

Spring 2022

INTRODUCTION TO COMPUTER VISION

Atlas Wang

Assistant Professor, The University of Texas at Austin

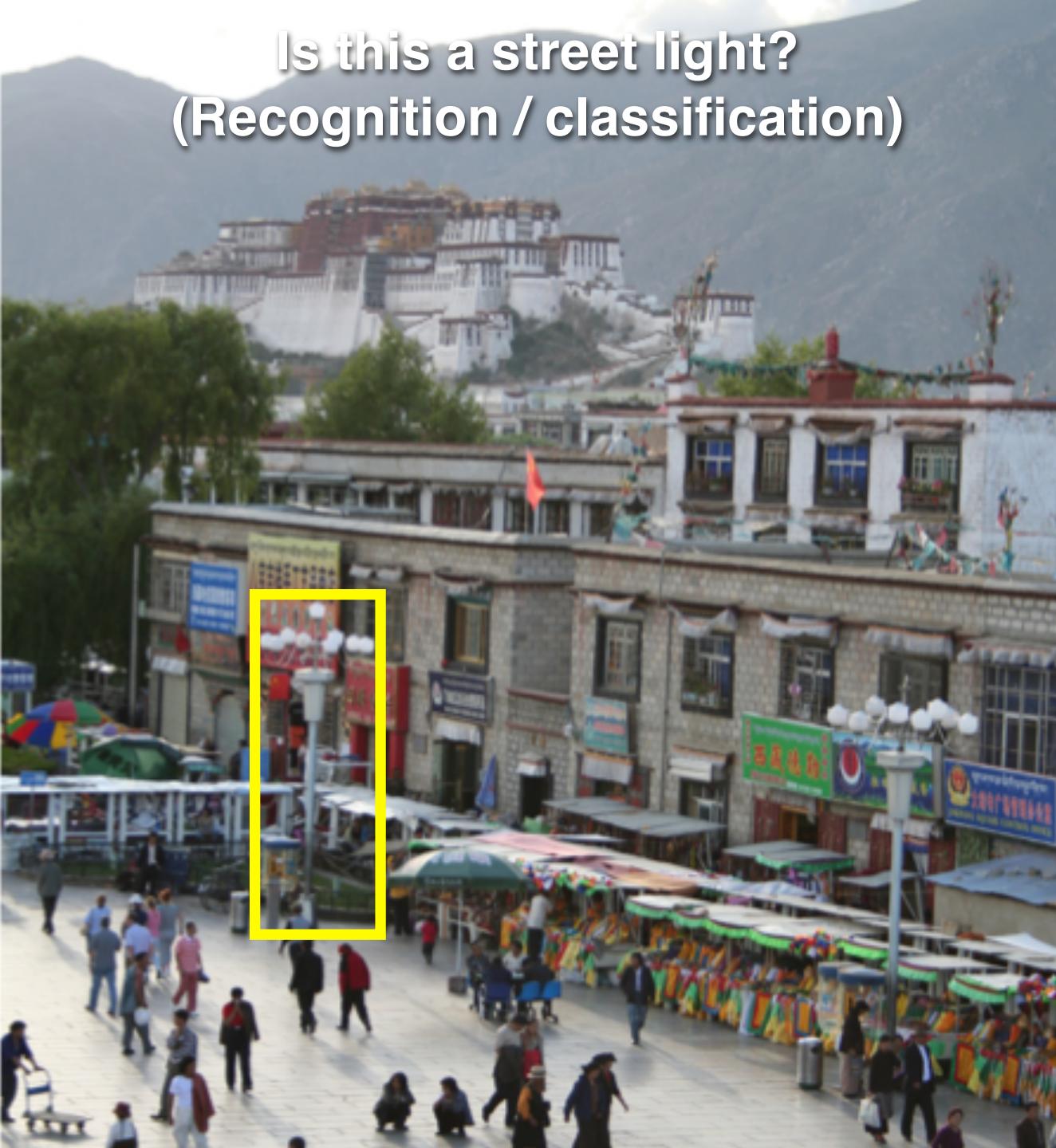
Visual Informatics Group@UT Austin
<https://vita-group.github.io/>

Many slides here were adapted from CMU 16-385

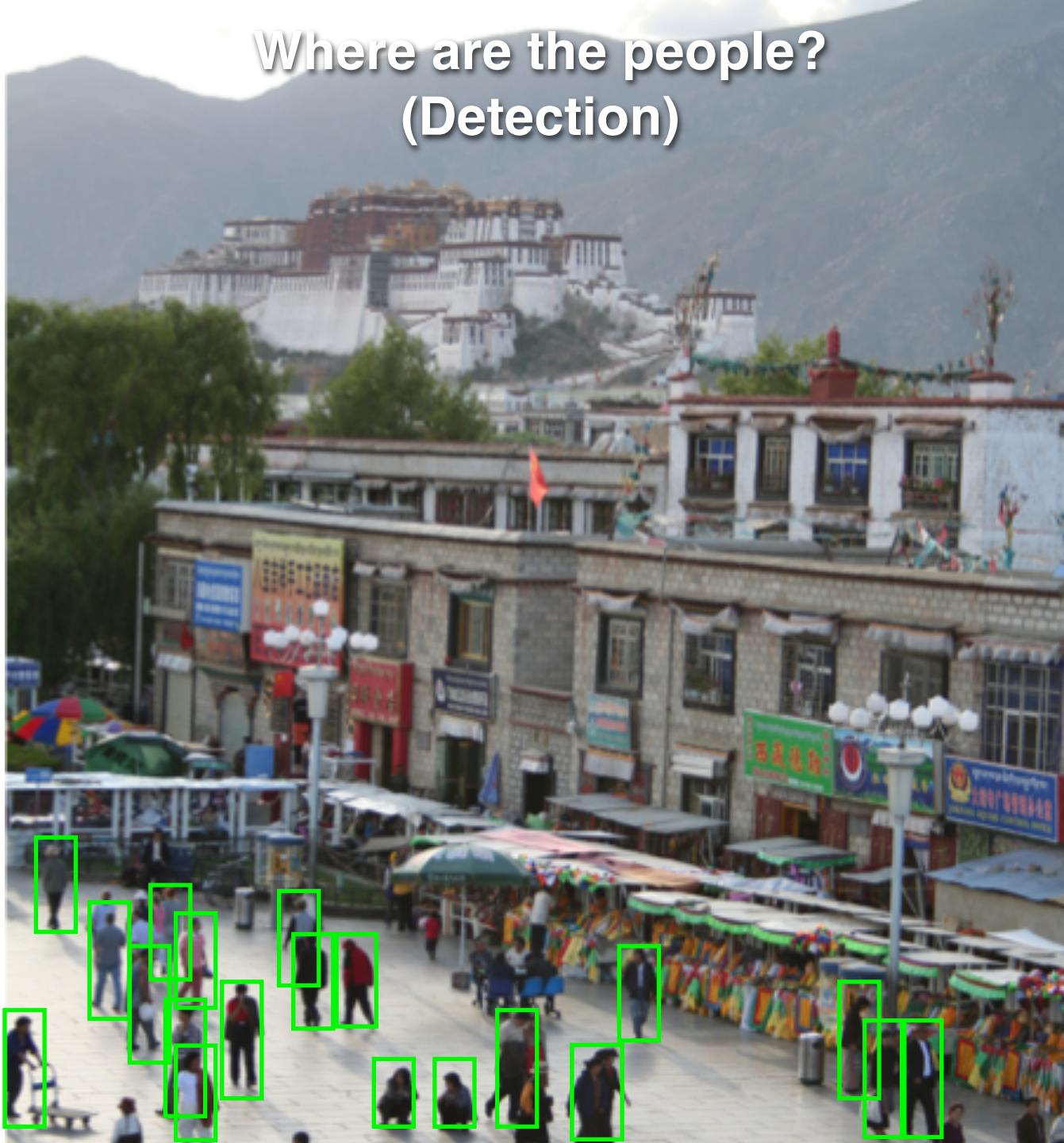
What do we mean by learning-based vision or “semantic vision”?

Is this a street light?

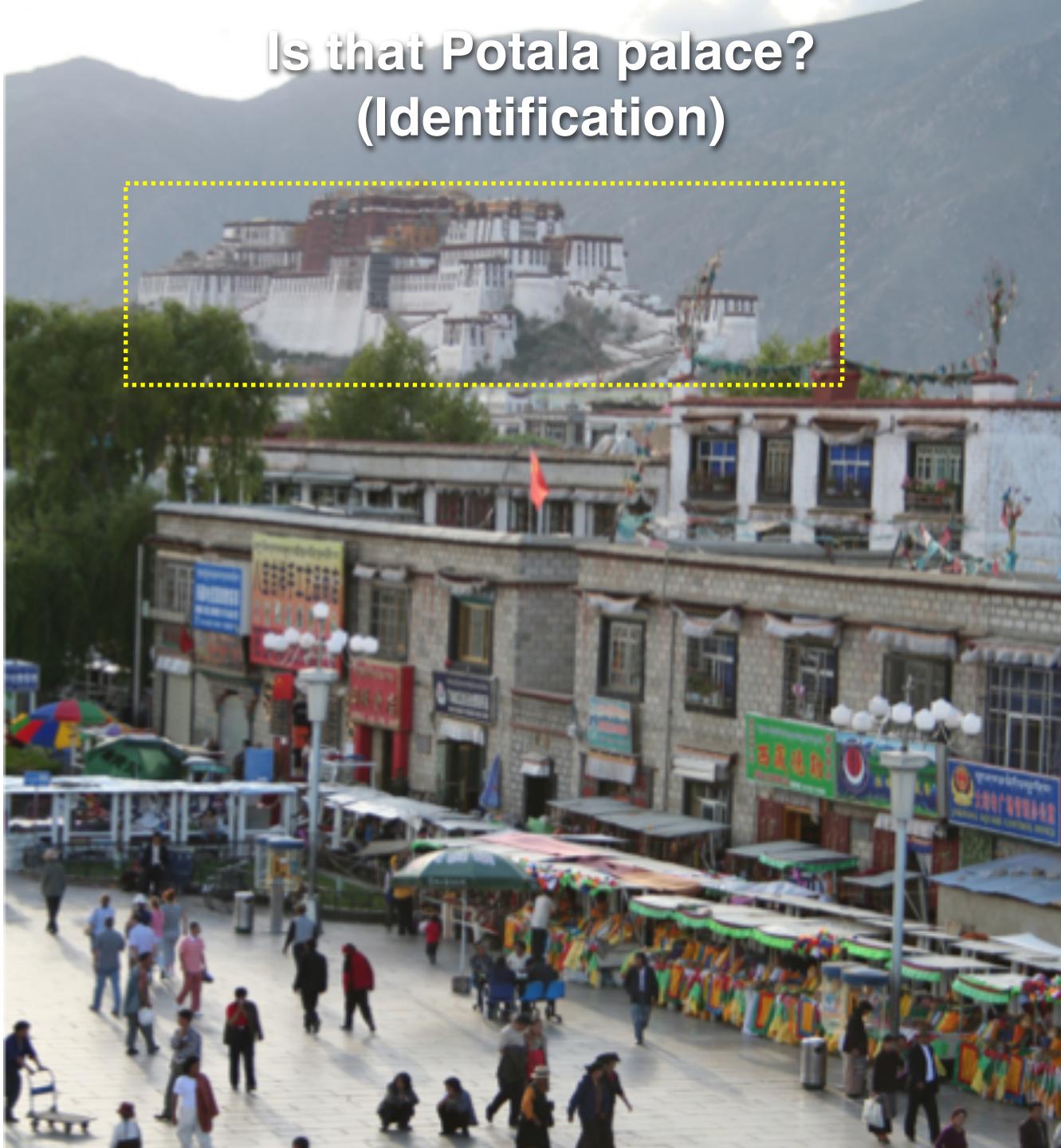
(Recognition / classification)



Where are the people? (Detection)

