CS688/WST665: Web-Scale Image Retrieval

Bag-of-Words (BoW) Models

Sung-Eui Yoon
(윤성의)

Course URL:
http://sglab.kaist.ac.kr/~sungeui/IR

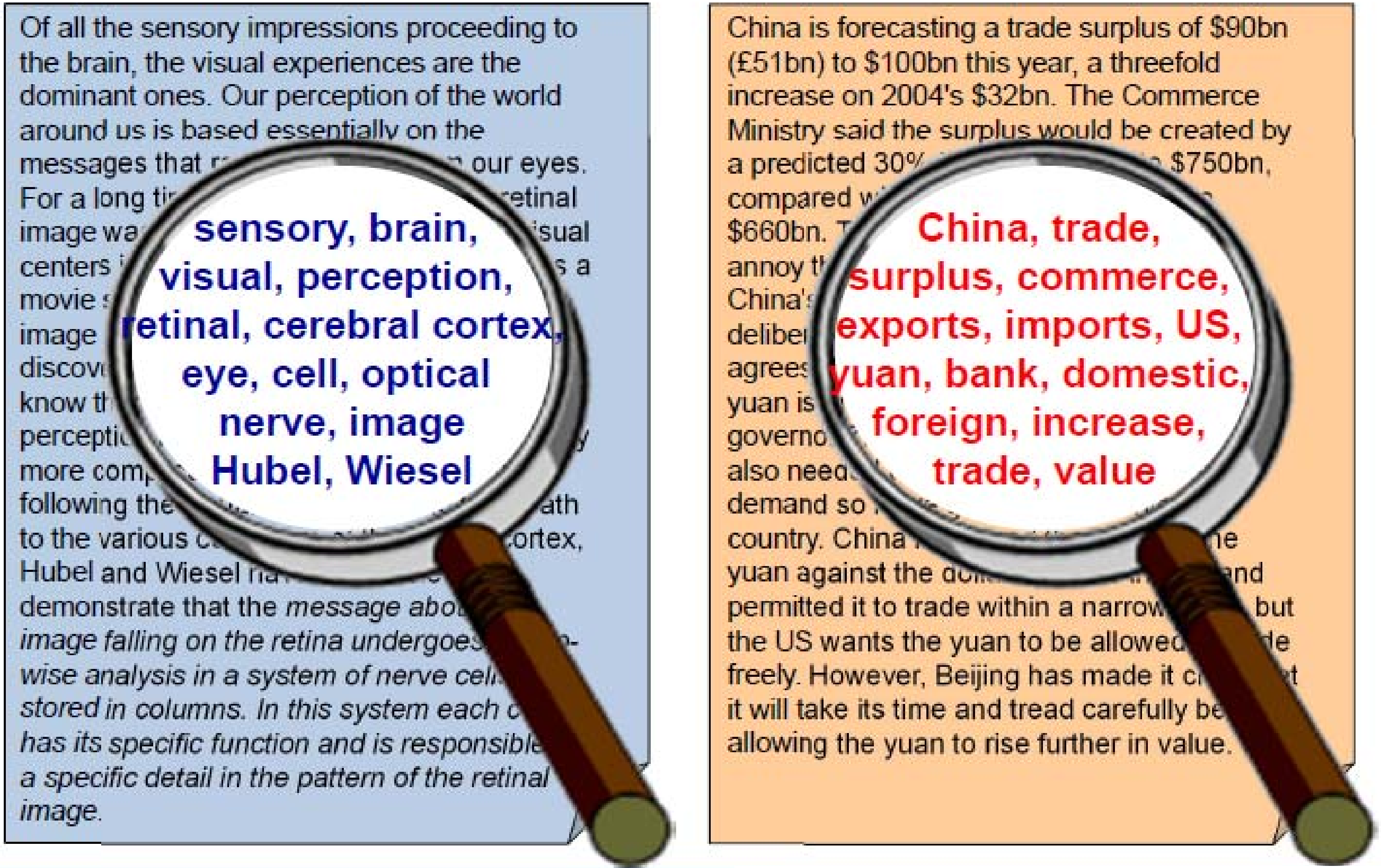
KAIST

What we will learn today?

- Bag of Words models
  - Basic representation
  - Different learning and recognition algorithms

Object → **Bag of ‘words’**

Analogy to documents

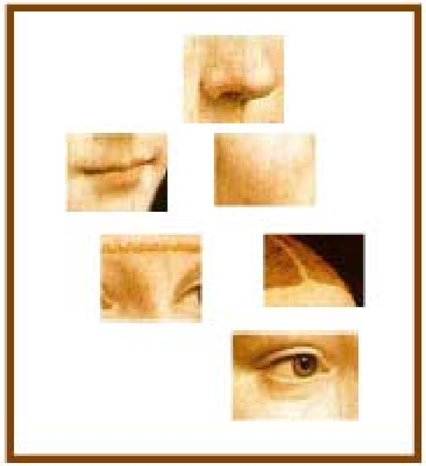


Fei-Fei Li

definition of "BoW"

– Independent features

face

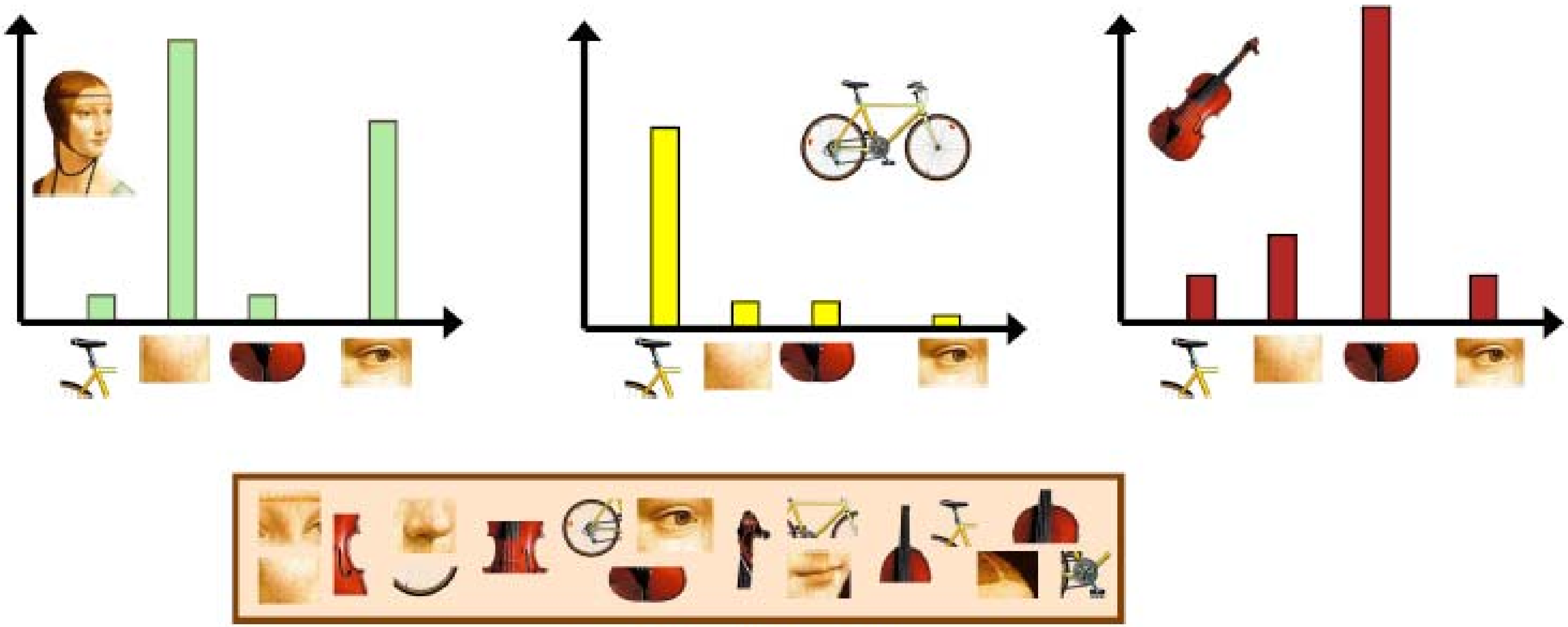


bike

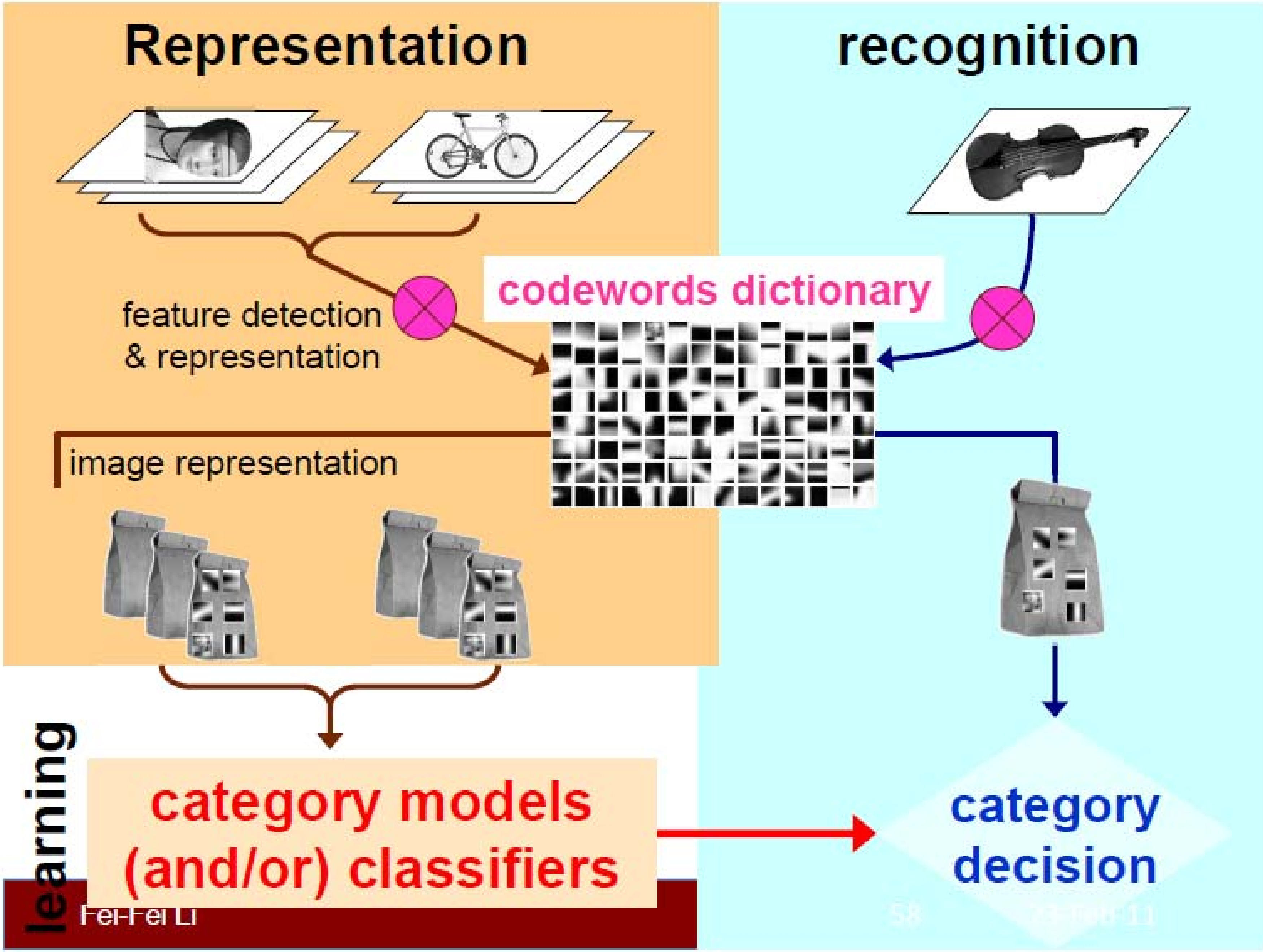
violin

definition of "BoW"

