Recap

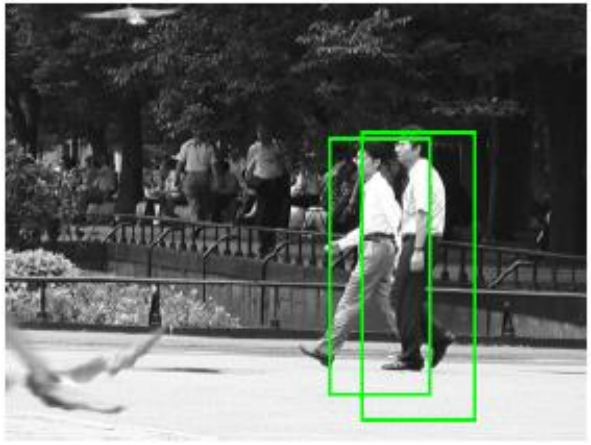
- Segmentation vs Boundary Detection vs semantic segmentation / scene parsing
- Why boundaries / Grouping?
- Recap: Canny Edge Detection
- The Berkeley Segmentation Data Set
- pB boundary detector ~2001
- Sketch Tokens 2013

Today: Scene Parsing / Semantic Segmentation

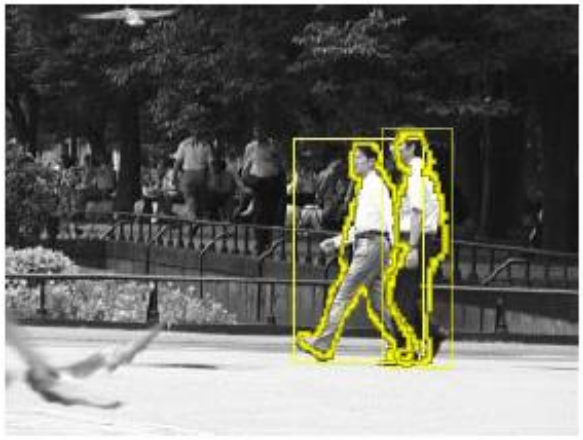
- Label every pixel of an image with a category label (usually with the help of contextual reasoning).
- Well known example: TextonBoost
- Detailed look at the “non parametric” approach of Tighe and Lazebnik

Object Recognition and Segmentation are Coupled

No Segmentation



Approximate Segmentation



Good Segmentation

Images from
[Leibe *et al.* 2005]

The Three Approaches

- Segment → Detect
- Detect → Segment
- Segment ←→ Detect

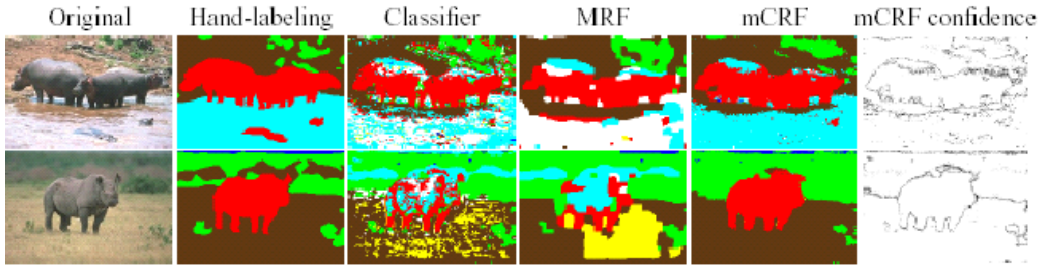
Segment first and ask questions later.

- Reduces possible locations for objects
- Allows use of shape information and makes long-range cues more effective
- But what if segmentation is wrong?

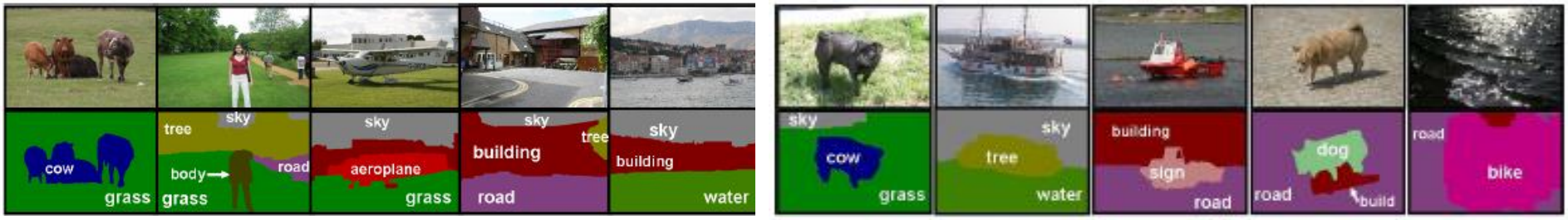
[Duygulu *et al* ECCV 2002]

Object recognition + data-driven smoothing

- Object recognition drives segmentation
- Segmentation gives little back



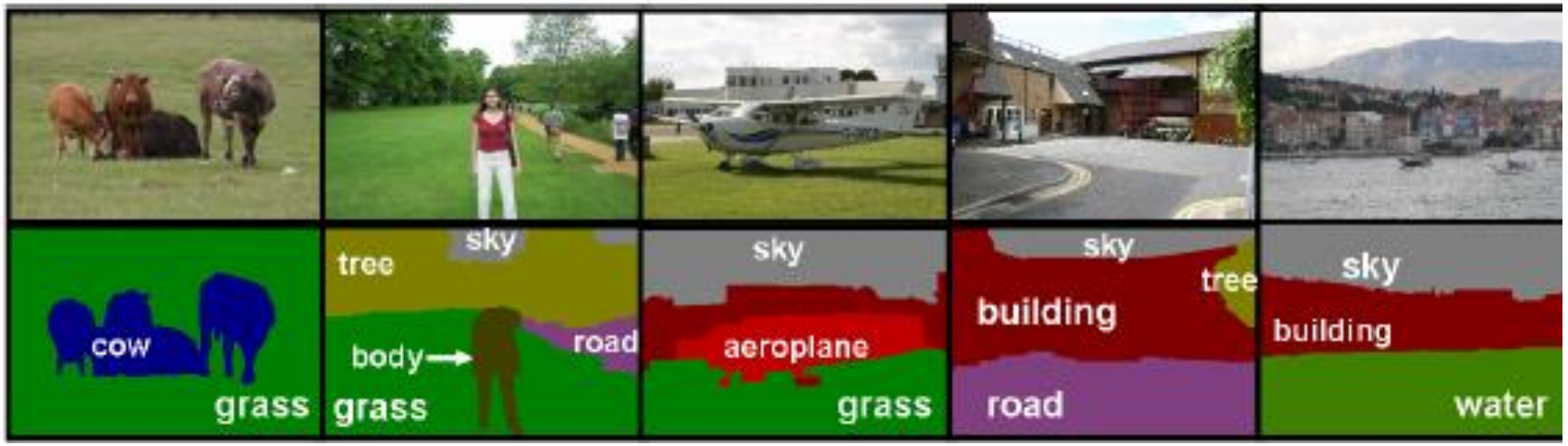
He *et al.* 2004



TextonBoost

TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation

J. Shotton ; University of Cambridge
J. Jinn, C. Rother, A. Criminisi ; MSR Cambridge



Slides modified from Derek Hoiem

The Ideas in TextonBoost

- Textons from Universal Visual Dictionary paper [Winn Criminisi Minka ICCV 2005]
- Color models and GC from “Foreground Extraction using Graph Cuts” [Rother Kolmogorov Blake SG 2004]
- Boosting + Integral Image from Viola-Jones
- Joint Boosting from [Torralba Murphy Freeman CVPR 2004]

What's good about this paper

- Provides recognition + segmentation for many classes (for the time it was published)

- Combines several good ideas

- Very thorough evaluation

TextonBoost Overview

$$\log P(\mathbf{c}|\mathbf{x},\boldsymbol{\theta}) = \sum_i \overbrace{\psi_i(c_i,\mathbf{x};\boldsymbol{\theta}_\psi)}^{\text{shape-texture}} + \overbrace{\pi(c_i,\mathbf{x}_i;\boldsymbol{\theta}_\pi)}^{\text{color}} + \overbrace{\lambda(c_i,i;\boldsymbol{\theta}_\lambda)}^{\text{location}}$$

$$+ \sum_{(i,j)\in\mathcal{E}} \overbrace{\phi(c_i,c_j,\mathbf{g}_{ij}(\mathbf{x});\boldsymbol{\theta}_\phi)}^{\text{edge}} - \log Z(\boldsymbol{\theta},\mathbf{x})$$

Shape-texture: localized textons

$\psi_i(c_i,\mathbf{x};\boldsymbol{\theta}_\psi) = \log \widetilde{P}_i(c_i|\mathbf{x})$

Color: mixture of Gaussians

$P(x|c) = \sum_k P(k|c)\mathcal{N}(x \mid \bar{x}_k, \Sigma_k) \qquad \pi(c_i,x_i;\boldsymbol{\theta}_\pi) = \log \sum_k \boldsymbol{\theta}_\pi(c_i,k)P(k|x_i)$

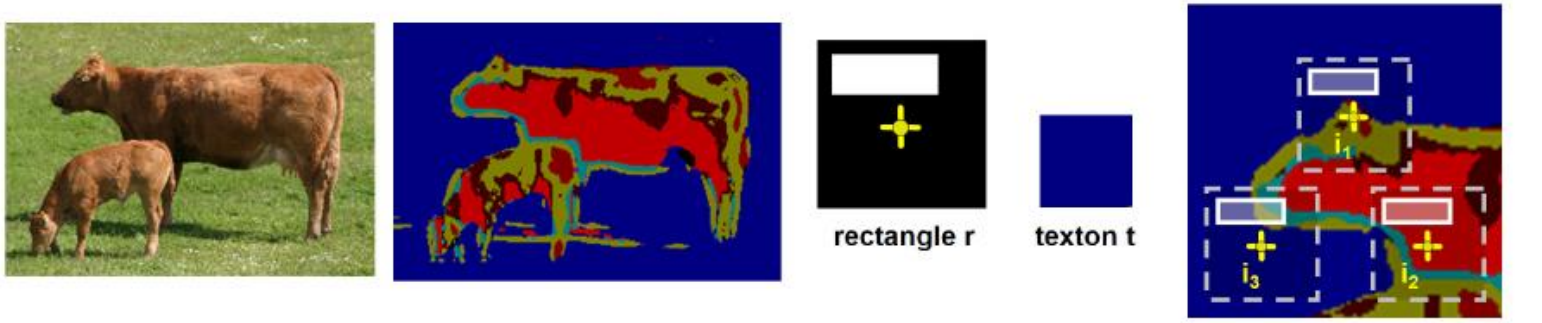
Location: normalized x-y coordinates

$\lambda_i(c_i,i;\boldsymbol{\theta}_\lambda) = \log \boldsymbol{\theta}_\lambda(c_i,\hat{i})$

Edges: contrast-sensitive Pott's model

$\phi(c_i,c_j,\mathbf{g}_{ij}(\mathbf{x});\boldsymbol{\theta}_\phi) = -\boldsymbol{\theta}_\phi^T \mathbf{g}_{ij}(\mathbf{x})\delta(c_i \neq c_j) \qquad \mathbf{g}_{ij} = [\exp(-\beta\|x_i - x_j\|^2), 1]^T$

Texture-Shape

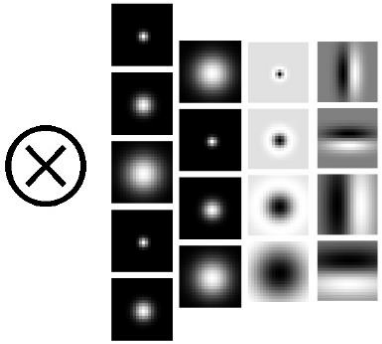


(a) Input image (b) Texton map (c) Feature pair = (r,t) (d) Superimposed rectangles

- 17 filters (oriented gaus/lap + dots)
- Cluster responses to form textons
- Count textons within white box (relative to position i)
- Feature = texton + rectangle

