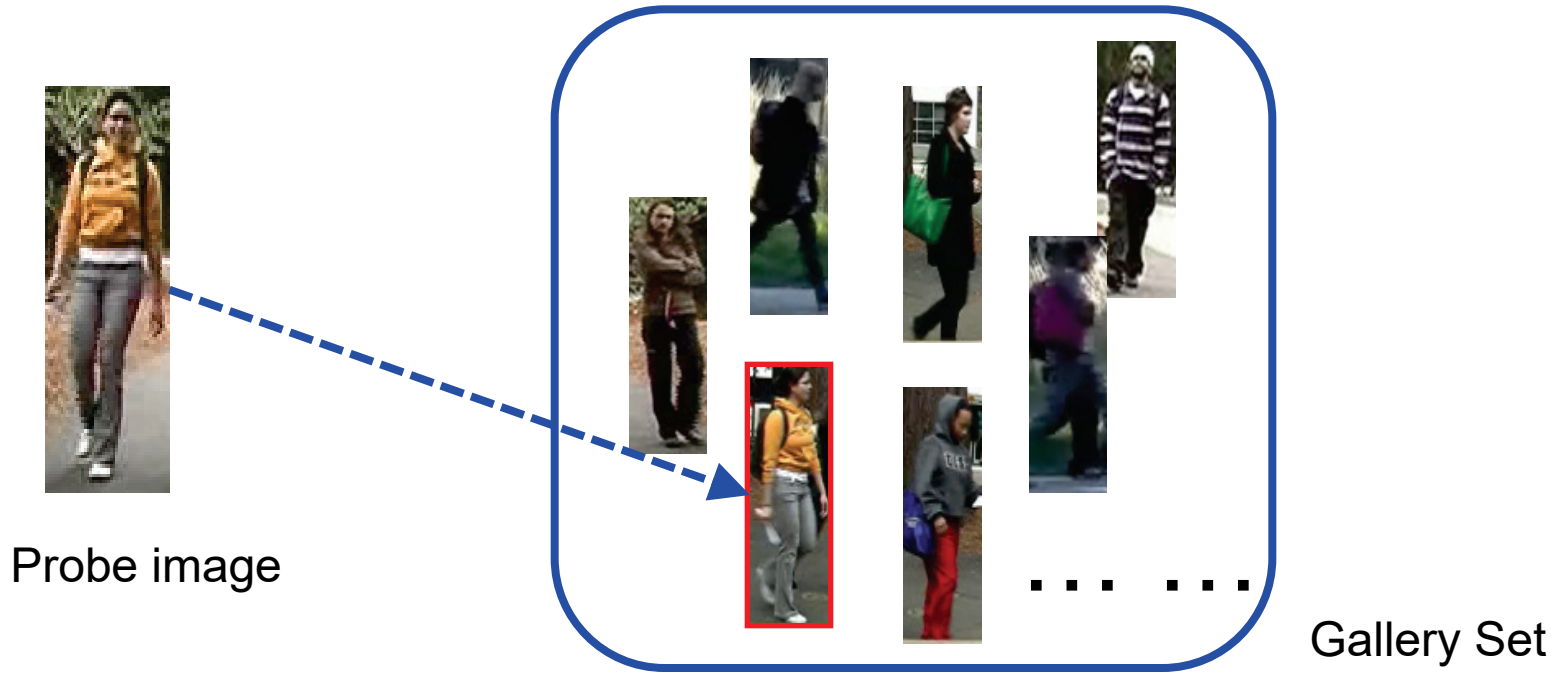
Attributes, Domain Transfer Learning & R-CNN

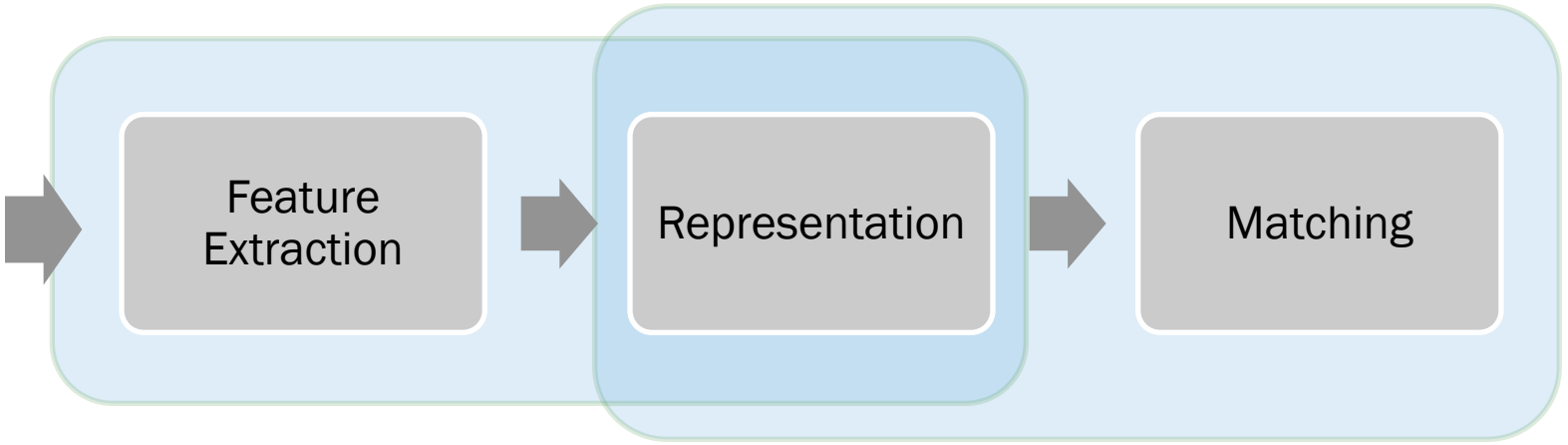
Person Detection/Localisation, Description & (Re-)Identification

- Visual recognition of an individual at different locations, different times over distributed camera views



Person Recognition (Object Recognition)

In many cases cannot be clearly separated



Low-level features:
Colour histograms, SIFT, LBP, Gabor, Early deep features of CNN

Robust features – more domain transferable:
Attribute, Shape symmetry, Colour coding, Patches & alignment, Soft-metrics (e.g. height factoring), late deep features of CNN

Matching functions:
Distance metric learning, Saliency-weighting, Sequence match/multi-shot

Challenges

- *Not all features are equal:* What features are more important under what circumstances?

- *On-the-fly selection:* How to selectively distribute some weights to informative feature given different appearance attributes?

- *Features are inherently noisy and ambiguous:* How to filter noisy and possibly redundant features?

- What are more robust features? (transferable across domains)

What are Attributes?

- Mid-level descriptions of classes or instances
  - Encoding domain knowledge:
    - "Red", "Stripy", "Dark", "Tall", "Adult".
  - Semantic attribute space:
    - Project images onto "humanly nameable" concept basis as projection axes:
      - "people wearing red", "tall people", "old people", "male", "people in suits"
    - Not statistical orthogonal basis (e.g. PCA)
    - Semantically meaningful to human interpretation (not necessarily orthogonal, e.g. Tensor space)

What are Attributes?

1 white shirt, dark trousers

2 bright and colorful shirt

3 dark jacket, jeans

4 with backpack, side pose

5 dark jacket, light color trousers

6 dark shirt with texture, back pose

7 dark shirt, side pose

8 dark shirt, dark trousers

9 colorful shirt, jeans

10 colorful shirt, dark trousers

Why Attributes?

- More robust & data independent (good for sparse data)
- Enable more “transfer learning” if learned from large “auxiliary data” pool (independent) – domain independent
- Readily interpretable for human interaction (e.g. query and search)

Attribute Ontology (hand-crafted)

- Bottom-up generative models:
  - + Clustering (e.g. random forests)
  - + Feature mapping (e.g. deep learning)
    - × Not very discriminative
- Top-down discriminative models:
  - + Ask an expert / ontology
  - + Use intuition
    - × … cannot easily guarantee classifiable or detectable
  - + Train a classifier!
  - × Reliant on *individual* SVM performance

e.g. human defined 15 attributes

Challenge: Attribute Consistency & Scalability

- Learning discriminative models:
  - + Ask an expert / ontology
  - + Use intuition
    - × … cannot easily guarantee classifiable or detectable, nor consistent
  - + Train multiple classifiers
    - × … lots of labelling… not just class labels, but all the parts (fine-grained)
    - × The curse of labelling
  - + Deep learning?

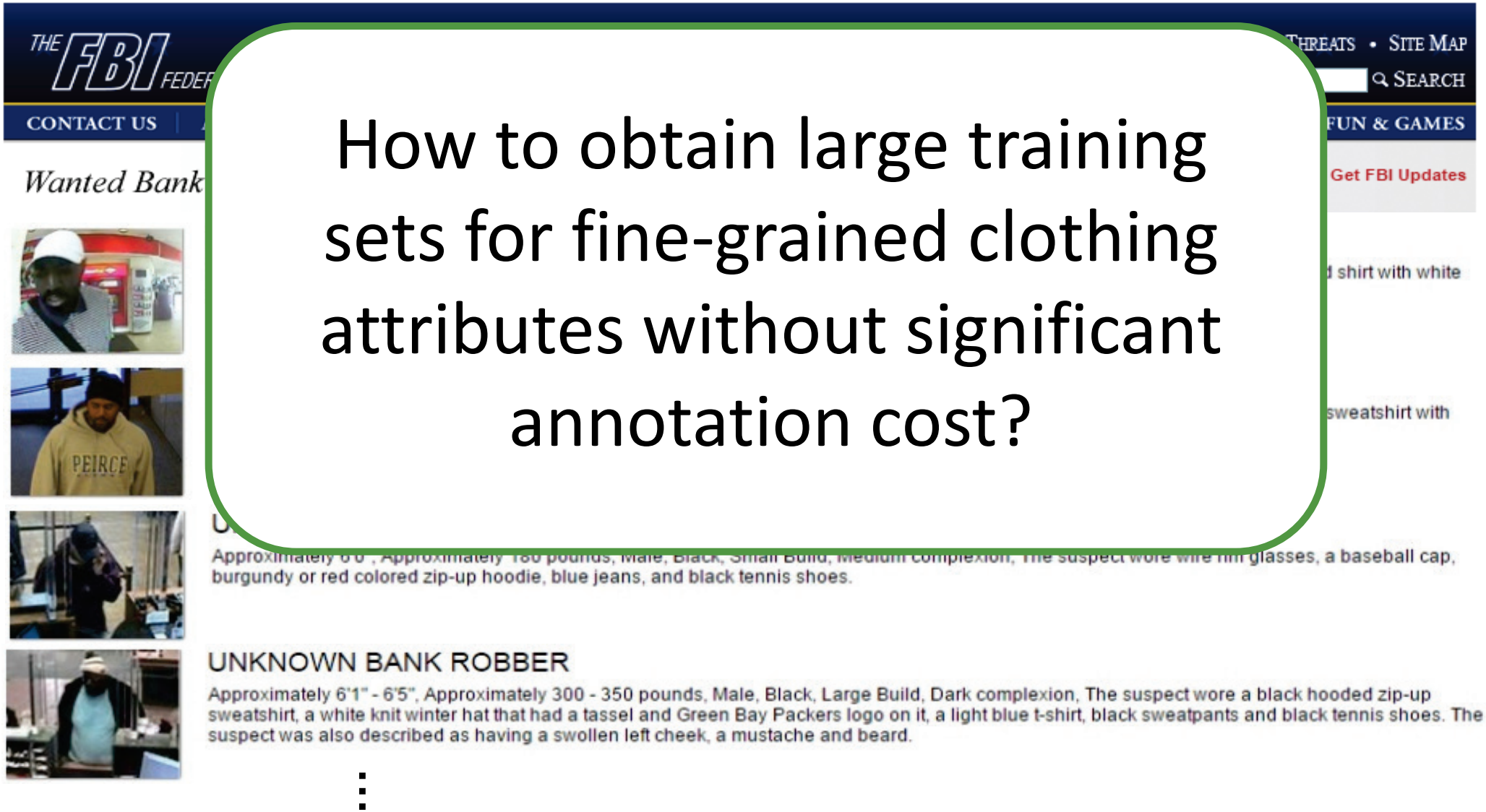
Deep Domain Adaptation Network (CVPR 2016)

Chen, Huang, Feris, Brown, Dong, Yan

Multi-Task Curriculum Transfer Learning (WACV 2017)

Dong, Gong, Zhu

Fine-grained clothing attributes: “light blue” “khaki”, “burgundy”, ‘leather jacket”, “polo-style shirt”, “zip-up windbreaker”, “narrow horizontal stripes”, “LA printed text”, “checkered”, etc.



How to obtain large training sets for fine-grained clothing attributes without significant annotation cost?

Alibaba.com
Global trade starts here.

Categories

< Any Category
< Apparel
T-Shirts

Sleeve Style

Long Sleeve (102561)
Short Sleeve (862176)
Sleeveless (18444)

Gender

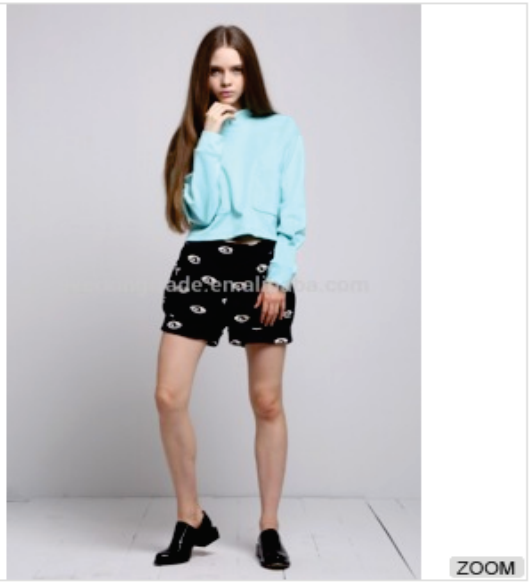
Men (553257)
Unisex (271486)
Women (189514)

Collar

O-Neck (660034)
Polo (231109)
V-Neck (55682)
Turtleneck (13294)

Design

Blank (318364)
With Pattern (614563)



Wholesale Product sky blue color woman long sleeve shirt

Promotion Price: **US $18.80** / Piece
Wholesale Price: ~~US $28.00~~ / Piece
Min. Order: 1 Piece

Shipping: US $24.21 to United States by Express DHL
Processing Time: 7 days.
Quantity: — 1 +

Buy Now | Add To Cart

Add to My Favorites

Product Details | Company Profile

Report Suspicious Activity

Product Description | Packaging & Shipping | Company Information | Our Services | FAQ

Quick Details

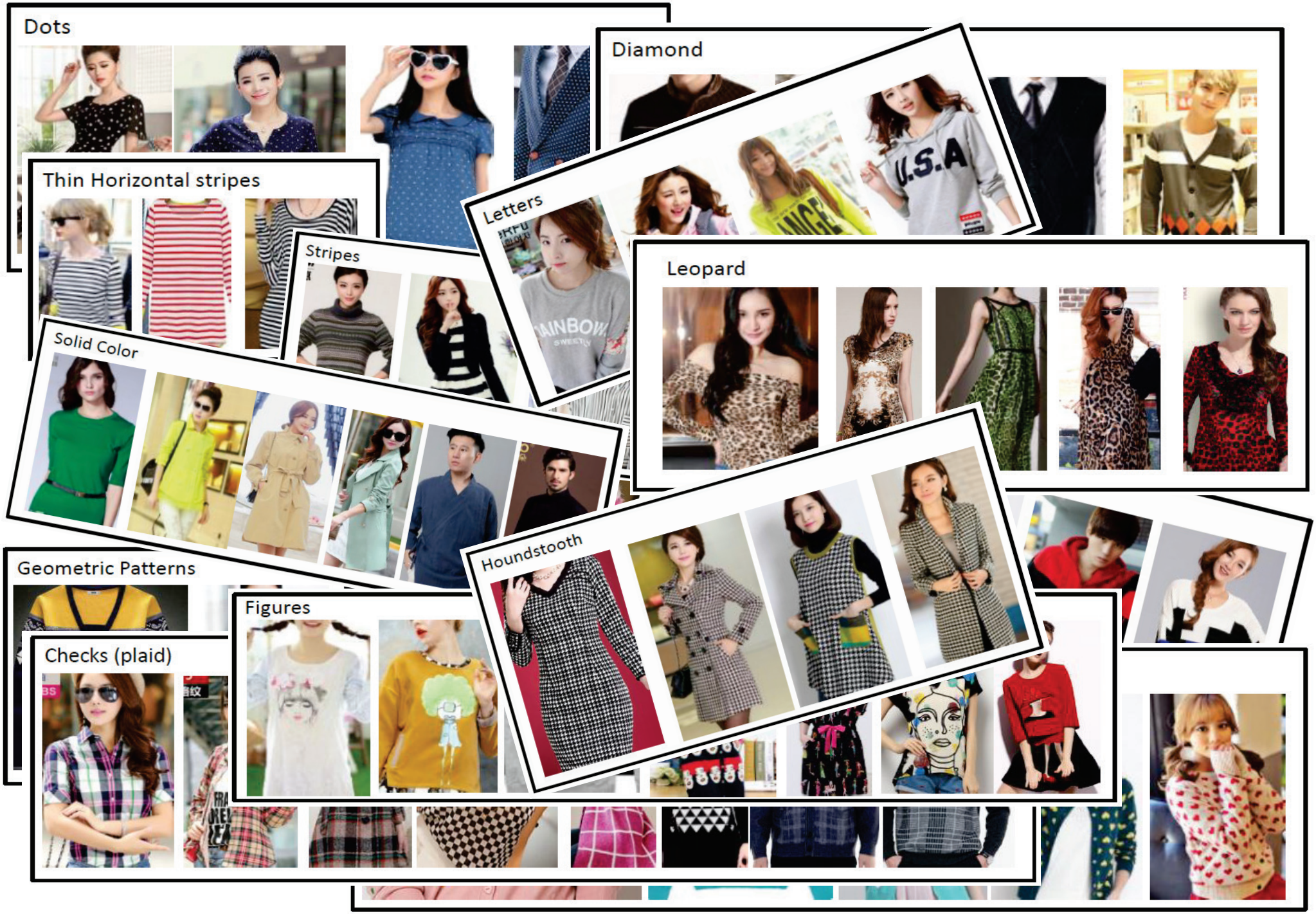
Product Type:	T-Shirts	Supply Type:	In-Stock Items	Age Group:	Adults
Material:	Spandex / Cotton, 95% POLYESTER, ...	Available Quantity:	200	Available Sizes:	S,M,Free
Color:	Blue	Gender:	Women	Collar:	Other
Design:	Blank	Sleeve Style:	Long Sleeve	Place of Origin:	Guangdong China (Mainland)
style:	shirt	OEM/ODM:	YES	quotation time:	within 24 hours
size:	S-M	STOCK:	YES		

Specifications

sky blue color woman long sleeve shirt
Size:S-L
Material: 95% polyester, 5% spandex
zipper in back


Jacket (other)
Harness Vest
Leisure Suit
Down Jacket
T-Shirt
Wedding Dress
Chiffon and Lace Shirt
Denim Jacket
Denim Shirt
Sleeveless Tank Dress
Sweater
Coat
Cardigan
Vest / Waistcoat
Leather


Watermelon Red
Red
Purplish Red
Rose Red
Yellow
Lemon Yellow
Beige
Ginger Yellow
Pink
Purple
Green
Orange
Dark Green
Fluorescent Green
Light Green
Tan
Military/Navy Green
Navy Blue


Dots
Diamond
Thin Horizontal stripes
Letters
Stripes
Leopard
Solid Color
Houndstooth
Geometric Patterns
Figures
Checks (plaid)

Nearly One Million Images
Hundreds of Clothing Attributes

After Data Curation:
341,021 images and 67 attributes

Clothing Attributes in the wild

- E-commercial
- Describing people in detail
- Automatic image annotation


(a) Query Image (b) Top-6 Retrieval Results

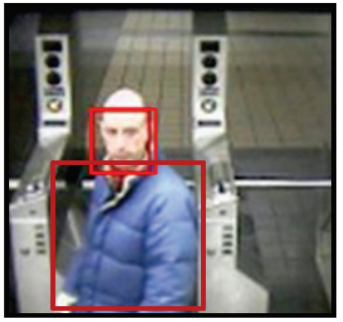
Purplishred dress, solid color

Transfer Learning: How to bridge the gap between the online shopping domain and the surveillance domain?



