Sketch Tokens: A Learned Mid-level Representation for Contour and Object Detection

By

Joseph J. Lim, C. Lawrence Zitnick and Piotr Dollár

Outline

- Introduction
- Related Work
- Method
- Results
- References

Introduction

- This paper offers a new mid-level feature extraction method to capture local edge structure

- Mid-level features provide a bridge between low-level and high-level features

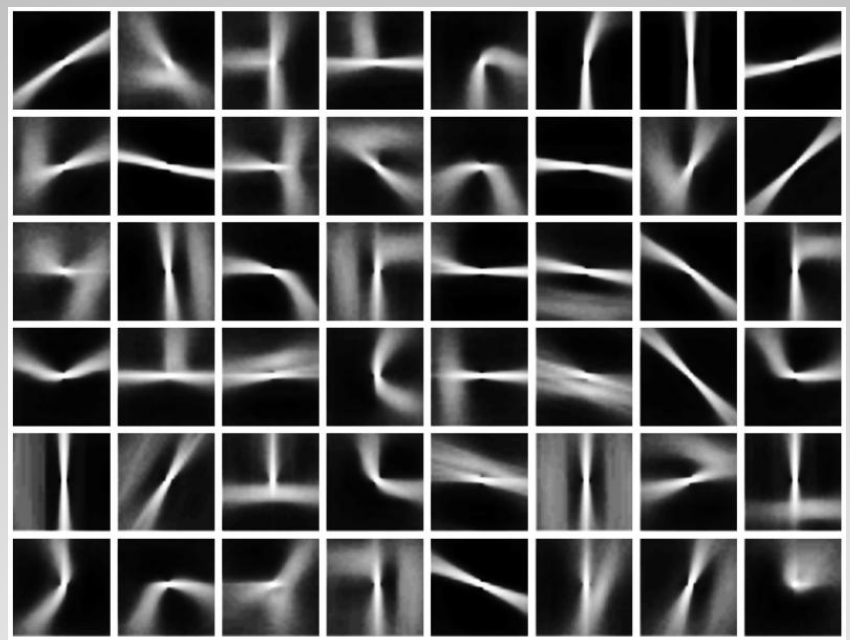
- Especially edge information was a popular early approach to designing mid-level features

Introduction

- The authors propose a novel approach to both learning and detecting local edge-based mid-level features called “sketch tokens”

- And demonstrate their effectiveness for both bottom-up and top-down tasks

Introduction



Examples of sketch tokens learned from hand drawn sketches represented using their mean contour structure.Notice the variety and richness of the sketch tokens.

Introduction

- Method summary
  - Defining sketch tokens: Learn informative sketch tokens
  - Detecting sketch tokens: predict the occurrence of sketch tokens given an input color image

Related Work

- Detecting object boundaries using low-, mid-, and high-level information (CVPR,2007)
  - uses contextual and shape information to refine the edge maps

- Discriminatively trained sparse code gradients for contour detection (NIPS, 2012)
  - learns a patch representation through sparse coding and measure local gradients of the resulting codes