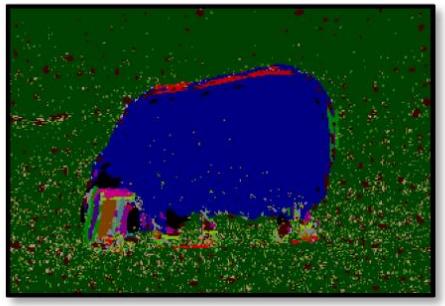
Texton Visualization

input image



filter bank

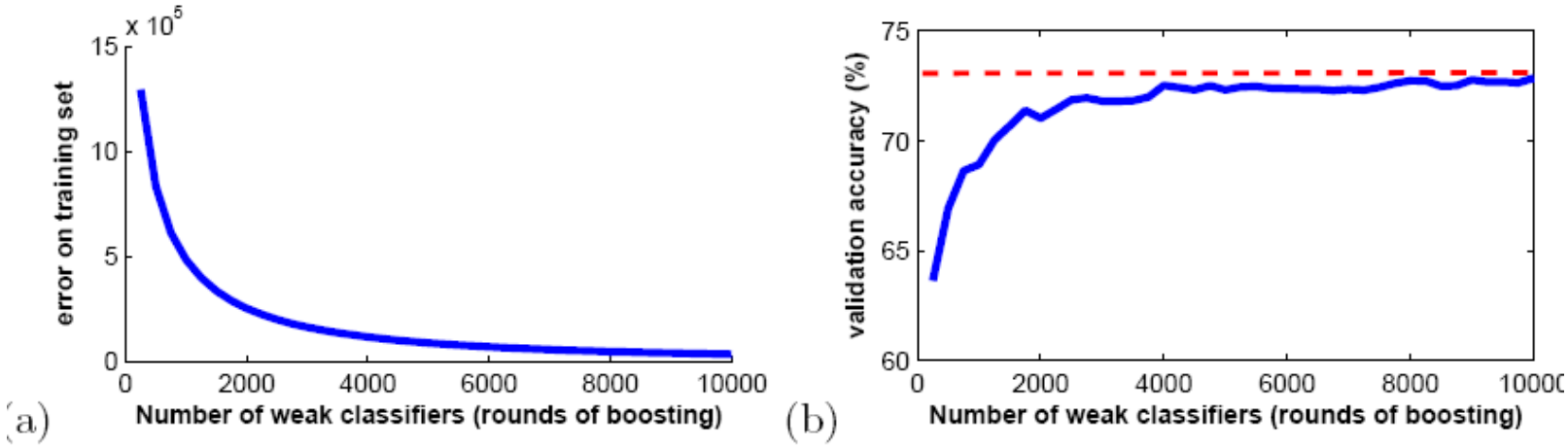
clustering and assignment



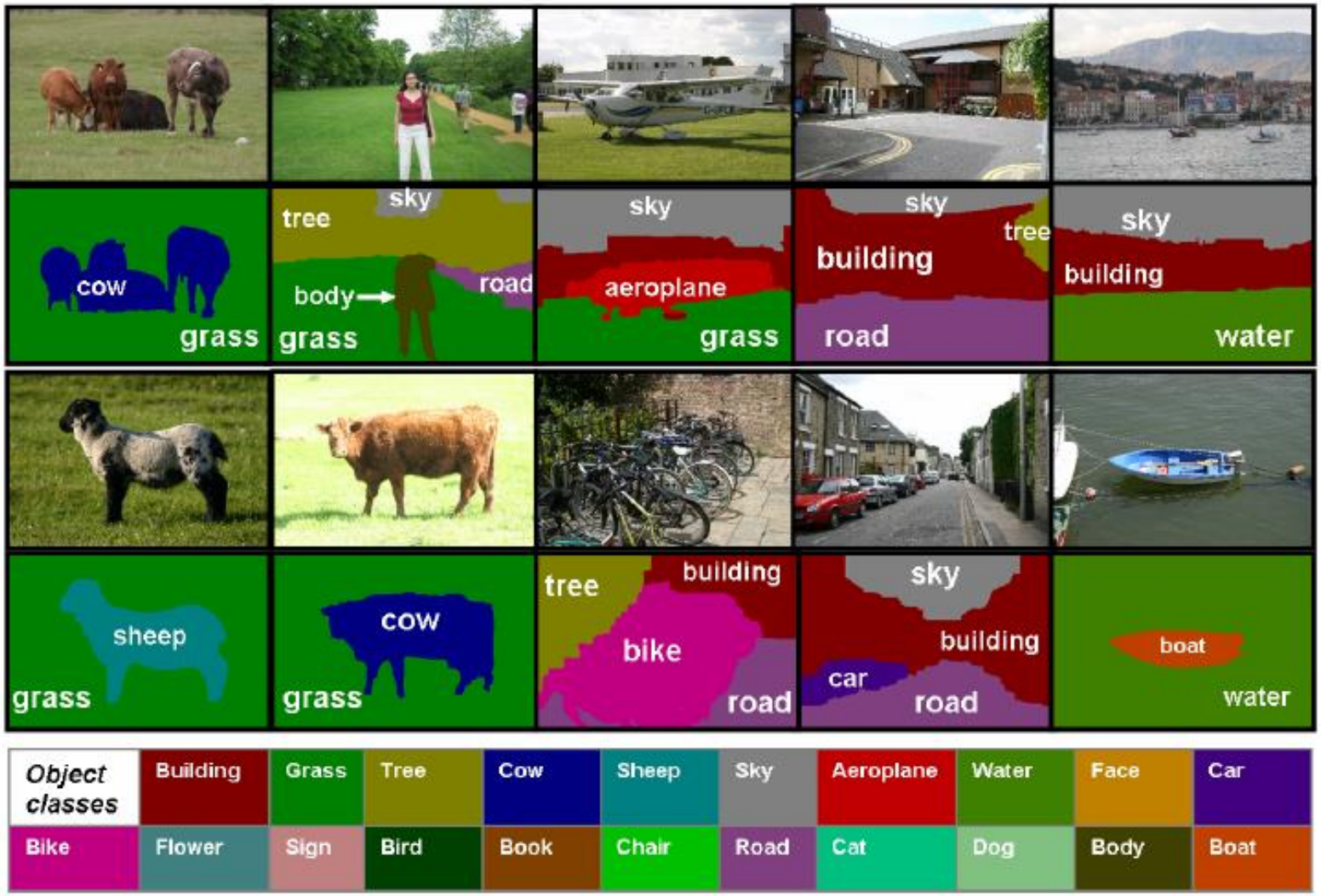
texton map
(colors ⇔ texton indices)

Results on Boosted Textons

- Boosted shape-textons in isolation
  - Training time: 42 hrs for 5000 rounds on 21-class training set of 276 images



Qualitative (Good) Results



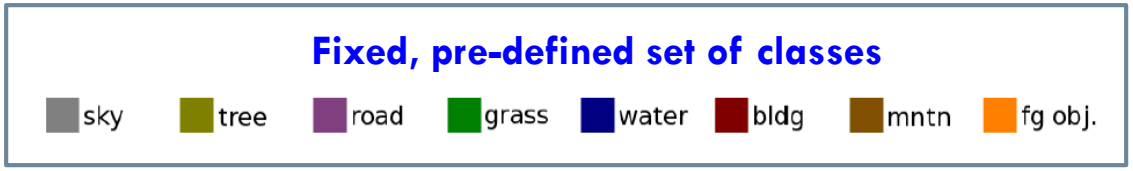
Qualitative (Bad) Results

- But notice good segmentation, even with bad labeling

Quantitative Results

True class \ Inferred class	building	grass	tree	cow	sheep	sky	aeroplane	water	face	car	bike	flower	sign	bird	book	chair	road	cat	dog	body	boat
building	**61.6**	4.7	9.7	0.3		2.5	0.6	1.3	2.0	2.6	2.1		0.6	0.2	4.8		6.3	0.4		0.5	
grass	0.3	**97.6**	0.5								0.1									1.3	
tree	1.2	4.4	**86.3**	0.5		2.9	1.4	1.9	0.8	0.1							0.1		0.2	0.1	
cow		30.9	0.7	**58.3**				0.9	0.4			0.4			4.2					4.1	
sheep	16.5	25.5	4.8	1.9	**50.4**									0.6			0.2				
sky	3.4	0.2	1.1			**82.6**		7.5									5.2				
aeroplane	21.5	7.2				3.0	**59.6**	8.5													
water	8.7	7.5	1.5	0.2		4.5		**52.9**		0.7	4.9			0.2	4.2		14.1	0.4			
face	4.1		1.1						**73.5**	7.1					8.4			0.4	0.2	5.2	
car	10.1		1.7							**62.5**	3.8		5.9	0.2			15.7				
bike	9.3		1.3							1.0	**74.5**		2.5			3.9	5.9		1.6		
flower		6.6	19.3	3.0								**62.8**			7.3		1.0				
sign	31.5	0.2	11.5	2.1		0.5		6.0		1.5		2.5	**35.1**		3.6	2.7	0.8	0.3		1.8	
bird	16.9	18.4	9.8	6.3	8.9	1.8		9.4						**19.4**			4.6	4.5			
book	2.6		0.6						0.4			2.0			**91.9**					2.4	
chair	20.6	24.8	9.6	18.2		0.2					3.7				1.9	**15.4**	4.5		1.1		
road	5.0	1.1	0.7					3.4	0.3	0.7	0.6		0.1	0.1	1.1		**86.0**			0.7	
cat	5.0		1.1	8.9				0.2		2.0					0.6		28.4	**53.6**	0.2		
dog	29.0	2.2	12.9	7.1				9.7							8.1		11.7		**19.2**		
body	4.6	2.8	2.0	2.1	1.3	0.2			6.0	1.1					9.9		1.7	4.0	2.1	**62.1**	
boat	25.1		11.5			3.8		30.6		2.0	8.6		6.4	5.1			0.3				**6.6**

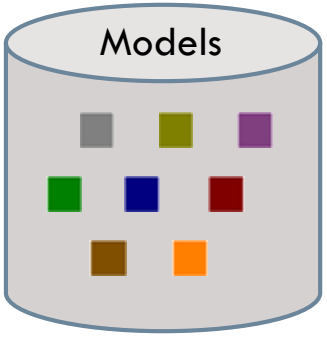
Closed-universe recognition



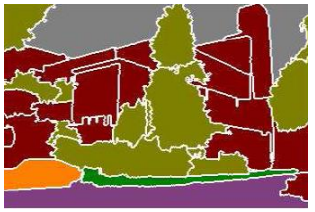
Test image

Learning

(offline)



Inference



Output

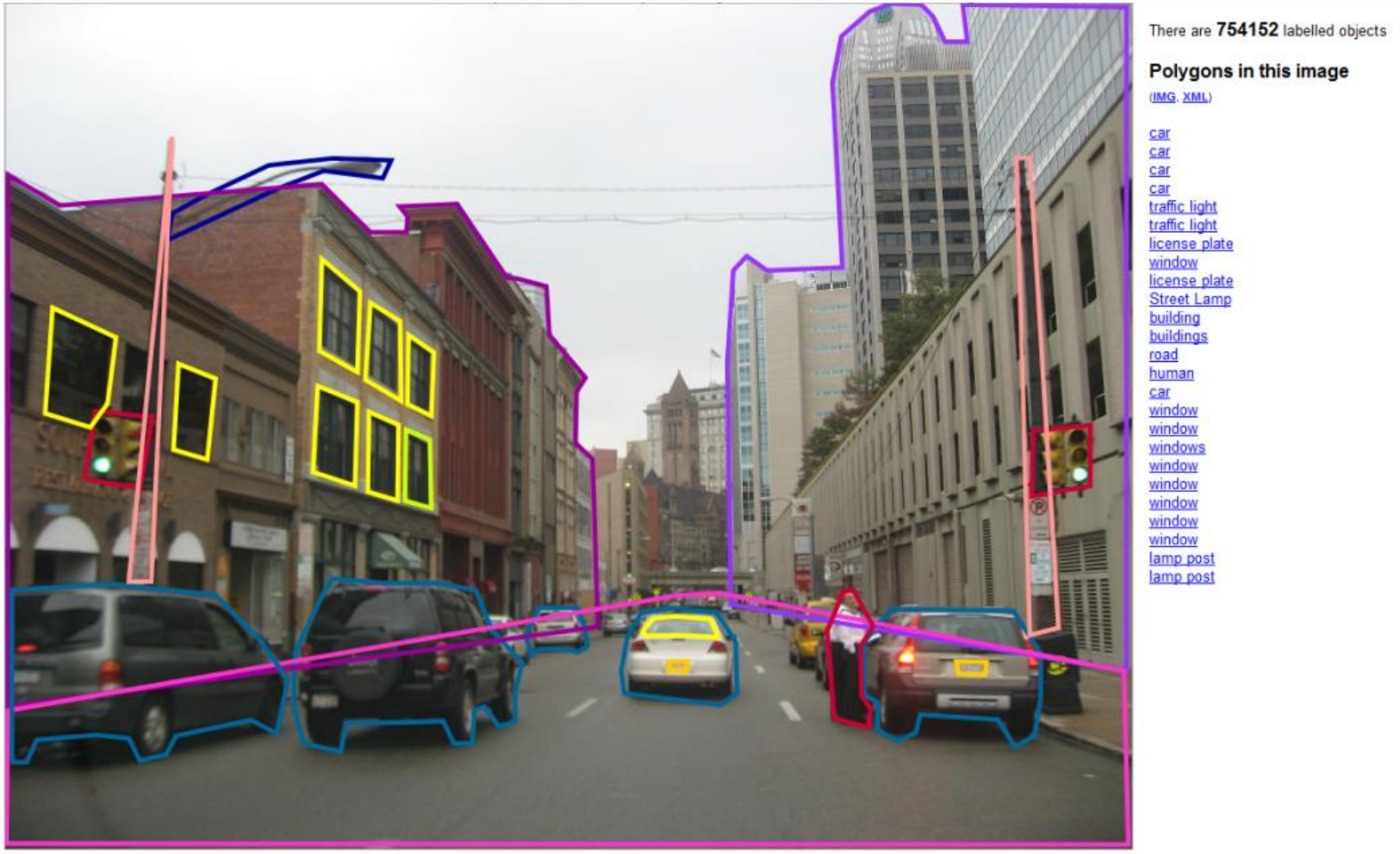
Closed-universe datasets

- Small amount of data
- Static datasets
- Limited variation
- Full annotation

Open-universe datasets

- Large amount of data
- Evolving datasets
- Wide variation
- Incomplete annotation

