Recognition

Encodings, pooling strategies, etc.

Reference(s): see slides

Today's lecture

Overview

- **Brief motivation**
- **Bag-of-Features (BoF) / Bag-of-Visual-Words (BoW)**
- **Spatial pyramid encoding**
- **"Semantic" encodings & Attributes**
- **Fisher vector encoding**

Introduction to recognition

Recognition tasks

Recognition Tasks …

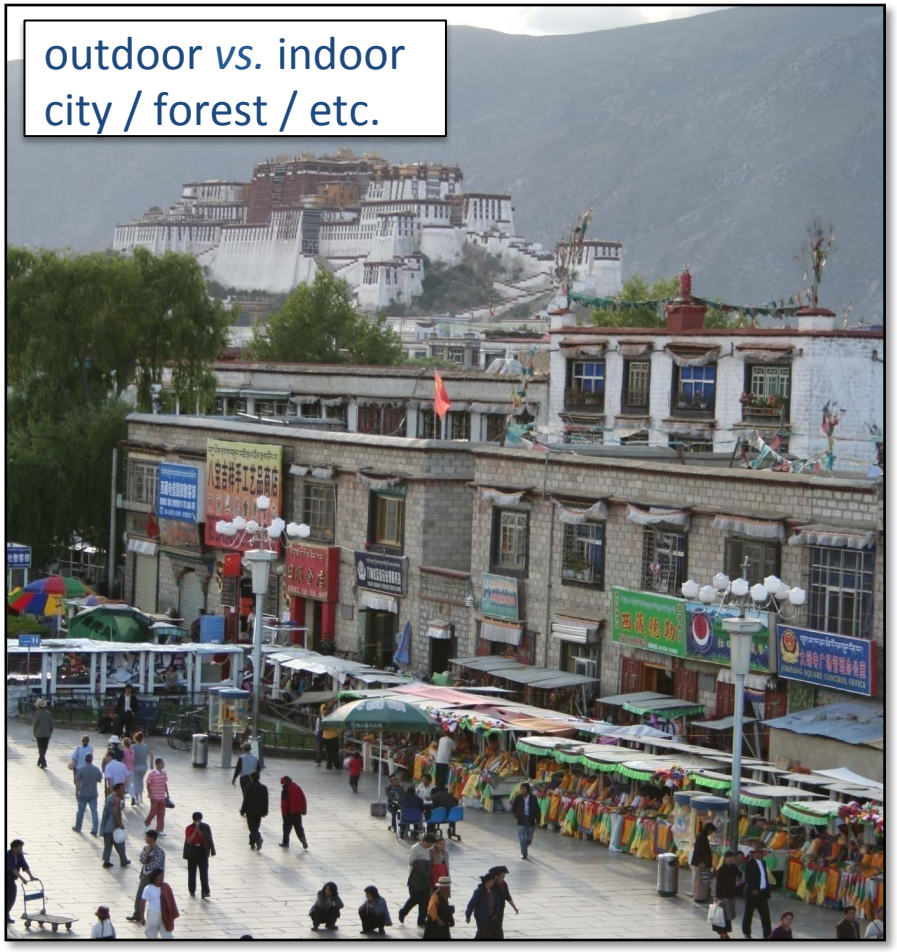
Slide adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

Introduction to recognition

Recognition tasks

Recognition Tasks …

- **Scene categorization**



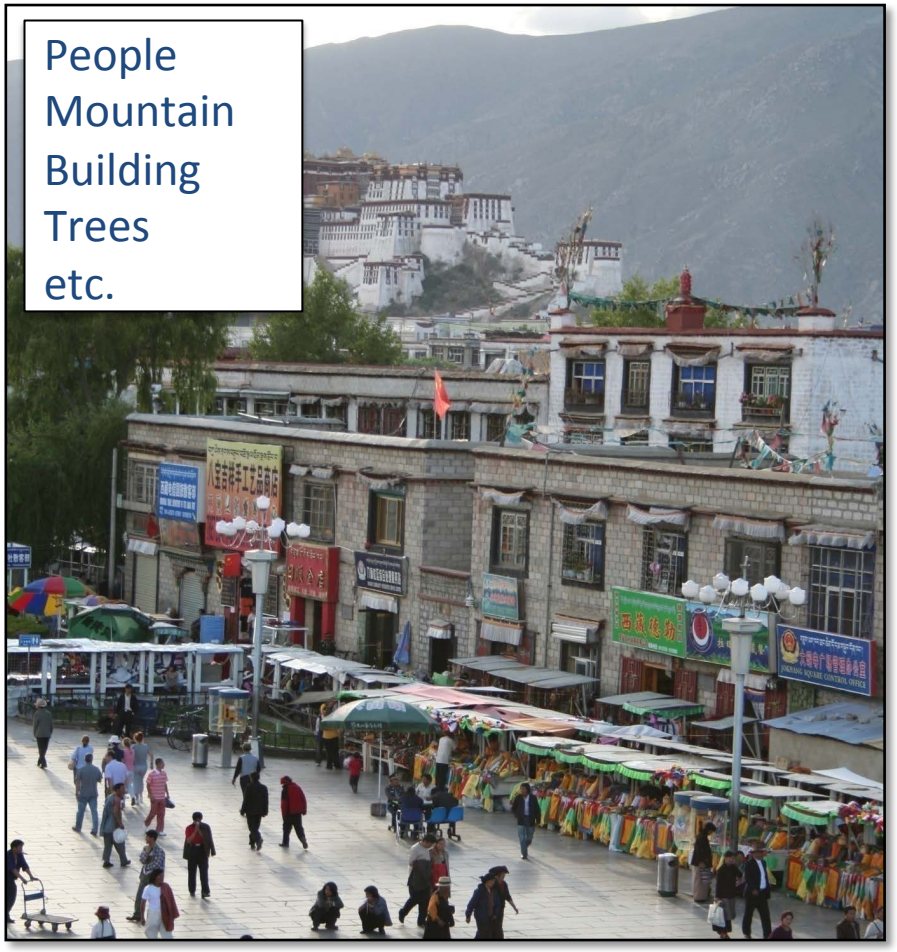
Slide adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