Method

- Defining sketch token classes
- Detecting sketch tokens

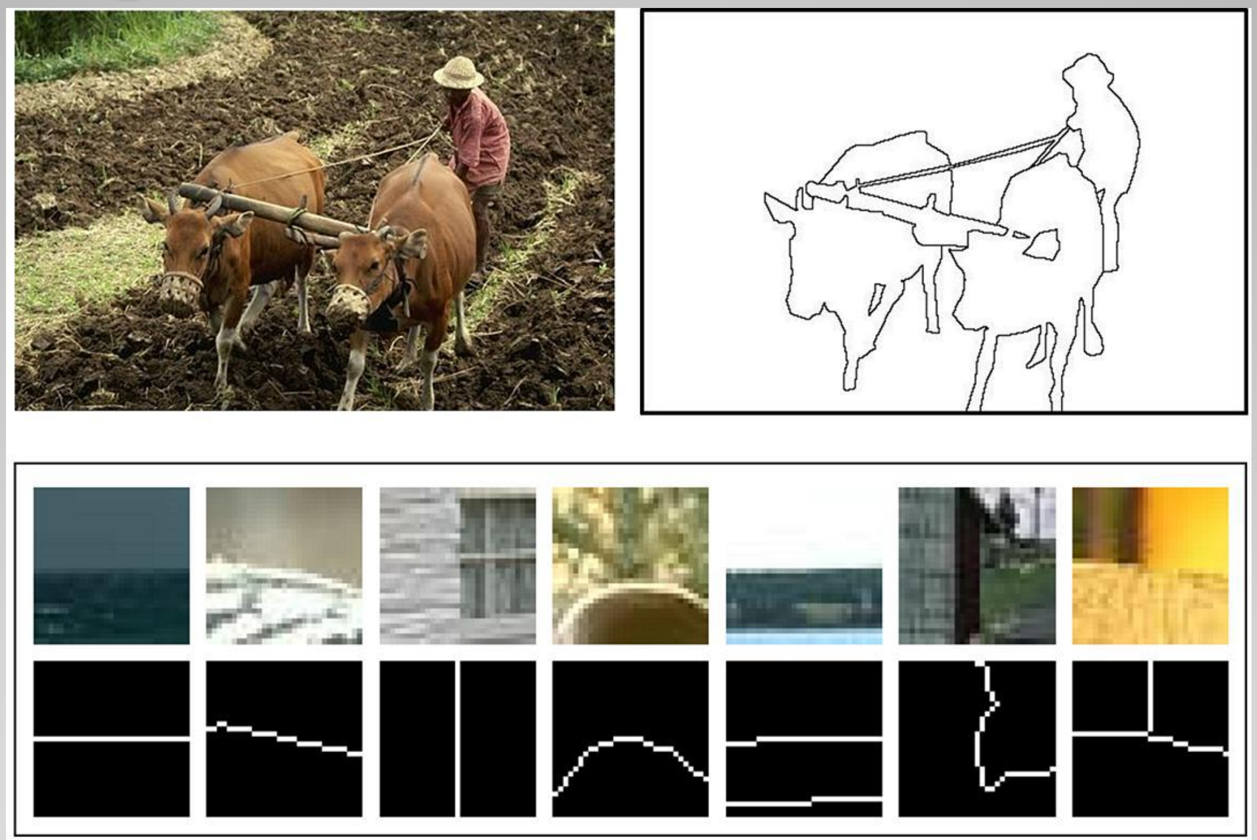
Defining sketch token classes

- Goal is to define a set of token classes that represent the wide variety of local edge structures

- These include straigth lines, t-junctions, y-junctions, corners,curves, parallel lines ...

- Sketch tokens are learned with a supervised method

Defining sketch token classes



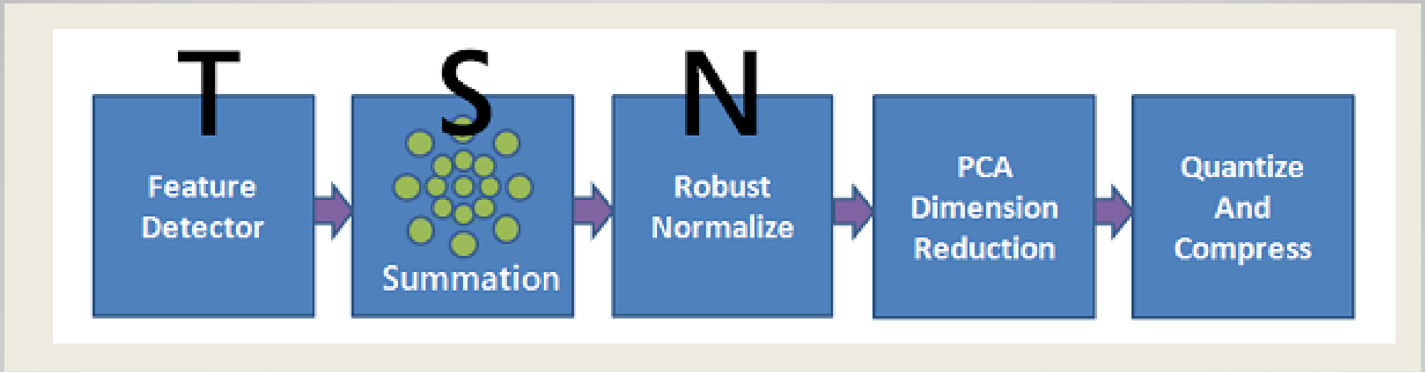
Defining sketch token classes

- Extract 35x35 patches from each binary image

- Calculate Daisy descriptors of patches

Defining sketch token classes

- Calculate Daisy descriptors of patches

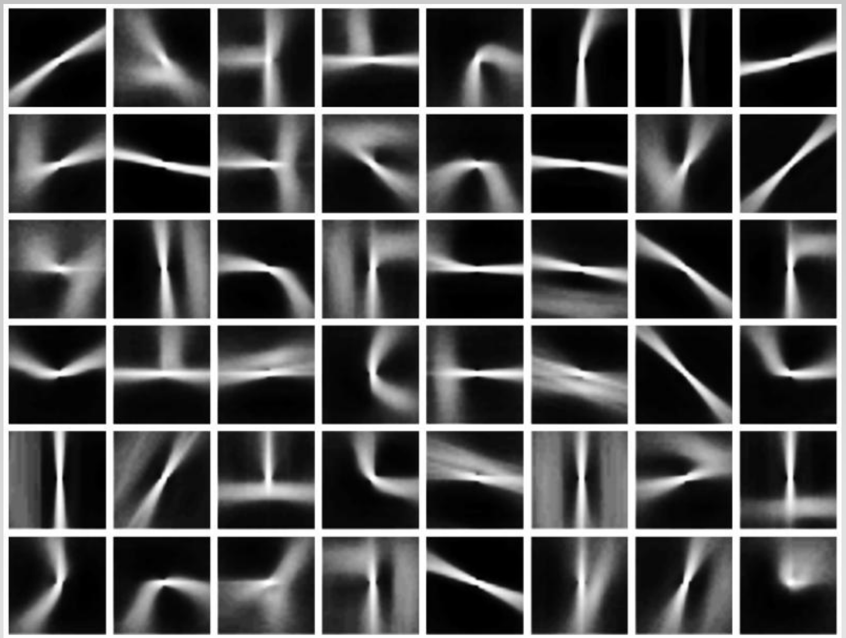


Picking the best DAISY. CVPR 2009

Defining sketch token classes

- Extract 35x35 patches from each binary image
- Calculate Daisy descriptors of patches
- Perform K-means clustering algorithm
  - Choose k=150

Defining sketch token classes



Detecting sketch tokens

- Contains two steps

  I. Feature extraction

  II. Classification

Detecting sketch tokens

I. Feature Extraction

- features directly indexing into the channels
- self-similarity features

Detecting sketch tokens

1. Features directly indexing into the channels

- Channels are composed of color, gradient and oriented gradient information
- Color channels are computed using CIE-LUV color space
- 3 gradient magnitude channels are computed with varying amounts of blur
- 8 oriented gradient channels are computed