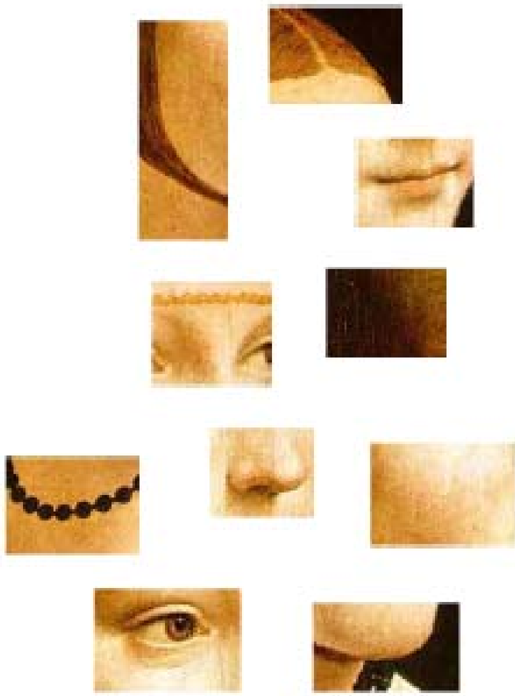
- Independent features
- histogram representation



codewords dictionary


Representation
recognition
codewords dictionary
feature detection
& representation
image representation
learning
category models
(and/or) classifiers
category
decision
Fei-Fei Li
58
23-Feb-11

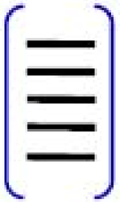
1.Feature detection and representation



1.Feature detection and representation

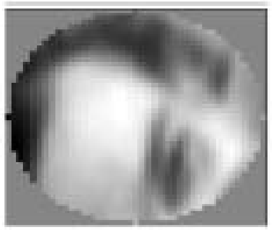
- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
- Interest point detector
  - Csurka, Bray, Dance & Fan, 2004
  - Fei-Fei & Perona, 2005
  - Sivic, Russell, Efros, Freeman & Zisserman, 2005
- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation based patches (Barnard, Duygulu, Forsyth, de Freitas, Blei, Jordan, 2003)

1.Feature detection and representation

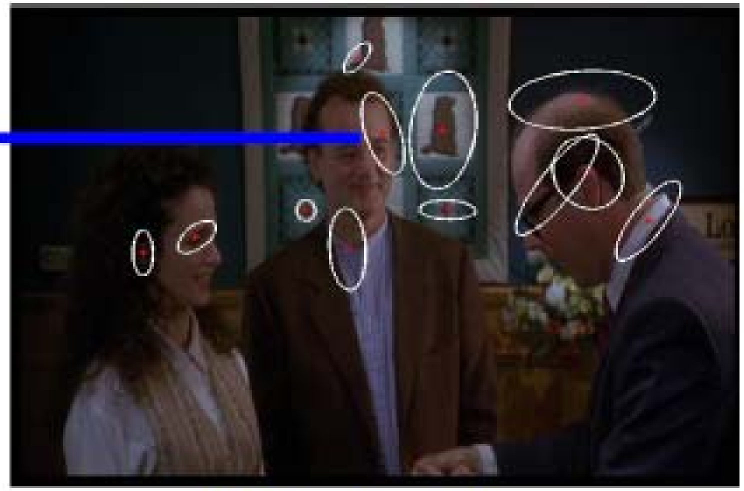


Compute SIFT descriptor

[Lowe'99]



Normalize patch



Detect patches

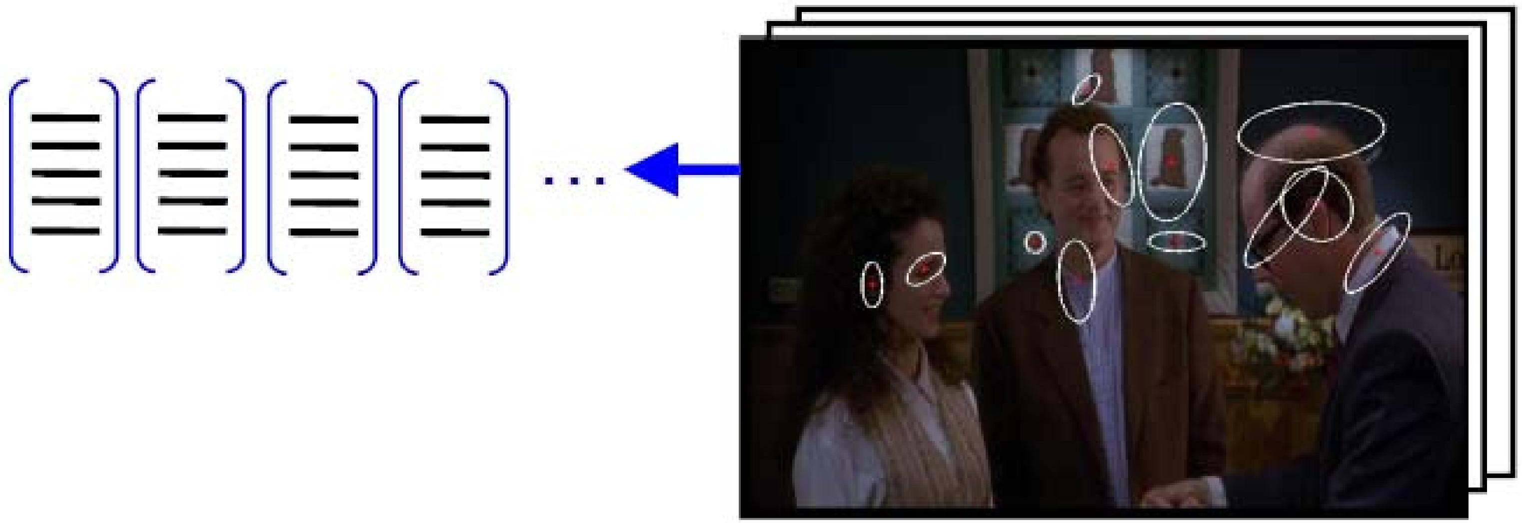
[Mikojaczyk and Schmid '02]

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

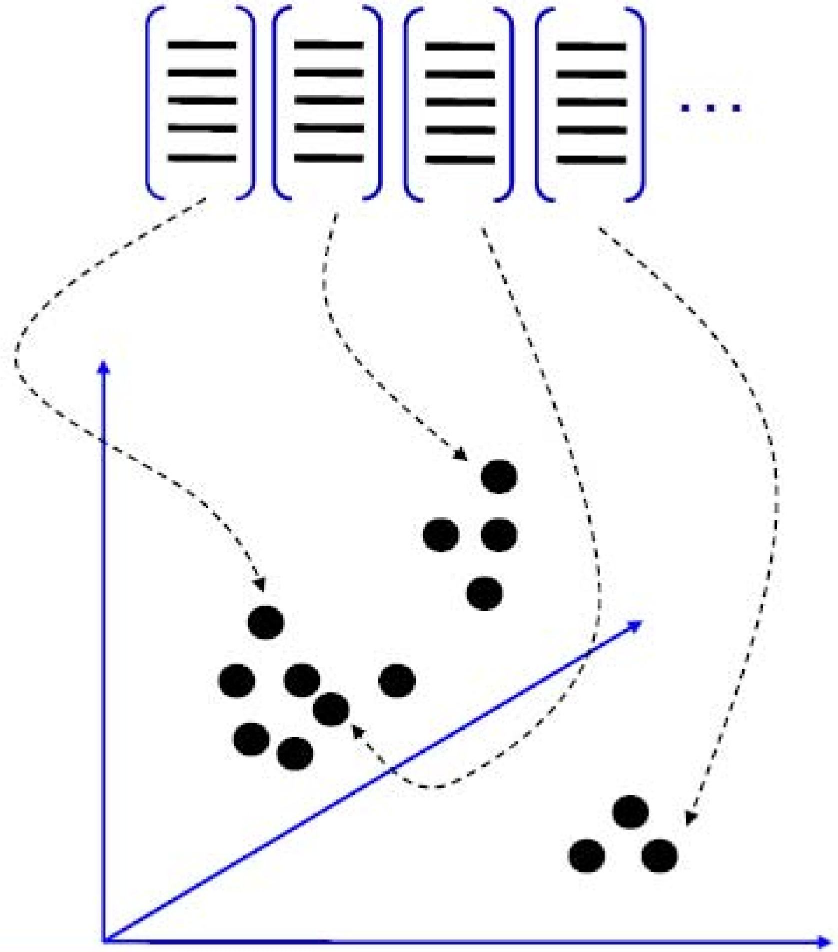
[Sivic & Zisserman, '03]