Introduction to recognition

Recognition tasks

Recognition Tasks …

- Scene categorization
- **Image Tagging**



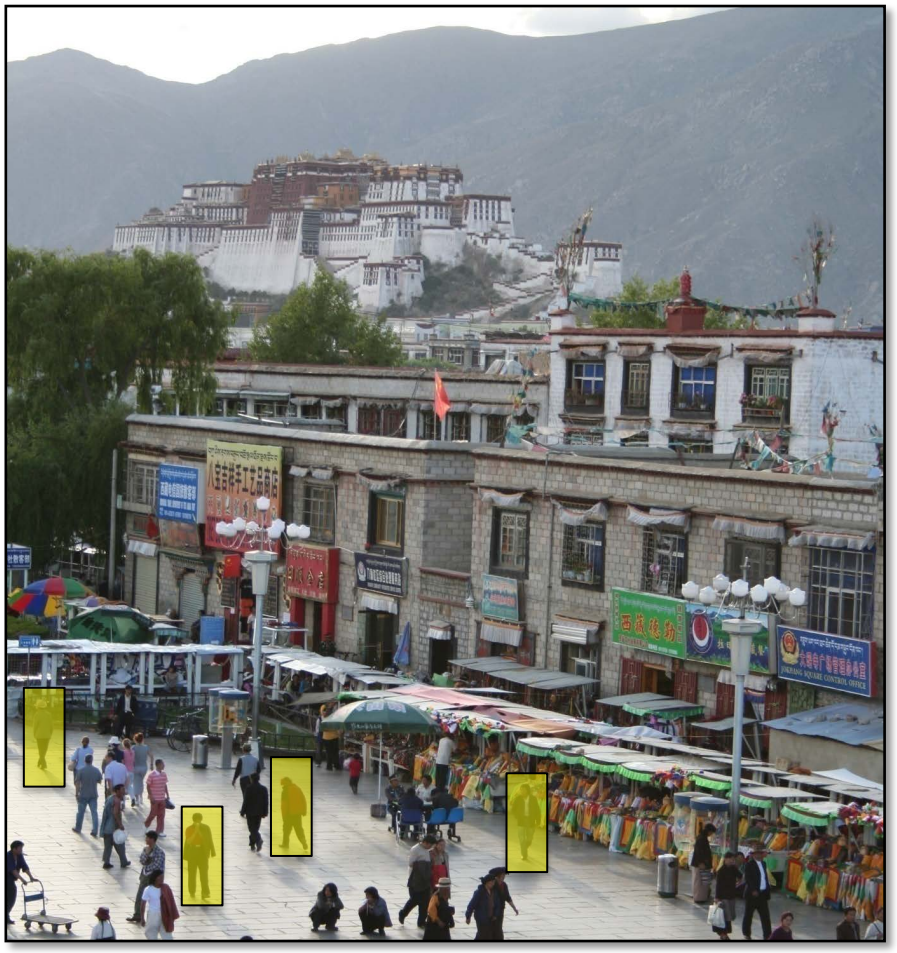
Slide adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

Introduction to recognition

Recognition tasks

Recognition Tasks …

- Scene categorization
- Image Tagging
- **Object Detection** (*e.g.,* people)



Slide adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

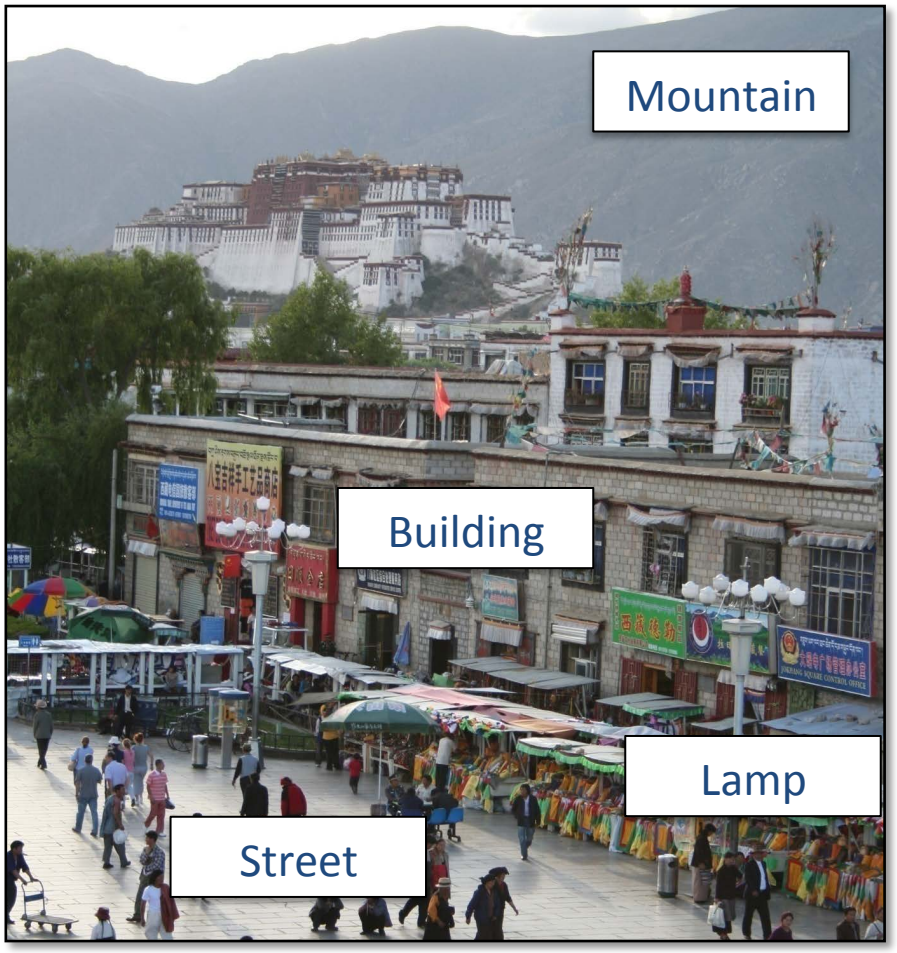
Introduction to recognition

Recognition tasks

Recognition Tasks …

- Scene categorization
- Image Tagging
- Object Detection (e.g., people)
- **Image "parsing"**

What about image understanding ?



Slides adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

Introduction to recognition

The magnitude of the problem …

How many visual categories do we distinguish?

