COMPUTATIONAL VISION:

OBJECT RECOGNITION BY BAG-OF-WORDS

Master in Artificial Intelligence

2021-2022

UNIVERSITAT DE BARCELONA

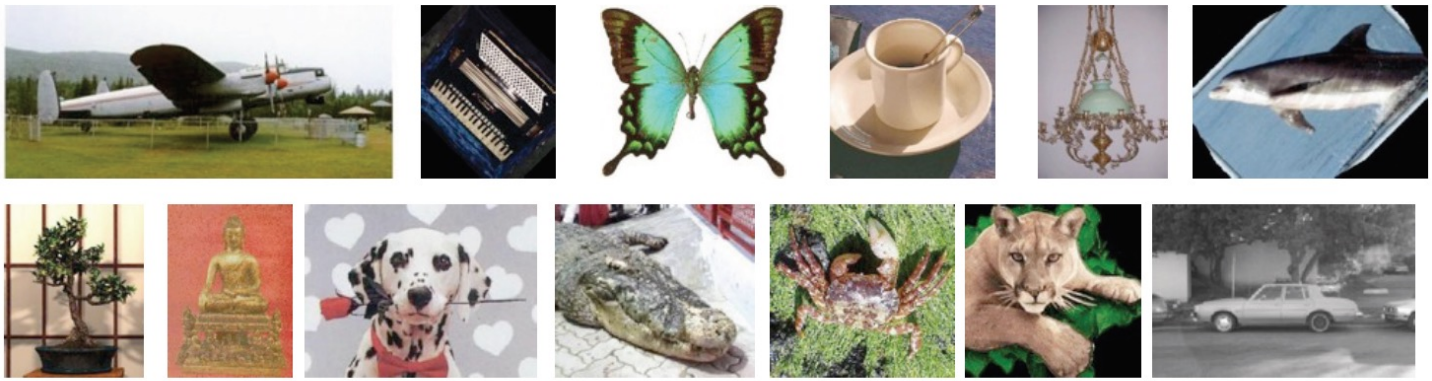
Overview: Bag-of-features models

- Motivation
- Image representation
  - Feature extraction
  - Visual vocabularies
- Discriminative methods for classification
  - Support vector machines
    - Kernels

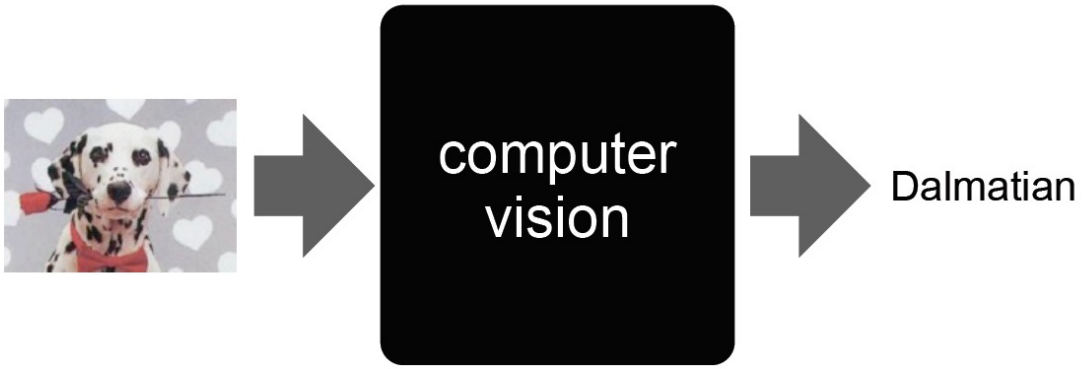
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Running example: Caltech-101

102 semantic labels



[Fei-Fei et al. 2003]



About 40 to 800 images per category, 9K in total. Most categories have about 50 images. Images are of variable sizes, with typical edge lengths of 200-300 pixels.

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Documents classification

Of all the sensory impressions proceeding to the brain, the visual experiences are the dominant ones. Our perception of the world around us is based essentially on the messages that r… our eyes. For a long ti… etinal image wa… isual centers i… s a movie s… image … discove… know th… perceptio… more comp… following the … ath to the various ce… ortex, Hubel and Wiesel hav… demonstrate that the *message abou… image falling on the retina undergoes … wise analysis in a system of nerve cells … stored in columns. In this system each c… has its specific function and is responsible … a specific detail in the pattern of the retinal image.*

sensory, brain, visual, perception, retinal, cerebral cortex, eye, cell, optical nerve, image Hubel, Wiesel

China is forecasting a trade surplus of $90bn (£51bn) to $100bn this year, a threefold increase on 2004's $32bn. The Commerce Ministry said the surplus would be created by a predicted 30% … $750bn, compared w… $660bn. T… annoy th… China's … deliber… agrees … yuan is … governo… also neede… demand so … country. China … he yuan against the dolla… nd permitted it to trade within a narrow … but the US wants the yuan to be allowed … le freely. However, Beijing has made it cl… t it will take its time and tread carefully be… allowing the yuan to rise further in value.

China, trade, surplus, commerce, exports, imports, US, yuan, bank, domestic, foreign, increase, trade, value

Origin 1: Bag-of-words models

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

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

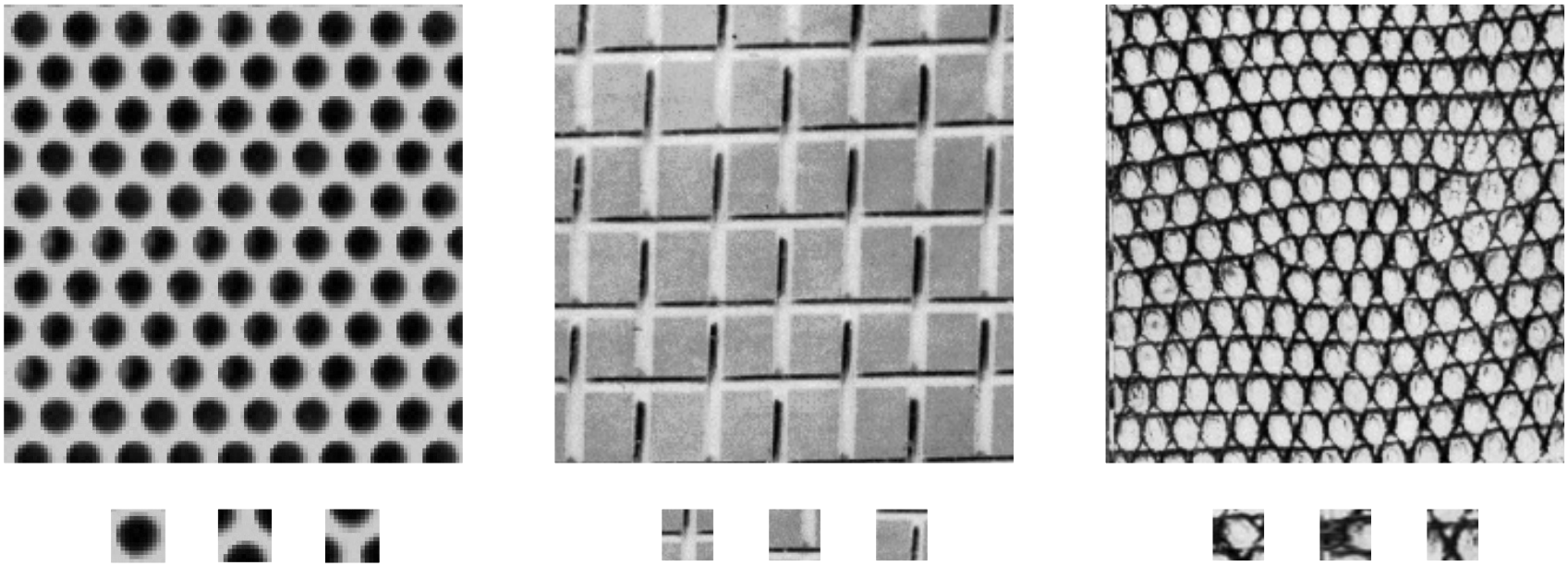
Origin 1: Bag-of-words models

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

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

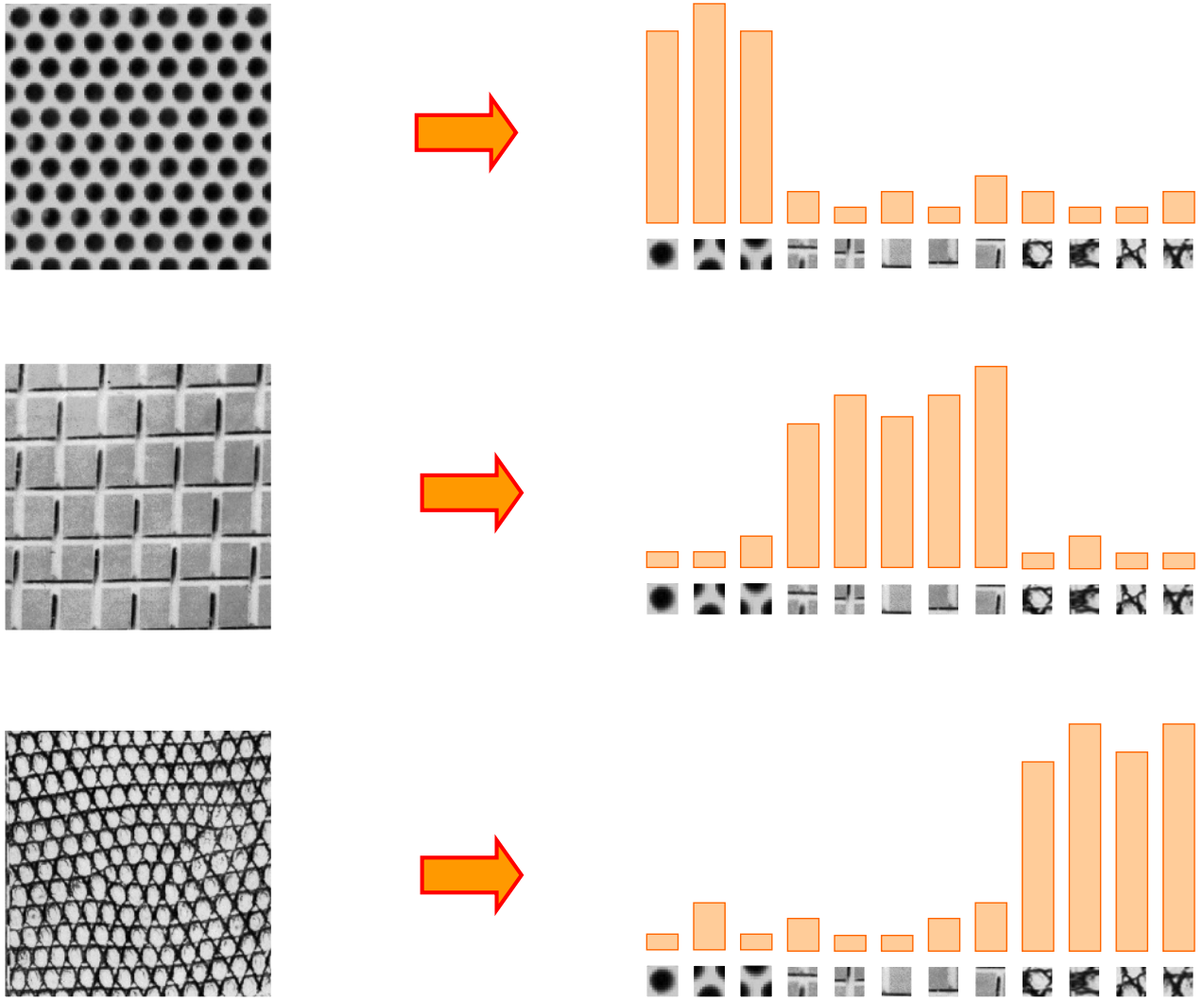
Origin 2: Texture recognition

- Texture is characterized by the repetition of basic elements or *textons*
- For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters



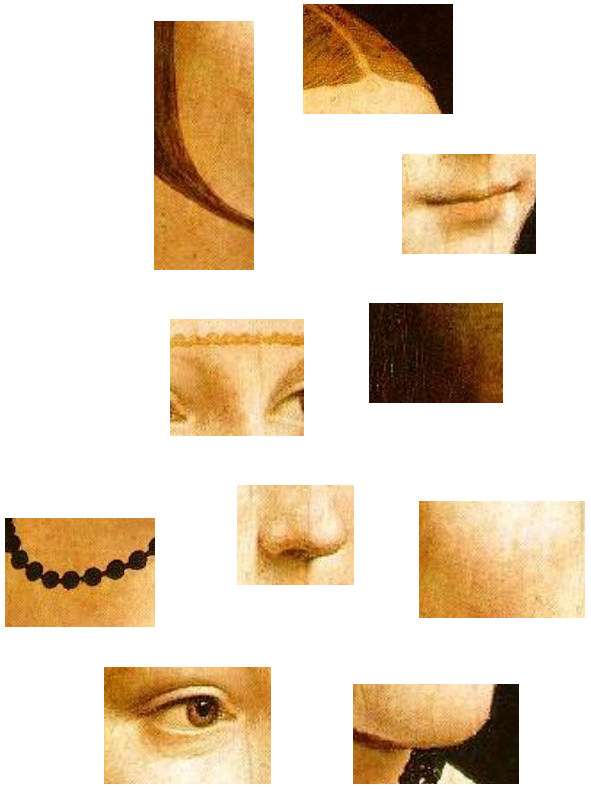
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Origin 2: Texture recognition



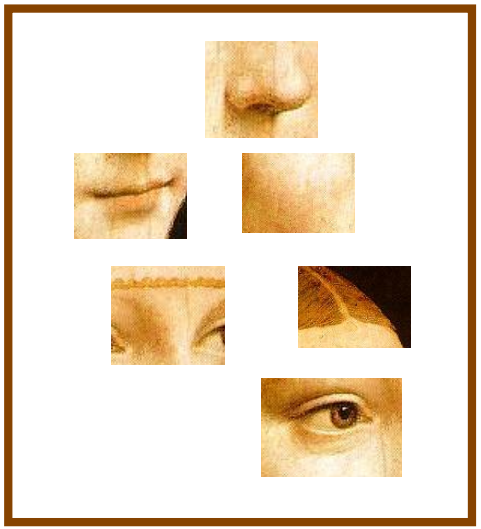
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Can we describe objects as bag-of-words?



