计算机视觉

邬向前

计算学部

多模态智能及应用研究中心

电子邮箱: xqwu@hit.edu.cn

Bag-of-Words models

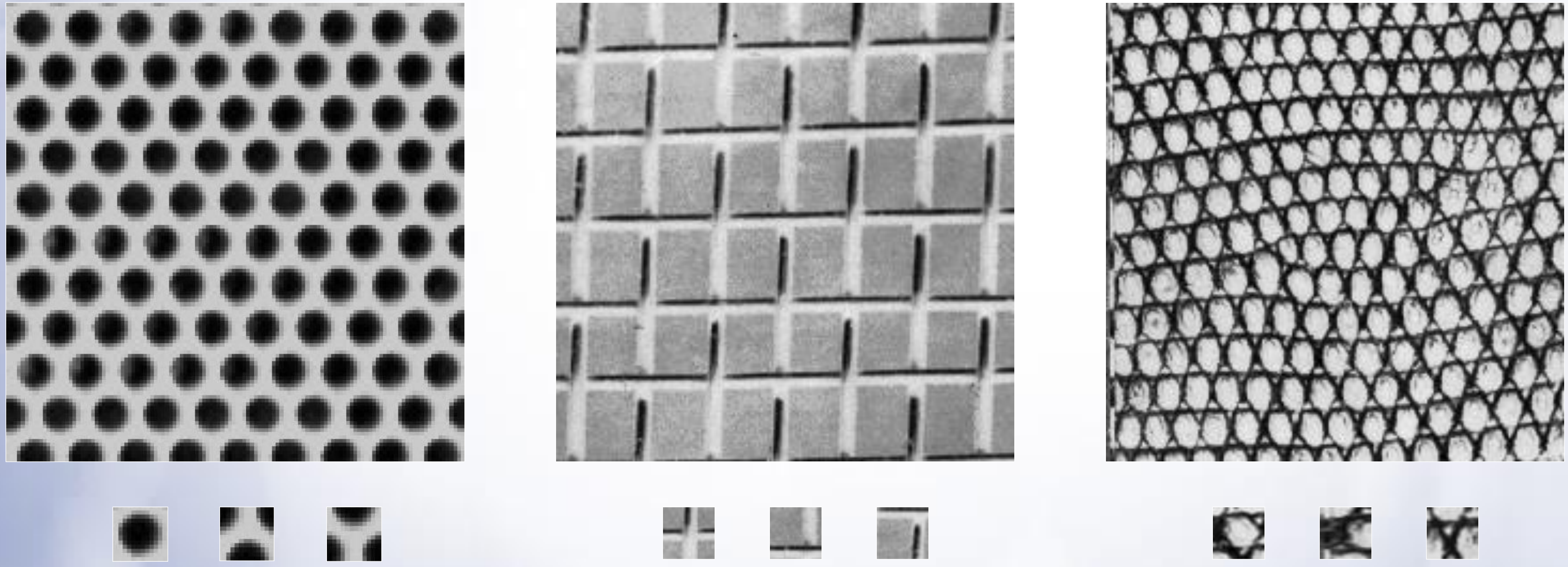
Bag-of-features models

Overview: Bag-of-features models

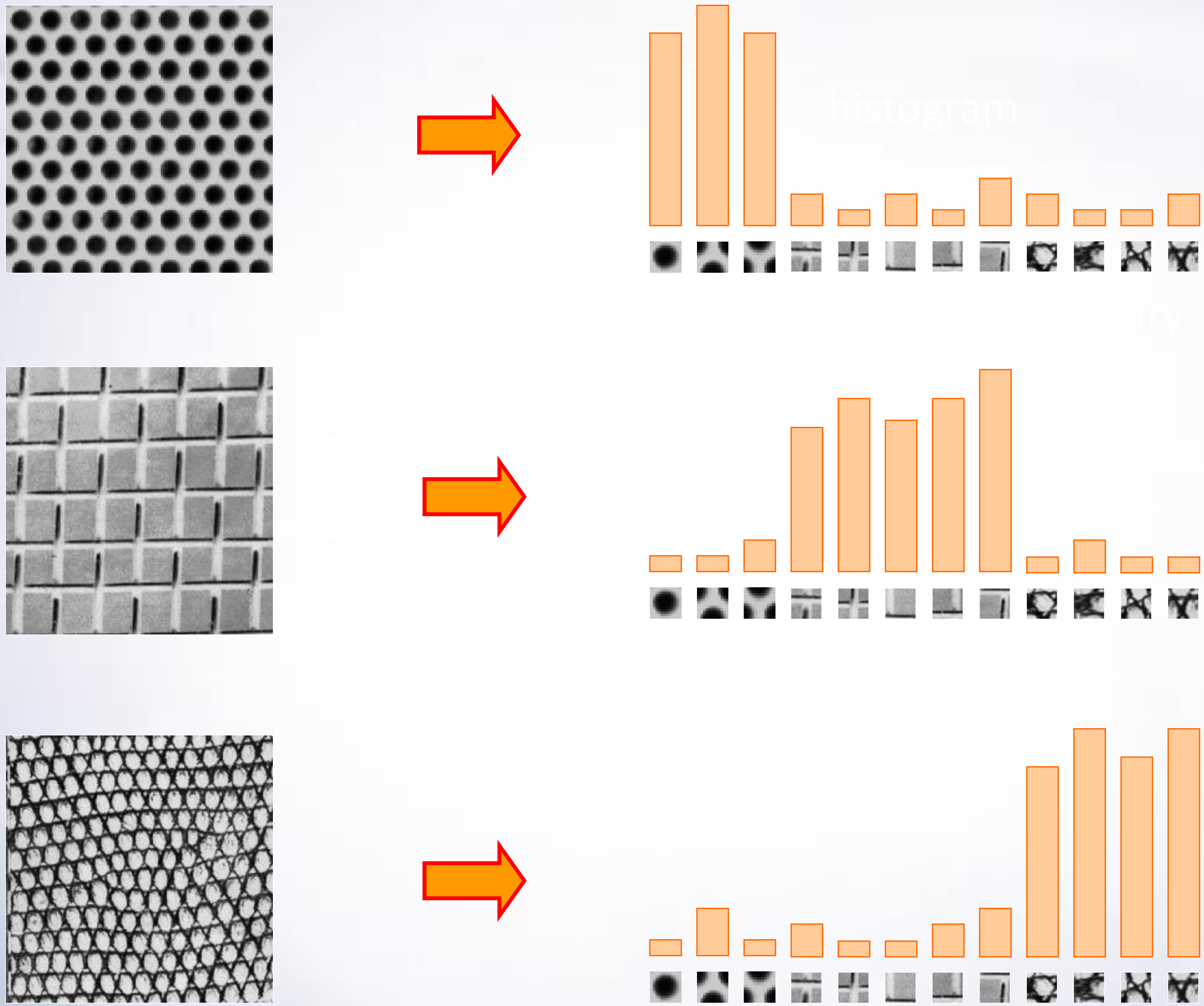
- Origins and motivation
- Image representation
- Discriminative methods
  – Nearest-neighbor classification
  – Support vector machines
- Generative methods
  – Naïve Bayes
- Extensions: incorporating spatial information

Origin 1: Texture recognition

- 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



Origin 1: Texture recognition



Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)

US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/

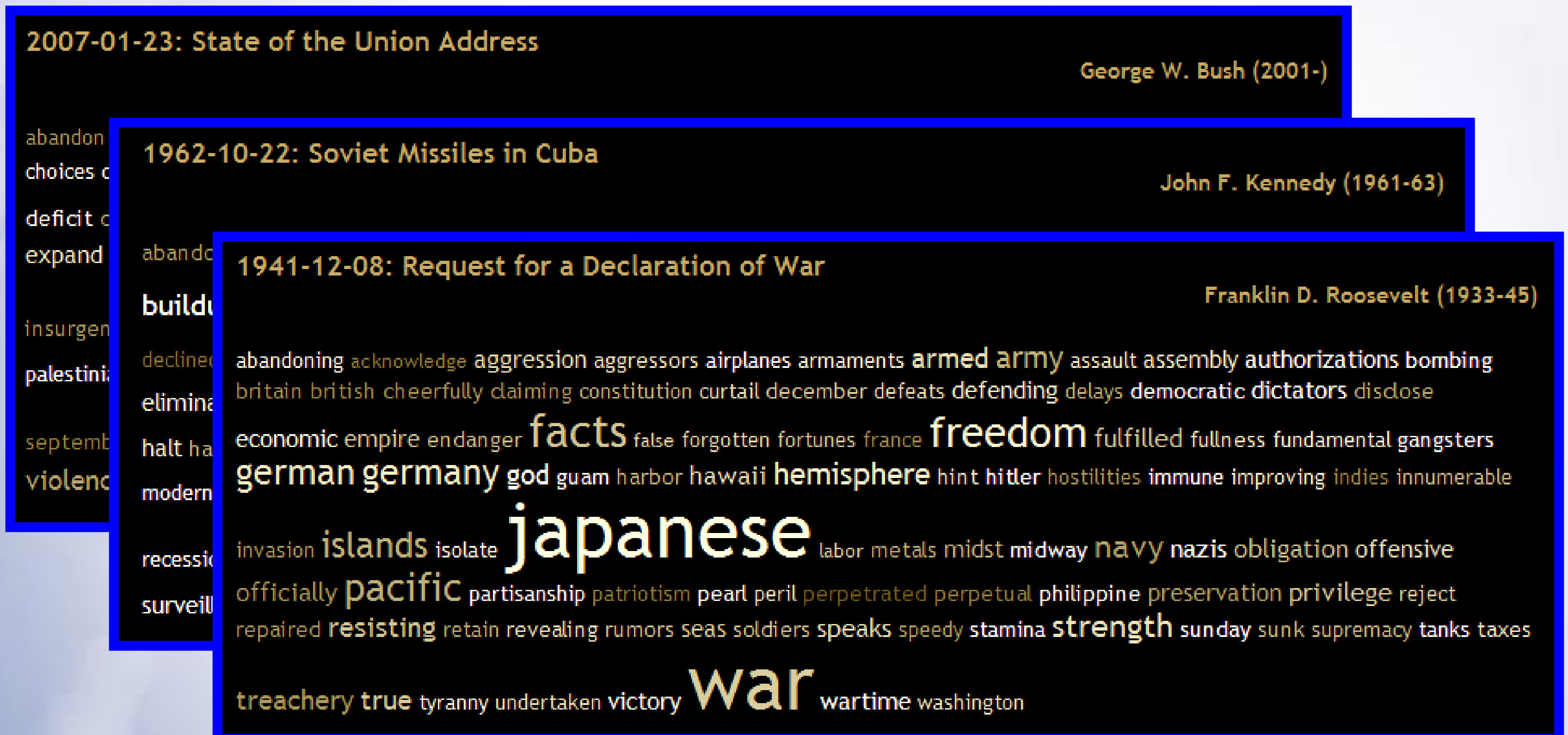
Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)

US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/

Origin 2: Bag-of-words models

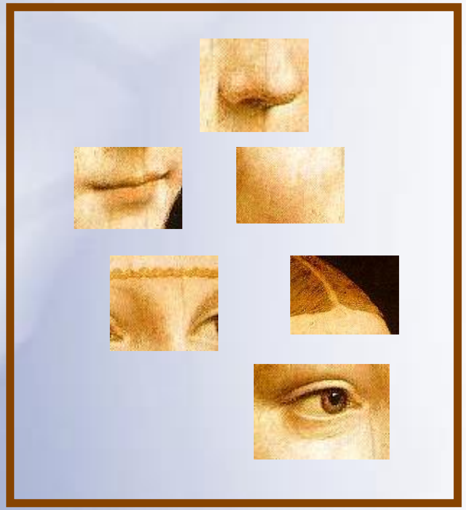
- Orderless document representation: frequencies of words from a dictionary Salton & McGill (1983)



US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/

Bags of features for image classification

1. Extract features



Bags of features for image classification

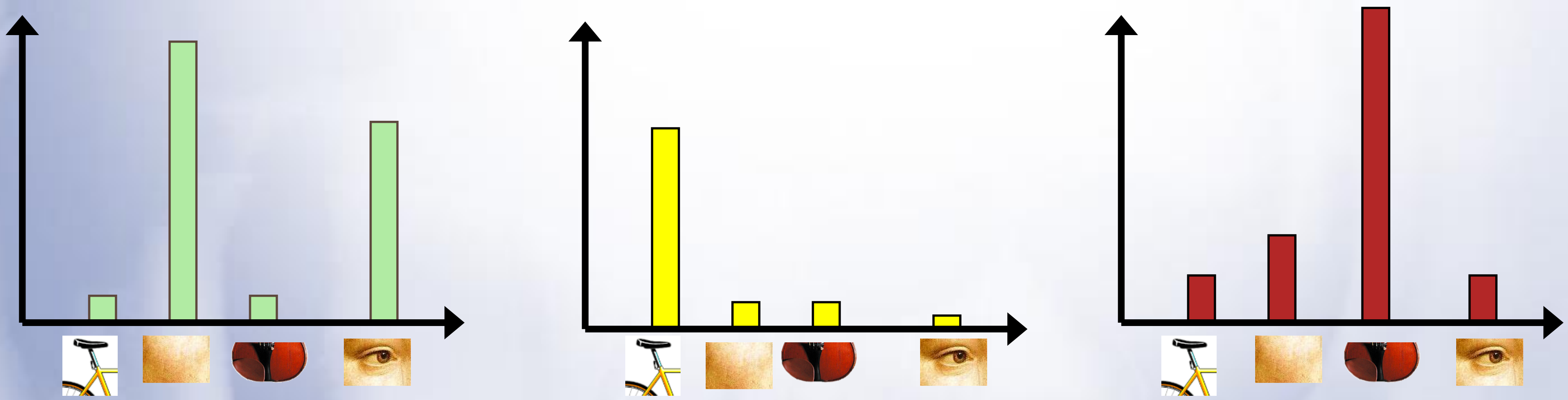
1. Extract features
2. Learn "visual vocabulary"