Courtesy of Aude Oliva

approx. 30000 [Biederman, 1987] (based on an estimate of about 3000 entry-level categories)

Slides adapted from S. Lazebnik, Fei-Fei Li, R. Fergus, A. Torralba, J. Ponce

Biederman, I., Recognition-by-Components: A Theory of Human Image Understanding, Psychol. Rev. 94(2), pp. 115-146, 1987

Introduction to recognition

A very very brief history …

1960s – early 1990: geometric era (*e.g.*, [Roberts, 1963])

1990s: appearance-based techniques

(*e.g.*, [Turk & Pentland, 1991], [Swain & Ballard, 1991], etc.)

1990s: sliding-window approaches

(*e.g.*, [Viola & Jones, 2000], [Dalal & Triggs, 2005], etc.)

late 1990s, early 2000s: local features, bag-of-features

(*e.g.*, [Lowe, 2004], [Fei-Fei & Perona, 2005])

Early 2000s: parts and shape models (*e.g.*, [Felzenszwalb et al., 2009])

Current trends: context, semantics, attributes, deep learning, etc. (too many :)

Slide adapted from S. Lazebnik

Roberts, L.G., "Machine Perception of Three Dimensional Solids", PhD thesis, MIT (Department of Electrical Engineering), 1963
Turk, M. A. and A.P. Pentland, "Face recognition using eigenfaces", CVPR, 1991
Swain, M. and D.H. Ballard, "Color Indexing", ICJV 7(1), pp. 11-32, 1991
Viola, P. and M. J. Jones, "Robust Real-Time Face Detection", IJCV 57(2), pp. 137-154, 2004
Dalal, N. and Triggs, B., "Histograms of Oriented Gradients for Human Detection", CVPR, 2005
Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60(2), pp. 91-110, 2004.
Felzenszwalb P., R. Girshick, D. McAllester,D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models", PAMI, 32(9), 2010
Fei-Fei Li and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories", CVPR, 2005

Introduction to recognition

Motivating example …

Training data

$f($  $) =$ "apple"

$f($  $) =$ "tomato"

$f($  $) =$ "cow"

Prediction function

Input Image

Output

How can we "encode" images?

Today's lecture

Overview

- Introduction
- **Bag-of-Features (BOF) / Bag-of-Visual-Words (BOV)**
- Spatial pyramid encoding
- Fisher vector encoding
- "Semantic" encodings

Bag-of-Features / Bag-of-Words

General concept

Bag-of-Features

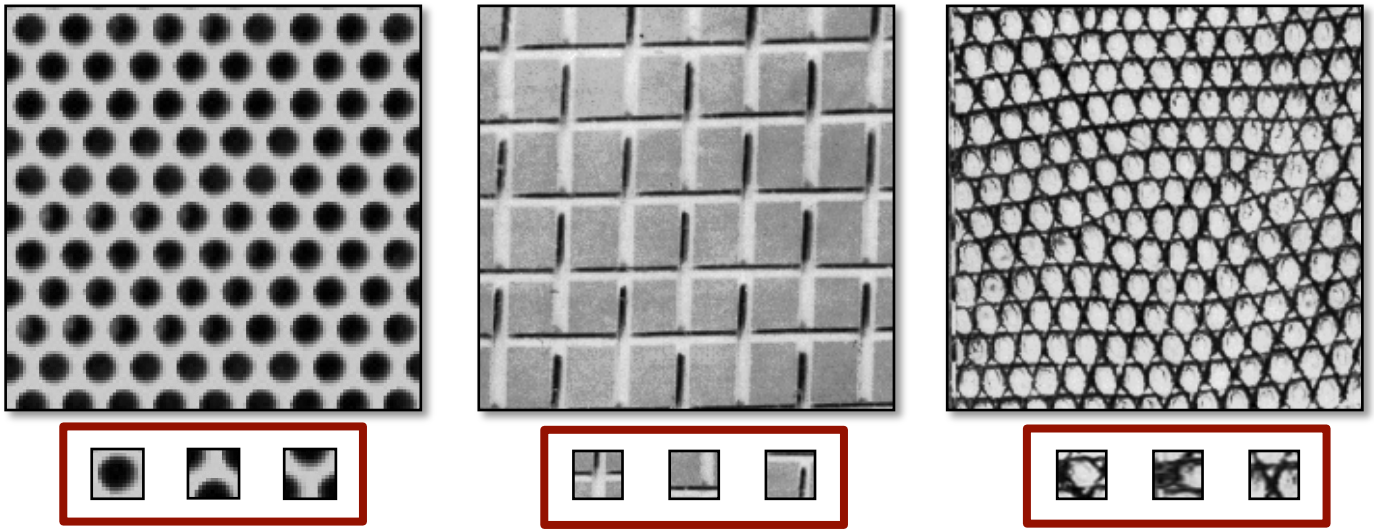
Csurka, G., Dance, C., Fan, L., Willamowski, J. and Bray, C., “Visual Categorization with Bags of Keypoints”, ECCV, 2004
Fei-Fei Li and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories”, CVPR, 2005

Bag-of-Features / Bag-of-Words

Origins … or "Where does this idea come from?"

Texture recognition

(*e.g.*, [Leung & Malik, 2001], [Varma & Zisserman, 2005])

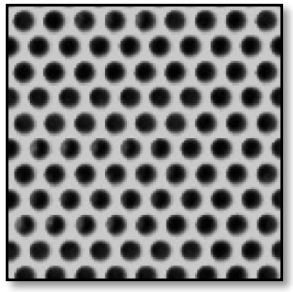


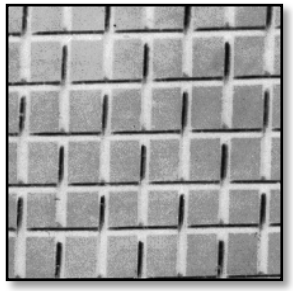
Texture is characterized by the repetition of "basic elements" (**textons**);
for stochastic textures, the "texton identity" matters, **NOT** the spatial arrangement!