Bag of features: outline

1. Extract features



Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

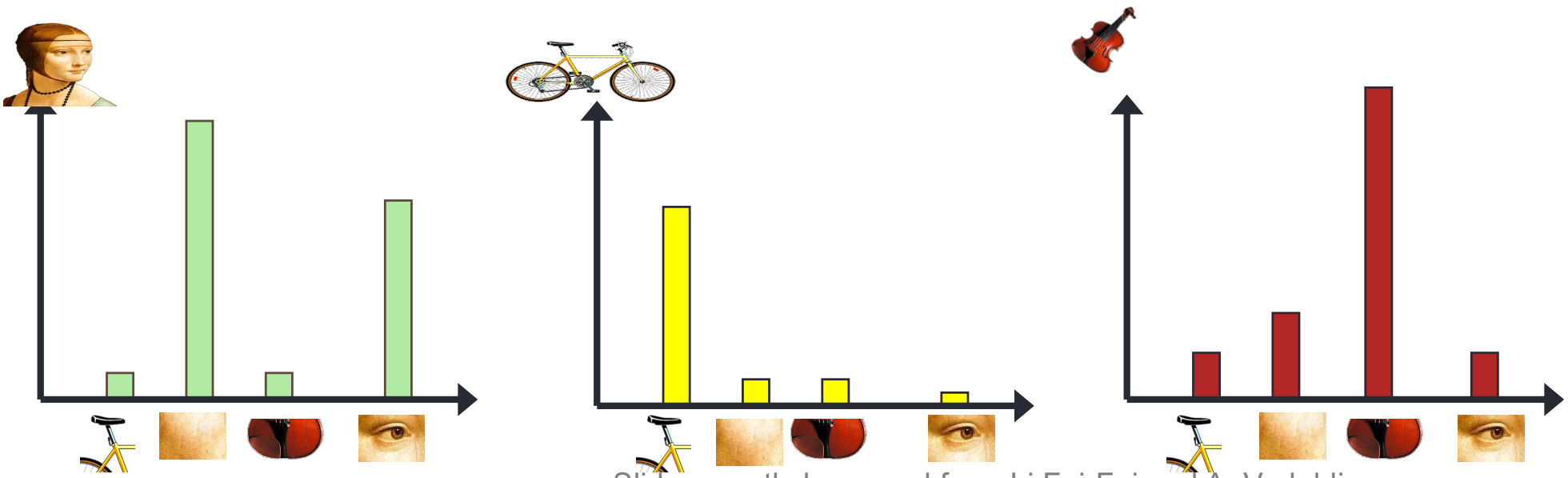
Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”

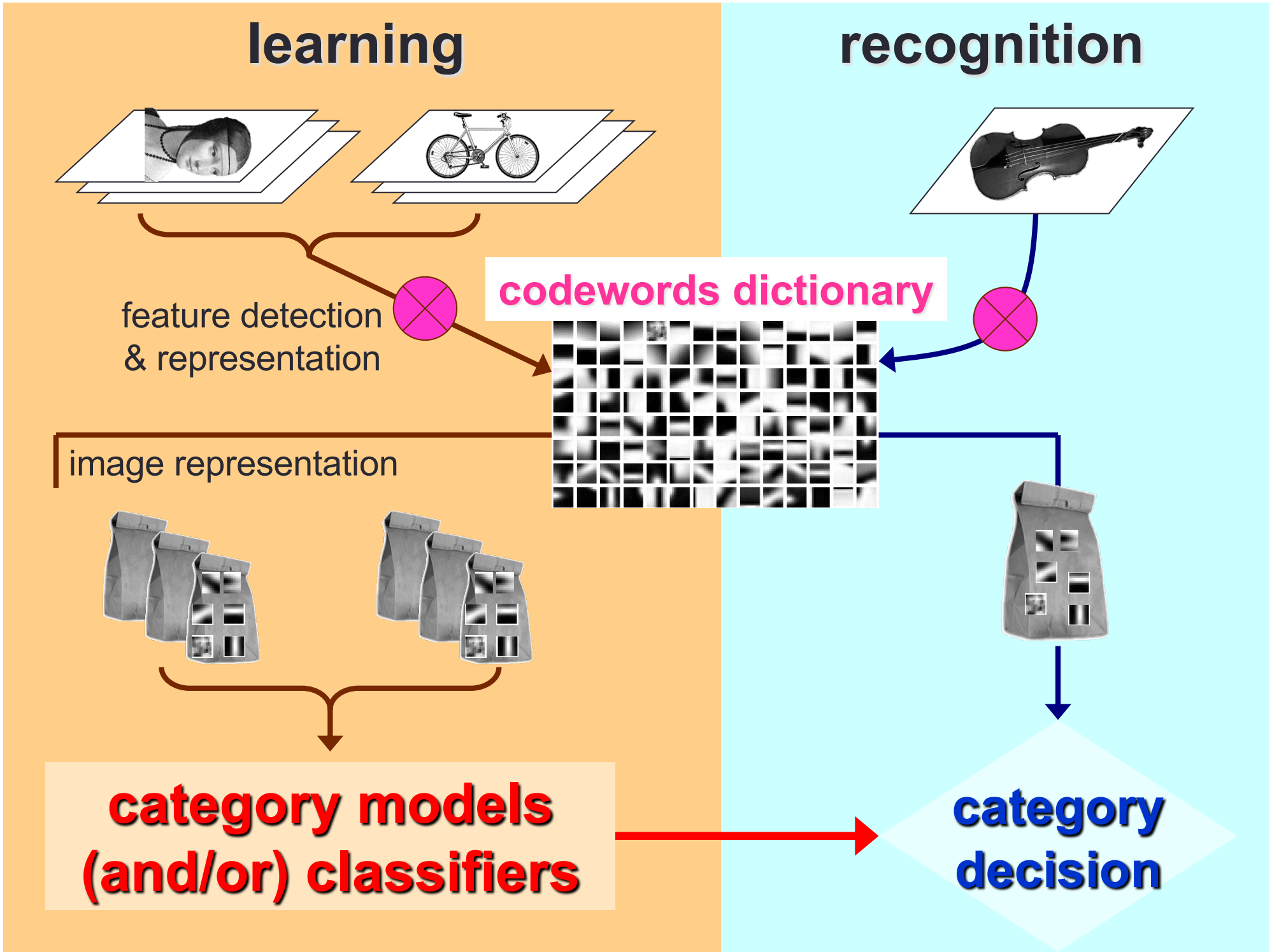
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Bag of features: outline

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

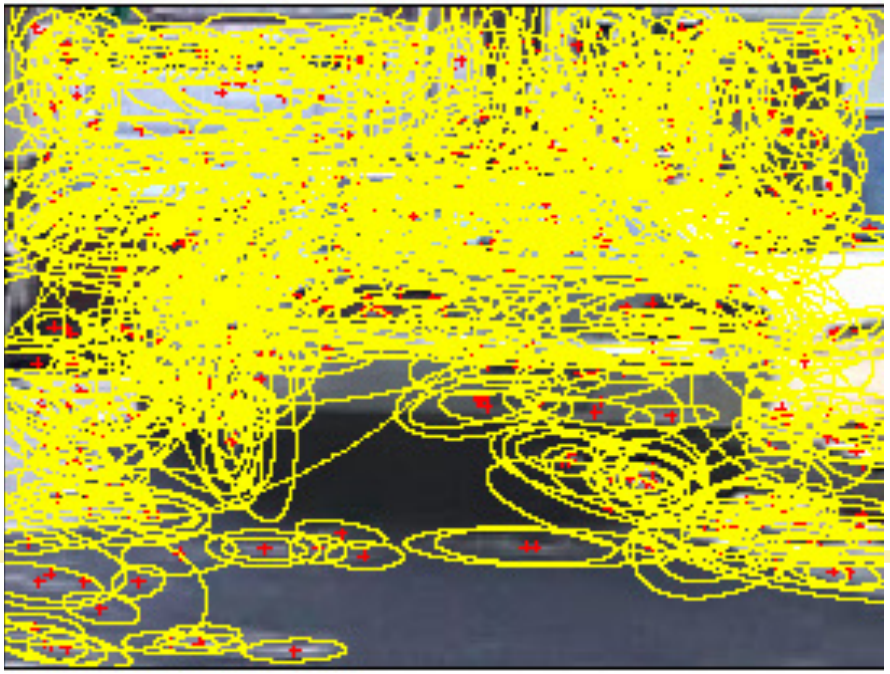


Slides mostly borrowed from Li Fei-Fei and A. Vedaldi


learning
recognition
feature detection
& representation
codewords dictionary
image representation
category models
(and/or) classifiers
category
decision

1. Feature extraction

- Regular grid
- Interest point detector

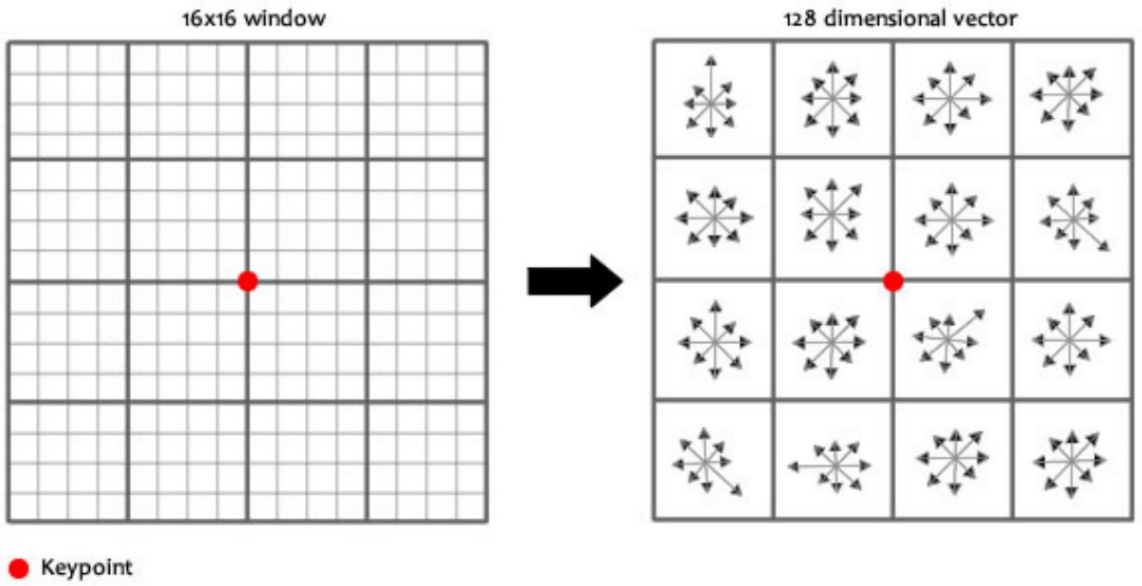


Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

SIFT descriptor

- SIFT **keypoint** [Lowe 2004]
  - extrema of the DoG scale space
  - blob-like image structure
  - oriented

- SIFT **descriptor**
  - 4 x 4 spatial histogram of gradient orientations
  - linear interpolation
  - Gaussian weighting



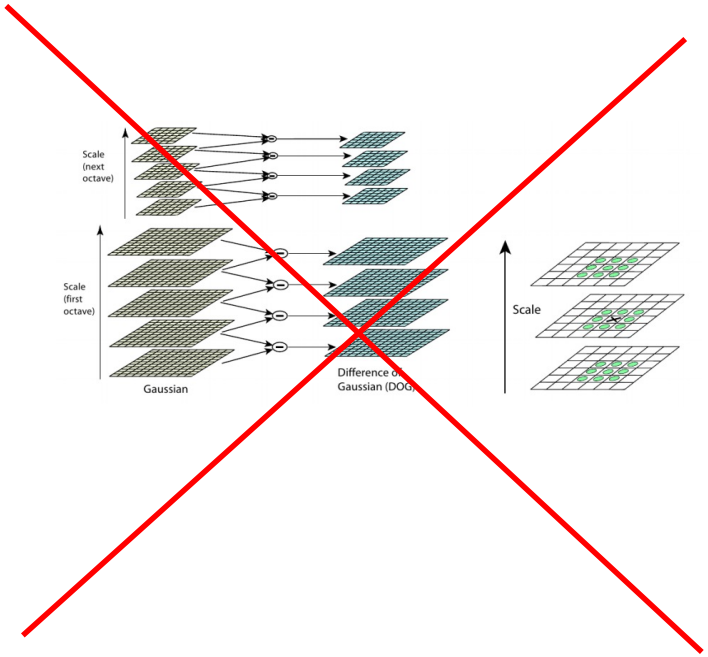
What is the size of the Sift descriptor?

