Bag-of-features
for category recognition

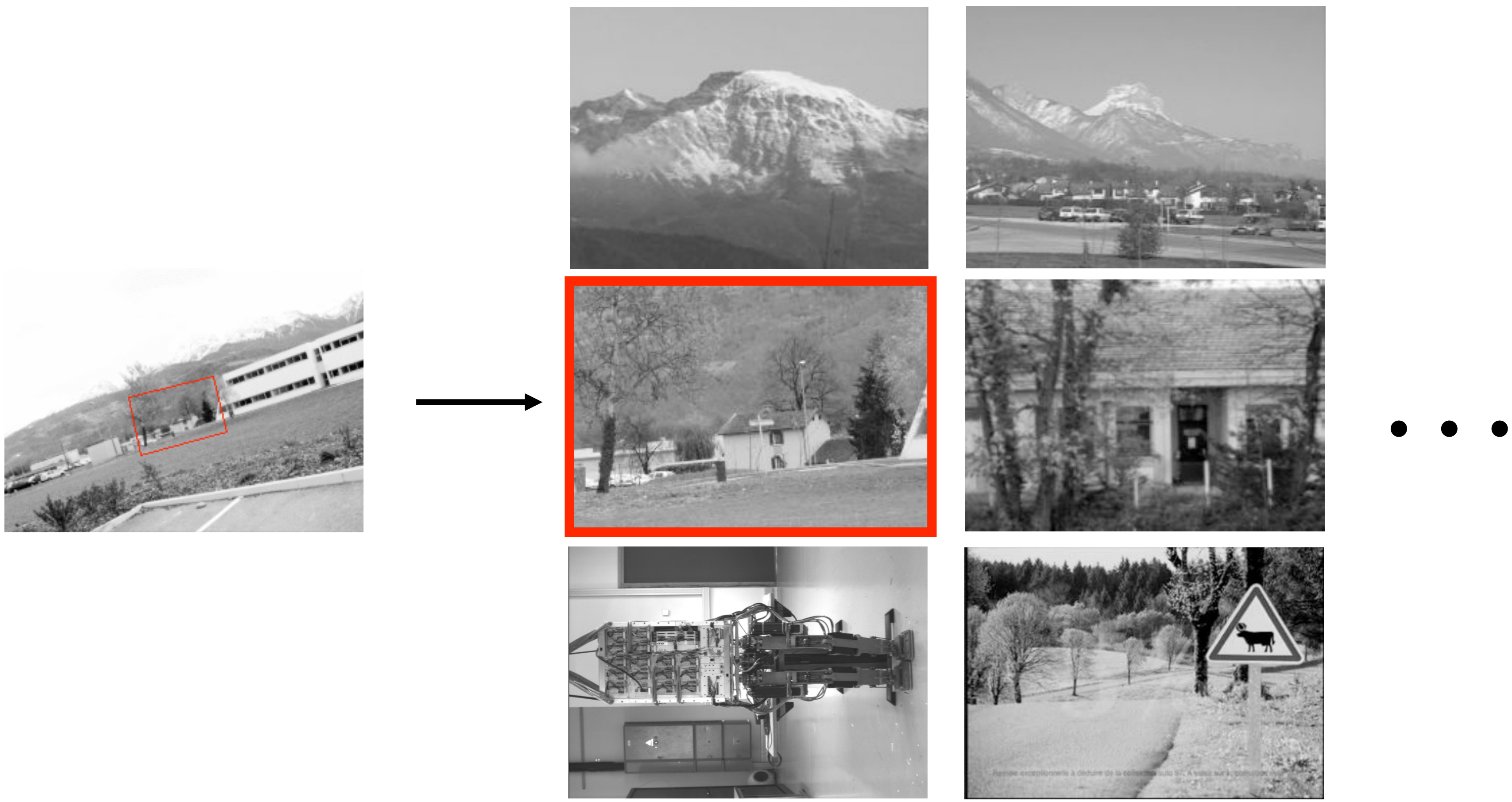
Cordelia Schmid

INRIA

LEAR

Visual search

- Particular objects and scenes, large databases



Category recognition

- Image classification: assigning a class label to the image

Car: present

Cow: present

Bike: not present

Horse: not present

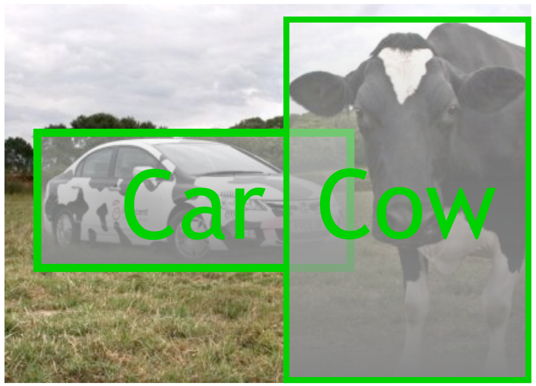
...

Category recognition

- Image classification: assigning a class label to the image

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

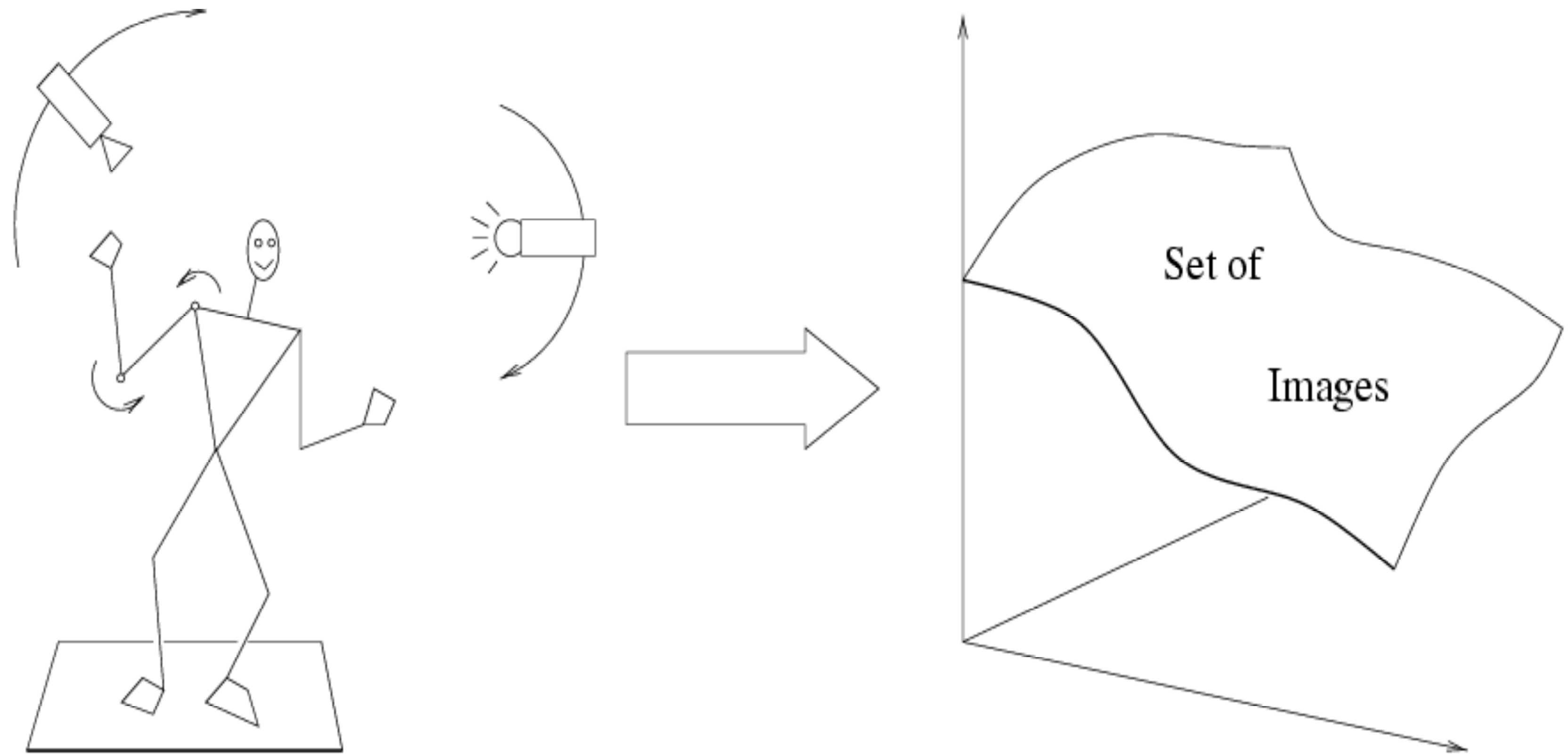
- Object localization: define the location and the category



