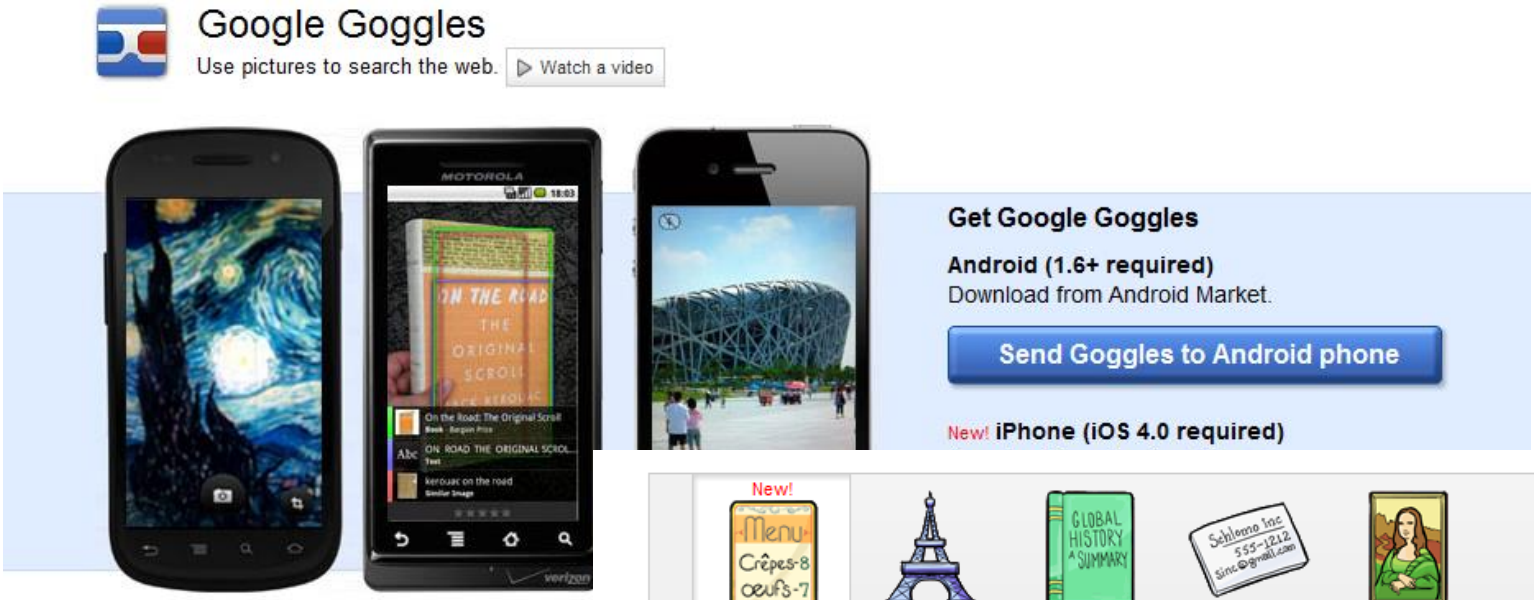
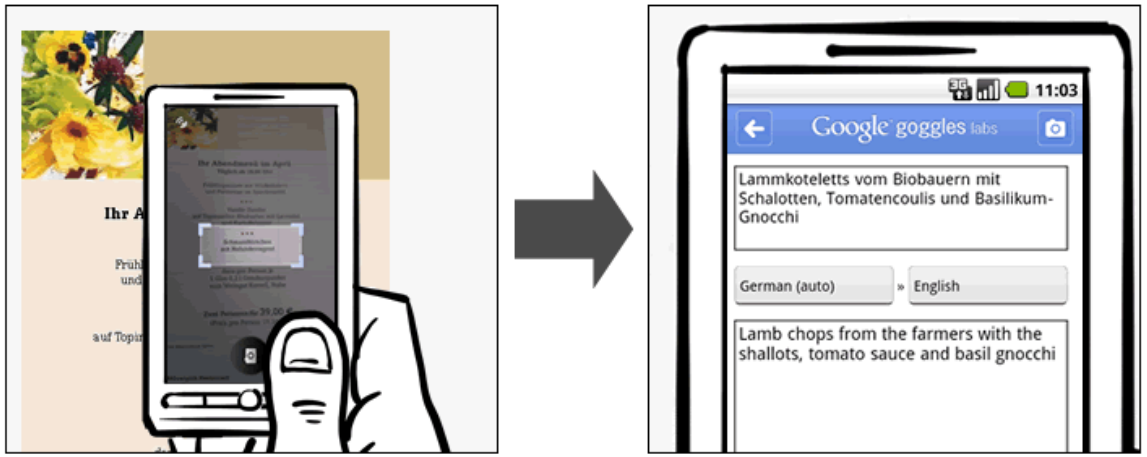
Object Recognition

Object Recognition

Introduction

Google Goggles





Slide credit: Kristen Grauman

Visual Recognition

- Design algorithms that have the capability to:
  - Classify images or videos
  - Detect and localize objects
  - Estimate semantic and geometrical attributes
  - Classify human activities and events

Slide credit: Juan Carlos Niebles

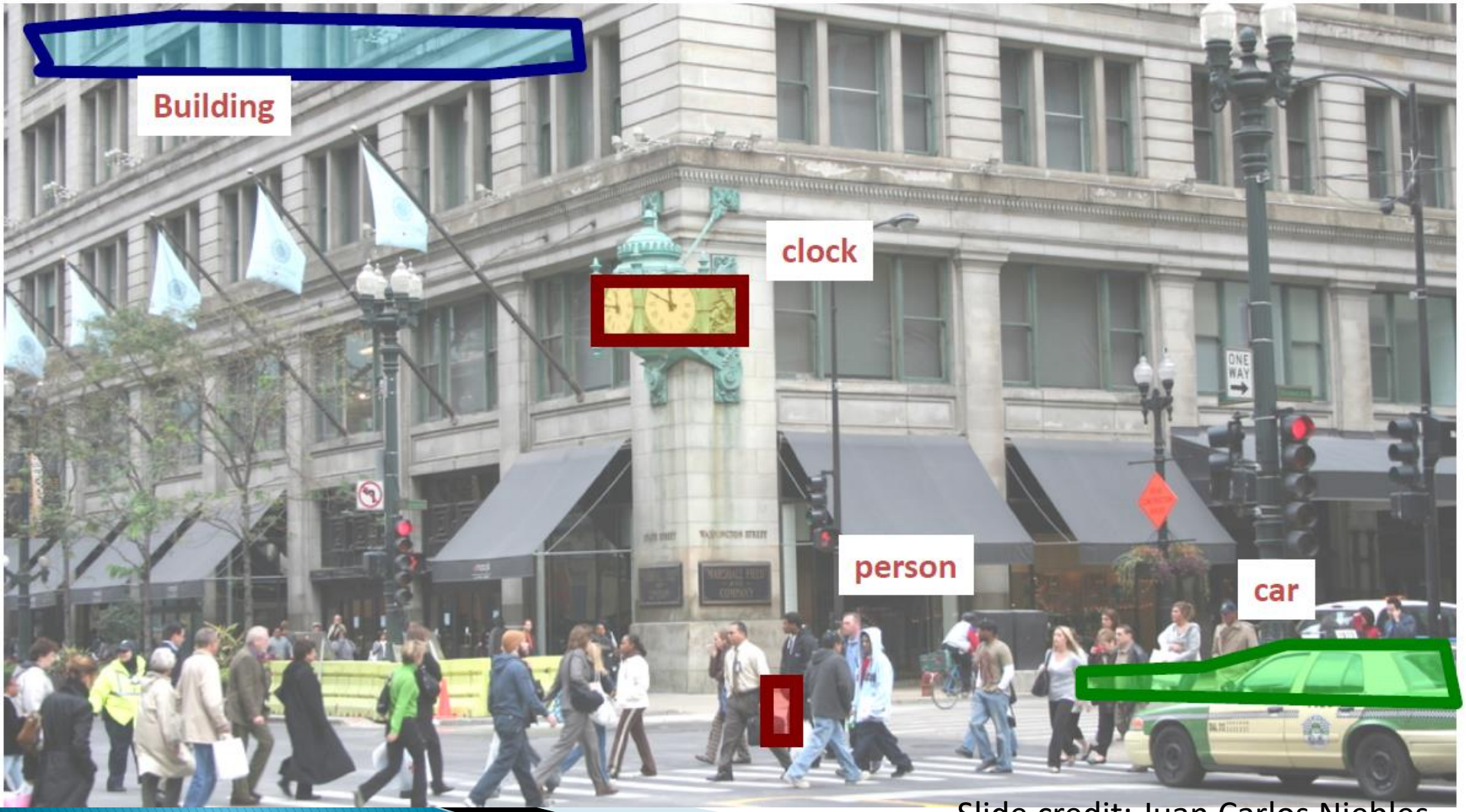
Does this image contain a building?

Slide credit: Juan Carlos Niebles

Does this image contain a car?

Slide credit: Juan Carlos Niebles

Which object does this image contain?



Slide credit: Juan Carlos Niebles

Estimating object semantic & geometric attributes

Slide credit: Juan Carlos Niebles

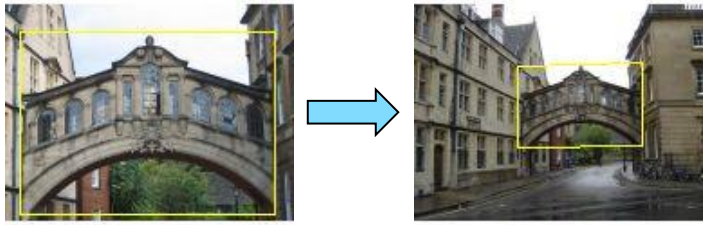
What are these people doing?

Slide credit: Juan Carlos Niebles

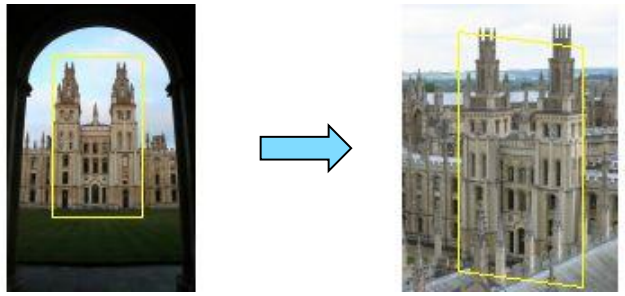
How many object categories are there?

Slide credit: Juan Carlos Niebles

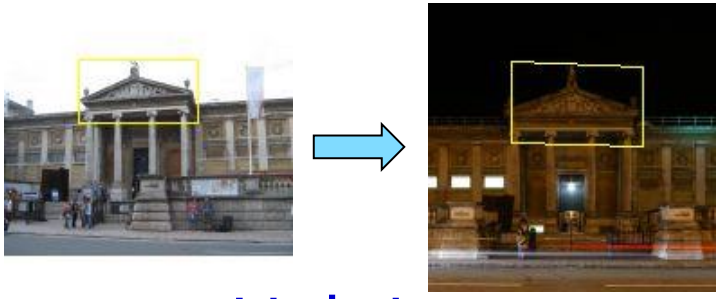
Why is it difficult?