PHOW
(dense SIFT)

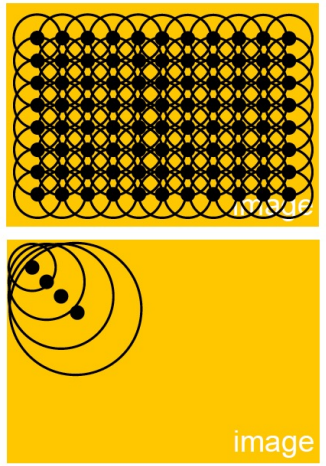
Pyramid Histogram of Visual **Words** (**PHOW**)

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Dense SIFT: PHOW features



- **PHOW features**
  - [Bosch et al. 2006], [Bosch et al. 2007]
  - dense multiscale SIFT
  - uniform spacing (e.g. 5 pixels)
  - four scales (e.g. 5, 7, 10, 12 pixels)
- **Direct implementation**
  - for each scale
    - create a list of keypoints

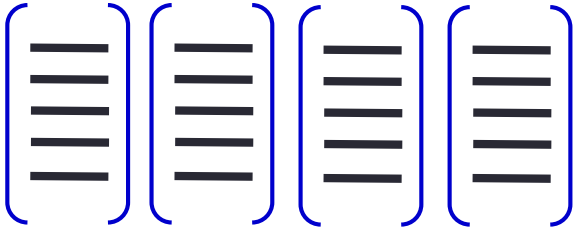


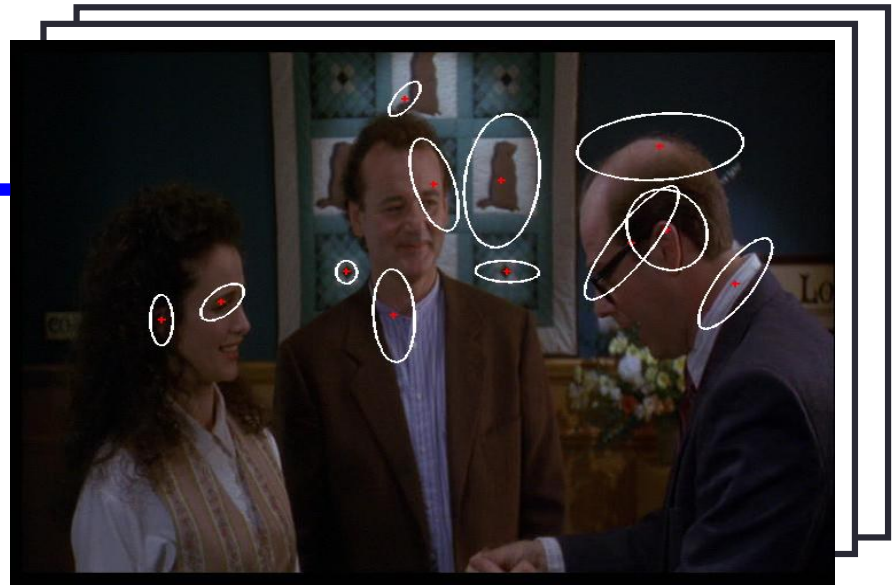
Large speed-up due to the uniform scale and sampling!