Location

Category

Difficulties: within object variations



Variability: Camera position, Illumination,Internal parameters

⇨ Within-object variations

Difficulties: within-class variations

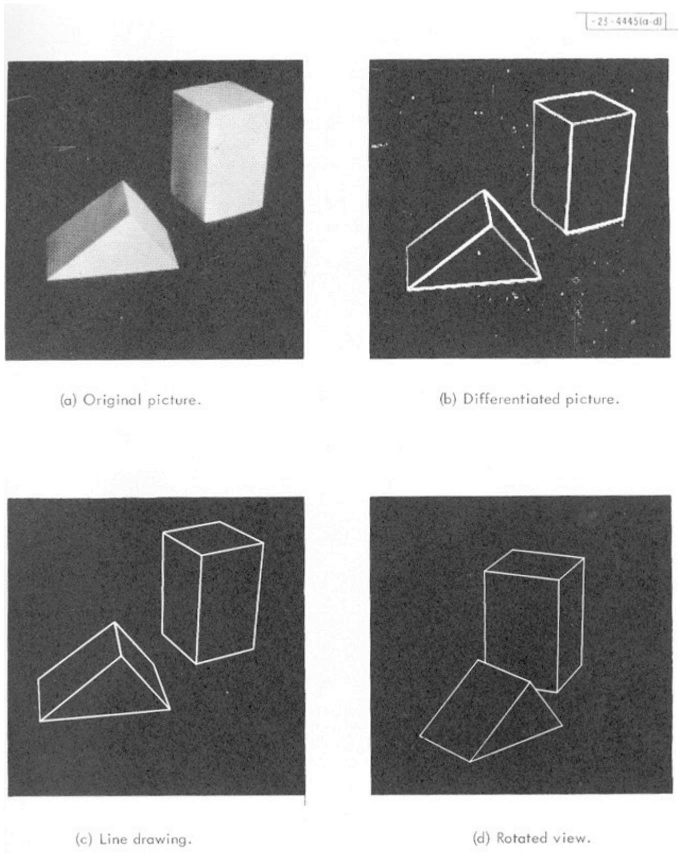
Category recognition

- 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

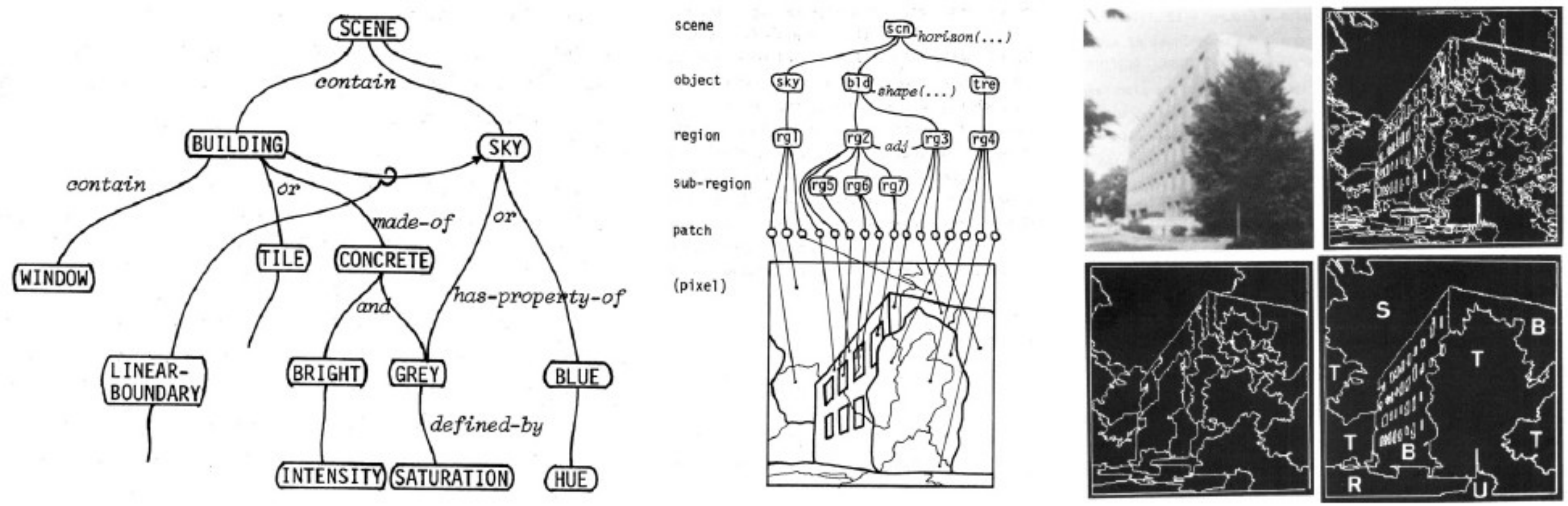


(a) Original picture. (b) Differentiated picture. (c) Line drawing. (d) Rotated view.

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

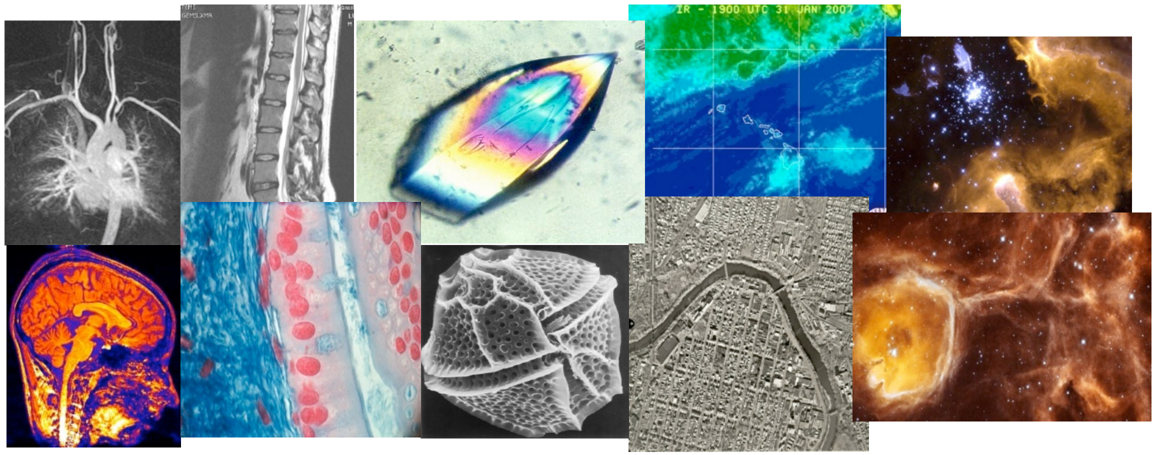
Internet images,
personal photo albums

Movies, news, sports

Why machine learning?

- Today lots of data, complex tasks

Surveillance and security



Medical and scientific images

Why machine learning?

- Today: Lots of data, complex tasks
- 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)

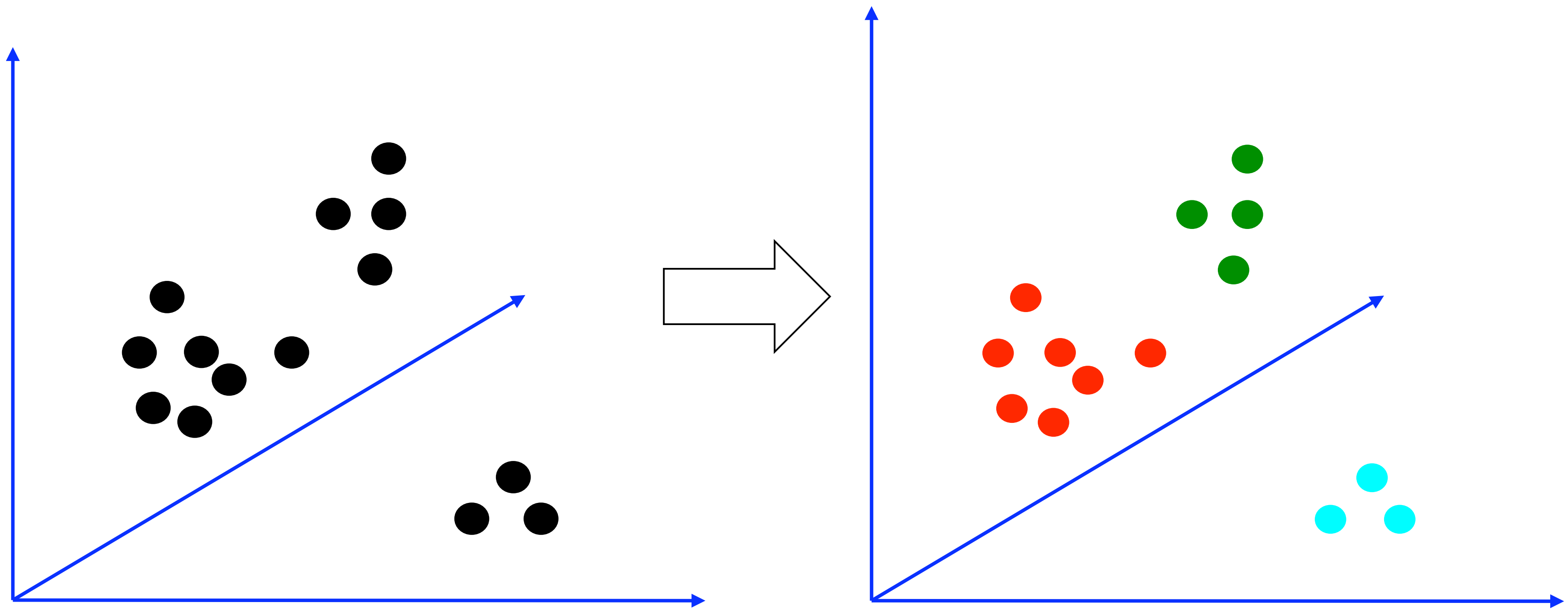
Unsupervised Learning

- 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 of an exploratory/ descriptive data analysis

