# Bag-of-features for category classification

Cordelia Schmid

INRIA

LEAR

# Category recognition

- Image classification: assigning a class label to the image

Car: present
Cow: present
Bike: not present
Horse: not present
...

# Category recognition

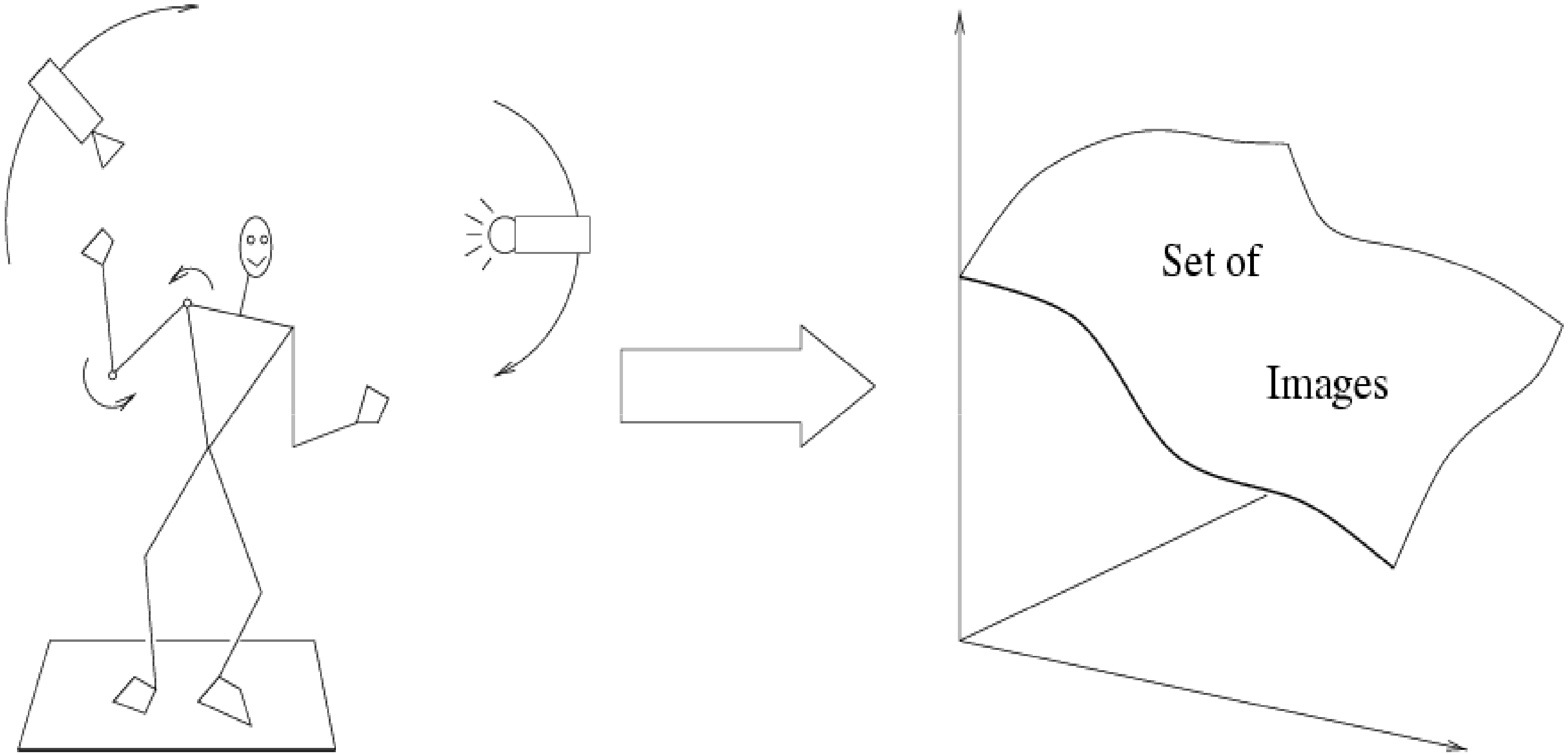
- Image classification: assigning a class label to the image

Car: present
Cow: present
Bike: not present
Horse: not present
...

- Object localization: define the location and the category


Location
Category

# Difficulties: within object variations




Variability: Camera position, Illumination,Internal parameters

⇨ Within-object variations

# Difficulties: within-class variations



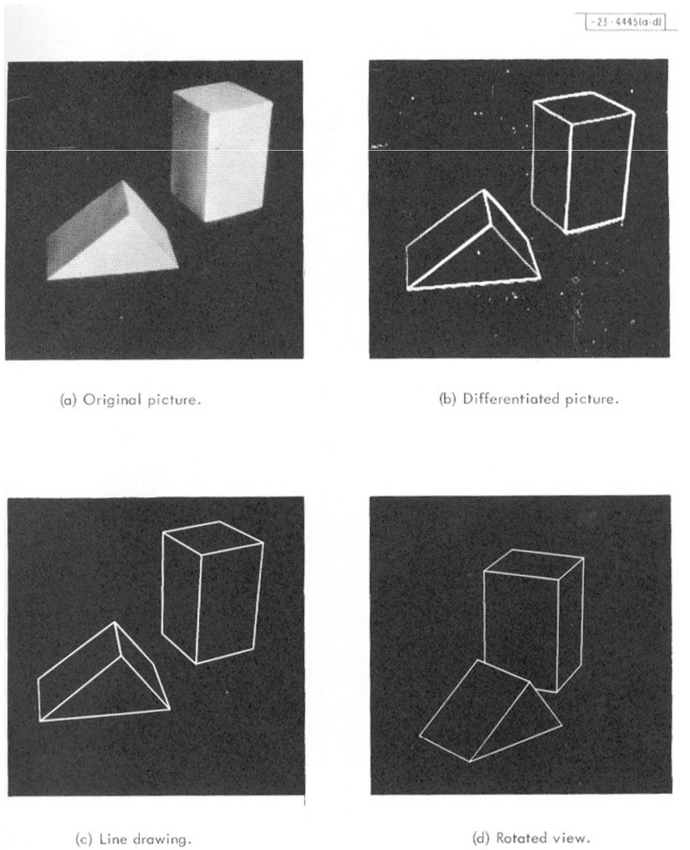



# Category recognition

- Robust image description
  - Appropriate descriptors for categories
- Statistical modeling and machine learning for vision
  - Use and validation of appropriate techniques

# Why machine learning?

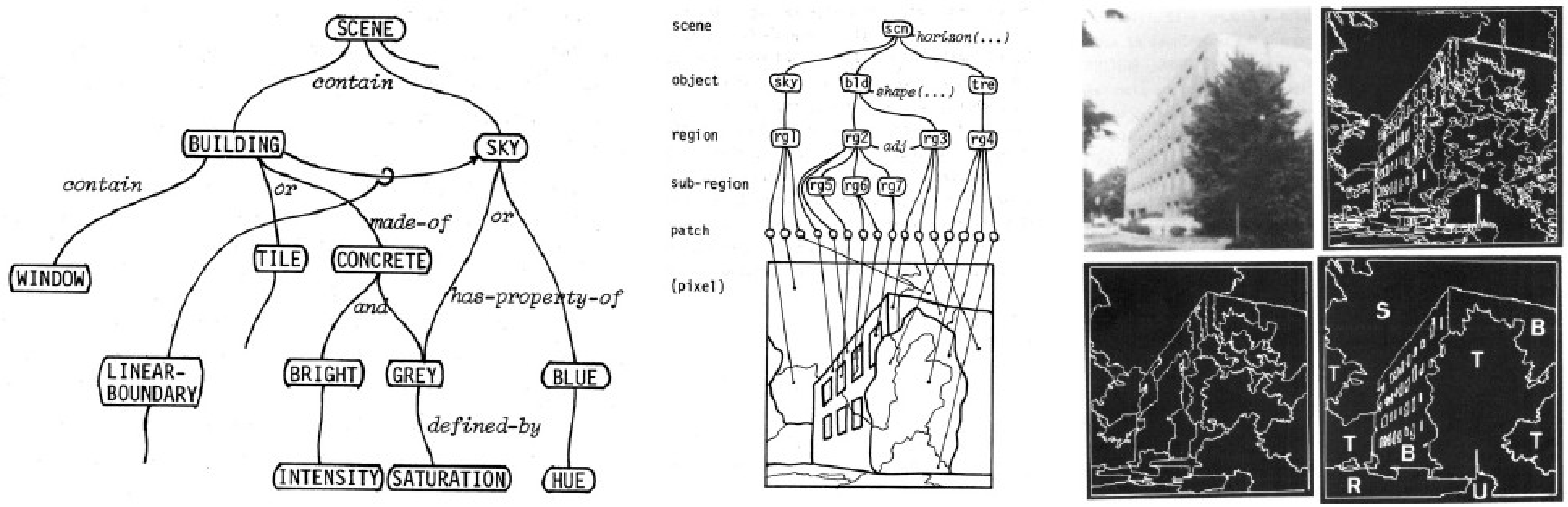
- Early approaches: simple features + handcrafted models
- Can handle only few images, simples tasks




L. G. Roberts, *Machine Perception of Three Dimensional Solids*, Ph.D. thesis, MIT Department of Electrical Engineering, 1963.

# Why machine learning?

- Early approaches: manual programming of rules
- Tedious, limited and does not take into accout the data




(a) Bottom-up process (b) Top-down process (c) Result

Figure 3. A system developed in 1978 by Ohta, Kanade and Sakai [33, 32] for knowledge-based interpretation of outdoor natural scenes. The system is able to label an image (c) into semantic classes: S-sky, T-tree, R-road, B-building, U-unknown.

*Y. Ohta, T. Kanade, and T. Sakai,* "An Analysis System for Scenes Containing objects with Substructures," *International Joint Conference on Pattern Recognition*, 1978.

# Why machine learning?

- Today lots of data, complex tasks

Internet images,
personal photo albums

Movies, news, sports

- Instead of trying to encode rules directly, learn them from examples of inputs and desired outputs

# Types of learning problems

- Supervised
  - Classification
  - Regression
- Unsupervised
- Semi-supervised
- Active learning
- ….

# Supervised learning

- Given training examples of inputs and corresponding outputs, produce the “correct” outputs for new inputs

- Two main scenarios:

  – **Classification:** outputs are discrete variables (category labels). Learn a decision boundary that separates one class from the other

  – **Regression:** also known as “curve fitting” or “function approximation.” Learn a continuous input-output mapping from examples (possibly noisy)

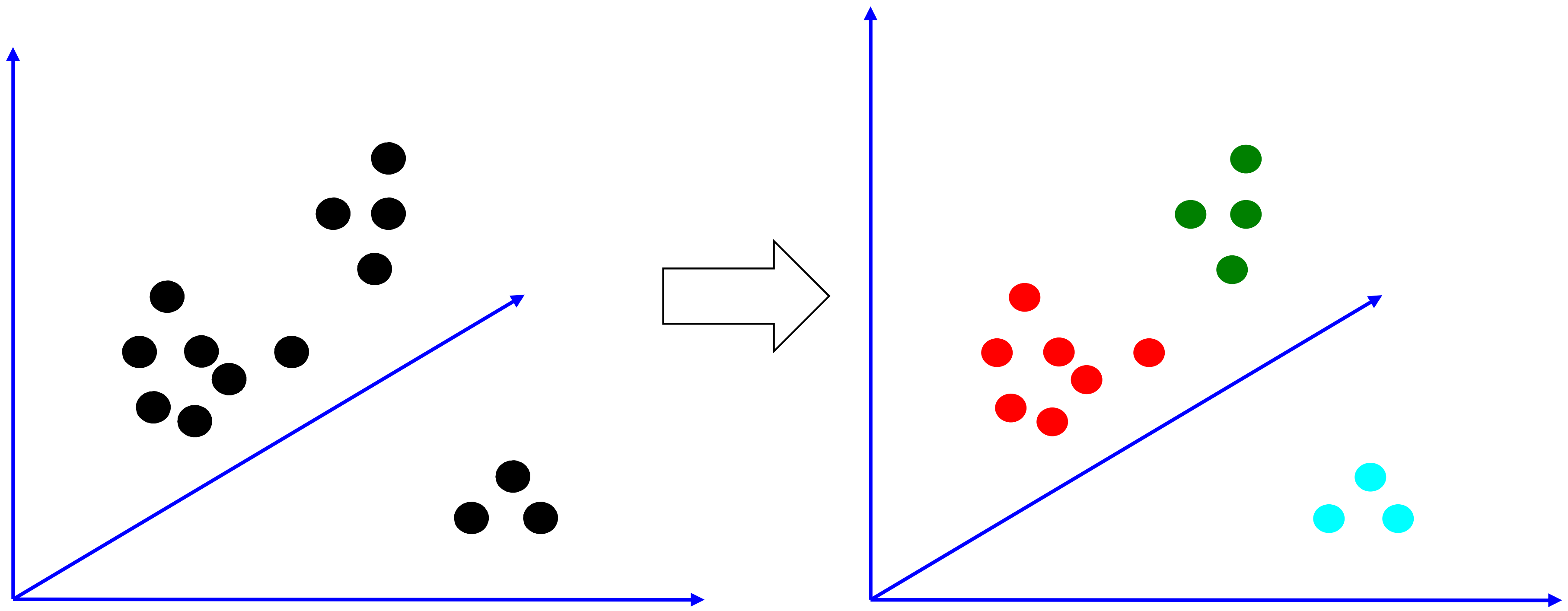
# Unsupervised Learning

- Given only *unlabeled* data as input, learn some sort of structure

- The objective is often more vague or subjective than in supervised learning. This is more an exploratory/descriptive data analysis

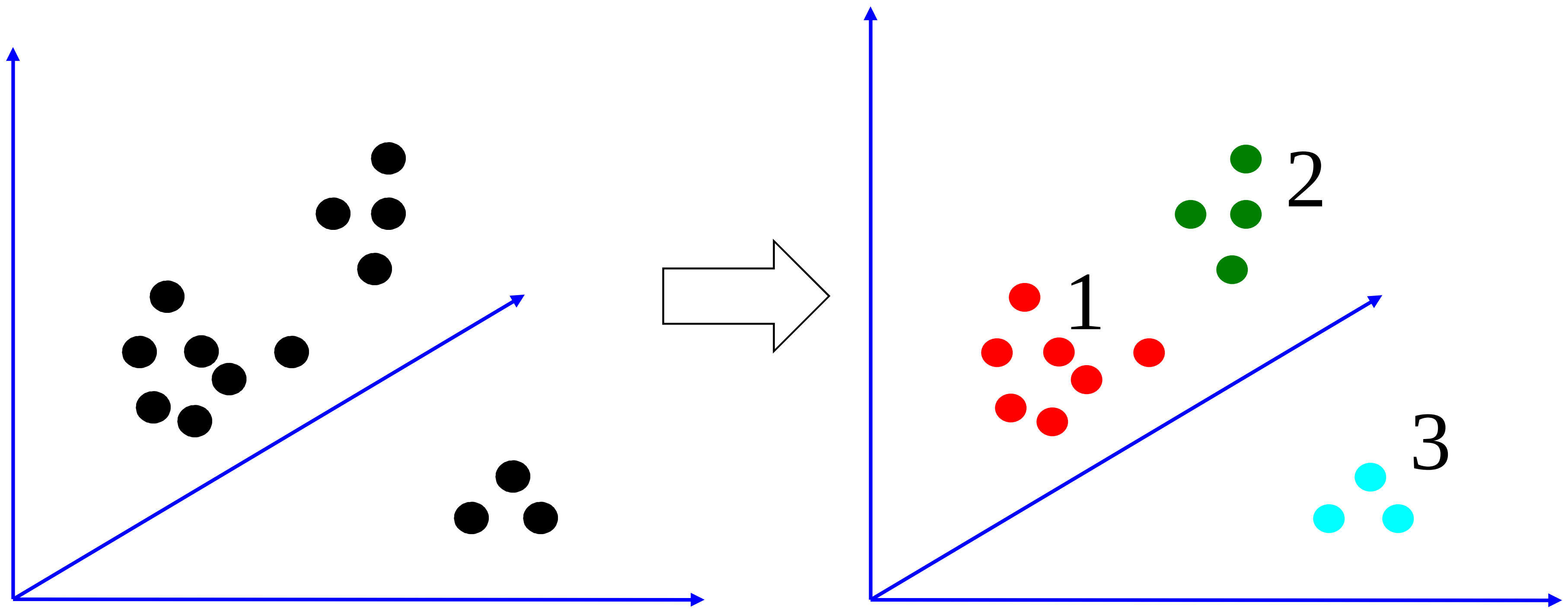
# Unsupervised Learning

- **Clustering**
  - Discover groups of “similar” data points



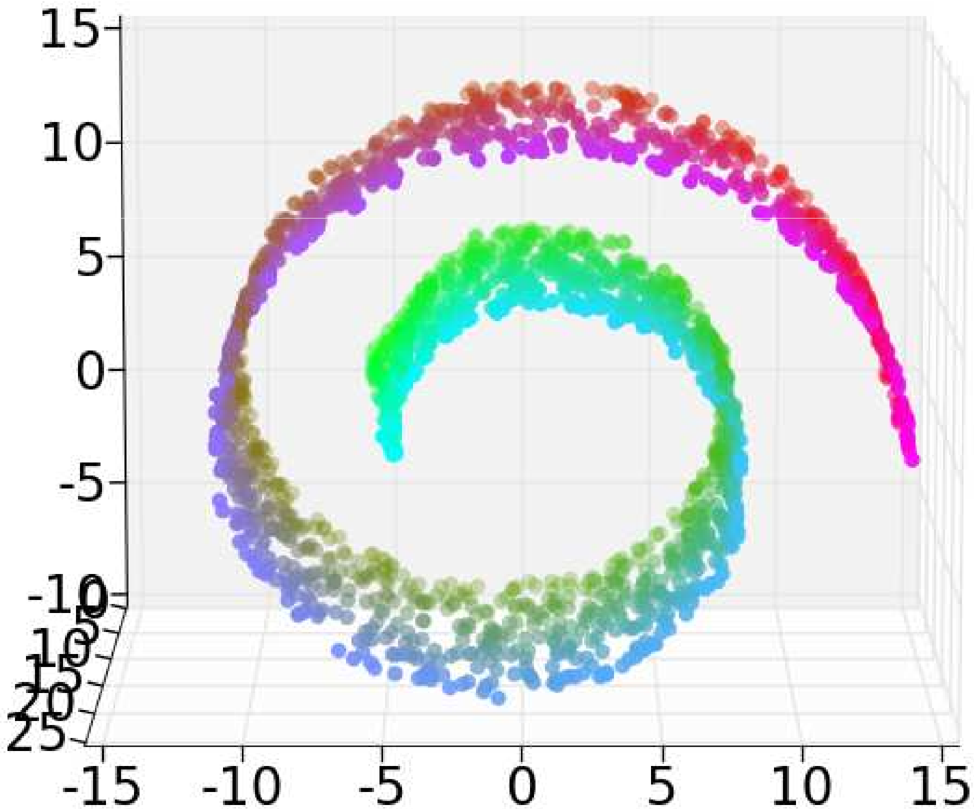
# Unsupervised Learning

- **Quantization**
  - Map a continuous input to a discrete (more compact) output



