High Level Computer Vision

Deep Learning for Computer Vision
Part 2

Bernt Schiele - schiele@mpi-inf.mpg.de
Mario Fritz - mfritz@mpi-inf.mpg.de

https://www.mpi-inf.mpg.de/hlcv

Overview Today

- ConvNet & Visualizations (left over from last lecture)
- Feature Generalization
  - “pre-training” on large dataset,
    “fine-tuning” on target dataset
- Object Detection
  - from image classification to object detection
- R-CNN - Regions with CNN features
  - Region-based Convolutional Networks for Accurate Object Detection and Semantic Segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik
    (CVPR’14, accepted in May’15 for PAMI)
  - Region Proposal Method: Selective Search for Object Recognition,
    J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, A. W. M. Smeulders
    In IJCV’13.

Large Convnets for Image Classification

Large Convnets for Image Classification

- Operations in each layer
- Architecture
- Training
- Results

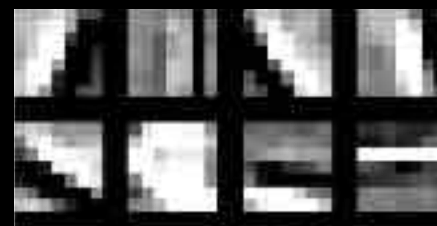
Components of Each Layer

Pixels /

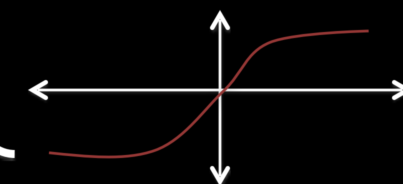
Features

Filter with Dictionary

(convolutional or tiled)

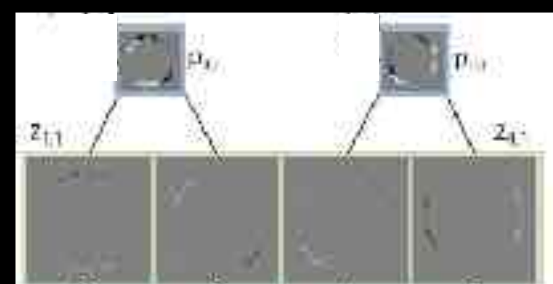


+ Non-linearity



Spatial/Feature

(Sum or Max)



[Optional]

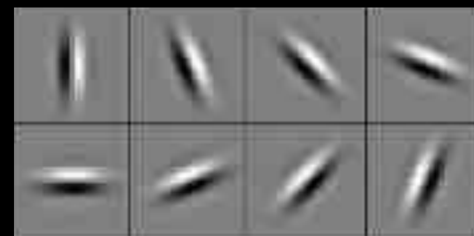
Normalization between feature responses

Output Features

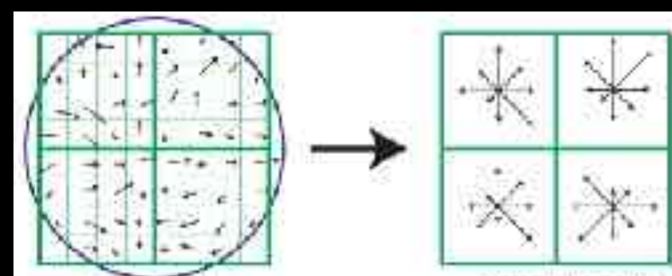
Compare: SIFT Descriptor

Image
Pixels

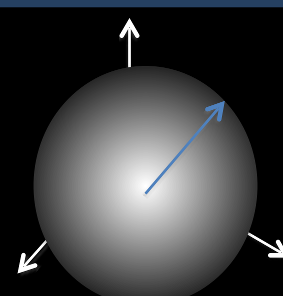
Apply
Gabor filters



Spatial pool
(Sum)



Normalize to unit
length

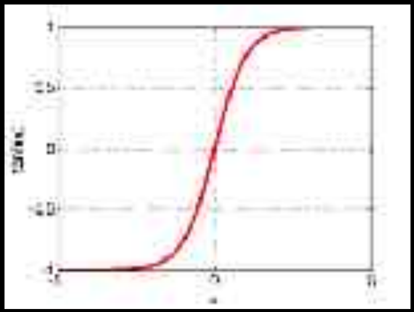


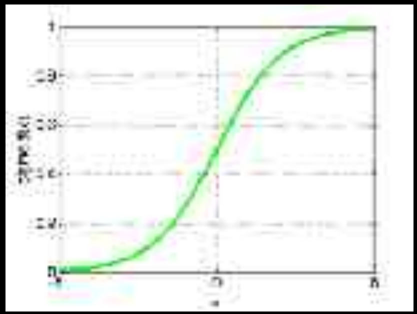
Feature
Vector

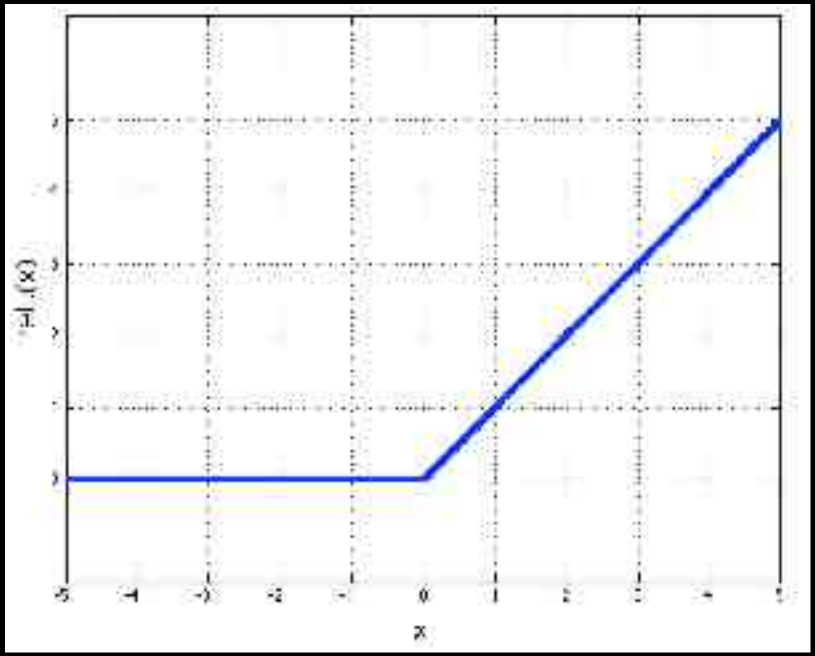
Non-Linearity

- Non-linearity
  - Per-feature independent
  - Tanh
  - Sigmoid: 1/(1+exp(-x))
  - Rectified linear
    - Simplifies backprop
    - Makes learning faster
    - Avoids saturation issues

→ Preferred option

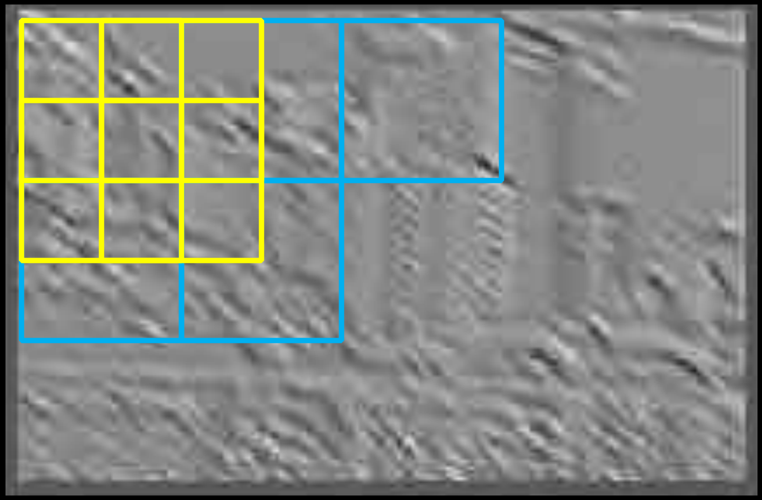






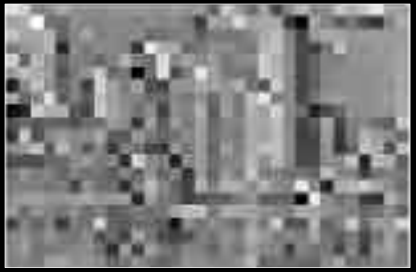
Pooling

- Spatial Pooling
  - Non-overlapping / overlapping regions
  - Sum or max
  - Boureau et al. ICML'10 for theoretical analysis



Max

Sum



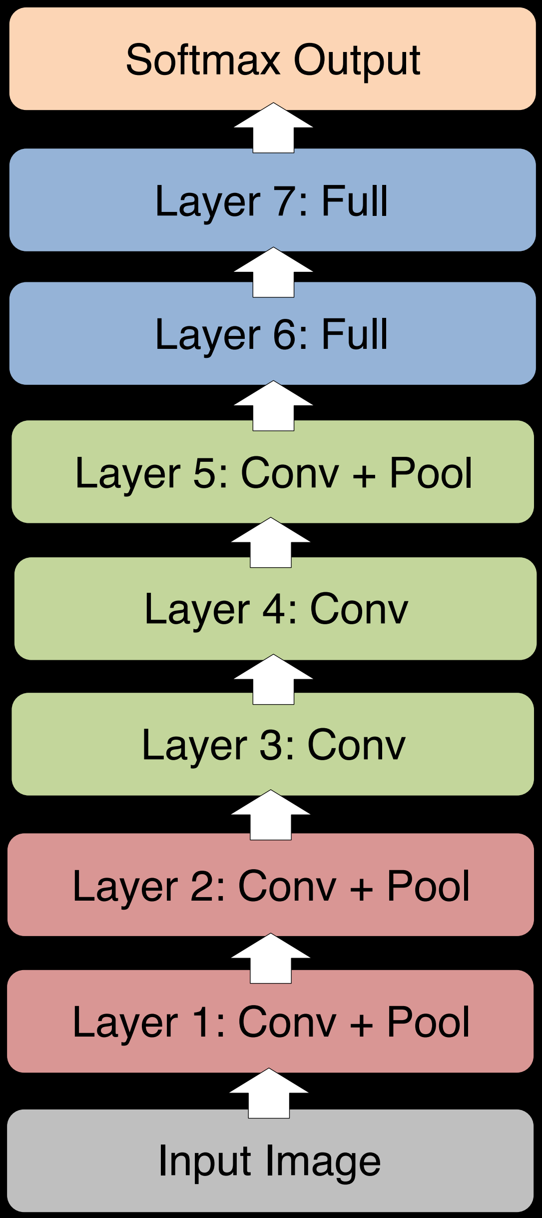
Architecture

Importance of Depth

Architecture of Krizhevsky et al.

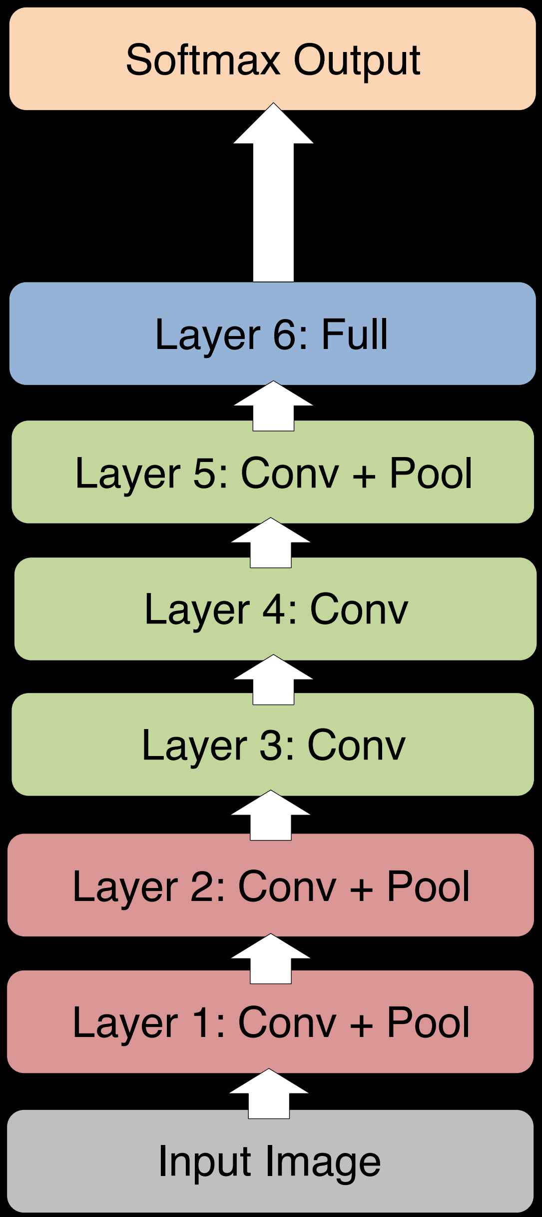
- 8 layers total
- Trained on Imagenet dataset [Deng et al. CVPR'09]
- 18.2% top-5 error
- Our reimplementation:

  18.1% top-5 error



Architecture of Krizhevsky et al.

- Remove top fully connected layer
  - Layer 7
- Drop 16 million parameters
- Only 1.1% drop in performance!

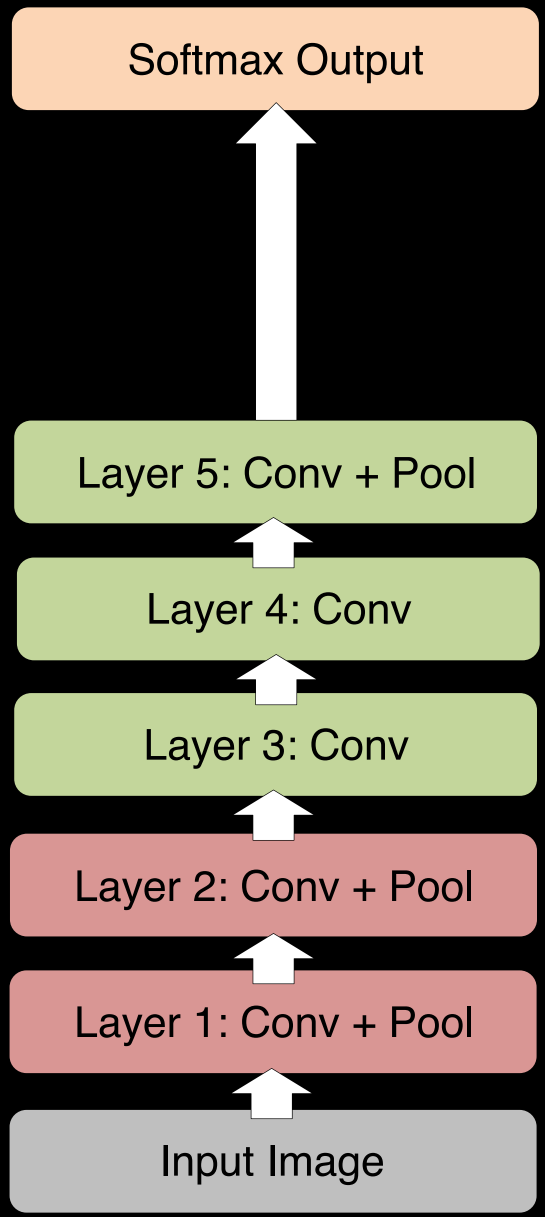


Architecture of Krizhevsky et al.

- Remove both fully connected layers
  - Layer 6 & 7

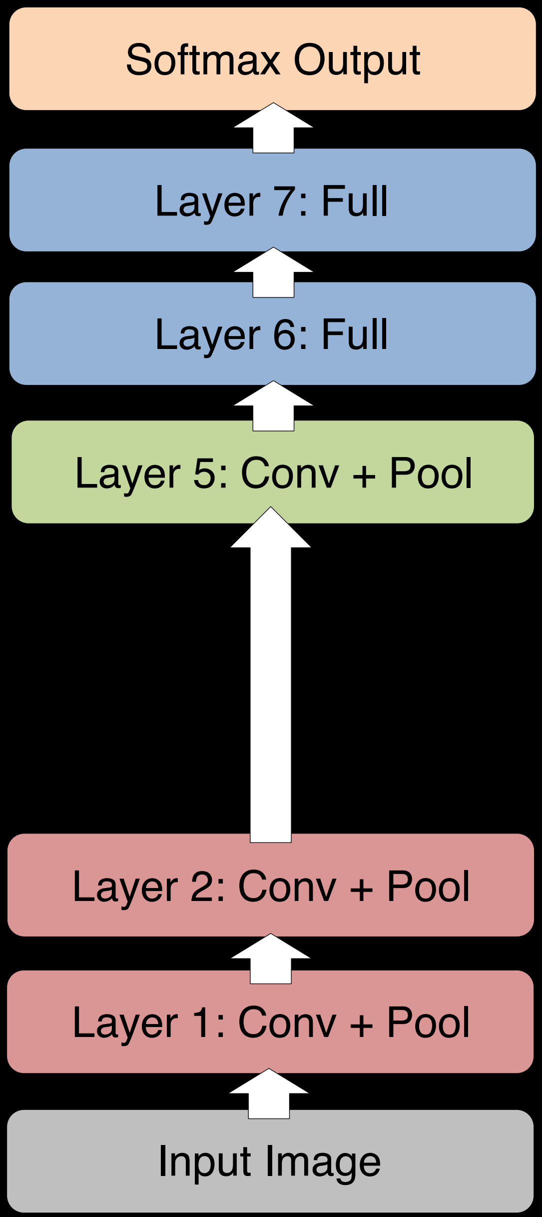
- Drop ~50 million parameters

- 5.7% drop in performance



Architecture of Krizhevsky et al.

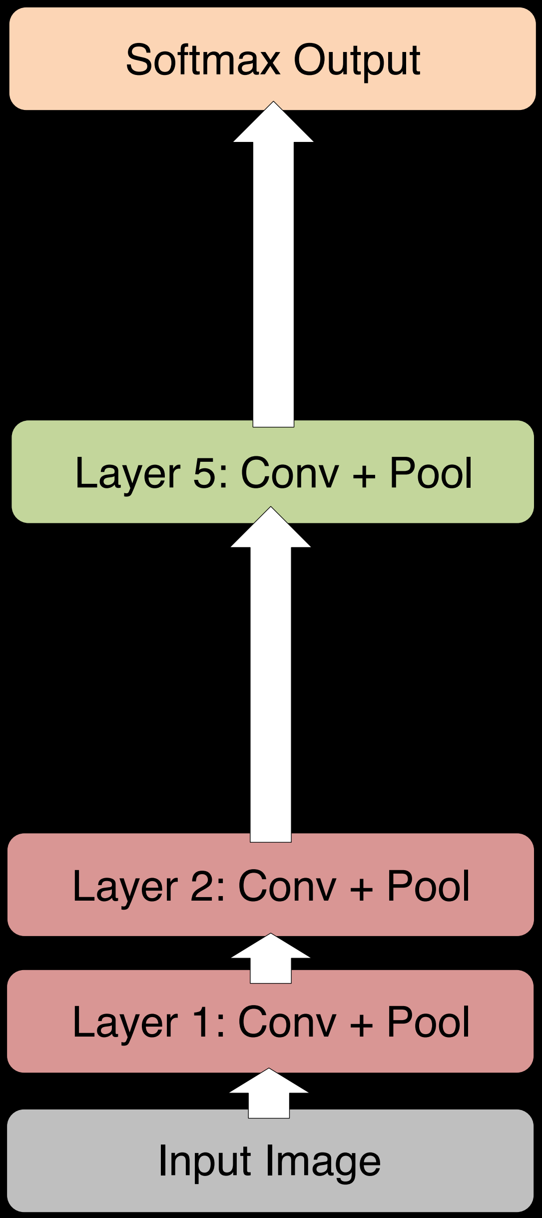
- Now try removing upper feature extractor layers:
  - Layers 3 & 4
- Drop ~1 million parameters
- 3.0% drop in performance



Architecture of Krizhevsky et al.

- Now try removing upper feature extractor layers & fully connected:
  - Layers 3, 4, 6 ,7
- Now only 4 layers
- 33.5% drop in performance

→Depth of network is key



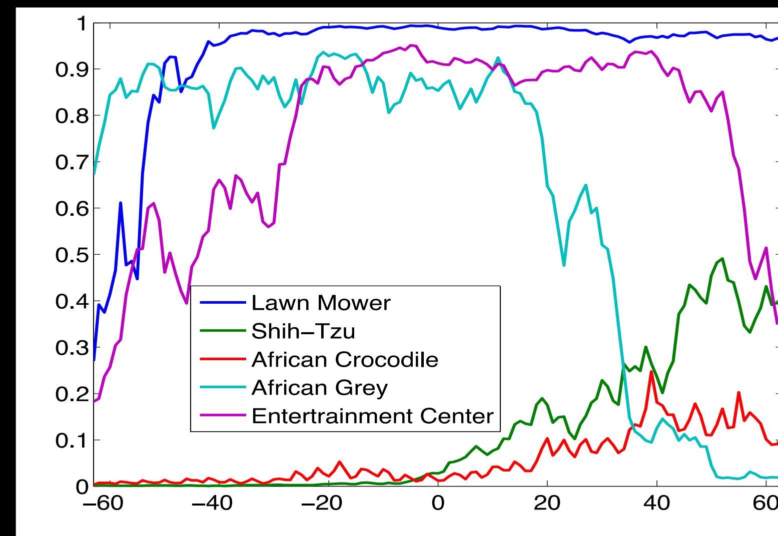
Tapping off Features at each Layer

Plug features from each layer into linear SVM or soft-max

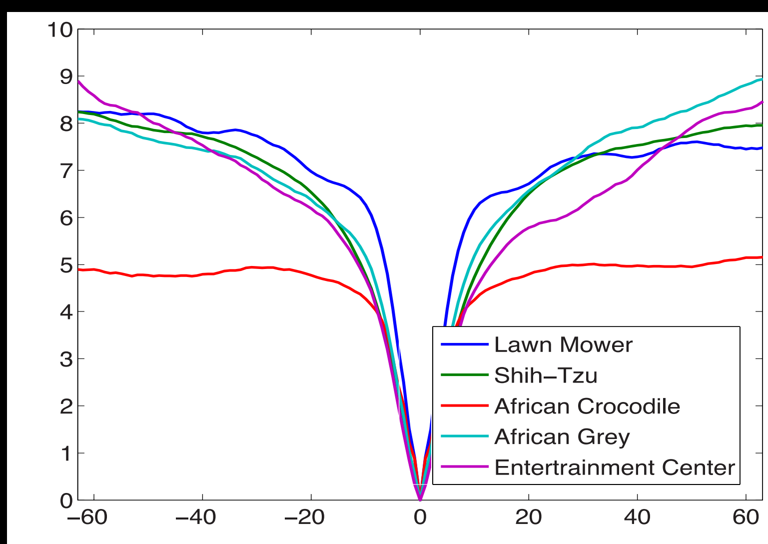
	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 ± 0.7	24.6 ± 0.4
SVM (2)	66.2 ± 0.5	39.6 ± 0.3
SVM (3)	72.3 ± 0.4	46.0 ± 0.3
SVM (4)	76.6 ± 0.4	51.3 ± 0.1
SVM (5)	$\mathbf{86.2 \pm 0.8}$	65.6 ± 0.3
SVM (7)	$\mathbf{85.5 \pm 0.4}$	$\mathbf{71.7 \pm 0.2}$
Softmax (5)	82.9 ± 0.4	65.7 ± 0.5
Softmax (7)	$\mathbf{85.4 \pm 0.4}$	$\mathbf{72.6 \pm 0.1}$

Translation (Vertical)