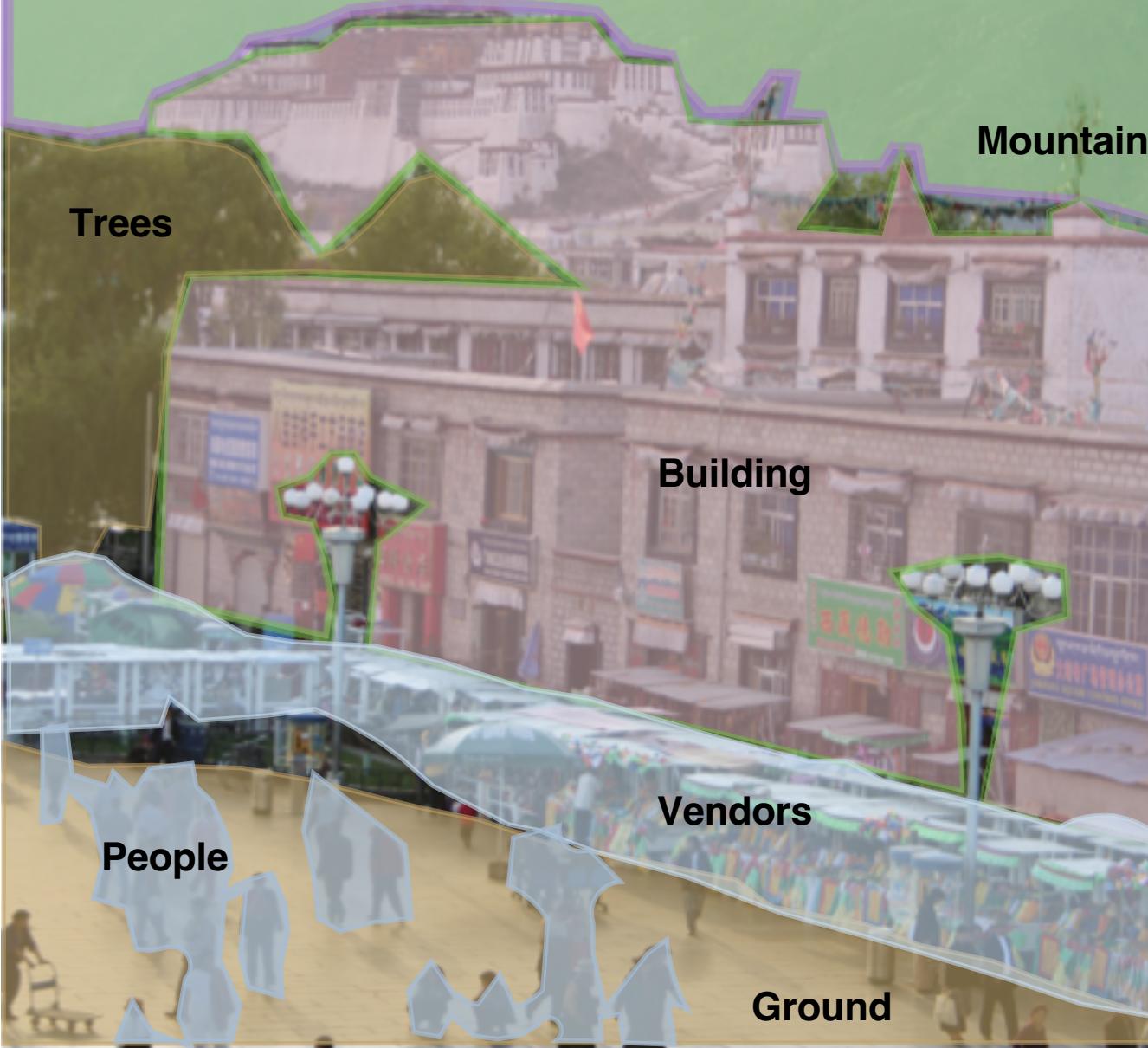


Is that Potala palace? (Identification)



Sky

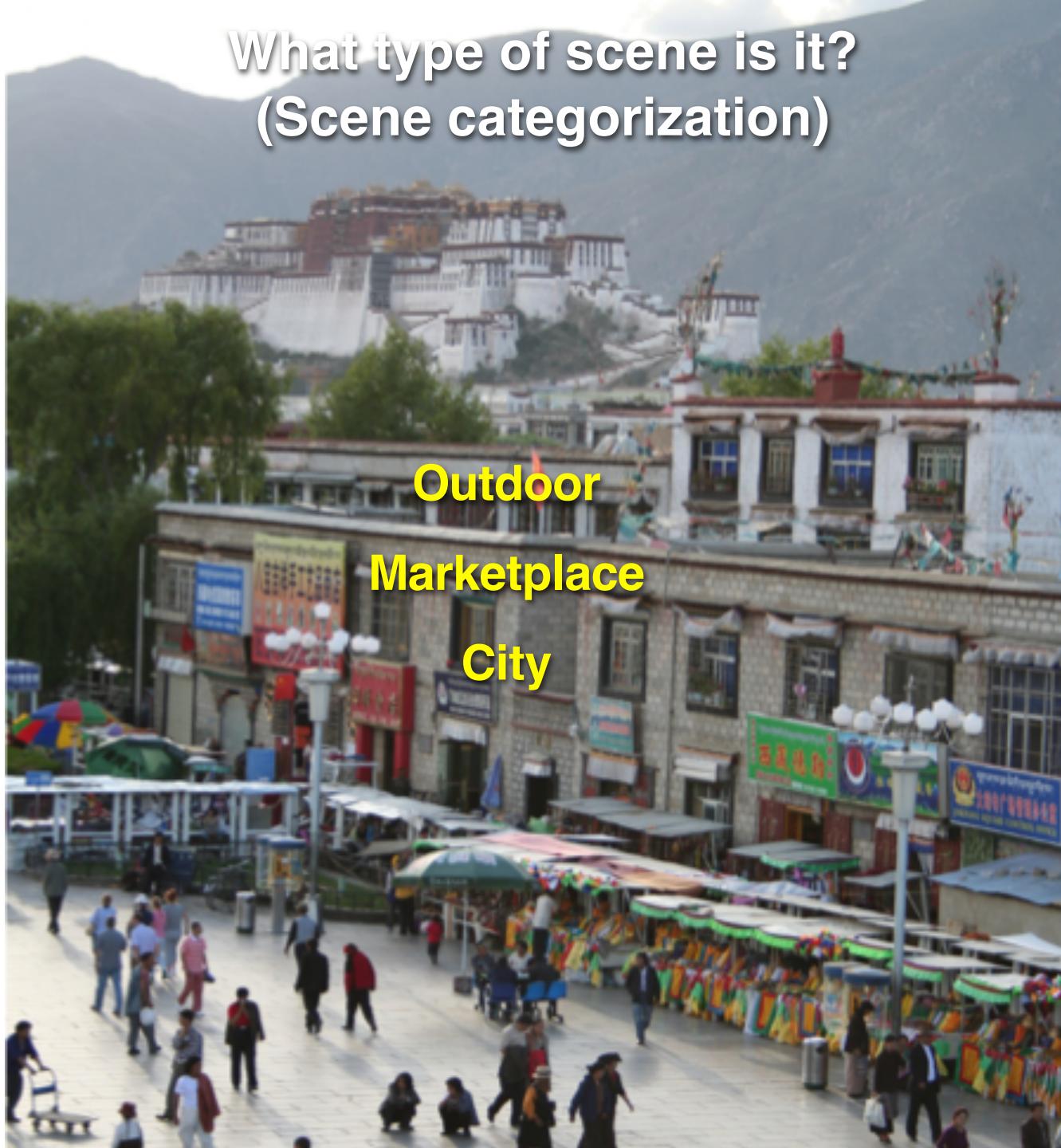
What's in the scene? (semantic segmentation)



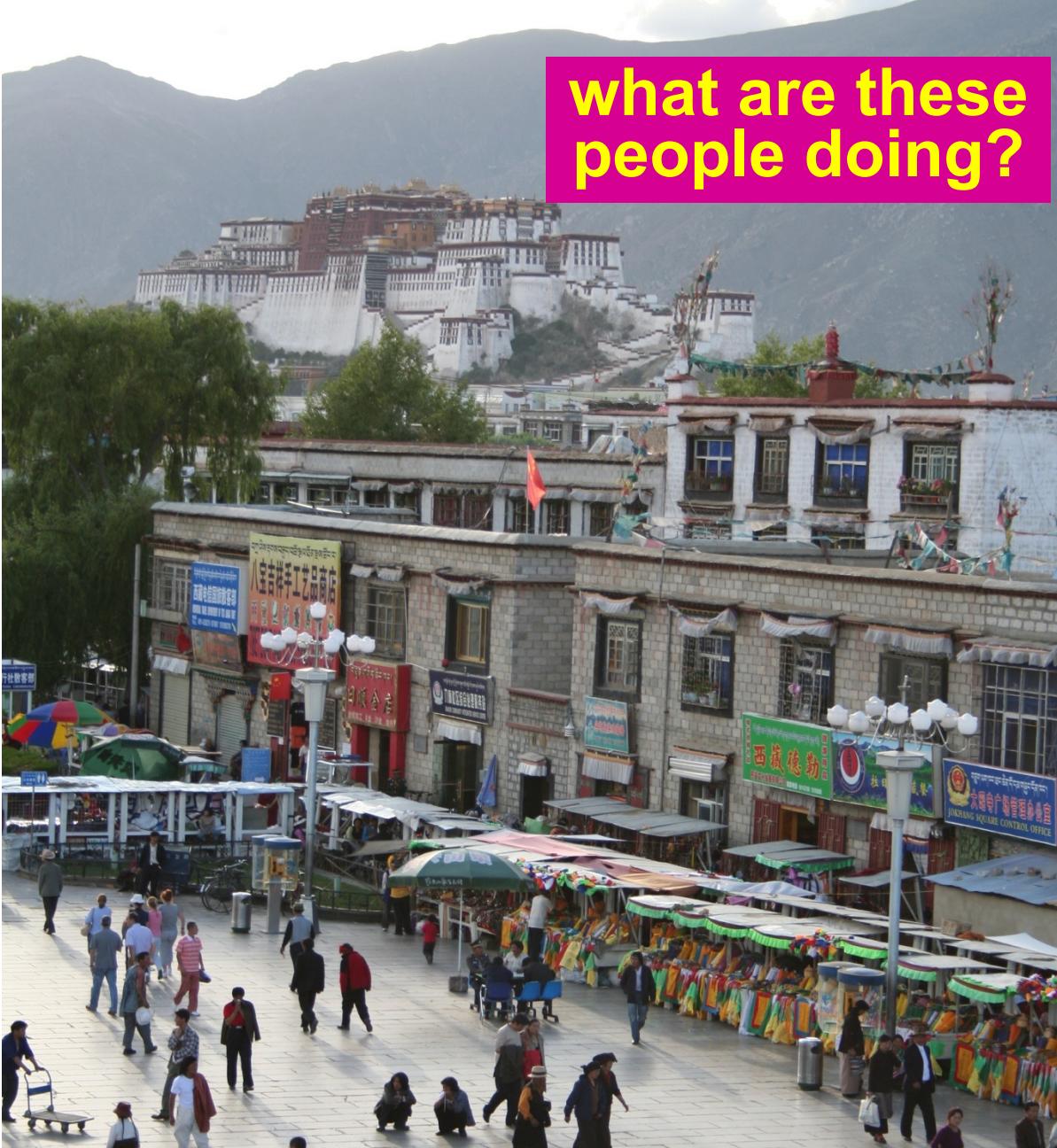
Object categorization



What type of scene is it? (Scene categorization)



Activity / Event Recognition



what are these
people doing?

Object recognition

Is it really so hard?