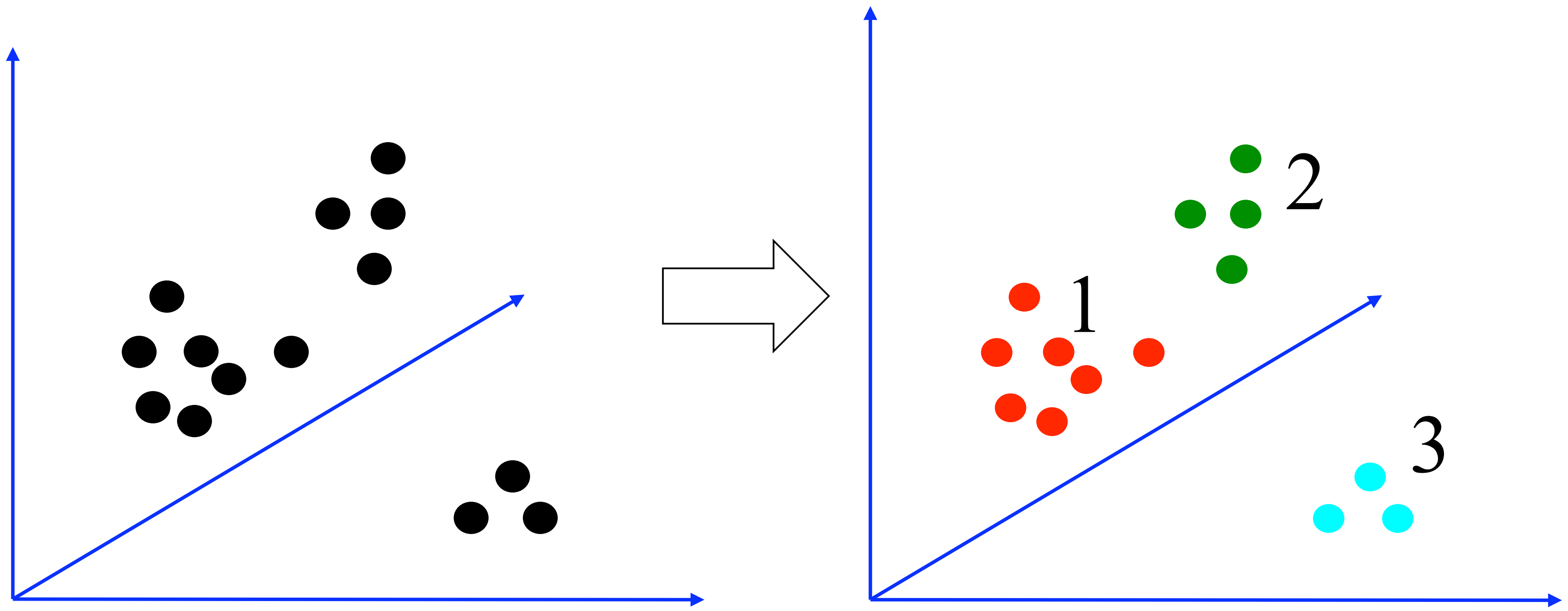
Unsupervised Learning

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



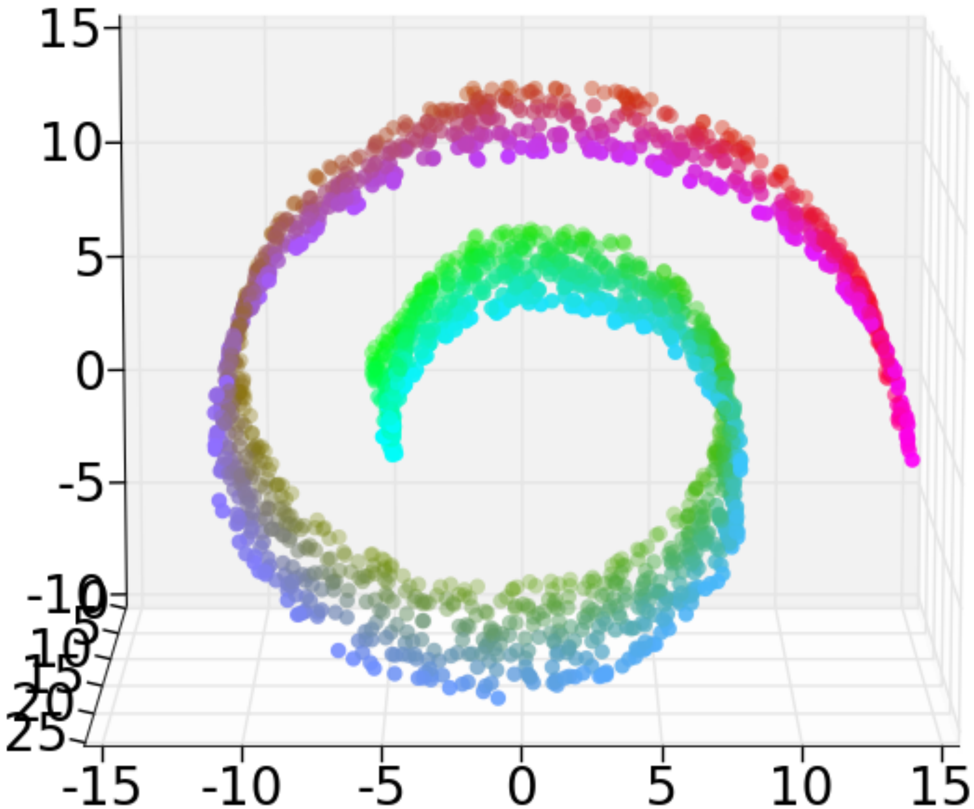
Unsupervised Learning

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



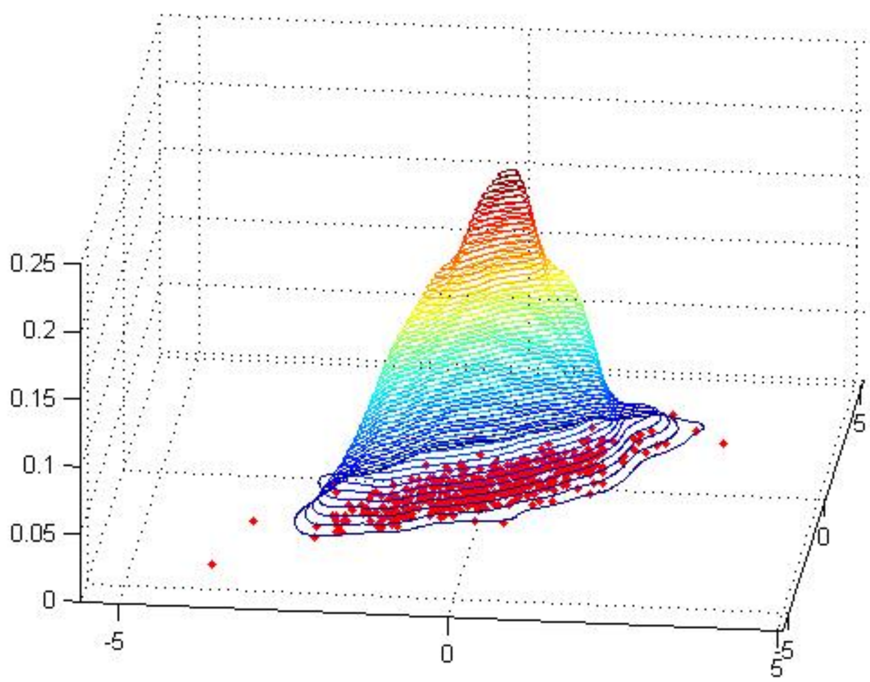
Unsupervised Learning

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



Unsupervised Learning

- **Density estimation**
  - Find a function that approximates the probability density of the data (i.e., value of the function is high for “typical” points and low for “atypical” points)
  - Can be used for **anomaly detection**

