

# Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

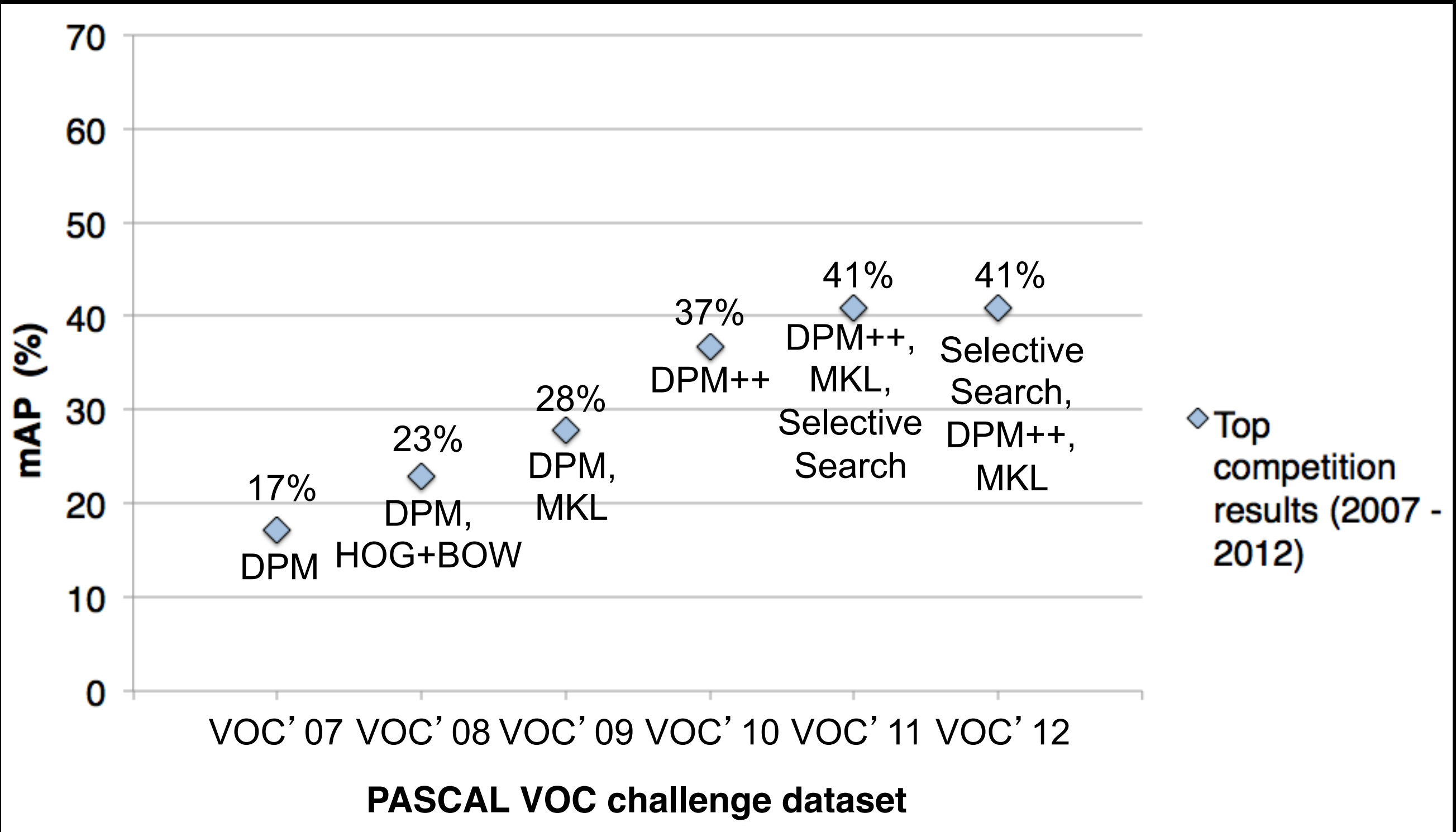


# Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation

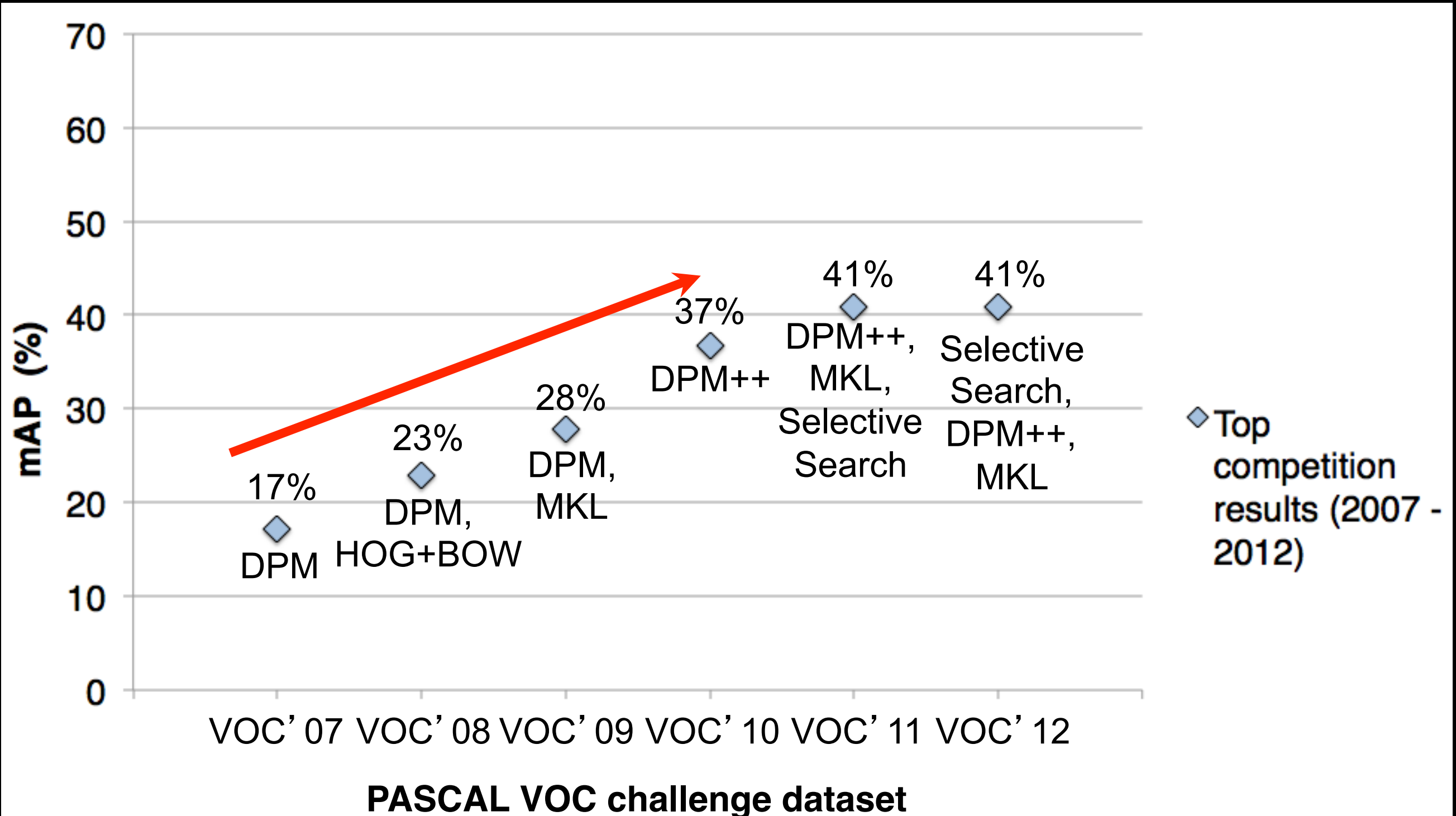
Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik



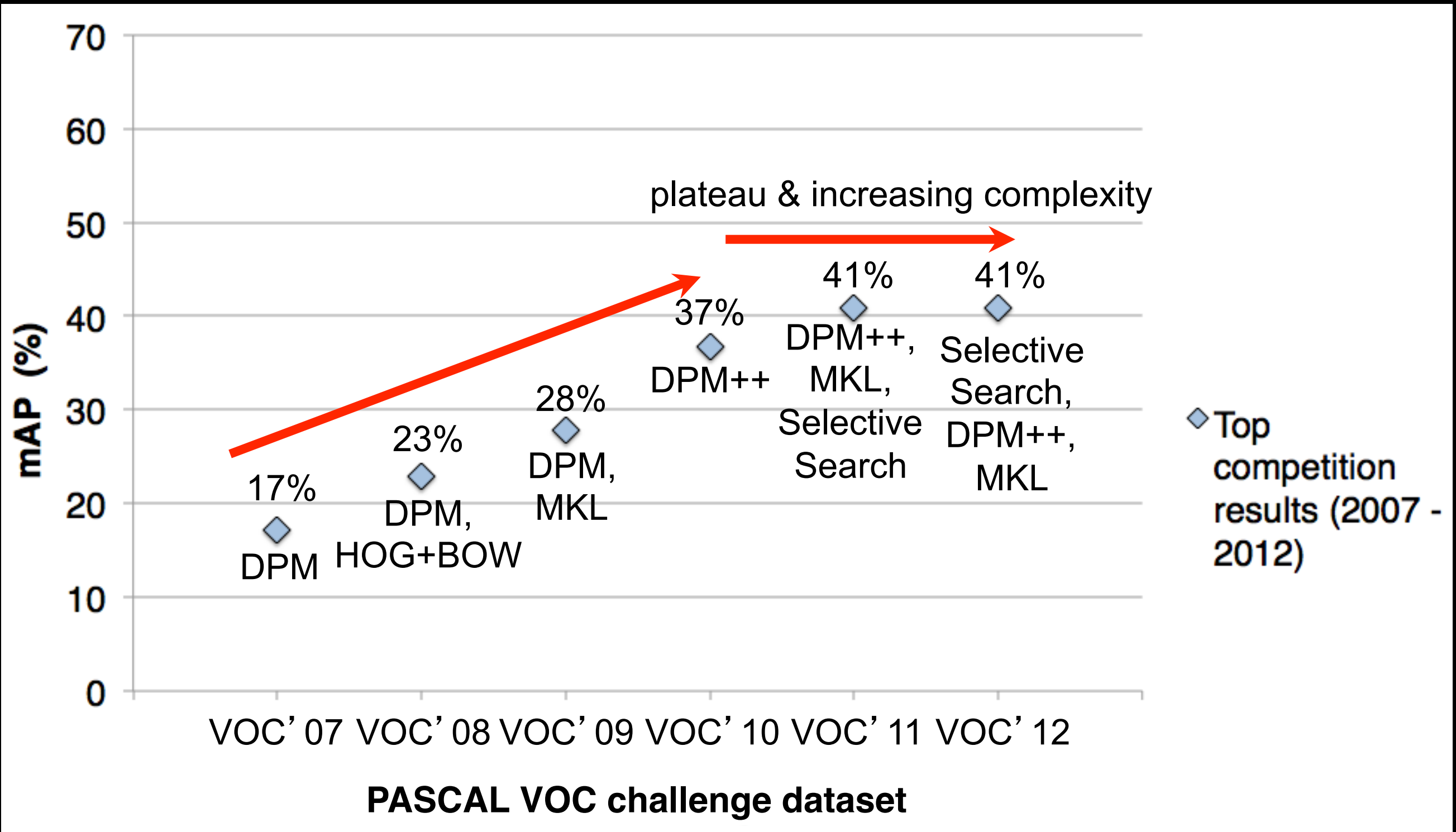
# PASCAL VOC detection history



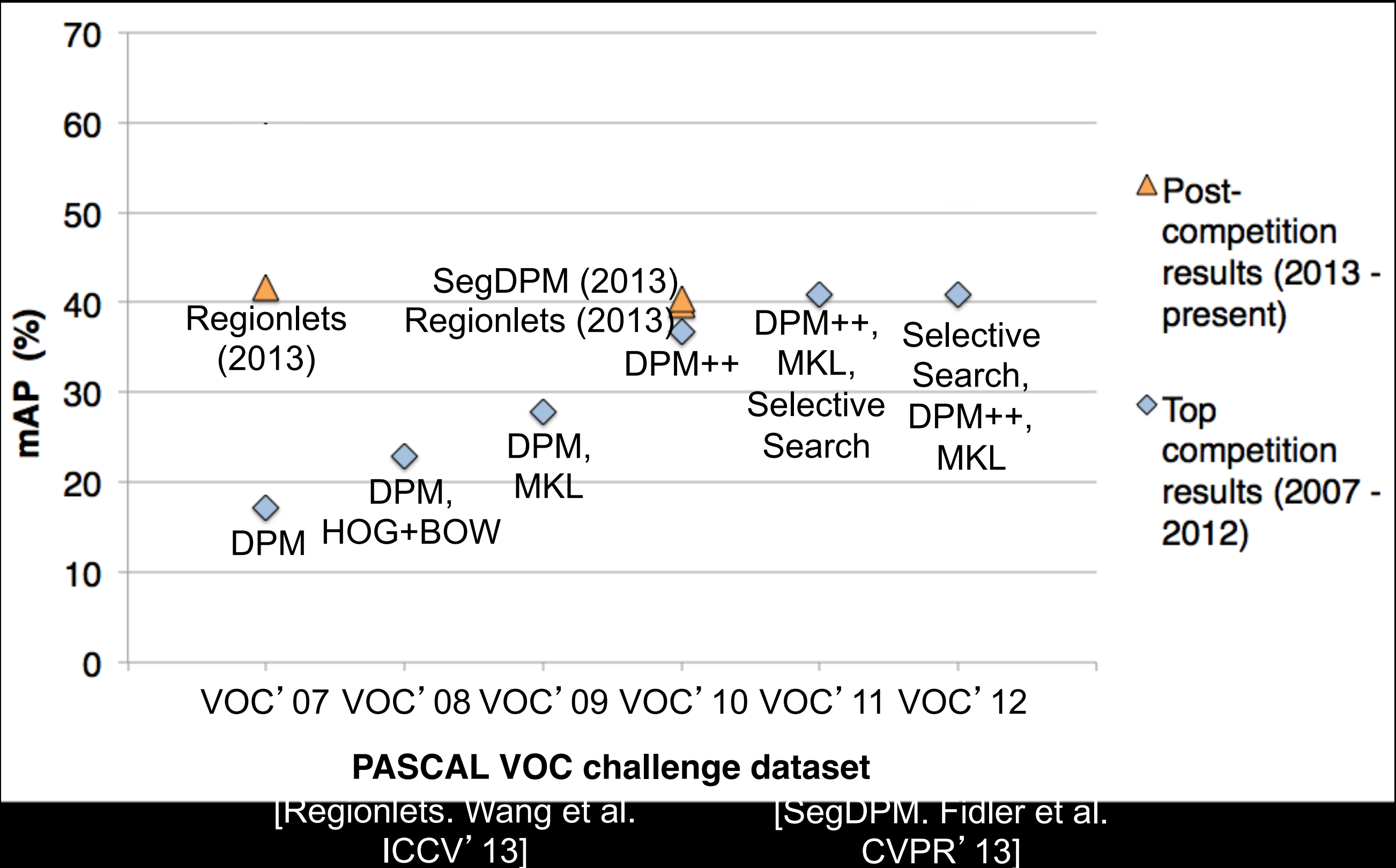
# A rapid rise in performance



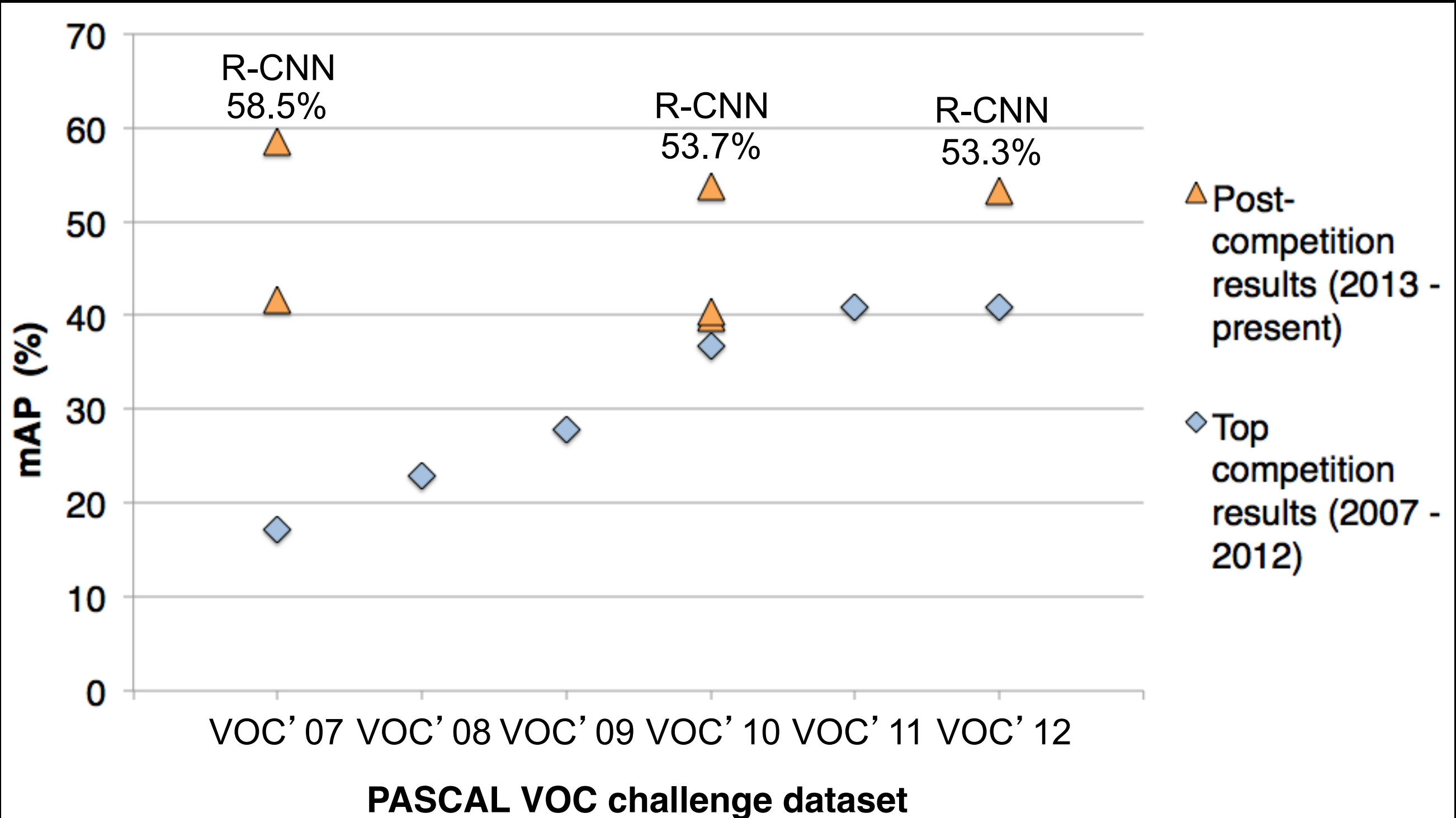
# Complexity and the plateau



# SIFT, HOG, LBP, ...

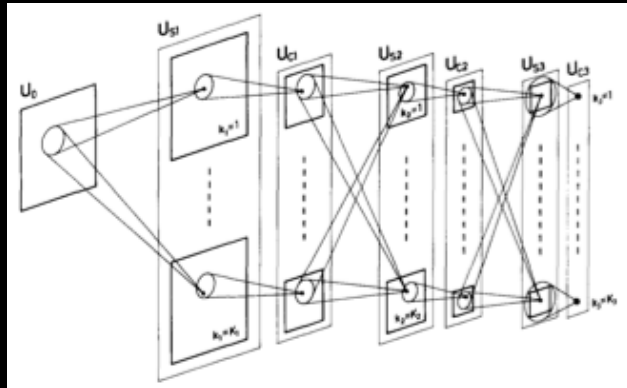


# R-CNN: Regions with CNN features



# Feature learning with CNNs

---

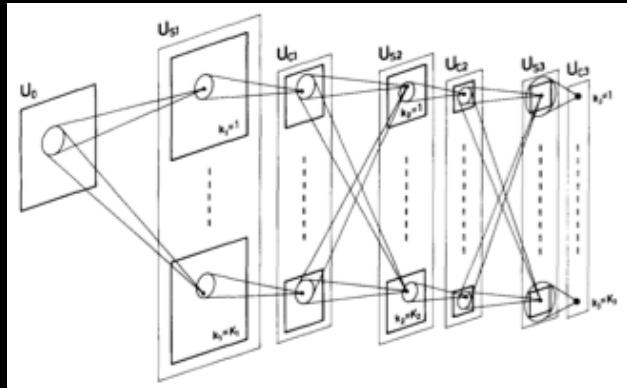


Fukushima 1980  
Neocognitron



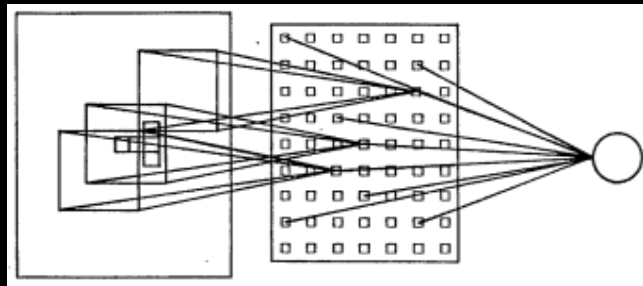
# Feature learning with CNNs

---



Fukushima 1980

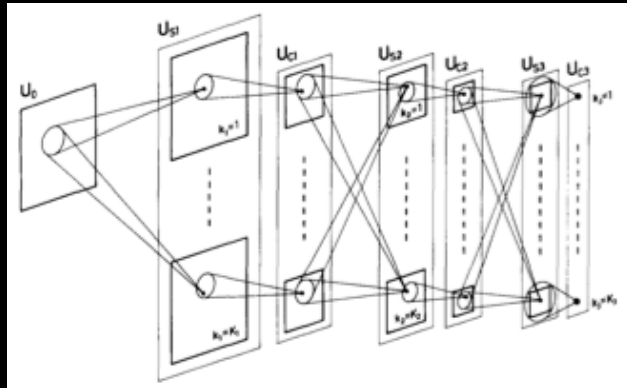
Neocognitron



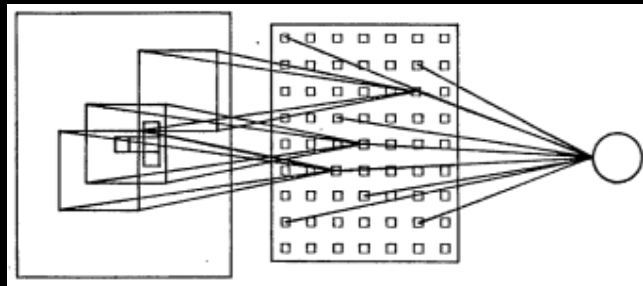
Rumelhart, Hinton, Williams  
1986

“T” versus “C” problem

# Feature learning with CNNs

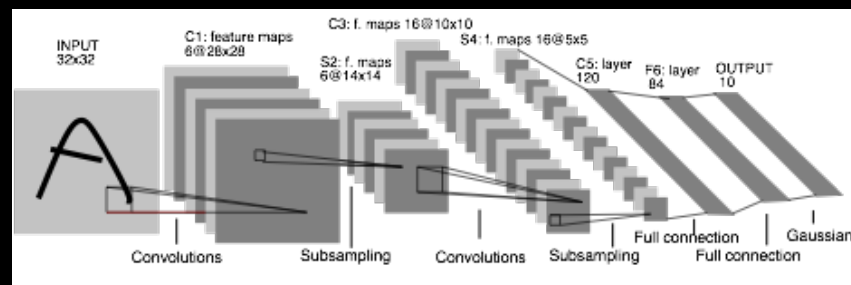


Fukushima 1980  
Neocognitron



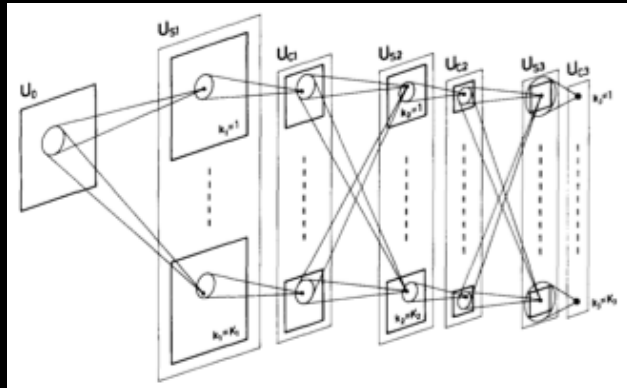
Rumelhart, Hinton, Williams  
1986

“T” versus “C” problem

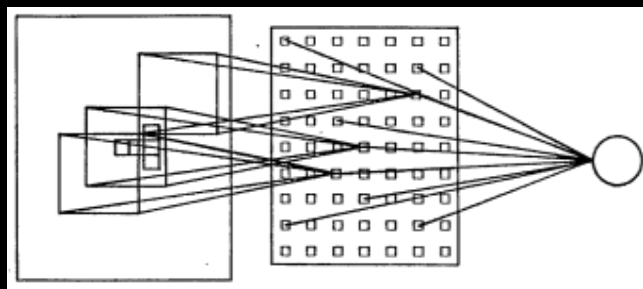


LeCun et al. 1989-1998  
Handwritten digit reading / OCR

# Feature learning with CNNs

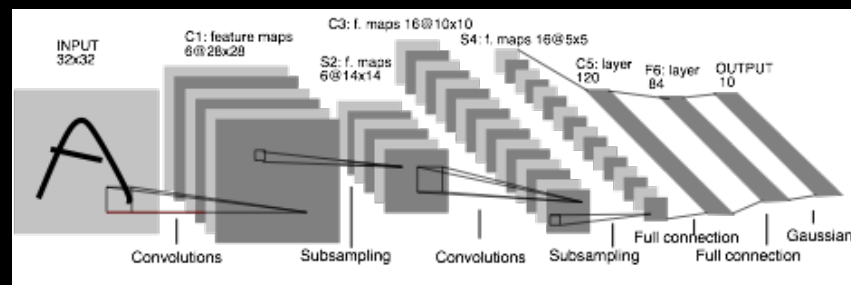


Fukushima 1980  
Neocognitron



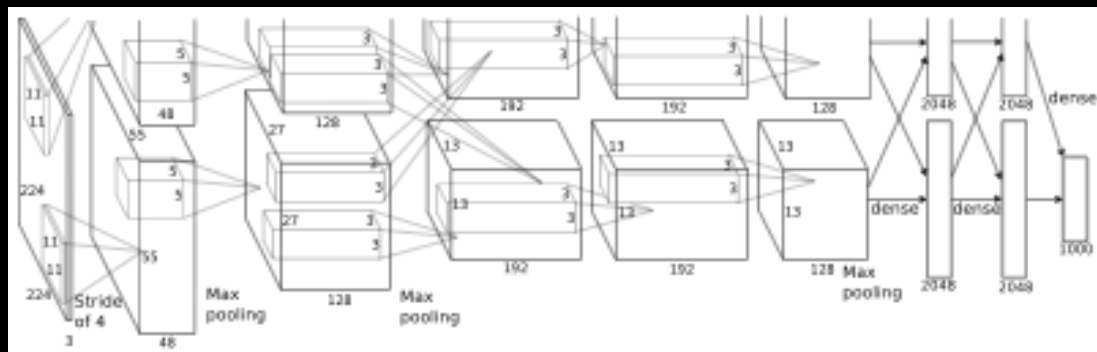
Rumelhart, Hinton, Williams  
1986

“T” versus “C” problem



LeCun et al. 1989-1998  
Handwritten digit reading / OCR

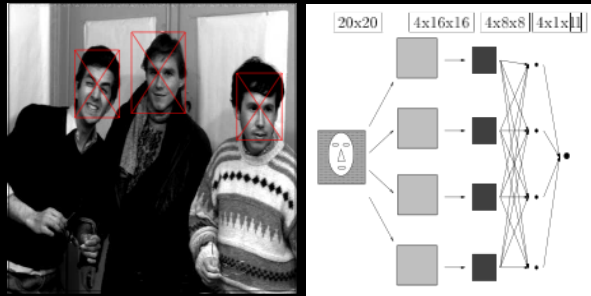
...



Krizhevsky, Sutskever,  
Hinton 2012

ImageNet classification  
breakthrough  
“SuperVision” CNN

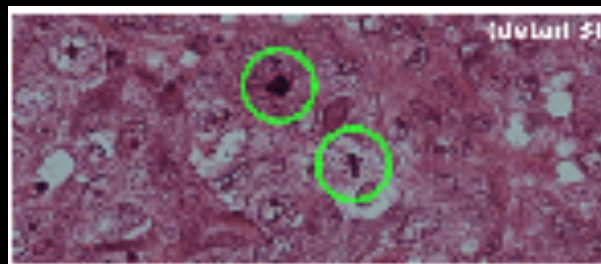
# CNNs for object detection



Vaillant, Monrocq, LeCun 1994  
Multi-scale face detection



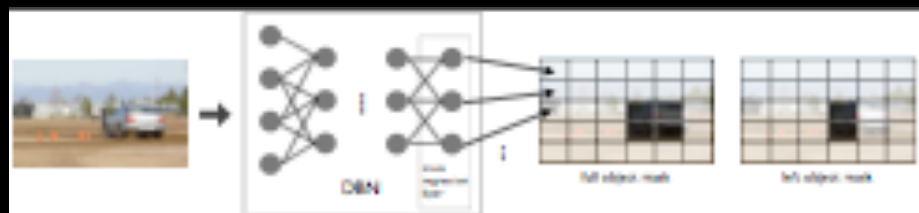
LeCun, Huang, Bottou 2004  
NORB dataset



Cireşan et al. 2013  
Mitosis detection



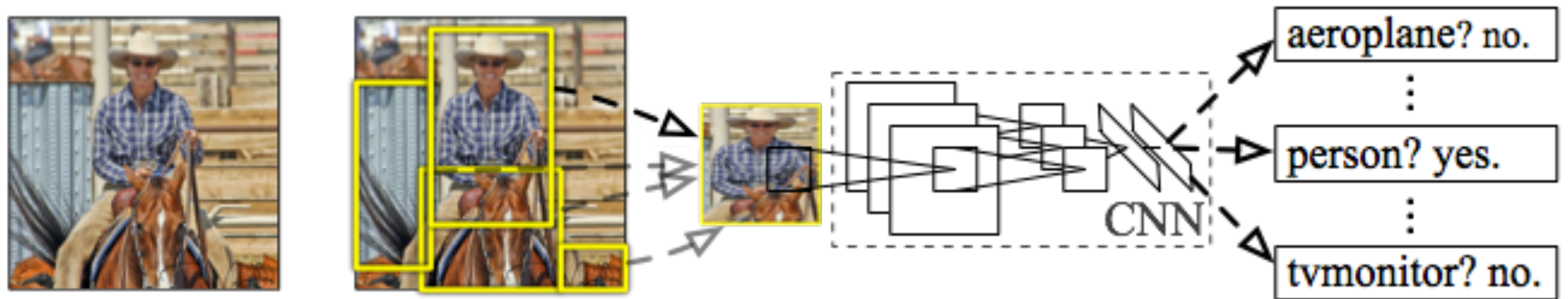
Sermanet et al. 2013  
Pedestrian detection



Szegedy, Toshev, Erhan 2013  
PASCAL detection (VOC' 07 mAP 30.5)

Can we break through the PASCAL plateau  
with feature learning?