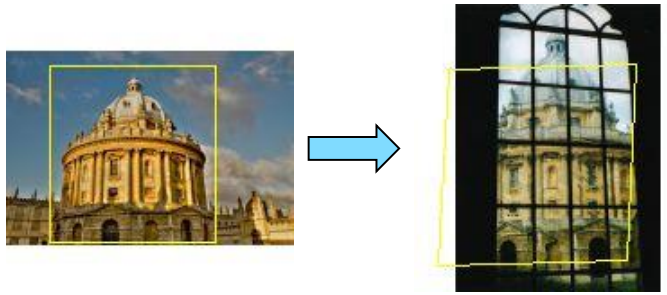
Scale



Viewpoint



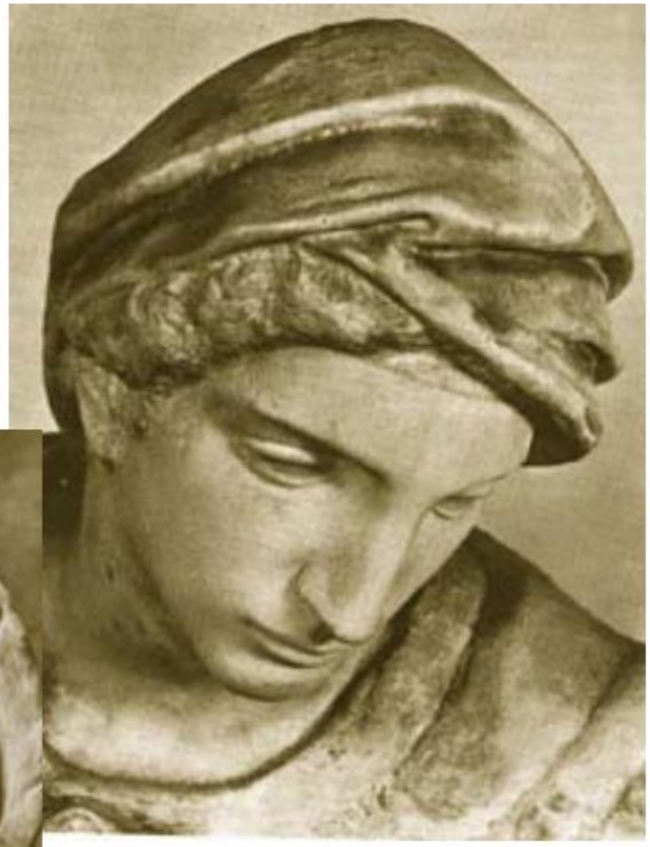
Lighting



Occlusion

Slide credit: J. Sivic

Challenges: viewpoint variation



Michelangelo 1475-1564

Slide credit: Juan Carlos Niebles

Scale

Slide credit: Juan Carlos Niebles

illumination

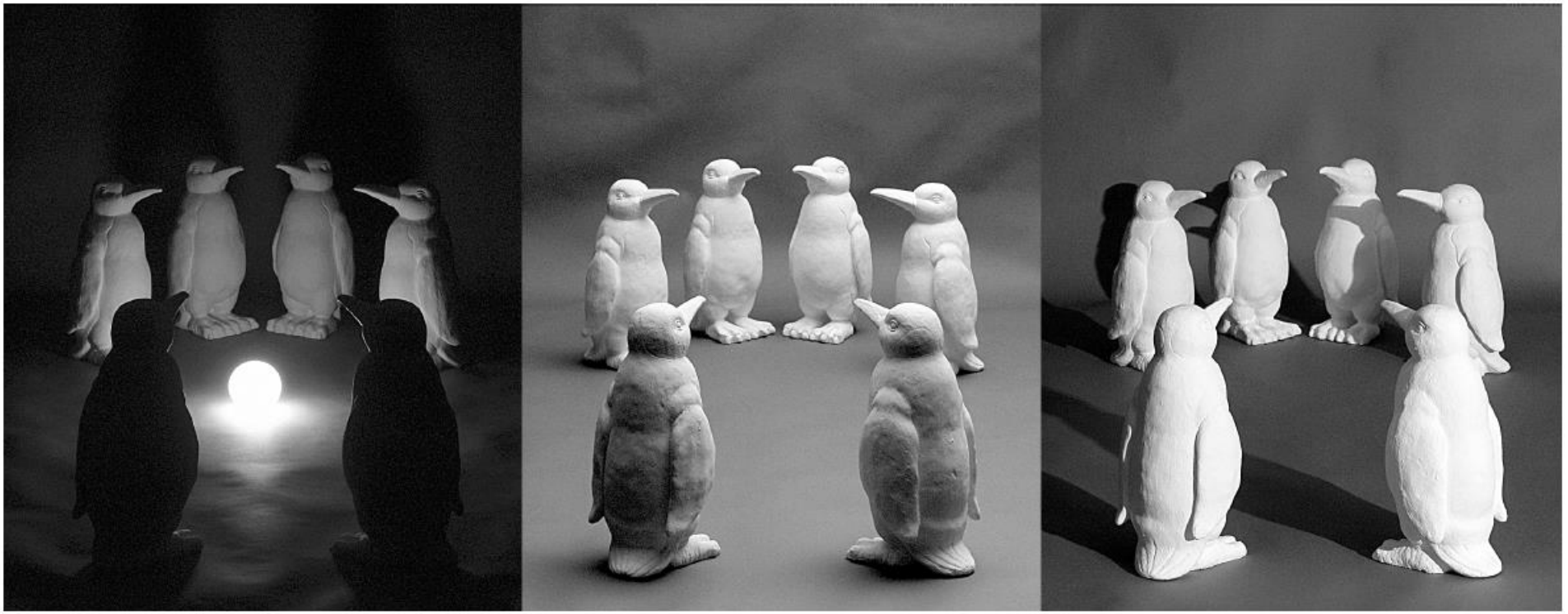


image credit: J. Koenderink

Deformation

Slide credit: Juan Carlos Niebles

Occlusion

Magritte, 1957

Slide credit: Juan Carlos Niebles

Background

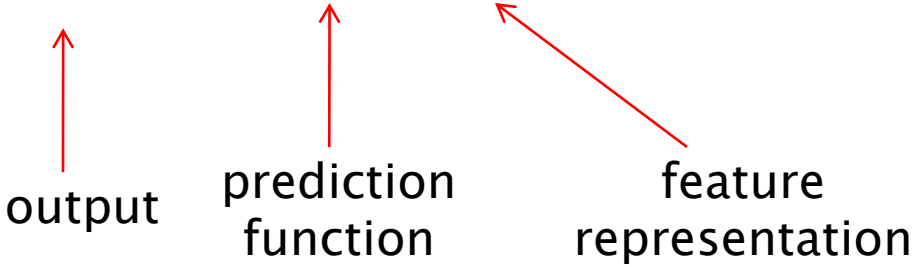
Kilmeny Niland. 1995

Intra-class variation

Slide credit: Juan Carlos Niebles

Machine Learning Framework

$$y = f(\mathbf{x})$$



- **Training**: given a *training set* of labeled examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing**: apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: Svetlana Lazebnik