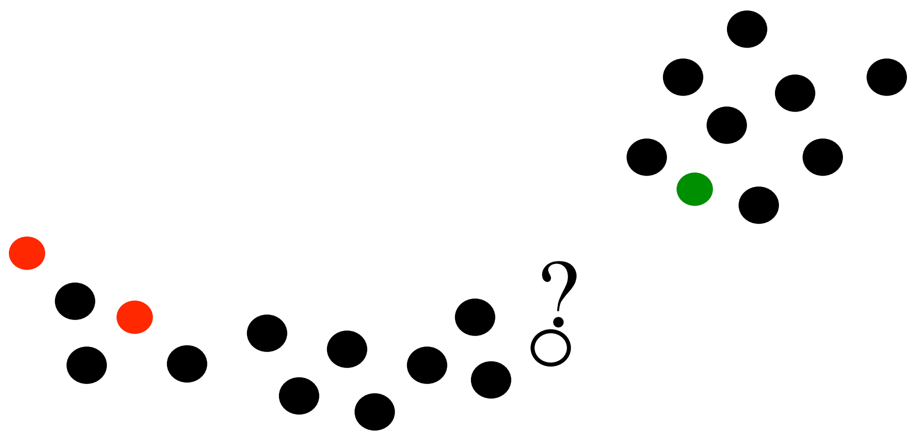


Other types of learning

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

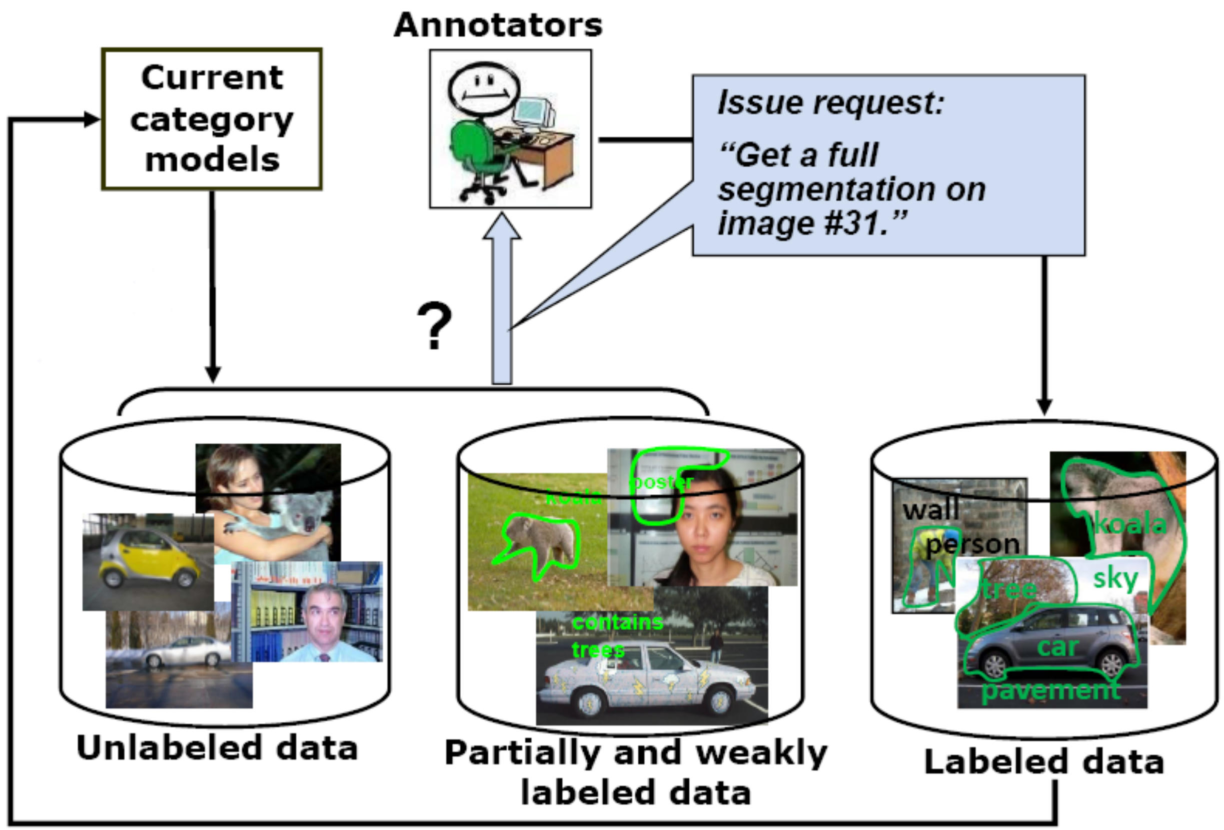
Other types of learning

- **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?



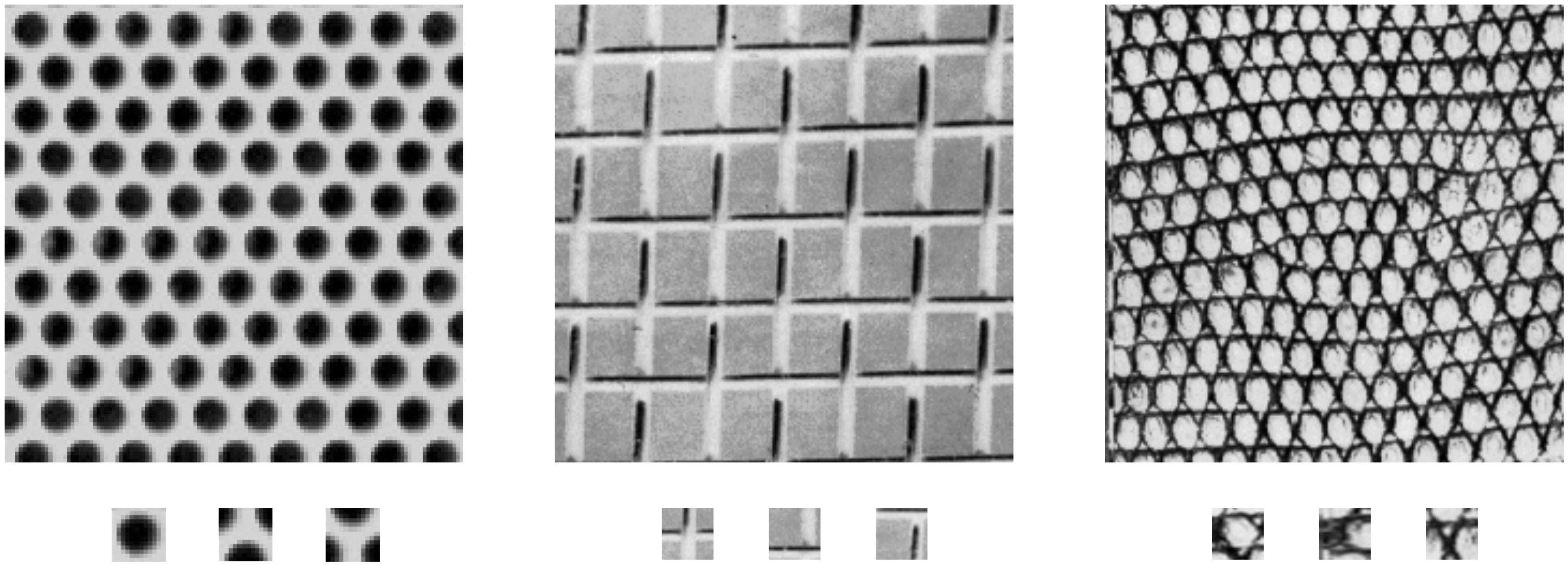
Other types of learning

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



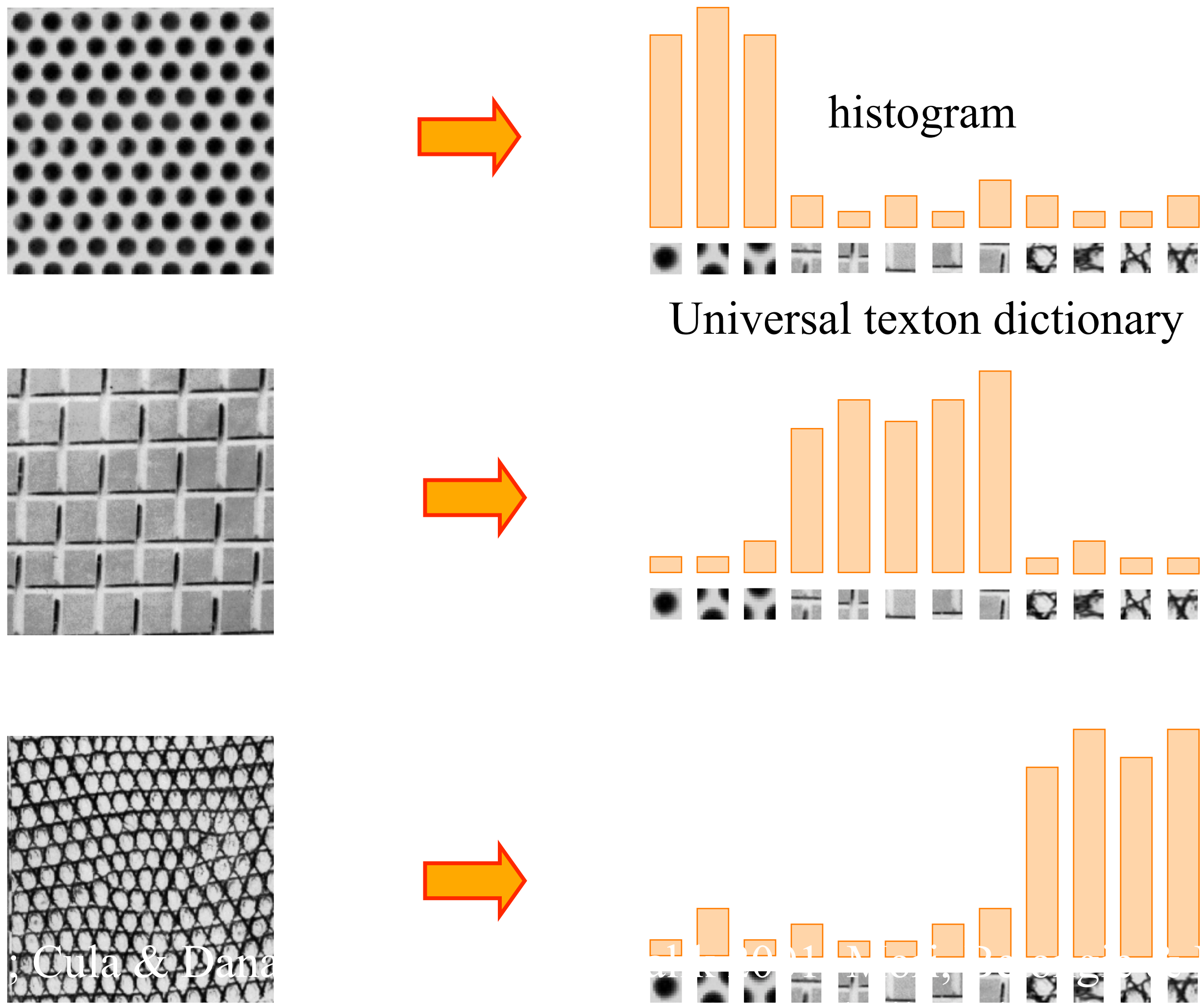
Bag-of-features for image classification