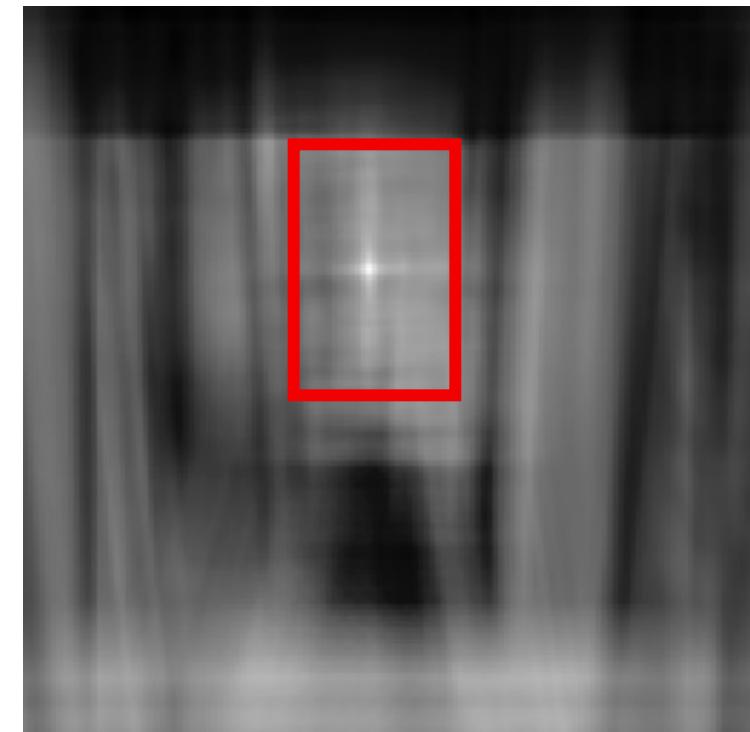
This is a chair



Find the chair in this image



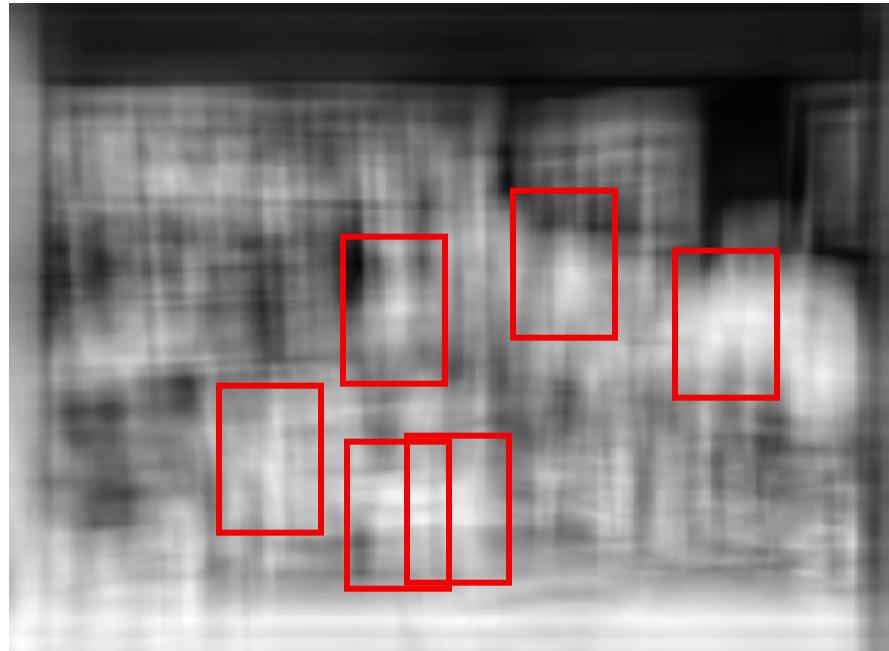
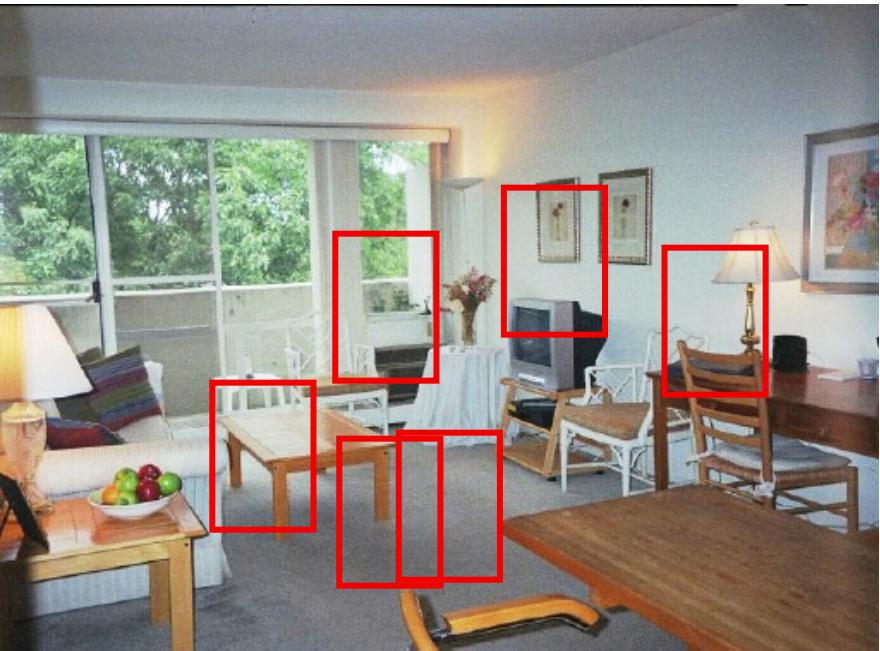
Output of normalized correlation



Object recognition

Is it really so hard?

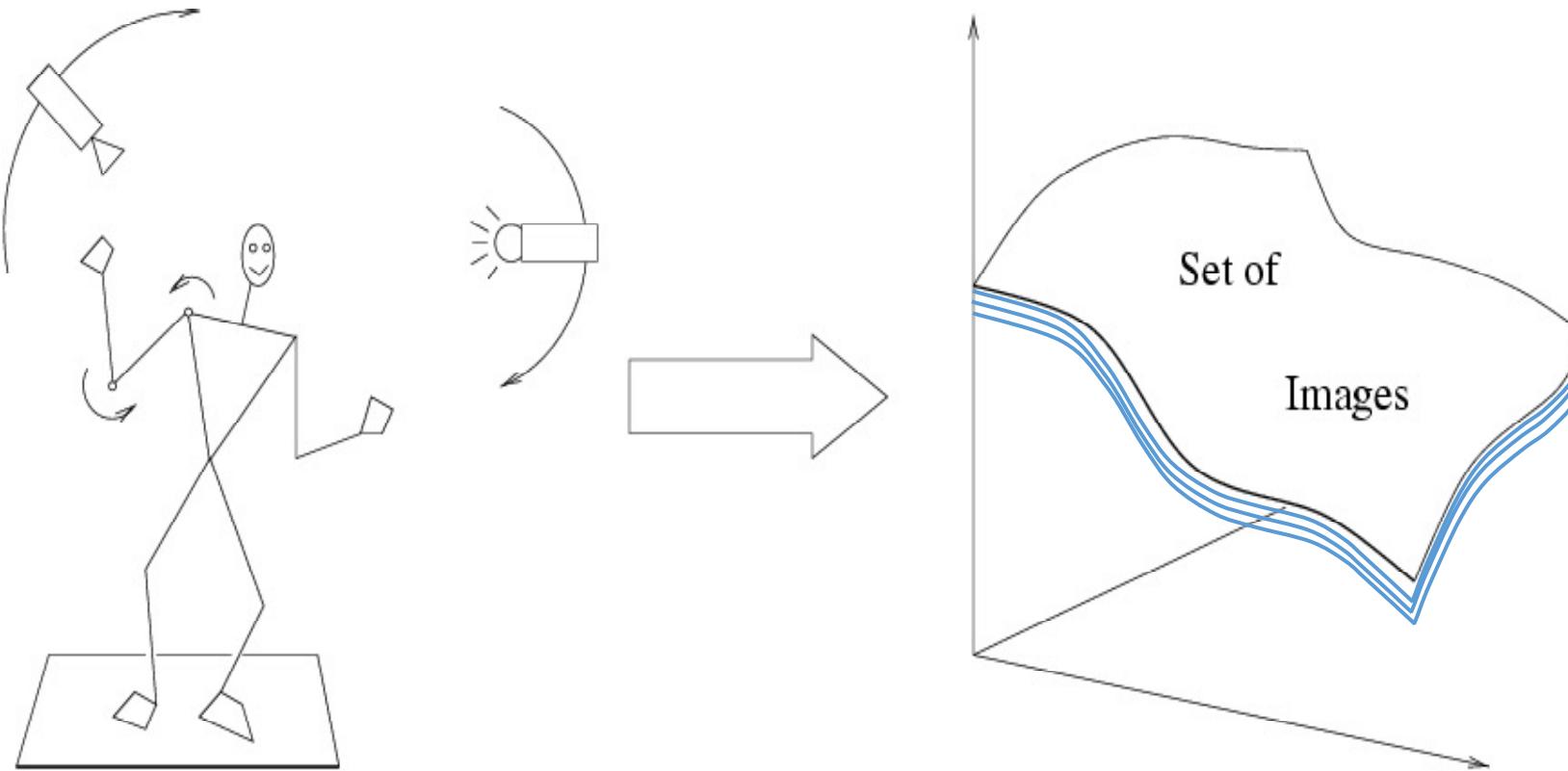
Find the chair in this image



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.

Why is this hard?



Variability:

- Camera position
- Illumination
- Shape parameters



How many object categories are there?

~10,000 to 30,000



Challenge: variable viewpoint



Michelangelo 1475-1564

Challenge: variable illumination

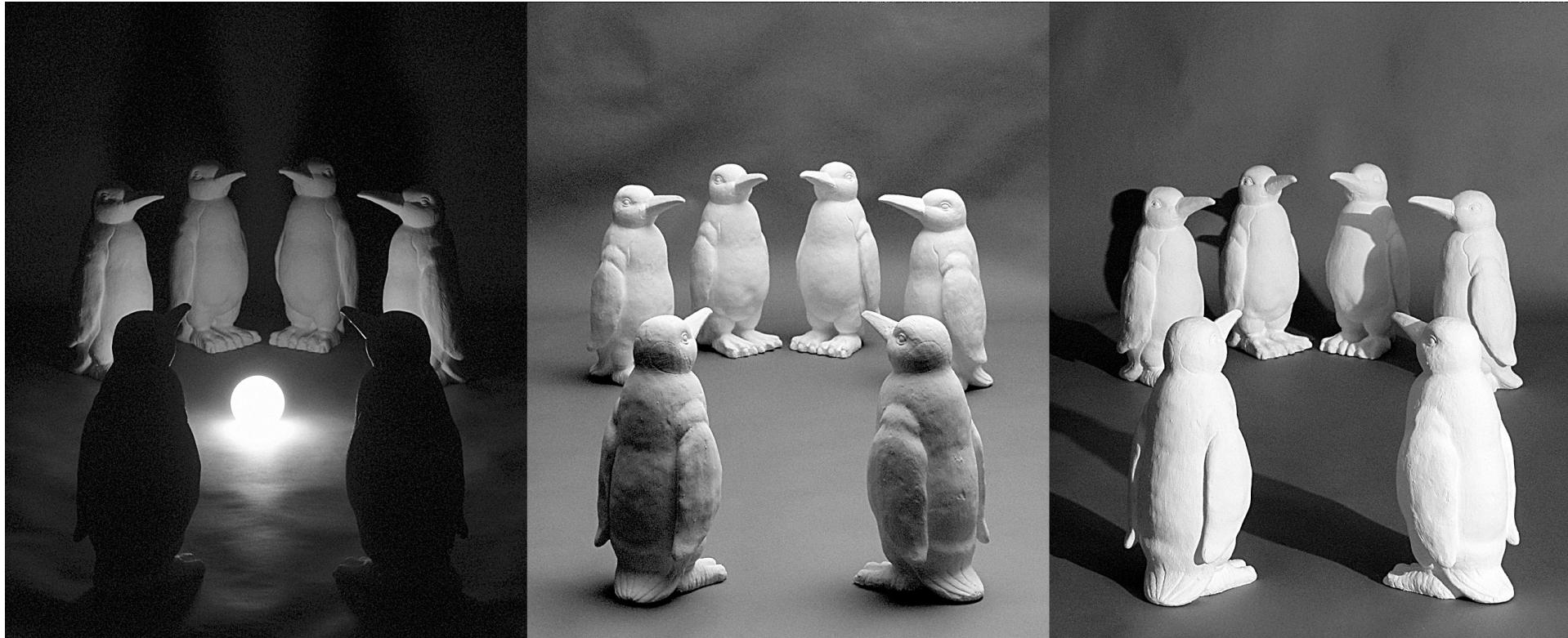


image credit: J. Koenderink

and small things

from Apple.

(Actual size)