# R-CNN: Regions with CNN features



Input  
image

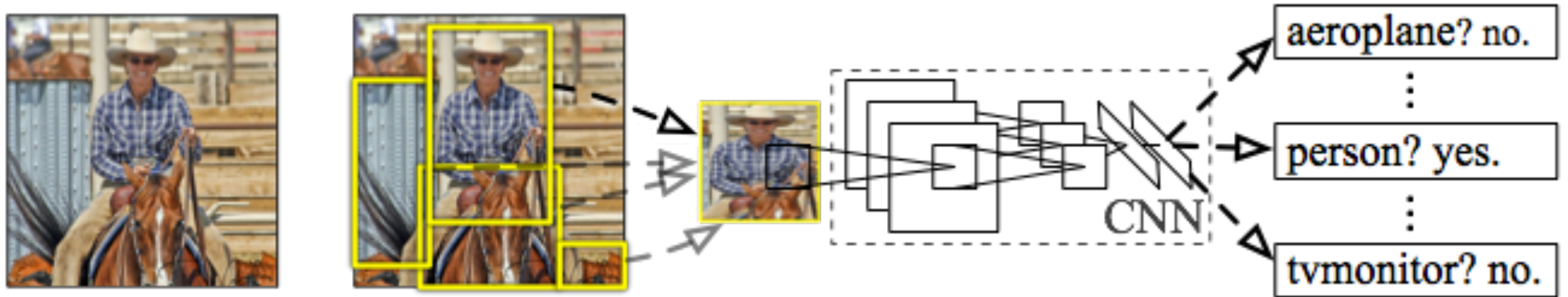
Extract region  
proposals (~2k / image)

Compute CNN  
features

Classify regions  
(linear SVM)



# R-CNN at test time: Step 1



Input image  $\xrightarrow{\text{Extract region proposals (~2k / image)}}$

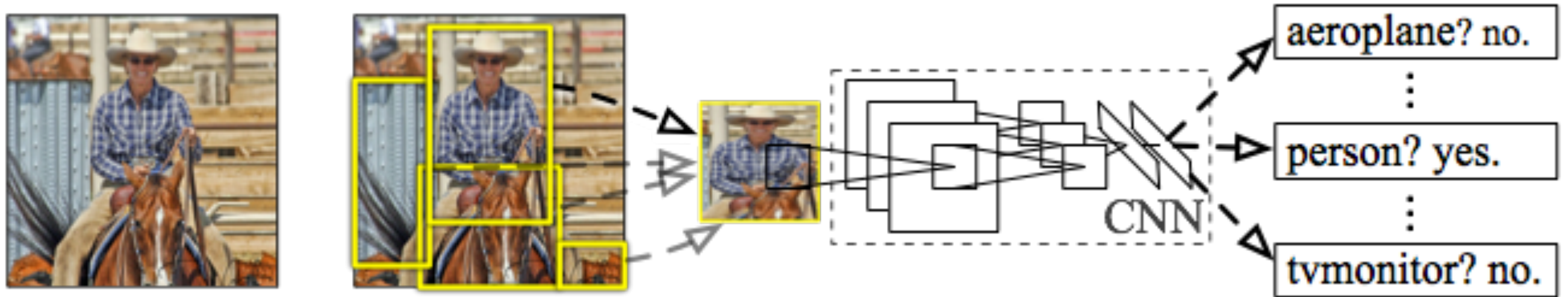
Proposal-method agnostic, many choices

- 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

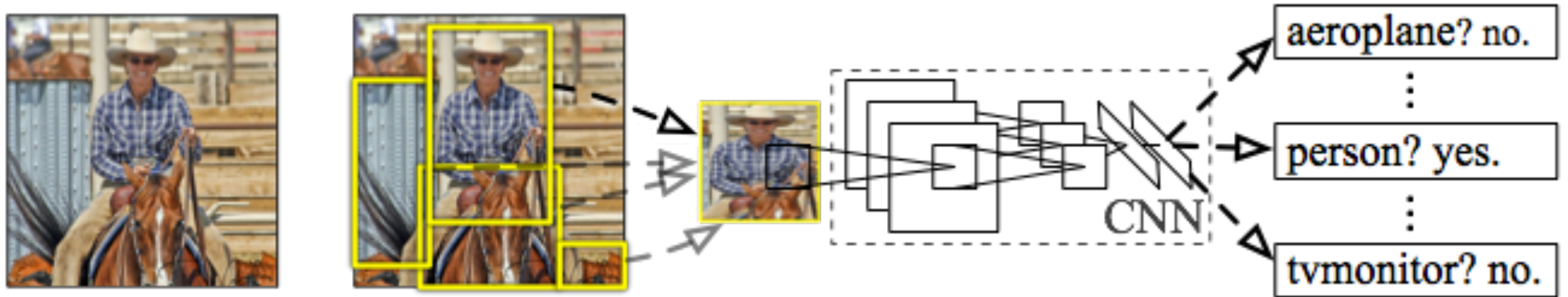


Input image      Extract region proposals (~2k / image)      **Compute CNN features**





# R-CNN at test time: Step 2

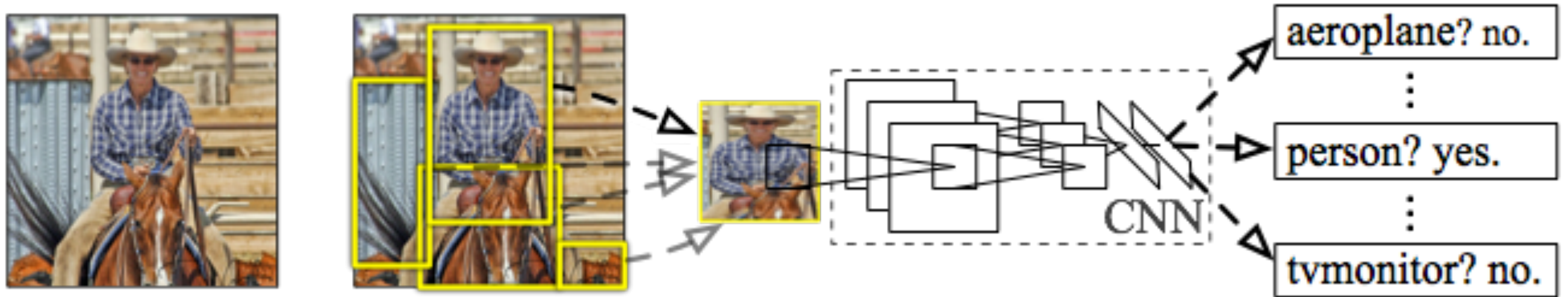


Input image      Extract region proposals (~2k / image)      **Compute CNN features**



**Dilate proposal**

# R-CNN at test time: Step 2



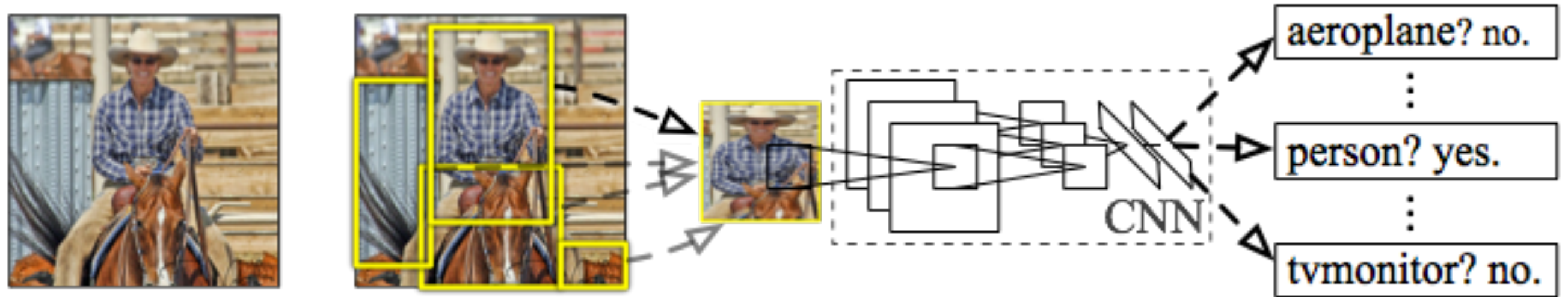
Input image      Extract region proposals (~2k / image)      Compute CNN features



a. Crop



# R-CNN at test time: Step 2



Input  
image

Extract region  
proposals (~2k / image)

Compute CNN  
features



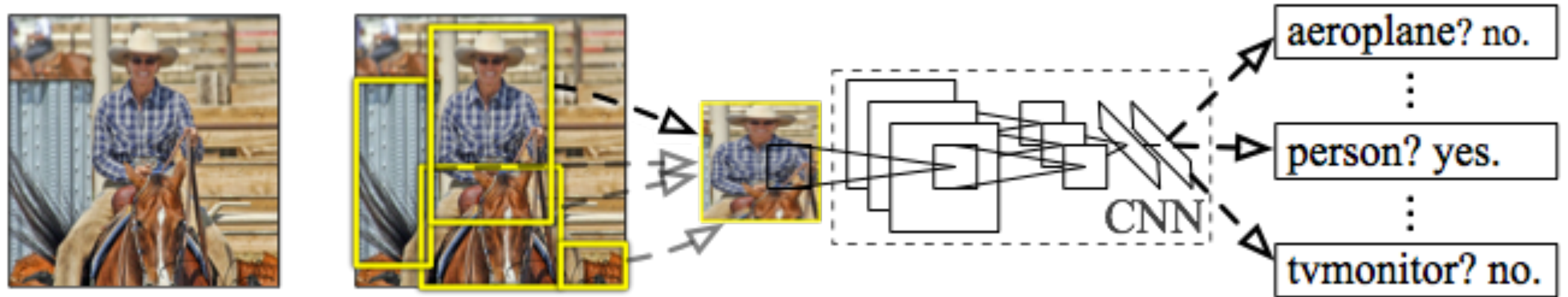
a. Crop



227 x 227

b. Scale (anisotropic)

# R-CNN at test time: Step 2



Input  
image

Extract region  
proposals (~2k / image)

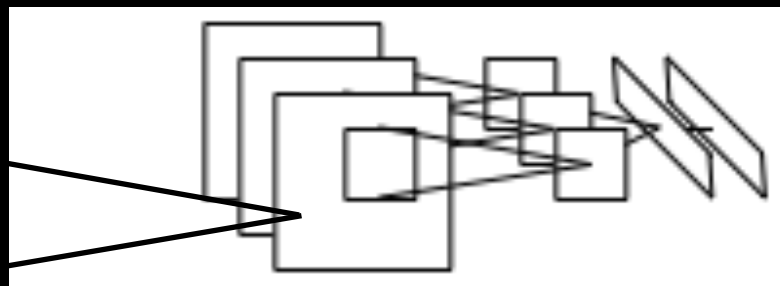
Compute CNN  
features



Crop



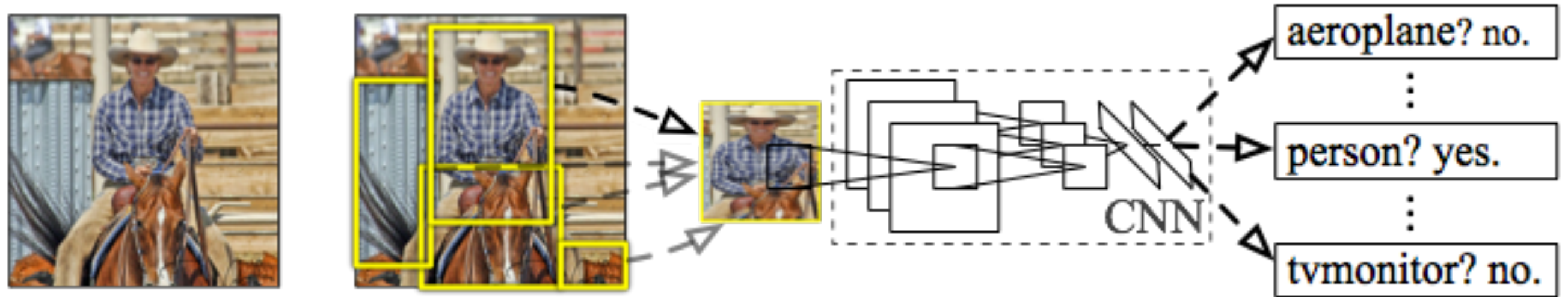
b. Scale (anisotropic)



c. Forward propagate  
Output: "fc<sub>7</sub>" features



# R-CNN at test time: Step 3

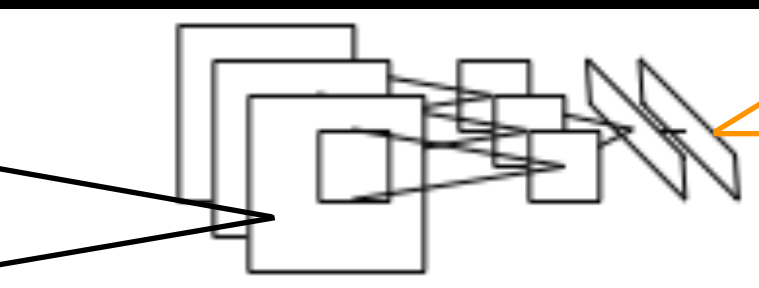


Input  
image

Extract region  
proposals (~2k / image)

Compute CNN  
features

Classify  
regions



person? 1.6

...

horse? -0.3

...

proposal 4096-dimensional  
fc7 feature vector

linear classifiers  
(SVM or softmax)

# Step 4: Object proposal refinement

---



Original  
proposal

Linear regression  
→  
on CNN features

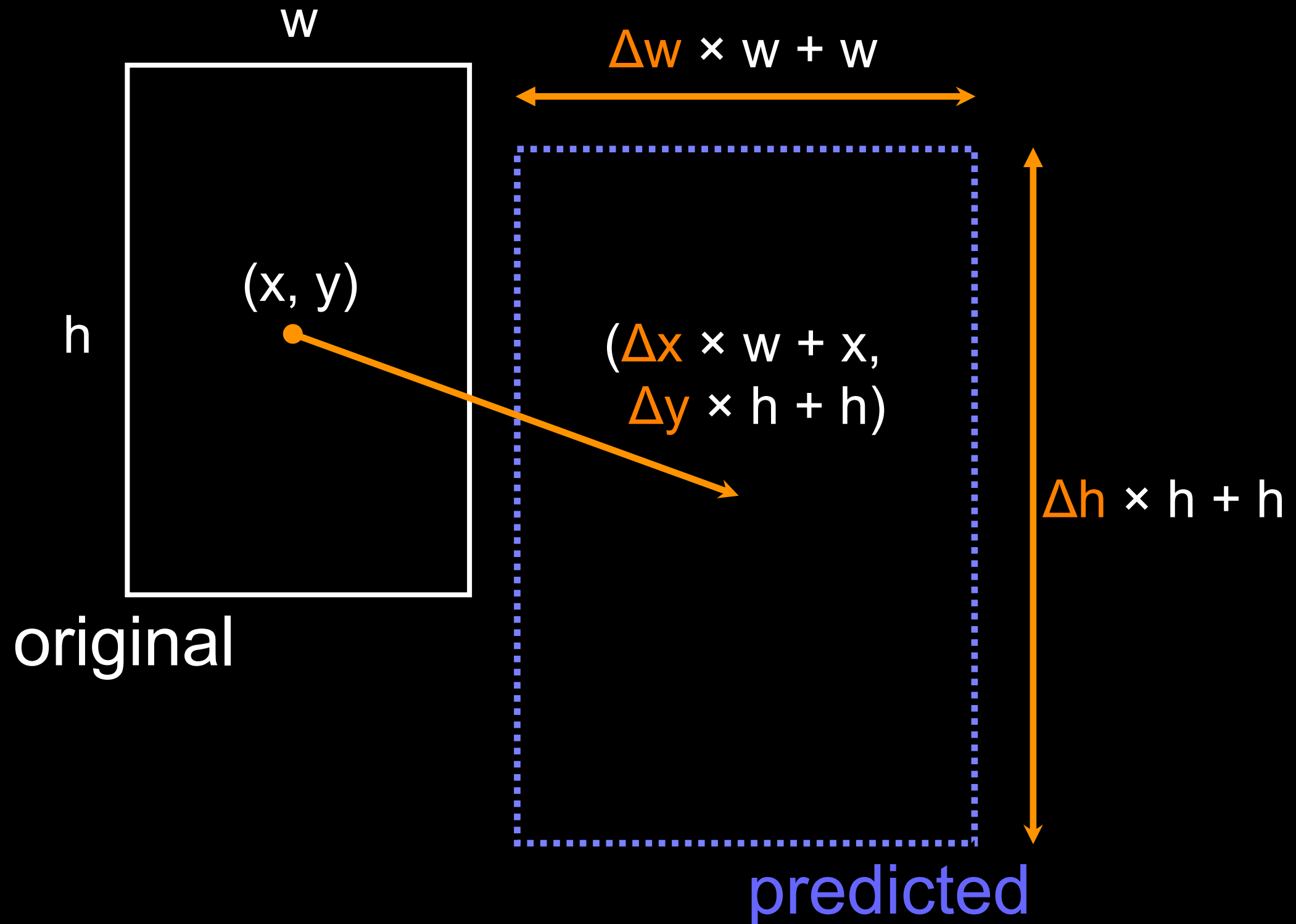


Predicted  
object bounding box

Bounding-box regression

# 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)

# Top bicycle FPs (AP = 72.8%)



bicycle [box]: ov=0.43 1-r=0.84



bicycle [box]: ov=0.35 1-r=0.81



bicycle [box]: ov=0.15 1-r=0.55



bicycle [box]: ov=0.44 1-r=0.57



bicycle [box]: ov=0.00 1-r=0.56



bicycle [box]: ov=0.35 1-r=0.52



bicycle [box]: ov=0.33 1-r=0.41



bicycle [box]: ov=0.35 1-r=0.47



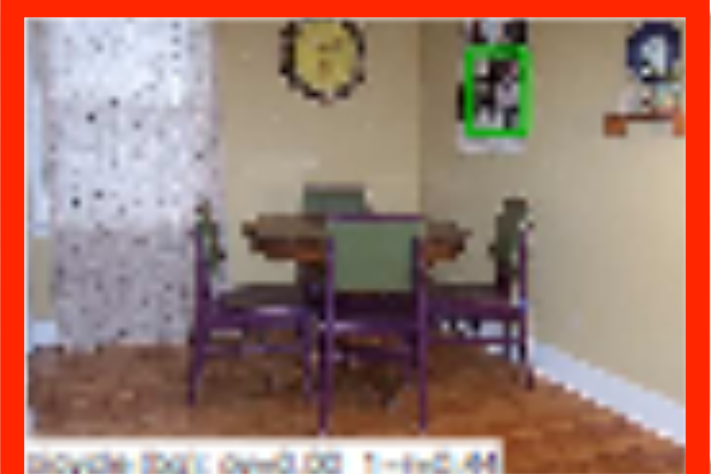
bicycle [box]: ov=0.48 1-r=0.45



bicycle [box]: ov=0.12 1-r=0.45



bicycle [box]: ov=0.42 1-r=0.45



bicycle [box]: ov=0.00 1-r=0.44

# False positive #15

---





# False positive #15



(zoom)