Bags of features for image classification

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary

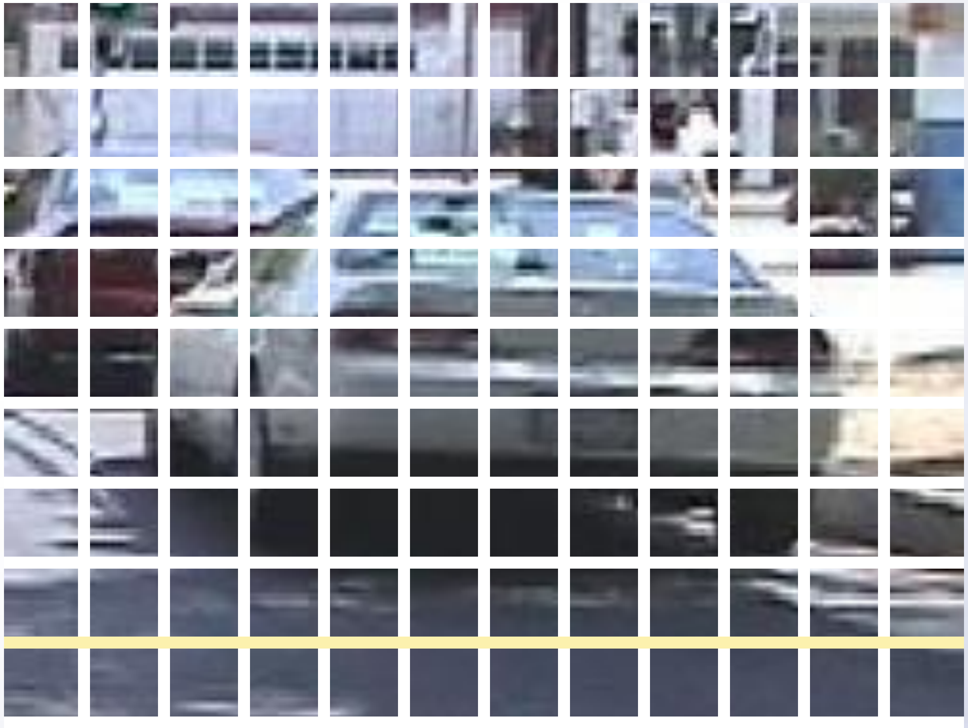
Bags of features for image classification

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”



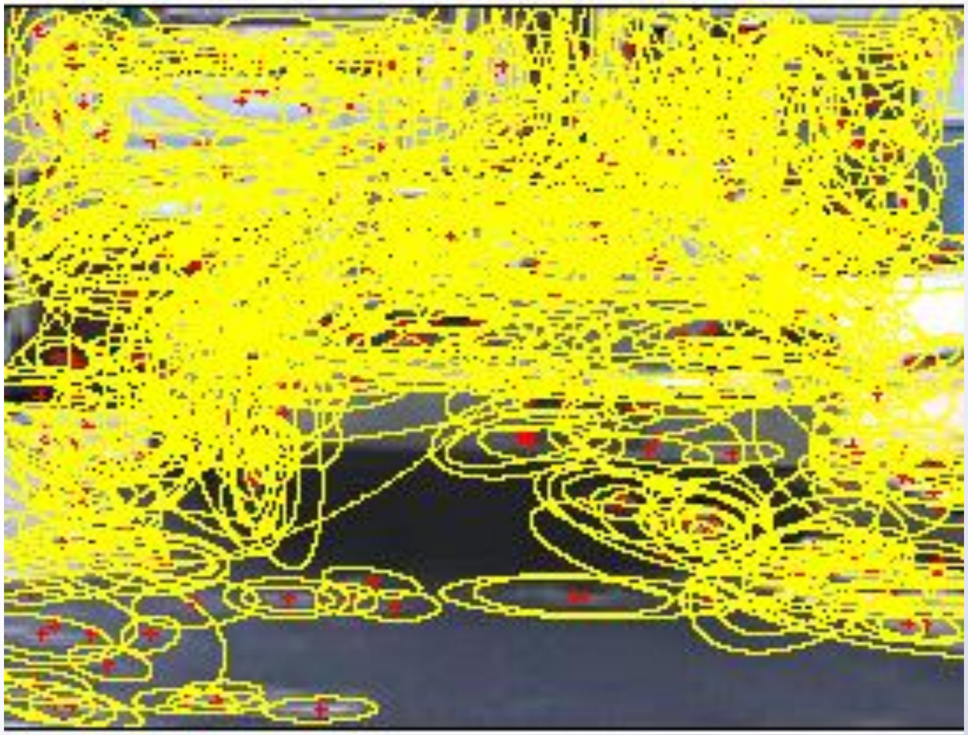
1. Feature extraction

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005



1. Feature extraction

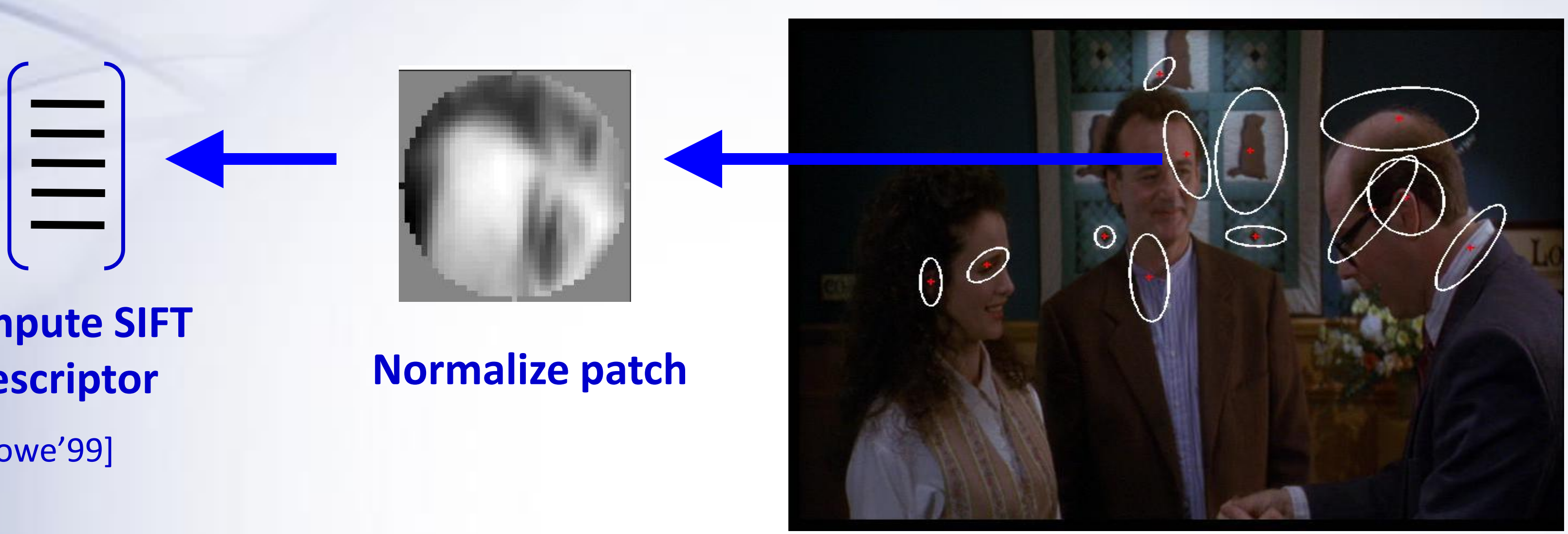
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005



1. Feature extraction

- 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)

1. Feature extraction



Compute SIFT descriptor

[Lowe'99]

Normalize patch

Detect patches

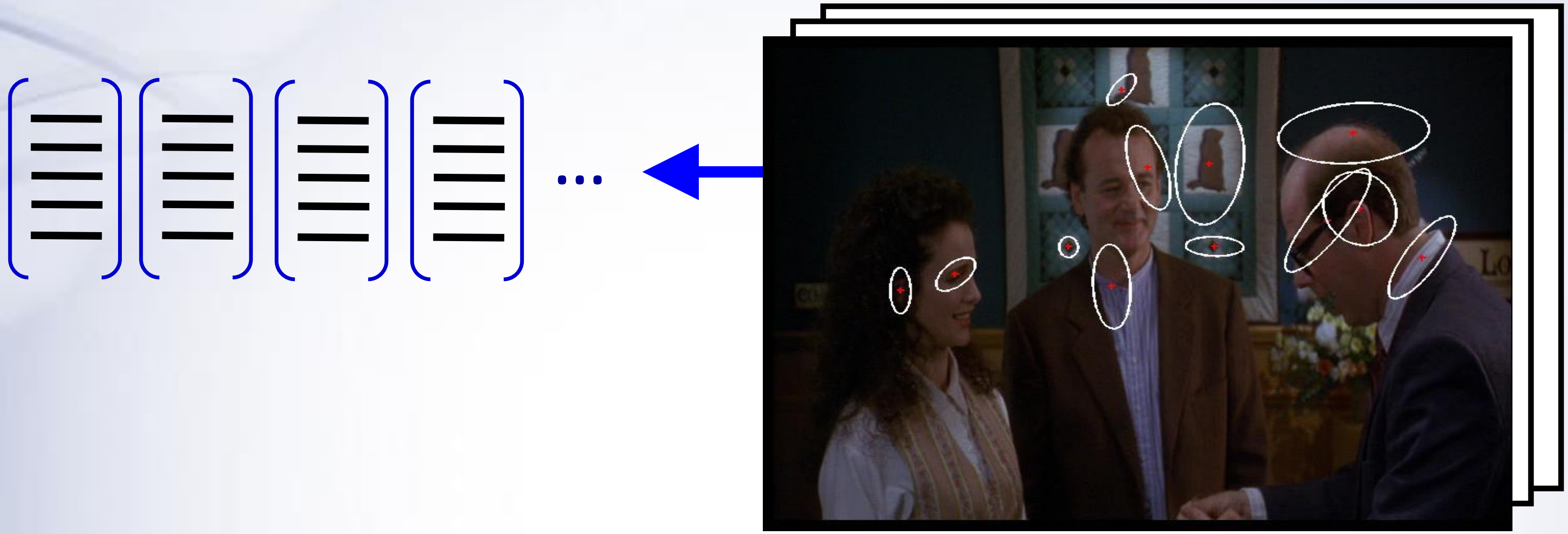
[Mikojaczyk and Schmid '02]

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

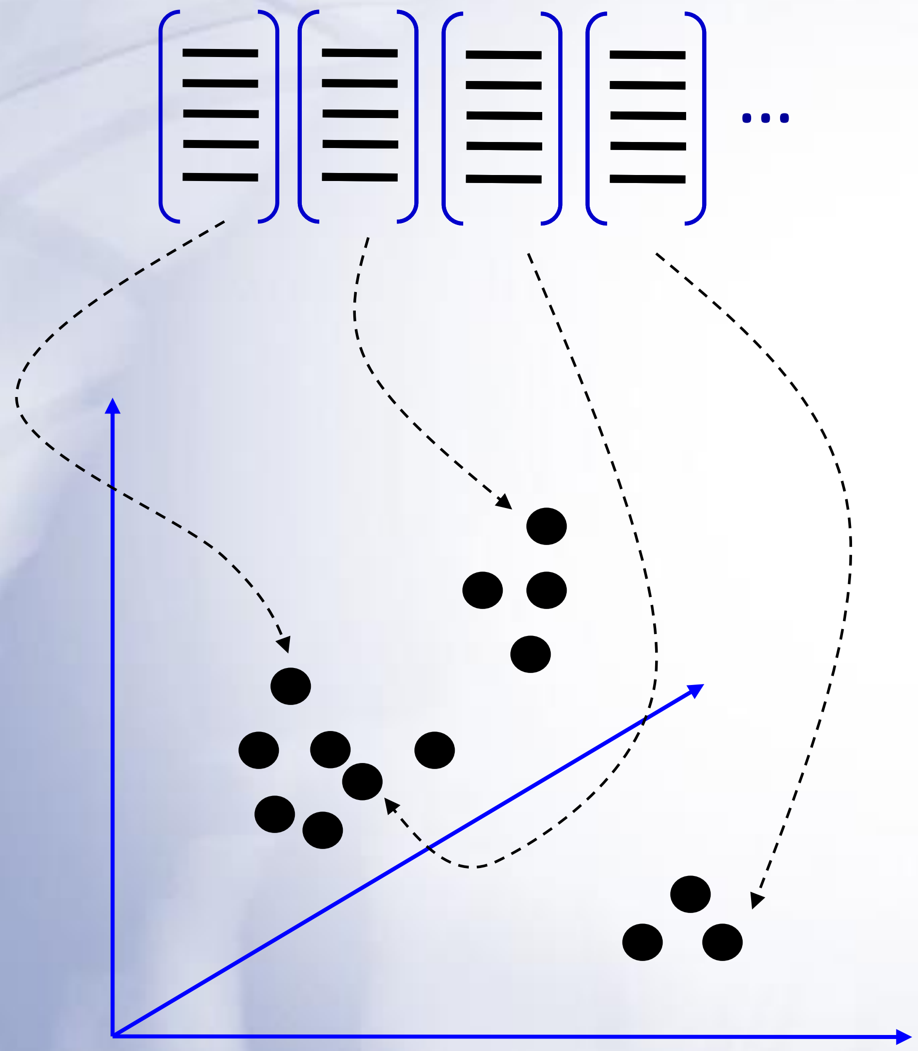
[Sivic & Zisserman, '03]