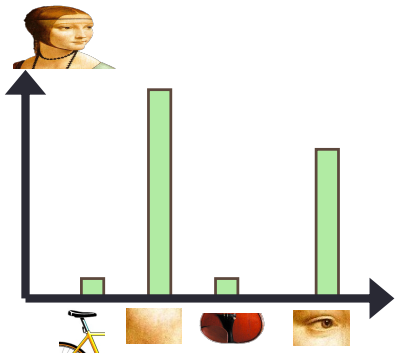
Feature extraction

PHOW

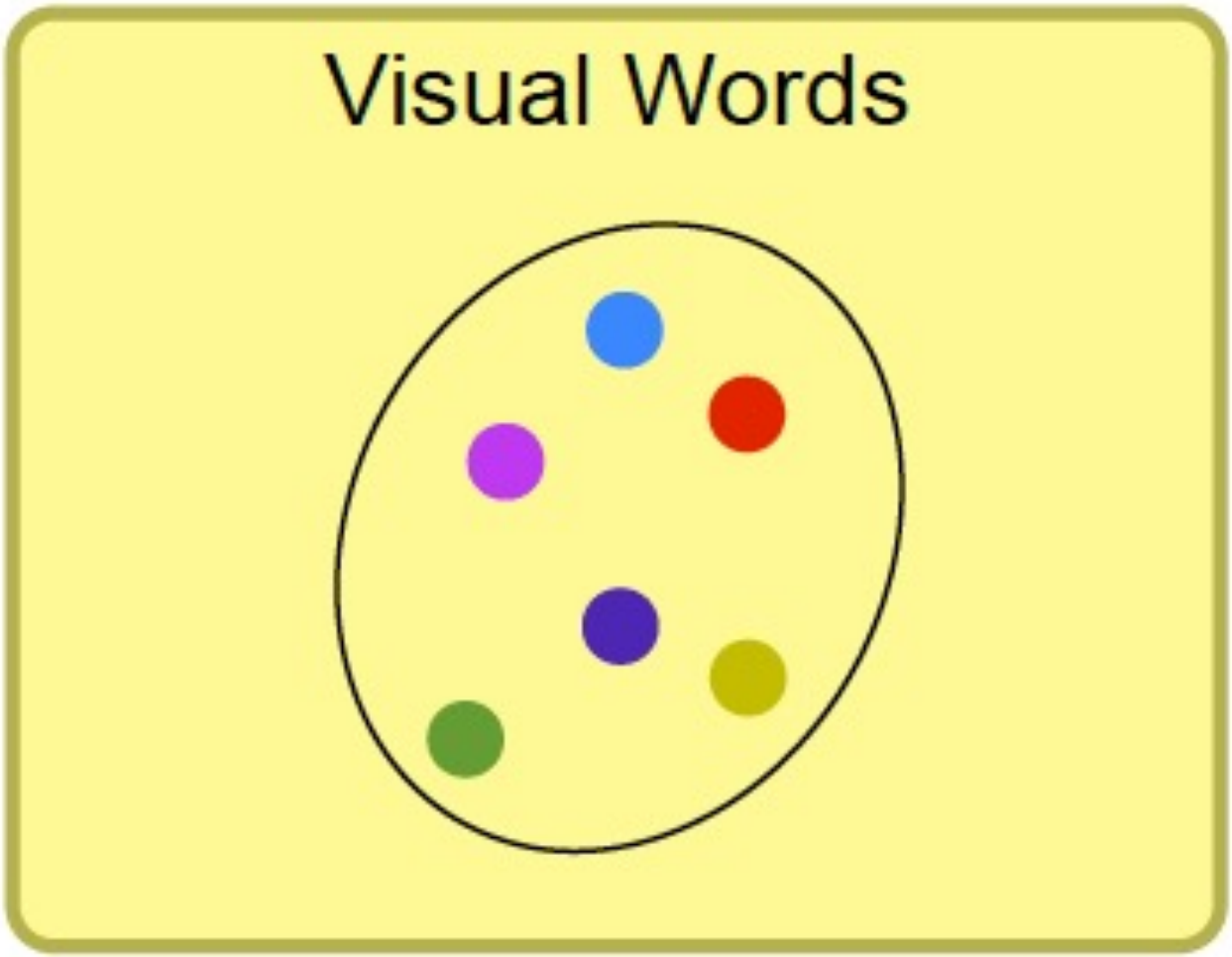
descriptors





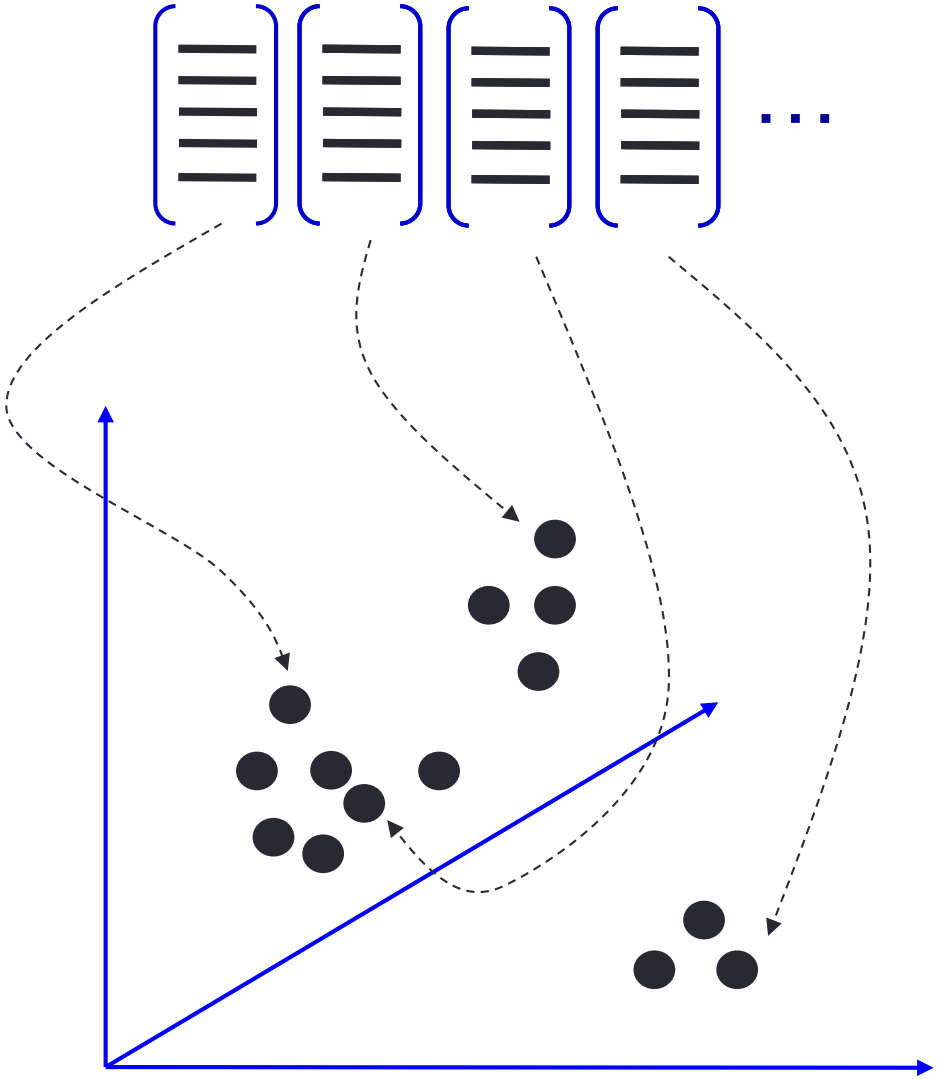


Visual Words



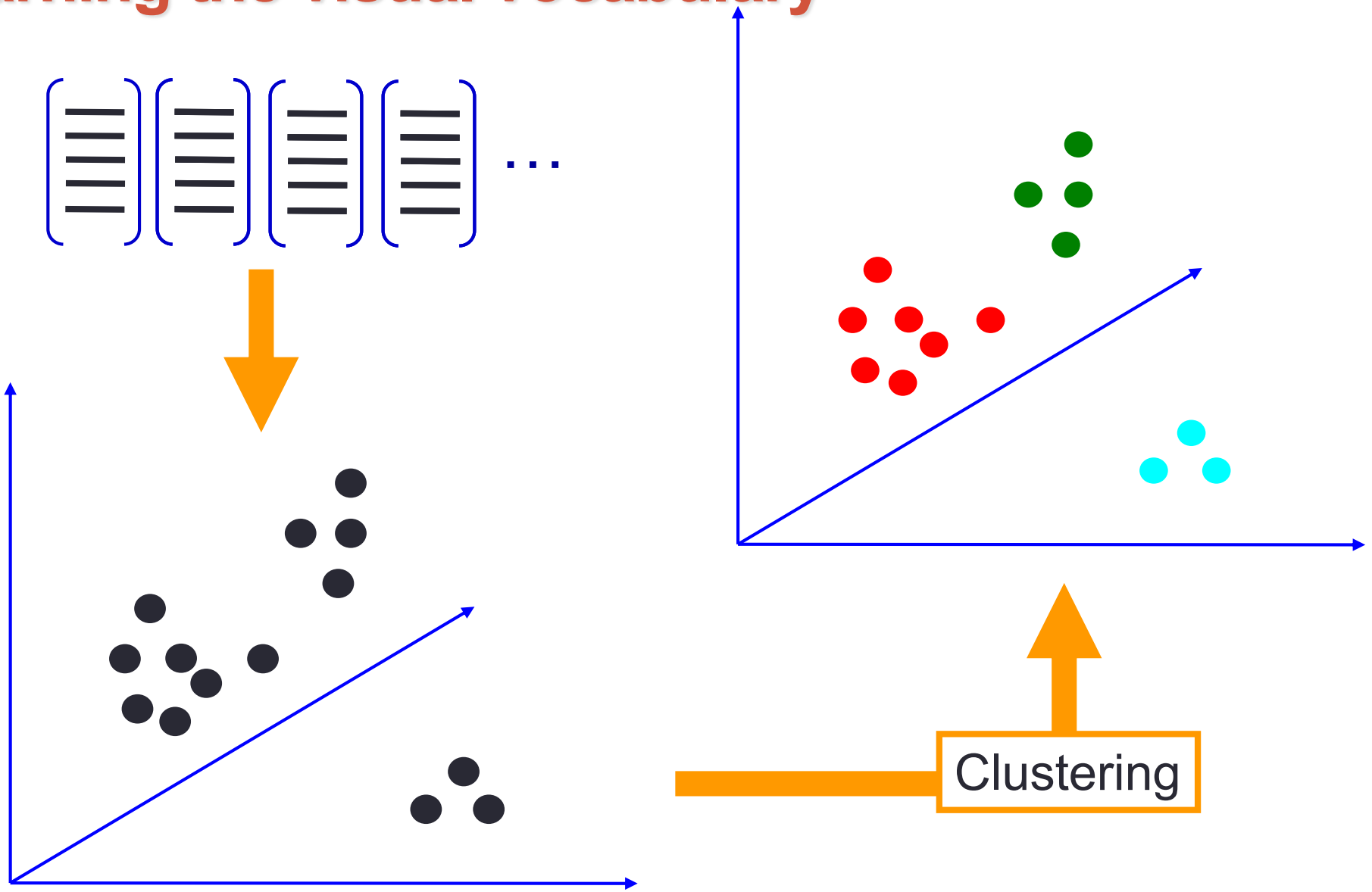
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Learning the visual vocabulary



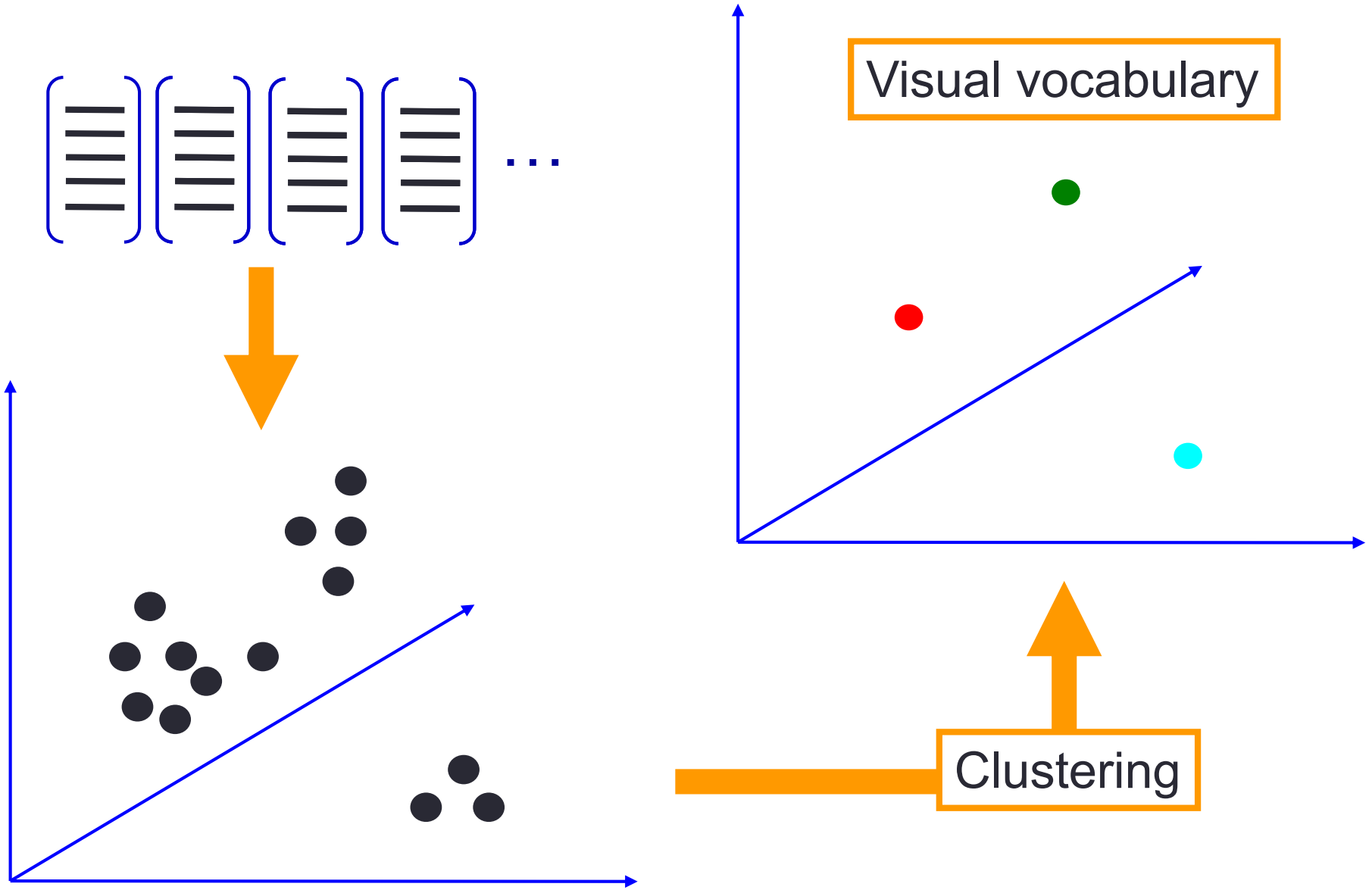
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Learning the visual vocabulary



Slide credit: Josef Sivic

Learning the visual vocabulary



Slide credit: Josef Sivic

K-means clustering

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

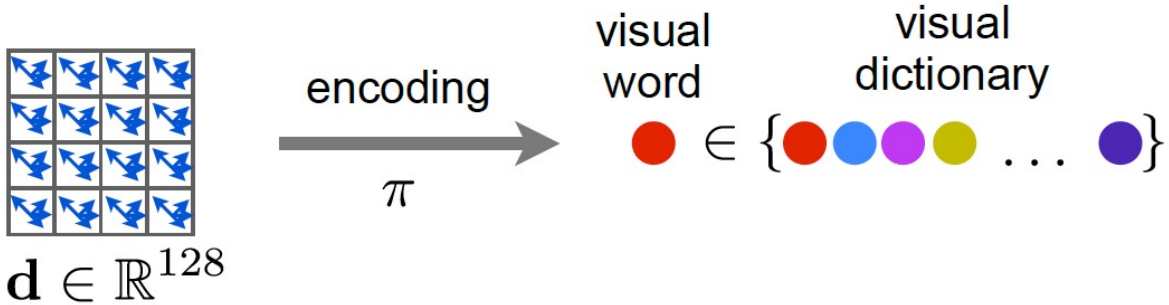
$$D(X, M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (x_i - m_k)^2$$

Algorithm:

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

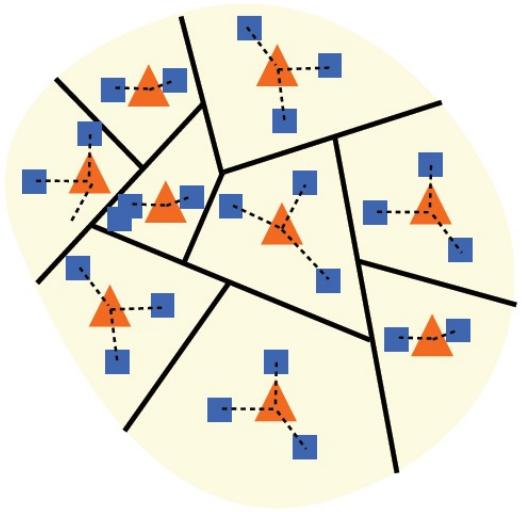
From clustering to vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be "universal"
- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary, Codevector = visual word
  - Visual words



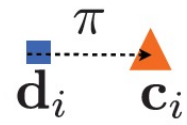
Visual words

- Optimize **reconstruction cost**

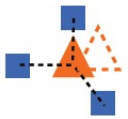


$$E(\pi, \mathbf{c}_1, \ldots, \mathbf{c}_K) = \sum_{i=1}^{M} \|\mathbf{d}_i - \mathbf{c}_{\pi(\mathbf{d}_i)}\|^2$$

1. reassign points to closest centers



2. fit centers to assigned points



- As simple as you expect

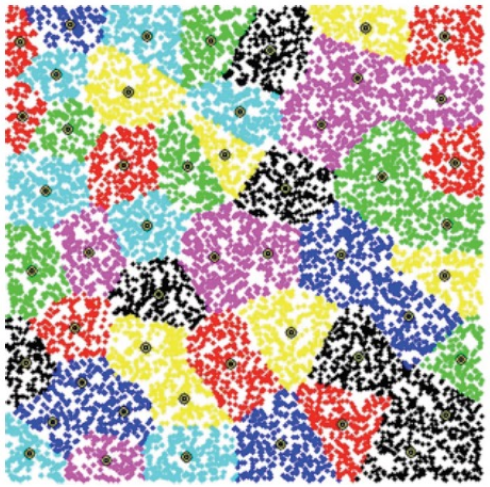
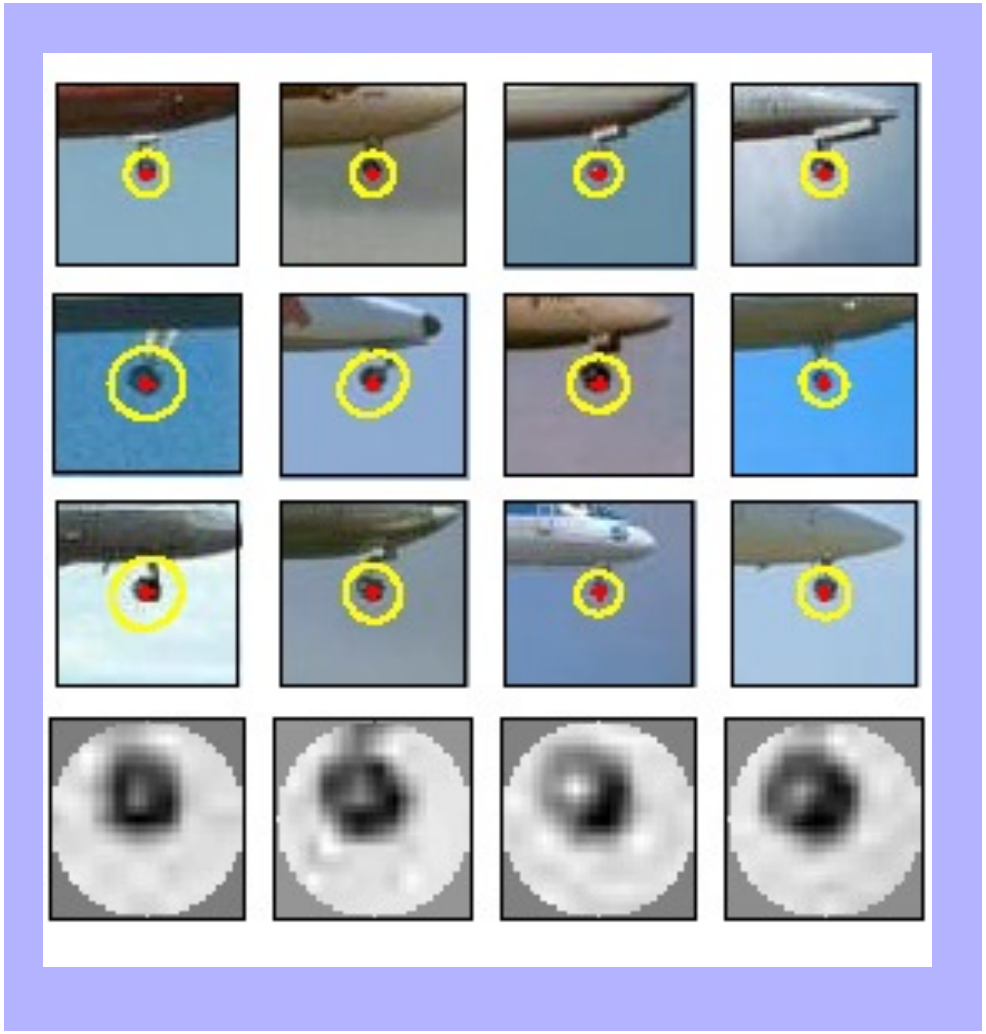
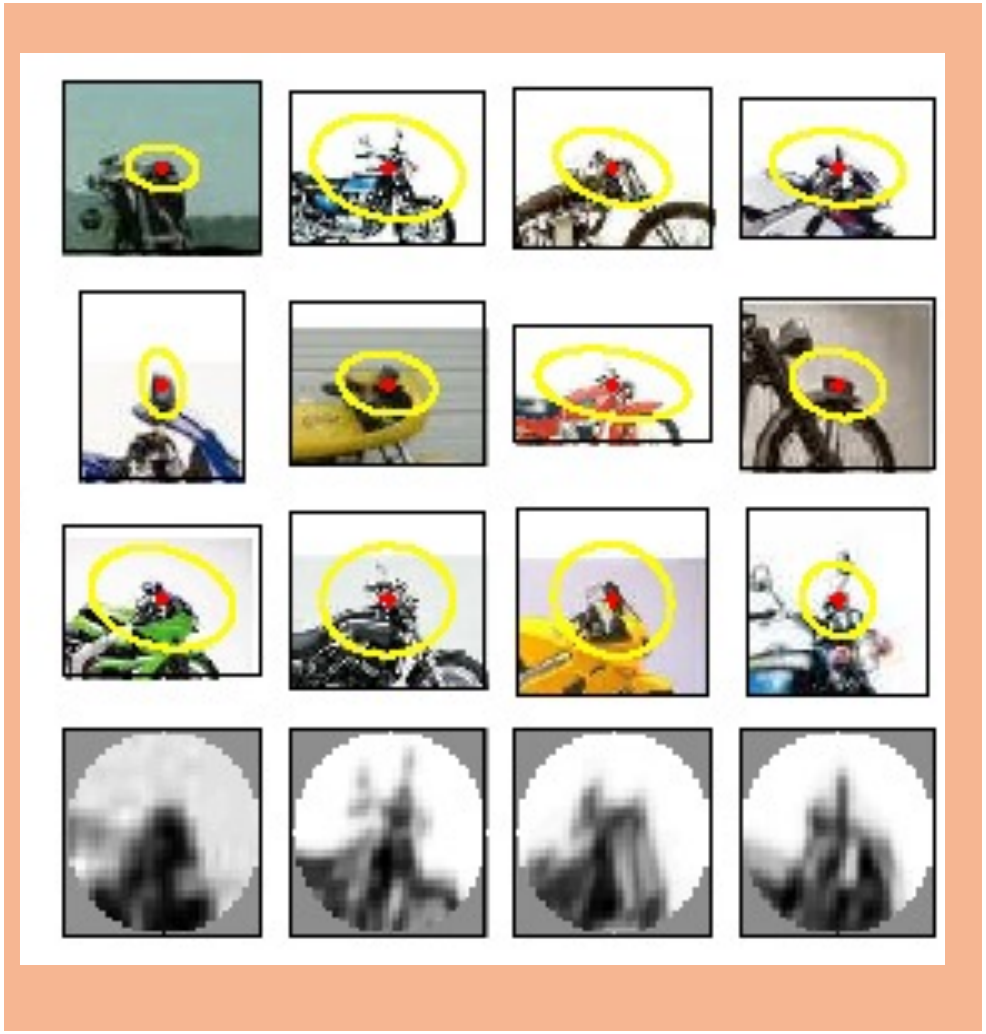