Detecting sketch tokens

2. Self-similarity features

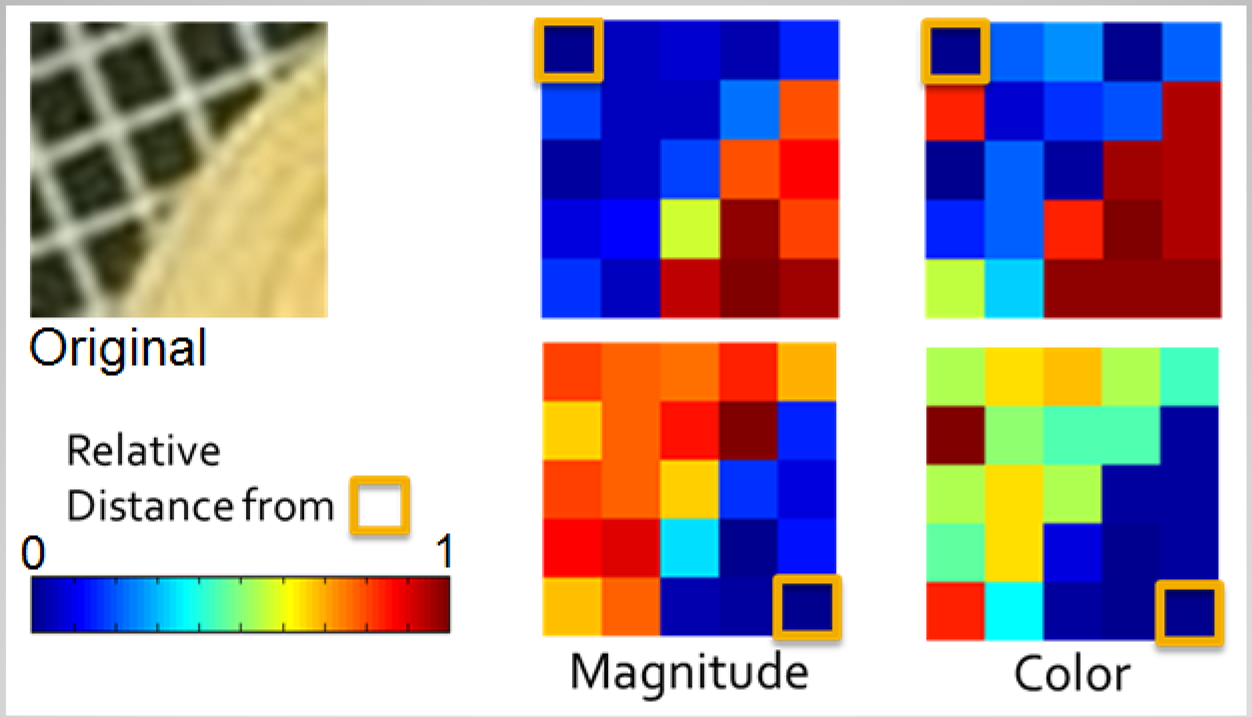
- The self-similarity features capture the portions of an image patch that contain similar textures based on color or gradient information

- For channel k and grid cells i and j, we define the self-similarity feature f_{ijk} as:

$$f_{ijk} = s_{jk} - s_{ik},$$

- where s_{jk} is the sum of grid cell j in channel k.

Detecting sketch tokens



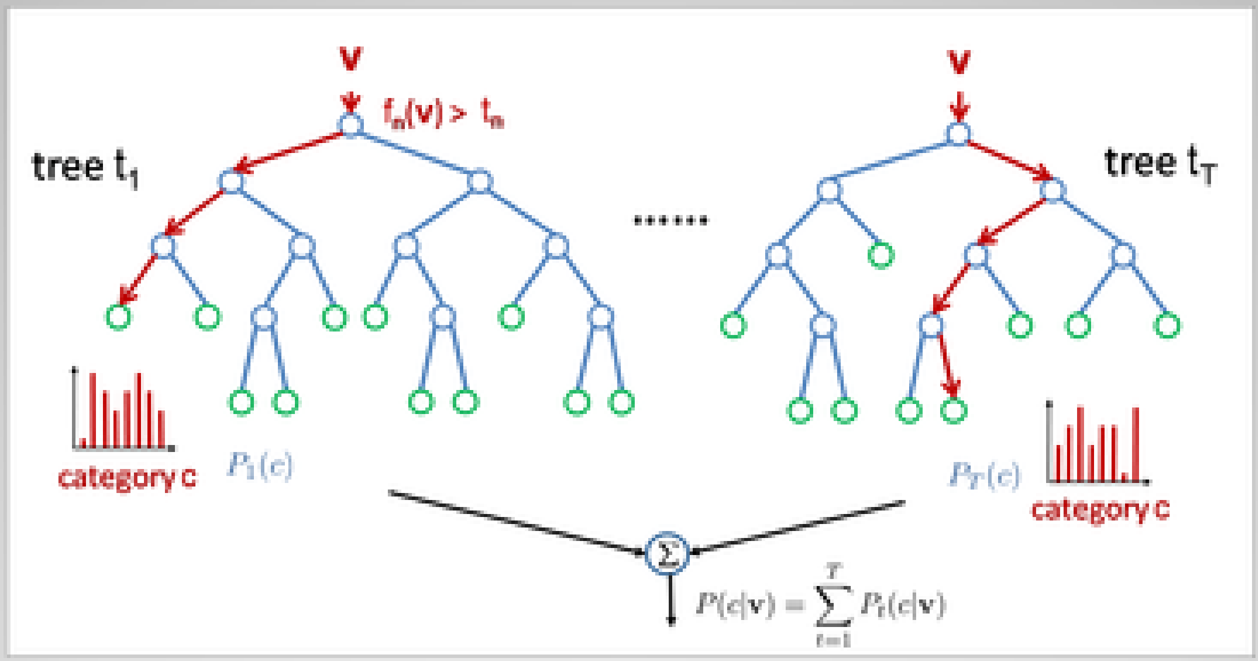
Detecting sketch tokens

II. Classification

- They use random forest classifier
- The leaf nodes contain the probabilities of belonging to each class
- Randomly sample 150000 contour patches and 160000 no contour patches