Slide credit: Josef Sivic

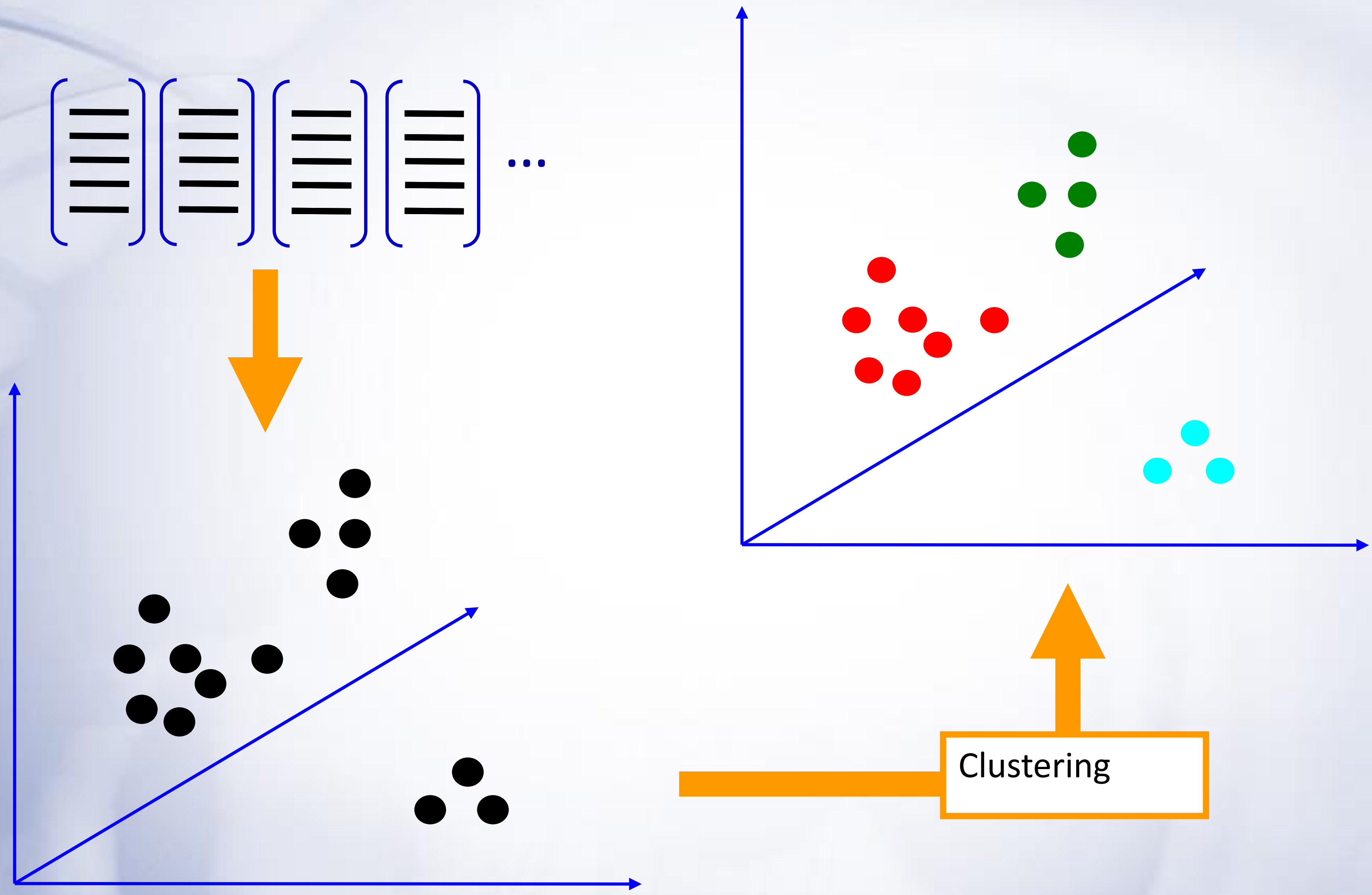
1. Feature extraction



2. Learning the visual vocabulary

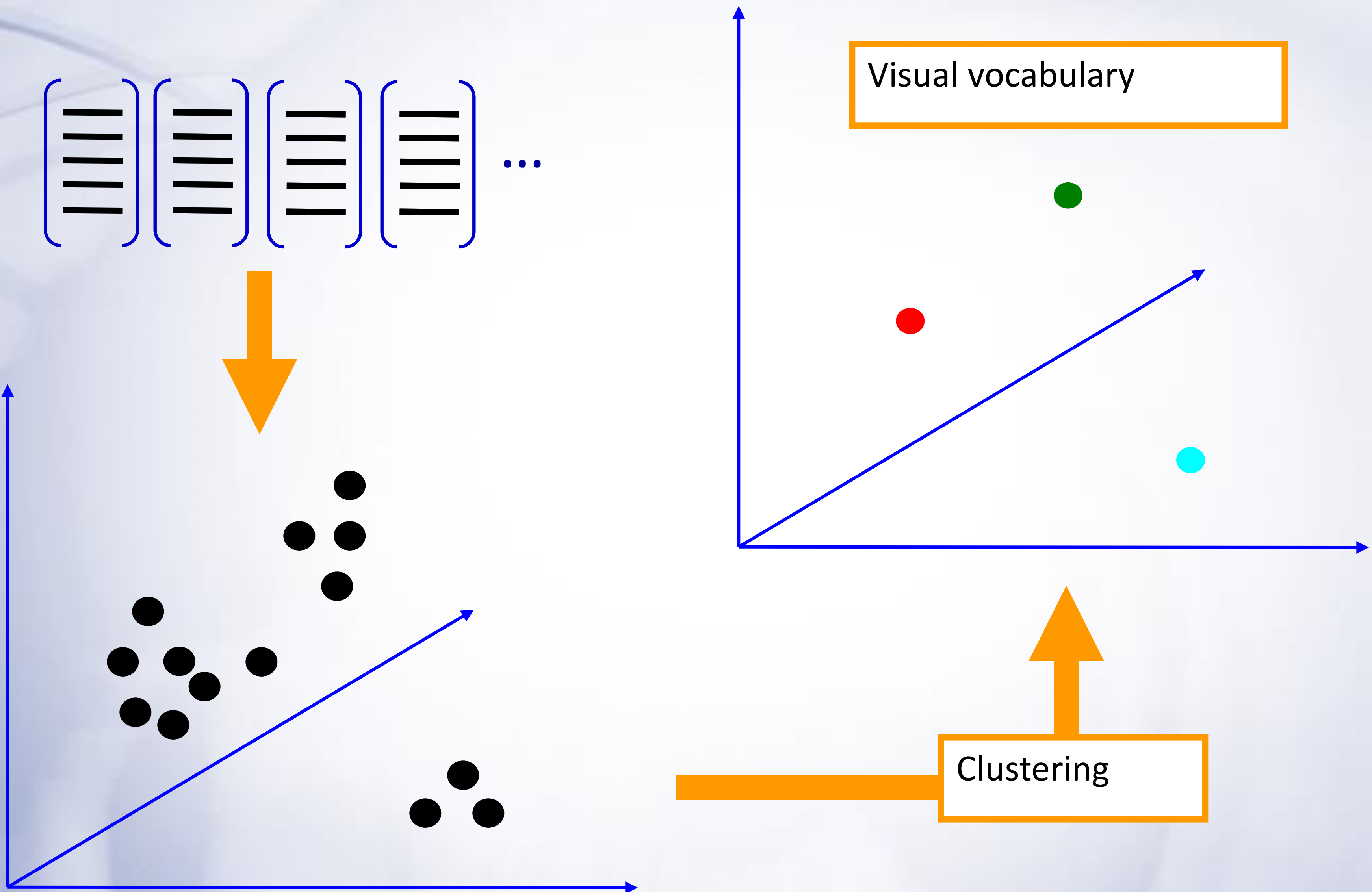


2. Learning the visual vocabulary



Slide credit: Josef Sivic

2. Learning the visual vocabulary



Slide credit: Josef Sivic

K-means clustering

- Want to minimize sum of squared Euclidean distances between points x_i and their nearest cluster centers m_k

$$D(X, M) = \sum_{\text{cluster}\,k} \sum_{\substack{\text{point}\,i\text{ in} \\ \text{cluster}\,k}} (x_i - m_k)^2$$

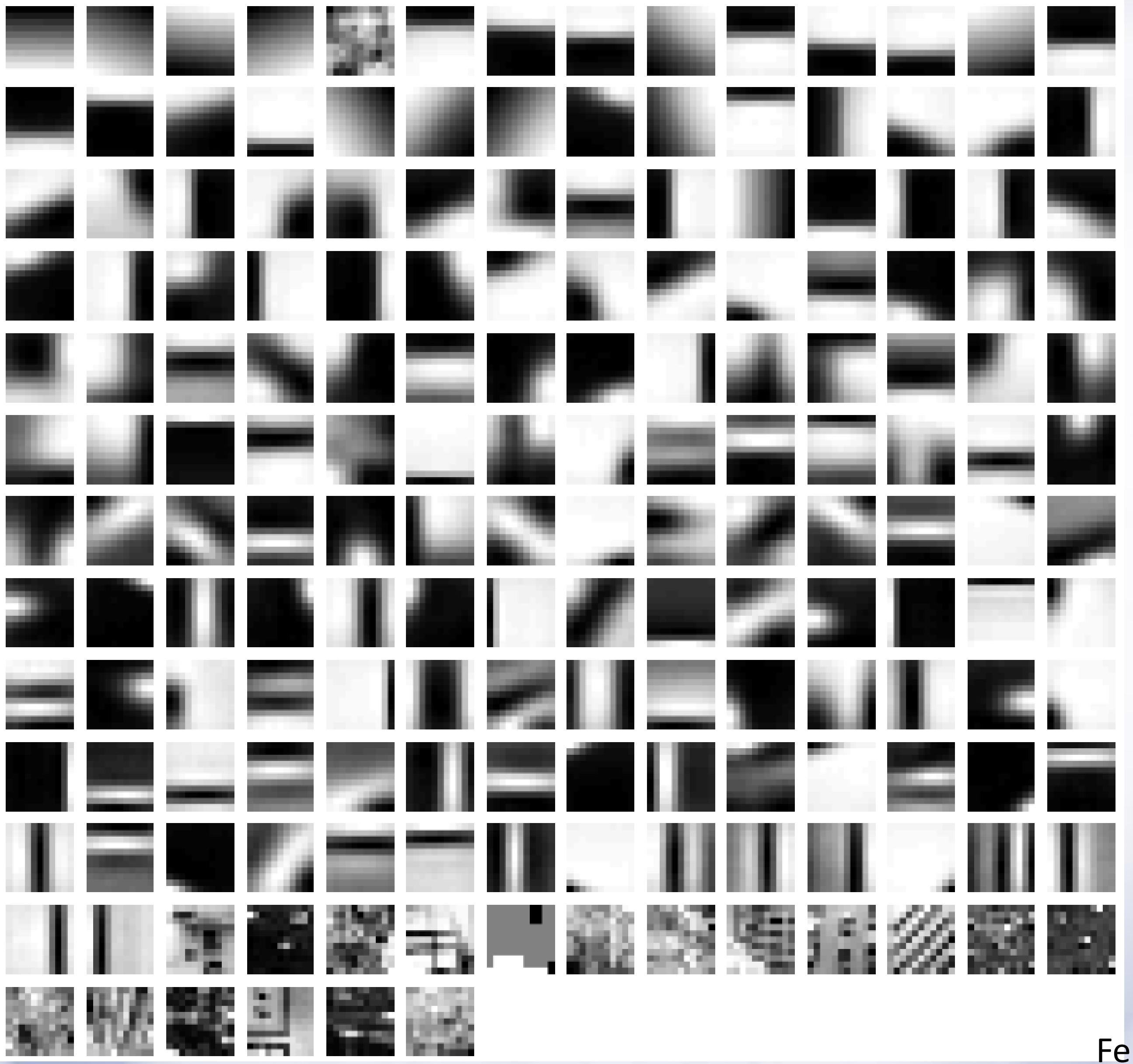
- Algorithm:
- Randomly initialize K cluster centers
- Iterate until convergence:
  - Assign each data point to the nearest center
  - Recompute each cluster center as the mean of all points assigned to it