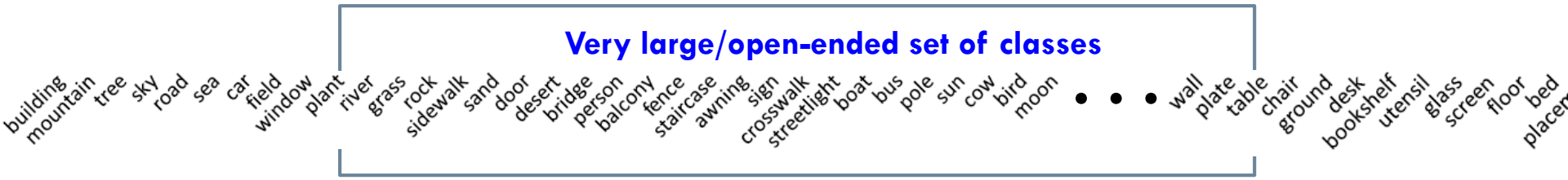
Open-universe recognition



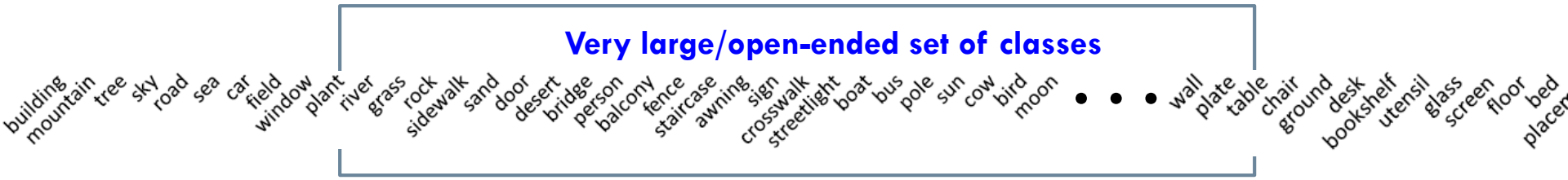
Evolving training set

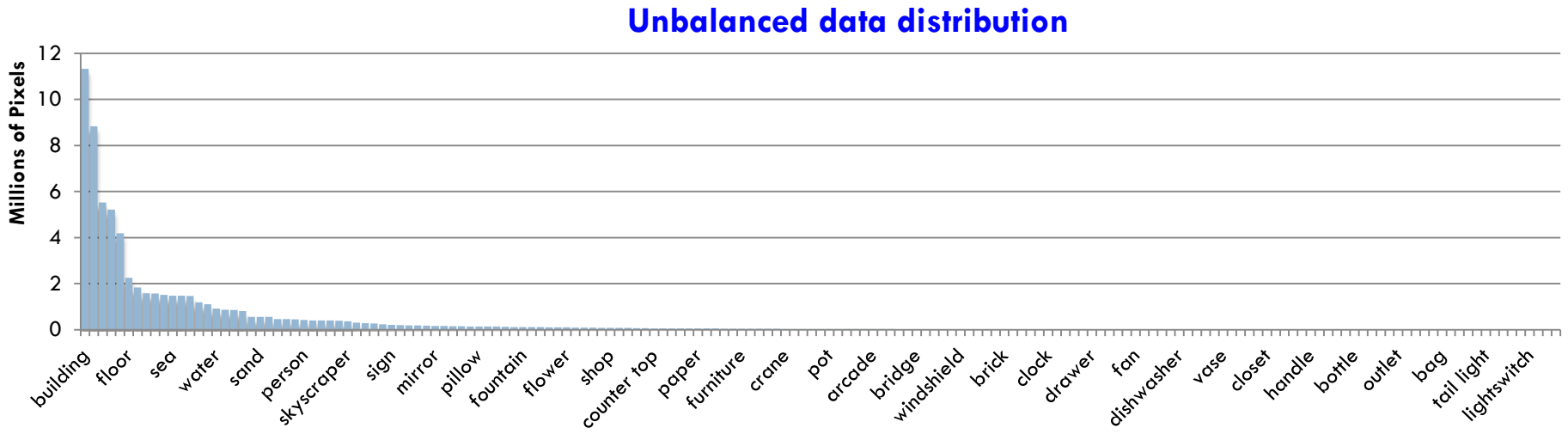
http://labelme.csail.mit.edu/

Open-universe recognition

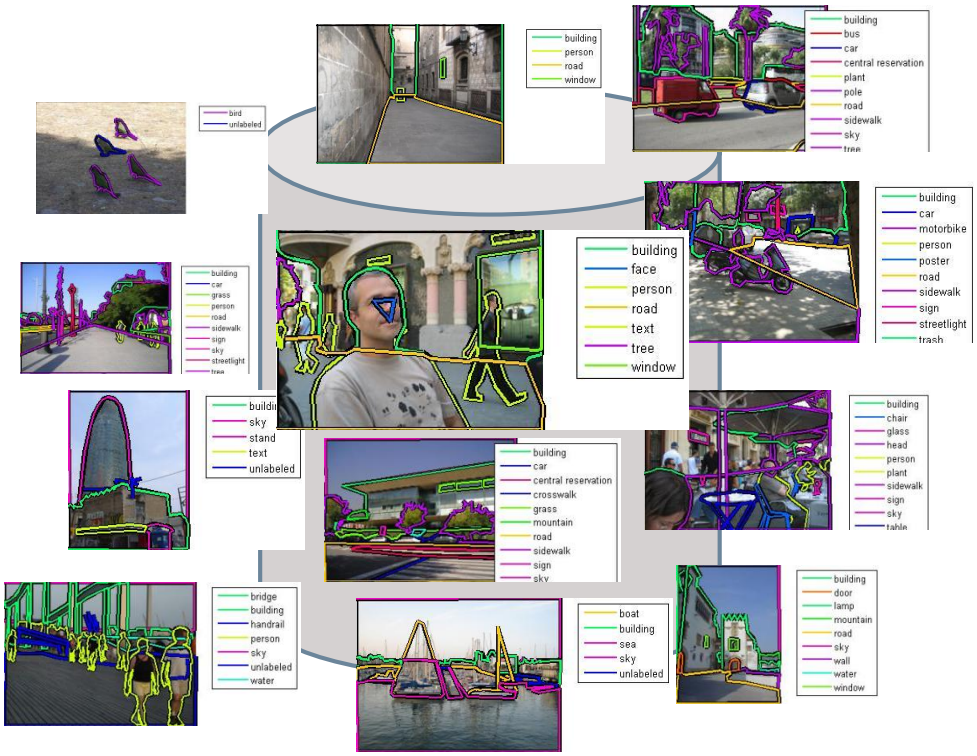


Open-universe recognition





Potential solution: Lazy learning



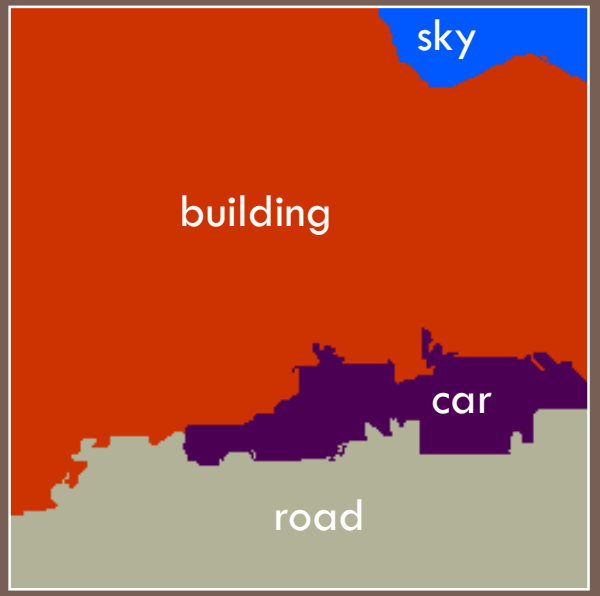
Test image

On-the-fly inference

LARGE-SCALE NONPARAMETRIC IMAGE PARSING

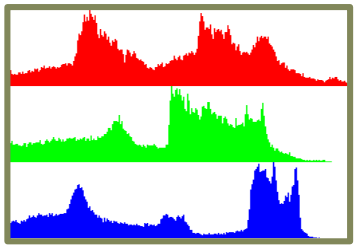
Joseph Tighe and Svetlana Lazebnik

ECCV 2010

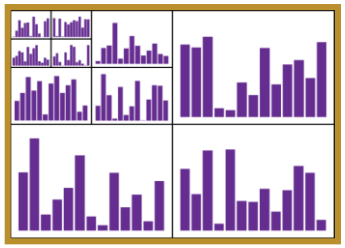


Step 1: Scene-level matching

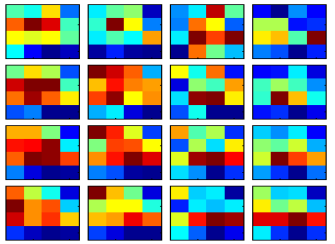
Color Histogram



Spatial Pyramid
(Lazebnik et al., 2006)



Gist
(Oliva & Torralba, 2001)



Step 2: Region-level matching

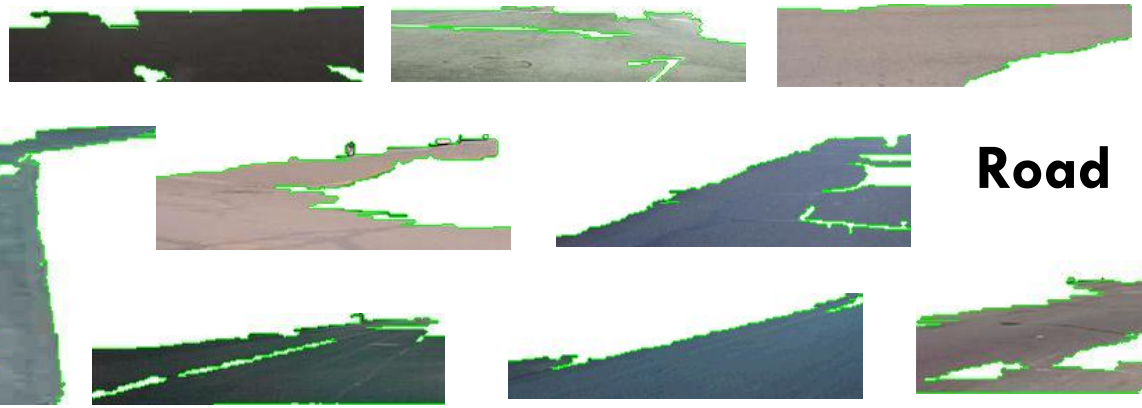
Superpixels
(Felzenszwalb & Huttenlocher, 2004)

Superpixel features

Shape	Mask of superpixel shape over its bounding box (8×8)	64
	Bounding box width/height relative to image width/height	2
	Superpixel area relative to the area of the image	1
Location	Mask of superpixel shape over the image	64
	Top height of bounding box relative to image height	1
Texture/SIFT	Texton histogram, dilated texton histogram	100×2
	SIFT histogram, dilated SIFT histogram	100×2
	Left/right/top/bottom boundary SIFT histogram	100×4
Color	RGB color mean and std. dev.	3×2
	Color histogram (RGB, 11 bins per channel), dilated hist.	33×2
Appearance	Color thumbnail (8×8)	192
	Masked color thumbnail	192
	Grayscale gist over superpixel bounding box	320

Step 2: Region-level matching

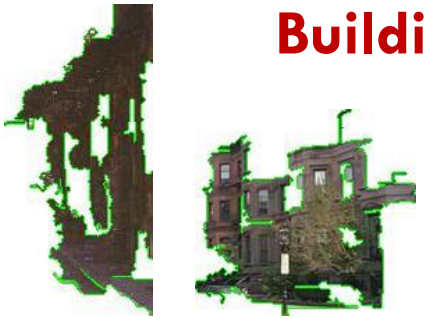
Pixel Area (size)



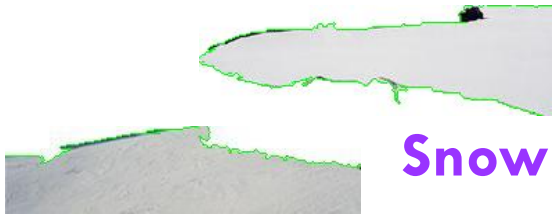
Road

Tree

Sky



Building

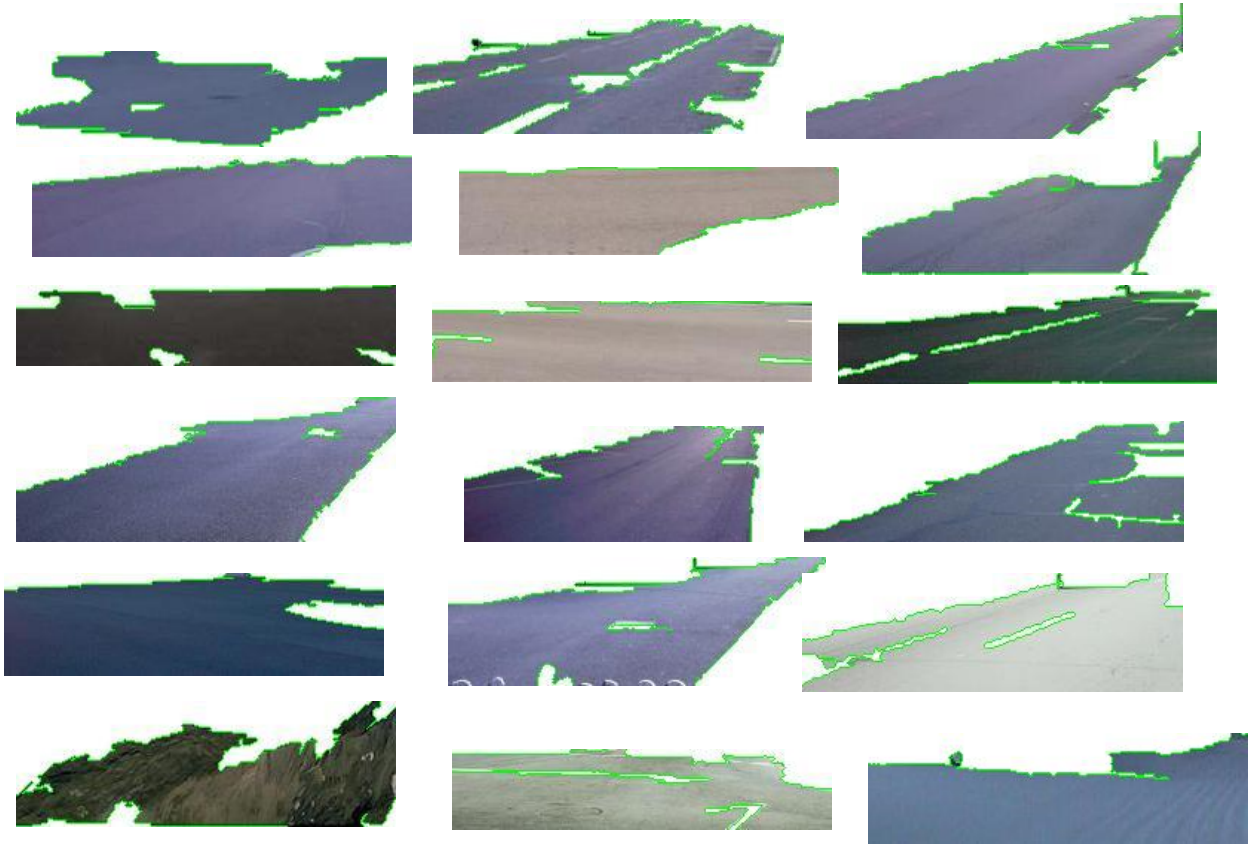


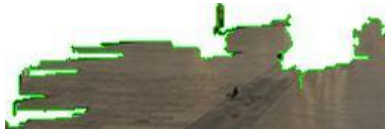
Snow

Step 2: Region-level matching

Absolute mask (location)