# Unsupervised Learning

- **Dimensionality reduction, manifold learning**
  – Discover a lower-dimensional surface on which the data lives

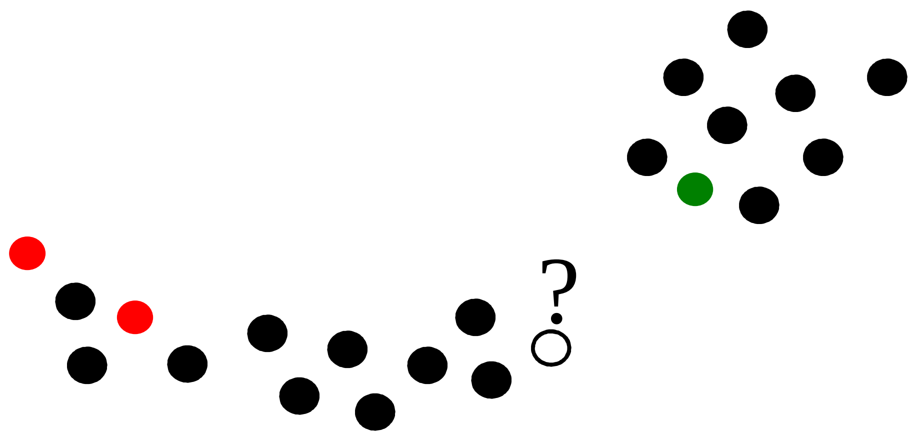


# Other types of learning

- **Semi-supervised learning:** lots of data is available, but only small portion is labeled (e.g. since labeling is expensive)

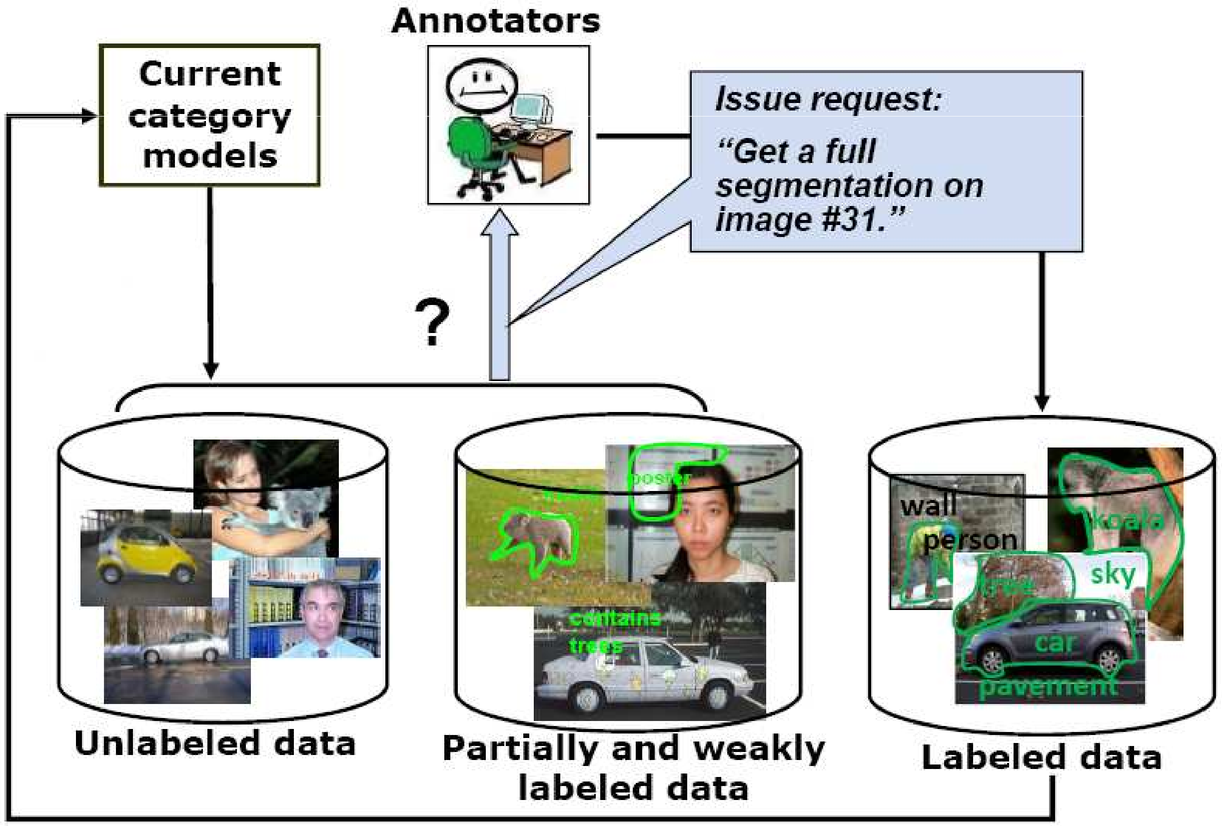
# Other types of learning

- **Semi-supervised learning:** lots of data is available, but only small portion is labeled (e.g. since labeling is expensive)
  - Why is learning from labeled and unlabeled data better than learning from labeled data alone?



# Other types of learning

- **Active learning:** the learning algorithm can choose its own training examples, or ask a “teacher” for an answer on selected inputs



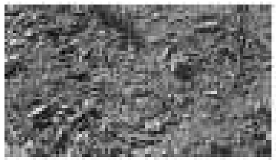
# Image classification

- Given

  Positive training images containing an object class

  Negative training images that don’t

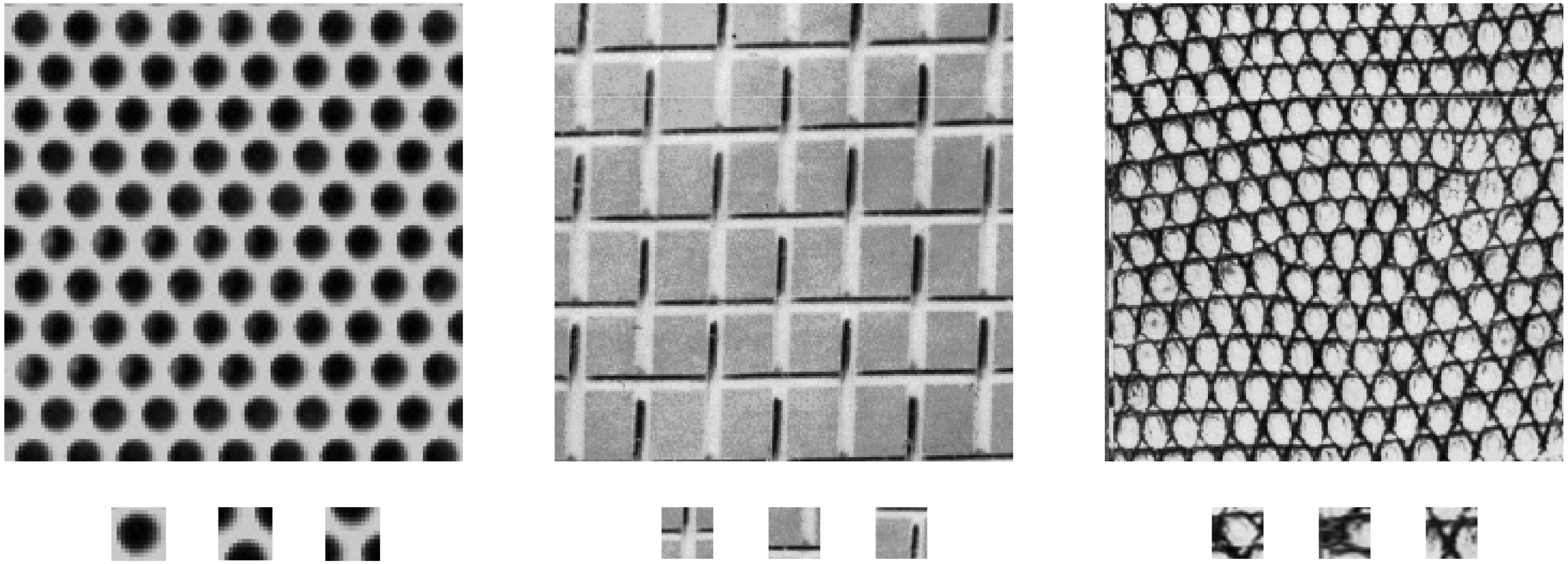
- Classify

  A test image as to whether it contains the object class or not

?

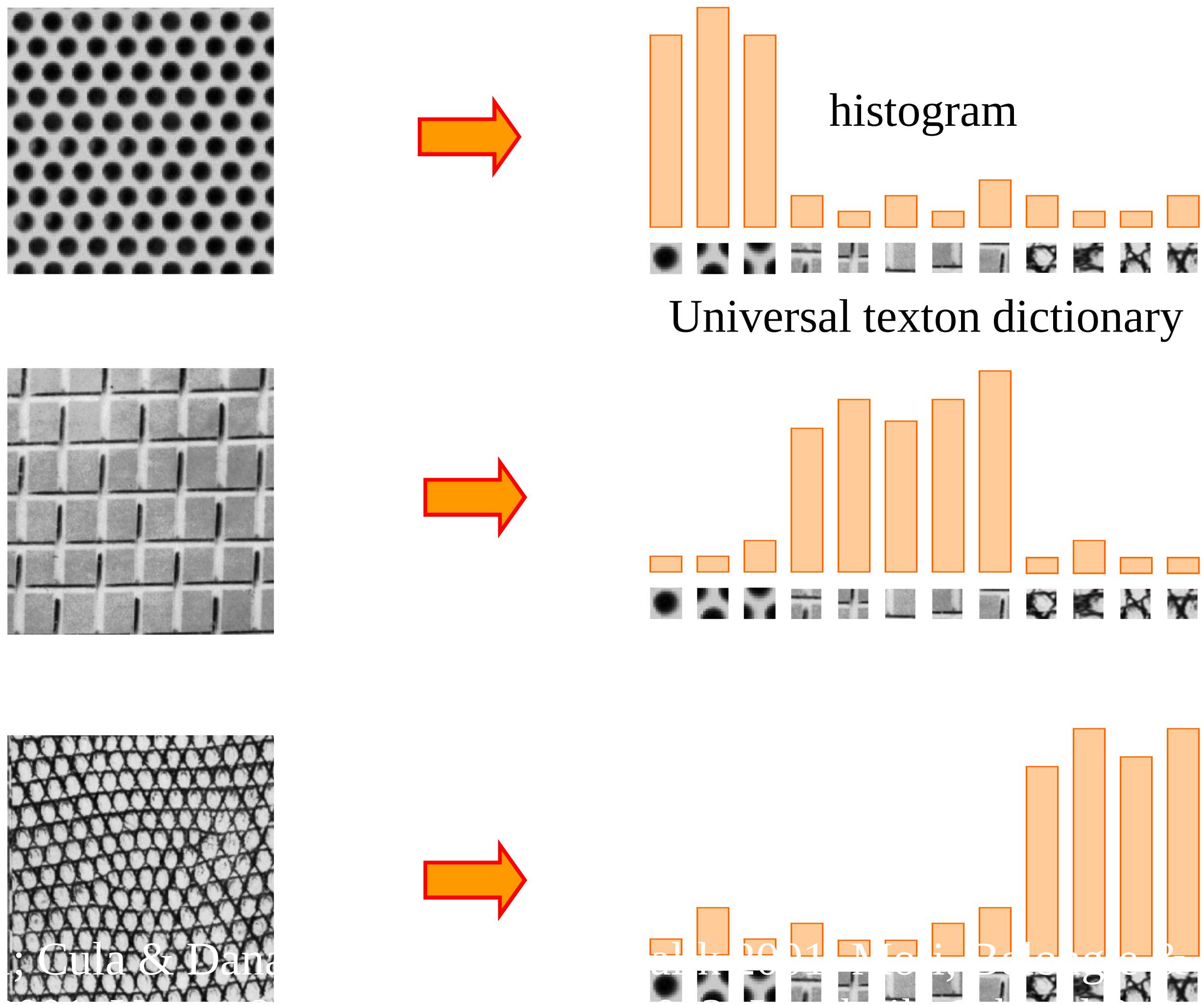
# Bag-of-features for image classification

- Origin: texture recognition
  - Texture is characterized by the repetition of basic elements or *textons*



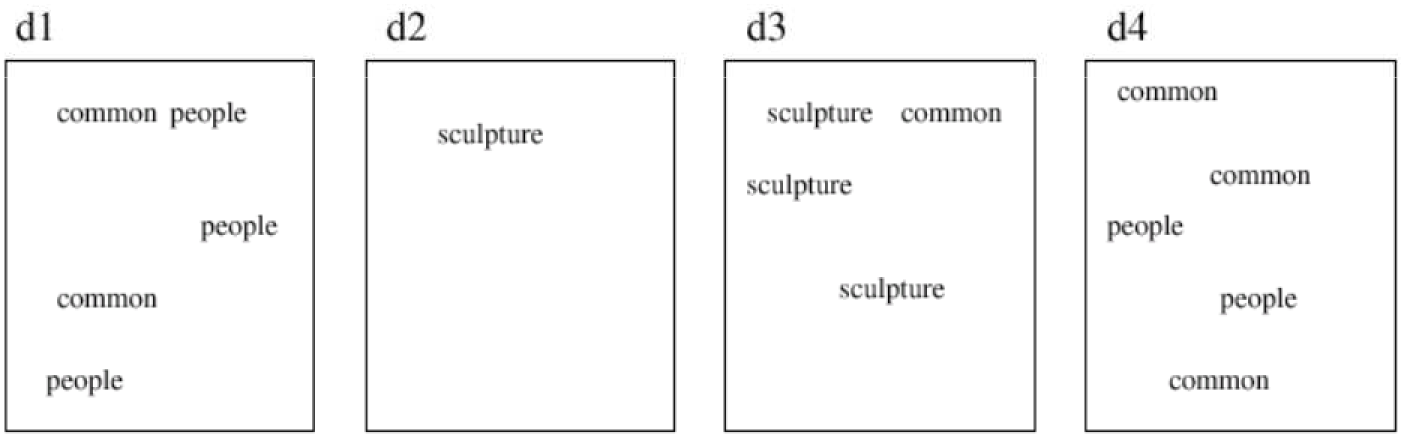
Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001
Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

# Texture recognition



# Bag-of-features – Origin: bag-of-words (text)

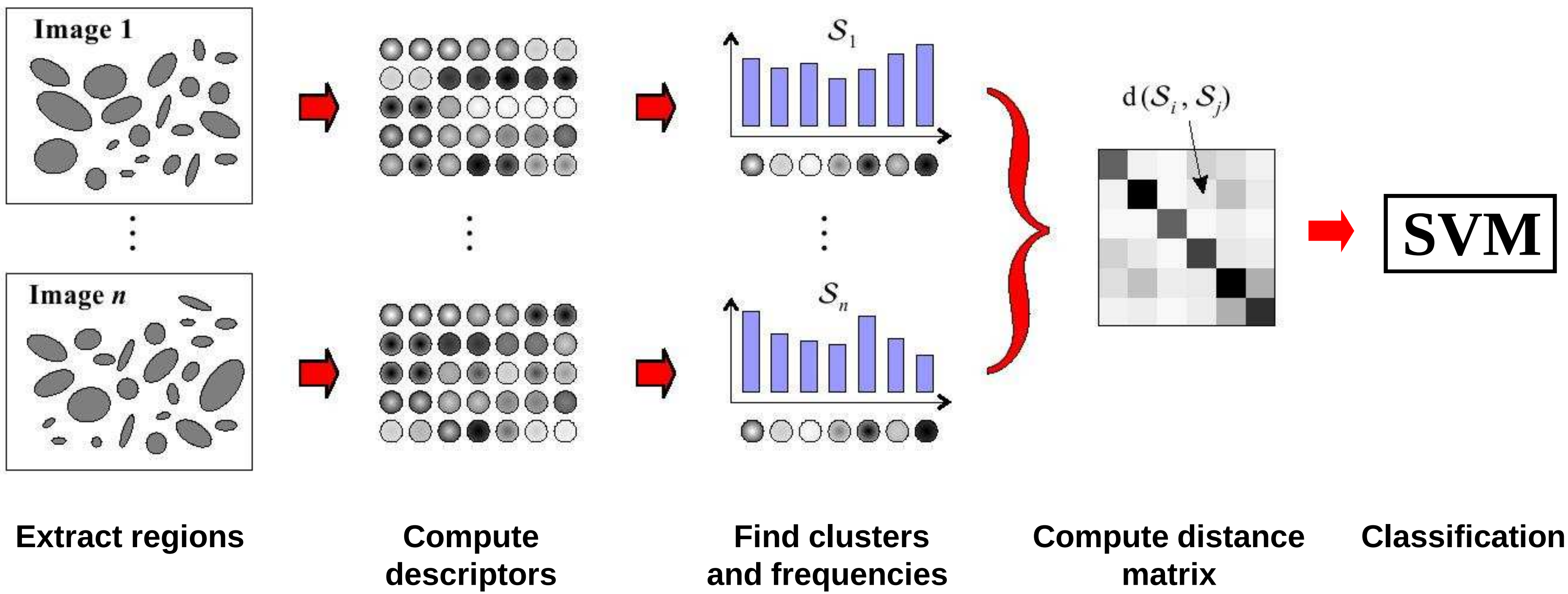
- Orderless document representation: frequencies of words from a dictionary
- Classification to determine document categories




