

Introduction to Computer Vision

Kaveh Fathian

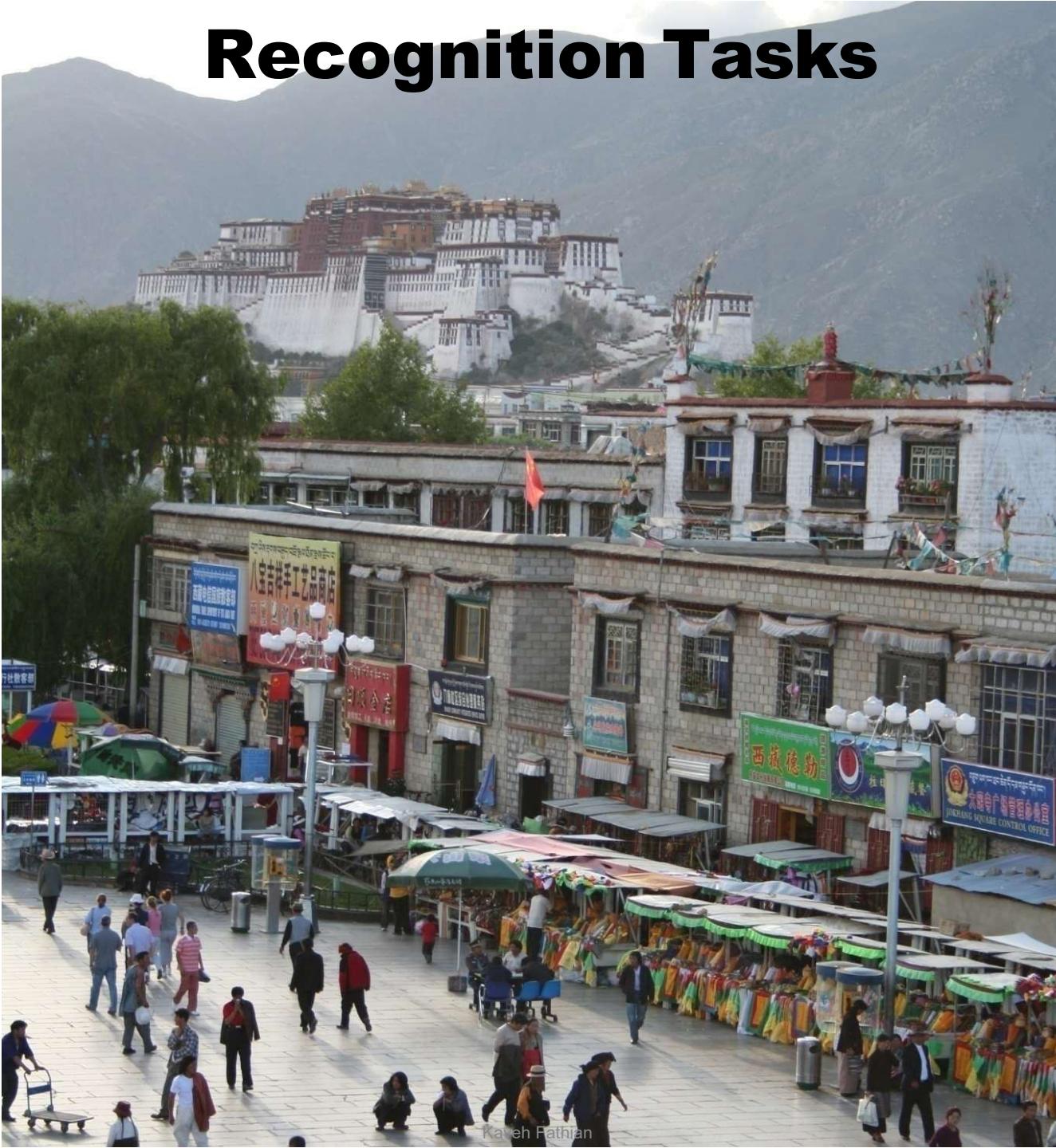
Assistant Professor

Computer Science Department

Colorado School of Mines

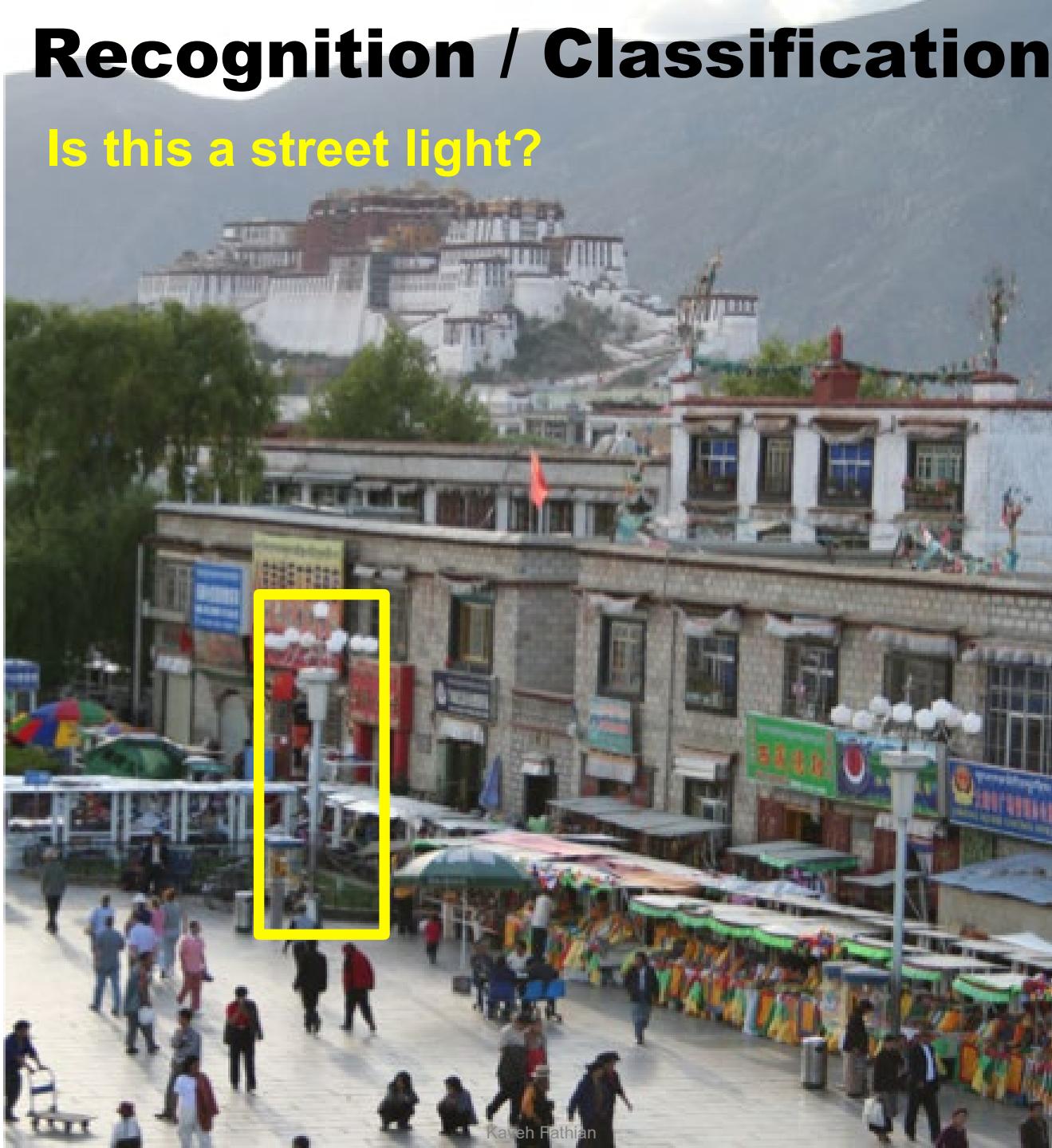
Lecture 15

Recognition Tasks



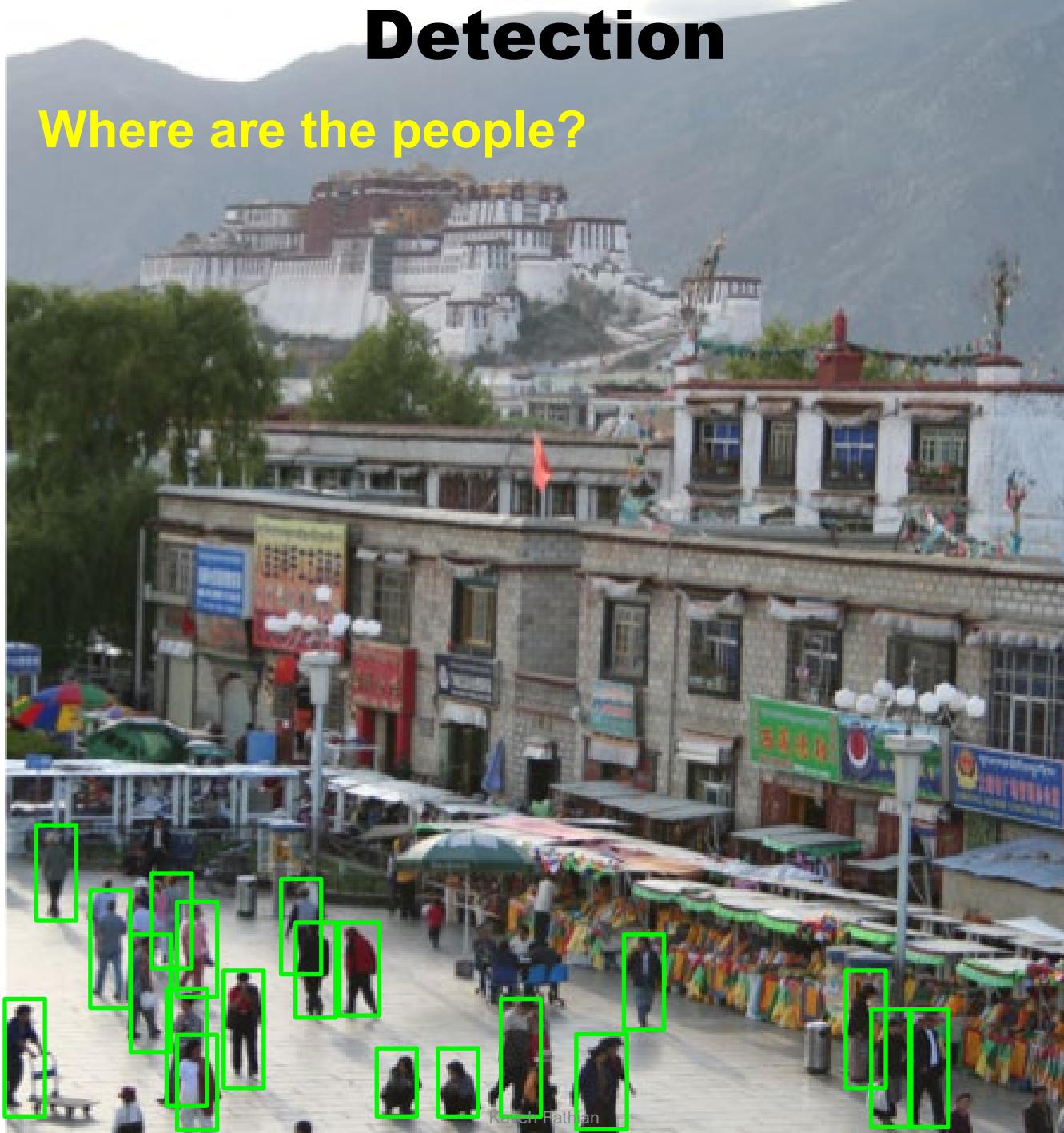
Recognition / Classification

Is this a street light?



Detection

Where are the people?



Identification

Is that Potala palace?



Sky Semantic Segmentation

What's in the scene?



Scene Categorization

What type of scene is it?

Outdoor

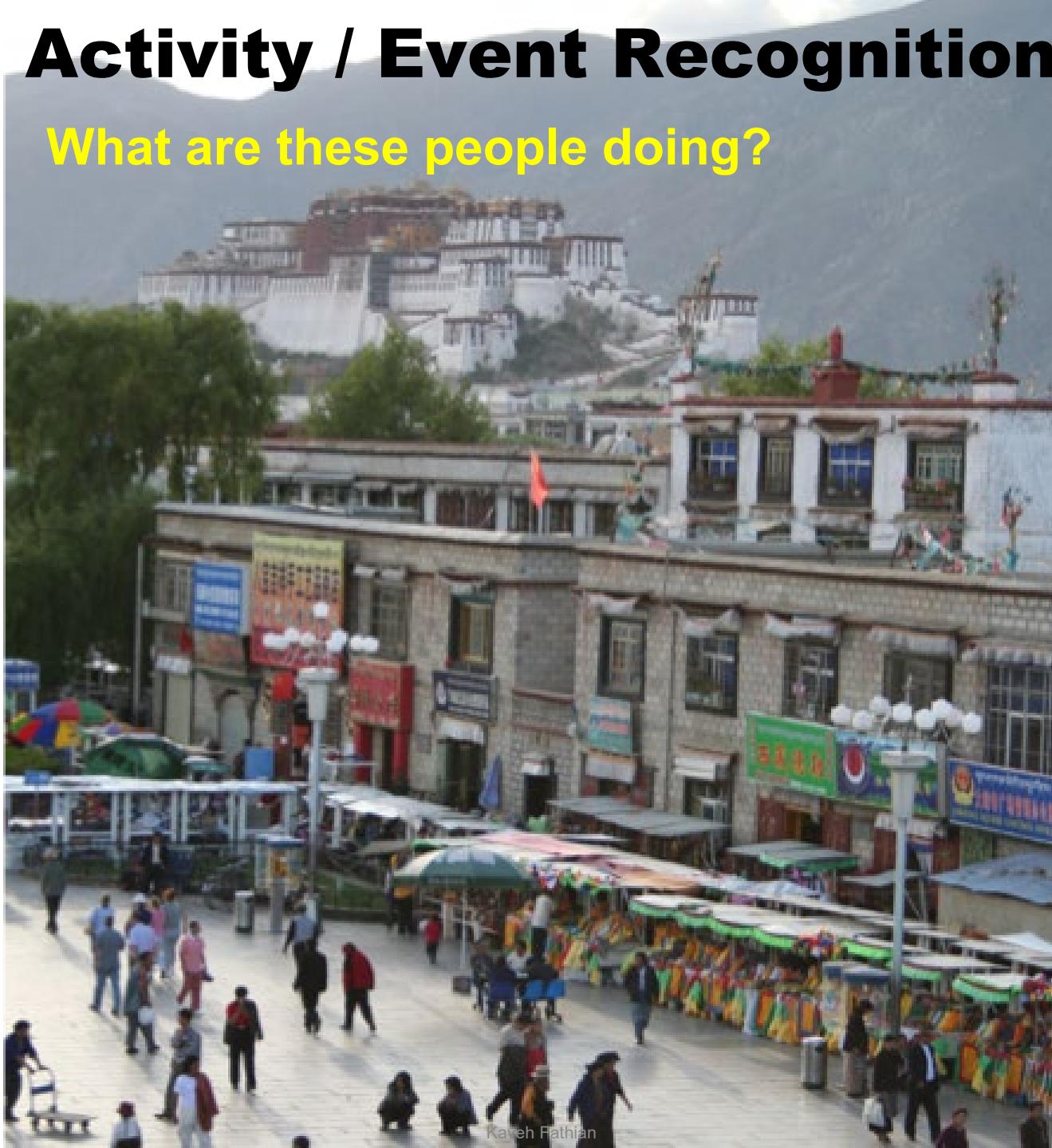
Marketplace

City



Activity / Event Recognition

What are these people doing?



Object Recognition: Is it really so hard?

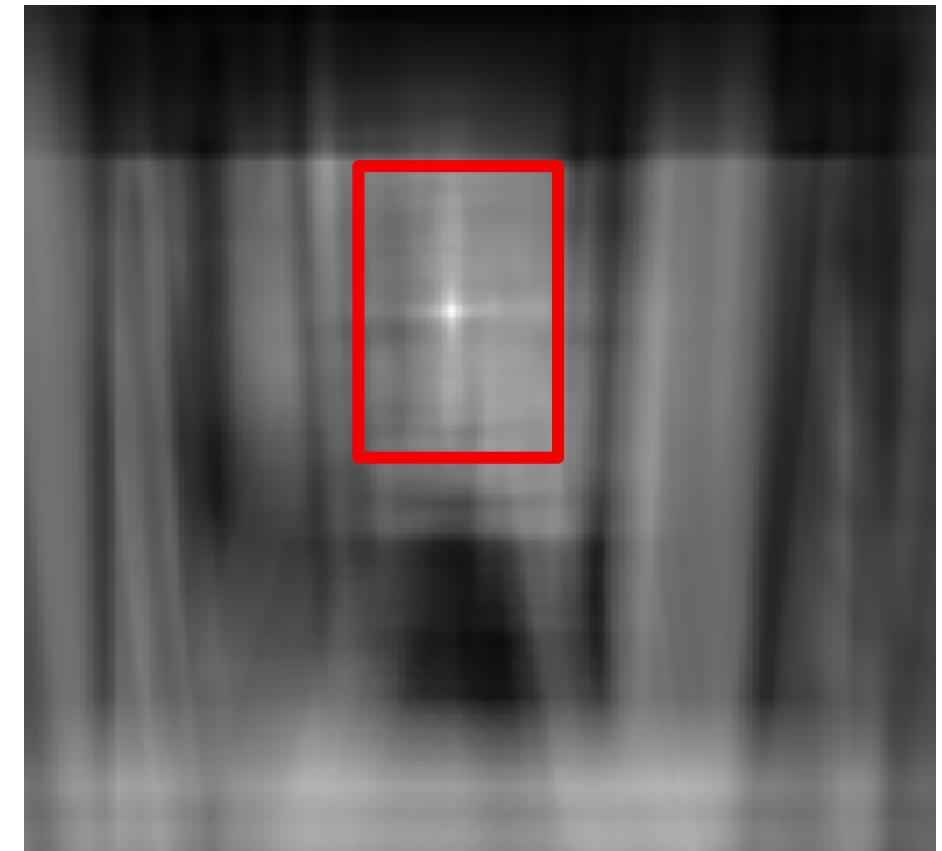
This is a chair



Find the chair in this image



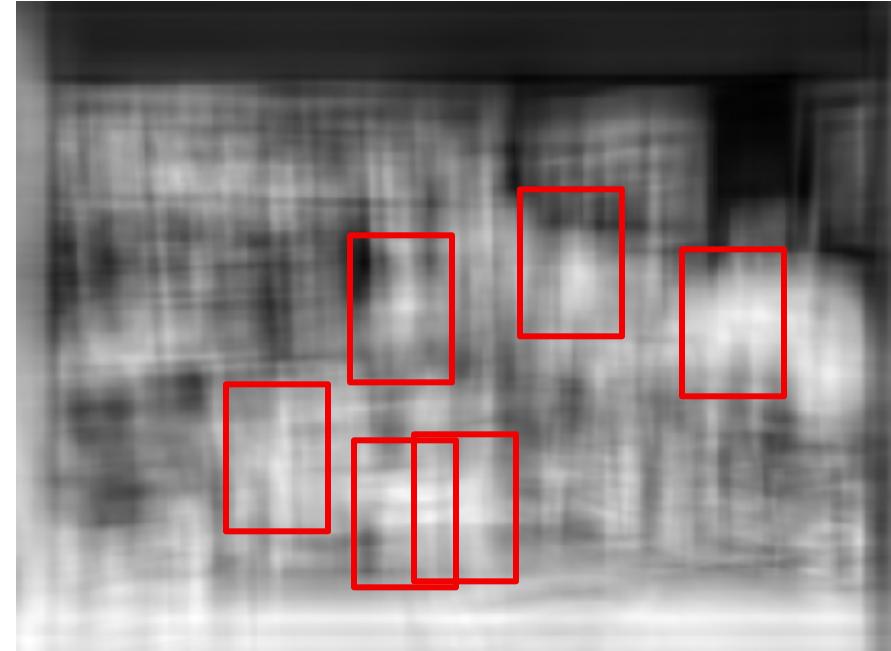
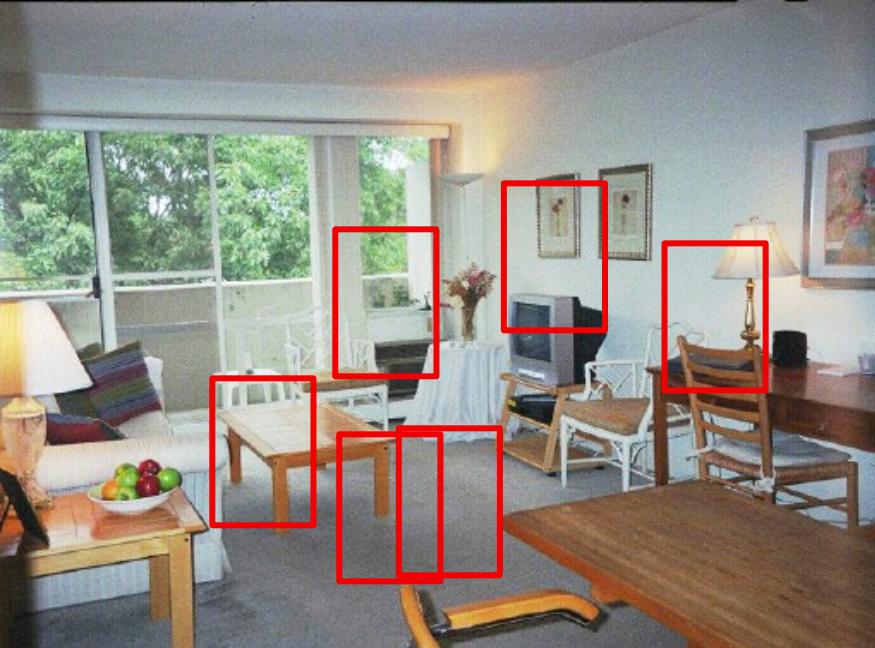
Output of normalized correlation



Object Recognition: Is it really so hard?



Find the chair in this image



Pretty much garbage

Simple template matching is not going to make it

A “popular method is that of template matching, by point to point correlation of a model pattern with the image pattern. These techniques are inadequate for three-dimensional scene analysis for many reasons, such as occlusion, changes in viewing angle, and articulation of parts.” Nivatia & Binford, 1977.

And it can get a lot harder...



Brady, M. J., & Kersten, D. (2003). Bootstrapped learning of novel objects. *J Vis*, 3(6), 413-422

Let's ask ChatGPT...

ChatGPT



Explain to me what is in this image?

This image shows the trunk of a tree, but it contains a well-camouflaged animal blending almost perfectly with the bark. If you look closely at the center of the image, slightly toward the left, you can spot a lizard — specifically, a **mossy leaf-tailed gecko (Uroplatus sikorae)**. This species is known for its remarkable camouflage that mimics tree bark, moss, and lichen. Its body shape, coloration, and texture make it almost indistinguishable from the tree surface, which helps it avoid predators and ambush prey in its natural habitat (typically Madagascar).