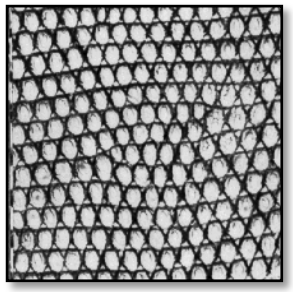
Leung, T. and Malik, J., "Representing and Recognizing the Visual Appearance of Materials using Three-Dimensional Textons", IJCV, 43(1), 2001
Varma, M. and Zisserman, A., "A statistical approach to texture classification from single images", IJCV 62(1), pp. 61-81, 2005

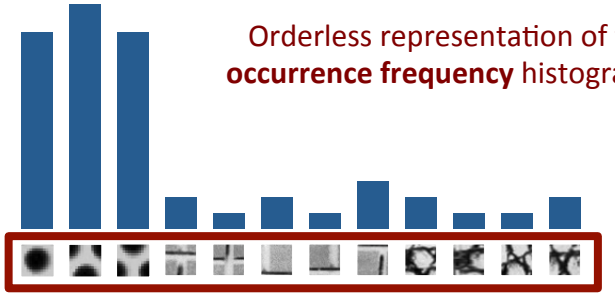
Bag-of-Features / Bag-of-Words

Origins … or “Where does this idea come from?”



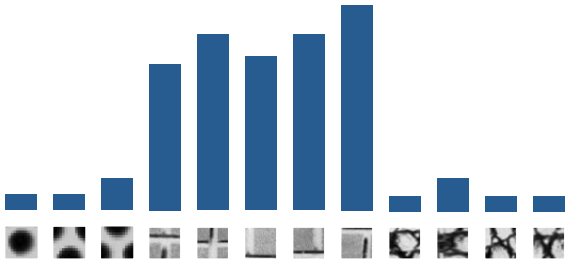


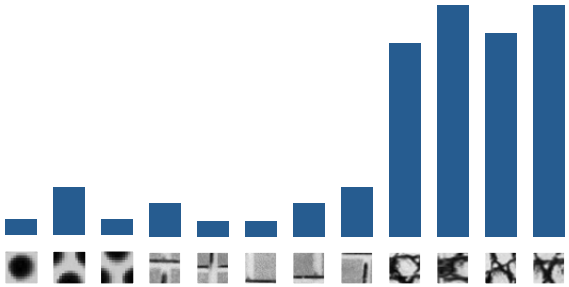




Orderless representation of the image as the **occurrence frequency** histogram of the textons!

universal “texton” dictionary

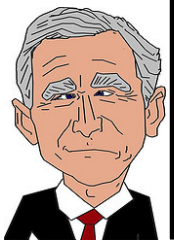




Bag-of-Features / Bag-of-Words

Origins … or "Where does this idea come from?"

The idea of a BoW representations dates back to early works in text document analysis (*cf.* [Salton & McGill, 1983]) → Document category classification!



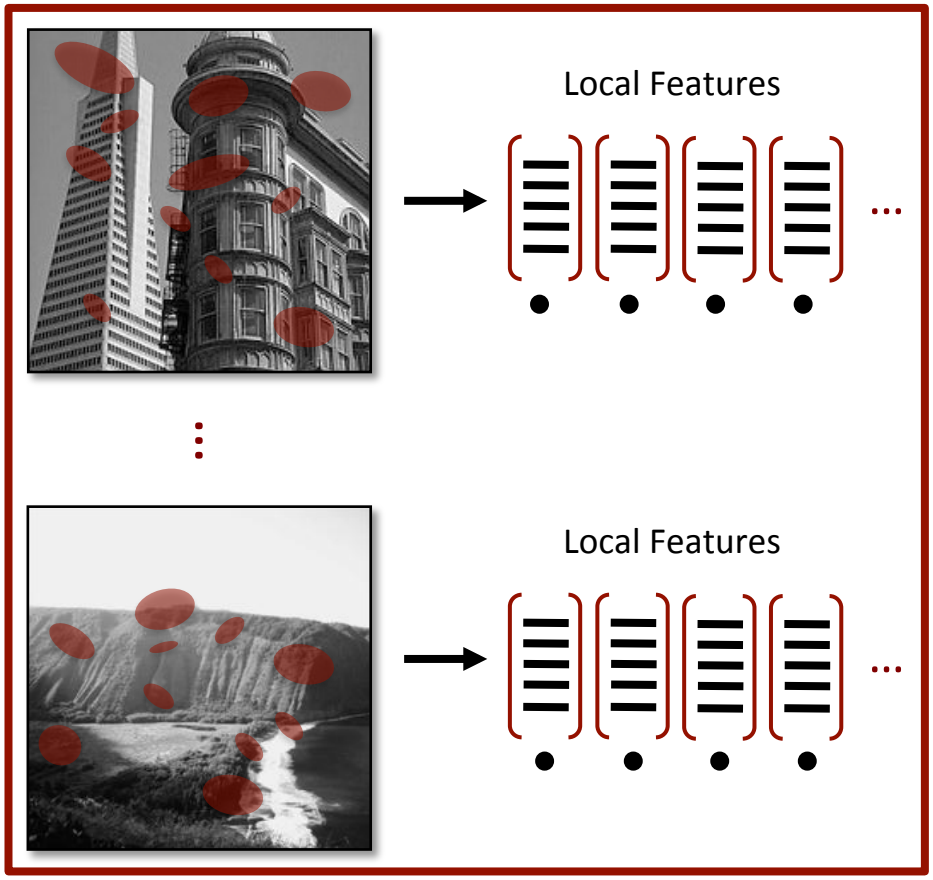
Tag cloud; Courtesy of http://chir.ag/projects/preztags/)

Salton, J. and McGill, M. J., "Introduction to Modern Information retrieval", McGraw-Hill, 1983

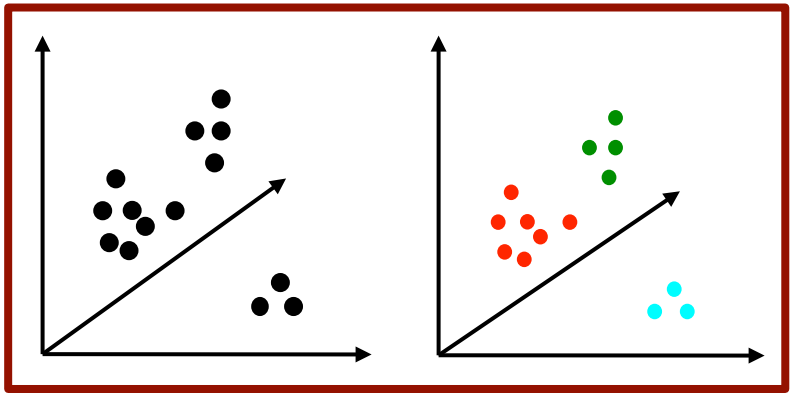
Bag-of-Features / Bag-of-Words

Basic pipeline – Establishing a "dictionary"

Bag-of-Features (BoF), *e.g.*, using SIFT, SURF, etc.