↳ Color
↳ Red
↳ **Watermelon Red**





↳ Type
↳ Jacket
↳ **Down Jacket**



↳ Pattern
↳ Stripes
↳ **Thin Horizontal Stripes**

Challenges

- The variance of Intra-class is very large caused by poor lighting, cluttered scenes, unknown viewpoint…

Deep Learning Frameworks

well-annotated, fine-grained and **large-scaled** samples.

a **rich** source of web images and their meta-text on online shops

Challenges

Shop images are:

- Well-posed by models
- Clean background
- Good lighting

How to transfer?

Shop images

Domain Drift

In the wild images

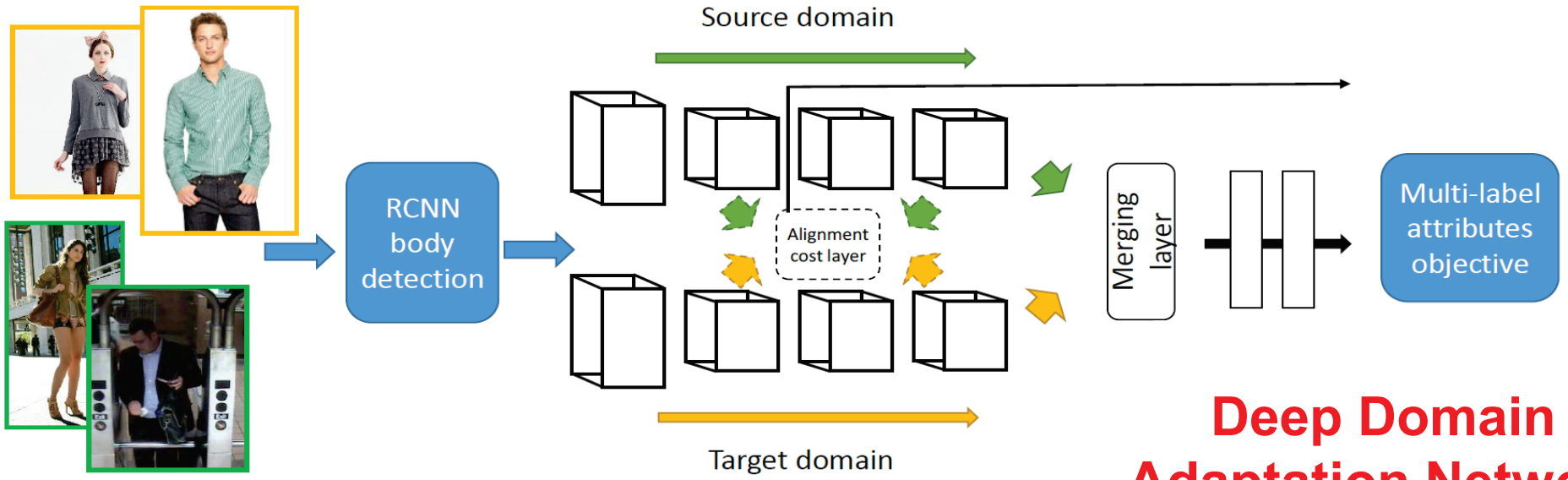
Traditional deep learning approaches to domain adaptation:

- Pre-train a source domain CNN deep model
  - Using a large set of source domain data
- Re-train the last layer for the new class labels
  - Assumes that all previous feature layers can be re-used across domains which may not be optimal.
  - Cannot handle prediction of attribute labels for which training samples are available only in the source domain
- Fine-tuning through backpropagation at a lower learning rate
  - Requires larger amount of target domain samples
  - After adaptation, learned features are biased to the target domain and may not be consistent with the source domain

Current approaches rely on two separate stages: 1) source domain is modeled 2) model is adapted to the target domain

A single model learning both source and target domains **jointly** through a double-path deep neural network architecture

- Joint learning of domain-invariant hierarchical features. The domain information is transferred at multiple levels within intermediate layers.
- Alignment cost layers connect the two paths to ensure both feature alignment and attribute label consistency of the representation across the domains.
- A merging layer for feature fusion by a direct max operation without pooling (not maxpooling)



Deep Domain Adaptation Network (DDAN)

A single model learning both source and target domains **jointly** through a double-path deep neural network architecture

- Joint learning of domain-invariant hierarchical features. The domain information is transferred at multiple levels within inte
- Alignment cost layers connec
  label consistency of the repre
- A merging layer for feature fu

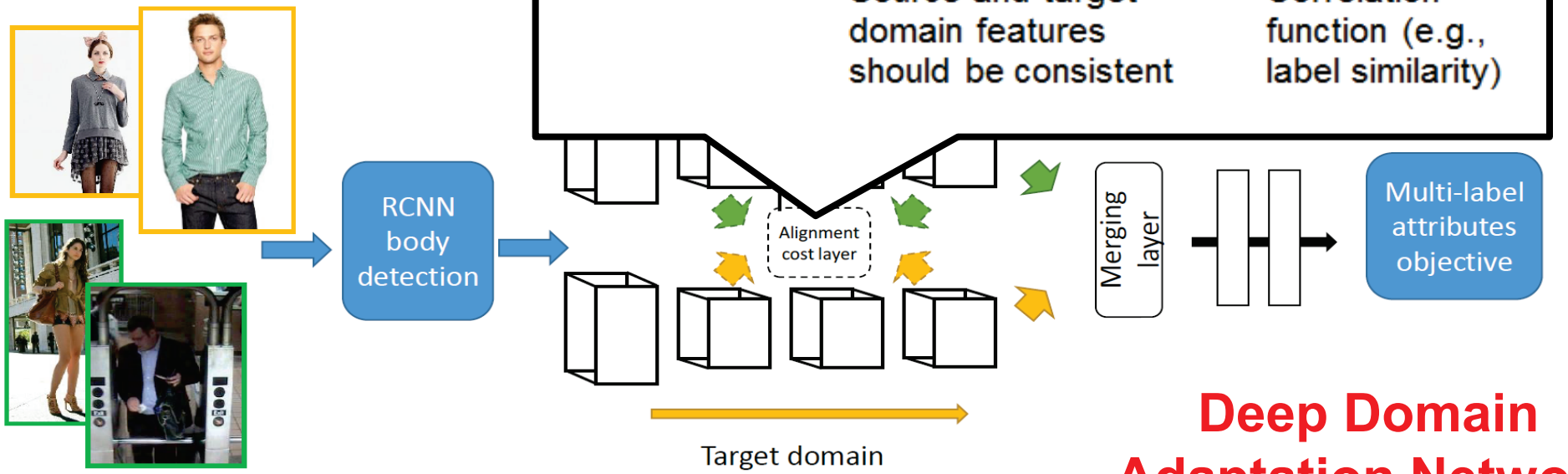
ng)

Alignment cost layers:

$$f(s,t) = ||X_s - X_t|| \times \lambda\phi(s,t)$$

Source and target domain features should be consistent

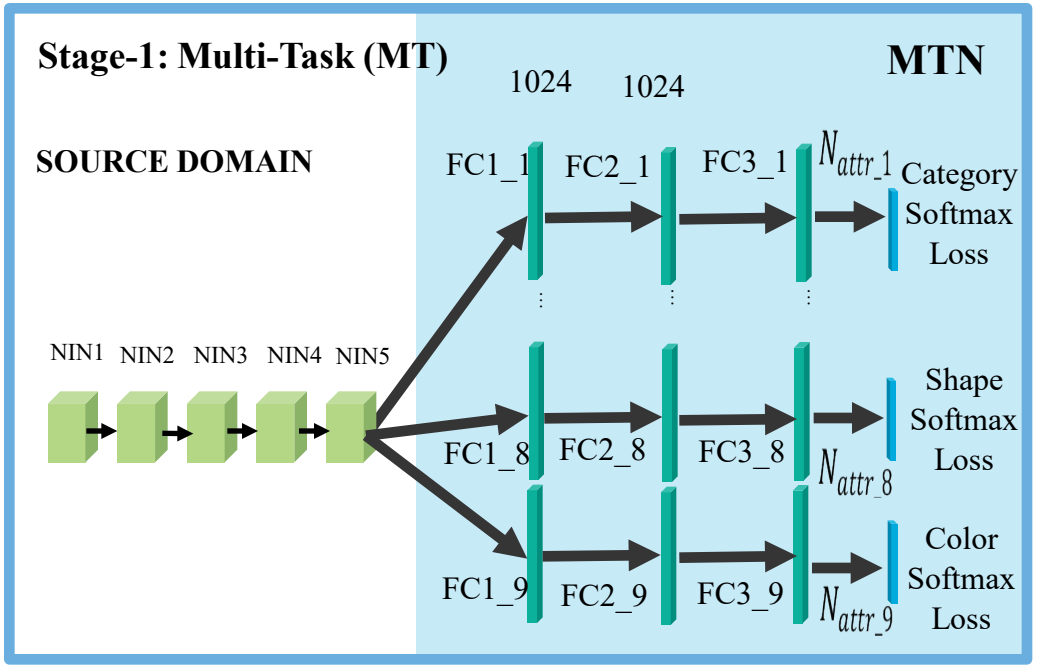
Correlation function (e.g., label similarity)



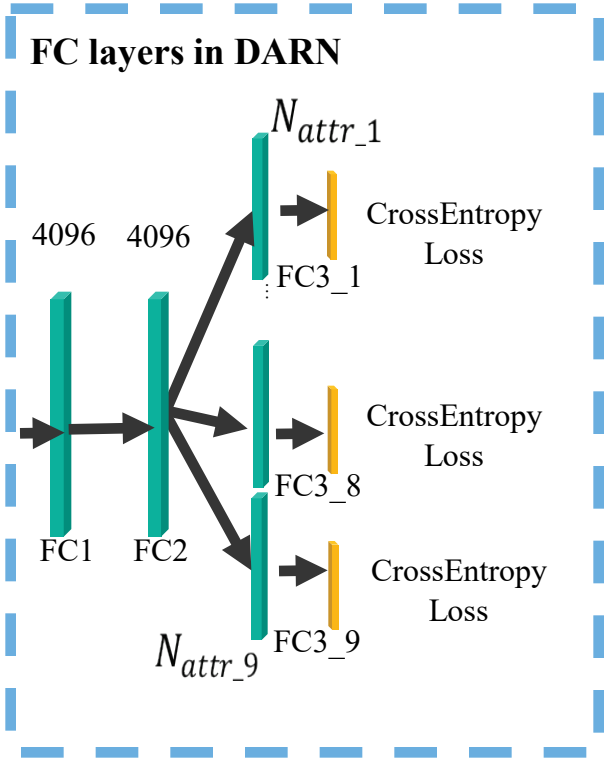
Deep Domain Adaptation Network (DDAN)

Multi-Task Curriculum Transfer Network

- Multi-Task Attributes Network (**MTN**)

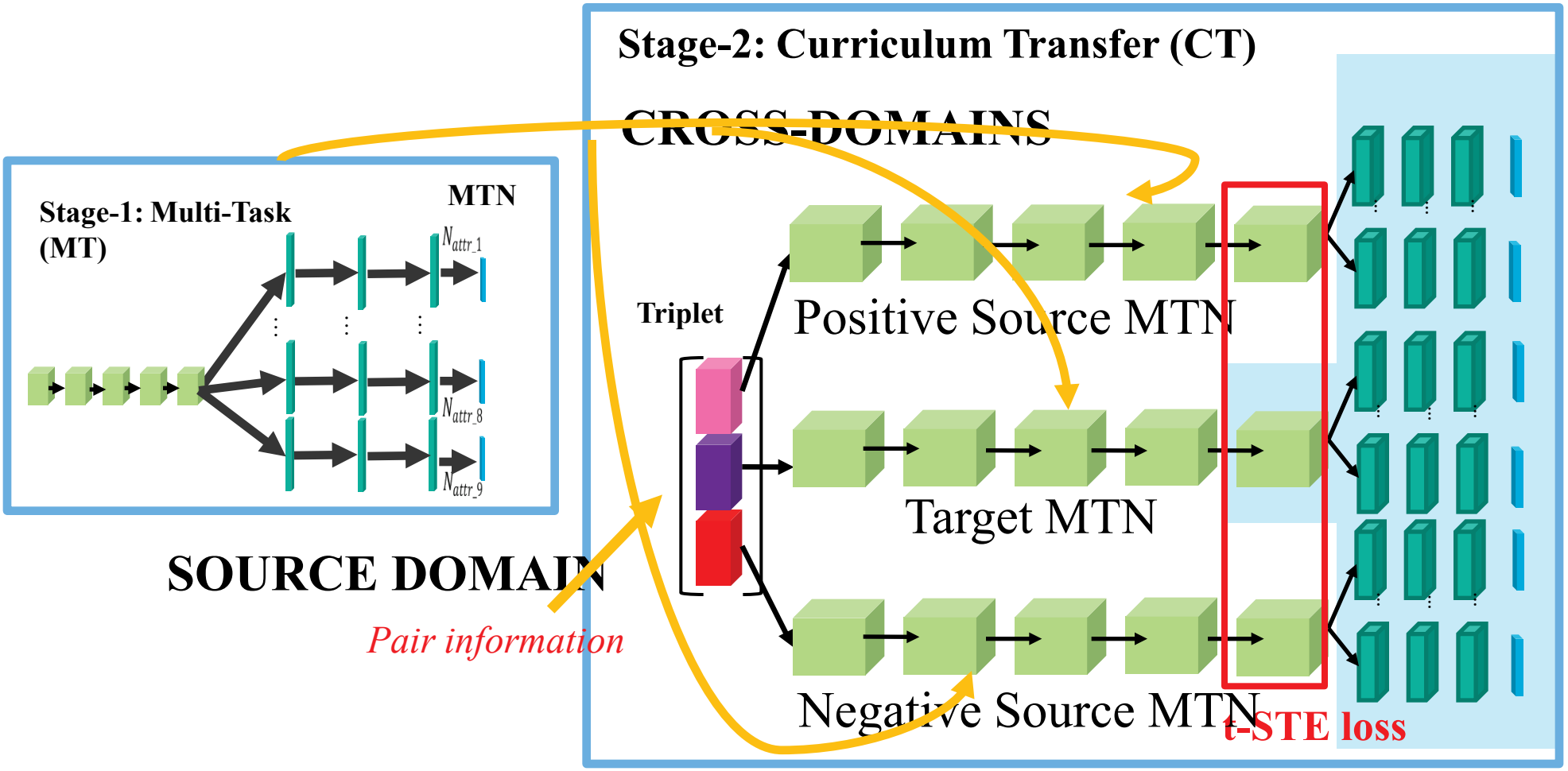


Single Task model (DDAN)



Multi-Task Curriculum Transfer Network

- Cross domain Curriculum learning (**MTCT**)



Experiments

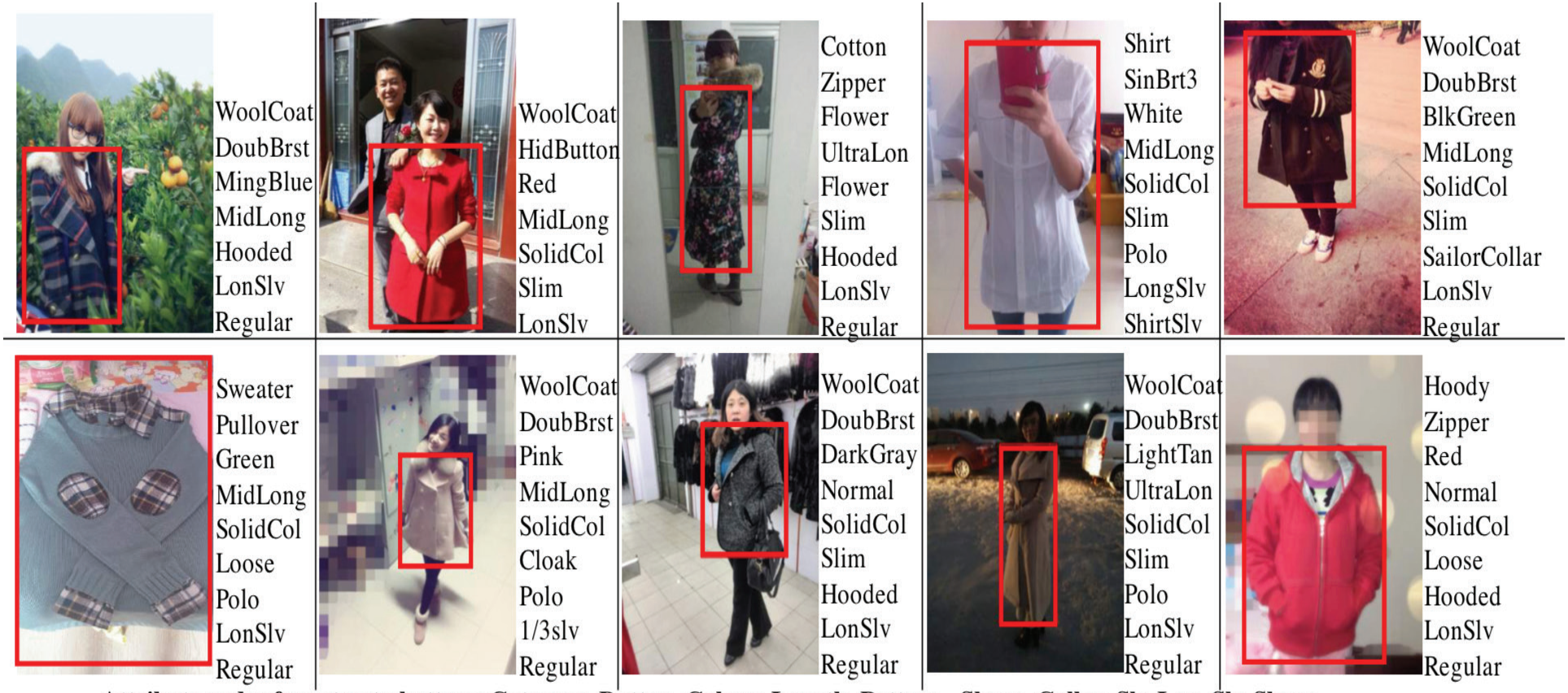
- Comparison the State-Of-The-Art

Method	Category	Button	Colour	Length	Pattern	Shape	Collar	Slv-Len	Slv-Shp	mAP^{cls}	mP^{ins}	mR^{ins}
DDAN [8]	12.56	24.13	20.72	35.91	61.67	47.14	31.17	80.63	73.96	43.10	45.41	52.20
DARN [27]	52.55	37.48	58.24	51.49	67.53	47.70	47.77	82.04	73.72	57.61	57.79	67.29
FashionNet [40]	55.85	39.52	60.33	53.08	68.65	49.79	52.17	83.79	75.34	59.84	59.97	69.74
MTCT	**65.96**	**43.57**	**66.86**	**58.27**	**70.55**	**51.40**	**58.97**	**86.05**	**77.54**	**64.35**	**64.97**	**75.66**