Road

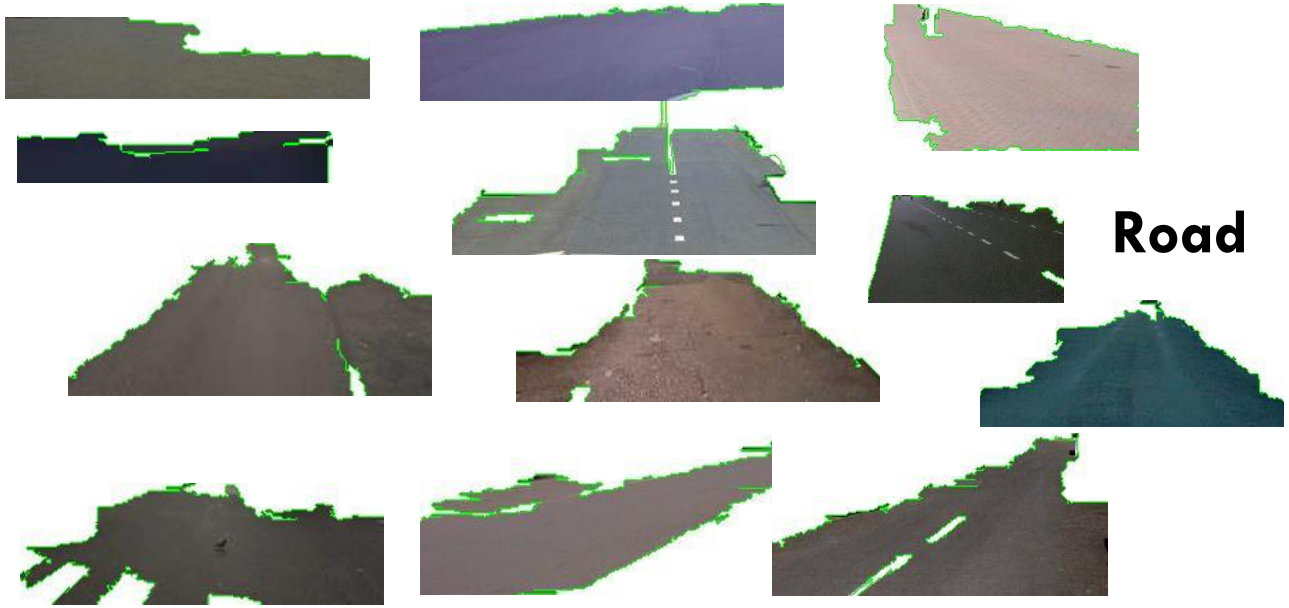




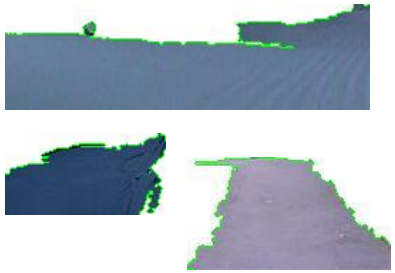
Sidewalk

Step 2: Region-level matching

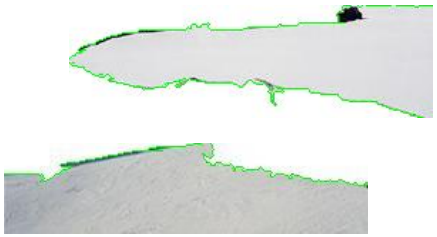
Texture



Road



Sidewalk

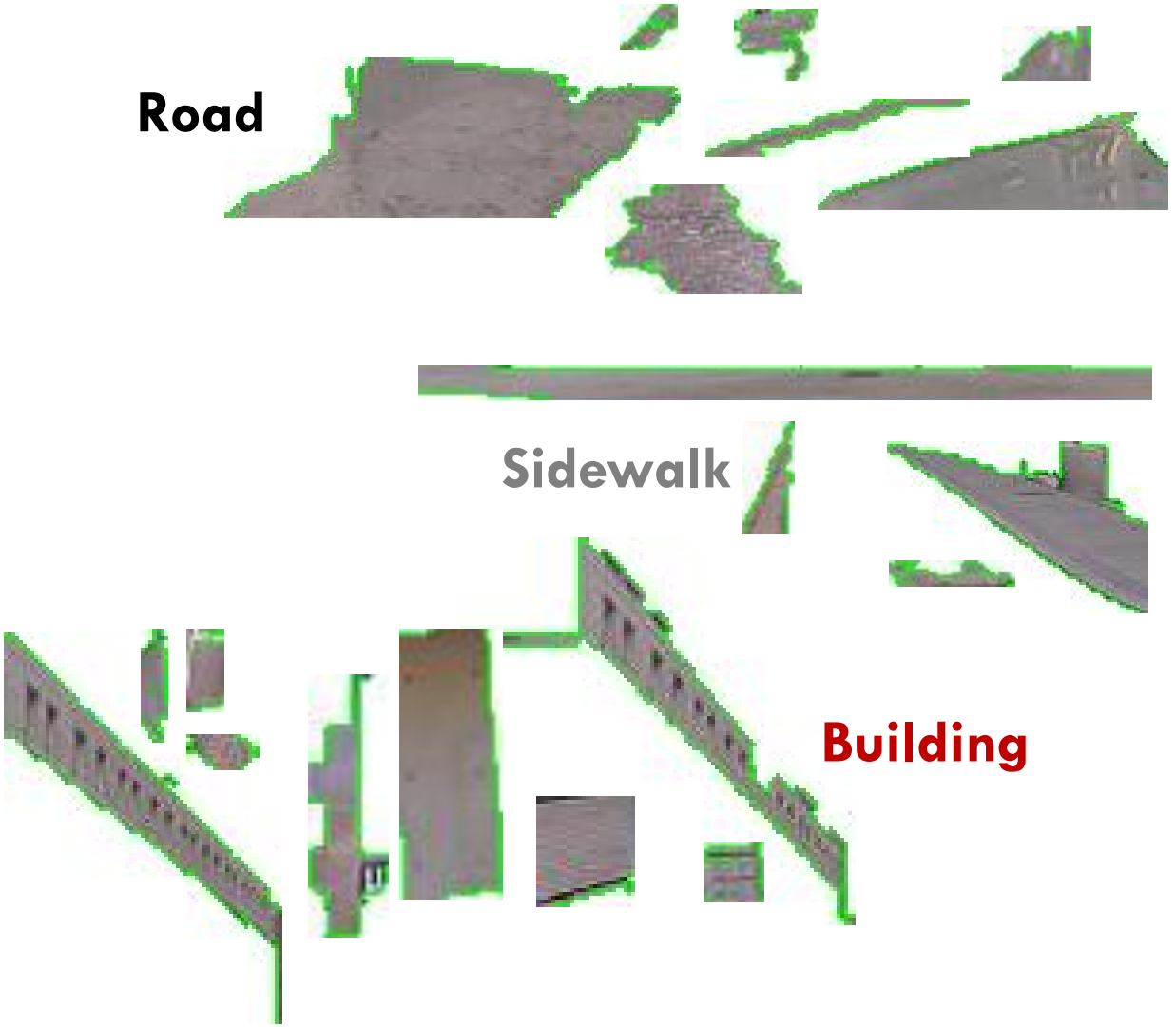


Snow

Sky

Step 2: Region-level matching

Color histogram



Region-level likelihoods

- Nonparametric estimate of class-conditional densities for each class c and feature type *k*:

$$\hat{P}(f_k(r_i) \mid c) = \frac{\#(N(f_k(r_i)), c)}{\#(D, c)}$$

kth feature type of ith region

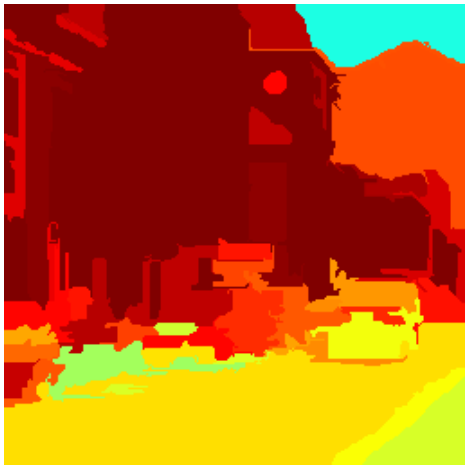
Features of class c within some radius of r_i

Total features of class c in the dataset

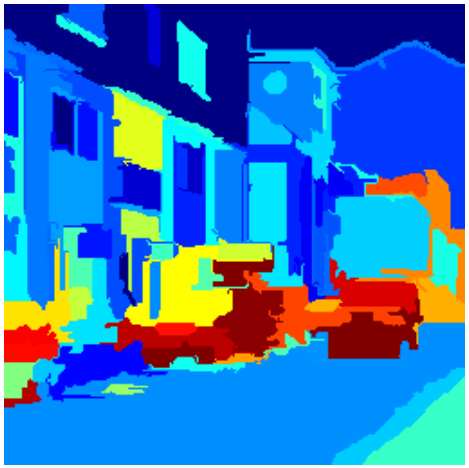
- Per-feature likelihoods combined via Naïve Bayes:

$$\hat{P}(r_i \mid c) = \prod_{\text{features } k} \hat{P}(f_k(r_i) \mid c)$$

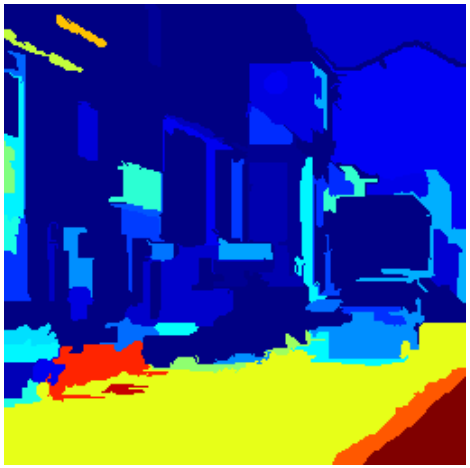
Region-level likelihoods



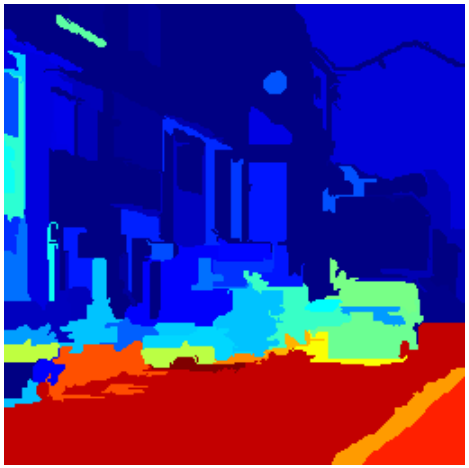
Building



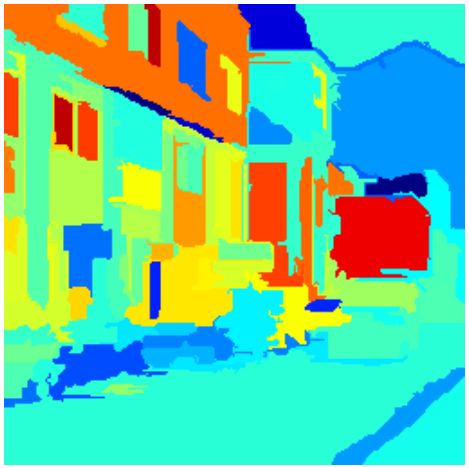
Car



Crosswalk

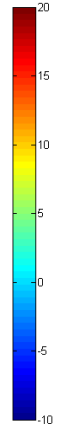


Road



Window

Sky

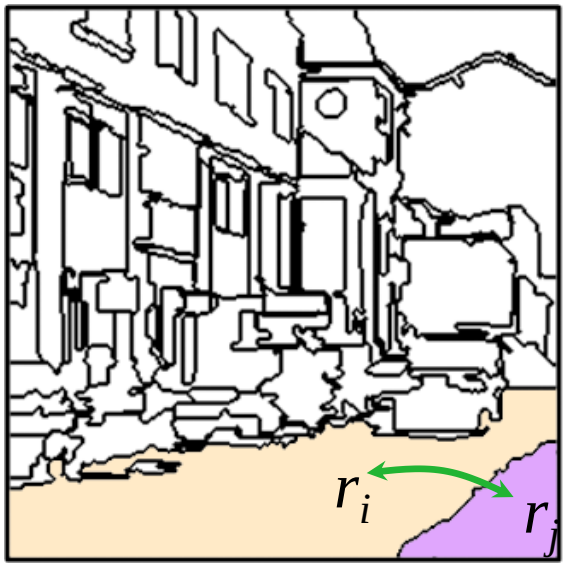


Step 3: Global image labeling

- Compute a global image labeling by optimizing a Markov random field (MRF) energy function:

$$E(\boldsymbol{c}) = \sum_{i} -\log L(r_i, c_i) \;+\; \lambda \sum_{i,j} \delta[c_i \neq c_j]\, \varphi(c_i, c_j)$$

- $\boldsymbol{c}$: *Vector of region labels*
- i: *Regions*
- $-\log L(r_i, c_i)$: *Likelihood score for region r_i and label c_i*
- i, j: *Neighboring regions*
- $\delta[c_i \neq c_j]$: *Smoothing penalty*
- $\varphi(c_i, c_j)$: *Co-occurrence penalty*



Efficient approximate minimization using α-expansion (Boykov et al., 2002)

Step 3: Global image labeling

- Compute a global image labeling by optimizing a Markov random field (MRF) energy function:

$$E(\boldsymbol{c}) = \sum_{i} -\log L(r_i, c_i) \;+\; \lambda \sum_{i,j} \delta[c_i \neq c_j]\, \varphi(c_i, c_j)$$

- $\boldsymbol{c}$: *Vector of region labels*
- i: *Regions*
- $-\log L(r_i, c_i)$: *Likelihood score for region r_i and label c_i*
- i, j: *Neighboring regions*
- $\delta[c_i \neq c_j]$: *Smoothing penalty*
- $\varphi(c_i, c_j)$: *Co-occurrence penalty*

Step 3: Global image labeling

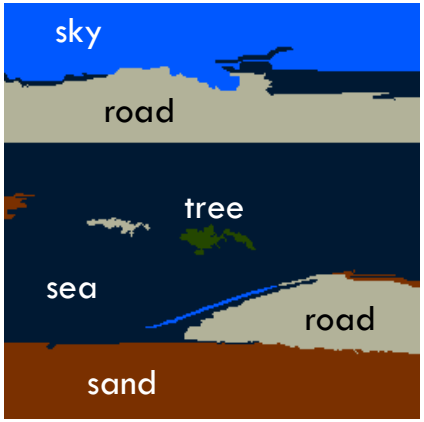
- Compute a global image labeling by optimizing a Markov random field (MRF) energy function:

$$E(\mathbf{c}) = \sum_i -\log L(r_i, c_i) + \lambda \sum_{i,j} \delta[c_i \neq c_j]\, \varphi(c_i, c_j)$$

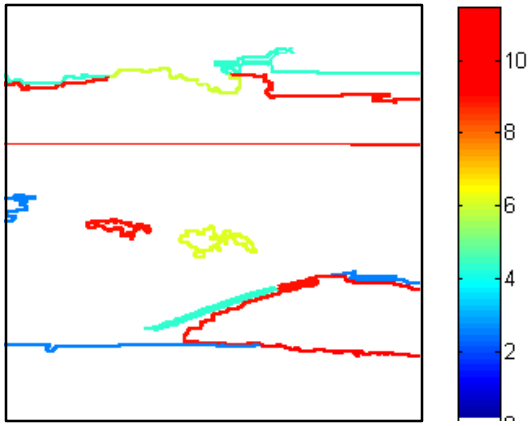
- $\mathbf{c}$: *Vector of region labels*
- i: *Regions*
- $-\log L(r_i, c_i)$: *Likelihood score for region r_i and label c_i*
- i, j: *Neighboring regions*
- $\delta[c_i \neq c_j]$: *Smoothing penalty*
- $\varphi(c_i, c_j)$: *Co-occurrence penalty*