Steps

Training

Training Images

Training Labels

Image Features

Training

Learned model

Testing

Test Image

Image Features

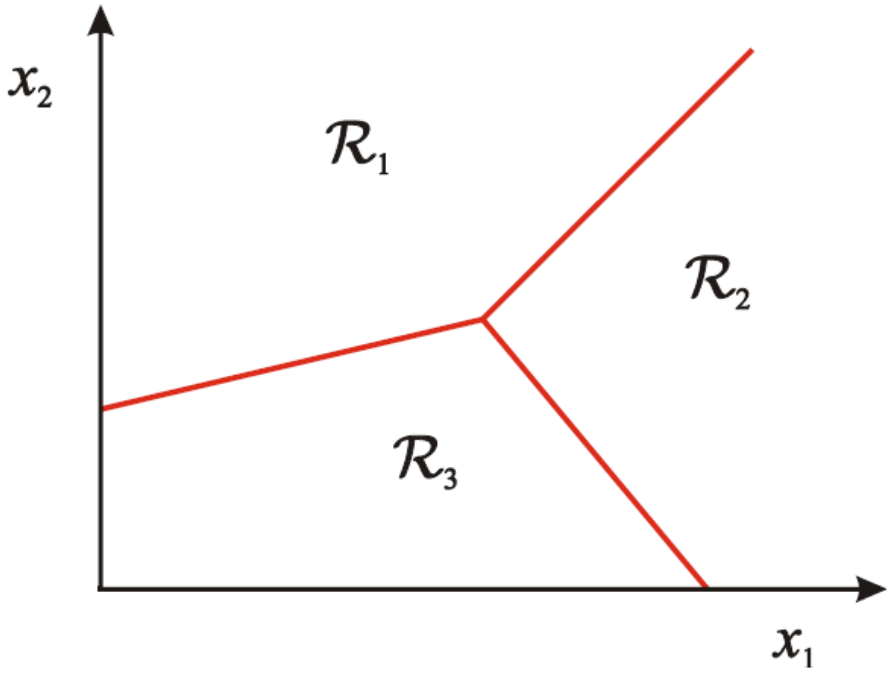
Learned model

Prediction

Slide credit: D. Hoiem

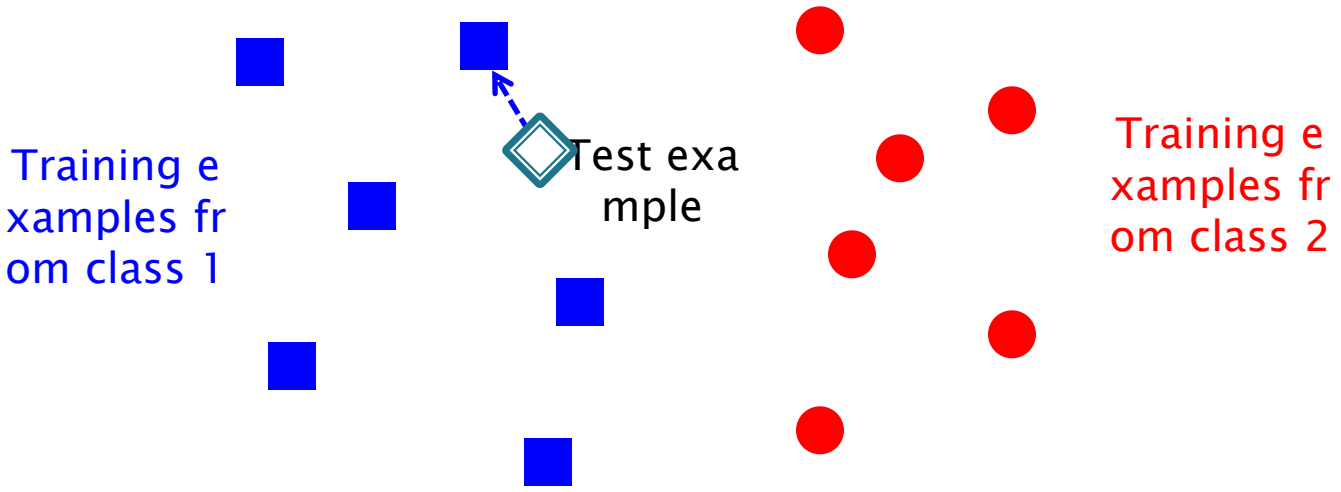
Classification

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



Slide credit: Svetlana Lazebnik

Nearest neighbor



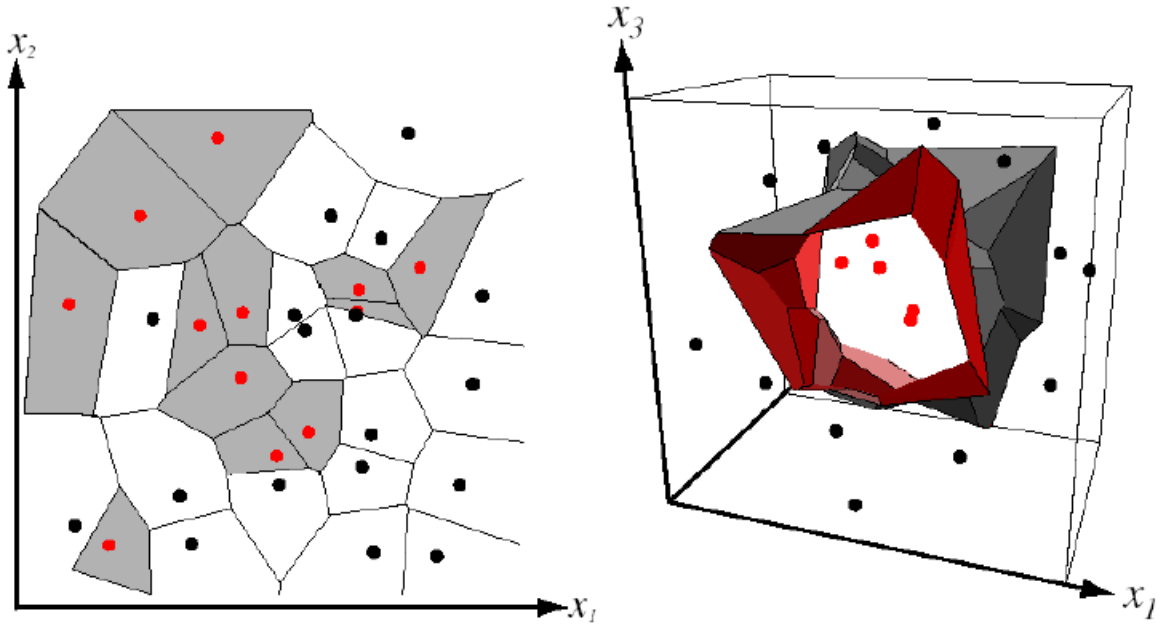
f(**x**) = label of the training example nearest to **x**

- All we need is a distance or similarity function for our inputs
- No training required!

Slide credit: Svetlana Lazebnik

Nearest Neighbor Classifier

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



from Duda *et al.*

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

Source: D. Lowe

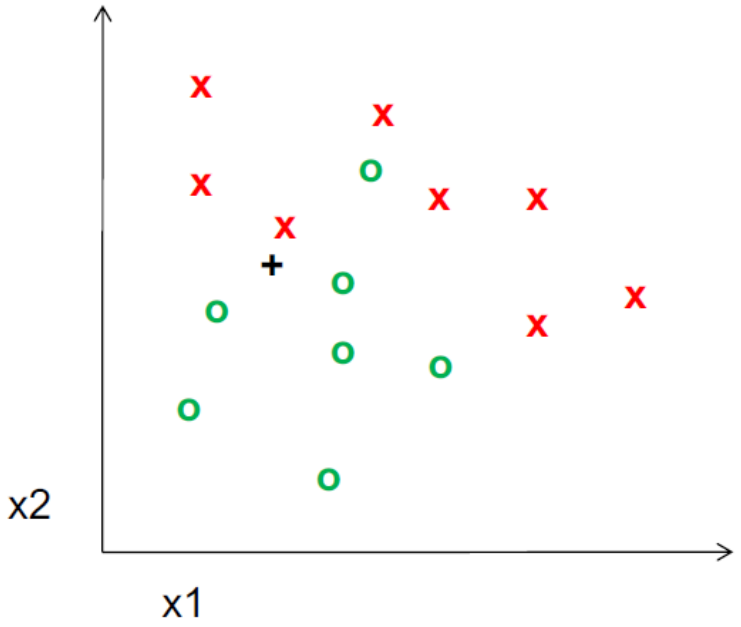
K-Nearest Neighbor

- For a new point, find the k closest points from training data
- Vote for class label with labels of the k points

Distance measure - Euclidean

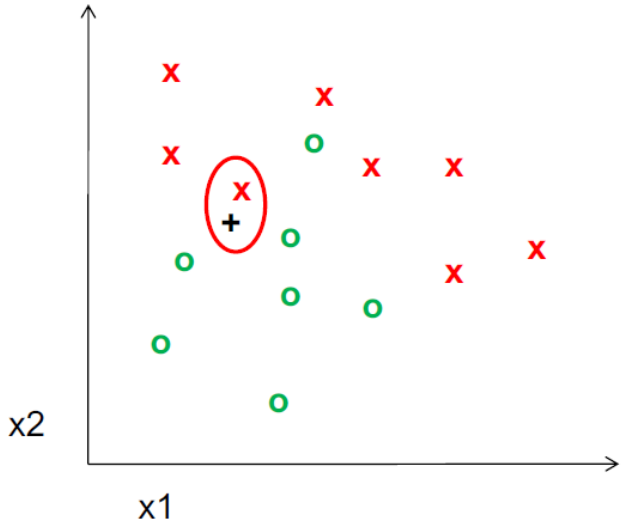
$$Dist(X^n, X^m) = \sqrt{\sum_{i=1}^{D} (X_i^n - X_i^m)^2}$$

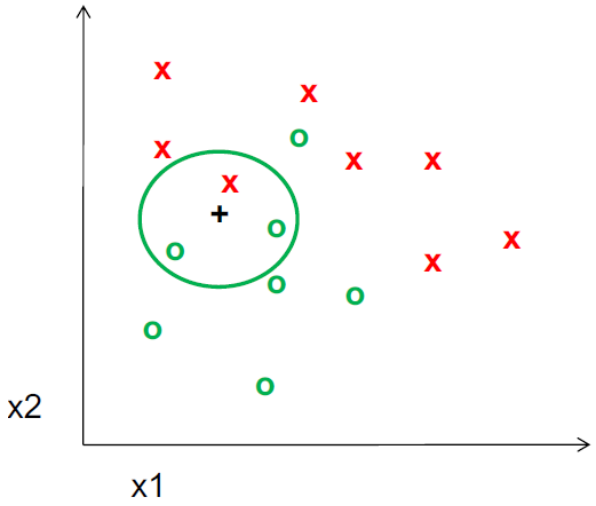
Where X^n and X^m are the n-th and m-th data points