Challenge: scale

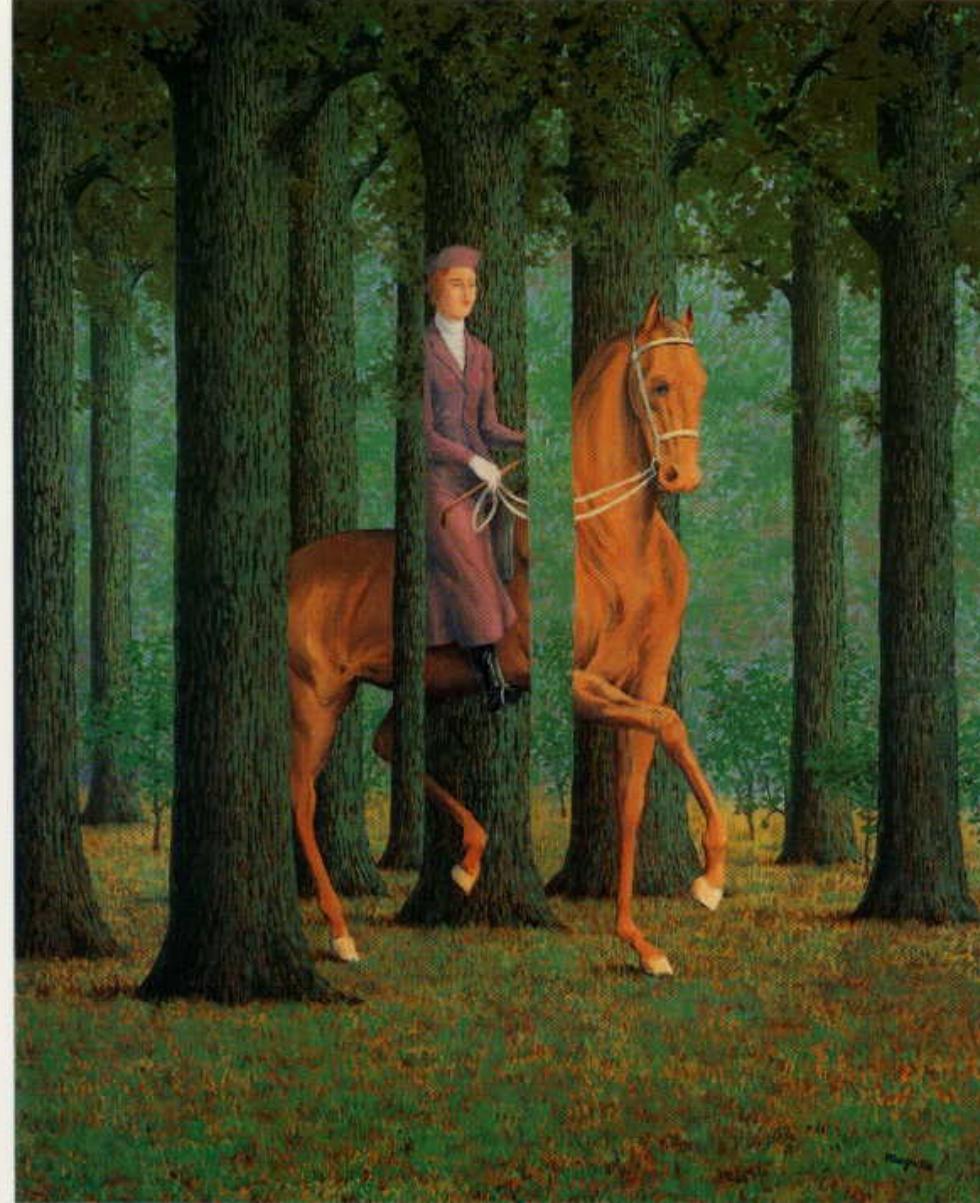
Challenge: deformation





Deformation

Challenge: Occlusion



Magritte, 1957

Challenge: background clutter



Challenge: intra-class variations

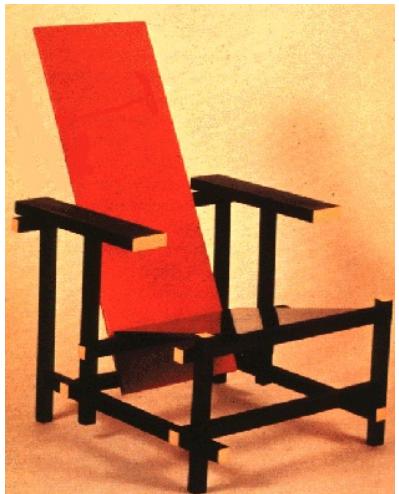


Image Classification

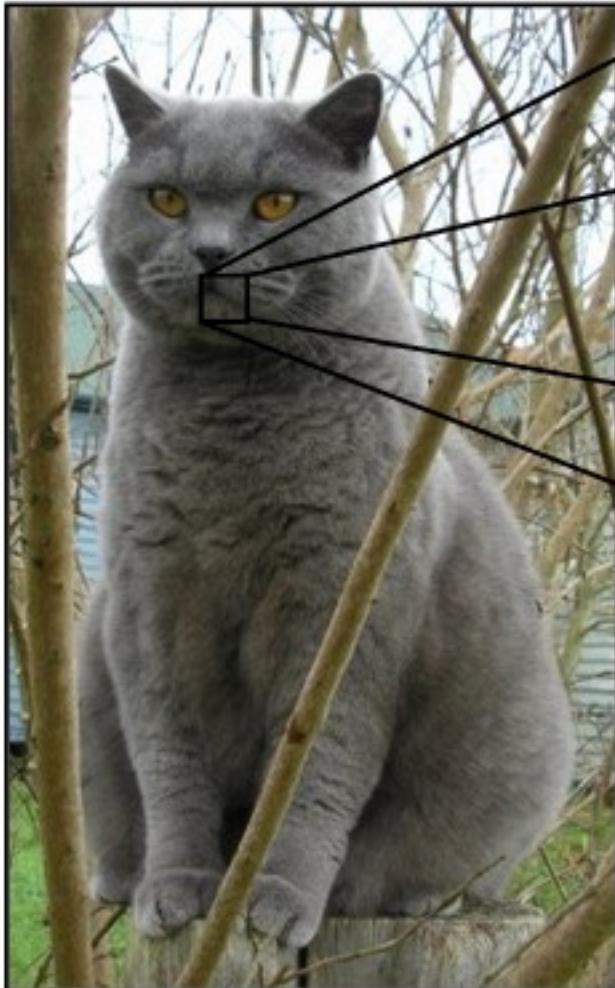


(assume given set of discrete labels)
{dog, cat, truck, plane, ...}



cat

Image Classification: Problem



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

What the computer sees

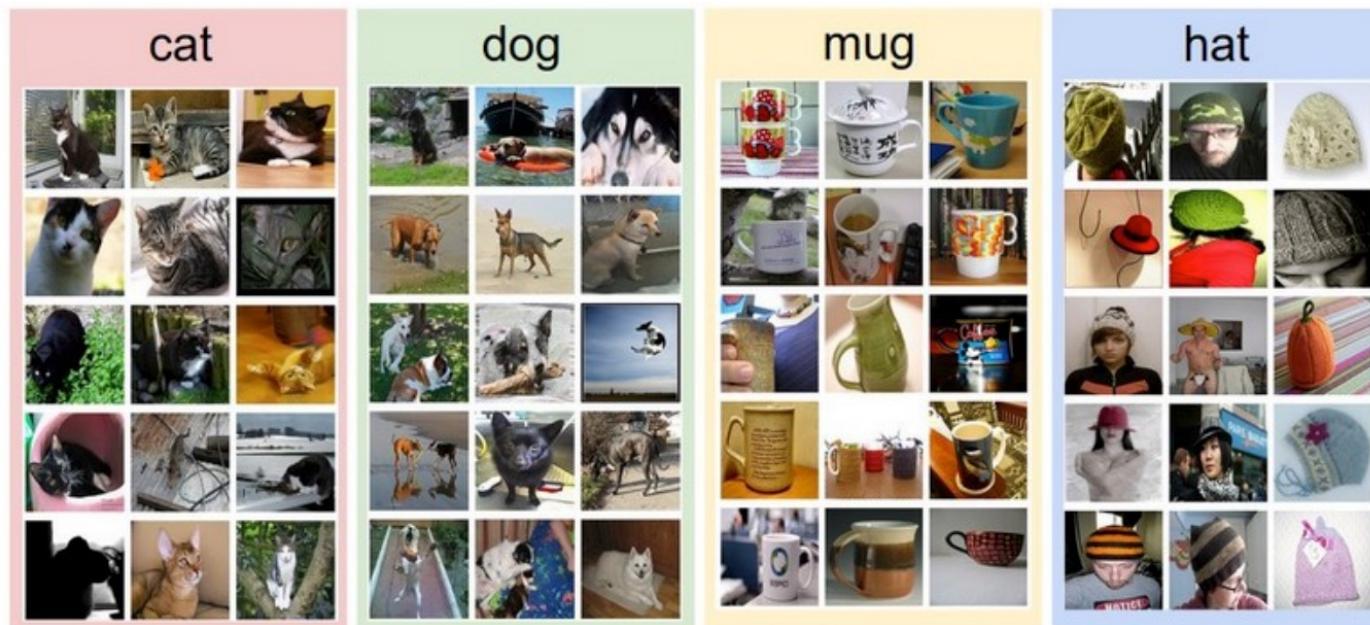
image classification

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



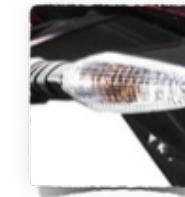
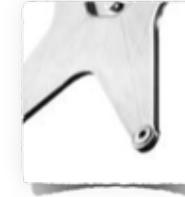
History of Image Classification

- 1960s –early 1990s: the geometric era
 - Recognition as an alignment problem: the simple “toy block” world ...
- 1990s: appearance-based models
 - PCA (eigenface), color histogram ...
- Mid-1990s: sliding window/template approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- **Mid-2000s: bags of features (Today)**
- *Present trends: deep learning (we will get there)*



Bag of words

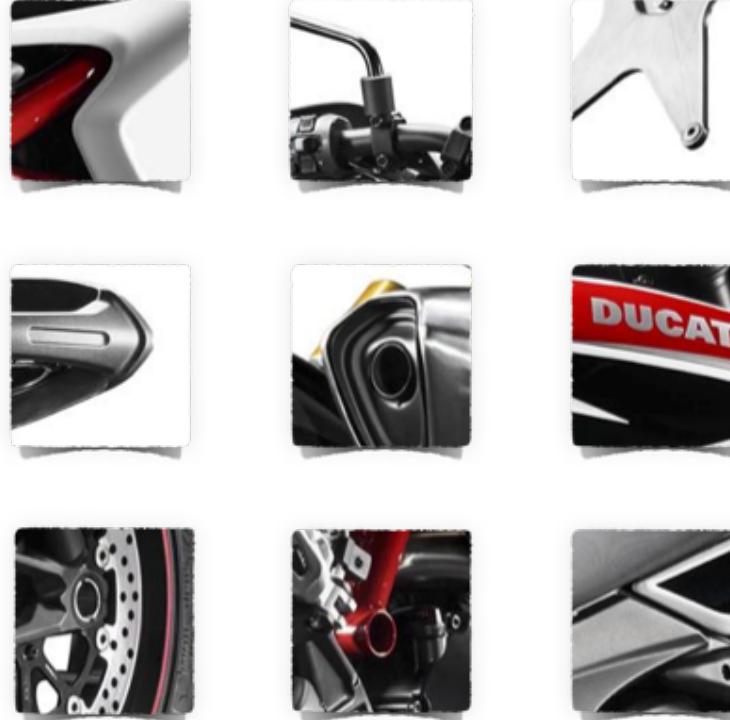
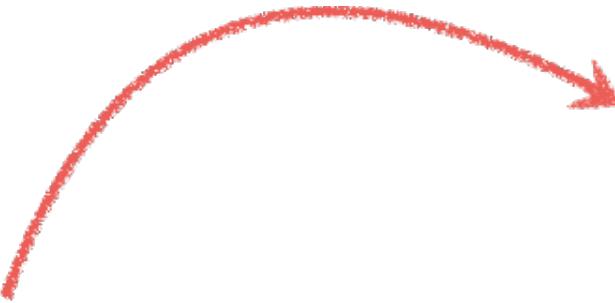
What object do these parts belong to?



Some local feature are
very informative



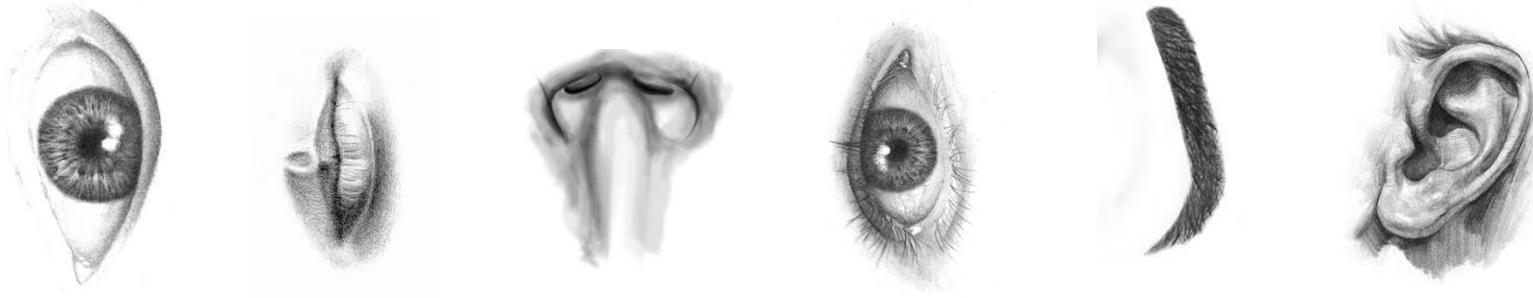
An object as



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

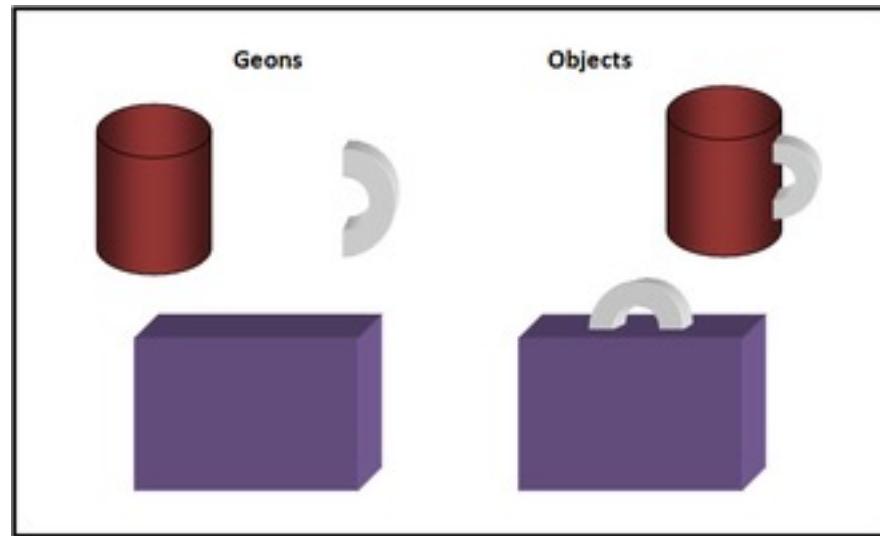
- deals well with occlusion
- scale invariant
- rotation invariant