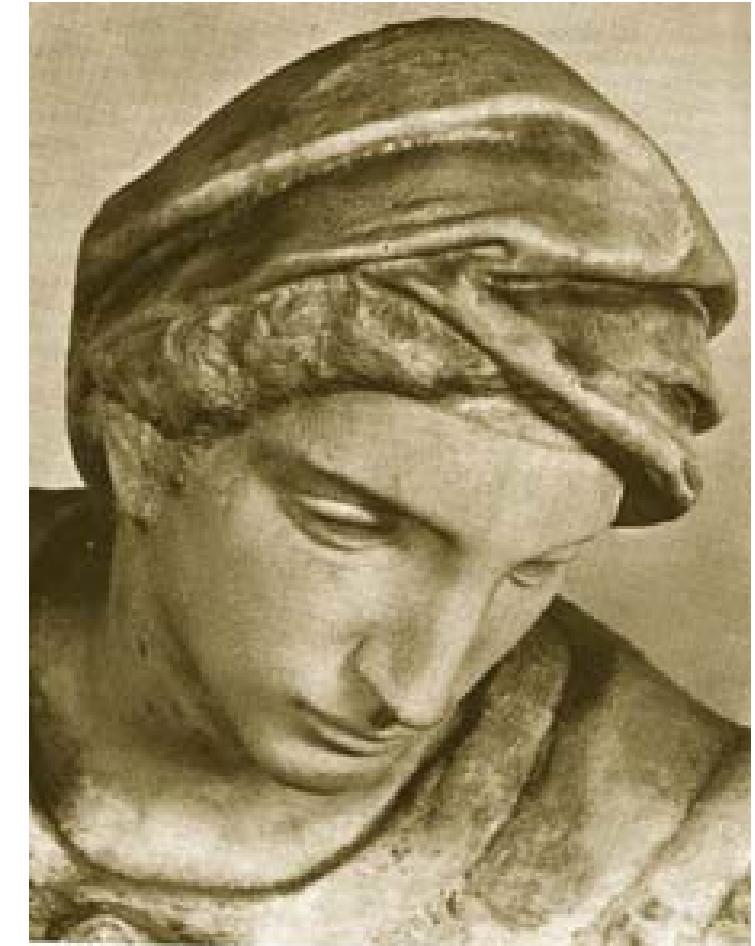
Challenge: Variable Viewpoint



Michelangelo 1475-1564



Kaveh Fathian



Challenge: Variable Illumination

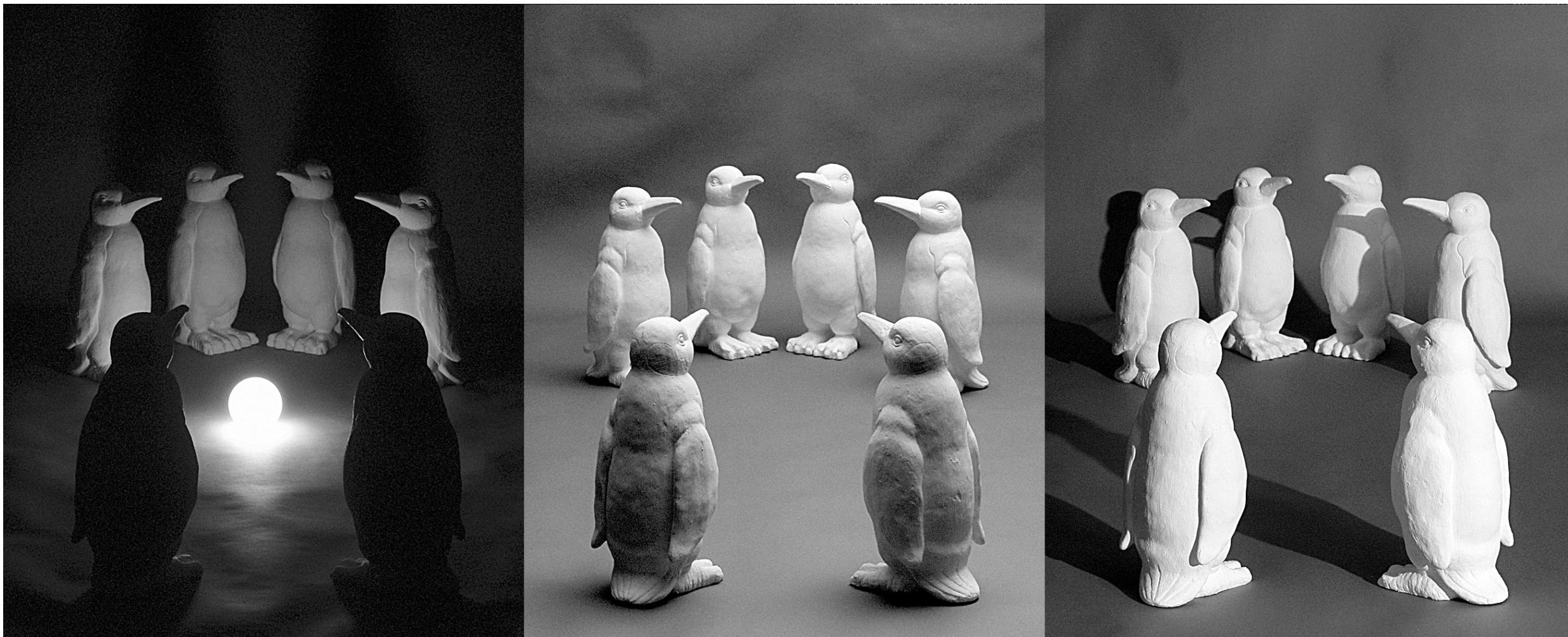


image credit: J. Koenderink

Challenge: Scale

and small things
from Apple.
(Actual size)



Kaveh Fathian

Challenge: Deformation



Challenge: Occlusion



Magritte, 1957



Kaveh Fathian

Challenge: Background Clutter

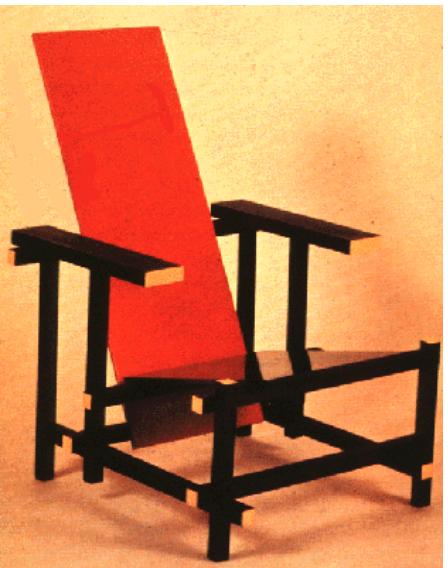


Kilmeny Niland. 1995



Kaveh Fathian

Challenge: Intra-Class Variations





How many object categories are there?



~10,000 to 30,000

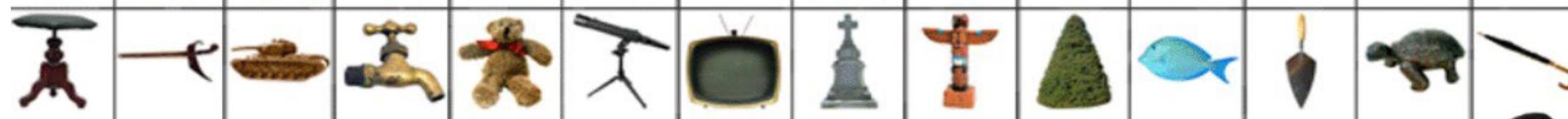
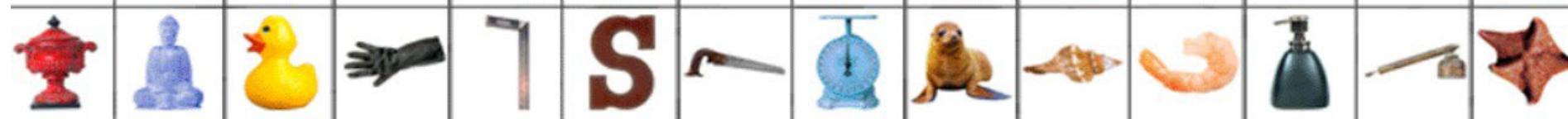


Image Classification

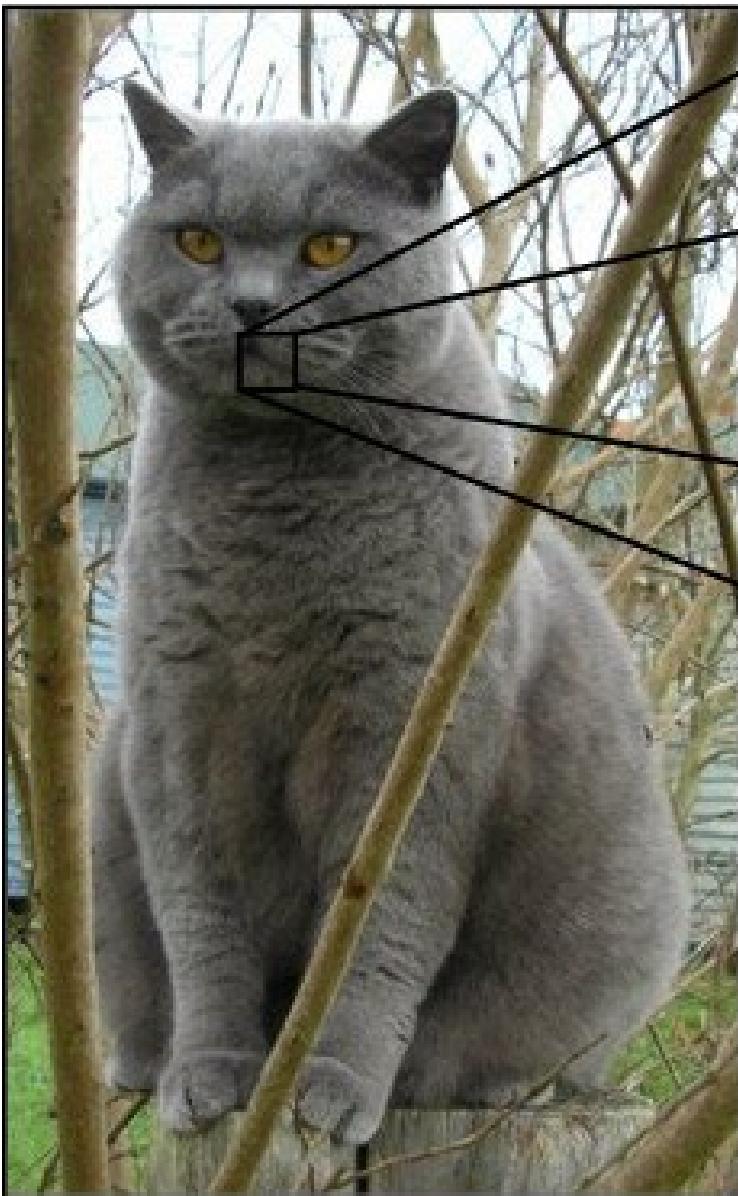


(assume given set of discrete labels)
{dog, cat, truck, plane, ...}



cat

Image Classification: Problem



| | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 08 | 02 | 22 | 97 | 38 | 15 | 00 | 40 | 00 | 75 | 04 | 05 | 07 | 78 | 52 | 12 | 50 | 77 | 91 | 65 |
| 19 | 49 | 99 | 40 | 17 | 81 | 18 | 57 | 60 | 87 | 17 | 40 | 95 | 43 | 69 | 45 | 07 | 56 | 62 | 00 |
| 81 | 49 | 31 | 73 | 58 | 79 | 14 | 29 | 93 | 71 | 40 | 67 | 56 | 88 | 30 | 03 | 49 | 13 | 36 | 65 |
| 52 | 70 | 95 | 23 | 04 | 60 | 11 | 42 | 62 | 31 | 66 | 56 | 01 | 32 | 56 | 71 | 37 | 02 | 36 | 91 |
| 22 | 31 | 16 | 71 | 51 | 62 | 03 | 89 | 41 | 92 | 36 | 54 | 22 | 40 | 40 | 28 | 66 | 33 | 13 | 80 |
| 24 | 47 | 38 | 65 | 99 | 03 | 45 | 02 | 44 | 75 | 35 | 55 | 78 | 36 | 84 | 20 | 35 | 17 | 12 | 50 |
| 32 | 95 | 81 | 28 | 64 | 23 | 67 | 10 | 26 | 38 | 40 | 67 | 59 | 54 | 70 | 66 | 18 | 38 | 64 | 70 |
| 67 | 26 | 20 | 68 | 02 | 62 | 12 | 20 | 95 | 63 | 94 | 39 | 63 | 08 | 40 | 91 | 66 | 49 | 94 | 21 |
| 24 | 55 | 58 | 05 | 66 | 73 | 99 | 26 | 97 | 17 | 78 | 78 | 96 | 83 | 14 | 88 | 34 | 89 | 63 | 72 |
| 21 | 36 | 23 | 09 | 75 | 00 | 76 | 44 | 20 | 45 | 35 | 14 | 00 | 61 | 33 | 97 | 34 | 31 | 33 | 95 |
| 78 | 17 | 53 | 28 | 22 | 75 | 31 | 67 | 15 | 94 | 03 | 80 | 04 | 62 | 16 | 14 | 09 | 53 | 56 | 92 |
| 16 | 39 | 05 | 42 | 96 | 35 | 31 | 47 | 55 | 58 | 88 | 24 | 00 | 17 | 54 | 24 | 36 | 29 | 85 | 57 |
| 86 | 36 | 00 | 48 | 35 | 71 | 89 | 07 | 05 | 44 | 44 | 37 | 44 | 60 | 21 | 58 | 51 | 34 | 17 | 58 |
| 19 | 80 | 81 | 68 | 05 | 94 | 47 | 69 | 28 | 73 | 92 | 13 | 86 | 32 | 17 | 77 | 04 | 89 | 55 | 40 |
| 04 | 52 | 08 | 83 | 97 | 35 | 99 | 16 | 07 | 97 | 57 | 32 | 16 | 26 | 26 | 79 | 33 | 27 | 98 | 66 |
| 83 | 94 | 68 | 87 | 57 | 62 | 20 | 72 | 03 | 46 | 35 | 67 | 46 | 55 | 12 | 32 | 63 | 93 | 53 | 69 |
| 04 | 42 | 16 | 73 | 58 | 45 | 39 | 11 | 24 | 94 | 72 | 18 | 05 | 46 | 29 | 32 | 40 | 62 | 76 | 36 |
| 20 | 69 | 36 | 41 | 72 | 30 | 23 | 88 | 31 | 40 | 99 | 69 | 82 | 67 | 59 | 85 | 74 | 04 | 36 | 16 |
| 20 | 73 | 35 | 29 | 78 | 31 | 90 | 01 | 74 | 31 | 49 | 71 | 98 | 86 | 41 | 16 | 23 | 57 | 05 | 54 |
| 01 | 70 | 54 | 71 | 83 | 51 | 54 | 69 | 16 | 92 | 33 | 48 | 61 | 43 | 52 | 01 | 89 | 24 | 47 | 48 |

What the computer sees

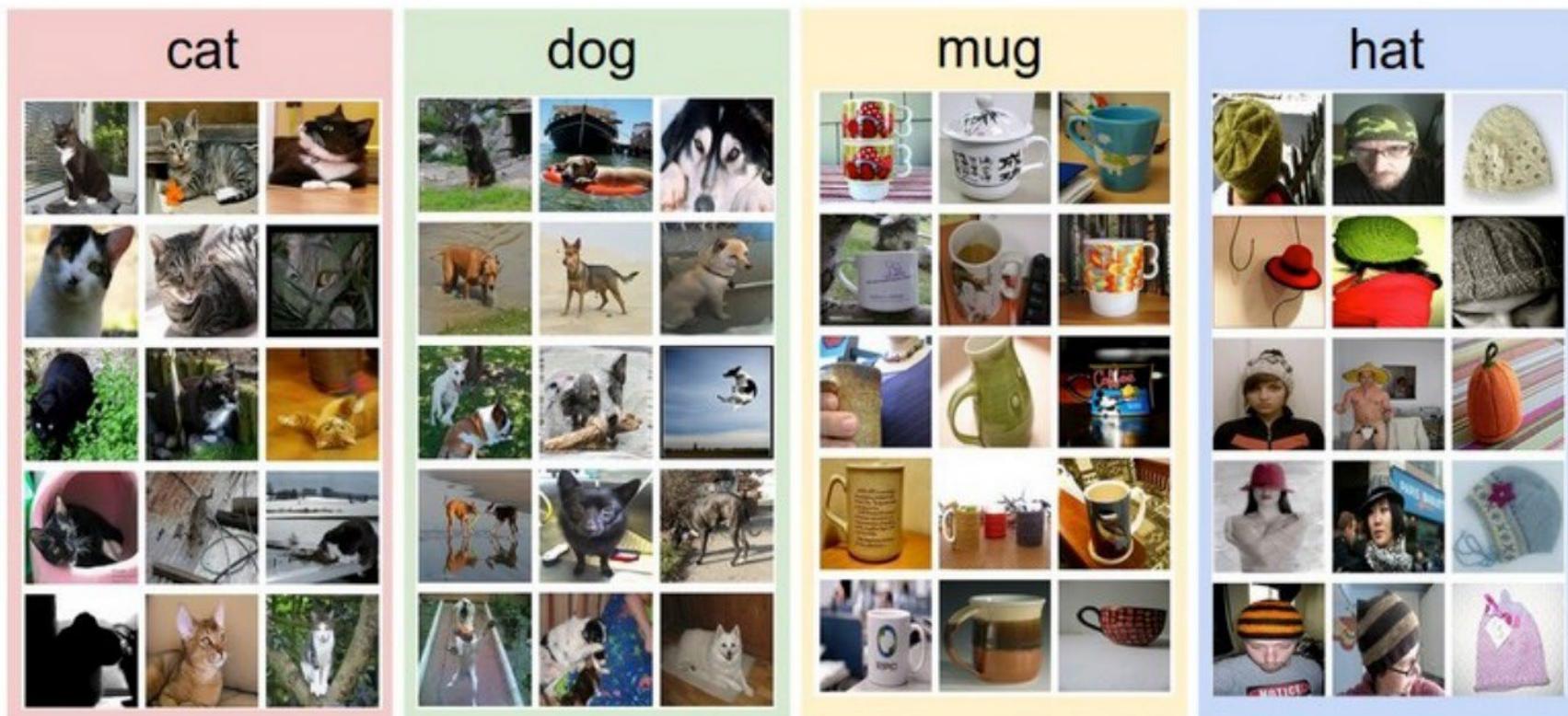
image classification

82% cat
15% dog
2% hat
1% mug

Data-Driven Approach

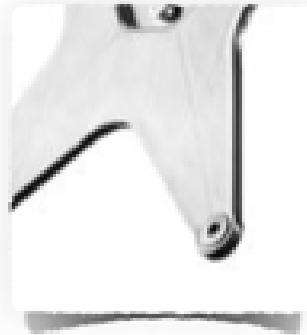
- Collect a database of images with labels
- Use ML to train an image classifier
- Evaluate the classifier on test images

Example training set



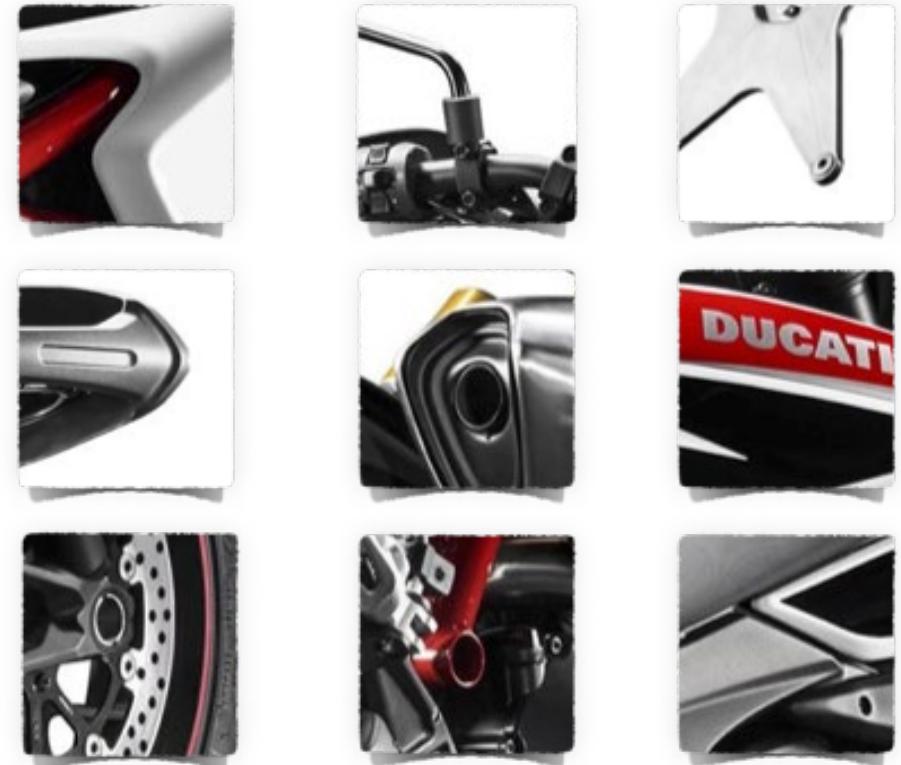
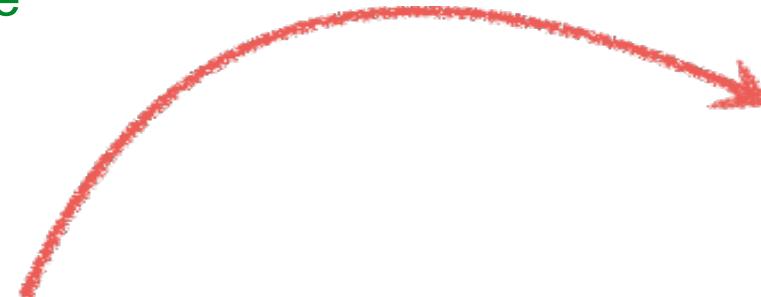
Bag of Words (BoW)

What object do these parts belong to?



Some local feature are very informative

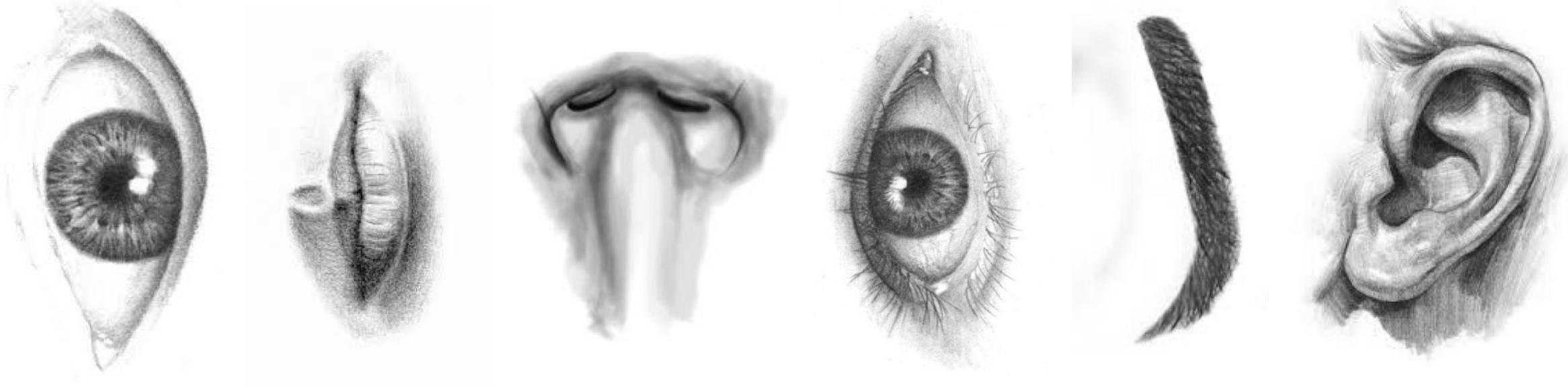
An object as



A collection of local features
(bag-of-features)

- Deals well with occlusion
- Scale invariant
- Rotation invariant

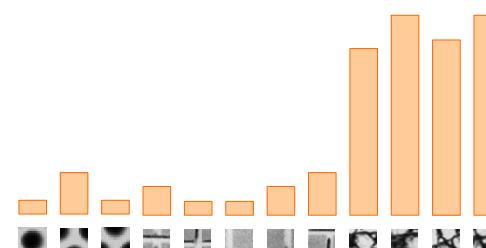
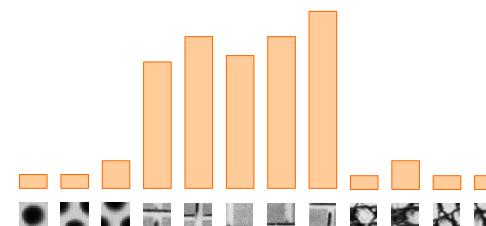
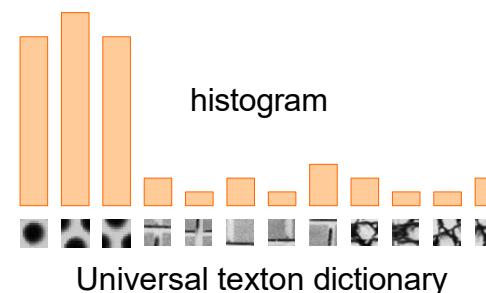
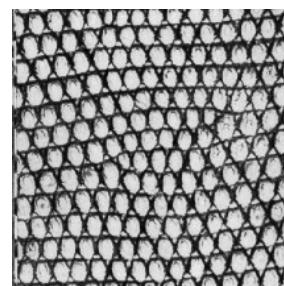
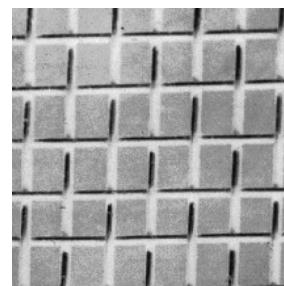
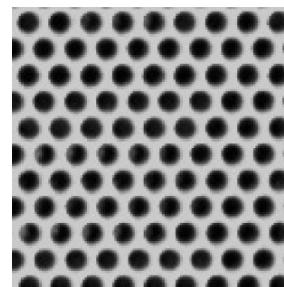
Not So Crazy Assumption!



