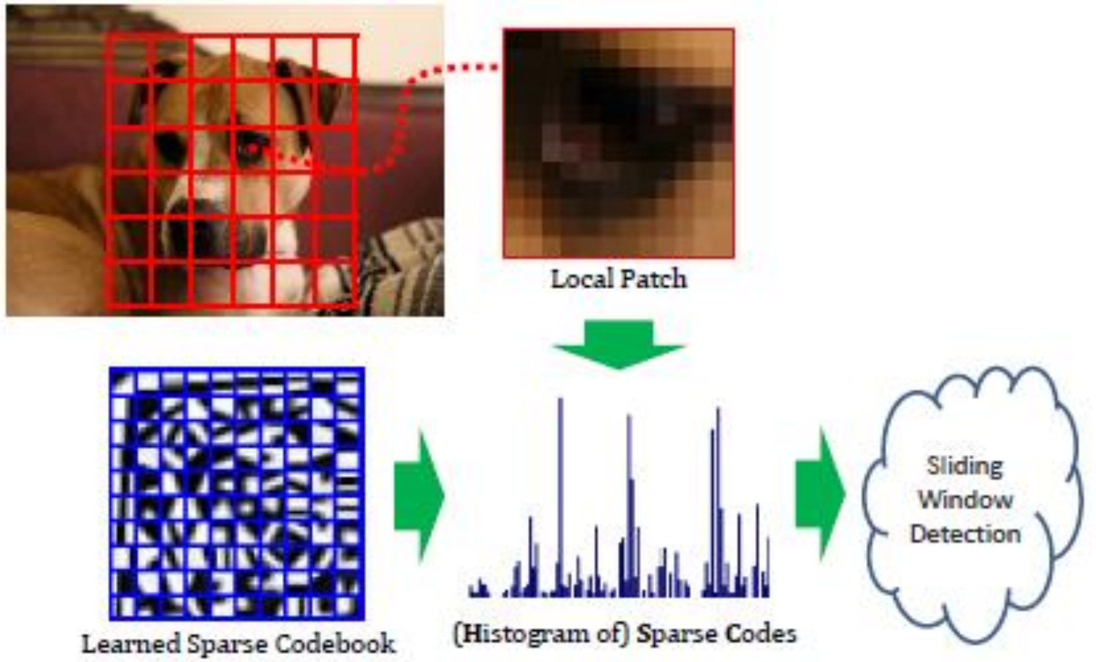
Histograms of Sparse Codes for Object Detection

Xiaofeng Ren (Amazon), Deva Ramanan (UC Irvine)

Presented by Hossein Azizpour

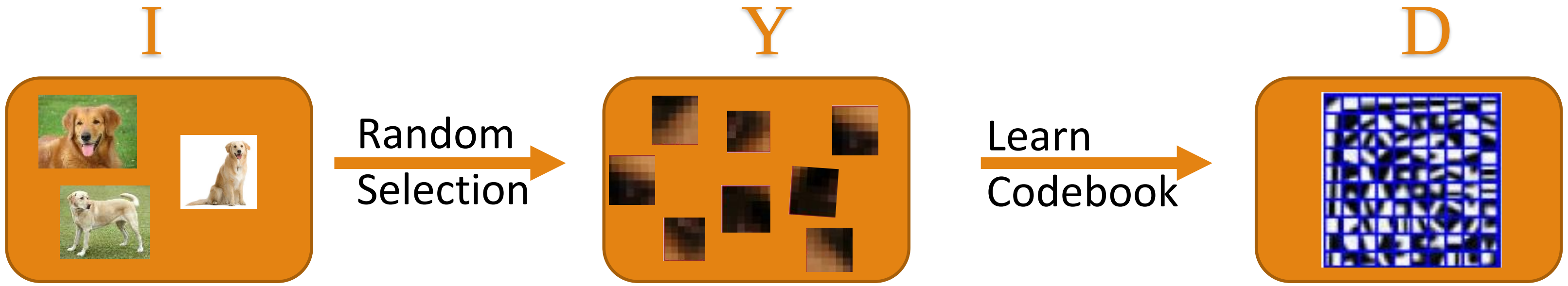
What does the paper do?

- (learning) a new representation
- local histograms of sparse encodings
- replaces HOG…
- (sliding window) detection
- and improves result



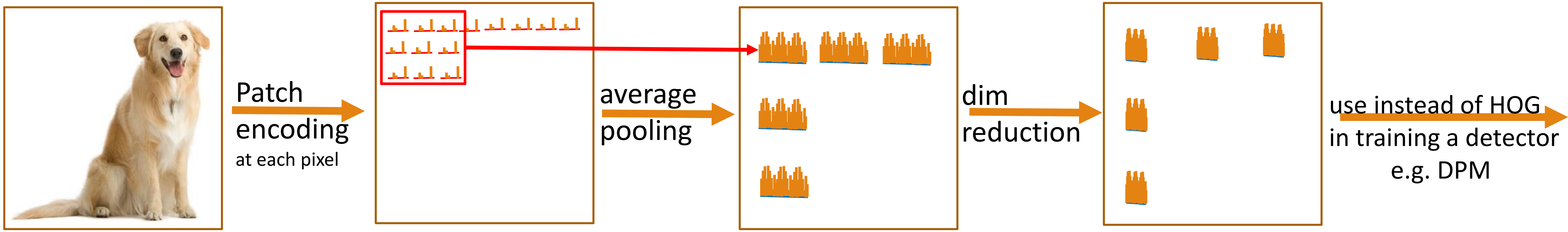
How is the feature extracted? (summary)