Bag-of-words

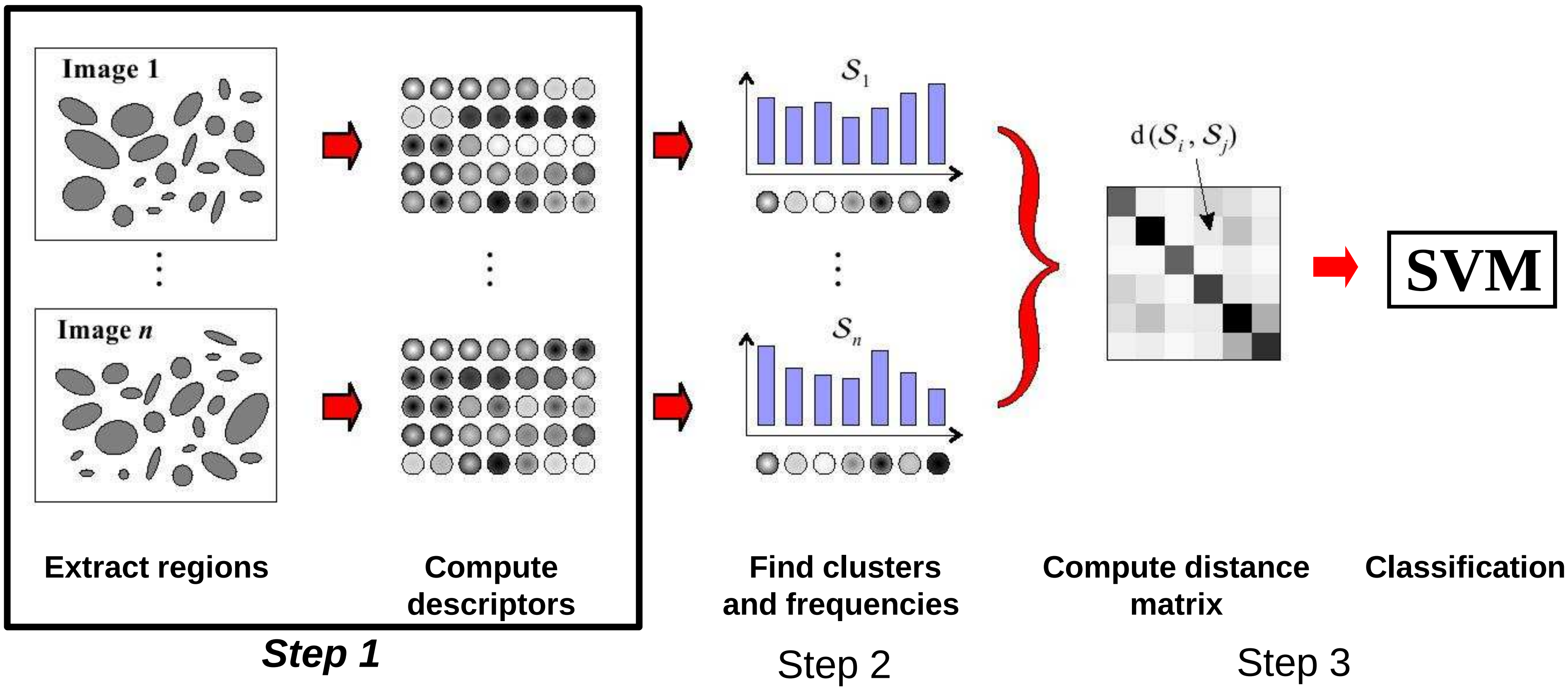
| | | | | |
|---|---|---|---|---|
| Common | 2 | 0 | 1 | 3 |
| People | 3 | 0 | 0 | 2 |
| Sculpture | 0 | 1 | 3 | 0 |
| … | … | … | … | … |

# Bag-of-features for image classification




[Nowak,Jurie&Triggs,ECCV'06], [Zhang,Marszalek,Lazebnik&Schmid,IJCV'07]

# Bag-of-features for image classification




[Nowak,Jurie&Triggs,ECCV'06], [Zhang,Marszalek,Lazebnik&Schmid,IJCV'07]

# Step 1: feature extraction

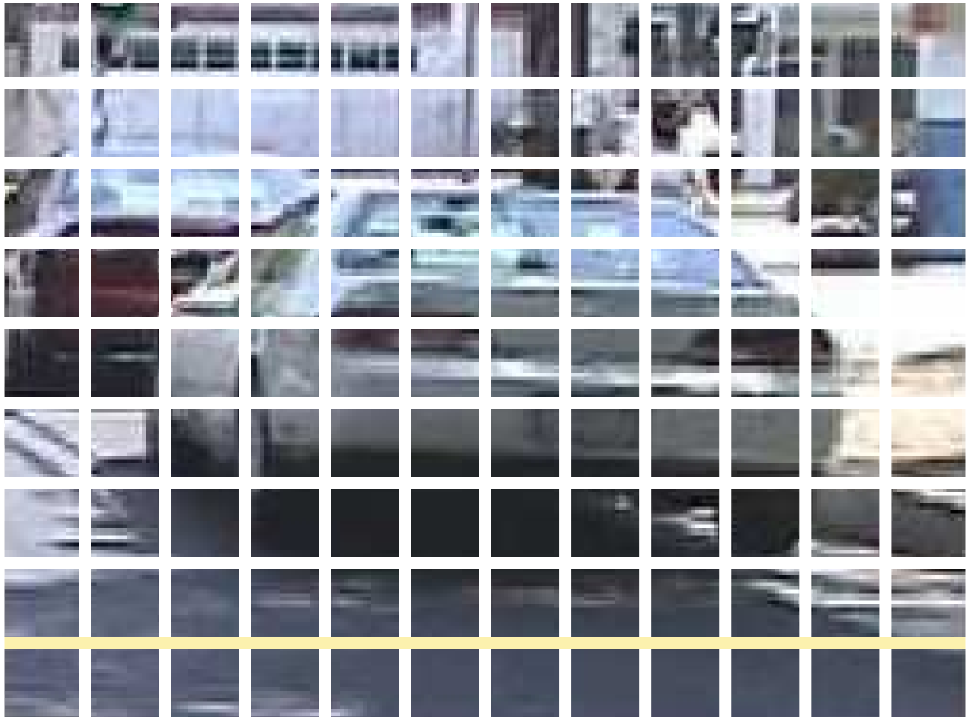
- Scale-invariant image regions + SIFT (see lecture 2)
  - Affine invariant regions give “too” much invariance
  - Rotation invariance for many realistic collections “too” much invariance

- Dense descriptors
  - Improve results in the context of categories (for most categories)
  - Interest points do not necessarily capture “all” features

- Color-based descriptors

- Shape-based descriptors

# Dense features

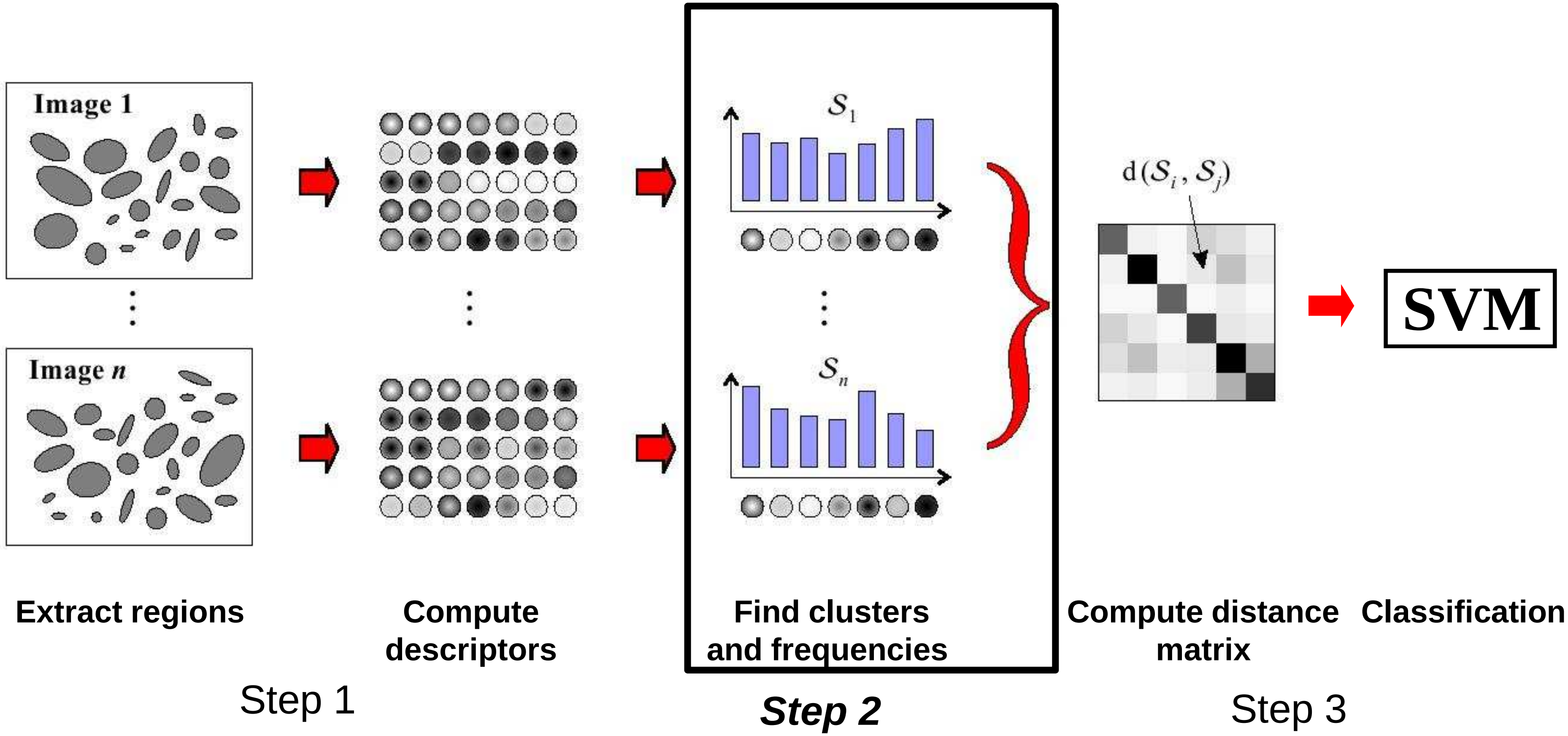


- Multi-scale dense grid: extraction of small overlapping patches at multiple scales

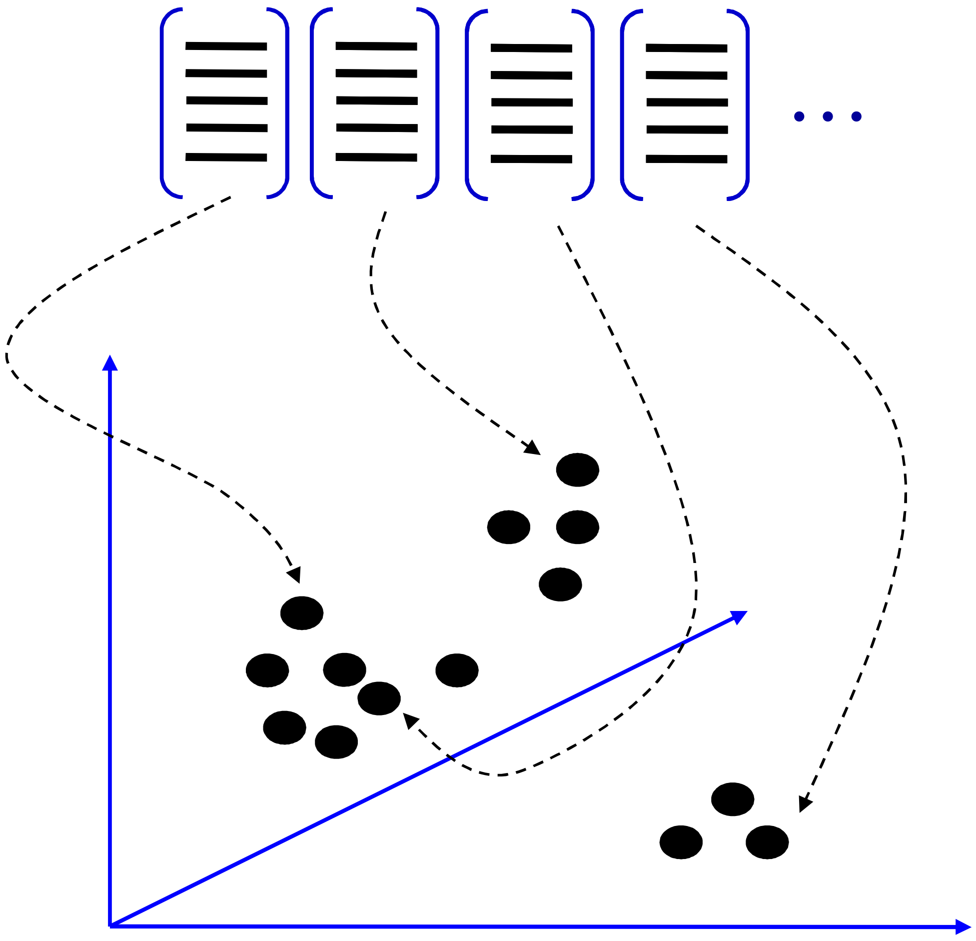
-Computation of  the SIFT descriptor  for each grid cells

-Exp.: Horizontal/vertical step size 6 pixel, scaling factor of 1.2 per level

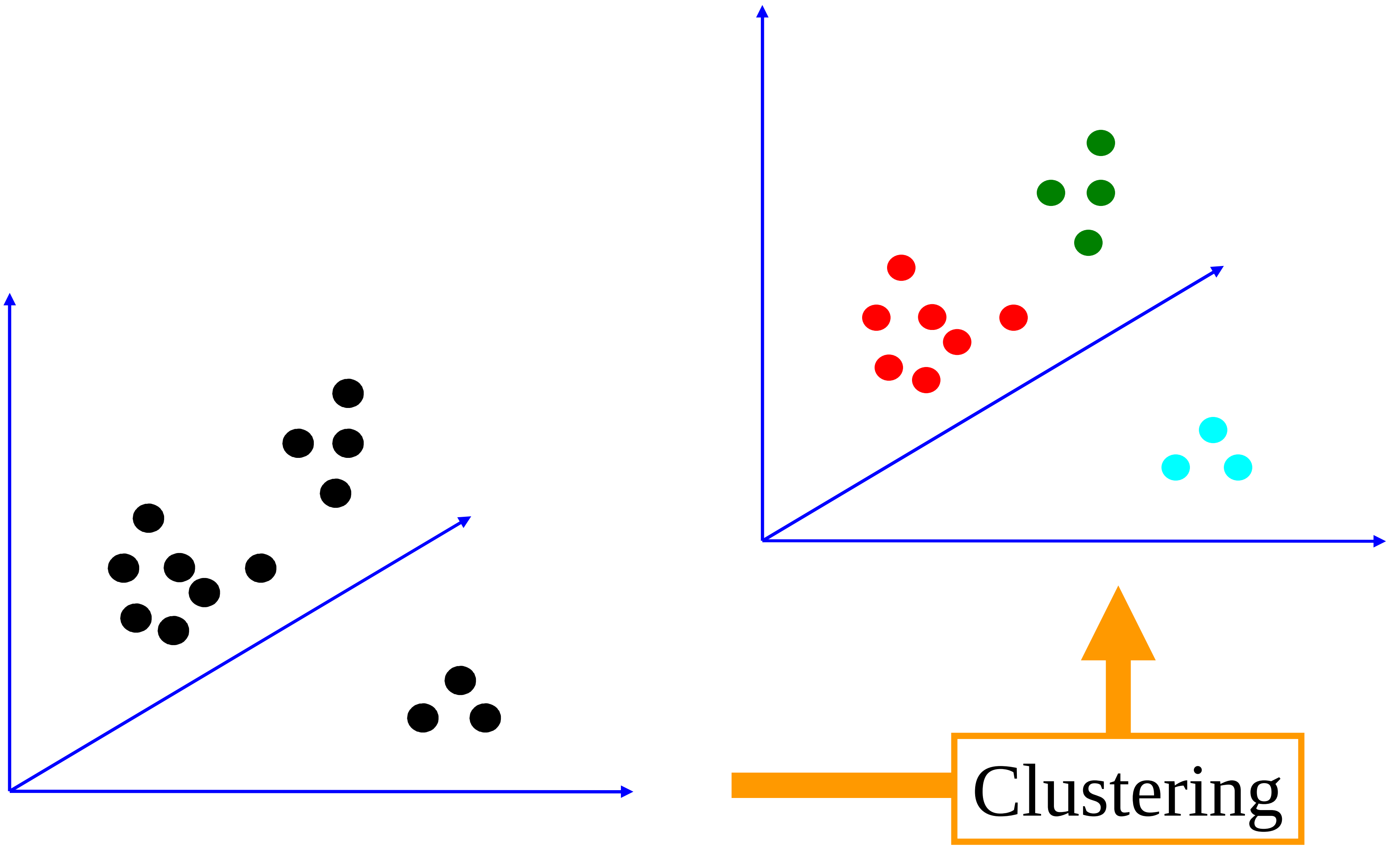
# Bag-of-features for image classification