Spatial information of local features
can be ignored for object recognition

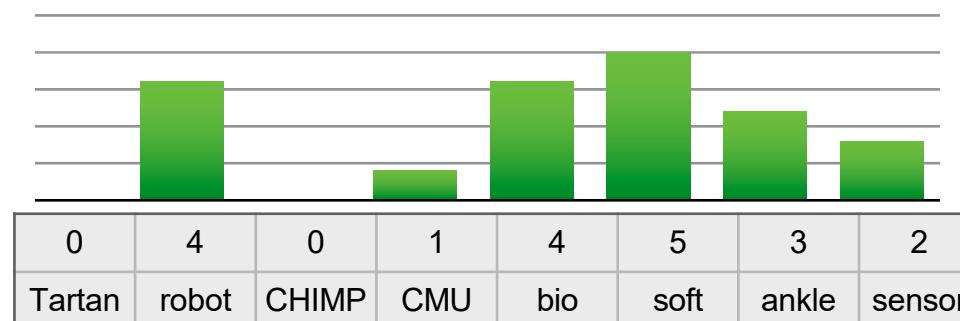
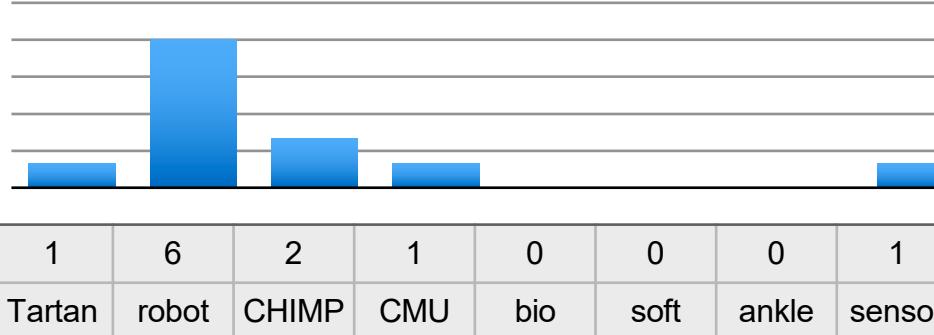
Bag-of-Features

- Represent a data item (document, texture, image) as a histogram over features
- An old idea (e.g., texture recognition and information retrieval)



Vector Space Model

G. Salton. 'Mathematics and Information Retrieval' Journal of Documentation, 1979



<http://www.fodey.com/generators/newspaper/snippet.asp>

A document (datapoint) is a vector of counts over each word (feature)

$$\mathbf{v}_d = [n(w_{1,d}) \ n(w_{2,d}) \ \cdots \ n(w_{T,d})]$$

$n(\cdot)$ counts the number of occurrences



just a histogram over words

What is the similarity between two documents?



A document (datapoint) is a vector of counts over each word (feature)

$$\mathbf{v}_d = [n(w_{1,d}) \ n(w_{2,d}) \ \cdots \ n(w_{T,d})]$$

$n(\cdot)$ counts the number of occurrences



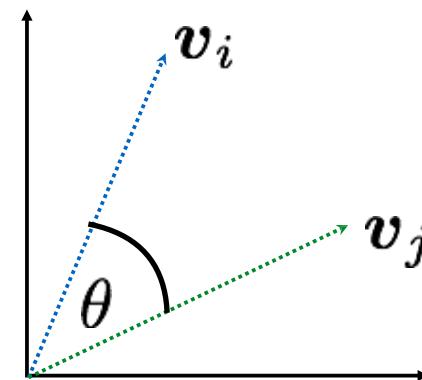
just a histogram over words

What is the similarity between two documents?

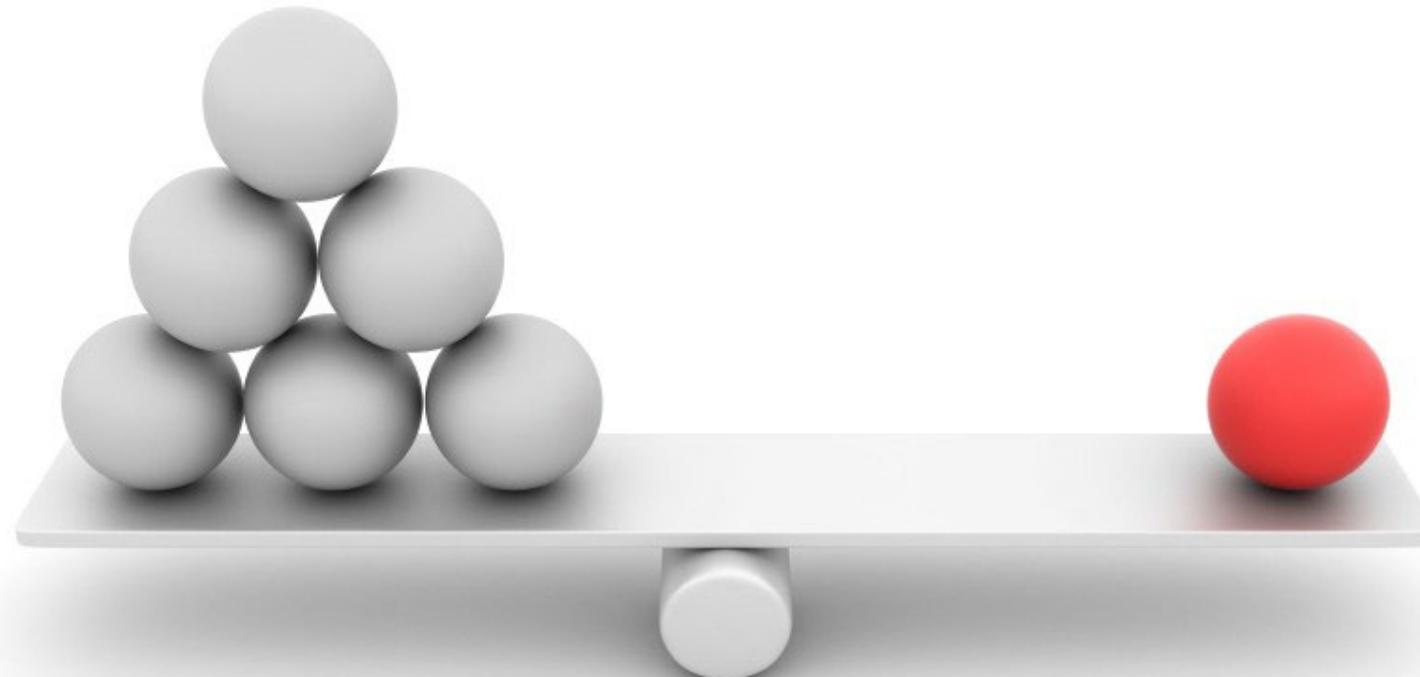


Use any distance you want but the cosine distance is fast.

$$d(\mathbf{v}_i, \mathbf{v}_j) = \cos \theta$$
$$= \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}$$



But Not All Words Are Created Equal



TF-IDF

TF-IDF: Term Frequency Inverse Document Frequency

$$\mathbf{v}_d = [n(w_{1,d}) \ n(w_{2,d}) \ \cdots \ n(w_{T,d})]$$

weigh each word by a heuristic

$$\mathbf{v}_d = [n(w_{1,d})\alpha_1 \ n(w_{2,d})\alpha_2 \ \cdots \ n(w_{T,d})\alpha_T]$$

$$n(w_{i,d})\alpha_i = n(w_{i,d}) \log \left\{ \frac{D}{\sum_{d'} \mathbf{1}[w_i \in d']} \right\}$$

term frequency inverse document frequency

(down-weights **common** terms)

Standard BOW Pipeline (for image classification)

Standard BOW Pipeline

Dictionary Learning:
Learn Visual Words using clustering

Encode:
Build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Standard BOW Pipeline

Dictionary Learning:
Learn Visual Words using clustering

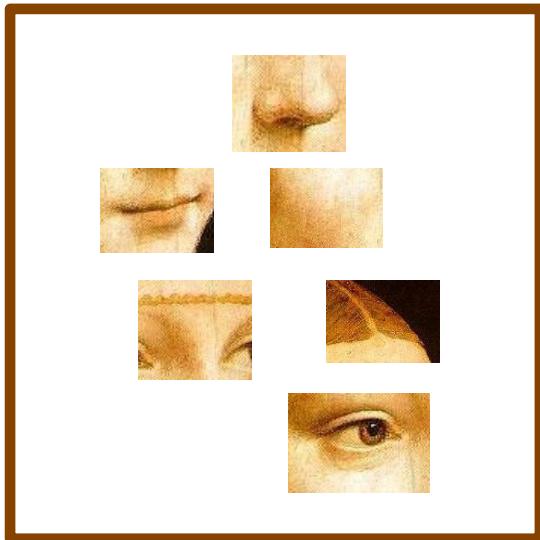
Encode:
Build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Standard BOW Pipeline

Dictionary Learning:
Learn Visual Words using clustering

1. Extract features (e.g., SIFT) from images



Standard BOW Pipeline

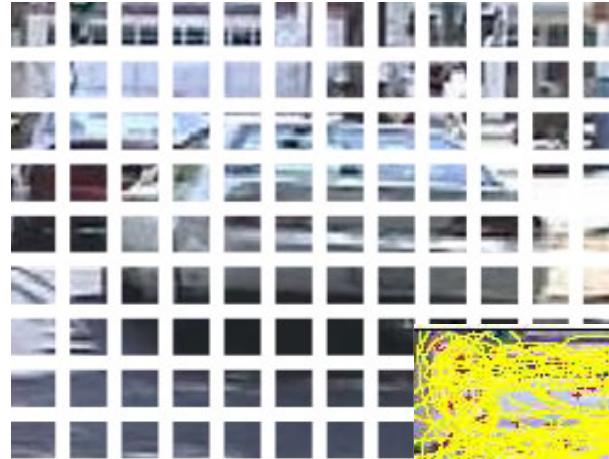
Dictionary Learning:
Learn Visual Words using clustering

2. Learn visual dictionary (e.g., K-means clustering)

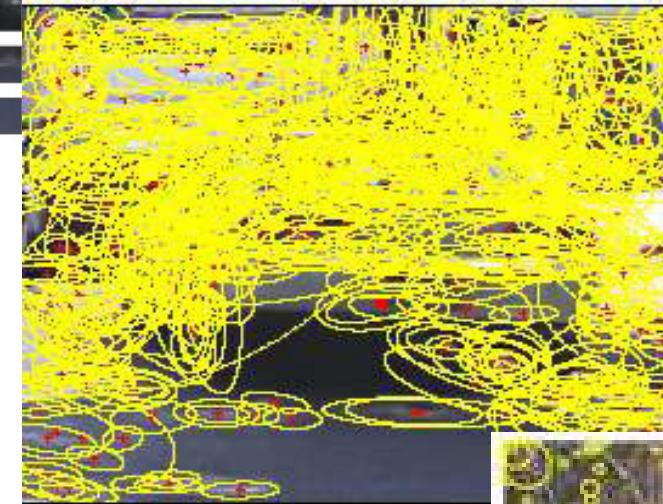


What Kinds of Features Can We Extract?

- Regular grid
 - Vogel & Schiele, 2003
 - Fei-Fei & Perona, 2005
- Interest point detector
 - Csurka et al. 2004
 - Fei-Fei & Perona, 2005
 - Sivic et al. 2005
- Other methods
 - Random sampling (Vidal-Naquet & Ullman, 2002)
 - Segmentation-based patches (Barnard et al. 2003)



Dense, regular grid

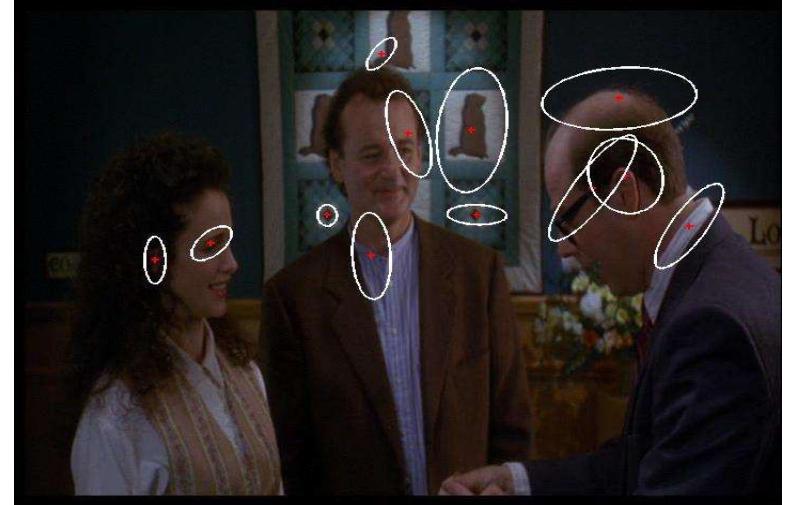
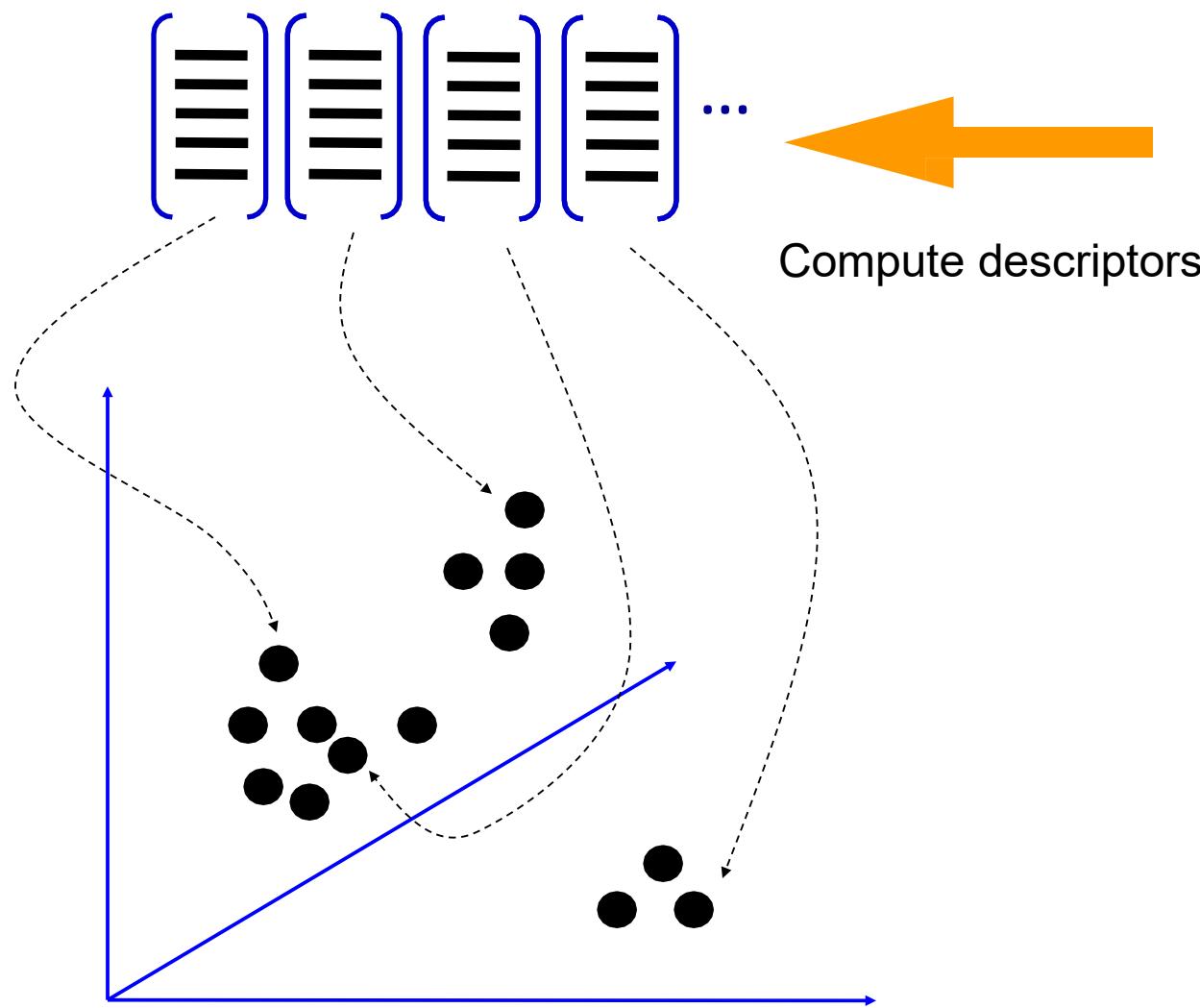


Sparse at
interest points



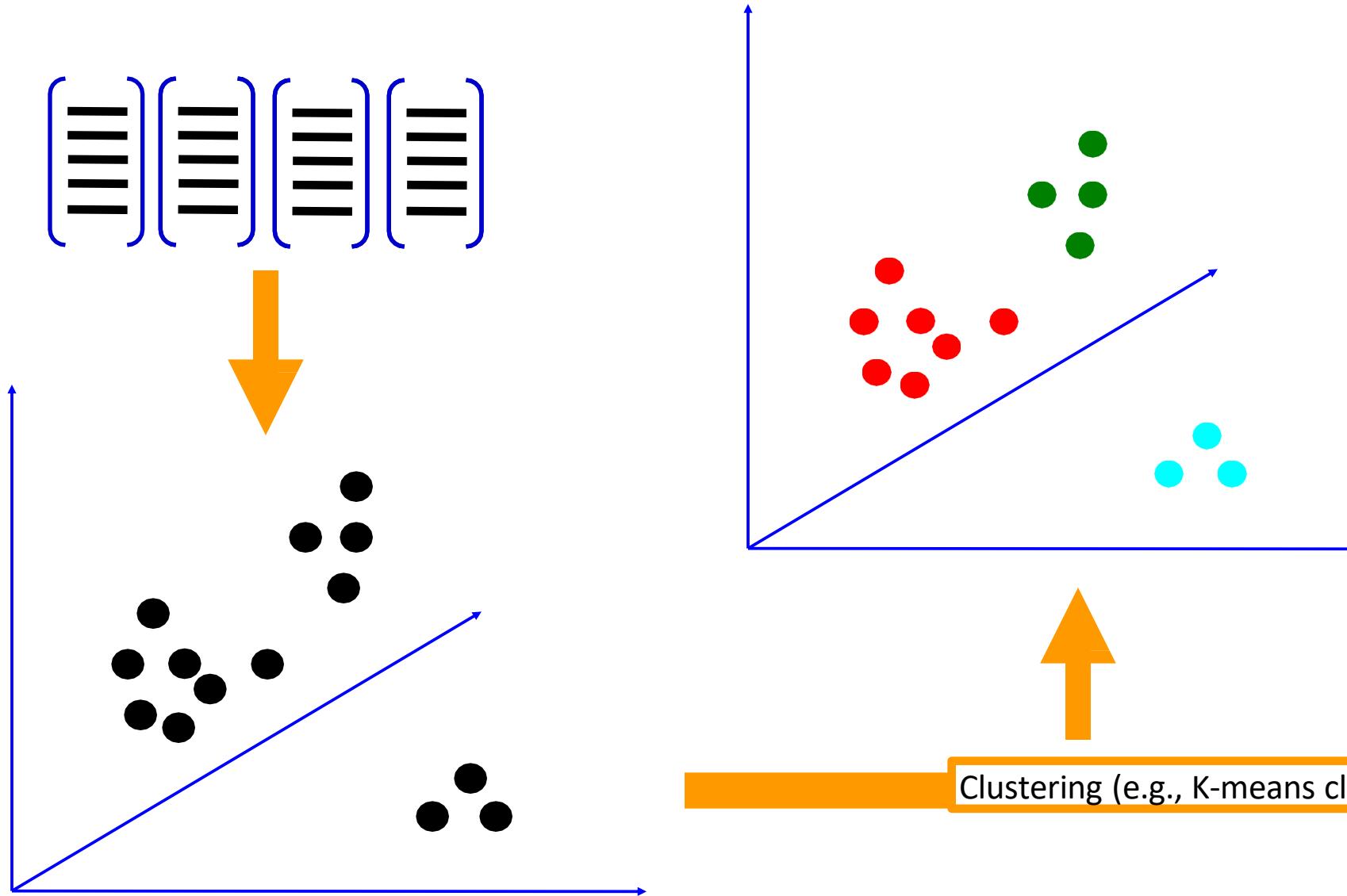
Randomly

How Do We Learn the Dictionary?