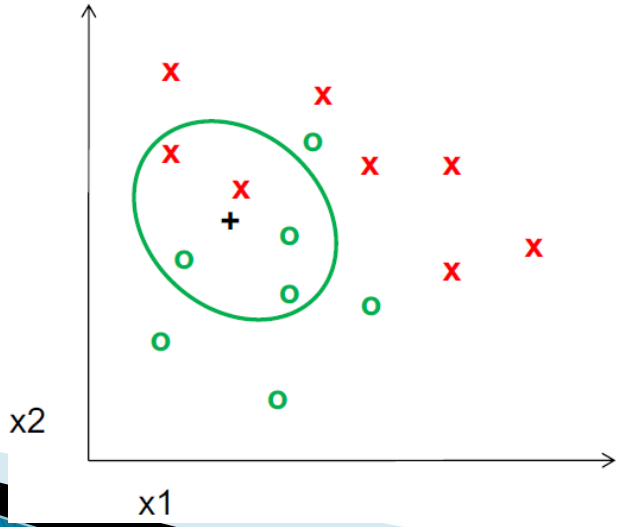


Slide credit: Svetlana Lazebnik

K-Nearest Neighbor

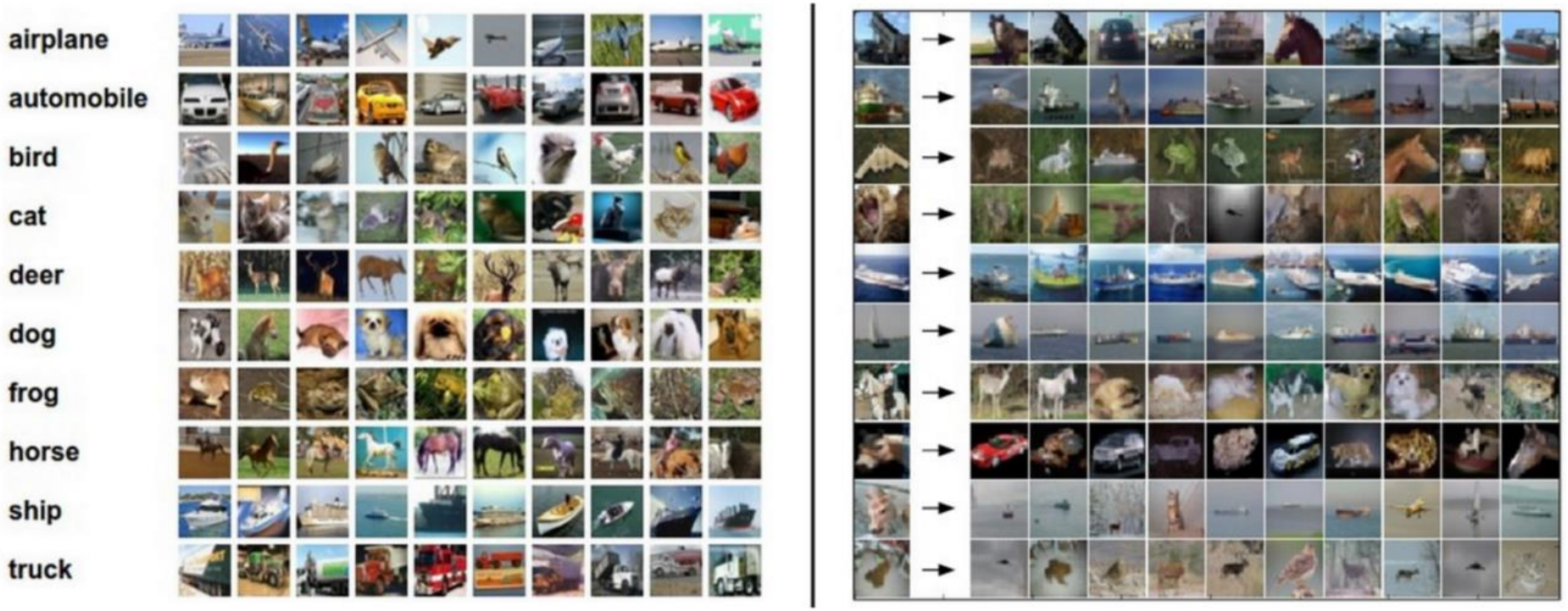






Slide credit: Svetlana Lazebnik

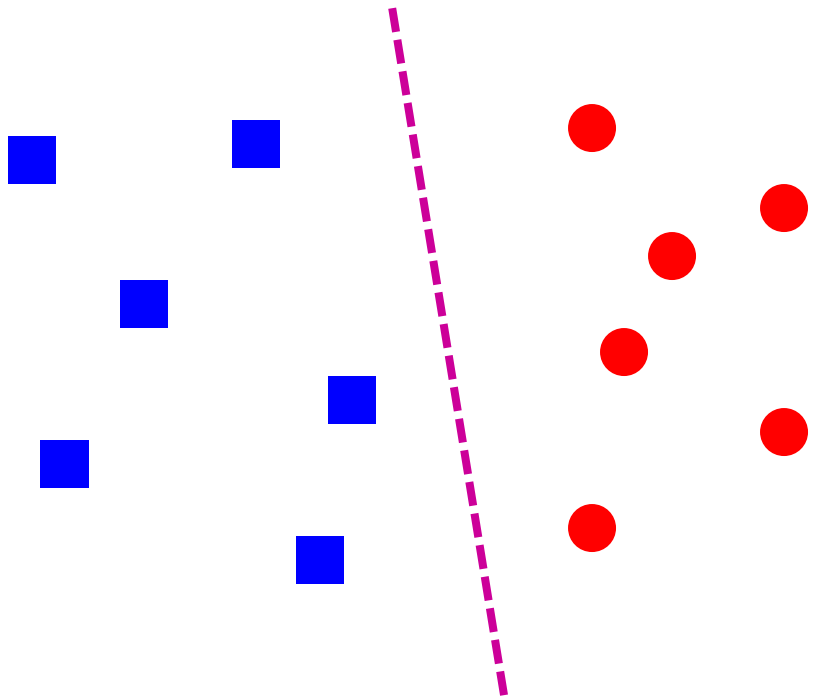
K-nearest neighbor classifier



Left: Example images from the CIFAR-10 dataset. Right: first column shows a few test images and next to each we show the top 10 nearest neighbors in the training set according to pixel-wise difference.

Credit: Andrej Karpathy

Linear Classifiers

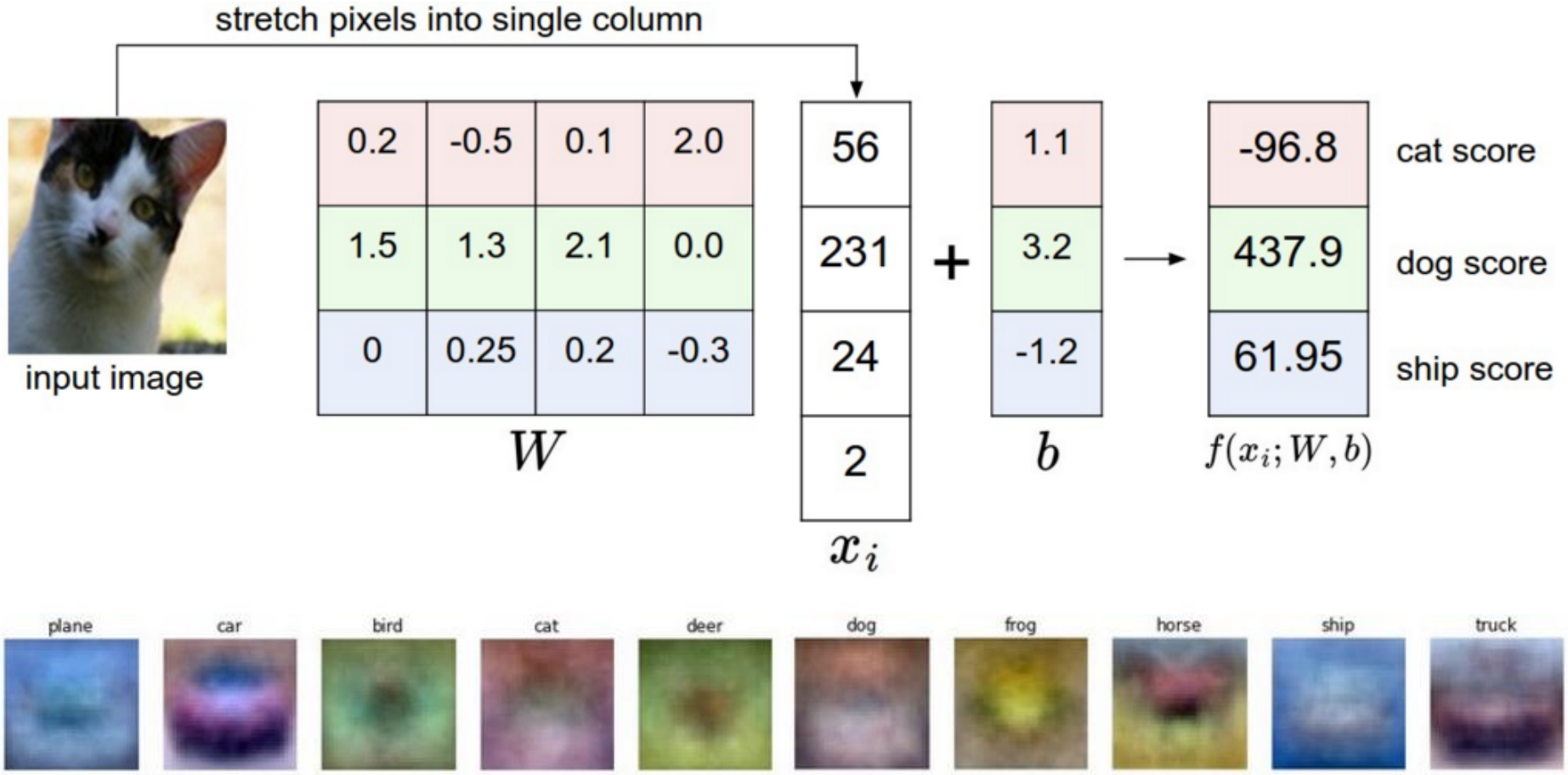


- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Slide credit: Svetlana Lazebnik

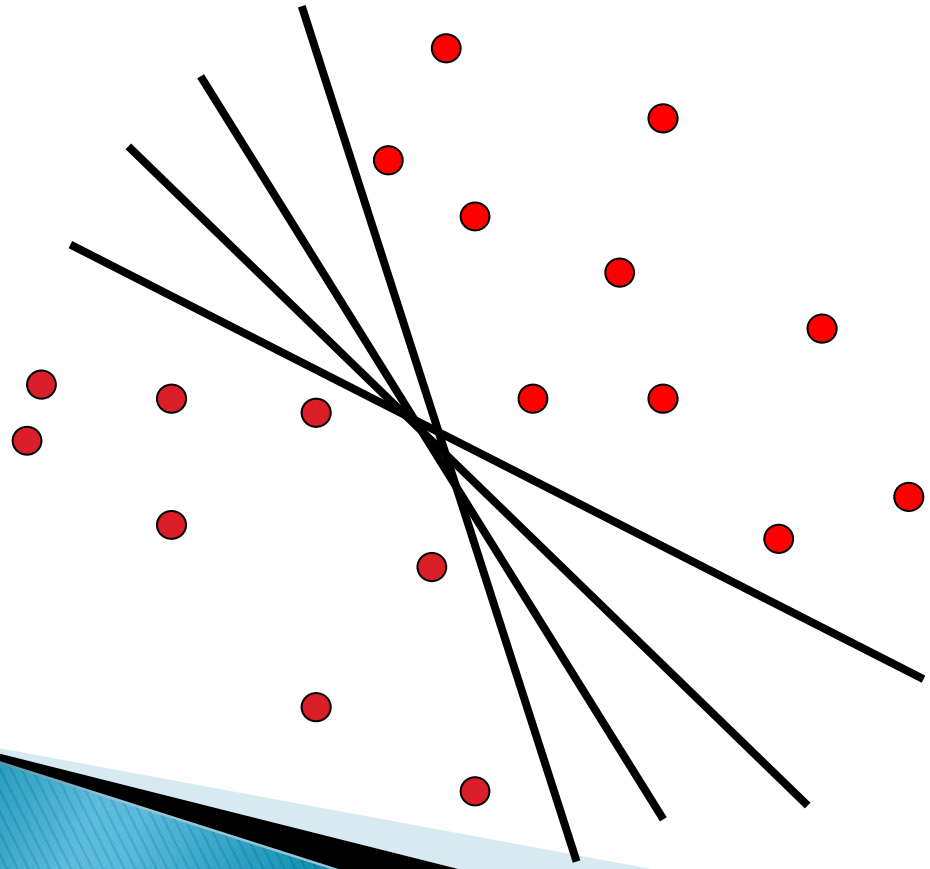
Visualizing linear classifiers



Source: Andrej Karpathy

Linear classifiers

- When the data is linearly separable, there may be more than one separator (hyperplane)



Which separator is best?