Original image

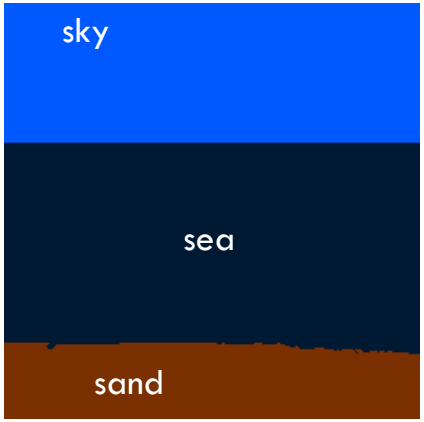
Maximum likelihood labeling



Edge penalties

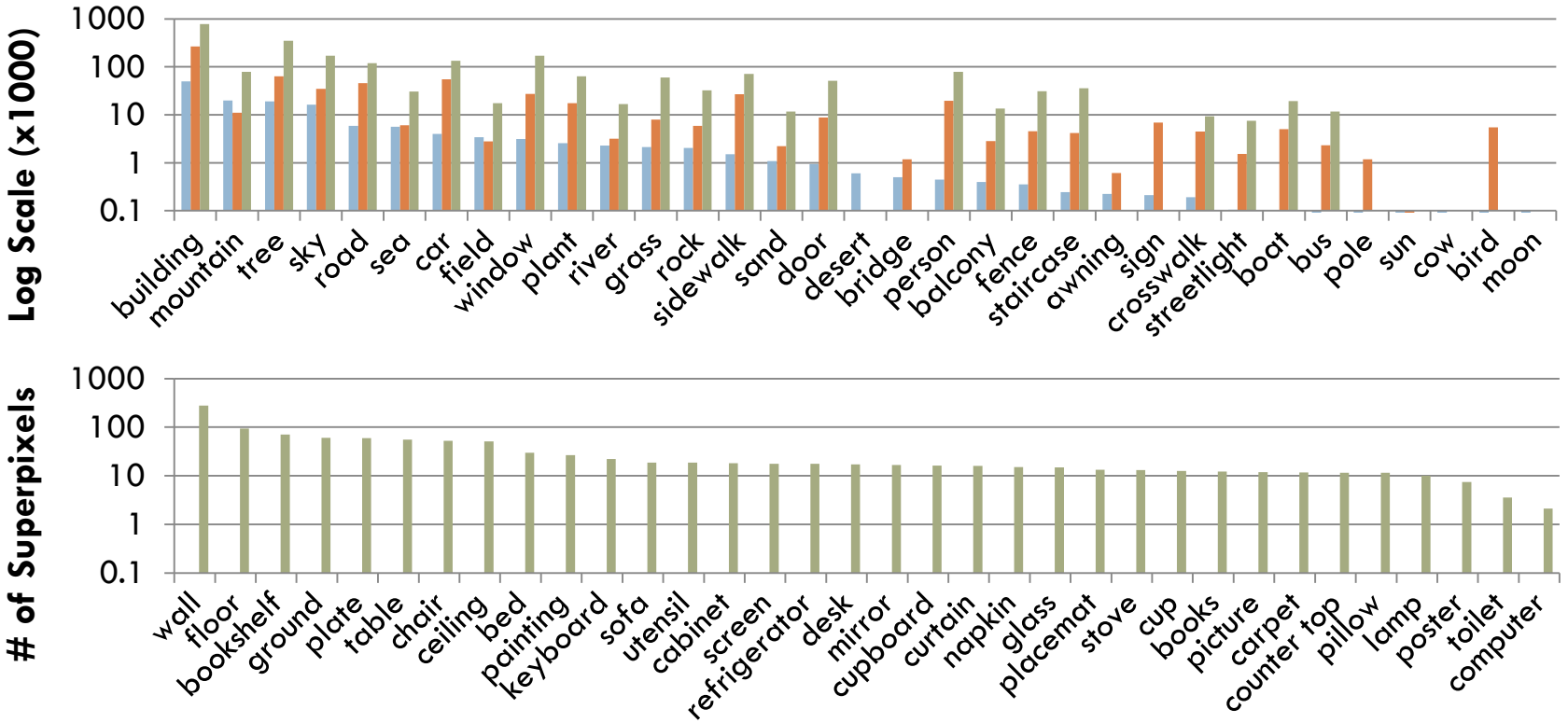


MRF labeling

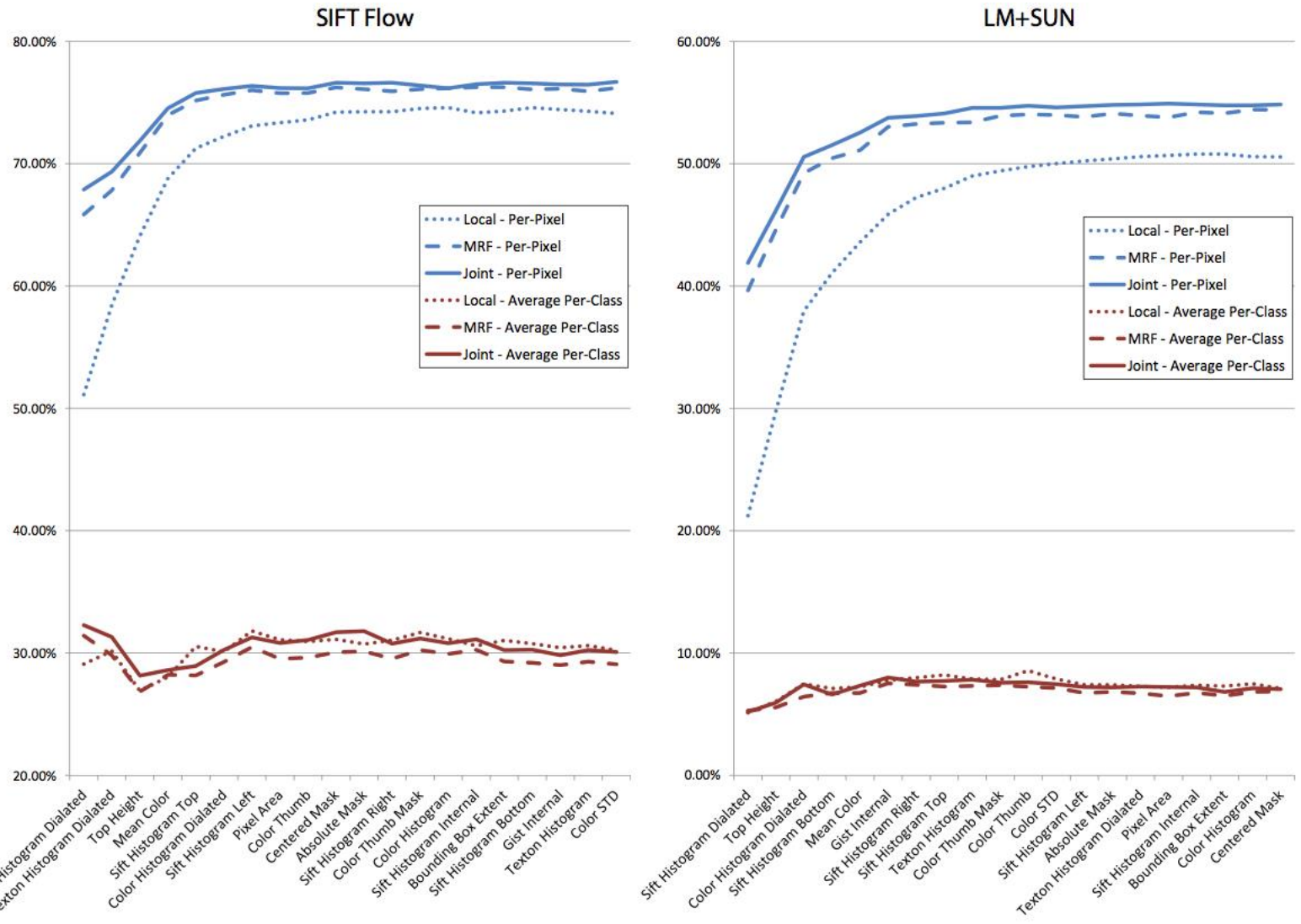


Datasets

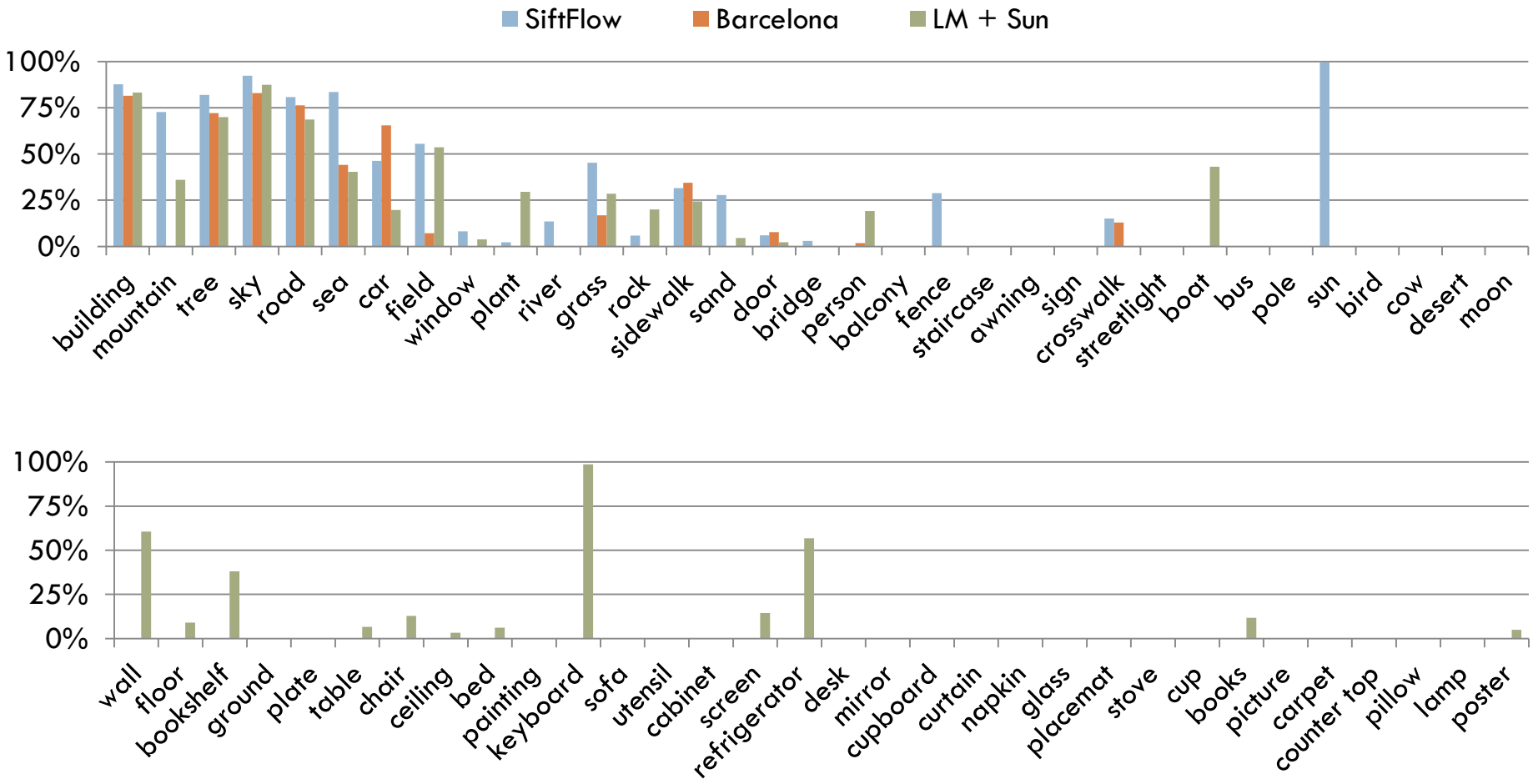
	Training images	Test images	Labels
SIFT Flow (Liu et al., 2009)	2,488	200	33
Barcelona	14,871	279	170
LabelMe+SUN	50,424	300	232