- Offline part
  - randomly select a set of n local image intensity patches $Y = [y_1, y_2, \ldots, y_n]$
  - generate a codebook $\mathrm{D} = [d_1, d_2, \ldots, d_m]$ able to reconstruct Y:
  - $Y = DX$ where $\mathrm{X} = [x_1, x_2, \ldots, x_n]$ is the new encoding of the patches



How is the feature extracted? (summary)

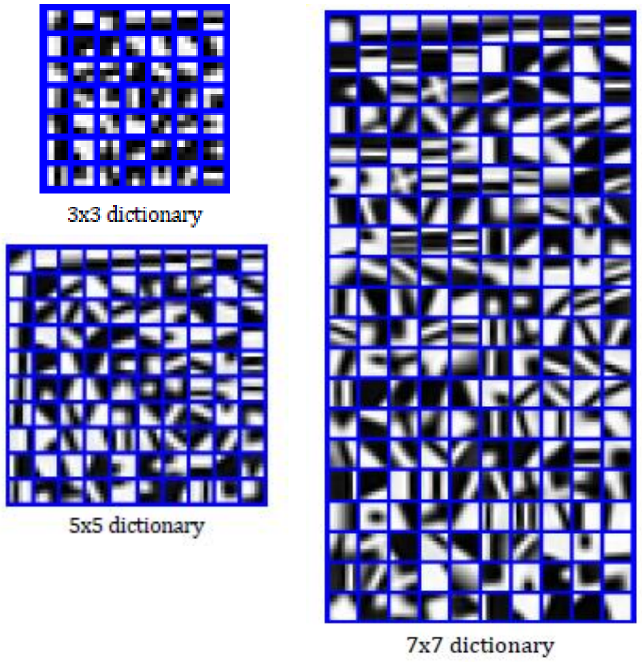
- Online part
  - compute the encodings for patches around each *pixel*.
  - do average pooling over fixed regions of pixels
  - (optionally) reduce dimensions using a projection matrix
  - post process the extracted feature (e.g. L2 normalization)



Generating the Codebook

- Given a set of image patches $Y = [y_1, \cdots, y_n]$
- Jointly find
  - dictionary $D = [d_1, \cdots, d_m]$
  - ***sparse*** code matrix $X = [x_1, \cdots, x_n]$
- K is a predefined sparsity level
- ***minimize residual*** rather than a complete reconstruction ($Y = DX + \epsilon$)
- solve for X and D in the following objective using K-SVD

$$\min_{D,X} \|Y - DX\|_F^2 \ s.t. \ \forall i, \ \|x_i\|_0 \le K$$



Generating the Codebook (K-SVD)

- alternate between the estimation of X and D
- given a dictionary D, use Orthogonal Matching Pursuit (OMP) to find the codes X
  - a greedy method to select K codes
- given the codes X, update dictionary D using SVD

$$\min_{D,X} \|Y - DX\|_F^2 \; s.t. \; \forall i, \; \|x_i\|_0 \leq K$$

Generating the Codebook (K-SVD)

- K-SVD as generalization of K-means
- Alternates between estimation of D and X
- A good approximate solver of the optimization

Task: Find the best possible codebook to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ by nearest neighbor, by solving

$$\min_{\mathbf{C},\mathbf{X}} \left\{\|\mathbf{Y}-\mathbf{C}\mathbf{X}\|_F^2\right\} \quad \text{subject to} \quad \forall i,\ \mathbf{x}_i = \mathbf{e}_k \text{ for some } k.$$

Initialization : Set the codebook matrix $\mathbf{C}^{(0)} \in \mathbb{R}^{n\times K}$. Set $J=1$.
Repeat until convergence (use stop rule):

- *Sparse Coding Stage*: Partition the training samples $\mathbf{Y}$ into K sets

$$(R_1^{(J-1)},\ R_2^{(J-1)},\ \ldots,\ R_K^{(J-1)}),$$

each holding the sample indices most similar to the column $\mathbf{c}_k^{(J-1)}$,

$$R_k^{(J-1)} = \left\{ i \mid \forall_{l\neq k},\ \|\mathbf{y}_i - \mathbf{c}_k^{(J-1)}\|_2 < \|\mathbf{y}_i - \mathbf{c}_l^{(J-1)}\|_2 \right\}.$$

- *Codebook Update Stage*: For each column k in $\mathbf{C}^{(J-1)}$, update it by

$$\mathbf{c}_k^{(J)} = \frac{1}{|R_k|} \sum_{i\in R_k^{(J-1)}} \mathbf{y}_i.$$

- Set $J = J+1$.

Fig. 1. The K-means algorithm.