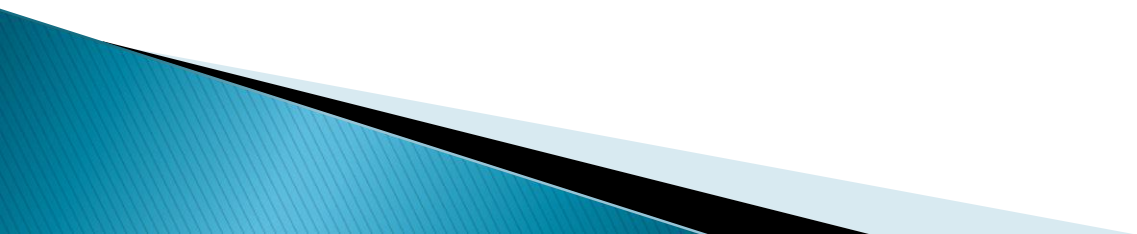
Slide credit: Svetlana Lazebnik

Nearest neighbor vs. linear classifiers

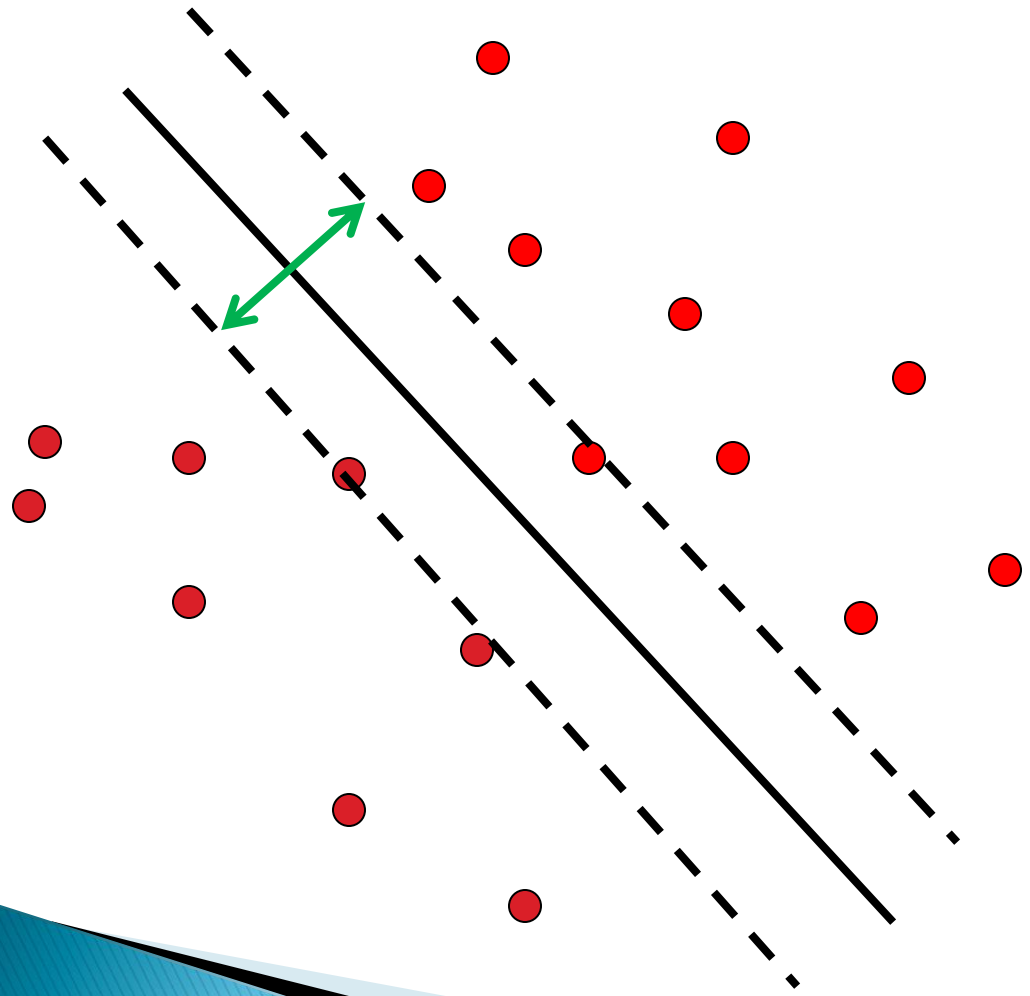
- NN pros:
  - Simple to implement
  - Decision boundaries not necessarily linear
  - Works for any number of classes
  - *Nonparametric* method
- NN cons:
  - Need good distance function
  - Slow at test time
- Linear pros:
  - Low-dimensional *parametric* representation
  - Very fast at test time
- Linear cons:
  - Works for two classes
  - How to train the linear function?
  - What if data is not linearly separable?

Slide credit: Svetlana Lazebnik

Support Vector Machines (SVMs)

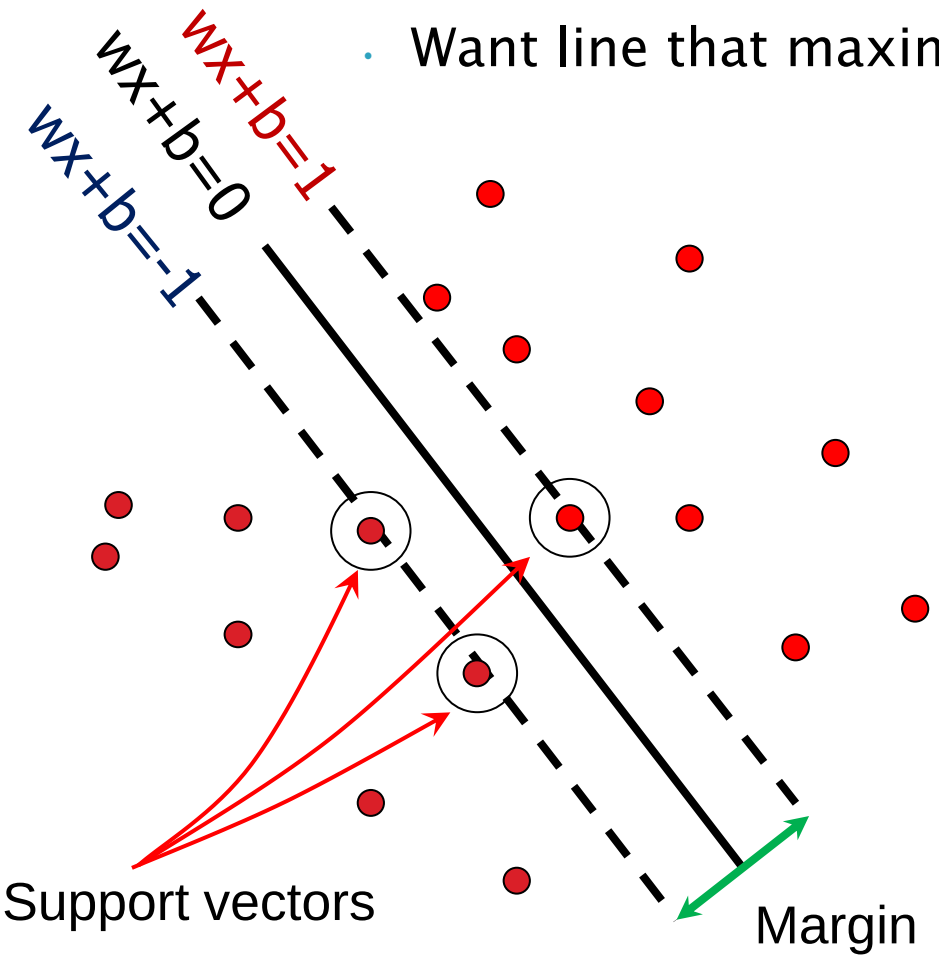


Support Vector Machines (SVMs)



- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

Support vector machines



- Want line that maximizes the margin.

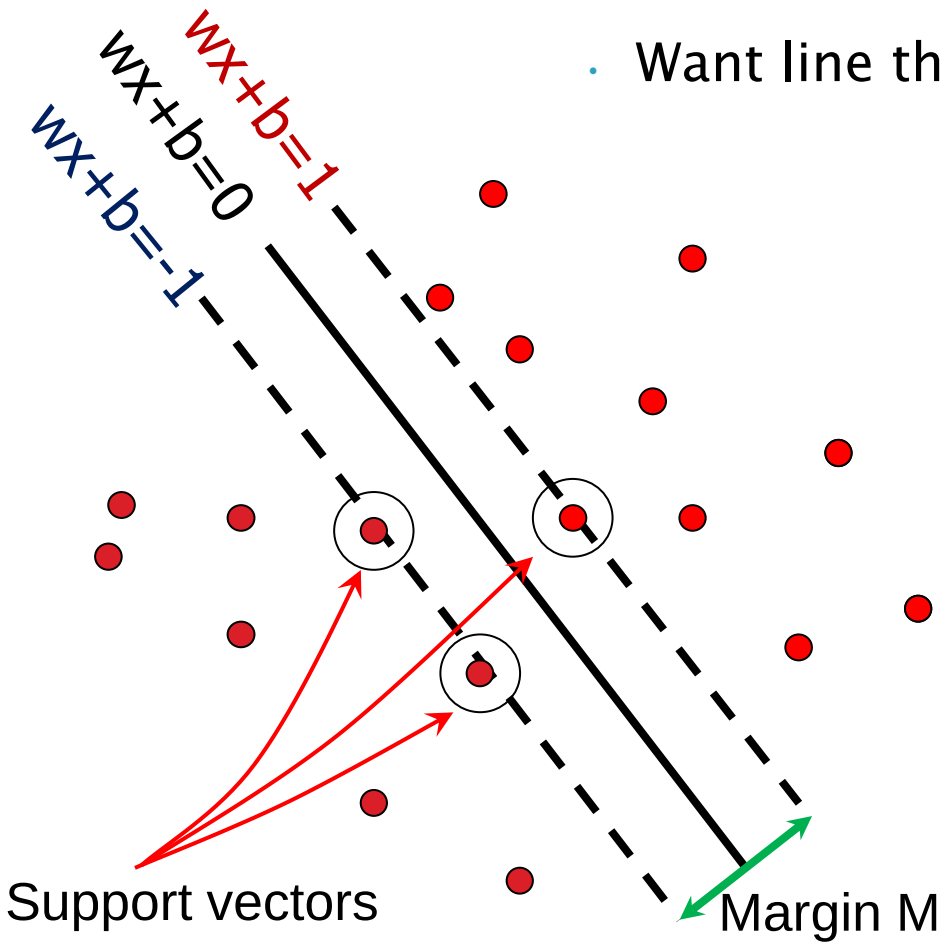
$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

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

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

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

Support vector machines



- Want line that maximizes the margin.

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

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

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

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

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Finding the maximum margin line

Maximize margin $2/\|\mathbf{w}\|$

Correctly classify all training data points:

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

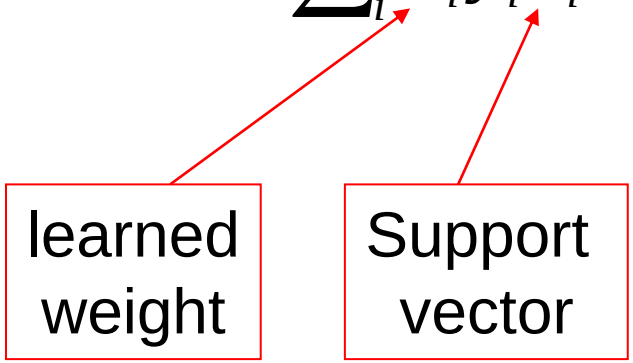
$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

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

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

Finding the maximum margin line

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



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

Finding the maximum margin line

- 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)

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

- Classification function:

$$f(x) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + \text{b})$$

$$= \text{sign}\left(\sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

If f(x) < 0, classify as negative, if f(x) > 0, classify as positive

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1

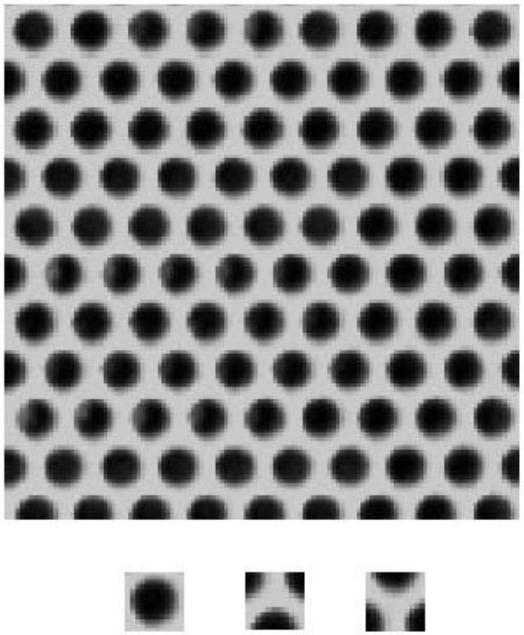
Bag of Features

Bag of Features

Object → Bag of 'words'