From clustering to vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be "universal"

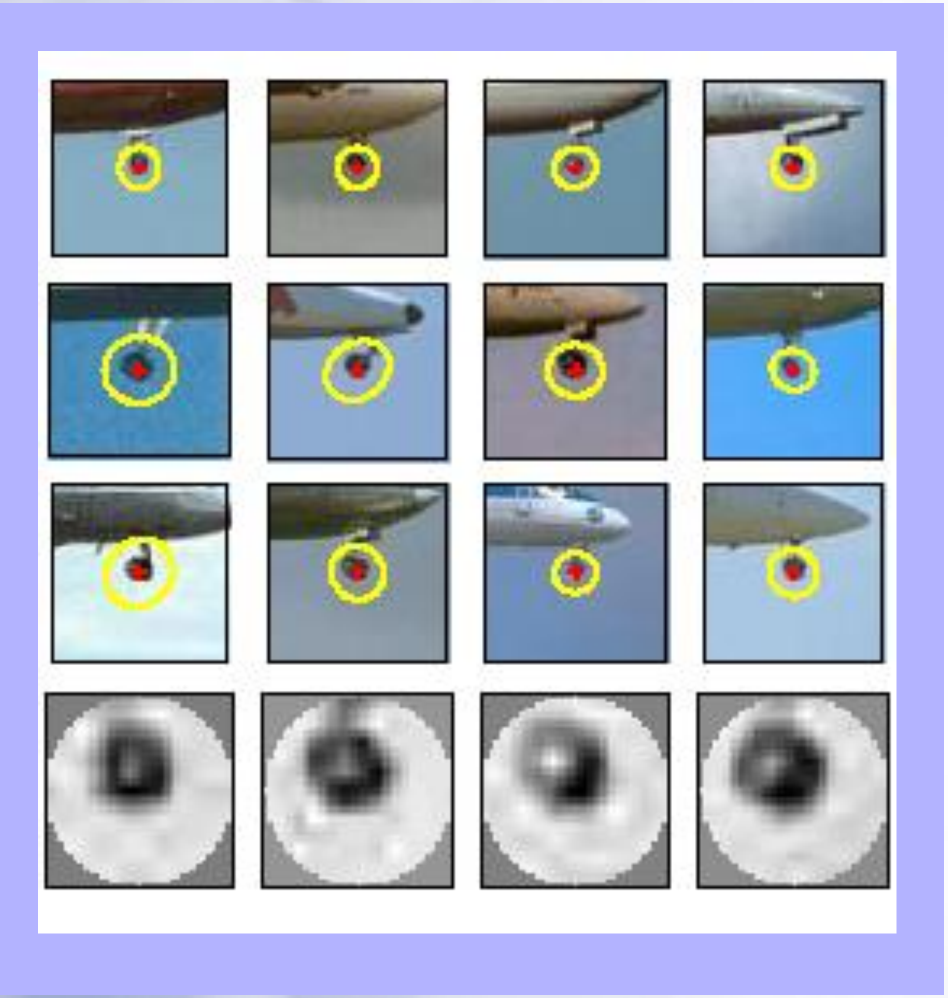
- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word

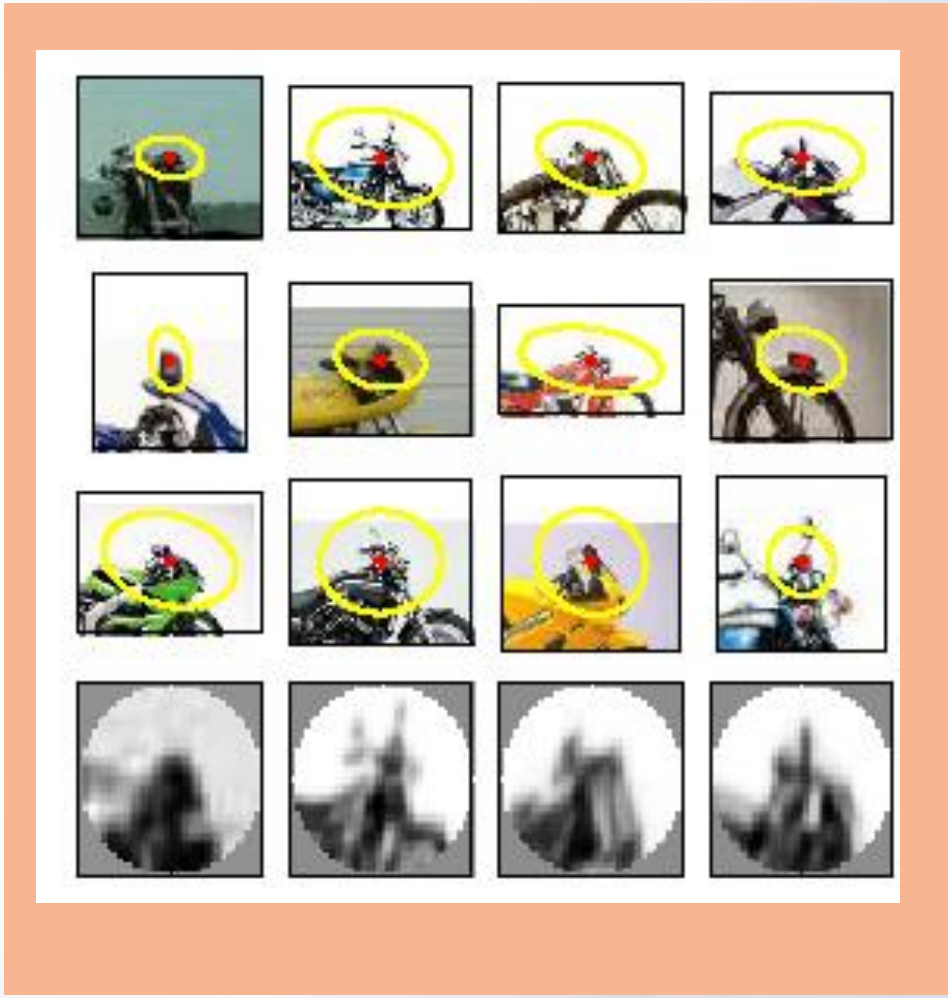
Example visual vocabulary



Fei-Fei et al. 2005

Image patch examples of visual words

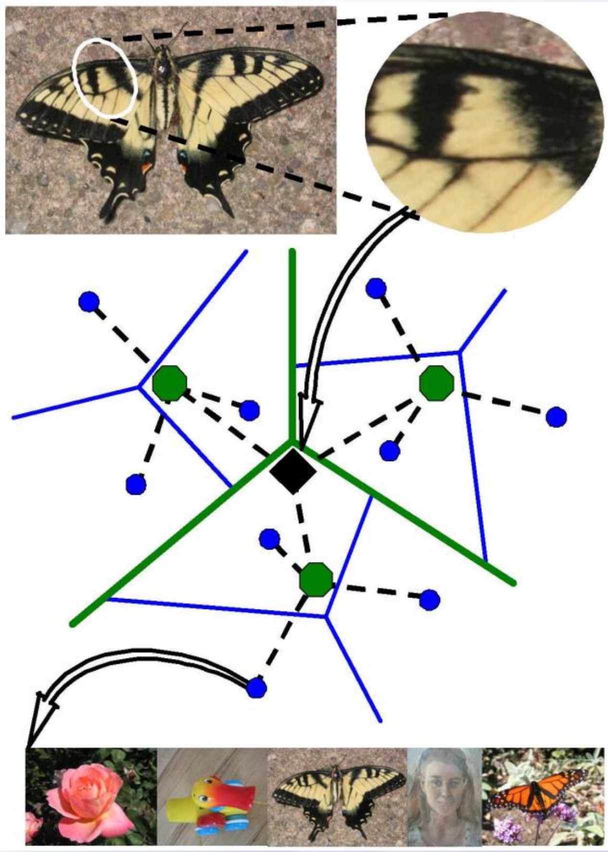




Sivic et al. 2005

Visual vocabularies: Issues

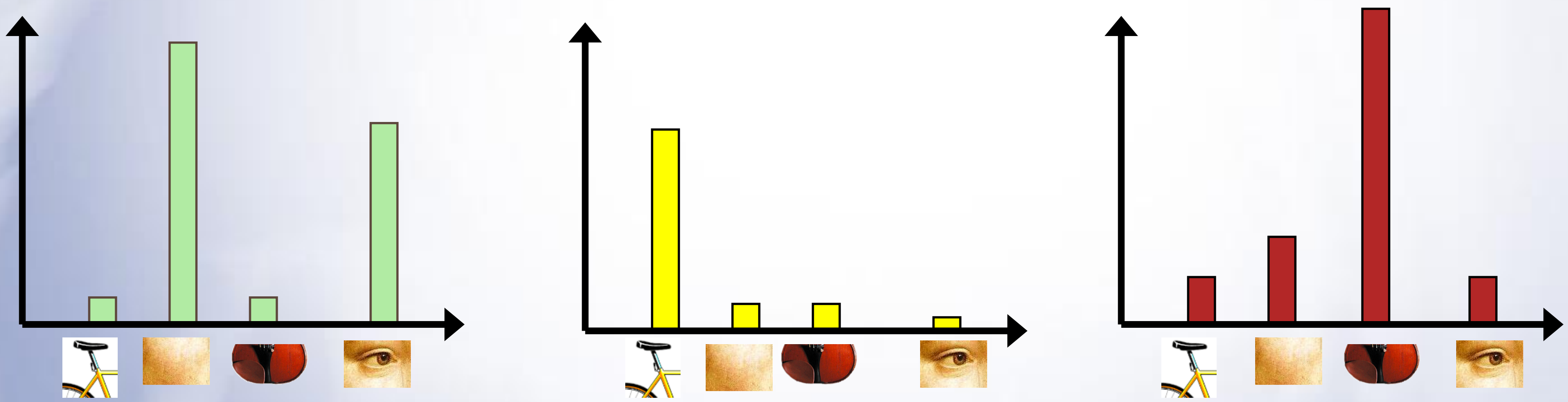
- How to choose vocabulary size?
  - Too small: visual words not representative of all patches
  - Too large: quantization artifacts, overfitting
- Generative or discriminative learning?
- Computational efficiency
  - Vocabulary trees (Nister & Stewenius, 2006)



3. Image representation

Image classification

- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?