Unannotated bicycle

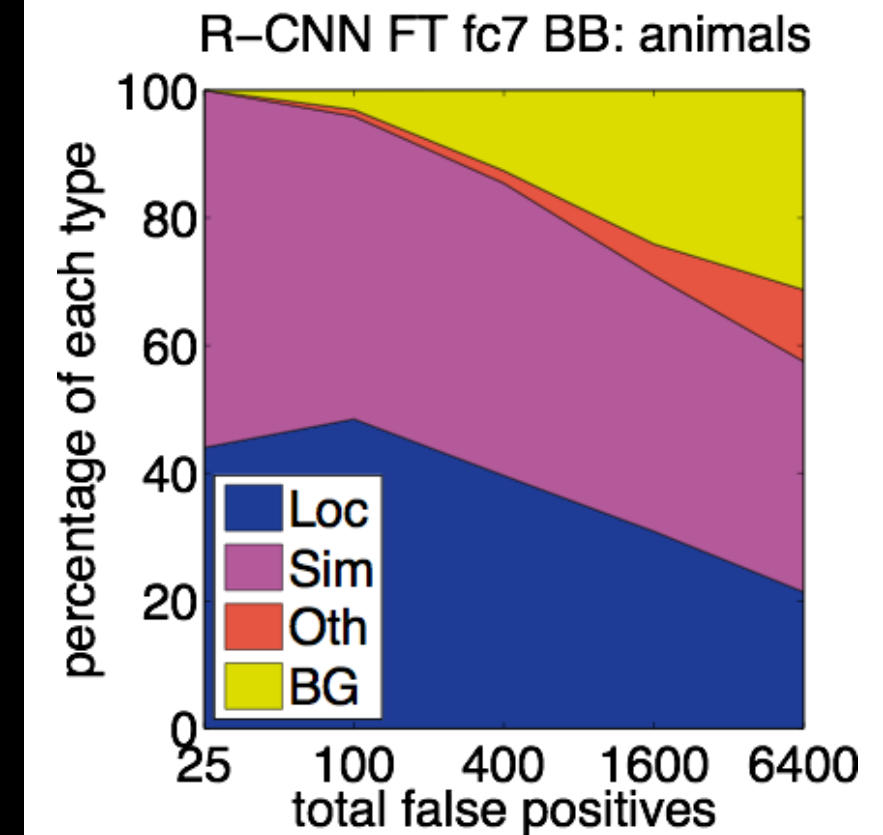
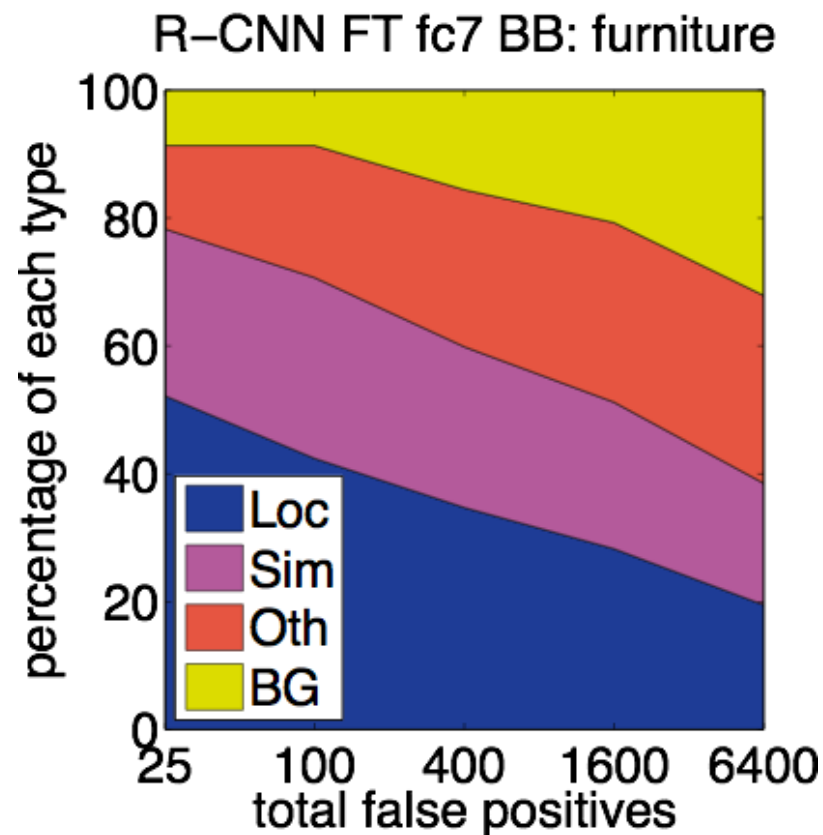
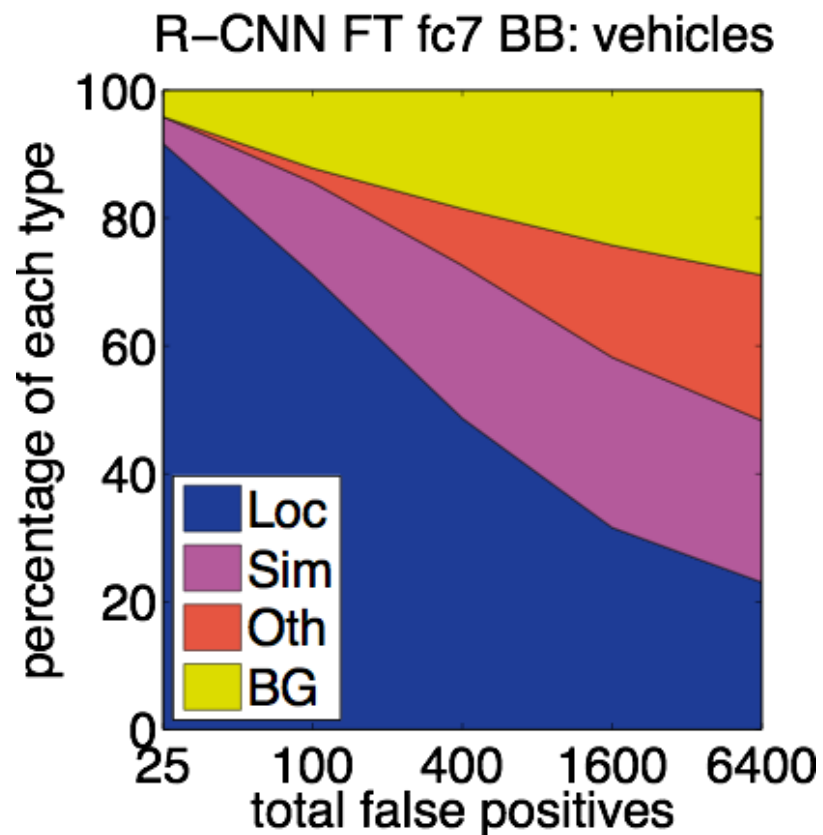
# False positive #15



1949 French comedy by Jacques Tati



# 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.

# 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

# ImageNet LSVR Challenge

---

- Image classification  
(not detection)
- 1000 classes  
(vs. 20)
- 1.2 million training labels  
(vs. 25k)



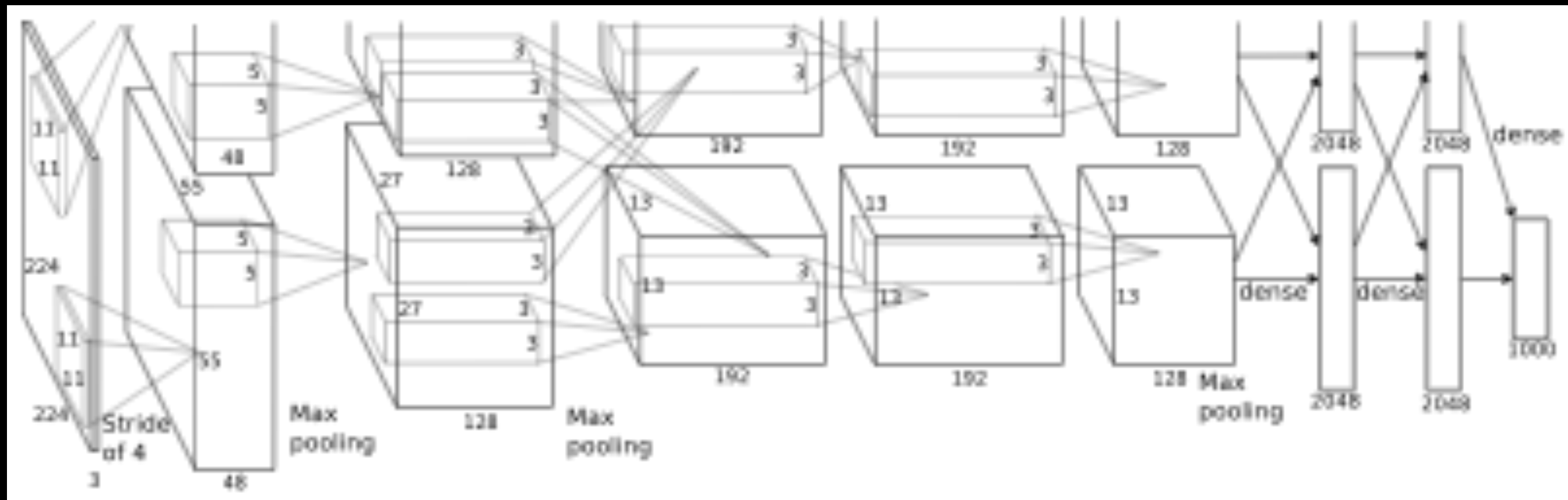
bus anywhere?

[Deng et al. CVPR' 09]



# ILSVRC 2012 winner

## “SuperVision” Convolutional Neural Network (CNN)



input ← 5 convolutional layers → fully connected

ImageNet Classification with Deep Convolutional Neural Networks.

Krizhevsky, Sutskever, Hinton. NIPS 2012.

# Impressive ImageNet results!

1000-way image classification

	Top-5 error
Fisher Vectors (ISI)	26.2%
5 SuperVision CNNs	16.4%
metric: c 7 SuperVision CNNs	15.3% s better) now: ~12%

But... does it generalize to other datasets and tasks?

[See also: DeCAF. Donahue et al., ICML 2014.]

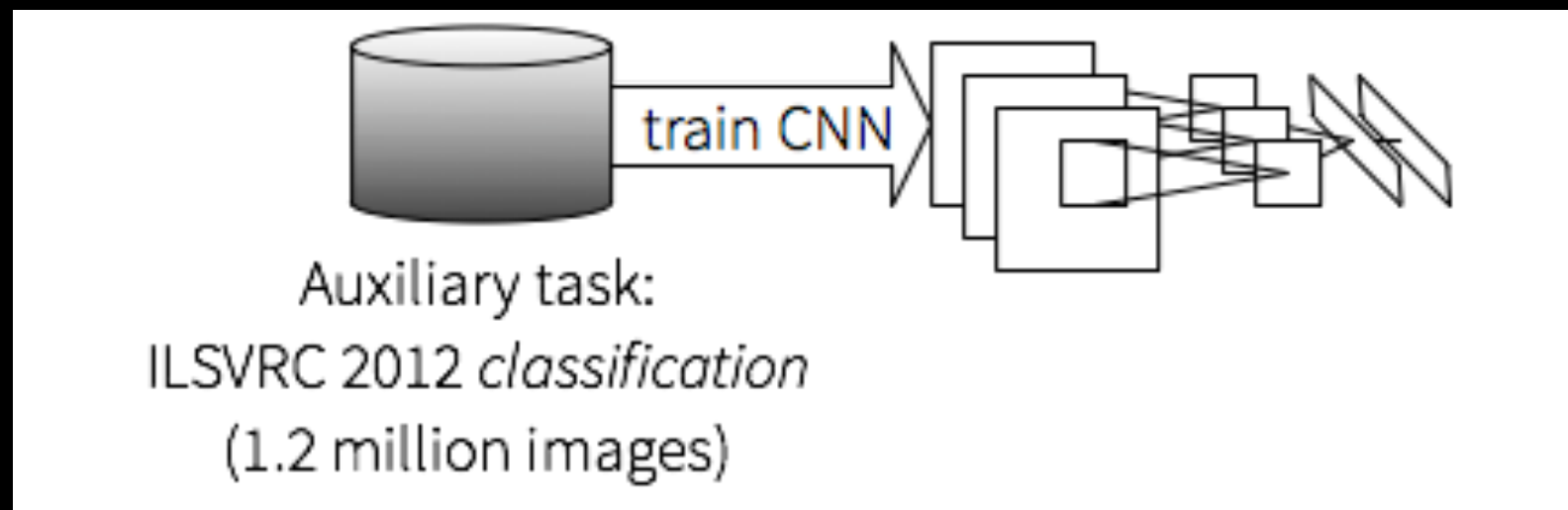
Spirited debate at ECCV 2012

# R-CNN training: Step 1

---

## Supervised pre-training

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



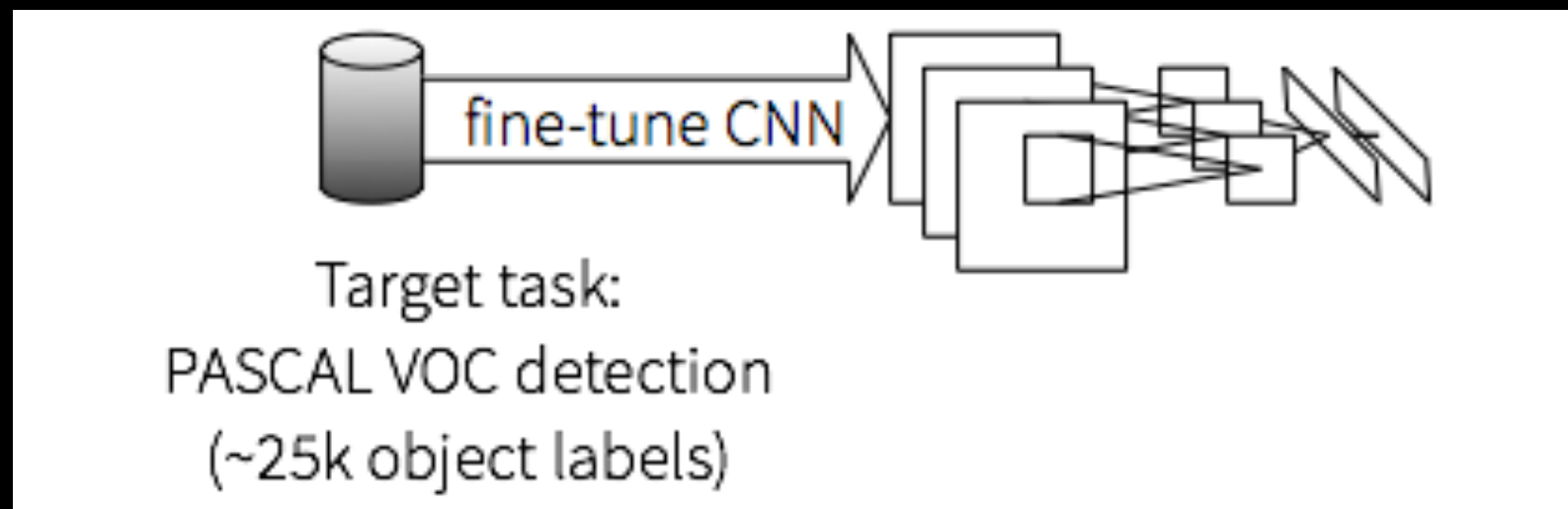
\*Network from Krizhevsky, Sutskever & Hinton. NIPS 2012  
Also called "AlexNet"