(not so) crazy assumption

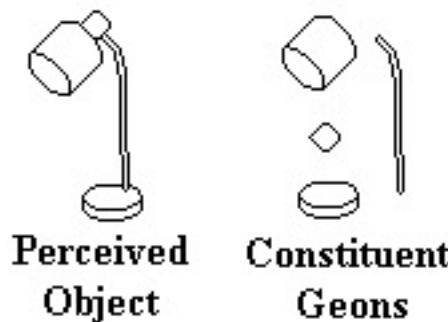


spatial information of local features
can be ignored for object recognition (i.e., verification)

Recognition-by-Components (RBC) Theory (1987)



- A human learning theory to explain object recognition
- According to RBC theory, we are able to recognize objects by separating them into **geons** (the object's main component parts).
- Geons are based on basic 3-dimensional shapes (cylinders, cones, etc.) that can be assembled in various arrangements to form a virtually unlimited number of objects.
- **Very impactful for computer vision recognition!**



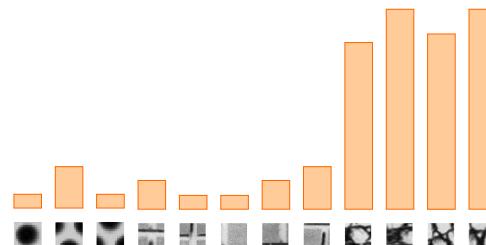
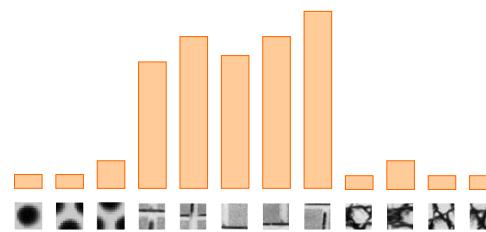
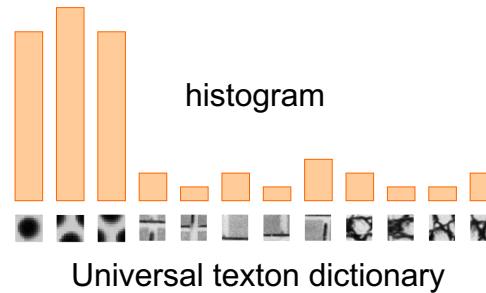
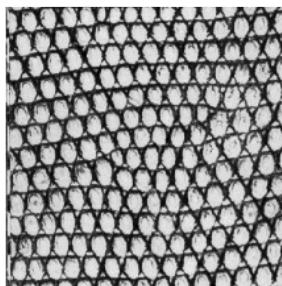
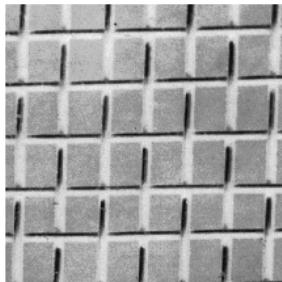
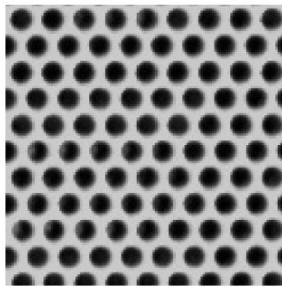
Bag-of-features

represent a data item (document, texture, image)
as a histogram over features

an old idea

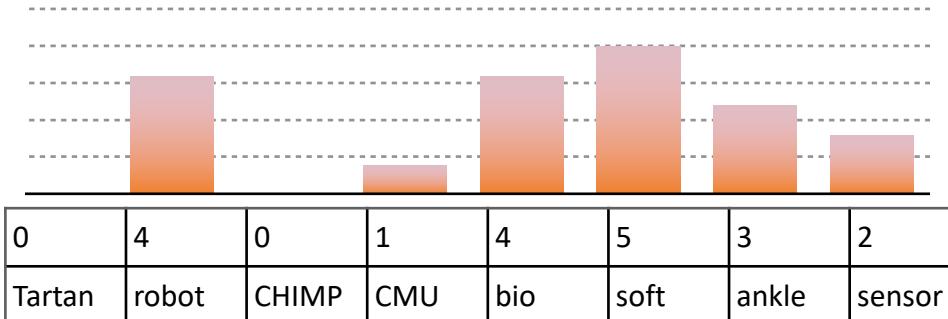
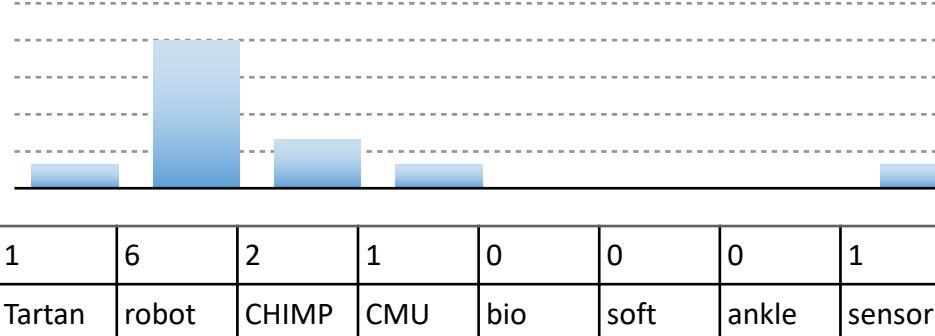
(e.g., texture recognition and information retrieval)

Texture recognition



Vector Space Model

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



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



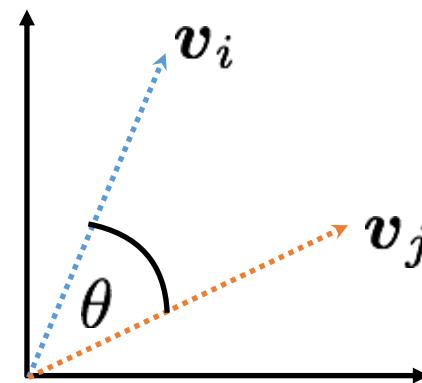
just a histogram over words

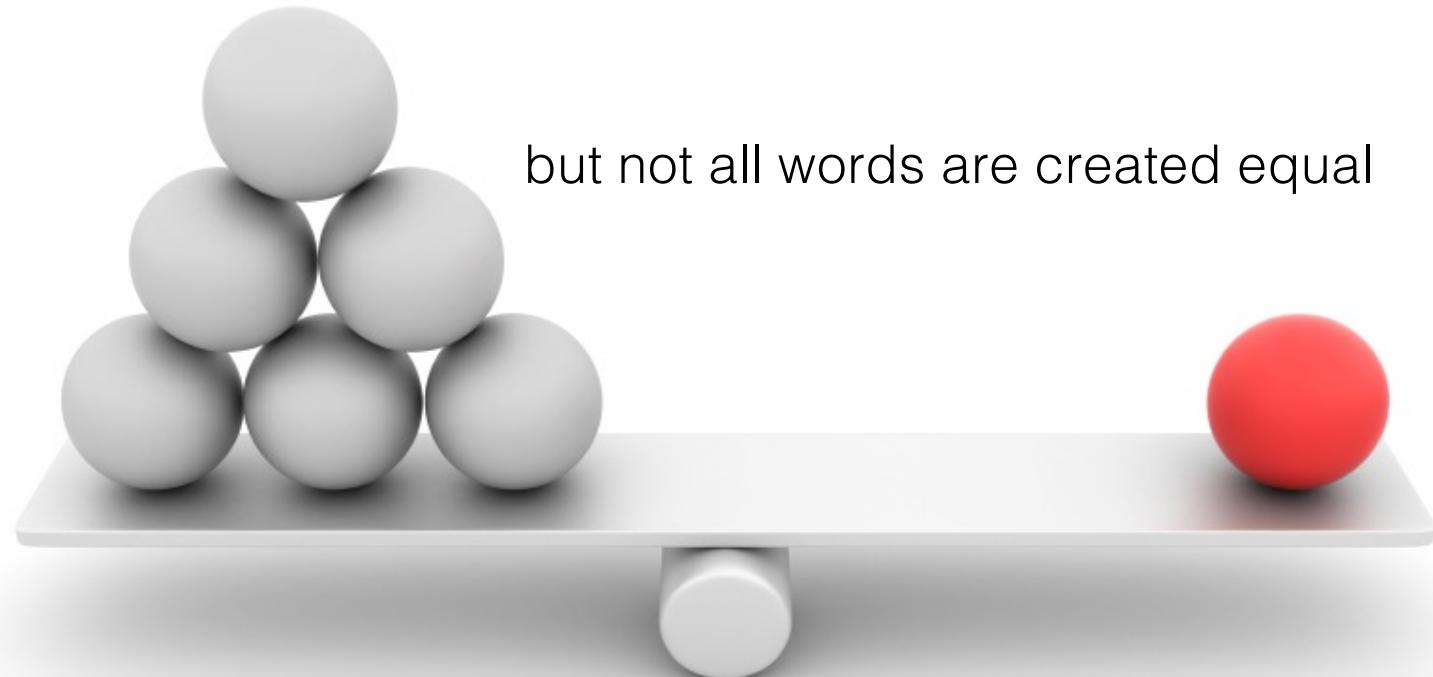
What is the similarity between two documents?



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

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





but not all words are created equal

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

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

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

(down-weights **common** terms)

Standard BOW pipeline (for image classification)

Dictionary Learning:
Learn Visual Words using clustering

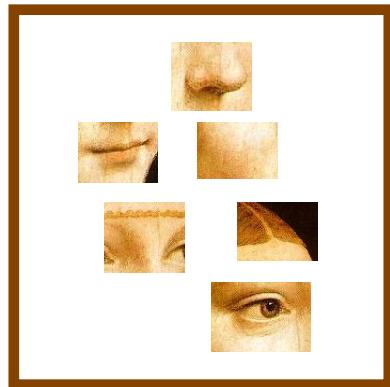
Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Dictionary Learning:

Learn Visual Words using clustering

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



Dictionary Learning:

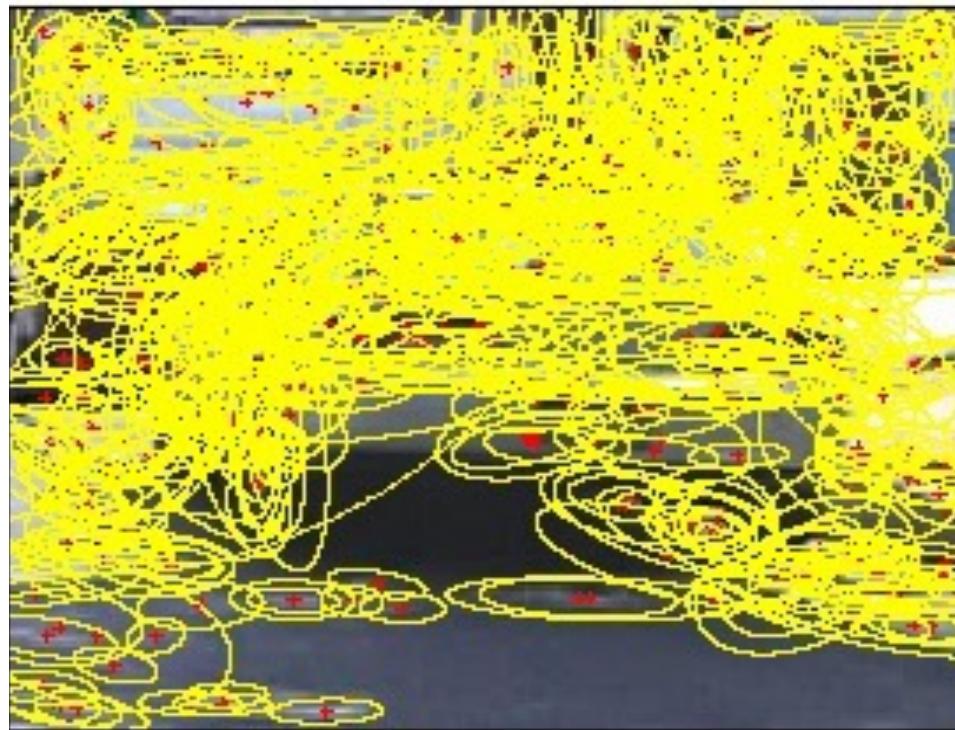
Learn Visual Words using clustering

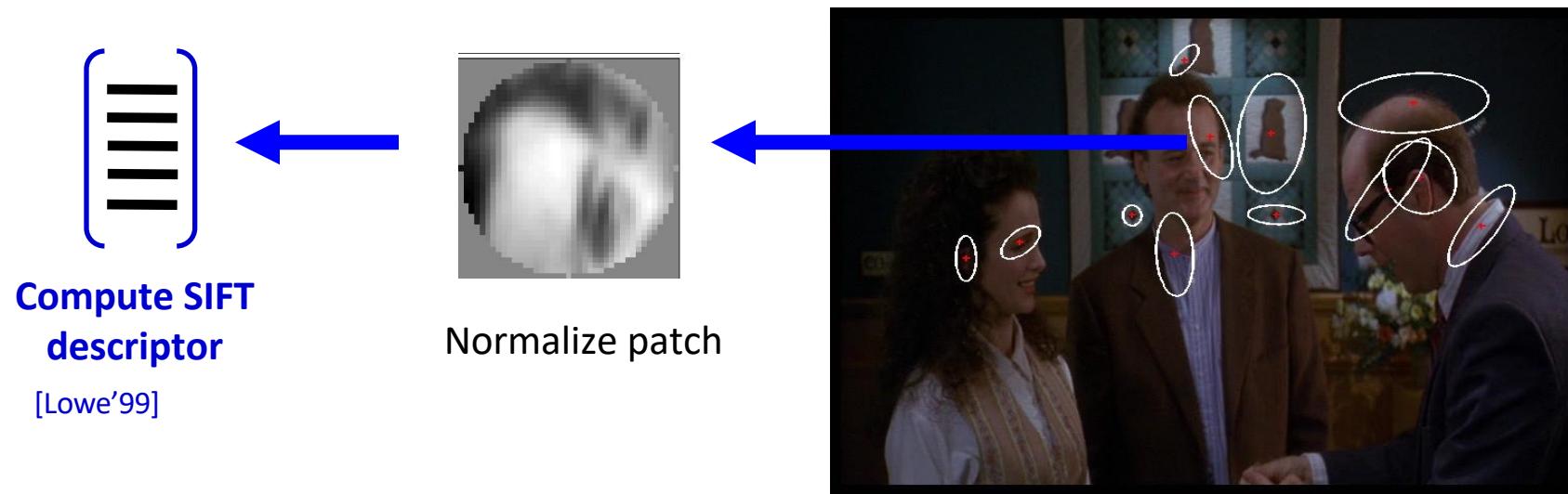
2. Learn visual dictionary (e.g., K-means clustering)



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)

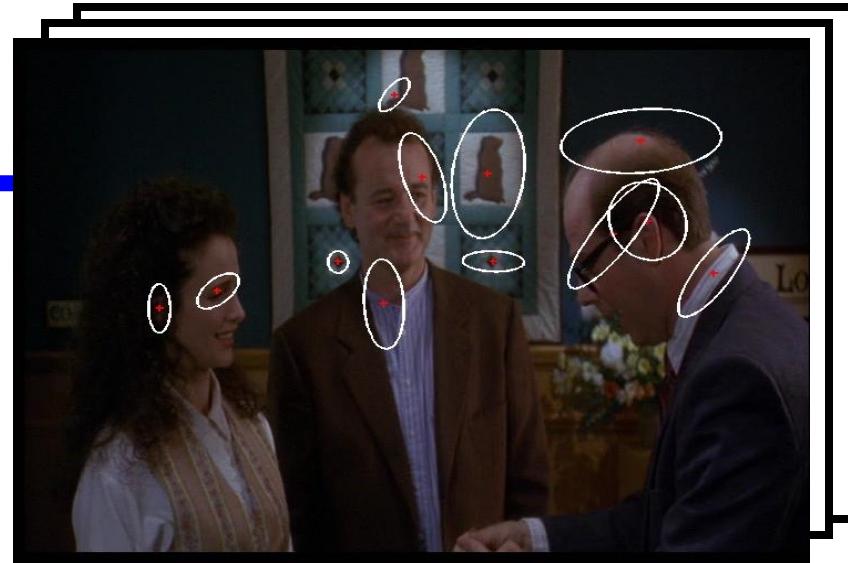
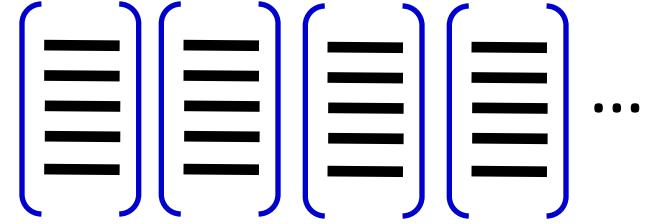




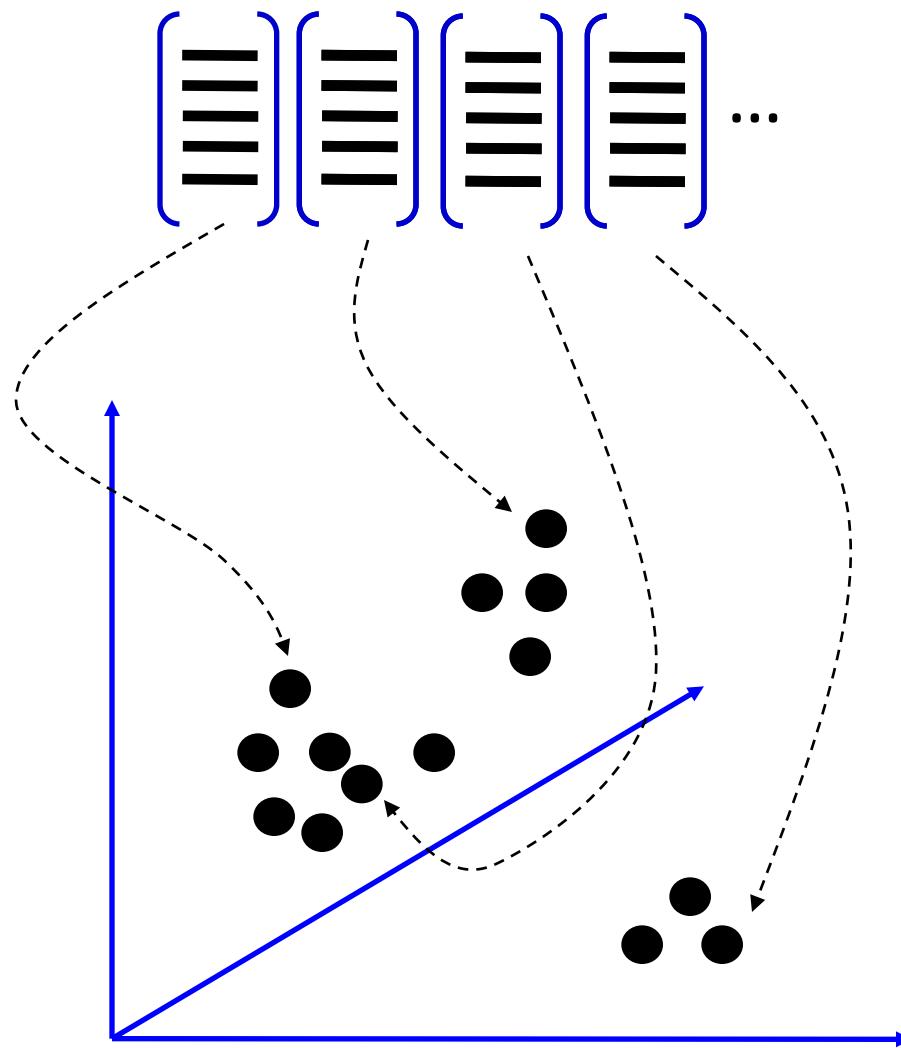
[Mikojaczyk and Schmid '02]

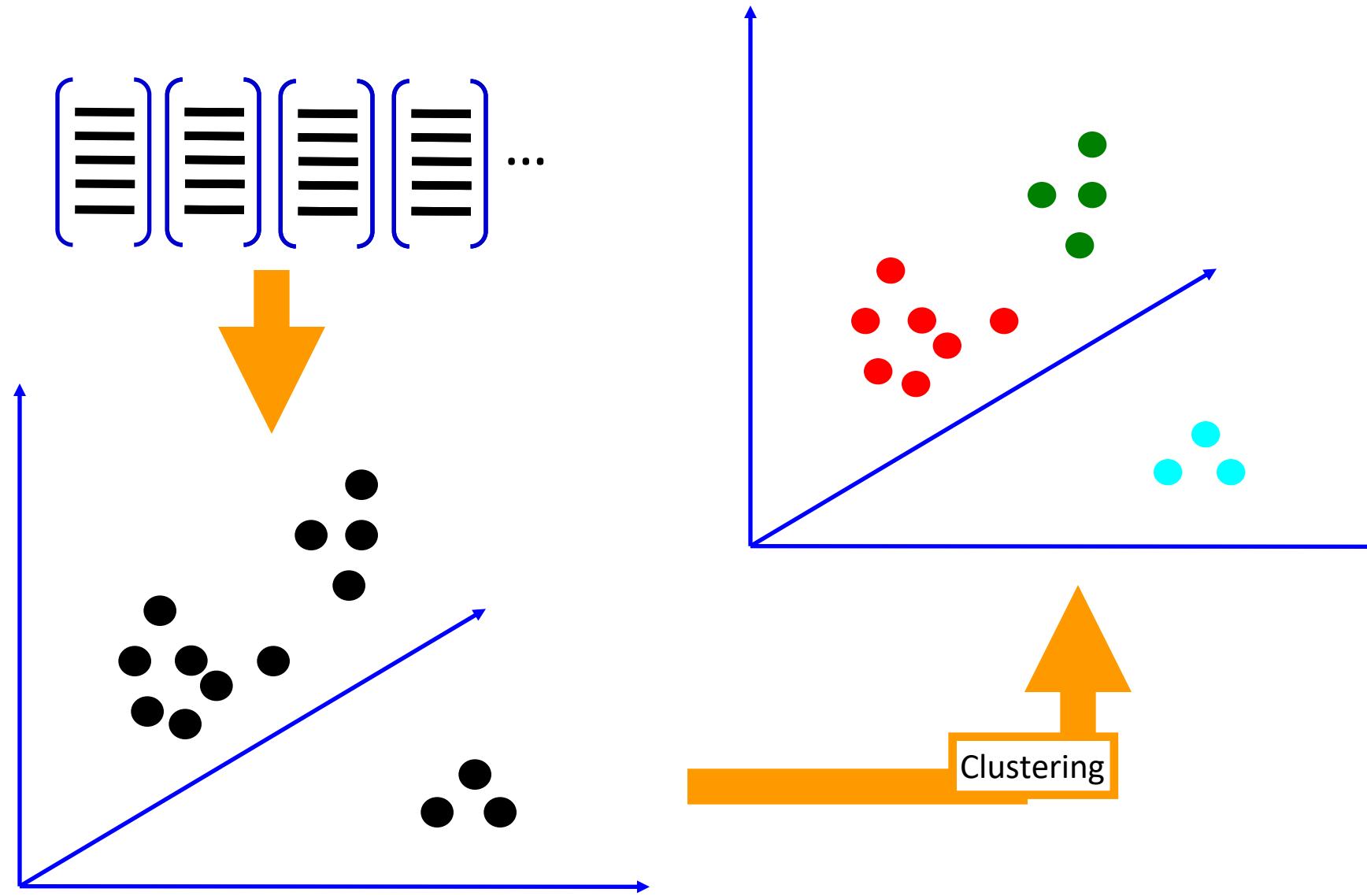
[Mata, Chum, Urban & Pajdla, '02]