Discriminative and generative methods for bags of features

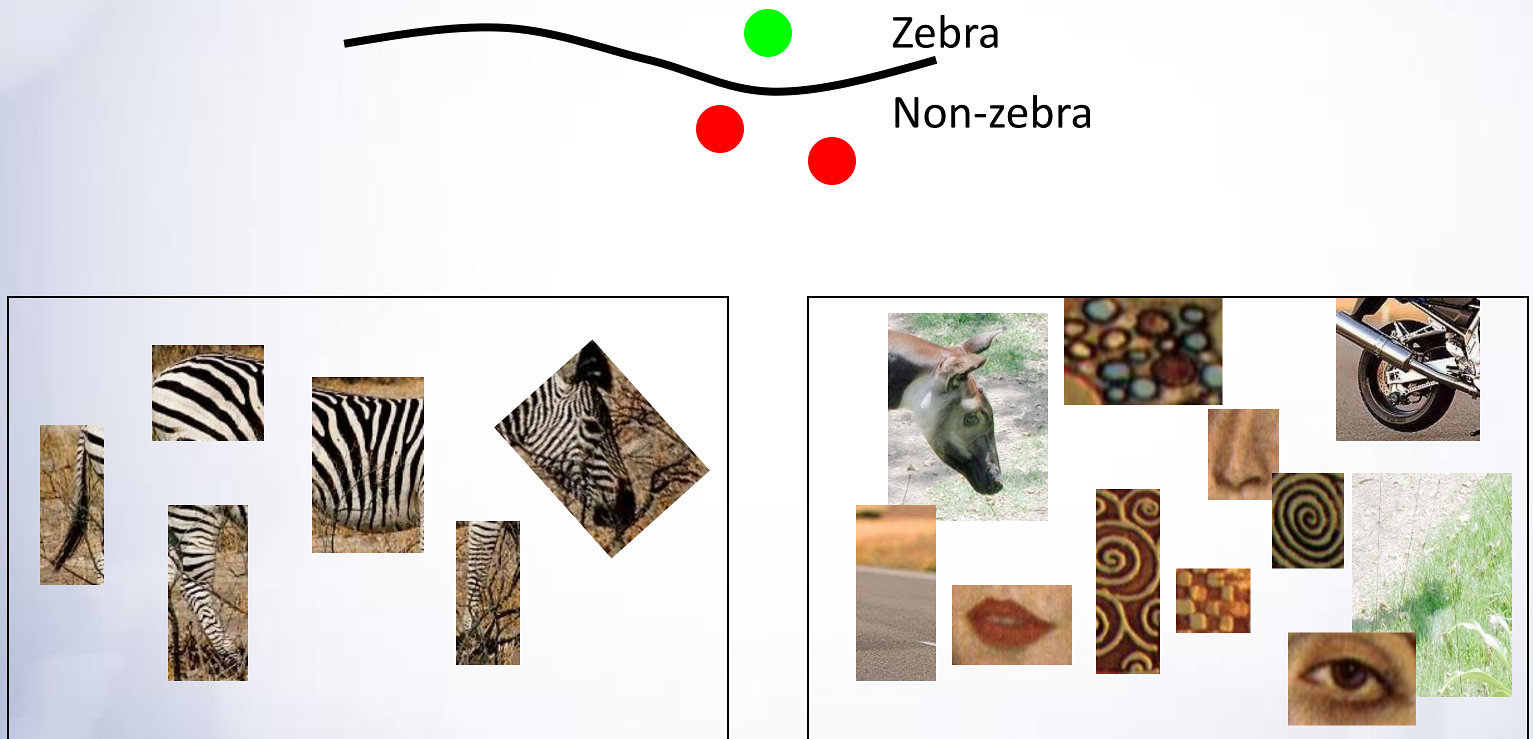
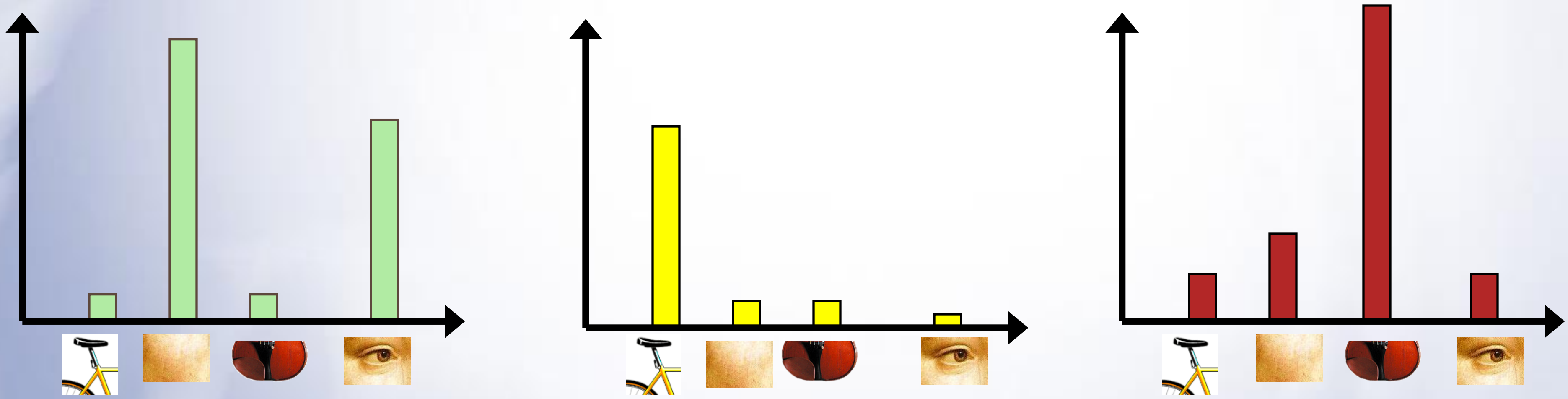


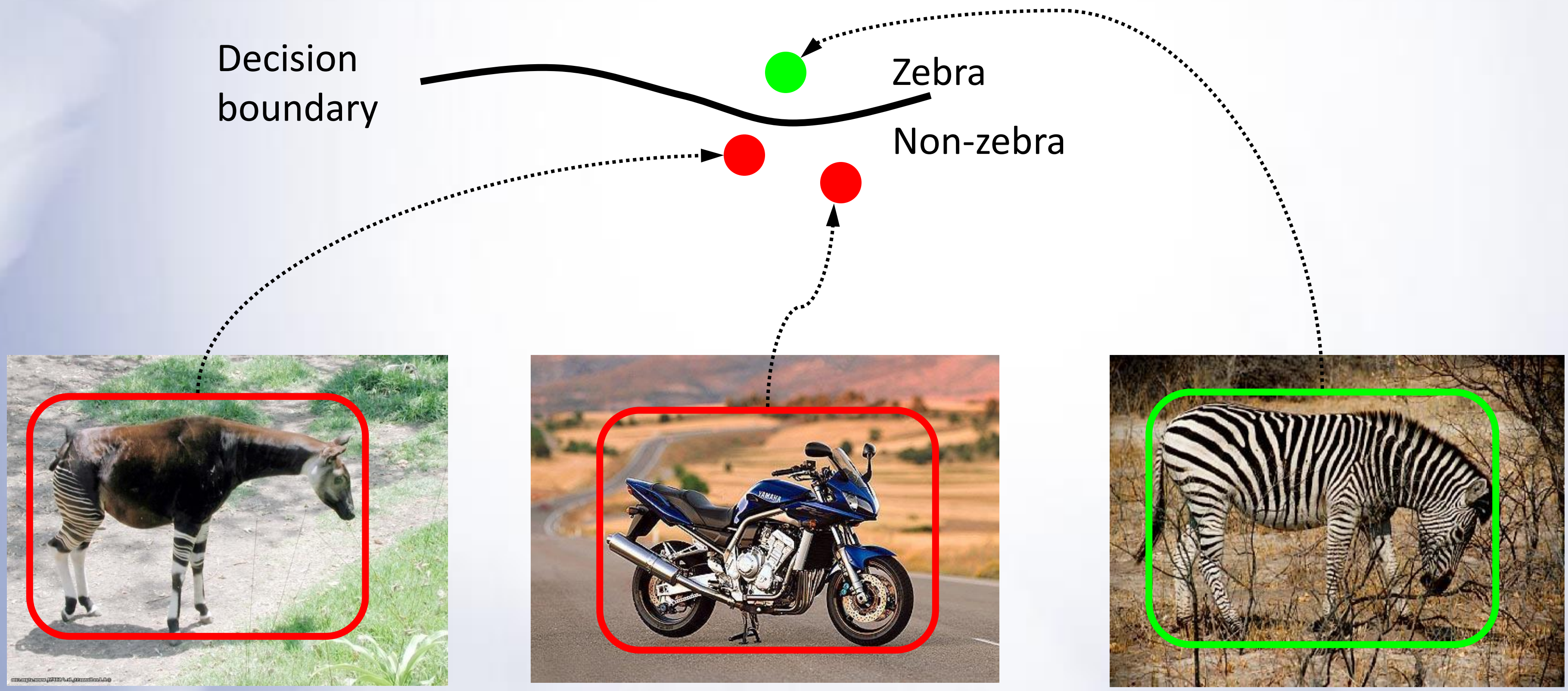
Image classification

- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?



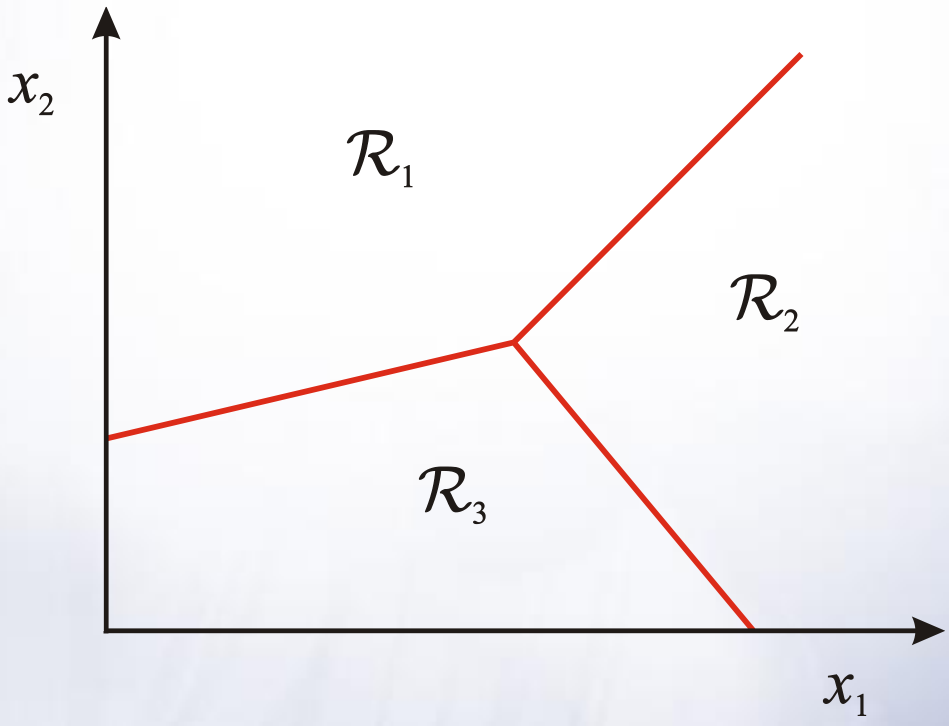
Discriminative methods

- Learn a decision rule (classifier) assigning bag-of-features representations of images to different classes



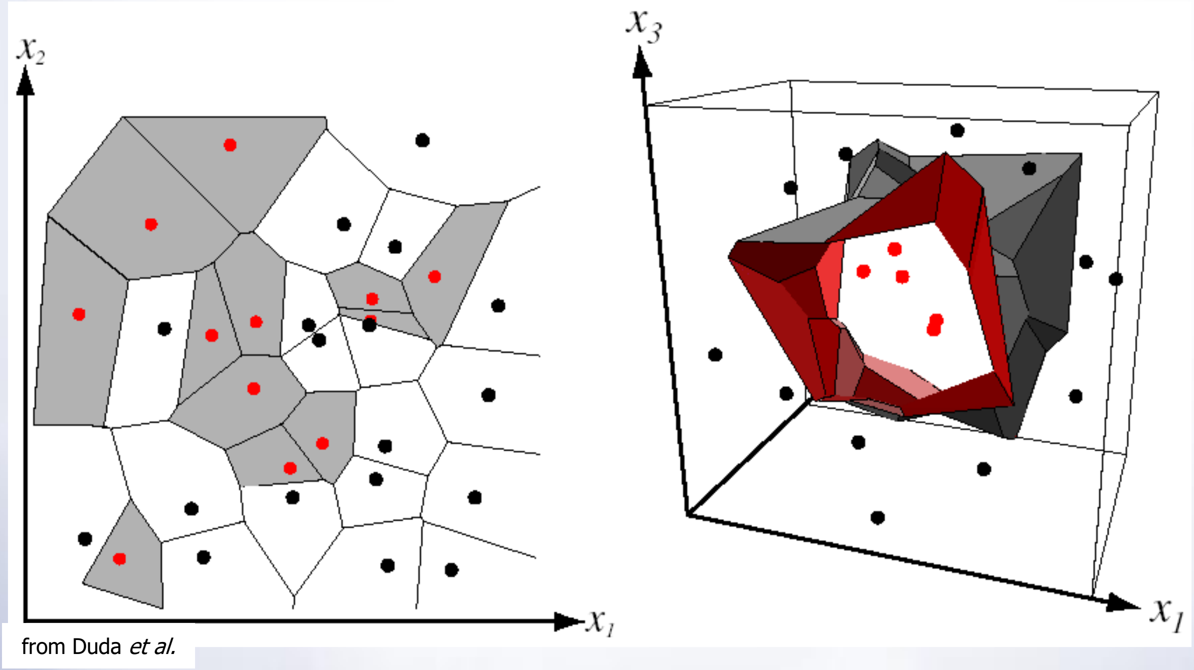
Classification

- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*



Nearest Neighbor Classifier

- Assign label of nearest training data point to each test data point

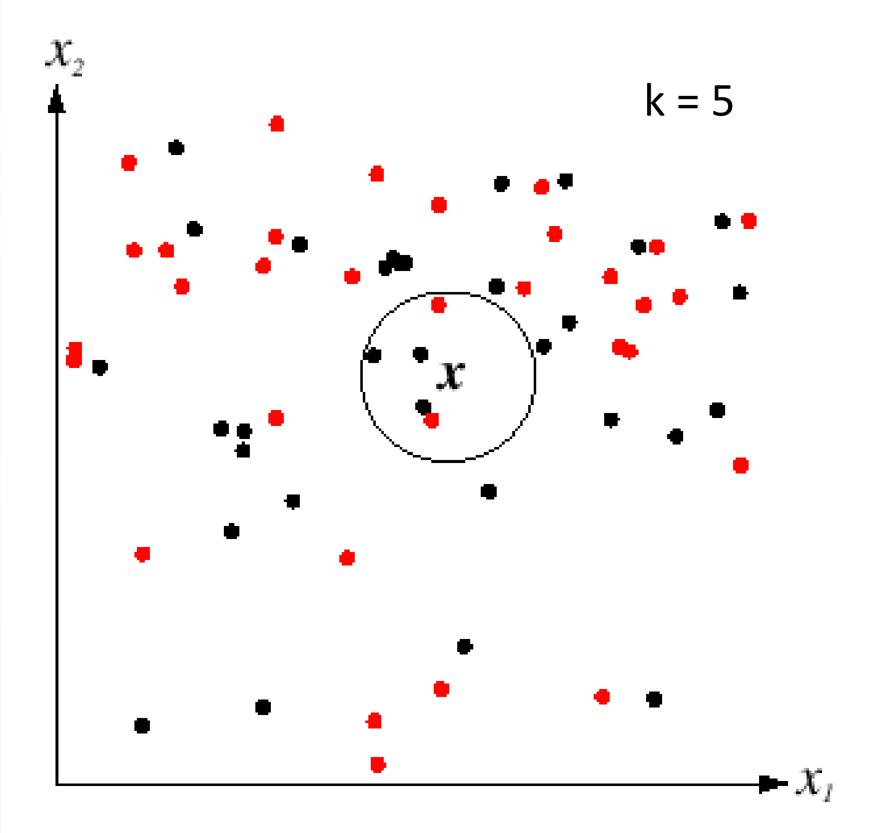


Voronoi partitioning of feature space for two-category 2D and 3D data

Source: D. Lowe

K-Nearest Neighbors

- For a new point, find the k closest points from training data
- Labels of the k points "vote" to classify
- Works well provided there is lots of data and the distance function is good