Slide credit: Josef Sivic

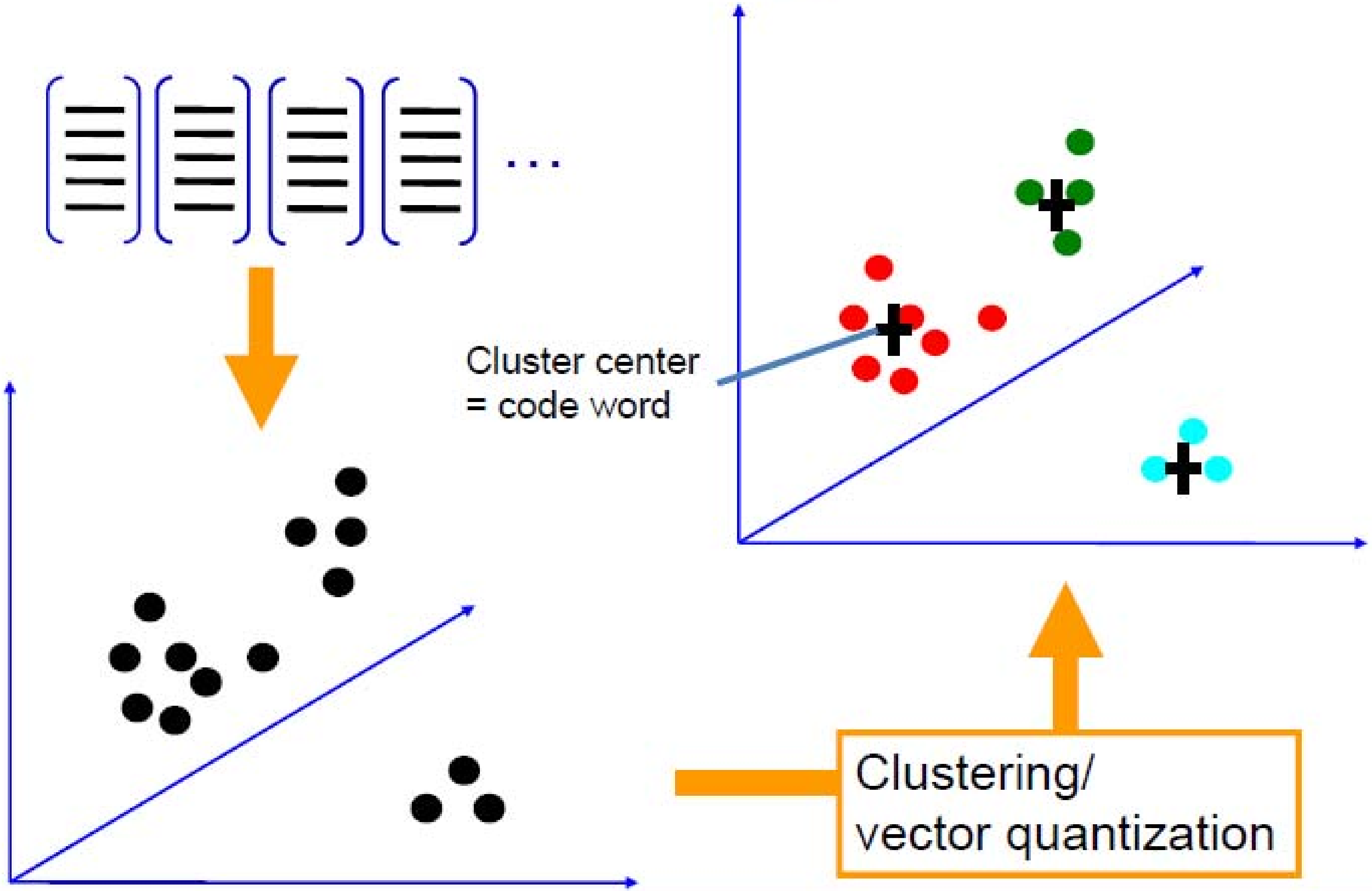
1.Feature detection and representation



2. Codewords dictionary formation



2. Codewords dictionary formation

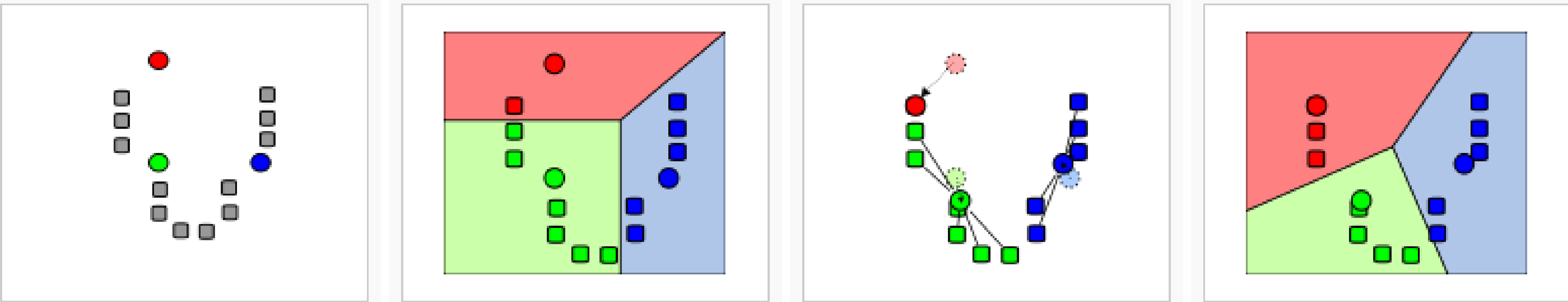


K-Means Clustering

- **Minimizing the within-cluster sum of squares (WCSS)**

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

Demonstration of the standard algorithm



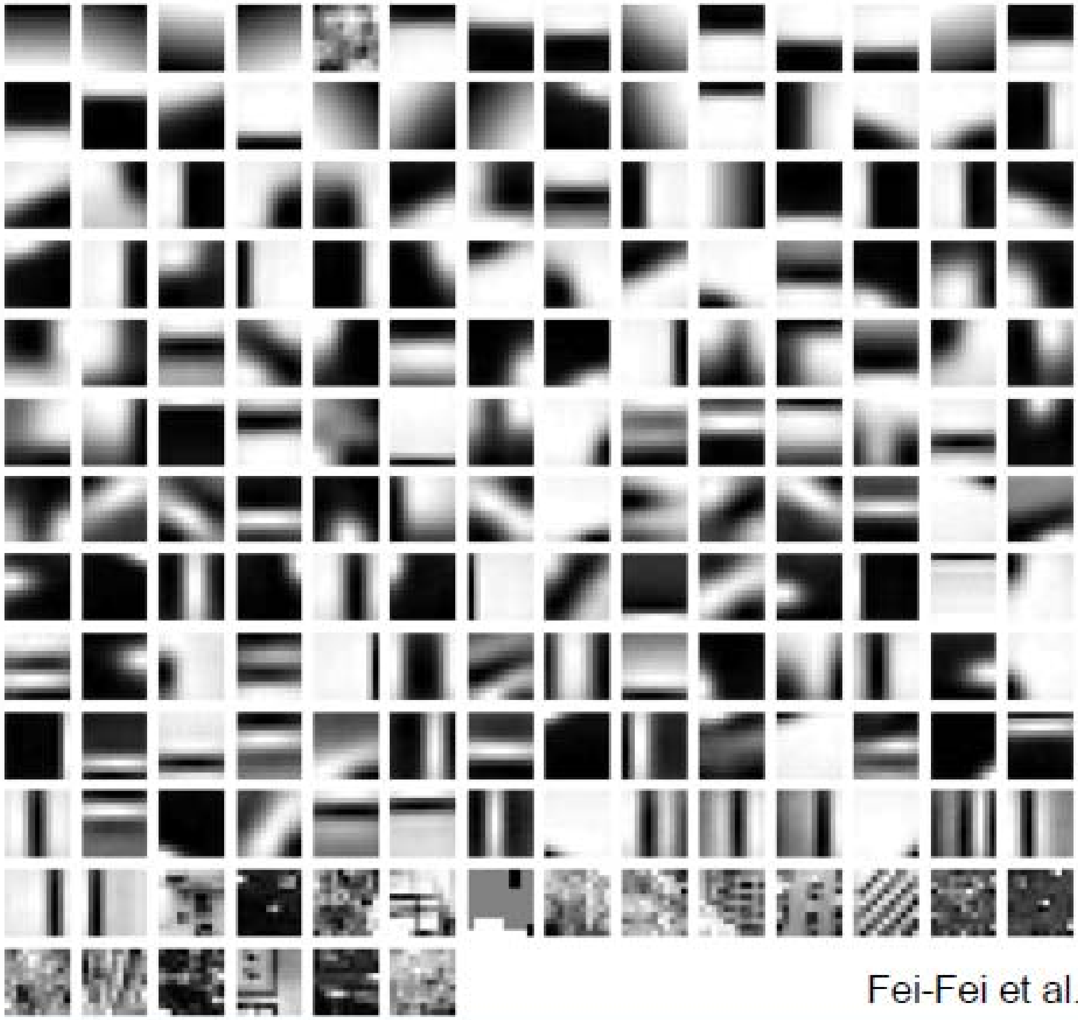
1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).

2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the *k* clusters becomes the new means.

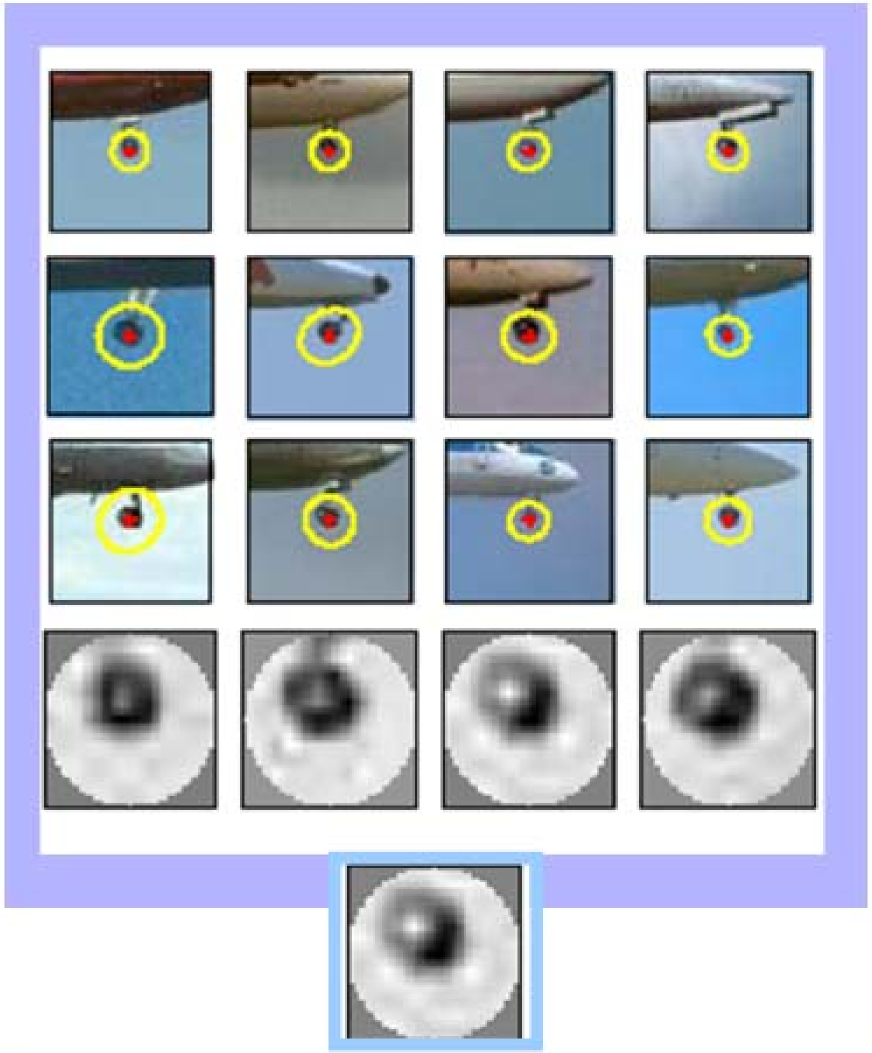
4) Steps 2 and 3 are repeated until convergence has been reached.