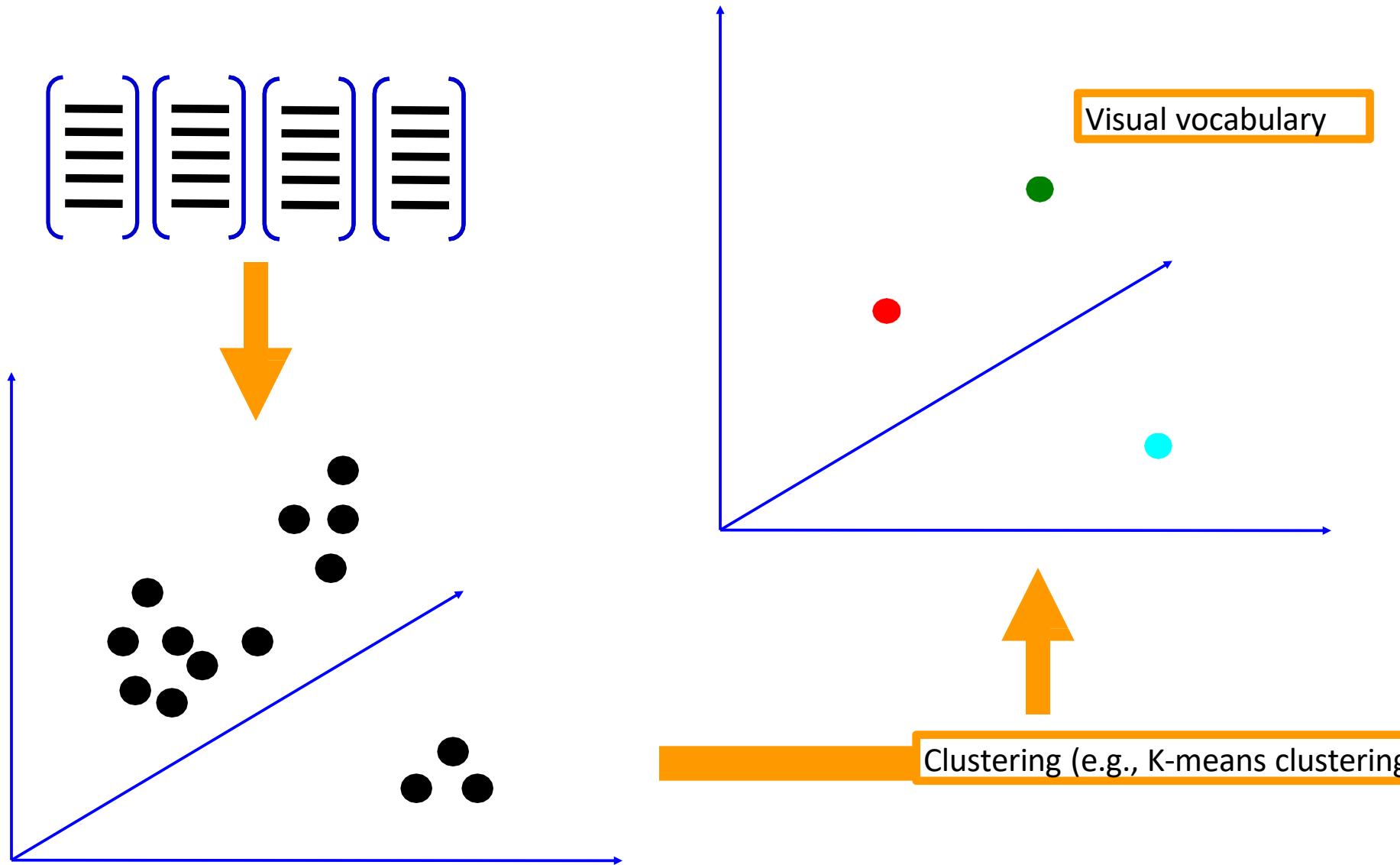


Detect features

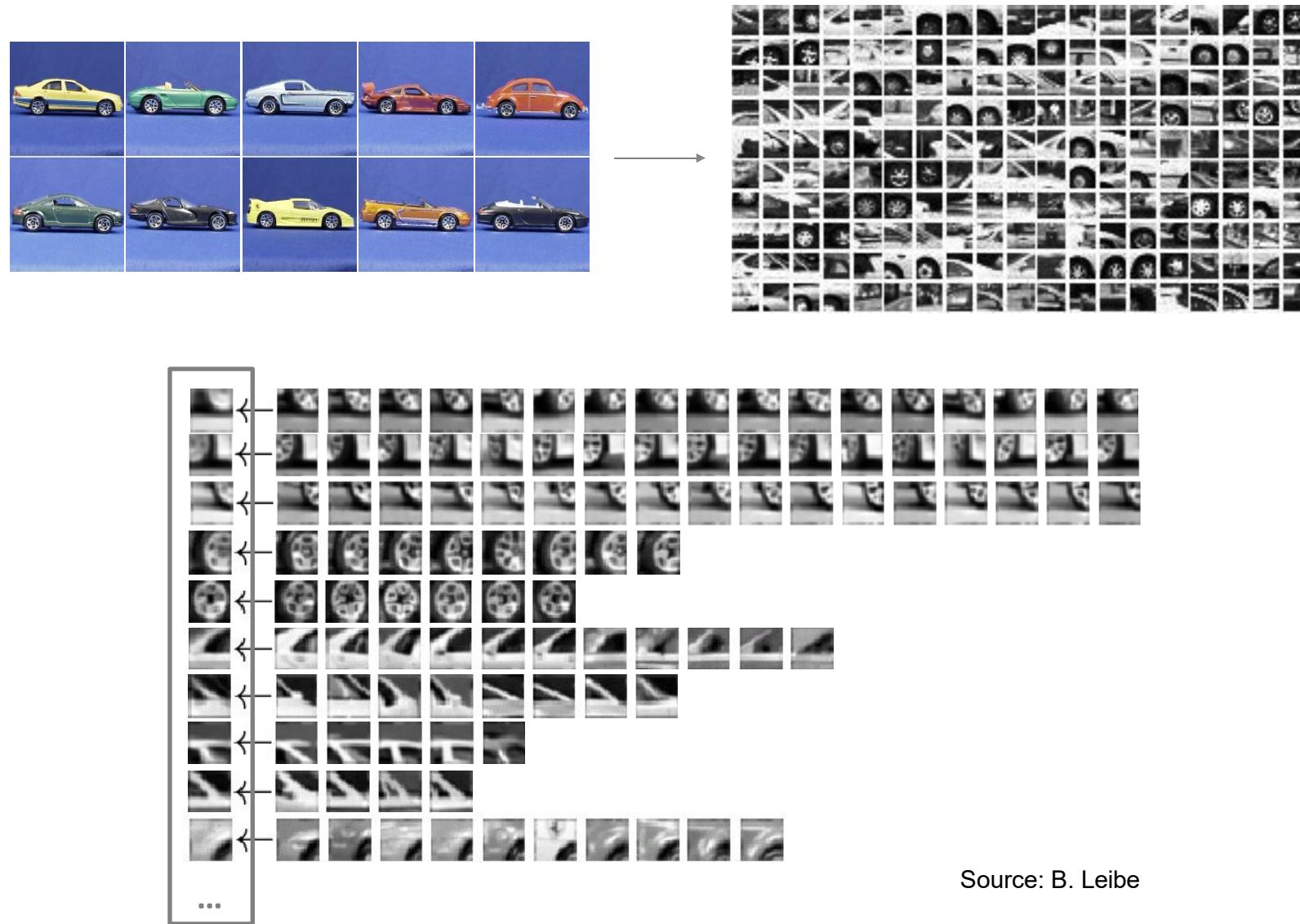
How Do We Learn the Dictionary?



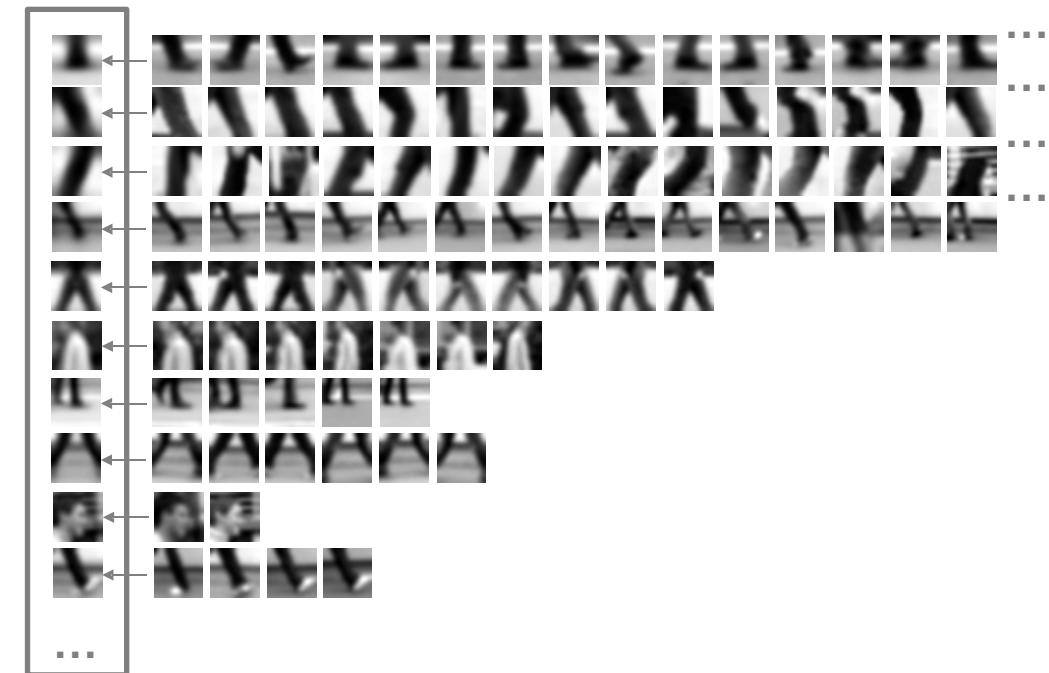
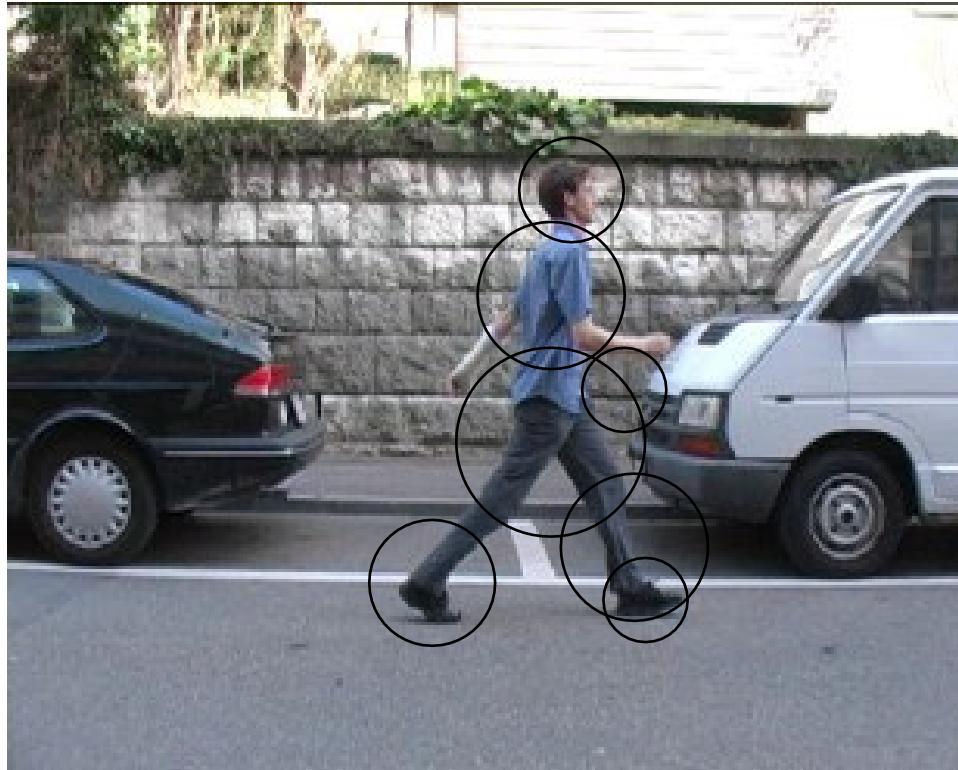
How Do We Learn the Dictionary?



Example of Visual Dictionary



Another Dictionary Example



Standard BOW Pipeline

Dictionary Learning:
Learn Visual Words using clustering

Encode:
Build Bags-of-Words (BOW) vectors
for each image

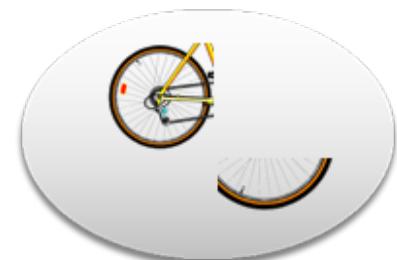
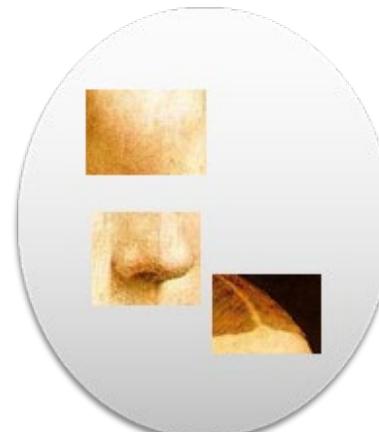
Classify:
Train and test data using BOWs

Encode:

Build Bags-of-Words (BOW) vectors for each image

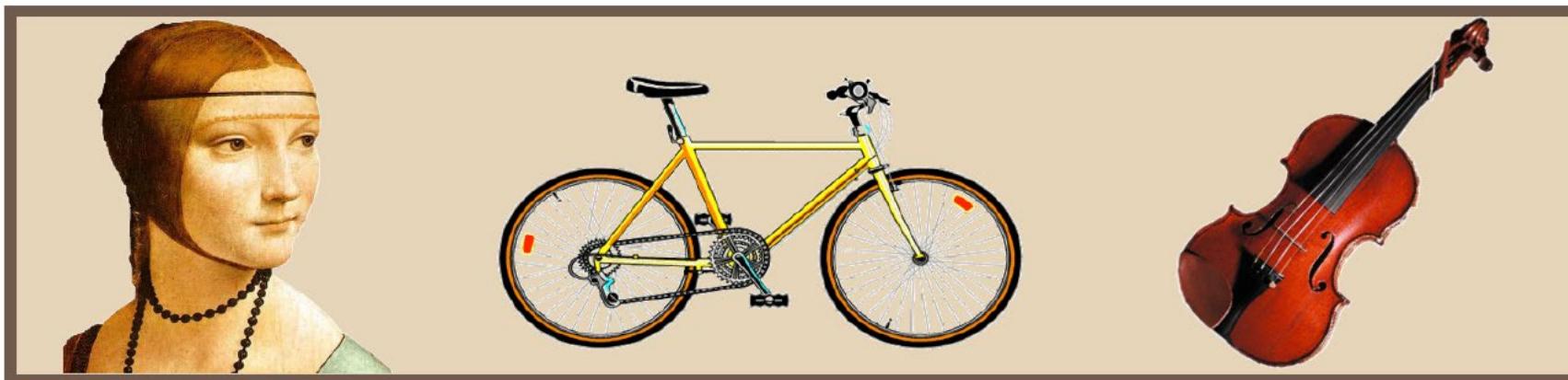
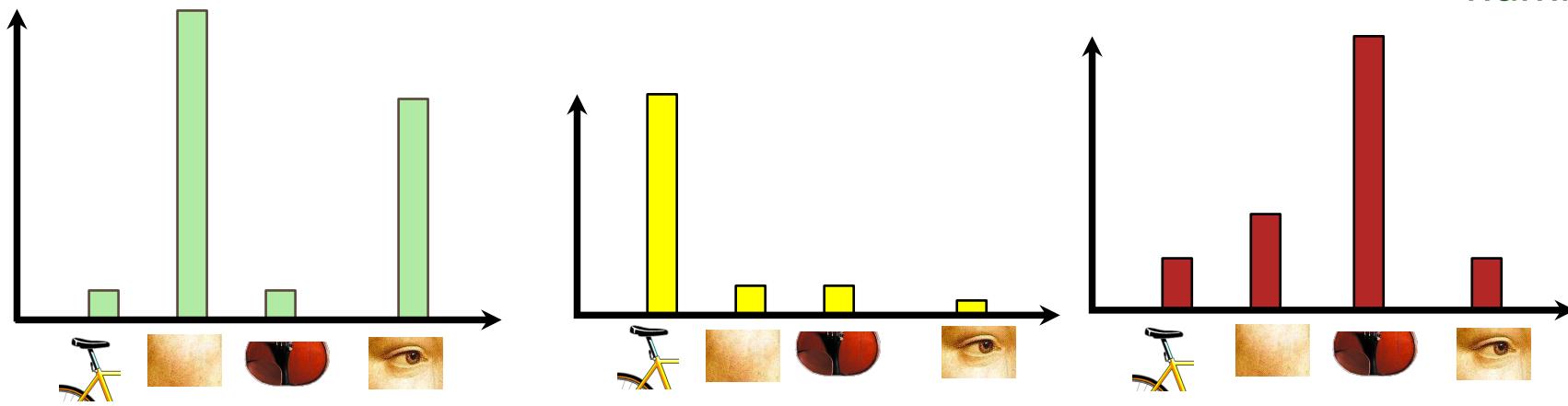


1. Quantization: image features gets associated to a visual word (nearest cluster center)



Encode: Build Bags-of-Words (BOW) vectors for each image

2. Histogram: count the number of visual word occurrences



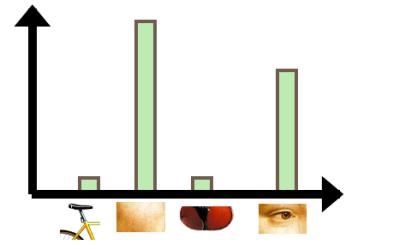
Standard BOW Pipeline

Dictionary Learning:
Learn Visual Words using clustering

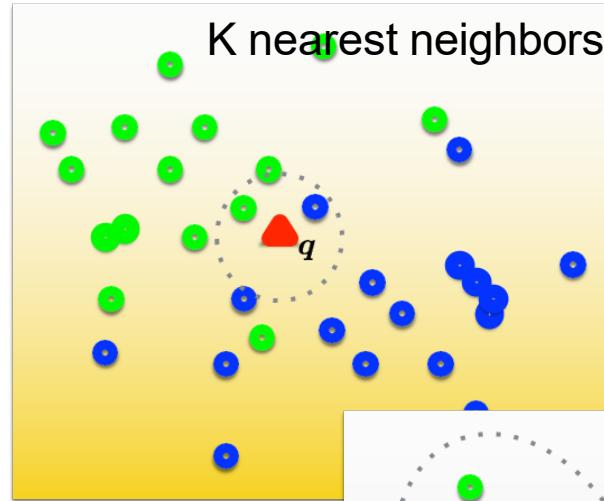
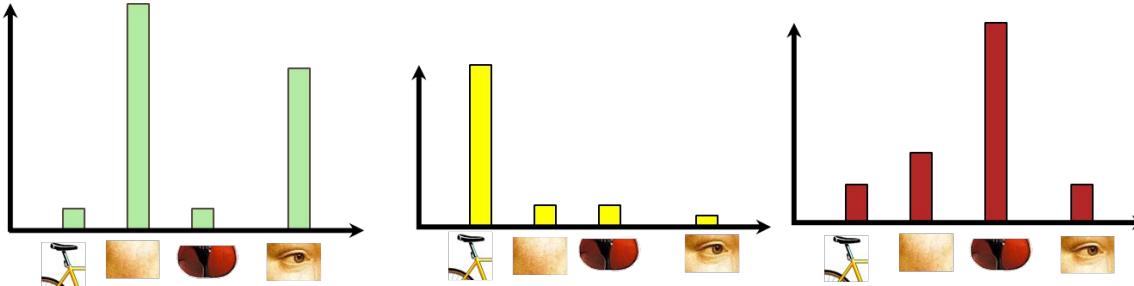
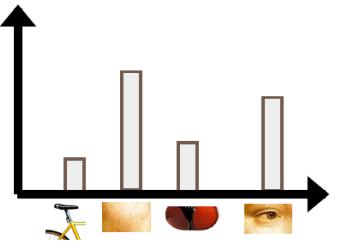
Encode:
Build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

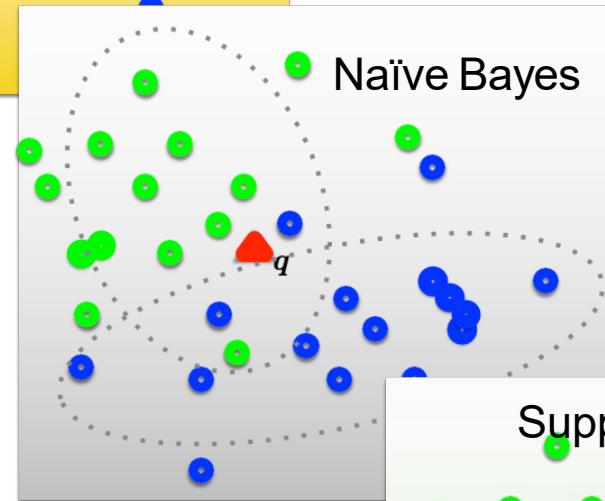
Classification Methods For BOW



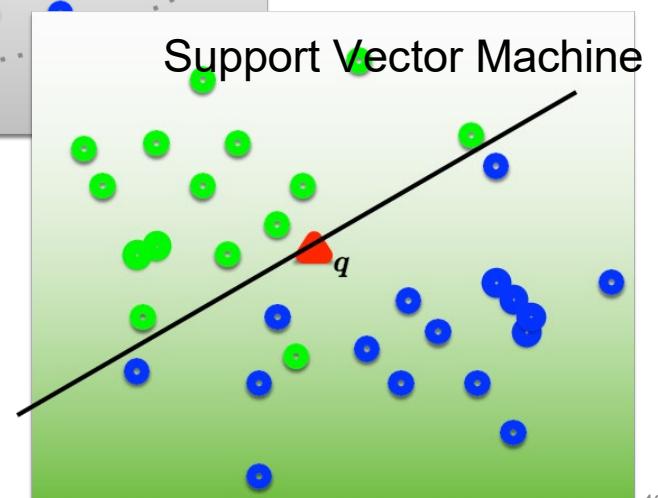
?



K nearest neighbors



Naïve Bayes



Support Vector Machine

CalTech-256 Dataset



| class | bag of features | bag of features | Parts-and-shape model |
|--------------|---------------------|---------------------------|-----------------------|
| | Zhang et al. (2005) | Willamowski et al. (2004) | Fergus et al. (2003) |
| airplanes | 98.8 | 97.1 | 90.2 |
| cars (rear) | 98.3 | 98.6 | 90.3 |
| cars (side) | 95.0 | 87.3 | 88.5 |
| faces | 100 | 99.3 | 96.4 |
| motorbikes | 98.5 | 98.0 | 92.5 |
| spotted cats | 97.0 | — | 90.0 |

- Caltech-256 object recognition dataset contains 30,607 real-world images
- Spans 257 classes (256 objects and a clutter class)
- Bag of features works pretty well for image-level classification!

Csurka et al. (2004), Willamowski et al. (2005), Grauman & Darrell (2005), Sivic et al. (2003, 2005)

How to score the retrieval results?