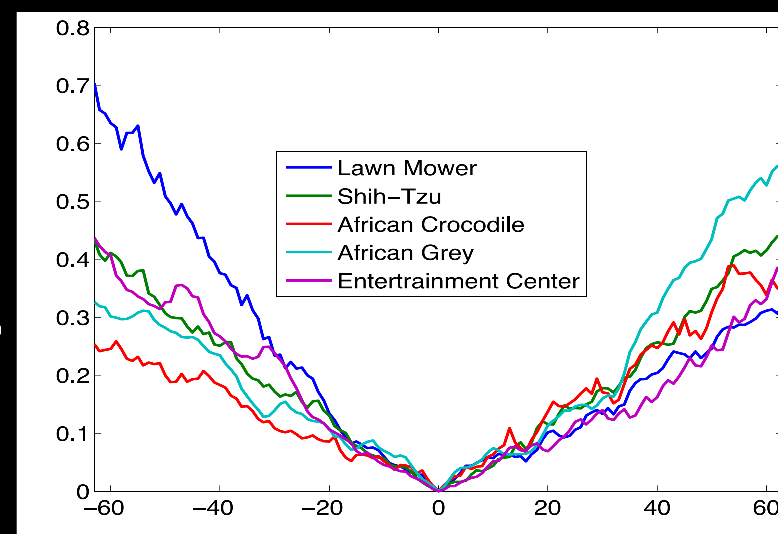
Output



Layer 1

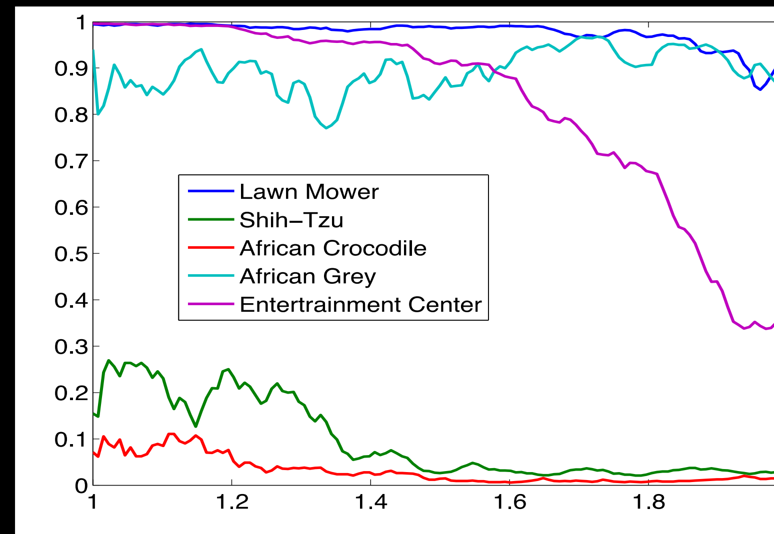


Layer 7

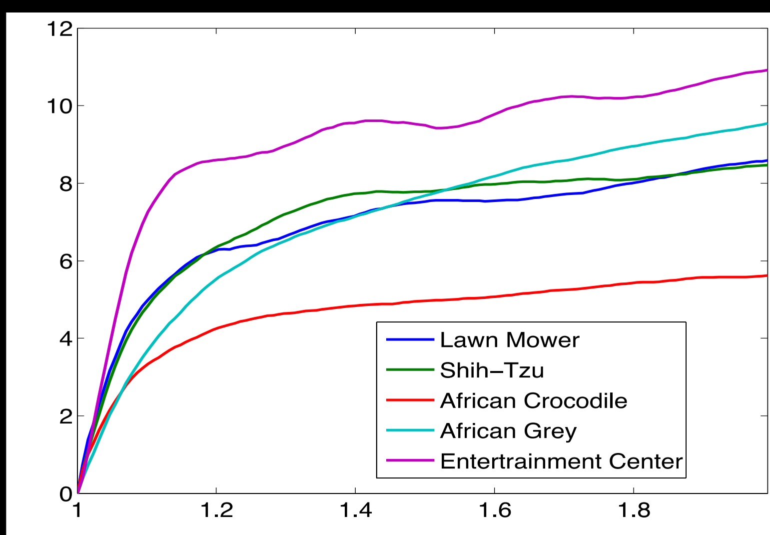


Scale Invariance

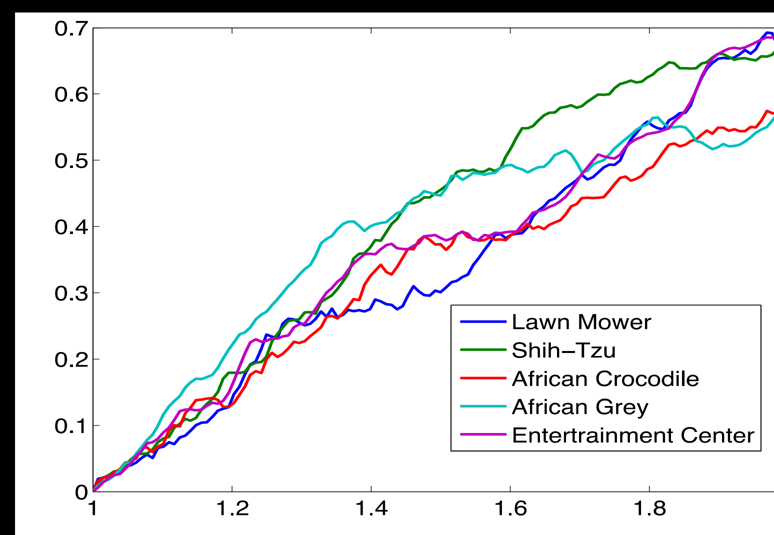
Output



Layer 1

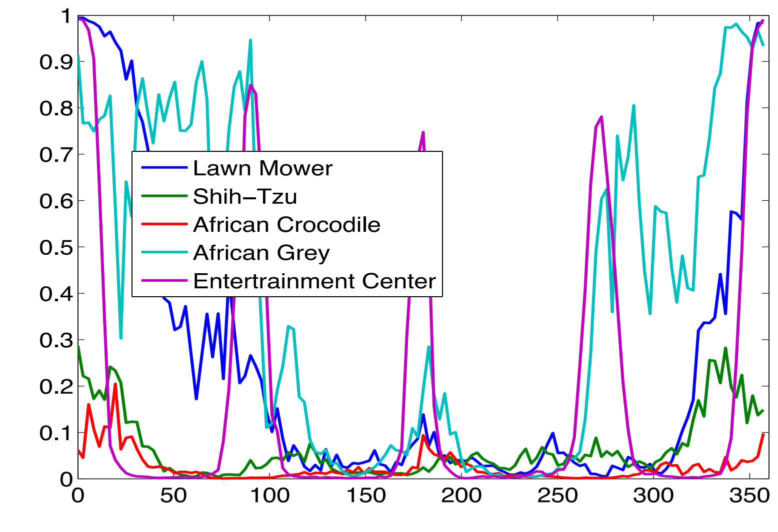


Layer 7

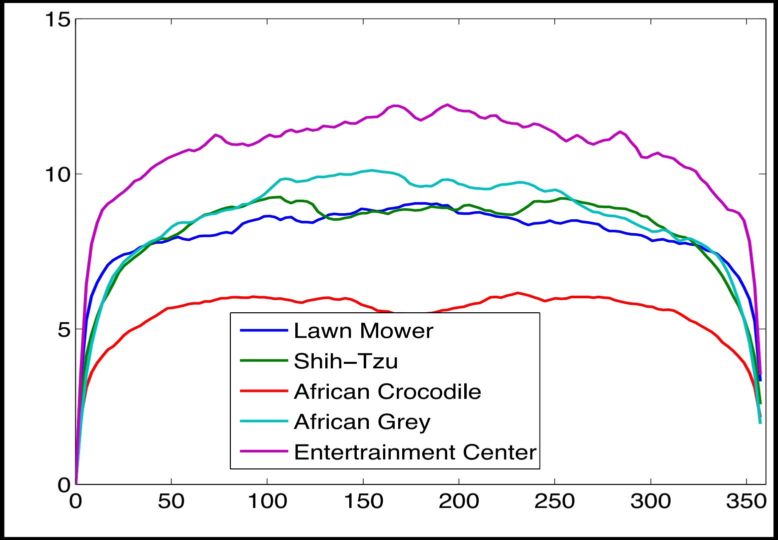


Rotation Invariance

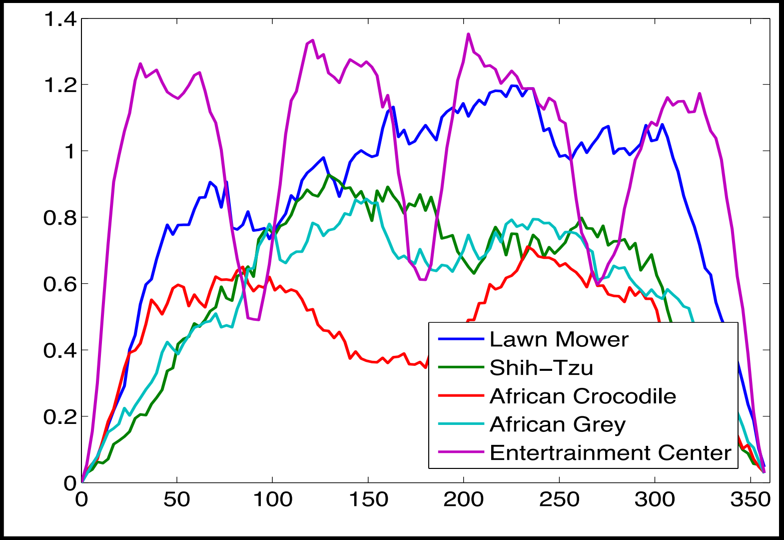
Output



Layer 1



Layer 7



Visualizing ConvNets

Visualizing Convnets

- Raw coefficients of learned filters in higher layers difficult to interpret
- Several approaches look to optimize input to maximize activity in a high-level feature
  - Erhan et al. [Tech Report 2009]
  - Le et al. [NIPS 2010]
  - Depend on initialization
  - Model invariance with Hessian about (locally) optimal stimulus

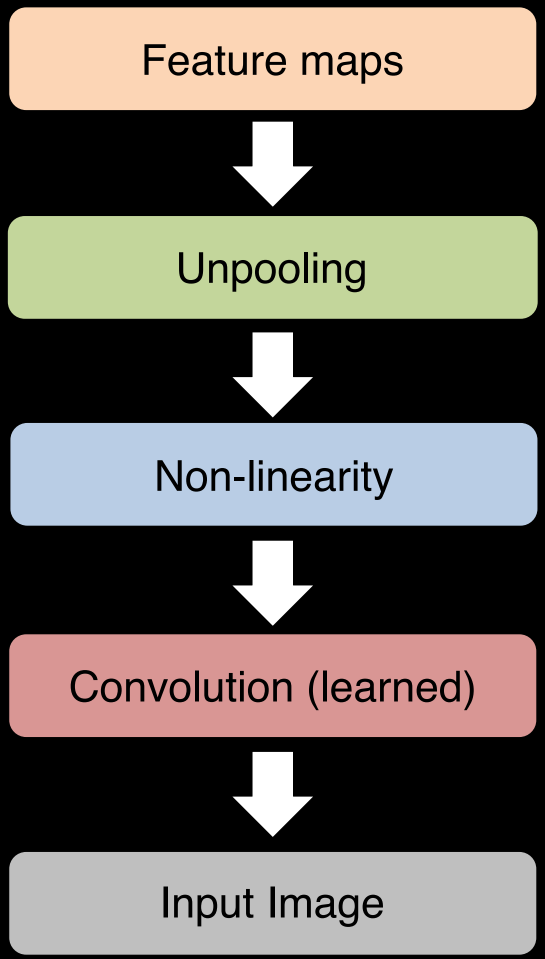
Visualization using Deconvolutional Networks

[Zeiler et al. CVPR'10, ICCV'11, arXiv'13]

- Provide way to map activations at high layers back to the input

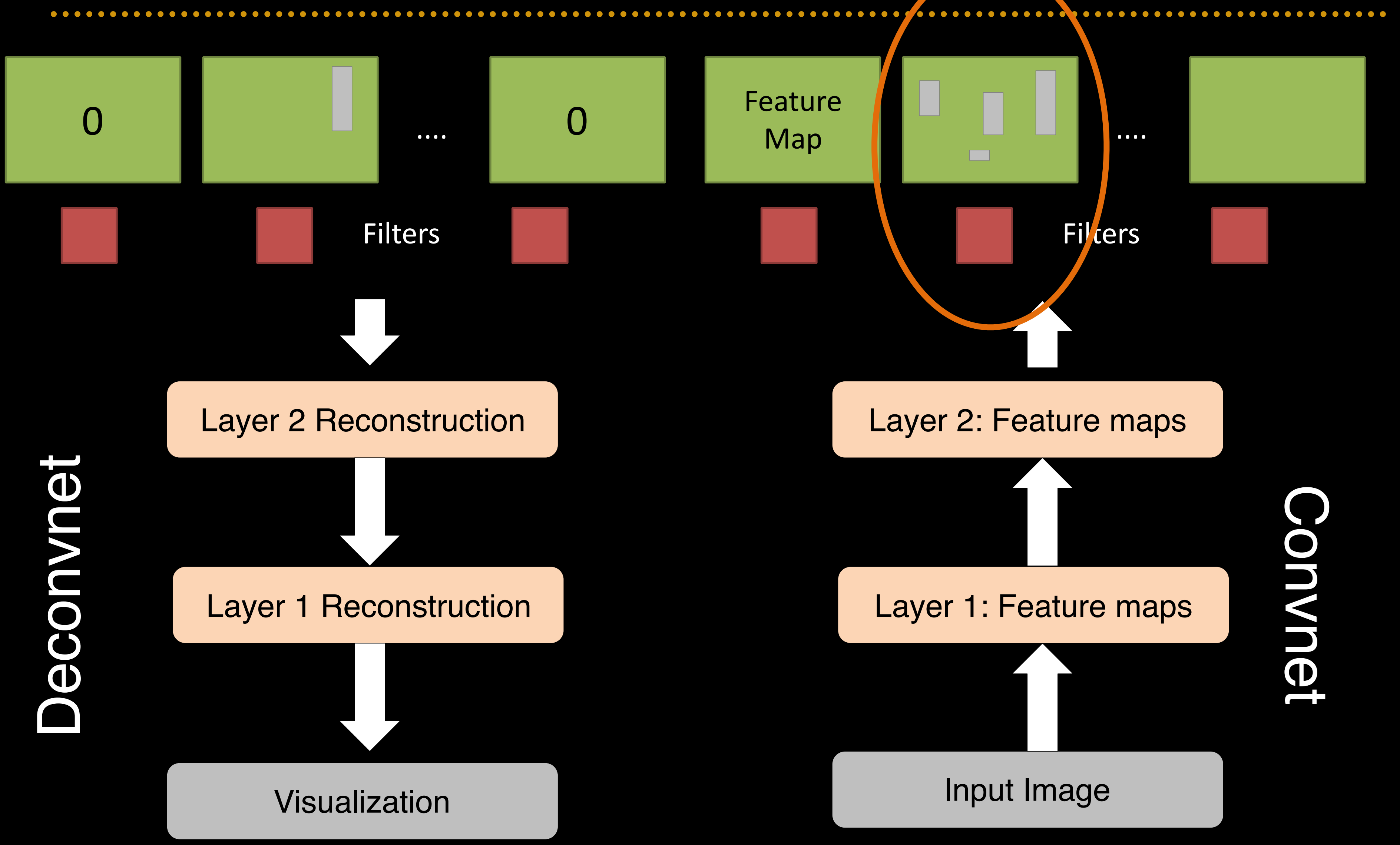
- Same operations as Convnet, but in reverse:
  - Unpool feature maps
  - Convolve unpooled maps
    - Filters copied from Convnet

- Used here purely as a probe
  - Originally proposed as unsupervised learning method
  - No inference, no learning

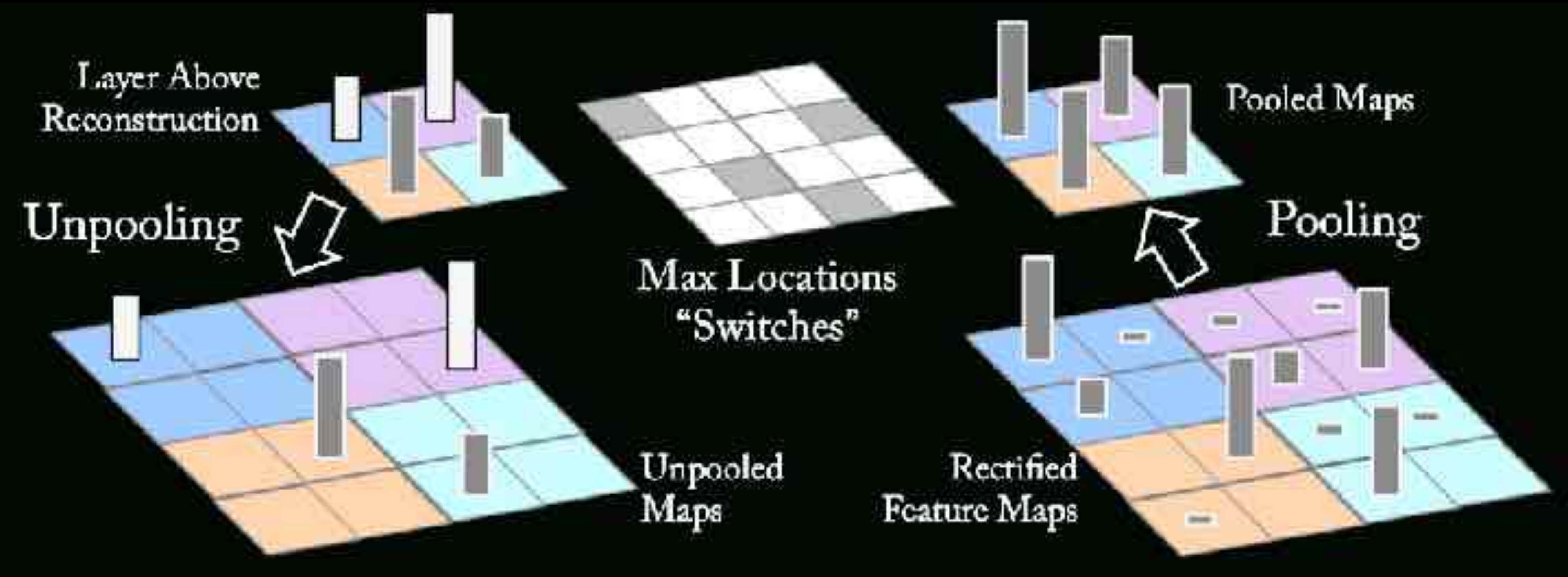


Deconvnet Projection from Higher Layers

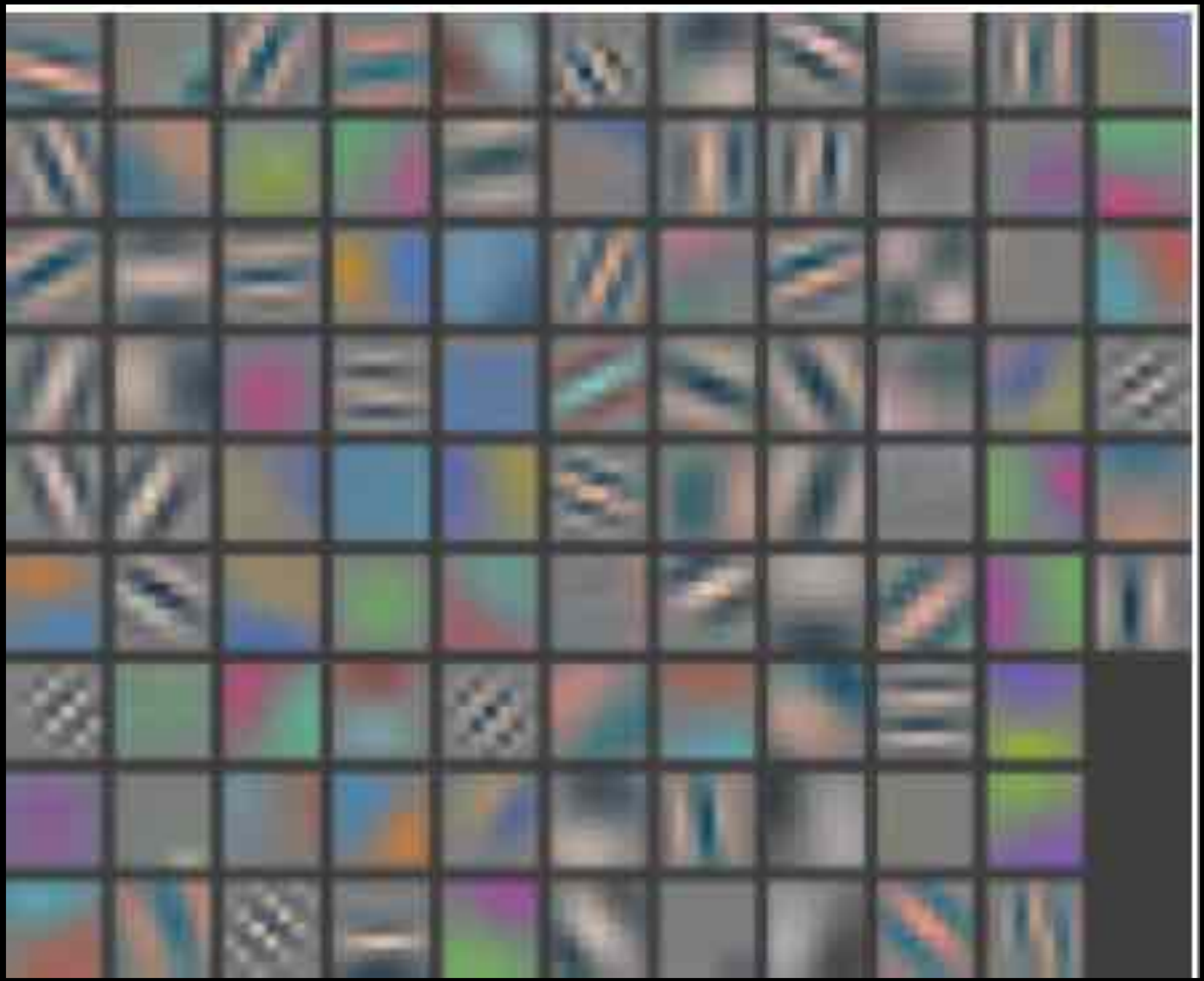
[Zeiler and Fergus. arXiv'13]



Unpooling Operation



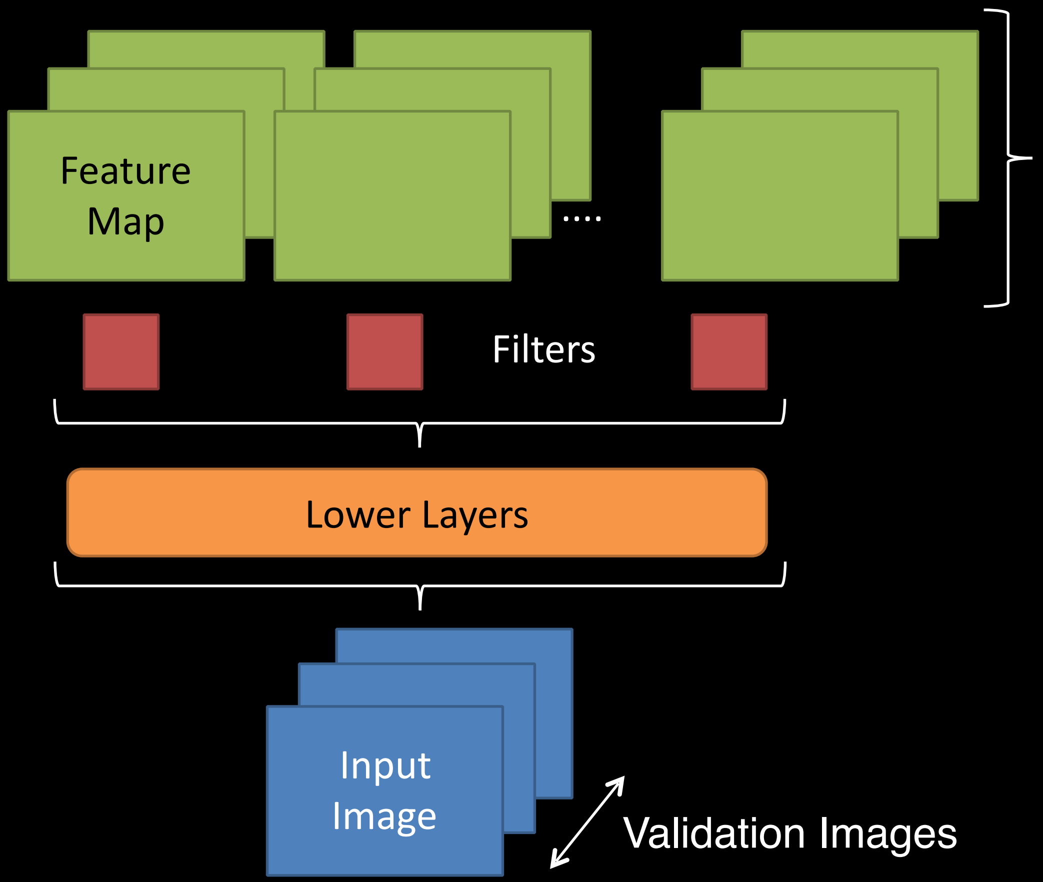
Layer 1 Filters



Visualizations of Higher Layers

[Zeiler and Fergus. arXiv'13]

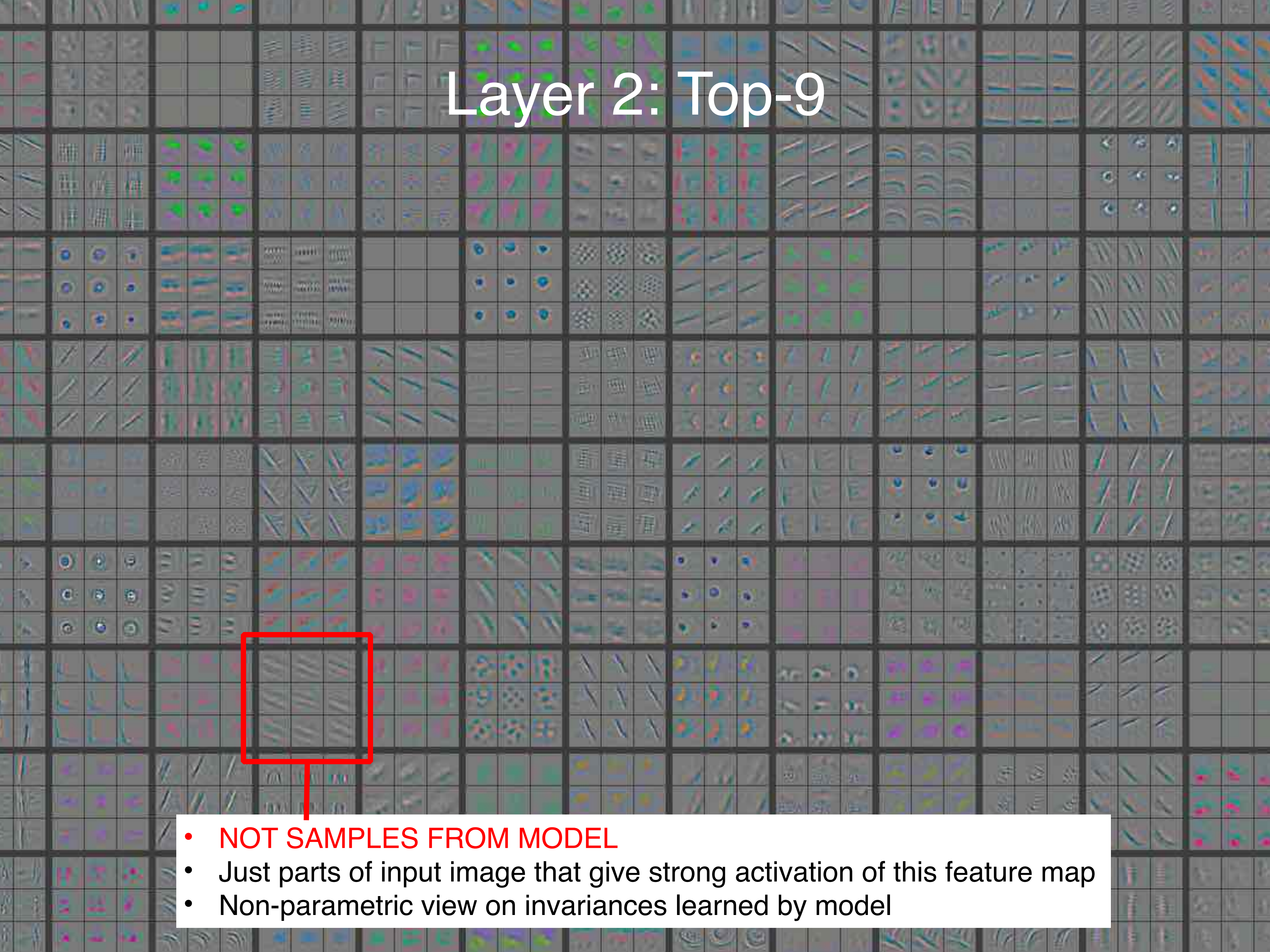
- Use ImageNet 2012 validation set
- Push each image through network