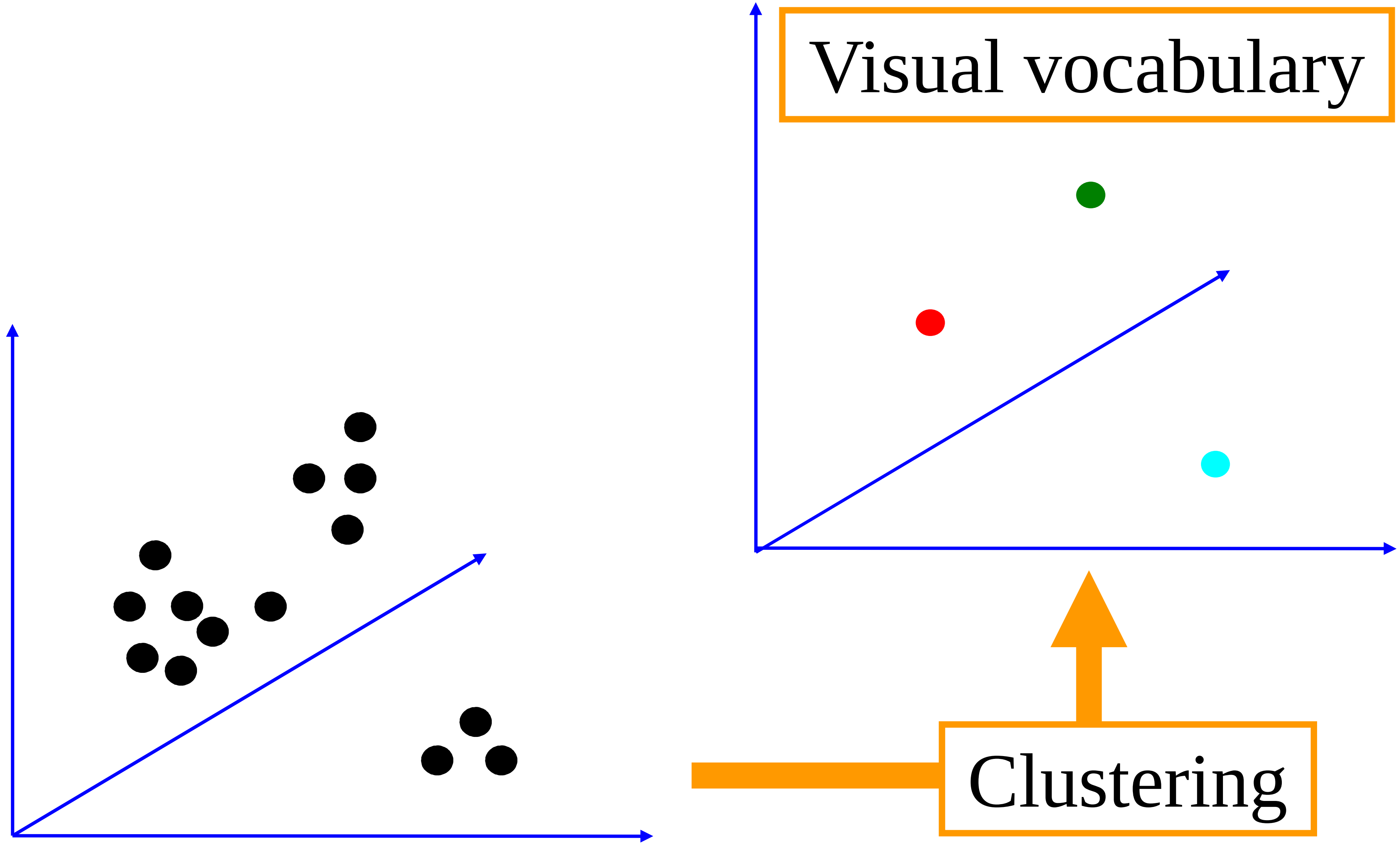
# Step 2: Quantization



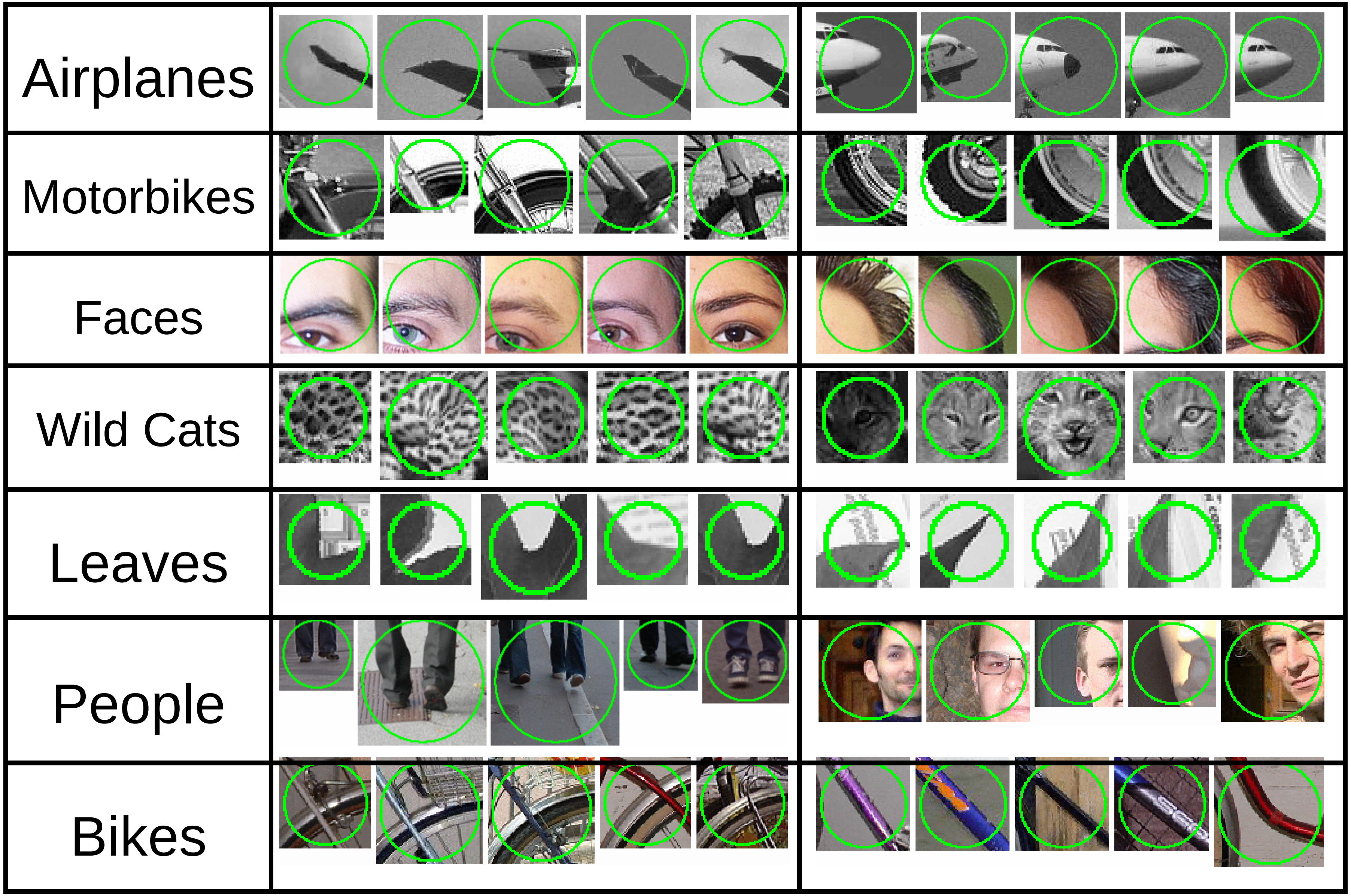
# Step 2:Quantization



# Step 2: Quantization



# Examples for visual words



# Step 2: Quantization

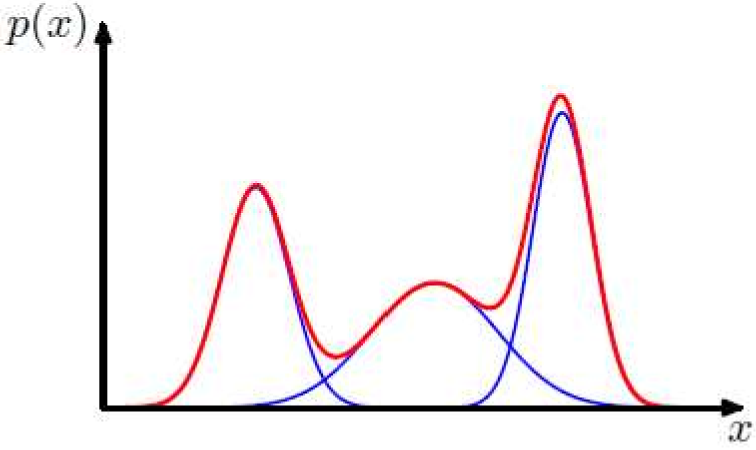
- Cluster descriptors
  - K-means
  - Gaussian mixture model
- Assign each visual word to a cluster
  - Hard or soft assignment
- Build frequency histogram

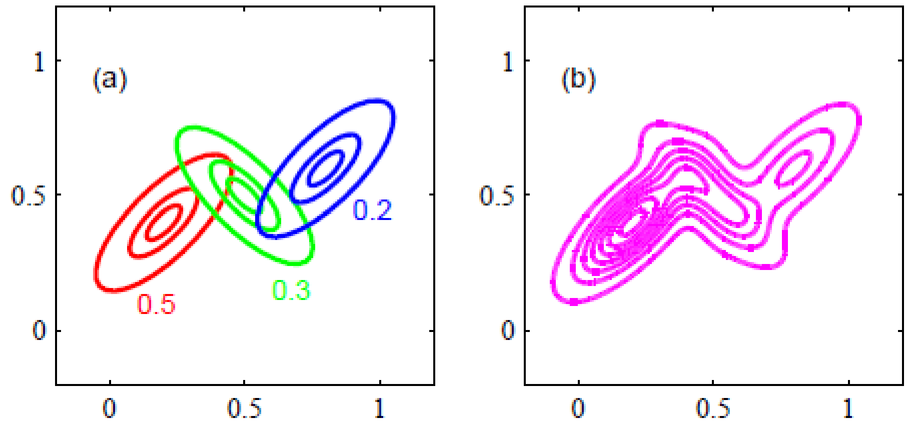
# Gaussian mixture model (GMM)

- Mixture of Gaussians: weighted sum of Gaussians

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k)$$

where $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma) = (2\pi)^{(-d/2)} |\Sigma|^{-1/2} \exp\left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$

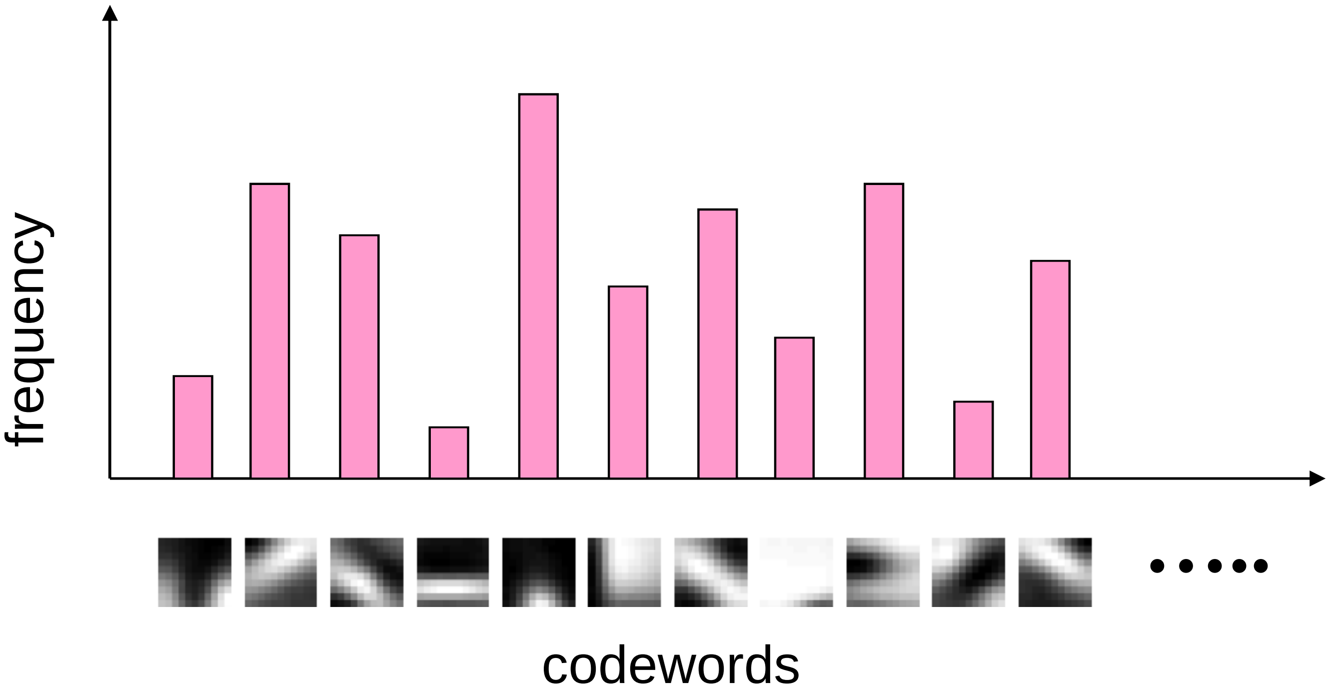


# Hard or soft assignment

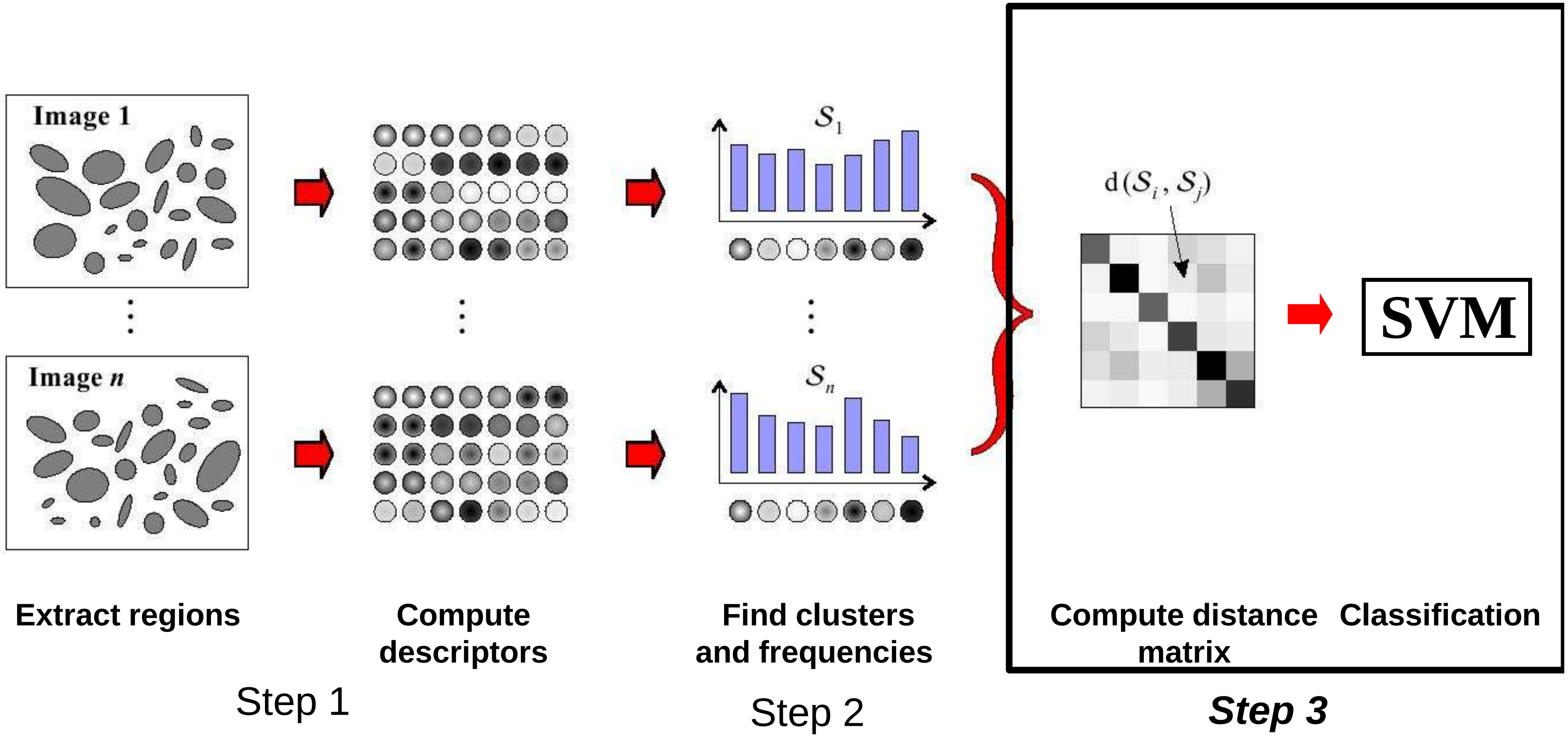
- K-means → hard assignment
  - Assign to the closest cluster center
  - Count number of descriptors assigned to a center
- Gaussian mixture model → soft assignment
  - Estimate distance to all centers
  - Sum over number of descriptors
- Represent image by a frequency histogram

# Image representation



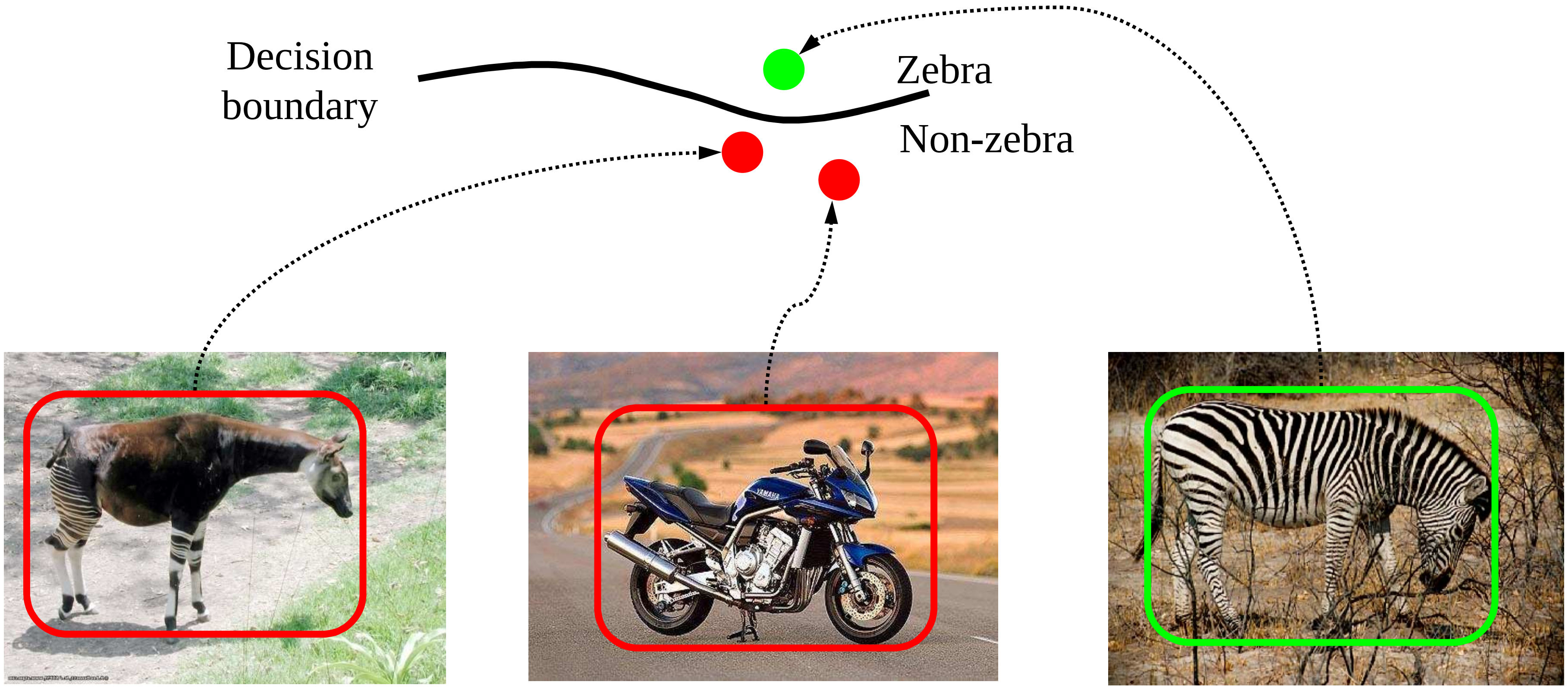

- Each image is represented by a vector, typically 1000-4000 dimension, normalization with L1 norm
- fine grained – represent model instances
- coarse grained – represent object categories