Visual Dictionary (*e.g.*, using K-means clustering)



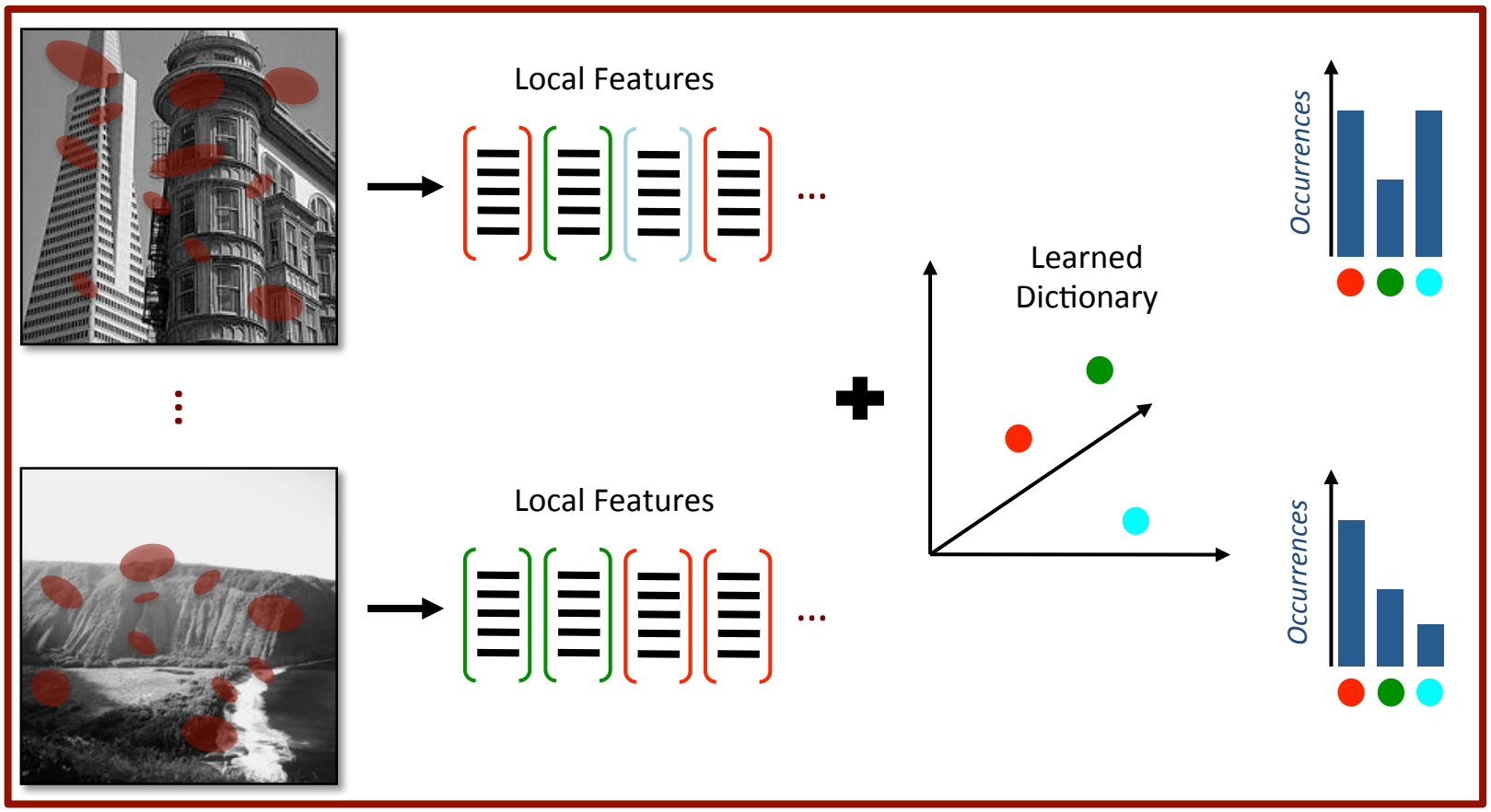
Note: The clusters do NOT necessarily correspond to the classes in our data!

Slides adapted from C. Schmid

Bag-of-Features / Bag-of-Words

Basic pipeline – Encoding as a BoW histogram

Encoding (through vector quantization)



These steps (except for clustering) are computed for **every** new image!

Slides adapted from C. Schmid

Bag-of-Features / Bag-of-Words

Dense grid vs. interest points

Descriptors: at interest points *vs.* at positions of a dense grid

Dense grids typically lead to favorable performance on many recognition problems (*cf.* [Fei-Fei & Perona, 2005])

Slide adapted from S. Lazebnik

Fei-Fei Li and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories", CVPR, 2005

*Interlude

K-means clustering

Objective: identify K groups / clusters in a high-dimensional space given a set of data points

$$\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}, \mathbf{x}_i \in \mathbb{R}^D$$

What is a cluster? Intuitively, a group of data points whose inter-point distances are smaller compared to distances to points outside of the cluster.

Formally, K-means **minimizes**

$$\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

with respect to

$$\{r_{nk}\}, \{\boldsymbol{\mu}_k\}$$

$\{r_{nk}\}$: 1 if point $\mathbf{x}_n$ is assigned to cluster k, 0 else

$\{\boldsymbol{\mu}_k\}$: represent the "center" of k^{th} cluster

Finding an optimal solution is NP-hard

*Interlude

K-means clustering

Approximate solution: the problem can be "solved" with a simple two-step alternating optimization algorithm:

(1) $$r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0, & \text{else} \end{cases}$$

fixed in this step!

In other words, simply "assign the n-th point to its closest cluster"!

fixed in this step!

(2) $$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

In other words, update the "representative" of each cluster with the mean of all points assigned to that cluster → K-means clustering!

Iterate until no more changes in the assignment of the points! (Easy way to initialize, not a good one though: **randomly** pick K points; better K-Means++)

*Interlude

K-means clustering – Practical issues

When you have many images + descriptors (in a high-dimensional space, e.g., SIFT descriptors), K-means can be quite **computationally expensive**!

Pragmatic solution (typically done in the literature): Take a random sample of M points and run K-means clustering. Finally assign all points to their closest clusters!

*Interlude

K-medians clustering

When we deal with descriptors where the Euclidean distances makes sense*, e.g.*, SIFT, SURF, etc., K-means clustering gives us good results, but …

What are we going to do with bit vectors?