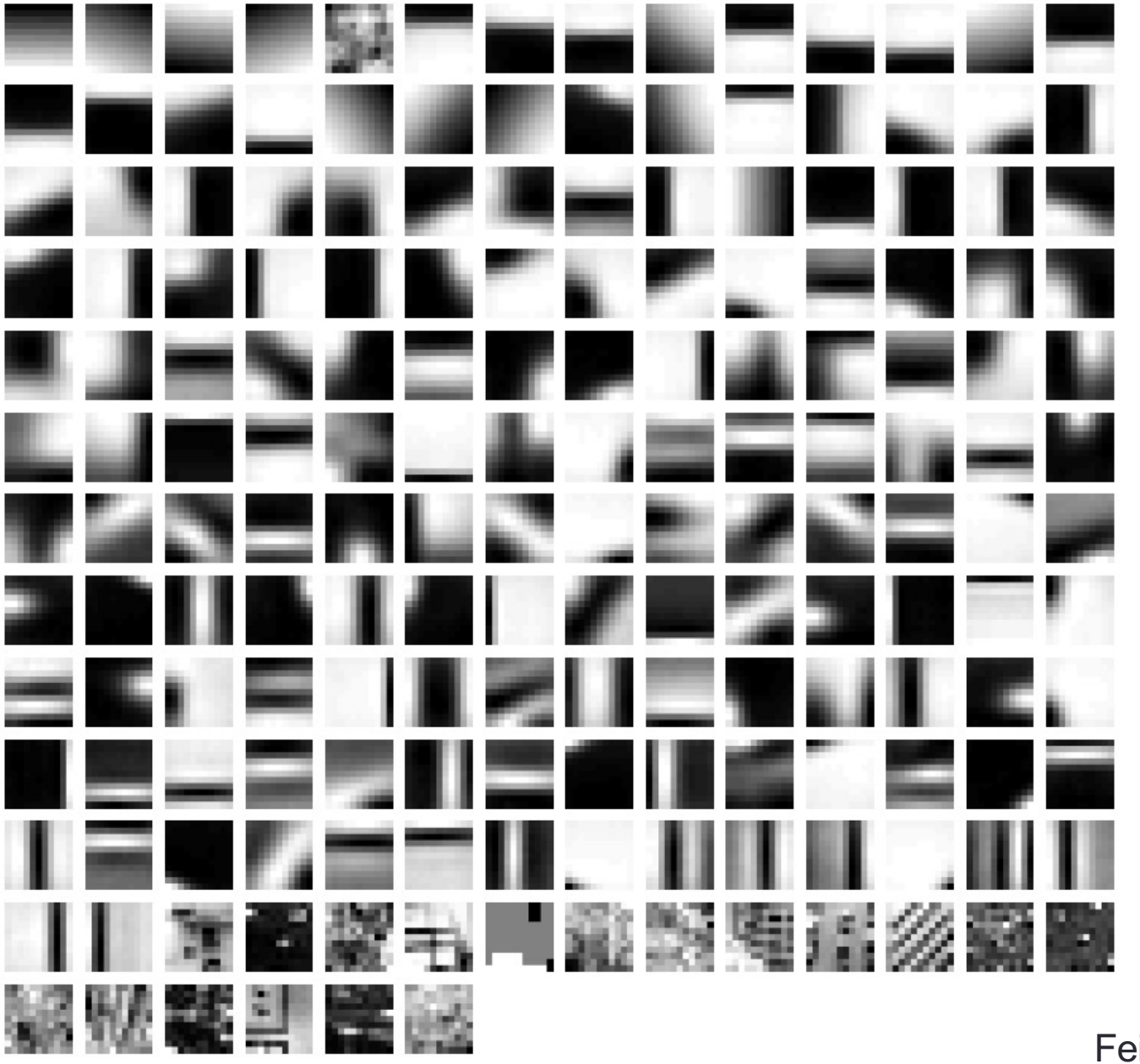
Image patch examples of visual words





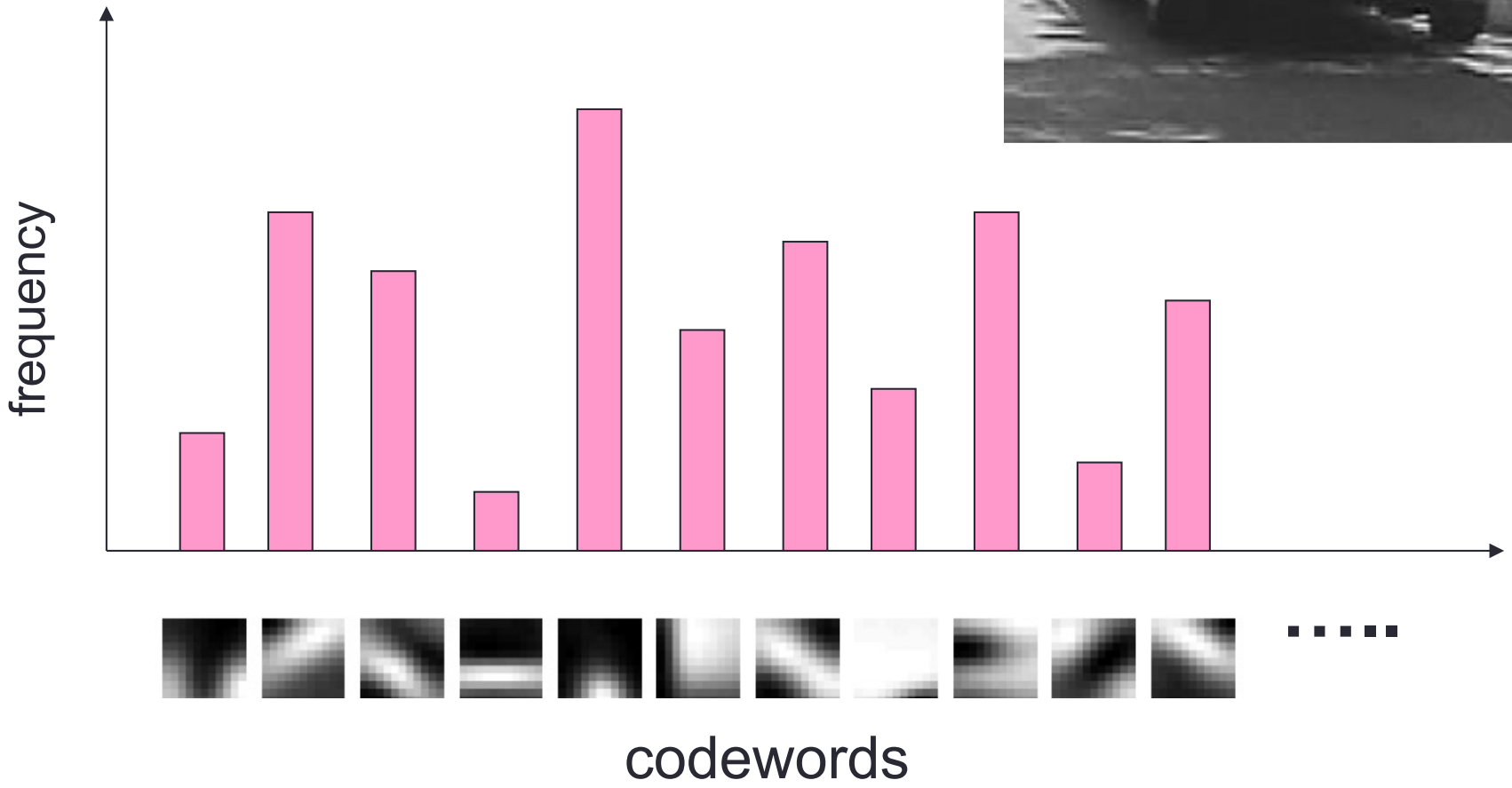
Sivic et al. 2005

Example visual vocabulary



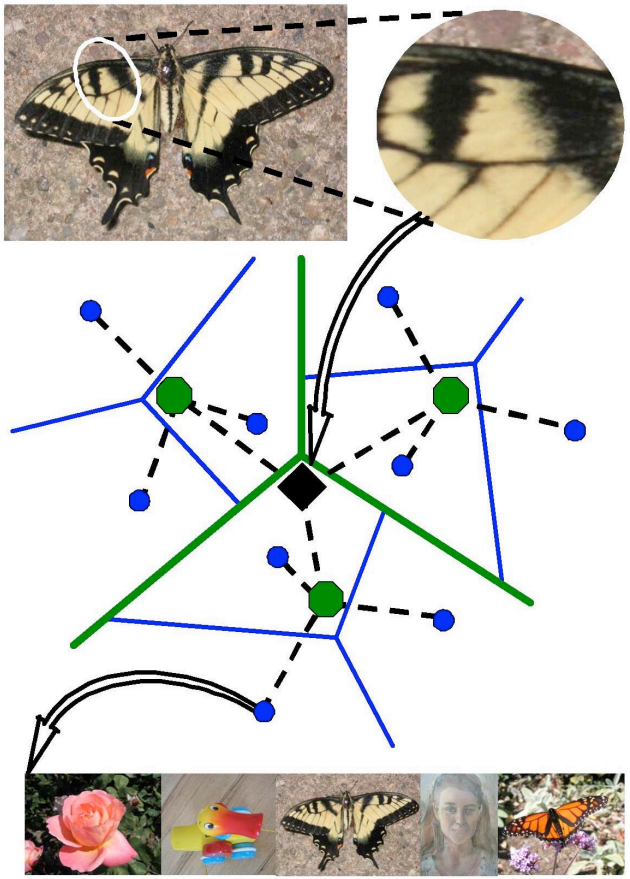
Fei-Fei et al. 2005

Image representation



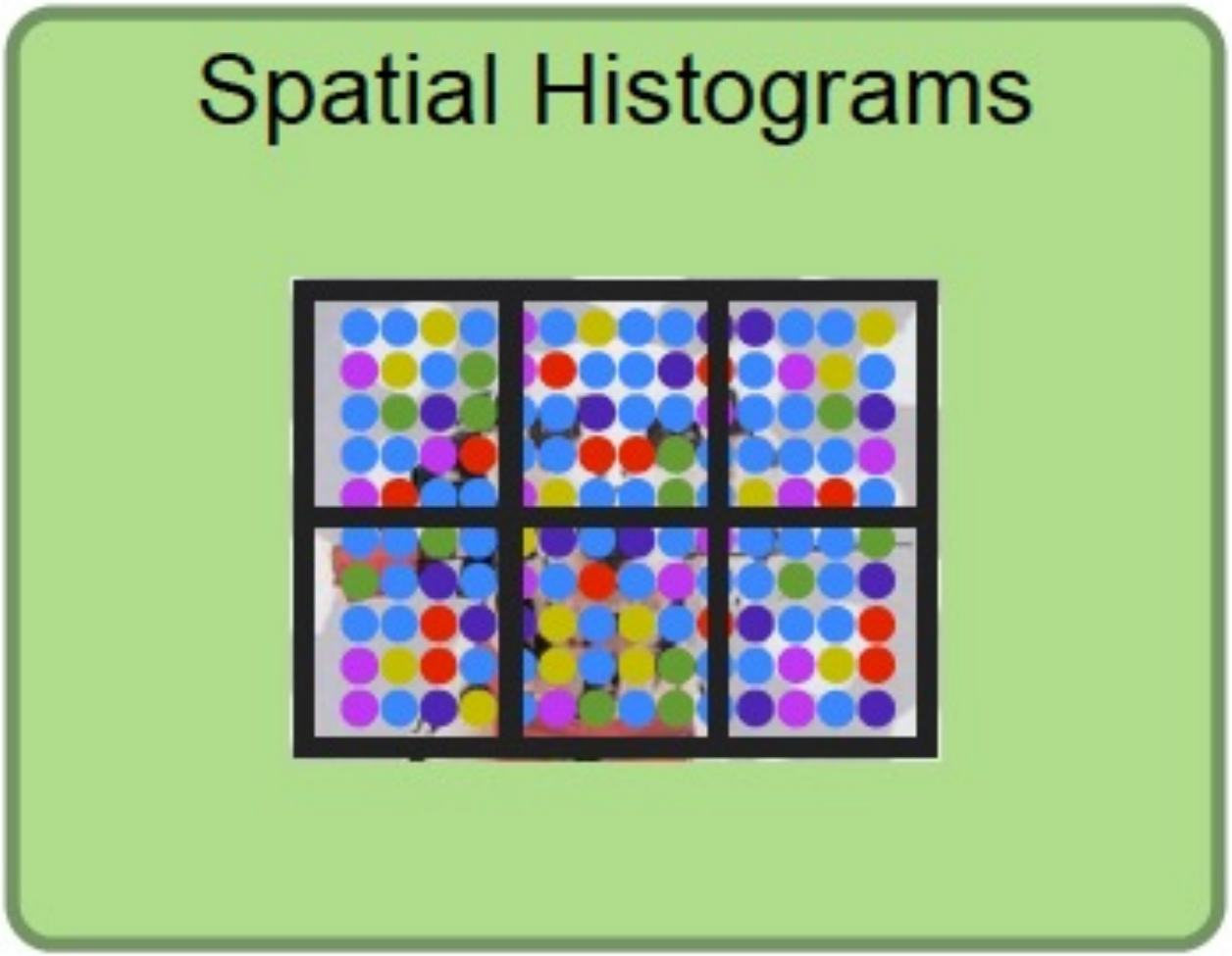
Visual vocabularies: Issues

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



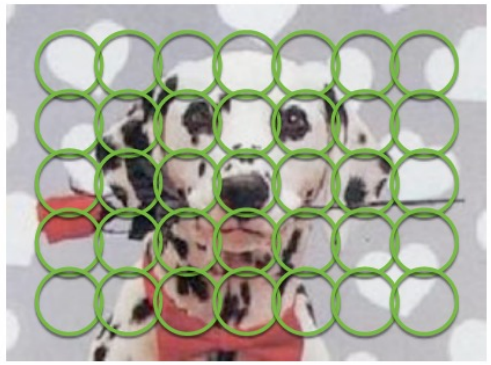
Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

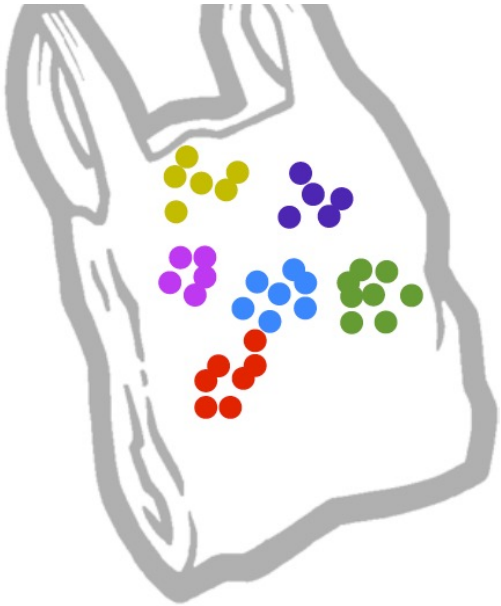
Spatial histograms

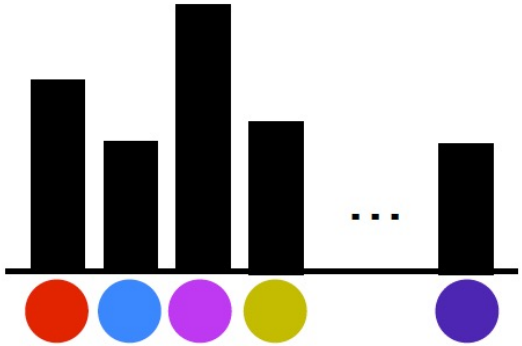


Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Bag-of-words





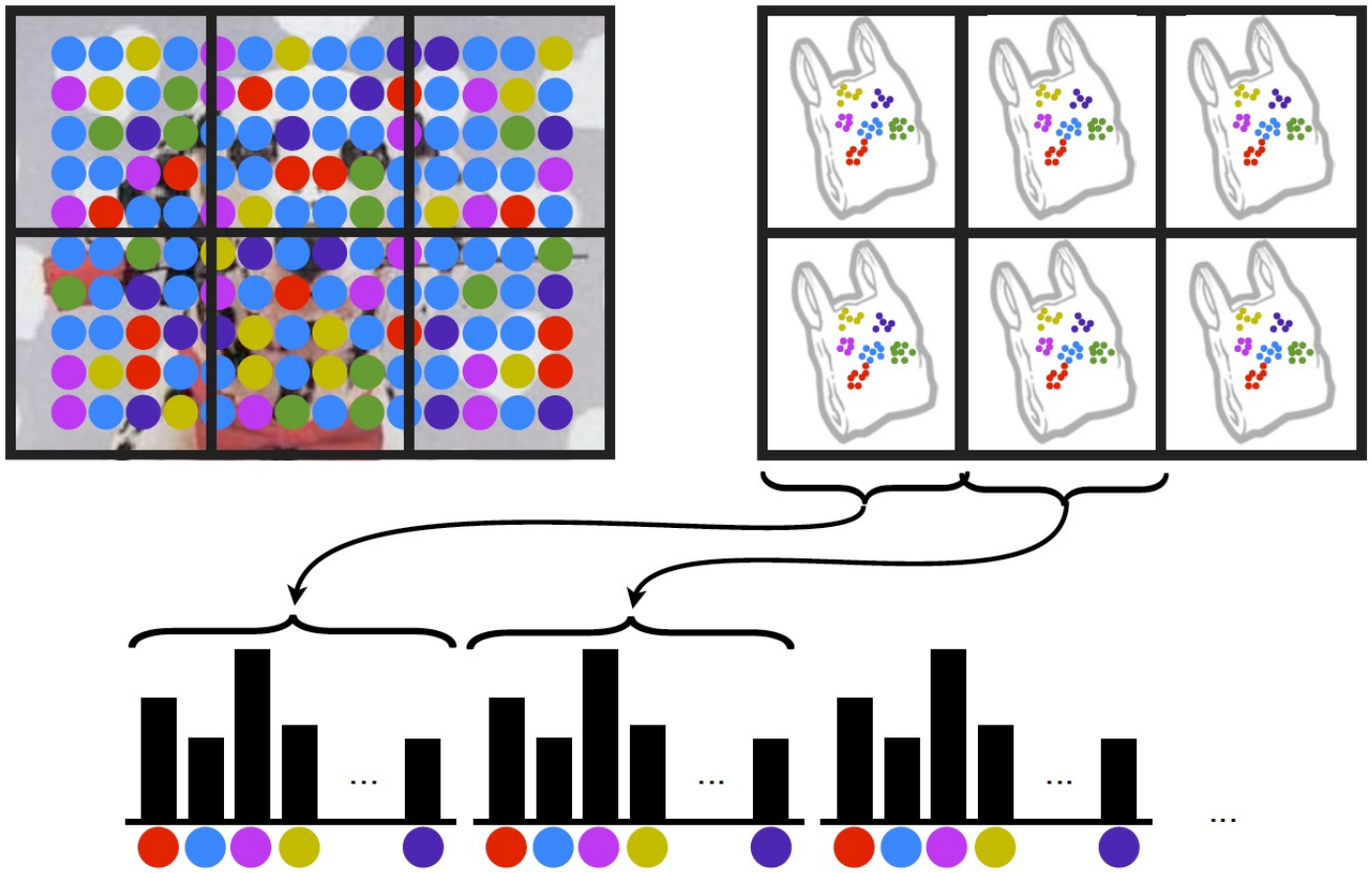


[Csurka et al. 2004]

histogram (bag) of visual words

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

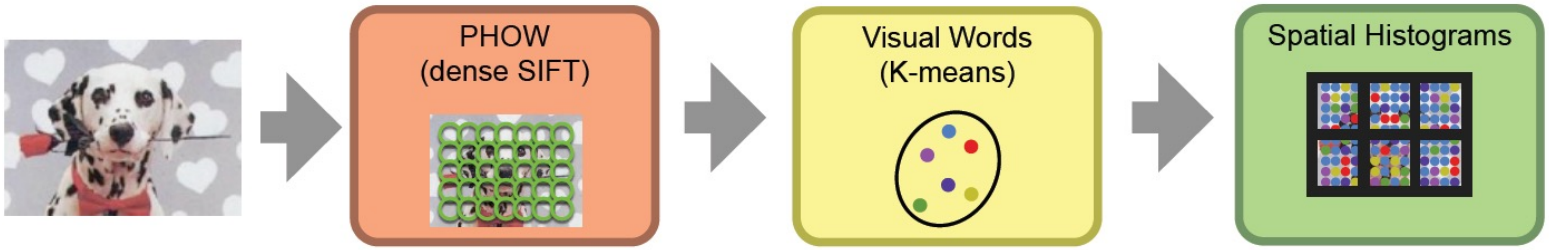
Spatial histograms

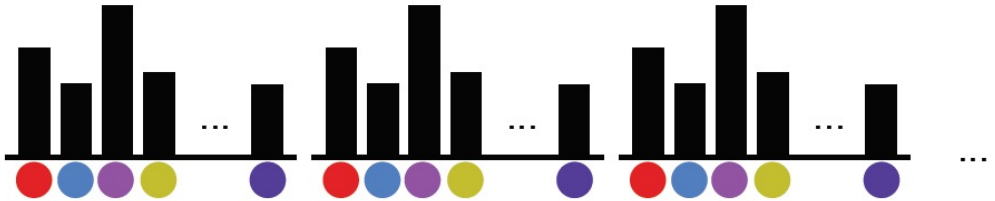


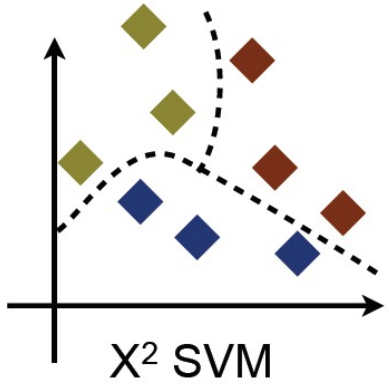
[Lazebnik et al. 2004]