Detecting sketch tokens

II. Classification



Boosting & Randomized Forests for Visual Recognition , ICCV 2009

Detecting sketch tokens

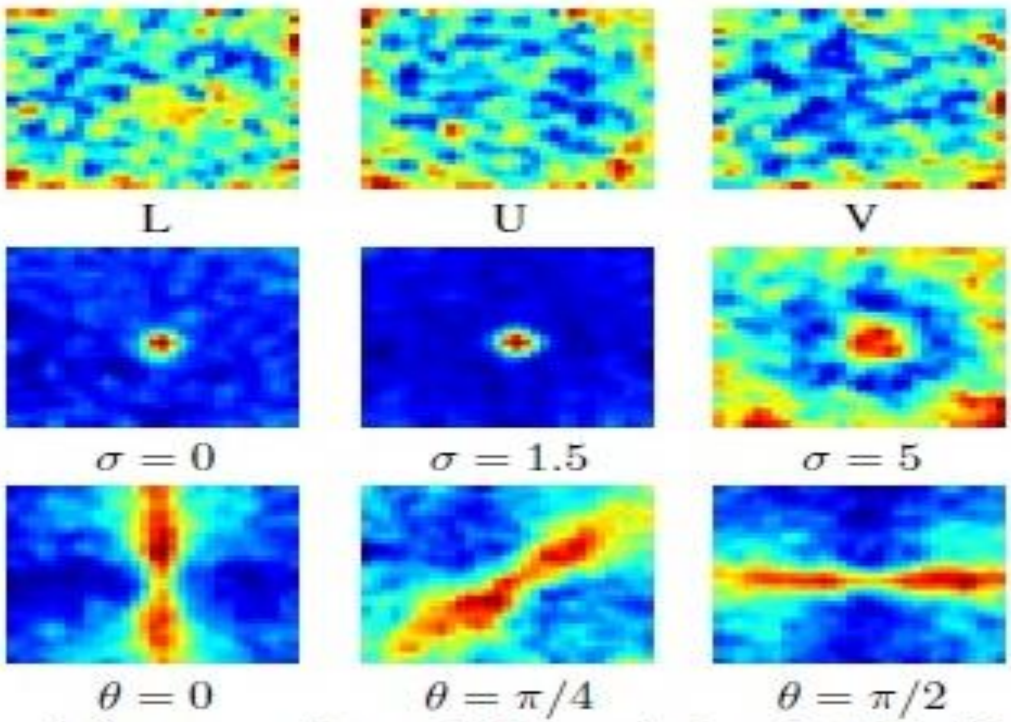


Figure 4. Frequency of example features being selected by the random forest: (first row) color channels, (second row) gradient magnitude channels, (third row) selected orientation channels.