# Bag-of-features for image classification



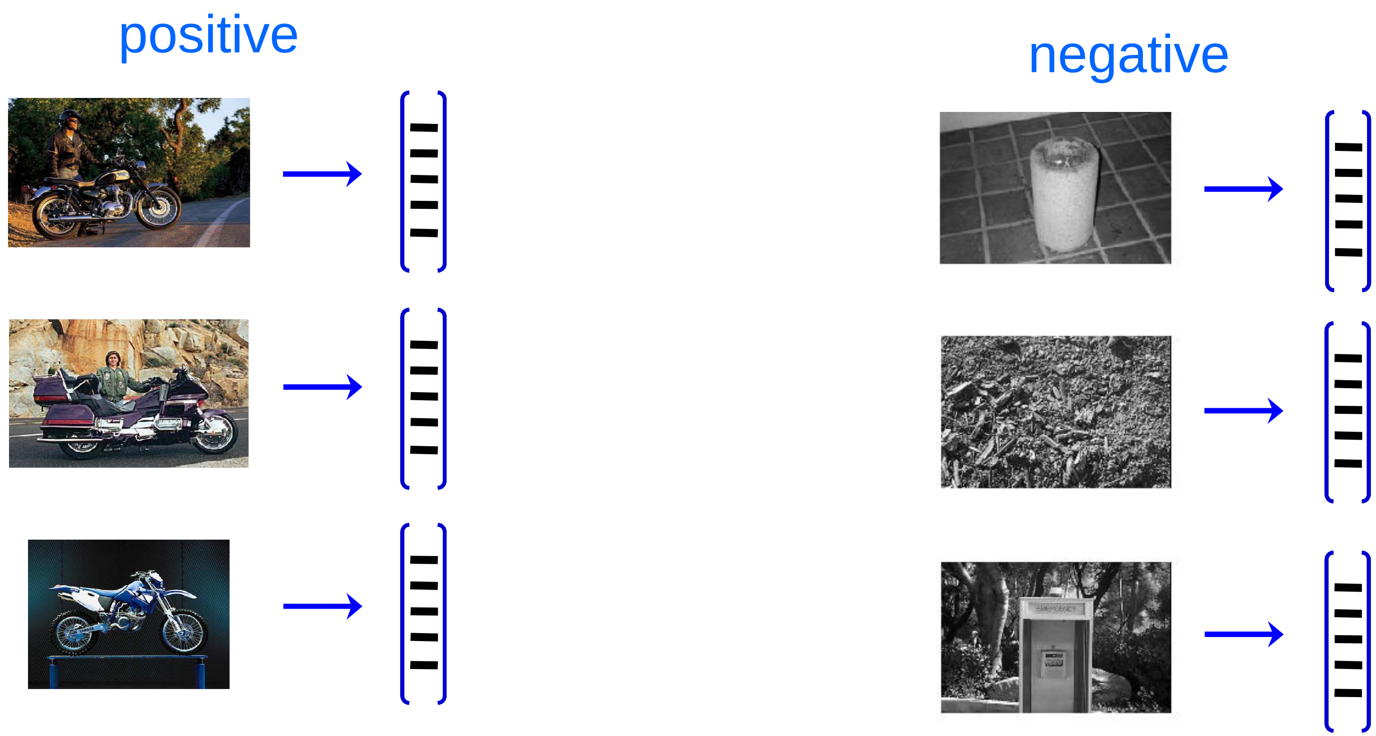
# Step 3: Classification

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



# Training data

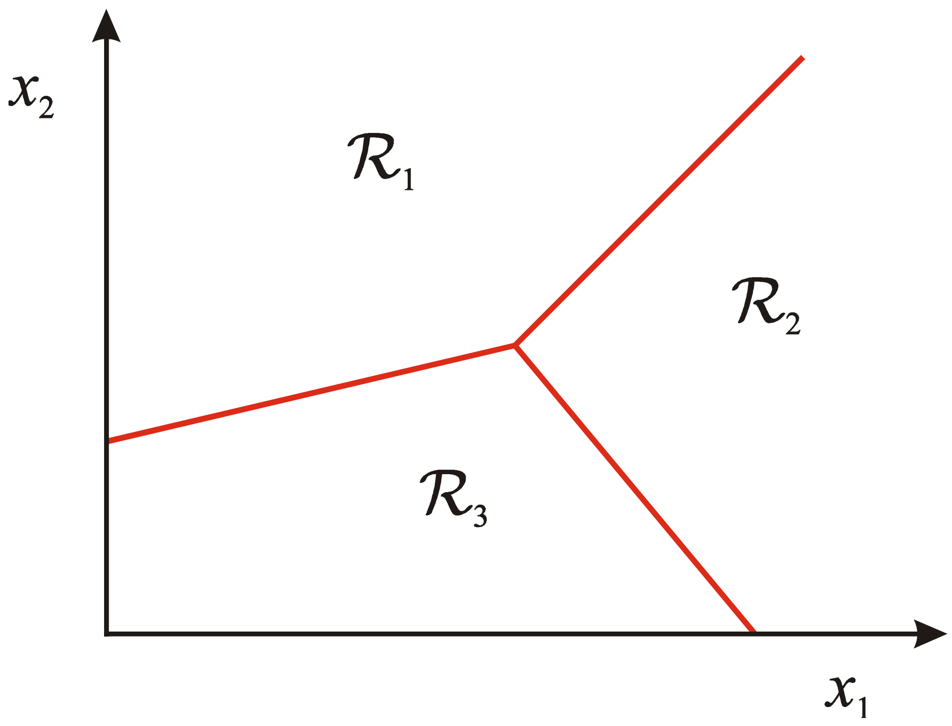
Vectors are histograms, one from each training image




Train classifier,e.g.SVM

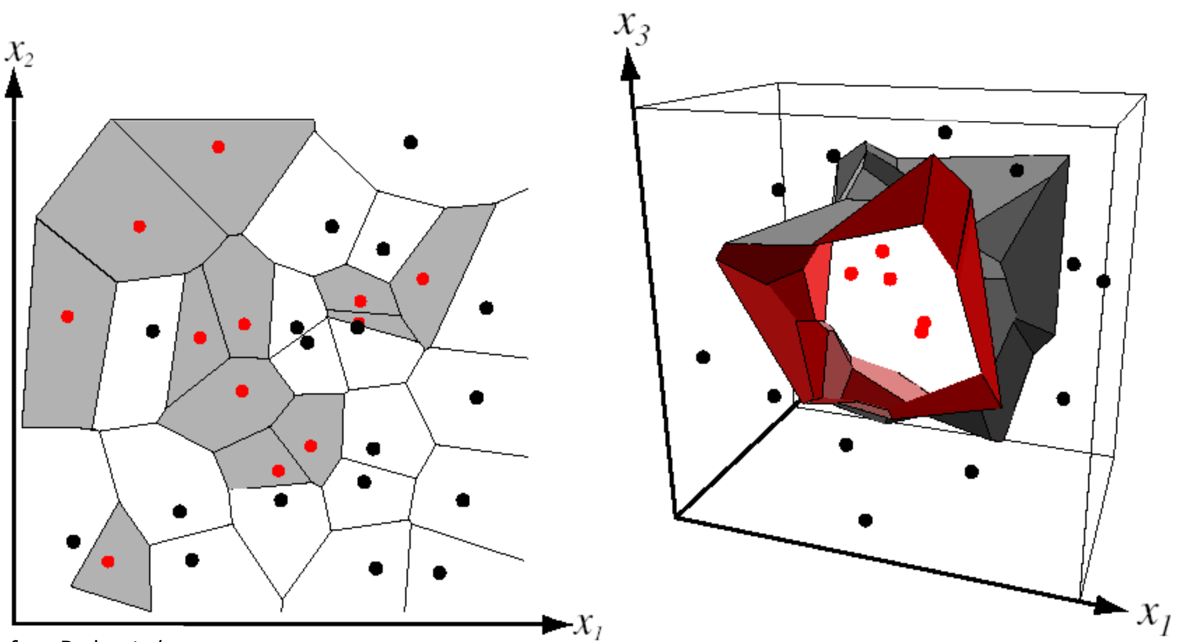
# Classification

- 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

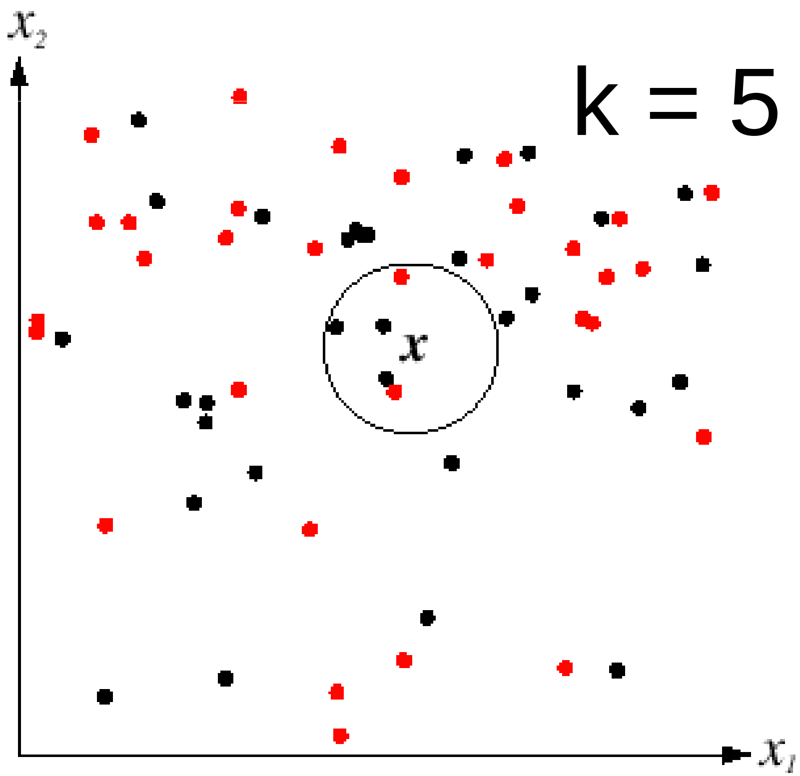



from Duda *et al.*

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

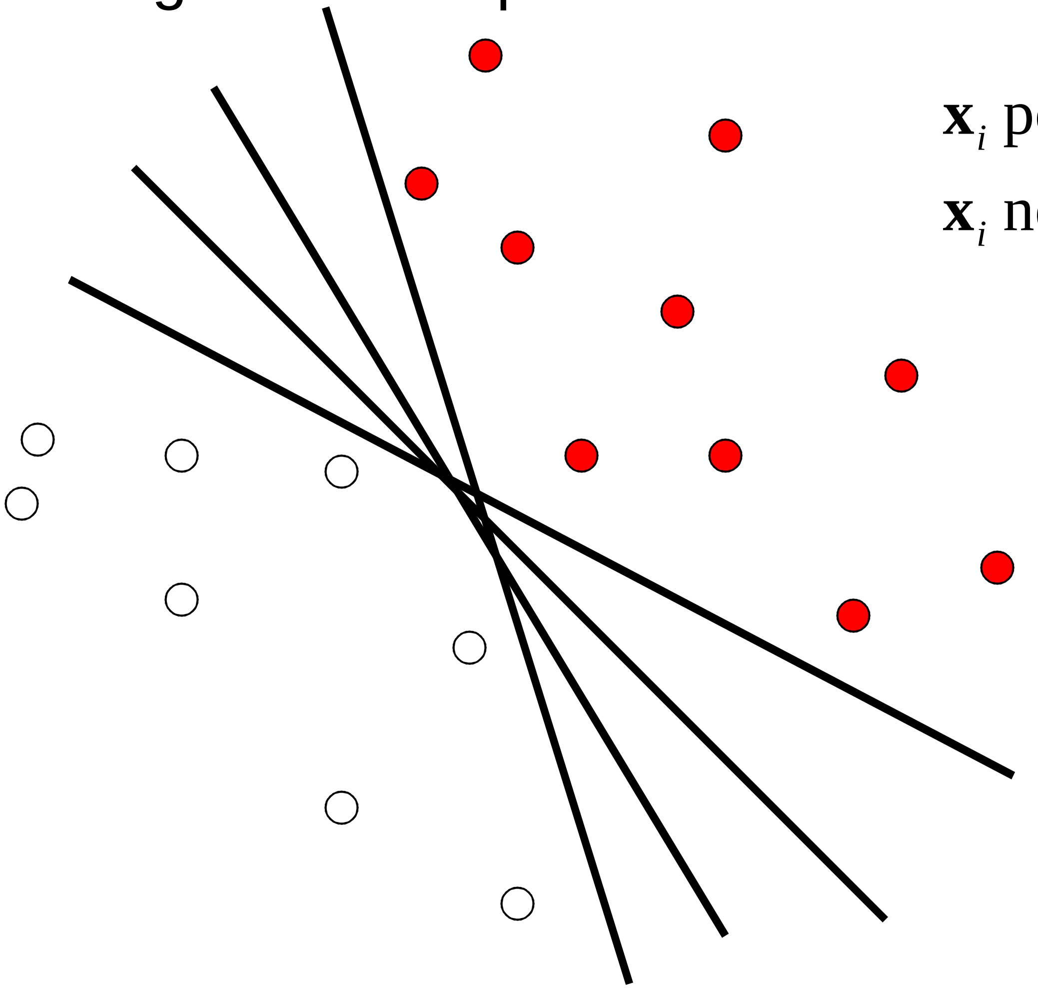
# k-Nearest Neighbors

- 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



# Linear classifiers

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



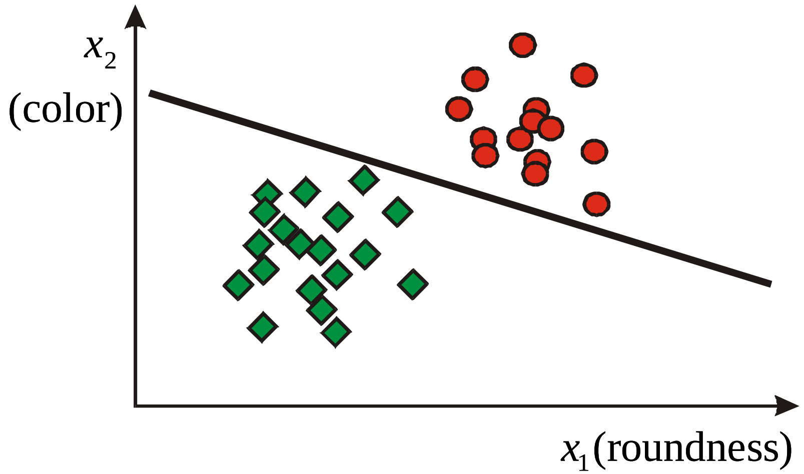
$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

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

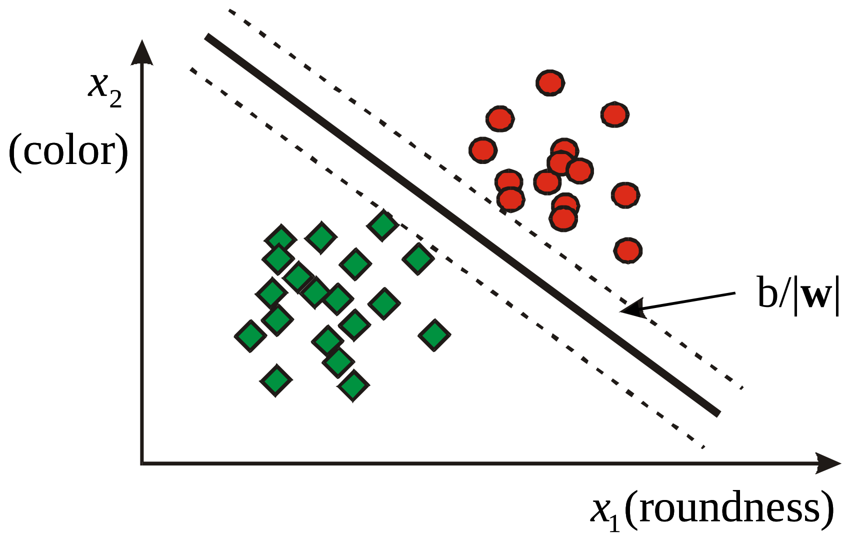
Which hyperplane
is best?

# Linear classifiers - margin

- Generalization is not good in this case:

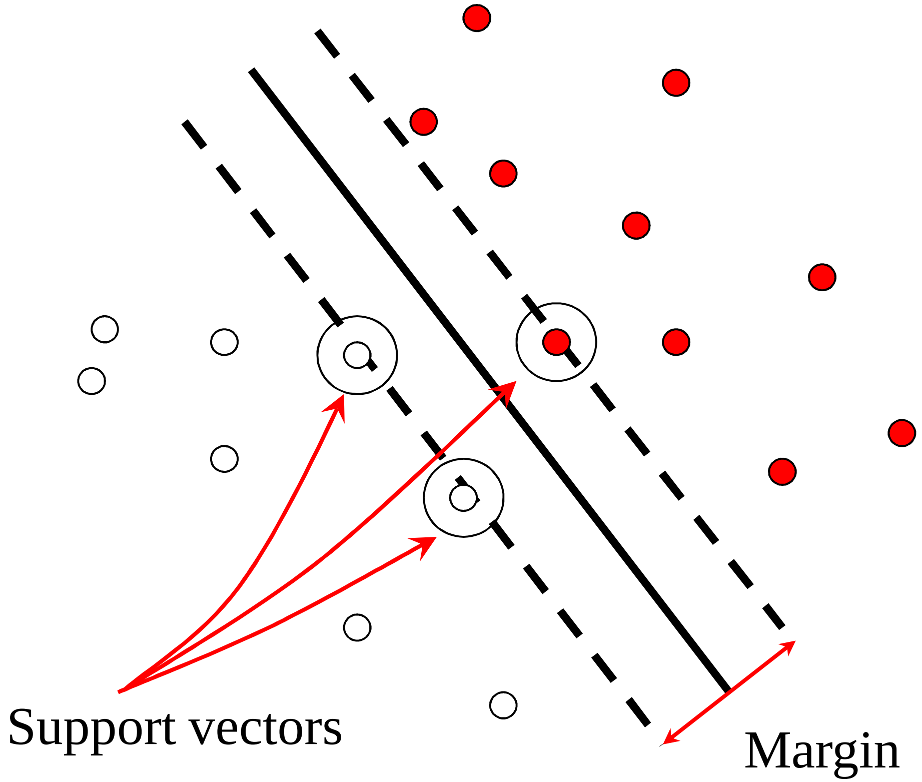



- Better if a margin is introduced:



# Support vector machines

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




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

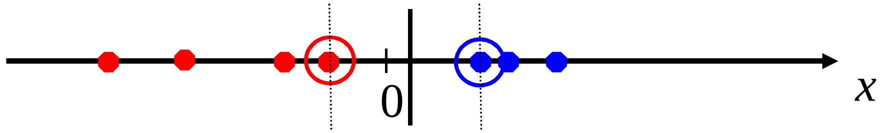
$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

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

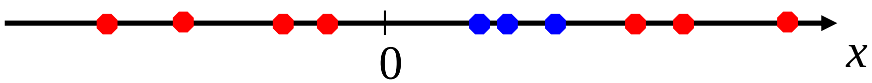
The margin is $2 \,/\, \|\mathbf{w}\|$

# Nonlinear SVMs

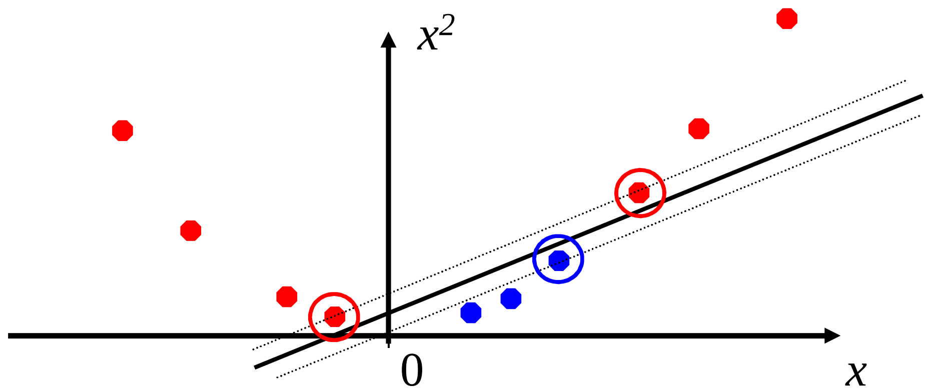
- Datasets that are linearly separable work out great:




- But what if the dataset is just too hard?