- *E.g.*, use K-medians with Hamming distance
  (implemented in `pycluster` , https://pypi.python.org/pypi/Pycluster)

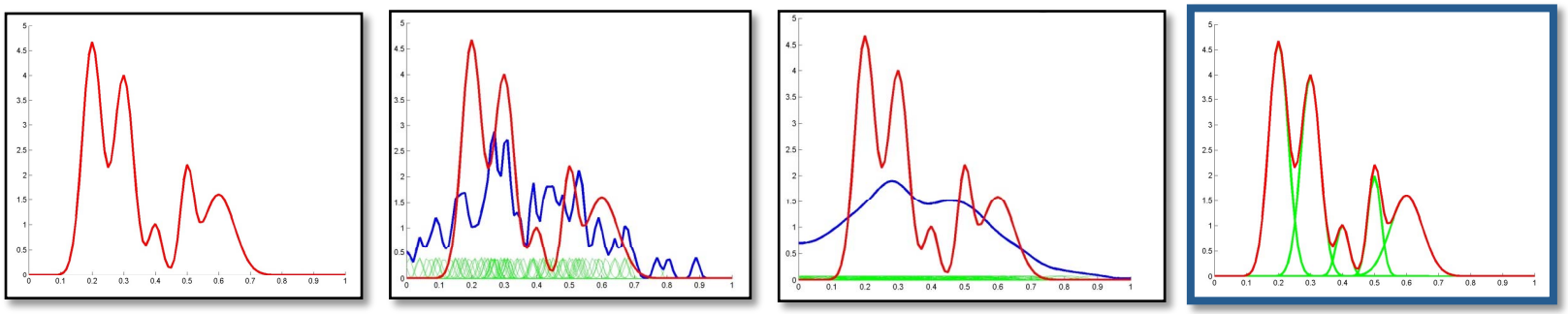
Another interesting idea could be [Grana et al., 2013]

C. Grana et al., "A Fast Approach for Integrating ORB Descriptors in the Bag of Words Model", SPIE Electronic Imaging, 2013

*Interlude

"Soft"-clustering via Gaussian mixtures

Say, we have data where the empirical distribution looks like…



large variance large bias

Clearly not Gaussian!

What can we do about it?

e.g., use kernel density estimation (but: bias/variance tradeoff!)

We could to try to find the "right" number of Gaussians, though!

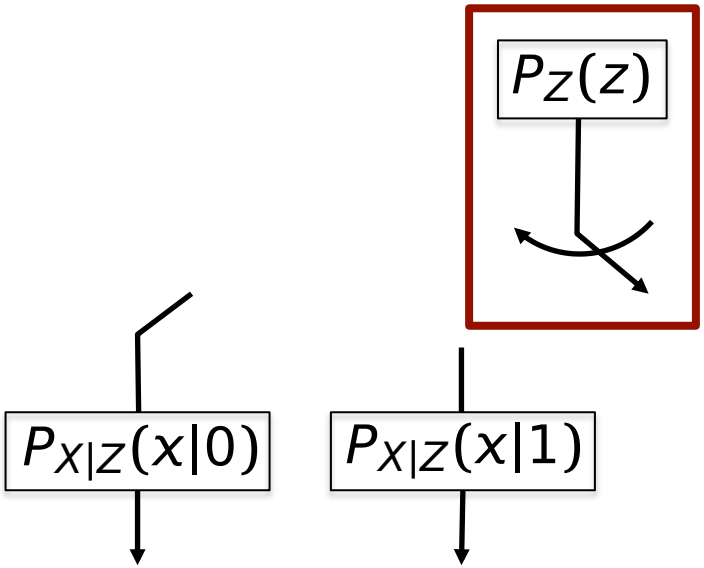
Slide adapted from N. Vasconcelos

*Interlude

"Soft"-clustering via Gaussian mixtures

We have **two types** of random variables

- Z – **hidden** state variable
- X – **observed** variable



sample a state
(from distribution of the hidden variable Z)

$$P_Z(z) = z_i$$

draw observation from class-conditional distribution for the selected state

$$P_{X|Z}(x|z_i) \rightarrow x_i$$

Slide adapted from N. Vasconcelos

*Interlude

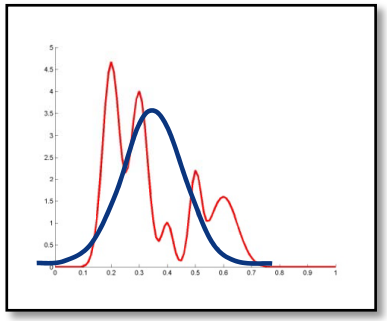
"Soft"-clustering via Gaussian mixtures

- We **never** get to see the realization of the hidden variable Z
- The probability density function (pdf) of the observed data is

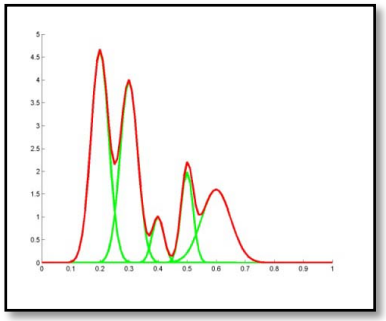
$$P_{\mathbf{X}}(\mathbf{x}) = \sum_{c=1}^{C} P_{\mathbf{X}|Z}(\mathbf{x}|c)P_Z(c)$$

$$= \sum_{c=1}^{C} P_{\mathbf{X}|Z}(\mathbf{x}|c)\pi_c, \text{ with } \sum_c \pi_c = 1$$

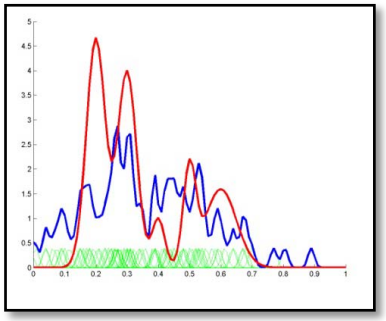
#components (C)



parametric, C=1



mixture of C components



mixture with C=N

Slide adapted from N. Vasconcelos

*Interlude

Gaussian mixtures

Advantages, …

- with respect to parametric estimates
  - more degrees of freedom → **less bias**
- with respect to kernel density estimates (KDE)
  - fewer #components → less parameters, **less variance**