Detecting sketch tokens

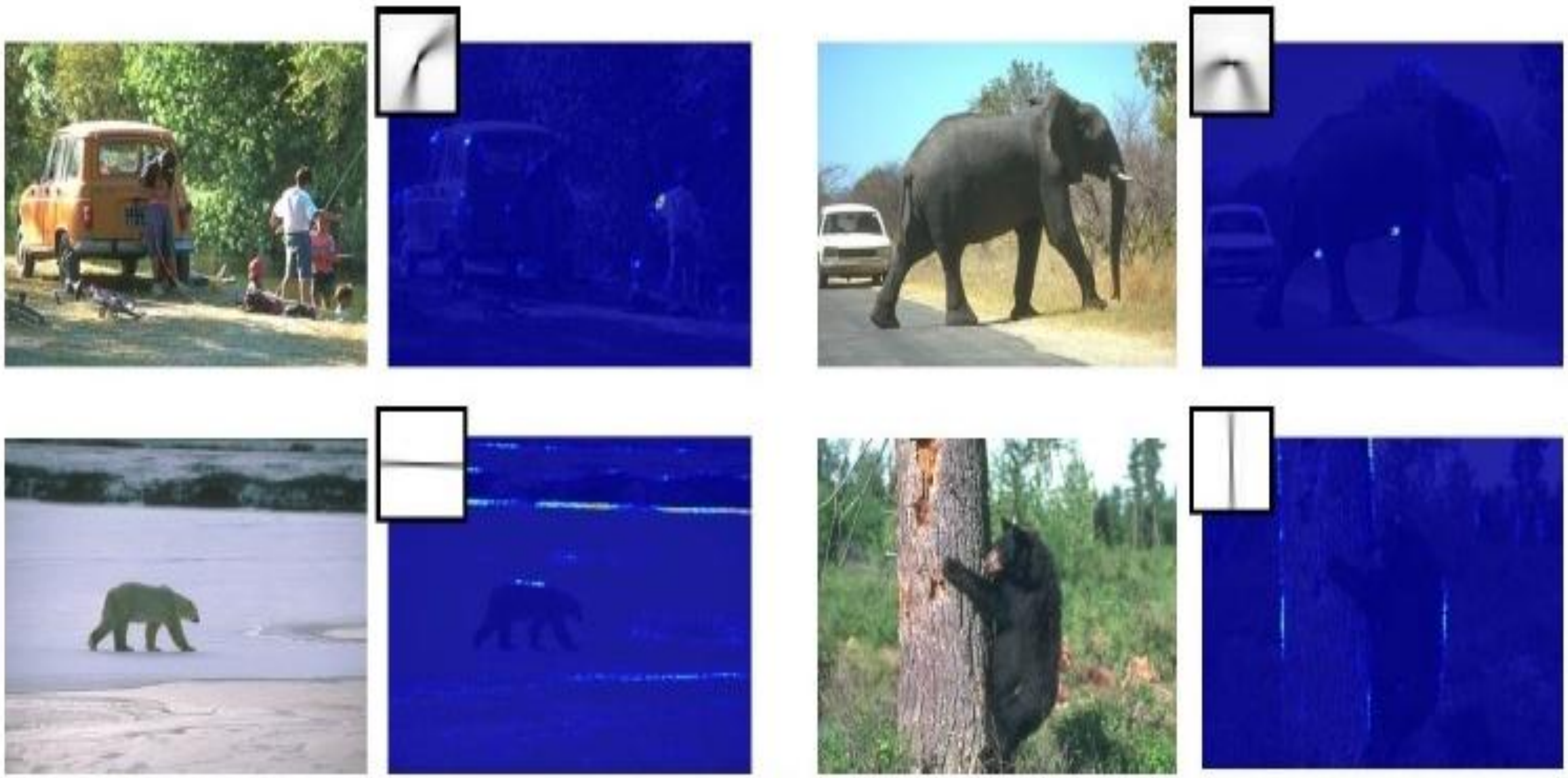


Figure 5. Illustration of the sketch token responses for four tokens. Notice the high selectivity of each sketch token (best viewed in color.)

Results

1. Contour Detection

- estimated probability of the patch's center containing a contour is:

$$e_i = \sum_j t_{ij} = 1 - t_{i0}.$$

- calculate it for each pixel

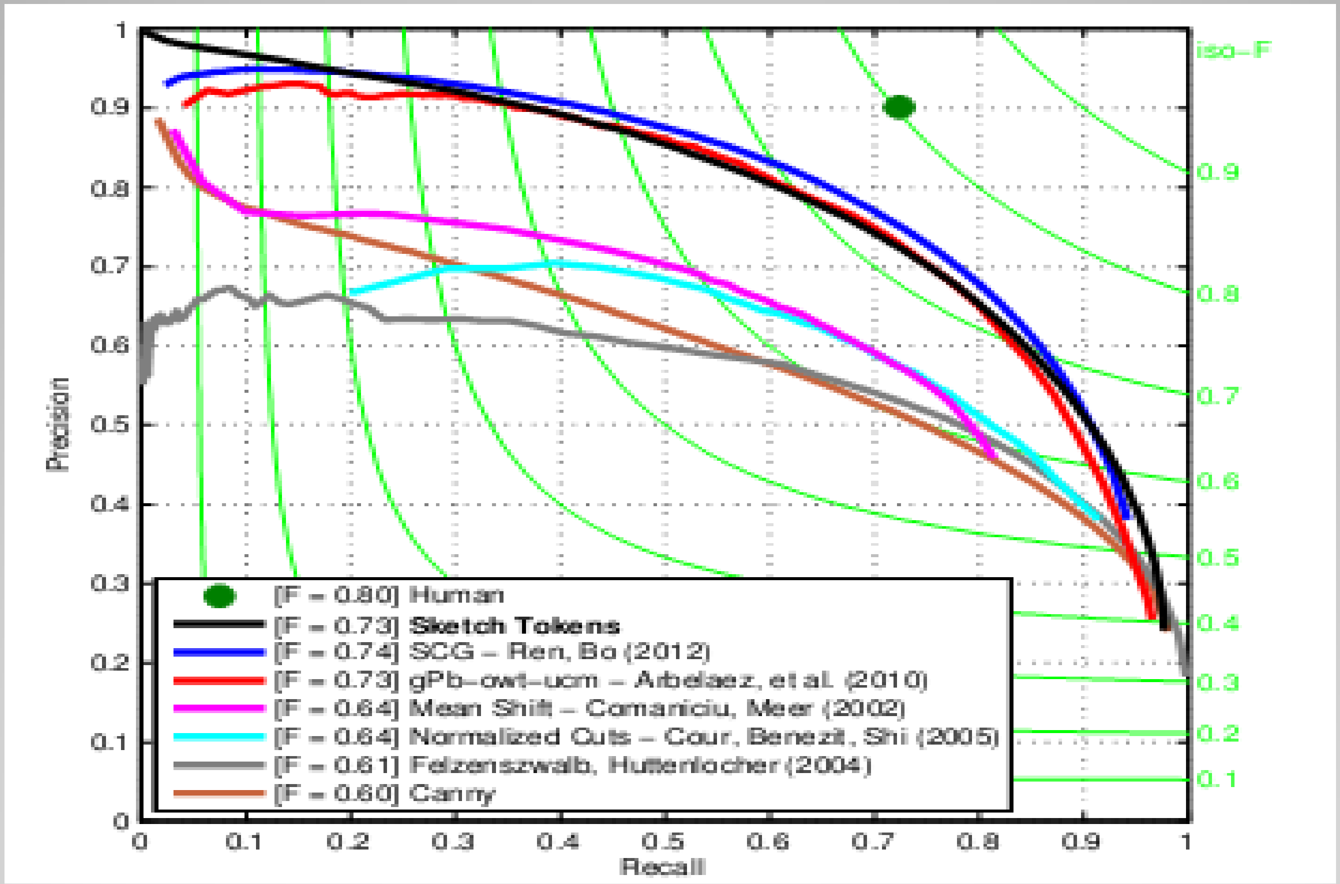
Results

1. Contour Detection

- Results on Berkeley Segmentation Dataset and Benchmark (BSDS500).