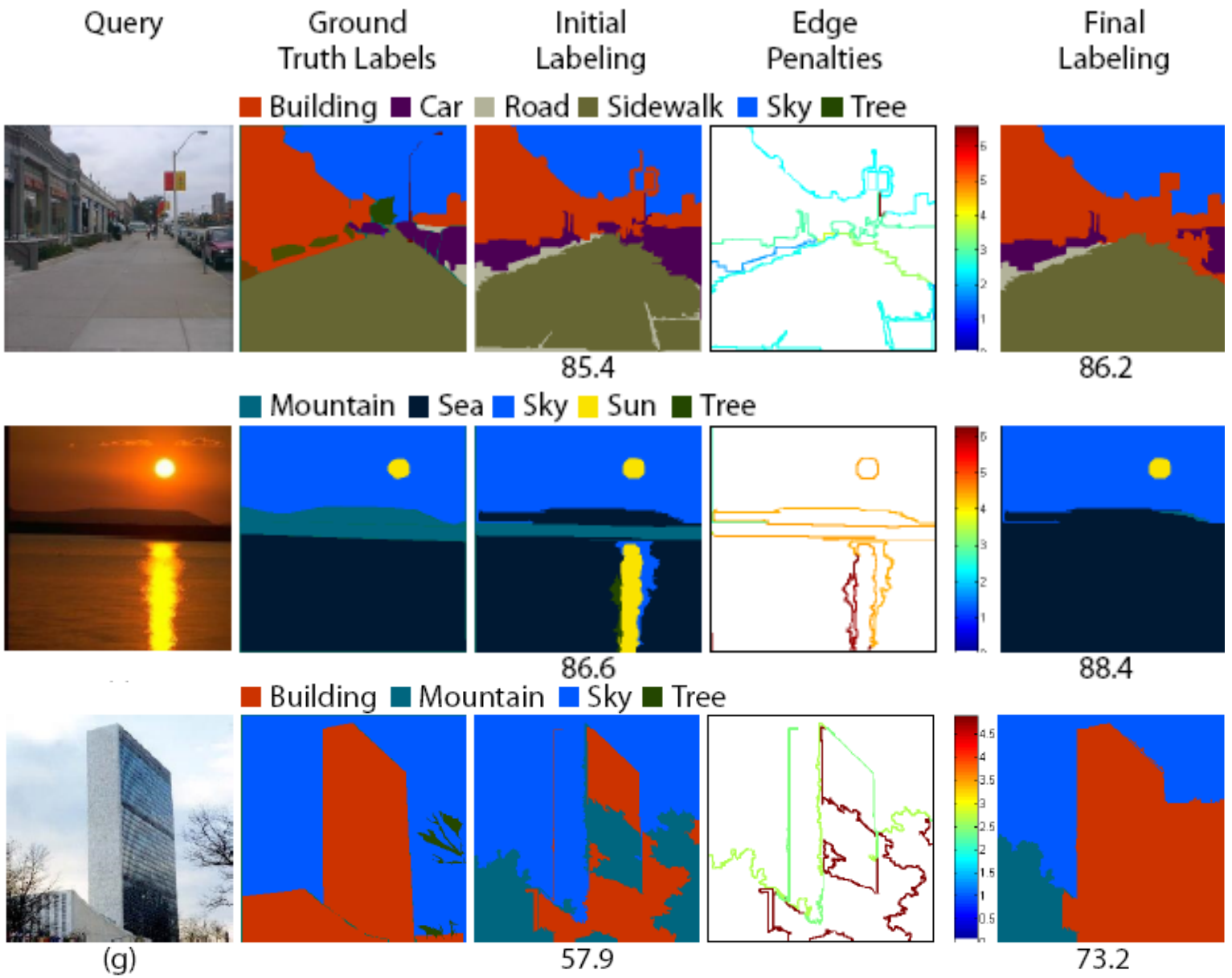
Overall performance



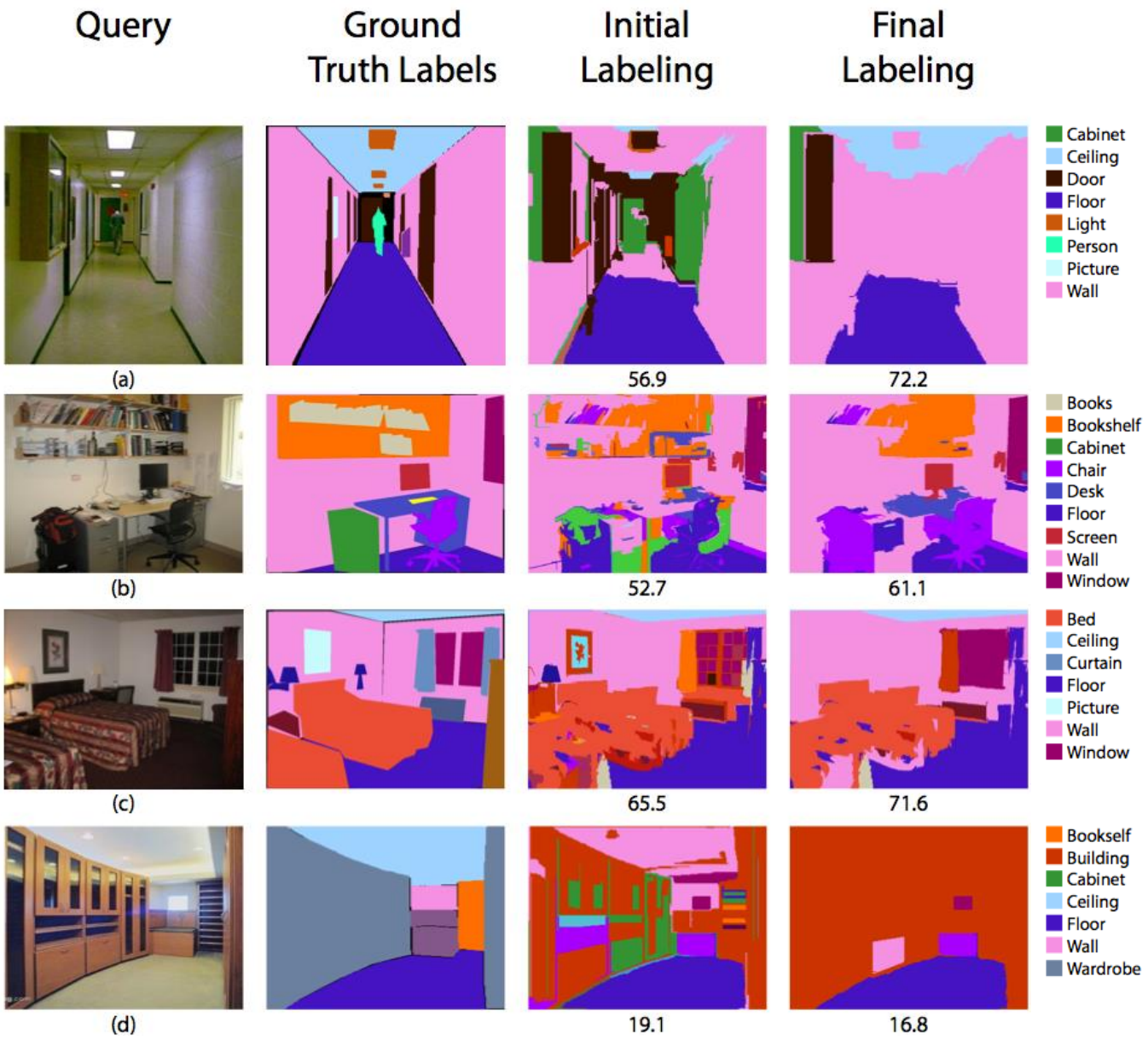
Per-class classification rates



Results on SIFT Flow dataset



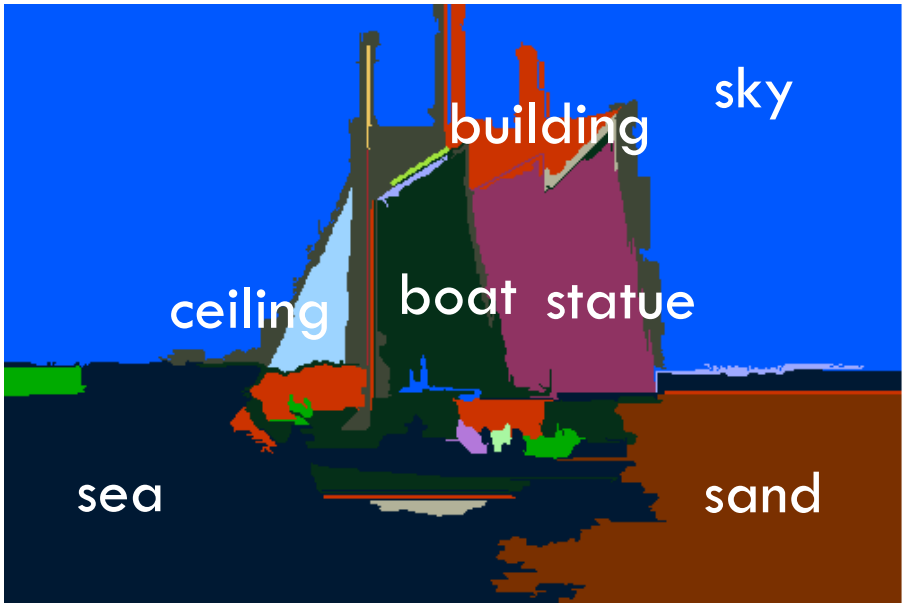
Results on LM+SUN dataset



Summary so far

- A lazy learning method for image parsing:
  - Global scene matching
  - Superpixel-level matching
  - MRF optimization
- Challenges
  - Indoor images are hard!
  - We do well on “stuff” but not on “things”

We get the “stuff” but not the “things”



FINDING THINGS: IMAGE PARSING WITH REGIONS AND PER-EXEMPLAR DETECTORS

Joseph Tighe and Svetlana Lazebnik

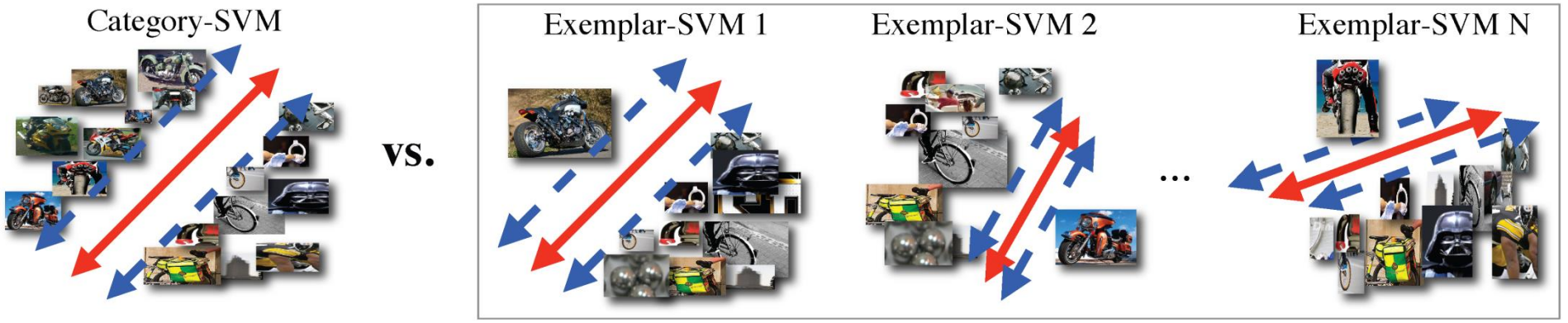
CVPR 2013

Superparsing Result

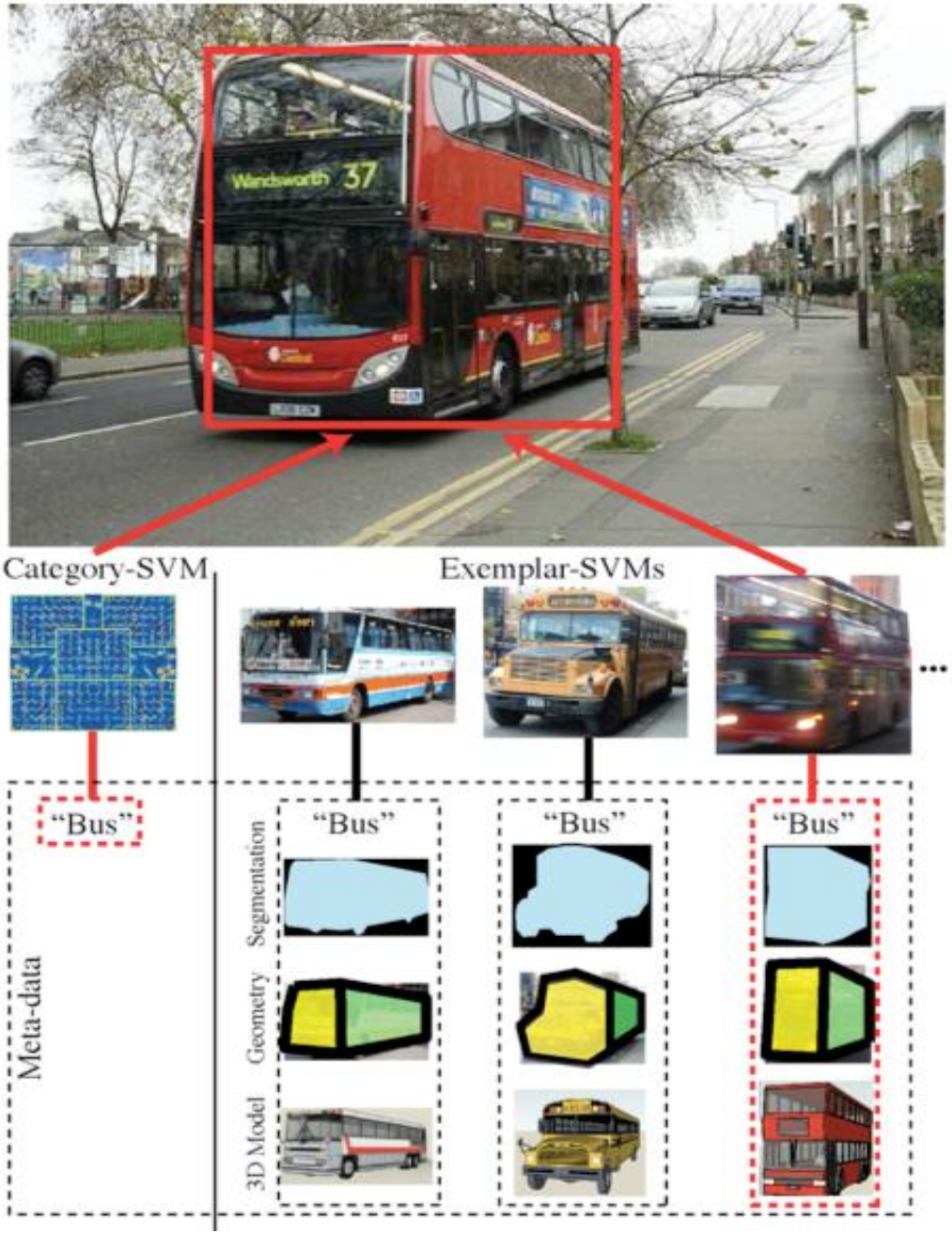
Detector Based Parsing Result

Per-exemplar detectors

- For each instance of a class: train SVM based on HOG features
- Negative examples are taken from all images that do not contain the class



Tomasz Malisiewicz, Abhinav Gupta, Alexei A. Efros. Ensemble of Exemplar-SVMs for Object Detection and Beyond. In ICCV, 2011

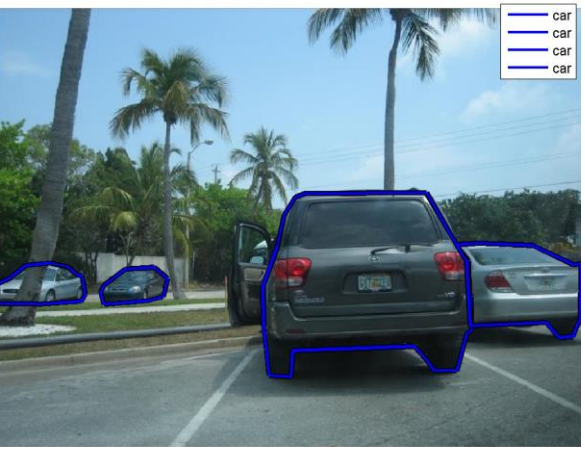


Tomasz Malisiewicz, Abhinav Gupta, Alexei A. Efros. Ensemble of Exemplar-SVMs for Object Detection and Beyond . In ICCV, 2011

Per-exemplar detectors for parsing

- Retrieve a set of similar images using global image descriptors
- Train per-exemplar detectors for “things” in retrieval set
- Run trained detectors on query and transfer weighted masked for all positive detections