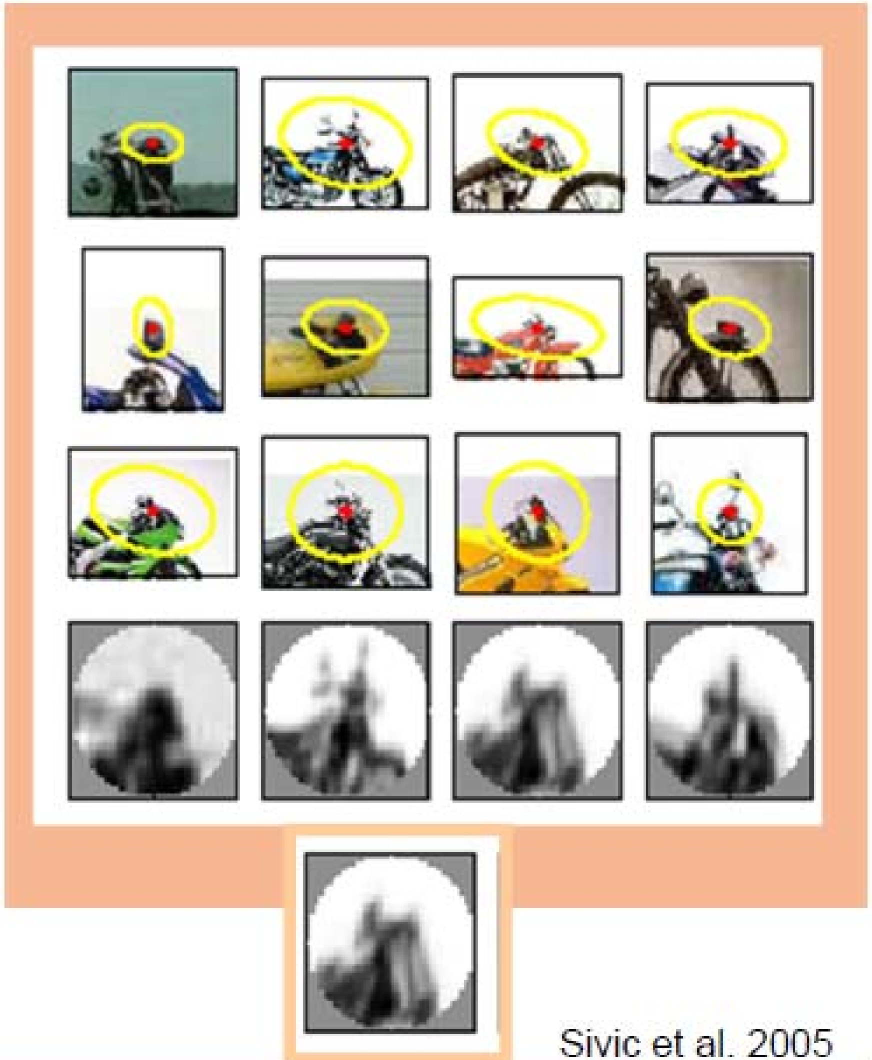
2. Codewords dictionary formation



Fei-Fei et al. 2005

Image patch examples of codewords



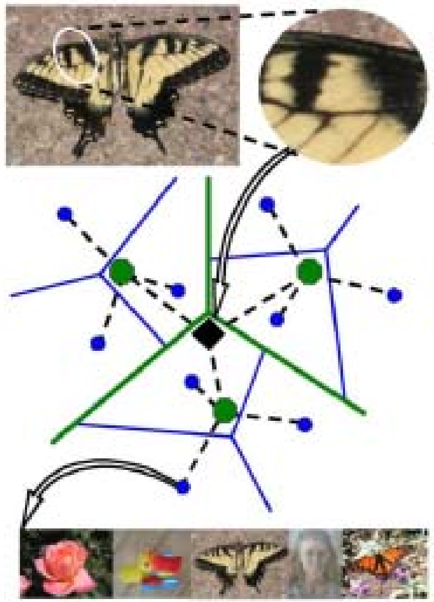


Sivic et al. 2005

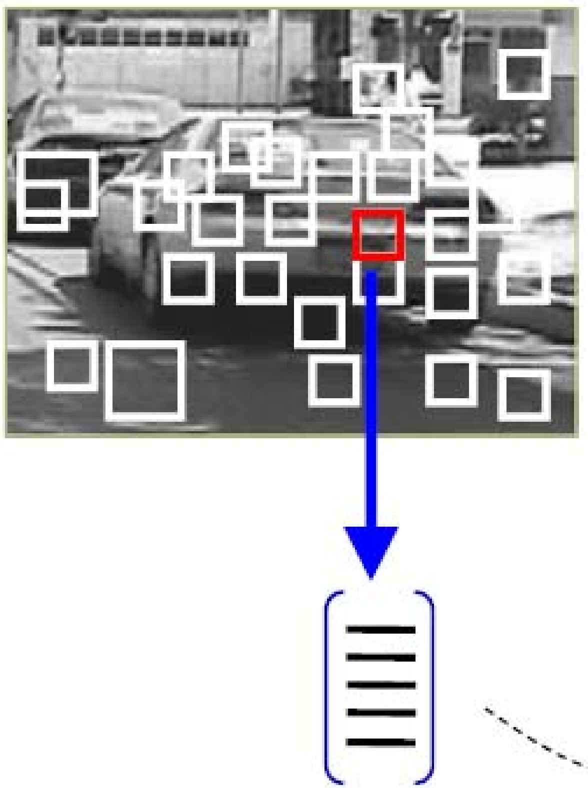
Visual vocabularies: Issues

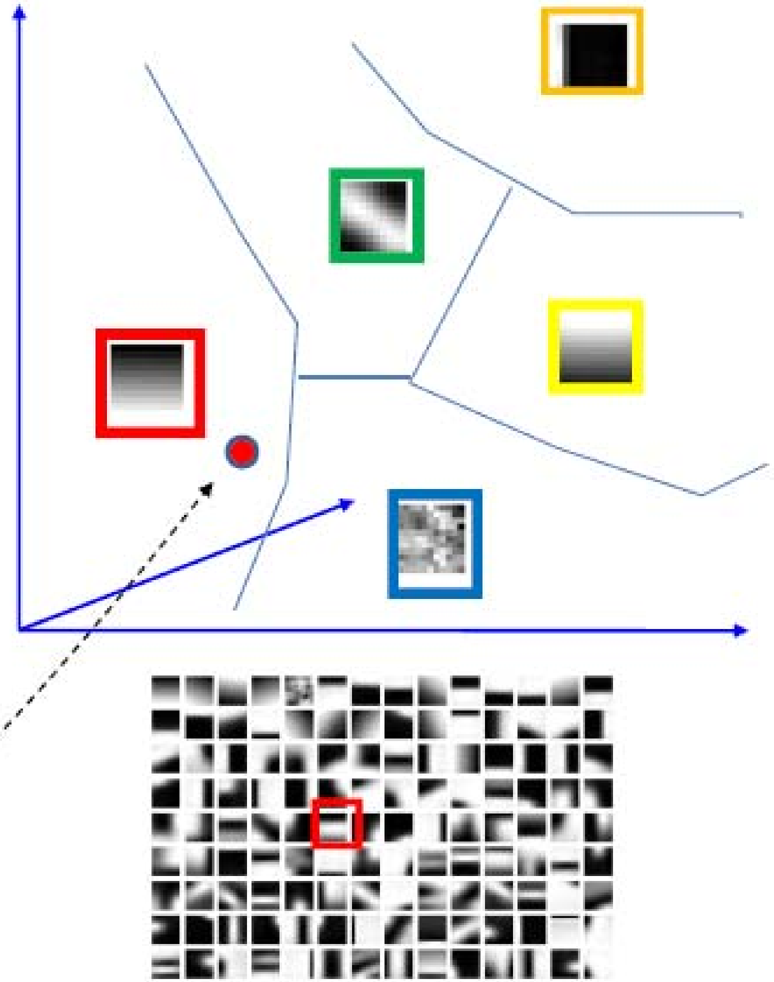
- How to choose vocabulary size?
  - Too small: visual words not representative of all patches
  - Too large: quantization artifacts, overfitting

- Computational efficiency
  - Vocabulary trees
    (Nister & Stewenius, 2006)



3. Bag of word representation

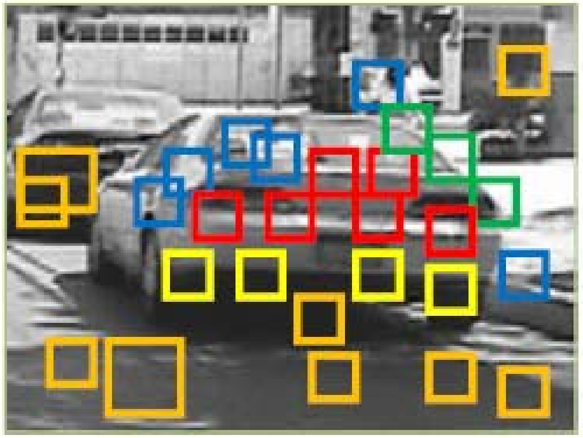


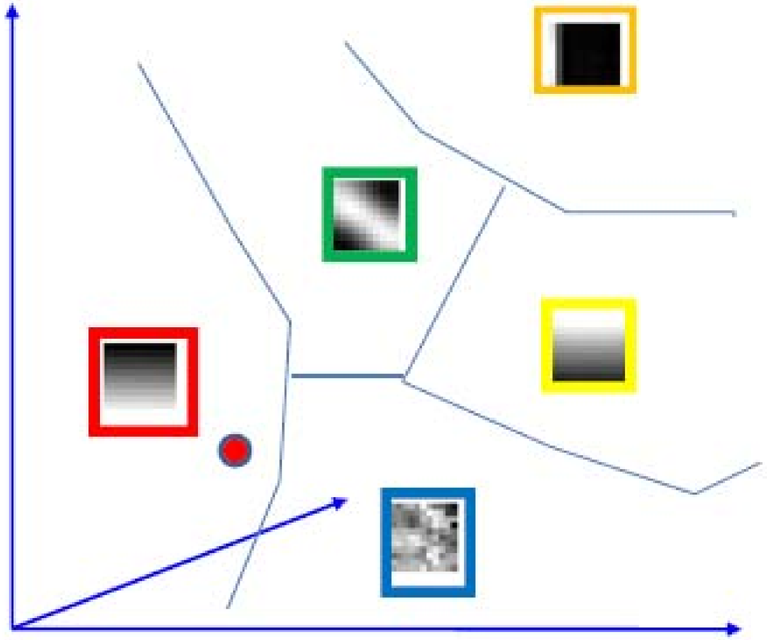


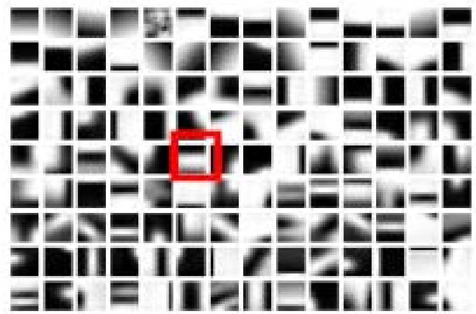
- Nearest neighbors assignment
- K-D tree search strategy