- Take max activation from feature map associated with each filter
- Use Deconvnet to project back to pixel space
- Use pooling "switches" peculiar to that activation

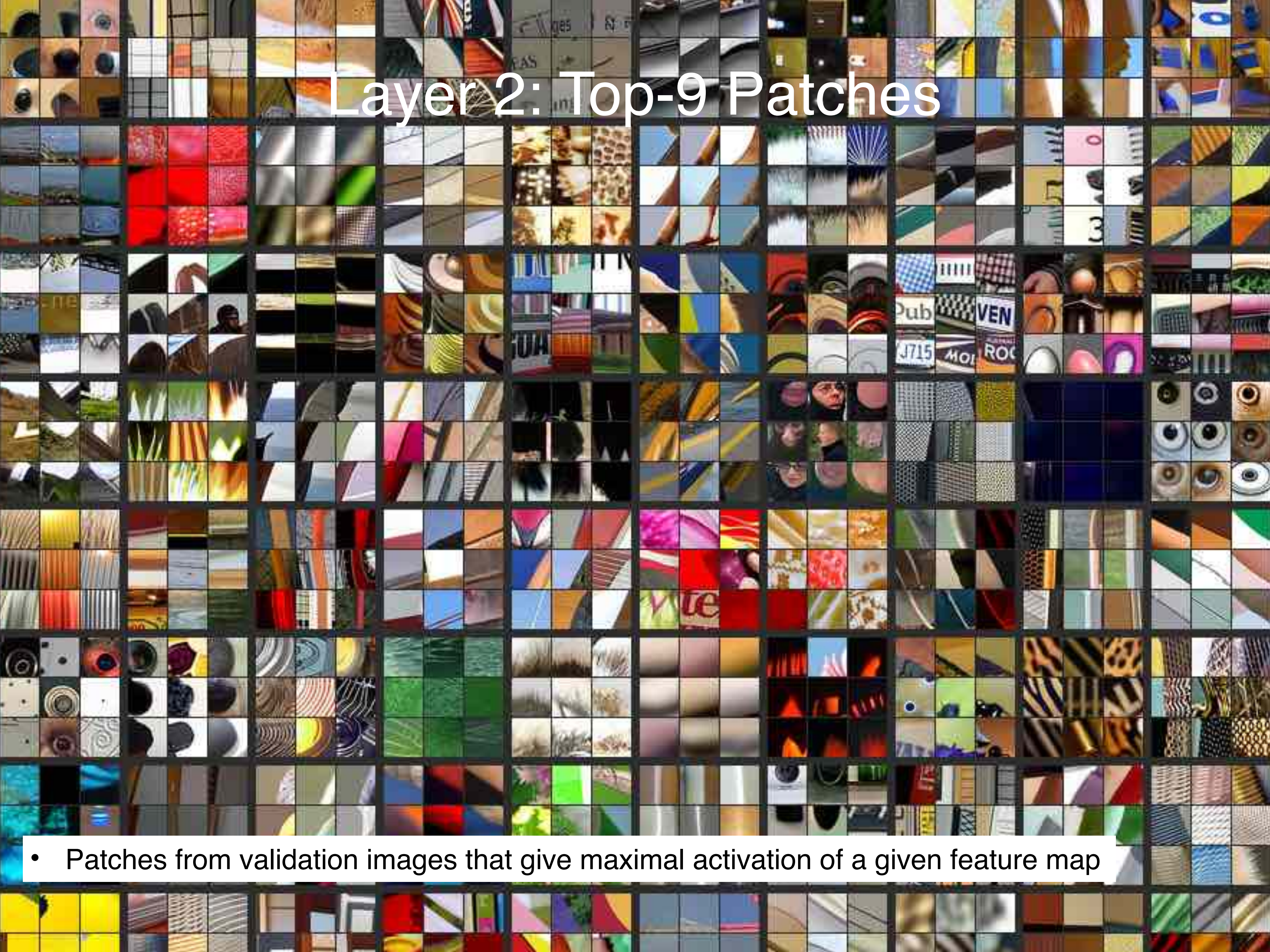
Layer 1: Top-9 Patches

Layer 2: Top-9



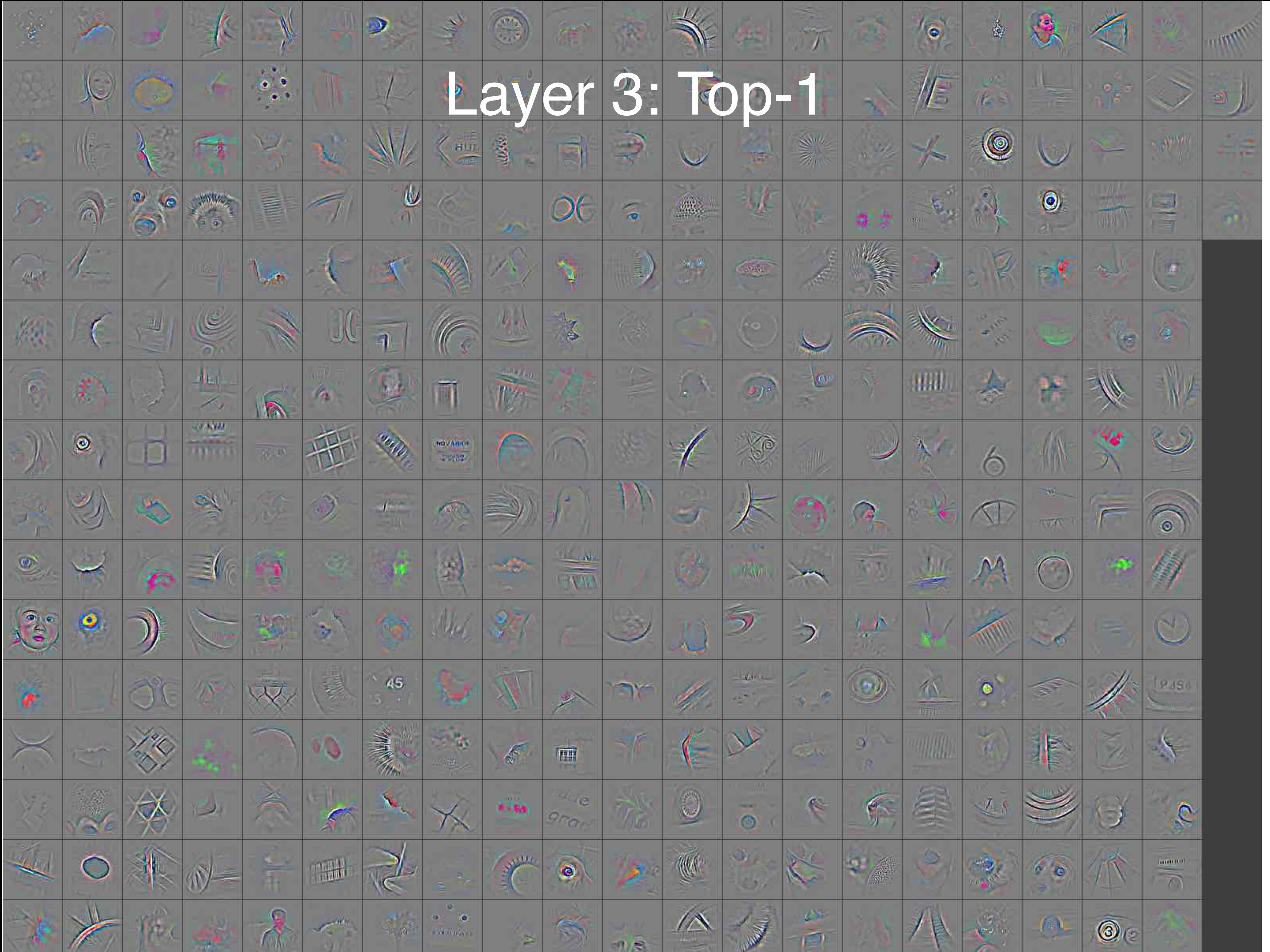
- NOT SAMPLES FROM MODEL
- Just parts of input image that give strong activation of this feature map
- Non-parametric view on invariances learned by model

Layer 2: Top-9 Patches

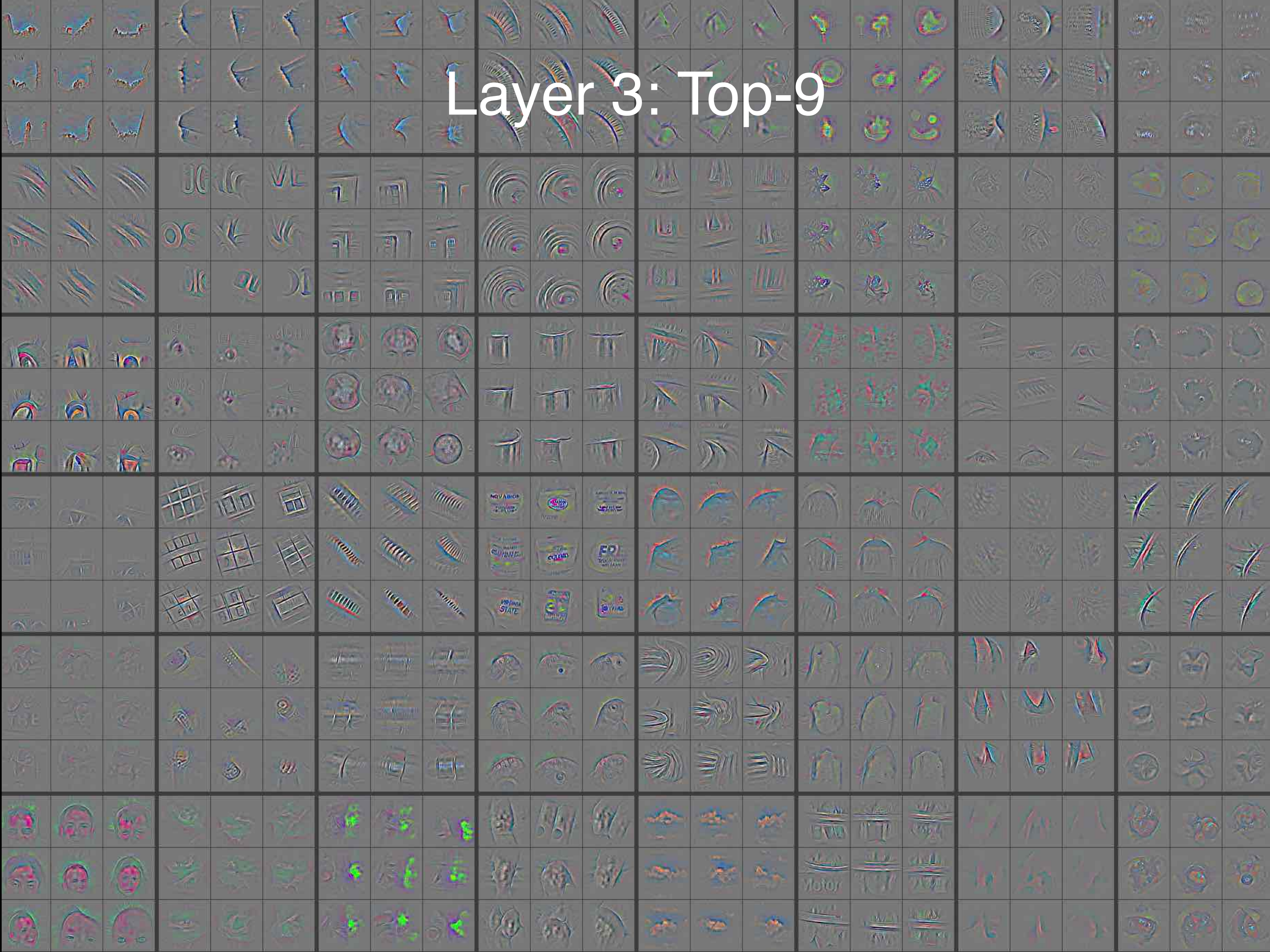


- Patches from validation images that give maximal activation of a given feature map