Source: D. Lowe

Functions for comparing histograms

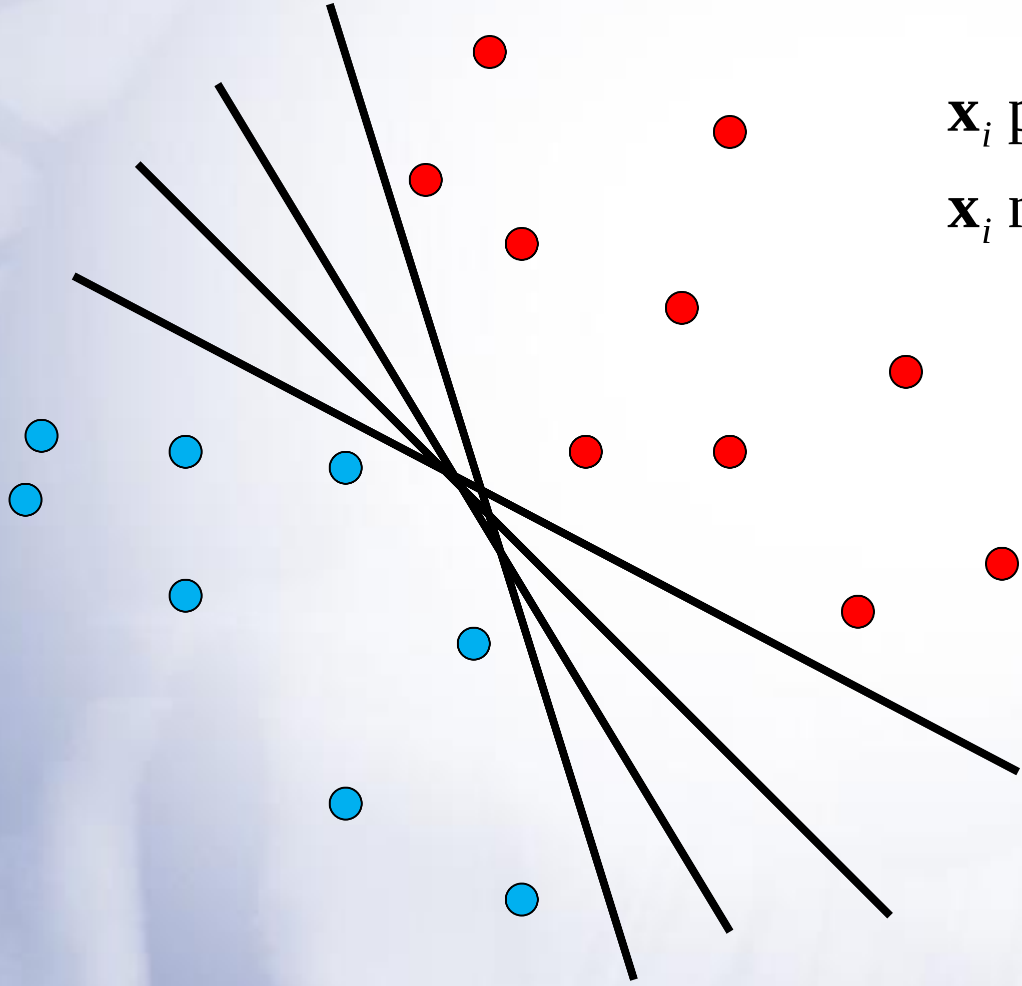
- L1 distance $$D(h_1, h_2) = \sum_{i=1}^{N} | h_1(i) - h_2(i) |$$

- χ^2 distance $$D(h_1, h_2) = \sum_{i=1}^{N} \frac{\left(h_1(i) - h_2(i)\right)^2}{h_1(i) + h_2(i)}$$

Jan Puzicha, Yossi Rubner, Carlo Tomasi, Joachim M. Buhmann: Empirical Evaluation of Dissimilarity Measures for Color and Texture. ICCV 1999

Linear classifiers

- Find linear function (*hyperplane*) to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Which hyperplane is best?

Slide: S. Lazebnik

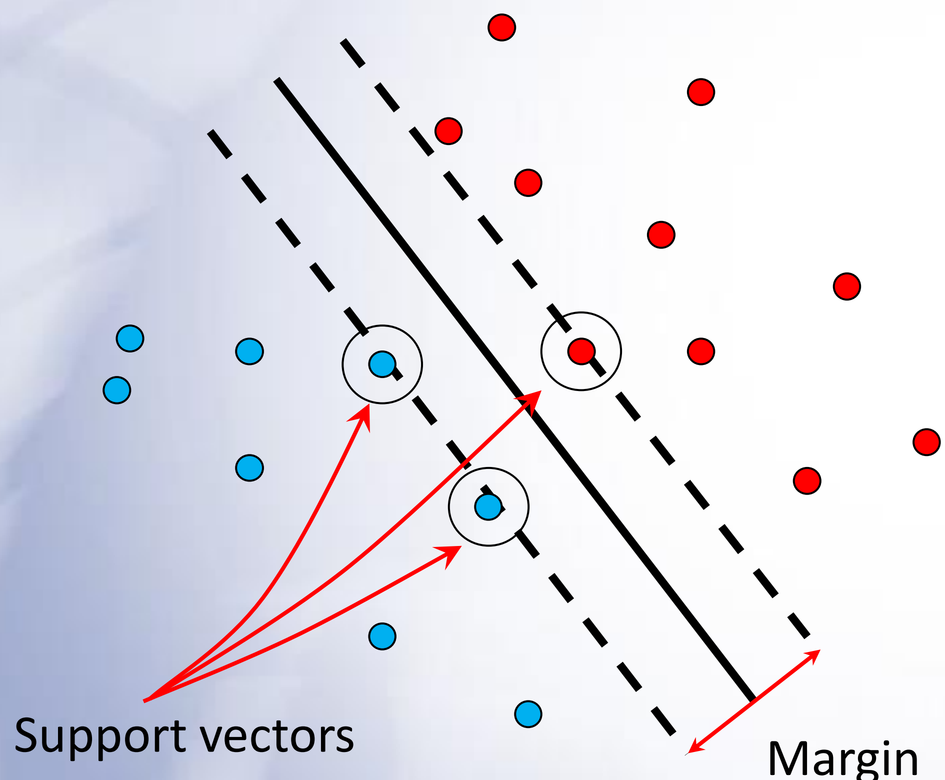
Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and hyperplane: $\quad \dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

- *Quadratic optimization problem*:
- Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

  Subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i+b) \geq 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $y_i(\mathbf{w}\cdot\boldsymbol{x}_i+b) \geq 1$
i=1，2，…，m.

使用拉格朗日乘数法

$$L(\omega, b, \alpha) = \frac{1}{2}\|\omega\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i(\omega^T x_i + b)\right)$$

首先固定α，以w和b为参数，求$L(\omega, b, \alpha)$的极小值，令偏导数为0

$$\frac{\partial L(\omega, b, \alpha)}{\partial \omega} = \omega - \sum_{i=1}^{m} \alpha_i\, y_i x_i = 0$$

$$\omega = \sum_{i=1}^{m} \alpha_i\, y_i x_i$$

Finding the maximum margin hyperplane

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

learned weight

Support vector

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin hyperplane

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ for any support vector

- Classification function (超平面):

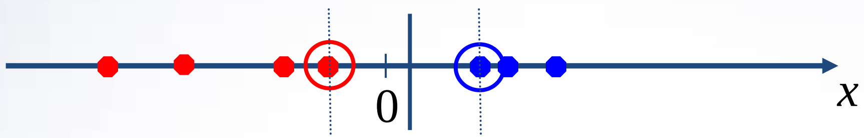
$$\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

- Notice that it relies on an *inner product* between the test point $\boldsymbol{x}$ and the support vectors $\boldsymbol{x}_i$
- Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points

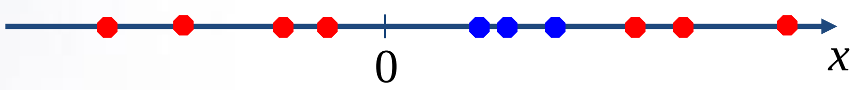
C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Nonlinear SVMs

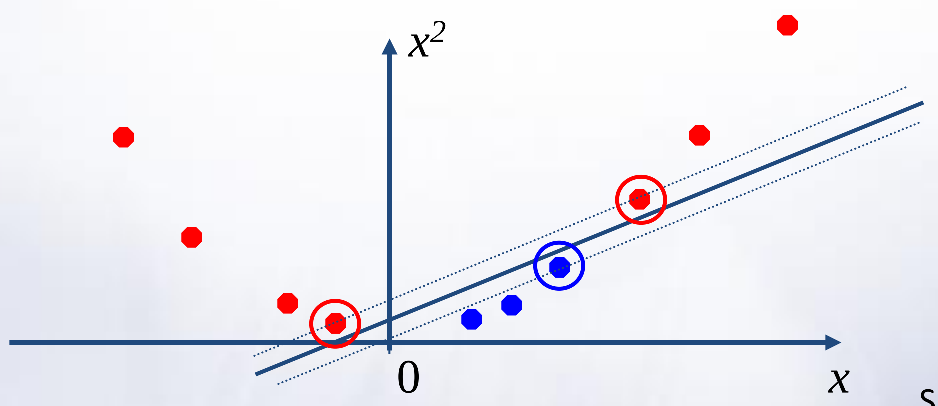
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



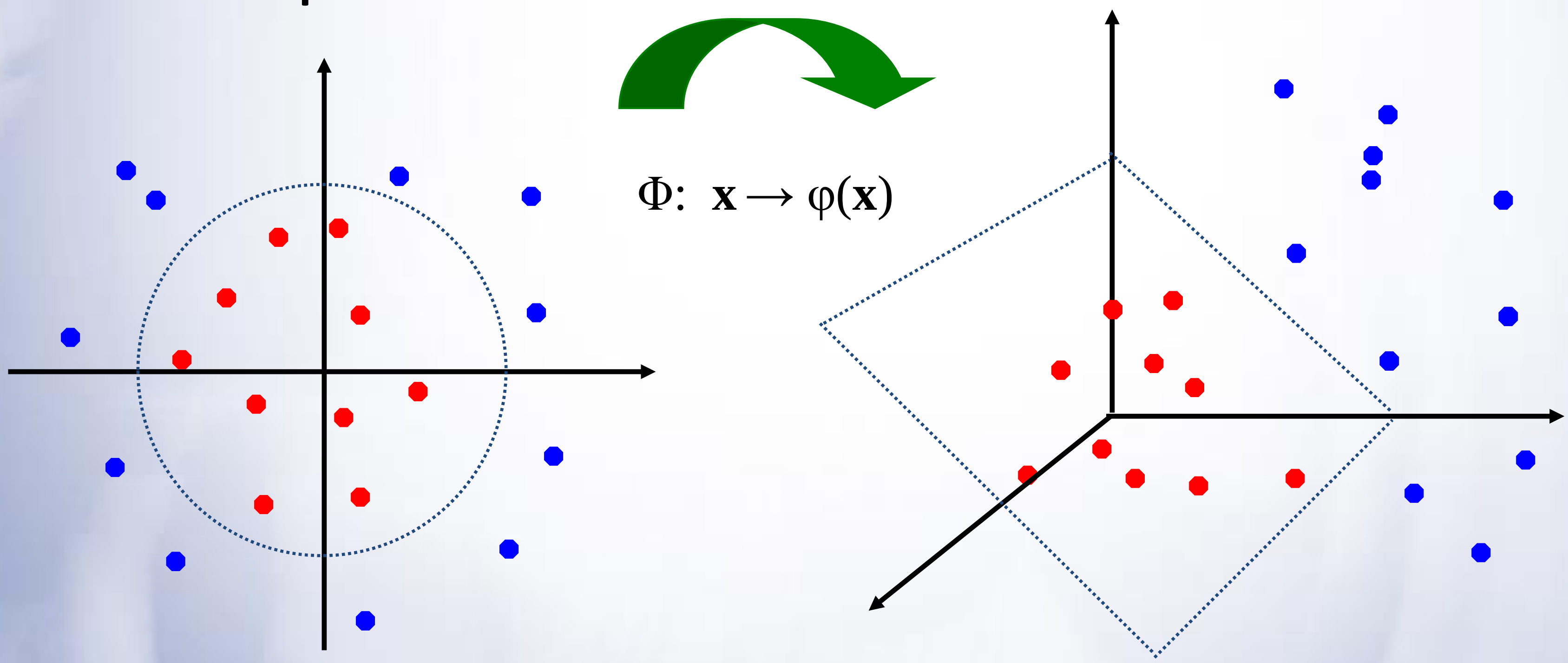
- We can map it to a higher-dimensional space:



Slide credit: Andrew Moore

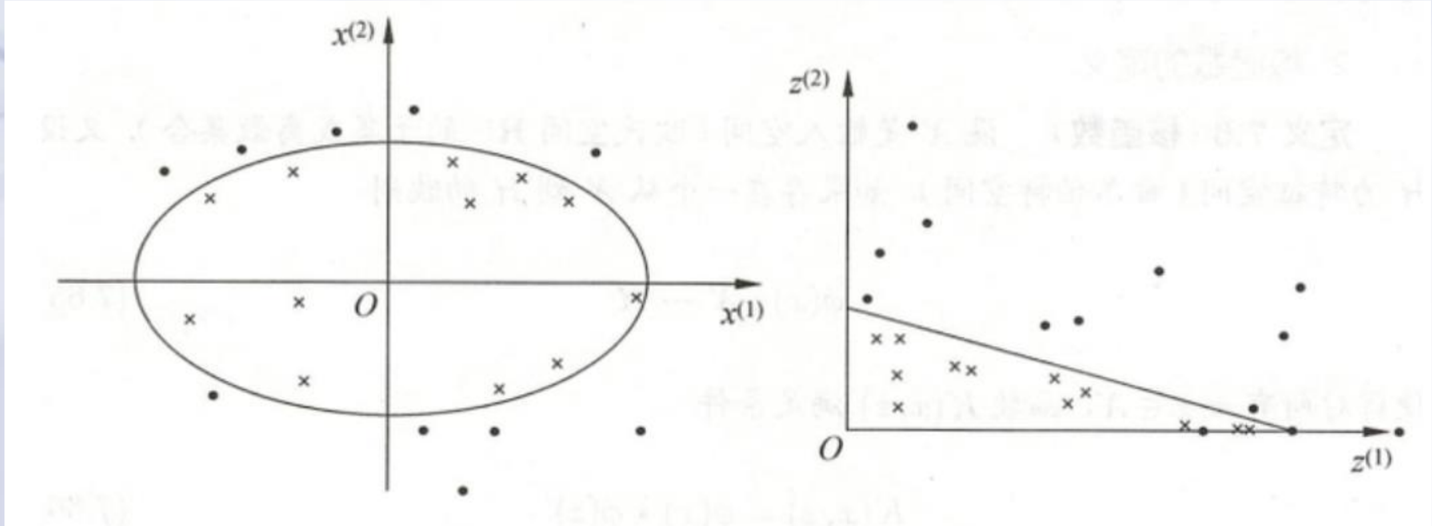
Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

Nonlinear SVMs



原空间 $x \subset \mathbf{R}^2,\ z = \left(x^{(1)},\ x^{(2)}\right)^{\mathrm{T}} \in X$

新空间 $z \subset \mathbf{R}^2,\ z = \left(z^{(1)},\ z^{(2)}\right)^{\mathrm{T}} \in Z$

原空间到新空间的映射为 $z = \phi(x) = \left(\left(x^{(1)}\right)^2,\ \left(x^{(2)}\right)^2\right)^{\mathrm{T}}$

经过变换，原空间 $X \subset \mathbf{R}^2$ 变换为新空间 $Z \subset \mathbf{R}^2$，原空间中的点相应地变换为新空间中的点。原空间中的椭圆可以表示为:

$$w_1 \left(x^{(1)}\right)^2 + w_2 \left(x^{(2)}\right)^2 + b = 0$$

变换成为新空间中的直线:

$$w_1 z^{(1)} + w_2 z^{(2)} + b = 0$$

Slide credit: Andrew Moore

Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i) \cdot \boldsymbol{\varphi}(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b \longrightarrow \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Kernels for bags of features

- Histogram intersection kernel:

$$I(h_1, h_2) = \sum_{i=1}^{N} \min(h_1(i), h_2(i))$$

- Generalized Gaussian kernel:

$$K(h_1, h_2) = \exp\left(-\frac{1}{A} D(h_1, h_2)^2\right)$$

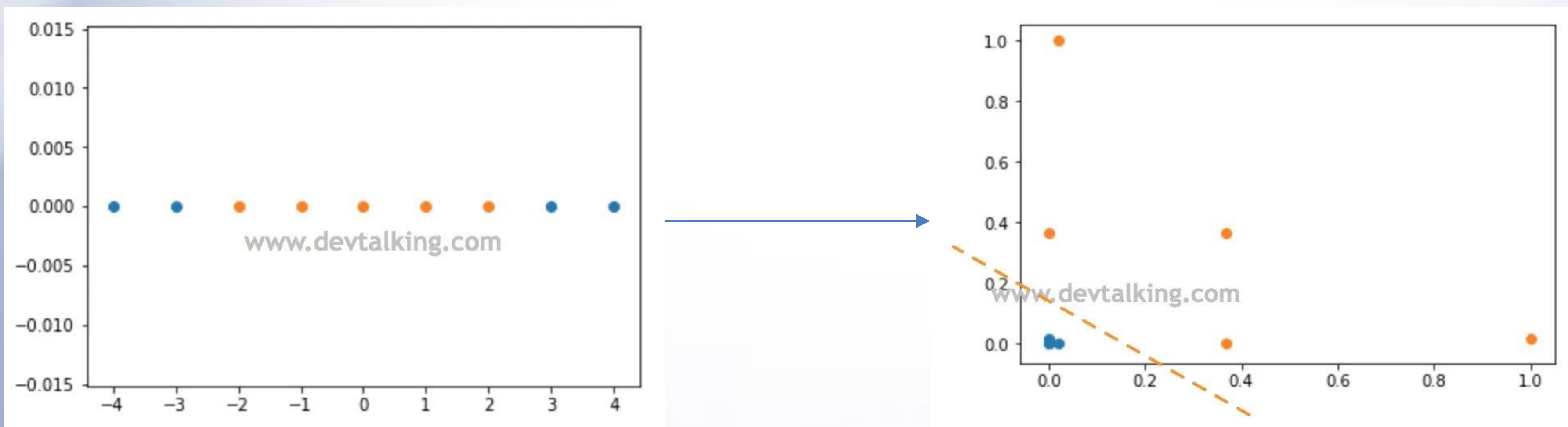
- *D* can be Euclidean distance, χ^2 distance, Earth Mover's Distance, etc.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, Local Features and Kernels for Classifcation of Texture and Object Categories: A Comprehensive Study, IJCV 2007

Kernels for bags of features

- Generalized Gaussian kernel:

$$K(h_1, h_2) = \exp\left(-\frac{1}{A} D(h_1, h_2)^2\right)$$



可以看到通过高斯函数将原本的一维样本数据转换为
二维后，新样本数据明显成为线性可分的状态。

Summary: SVMs for image classification

1. Pick an image representation (in our case, bag of features)
2. Pick a kernel function for that representation
3. Compute the matrix of kernel values between every pair of training examples
4. Feed the kernel matrix into your favorite SVM solver to obtain support vectors and weights
5. At test time: compute kernel values for your test example and each support vector, and combine them with the learned weights to get the value of the decision function

SVMs: Pros and cons

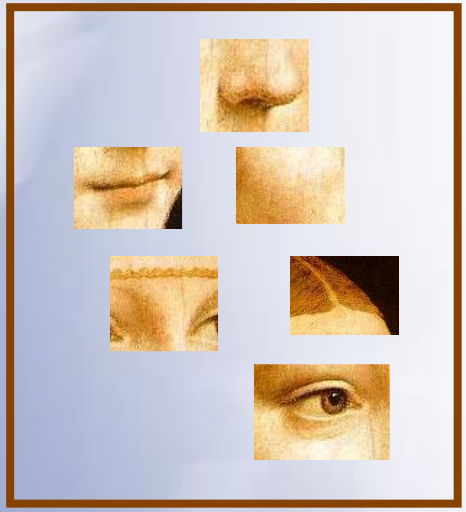
- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
  - Kernel-based framework is very powerful, flexible
  - SVMs work very well in practice, even with very small training sample sizes

- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Slide: S. Lazebnik

Generative learning methods for bags of features

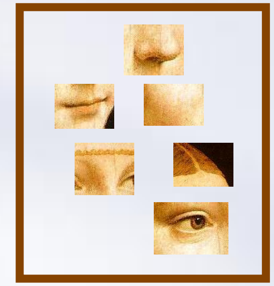
- Model the probability of a bag of features given a class



Generative methods

- We will cover two models, both inspired by text document analysis:
  - Naïve Bayes
  - Probabilistic Latent Semantic Analysis

The Naïve Bayes model



- Assume that each feature is conditionally independent *given the class*

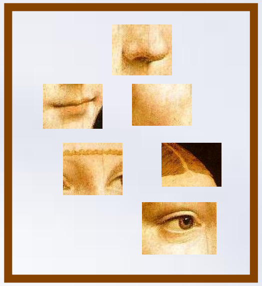
$$p(f_1,\ldots,f_N \mid c) = \prod_{i=1}^{N} p(f_i \mid c) = \prod_{j=1}^{M} p(w_j \mid c)^{n(w_j)}$$

f_i: ith feature in the image
N: number of features in the image

w_j: jth visual word in the vocabulary
M: size of visual vocabulary
$n(w_j)$: number of features of type w_j in the image

Csurka et al. 2004

The Naïve Bayes model



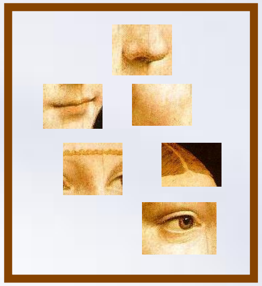
- Assume that each feature is conditionally independent *given the class*

$$p(f_1,\ldots,f_N \mid c)=\prod_{i=1}^{N} p(f_i \mid c)=\prod_{j=1}^{M} p(w_j \mid c)^{n(w_j)}$$

$$P(w_j|c)=\frac{\text{类c下特征类}w_j\text{出现的次数和}}{\text{类c下特征总和}}$$

Csurka et al. 2004

The Naïve Bayes model



- Assume that each feature is conditionally independent *given the class*

有些情况下，类c下某特征wj出现次数为0，导致类条件概率为0。
这样不合适，不能因为一个特征不出现就否定了其他所有特征。

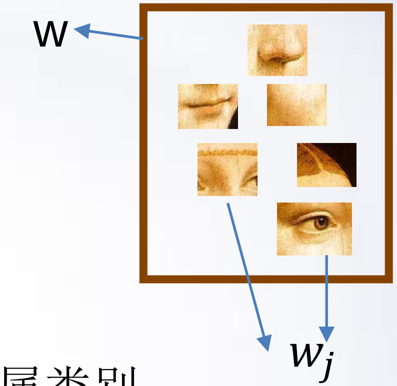
$$P(w_j|c) = \frac{\text{类c下特征类}w_j\text{出现的次数和} + 1}{\text{类c下特征总和} + M}$$