- 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

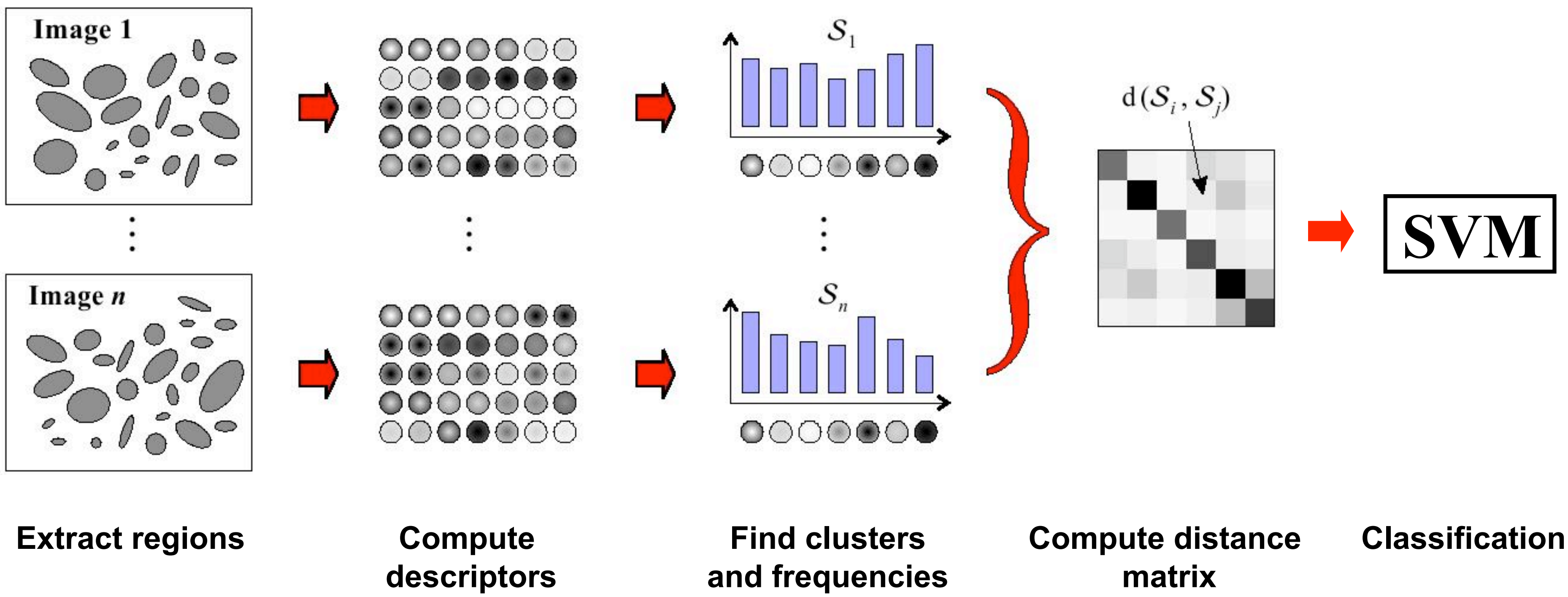
Texture recognition



Bag-of-features for image classification

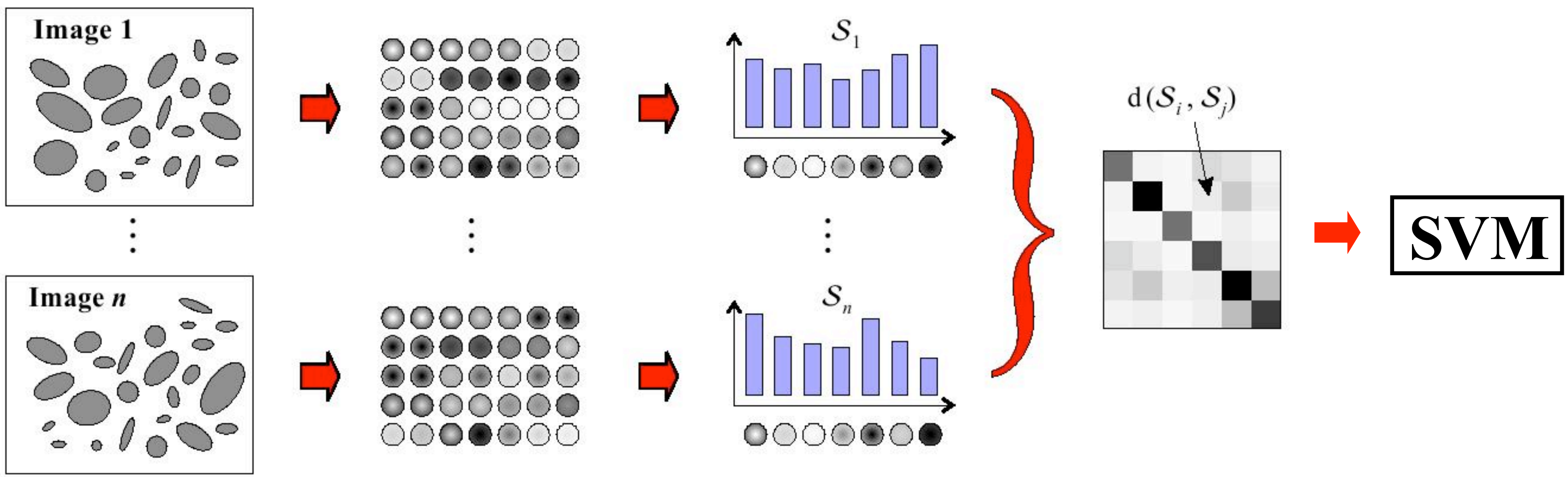
- Origin: bag-of-words
  - Orderless document representation: frequencies of words from a dictionary
  - Classification to determine document categories

Bag-of-features for image classification



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

Bag-of-features for image classification



Extract regions **Compute descriptors** **Find clusters and frequencies** **Compute distance matrix** **Classification**

Step 1 Step 2 Step 3

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

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

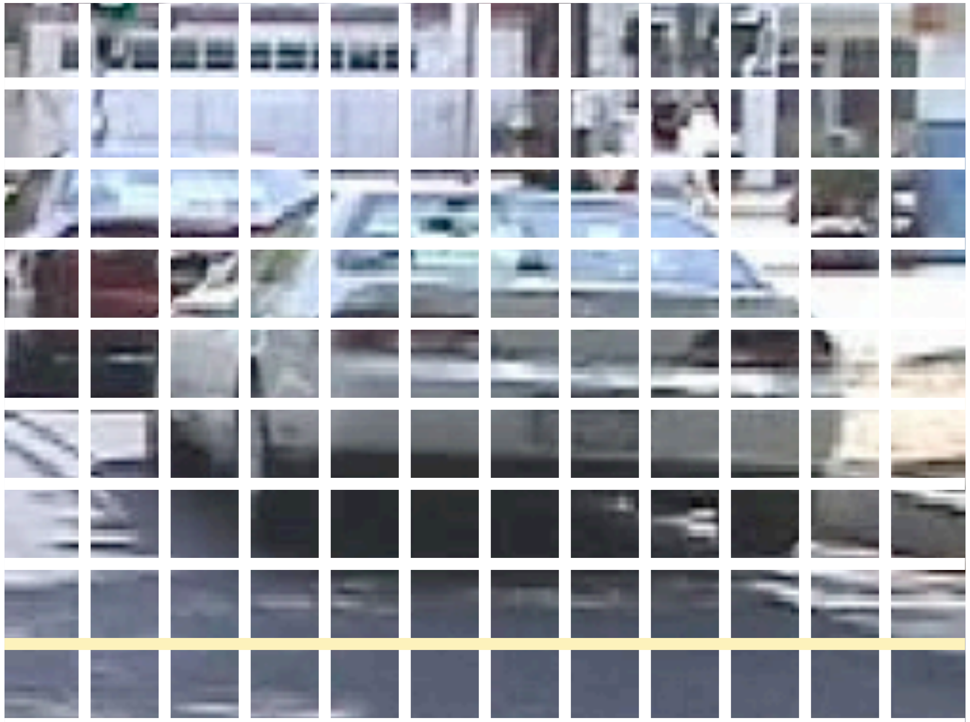
Cars- misclassified into buildings, phones, phones