Task: Find the best dictionary to represent the data samples $\{y_i\}_{i=1}^N$ as sparse compositions, by solving

$$\min_{\mathbf{D},\mathbf{X}} \left\{\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2\right\} \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \le T_0.$$

Initialization : Set the dictionary matrix $\mathbf{D}^{(0)} \in \mathbb{R}^{n\times K}$ with ℓ^2 normalized columns. Set $J=1$.
Repeat until convergence (stopping rule):

- *Sparse Coding Stage*: Use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$, by approximating the solution of

$$i = 1,2,\ldots,N, \quad \min_{\mathbf{x}_i} \left\{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2\right\} \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \le T_0.$$

- *Codebook Update Stage*: For each column $k = 1,2,\ldots,K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Define the group of examples that use this atom, $\omega_k = \{i \mid 1 \le i \le N,\ \mathbf{x}_T^k(i) \neq 0\}$.
  - Compute the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j\neq k} \mathbf{d}_j \mathbf{x}_T^j.$$

  - Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to ω_k, and obtain $\mathbf{E}_k^R$.
  - Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U}\boldsymbol{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\boldsymbol{\Delta}(1,1)$.
- Set $J = J+1$.

$$\min_{D,X} \|Y - DX\|_F^2 \ s.t.\ \forall i,\ \|x_i\|_0 \le K$$

Generating the Codebook (OMP)

- Greedily selects the codes for the encodings
- Updates all the coefficients
- *There is a fast version Called* ***Batch OMP***

$$\min_{\alpha \in \mathbb{R}^p} \|\mathbf{y} - \mathbf{D}\alpha\|_2^2 \text{ s.t. } \|\alpha\|_0 \leq L$$

1: $\Gamma = \emptyset$.
2: **for** *iter* $= 1, \dots, L$ **do**
3: Select the atom which most reduces the objective

$$\hat{\imath} \leftarrow \arg\min_{i \in \Gamma^C} \left\{ \min_{\alpha'} \|\mathbf{y} - \mathbf{D}_{\Gamma \cup \{i\}} \alpha'\|_2^2 \right\}$$

4: Update the active set: $\Gamma \leftarrow \Gamma \cup \{\hat{\imath}\}$.
5: Update the residual (orthogonal projection)

$$\mathbf{r} \leftarrow (\mathbf{I} - \mathbf{D}_\Gamma (\mathbf{D}_\Gamma^T \mathbf{D}_\Gamma)^{-1} \mathbf{D}_\Gamma^T)\mathbf{y}.$$

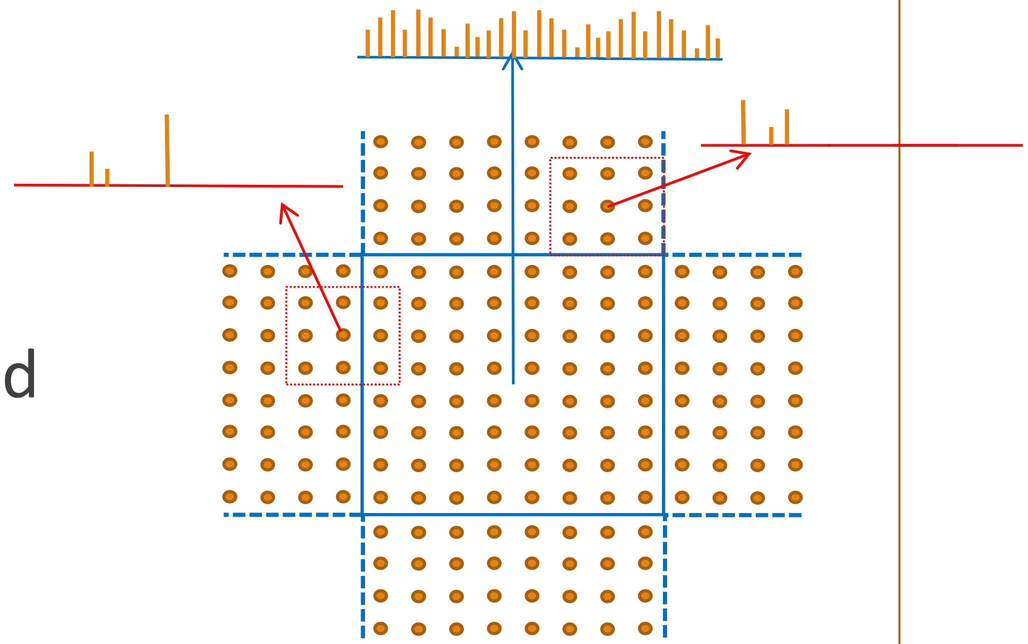
6: Update the coefficients

$$\alpha_\Gamma \leftarrow (\mathbf{D}_\Gamma^T \mathbf{D}_\Gamma)^{-1} \mathbf{D}_\Gamma^T \mathbf{y}.$$

7: **end for**

Feature Extraction (Binning)

- 8x8 cells
- soft binning (bilinear interpolation)
- 4 – neighborhood
- average pooling on 16x16 neighborhood
- absolute value of sparse codes
- $F = (|x_1|, |x_2|, \ldots, |x_m|)$
- contrast sensitive features $[|x_i|, \max(x_i, 0), \max(-x_i, 0)]$



Feature Extraction (post processing)