Layer 3: Top-1

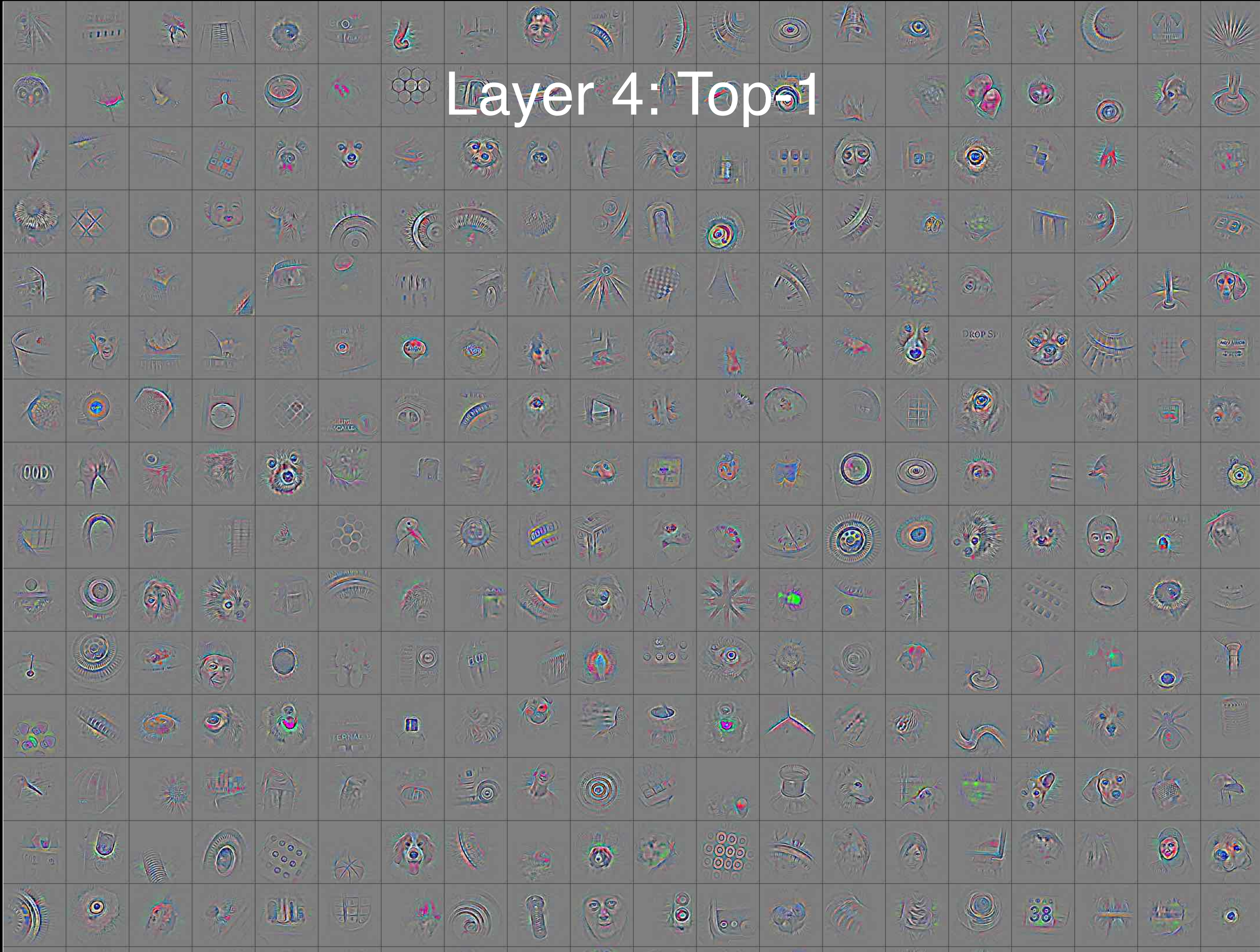


Layer 3: Top-9

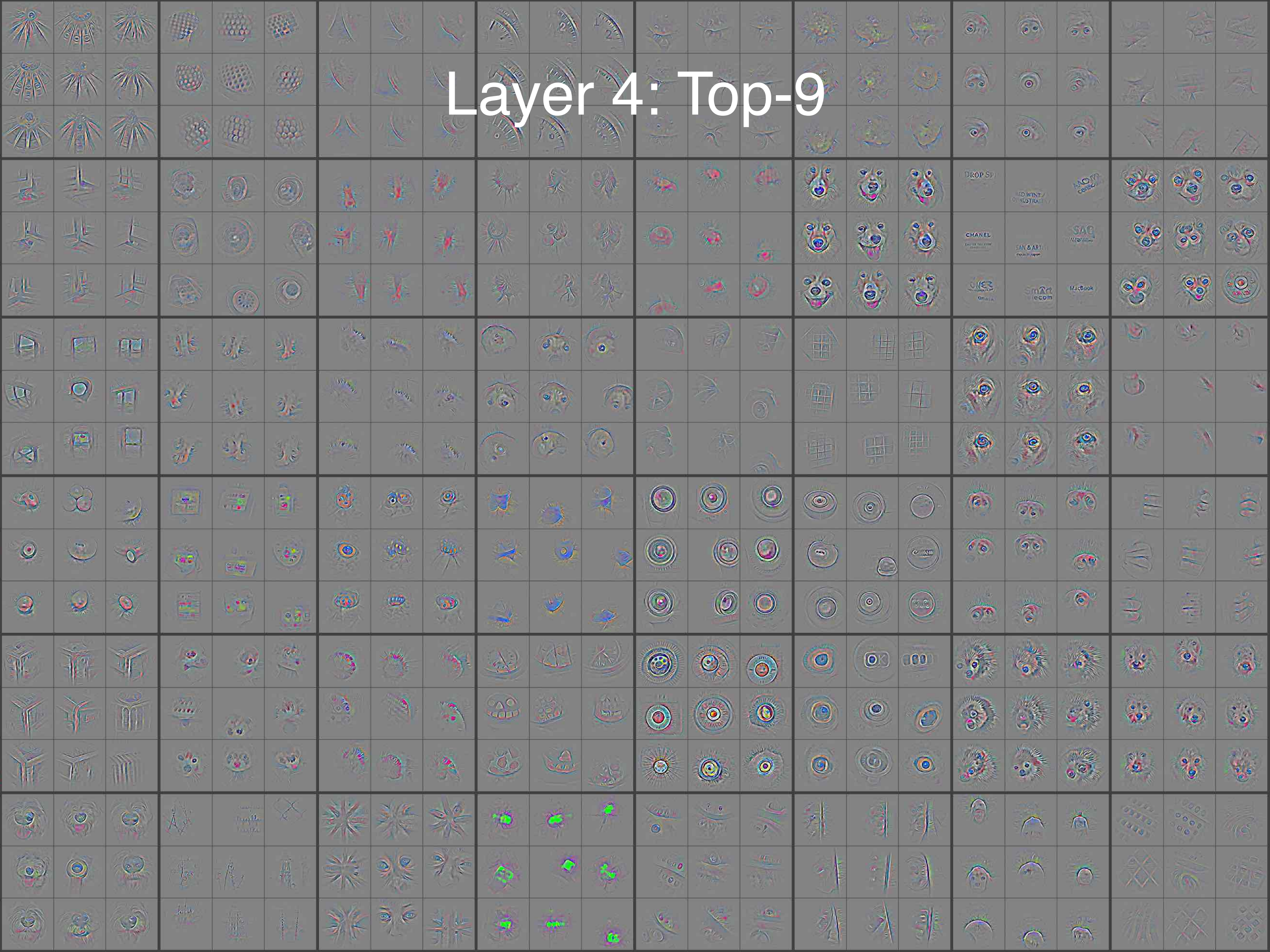


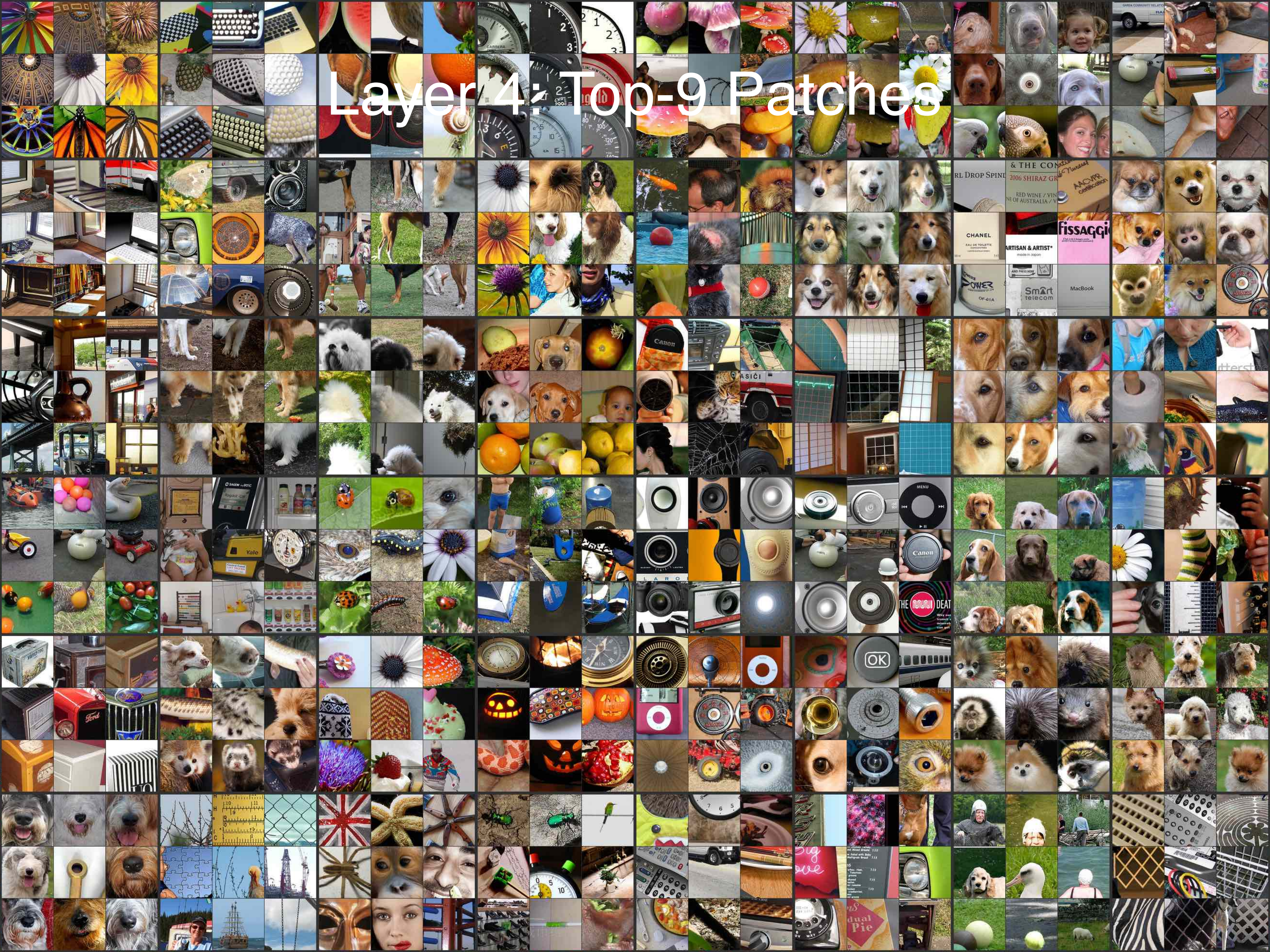
Layer 3: Top-9 Patches

Layer 4: Top-1

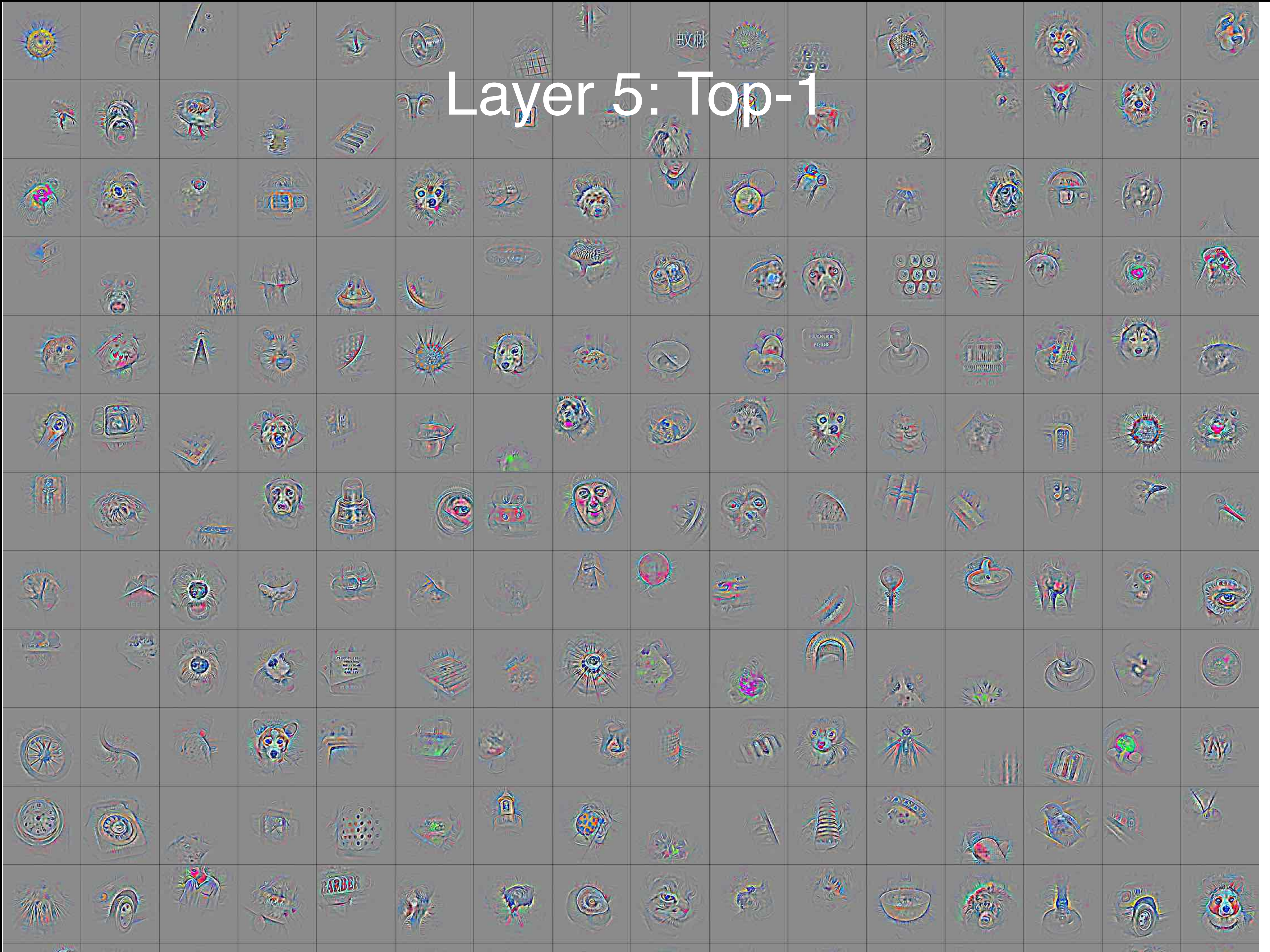


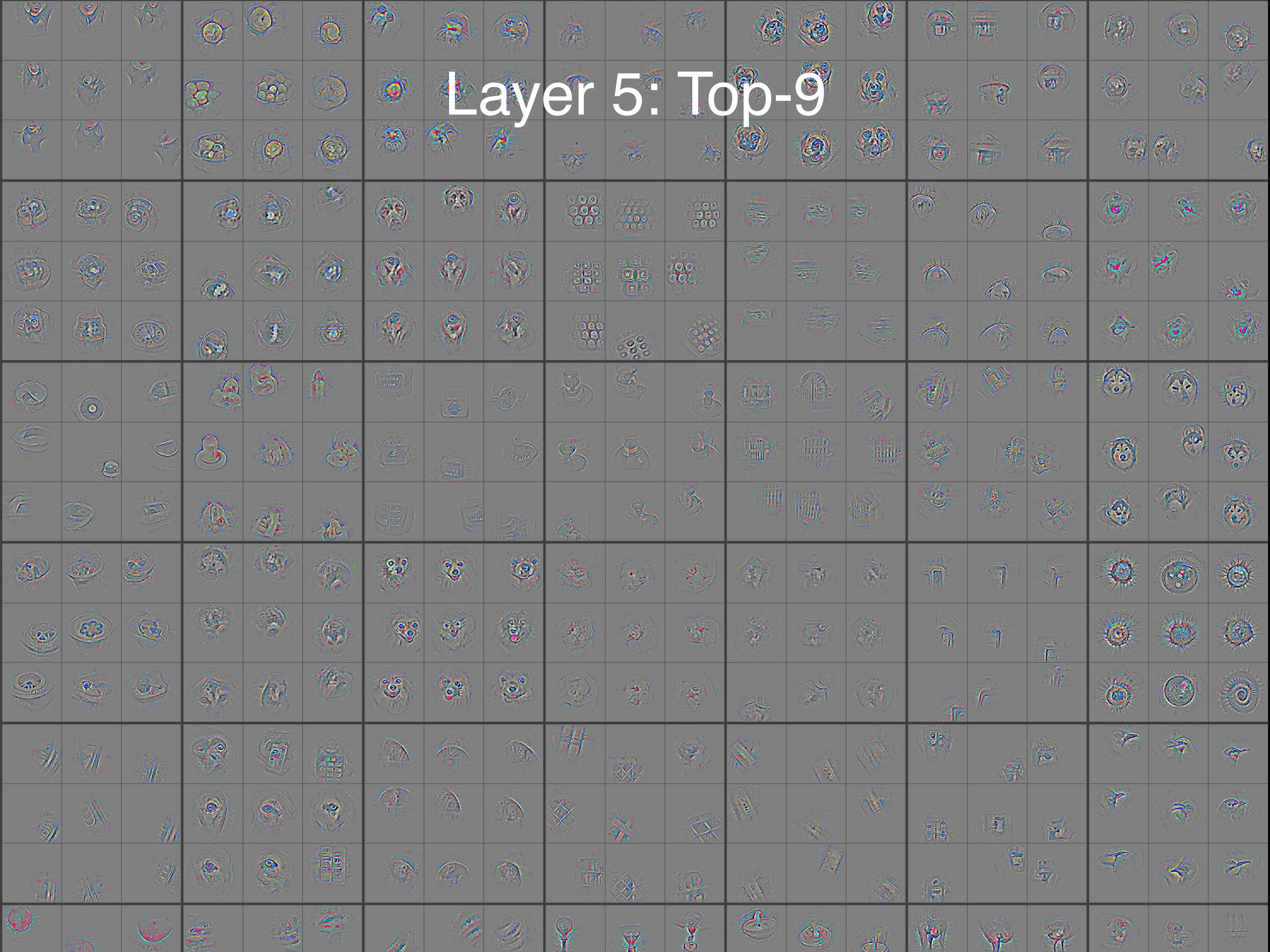
Layer 4: Top-9

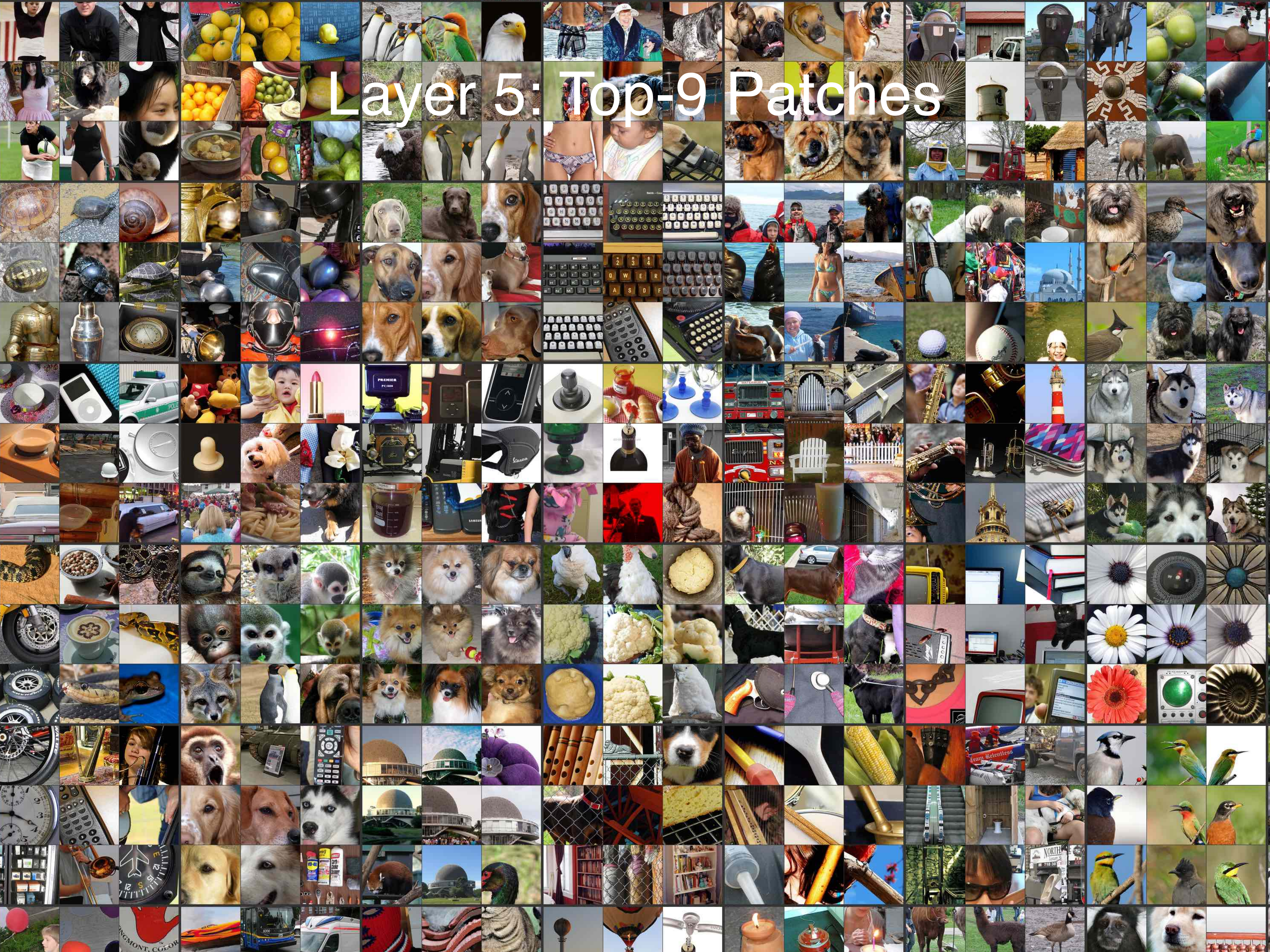



Layer 4: Top-9 Patches

Layer 5: Top-1

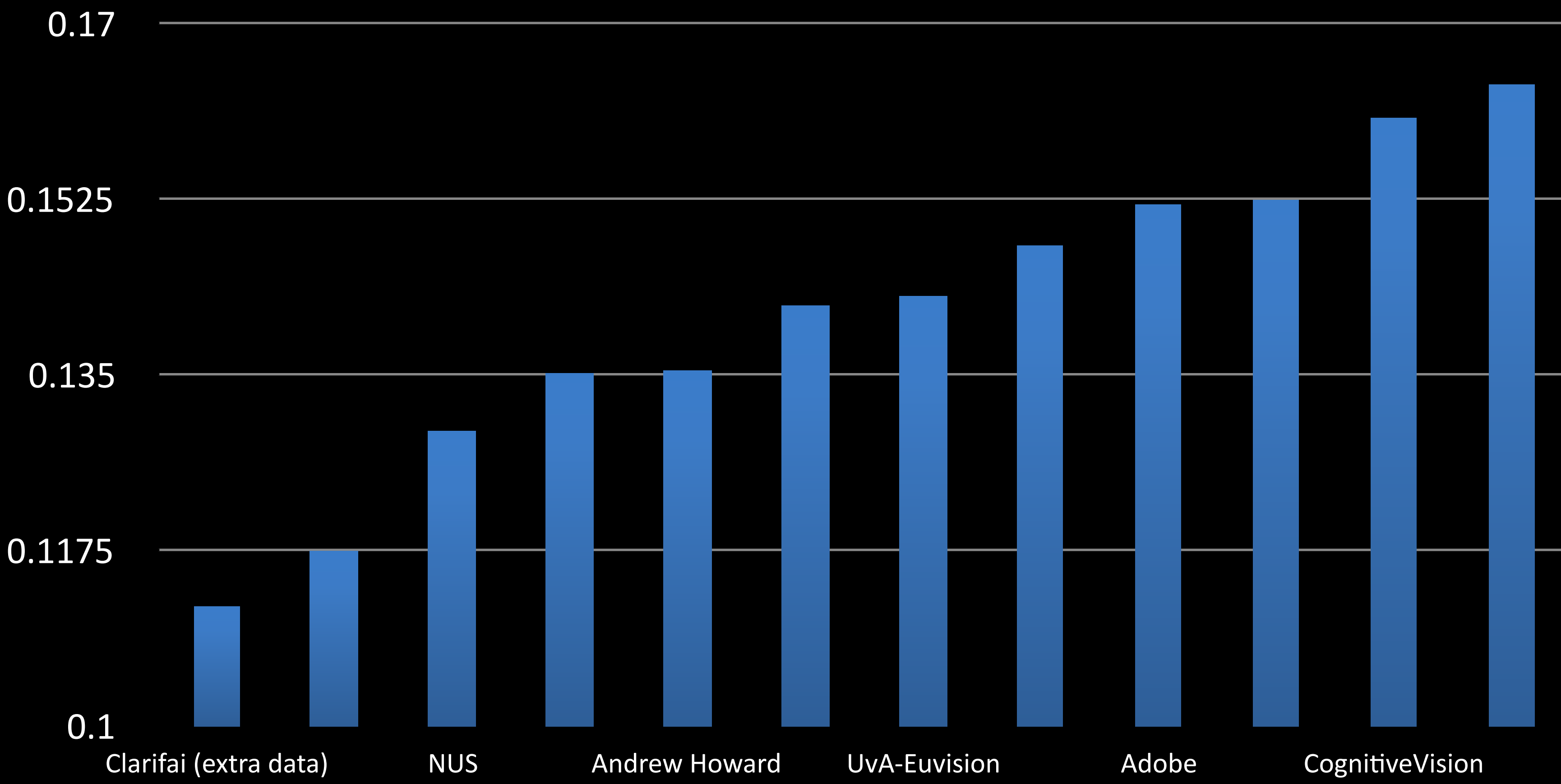



Layer 5: Top-9


Layer 5: Top-9 Patches

ImageNet Classification 2013 Results

- http://www.image-net.org/challenges/LSVRC/2013/results.php



- Pre-2012: 26.2% error → 2012: 16.5% error → 2013: 11.2% error

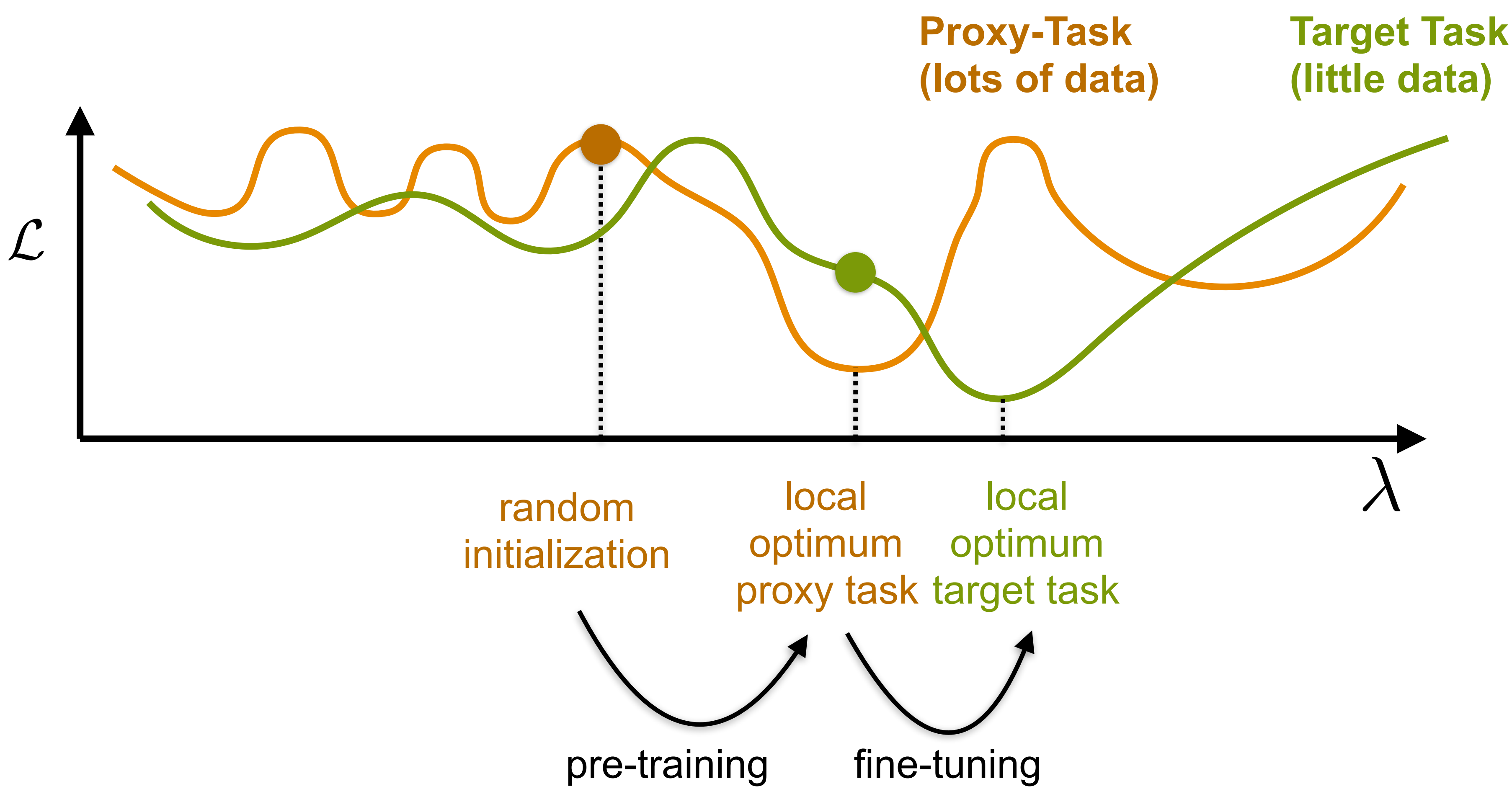
Sample Classification Results

[Krizhevsky et al. NIPS'12]

Feature Generalization and Pretraining: Overview

- Typically we are lacking data
- But there are large datasets for some tasks
- Idea:
  - Can we use learnt features from other trasks?
  - How can we transfer learnt features from other tasks?
  - Can we still do end-to-end learning?

Feature Generalization and Pretraining: Overview



Feature Generalization

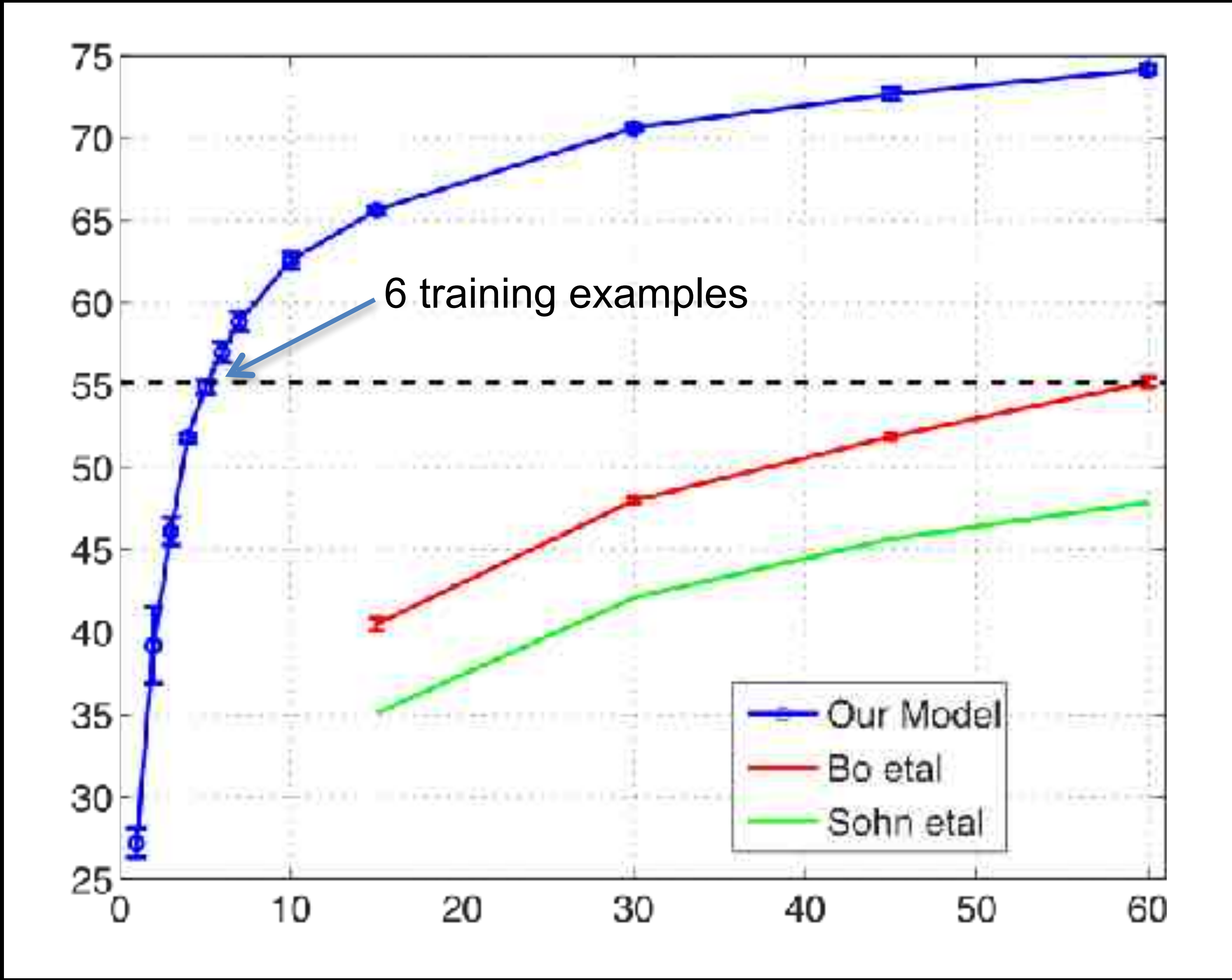
slides from: Rob Fergus, NIPS'13 tutorial

Training Features on Other Datasets

- Train model on ImageNet 2012 training set
- Re-train classifier on new dataset
  - Just the softmax layer
- Classify test set of new dataset

Caltech 256

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, arXiv 1311.2901, 2013



Caltech 256

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, arXiv 1311.2901, 2013

# Train	Acc % 15/class	Acc % 30/class	Acc % 45/class	Acc % 60/class
Sohn *et al.* [16]	35.1	42.1	45.7	47.9
Bo *et al.* [3]	40.5 ± 0.4	48.0 ± 0.2	51.9 ± 0.2	55.2 ± 0.3
Non-pretr.	9.0 ± 1.4	22.5 ± 0.7	31.2 ± 0.5	38.8 ± 1.4
ImageNet-pretr.	$\mathbf{65.7 \pm 0.2}$	$\mathbf{70.6 \pm 0.2}$	$\mathbf{72.7 \pm 0.4}$	$\mathbf{74.2 \pm 0.3}$

[3] L. Bo, X. Ren, and D. Fox. Multipath sparse coding using hierarchical matching pursuit. In CVPR, 2013.

[16] K. Sohn, D. Jung, H. Lee, and A. Hero III. Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In ICCV, 2011.