Step 1: feature extraction

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

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

Dense features

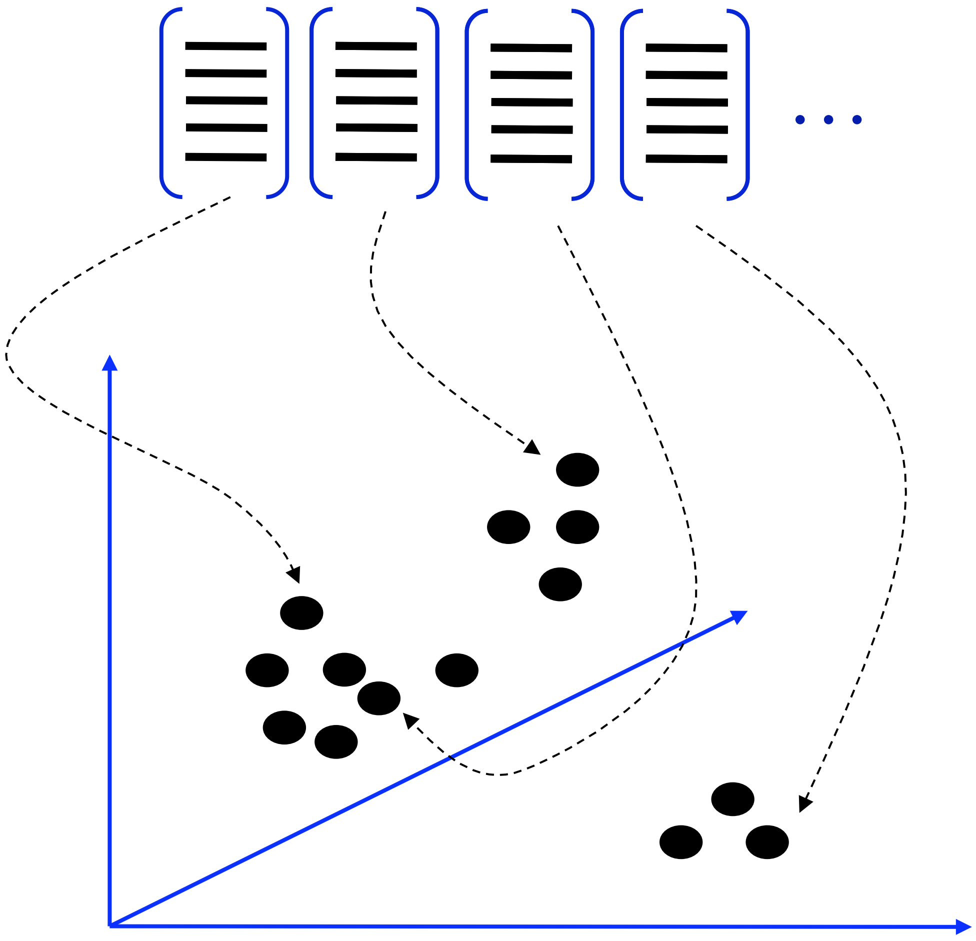


- Multi-scale dense grid: extraction of small overlapping patches at multiple scales
- Computation of the SIFT descriptor for each grid cells