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Summary: image descriptors

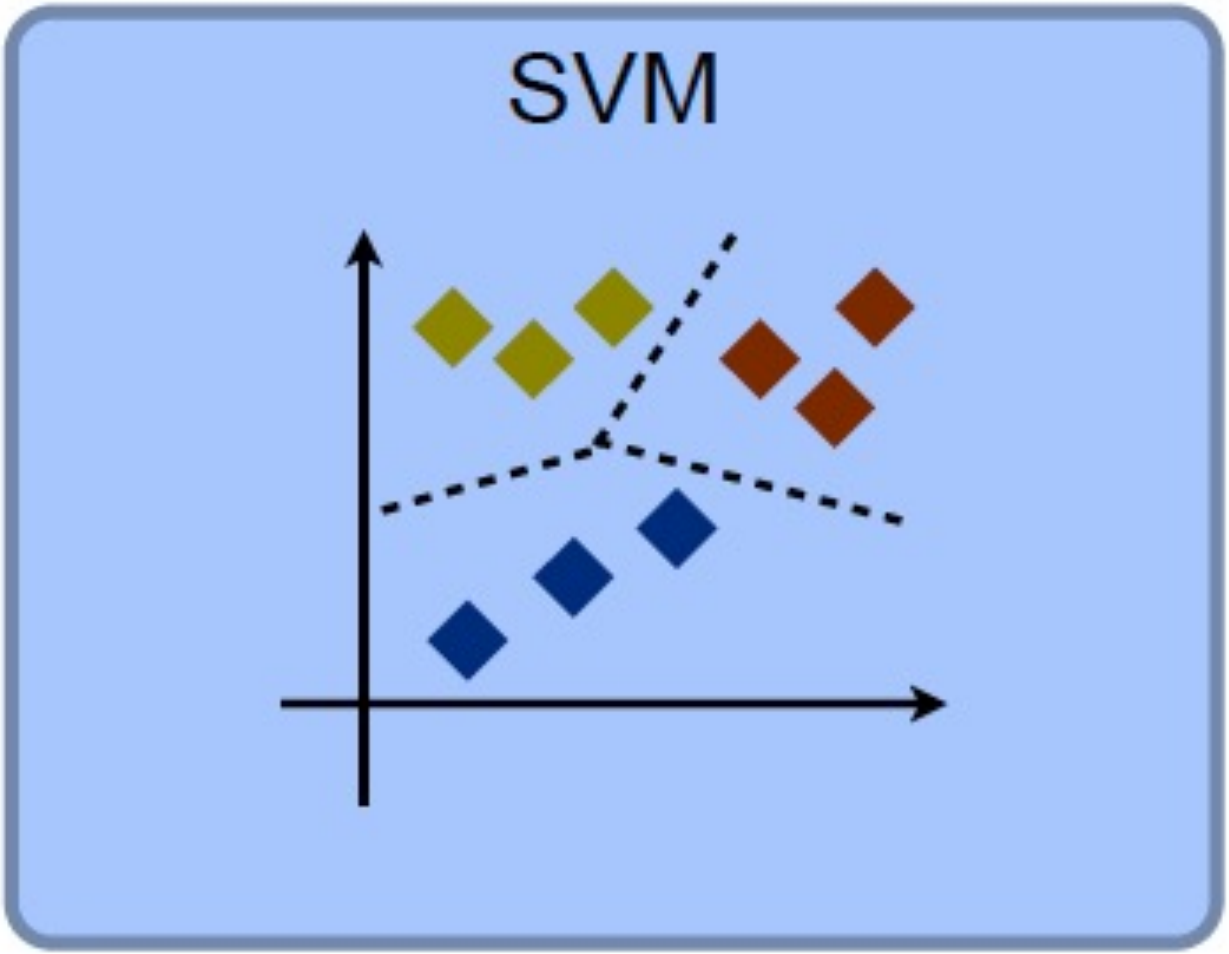






Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

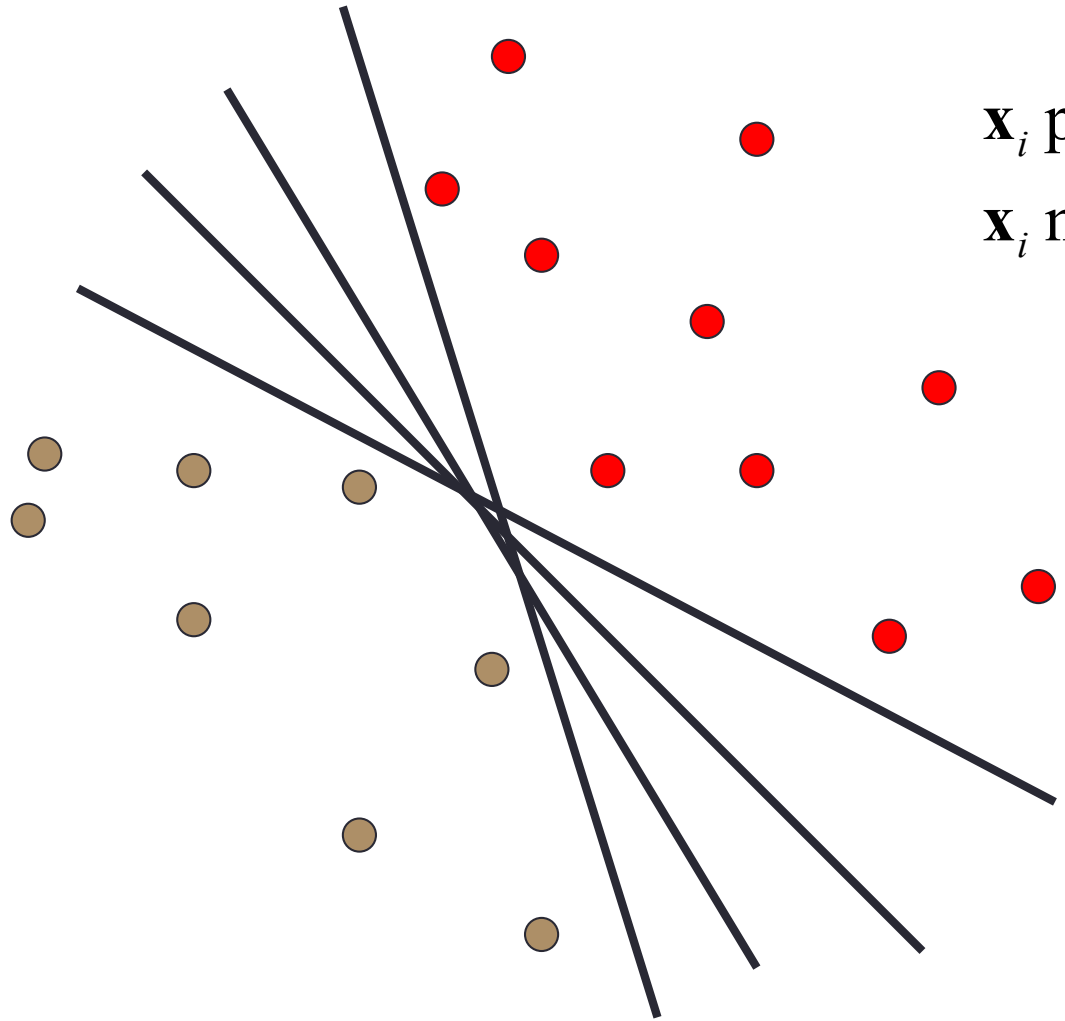
Classification



Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Linear classifiers

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



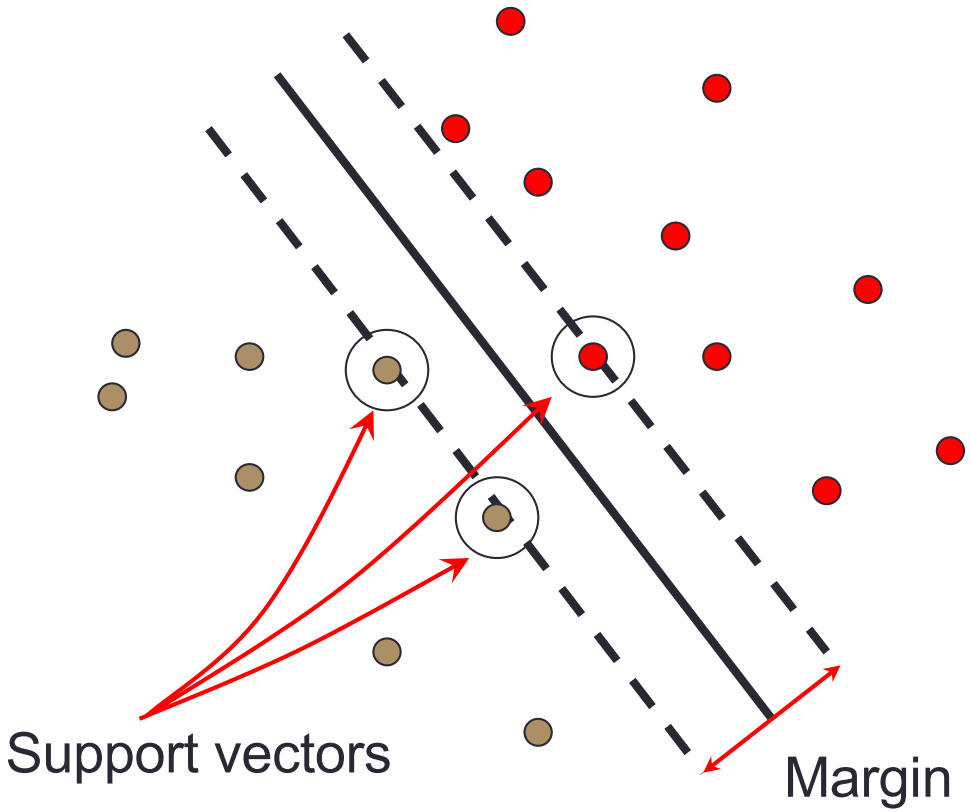
$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

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

Which hyperplane is best?

Support vector machines

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



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

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

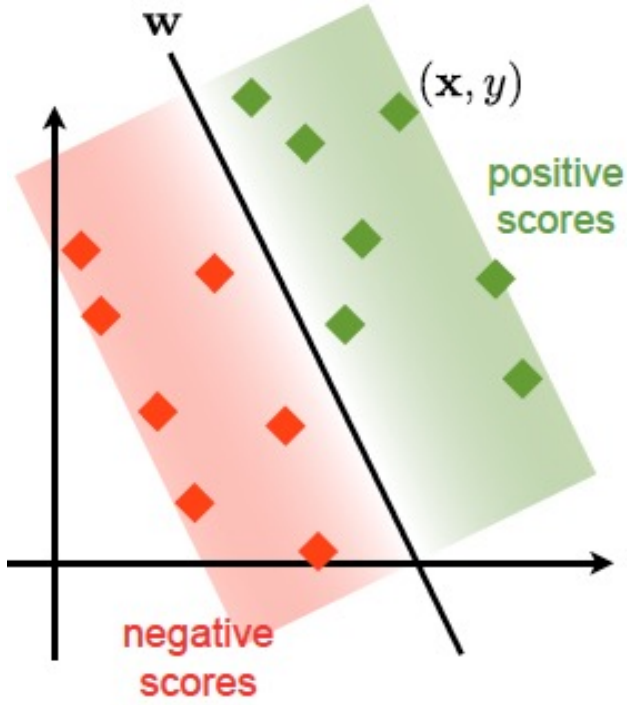
For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

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

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

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

Linear SVM



Discriminant score

$$f(\mathbf{x}; \mathbf{w}) = \langle \mathbf{w}, \mathbf{x} \rangle$$

$$\min_{\mathbf{w}} \frac{\lambda}{2} \|\mathbf{w}\|^2 + \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{w}; (\mathbf{x}_i, y_i))$$

where N is the number of points.