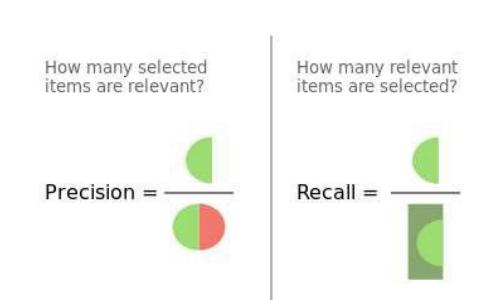
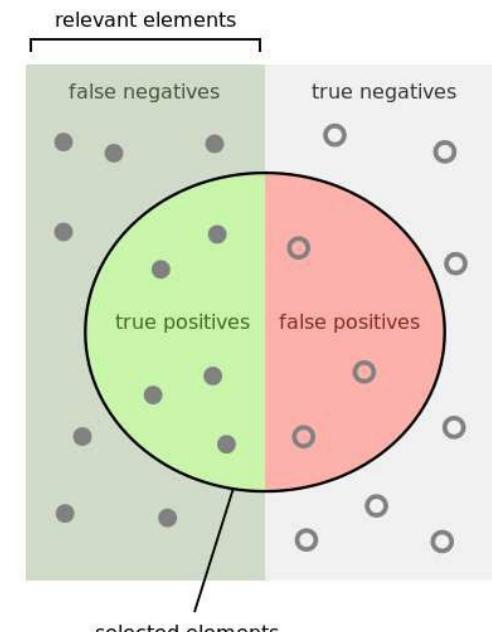
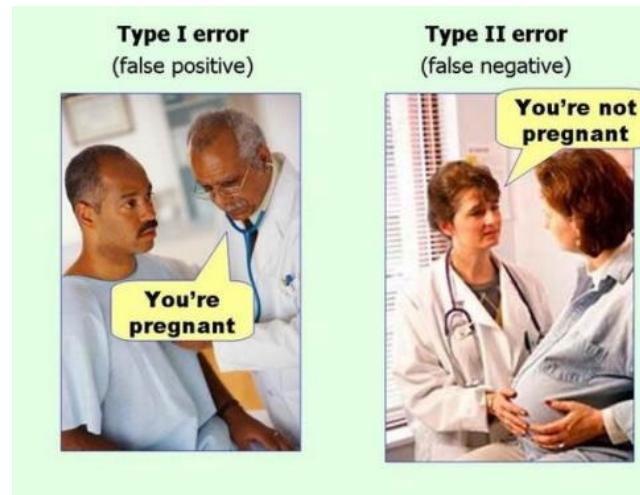
Precision and Recall

True positive (tp) – correct attribution

True negative (tn) – correct rejection

False positive (fp) – incorrect attribution

False negative (fn) – incorrect rejection



<https://commons.wikimedia.org/w/index.php?curid=36926283>

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Precision} = \frac{\text{#relevant}}{\text{#returned}}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{Recall} = \frac{\text{#relevant}}{\text{#total relevant}}$$

Query



Scoring Retrieval Quality Example

Results (ordered):

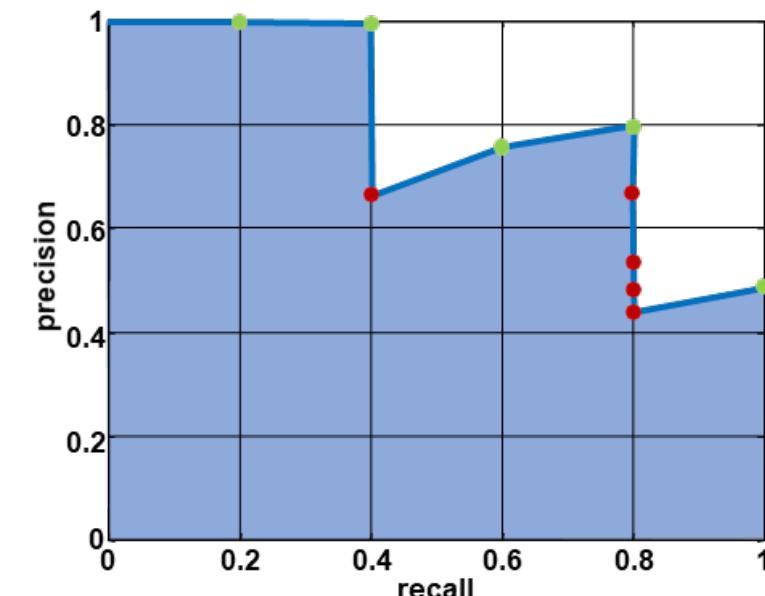


Database size: 10 images

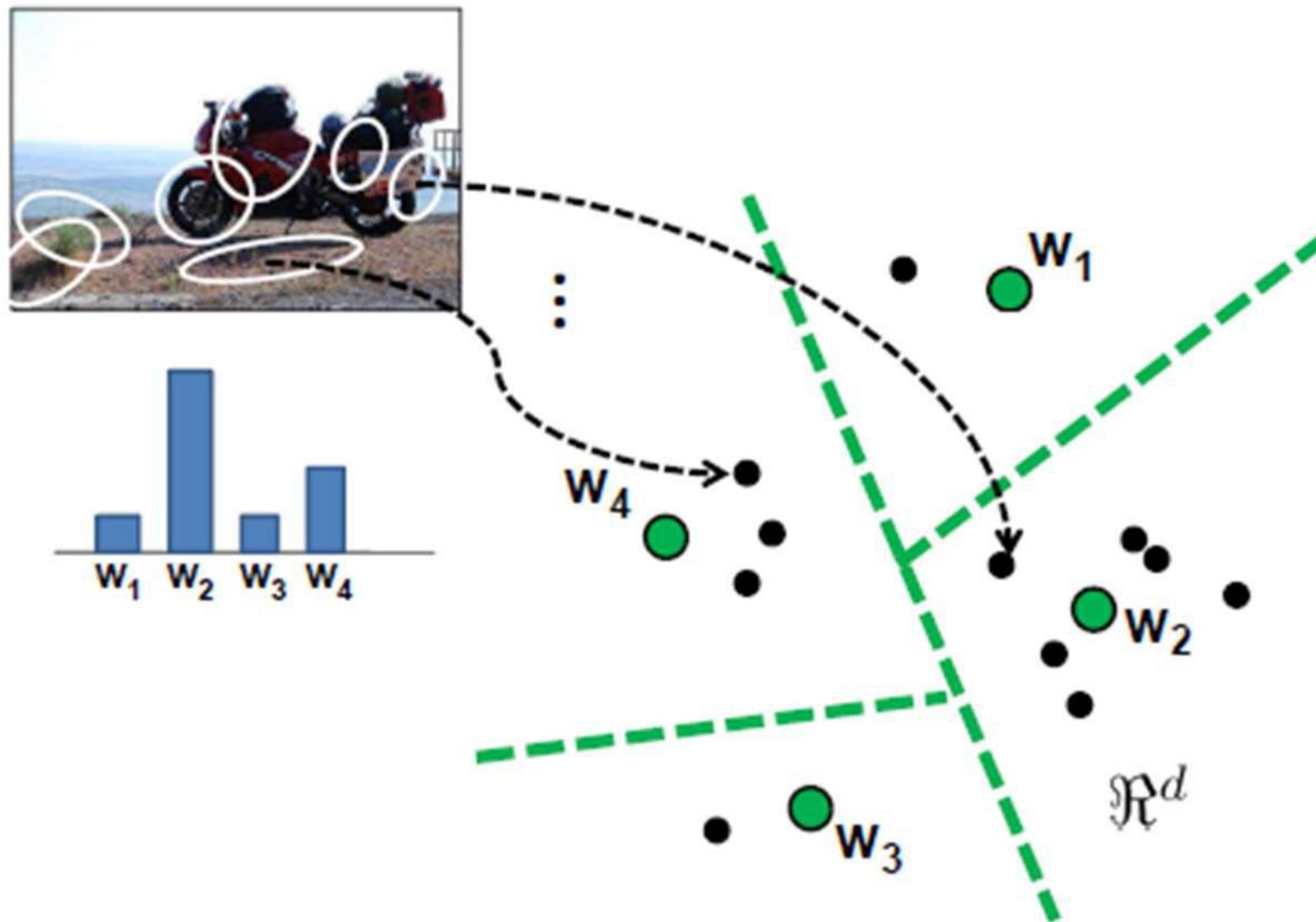
Relevant (total): 5 images

Precision = #relevant / #returned

Recall = #relevant / #total relevant



Standard K-means Bag of Words



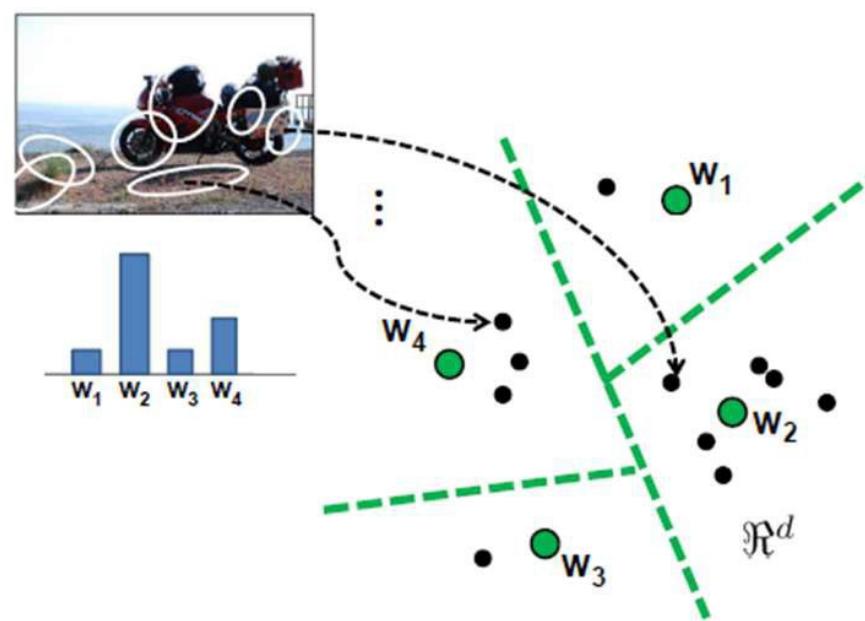
Better Bags of Visual Features



- More advanced quantization / encoding methods
 - Mixtures of Gaussians
 - Soft assignment (a.k.a. Kernel Codebook)
 - VLAD – Vectors of Locally-Aggregated Descriptors
- Deep learning has taken attention away from these methods...

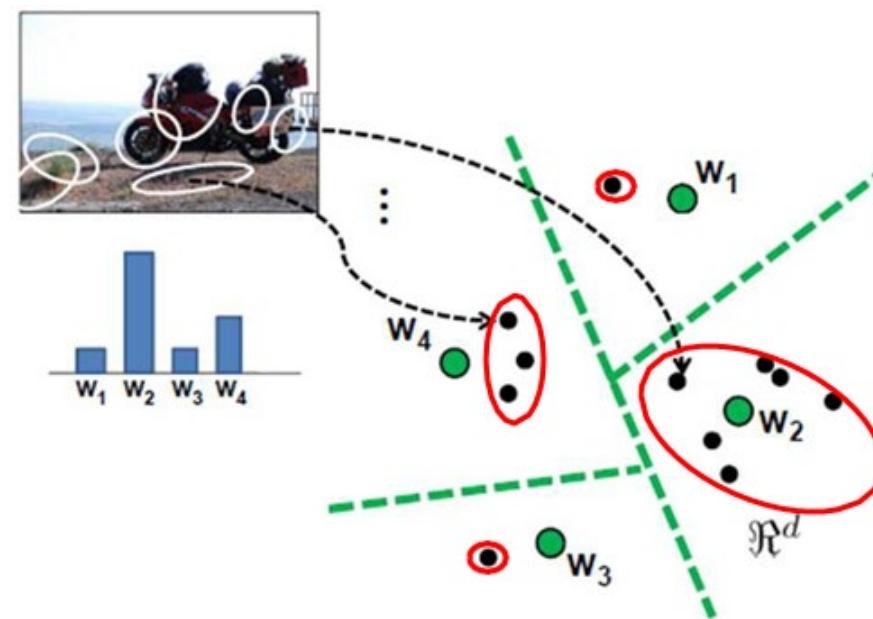
Motivation

- Bag of Visual Words is only about **counting** the number of local descriptors assigned to each Voronoi region
- Why not including **other statistics**?



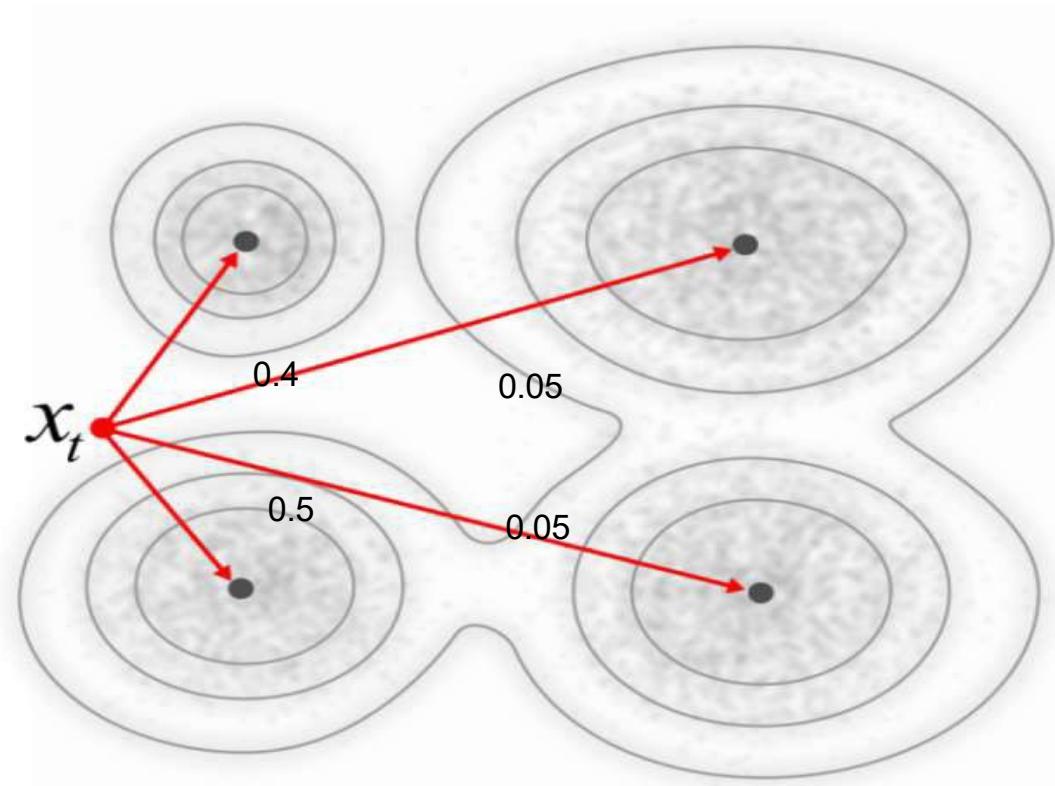
Motivation

- Bag of Visual Words is only about **counting** the number of local descriptors assigned to each Voronoi region
- Why not including **other statistics**?
- For instance:
 - Mean of local descriptors
 - (Co)variance of local descriptors



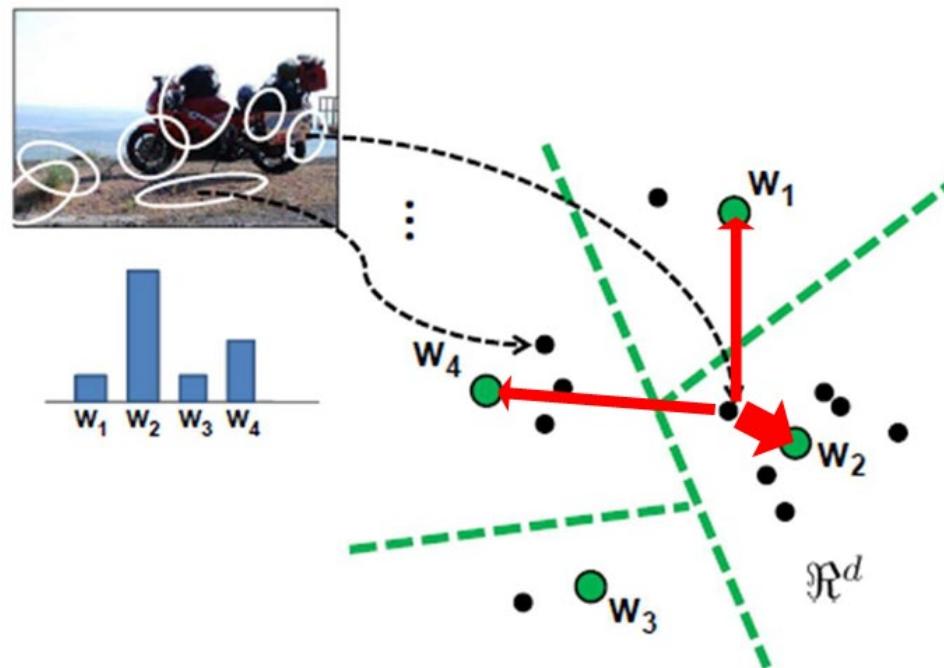
Gaussian Mixture Model (GMM)

- GMM can be thought of as “soft” k-means.
- Each component has a mean and a standard deviation along each direction (or full covariance)
- Can easily represent non-circular distributions



Simple case: Soft Assignment

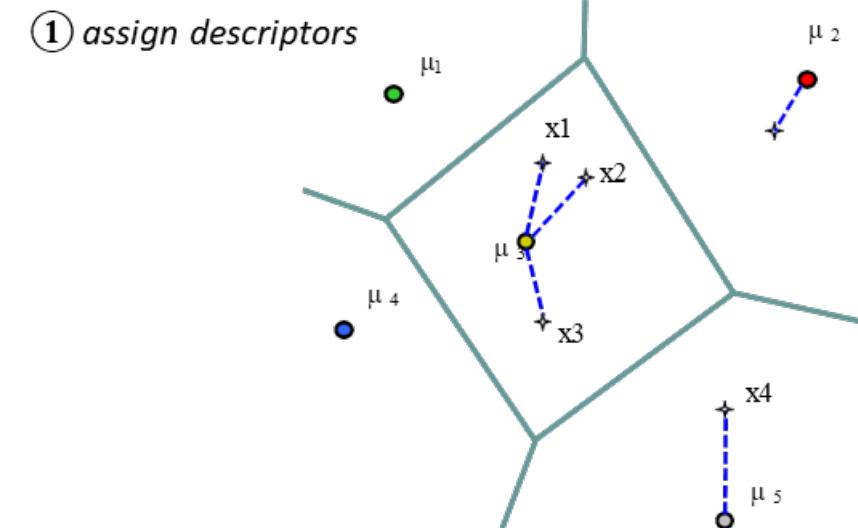
- “Kernel codebook encoding” by Chatfield et al. 2011
- Cast a set of proportional votes (weights) to n most similar clusters, rather than a single ‘hard’ vote
- This is fast and easy to implement, but it makes an inverted file index *less sparse*



VLAD – Vectors of Locally-Aggregated Descriptors

Given a codebook $\{\mu_i, i = 1 \dots N\}$,
e.g. learned with K-means, and a set of
local descriptors $X = \{x_t, t = 1 \dots T\}$:

① assign: $\text{NN}(x_t) = \arg \min_{\mu_i} \|x_t - \mu_i\|$



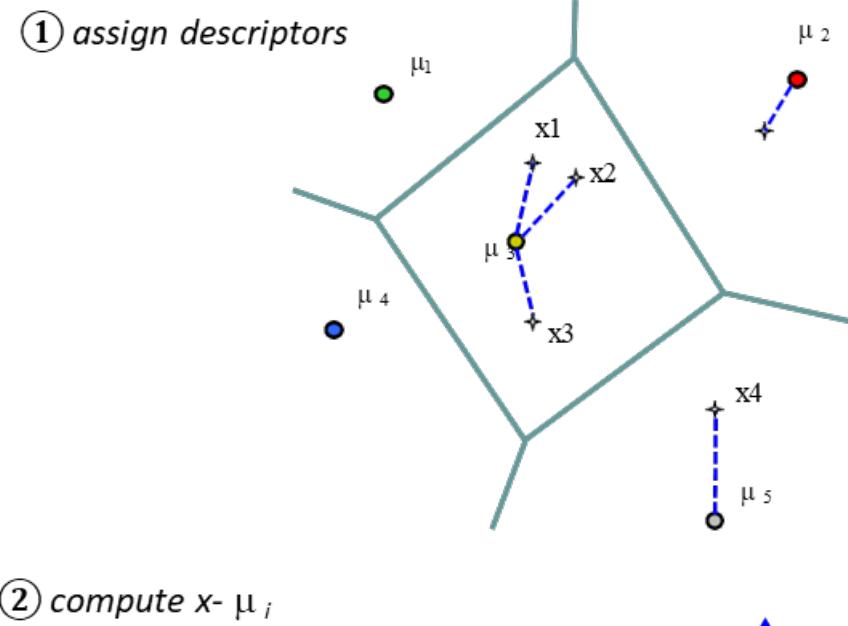
Jégou, Douze, Schmid and Pérez. "Aggregating local descriptors into a compact image representation", CVPR'10.

VLAD – Vectors of Locally-Aggregated Descriptors

Given a codebook $\{\mu_i, i = 1 \dots N\}$,
e.g. learned with K-means, and a set of
local descriptors $X = \{x_t, t = 1 \dots T\}$:

① assign: $\text{NN}(x_t) = \arg \min_{\mu_i} \|x_t - \mu_i\|$

②③ compute: $v_i = \sum_{x_t: \text{NN}(x_t)=\mu_i} x_t - \mu_i$



Jégou, Douze, Schmid and Pérez. "Aggregating local descriptors into a compact image representation", CVPR'10.

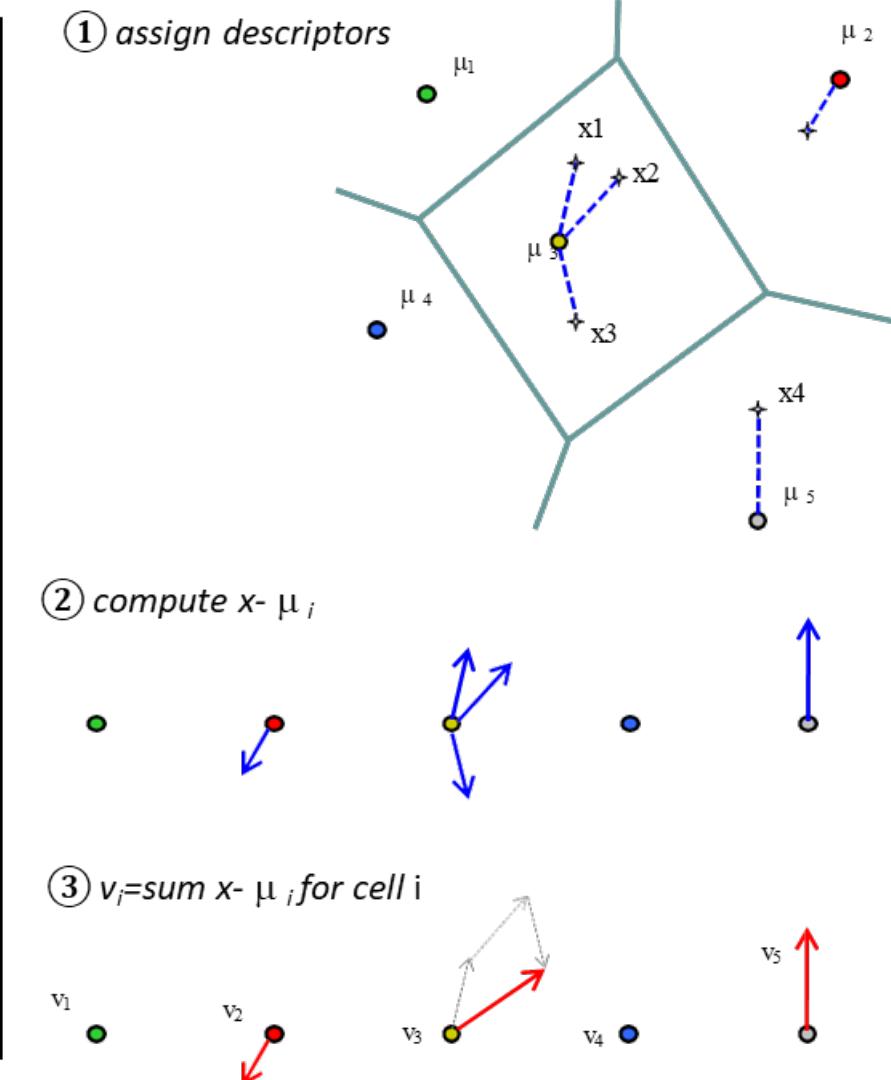
VLAD – Vectors of Locally-Aggregated Descriptors

Given a codebook $\{\mu_i, i = 1 \dots N\}$,
e.g. learned with K-means, and a set of
local descriptors $X = \{x_t, t = 1 \dots T\}$:

① assign: $\text{NN}(x_t) = \arg \min_{\mu_i} \|x_t - \mu_i\|$

②③ compute: $v_i = \sum_{x_t: \text{NN}(x_t)=\mu_i} x_t - \mu_i$

concatenate v_i 's + ℓ_2 normalize

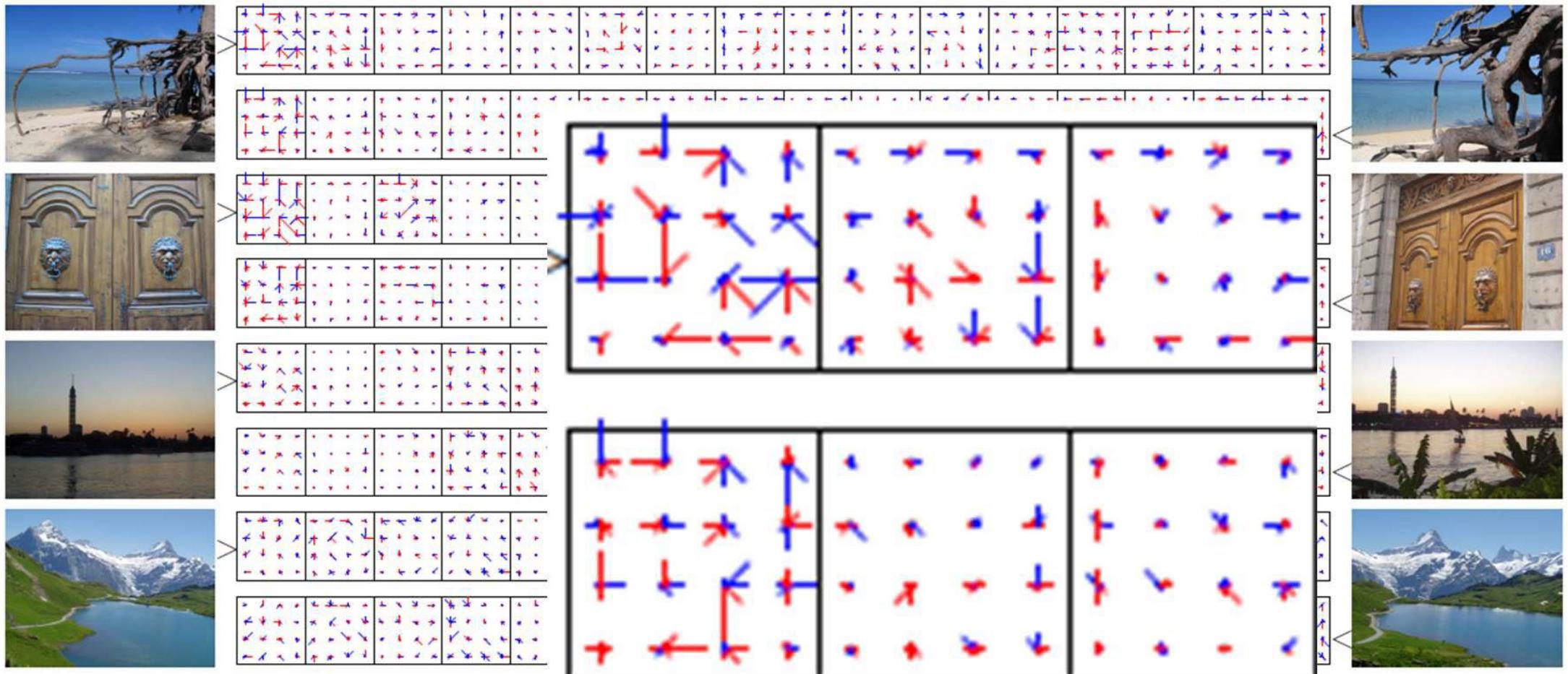


Jégou, Douze, Schmid and Pérez. "Aggregating local descriptors into a compact image representation", CVPR'10.