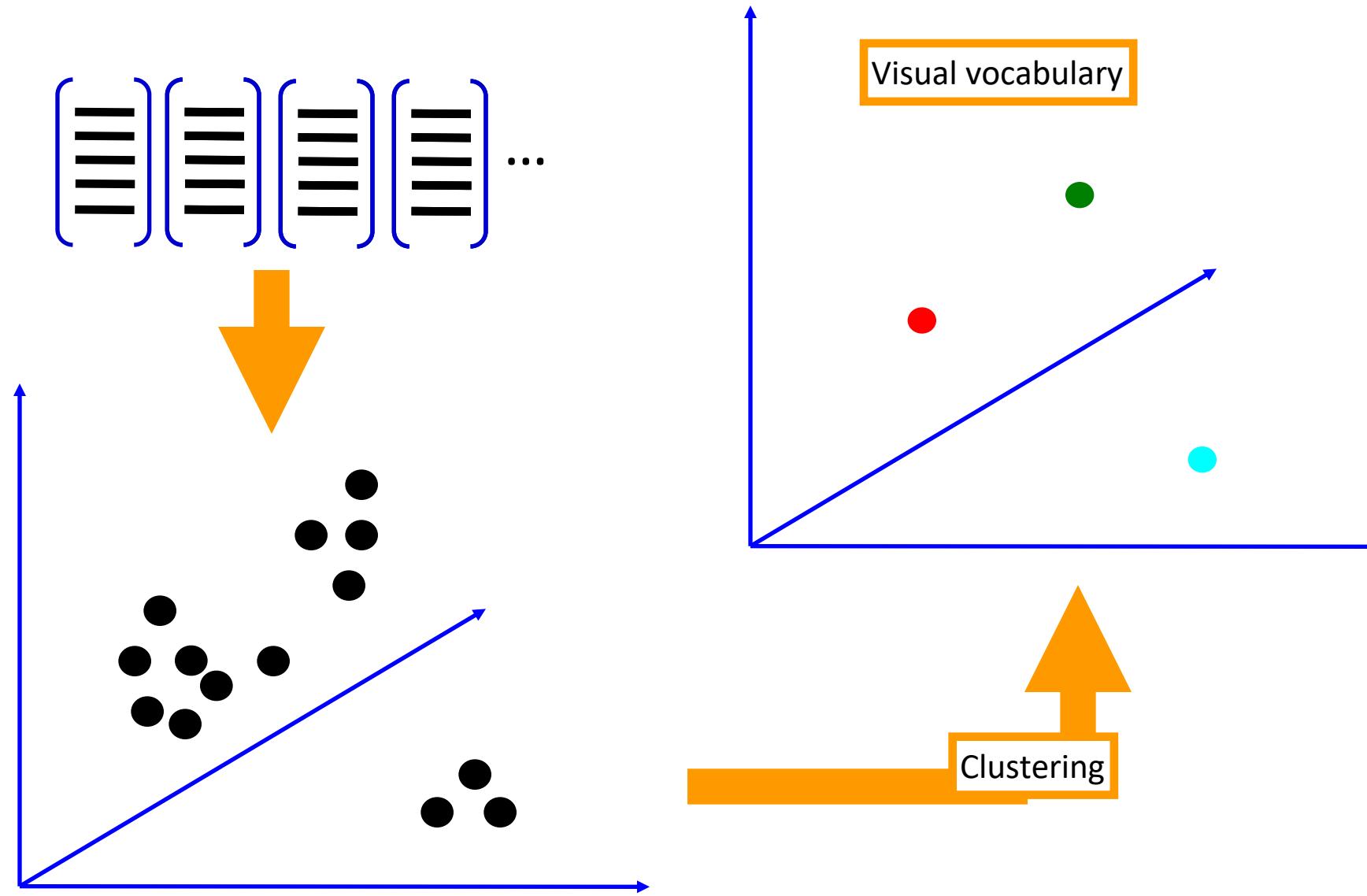
[Sivic & Zisserman, '03]



How do we learn the dictionary?



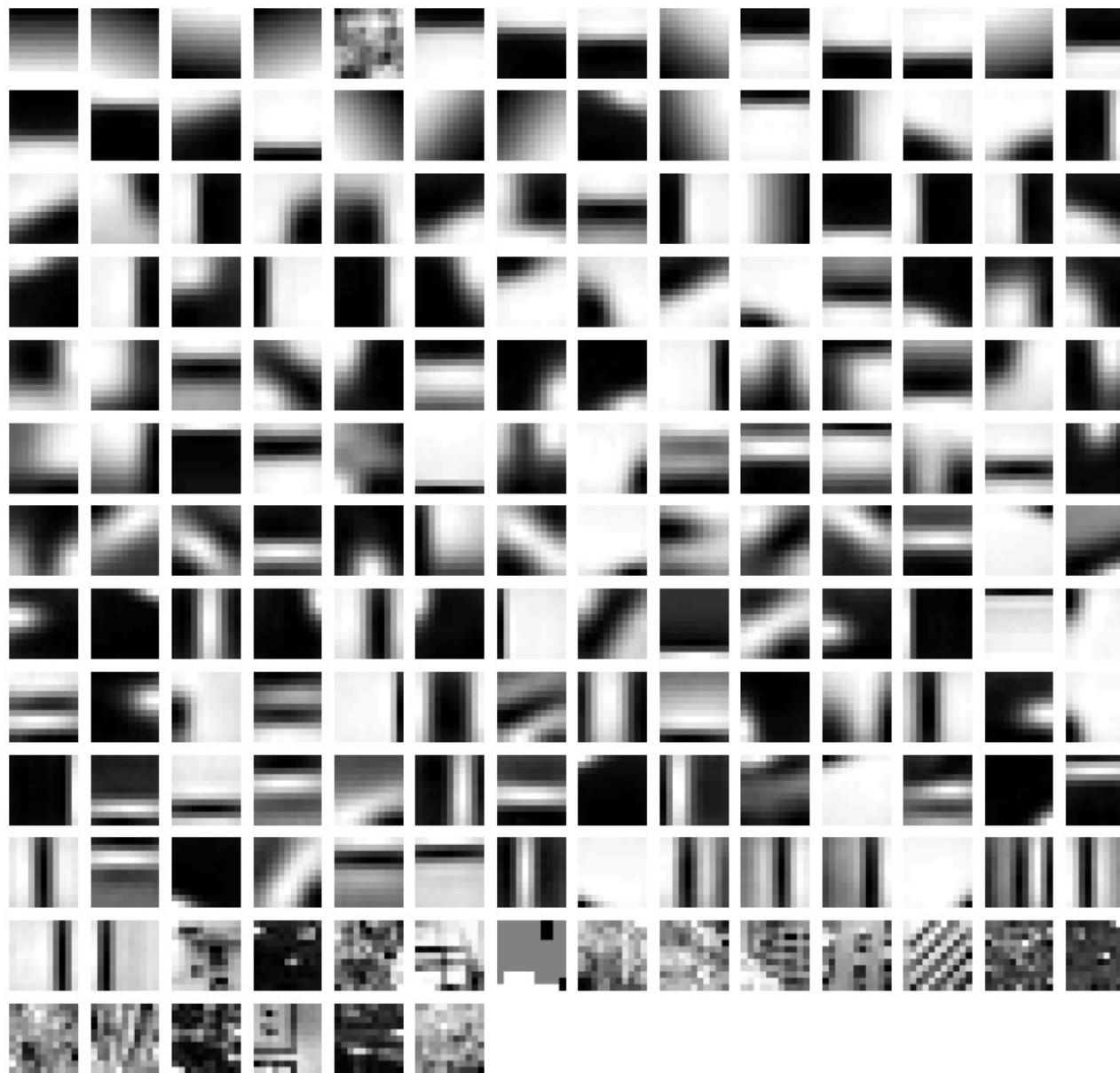




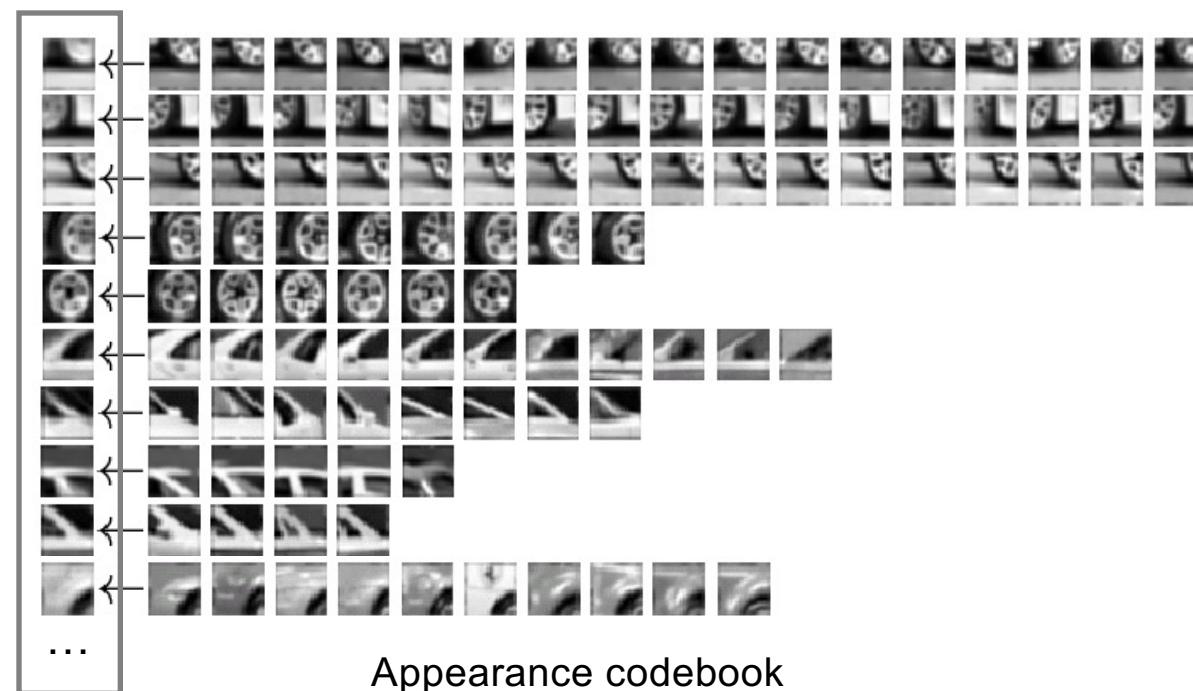
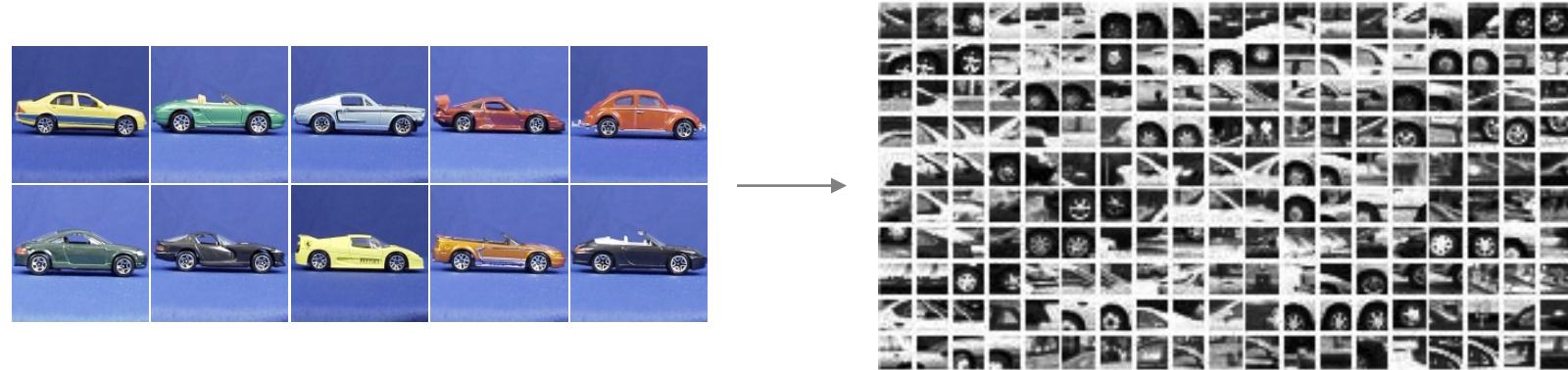
*From what **data** should I learn the dictionary?*

- Dictionary can be learned on separate training set
- Provided the training set is sufficiently representative, the dictionary will be “universal”

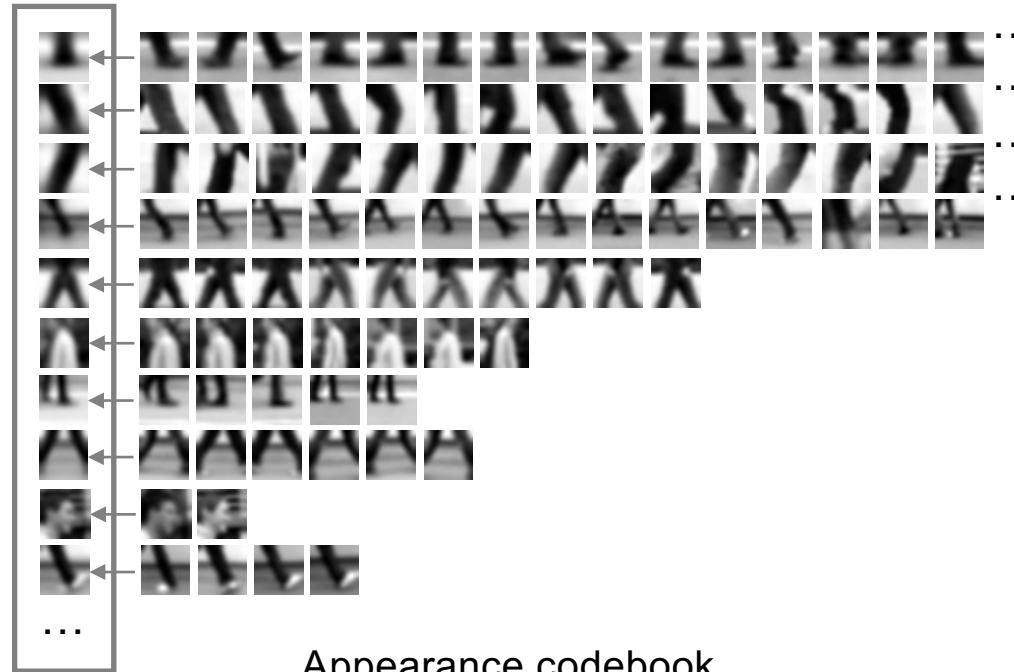
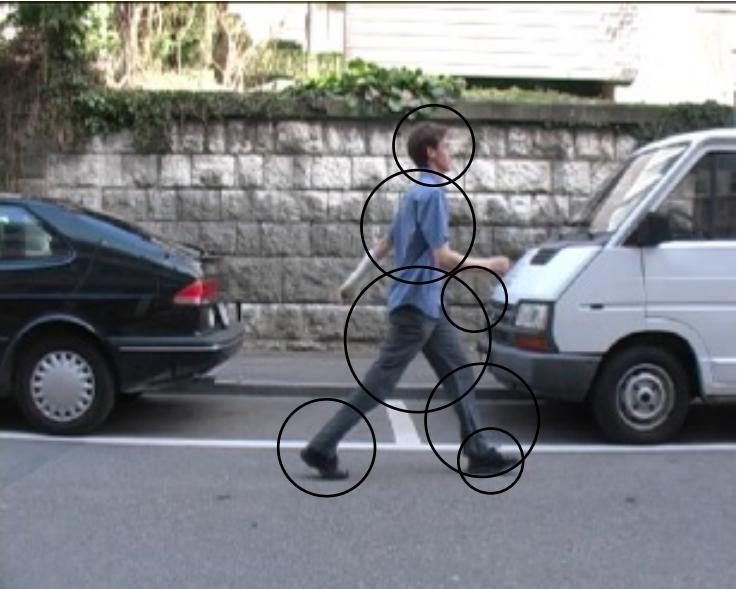
Example visual dictionary