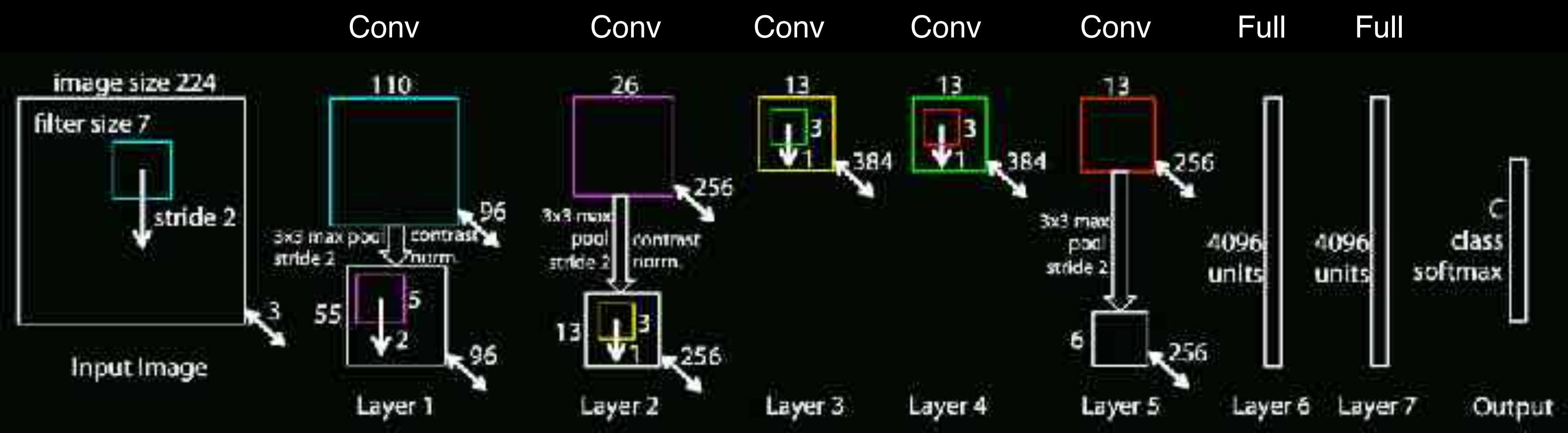
Object Detection

slides from: Rob Fergus, NIPS'13 tutorial

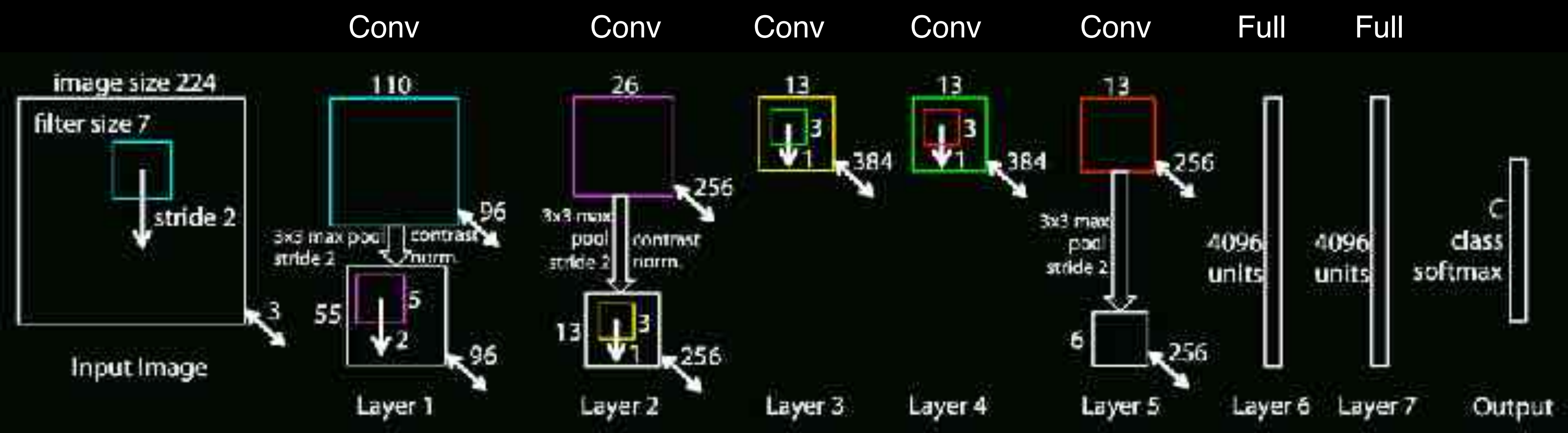
Detection with ConvNets

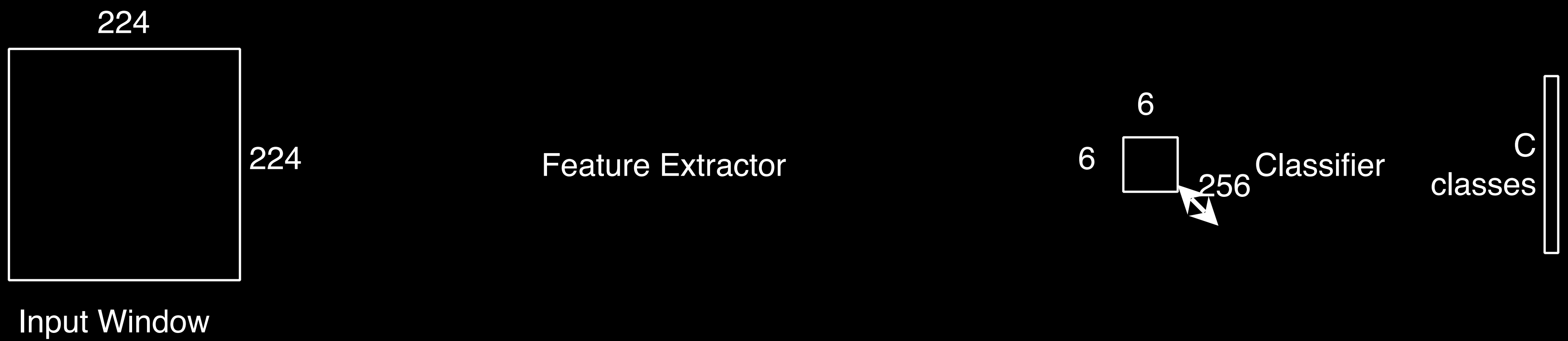
- So far, all about classification
- What about localizing objects within the scene?

Sliding Window with ConvNet

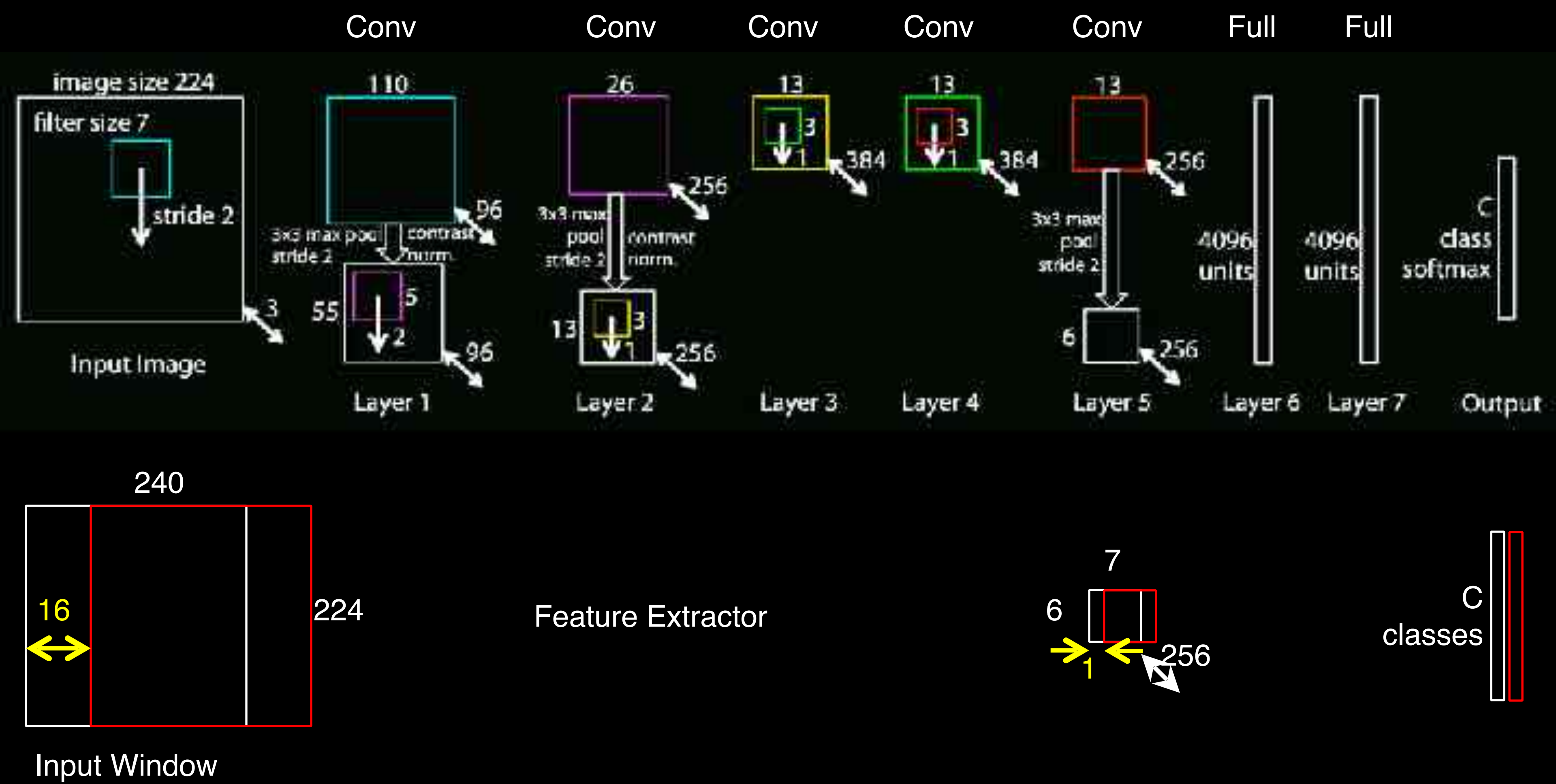


Sliding Window with ConvNet



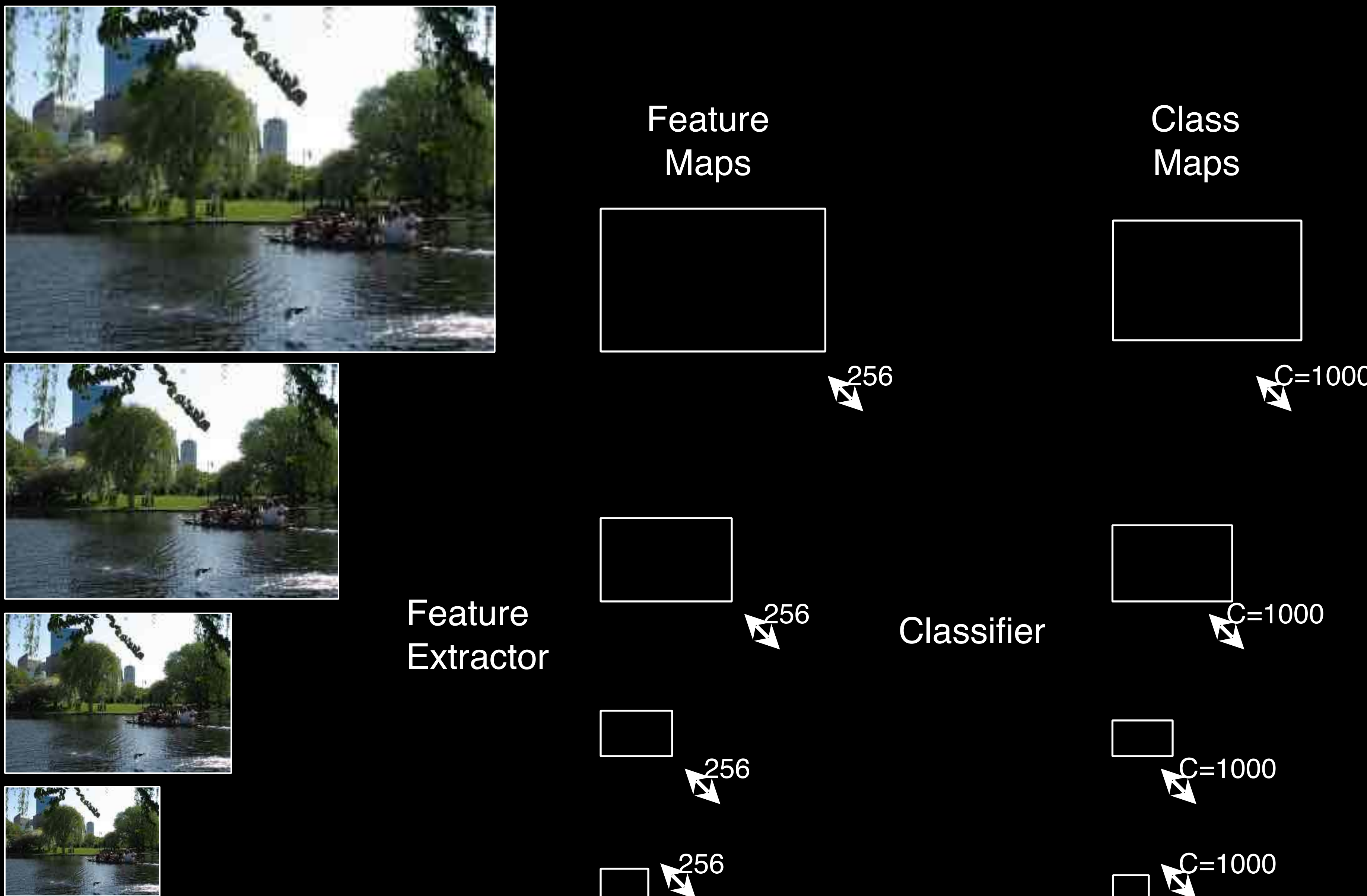


Sliding Window with ConvNet

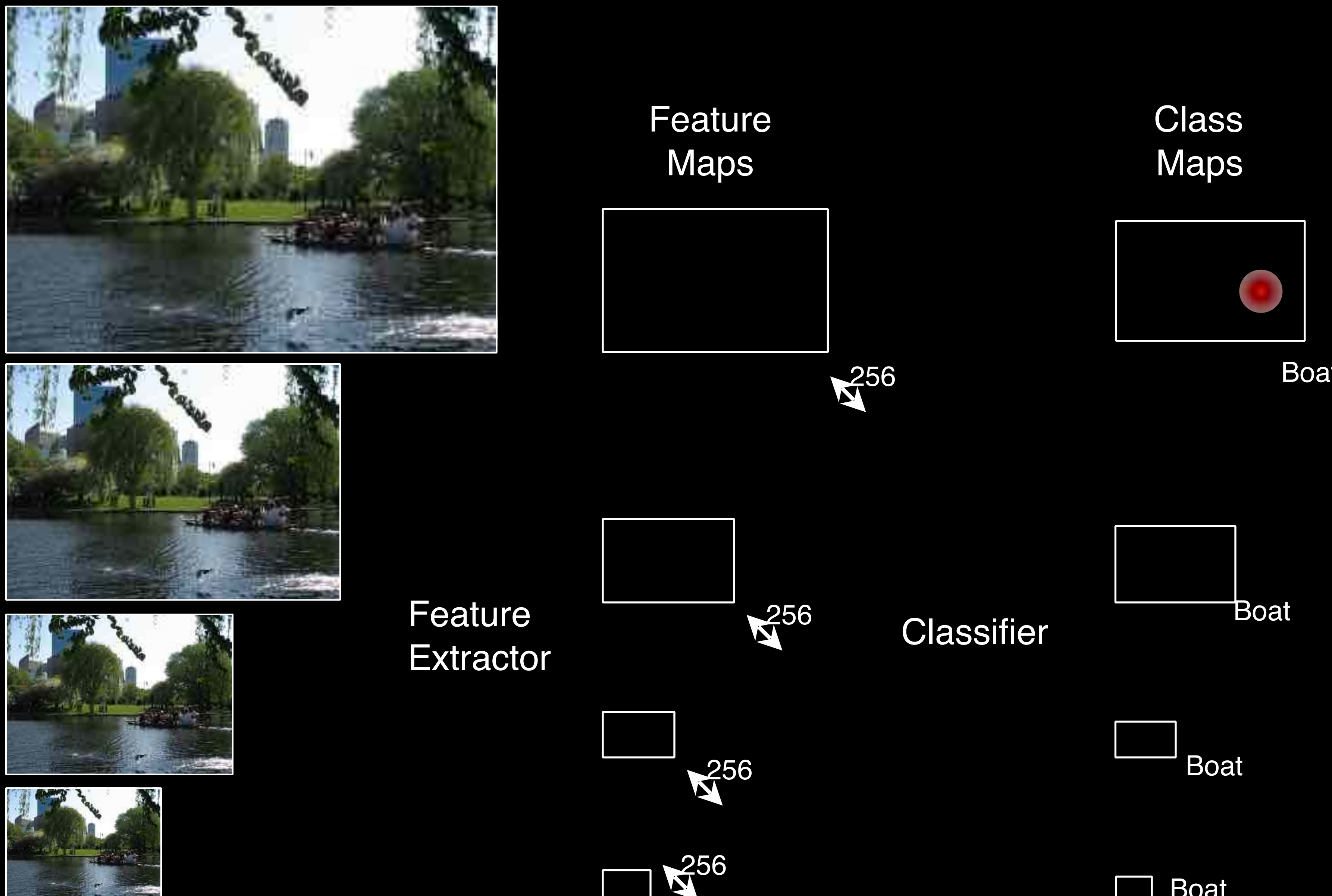


No need to compute two separate windows
Just one big input window, computed in a single pass

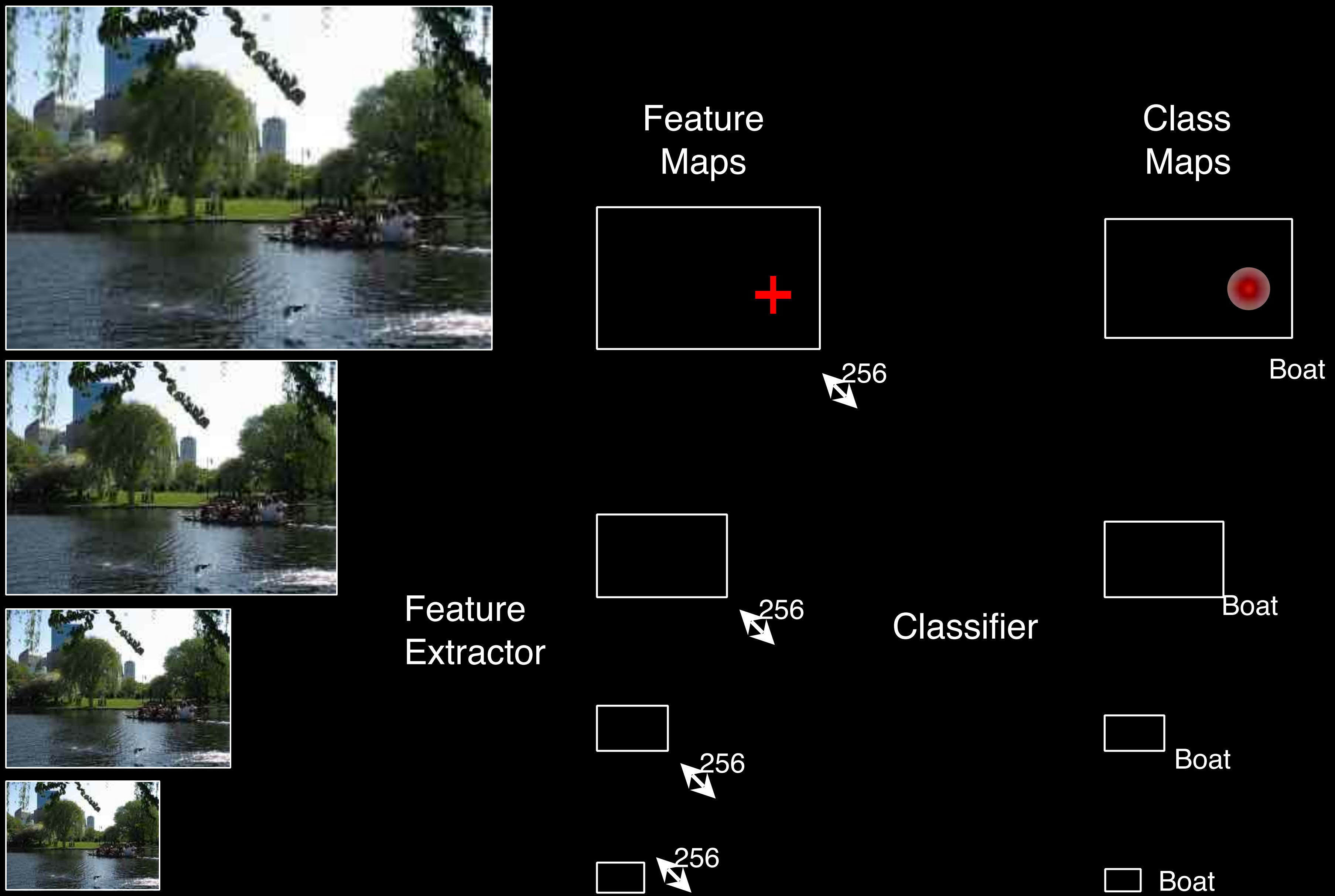
ConvNets for Detection



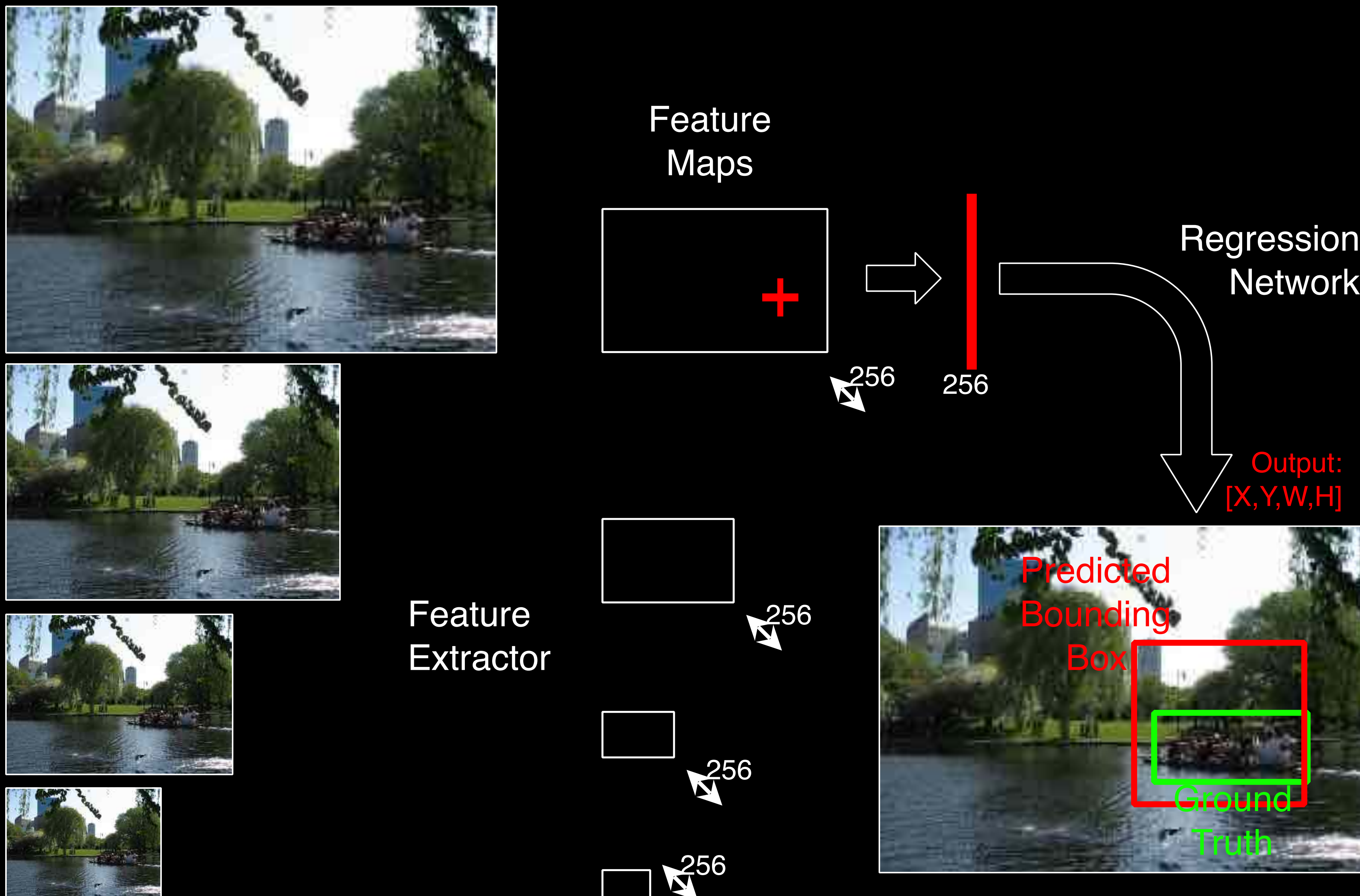
ConvNets for Detection



ConvNets for Detection



ConvNets for Detection



Bounding Box prediction example

[Sermanet et al. CVPR'14]

Detection Results

[Sermanet et al. CVPR'14]

Top predictions:

tv or monitor (confidence 11.5)
person (confidence 4.5)
miniskirt (confidence 3.1)

Groundtruth:

tv or monitor
tv or monitor (2)
tv or monitor (3)
person
remote control
remote control (2)

Detection Results

[Sermanet et al. CVPR'14]

Top predictions:
trombone (confidence 26.8)
oboe (confidence 17.5)
oboe (confidence 11.5)

Groundtruth:
person
hat with a wide brim
hat with a wide brim (2)
hat with a wide brim (3)
oboe
oboe (2)
saxophone
trombone
person (2)
person (3)
person (4)

Top predictions:
microwave (confidence 5.6)
refrigerator (confidence 2.5)

Groundtruth:
bowl
microwave

Top predictions:
watercraft (confidence 72.2)
watercraft (confidence 2.1)

Groundtruth:
watercraft
watercraft (2)

Top predictions:
tennis ball (confidence 3.5)
banana (confidence 2.4)
banana (confidence 2.1)
hotdog (confidence 2.0)
banana (confidence 1.9)

Groundtruth:
strawberry
strawberry (2)
strawberry (3)
strawberry (4)
strawberry (5)
strawberry (6)
strawberry (7)
strawberry (8)
strawberry (9)
strawberry (10)
apple
apple (2)
apple (3)