Example

A graphical representation of

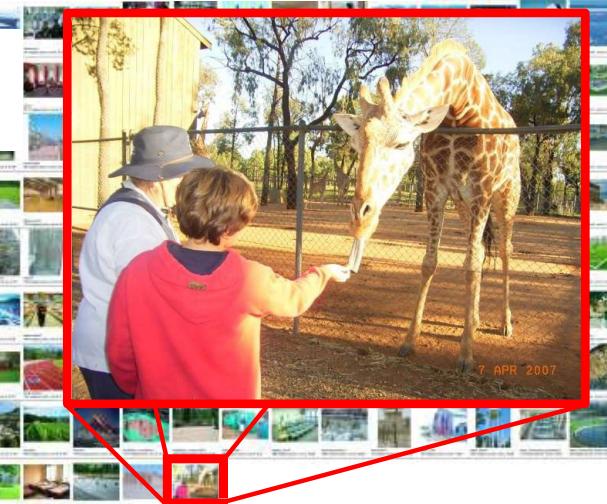
$$v_i = \sum_{x_t: \text{NN}(x_t) = \mu_i} x_t - \mu_i$$





SUN Database – Xiao et al. CVPR 2010

130k images
899 categories



<https://groups.csail.mit.edu/vision/SUN/>

Global Image Descriptors

- Tiny images (Torralba et al, 2008)
- Color histograms
- Self-similarity (Shechtman and Irani, 2007)
- Geometric class layout (Hoiem et al, 2005)
- Geometry-specific histograms (Lalonde et al, 2007)
- Dense and Sparse SIFT histograms
- Berkeley texton histograms (Martin et al, 2001)
- HoG 2x2 spatial pyramids
- Gist scene descriptor (Oliva and Torralba, 2008)

Texture
Features



How do we describe an image patch?

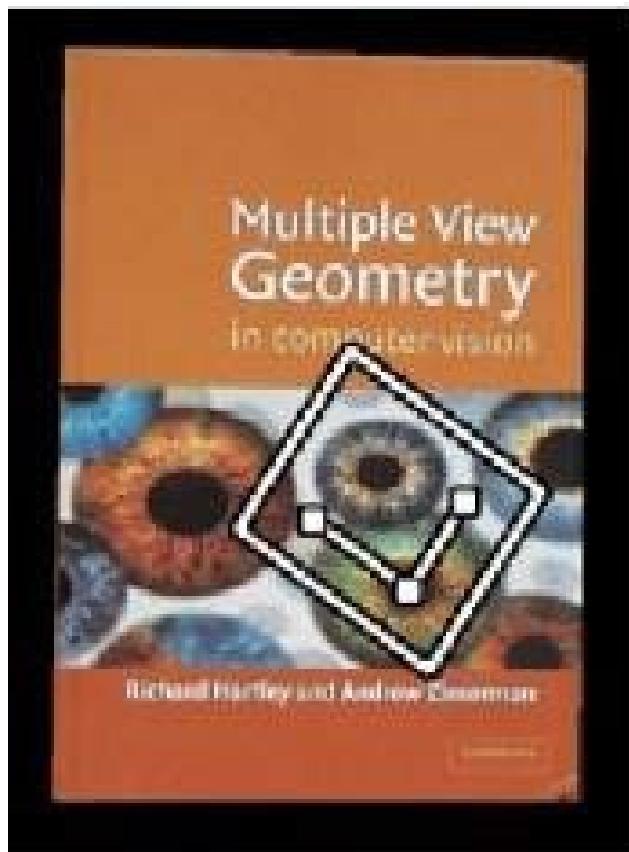
Patches with similar content should have similar descriptors.



Photometric transformations



Geometric transformations



objects will appear at different scales,
translation and rotation

What is the best descriptor for an image feature?

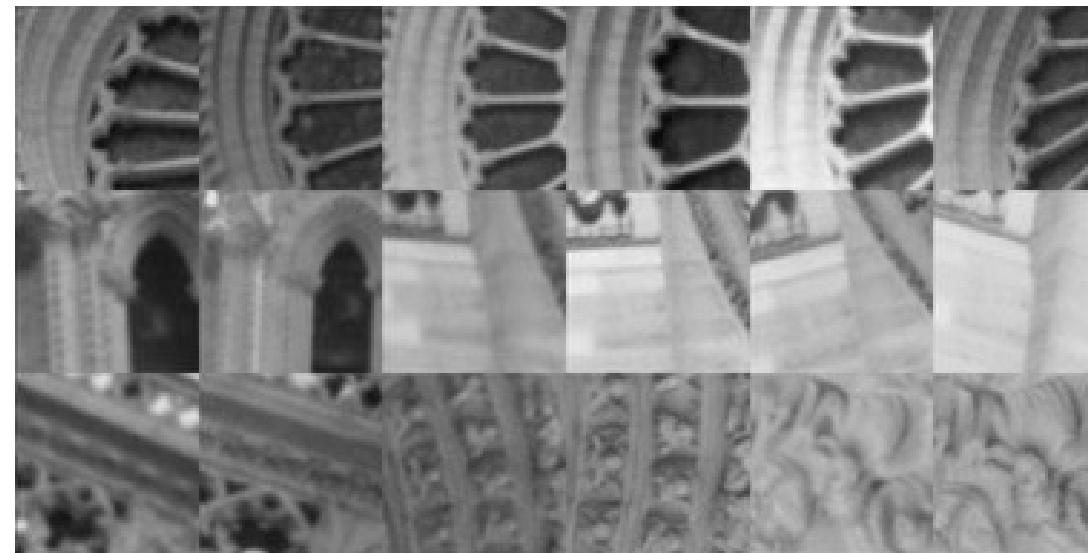
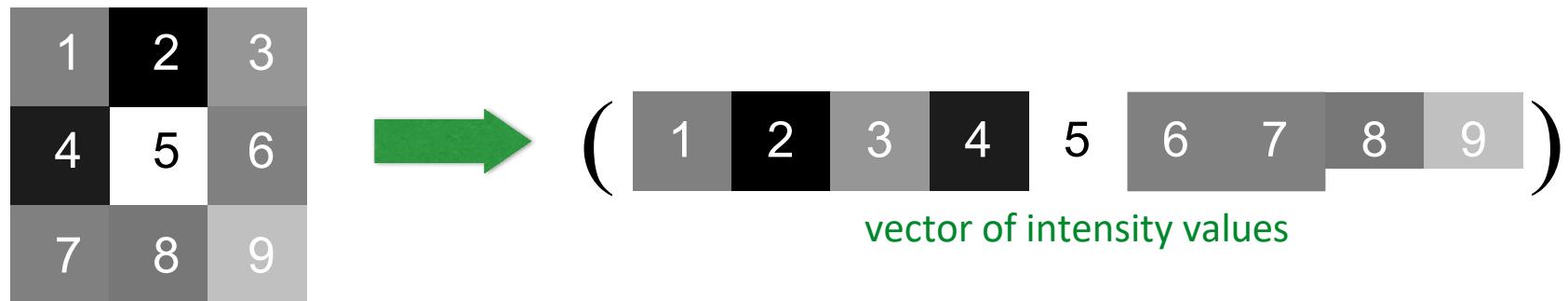


Image patch

Just use the pixel values of the patch!



Perfectly fine if geometry and appearance is unchanged
(a.k.a. template matching)

Tiny Images



Tiny Images

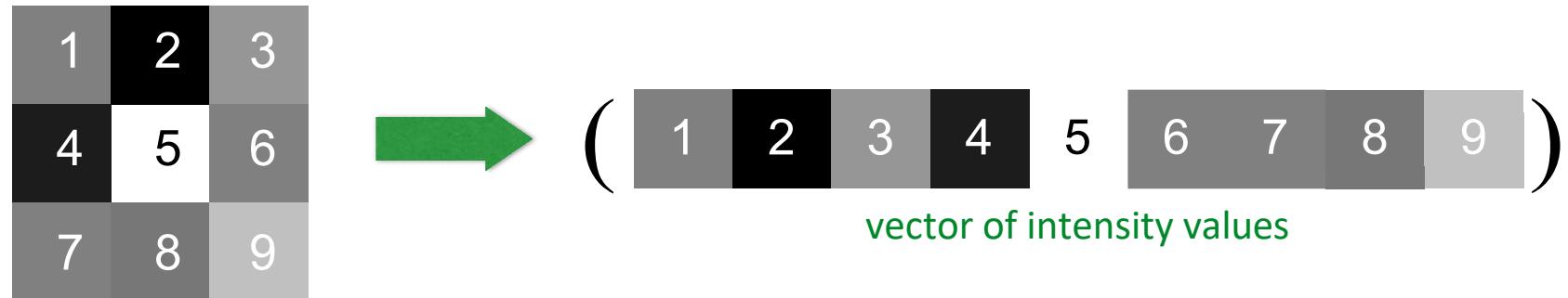
Just down-sample it!

Simple, fast, robust to small affine transforms.



Image patch

Just use the pixel values of the patch!



Perfectly fine if geometry and appearance is unchanged
(a.k.a. template matching)

What are the problems?

How can you be less sensitive to absolute intensity values?

Image gradients

Use pixel differences

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |



$$\left(\begin{array}{ccccccc} - & + & + & - & - & + \end{array} \right)$$

vector of x derivatives

'binary descriptor'

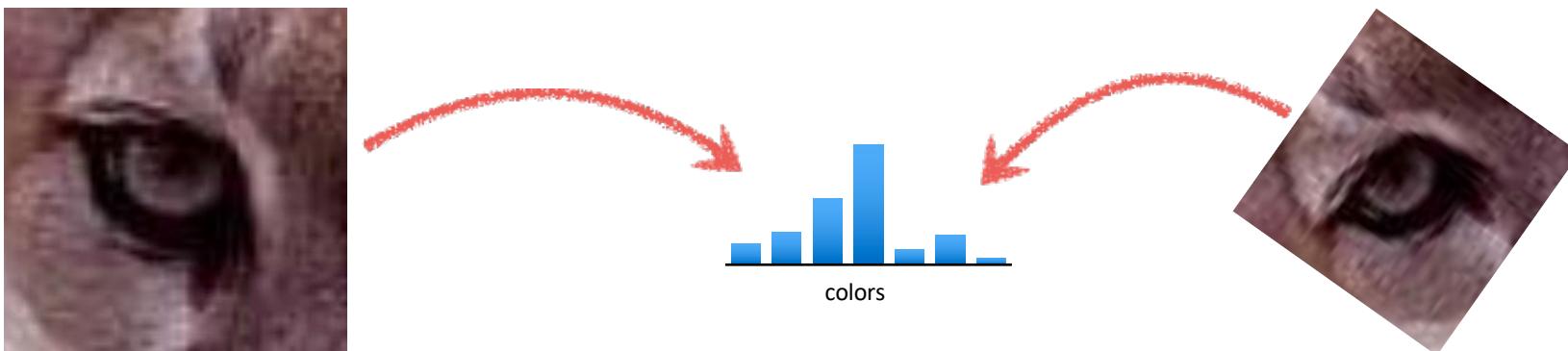
Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Color histogram

Count the colors in the image using a histogram

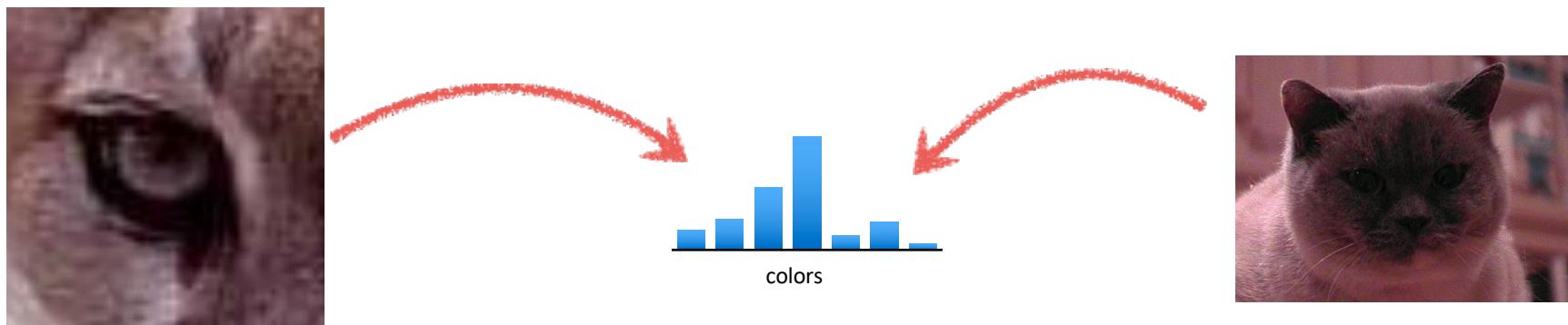


Invariant to changes in scale and rotation

What are the problems?

Color histogram

Count the colors in the image using a histogram



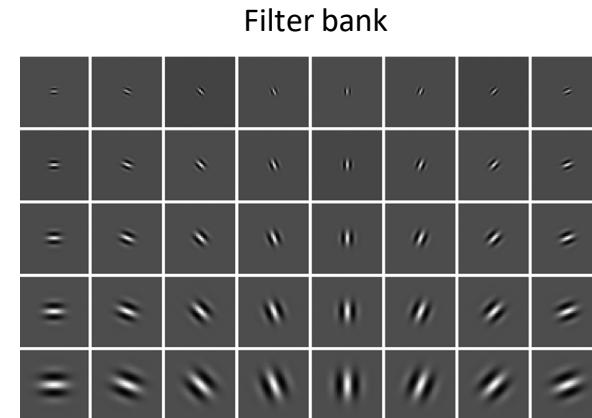
Invariant to changes in scale and rotation

What are the problems?

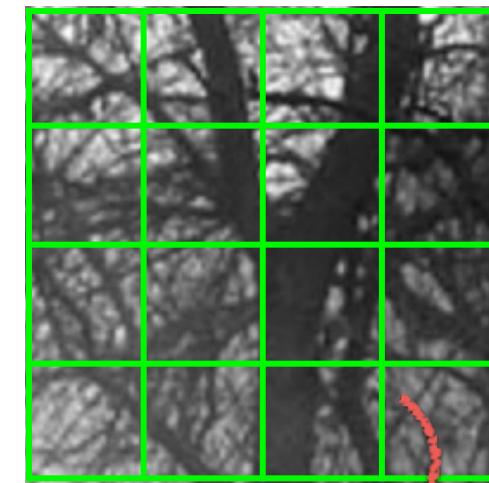
How can you be more sensitive to spatial layout?

GIST

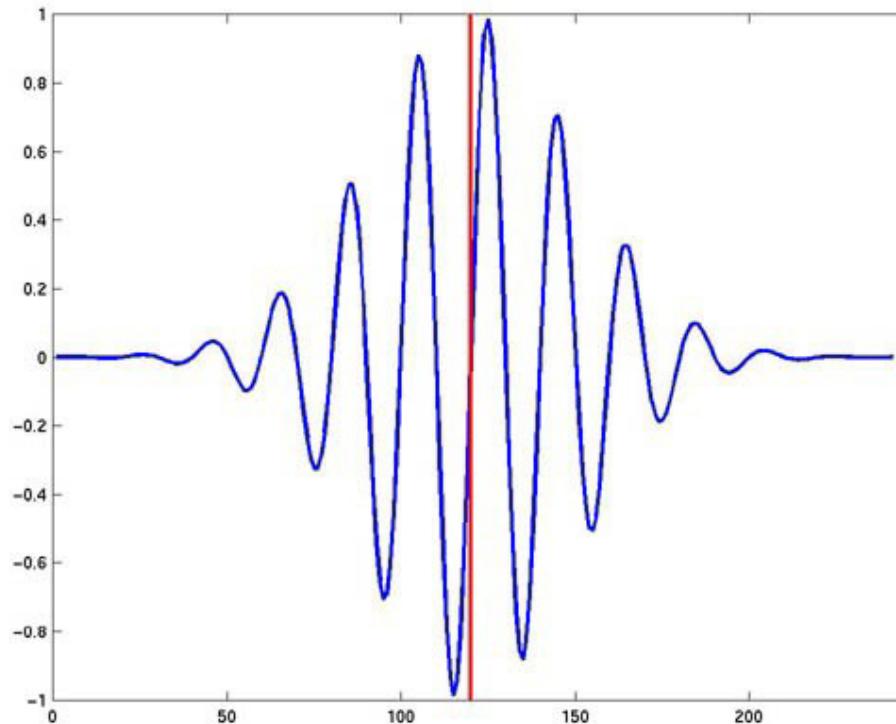
1. Compute filter responses
(filter bank of Gabor filters)
2. Divide image patch into 4×4 cells
3. Compute filter response averages for each cell
4. Size of descriptor is $4 \times 4 \times N$,
where N is the size of the
filter bank



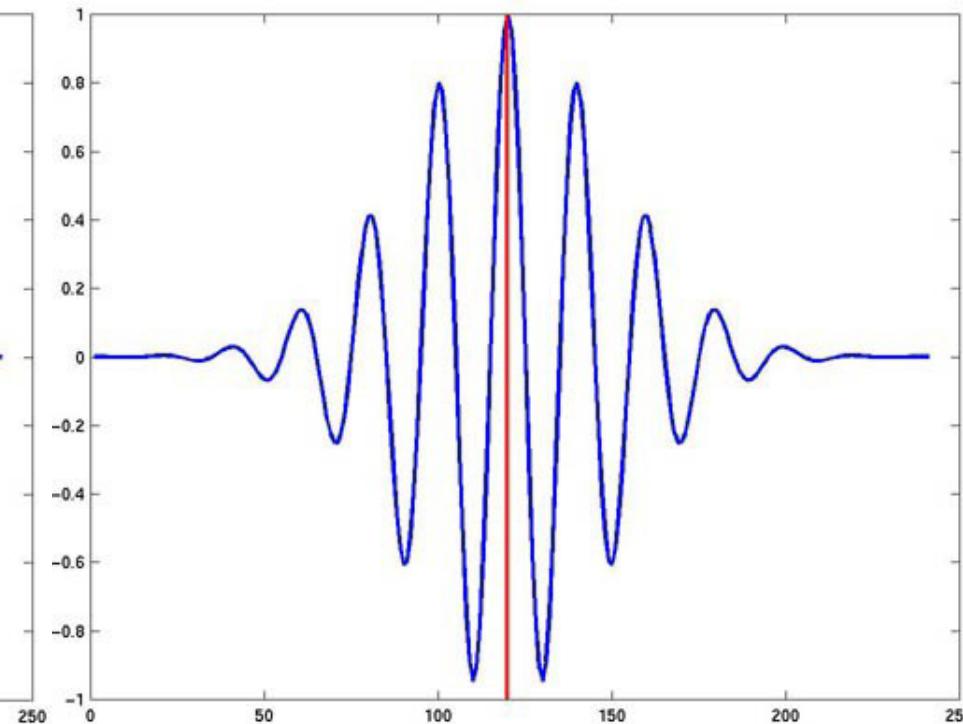
4×4 cell



Gabor Filters (1D examples)



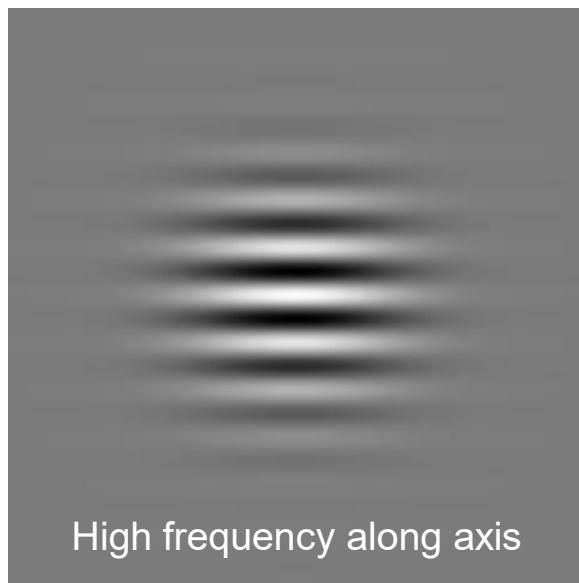
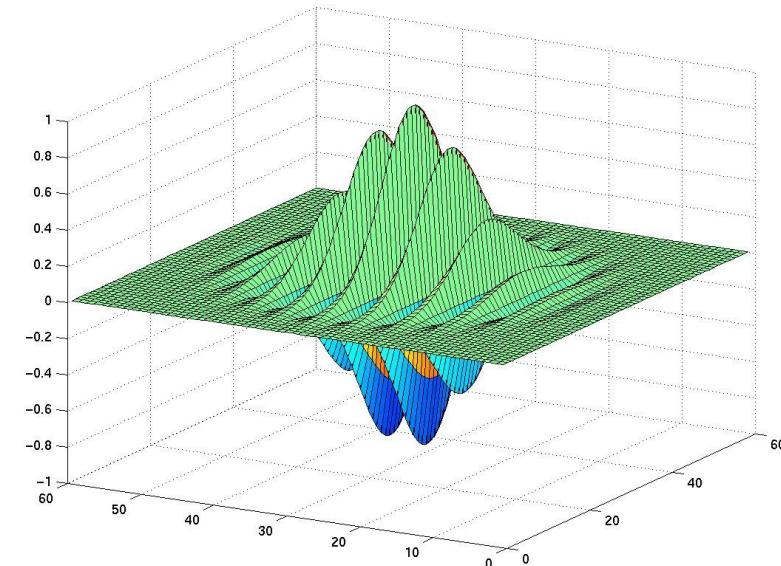
$$e^{-\frac{x^2}{2\sigma^2}} \sin(2\pi\omega x)$$