- Effects of Multi-Task and Transfer Learning

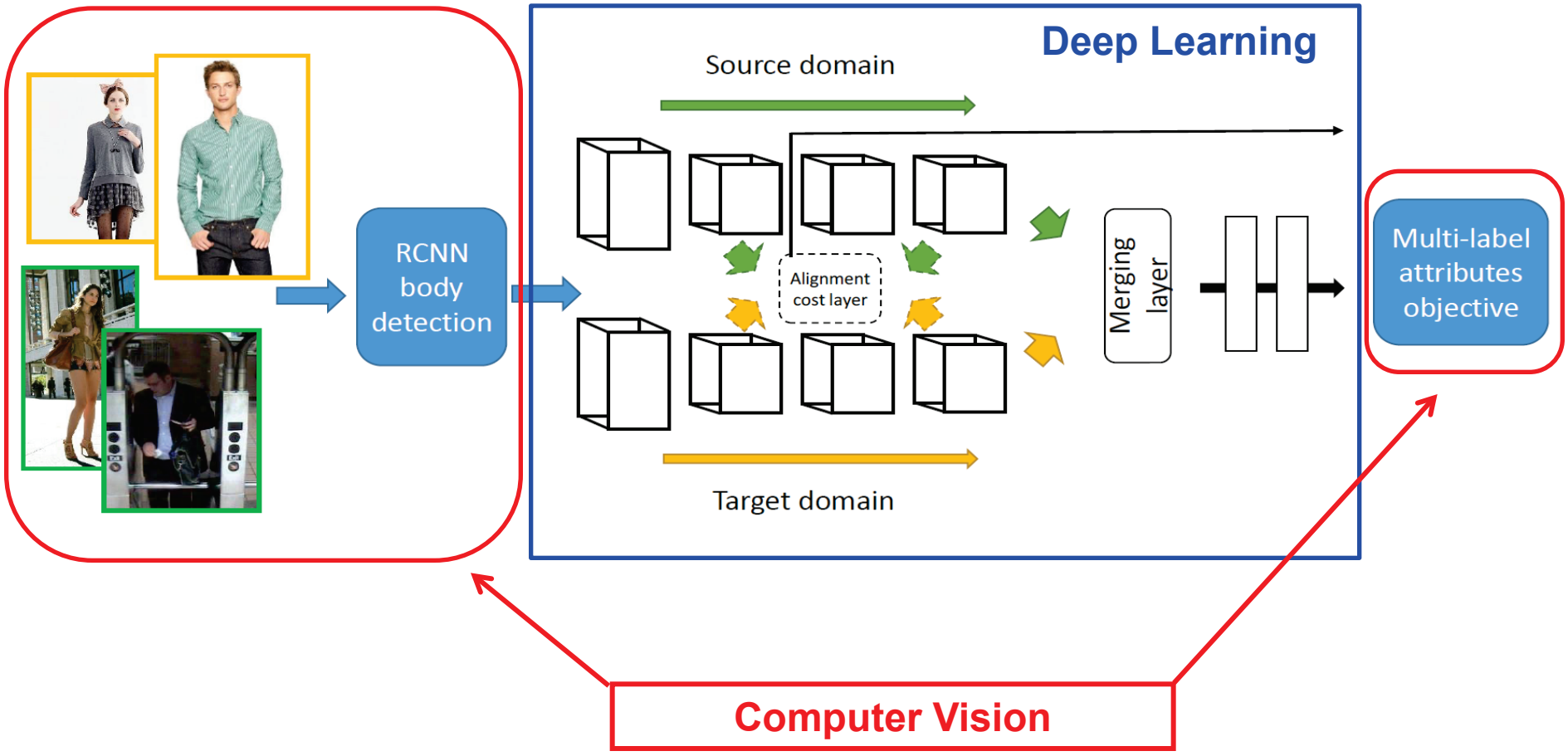
Method	Category	Button	Colour	Length	Pattern	Shape	Collar	Slv-Len	Slv-Shp	mAP^{cls}	mP^{ins}	mR^{ins}
JAN(NoAdpt) [27]	34.08	35.87	43.08	44.21	63.76	43.40	40.50	78.13	71.08	50.46	50.39	58.40
MTN(NoAdpt)	35.77	33.77	44.13	44.76	65.26	45.75	40.85	79.76	72.40	51.38	51.82	60.00
MTN(UD)	54.10	40.65	57.88	51.35	67.80	49.79	49.09	83.61	74.60	58.76	60.16	70.00
MTN(FTT)	61.92	42.65	65.43	55.16	70.06	49.00	50.55	85.54	76.04	61.82	62.53	72.76
MTCT	**65.96**	**43.57**	**66.86**	**58.27**	**70.55**	**51.40**	**58.97**	**86.05**	**77.54**	**64.35**	**64.97**	**75.66**

Examples



Attribute order from top to bottom: Category, Button, Colour, Length, Pattern, Shape, Collar, Slv-Len, Slv-Shape

From Image Classification to Object Recognition



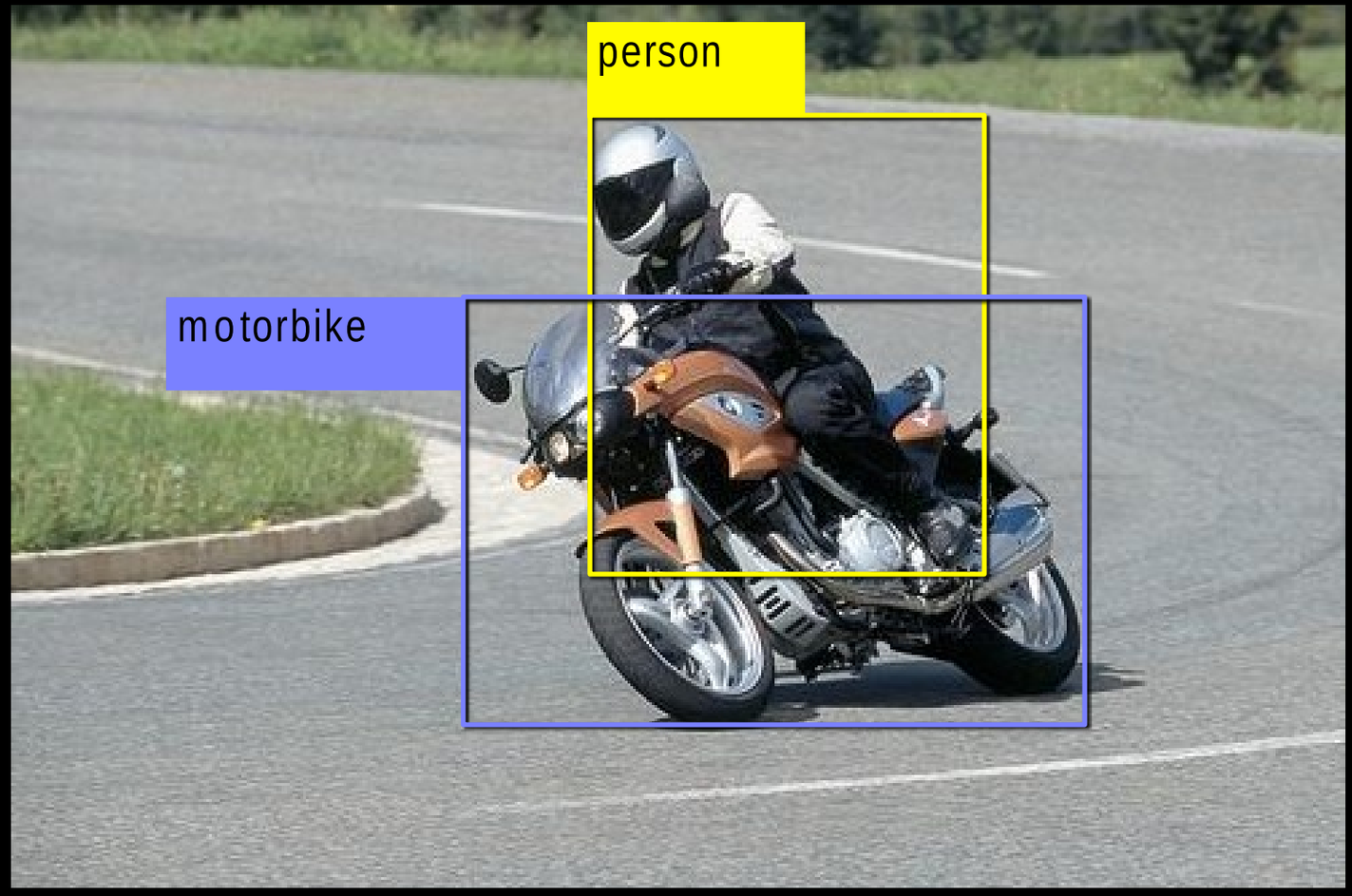
R-CNN
(Rich feature hierarchies for accurate object detection and semantic segmentation)

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

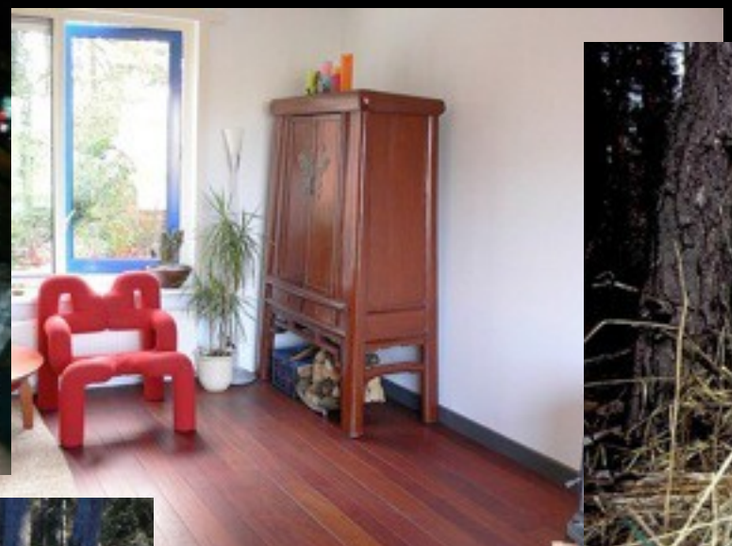
Detection & Segmentation

input

background



PASCAL VOC

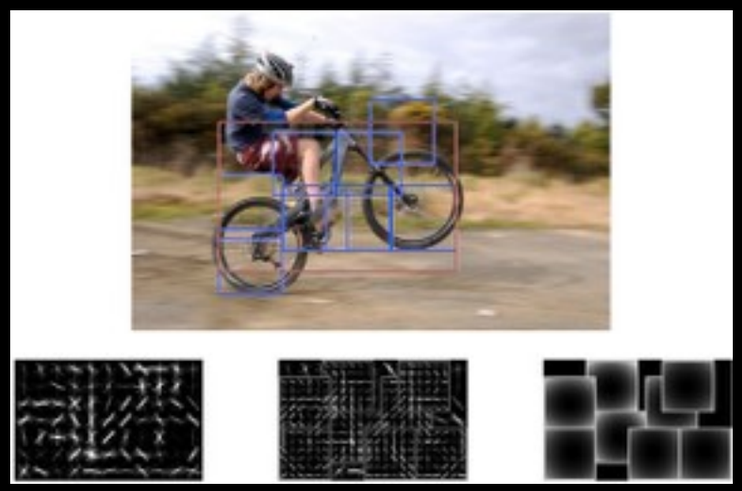





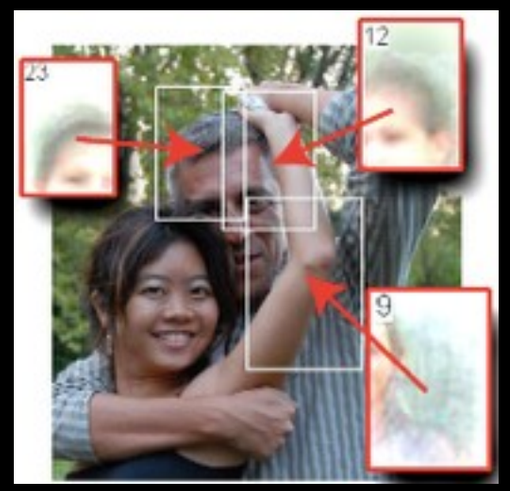

Example PASCAL VOC images

Dominant detection methods

1. Part-based sliding window methods (HOG)

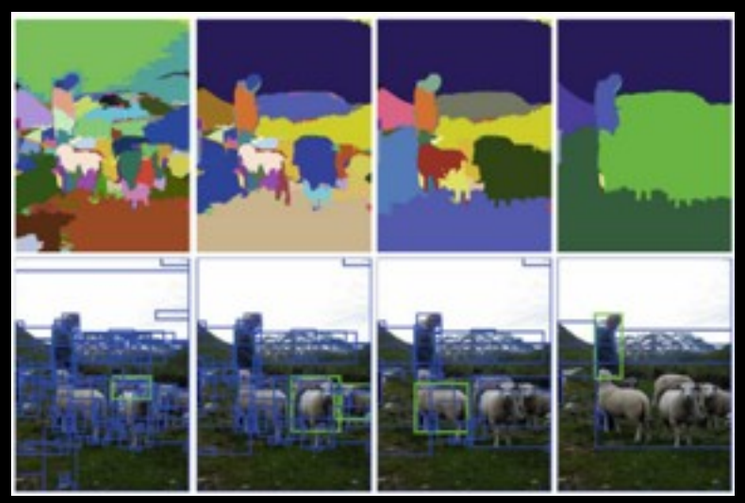


DPM



Poselets

2. Region-proposal classifiers (SIFT++ BoW)



Russell et al. 2006

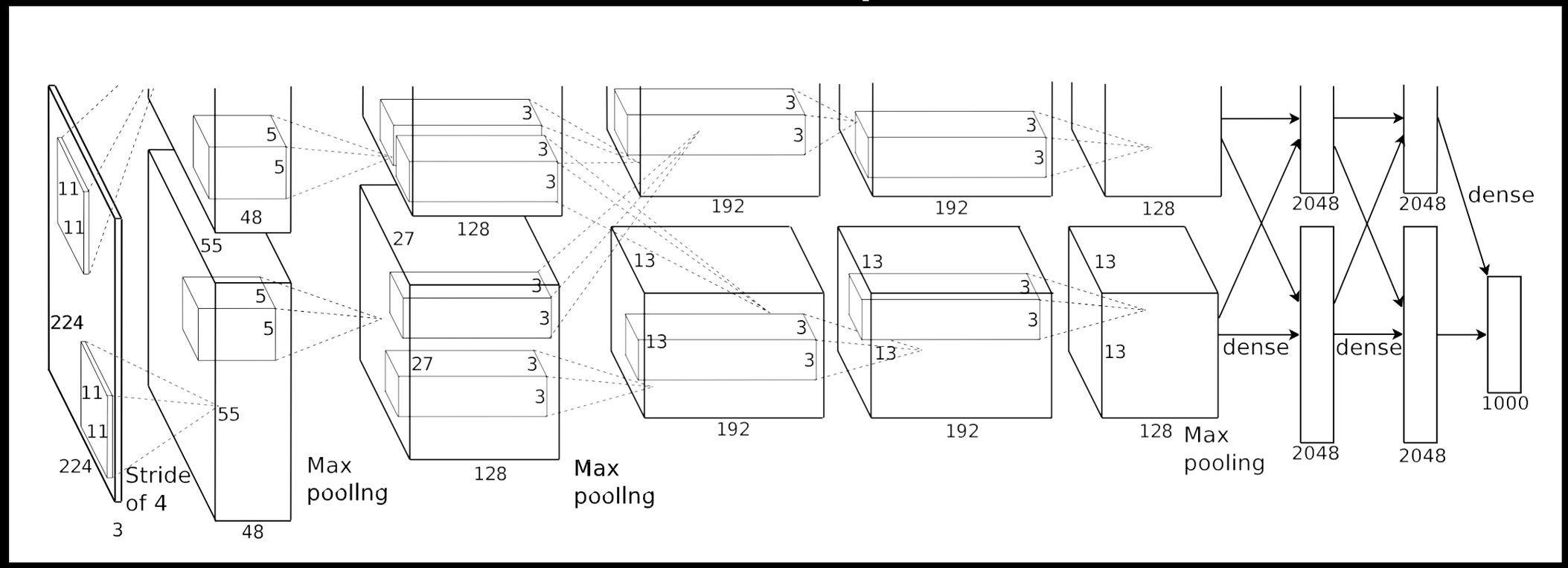
Gu et al. 2009

van de Sande et al. 2011

> “selective search”

ImageNet LSVRC'12 winner

AlexNet – Toronto "SuperVision" CNN



Krizhevsky, Sutskever, and Hinton.
ImageNet Classification with Deep Convolutional Neural Networks.
NIPS 2012.

cf. LeCun et al. Neural Comp. '89 & Proc. of the IEEE '98

Impressive ImageNet results!