Codewords dictionary

3. Bag of word representation

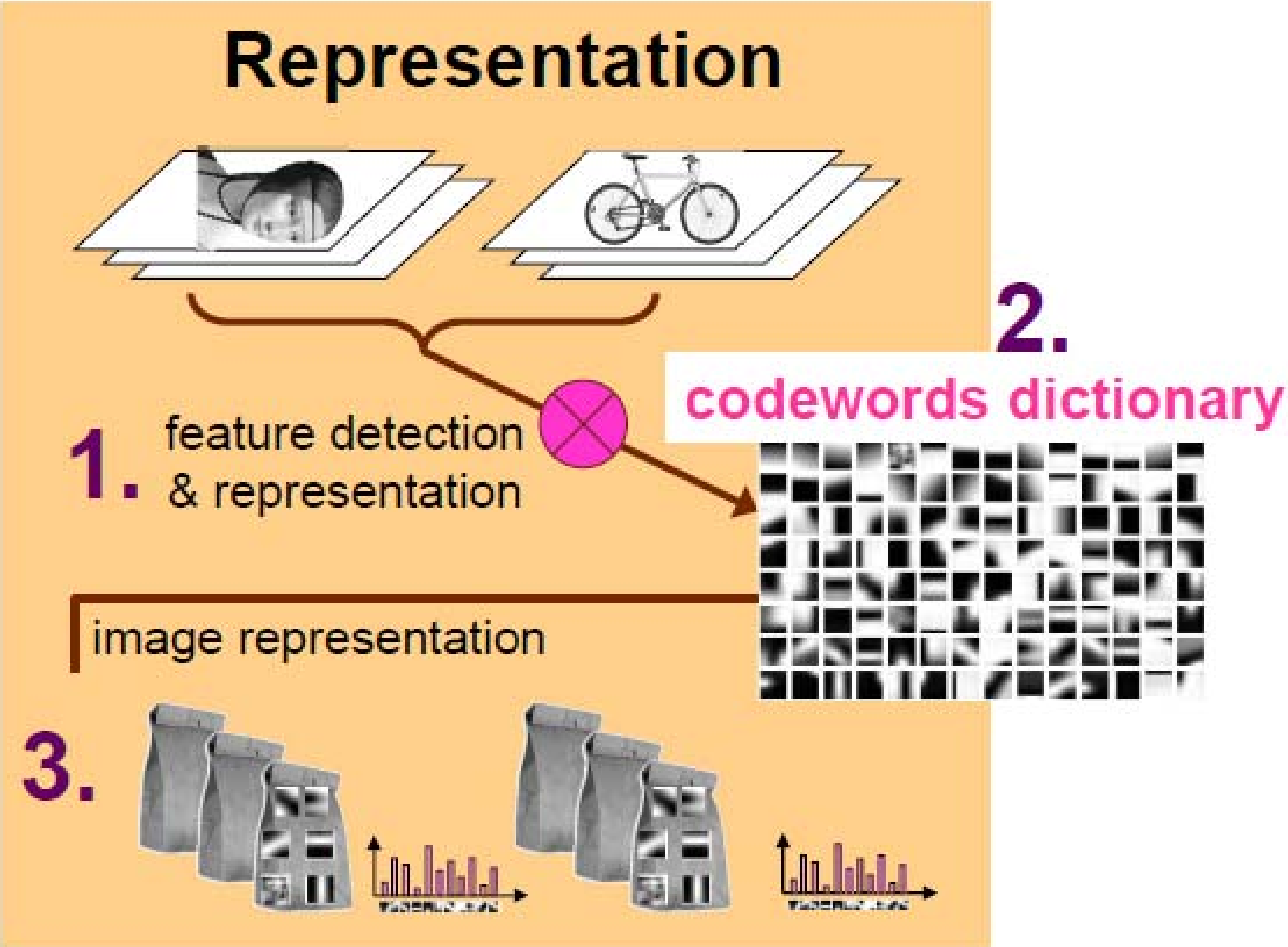




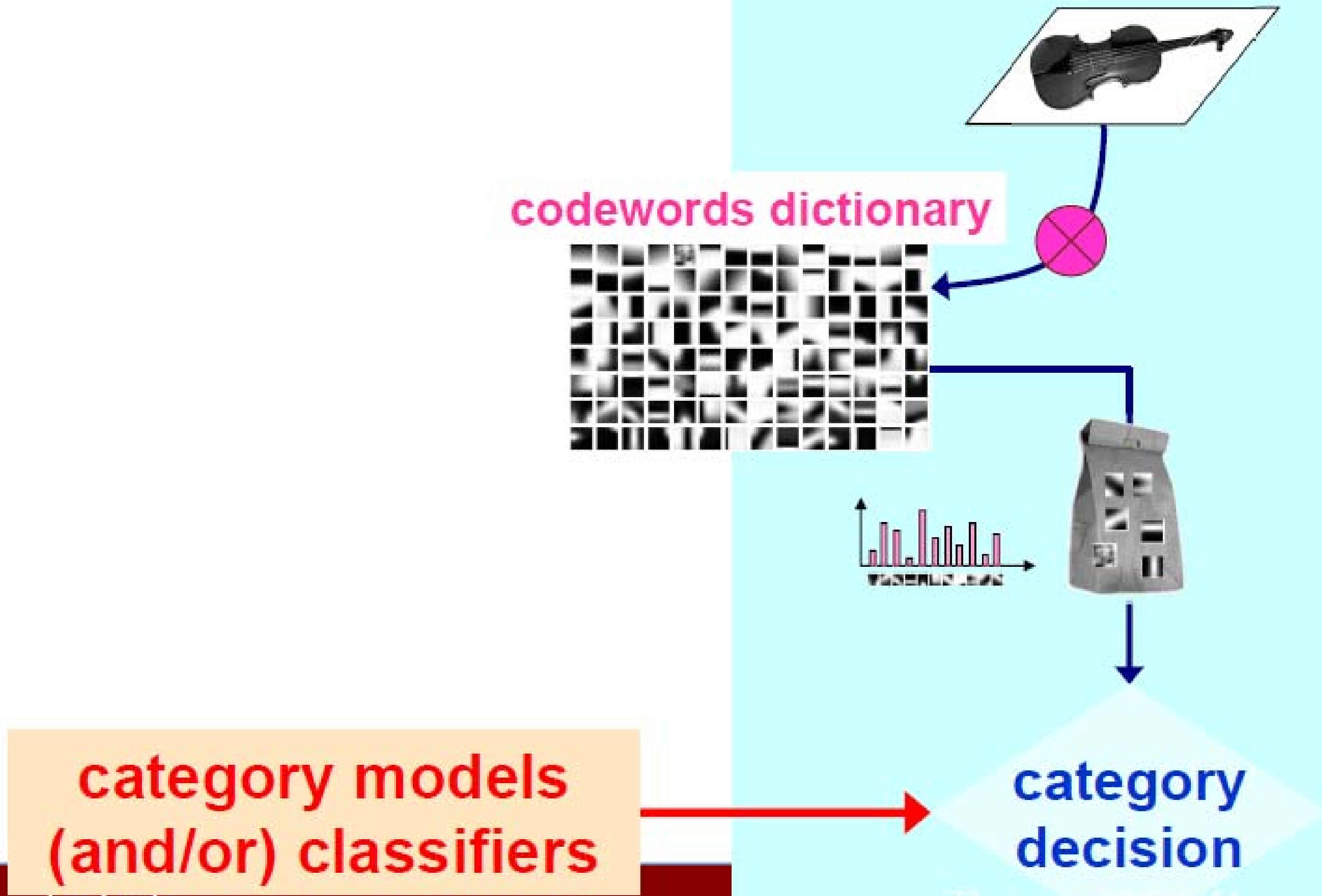


Codewords dictionary

Fei-Fei Li

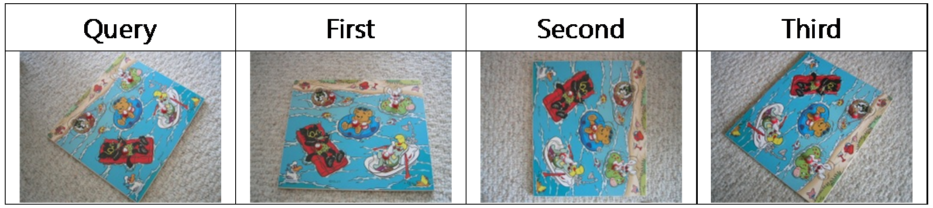

Representation
2.
codewords dictionary
1. feature detection
& representation
image representation
3.

Learning and Recognition



PA2

- Understand and implement a basic image retrieval system
- Use the original UKBenchmark



Learning and Recognition

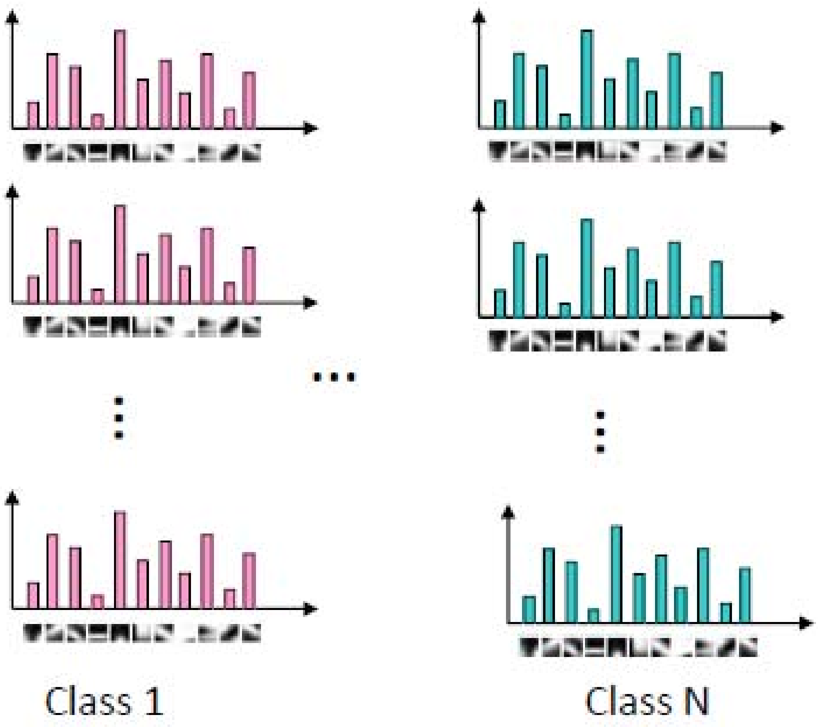
1. Discriminative method:
   - NN
   - SVM

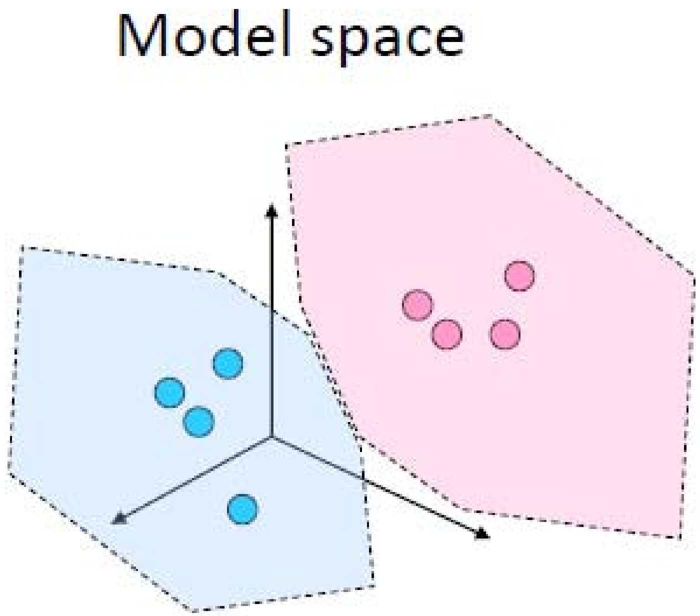
2. Generative method:
   - graphical models

category models (and/or) classifiers

Discriminative classifiers

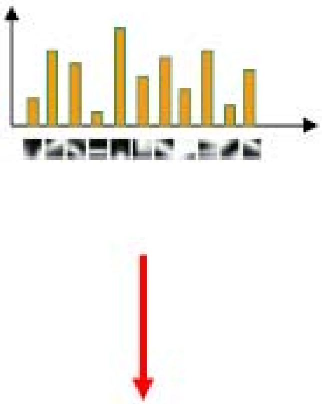
category models





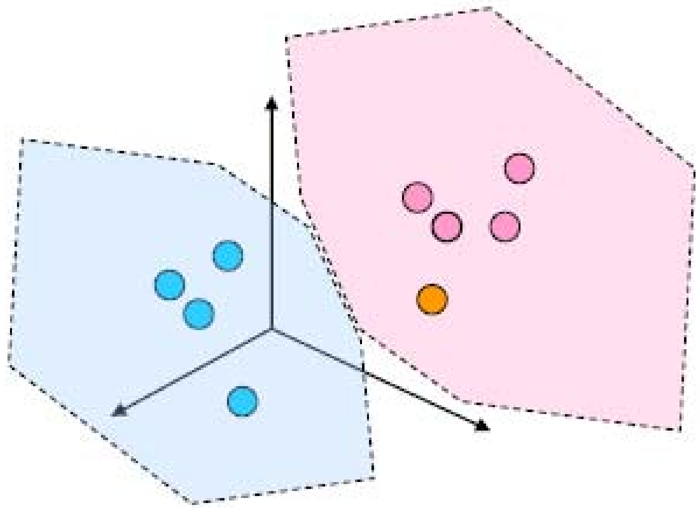
Discriminative classifiers

Query image



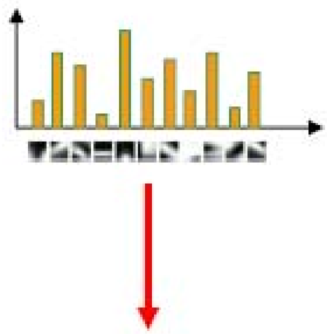
Winning class: pink

Model space



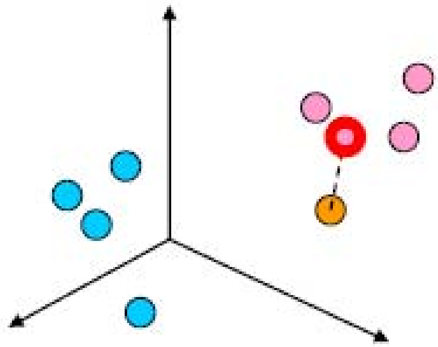
Nearest Neighbors classifier

Query image



Winning class: pink

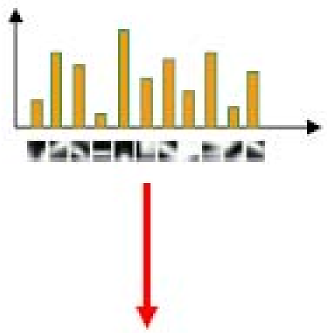
Model space



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

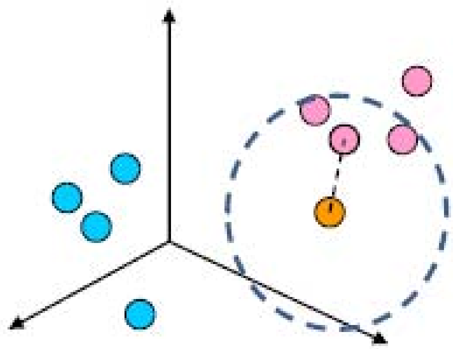
K- Nearest Neighbors classifier

Query image



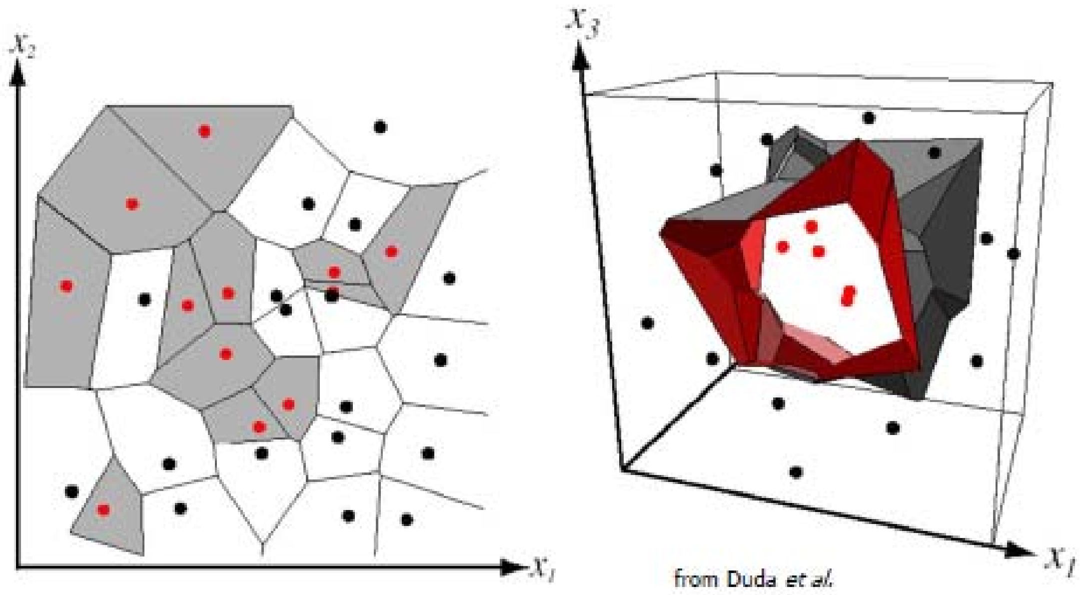
Winning class: pink

Model space



- 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

K- Nearest Neighbors classifier



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

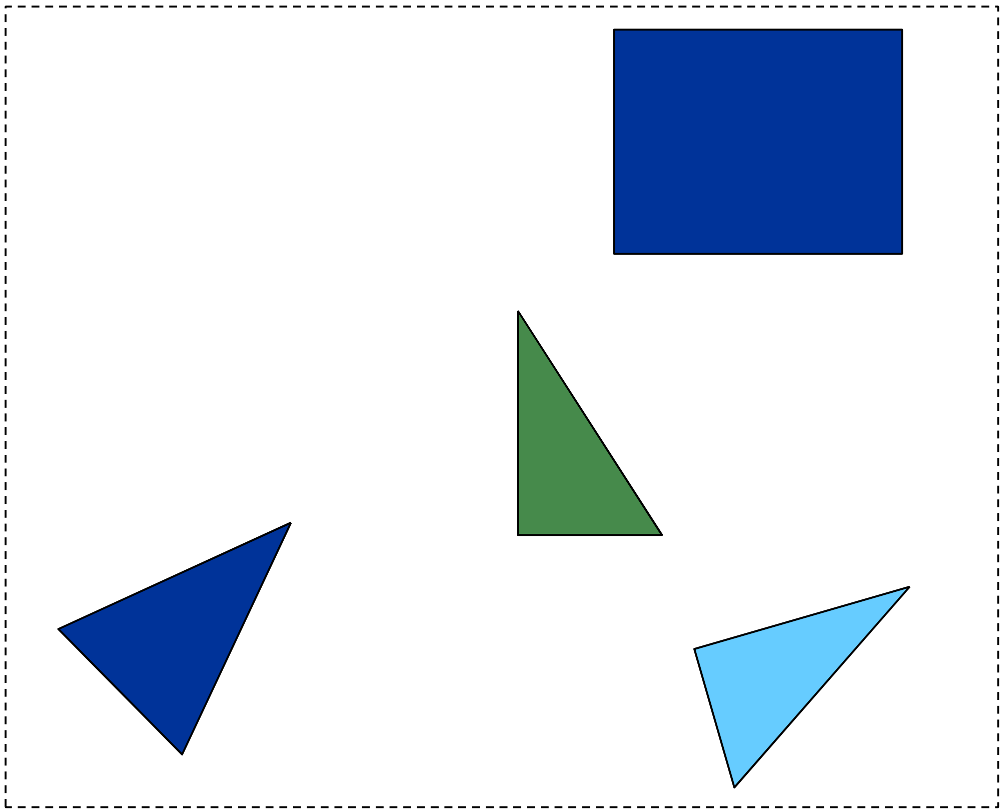
- For k dimensions: k-D tree = space-partitioning data structure for organizing points in a *k*-dimensional space
- Enable efficient search