Disadvantages, …

- **learning complexity** (we always need to resort to numerical procedures)
  (standard way of learning: expectation-maximization (EM) algorithm)
- compared to: parametric estimates:
  - if MLE's have closed-form solution → simple estimation (otherwise: numerical solutions)
- compared to: non-parametric estimates:
  - *e.g*, NN: just store samples; *e.g.*, KDE: place kernel on top of each sample

Slide adapted from N. Vasconcelos

*Interlude

"Soft"-clustering via Gaussian mixtures

Gaussian Mixture Model (GMM):

$$P_{\mathbf{X}}(\mathbf{x}) = \sum_{c=1}^{C} \pi_c \underbrace{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}_{P_{\mathbf{X}|Z}(\mathbf{x}|c)}$$

Estimation using EM: *e.g.*, [Bishop, 2006; Chapter 9.2]

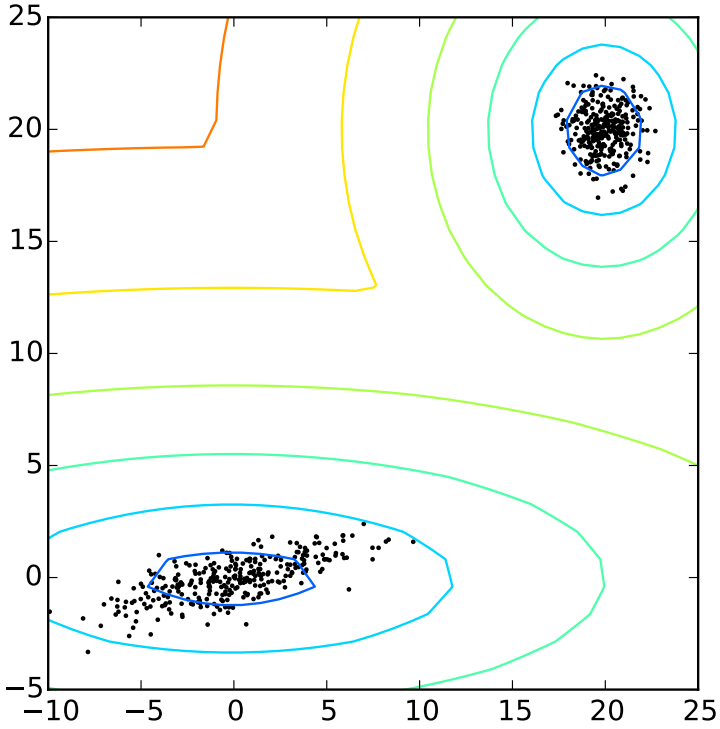
The **basic principle** of EM is simple:

1. Initialize parameters (*e.g.*, using k-means) from the data $\mathbf{x}_1,\ldots,\mathbf{x}_N$
2. **E-step:** "Guess" the hidden values of the z_i (using the current parameter set)
3. **M-step:** Use the new z_i to solve for the parameters, *i.e.*, compute new estimates
4. goto step 2)

Bishop, C., "Pattern Recognition and Machine Learning", Springer, 2006

*Interlude

“Soft”-clustering via Gaussian mixtures (using scikit-learn)




```
from sklearn import mixture

model = mixture.GMM(
     n_components=2,
     covariance_type='diag',
     random_state=1234)
model.fit(X_train)


# E.g., predict class membership
pnt0 = np.zeros((1,2)) # at (0,0)
pnt1 = np.ones((1,2))*20 # at (20,20)

print model.predict(pnt0)
Print model.predict(pnt1)
```

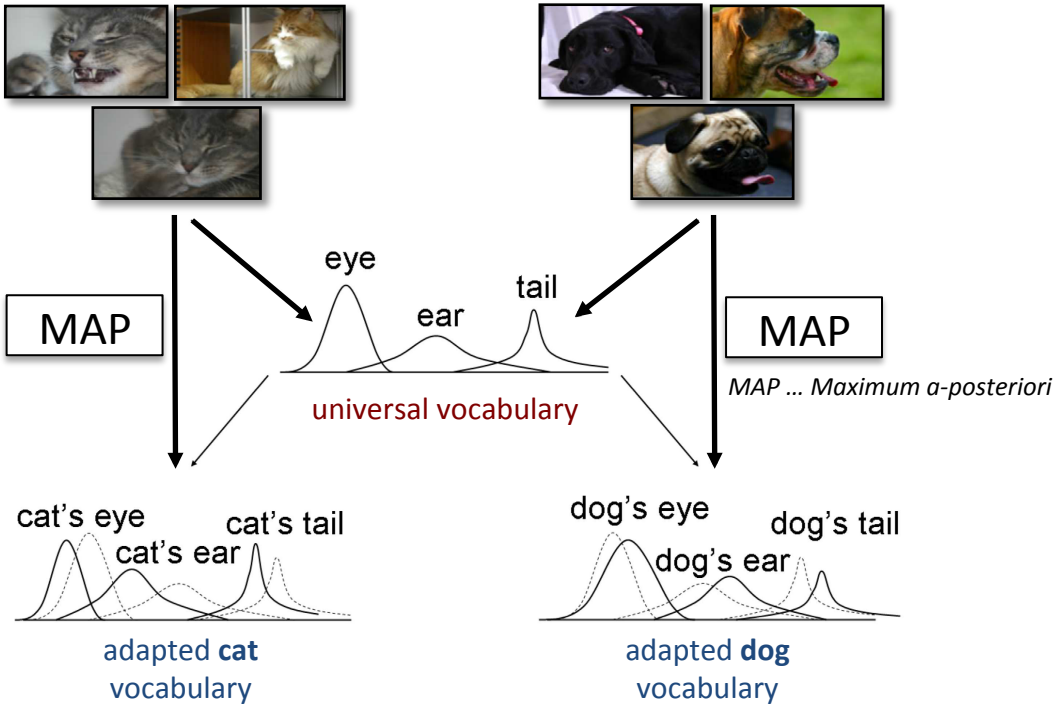
Adapted form the scikit-learn documentation