Task: 1000-way whole-image classification

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

metric: classification error rate (lower is better)

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

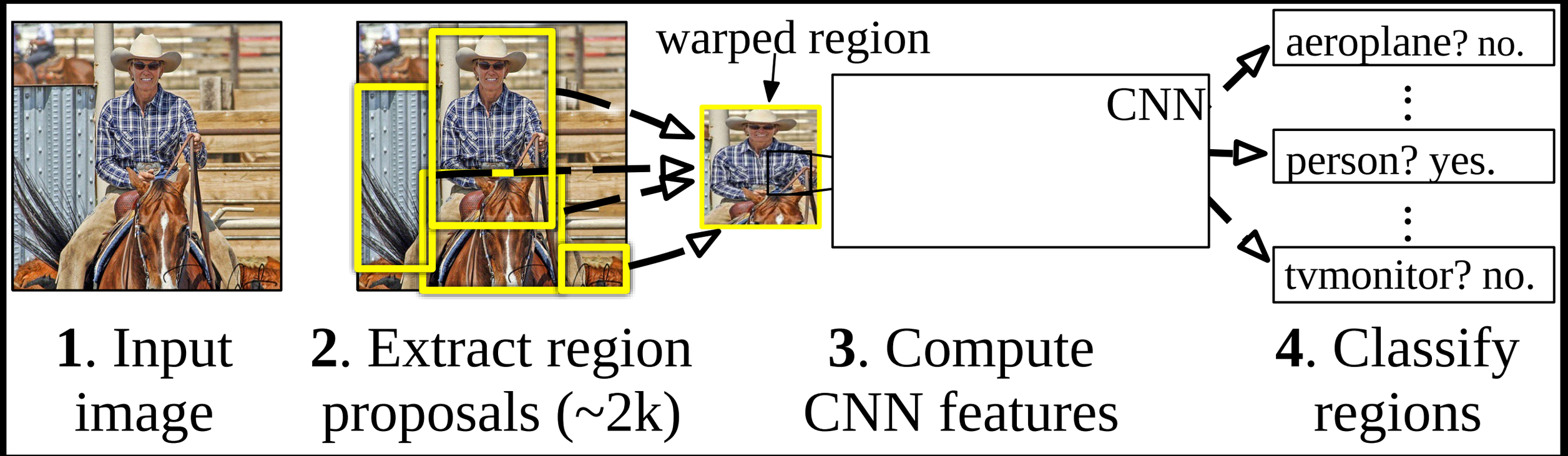
(Donahue, Jia, et al. DeCAF Tech Report)

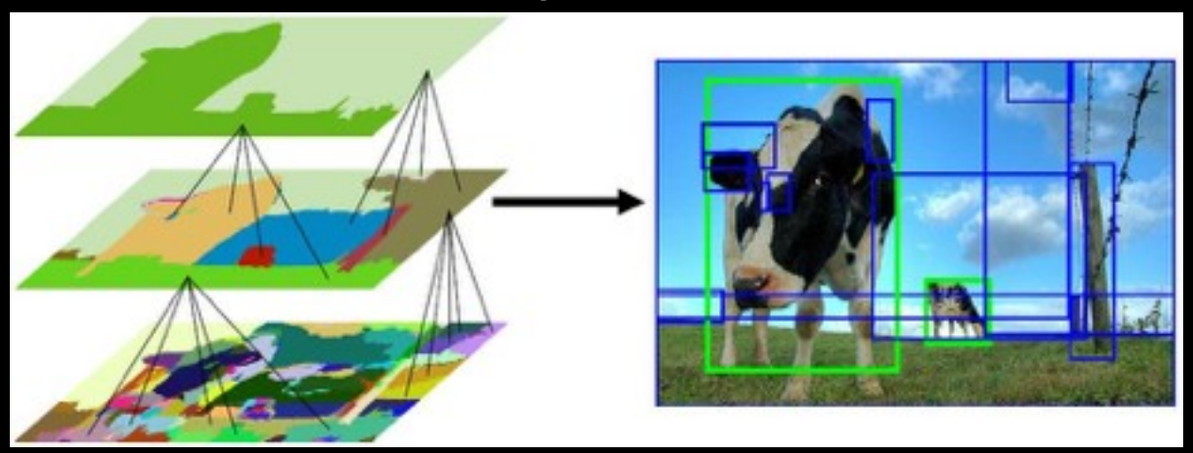
Objective

Question: If the AlexNet CNN model can be made to work as an object detector?

Object detection system

R-CNN: "Regions with CNN features"



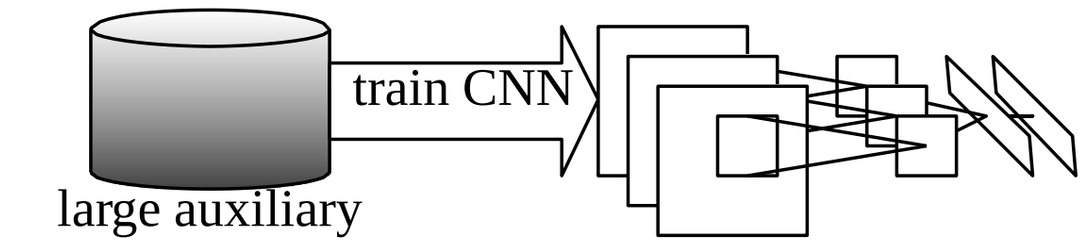


(e.g. selective search)

(With a few minor tweaks: semantic segmentation)

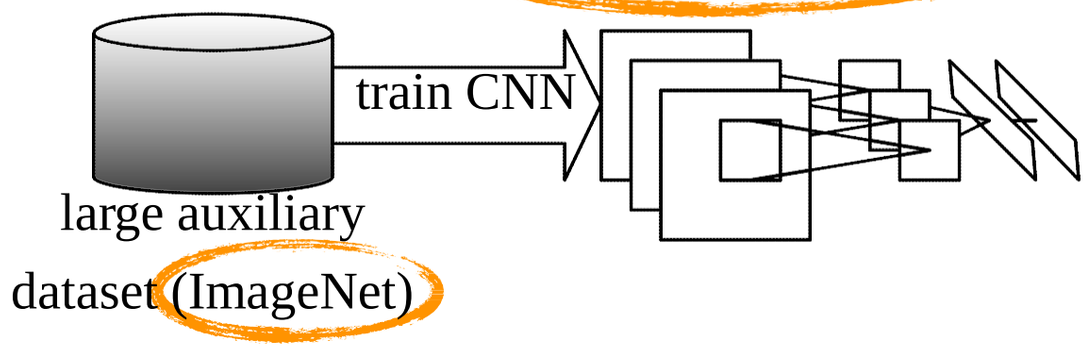
Training

1. Pre-train CNN for **image classification**



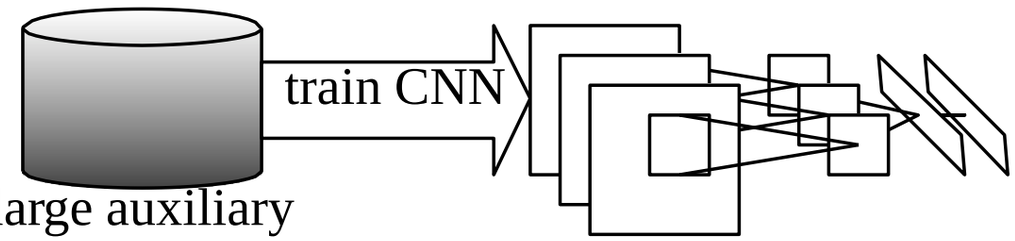
Training

1. Pre-train CNN for **image classification**

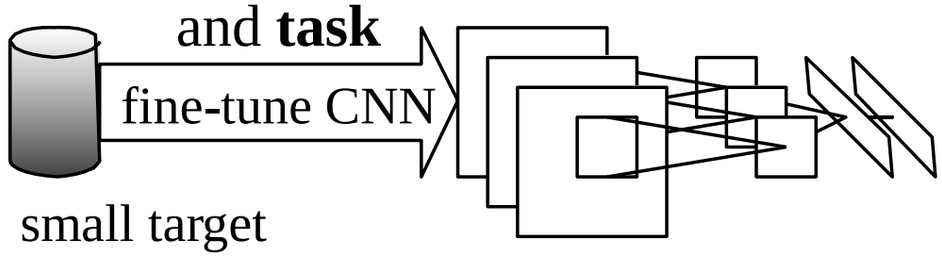


Training

1. Pre-train CNN for **image classification**



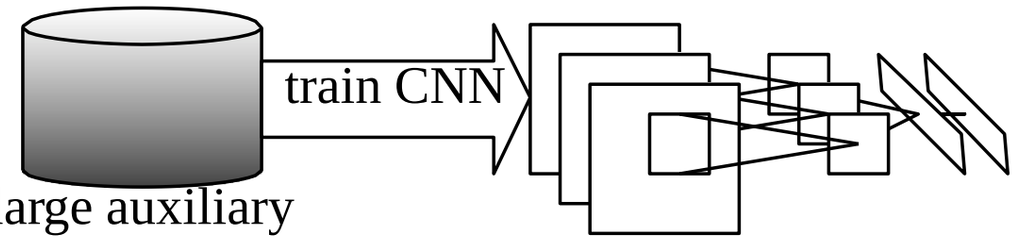
2. Fine-tune CNN on **target dataset** and **task**



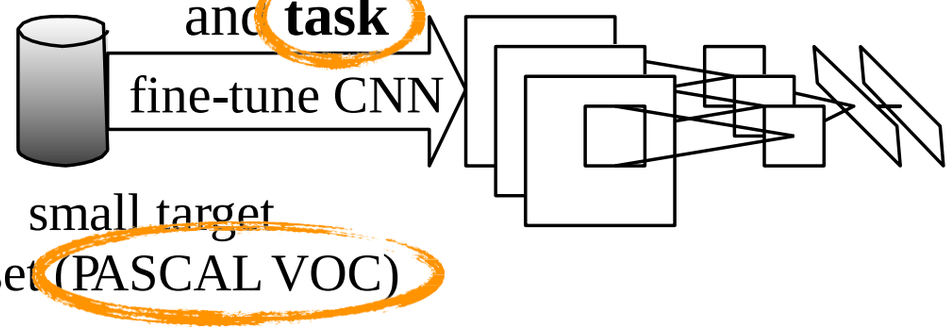
(optional)

Training

1. Pre-train CNN for **image classification**



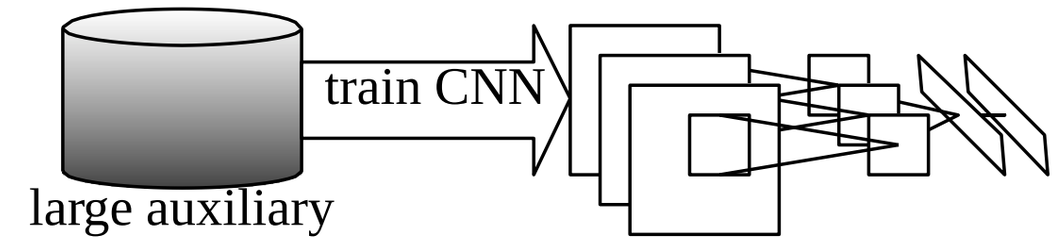
2. Fine-tune CNN on **target dataset** and **task**



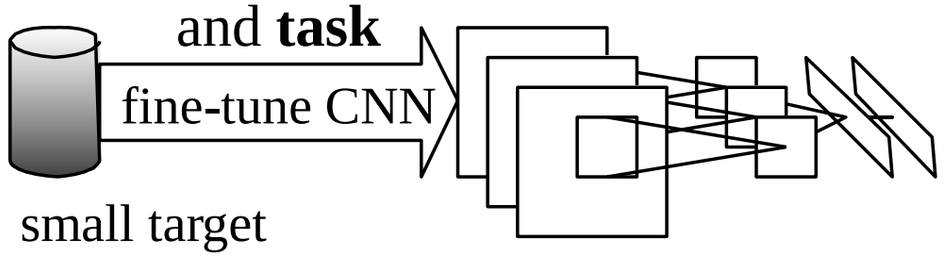
(optional)

Training

1. Pre-train CNN for **image classification**

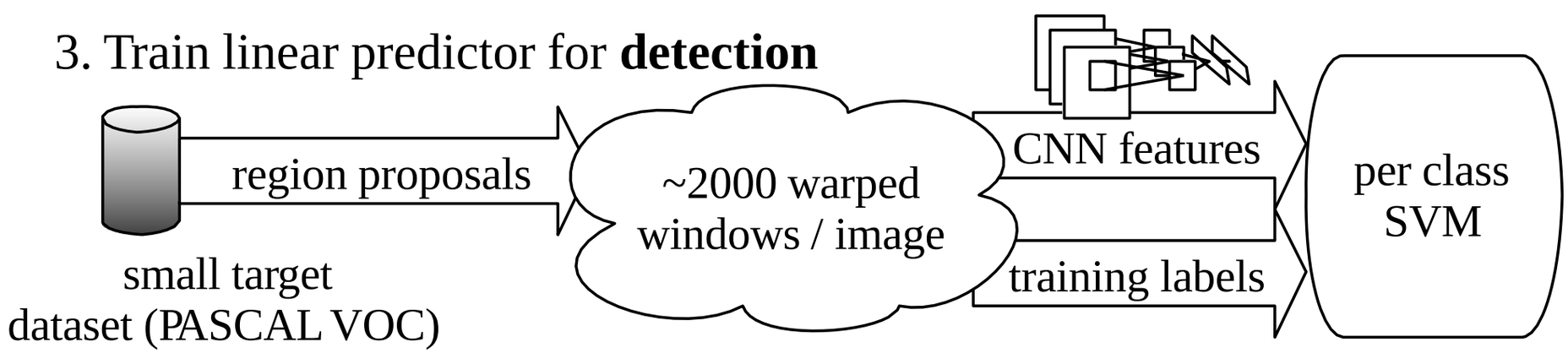


2. Fine-tune CNN on **target dataset** and **task**

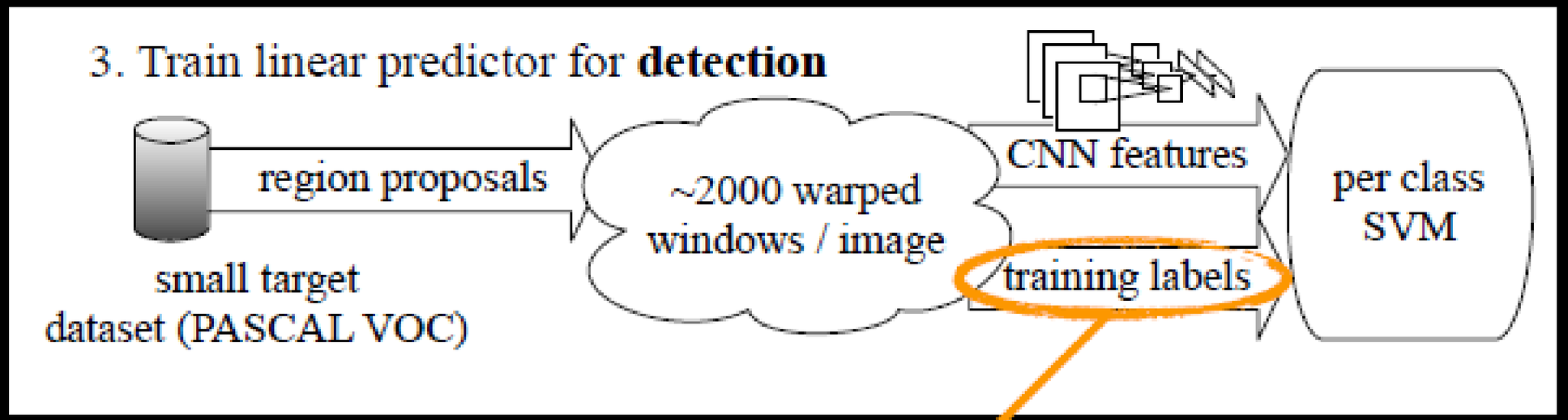


(optional)

3. Train linear predictor for **detection**



Training labels

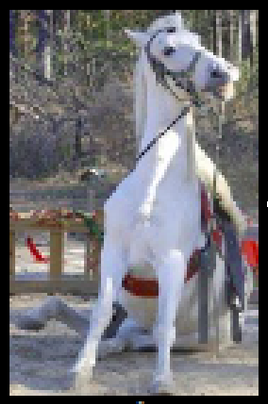


labeling protocol

positives = ground truth

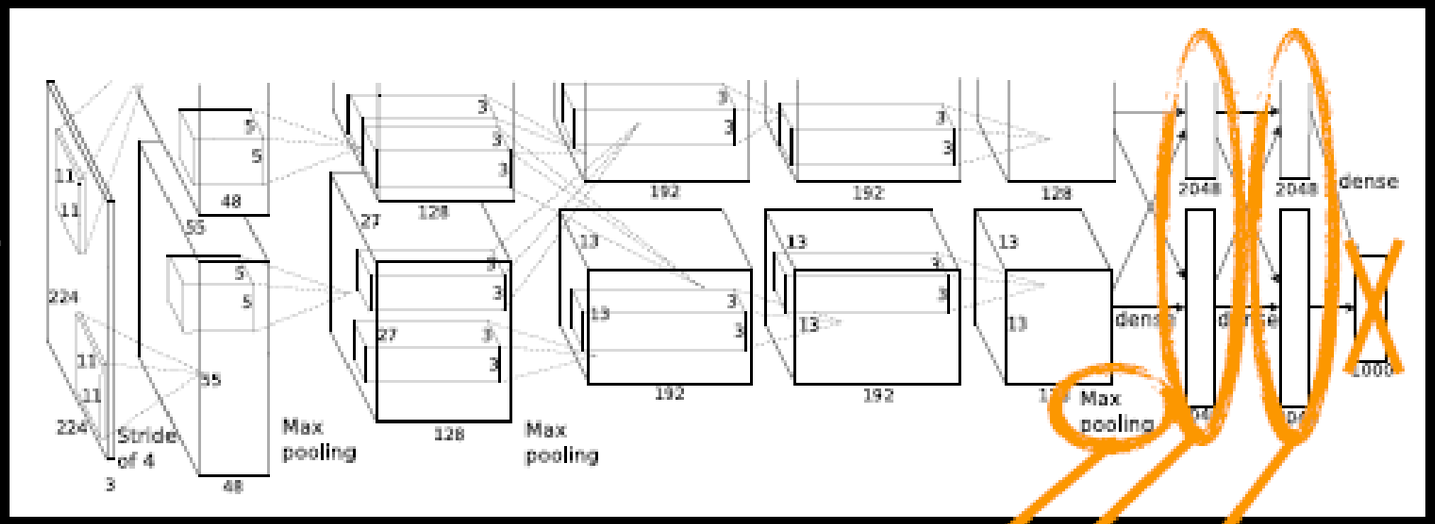
negatives = max IoU < 0.3

CNN features for detection



region

warped region



$pool_5$: 6 x 6 x 256 = 9216-dim

6.4% / 15% non-zero

fc_6: 4096-dimensional

71.2% / 20% nz

fc_7: 4096-dimensional

100% / 20% nz

Results

reference	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
Selective Search (Uijlings et al. 2012)		35.1%
Regionlets (Wang et al. 2013)	41.7%	

metric: mean average precision (higher is better)