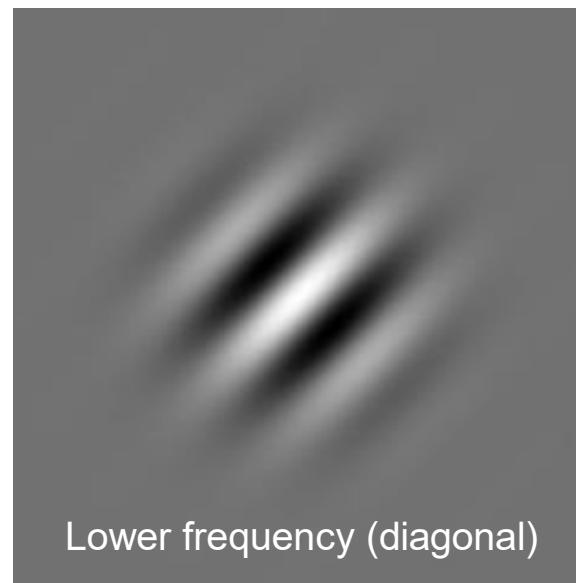
$$e^{-\frac{x^2}{2\sigma^2}} \cos(2\pi\omega x)$$

2D Gabor Filters

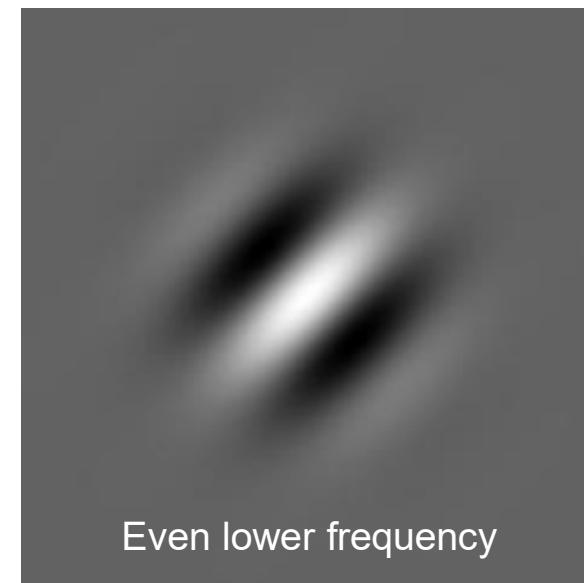
$$e^{-\frac{x^2+y^2}{2\sigma^2}} \cos(2\pi(k_x x + k_y y))$$



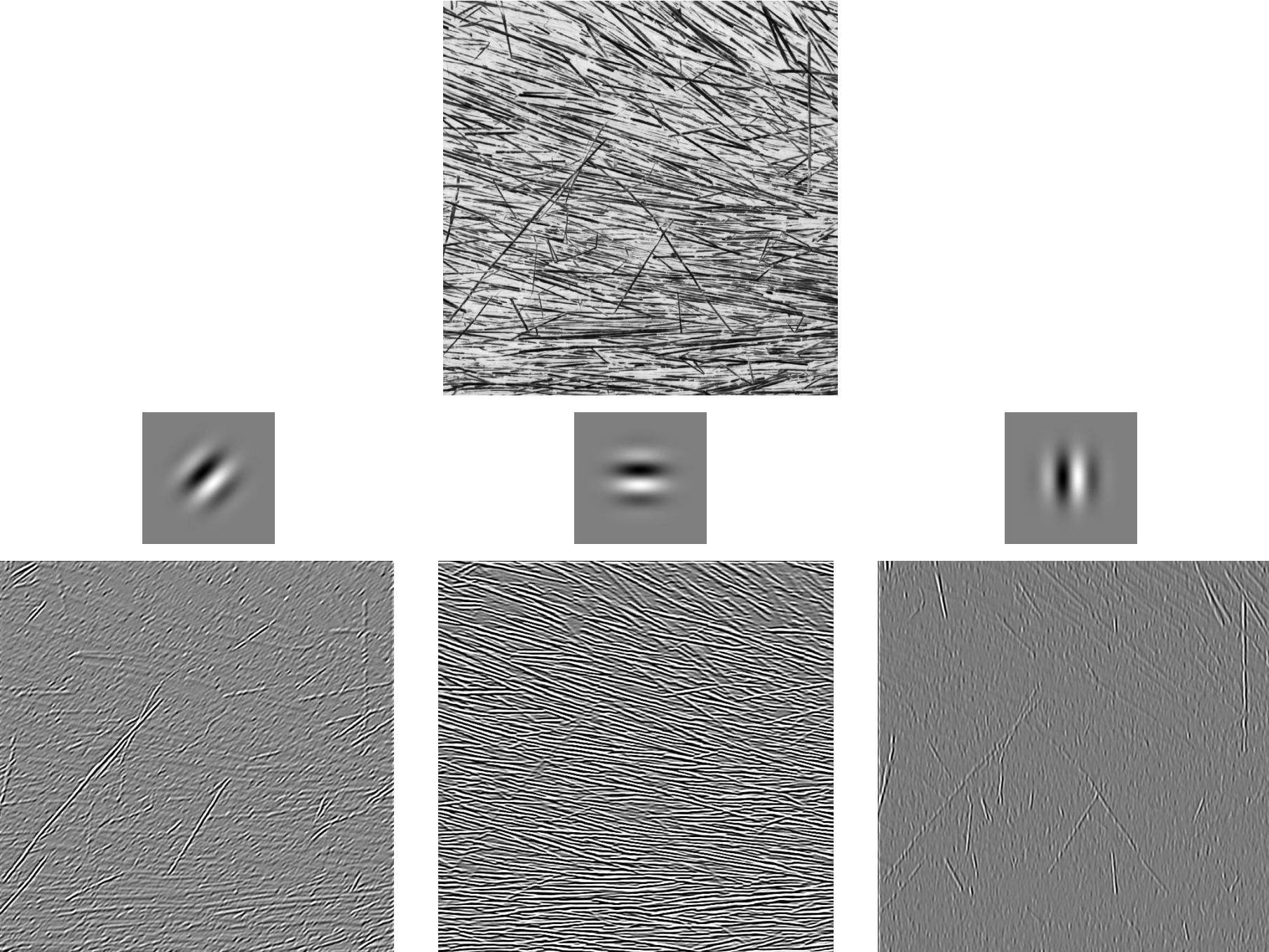
High frequency along axis



Lower frequency (diagonal)

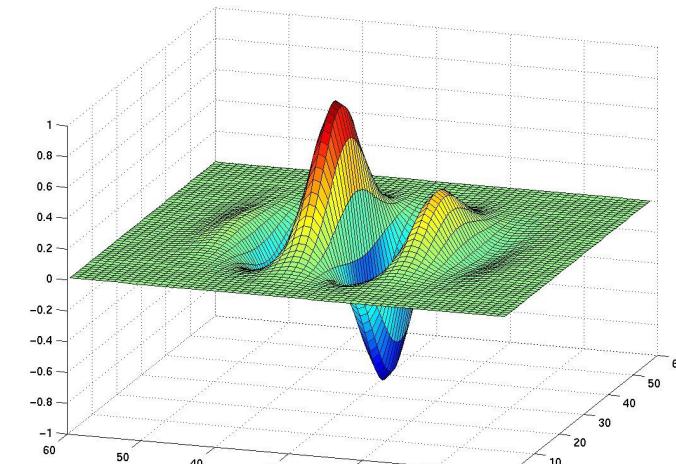
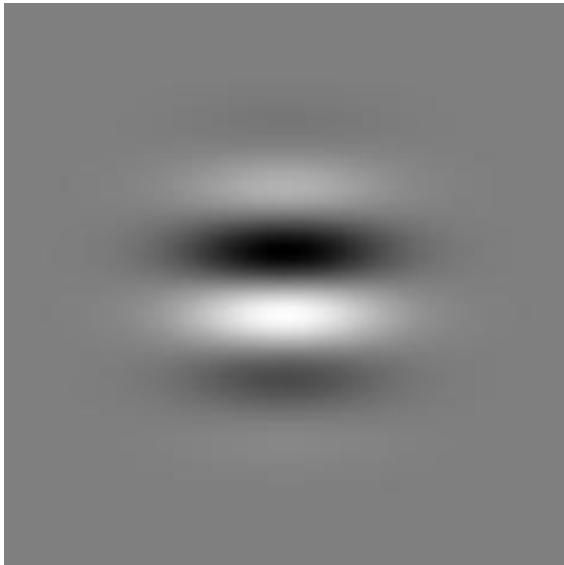


Even lower frequency

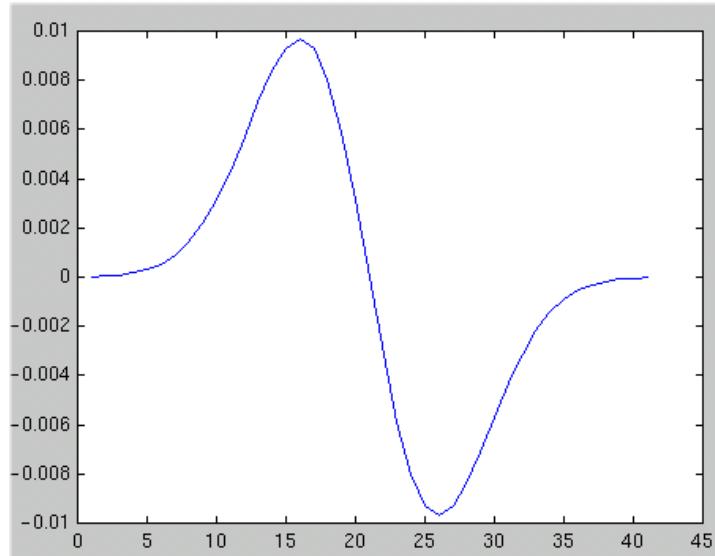


Kaveh Fathian

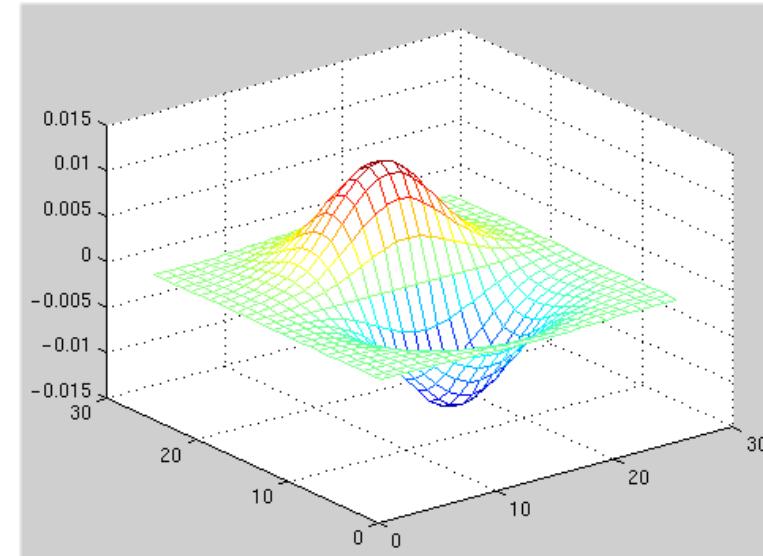
Odd Gabor filter



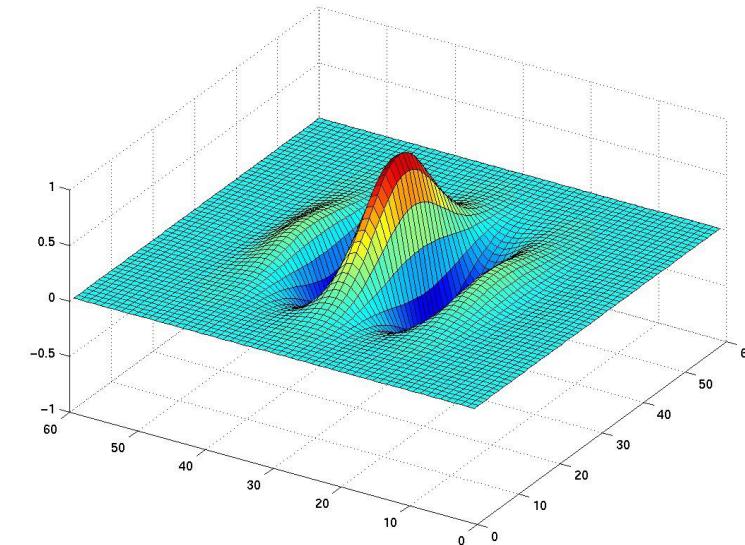
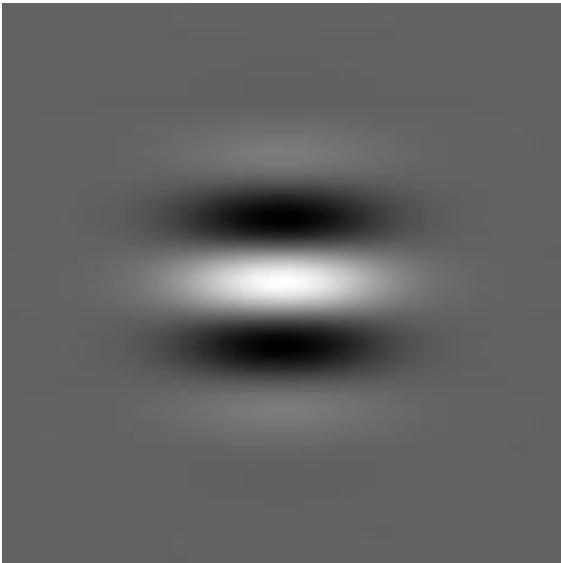
... looks a lot like...



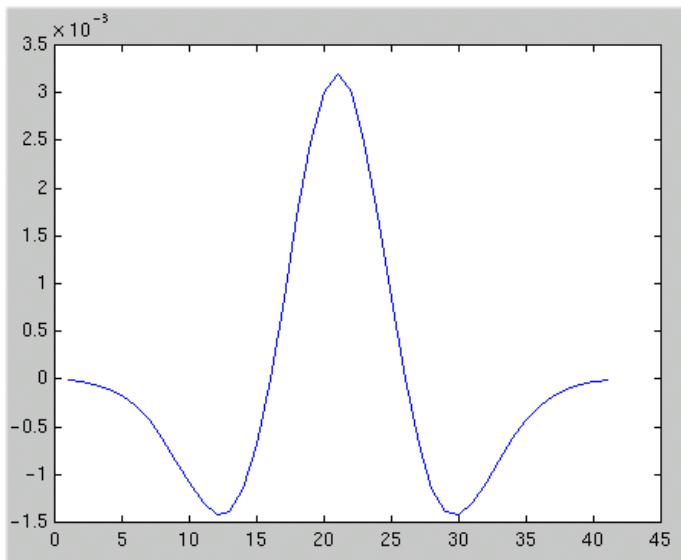
Gaussian
Derivative



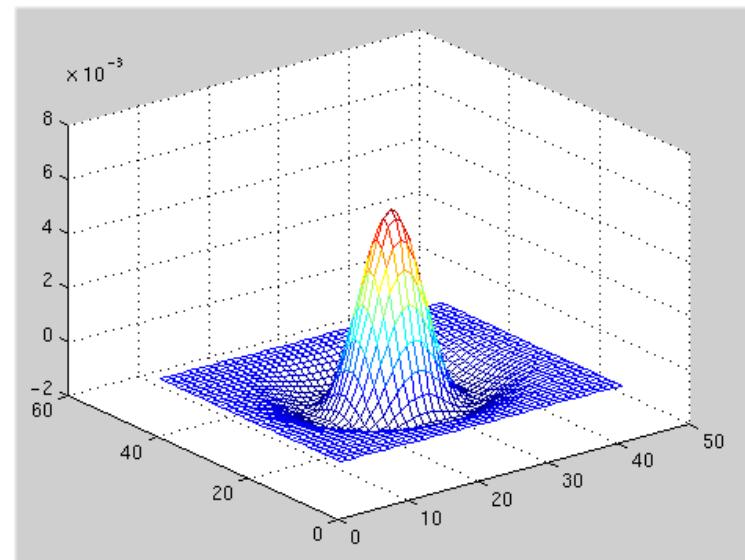
Even Gabor filter



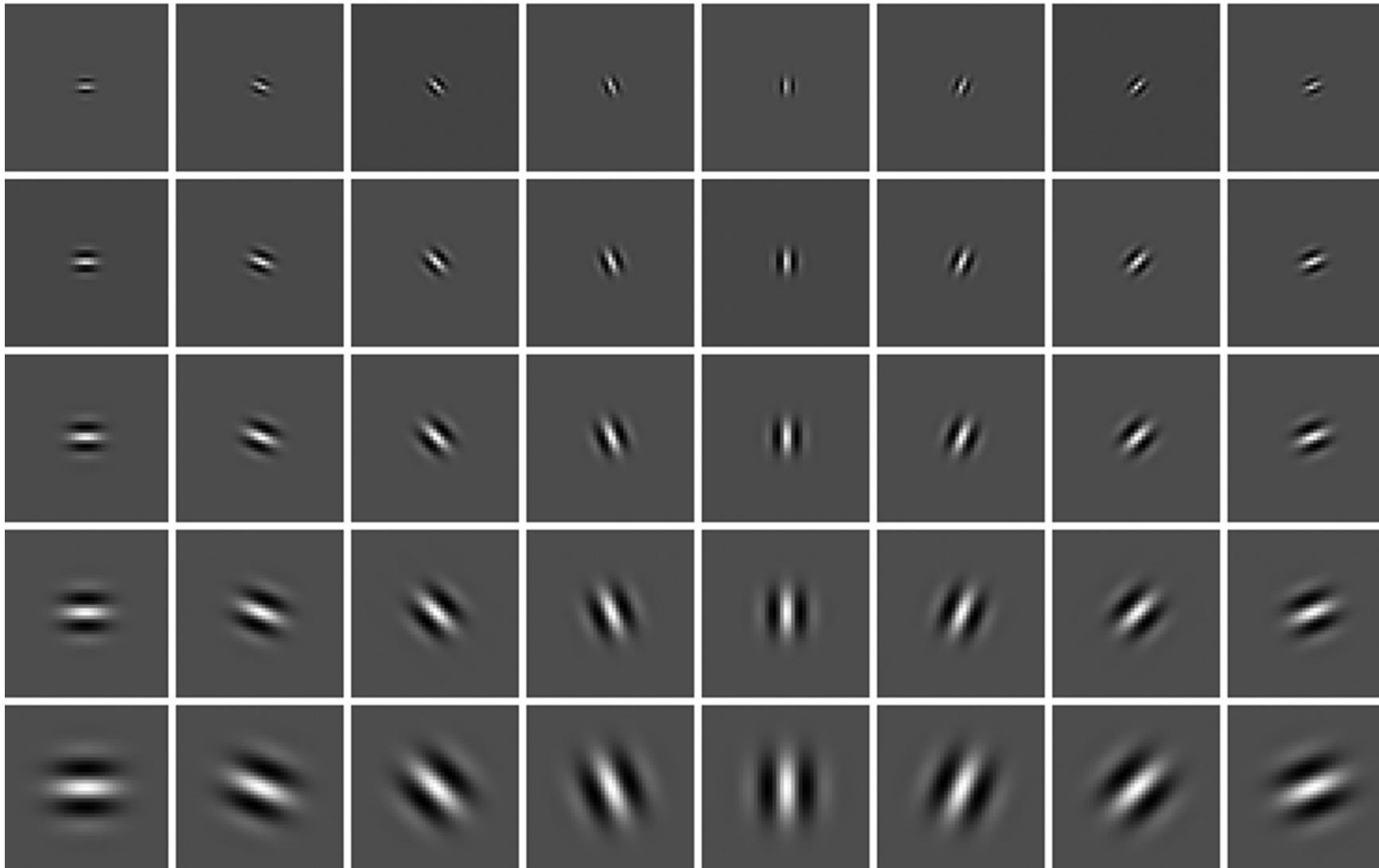
... looks a lot like...



Laplacian

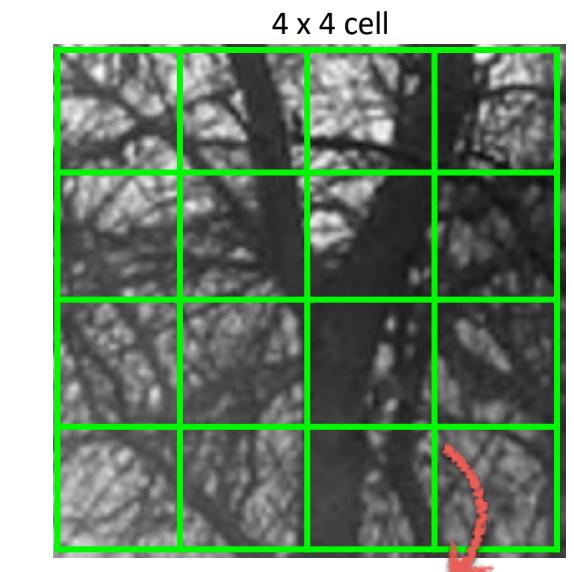
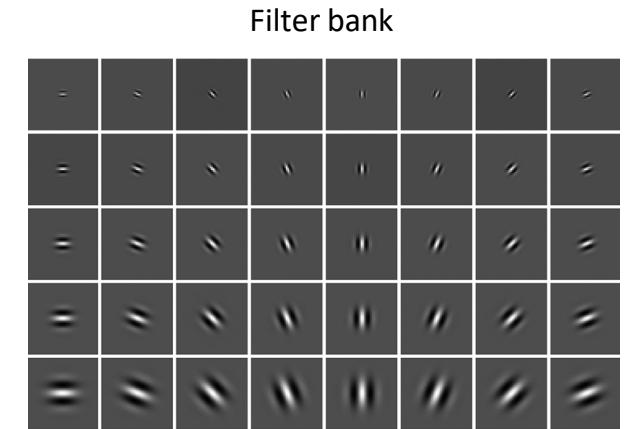


Directional edge detectors



GIST

1. Compute filter responses (filter bank of Gabor filters)
2. Divide image patch into 4×4 cells
3. Compute filter response averages for each cell
4. Size of descriptor is $4 \times 4 \times N$, where N is the size of the filter bank



averaged filter responses

What is the GIST descriptor encoding?