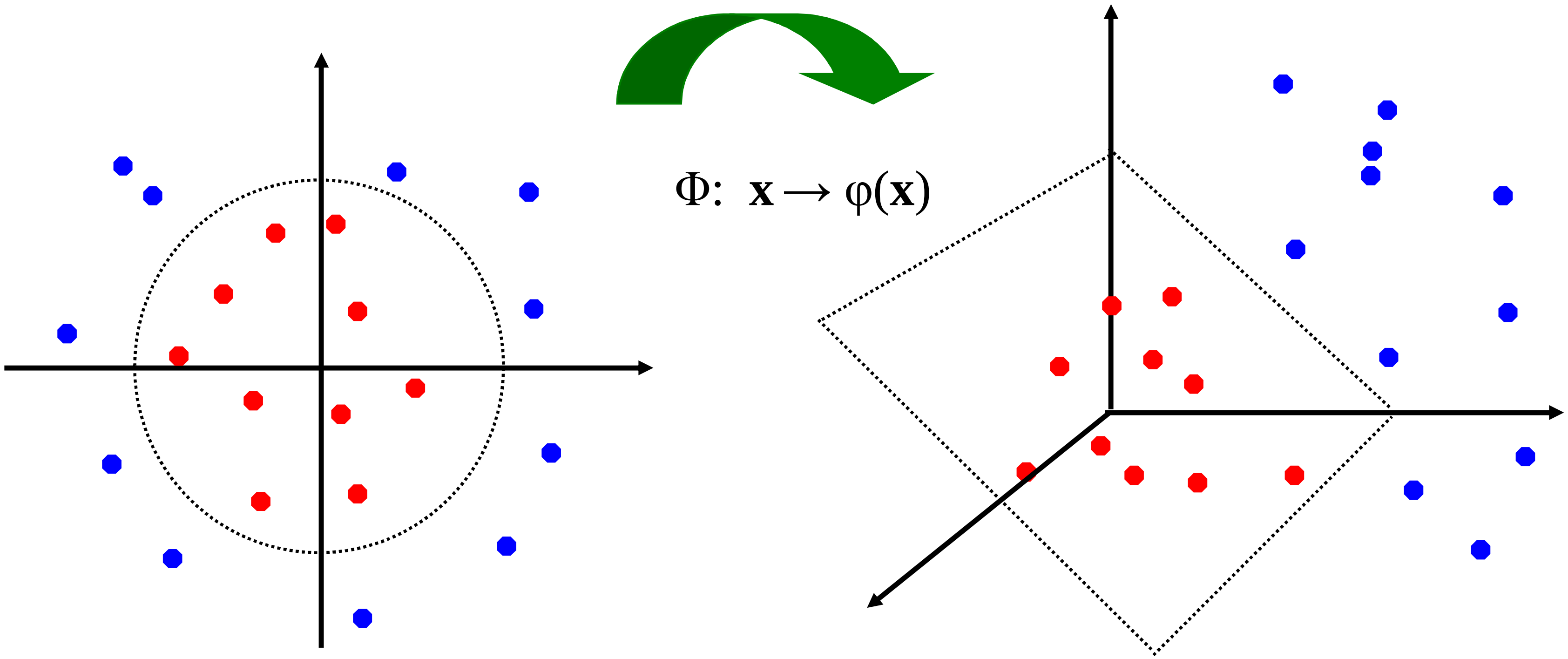



- We can map it to a higher-dimensional space:



# Nonlinear SVMs

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



# Nonlinear SVMs

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

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

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

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

# Kernels for bags of features

- Hellinger kernel $K(h_1, h_2) = \sum_{i=1}^{N} \sqrt{h_1(i) h_2(i)}$
- 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, $\chi^2$ distance etc.

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

# Combining features

- SVM with multi-channel chi-square kernel

$$K(H_i, H_j) = \exp\Big(-\sum_{c\in\mathcal{C}} \frac{1}{A_c} D_c(H_i, H_j)\Big)$$

- Channel *c* is a combination of detector, descriptor
- $D_c(H_i, H_j)$ is the chi-square distance between histograms

$$D_c(H_1, H_2) = \frac{1}{2}\sum_{i=1}^{m}[(h_{1i} - h_{2i})^2 / (h_{1i} + h_{2i})]$$

- $A_c$ is the mean value of the distances between all training sample
- Extension: learning of the weights, for example with Multiple Kernel Learning (MKL)

J. Zhang, M. Marszalek, S. Lazebnik and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study, IJCV 2007.

# Multi-class SVMs

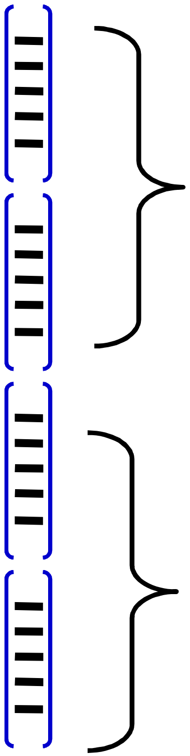
- Various direct formulations exist, but they are not widely used in practice. It is more common to obtain multi-class SVMs by combining two-class SVMs in various ways.

- One versus all:
  - Training: learn an SVM for each class versus 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 versus one:
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM "votes" for a class to assign to the test example

# Why does SVM learning work?

- Learns foreground and background visual words

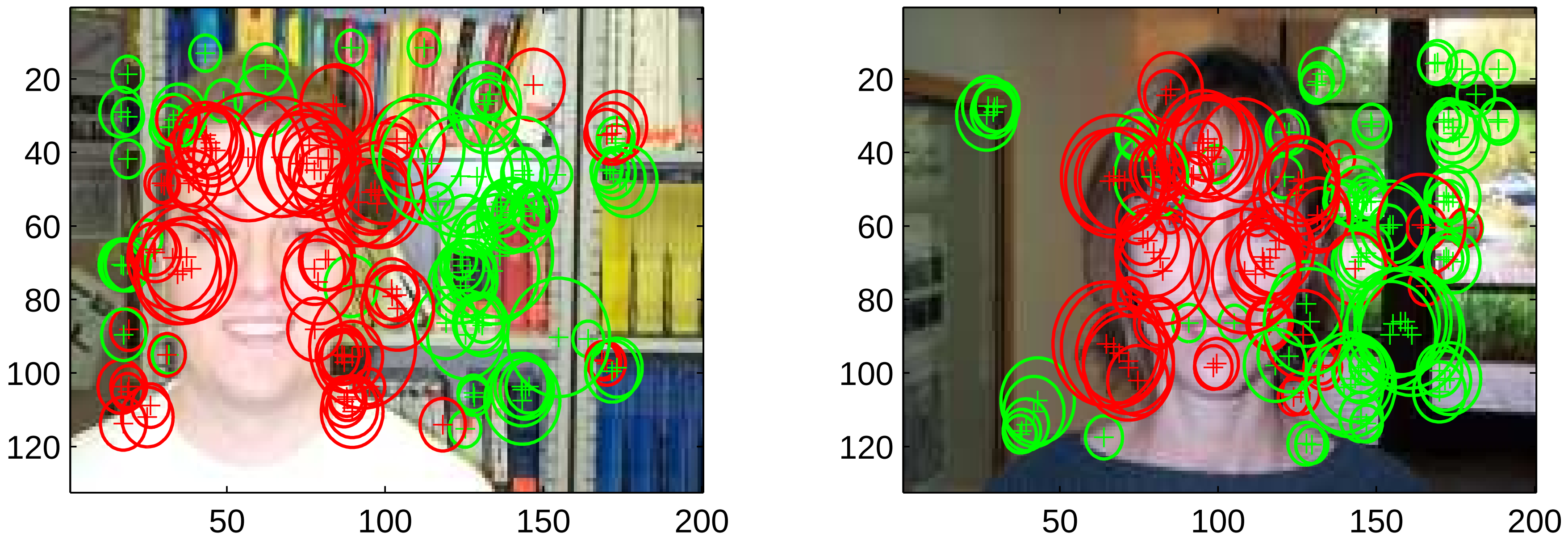


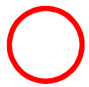
foreground words – high weight

background words – low weight

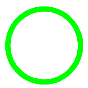
# Illustration

## Localization according to visual word probability



foreground word more probable

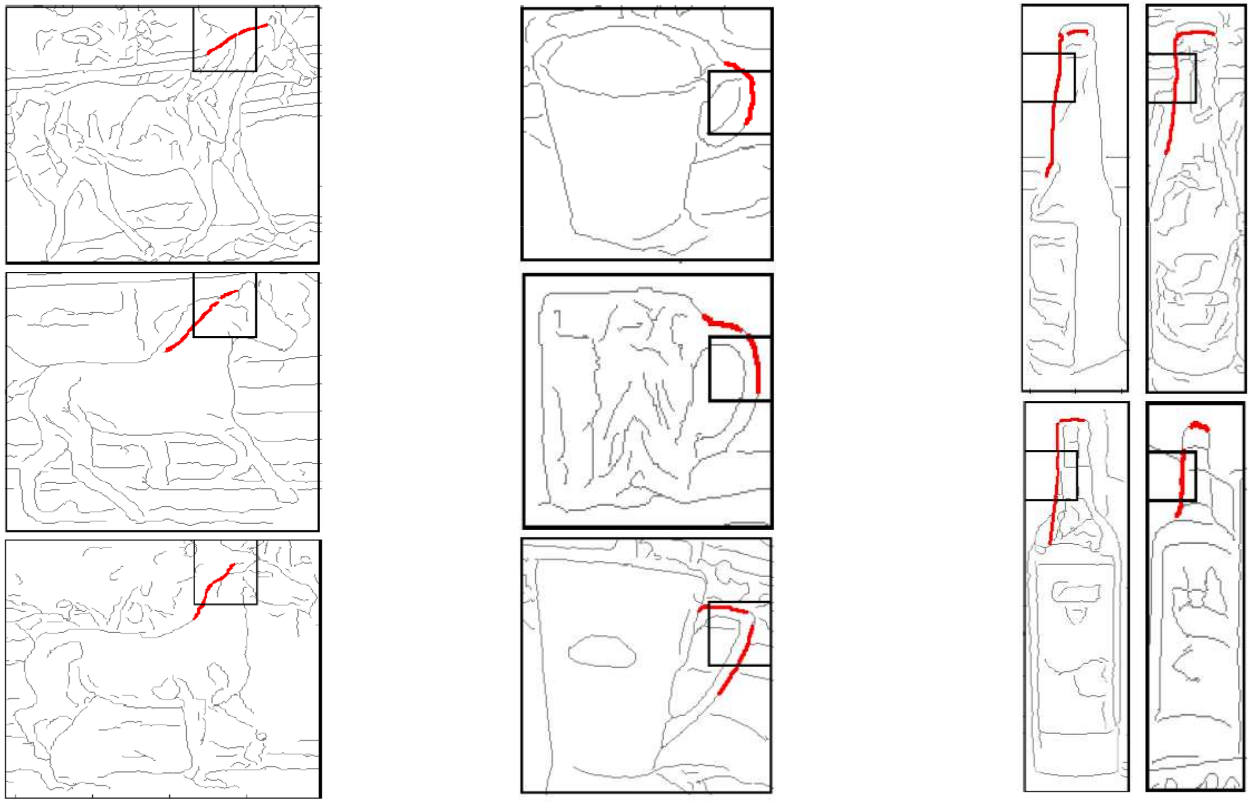


background word more probable

# Illustration

A linear SVM trained from positive and negative window descriptors

A few of the highest weighed descriptor vector dimensions (= 'PAS + tile')



+ lie on object boundary (= local shape structures common to many training exemplars)

# Bag-of-features for image classification

- Excellent results in the presence of background clutter

bikes books building cars people phones trees

# Examples for misclassified images

Books- misclassified into faces, faces, buildings

Buildings- misclassified into faces, trees, trees

Cars- misclassified into buildings, phones, phones

# Bag of visual words summary

- Advantages:
  - largely unaffected by position and orientation of object in image
  - fixed length vector irrespective of number of detections
  - very successful in classifying images according to the objects they contain

- Disadvantages:
  - no explicit use of configuration of visual word positions
  - poor at localizing objects within an image

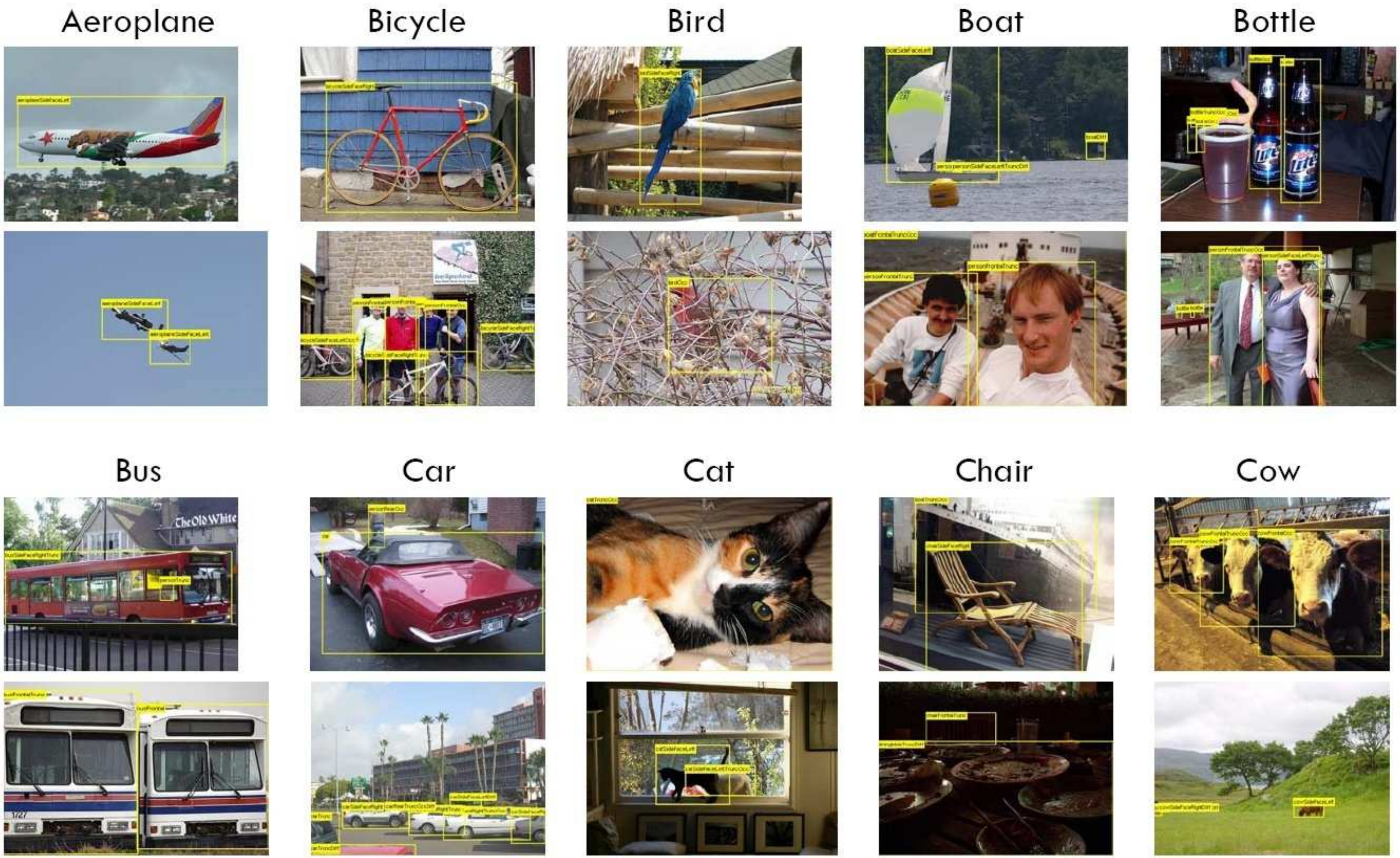
# Evaluation of image classification

- PASCAL VOC [05-10] datasets

- PASCAL VOC 2007
  - Training *and* test dataset available
  - Used to report state-of-the-art results
  - Collected January 2007 from Flickr
  - 500 000 images downloaded and random subset selected
  - 20 classes
  - Class labels per image + bounding boxes
  - 5011 training images, 4952 test images

- Evaluation measure: average precision

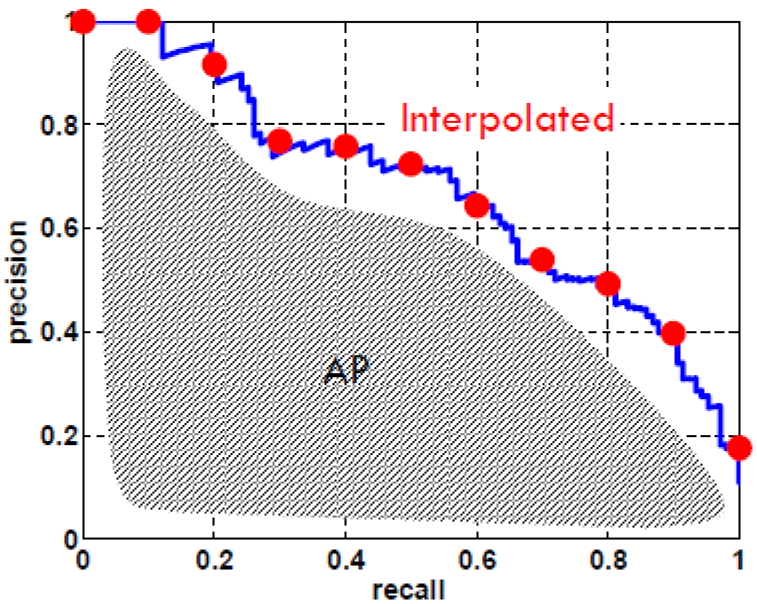
# PASCAL 2007 dataset



# PASCAL 2007 dataset

# Evaluation

- Average Precision [TREC] averages precision over the entire range of recall
  - Curve interpolated to reduce influence of “outliers”




- A good score requires both high recall and high precision
- Application-independent
- Penalizes methods giving high precision but low recall

# Results for PASCAL 2007

- Winner of PASCAL 2007 [Marszalek et al.] : mAP 59.4
  - Combination of several different channels (dense + interest points, SIFT + color descriptors, spatial grids)
  - Non-linear SVM with Gaussian kernel
- Multiple kernel learning [Yang et al. 2009] : mAP 62.2
  - Combination of several features
  - Group-based MKL approach
- Combining object localization and classification [Harzallah et al.'09] : mAP 63.5
  - Use detection results to improve classification

# Comparison interest point - dense

**Image classification** results on PASCAL'07 train/val set

| | AP |
|---|---|
| (SHarris + Lap) x SIFT | 0.452 |
| MSDense x SIFT | 0.489 |
| (SHarris + Lap + MSDense) x SIFT | 0.515 |

Method: bag-of-features + SVM classifier

# Comparison interest point - dense

**Image classification** results on PASCAL'07 train/val set

| | AP |
|---|---|
| (SHarris + Lap) x SIFT | 0.452 |
| MSDense x SIFT | 0.489 |
| (SHarris + Lap + MSDense) x SIFT | 0.515 |

Dense is on average a bit better!