Results

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
Selective Search (Uijlings et al. 2012)		35.1%
Regionlets (Wang et al. 2013)	41.7%	39.7%
pre-trained only: R-CNN $pool_5$	40.1%	
pre-trained only: R-CNN fc_6	43.4%	
pre-trained only: R-CNN fc_7	42.6%	

metric: mean average precision (higher is better)

Results

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
Selective Search (Uijlings et al. 2012)		35.1%
Regionlets (Wang et al. 2013)	41.7%	39.7%
R-CNN $pool_5$	40.1%	
R-CNN fc_6	43.4%	
R-CNN fc_7	42.6%	
R-CNN FT $pool_5$	42.1%	
R-CNN FT fc_6	47.2%	
R-CNN FT fc_7	48%	

fine-tuned

metric: mean average precision (higher is better)

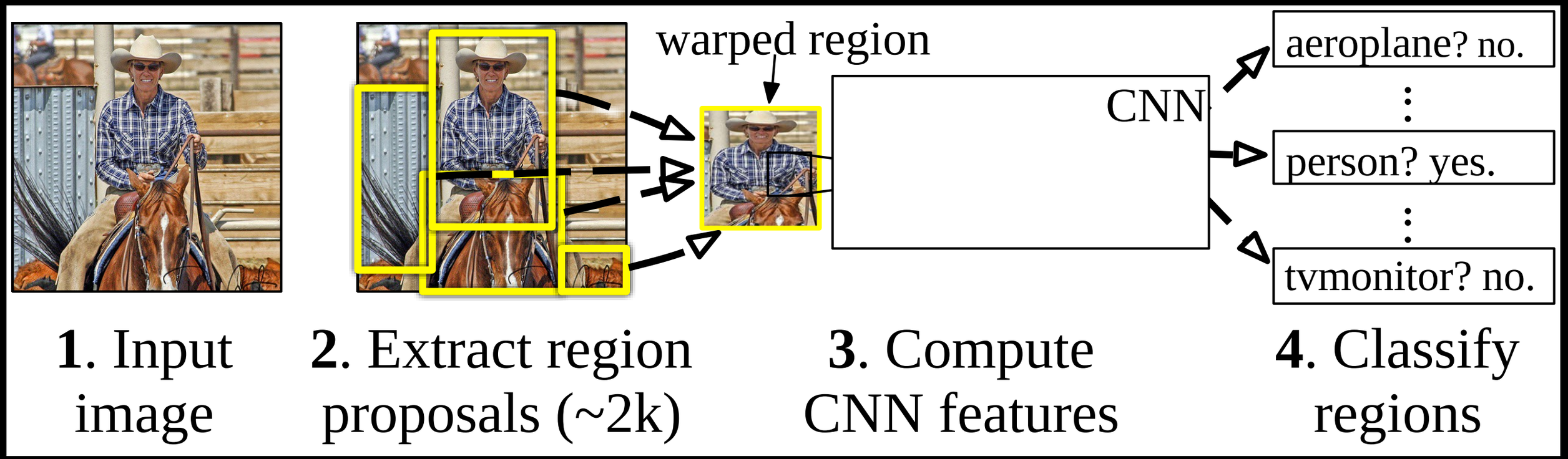
Results — more

	VOC 2007	VOC 2010
DPM v5 (Girshick et al. 2011)	33.7%	29.6%
Selective Search (Uijlings et al. 2012)		35.1%
Regionlets (Wang et al. 2013)	41.7%	39.7%
pre-trained only: R-CNN $pool_5$	~~40.1%~~ 44.0%	
pre-trained only: R-CNN fc_6	~~43.4%~~ 46.2%	
pre-trained only: R-CNN fc_7	~~42.6%~~ 43.5%	

metric: mean average precision (higher is better)

CV and DL together

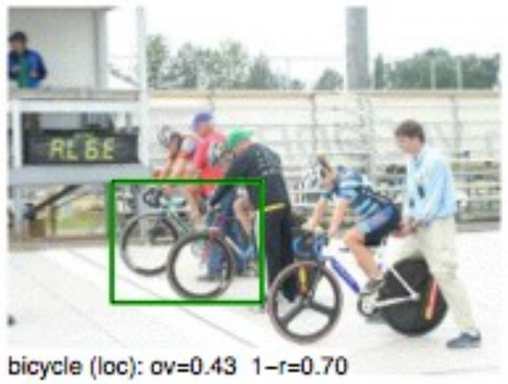
Good features are not enough!



Computer Vision | Deep Learning | Computer Vision

Top bicycle FPs (AP 62.5%)

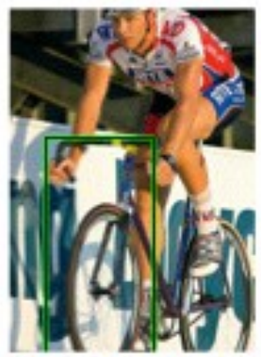

bicycle (loc): ov=0.36 1−r=0.78


bicycle (loc): ov=0.43 1−r=0.70


bicycle (loc): ov=0.32 1−r=0.69


bicycle (loc): ov=0.43 1−r=0.67


bicycle (loc): ov=0.34 1−r=0.66


bicycle (loc): ov=0.47 1−r=0.65


bicycle (loc): ov=0.33 1−r=0.61


bicycle (loc): ov=0.28 1−r=0.61

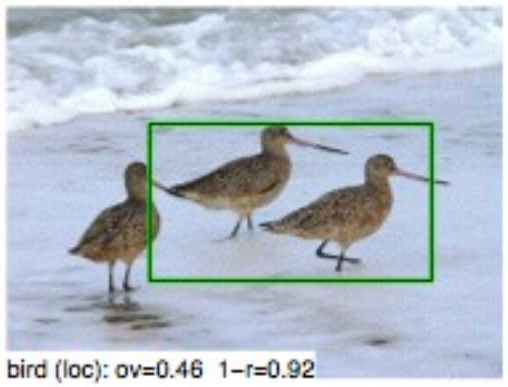

bicycle (sim): ov=0.00 1−r=0.60

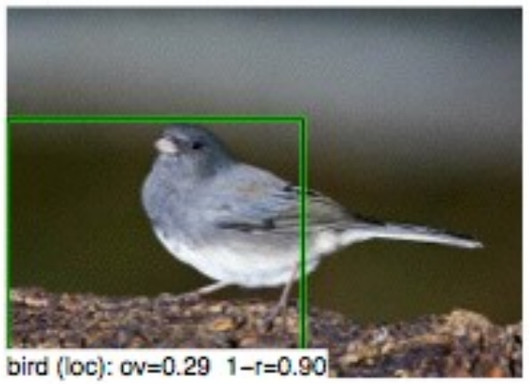

bicycle (sim): ov=0.00 1−r=0.59


bicycle (loc): ov=0.18 1−r=0.59


bicycle (loc): ov=0.46 1−r=0.58

Top bird FPs (AP 41.4%)


bird (loc): ov=0.46 1−r=0.92


bird (loc): ov=0.29 1−r=0.90


bird (loc): ov=0.49 1−r=0.89

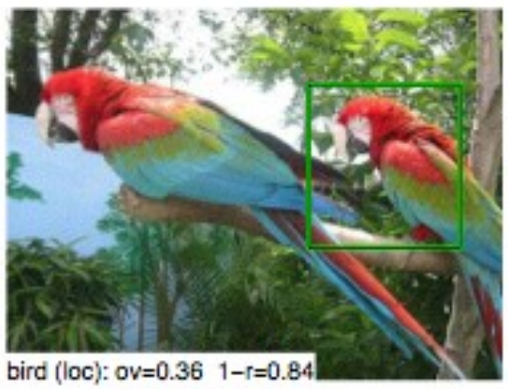

bird (loc): ov=0.35 1−r=0.92

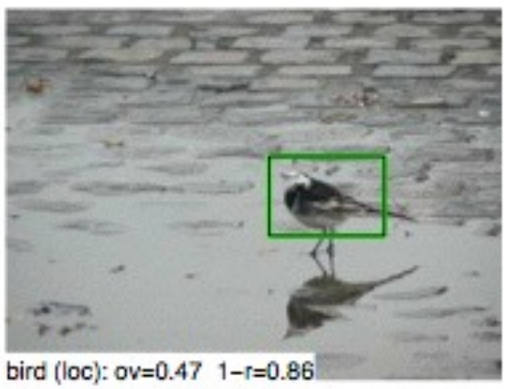

bird (loc): ov=0.36 1−r=0.87

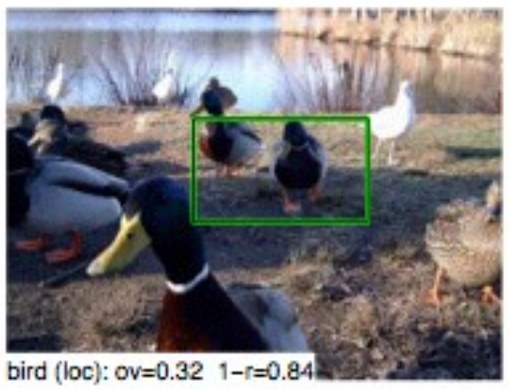

bird (loc): ov=0.39 1−r=0.86

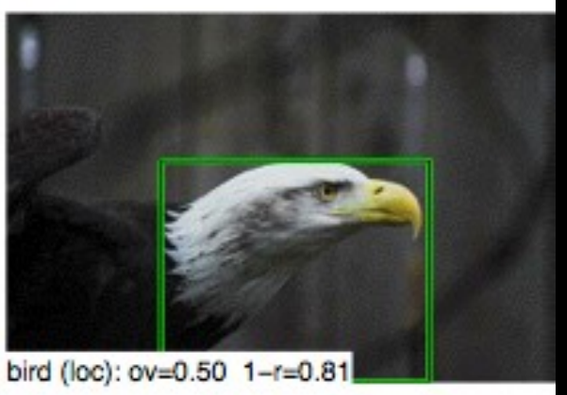

bird (loc): ov=0.32 1−r=0.89


bird (loc): ov=0.38 1−r=0.86

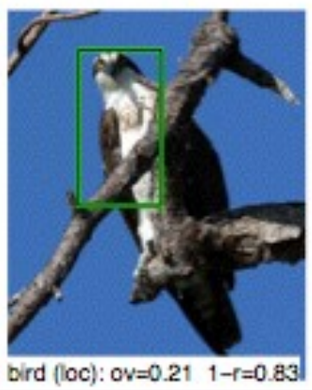

bird (loc): ov=0.36 1−r=0.84


bird (loc): ov=0.47 1−r=0.86

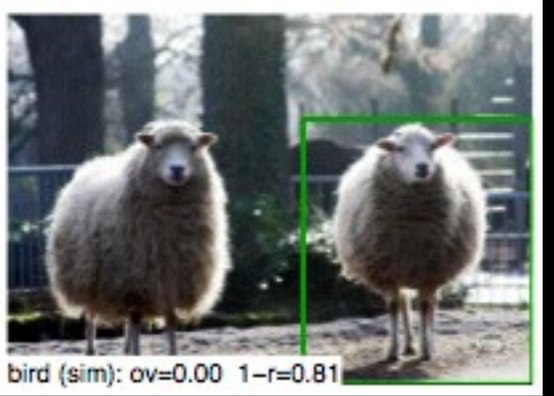

bird (loc): ov=0.32 1−r=0.84


bird (loc): ov=0.50 1−r=0.81


bird (loc): ov=0.34 1−r=0.81


bird (loc): ov=0.21 1−r=0.83


bird (loc): ov=0.18 1−r=0.81


bird (sim): ov=0.00 1−r=0.81

Visualizing features

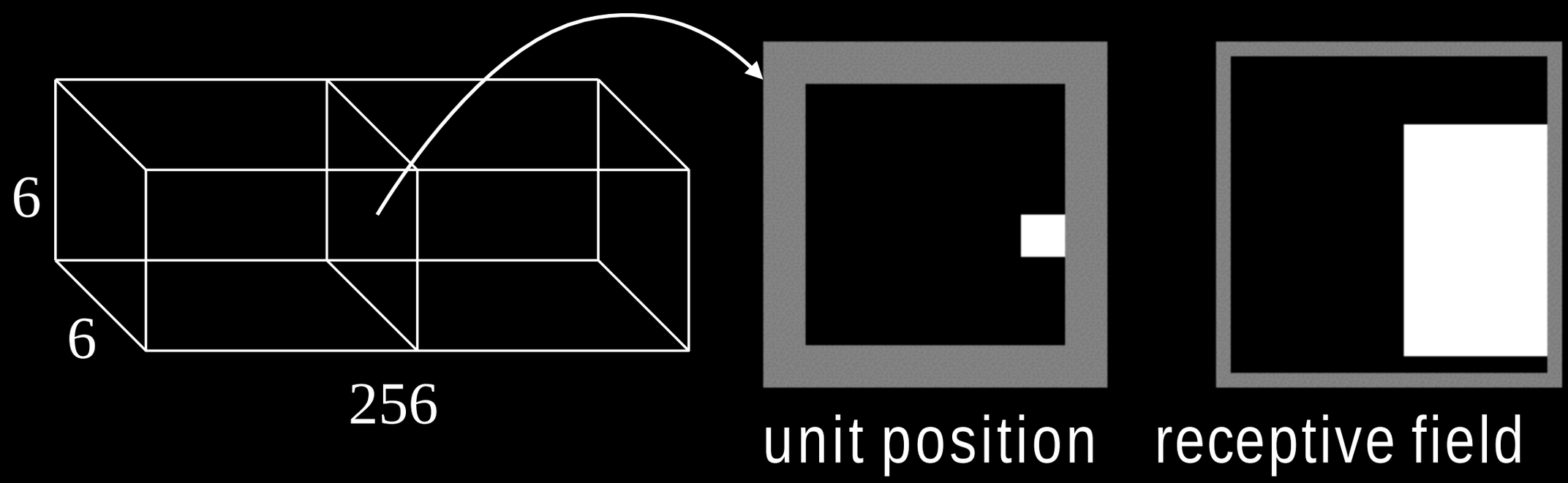
> What does $pool_5$ learn?

> Recap:

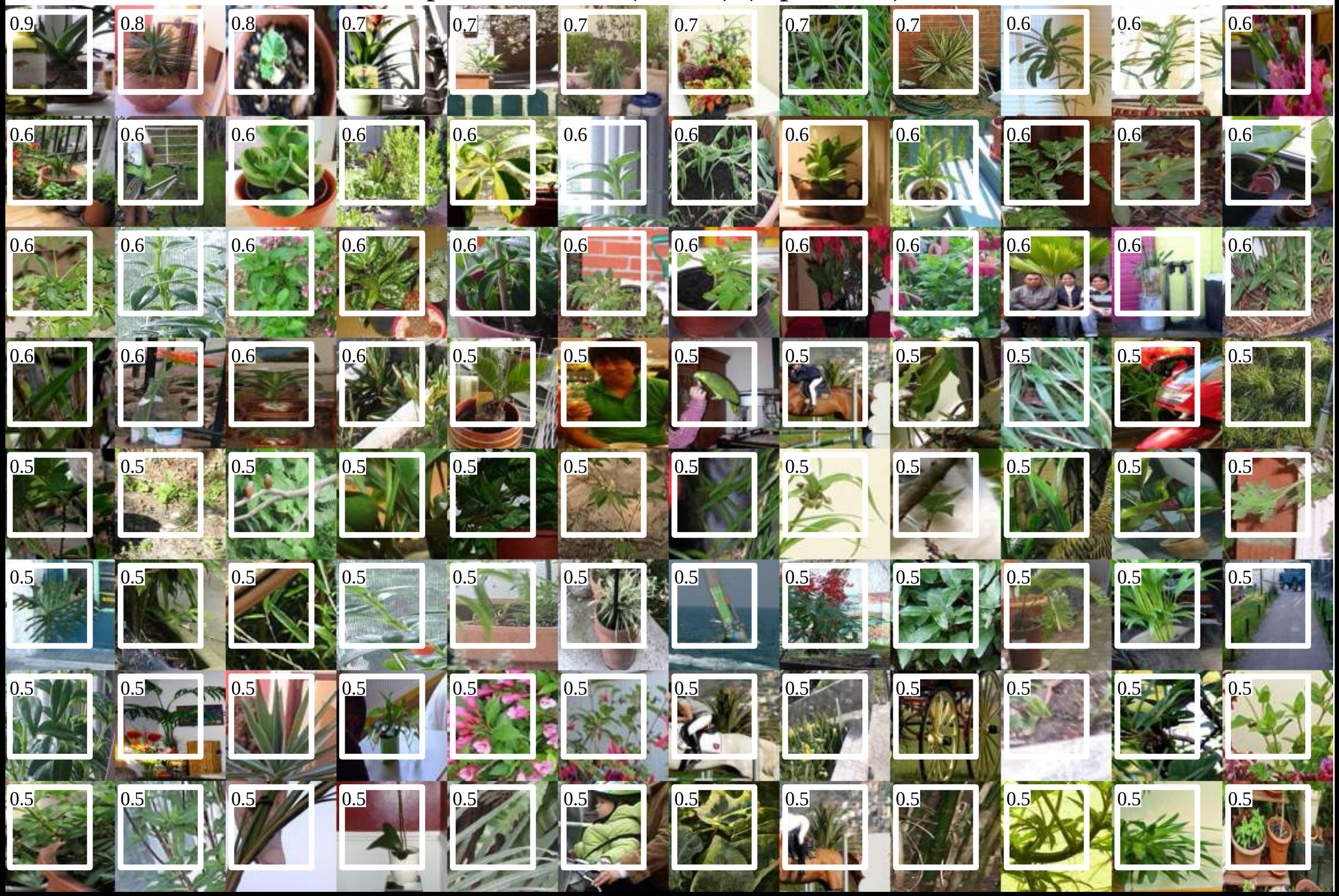
> $pool_5$: max-pooled output of last conv. layer

> 6 x 6 spatial structure (with 256 channels)

> receptive field size 163 x 163 (of 224 x 224)