- L2 normalization
- power transform $\bar{F} = F^{\alpha}$ (element-wise)

Feature Extraction (example)

- 3m dimensional feature
- HOG can be replaced

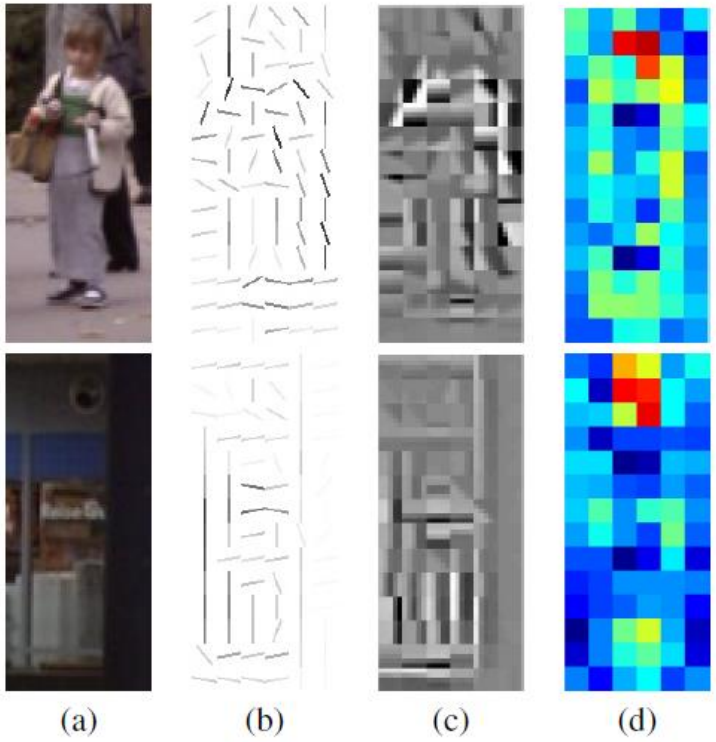


Figure 3: Visualizing HSC vs HOG: (a) image; (b) dominant orientation in HOG, weighted by gradient magnitude; (c) dominant codeword in HSC, weighted by histogram value; (d) per-cell responses of HSC features when multiplied with a linear SVM model trained on INRIA (colors are on the same scale).

Feature Extraction (dim reduction)

- Too long feature vectors -> slow training and testing
- (somewhat) supervised dimensionality reduction
- Train root filters $(w_1, w_2, \dots, w_q)$ for different classes/subclasses using original 3m dimensional features.
- $w_i = w_i^1 || w_i^2 || \dots || w_i^C$ where w_i^c is the corresponding part of cell c.
- stack all cells of all weight vectors to produce matrix $W = \begin{bmatrix} {w_1^1}^T \\ \vdots \\ {w_i^c}^T \\ \vdots \\ {w_q^C}^T \end{bmatrix}$
- Do PCA dimensionality reduction on $W' = WP, \quad W = USP^T,$
- Use the projection matrix to transform original cell features to lower dimensions $F' = FP$

Training Detector

- Deformable Parts Model (DPM)
  - Root only
  - Root with Parts

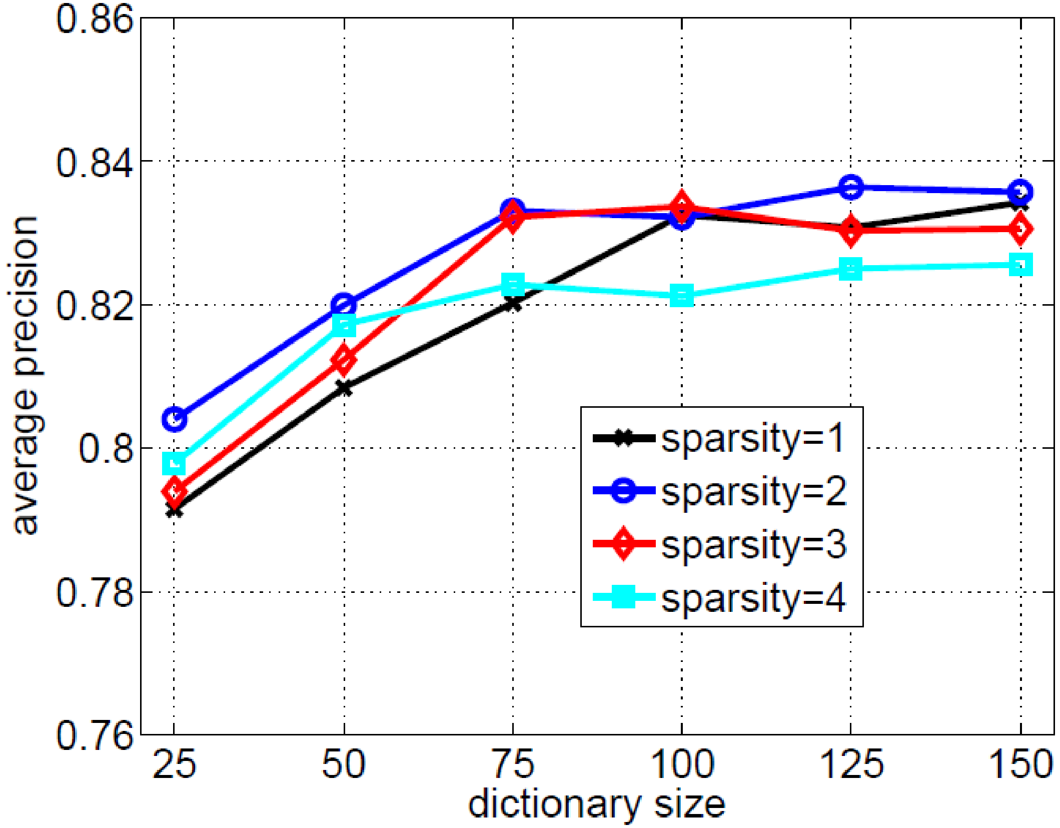
$$\sum_{i \in V} w_i^m \phi(x, p_i) + \sum_{ij \in E} w_{ij}^m \psi(p_i, p_j) + b_m$$