IP and dense are complementary, combination improves results.

# Comparison interest point - dense

**Image classification** results on PASCAL'07 train/val set for individual categories

| | (SHarris + Lap) x SIFT | MSDense x SIFT |
|---|---|---|
| Bicycle | **0.534** | 0.443 |
| PottedPlant | **0.234** | 0.167 |
| Bird | 0.342 | **0.497** |
| Boat | 0.482 | **0.622** |

Results are category dependent!

# Evaluation BoF – spatial

**Image classification** results on PASCAL'07 train/val set

| (SH, Lap, MSD) x (SIFT,SIFTC) spatial layout | AP |
|---|---|
| 1 | 0.53 |
| 2x2 | 0.52 |
| 3x1 | 0.52 |
| 1,2x2,3x1 | 0.54 |

Spatial layout not dominant for PASCAL'07 dataset

Combination improves average results, i.e., it is appropriate for some classes

# Evaluation BoF - spatial

Image classification results on PASCAL'07 train/val set for individual categories

| | 1 | 3x1 |
|---|---|---|
| Sheep | **0.339** | 0.256 |
| Bird | **0.539** | 0.484 |
| DiningTable | 0.455 | **0.502** |
| Train | 0.724 | **0.745** |

Results are category dependent!

➔ Combination helps somewhat

# Spatial pyramid matching

- Add spatial information to the bag-of-features

- Perform matching in 2D image space

[Lazebnik, Schmid & Ponce, CVPR 2006]

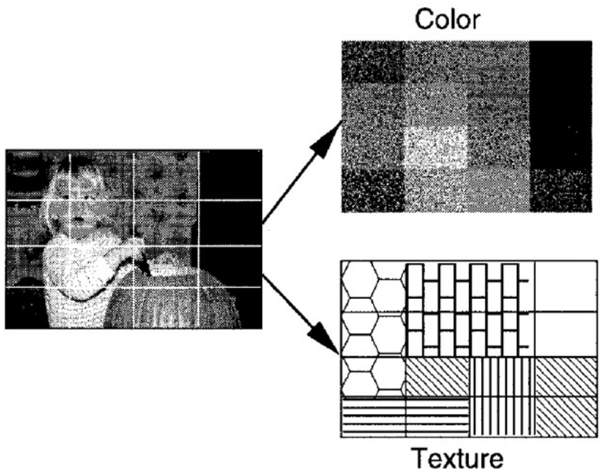
# Related work

Similar approaches:

Subblock description [Szummer & Picard, 1997]

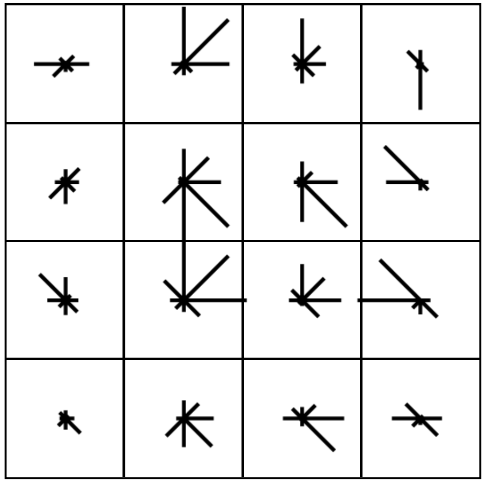
SIFT [Lowe, 1999]

GIST [Torralba et al., 2003]




Szummer & Picard (1997)

SIFT

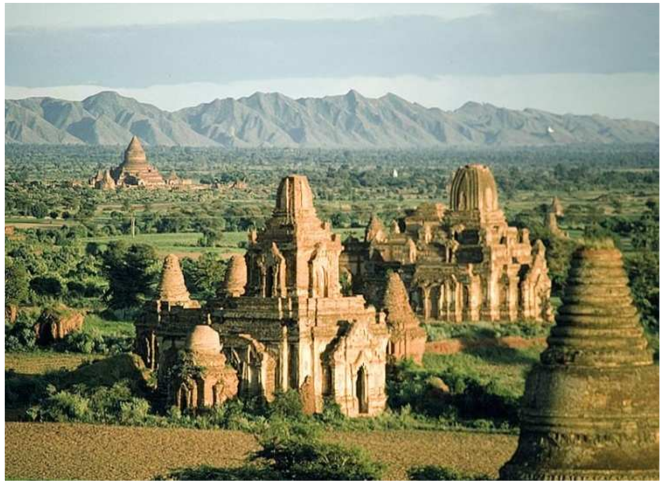


Lowe (1999, 2004)

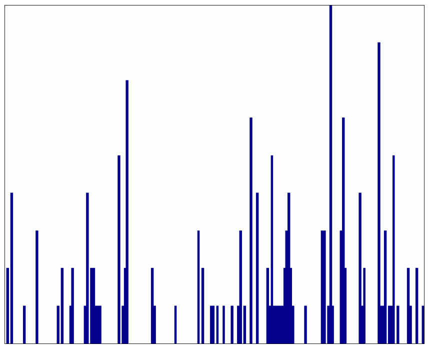
Gist

Torralba et al. (2003)

# Spatial pyramid representation

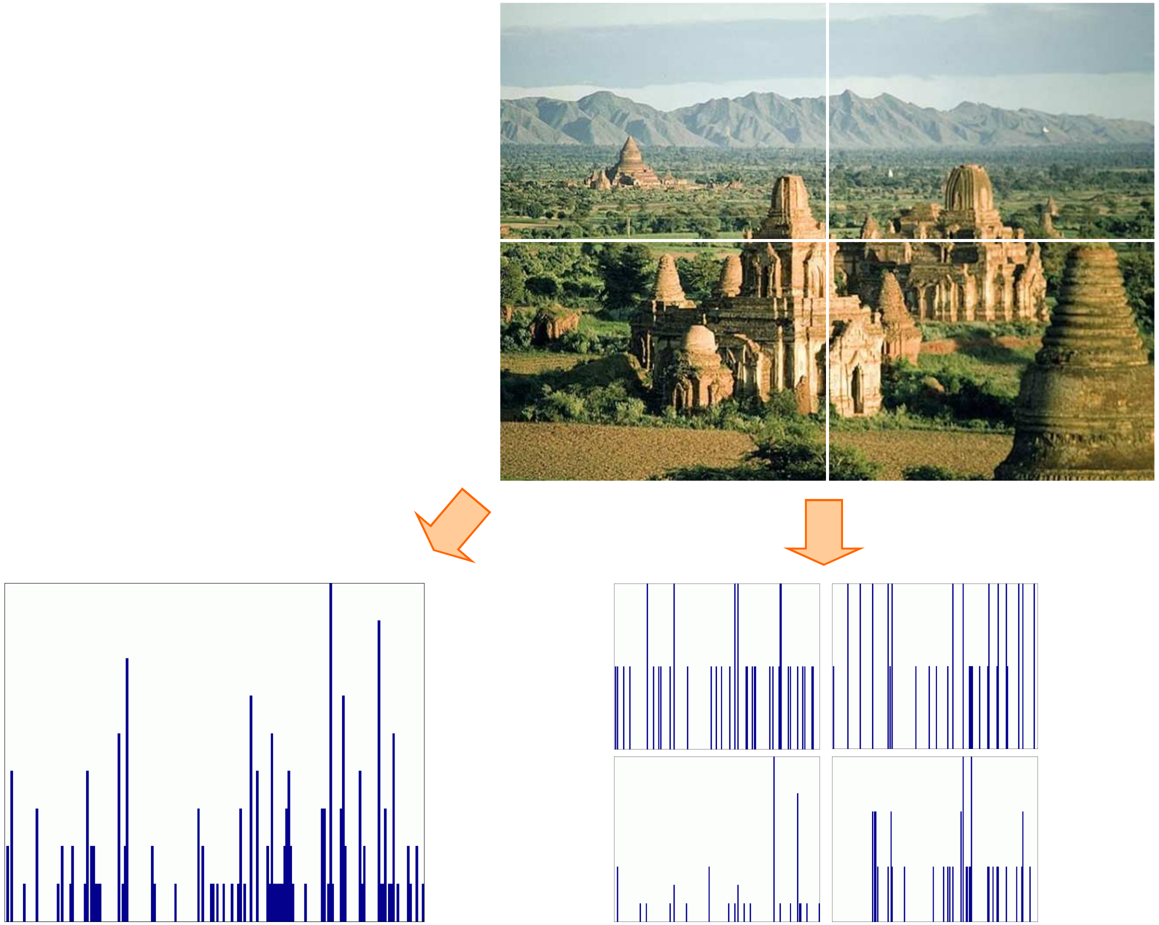


Locally orderless representation at several levels of spatial resolution



level 0

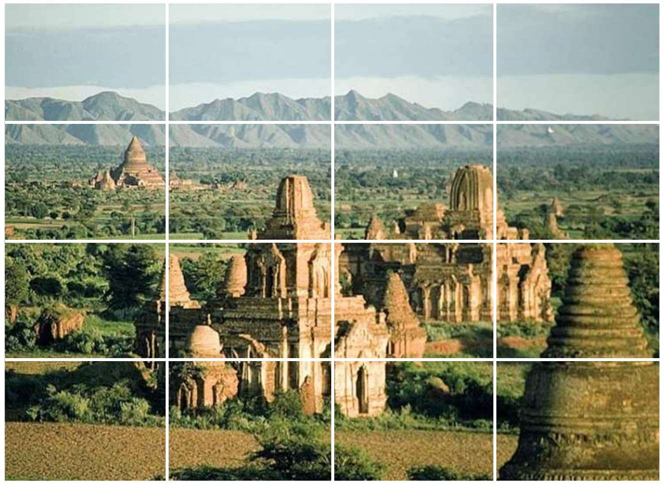
# Spatial pyramid representation



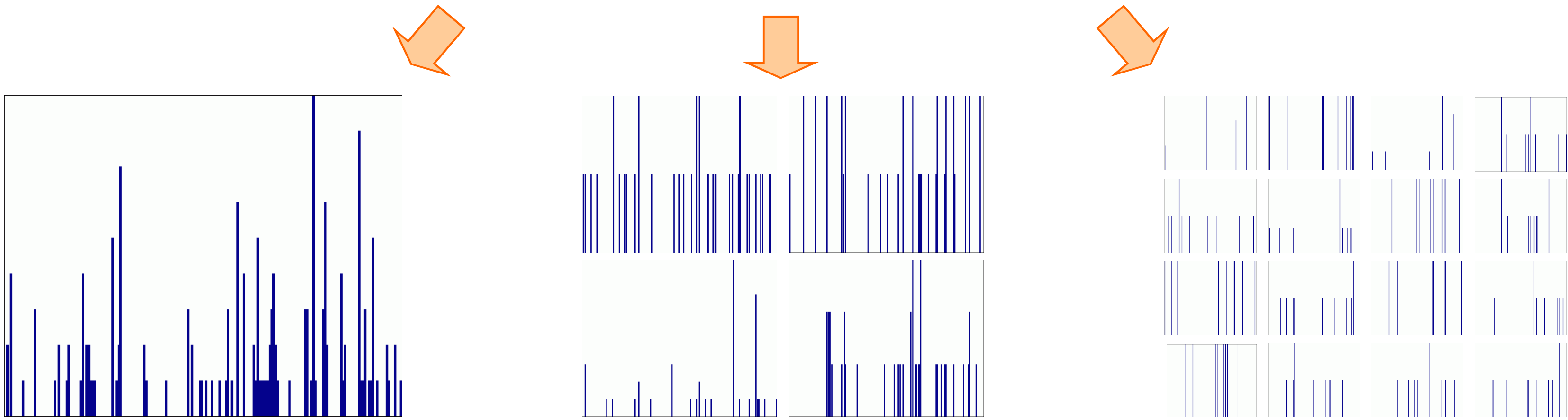
level 0 level 1

Locally orderless representation at several levels of spatial resolution

# Spatial pyramid representation



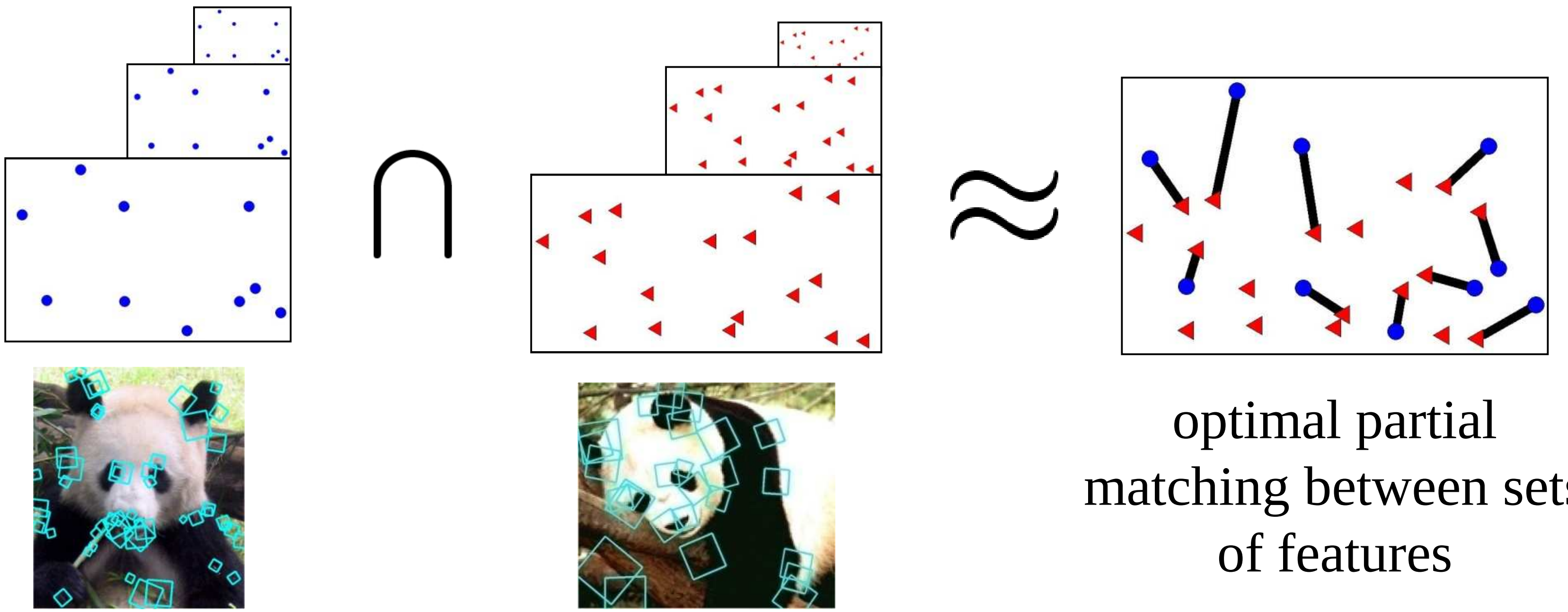
Locally orderless representation at several levels of spatial resolution



level 0 level 1 level 2

# Pyramid match kernel

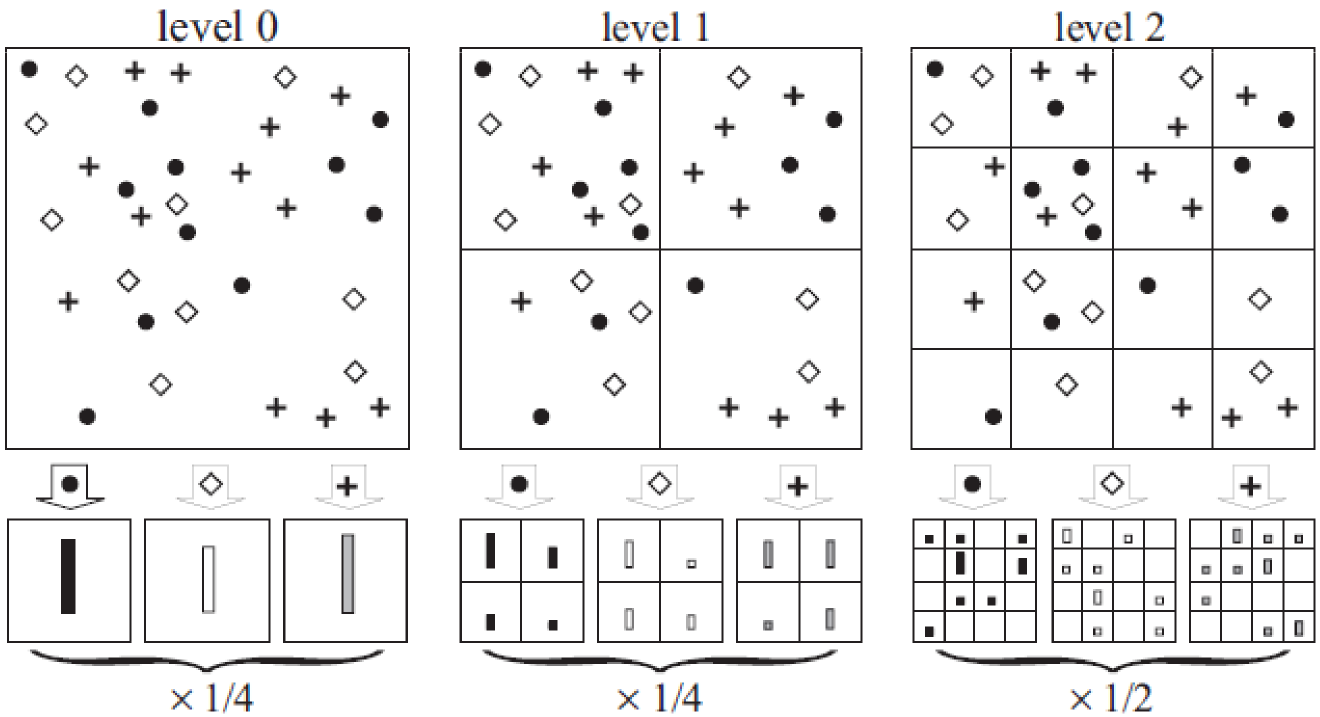
- Weighted sum of histogram intersections at multiple resolutions (linear in the number of features instead of cubic)