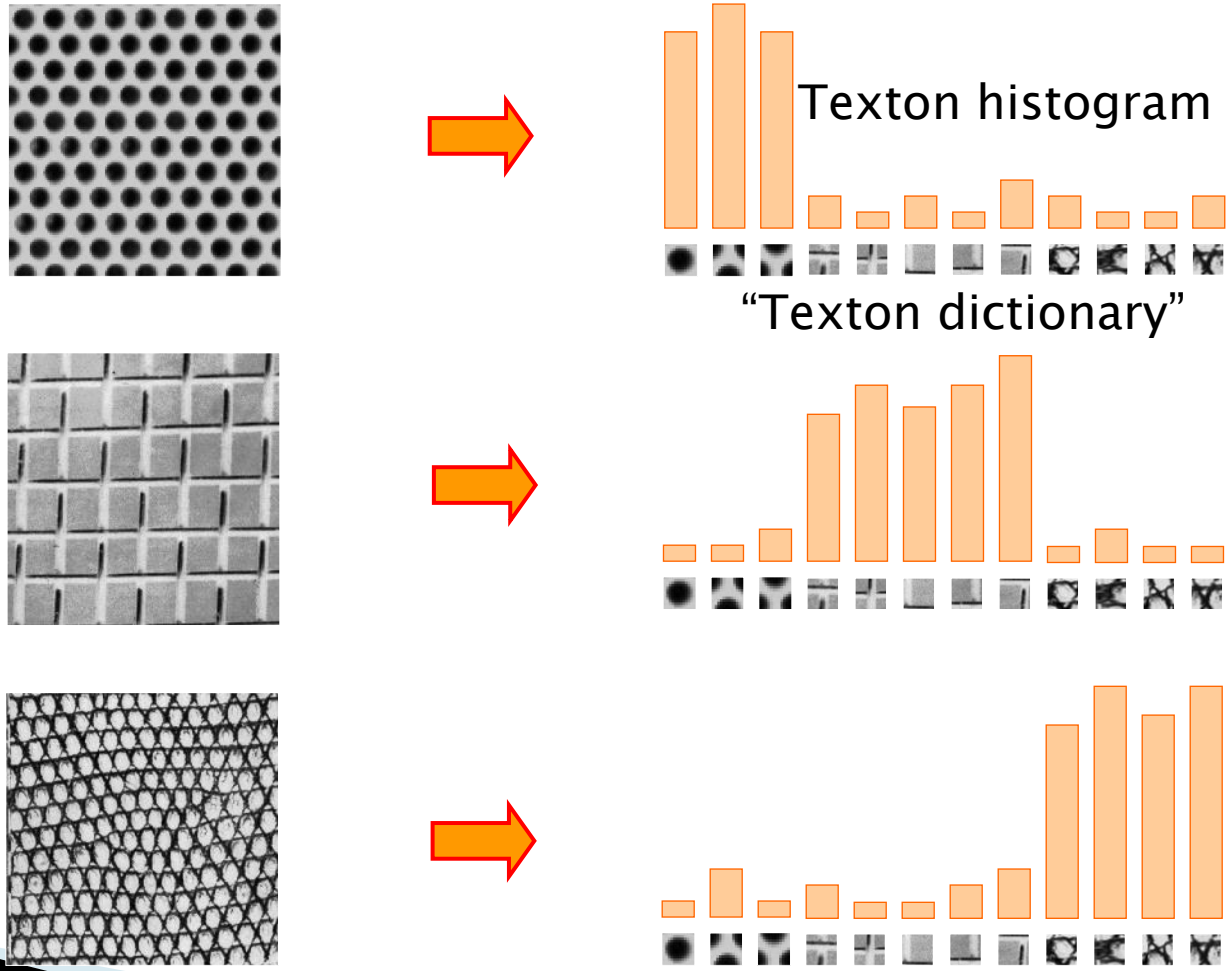
Texture models

- 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 models



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

Bags of words

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

Bags of words

Bags of words

Bags of words

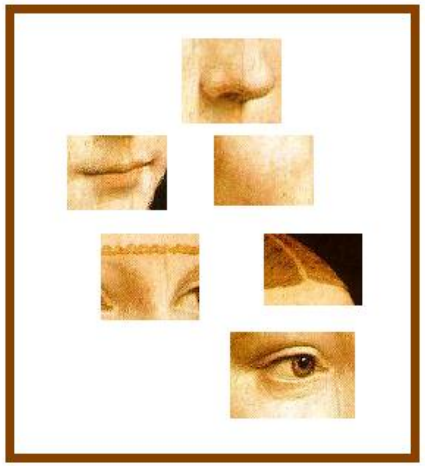
Bag of Features for Object Recognition

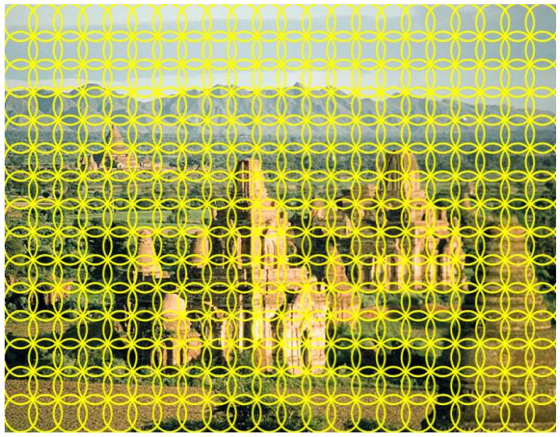
Csurka et al. (2004), Willamowski et al. (2005), Grauman & Darrell (2005), Sivic et al. (2003, 2005)

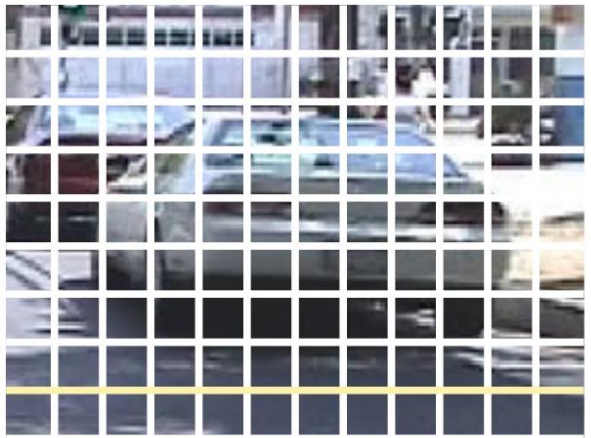
Overview of Bag of Features

- Extract local features.
- Learn visual vocabulary.
- Quantize local features.
- Represent images by frequencies of visual words.

Step 1: Extract Local Features

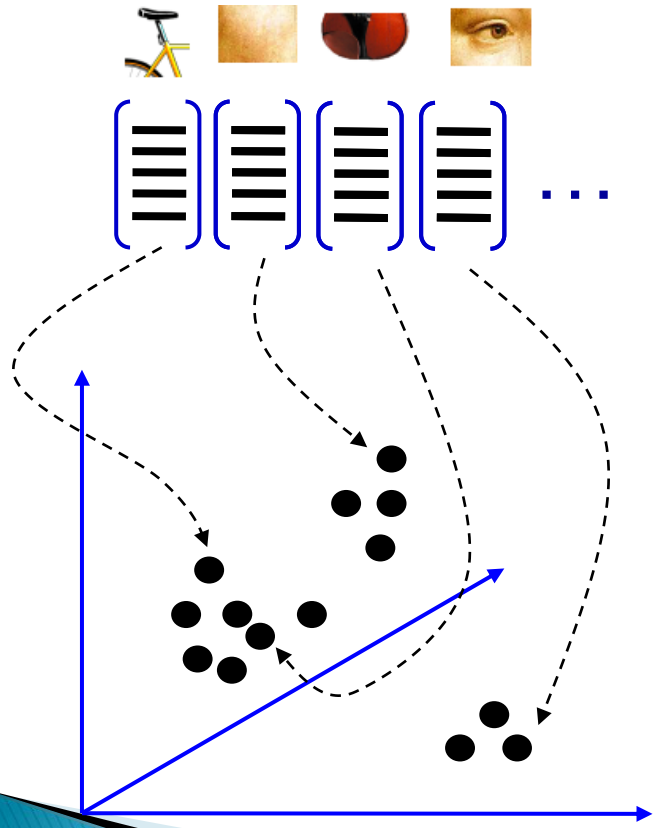






Slide credit: Juan Carlos Niebles

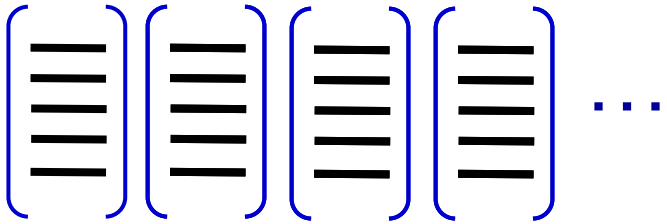
Step 2: Learn Visual Vocabulary

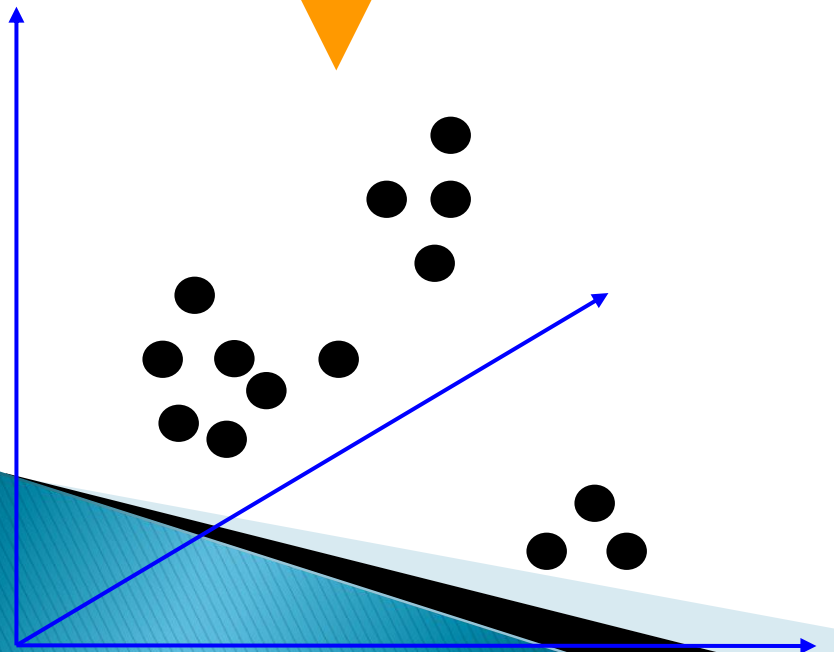


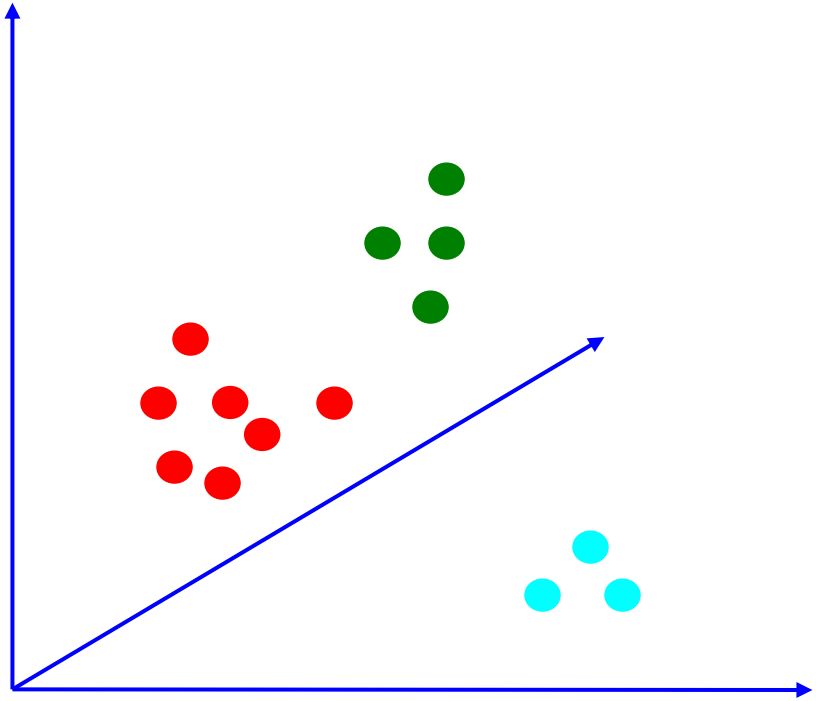
Extracted descriptors
from the training set