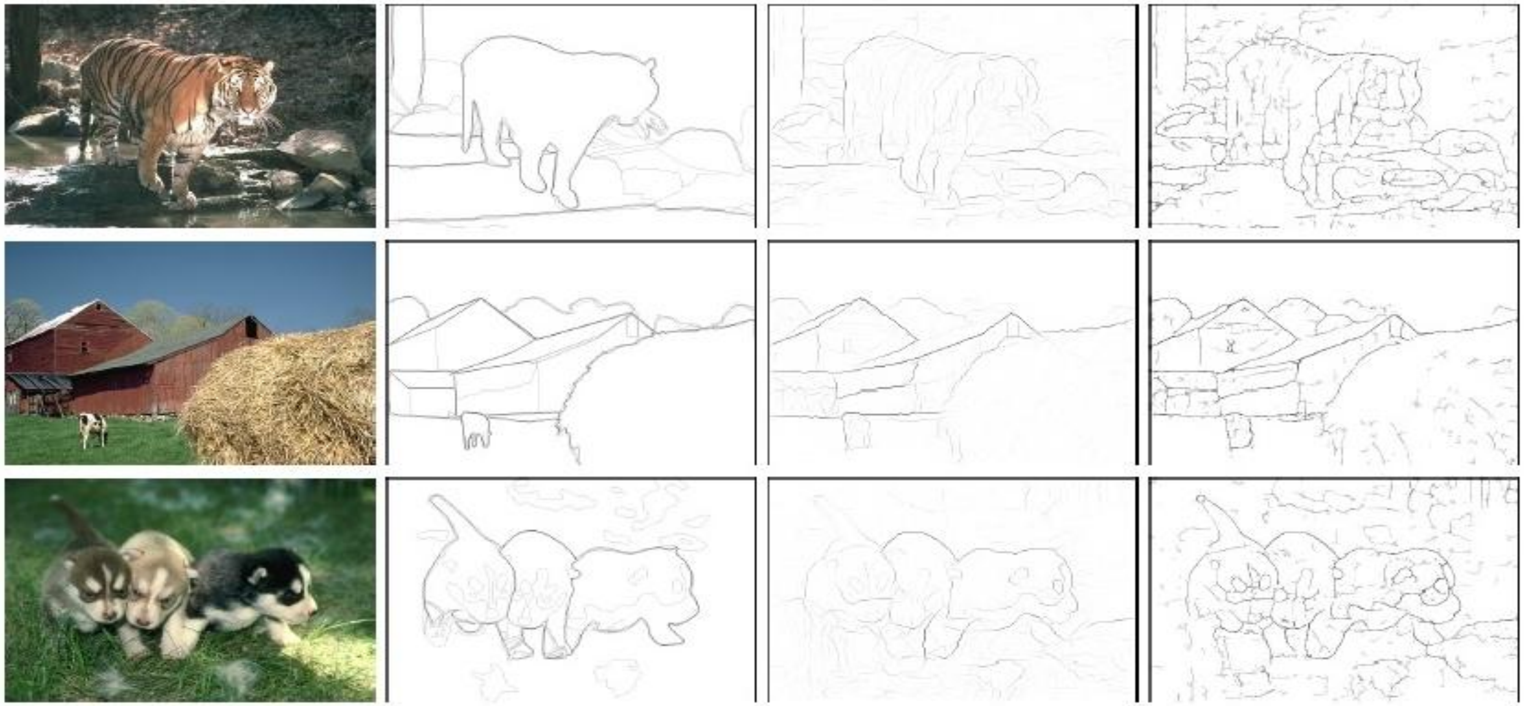
Method	ODS	OIS	AP	Speed
Human	.80	.80	-	-
Canny	.60	.64	.58	1/15 s
Felz-Hutt [12]	.61	.64	.56	1/10 s
gPb (local) [1]	.71	.74	.65	60 s
SCG (local) [24]	.72	.74	.75	100 s
Sketch tokens	**.73**	**.75**	**.78**	**1 s**
gPb (global) [1]	.73	.76	.73	240 s
SCG (global) [24]	.74	.76	.77	280 s

Results



Precision
Recall
iso-F
[F = 0.80] Human
[F = 0.73] Sketch Tokens
[F = 0.74] SCG – Ren, Bo (2012)
[F = 0.73] gPb-owt-ucm – Arbelaez, et al. (2010)
[F = 0.64] Mean Shift – Comaniciu, Meer (2002)
[F = 0.64] Normalized Cuts – Cour, Benezit, Shi (2005)
[F = 0.61] Felzenszwalb, Huttenlocher (2004)
[F = 0.60] Canny

Results



(a) Original image (b) Ground truth (c) SCG [24] (d) Sketch Tokens

Figure 10. Examples of contour detection on the BSDS500 [1]. For Sketch Tokens we define edge strength according to Equation 2 and apply smoothing and standard non-maximal suppression to obtain peak edge responses [3]. Note how our method captures finer details such as the structure of Sydney Opera House on the 1st row and human legs on the 2nd row.