- fixing part latency
  - using originally trained DPMs
  - to make training faster!

$$\begin{aligned} \underset{\beta, \xi_n \geq 0}{\operatorname{argmin}} \quad & \frac{1}{2}\beta \cdot \beta + C \sum_n \xi_n \\ \text{s.t.} \quad \forall n \in \text{pos} \quad & \beta \cdot \Phi(I_n, z_n) \geq 1 - \xi_n \\ \forall n \in \text{neg}, \forall z \quad & \beta \cdot \Phi(I_n, z) \leq -1 + \xi_n \end{aligned}$$

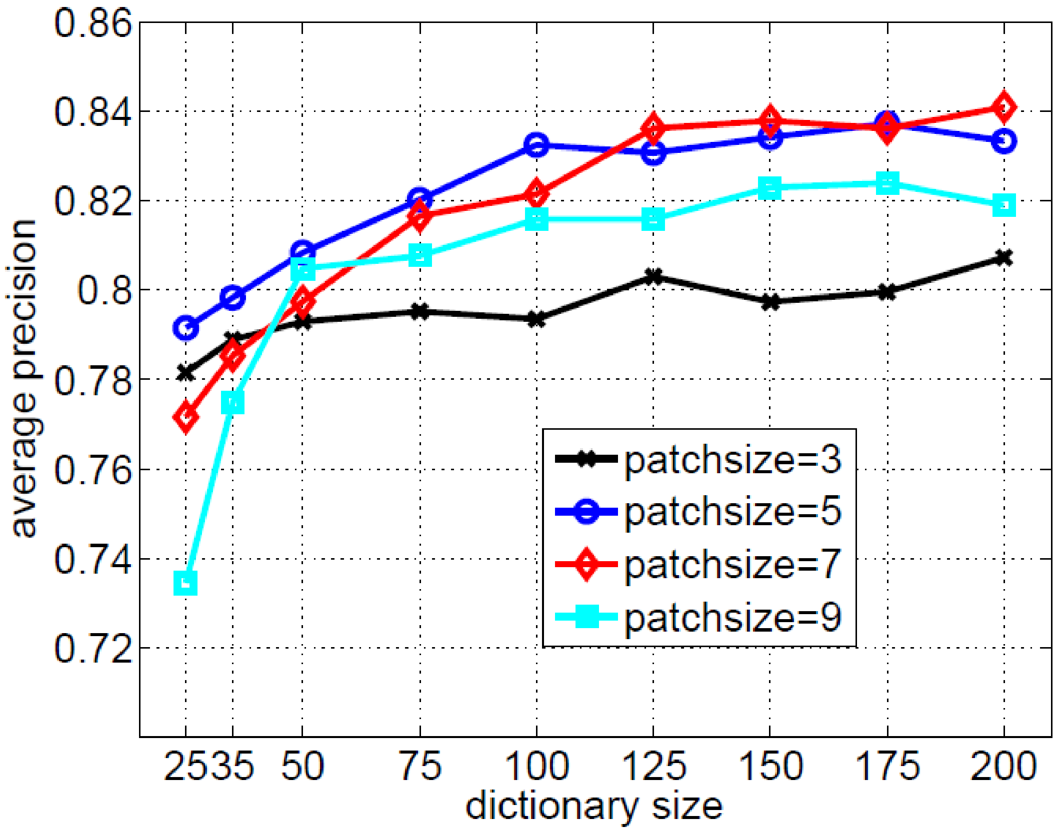
Experiments (Different Parameters)

- INRIA pedestrian
- root-only
- HOG AP = 80.2%
- sparsity level vs Dictionary Size
- fix K = 1
- histogram of sparse codes