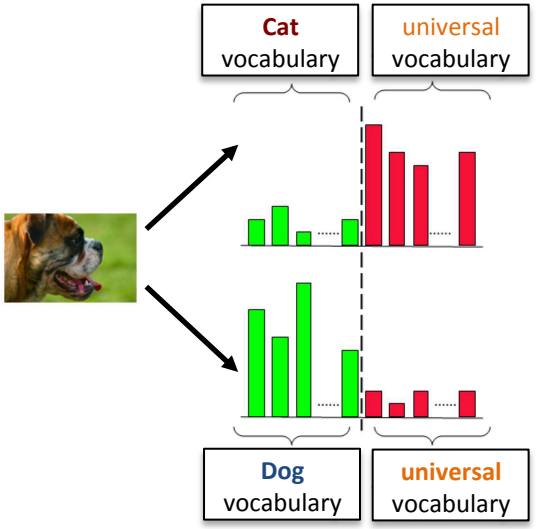
Bag-of-Features / Bag-of-Words

Example of codebook generation with GMMs

Codebook generation:

- using k-means (*e.g.*, [Csurka et al., 2004])
- using **mixtures of Gaussians** (*e.g.*, [Perronnin et al., 2006])





Graphics adapted from [Perronnin et al., 2006]

Csurka, G., Dance, C., Fan, L., Willamowski, J. and Bray, C., "Visual Categorization with Bags of Keypoints", ECCV, 2004

Perronnin, F., Dance, C., G. Csurka and M. Bressan, "Adapted Vocabularies for Generic Visual Categorization", ECCV, 2006

Bag-of-Features / Bag-of-Words

Summary

Advantages:

- **Unsupervised** learning
- Codebook can be learned on separate training set of images
- "Sufficiently large" training corpus → "Universal" dictionary

But, …

- No spatial information!
- Possible loss of information due to hard assignments!
- How do we choose the right vocabulary size?
  - Too small: Vocabulary will not be rich enough
  - Too large: Danger of "overfitting"

  (In practice: typical vocabulary sizes range from 500 – 6000)

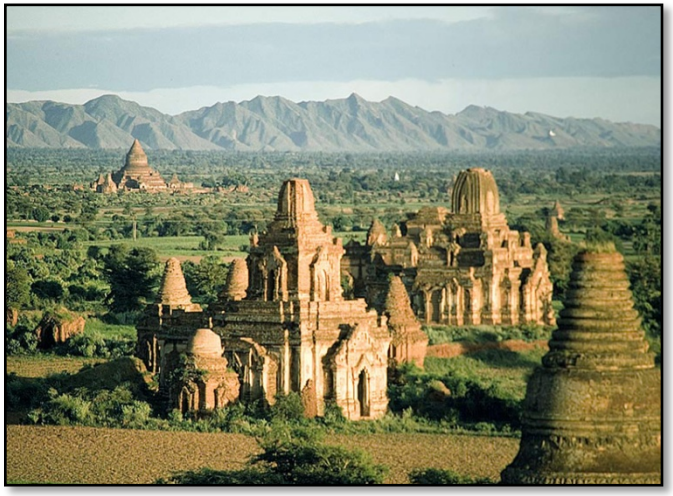
Today's lecture

Overview

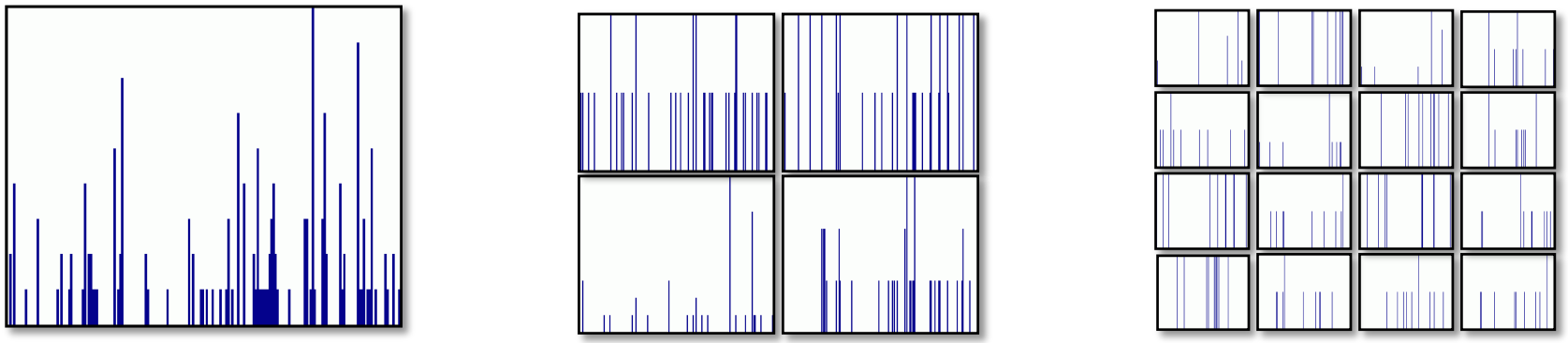
- Introduction
- Bag-of-Features (BOF) / Bag-of-Visual-Words (BOV)
- **Spatial pyramid encoding**
- "Semantic" encodings & Attributes
- Fisher vector encoding

Spatial pyramid encoding

Basic principle [Lazebnik et al., 2006]



Finally, BoW histograms from all cells in the spatial pyramid are concatenated!



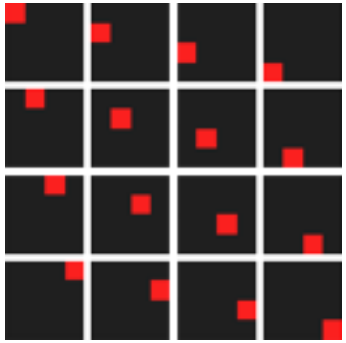
Slide adapted from S. Lazebnik

Lazebnik, S., Schmid, C. and Ponce, J., "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories", CVPR, 2006

Spatial pyramid encoding

Can we do better than spatial pyramids?

Recent work on adaptive pooling regions [Yangqing & Chang, 2011]



"Superpixel" (basic building blocks for pooling regions)

Pooling regions in SPM [Lazebnik et al., 2006]

Overcomplete rectangular regions

Some key points, …

- Even random selection from the over-complete features improves performance!
- By "learning" the receptive fields, considerable improvements can be achieved!

Another recent approach: [Xu & Vasconcelos, 2014]

Yangqing, J. and Chang, H., "Beyond Spatial Pyramids: Receptive Field Learning for Pooled Image Features", NIPS 2011
Xu, C. and Vasconcelos, N., "Learning Receptive Fields for Pooling from Tensors of Feature Response", CVPR 2014