Results

1. Object Detection

- INRIA pedestrian dataset

channels	# channels	miss rate
LUV	3	72.7%
M+O	7	20.7%
LUV+M+O	10	17.2%
ST	151	19.5%
ST+LUV	154	16.5%
ST+LUV+M+O	**161**	**14.7%**

Results

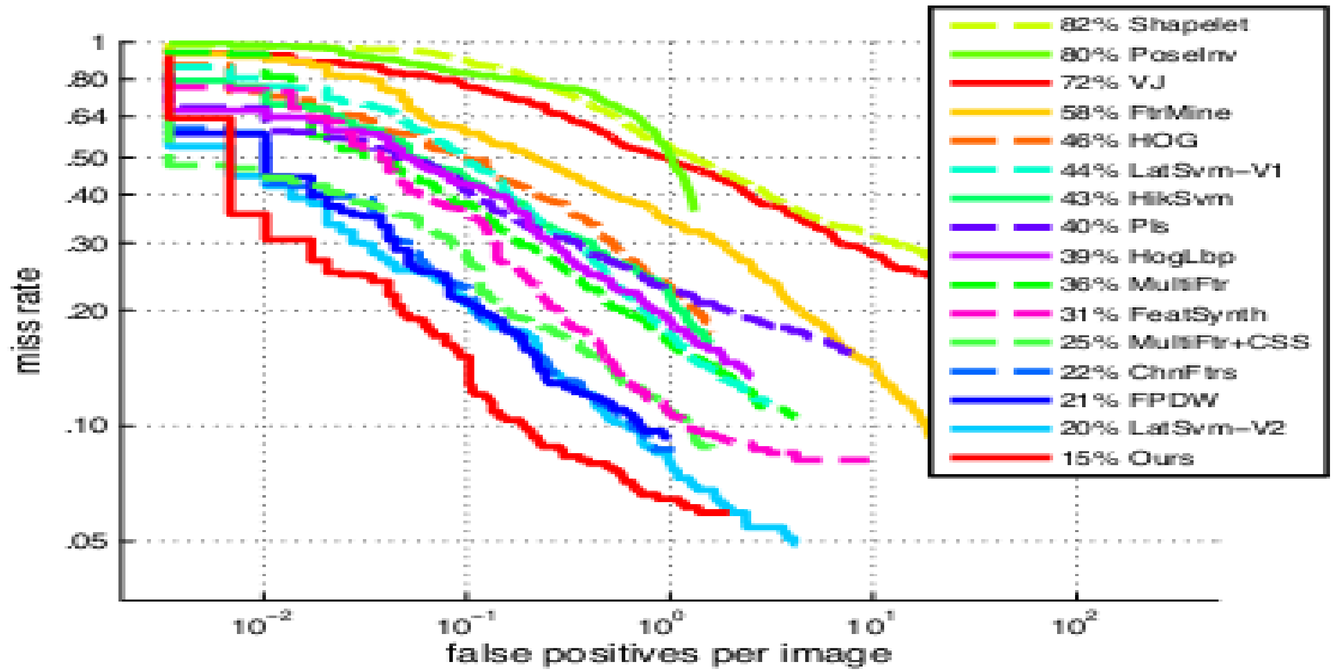


Figure 7. Mean log-average miss rate on the INRIA pedestrian dataset: notice the considerable improvement over previous techniques using our approach. At a 90% detection rate, we achieve a $10\times$ reduction in FPPI over the previous state-of-the-art.

Results

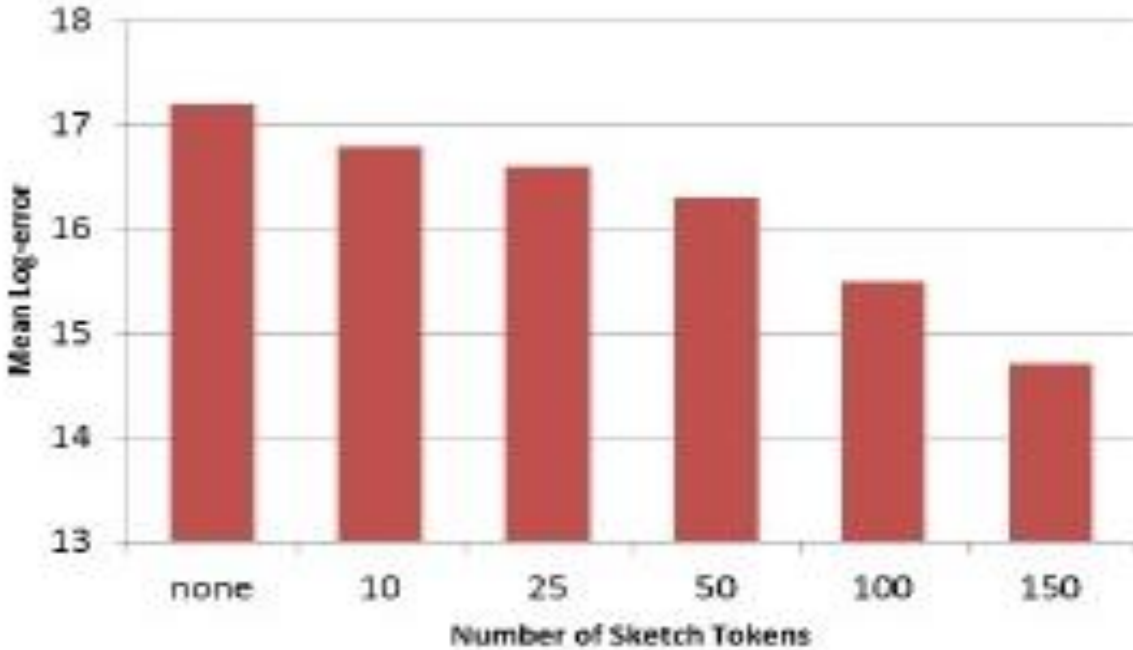


Figure 8. Mean log-error rates on the INRIA dataset using different numbers of sketch tokens. Notice that capturing a large number of varying edge structures leads to a large increase in performance.