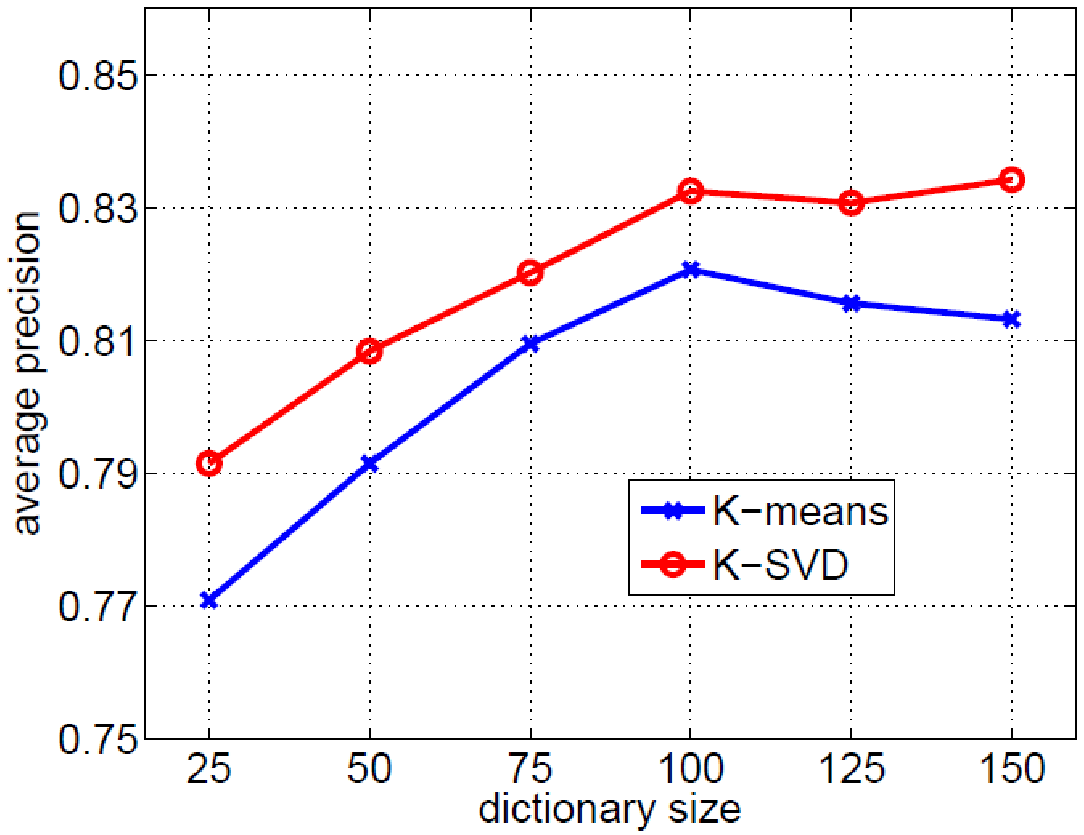
Experiments (Different Parameters)

- Patch size vs Dictionary size
- K-medoid clustering wouldn't gain performance larger than 3x3
- Fixed to 5x5



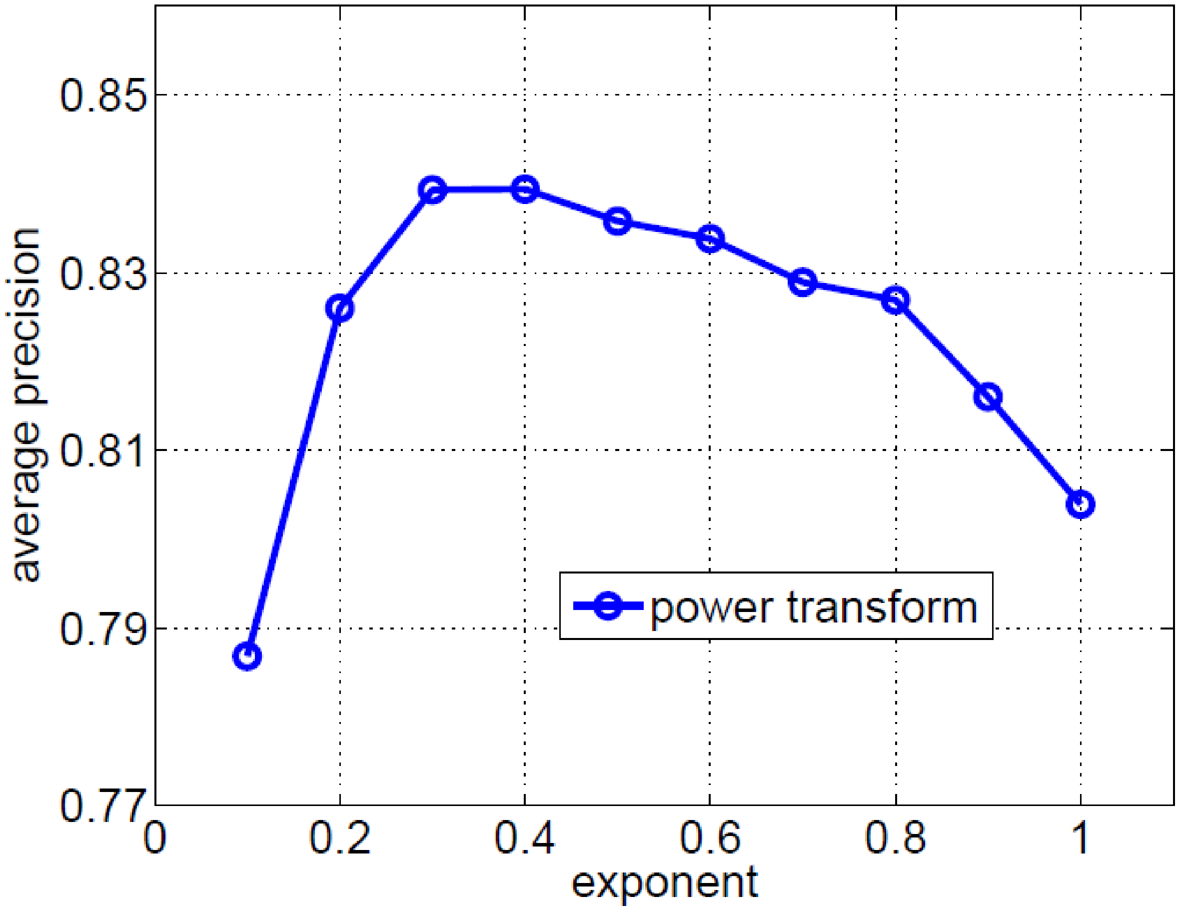
Experiments (Different Parameters)