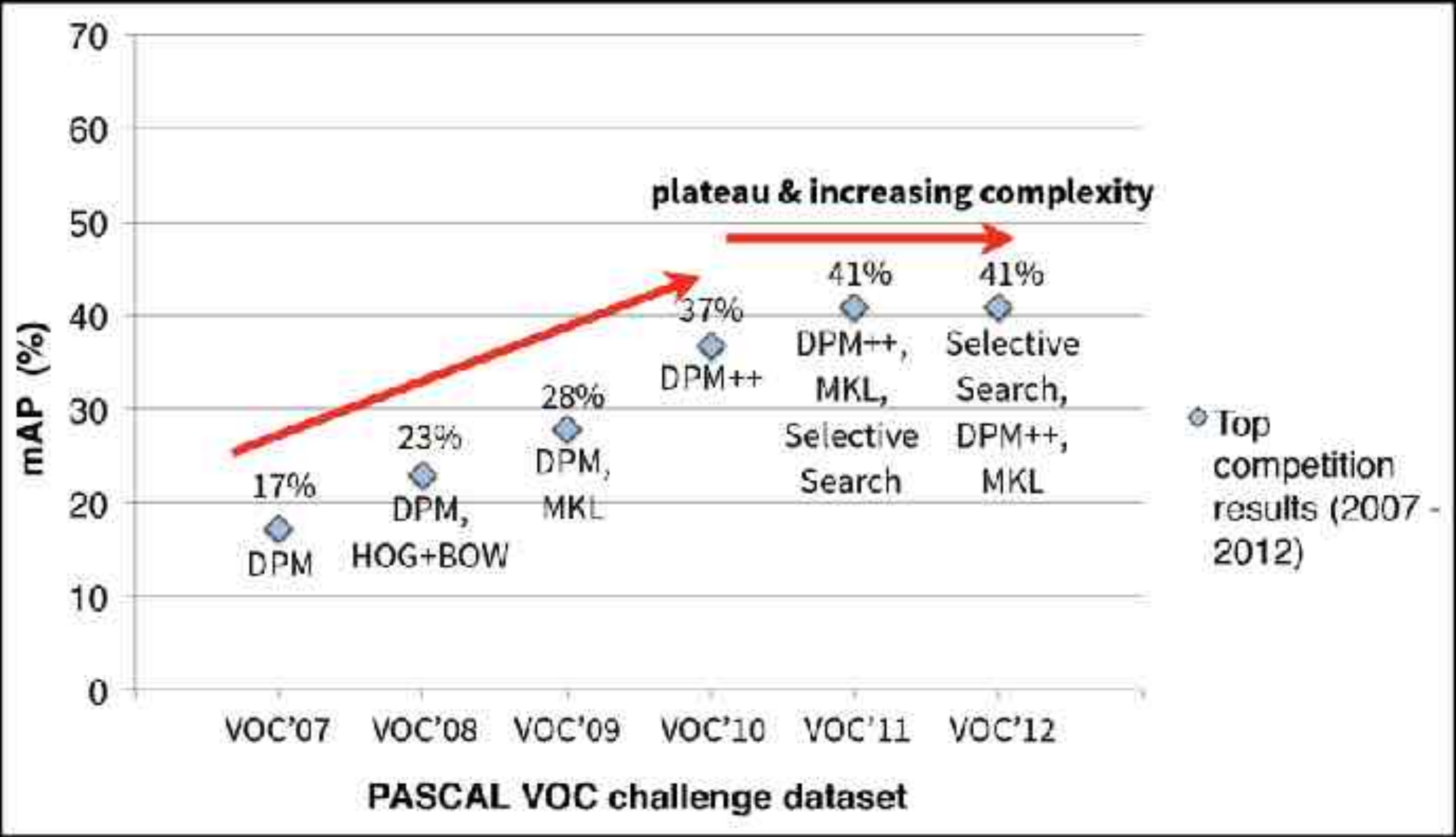
Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

EECS Berkeley

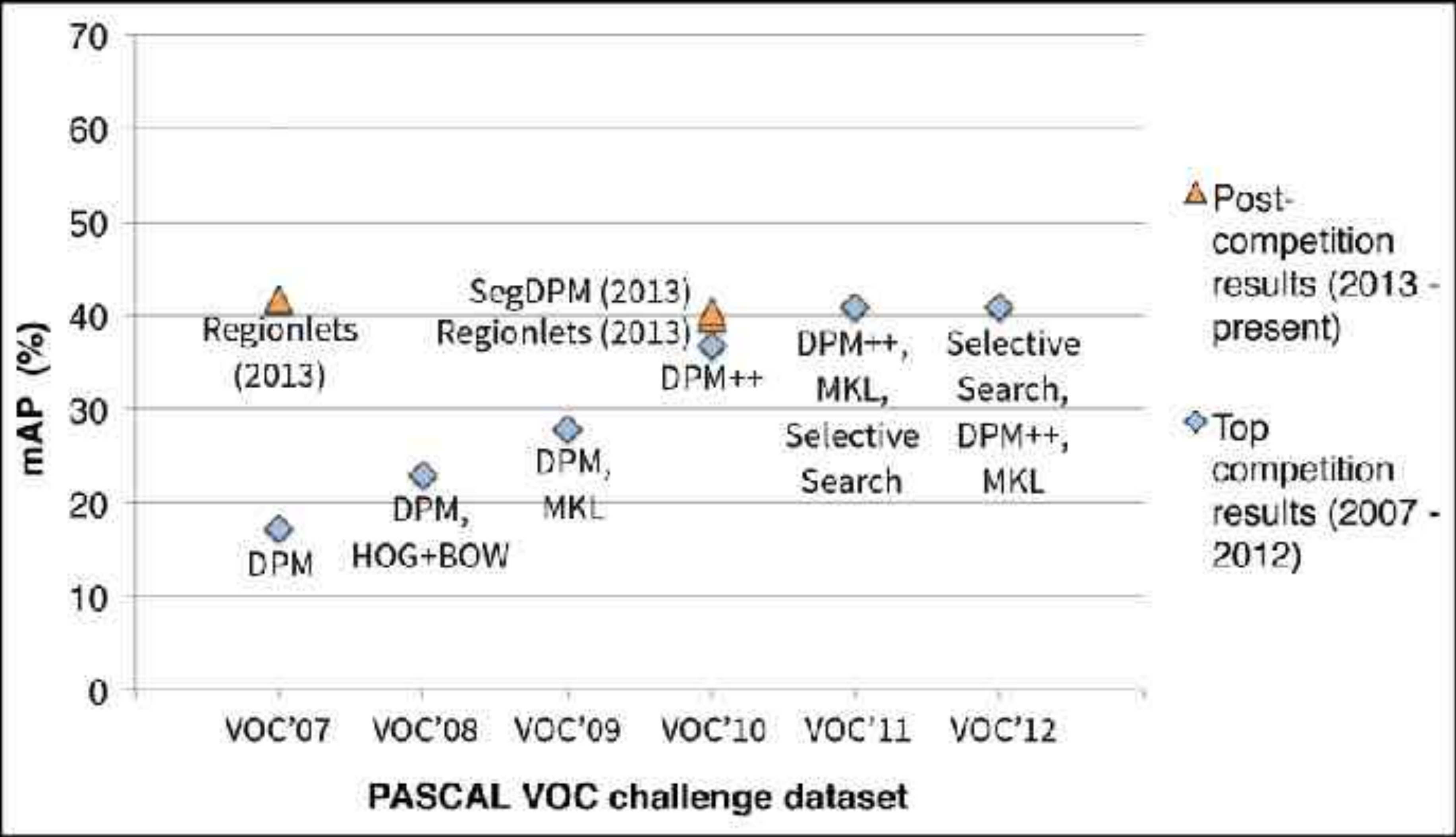
slides from: Ross Girschick - CVPR'14 talk

Complexity and the plateau



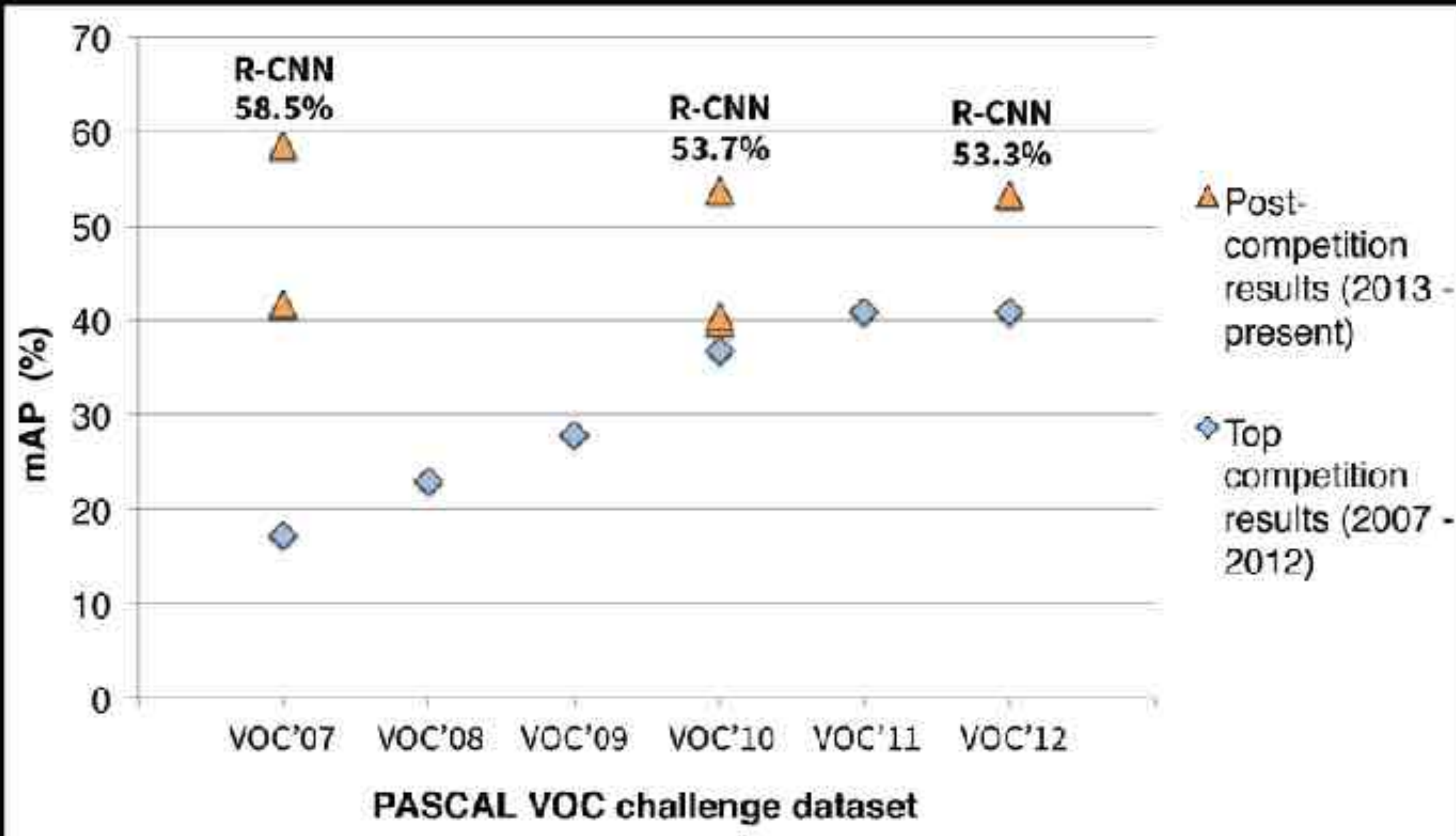
[Source: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc20{07,08,09,10,11,12}/results/index.html]

SIFT, HOG, LBP, ...



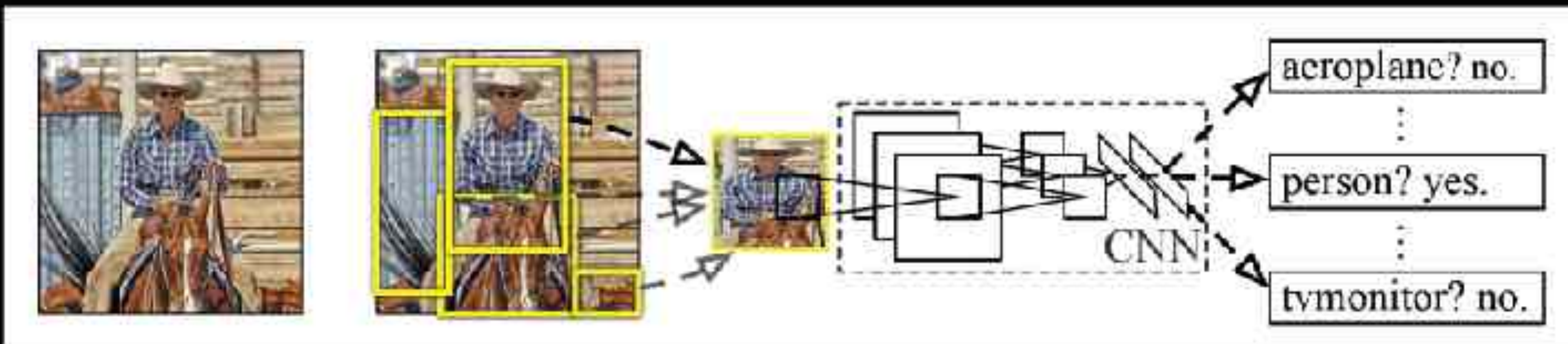
[Regionlets. Wang et al. ICCV'13] [SegDPM. Fidler et al. CVPR'13]

R-CNN: Regions with CNN features



Can we break through the PASCAL plateau with feature learning?

R-CNN: Regions with CNN features



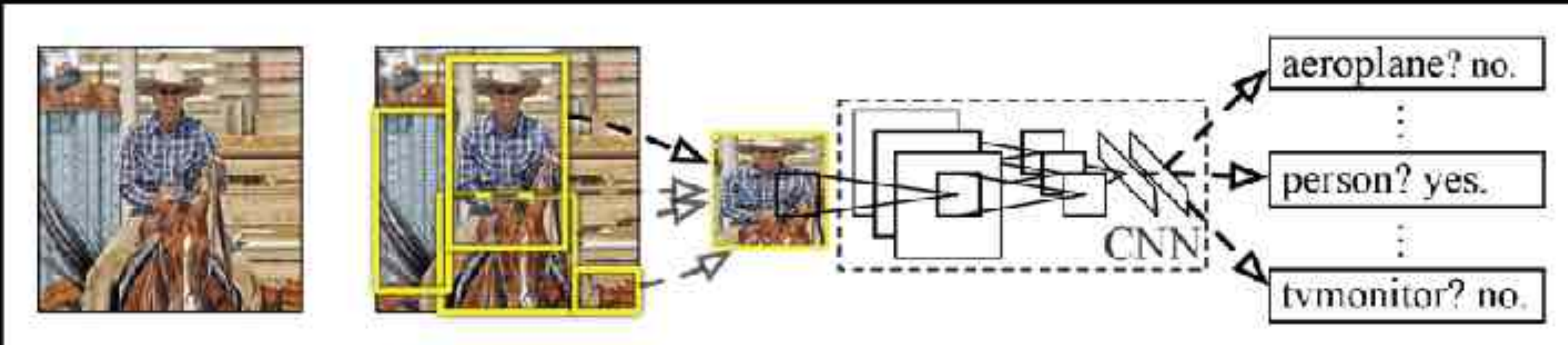
Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions (linear SVM)

R-CNN at test time: Step 1



Input image → Extract region proposals (~2k / image)

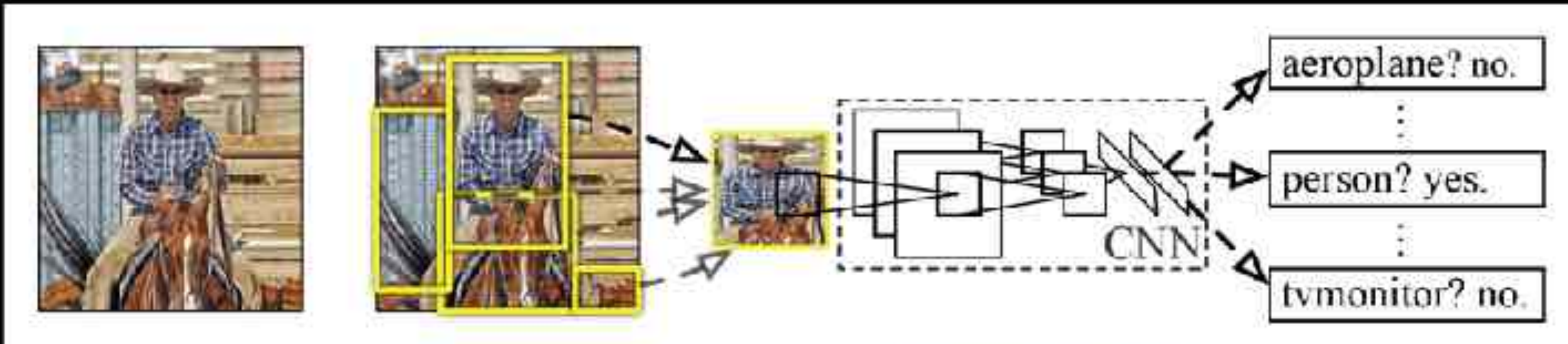
Proposal-method agnostic, many choices

- Selective Search [van de Sande, Uijlings et al.] (Used in this work)
- Objectness [Alexe et al.]
- Category independent object proposals [Endres & Hoiem]
- CPMC [Carreira & Sminchisescu]

Active area, at this CVPR

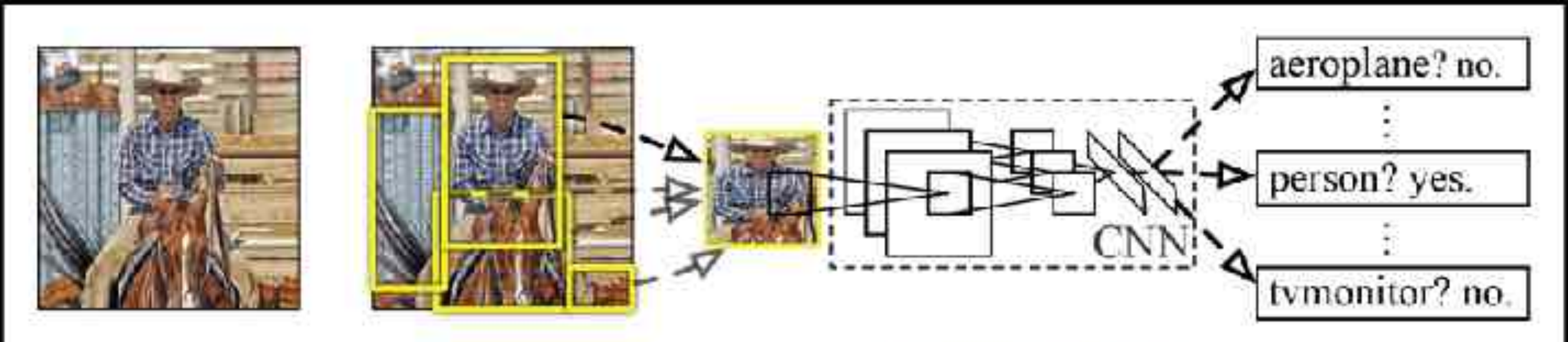
- BING [Ming et al.] – *fast*
- MCG [Arbelaez et al.] – *high-quality segmentation*

R-CNN at test time: Step 2



Input image | Extract region proposals (~2k / image) → Compute CNN features

R-CNN at test time: Step 2



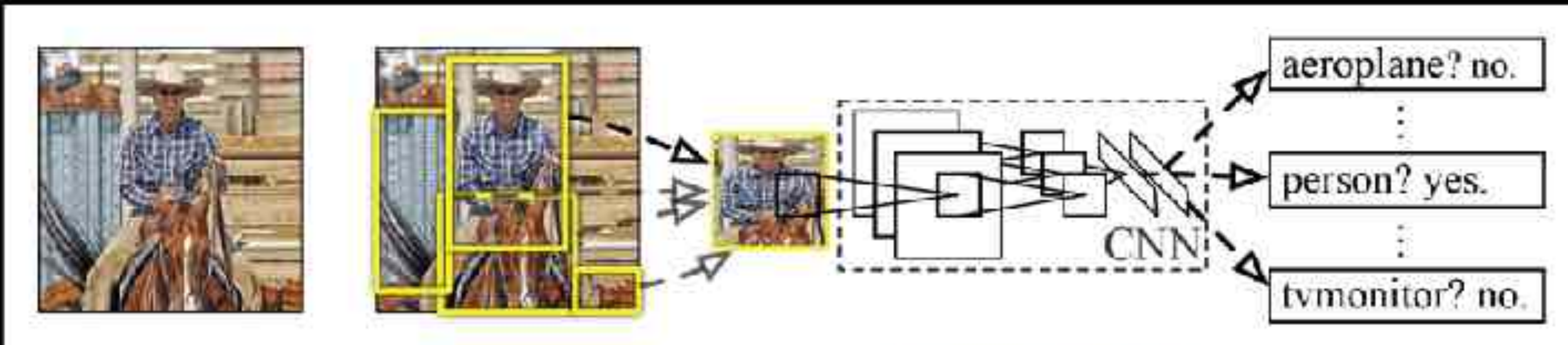
Input image

Extract region proposals (~2k / image)

→ Compute CNN features

Dilate proposal

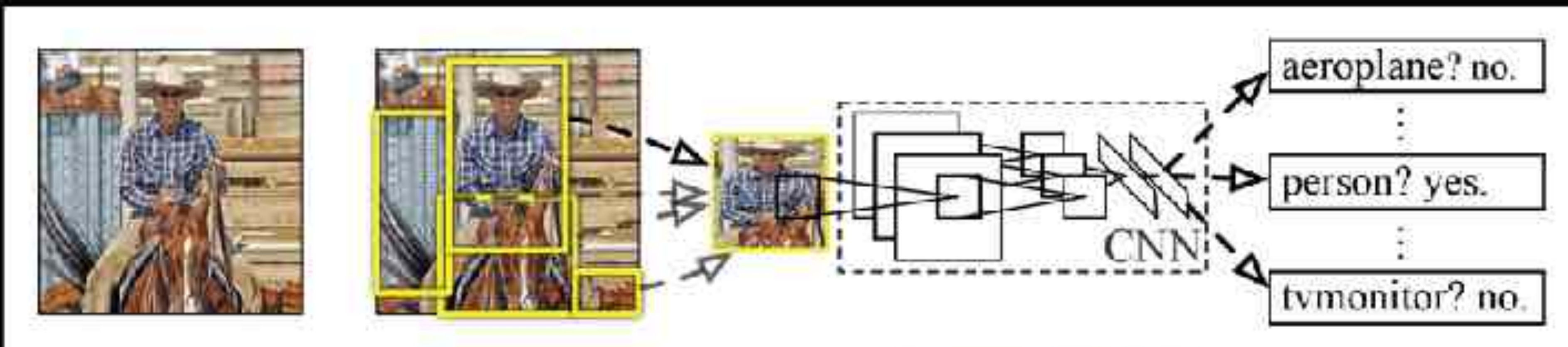
R-CNN at test time: Step 2



Input image | Extract region proposals (~2k / image) → Compute CNN features

a. Crop

R-CNN at test time: Step 2



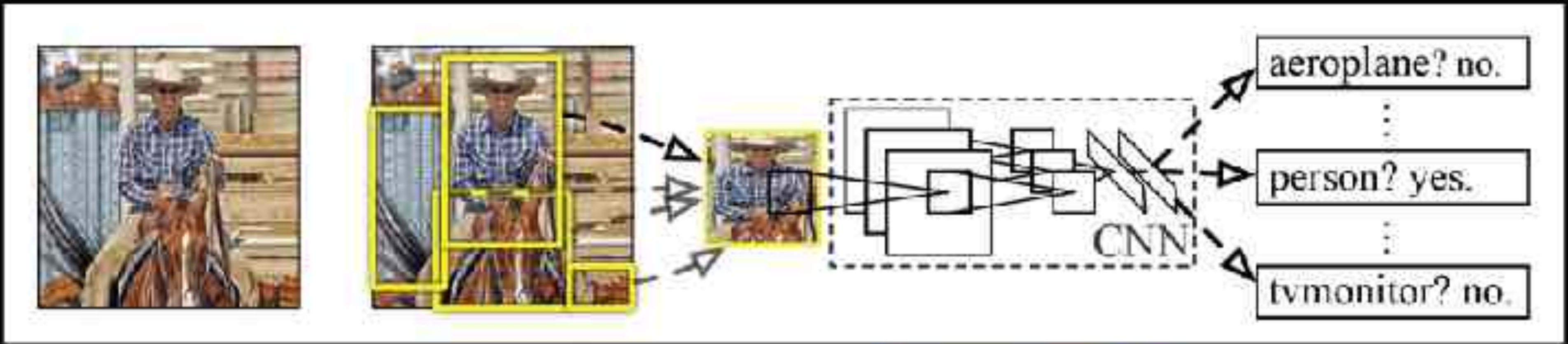
Input image | Extract region proposals (~2k / image) → Compute CNN features

a. Crop

227 x 227

b. Scale (anisotropic)

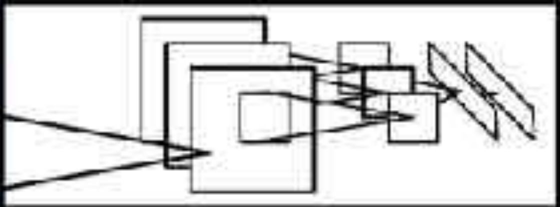
R-CNN at test time: Step 2



Input image

Extract region proposals (~2k / image)

Compute CNN features

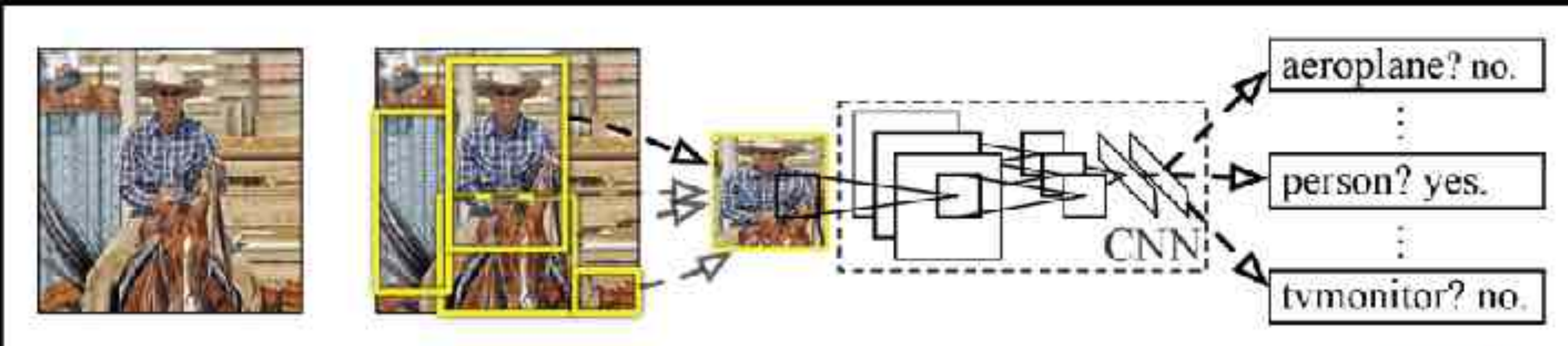


. Crop

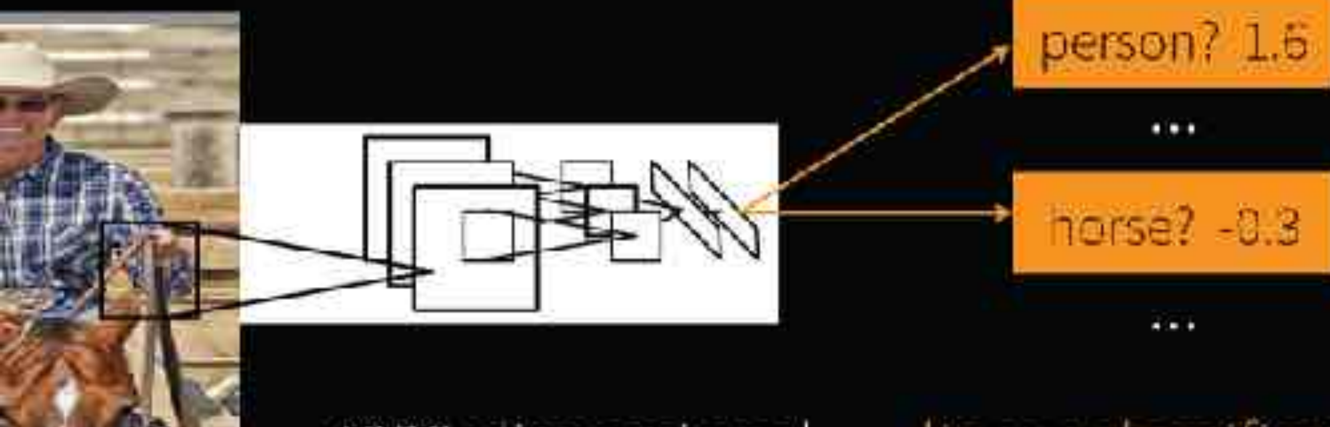
b. Scale (anisotropic)

c. Forward propagate
Output: “fc_7” features

R-CNN at test time: Step 3



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



proposal

4096-dimensional fc_7 feature vector

linear classifiers (SVM or softmax)