- Nice tutorial: http://www.cs.umd.edu/class/spring2002/cmsc420-0401/pbasic.pdf

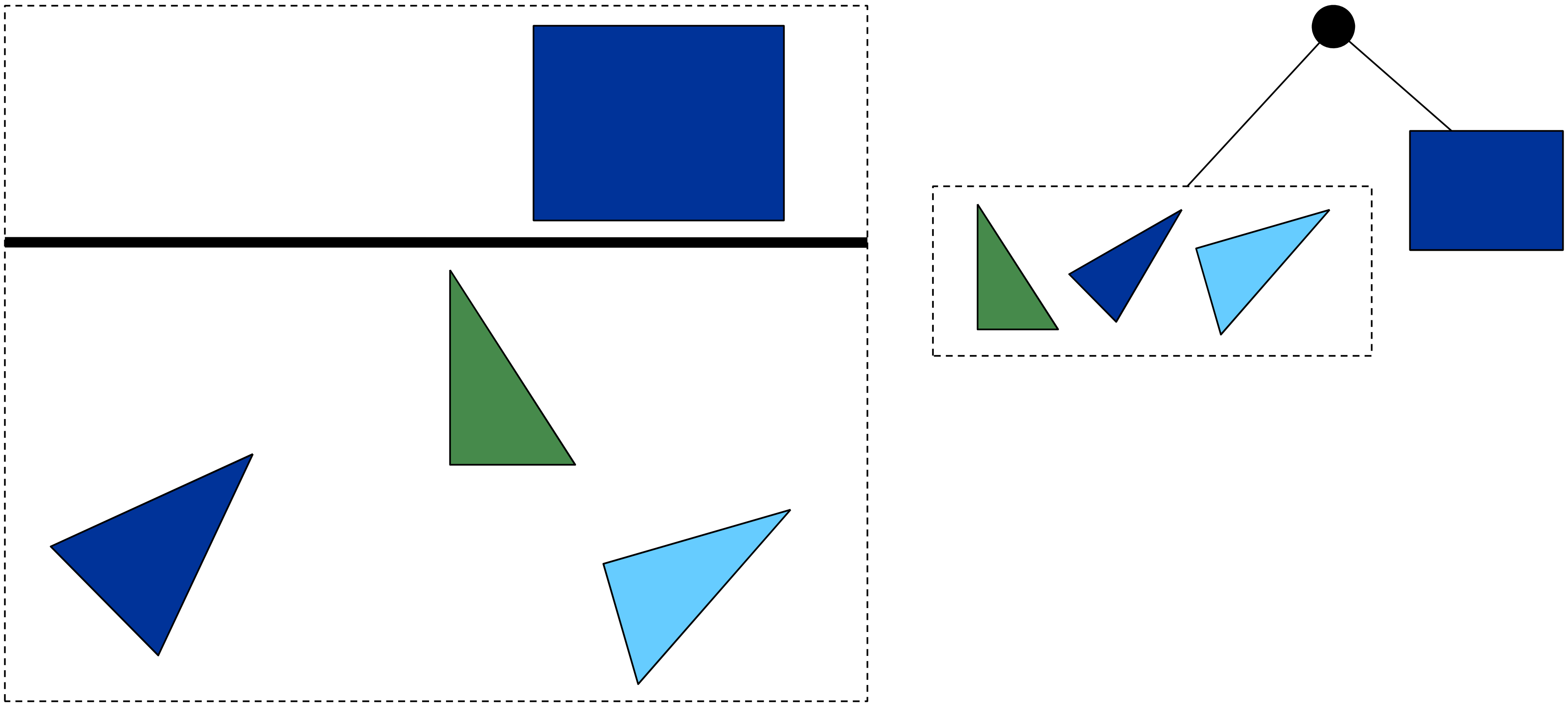
Overview of kd-Trees

- **Binary spatial subdivision (special case of BSP tree)**
- **Split planes aligned on main axis**
- **Inner nodes: subdivision planes**
- **Leaf nodes: points**

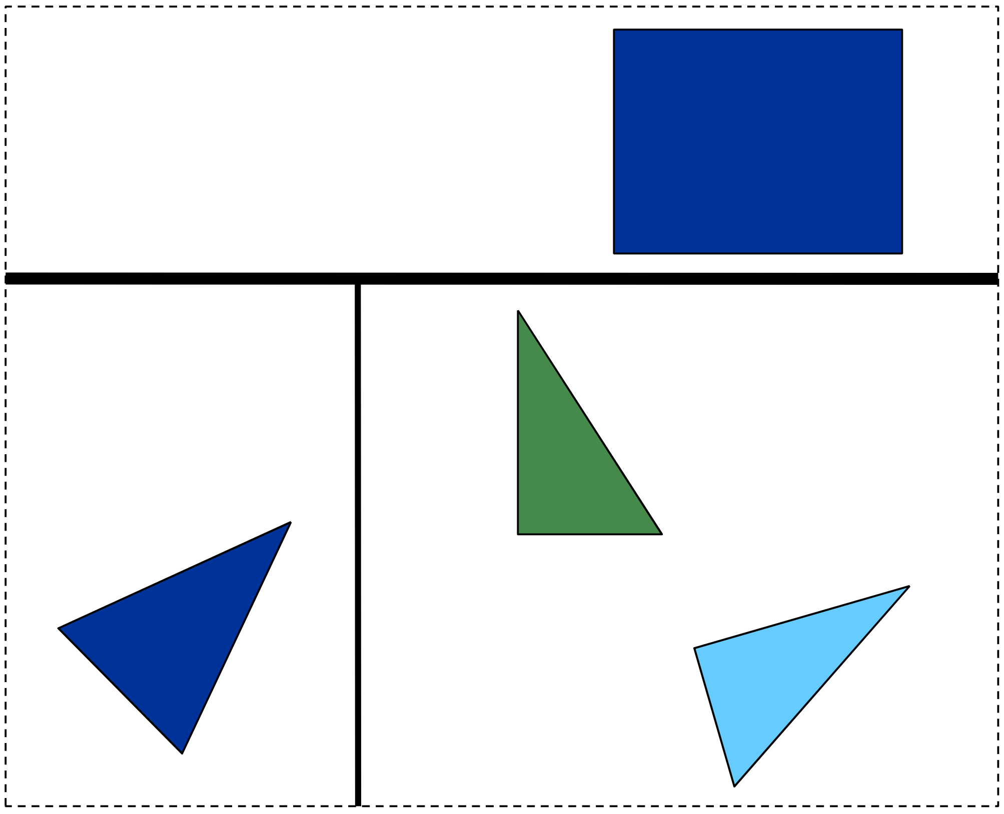
2D Example with Triangles

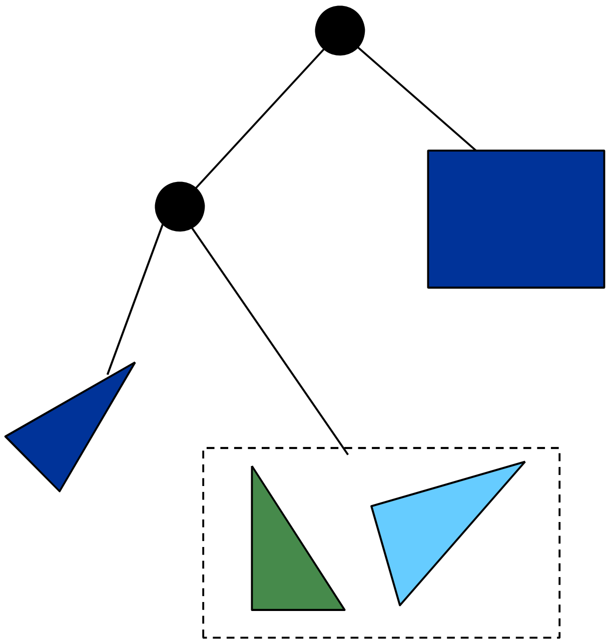


2D Example with Triangles

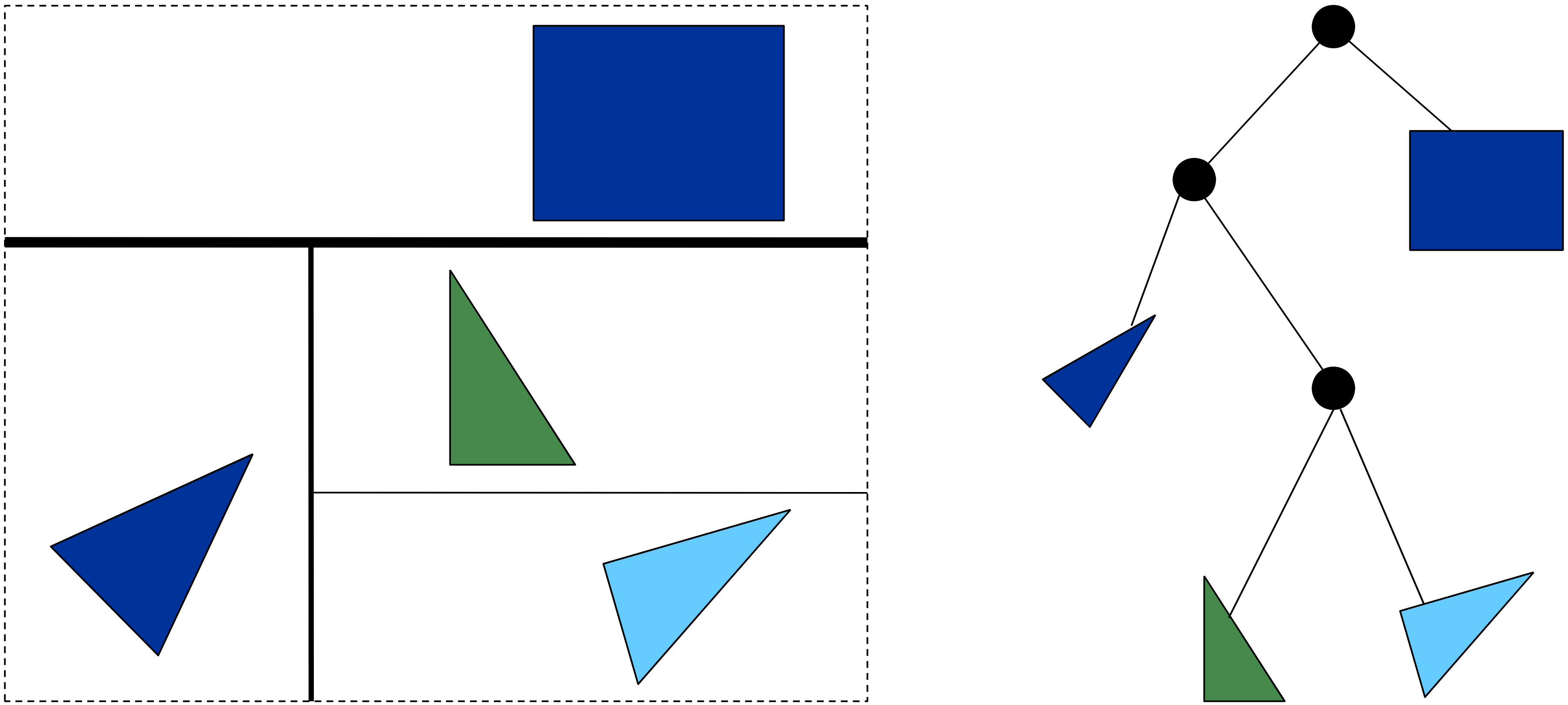


2D Example with Triangles





2D Example with Triangles



Split Planes

- How to select axis & split plane?