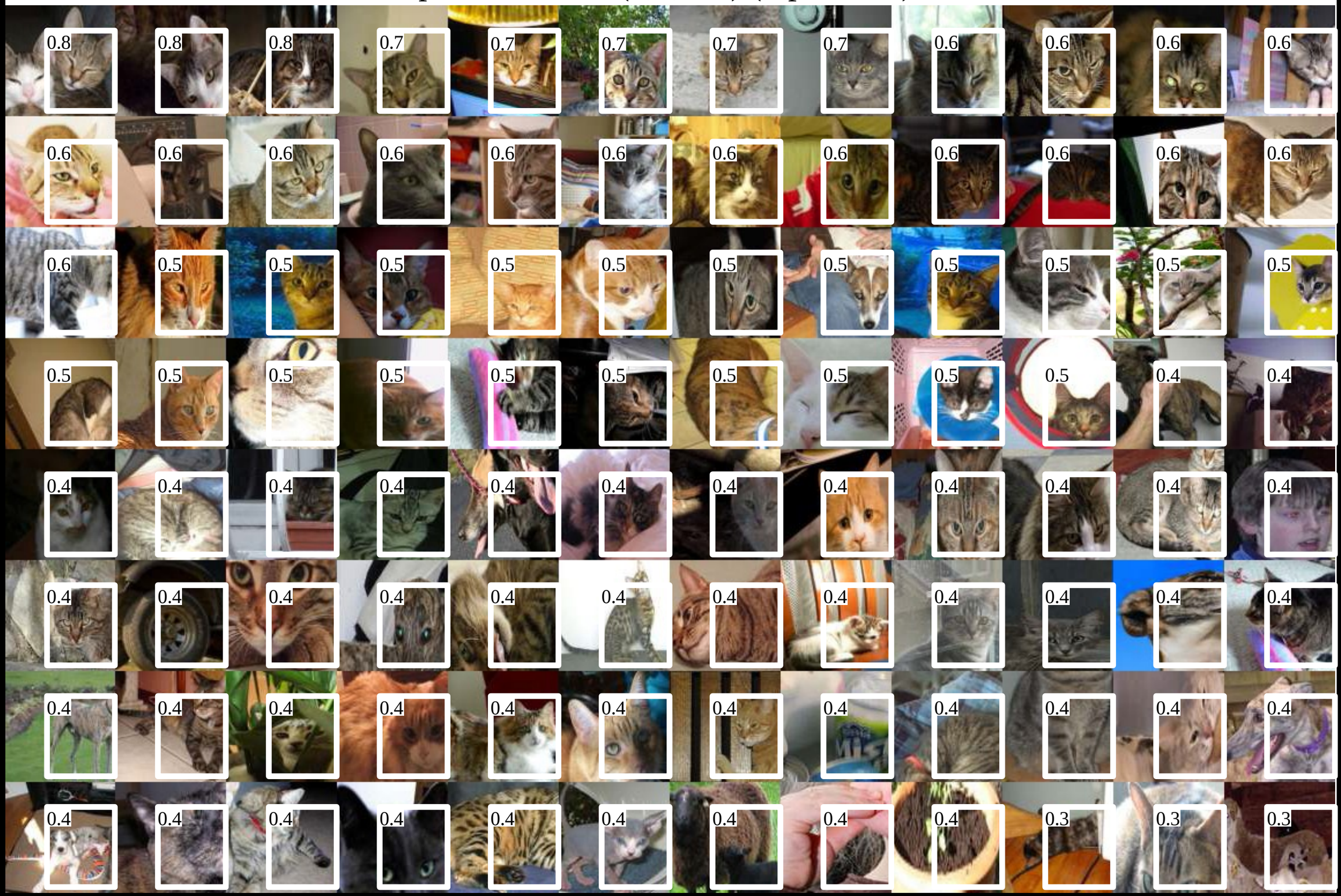


pool5 feature: (3,3,42) (top 1 − 96)

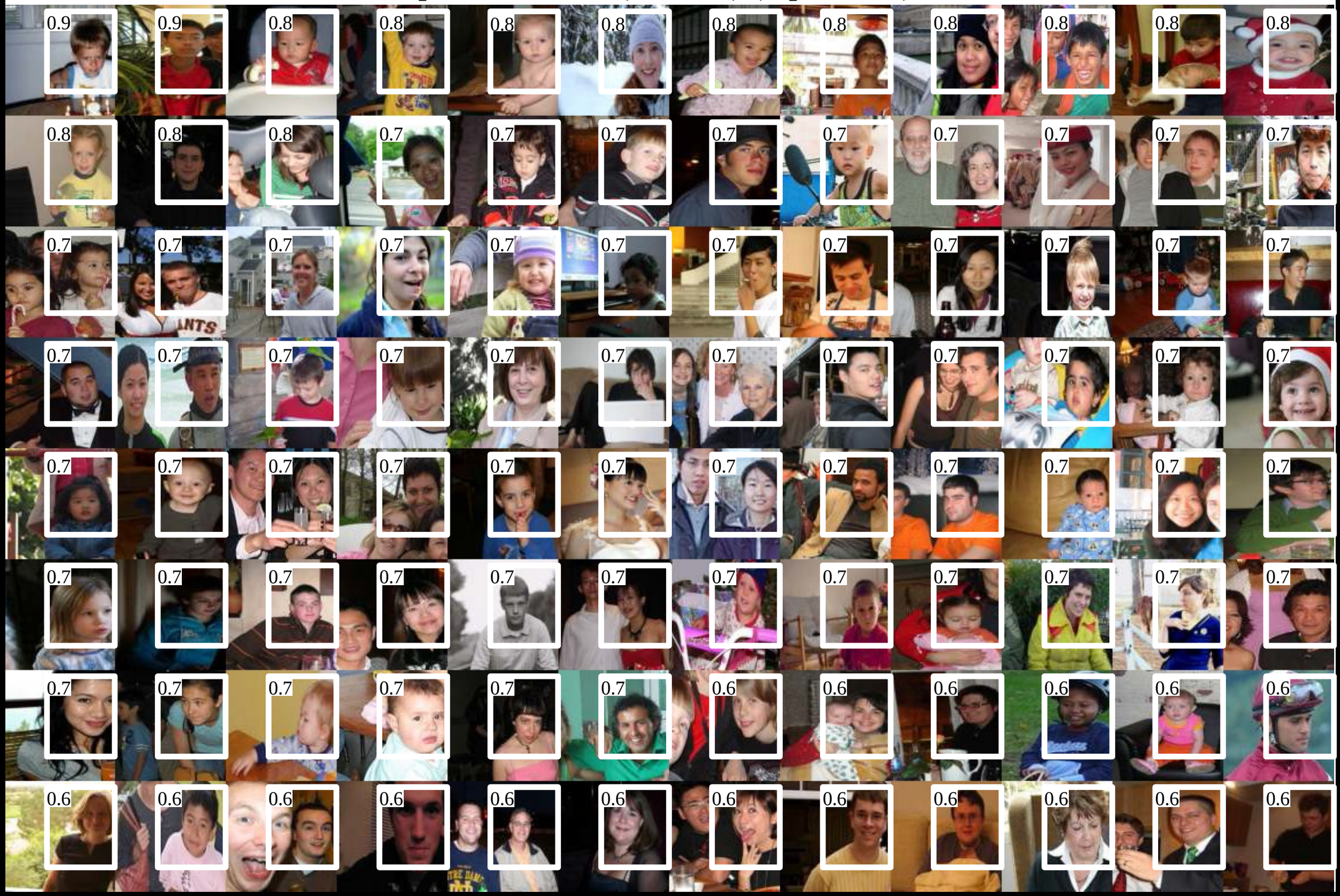


pool5 feature: (3,4,80) (top 1 − 96)

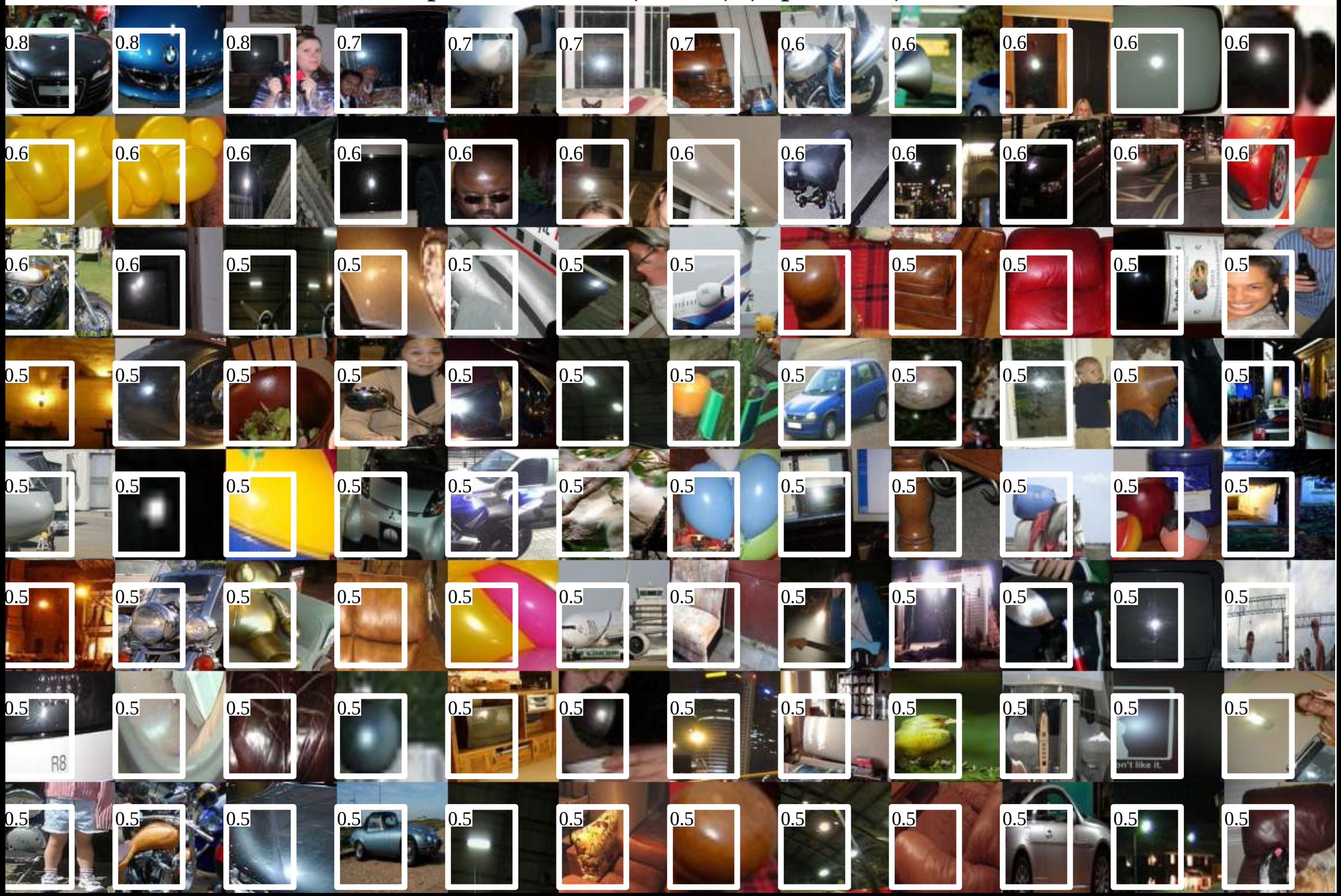
pool5 feature: (4,5,110) (top 1 − 96)



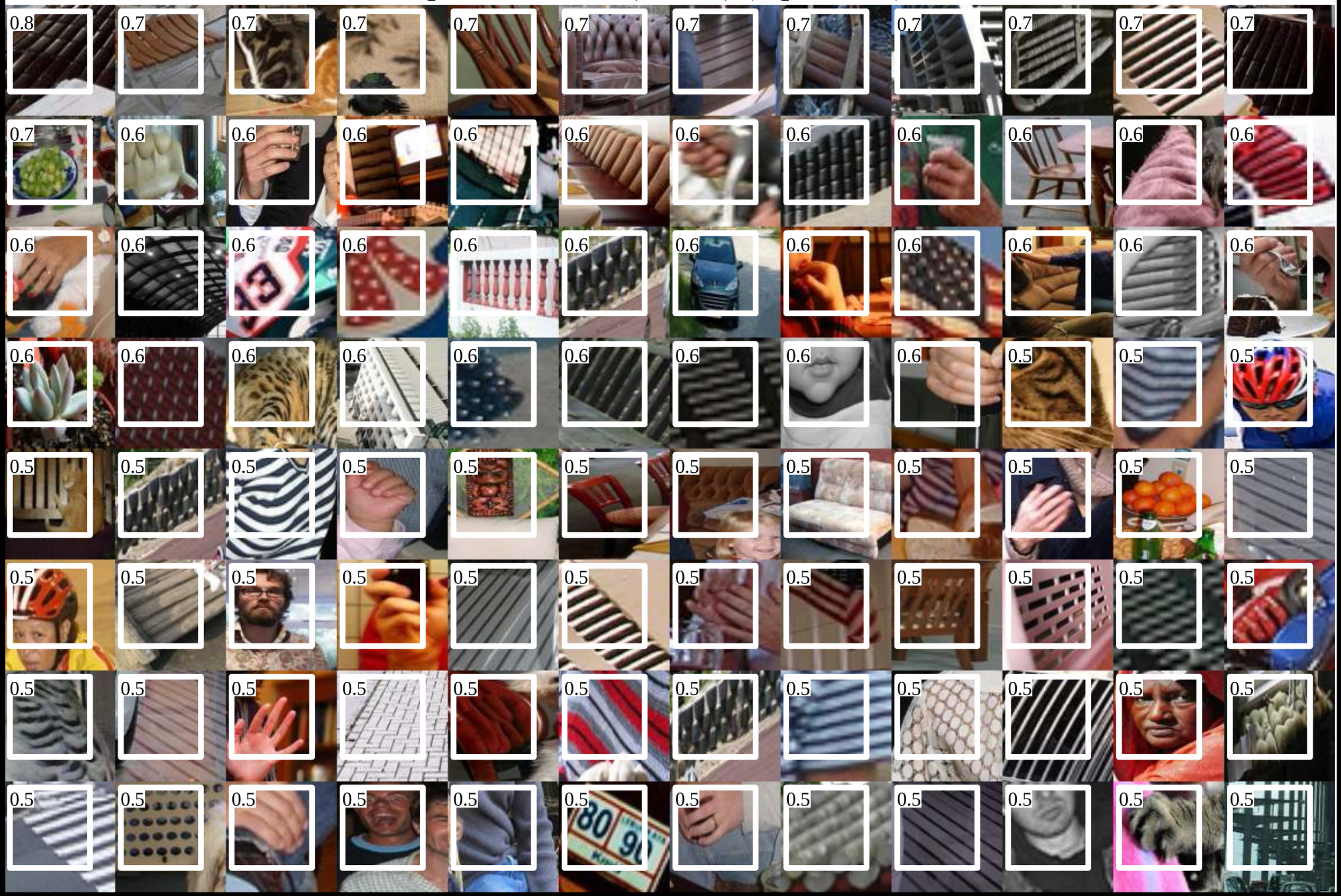
pool5 feature: (3,5,129) (top 1 − 96)



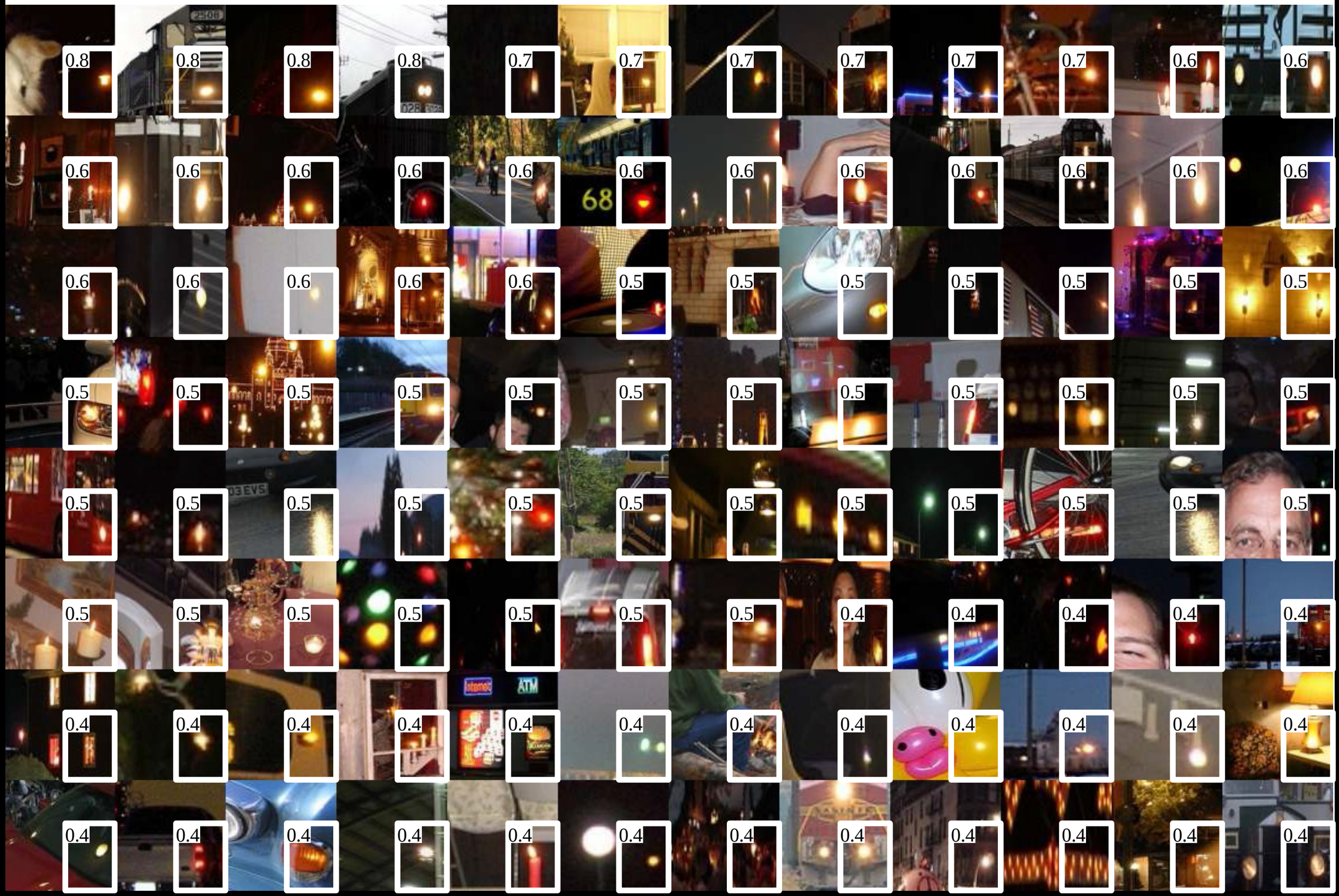
pool5 feature: (4,2,26) (top 1 − 96)