Step 4: Object proposal refinement

Linear regression

on CNN features

Original
proposal

Predicted
object bounding box

Bounding-box regression

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

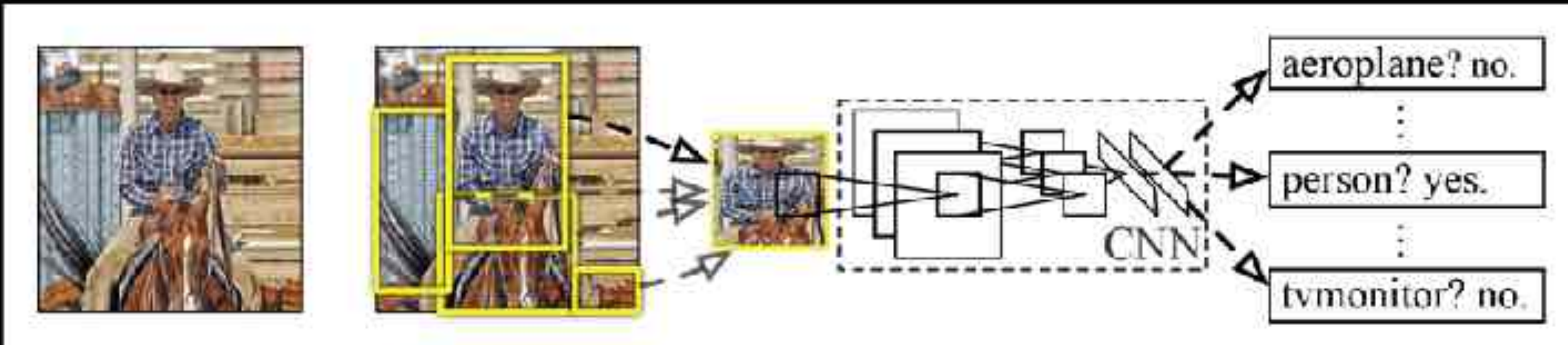
metric: mean average precision (higher is better)

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%**

metric: mean average precision (higher is better)

R-CNN at test time: Step 1



Input image → Extract region proposals (~2k / image)

Proposal-method agnostic, many choices

- Selective Search [van de Sande, Uijlings et al.] (Used in this work)
- Objectness [Alexe et al.]
- Category independent object proposals [Endres & Hoiem]
- CPMC [Carreira & Sminchisescu]

Active area, at this CVPR

- BING [Ming et al.] – *fast*
- MCG [Arbelaez et al.] – *high-quality segmentation*

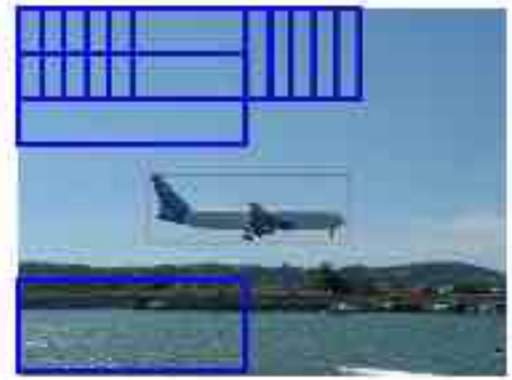
Segmentation as Selective Search for Object Recognition

K. van de Sande[1], J. Uijlings[2], T. Gevers[1], and A. Smeulders[1]
University of Amsterdam[1] and University of Trento[2]

Reading Group presentation by Esa Rahtu

(material taken from van de Sande's ICCV paper and PASCAL presentations)

Motivation



- Most current approaches use exhaustive search
  - Visit every location in an image
  - Imposes computational constraints on
    - Number of possible locations -> grid/fixed aspect ratio)
    - Evaluation cost per location -> simple features/classifiers
  - To go beyond this, we need something more sophisticated

Viola IJCV 2004
Dalal CVPR 2005
Felzenszwalb TPAMI 2010
Vedaldi ICCV 2009

Main design criteria

- **High recall**
  - We do not want to lose any objects, since they cannot be recovered later.
- Coarse locations are sufficient
  - Accurate delineation is not necessary for recognition
  - In contrary, nearby context might be useful
    -> use bounding boxes
- Fast to compute
  - Necessary when operating with large datasets
    -> <10s/image

How to obtain high recall?

- Images are intrinsically hierarchical


- Segmentation at single scale are not enough
-> hypotheses based on hierarchical grouping

Proposed method

- Start by oversegmenting the input image

"Efficient graph-based image segmentation"
Felzenszwalb and Huttenlocher, IJCV 2004

Method

- compute similarity measure between all adjacent region pairs a and b (e.g.) as:

$$S(a,b) = \alpha S_{zize}(a,b) + \beta S_{color}(a,b)$$

  - with

$$S_{size}(a,b) = 1 - \frac{\text{size}(a) + \text{size}(b)}{\text{size}(image)}$$

    encourages small regions to merge early

  - and

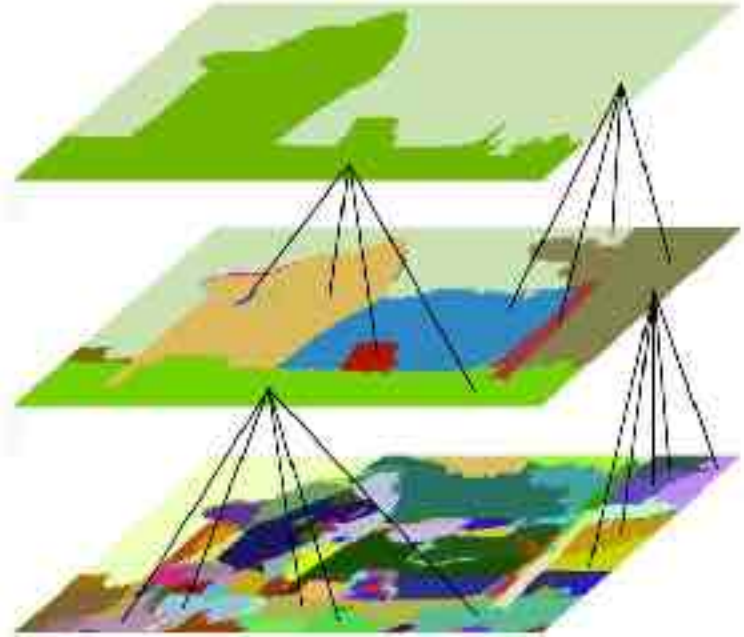
$$S_{color}(a,b) = \sum_{k=1}^{n} \min(a^k, b^k)$$

    a^k, b^k are color histograms, encouraging “similar” regions to merge

  - for slightly more elaborated similarities see their IJCV-paper

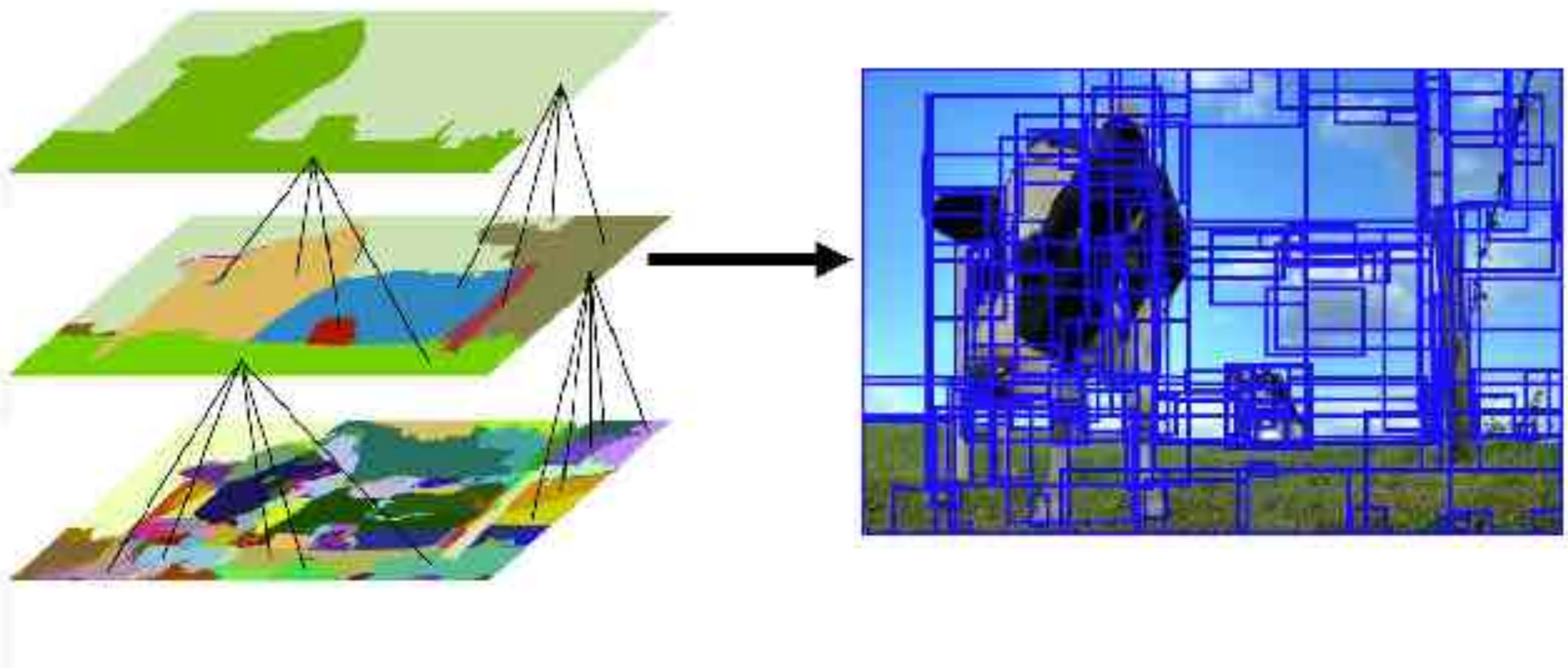
Proposed method

1. Merge two most similar regions based on *S*.
2. Update similarities between the new region and its neighbors.
3. Go back to step 1. until the whole image is a single region.

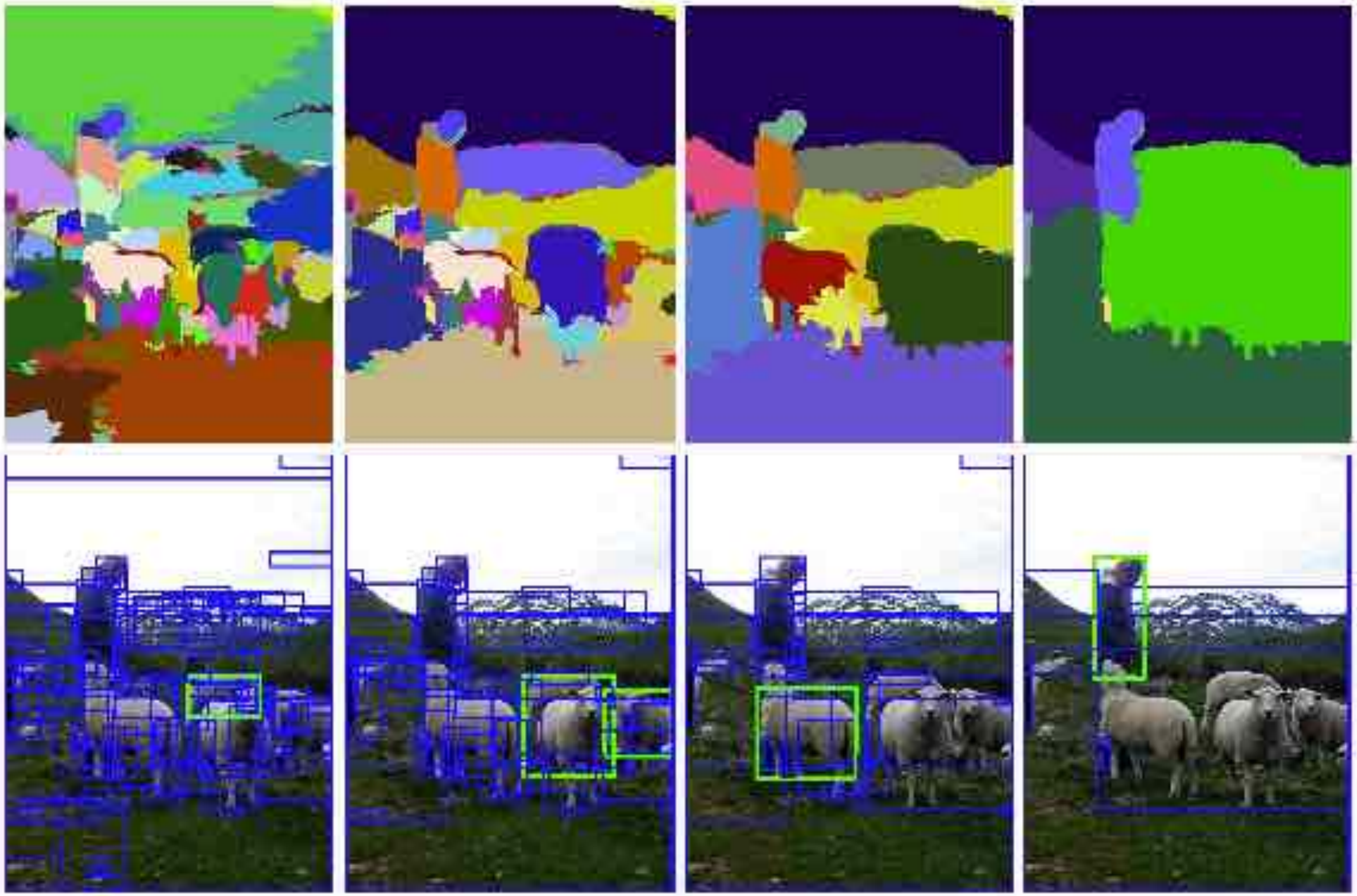


Proposed method

- Take bounding boxes of all generated regions and treat them as possible object locations.



Proposed method

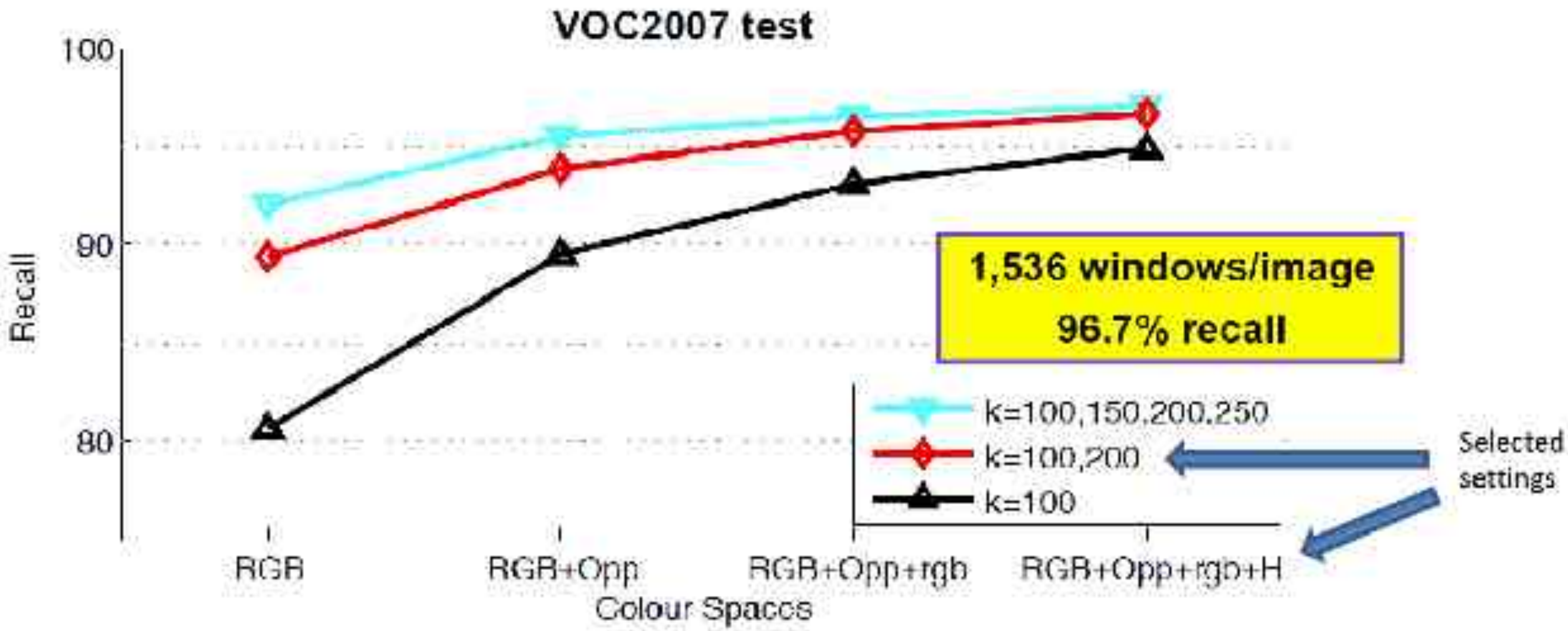


High recall revisited

- No single segmentation works for all cases
  -> diversify the set of segmentations
- Use different color spaces
  – RGB, Opponent color, normalized RGB, and hue
- Use different parameters in Felzenswalb method
  – k = [100, 150, 200, 250] (k = threshold parameter)

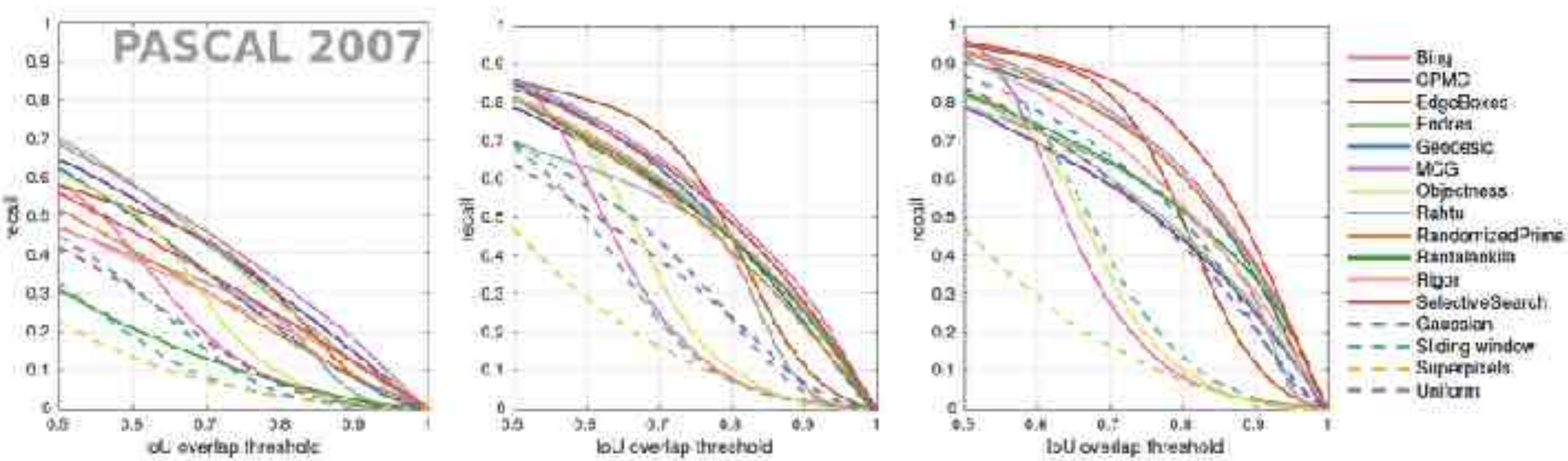
Evaluation of object hypotheses

- Recall is a proportion of objects that are covered by some box with >0.5 overlap



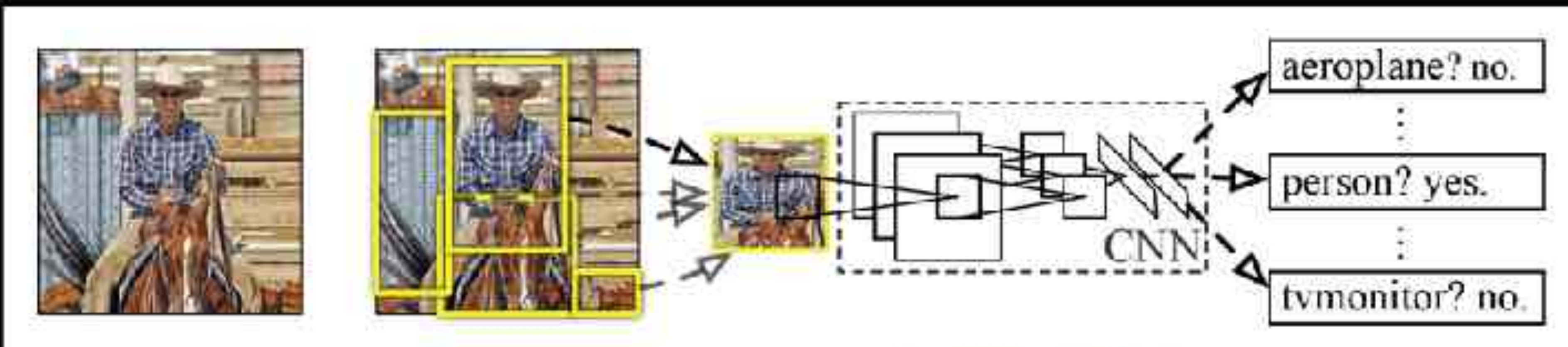
An Evaluation of Region Proposal Methods

- Hosang, Benenson, Dollar, Schiele @ Pami'15
- Recall (of ground truth bounding boxes) as a function of
  - proposal method
  - IoU (intersection over union)
  - number of proposals per image



(a) 100 proposals per image. (b) 1 000 proposals per image. (c) 10 000 proposals per image.