- Option 1:
  - Choose a random dimension
  - Subdivide in the middle

- Option 2:
  - Choose a dimension that has a high variance

- Any other options

Nearest Neighbor Search with kd-tree

- Goal: find k nearest neighbors given a point
  - Commonly identify approximate, not exact nearest neighbors

- Apply a depth-first search
  - Traverse the tree with a stack

- Or, we can apply a best-bin first search
  - Traverse more promising nodes first

- Traverse until we visit a certain number of nodes

Hashing techniques

- Kd-trees are not scalable
- Hashing arise as better technology

Functions for comparing histograms

- L1 distance

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

- χ^2 distance (Variance of count data follows mean of the m usually)

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

- Quadratic distance (*cross-bin*)

$$D(h_1, h_2) = \sum_{i,j} A_{ij} (h_1(i) - h_2(j))^2$$

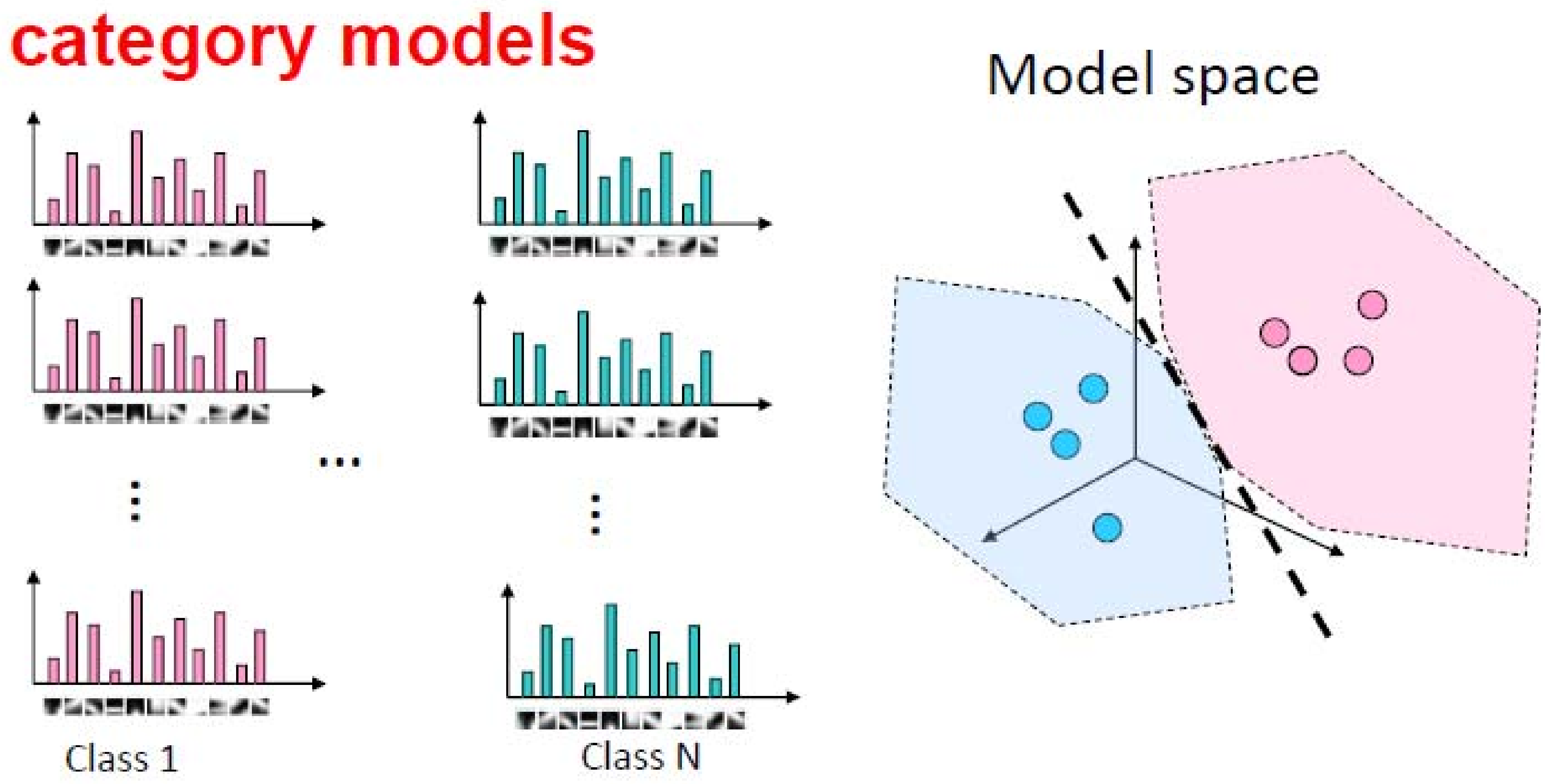
Jan Puzicha, Yossi Rubner, Carlo Tomasi, Joachim M. Buhmann: Empirical Evaluation of Dissimilarity Measures for Color and Texture. ICCV 1999

Learning and Recognition

1. Discriminative method:
    - NN
    - SVM

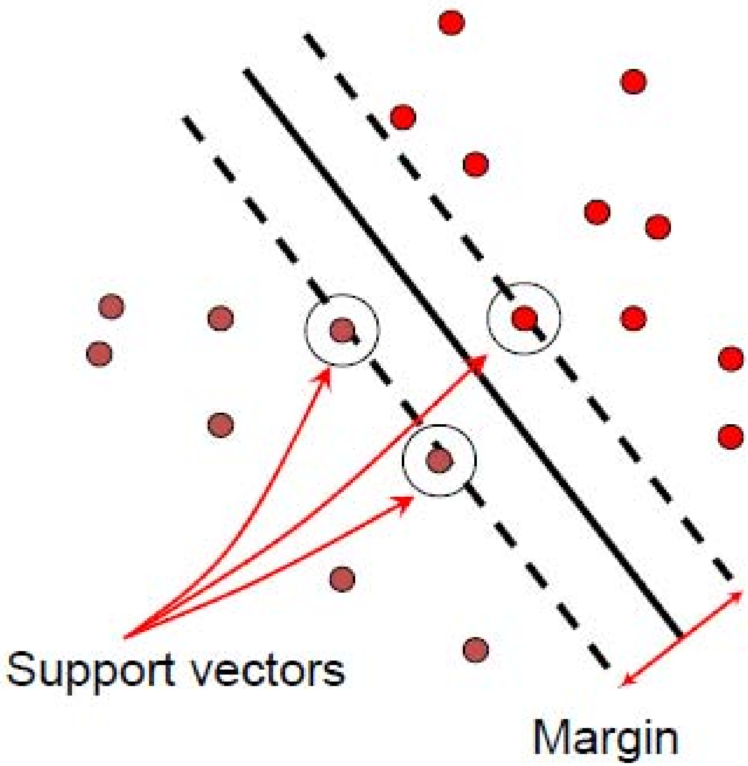
2. Generative method:
    - graphical models

Discriminative classifiers (linear classifier)



Support vector machines

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



Support vectors: $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

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

Margin = $2 / \|\mathbf{w}\|$

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

Classification function (decision boundary):

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

Credit slide: S. Lazebnik

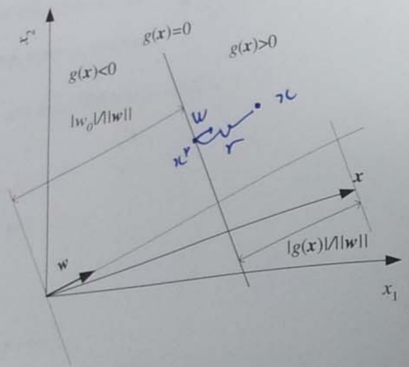


Figure 10.2 The geometric interpretation of the linear discriminant.

$+w_0=0$

where x_p is the normal projection of x onto the hyperplane and r gives us the distance from x to the hyperplane, negative if x is on the negative side, and positive if x is on the positive side (see figure 10.2). Calculating $g(x)$ and noting that $g(x_p) = 0$, we have

$$r = \frac{g(x)}{\|w\|} \tag{10.4}$$

We see then that the distance to origin is

$$r_0 = \frac{w_0}{\|w\|} \tag{10.5}$$

Thus w_0 determines the location of the hyperplane with respect to the origin, and w determines its orientation.

10.3.2 Multiple Classes

When there are $K > 2$ classes, there are K discriminant functions. When they are linear, we have

$$g_i(x|w_i, w_{i0}) = w_i^T x + w_{i0} \tag{10.6}$$

$$g(x) = w_1 \cdot x_1 + w_2 \cdot x_2 + w_0$$
$$= w^T \cdot x + w_0$$

$$x_p : \quad x = x_p + \frac{w}{\|w\|} \cdot r$$

$$\frac{g(x)}{\|w\|} = \frac{w^T}{\|w\|} \cdot x + \frac{w_0}{\|w\|}$$

$$\frac{g(x_p)}{\|w\|} = 0 = \frac{w^T}{\|w\|} \cdot x_p + \frac{w_0}{\|w\|}$$

$$\Rightarrow \frac{w_0}{\|w\|} = -\frac{w^T}{\|w\|} x_p$$

$$\frac{g(x)}{\|w\|} = \frac{w^T}{\|w\|} \cdot x - \frac{w^T}{\|w\|} \cdot x_p$$

$$= \frac{w^T}{\|w\|}(x - x_p)$$

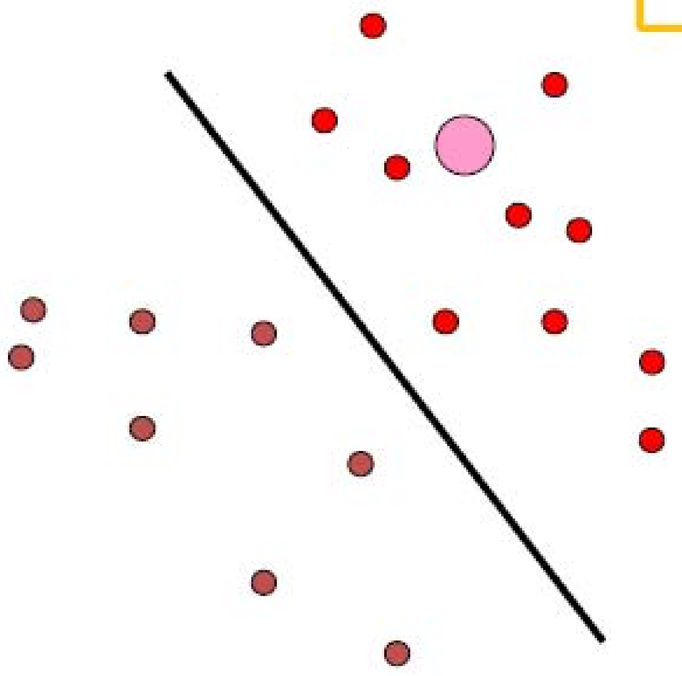
$$= 1 \cdot r \cdot \cos(0^\circ)$$

$$= r.$$

기 록 자	서명(Signature)	일자(Date)

Support vector machines

- Classification



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

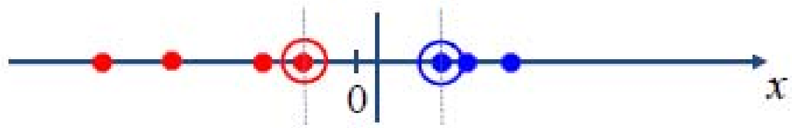
Test point

$$\begin{aligned} &if \quad \mathbf{x} \cdot \mathbf{w} + b \geq 0 \rightarrow \quad class\ 1 \\ &if \quad \mathbf{x} \cdot \mathbf{w} + b < 0 \rightarrow \quad class\ 2 \end{aligned}$$

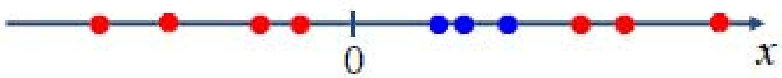
C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Nonlinear SVMs