- K-SVD vs K-means
- activated code can have a weight other than 1
- possible change of sign



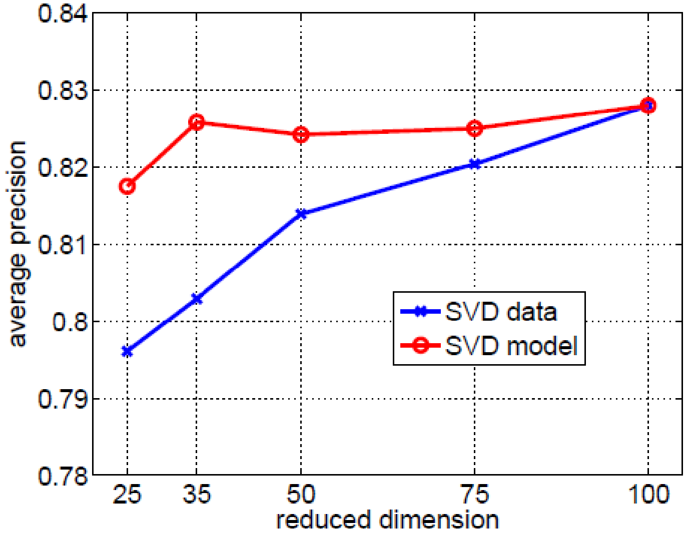
Experiments (Different Parameters)

- Power transform
- Fixed at 0.25
- double helinger kernel!

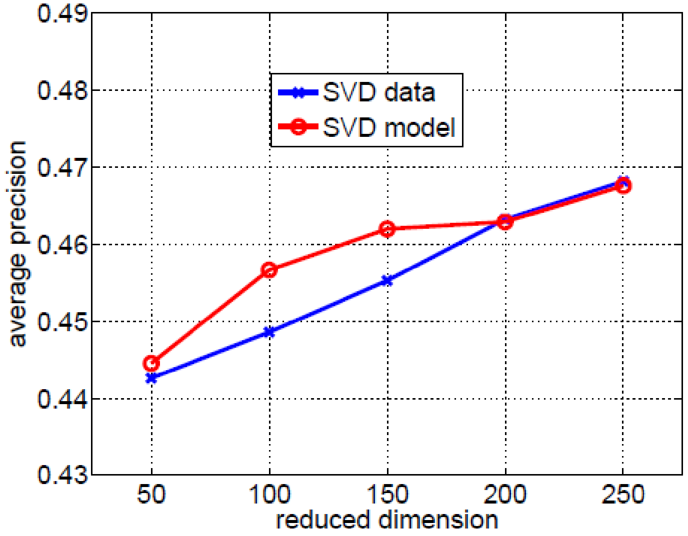


Experiments (Different Parameters)

- Supervised PCA (on models) vs PCA on data
- PASCAL 4 classes
  - bus
  - cat
  - diningtable
  - Motorbike
- More effective for person
- fixed at 100



INRIA



PASCAL

Final Experiments

- INRIA root only

HOG	HSC_{3x3}	HSC_{5x5}	HSC_{7x7}	[14]
80.2%	80.7%	84.0%	84.9%	84.9%

- PASCAL root only

	aero	bike	bird	boat	bttl	bus	car	cat	chair	cow	table	dog	hors	mbik	prsn	plnt	shep	sofa	train	tv	$\overline{\text{avg}}$
HOG	20.5	47.7	9.2	11.3	18.3	35.4	40.8	4.0	12.2	23.4	11.2	2.6	41.0	30.3	21.0	6.6	11.8	16.0	31.5	32.5	21.4
HSC	25.3	49.2	6.2	15.4	24.0	44.3	45.6	12.0	15.6	27.7	16.1	10.8	43.3	42.7	28.5	10.8	20.9	25.1	34.4	39.8	26.9
Δ_{HSC}	+4.7	+1.5	-3.0	+4.0	+5.6	+8.9	+4.9	+8.0	+3.4	+4.3	+4.9	+8.2	+2.3	+12.5	+7.5	+4.2	+9.1	+9.2	+2.8	+7.3	+5.5
[14]	25.2	50.2	5.8	11.8	17.2	41.4	43.6	3.5	15.9	21.0	15.6	7.9	44.1	34.8	30.3	9.9	14.6	18.4	36.4	33.7	24.1