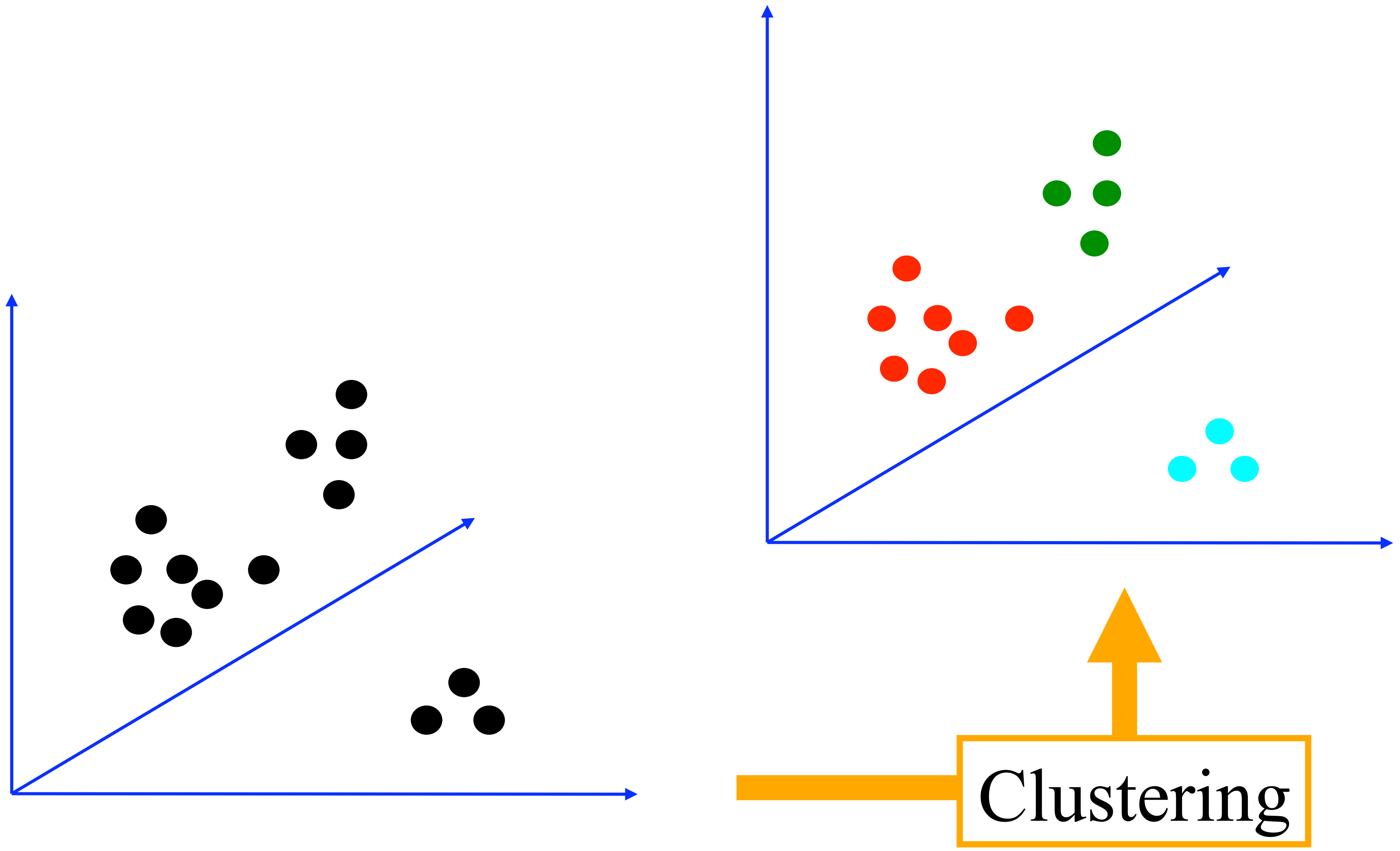
Step 1: feature extraction

- 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

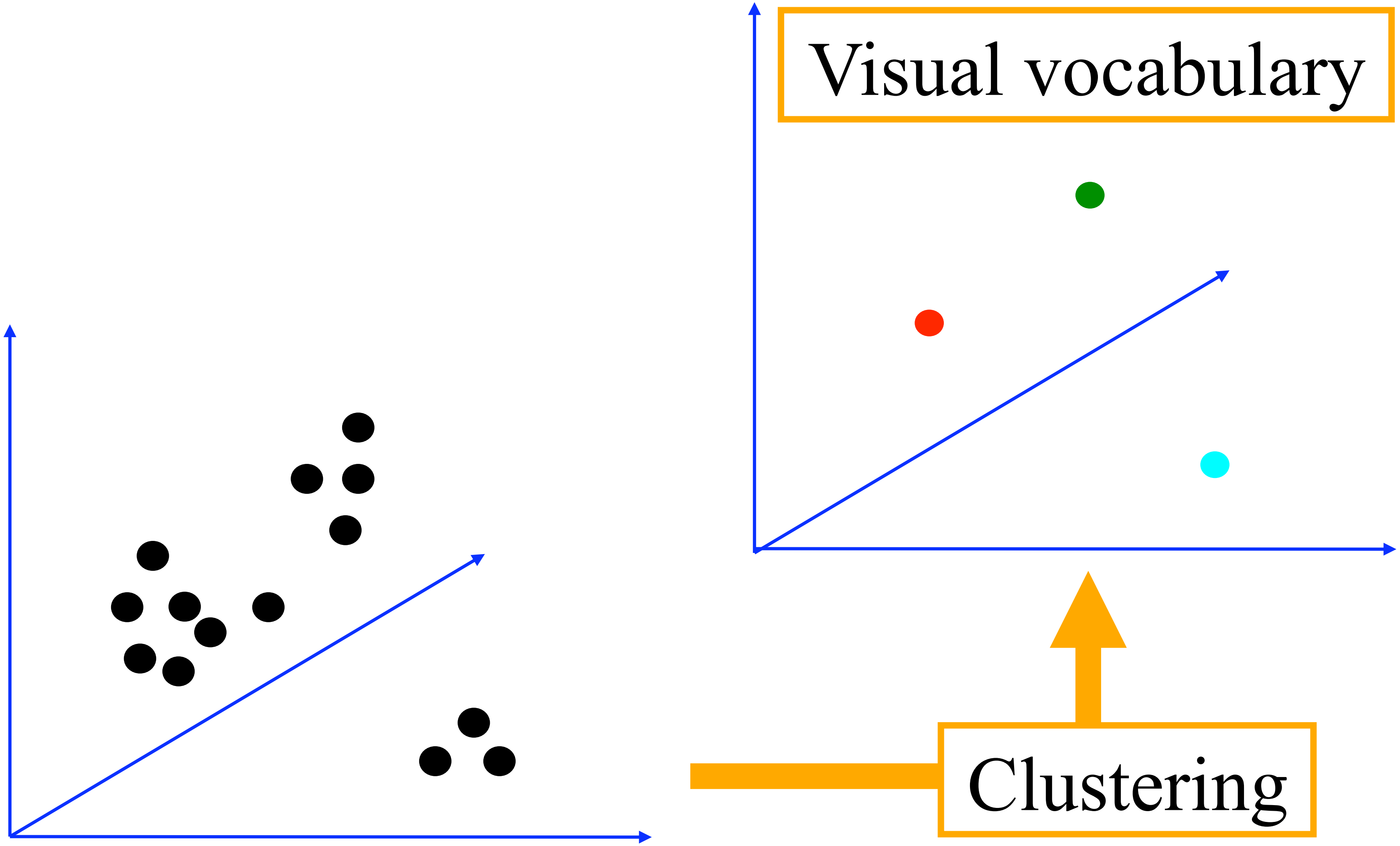
Step 2: Quantization



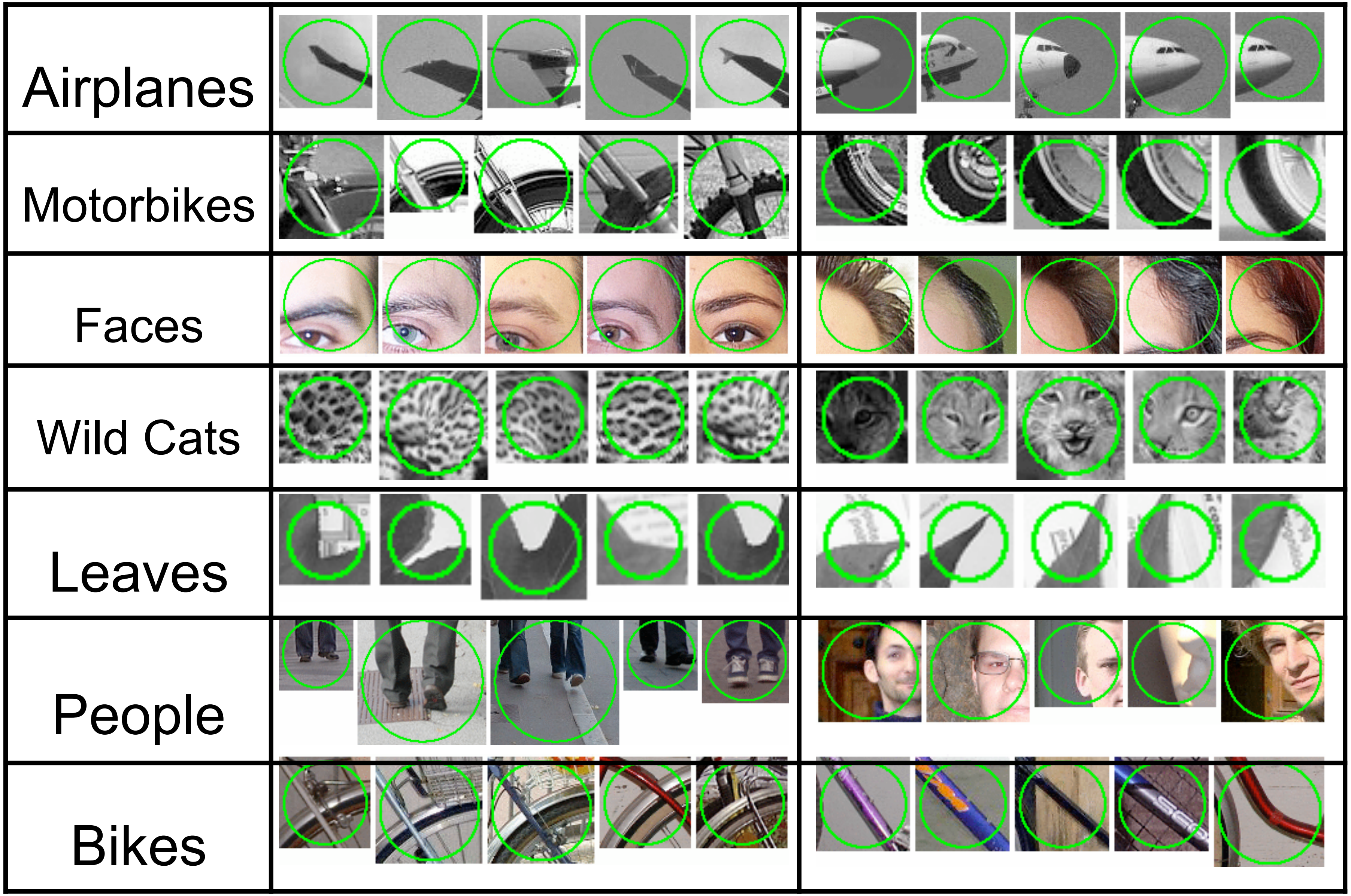
Step 2:Quantization



Step 2: Quantization



Examples for visual words



Step 2: Quantization

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

K-means Clustering: Cost function

- Partition dataset $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in K clusters
- Clusters characterized by cluster prototypes $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$
  - Assign $\mathbf{x}$ to closest prototype
- Cost function

$$J(\{\boldsymbol{\mu}_k\}) = \sum_{n=1}^{N} \min_k \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Non-differentiable, non-convex

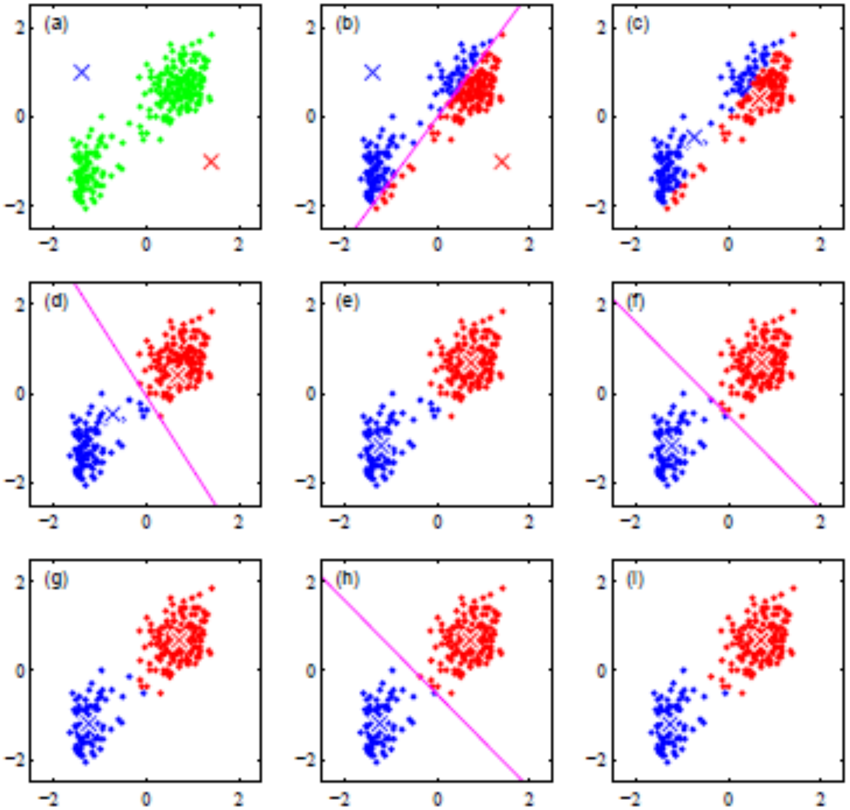
K-means clustering

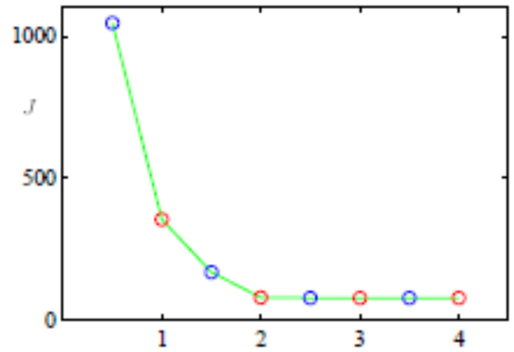
- We want to minimize sum of squared Euclidean distances between points x_i and their nearest cluster centers