# R-CNN training: Step 2

---

## Fine-tune the CNN for detection

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

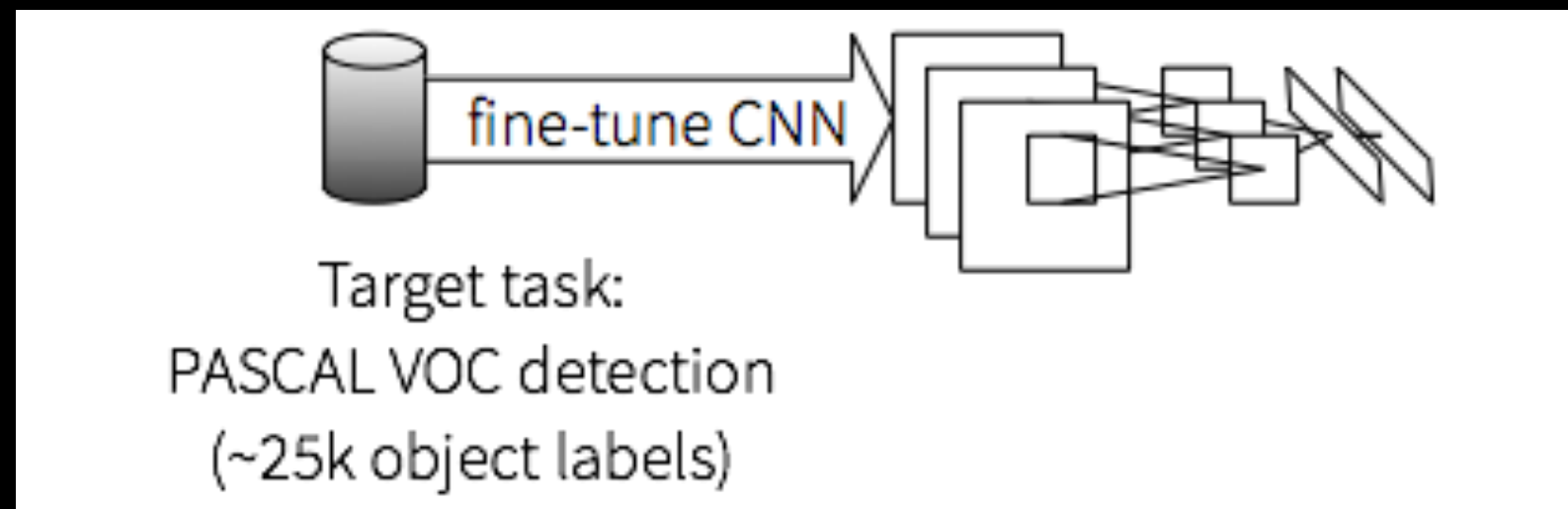


# R-CNN training: Step 2

---

## 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%)

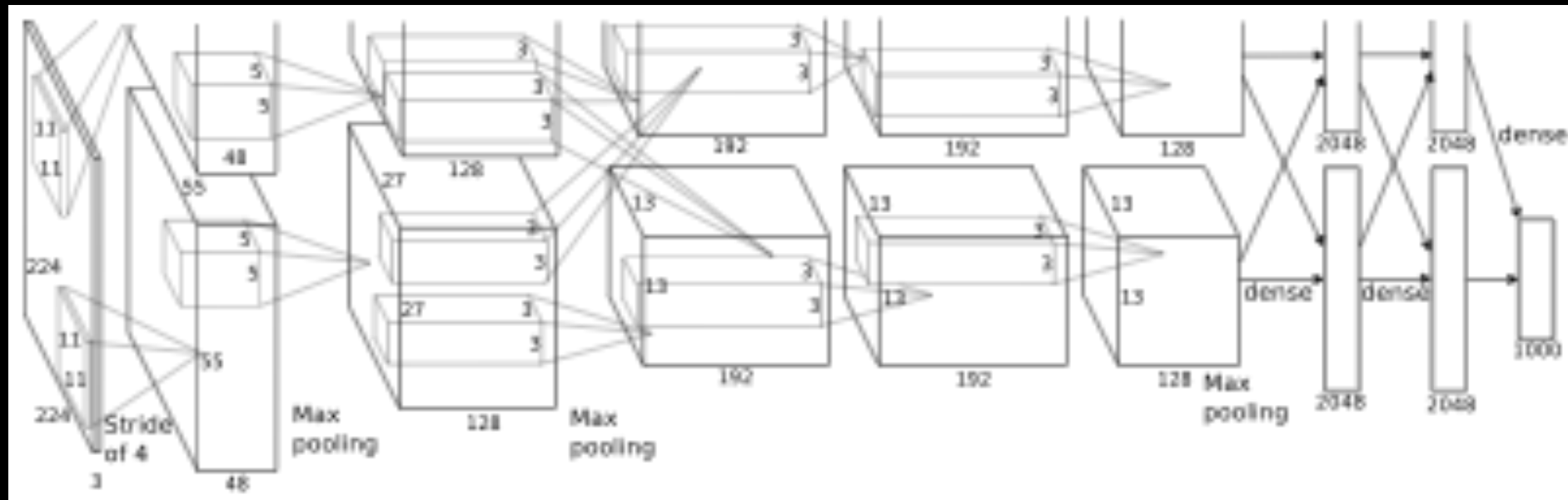


# Compare with fine-tuned R-CNN

fine-tuned		VOC 2007	VOC 2010
	Regionlets (Wang et al. 2013)	41.7%	39.7%
	SegDPM (Fidler et al. 2013)		40.4%
	R-CNN pool <sub>5</sub>	44.2%	
	R-CNN fc <sub>6</sub>	46.2%	
	R-CNN fc <sub>7</sub>	44.7%	
	R-CNN FT pool <sub>5</sub>	47.3%	
	R-CNN FT fc <sub>6</sub>	53.1%	
	R-CNN FT fc <sub>7</sub>	54.2%	50.2%

metric: mean average precision (higher is better)

# What did the network learn?

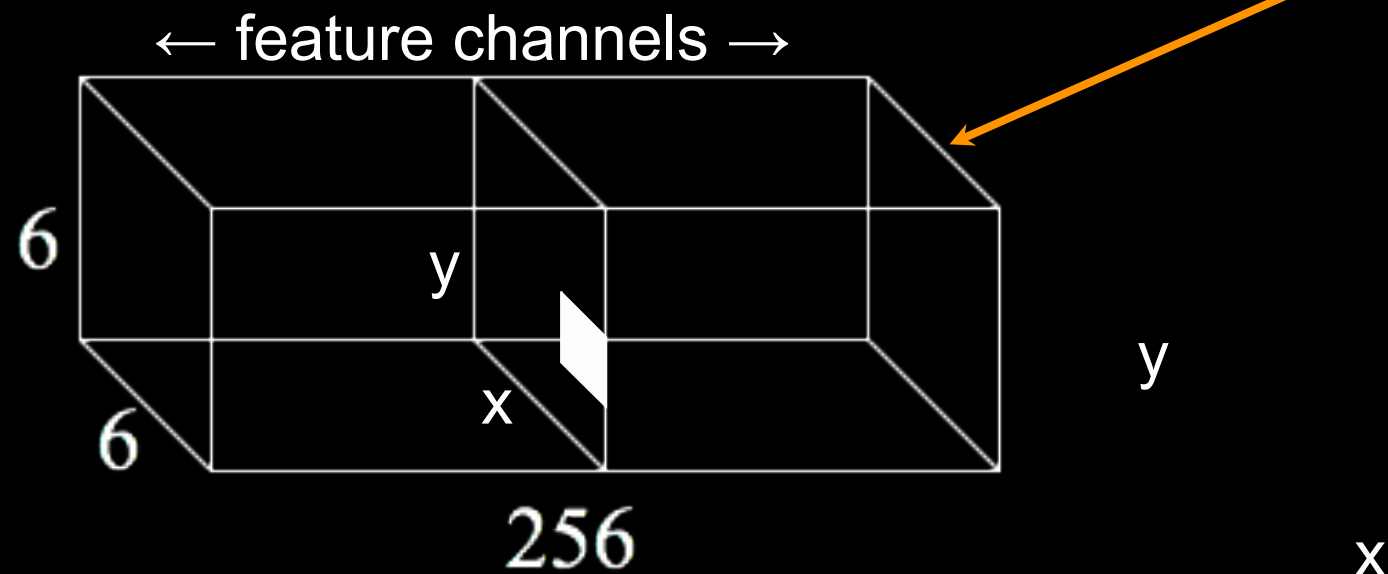
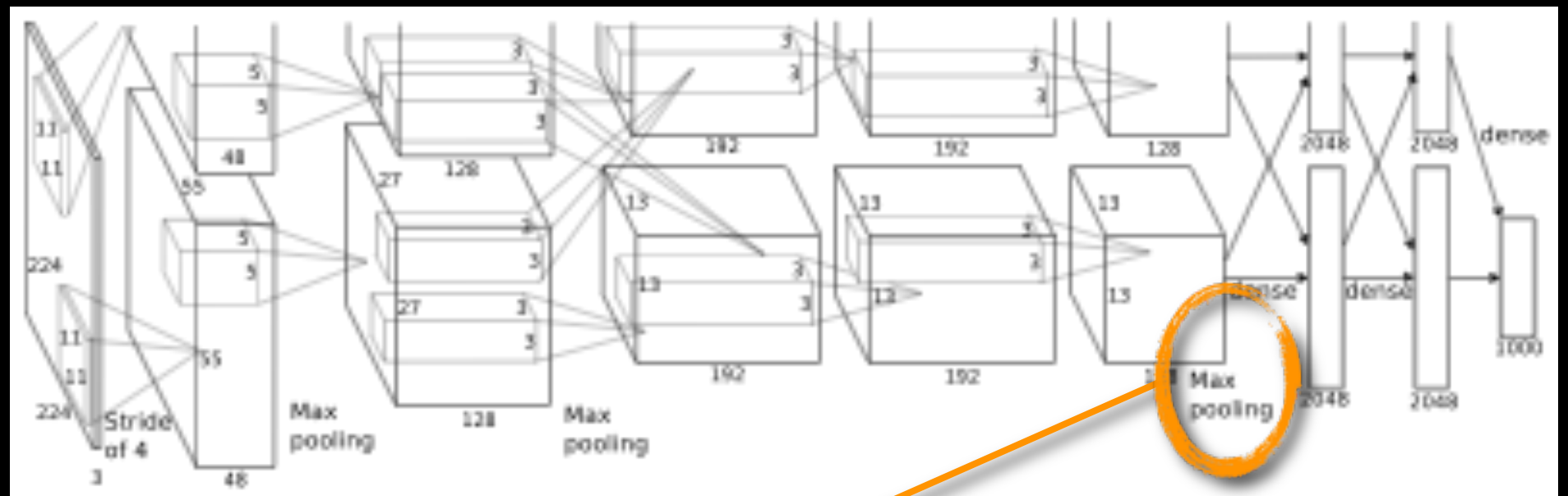




# What did the network learn?



“stimulus”

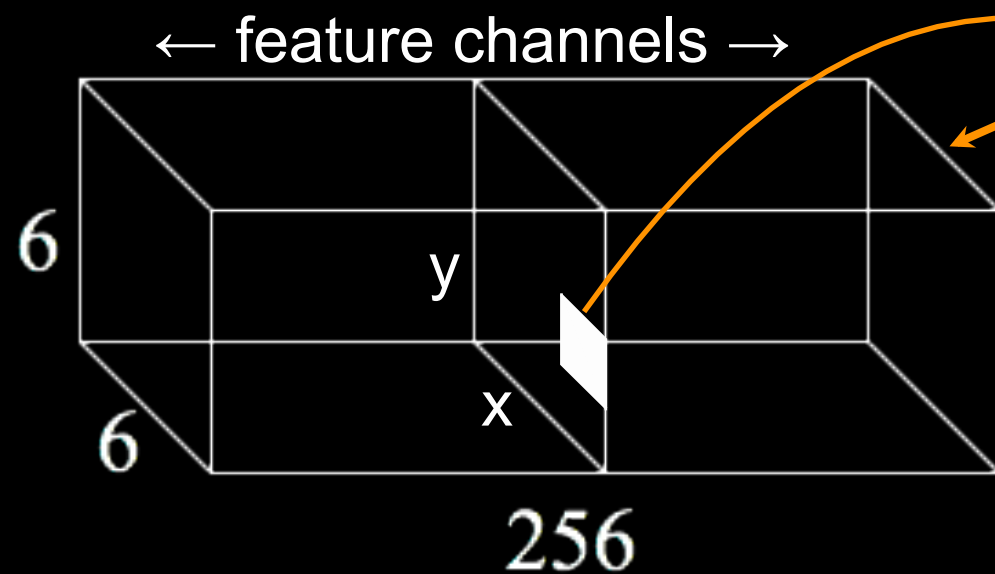
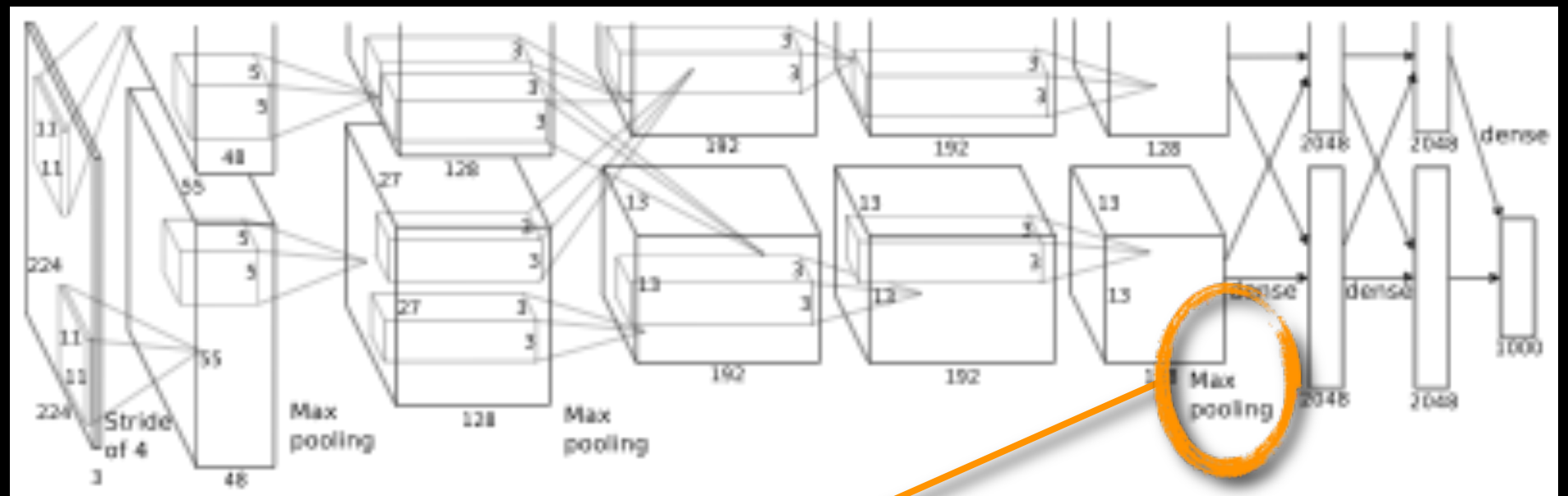


pool<sub>5</sub> feature map

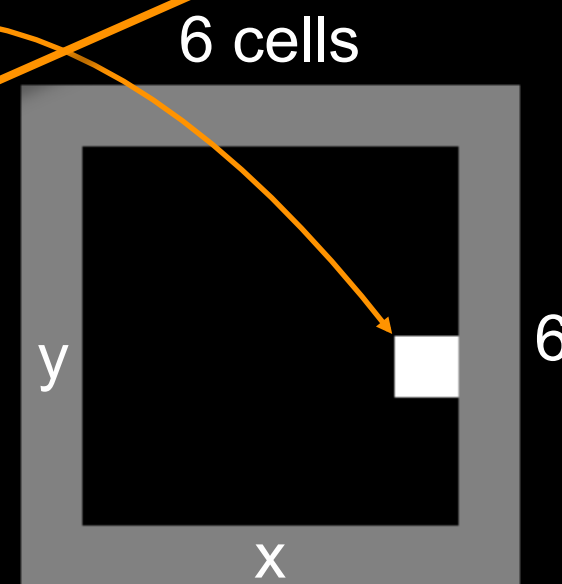
# What did the network learn?