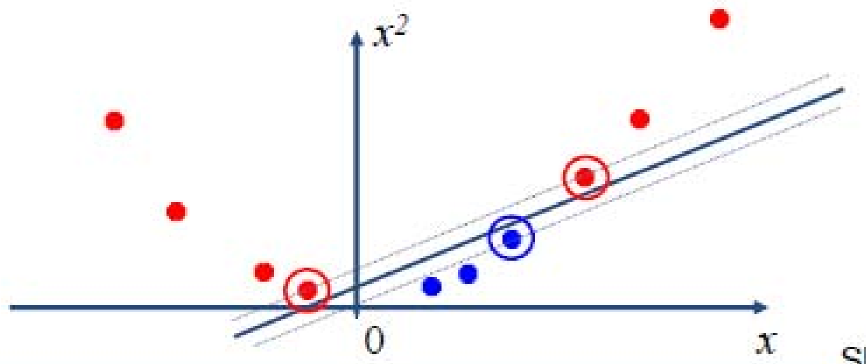
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



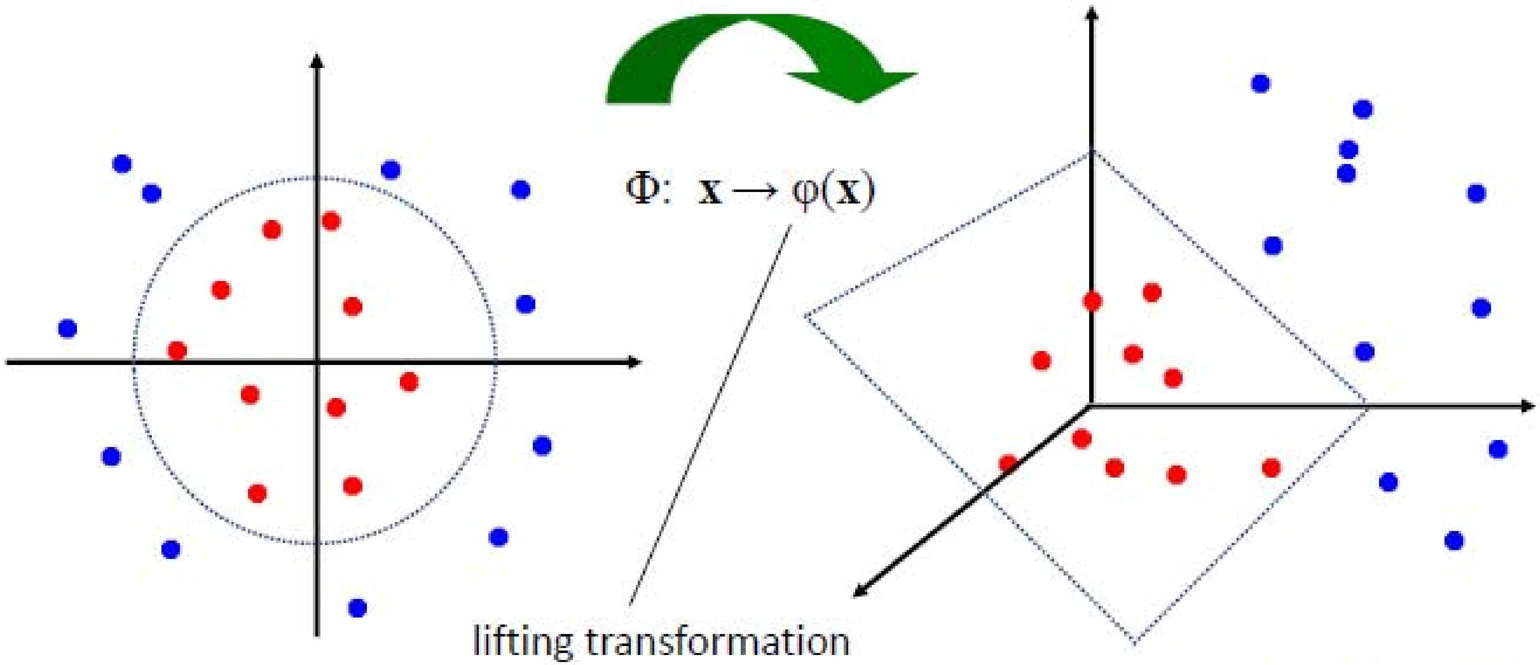
- We can map it to a higher-dimensional space:



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

What about multi-class SVMs?

- No "definitive" multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs
- One vs. others
  - Traning: learn an SVM for each class vs. the others
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM "votes" for a class to assign to the test example

Credit slide: S. Lazebnik

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

Object recognition results

- ETH-80 database of 8 object classes
  (Eichhorn and Chapelle 2004)
- Features:
  - Harris detector
  - PCA-SIFT descriptor, d=10

- Achieves a high accuracy (about 80%)

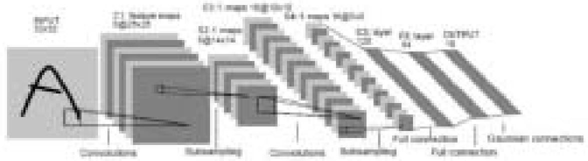
Slide credit: Kristen Grauman

Discriminative models

Nearest neighbor

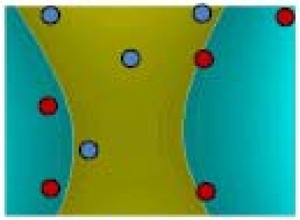
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

Neural networks



LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
...

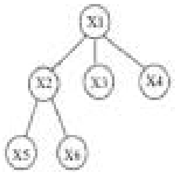
Support Vector Machines



Guyon, Vapnik, Heisele, Serre, Poggio...

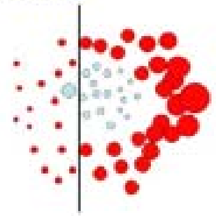
Latent SVM

Structural SVM



Felzenszwalb 00
Ramanan 03...

Boosting



Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,...

Source: Vittorio Ferrari, Kristen Grauman, Antonio Torralba

Learning and Recognition

1. Discriminative method:
    - NN
    - SVM

2. Generative method:
    - graphical models

→ Model the probability distribution that produces a given bag of features

Generative models

1. Naïve Bayes classifier
   – Csurka Bray, Dance & Fan, 2004

2. Hierarchical Bayesian text models (pLSA and LDA)
   – Background: Hoffman 2001, Blei, Ng & Jordan, 2004
   – Object categorization: Sivic et al. 2005, Sudderth et al. 2005
   – Natural scene categorization: Fei-Fei et al. 2005

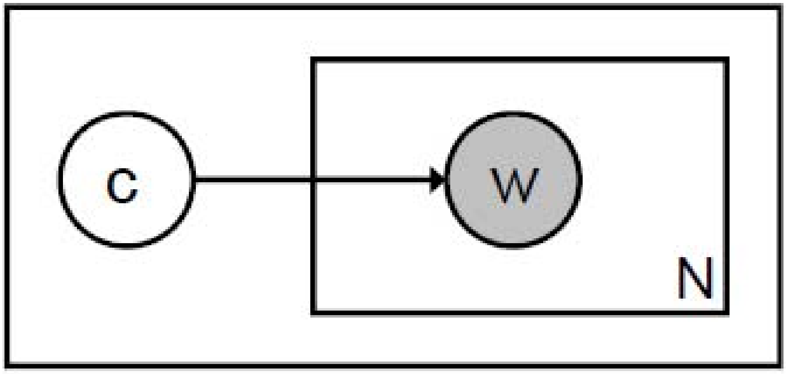
Some notations

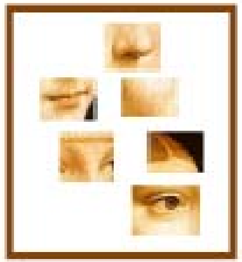
- **w:** a collection of all N codewords in the image

  $\mathbf{w} = [w_1, w_2, \ldots, w_N]$

- c: category of the image

the Naïve Bayes model





Graphical model

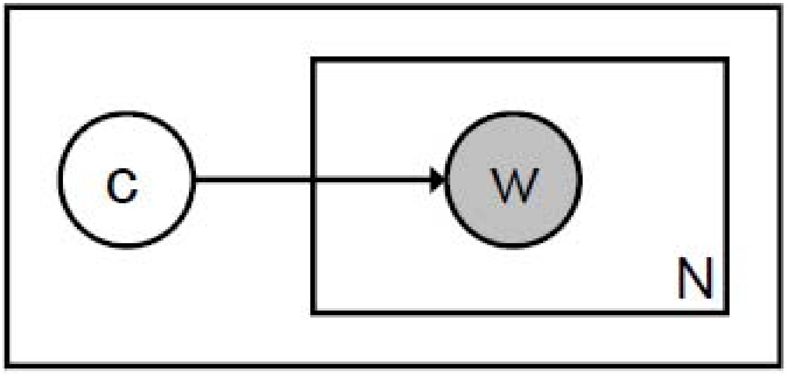
Posterior = $p(c \mid w) \propto p(c)\, p(w \mid c)$

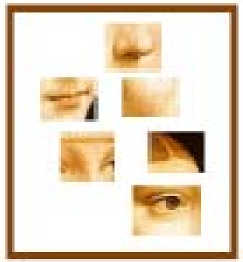
probability
that image **I** is
of category c

Prior prob. of
the object classes

Image likelihood
given the class

the Naïve Bayes model





Graphical model

$$c^* = \arg\max_c \; p(c \mid w) \propto p(c)\,p(w \mid c) = p(c)\prod_{n=1}^{N} p(w_n \mid c)$$

Object class decision

Likelihood of ith visual word given the class

Estimated by empirical frequencies of code words in images from a given class

Our in-house database contains 1776 images in seven classes[1]: faces, buildings, trees, cars, phones, bikes and books. Fig. 2 shows some examples from this dataset.

Csurka et al. 2004

Table 1. Confusion matrix and the mean rank for the best vocabulary (*k=1000*).

True classes →	*faces*	*buildings*	*trees*	*cars*	*phones*	*bikes*	*books*
faces	**76**	4	2	3	4	4	13
buildings	2	**44**	5	0	5	1	3
trees	3	2	**80**	0	0	5	0
cars	4	1	0	**75**	3	1	4
phones	9	15	1	16	**70**	14	11
bikes	2	15	12	0	8	**73**	0
books	4	19	0	6	7	2	**69**

Csurka et al. 2004

Generative vs discriminative

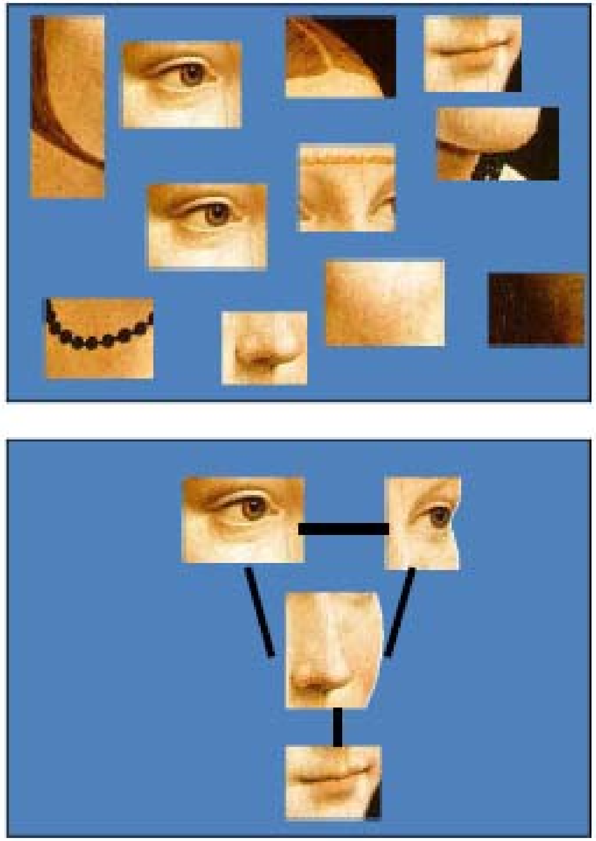
- Discriminative methods
  - Computationally efficient & fast

- Generative models

  - Flexibility in modeling parameters

Weakness of BoW the models

- No rigorous geometric information of the object components
- It's intuitive to most of us that objects are made of parts – no such information
- Not extensively tested yet for
  - View point invariance
  - Scale invariance
- Segmentation and localization unclear



What have we learned today?

- Bag of Words models
  - Basic representation
  - Different learning and recognition algorithms

Next Time…

- Various image retrieval systems

Homework for Every Class

- Go over the next lecture slides
- Come up with one question on what we have discussed today
  - 1 for typical questions (that were answered in the class)
  - 2 for questions with thoughts or that surprised me
- Write questions at least 4 times