Slide credit: Josef Sivic

Step 2: Learn Visual Vocabulary



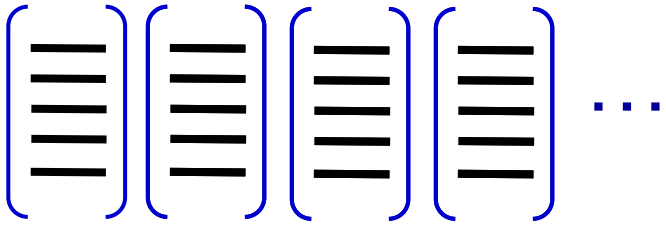


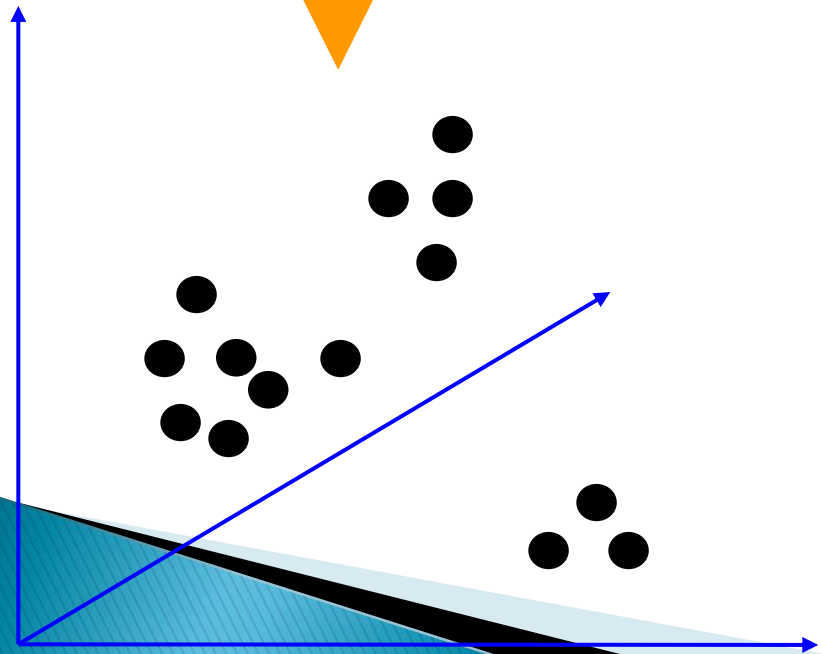


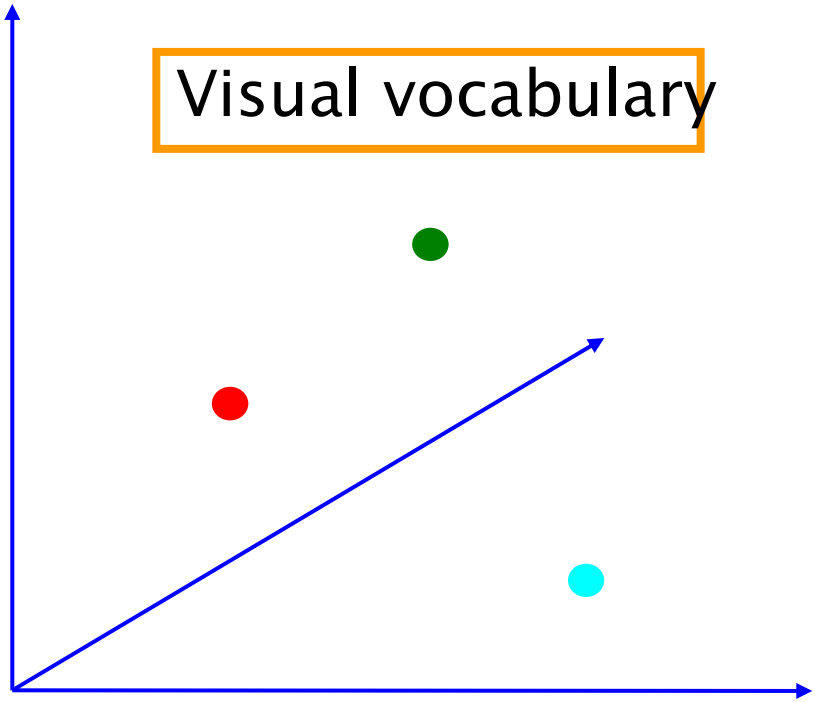
Clustering

Slide credit: Josef Sivic

Step 2: Learn Visual Vocabulary







Clustering

Slide credit: Josef Sivic

Recall: K-means clustering

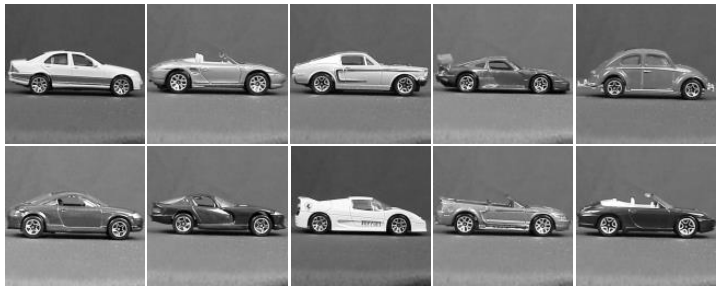
- Want to minimize sum of squared Euclidean distances between features $\mathbf{x}_i$ and their nearest cluster centers $\mathbf{m}_k$

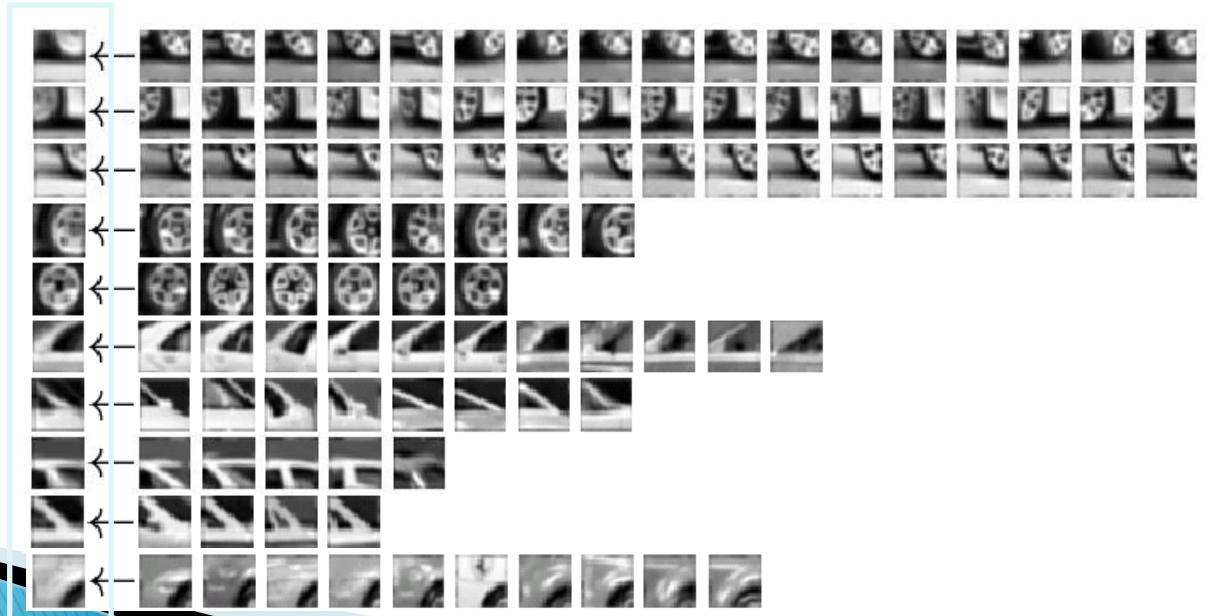
$$D(X, M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (\mathbf{x}_i - \mathbf{m}_k)^2$$