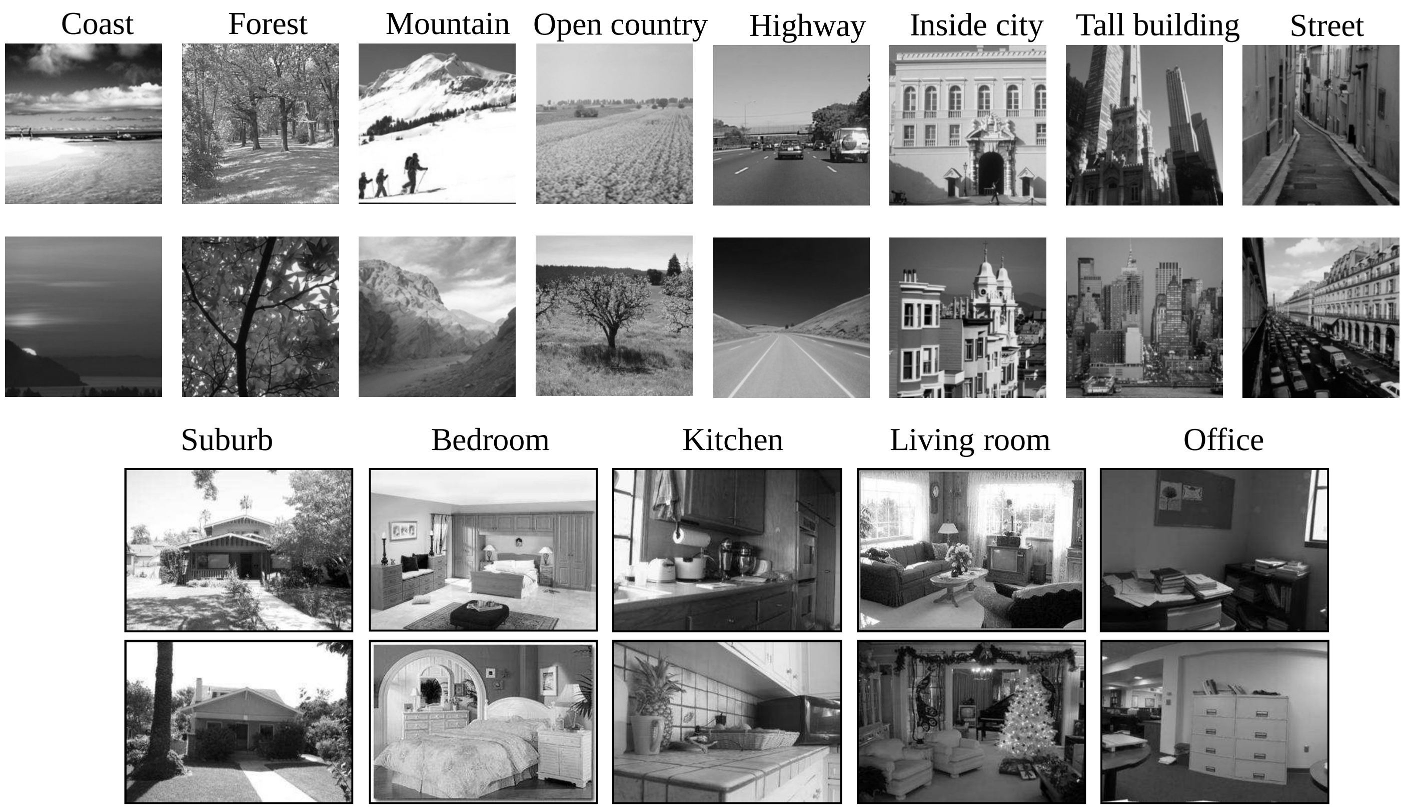
optimal partial matching between sets of features

# Spatial pyramid matching

- Combination of spatial levels with pyramid match kernel
  [Grauman & Darell'05]
- Intersect histograms, more weight to finer grids



# Scene dataset [Labzenik et al.'06]




4385 images

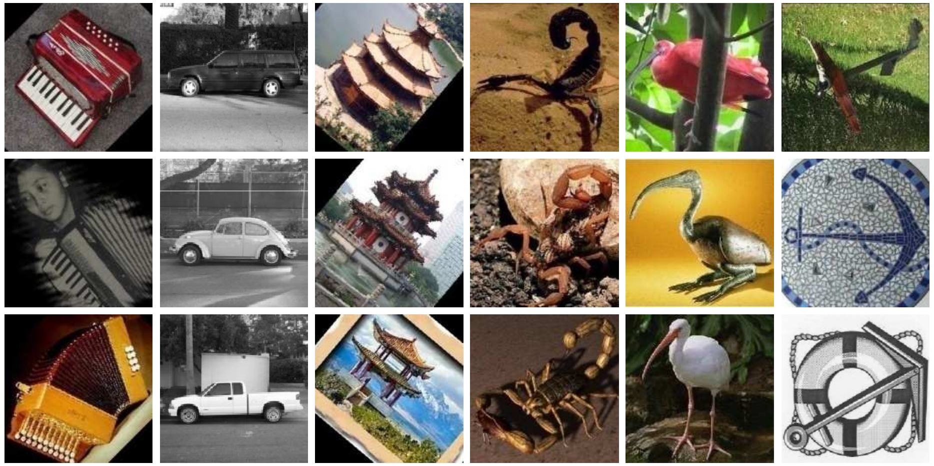
15 categories

# Scene classification


| L | Single-level | Pyramid |
|---|---|---|
| 0(1x1) | 72.2±0.6 | |
| 1(2x2) | 77.9±0.6 | 79.0 ±0.5 |
| 2(4x4) | 79.4±0.3 | *81.1 ±0.3* |
| 3(8x8) | 77.2±0.4 | 80.7 ±0.3 |

# Retrieval examples

# Category classification – CalTech101

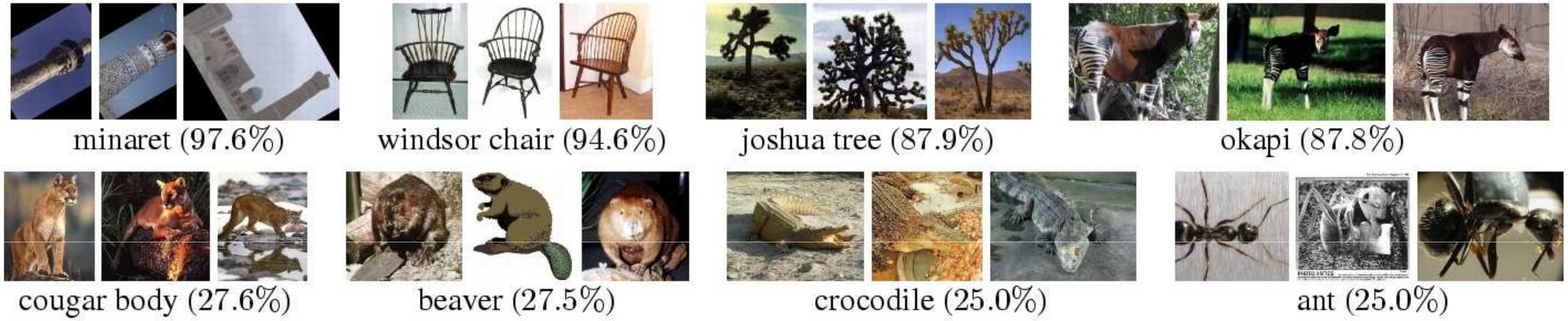


| L | Single-level | Pyramid |
|---|---|---|
| **0(1x1)** | **41.2±1.2** | |
| **1(2x2)** | **55.9±0.9** | **57.0 ±0.8** |
| **2(4x4)** | **63.6±0.9** | ***64.6 ±0.8*** |
| **3(8x8)** | **60.3±0.9** | **64.6 ±0.7** |

Bag-of-features approach by Zhang et al.'07: 54 %

# CalTech101

## Easiest and hardest classes



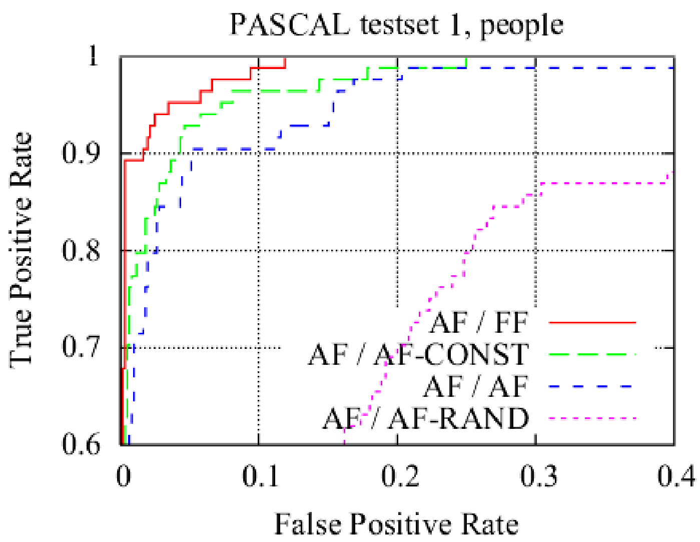

- Sources of difficulty:
  - Lack of texture
  - Camouflage
  - Thin, articulated limbs
  - Highly deformable shape

# Discussion

- Summary
  - Spatial pyramid representation: appearance of local image patches + coarse global position information
  - Substantial improvement over bag of features
  - Depends on the similarity of  image layout
- Extensions
  - Flexible, object-centered grid

# Motivation

- Evaluating the influence of background features [J. Zhang et al., IJCV'07]
  - Train and test on different combinations of foreground and background by separating features based on bounding boxes




*Training*: original training set

*Testing*: different combinations foreground + background features

Best results when testing with foreground features only

# Approach

- Better to train on a “harder” dataset with background clutter and test on an easier one without background clutter
- Spatial weighting for bag-of-features [Marszalek & Schmid, CVPR’06]
  - weight features by the likelihood of belonging to the object
  - determine likelihood based on shape masks

# Masks for spatial weighting

For each test feature:

- Select closest training features + corresponding masks
(training requires segmented images or bounding boxes)

- Align mask based on local co-ordinates system
(transformation between training and test co-ordinate systems)

Sum masks weighted by matching distance

three features agree on object localization,
the object has higher weights

Weight histogram features with the strength of the final mask

# Example masks for spatial weighting

# Classification for PASCAL dataset

| | Zhang et al. | Spatial weighting | Gain |
|---|---|---|---|
| bikes | 74.8 | 76.8 | +2.0 |
| cars | 75.8 | 76.8 | +1.0 |
| motorbikes | 78.8 | 79.3 | +0.5 |
| people | 76.9 | 77.9 | +1.0 |

Equal error rates for PASCAL test set 2

# Discussion

- Including spatial information improves results
- Importance of flexible modeling of spatial information
  - coarse global position information
  - object based models

# Recent extensions

- Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification. J. Yang et al., CVPR'09.
  - Local coordinate coding,  linear SVM, excellent results in 2009 PASCAL challenge
- Learning Mid-level features for recognition, Y. Boureau et al., CVPR'10.
  - Use of sparse coding techniques and max pooling

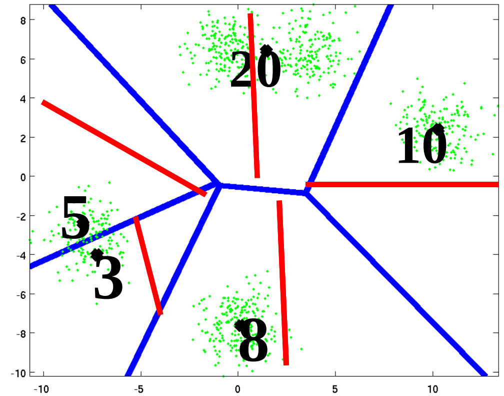
# Recent extensions

- Efficient Additive Kernels via Explicit Feature Maps, A. Vedaldi and Zisserman, CVPR'10.
  - approximation by linear kernels

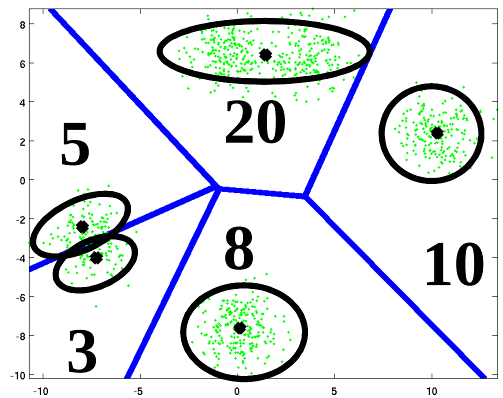
- Improving the Fisher Kernel for Large-Scale Image Classification, Perronnin et al., ECCV'10
  - More discriminative descriptor, power normalization, linear SVM

# Fisher vector image representation

- Mixture of Gaussian/ k-means stores nr of points per cell




- Fisher vector adds 1st & 2nd order moments
  - More precise description of regions assigned to cluster
  - Fewer clusters needed for same accuracy
  - Per cluster also store: mean and variance of data in cell



# Fisher vector image representation

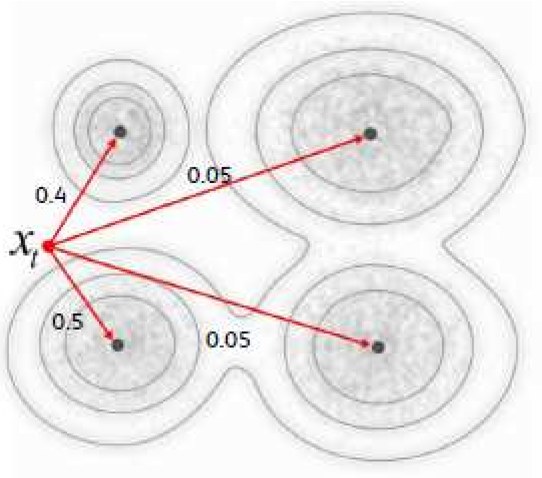
$X = \{x_t, t = 1 \ldots T\}$ is the set of T i.i.d. D-dim local descriptors (e.g. SIFT) extracted from an image:

$u_\lambda(x) = \sum_{i=1}^{K} w_i u_i(x)$ is a Gaussian Mixture Model (GMM) with parameters $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1 \ldots N\}$ trained on a large set of local descriptors: a **visual vocabulary**

FV formulas:

$$\mathcal{G}_{\mu,i}^{X} = \frac{1}{T\sqrt{w_i}} \sum_{t=1}^{T} \gamma_t(i) \left( \frac{x_t - \mu_i}{\sigma_i} \right)$$

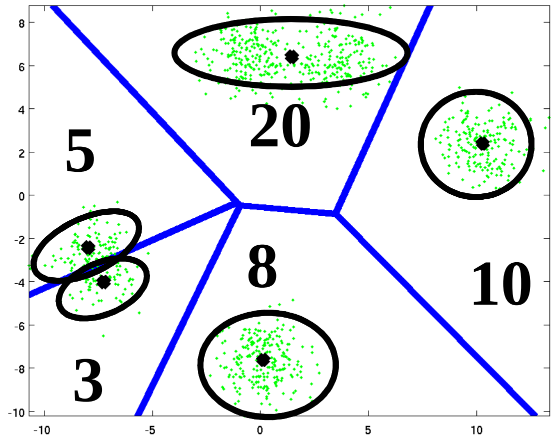
$$\mathcal{G}_{\sigma,i}^{X} = \frac{1}{T\sqrt{2w_i}} \sum_{t=1}^{T} \gamma_t(i) \left[ \frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right]$$




$\gamma_t(i)$ = soft-assignment of patch $x_t$ to Gaussian i

# Fisher vector image representation

- Fischer vector adds 1st & 2nd order moments
  - More precise description regions assigned to cluster
  - Fewer clusters needed for same accuracy
  - Representation 2D times larger, at same computational cost
  - High dimensional, robust representation



# Relation to BOF

FV formulas:

$$\mathcal{G}^X_{\mu,i} = \frac{1}{T\sqrt{w_i}} \sum_{t=1}^{T} \gamma_t(i) \left( \frac{x_t - \mu_i}{\sigma_i} \right)$$

$$\mathcal{G}^X_{\sigma,i} = \frac{1}{T\sqrt{2w_i}} \sum_{t=1}^{T} \gamma_t(i) \left[ \frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right]$$

Soft BOV formula:

$\frac{1}{T}\sum_{t=1}^{T} \gamma_t(i)$

Like the (original) BOV the FV is an average of local statistics.

The FV extends the BOV and includes higher-order statistics (up to 2nd order)

Results on VOC 2007: BOV = 43.6 % $\rightarrow$ FV = 57.7 % $\rightarrow$ √FV = 62.1 %