M表示训练样本包含多少种不同特征单词，即特征总数

为了解决零概率的问题，法国数学家拉普拉斯最早提出用加1的方法估计没有出现过的现象的概率，所以加法平滑也叫做拉普拉斯平滑。

The Naïve Bayes model

- Maximum A Posteriori decision:



$$欲求c^* = arg\max_c P(c|w) = arg\max_c \frac{P(c)P(w|c)}{P(w)}$$

$P(w)$可计算，给定训练样本后为固定常数

所以需要计算分子最大值对应的类别c

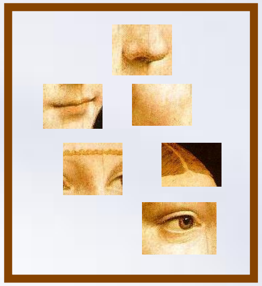
即为一张图片w(其中通过词袋模型建立特征词汇表)的所属类别

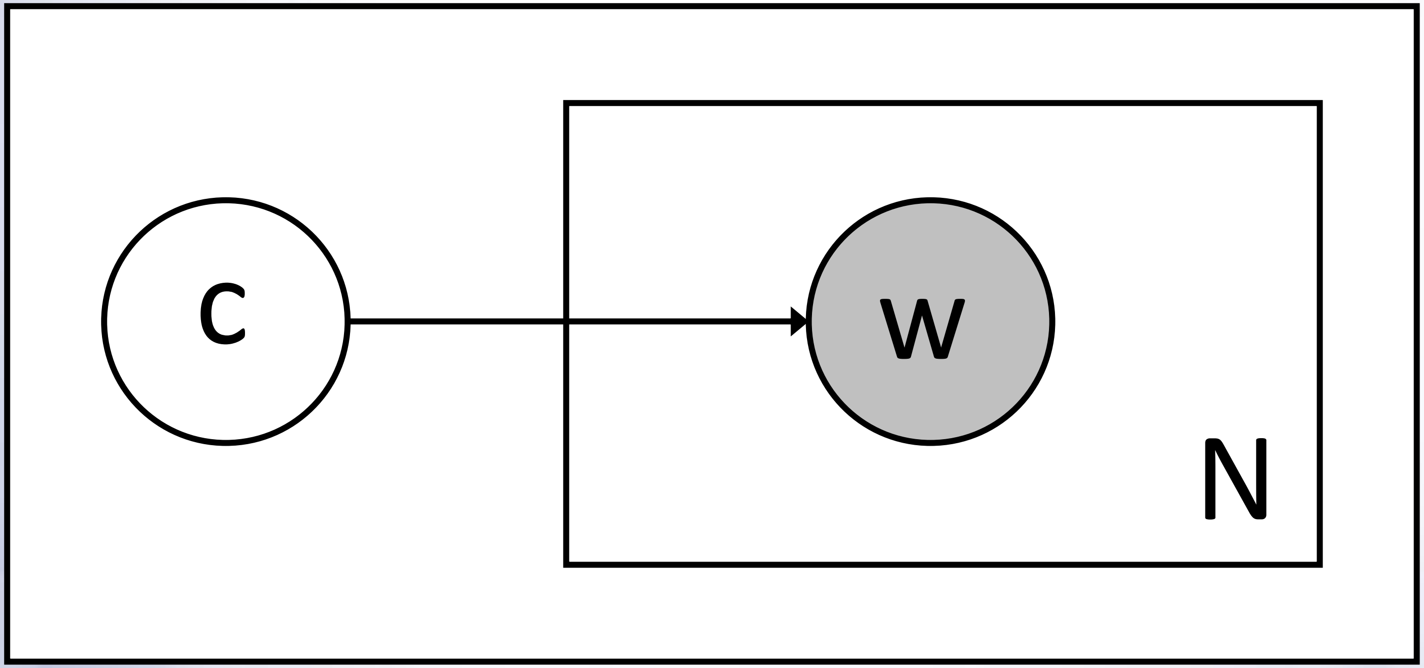
$$c^* = \arg\max_c \; p(c)\prod_{j=1}^{M} p(w_j \mid c)^{n(w_j)}$$

$$= \arg\max_c \; \log p(c) + \sum_{j=1}^{M} n(w_j) \log p(w_j \mid c)$$

Csurka et al. 2004

The Naïve Bayes model





由此，根据朴素贝叶斯公式，可以建立类别c与特征w的关系式。
即已知视觉单词w求类别c。

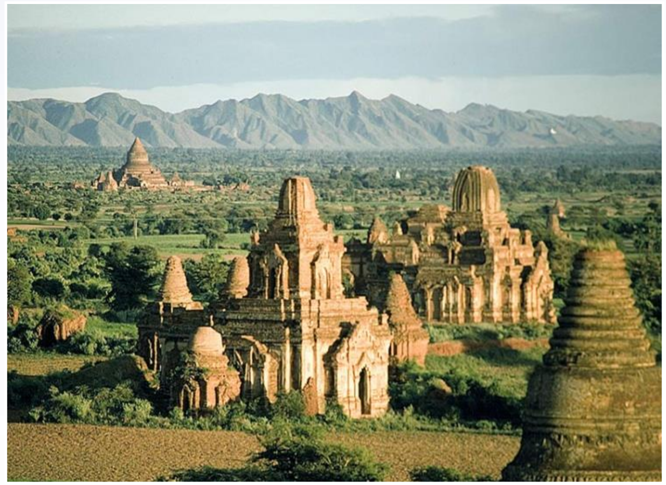
Csurka et al. 2004

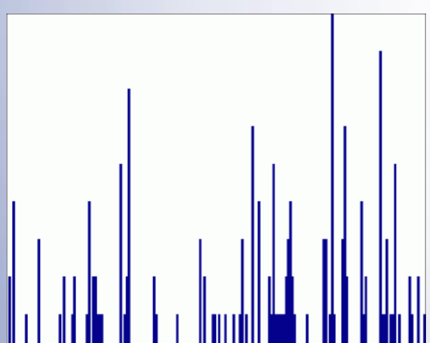
Adding spatial information

- Computing bags of features on sub-windows of the whole image
- Using codebooks to vote for object position
- Generative part-based models

Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution





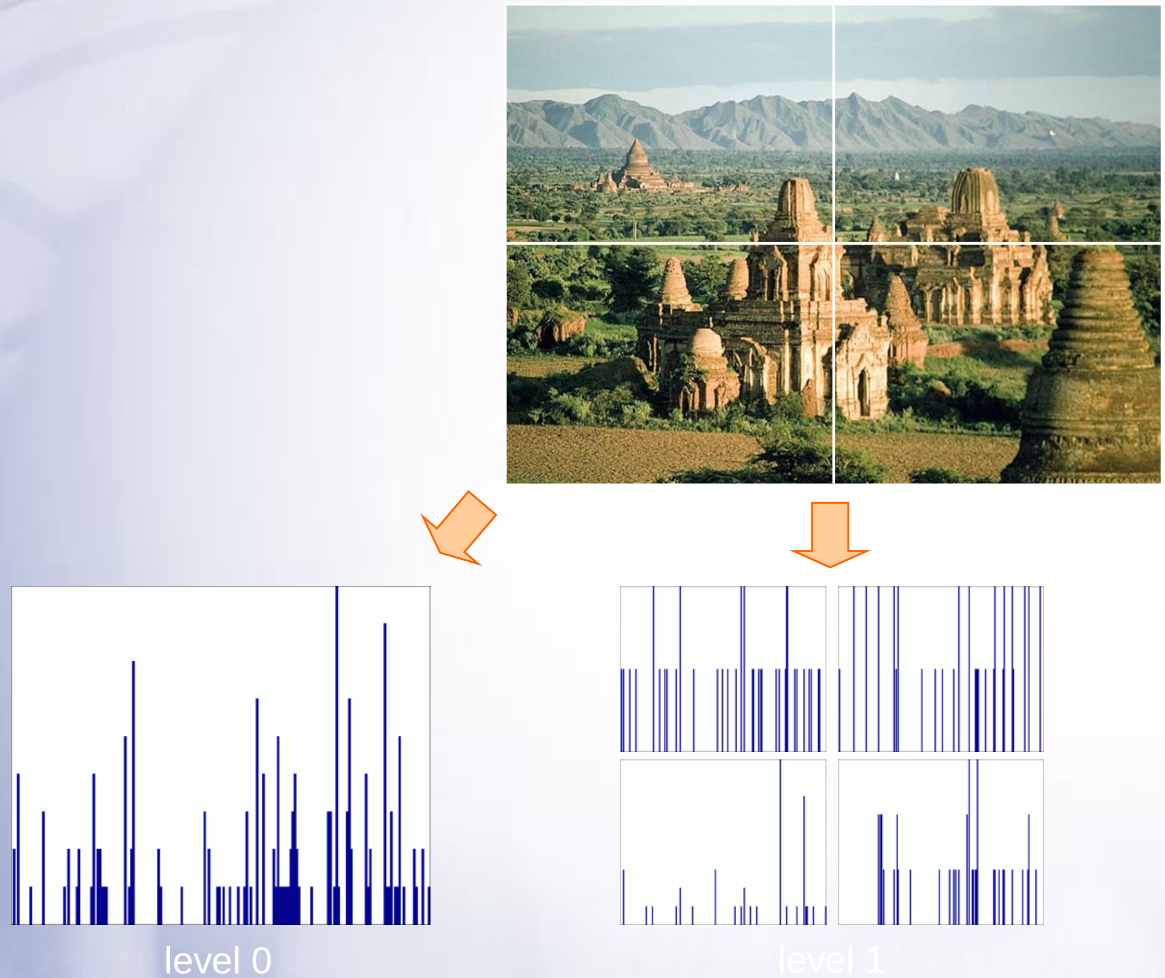
level 0

Lazebnik, Schmid & Ponce (CVPR 2006)

Slide: S. Lazebnik

Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Lazebnik, Schmid & Ponce (CVPR 2006)

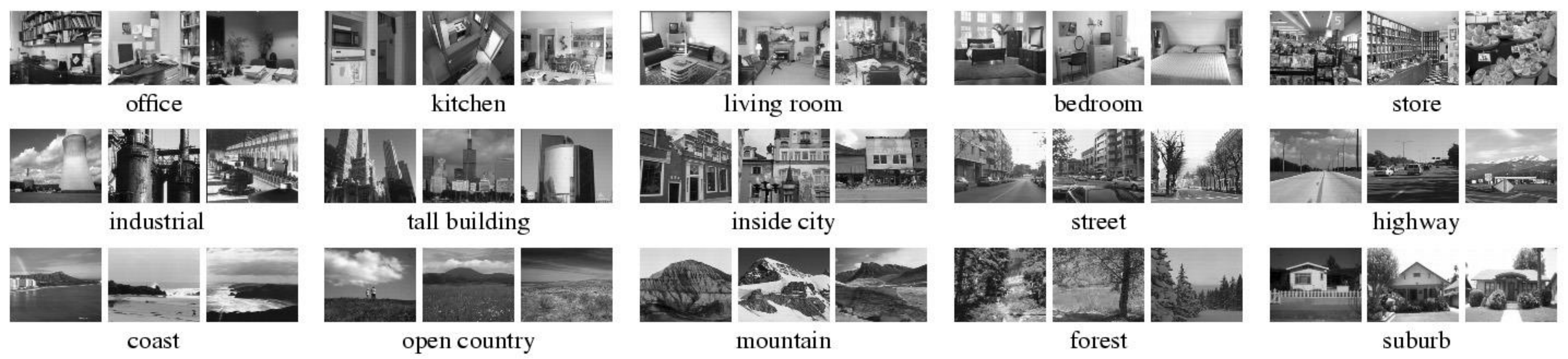
Slide: S. Lazebnik

Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)

Slide: S. Lazebnik



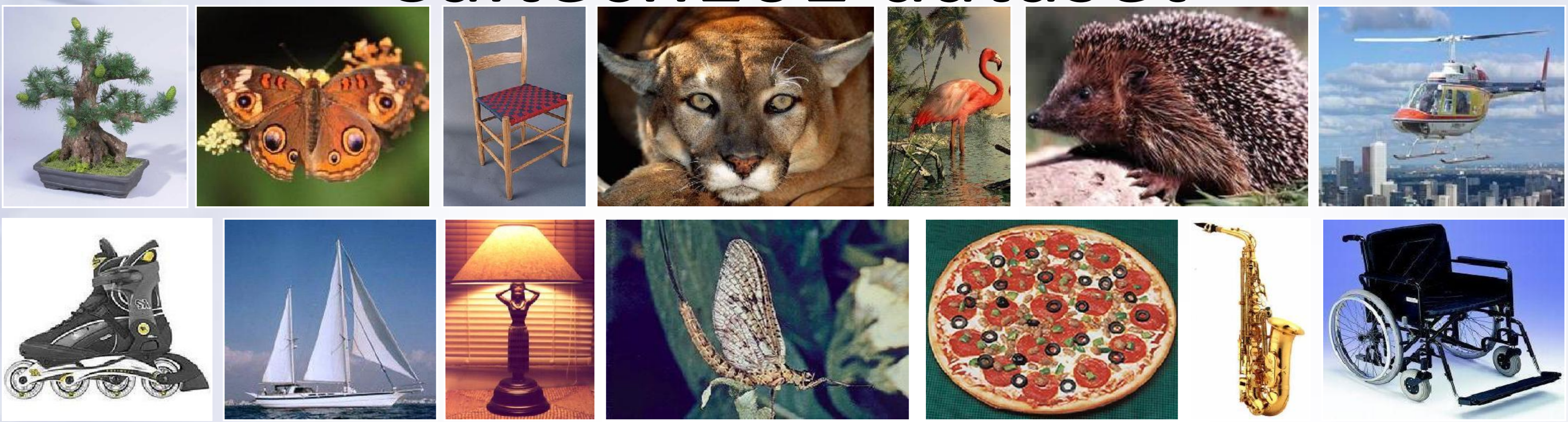
Multi-class classification results
(100 training images per class)

Level	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
	Single-level	Pyramid	Single-level	Pyramid
0 (1×1)	45.3 ±0.5		72.2 ±0.6	
1 (2×2)	53.6 ±0.3	56.2 ±0.6	77.9 ±0.6	79.0 ±0.5
2 (4×4)	61.7 ±0.6	64.7 ±0.7	79.4 ±0.3	**81.1** ±0.3
3 (8×8)	63.3 ±0.8	**66.8** ±0.6	77.2 ±0.4	80.7 ±0.3

Slide: S. Lazebnik

http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html

Caltech101 dataset



Multi-class classification results (30 training images per class)

	Weak features (16)		Strong features (200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7

Slide: S. Lazebnik

作业:

1. 编程实现Bag-of-Feature算法