Example dictionary



Another dictionary

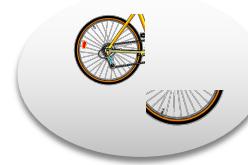
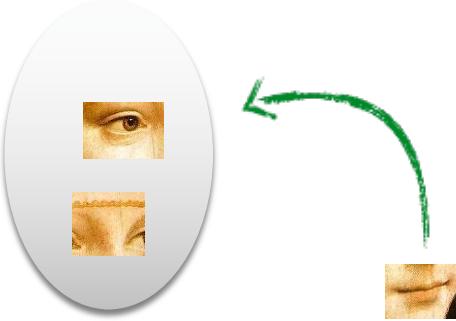


Dictionary Learning:

Learn Visual Words using clustering

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

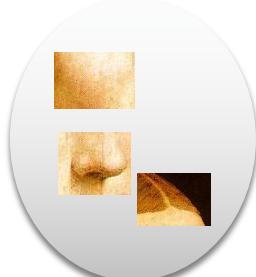
Classify:
Train and test data using BOWs



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

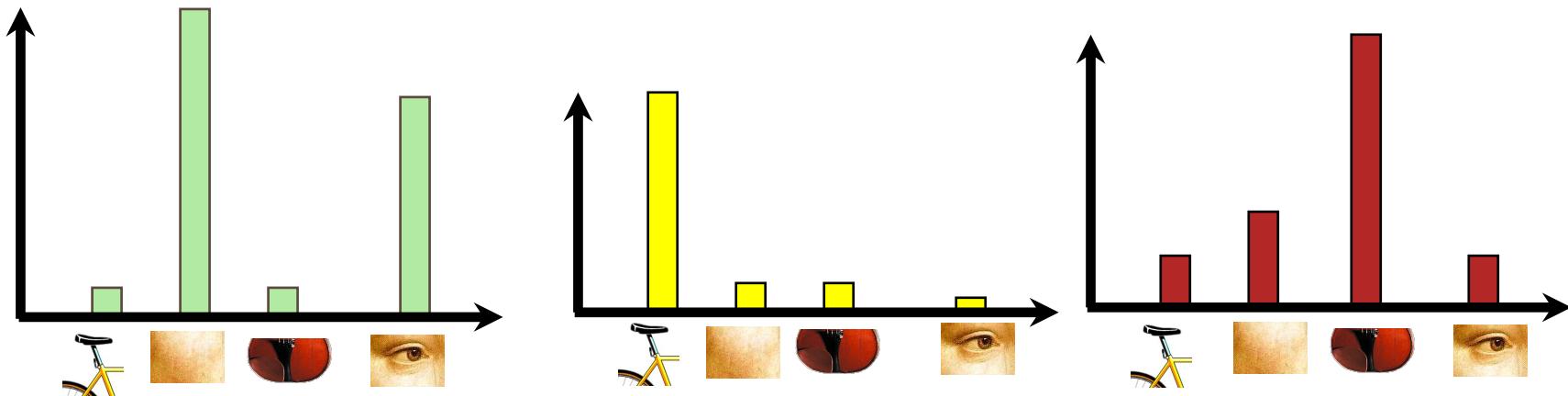
Encode:

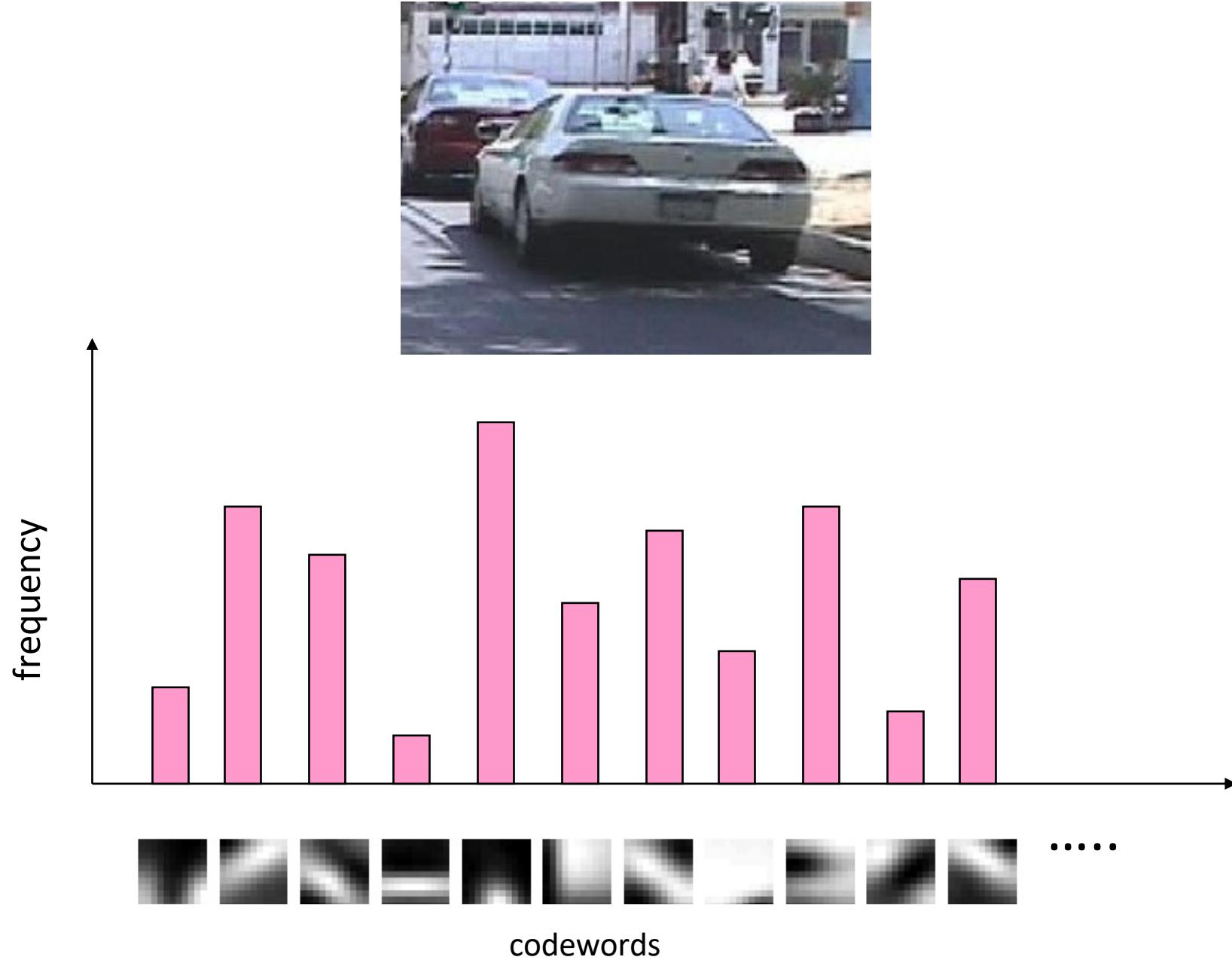
build Bags-of-Words (BOW) vectors
for each image



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

2. Histogram: count the
number of visual word
occurrences

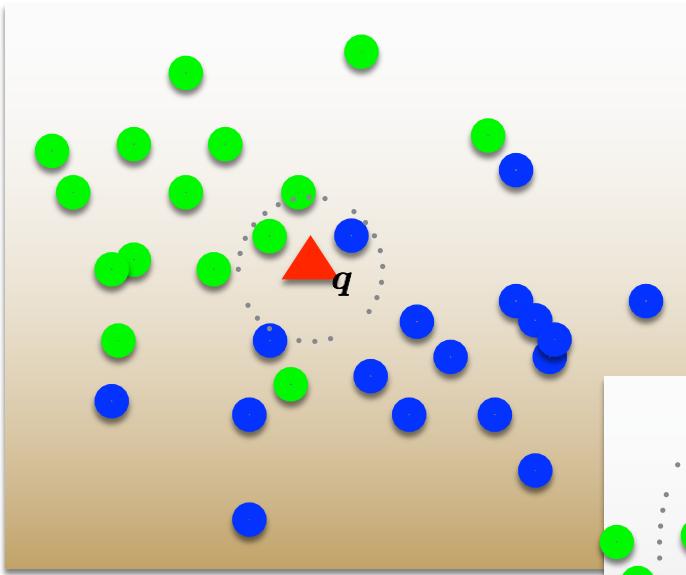




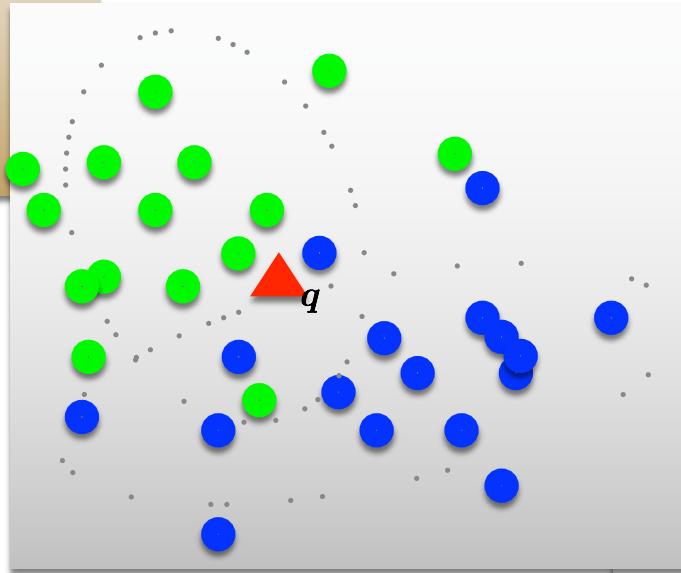
Dictionary Learning:
Learn Visual Words using clustering

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

Classify:
Train and test data using BOWs

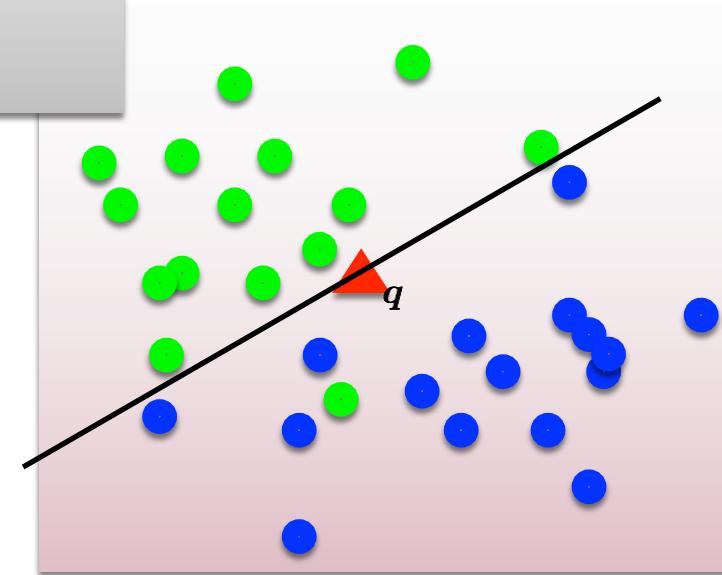


K nearest neighbors



Naïve Bayes

Support Vector Machine





The University of Texas at Austin
**Electrical and Computer
Engineering**
Cockrell School of Engineering