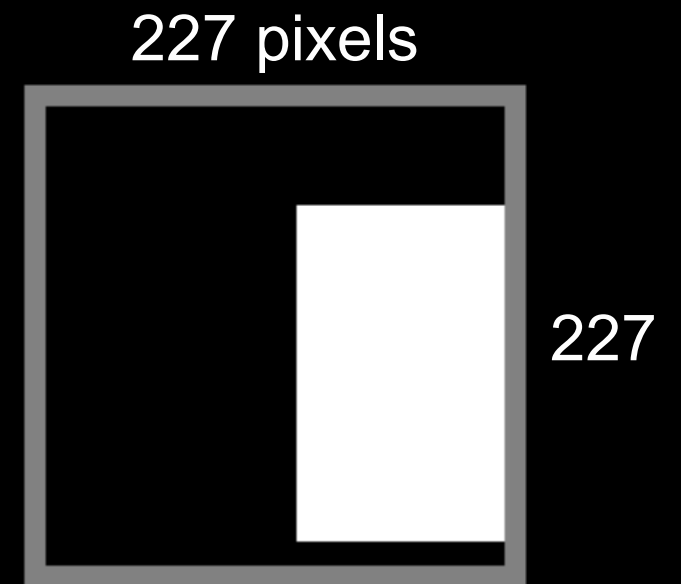
“stimulus”



pool<sub>5</sub> feature map



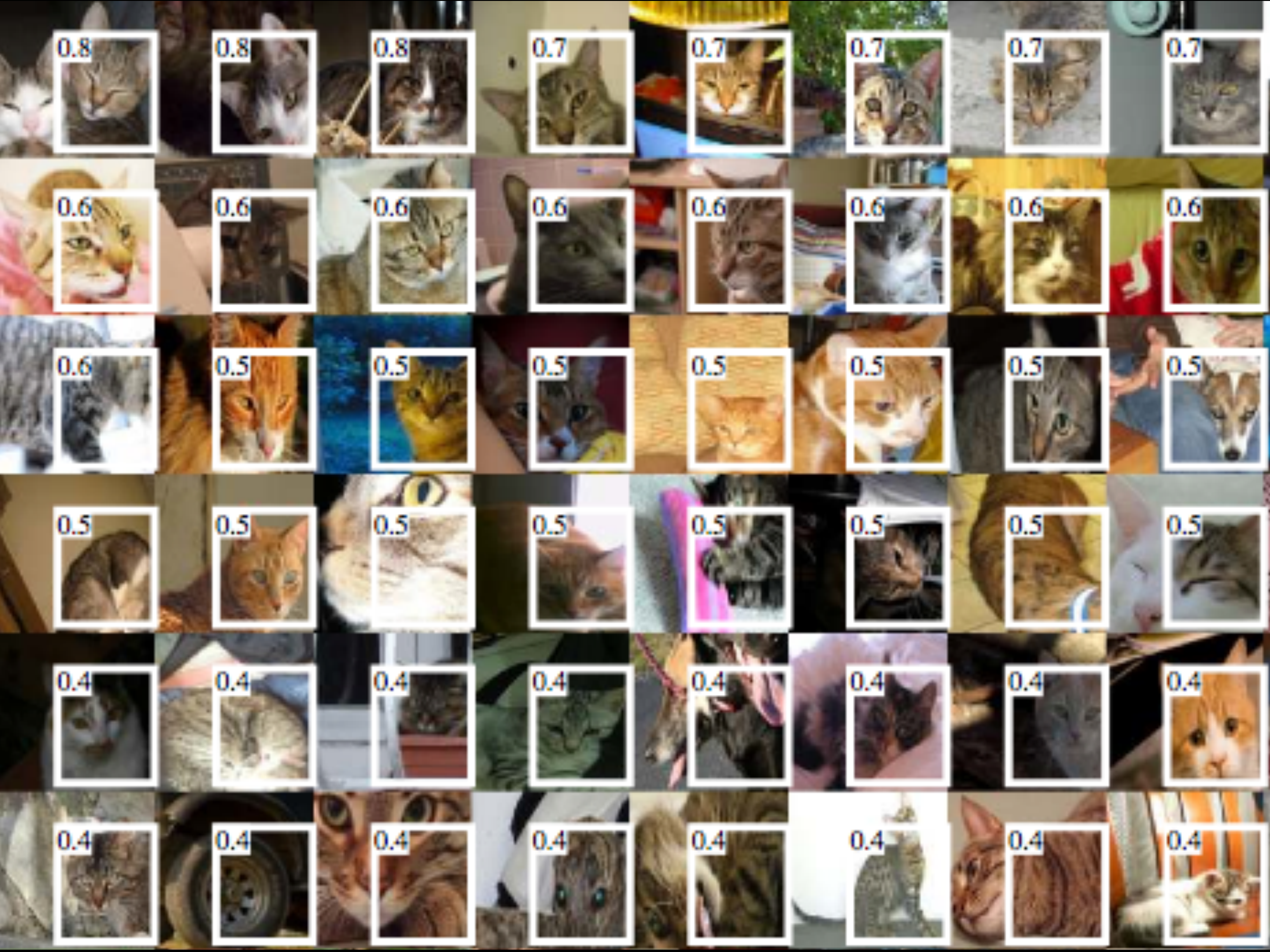
feature position



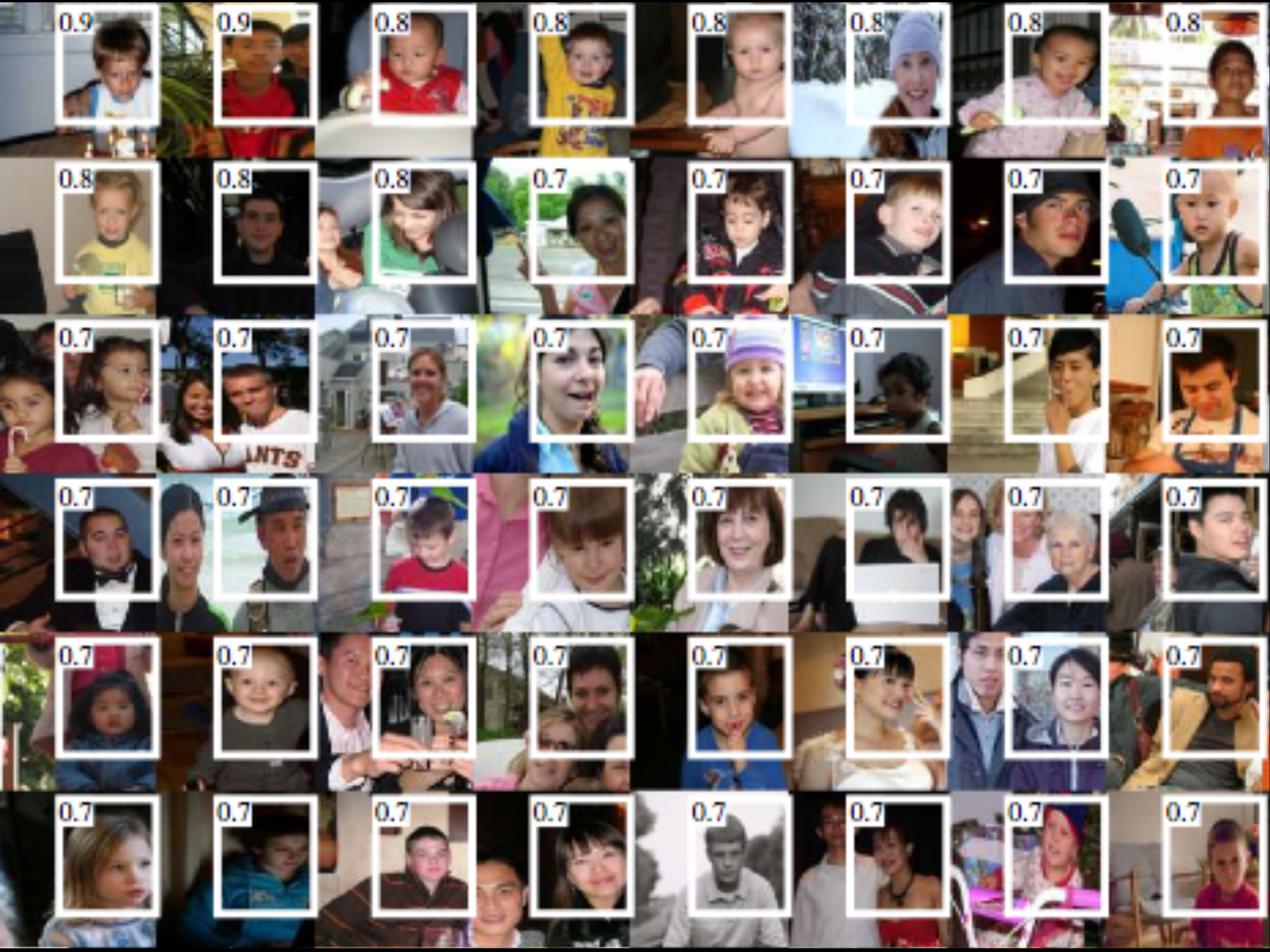
receptive field

Visualize images that activate pool<sub>5</sub> a feature

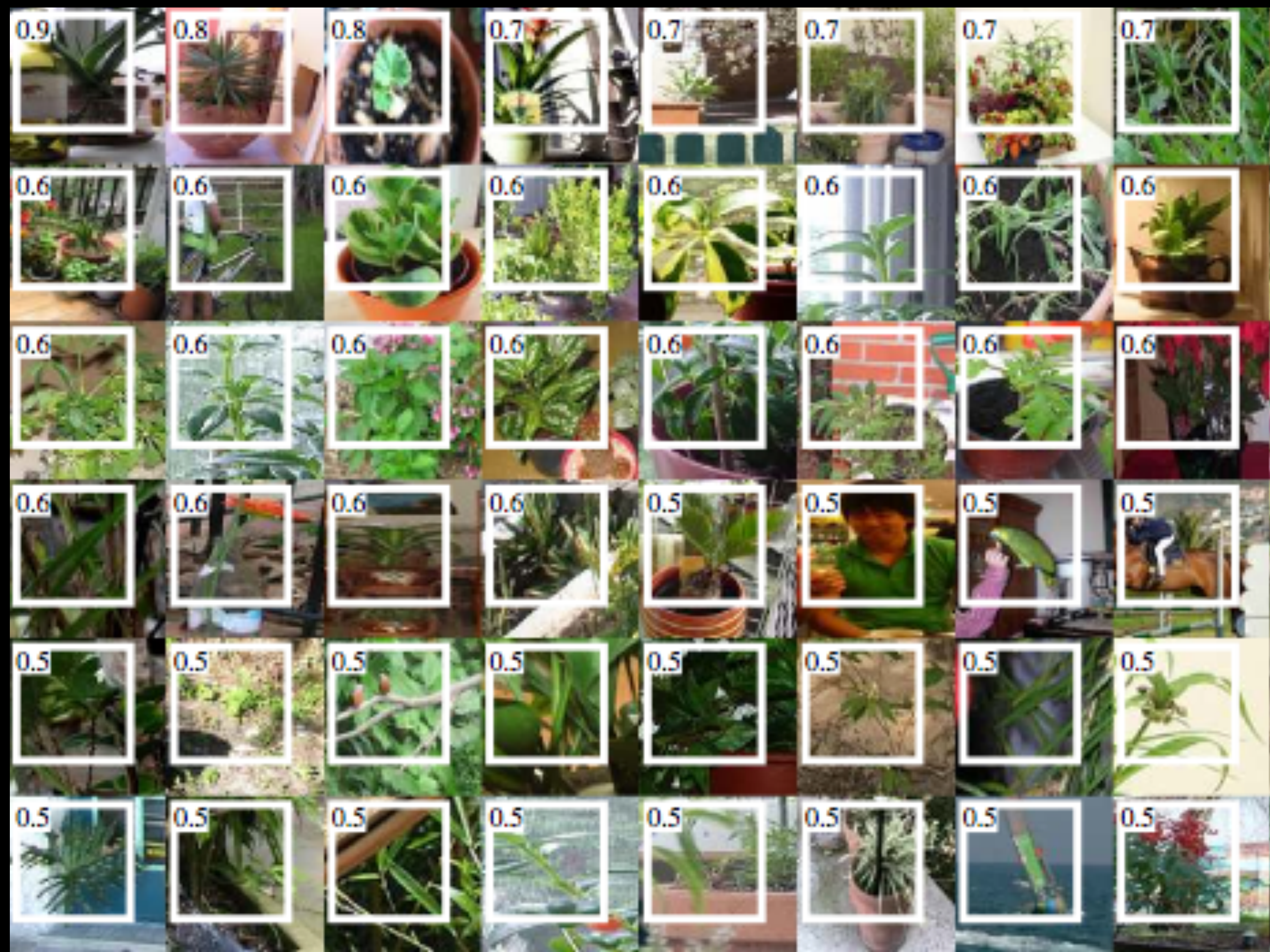




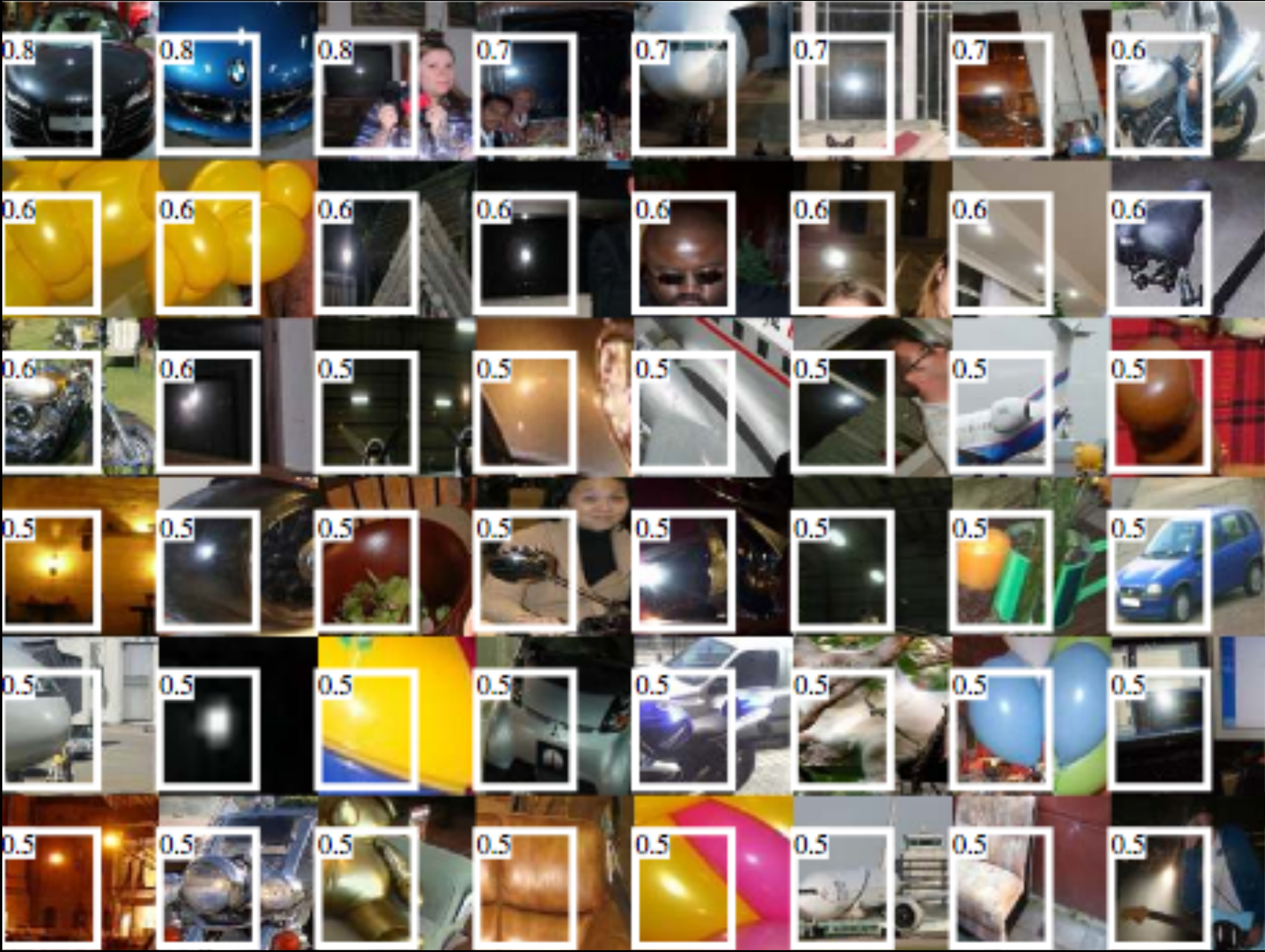




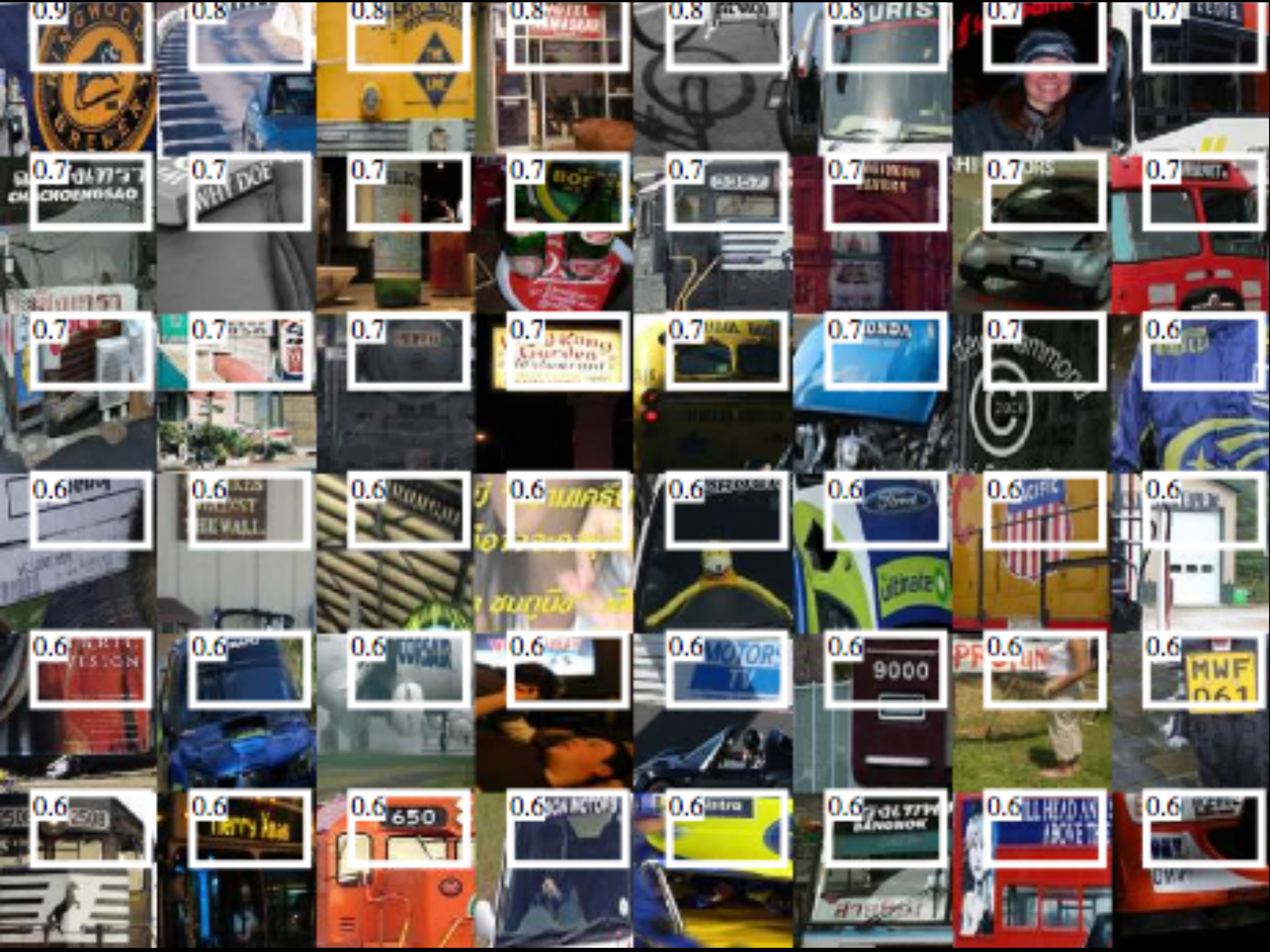














# Take away

---

- Dramatically better PASCAL mAP
- R-CNN outperforms other CNN-based methods on ImageNet detection
- Detection speed is manageable (~11s / image)
- Scales very well with number of categories (30ms for 20 → 200 classes!)
- R-CNN is simple and completely open source