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

Slide credit: Juan Carlos Niebles

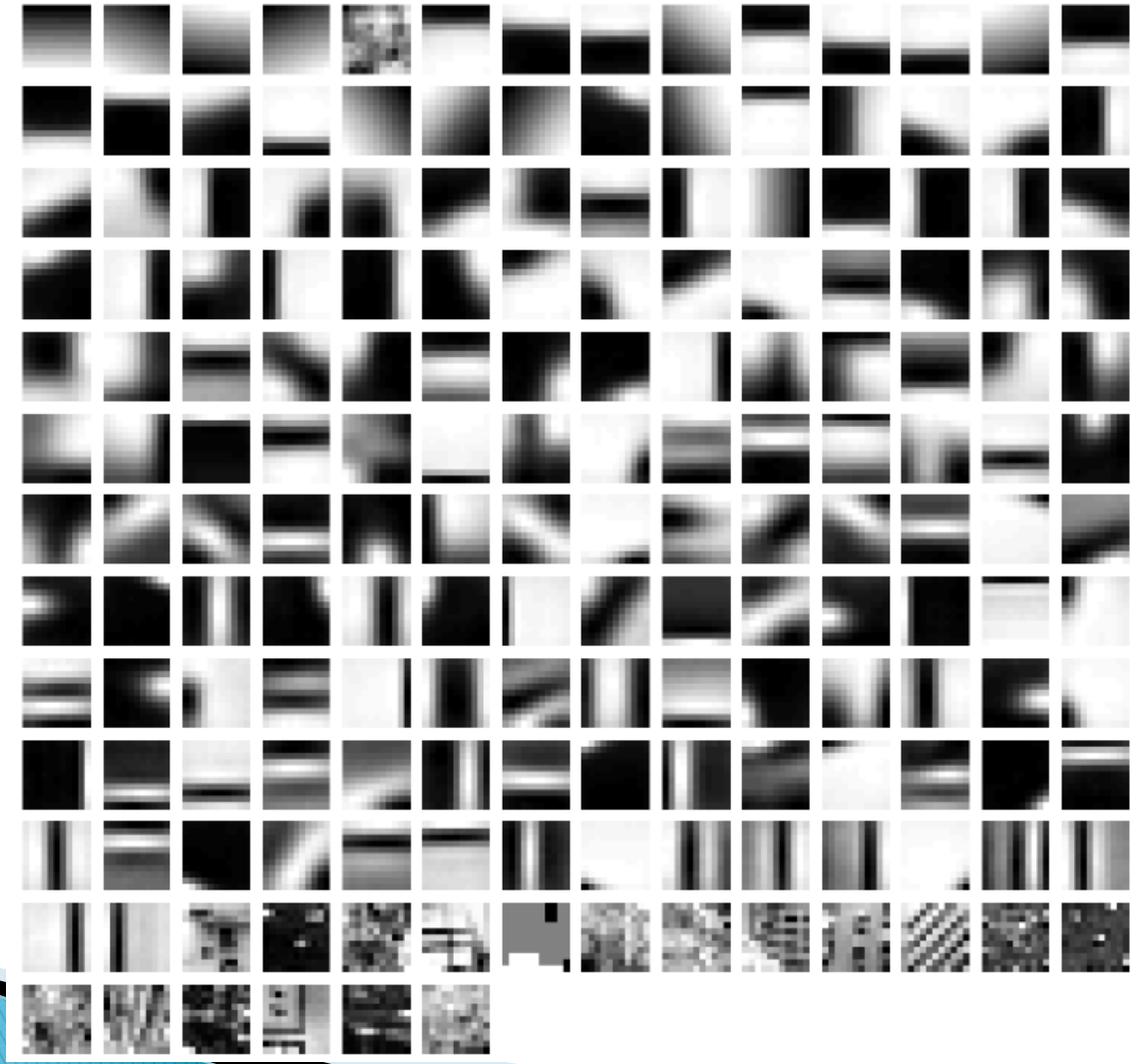
Visual Vacabularies





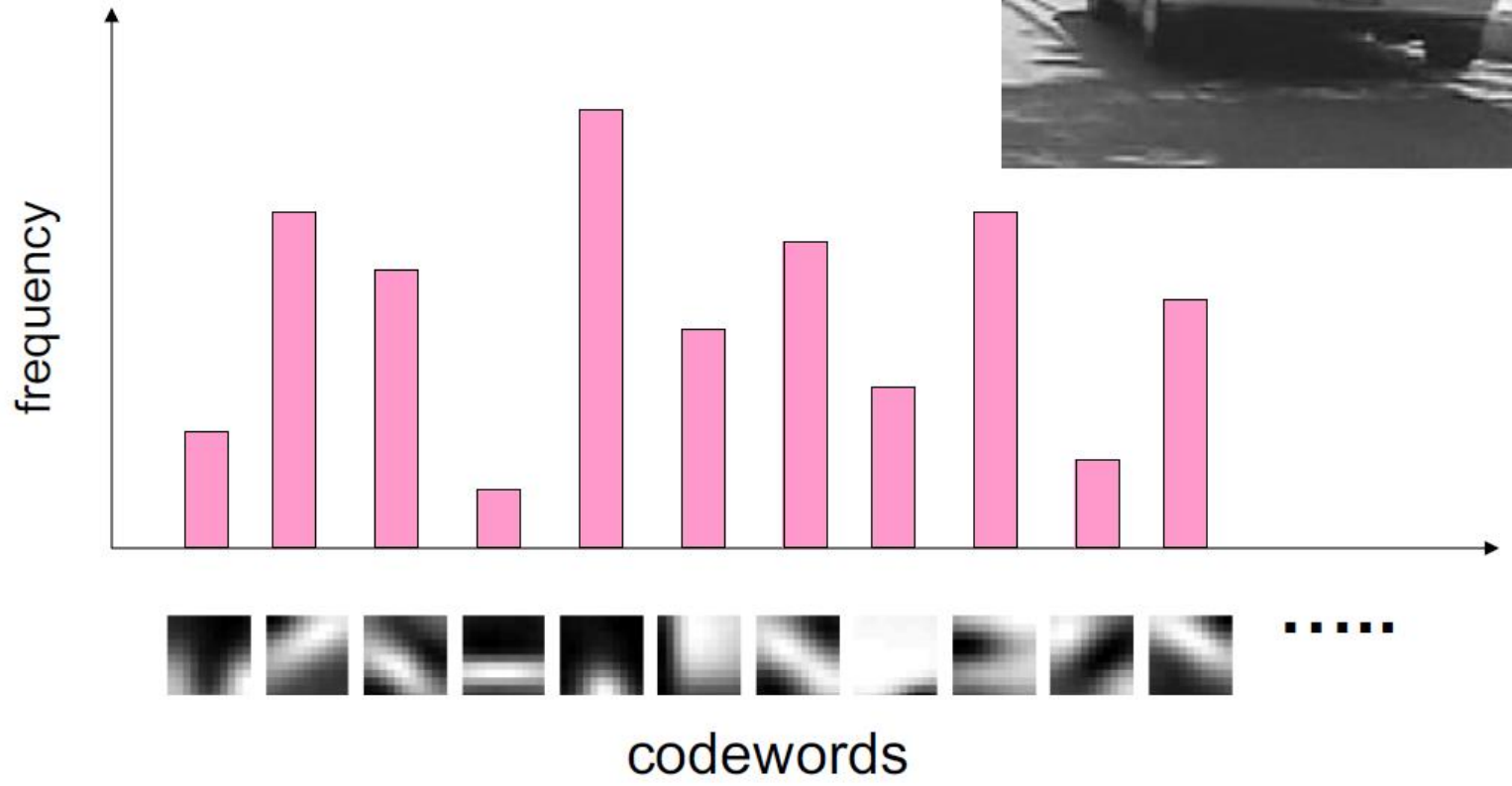
Source: B. Leibe

Example Visual Vocabularies



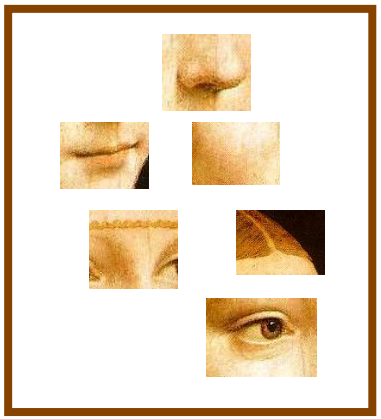
Fei-Fei et al. 2005

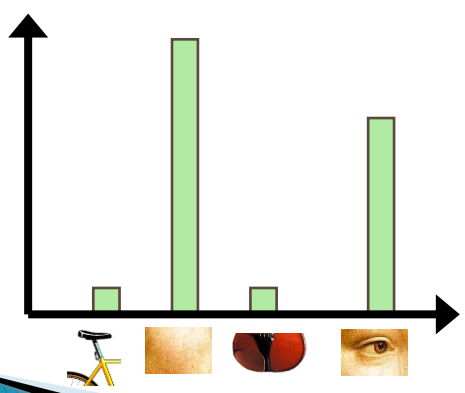
Step 3: Image Representation

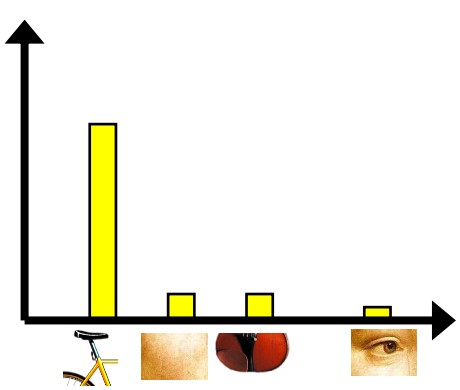


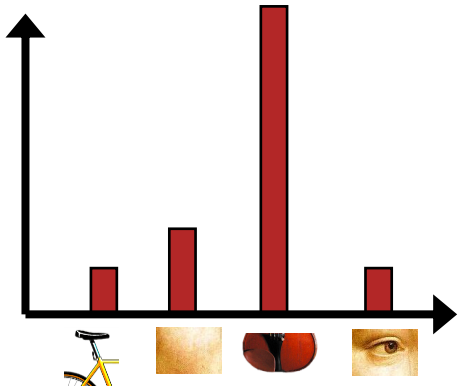
Slide credit: Juan Carlos Niebles

Step 3: Image Representation

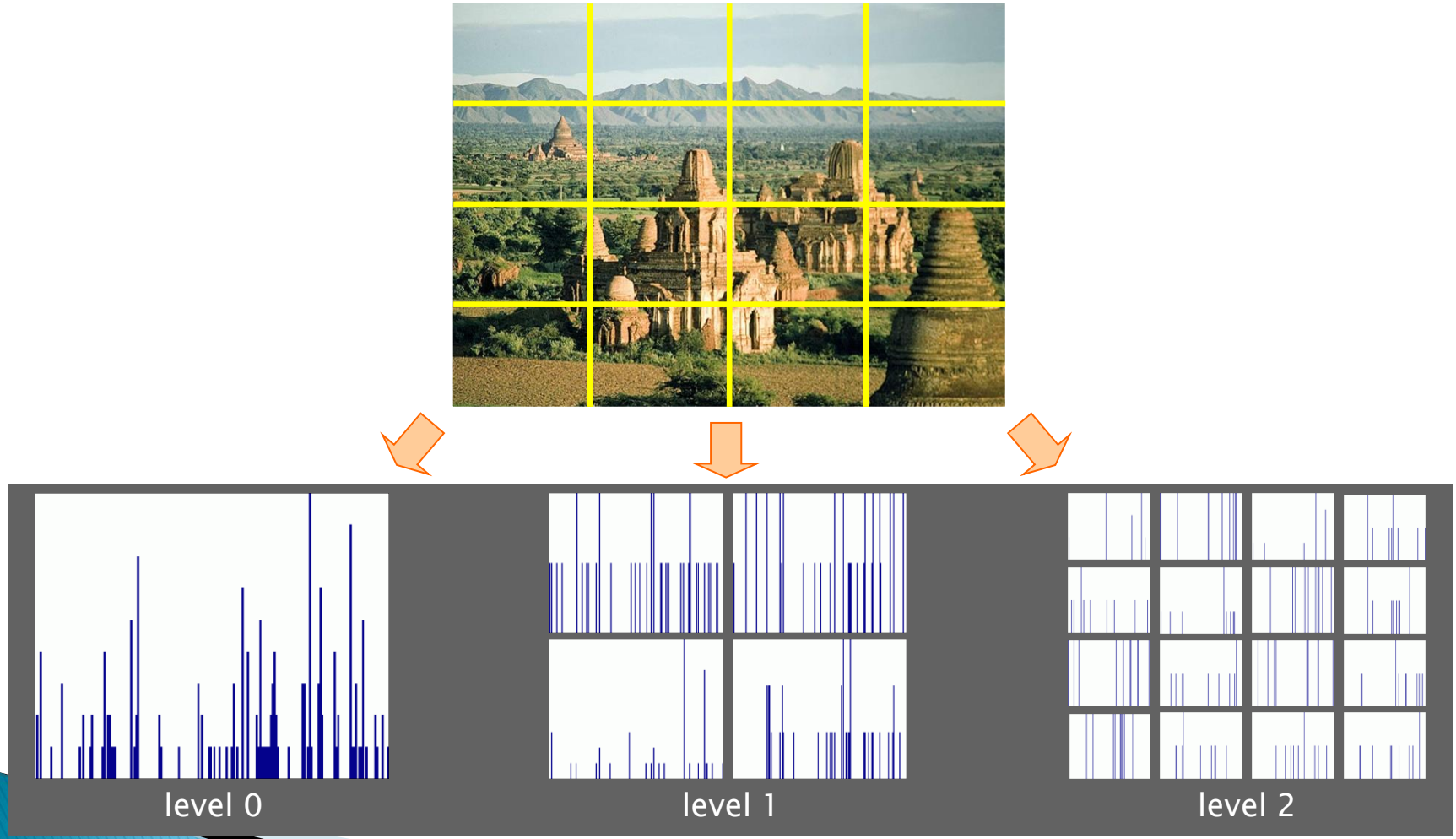








Spatial Pyramids



zebnik, Schmid & Ponce (CVPR 2006)