(a) Root-only models: HOG, HSC, their difference Δ_{HSC} (HSC-HOG); and DPM [14]

- PASCAL with parts

	aero	bike	bird	boat	bttl	bus	car	cat	chair	cow	table	dog	hors	mbik	prsn	plnt	shep	sofa	train	tv	$\overline{\text{avg}}$
HOG	30.3	56.4	9.7	15.6	23.2	49.1	51.1	14.9	19.6	21.6	19.6	10.7	56.0	47.3	40.0	12.8	16.7	27.9	41.0	39.5	30.1
HSC	32.2	58.3	11.5	16.3	30.6	49.9	54.8	23.5	21.5	27.7	34.0	13.7	58.1	51.6	39.9	12.4	23.5	34.4	47.4	45.2	34.3
Δ_{HSC}	+1.9	+1.9	+1.8	+0.7	+7.4	+0.8	+3.7	+8.7	+1.9	+6.1	+14.3	+3.0	+2.2	+4.2	-0.1	-0.4	+6.8	+6.5	+6.4	+5.7	+4.2
[14]	30.7	58.9	10.4	14.4	24.8	49.0	54.1	11.1	20.6	25.3	25.2	11.0	58.5	48.4	41.3	12.1	15.5	34.4	43.4	39.0	31.4

(b) Part-based models, with dimension reduction

Visualizing HSC with Reconstructions

(1) Image

(2) HSC

(3) HOG

Courtesy of Carl Voldrick et al 2012 “Inverting and Visualizing Features”

Some detections…

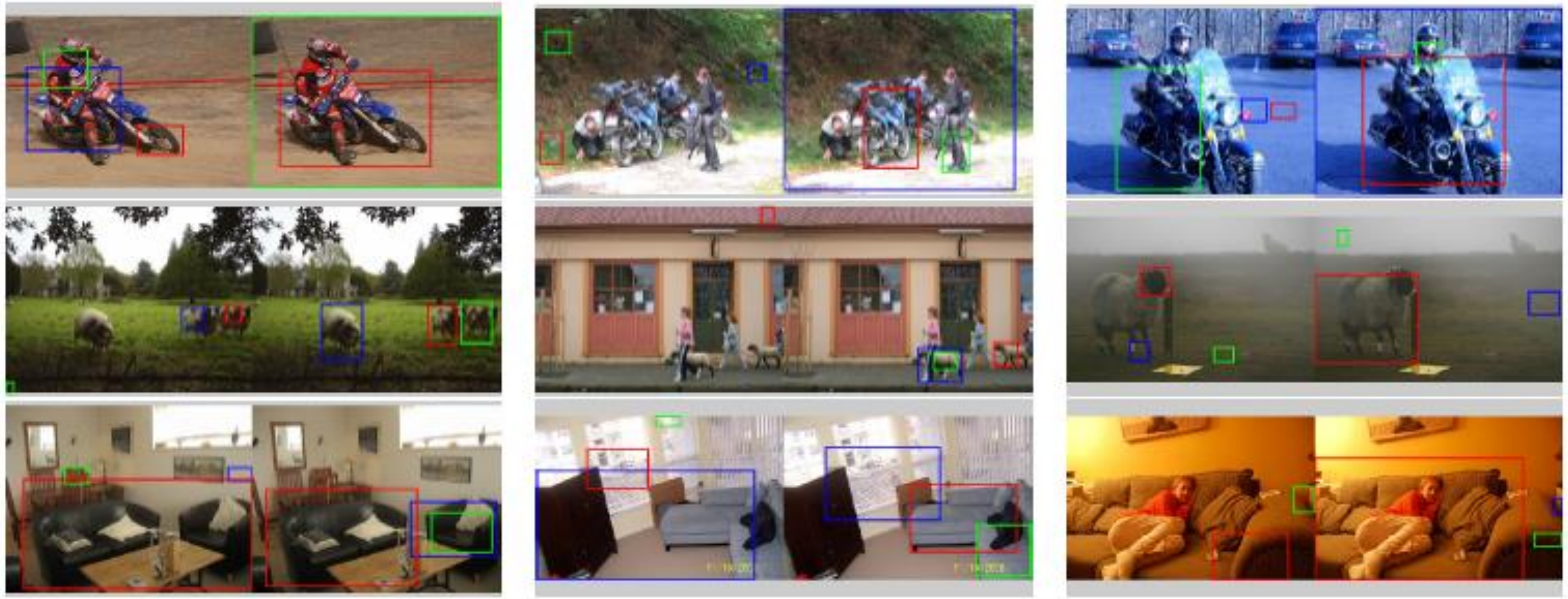


Figure 6: A few examples of HOG (left) vs HSC (right) based detection (root-only), showing top three candidates (in the order of red, green, blue). HSC behaves differently than HOG and tends to have different modes of success (and failure).

Conclusions

- we can easily? go beyond hand crafted HOG by a sort of feature learning
- deep learning
- primitive shape codes might work better than simple gradient orientation
- PCA on model instead of data seems promising