# Semantic segmentation

---



CPMC segments  
(Carreira & Sminchisescu)



full



fg

# Semantic segmentation



CPMC segments  
(Carreira & Sminchisescu)



full



fg

	VOC 2011 test
Bonn second-order pooling (Carreira et al.)	<b>47.6%</b>
R-CNN fc <sub>6</sub> full+fg (no fine-tuning)	<b>47.9%</b>

Improved to 50.5% in our upcoming ECCV' 14 work:  
**Simultaneous Localization and Detection. Hariharan et al.**

# Get the code and models!

[bit.ly/rcnn-cvpr14](https://github.com/rbgirshick/rcnn)

The screenshot shows the GitHub repository page for `rbgirshick/rcnn`. The repository is public and has 42 commits, 3 branches, 1 release, and 2 contributors. The main branch is `master`. The repository description is "R-CNN: Regions with Convolutional Neural Network Features". The commit history shows a recent commit by `rbgirshick` titled "upgrade prototxt to caffe proto v1" 4 days ago. The commit message details the changes made to various files and directories, including `bbox_regression`, `bin`, `cachedir`, `data`, `datasets`, `examples`, `experiments`, `external`, `feat_cache`, `finetuning`, and `imdb`. The right sidebar contains links to Code, Issues, Pull Requests, Wiki, Pulse, Graphs, Network, and Settings. The SSH clone URL is `git@github.com:rbgirshick/rcnn`. The page also includes a "Clone in Desktop" button and a "Download ZIP" button.

PUBLIC **rbgirshick / rcnn** Unwatch 36 Unstar 89 Fork 53

R-CNN: Regions with Convolutional Neural Network Features — Edit

42 commits 3 branches 1 release 2 contributors

branch: master rcnn / +

upgrade prototxt to caffe proto v1

rbgirshick authored 4 days ago latest commit 988411631b

bbox_regression	load cached results if they exist	2 months ago
bin	add missing bin directory	3 months ago
cachedir	make cachedir setup same with a local override file	3 months ago
data	update data download READMEs and main README	4 days ago
datasets	initial checkin	3 months ago
examples	improve demo; update models to caffe's v1 proto messages	4 days ago
experiments	show how to use RCNN_CONFIG_OVERRIDE	3 months ago
external	cleanup installation instructions after a walkthrough; remove externa...	3 months ago
feat_cache	add feat_cache/README.md	3 months ago
finetuning	upgrade prototxt to caffe proto v1	4 days ago
imdb	load cached results if they exist	2 months ago

SSH clone URL

git@github.com:rbgirshick/rcnn

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

# Supplementary slides follow

---

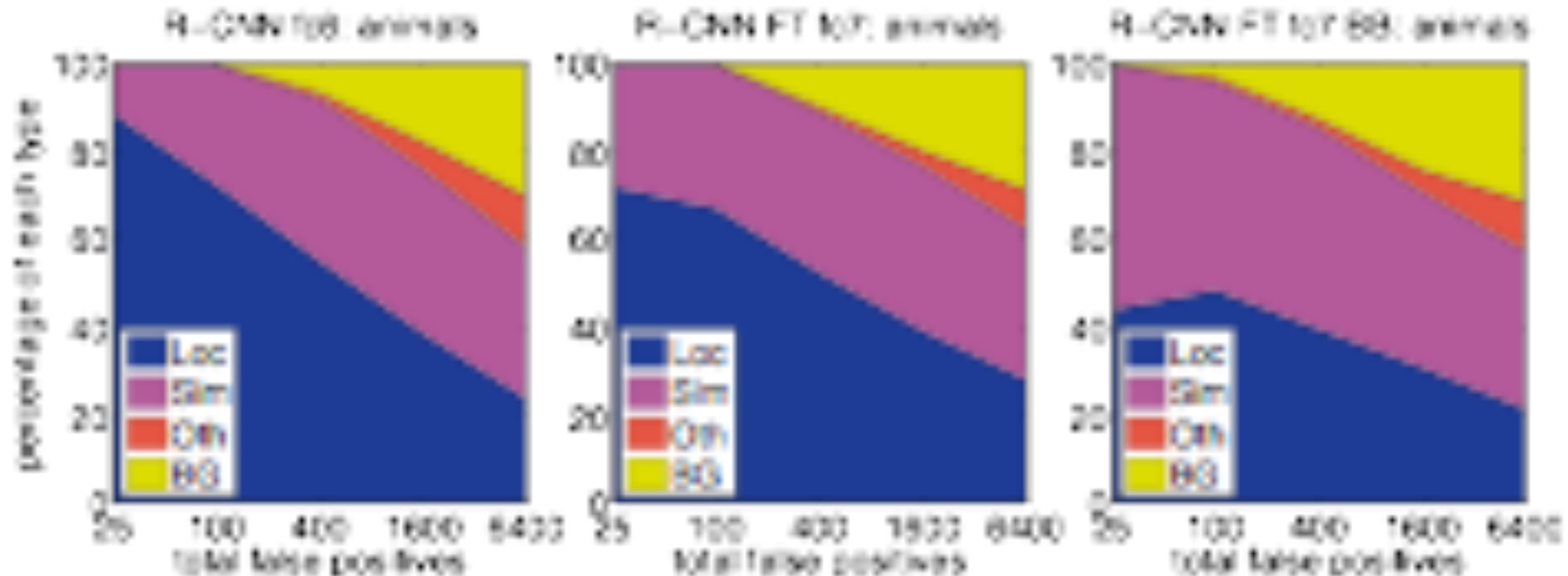
# Pre-trained CNN + SVMs (no FT)

	VOC 2007	VOC 2010
Regionlets (Wang et al. 2013)	41.7%	39.7%
SegDPM (Fidler et al. 2013)		40.4%
R-CNN pool <sub>5</sub>	44.2%	
R-CNN fc <sub>6</sub>	46.2%	
R-CNN fc <sub>7</sub>	44.7%	

metric: mean average precision (higher is better)



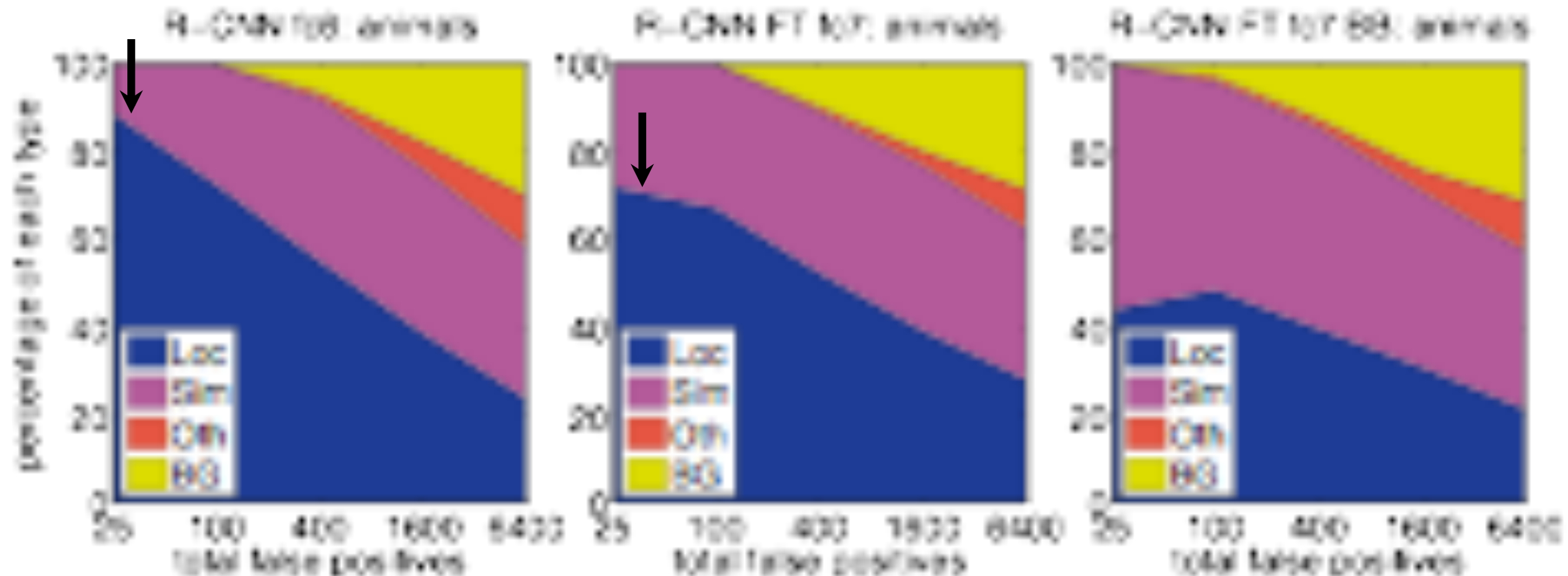
# False positive analysis



↑  
No fine-tuning

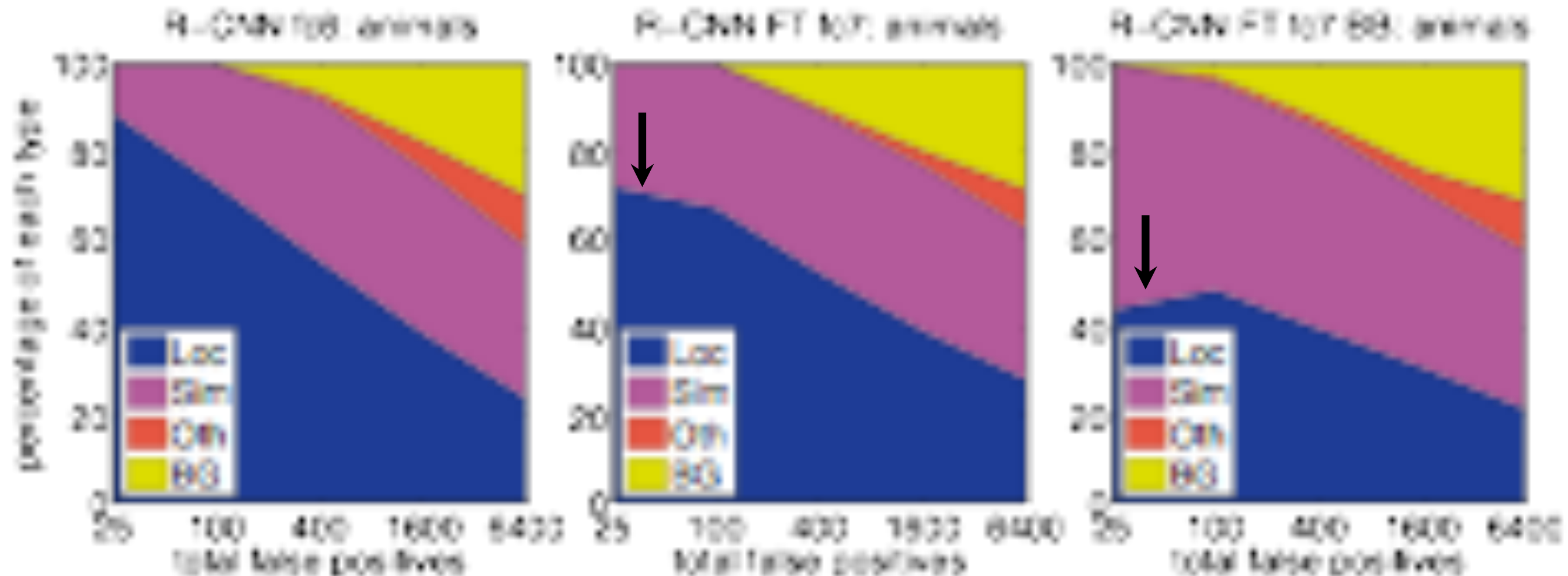
**Analysis software:** D. Hoiem, Y. Chodpathumwan, and Q. Dai. "Diagnosing Error in Object Detectors." ECCV, 2012.