Results

1. Object Detection

- PASCAL VOC 2007

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow
HOG	19.7	43.9	2.2	4.8	13.4	36.6	40.2	5.4	10.9	15.7
ST	17.8	41.1	4.8	5.7	11.1	31.9	33.8	5.1	10.8	16.1
ST+HOG	**21.9**	**48.5**	**6.3**	**6.4**	**14.6**	**41.5**	**43.3**	**6.1**	**15.7**	**19.2**

	table	dog	horse	moto	person	plant	sheep	sofa	train	tv
HOG	7.5	2.1	41.9	30.9	23.9	3.4	9.3	14.8	26.9	32.4
ST	7.4	3.1	32.9	27.0	20.9	4.6	8.6	10.4	18.9	26.3
ST+HOG	**14.2**	**3.8**	**46.1**	**34.5**	**30.9**	**8.1**	**15.3**	**18.9**	**30.3**	**36.6**

Table 3. PASCAL 2007 results for linear SVMs: Sketch tokens+HOG outperforms HOG on all classes by 3.8 AP on average.

Results

1. Object Detection

- PASCAL VOC 2007

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow
HOG	**27.9**	56.5	1.9	6.2	21.2	48.2	**52.7**	**7.6**	17.7	21.2
ST+HOG	23.8	**58.2**	**10.5**	**8.5**	**27.1**	**50.4**	52.0	7.3	**19.2**	**22.8**

	table	dog	horse	moto	person	plant	sheep	sofa	train	tv
HOG	14.7	3.0	55.4	42.9	**33.9**	6.0	11.9	21.7	43.2	37.7
ST+HOG	**18.1**	**8.0**	**55.9**	**44.8**	32.4	**13.3**	**15.9**	**22.8**	**46.2**	**44.9**

Table 4. PASCAL 2007 results for DPMs: On average Sketch Tokens+HOG outperformed HOG by 2.5 AP.

Results

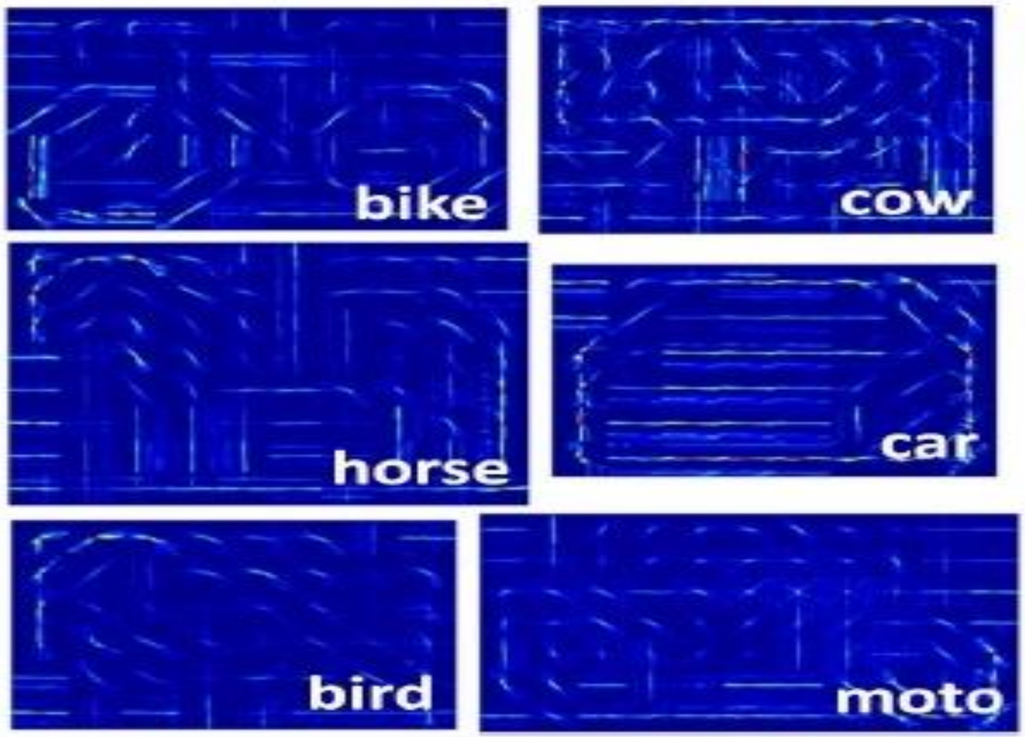


Figure 9. Sketch token weight visualization: We visualize the top 5 sketch tokens multiplied by the learned weight for each cell. Notice the many sketch tokens with rich edge structures that are used.

References

- P. Doll ´ar, Z. Tu, P. Perona, and S. Belongie. Integral channel features. InBMVC, 2009
- S. A. J. Winder, G. Hua, and M. Brown. Picking the best DAISY. InCVPR, 2009
- P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. PAMI, 32(9):1627–1645, 2010

Questions

?