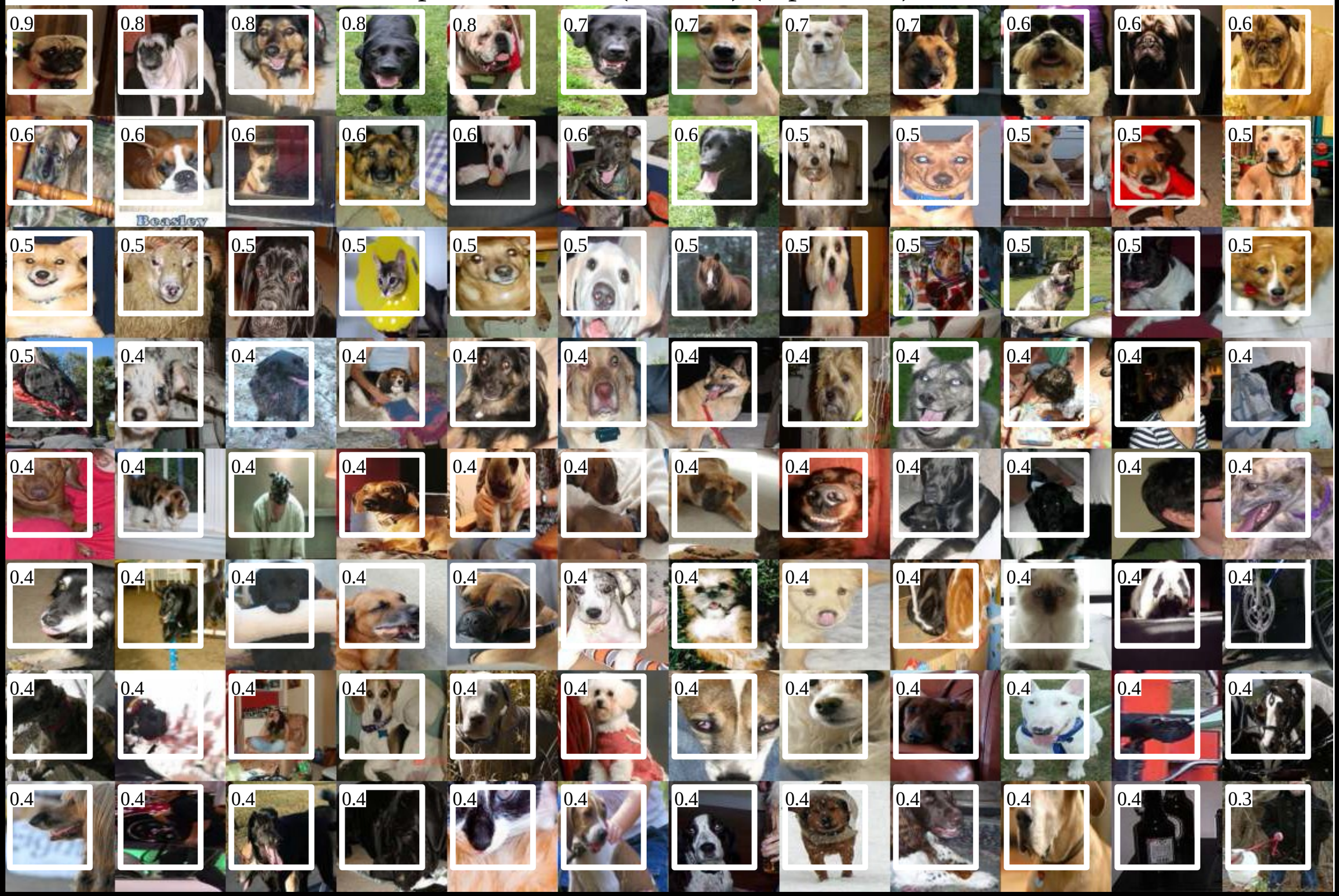
pool5 feature: (3,3,39) (top 1 − 96)



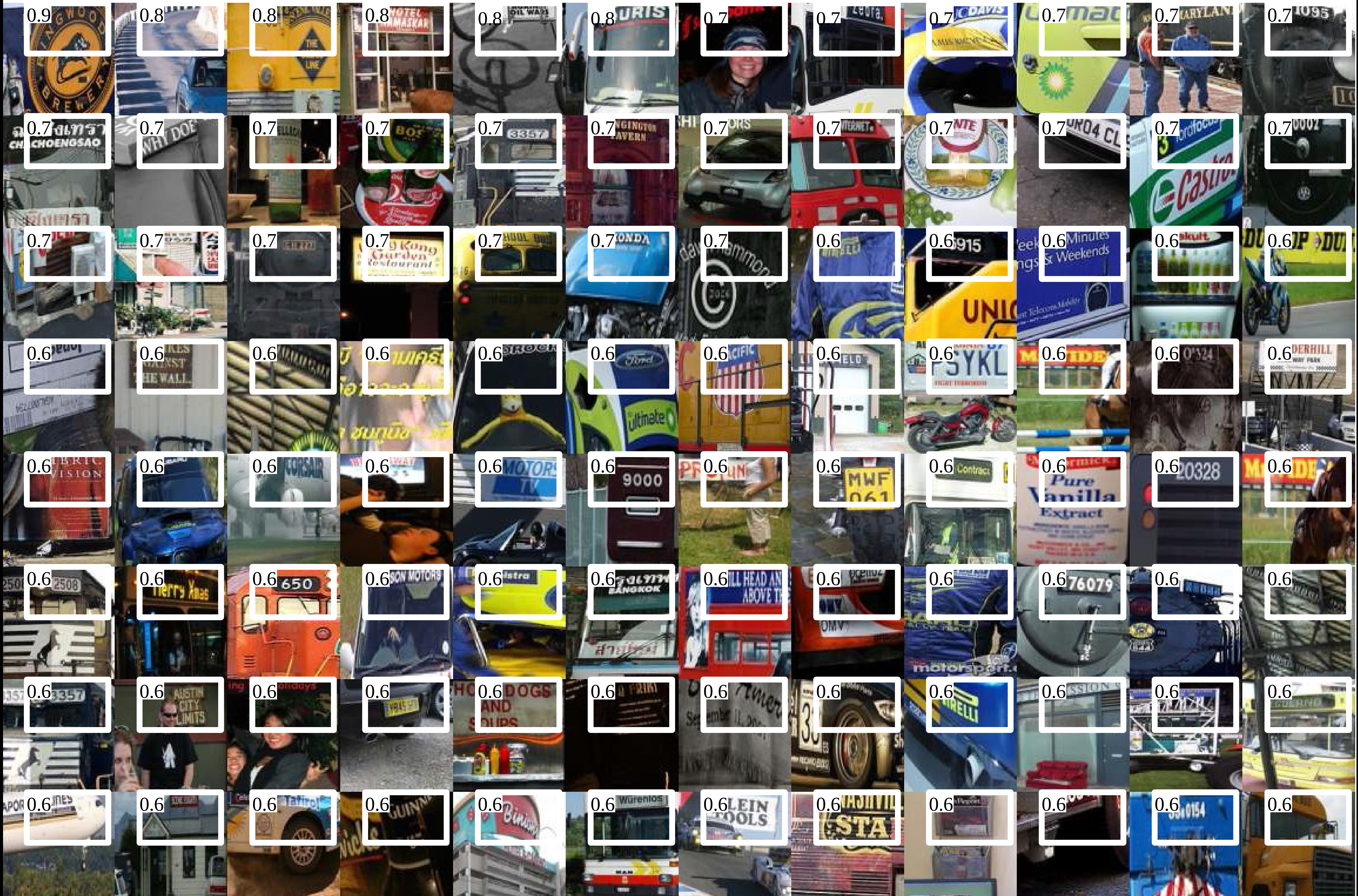
pool5 feature: (5,6,53) (top 1 − 96)



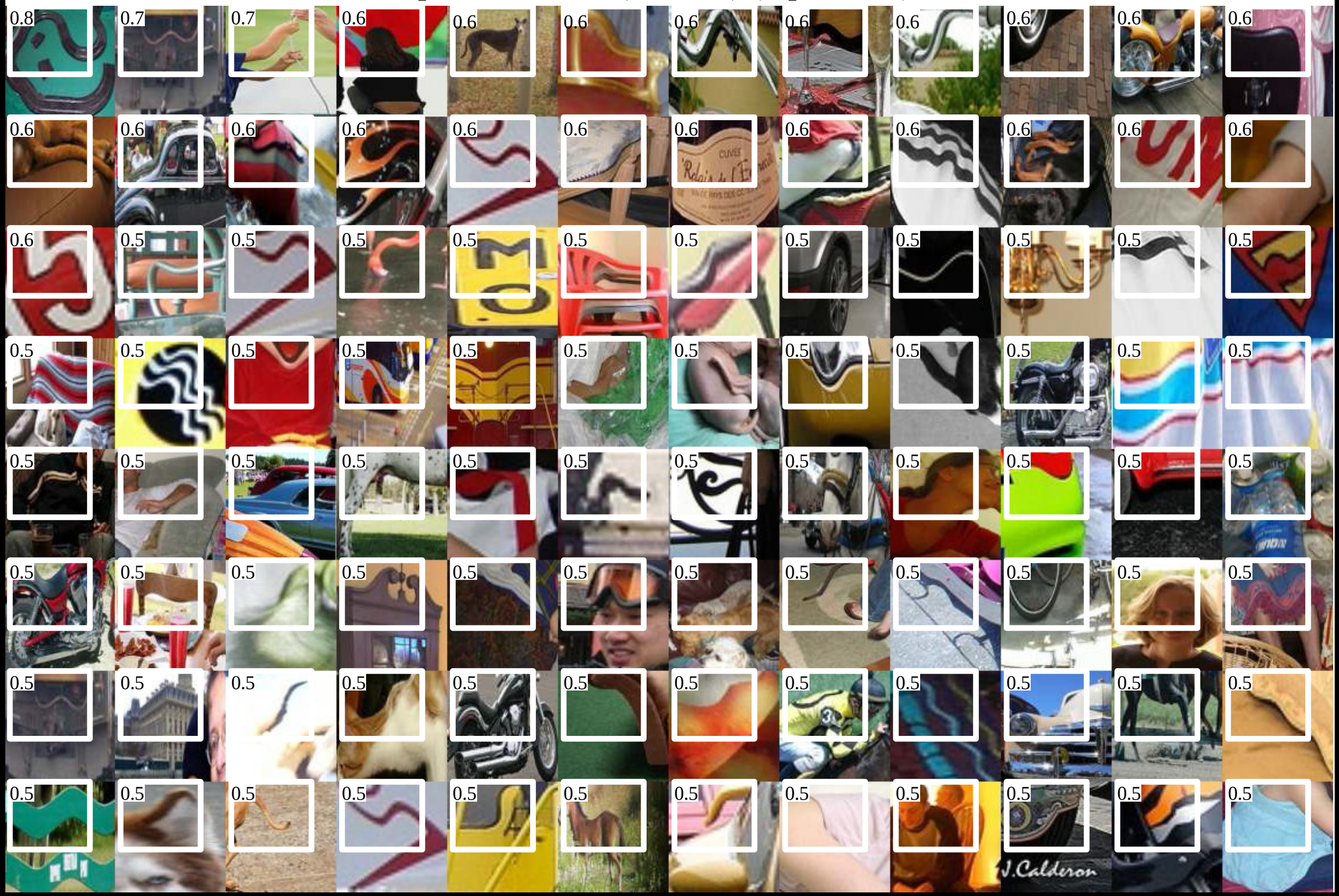
pool5 feature: (3,3,139) (top 1 − 96)



pool5 feature: (1,4,138) (top 1 − 96)



pool5 feature: (2,3,210) (top 1 − 96)

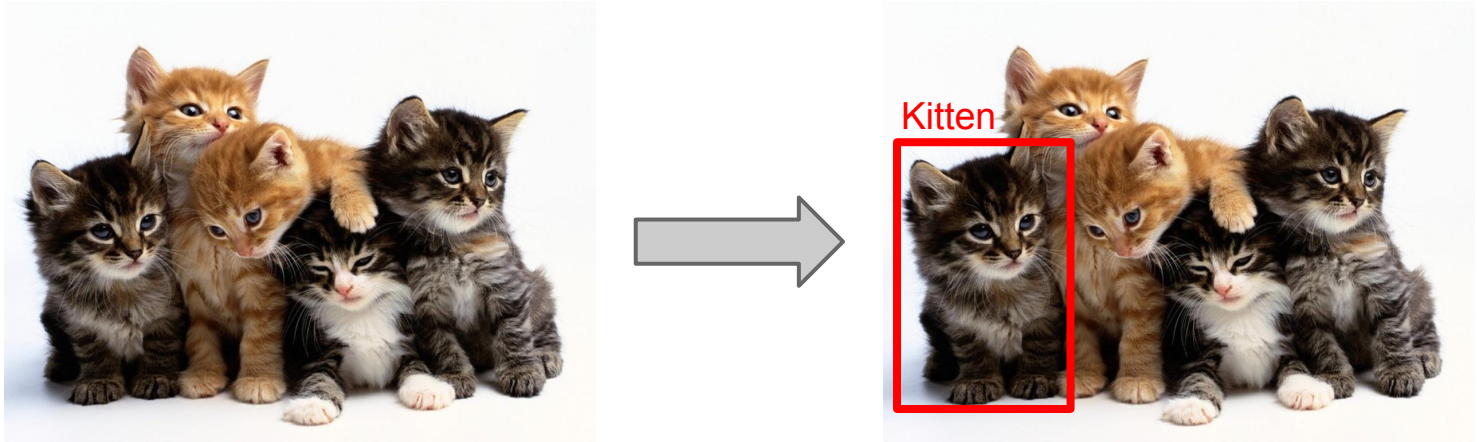


Selective Search

Uijlings, van de Sande, Gevers, Smeulders

Object Recognition

Goal:

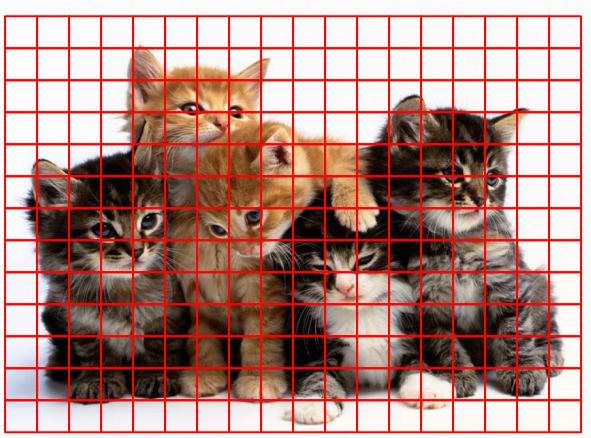


Problem: Where do we look in the image for the object?

One Solution

Idea: Exhaustively search for objects.

Problem: Extremely slow, must process tens of thousands of candidate objects.

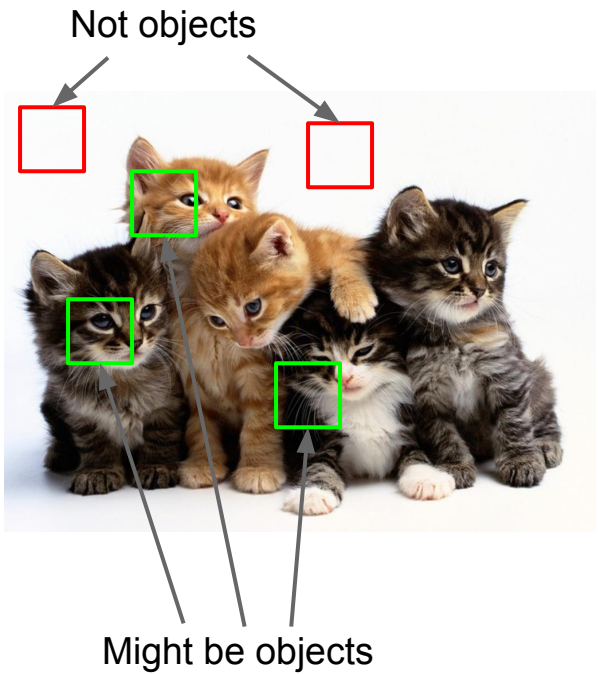


[N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection.” In CVPR, 2005.]

One Solution

Idea: Running a scanning **detector** is cheaper than running a recognizer, so do that first.

1. Exhaustively search for candidate objects with a generic detector.
2. Run recognition algorithm only on candidate objects.

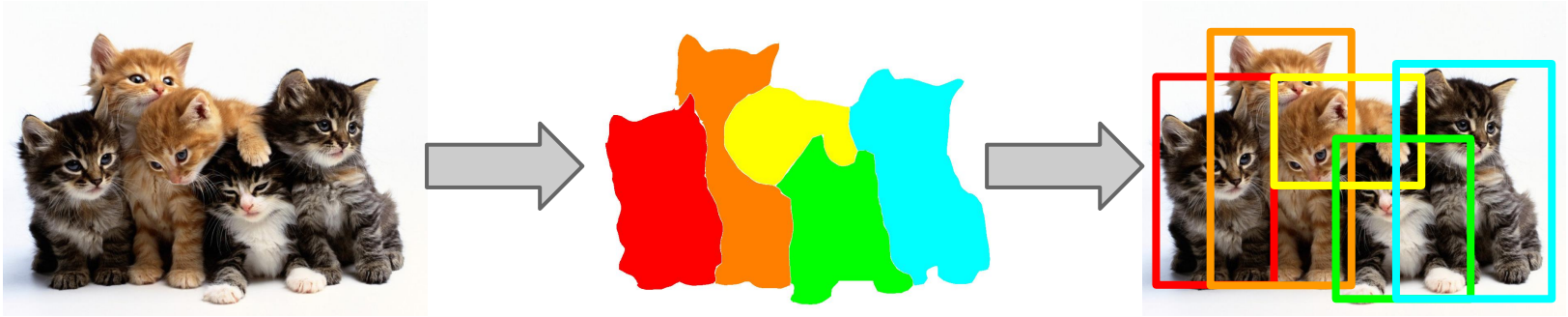


Problem: What about oddly-shaped objects? Will we need to scan with windows of many different shapes?

[B. Alexe, T. Deselaers, and V. Ferrari. “Measuring the objectness of image windows.” IEEE transactions on Pattern Analysis and Machine Intelligence, 2012.]

Segmentation

Idea: If we correctly segment the image before running object recognition, we can use our segmentations as candidate objects.



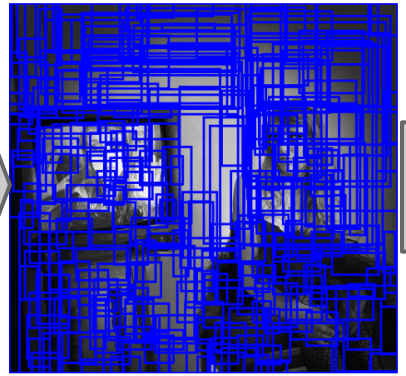
Advantages: Can be efficient, makes no assumptions about object sizes or shapes.