# False positive analysis



After fine-tuning

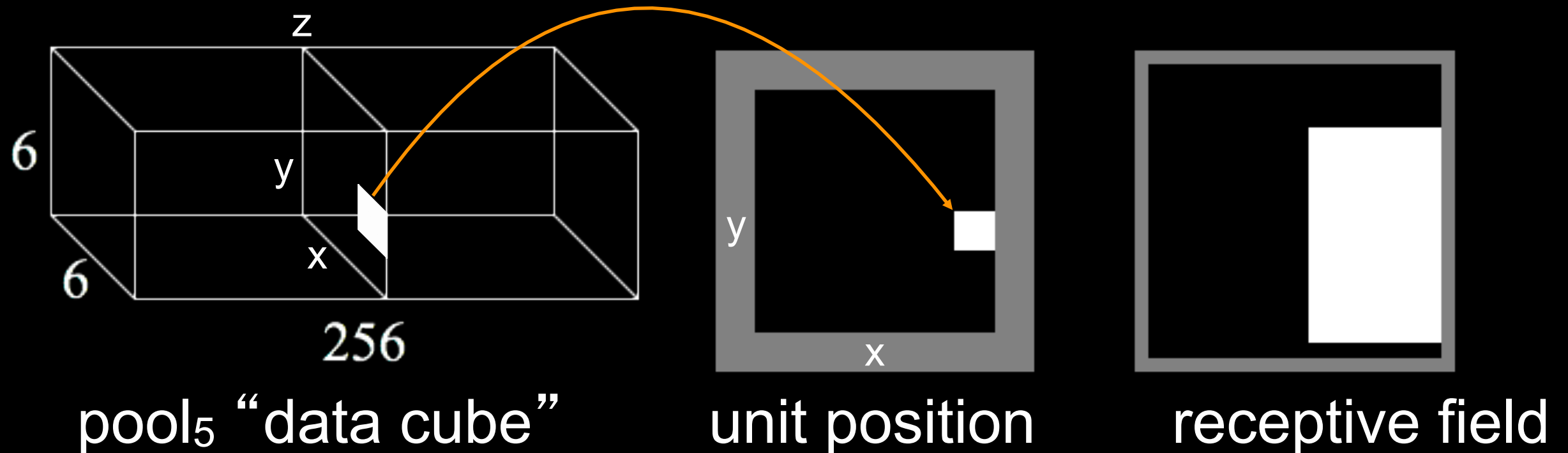
# False positive analysis



↑  
After bounding-  
box regression

# What did the network learn?

---



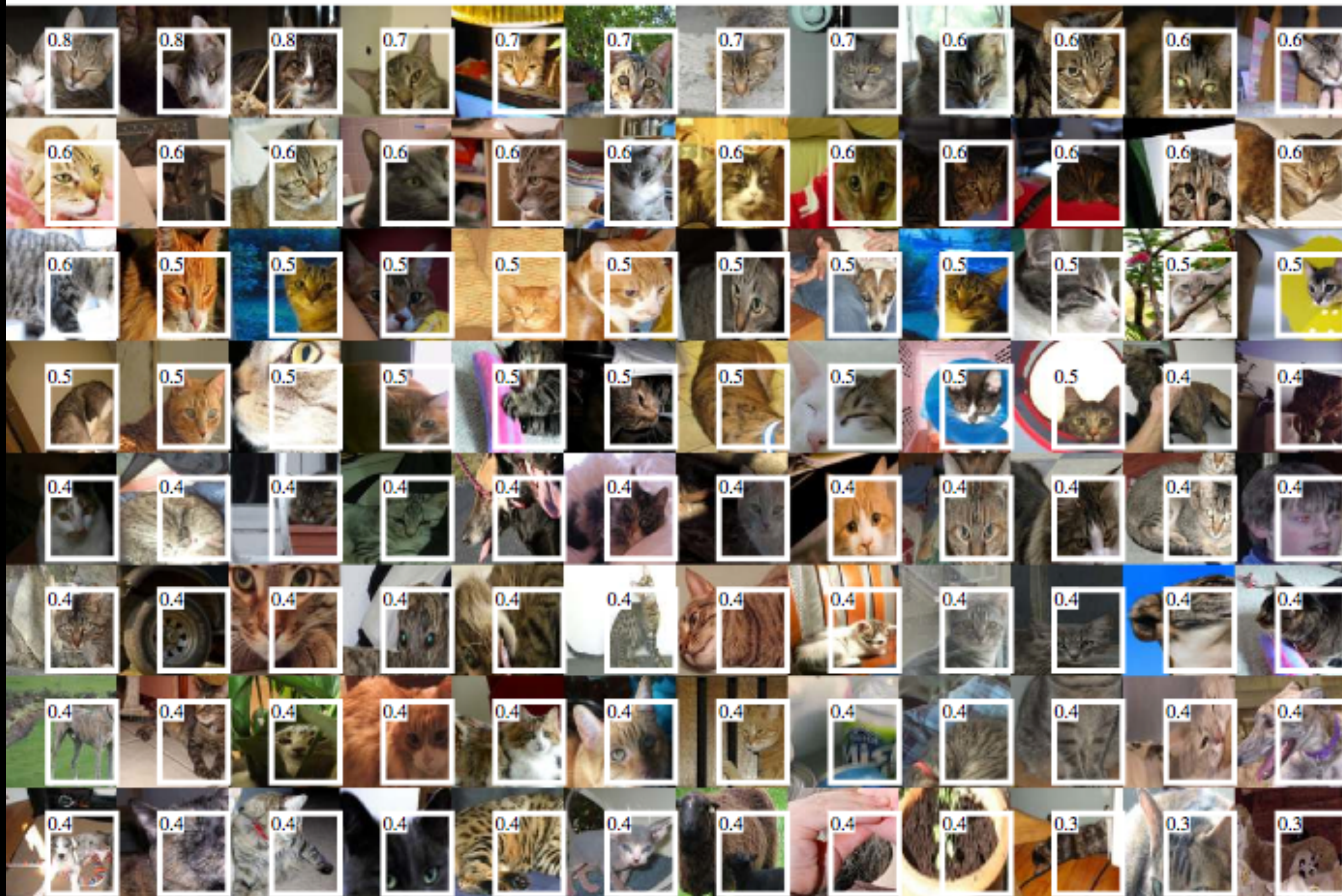
Visualize `pool5` units







pool5 feature: (4,5,110) (top 1 – 96)

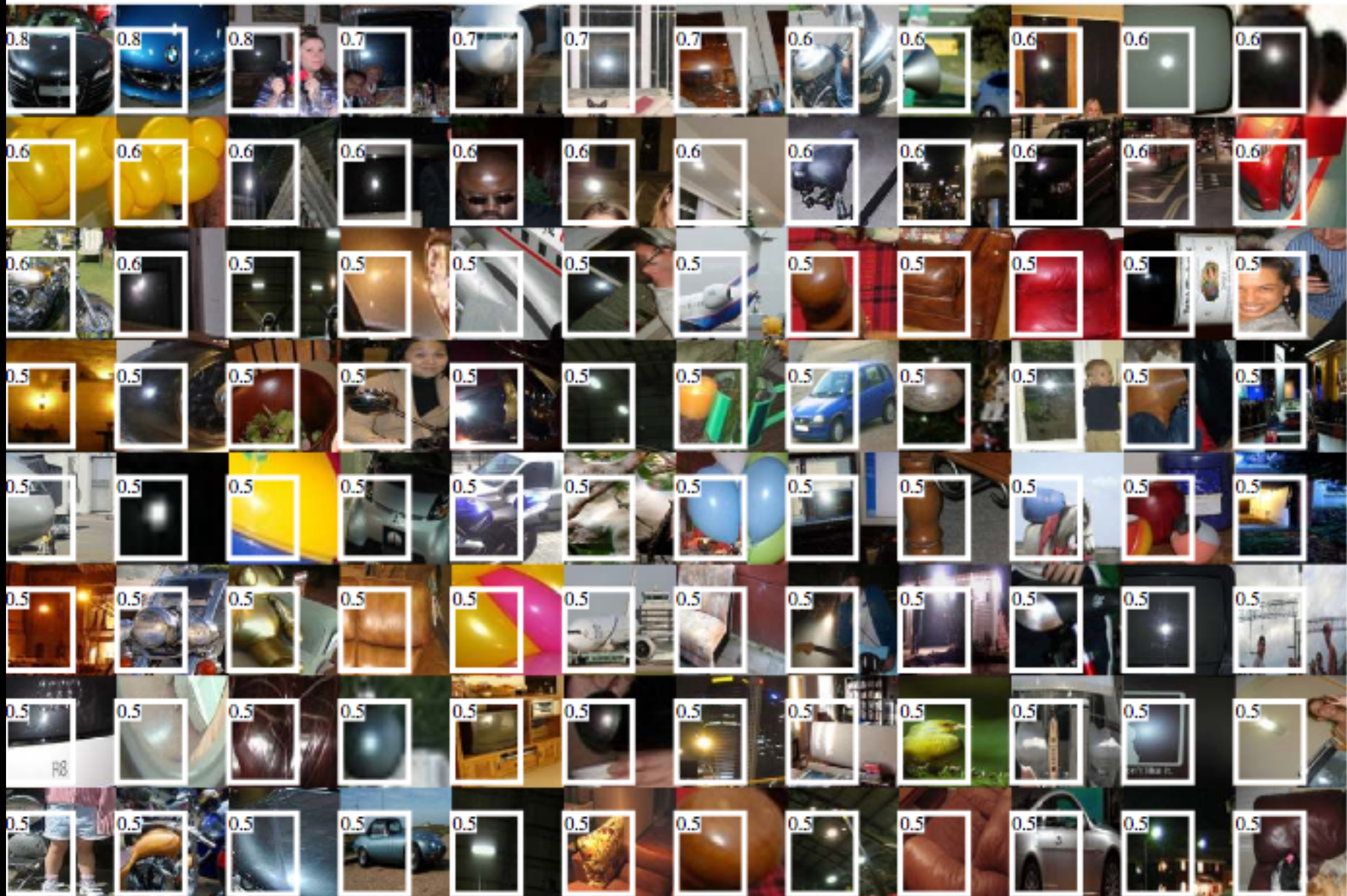








pool5 feature: (4,2,26) (top 1 – 96)





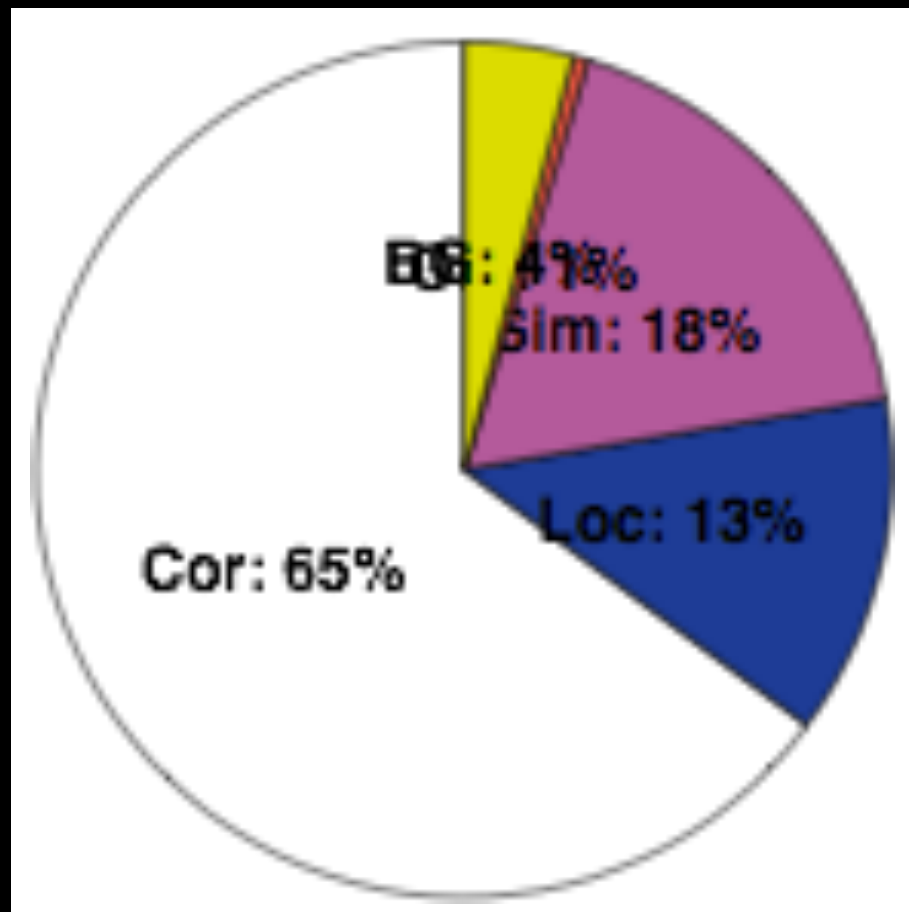




# Comparison with DPM

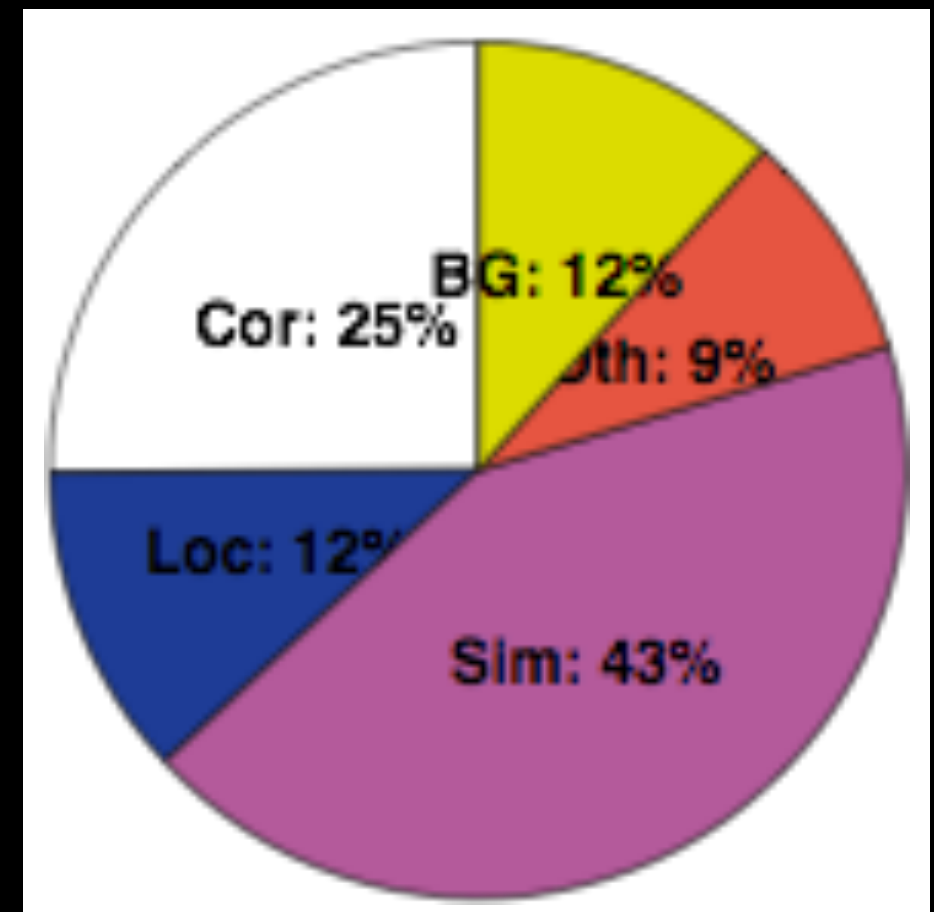
---

R-CNN



animals

DPM v5



animals



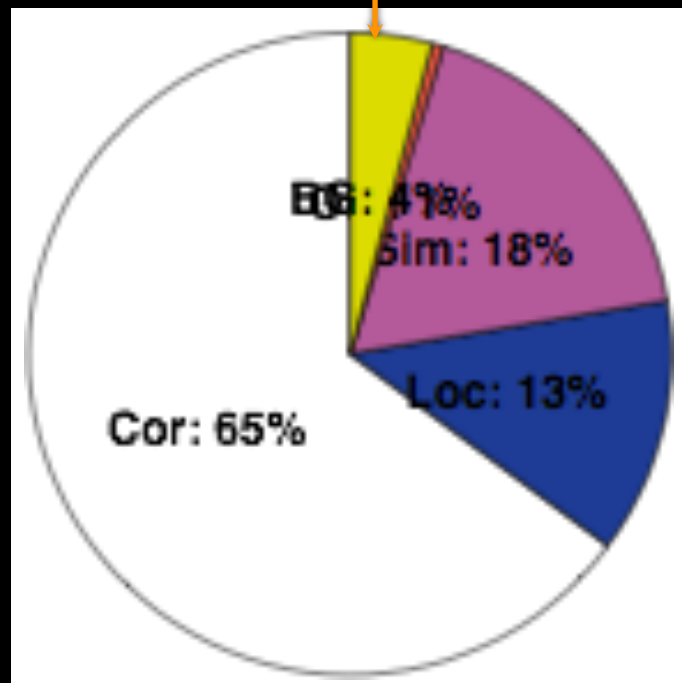
# Localization errors dominate

background

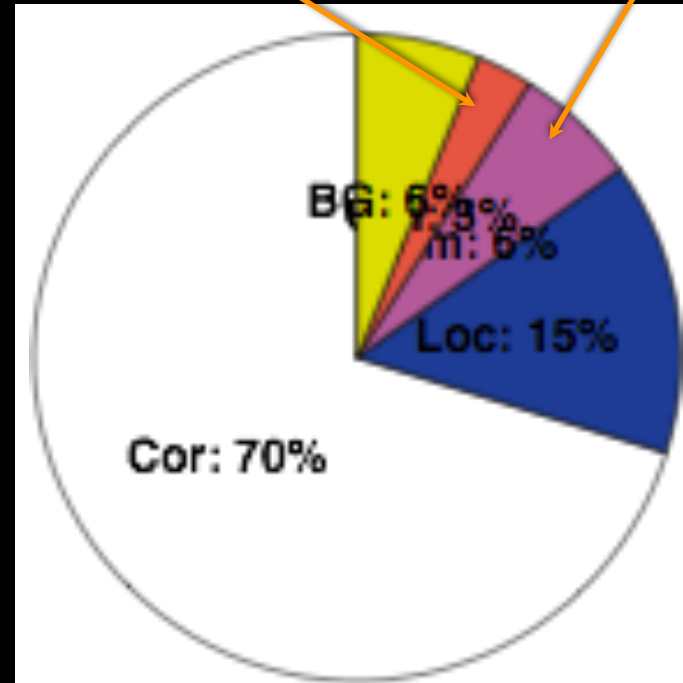
dissimilar  
classes

similar  
classes

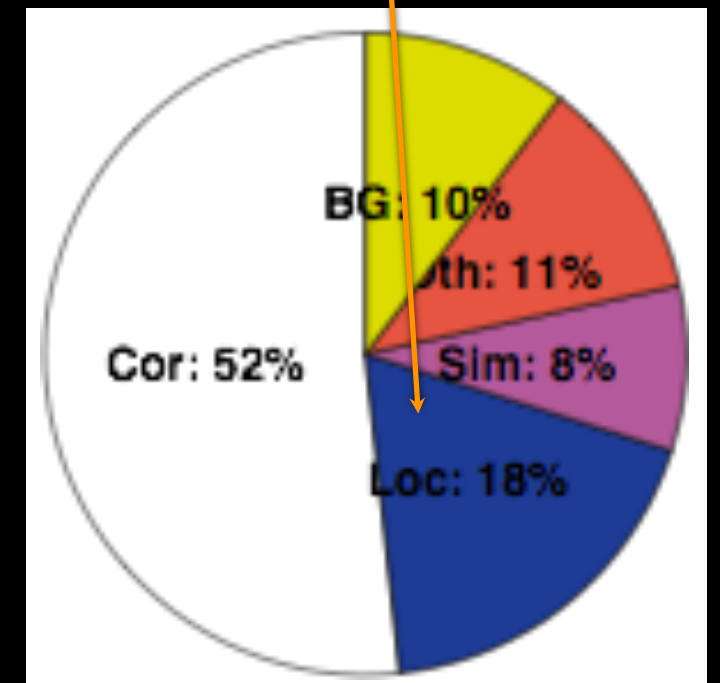
localization



animals



vehicles



furniture

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