R-CNN at test time: Step 2



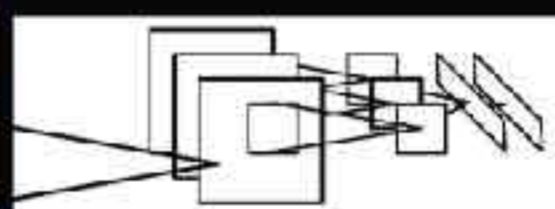
Input image

Extract region proposals (~2k / image)

Compute CNN features

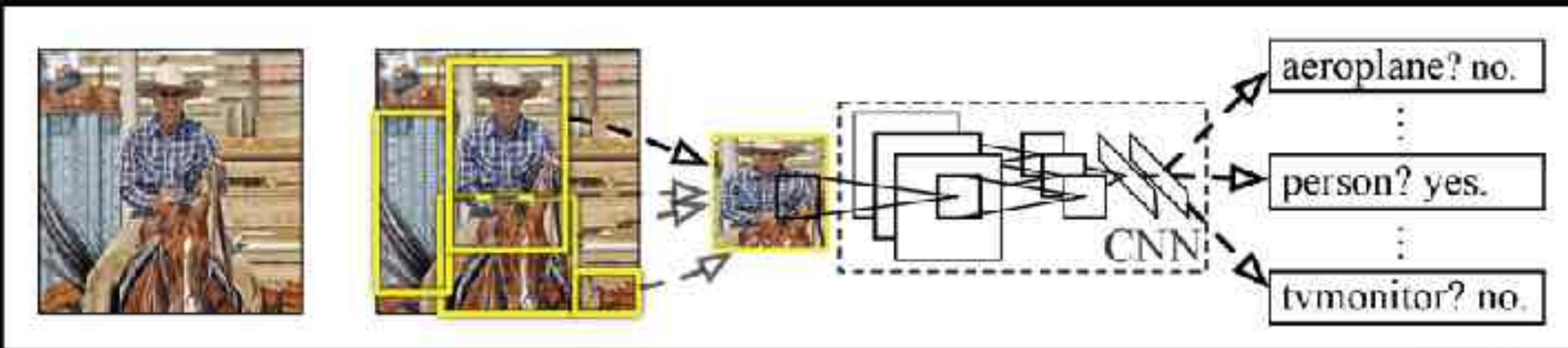
. Crop

b. Scale (anisotropic)



c. Forward propagate
Output: "fc_7" features

R-CNN at test time: Step 3

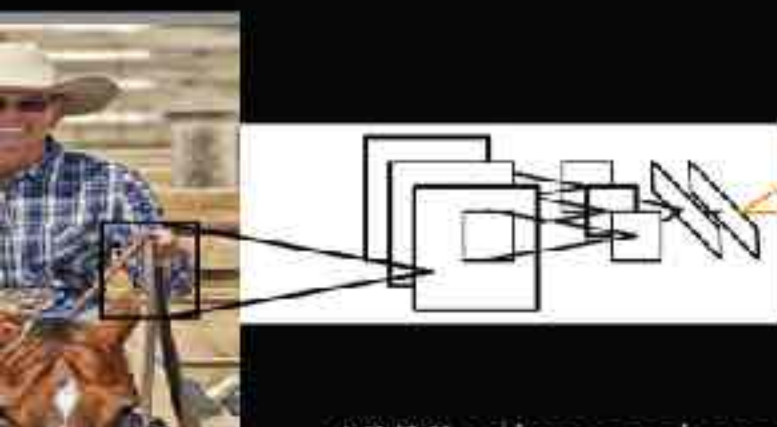


Input image

Extract region proposals (~2k / image)

Compute CNN features

→ Classify regions



person? 1.6

...

horse? -0.3

...

proposal

4096-dimensional fc_7 feature vector

linear classifiers (SVM or softmax)

Training R-CNN

Bounding-box labeled detection data is scarce

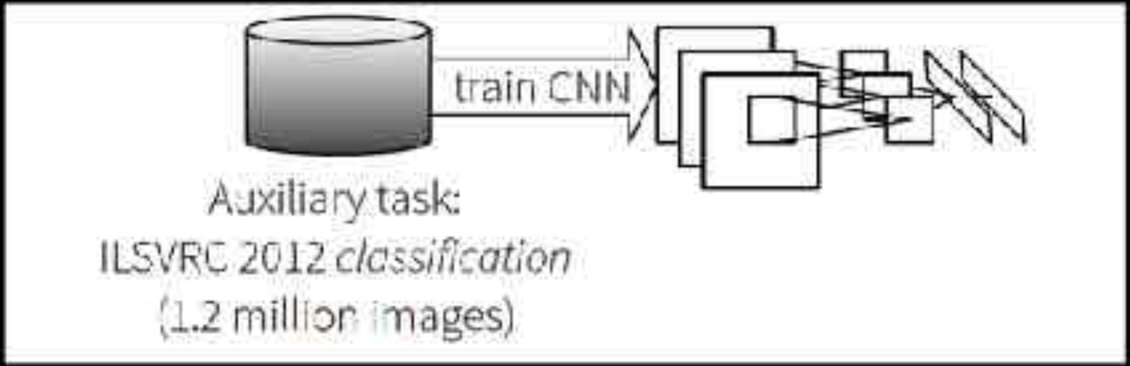
Key insight:

Use *supervised* pre-training on a data-rich auxiliary task and *transfer* to detection

R-CNN training: Step 1

Supervised pre-training

Train a SuperVision CNN* for the 1000-way ILSVRC image classification task

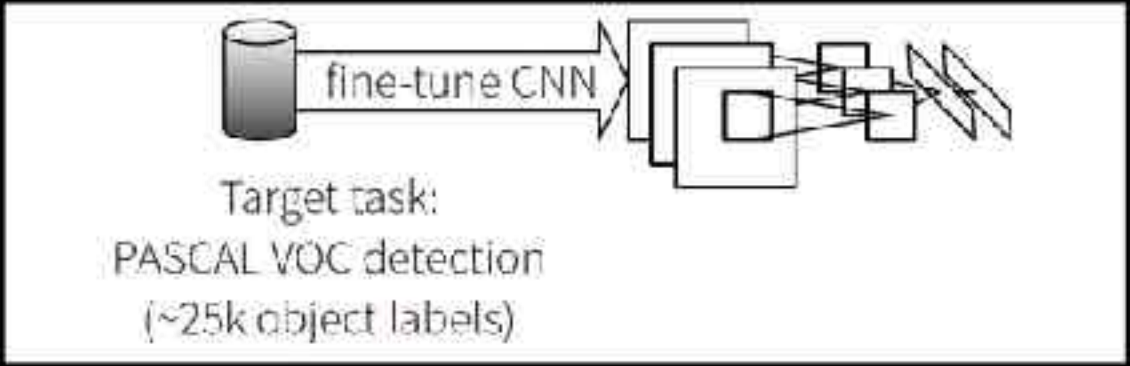


*Network from Krizhevsky, Sutskever & Hinton. NIPS 2012
Also called "AlexNet"

R-CNN training: Step 2

Fine-tune the CNN for detection

Transfer the representation learned for ILSVRC classification to PASCAL (or ImageNet detection)



Try Caffe! http://caffe.berkeleyvision.org

- Clean & fast CNN library in C++ with Python and MATLAB interfaces
- Used by R-CNN for training, fine-tuning, and feature computation

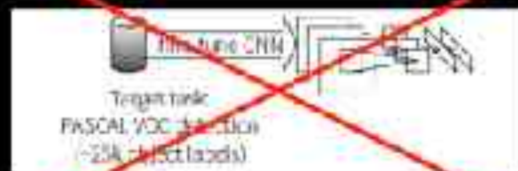
R-CNN training: Step 3

Train detection SVMs
(With the softmax classifier from fine-tuning mAP decreases from 54% to 51%)

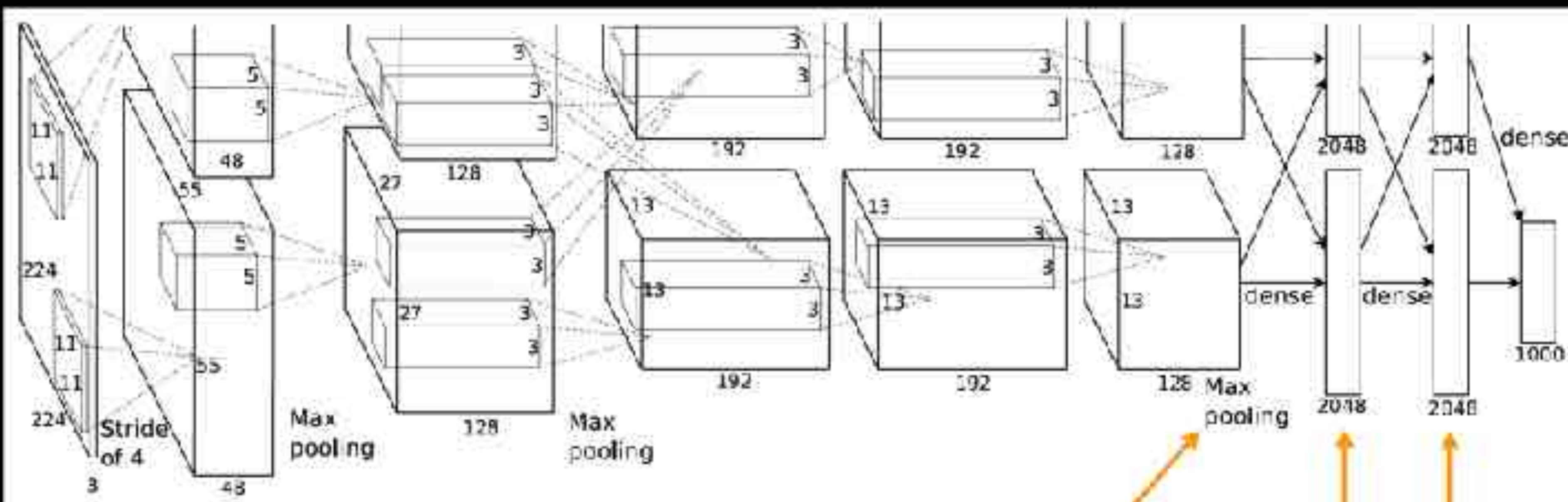
Ablation: skip fine-tuning

Step 1: pre-train

~~Step 2: fine-tune~~



Step 3: train SVMs



pool$_5$

fc$_6$

fc$_7$

Pre-trained CNN + SVMs (no FT)

	VOC 2007	VOC 2010
Regionlets (Wang et al. 2013)	41.7%	39.7%
SegDPM (Fidler et al. 2013)		**40.4%**
R-CNN pool_5	44.2%	
R-CNN fc_6	**46.2%**	
R-CNN fc_7	44.7%	

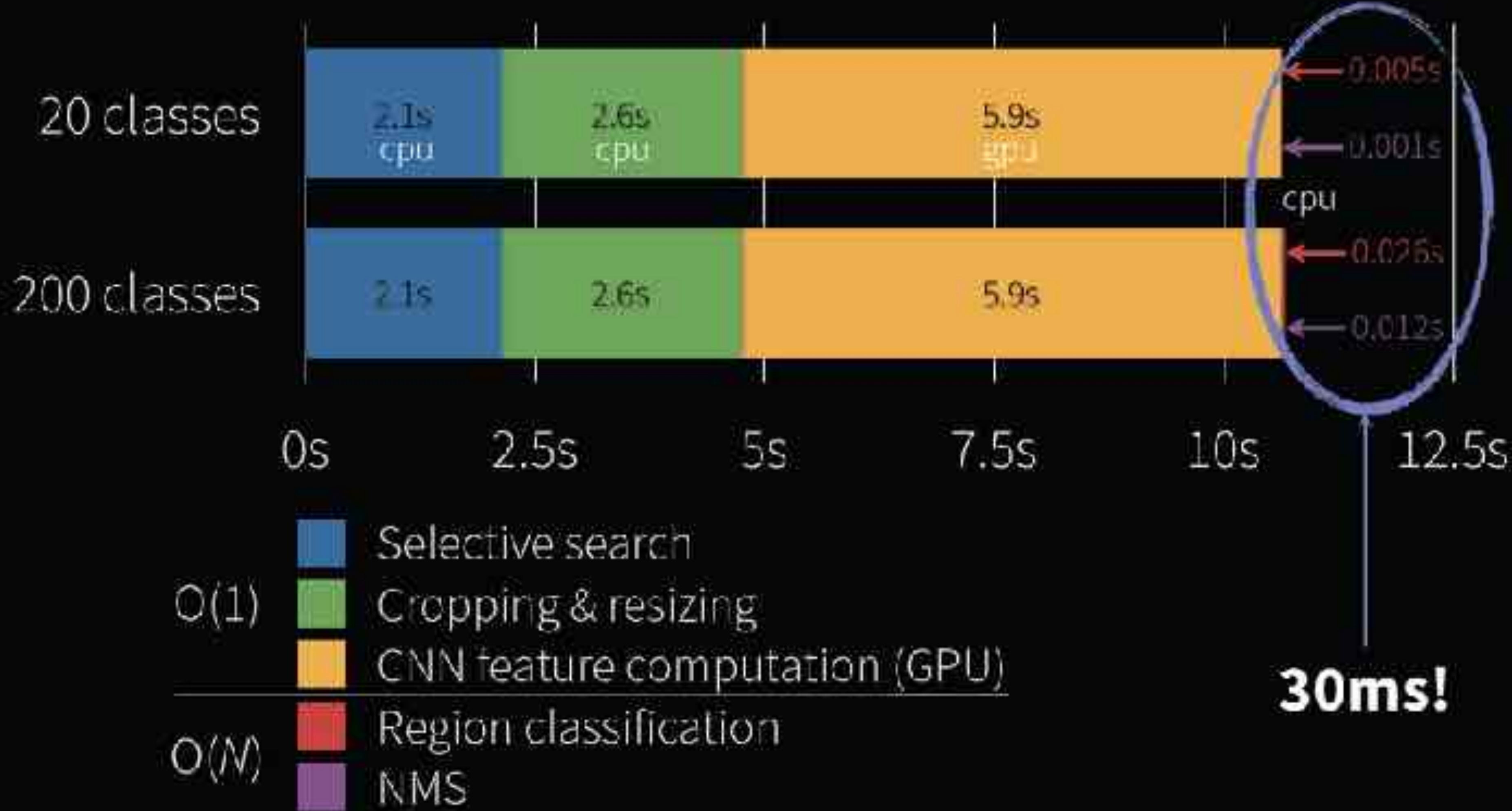
metric: mean average precision (higher is better)

Compare with fine-tuned R-CNN

		VOC 2007	VOC 2010
	Regionlets (Wang et al. 2013)	41.7%	39.7%
	SegDPM (Fidler et al. 2013)		40.4%
	R-CNN $pool_5$	44.2%	
	R-CNN fc_6	46.2%	
	R-CNN fc_7	44.7%	
fine-tuned	R-CNN FT $pool_5$	47.3%	
fine-tuned	R-CNN FT fc_6	53.1%	
fine-tuned	R-CNN FT fc_7	**54.2%**	**50.2%**

metric: mean average precision (higher is better)

Detection speed & scalability



Hardware: Intel Core i7-3930K 3.2Ghz and NVIDIA Tesla K20c

We thank NVIDIA for generous hardware donations.

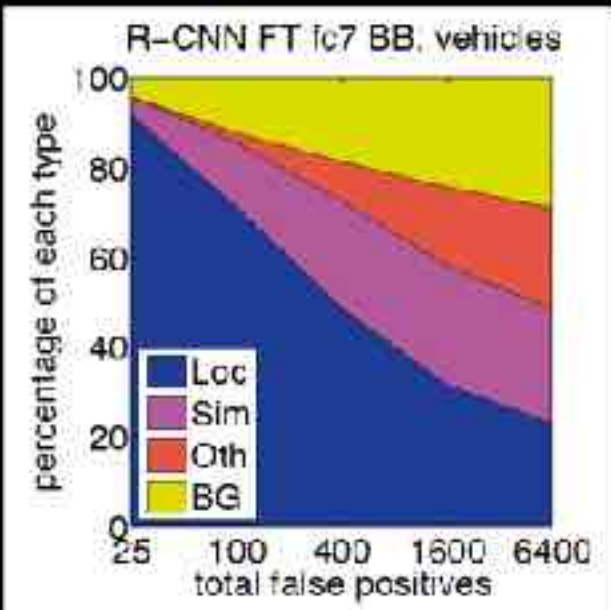
Top bicycle FPs (AP = 72.8%)

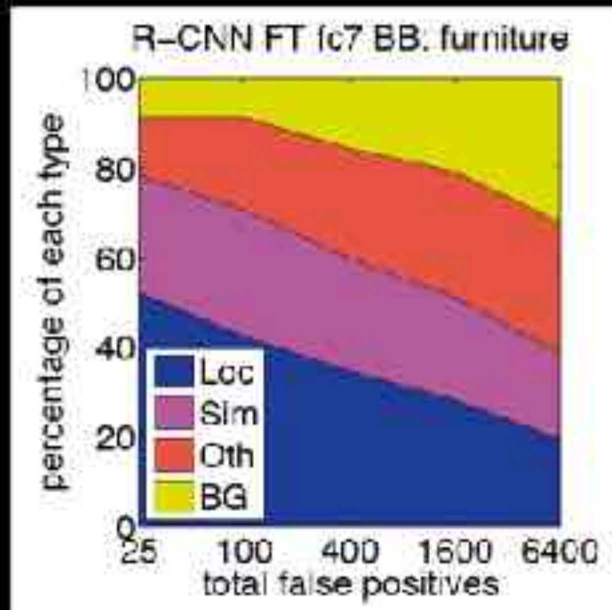
Top bicycle FPs (AP = 72.8%)

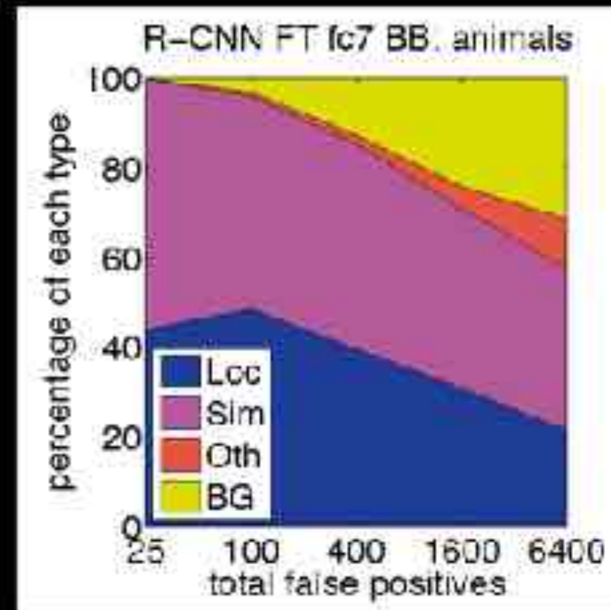
False positive #15

Unannotated bicycle

False positive type distribution







Loc = localization

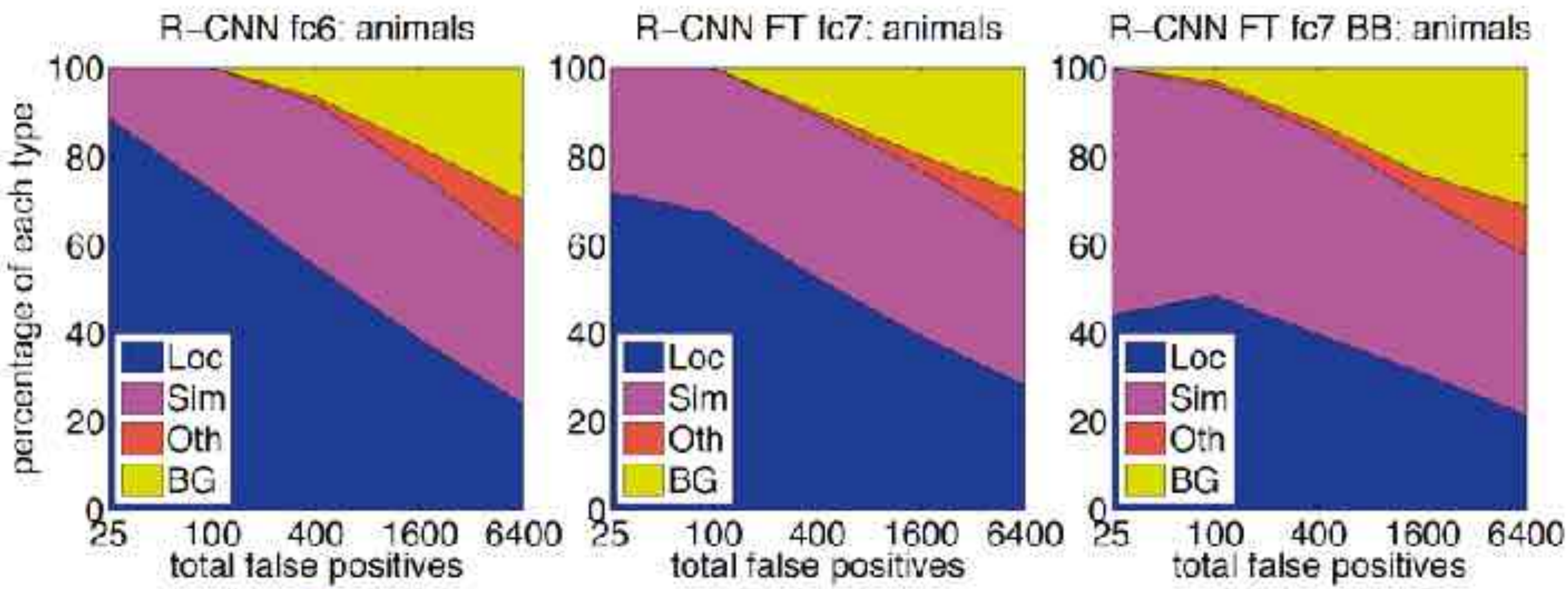
Oth = other / dissimilar classes

Sim = similar classes

BG = background

Analysis software: D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing Error in Object Detectors. ECCV, 2012.

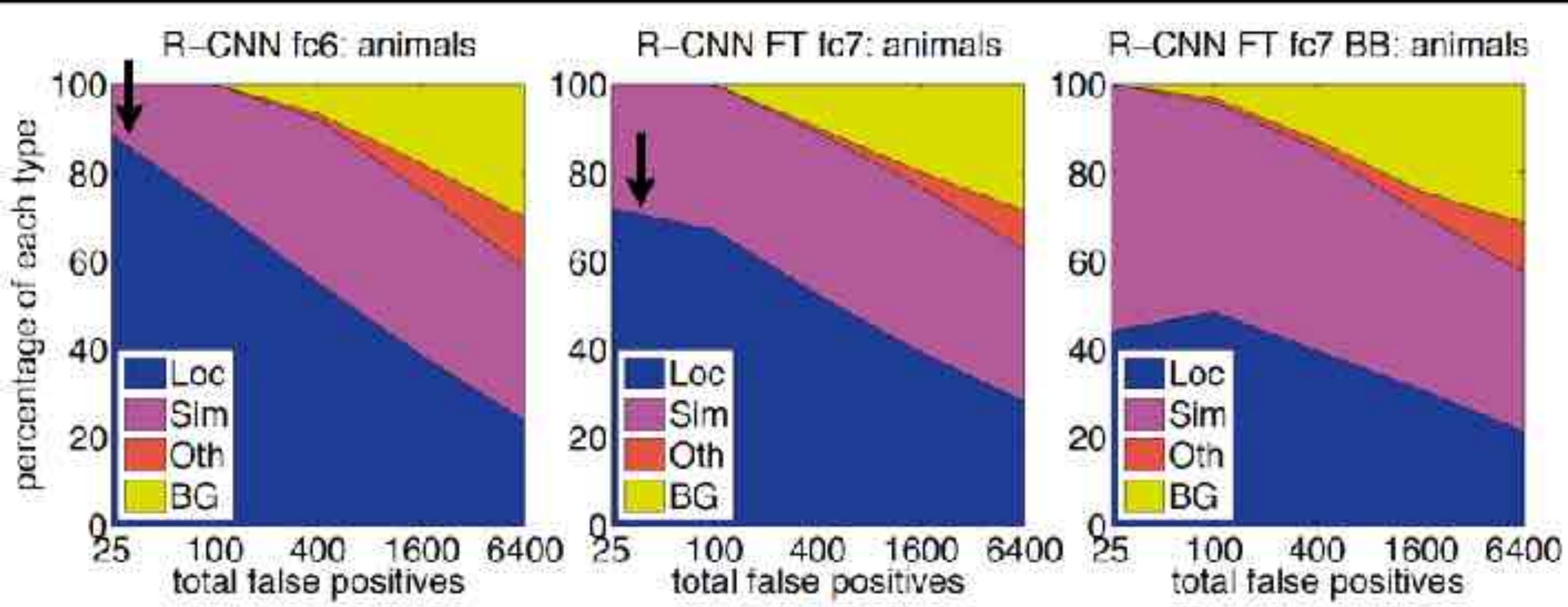
False positive analysis



No fine-tuning

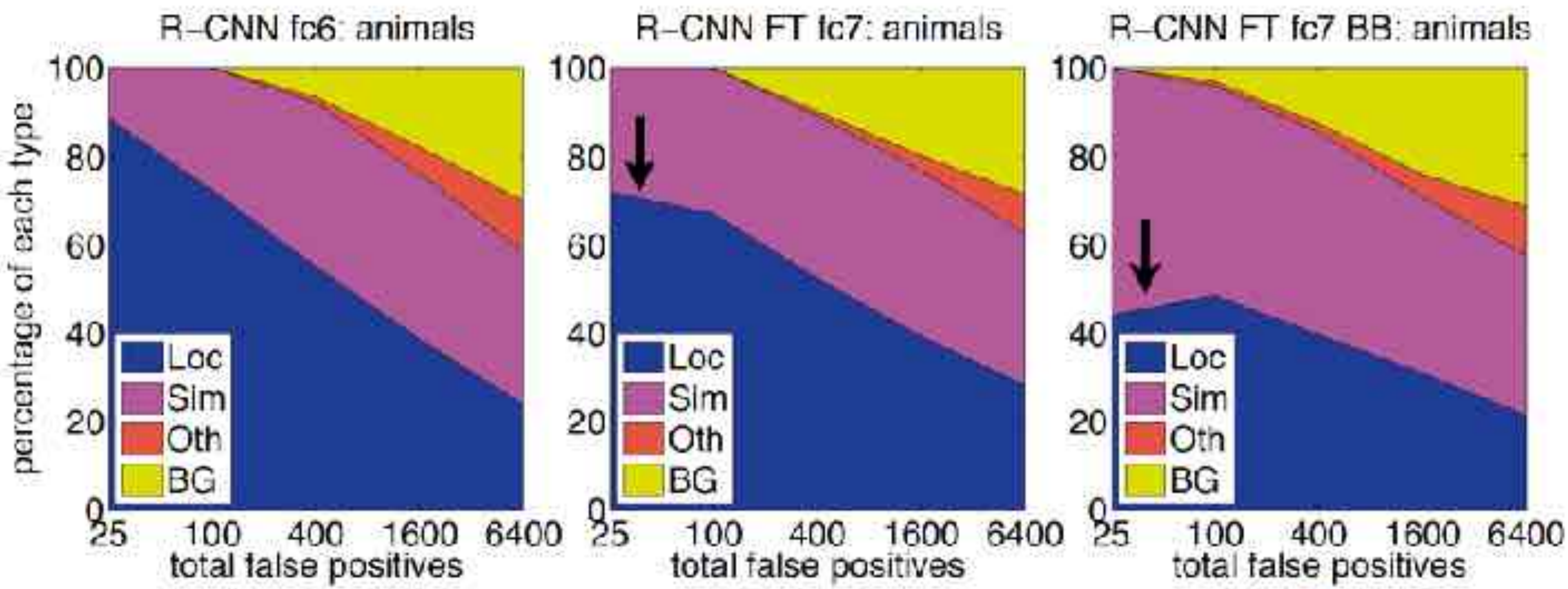
Analysis software: D. Hoiem, Y. Chodpathumwan, and Q. Dai. "Diagnosing Error in Object Detectors." ECCV, 2012.

False positive analysis



After fine-tuning

False positive analysis



After bounding-box regression