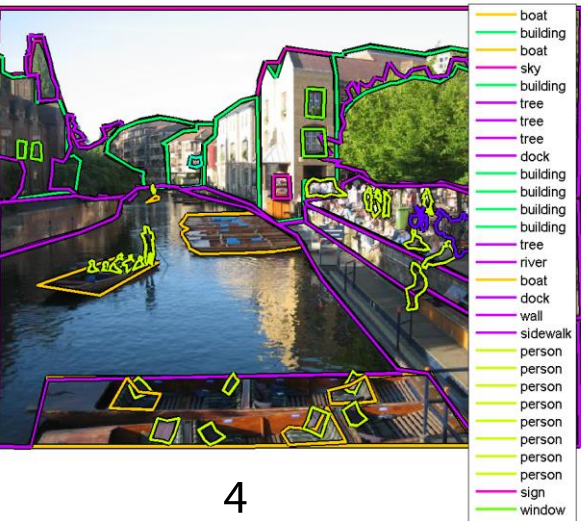
Retrieval set for



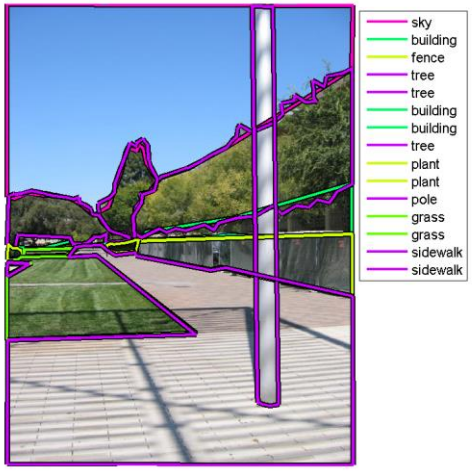
1

2

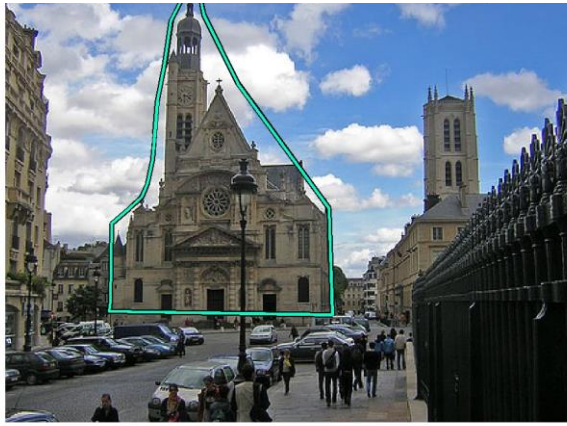
3



4

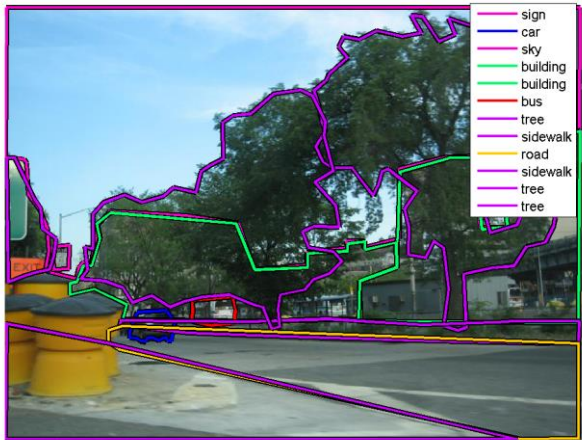


5

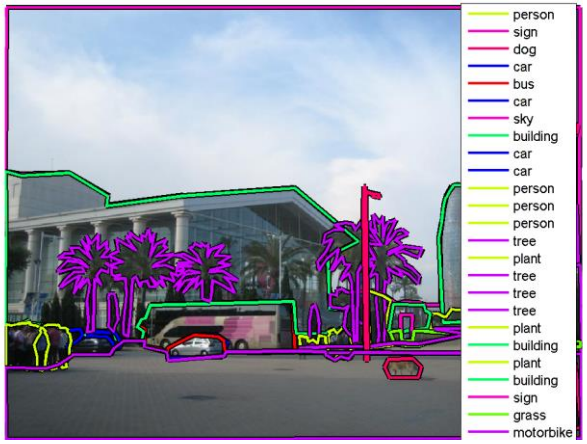


6

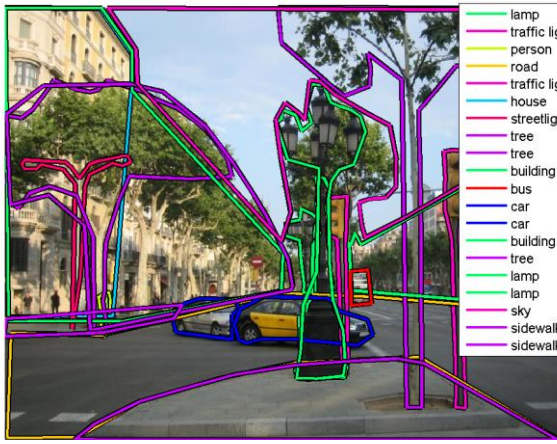
Retrieval set for



16



26



59

342

410

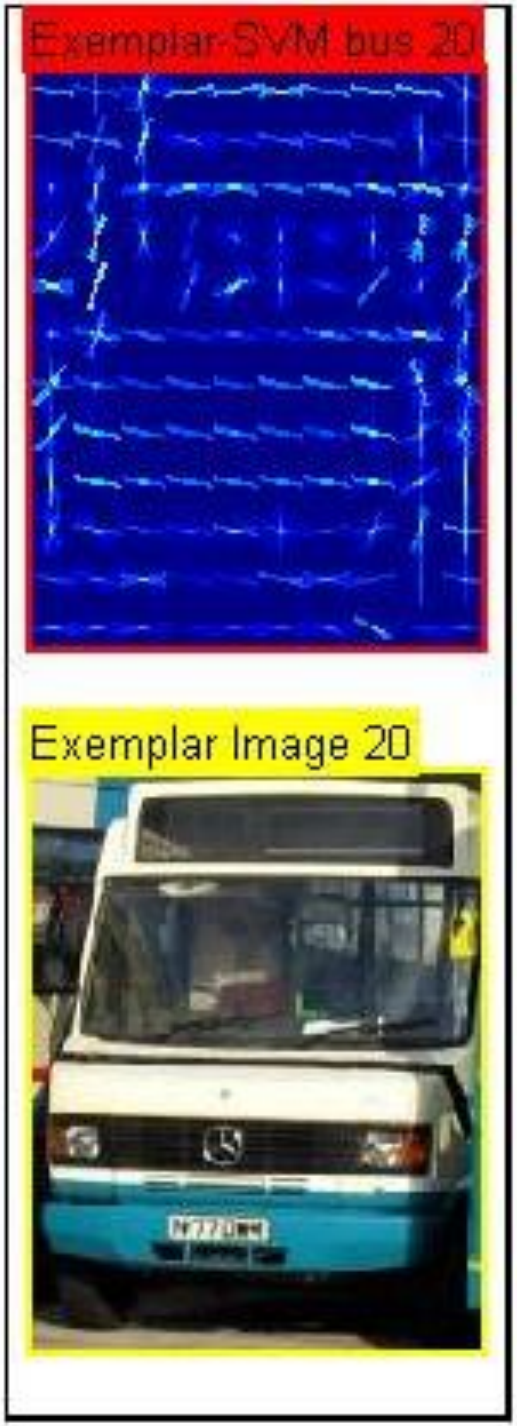
491

Per-exemplar detectors for parsing

- Retrieve a set of similar images using global image descriptors
- Train per-exemplar detectors for each object in retrieval set
- Run trained detectors on query and transfer weighted masked for all positive detections

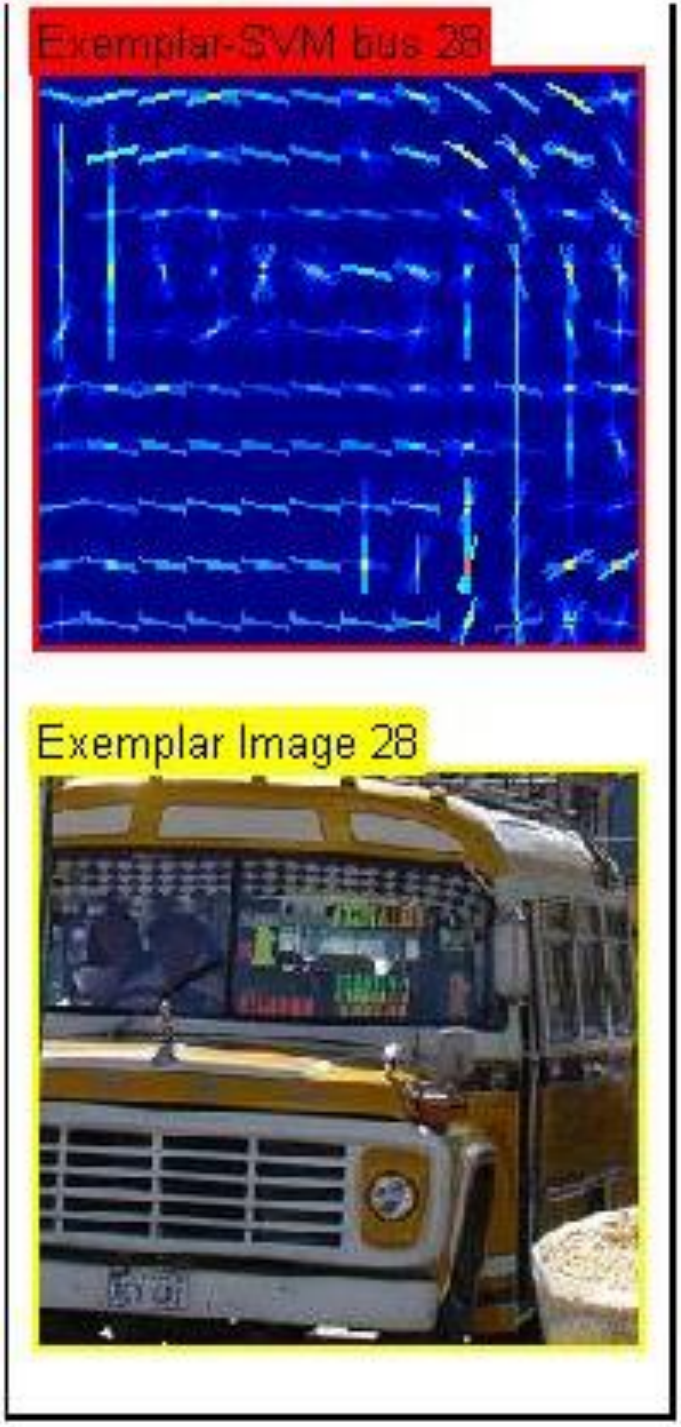
Per-exemplar detectors for parsing

Per-exemplar detectors for parsing


Exemplar SVM bus 20
Exemplar Image 20


bus: -0.625


Exemplar


Exemplar-SVM bus 28
Exemplar Image 28

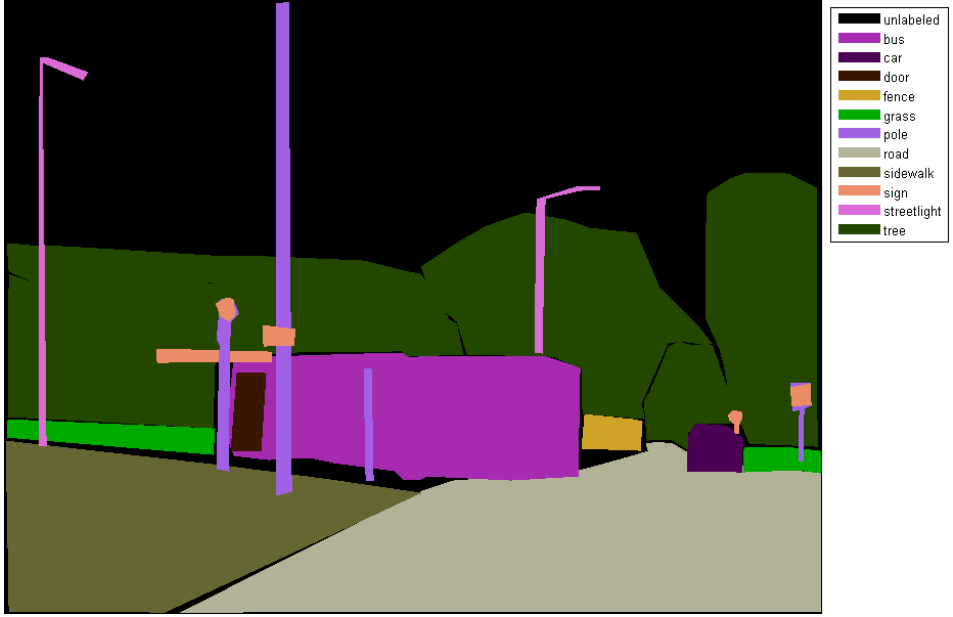

bus -0.692


Exemplar

Per-exemplar detectors for parsing

- Retrieve a set of similar images using global image descriptors
- Train per-exemplar detectors for “things” in retrieval set
- Run trained detectors on query and transfer weighted masks for all positive detections

Per-exemplar detectors for parsing



Superparsing Result

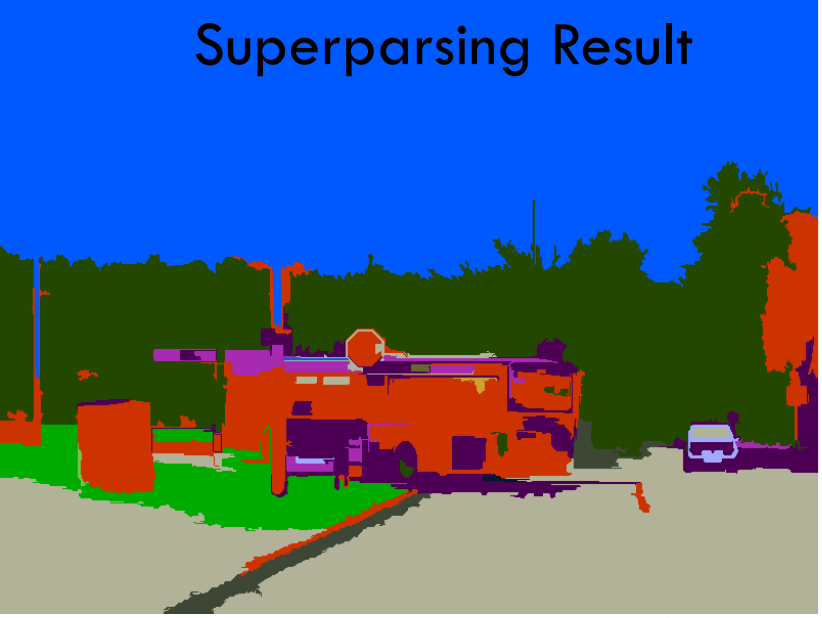
55% (23%)

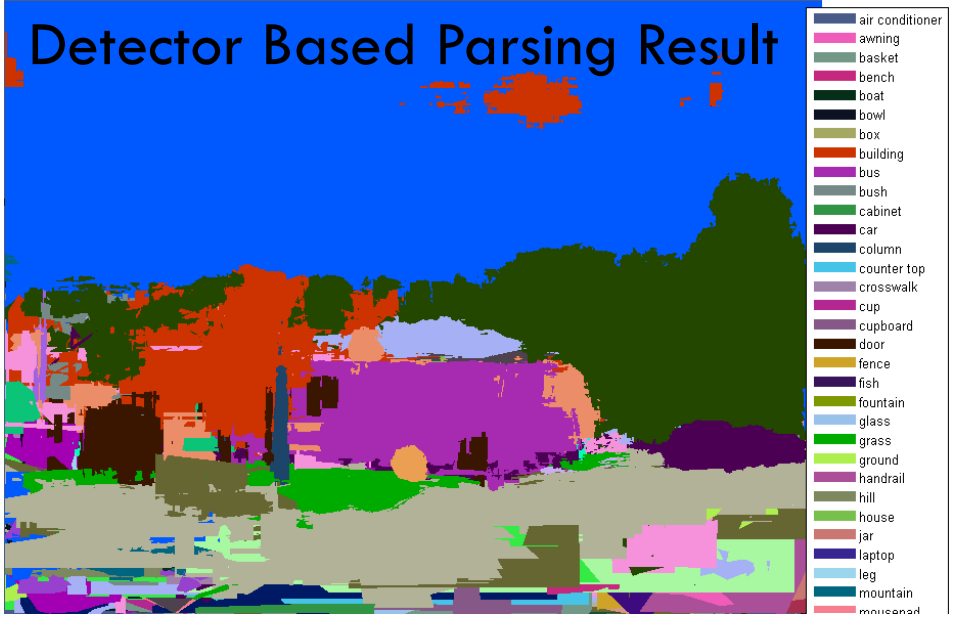
Detector-based Parsing Result

45% (26%)

How do we combine these?