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

K-means Clustering: Example





K-means clustering

- Local minimum, solution dependent on initialization
- Initialization important, run several times
  - Select best solution, min cost

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”

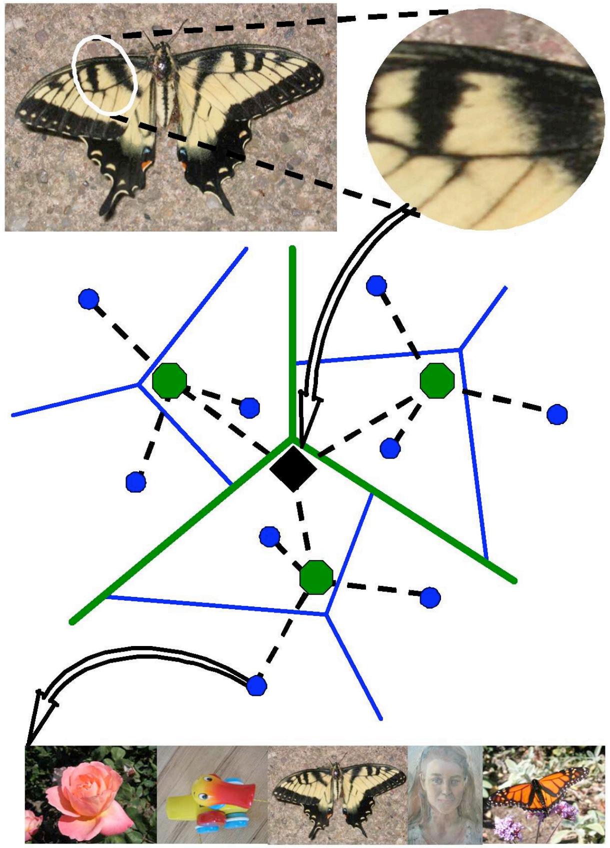
- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word

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)

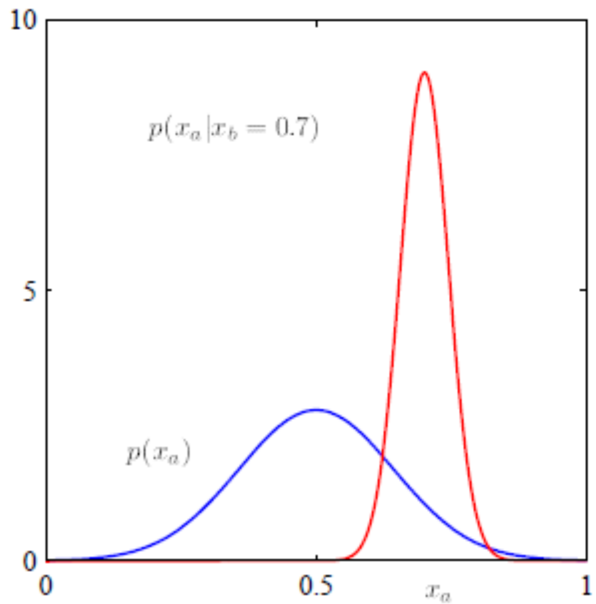
- Soft quantization: Gaussian
mixture instead of k-means

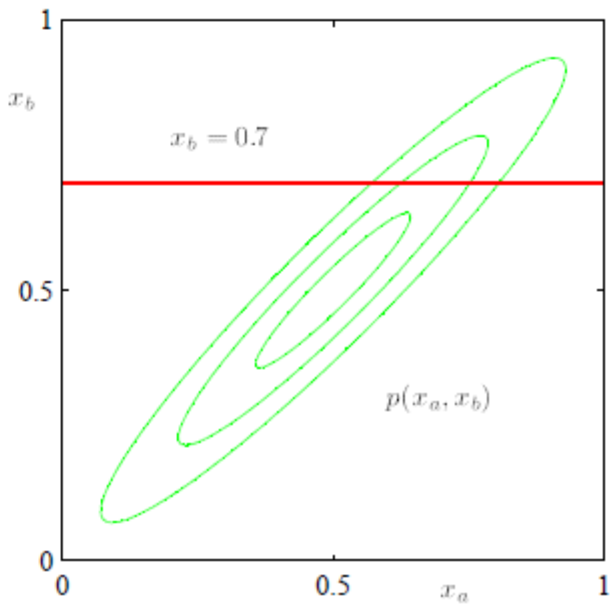


Gaussian mixture model (GMM)

Gaussian density

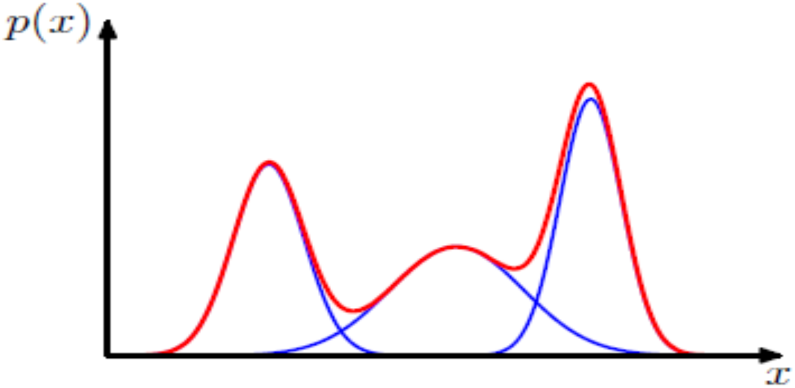
$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma) = (2\pi)^{(-d/2)} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)$$

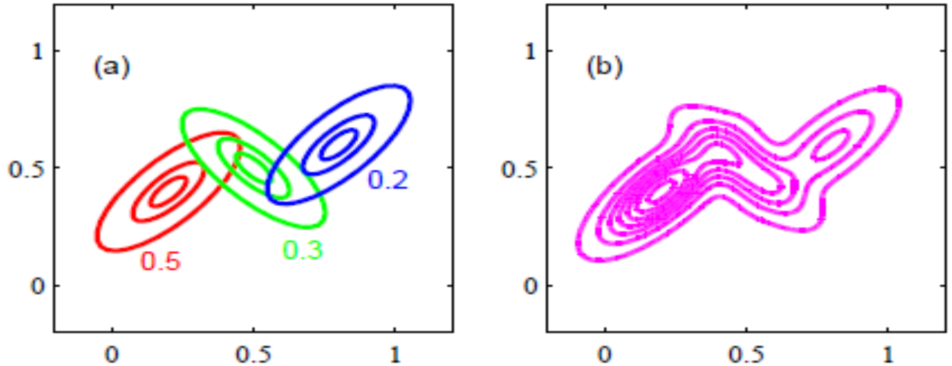




Mixture of Gaussians: weighted sum of Gaussians

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





$$p(z_n = k|\mathbf{x}_n) \quad = \frac{p(z_n{=}k)p(\mathbf{x}_n|z_n{=}k)}{p(\mathbf{x}_n)} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{k'}, \Sigma_{k'})} \equiv q_{nk}$$

Mixture of Gaussians: Maximum Likelihood Estimation

- Given a data set $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ find clustering
  - clustering induced by mixture model
  - fit mixture parameters $\{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$ to data

- Find parameters that maximize data (log-)likelihood
  - let the $\mathbf{x}_n$ independently distributed according mixture

$$
\begin{aligned}
\log p(\mathbf{X}) &= \log \prod_{n=1}^{N} p(\mathbf{x}_n) = \sum_{n=1}^{N} \log p(\mathbf{x}) \\
&= \sum_{n} \log \left\{ \sum_{k} \pi_k \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \right\}
\end{aligned}
$$

- Not convex, and not trivial to maximize.

Mixture of Gaussians: EM algorithm

1. Initialize parameters $\{\mu_k, \Sigma_k, \pi_k\}$
2. Expectation Step: Evaluate responsibilities:

$$q_{nk} = p(z_n = k | \mathbf{x}_n) \tag{1}$$

3. Maximization Step: Re-estimate parameters:

$$\pi_k^{\text{new}} = \frac{\sum_n q_{nk}}{N}$$
$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{\sum_n q_{nk}} \sum_n q_{nk} \mathbf{x}_n$$
$$\Sigma_k^{\text{new}} = \frac{1}{\sum_n q_{nk}} \sum_n q_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^\top$$

4. Evaluate log-likelihood $\log p(\mathbf{X})$, and check for convergence (go to step 2).

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
- Frequency histogram

Image representation