Rough spatial distribution of image gradients

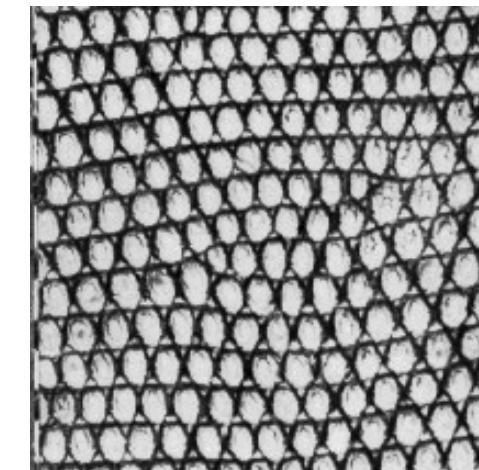
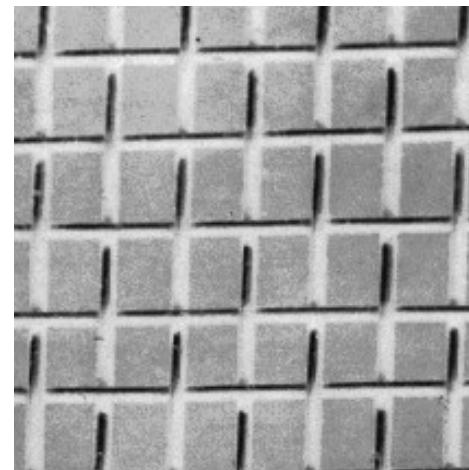
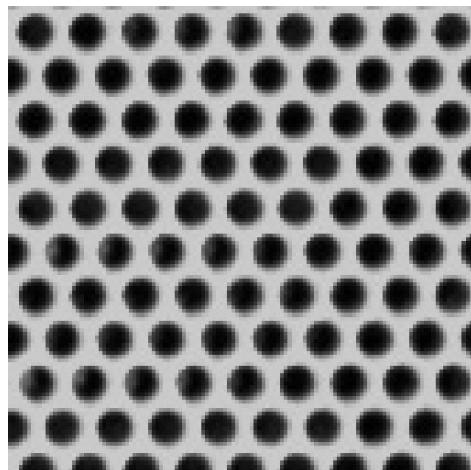
Textons

Julesz. Textons, the elements of texture perception, and their interactions. Nature 1981

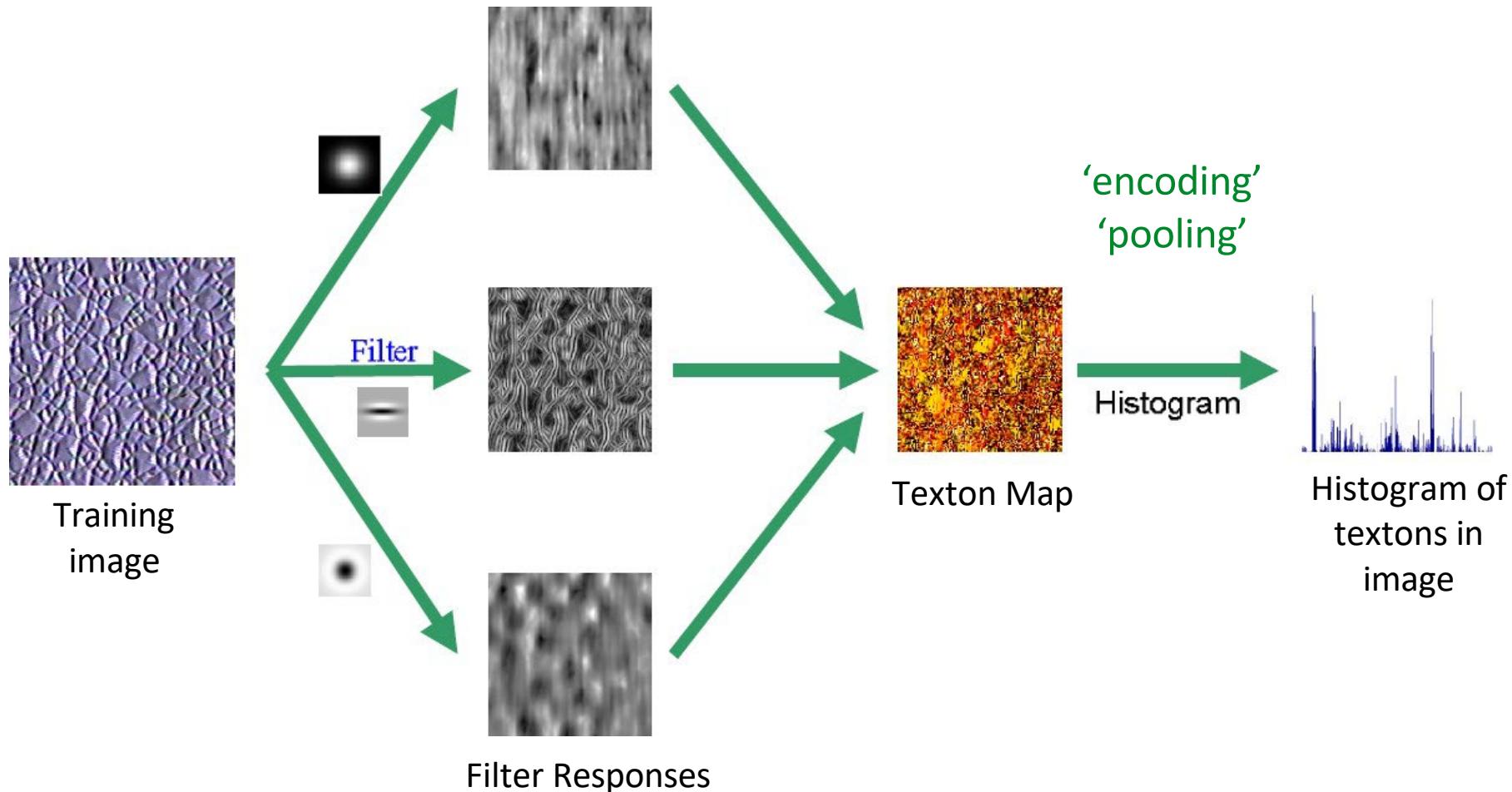
Texture is characterized by the repetition of basic elements or **textons**



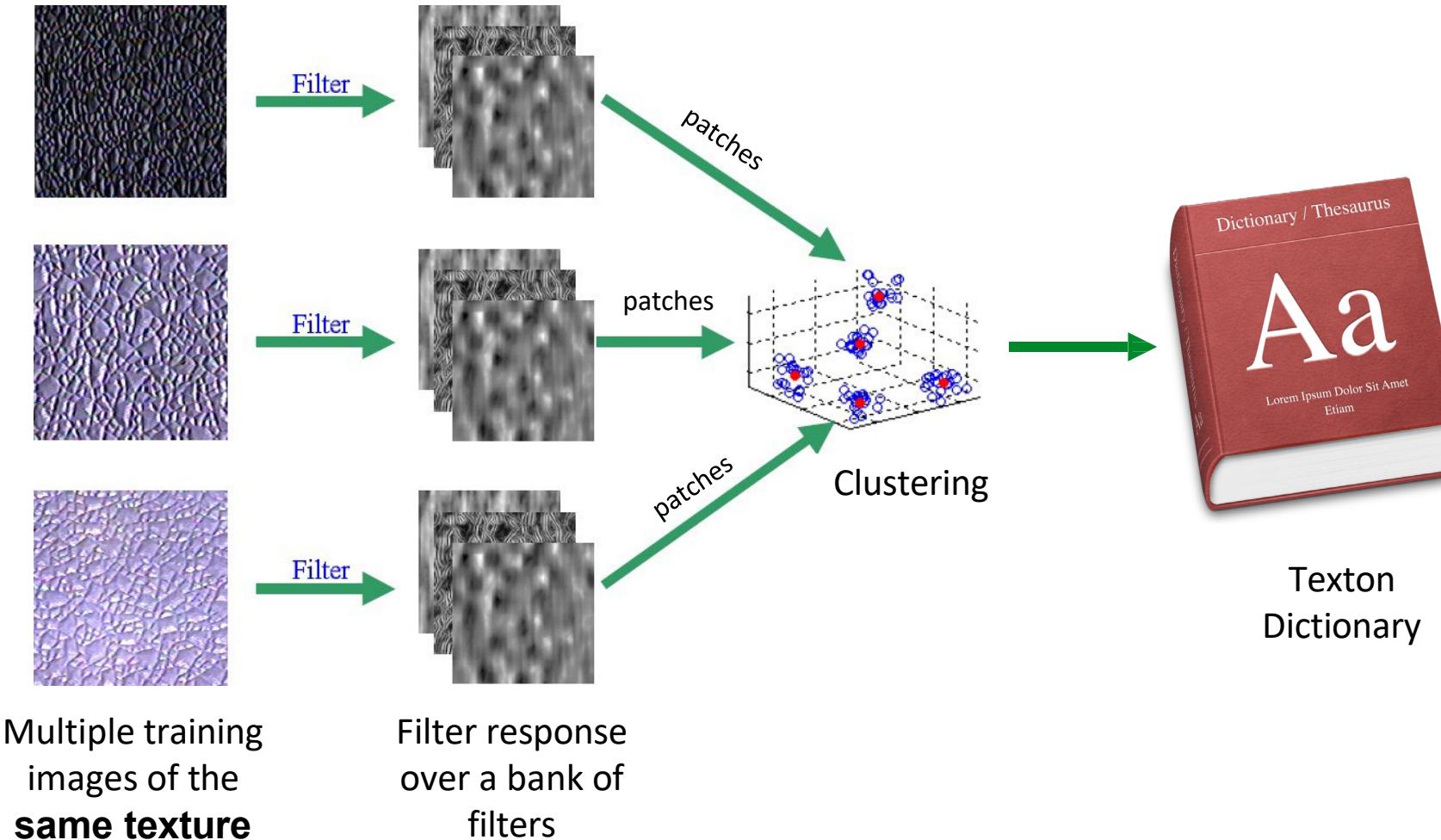
For stochastic textures, it is the identity of the **textons**, not
their spatial arrangement, that matters



Histogram of Textons descriptor

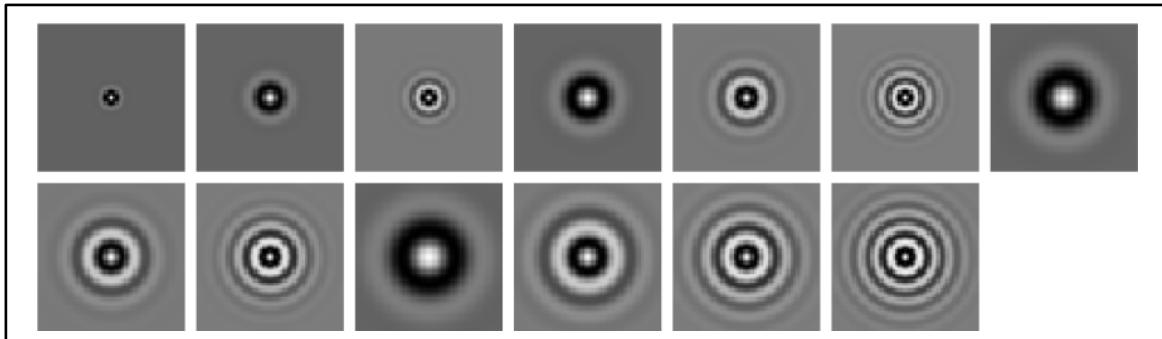


Learning Textons from data



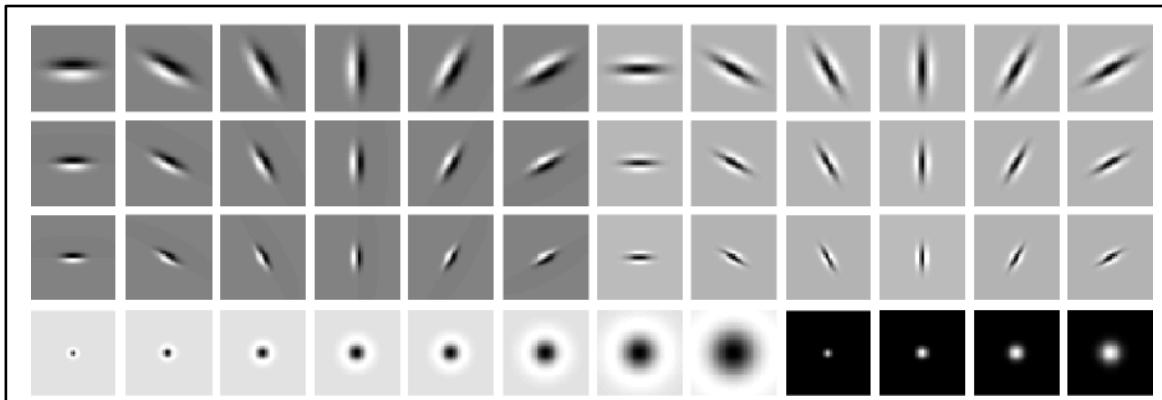
Example of Filter Banks

Isotropic Gabor

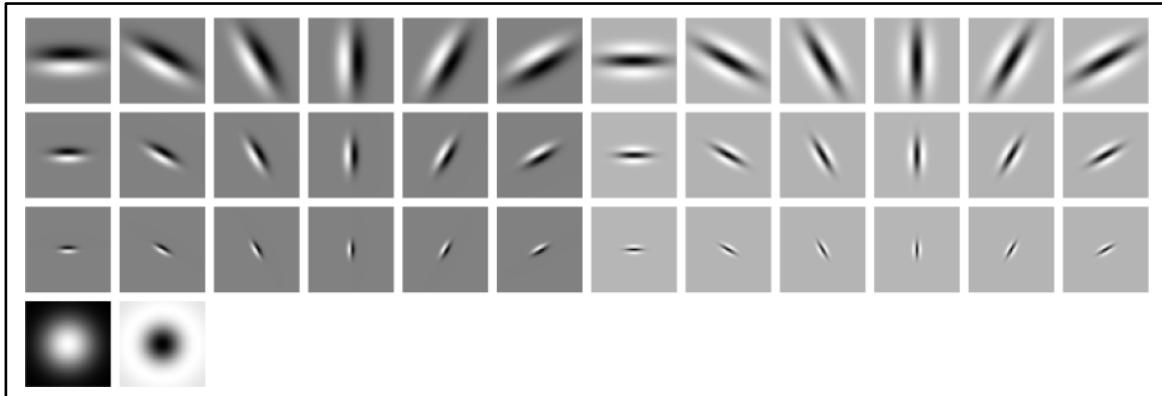


'S'

Gaussian derivatives at different scales and orientations

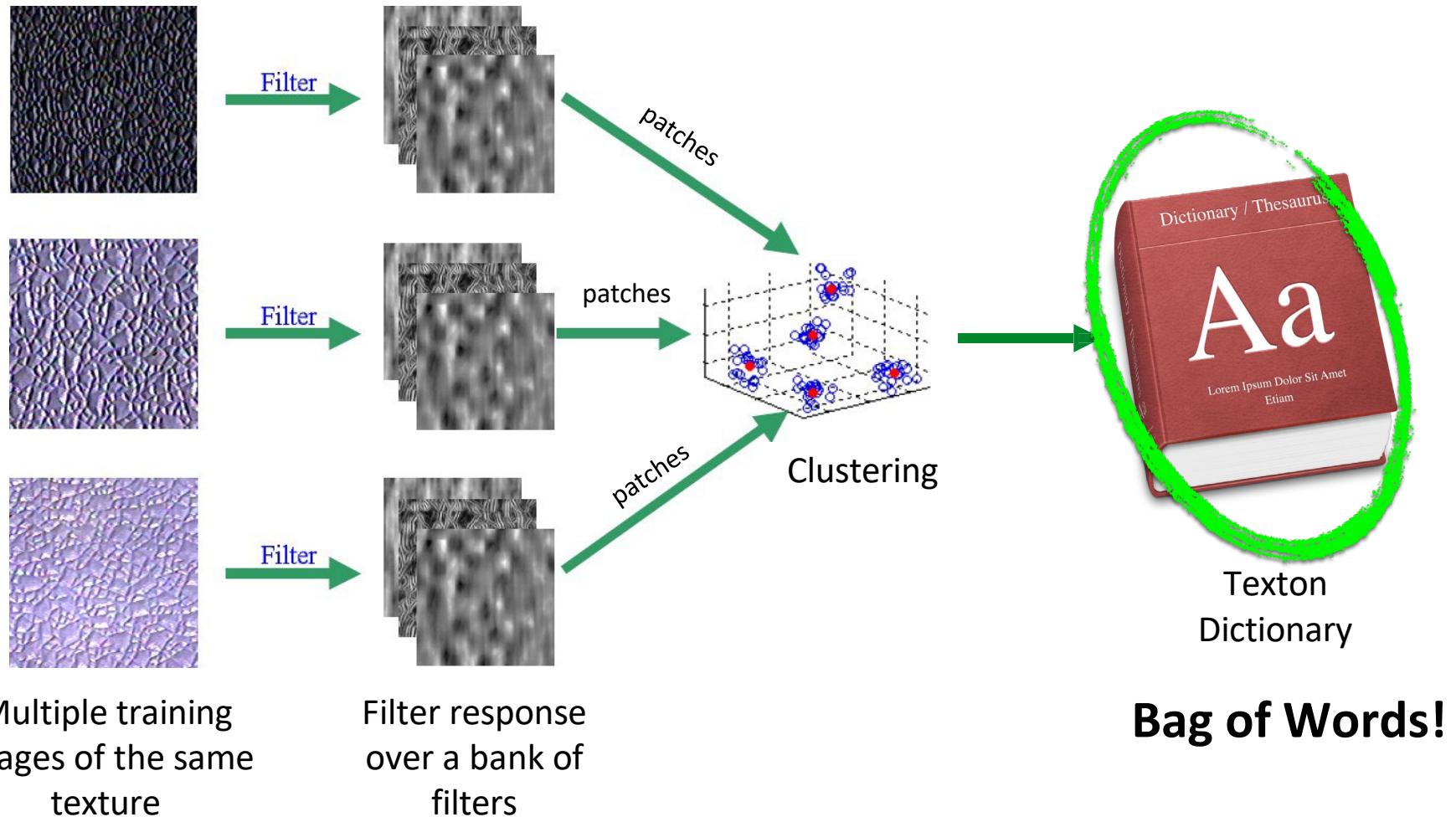


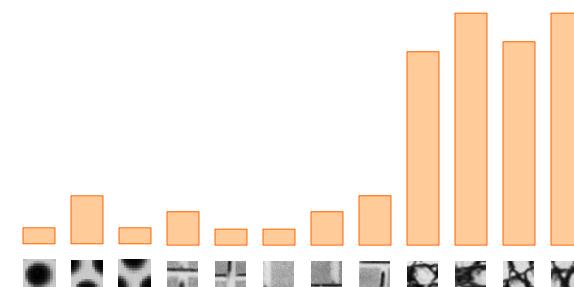
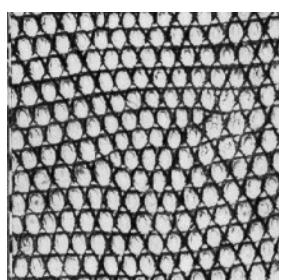
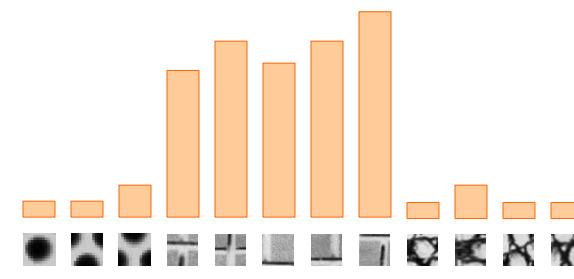
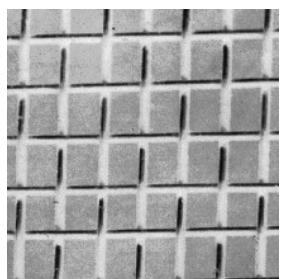
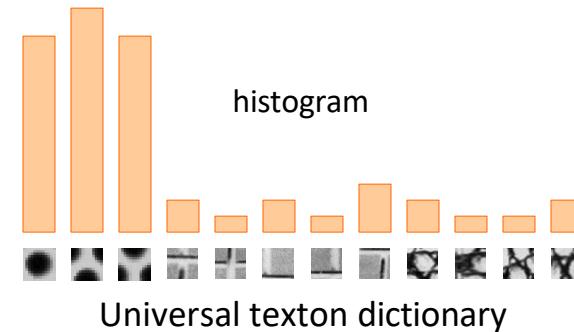
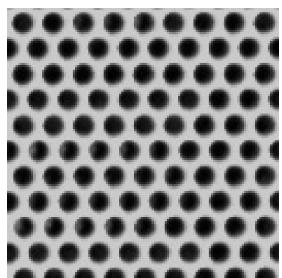
'LM'



'MR8'

Learning Textons from data

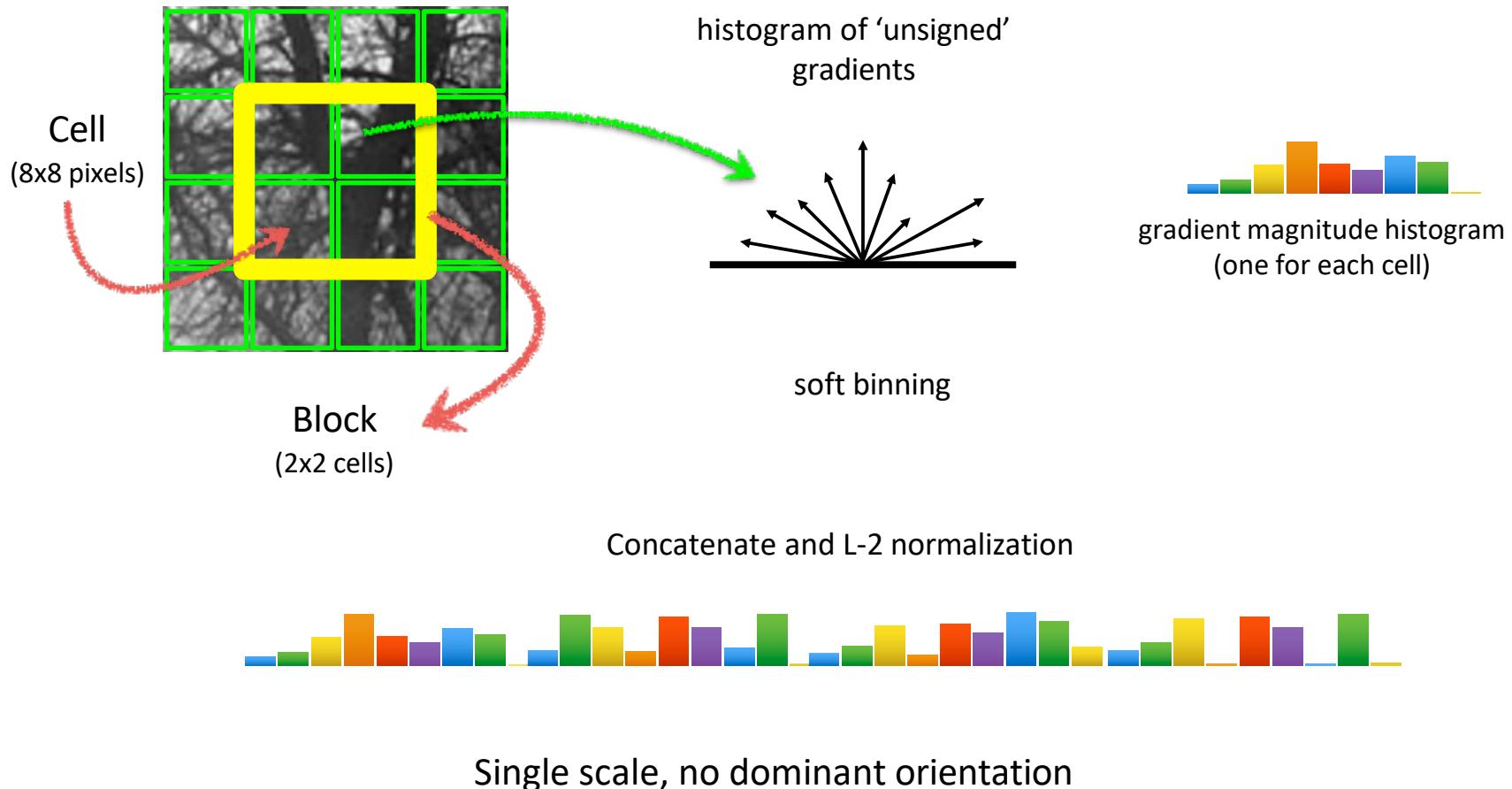




Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001;
Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

HOG Descriptor

Dalal, Triggs. **Histograms of Oriented Gradients** for Human Detection. CVPR, 2005



HOG Pedestrian Detection

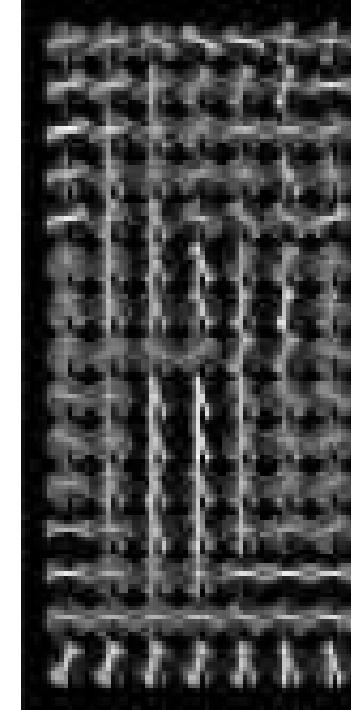
1 cell step size

128 pixels
16 cells
15 blocks



$$15 \times 7 \times 4 \times 9 = 3780$$

visualization



64 pixels
8 cells
7 blocks

Redundant representation due to overlapping blocks
How many times is each inner cell encoded?



<http://chrisjmccormick.wordpress.com/2013/05/09/hog-person-detector-tutorial/>