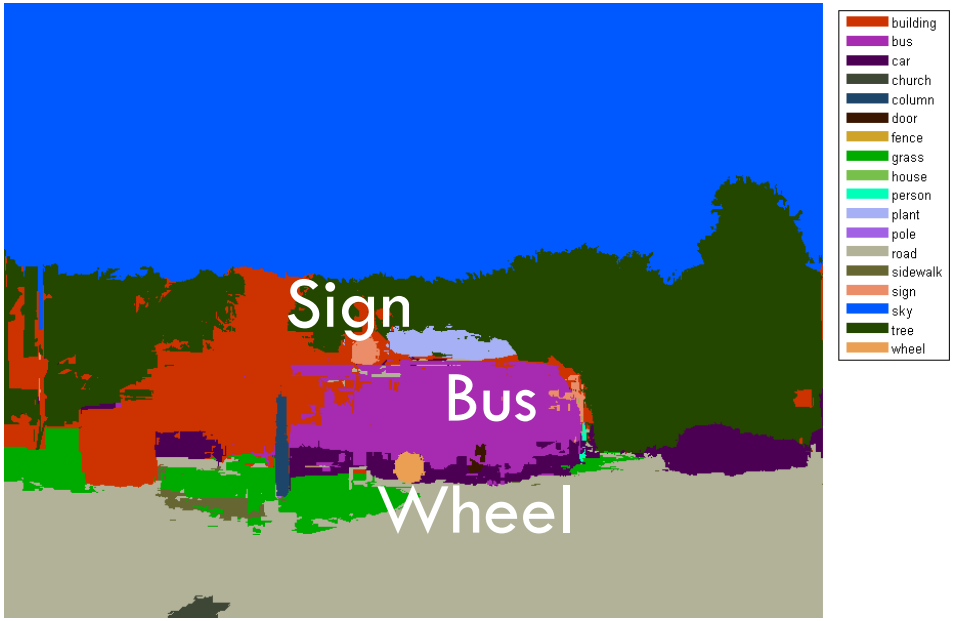
- Learn which labels to trust. If there are c classes, there are 2c predictions at each pixel (one from super-parsing, one from the object detectors).
- Learn an SVM to predict the best category from those 2c confidences.
- Then smooth with an MRF



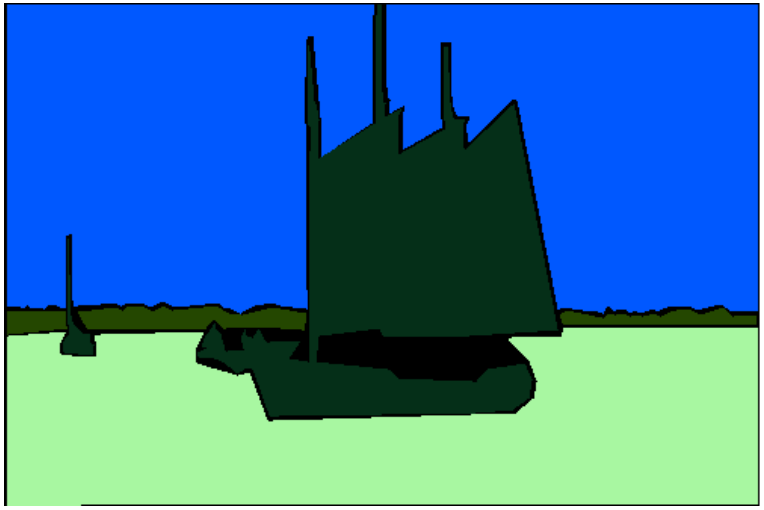


55% (23%)

45% (26%)

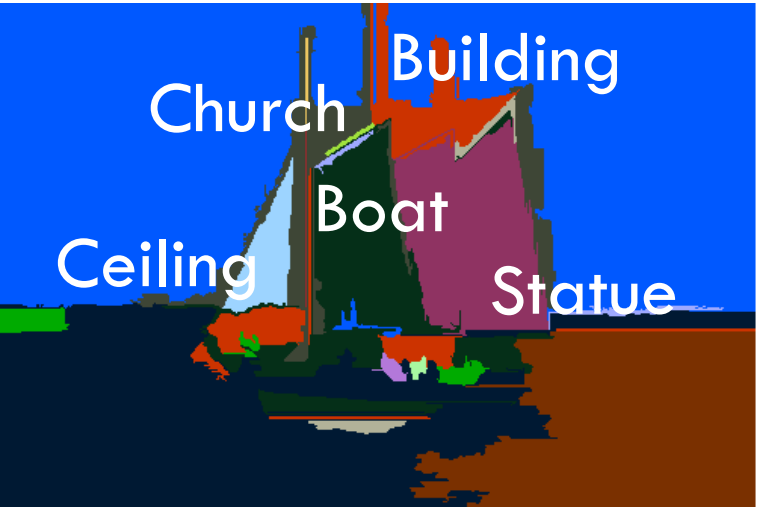


61% (31%)



Superparsing Result

Detector Based Parsing Result



52% (31%)

19% (25%)

Superparsing Result

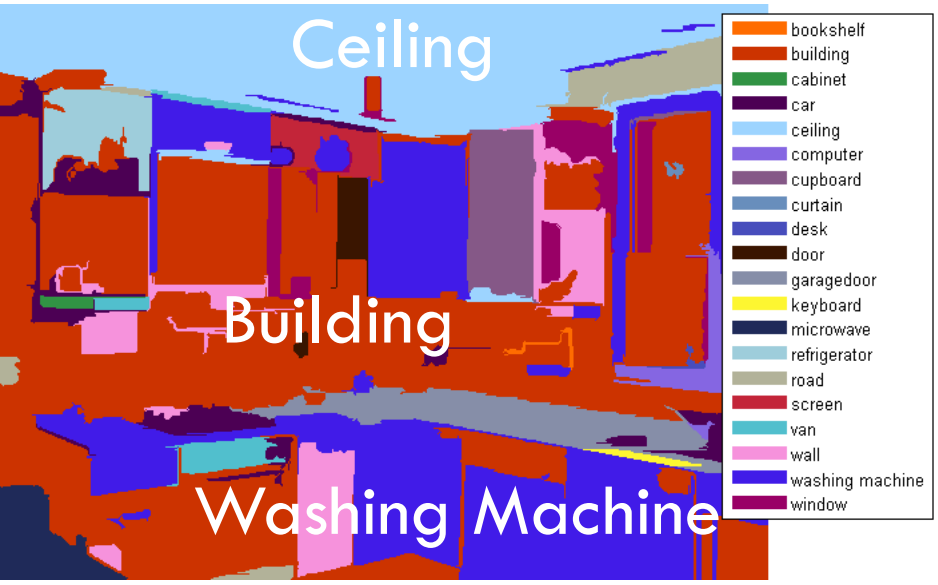
52% (31%)

Detector Based Parsing Result

19% (25%)

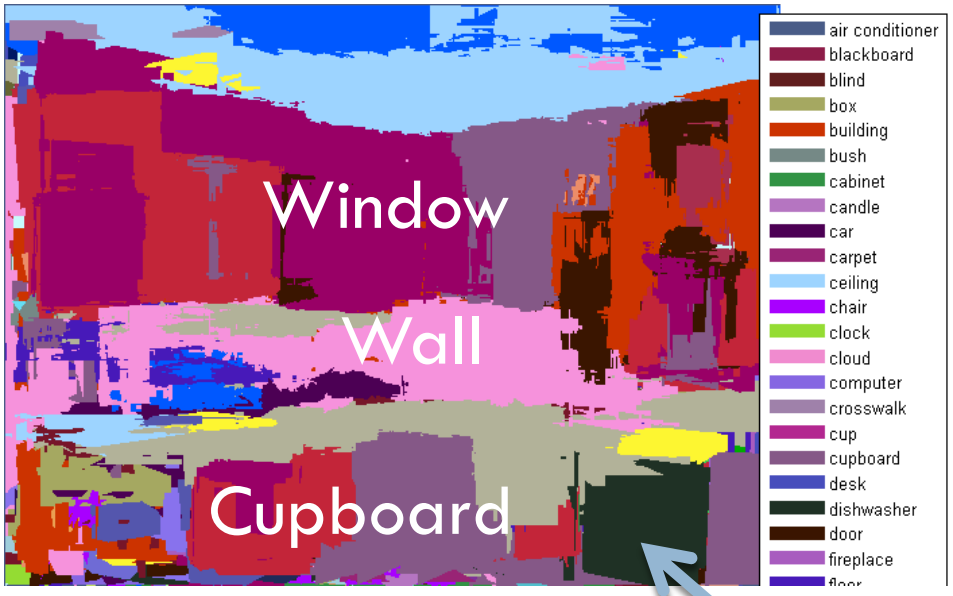
62% (46%)

Superparsing Result



12% (7%)

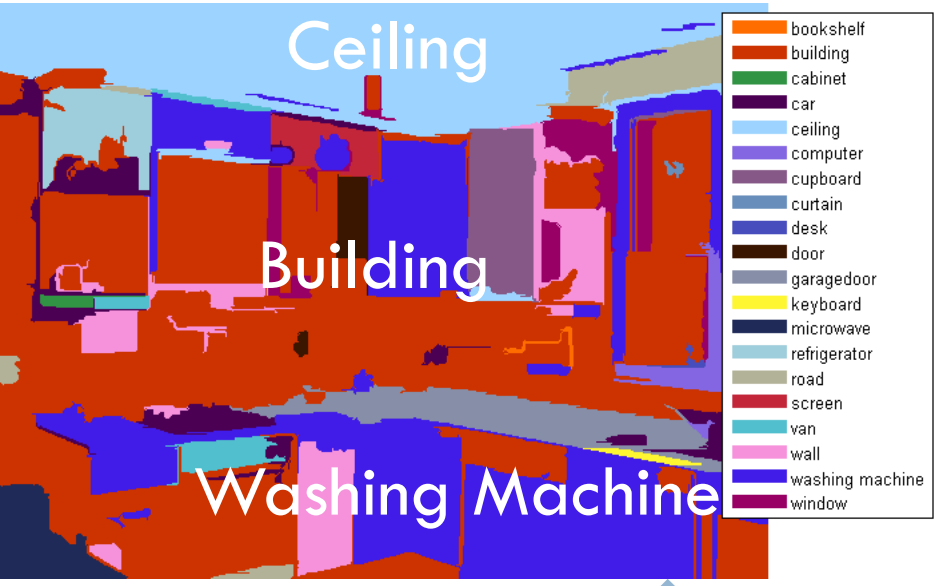
Detector Based Parsing Result



20% (9%)

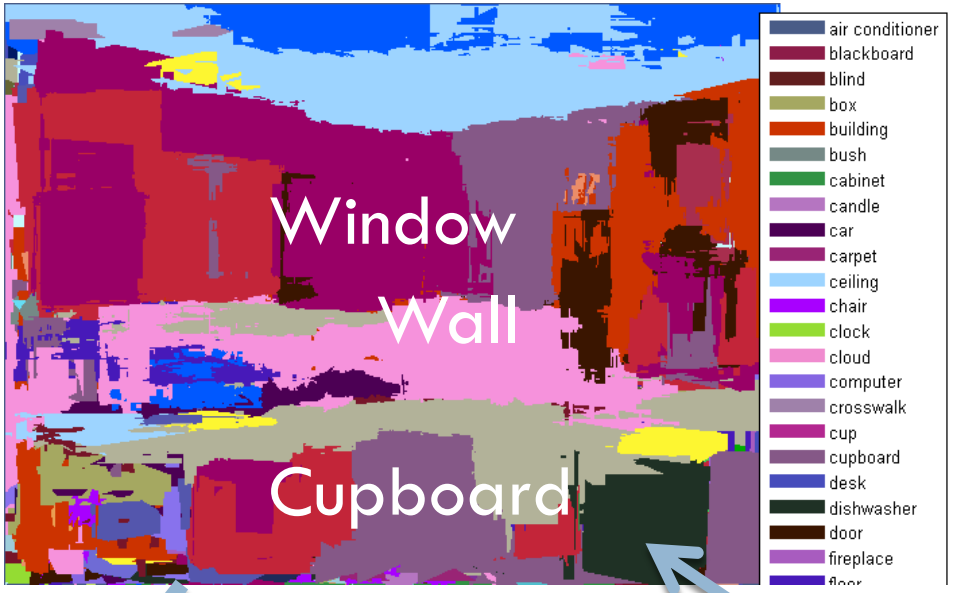
Dishwasher

Superparsing Result

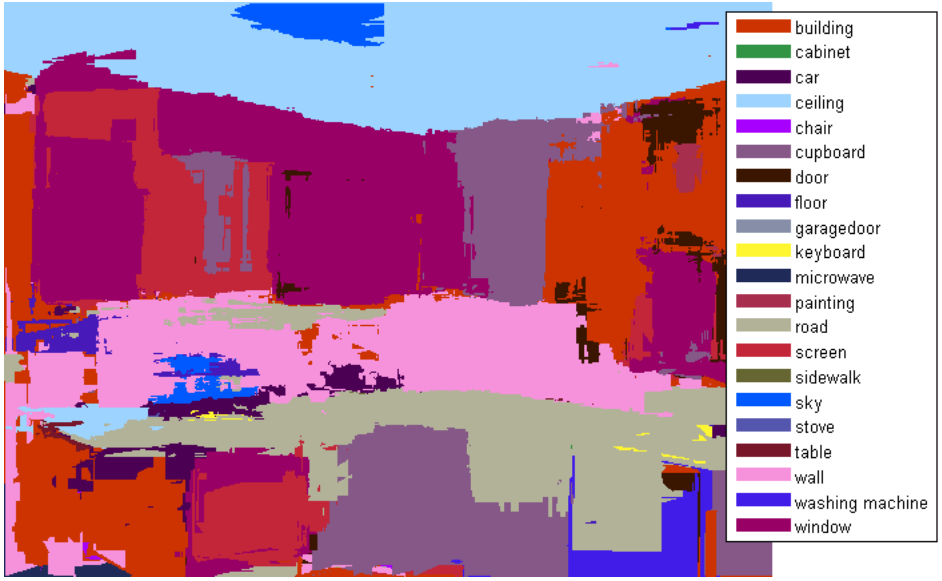


12% (7%)

Detector Based Parsing Result



20% (9%)



24% (10%)

SuperParsing Conclusion

- Image parsing with superpixels
  - Scene-level matching
  - Superpixel-level matching
  - MRF optimization
- Getting “things” with detectors
  - Use per-exemplar detectors of Malisiewicz et al.

Summary

- There are several ways to generate semantic segmentations.
  - Segment then classify
  - Detect then segment
  - Various things in between
- Not clear what is correct.
- Expect to see more research in this area as PASCAL VOC fades and MS COCO gets more attention.