Approach

Original Image

Search

Key concept



Candidate Boxes

Object Recognition

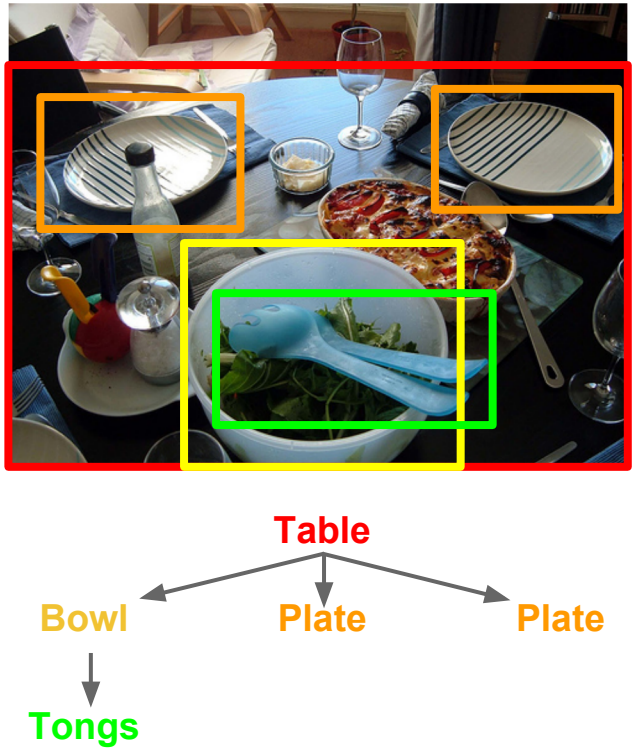
Final Detections

Hierarchical Image Representation

Images are actually 2D representations of a 3D world.

Objects can be on top of, behind, or parts of other objects.

We can encode this with an object/segment **hierarchy**.



Selective Search

Goals:

1. Detect objects at any scale.
    a. Hierarchical algorithms are good at this.
2. Consider multiple grouping criteria.
    a. Detect differences in color, texture, brightness, etc.
3. Be fast.

Idea: Use bottom-up grouping of image regions to generate a hierarchy of small to large regions.

Selective Search

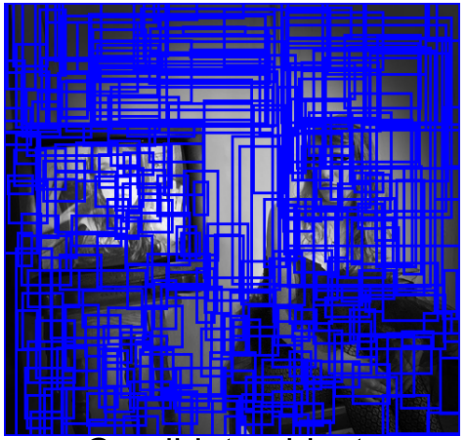
Step 1: Generate initial sub-segmentation

Goal: Generate many regions, each of which belongs to at most one object.

Using the method described by Felzenszwalb et al. from week 1 works well.

Input Image

Segmentation



Candidate objects

[P. F. Felzenszwalb and D. P. Huttenlocher. “Efficient Graph-Based Image Segmentation.” IJCV, 59:167–181, 2004.]

Selective Search

Step 2: Recursively combine similar regions into larger ones.

Greedy algorithm:

1. From set of regions, choose two that are most similar.
2. Combine them into a single, larger region.
3. Repeat until only one region remains.

This yields a hierarchy of successively larger regions, just like we want.

Selective Search

Step 2: Recursively combine similar regions into larger ones.

Input Image

Initial Segmentation

After some iterations

After more iterations

Selective Search

Step 3: Use the generated regions to produce candidate object locations.

Input Image

Similarity

What do we mean by “**similarity**”?

Goals:

1. Use multiple grouping criteria.
2. Lead to a balanced hierarchy of small to large objects.
3. Be efficient to compute: should be able to quickly combine measurements in two regions.

Similarity

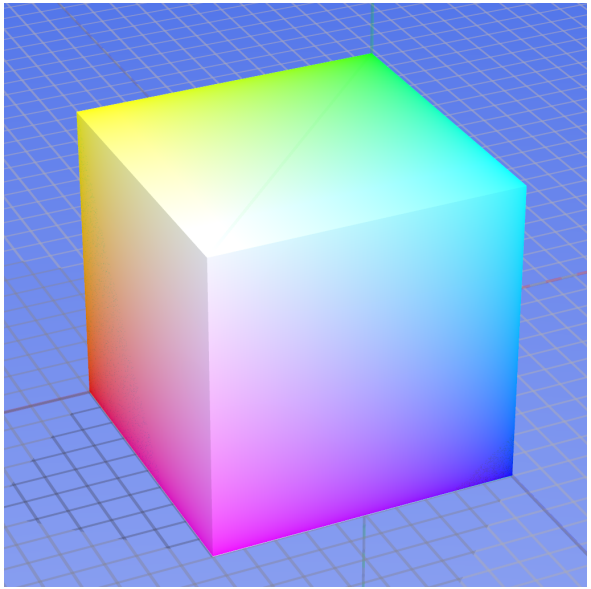
What do we mean by “**similarity**”?

Two-pronged approach:

1. Choose a **color space** that captures interesting things.
    a. Different color spaces have different invariants, and different responses to changes in color.
2. Choose a **similarity metric** for that space that captures everything we’re interested: color, texture, size, and shape.

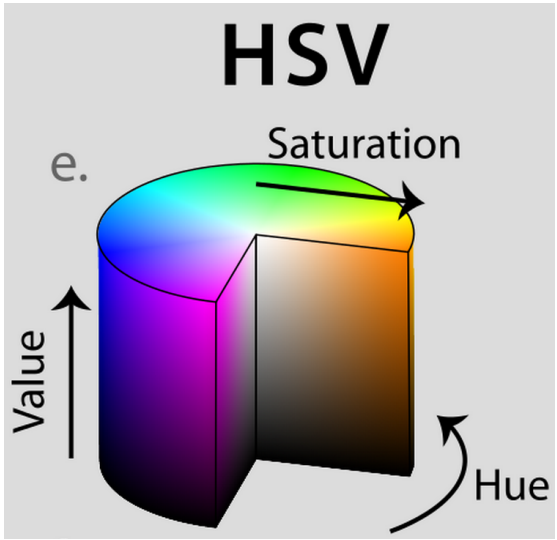
Similarity

RGB (red, green, blue) is a good baseline, but changes in illumination (shadows, light intensity) affect all three channels.



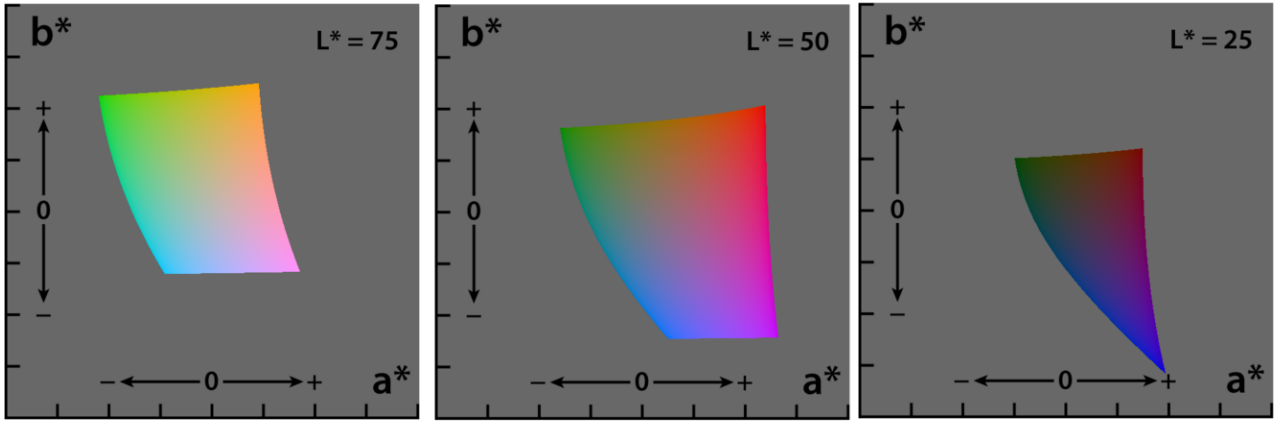
Similarity

HSV (hue, saturation, value) encodes color information in the hue channel, which is invariant to changes in lighting. Additionally, saturation is insensitive to shadows, and value is insensitive to brightness changes.



Similarity

Lab uses a lightness channel and two color channels (a and b). It's calibrated to be *perceptually uniform*. Like HSV, it's also somewhat invariant to changes in brightness and shadow.



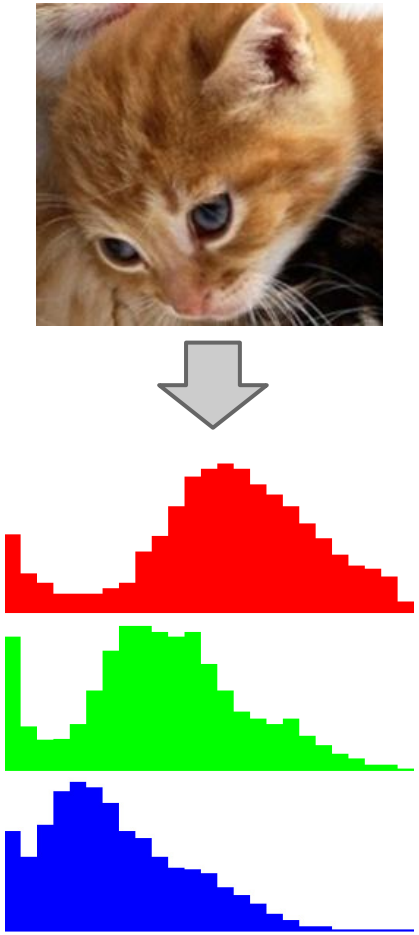
Similarity

Similarity Measures: Color Similarity

Create a color histogram *C* for each channel in region *r*.
In the paper, 25 bins were used, for 75 total dimensions.

We can measure similarity with histogram intersection:

$$s_{colour}(r_i, r_j) = \sum_{k=1}^{n} \min(c_i^k, c_j^k)$$

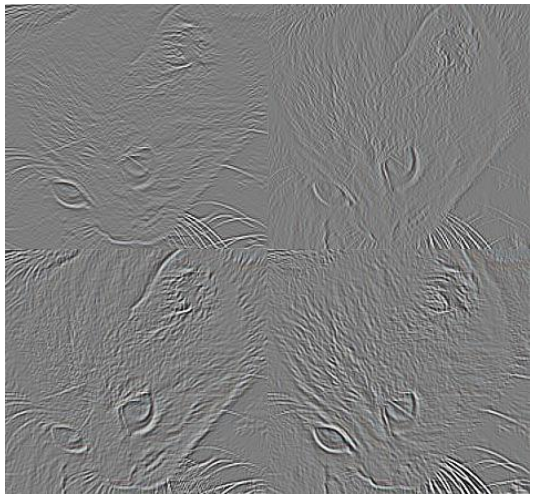


Similarity

Similarity Measures: Texture Similarity

Can measure textures with a HOG-like feature:

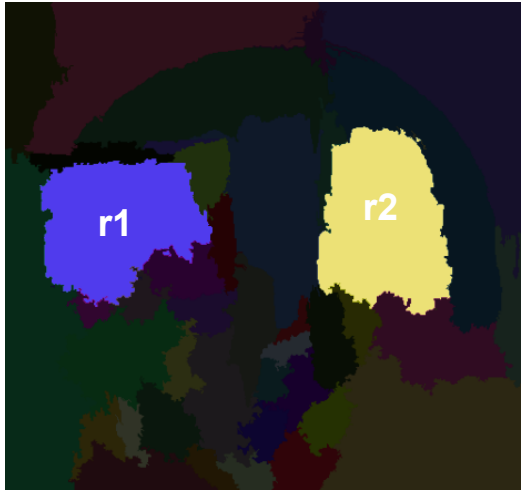
1. Extract gaussian derivatives of the image in 8 directions and for each channel.
2. Construct a 10-bin histogram for each, resulting in a 240-dimensional descriptor.



Similarity

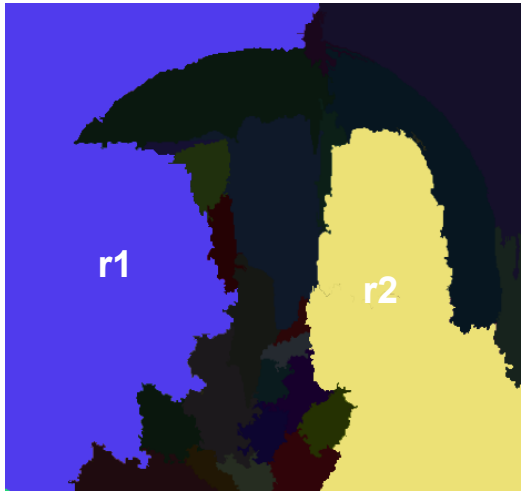
Similarity Measures: Size Similarity

We want small regions to merge into larger ones, to create a balanced hierarchy.



Solution: Add a size component to our similarity metric, that ensures small regions are more similar to each other.

$$s_{size}(r_i, r_j) = 1 - \frac{\text{size}(r_i) + \text{size}(\text{r}_\text{j})}{\text{size}(im)}$$

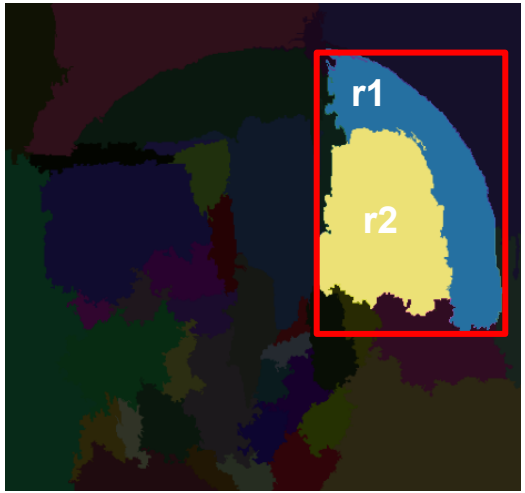


Similarity

Similarity Measures: Shape Compatibility

We also want our merged regions to be cohesive, so we can add a measure of how well two regions “fit together”.

$$fill(r_i, r_j) = 1 - \frac{\text{size}(BB_{ij}) - \text{size}(r_i) - \text{size}(r_i)}{\text{size}(im)}$$



Similarity

Final similarity metric:

We measure the similarity between two patches as a **linear combination** of the four given metrics:

$$\begin{aligned} s(r_i, r_j) &= a_1 s_{colour}(r_i, r_j) + a_2 s_{texture}(r_i, r_j) + \\ & \quad a_3 s_{size}(r_i, r_j) + a_4 s_{fill}(r_i, r_j), \end{aligned}$$

Then, we can create a diverse collection of region-merging strategies by considering different weighted combinations in different color spaces.

Evaluation

Measuring box quality:

A metric called **Average Best Overlap**:

$$\mathrm{ABO} = \frac{1}{|G^c|} \sum_{g_i^c \in G^c} \max_{l_j \in L} \mathrm{Overlap}(g_i^c, l_j)$$

Overlap between ground truth and best selected box.

Average of “best overlaps” across all images.

Segmentation Results

Note that HSV, Lab, and rgI do noticeably better than RGB.

Texture on its own performs worse than the color, size, and fill similarity metrics.

Similarities	MABO	# box
C	0.635	356
T	0.581	303
S	0.640	466
F	0.634	449
C+T	0.635	346
C+S	0.660	383
C+F	0.660	389
T+S	0.650	406
T+F	0.638	400
S+F	0.638	449
C+T+S	0.662	377
C+T+F	0.659	381
C+S+F	0.674	401
T+S+F	0.655	427
C+T+S+F	0.676	395

Colours	MABO	# box
HSV	0.693	463
I	0.670	399
RGB	0.676	395
rgI	0.693	362
Lab	0.690	328
H	0.644	322
rgb	0.647	207
C	0.615	125

Thresholds	MABO	# box
50	0.676	395
100	0.671	239
150	0.668	168
250	0.647	102
500	0.585	46
1000	0.477	19

The best similarity measure overall uses all four metrics.

Segmentation Results

Combining strategies improves performance even more:

Version	Diversification Strategies	MABO	# win	# strategies	time (s)
Single Strategy	HSV C+T+S+F $k = 100$	0.693	362	1	0.71
Selective Search Fast	HSV, Lab C+T+S+F, T+S+F $k = 50, 100$	0.799	2147	8	3.79
Selective Search Quality	HSV, Lab, rgI, H, I C+T+S+F, T+S+F, F, S $k = 50, 100, 150, 300$	0.878	10,108	80	17.15

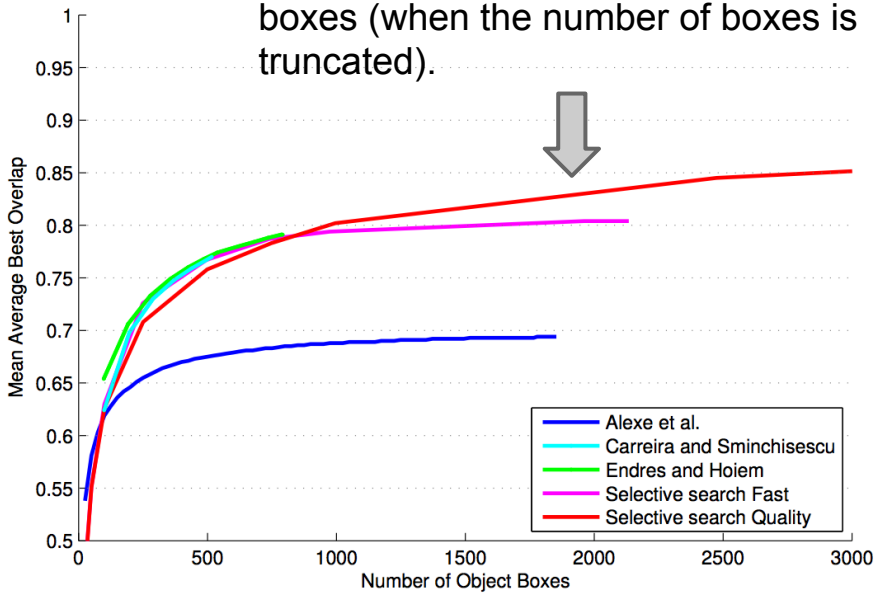
Using an ensemble greatly improves performance, at the cost of runtime (more candidate windows to check).

Segmentation Results

method	recall	MABO	# windows
Arbelaez *et al.* [3]	0.752	0.649 ± 0.193	418
Alexe *et al.* [2]	0.944	0.694 ± 0.111	1,853
Harzallah *et al.* [16]	0.830	-	200 per class
Carreira and Sminchisescu [4]	0.879	0.770 ± 0.084	517
Endres and Hoiem [9]	0.912	0.791 ± 0.082	790
Felzenszwalb *et al.* [12]	0.933	0.829 ± 0.052	100,352 per class
Vedaldi *et al.* [34]	0.940	-	10,000 per class
Single Strategy	0.840	0.690 ± 0.171	289
Selective search "Fast"	0.980	0.804 ± 0.046	2,134
Selective search "Quality"	0.991	0.879 ± 0.039	10,097

Excellent performance with fewer boxes than previous algorithms, which speeds up recognition.

"Quality" can outperform "Fast" even when returning the same number of boxes (when the number of boxes is truncated).



Segmentation Results

(a) Bike: 0.863

(b) Cow: 0.874

(c) Chair: 0.884

(d) Person: 0.882