Linear SVM

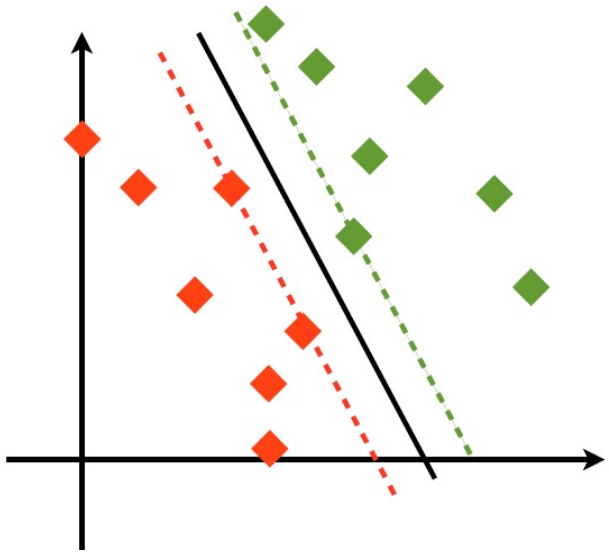
$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{w}; (\mathbf{x}_i, y_i))$$

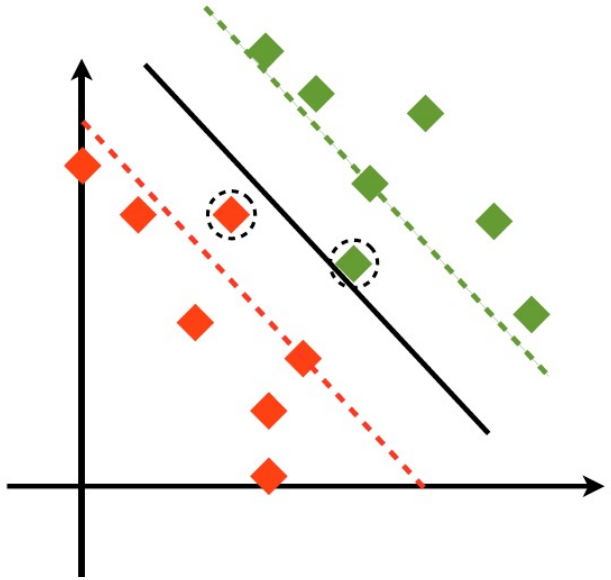
hard loss

$$\ell(\mathbf{w}; (\mathbf{x}, y)) = \begin{cases} 0, & y\langle \mathbf{w}, \mathbf{x}\rangle > 1, \\ +\infty, & \text{otherwise.} \end{cases}$$

hinge loss

$$\ell(\mathbf{w}; (\mathbf{x}, y)) = \begin{cases} 0, & y\langle \mathbf{w}, \mathbf{x}\rangle > 1, \\ 1 - y\langle \mathbf{w}, \mathbf{x}\rangle, & \text{otherwise.} \end{cases}$$





Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Finding the maximum margin hyperplane

1. Maximize margin 2/||**w**||
2. Correctly classify all training data:

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

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

- *Quadratic optimization problem*:

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

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

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

Finding the maximum margin hyperplane

- Solution:

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

learned weight

Support vector

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

Finding the maximum margin hyperplane

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

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

- Classification function (decision boundary):

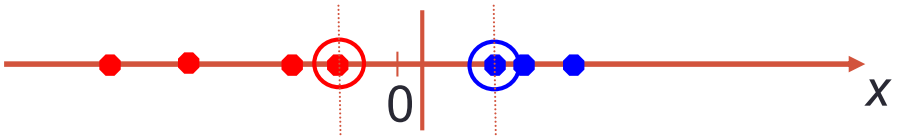
$$\mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

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

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

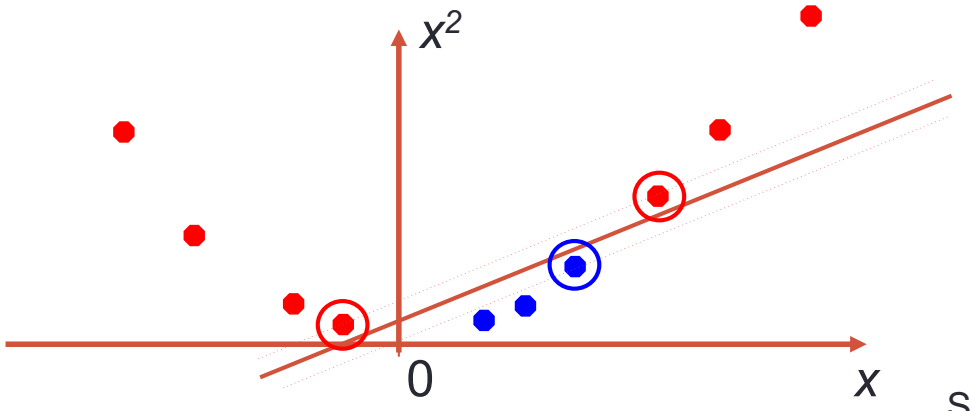
Nonlinear SVMs

- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?

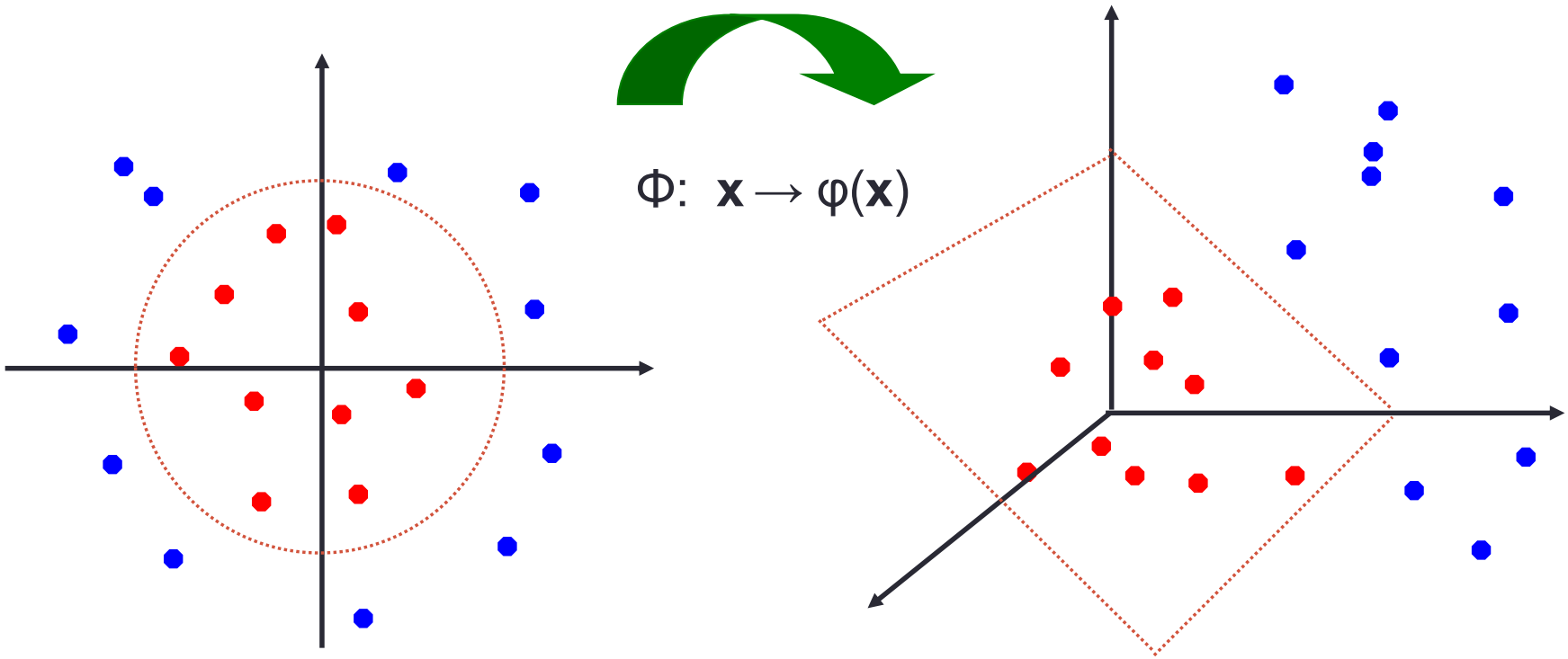
- We can map it to a higher-dimensional space:



Slide credit: Andrew Moore

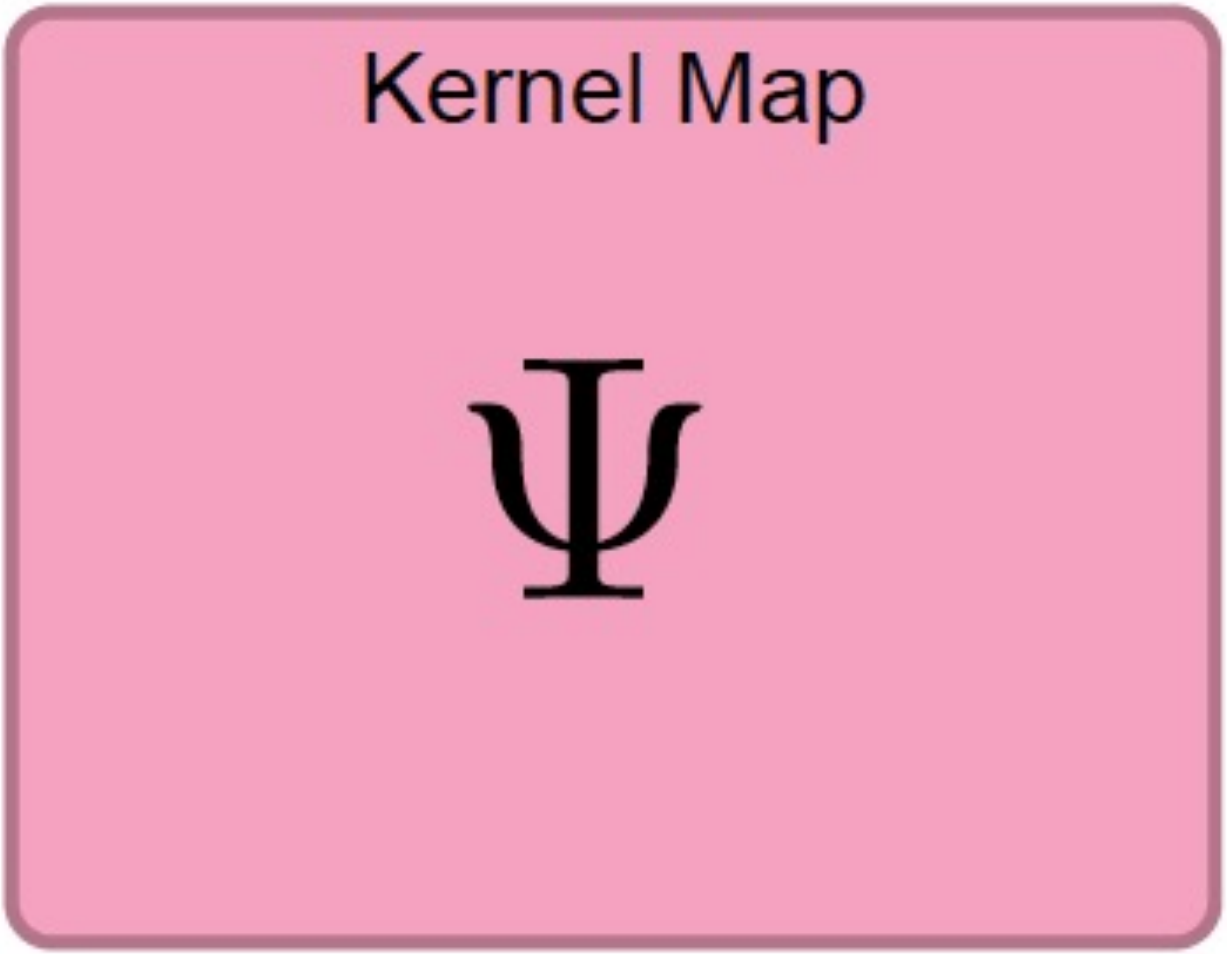
Nonlinear SVMs

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



Slide credit: Andrew Moore

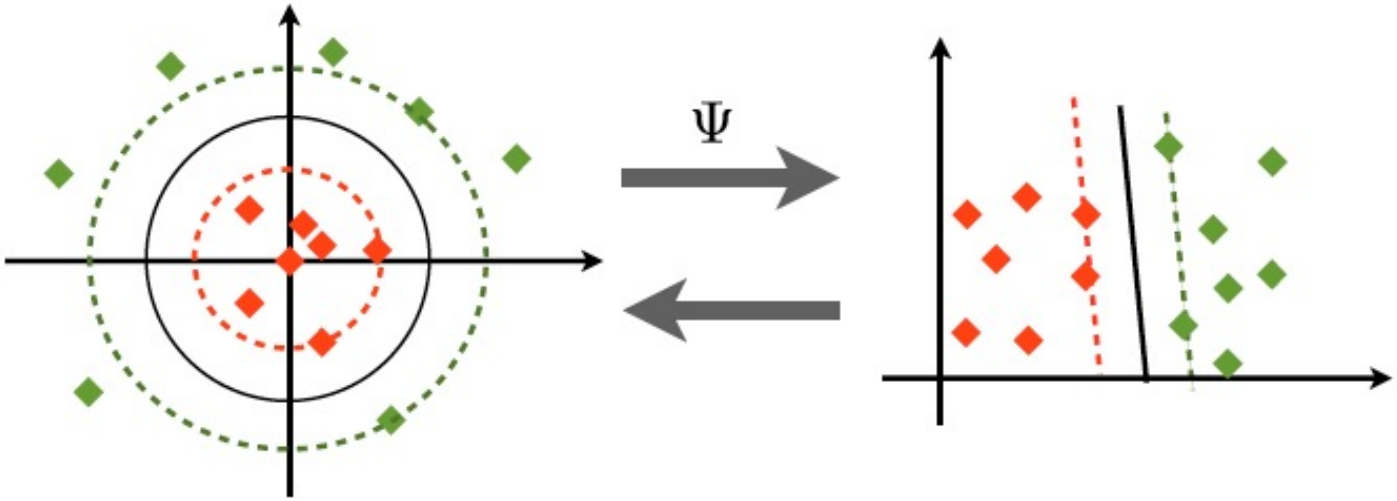
Kernel Map



Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Non-linear SVM

- Generalize SVM through a *feature map*



- The feature map may be defined implicitly by a **kernel function**

$$\mathbf{w} = \sum_{i=1}^{N} \beta_i \Psi(\mathbf{x}_i) \qquad \langle \mathbf{w}, \Psi(\mathbf{x}) \rangle = \sum_{i=1}^{N} \beta_i \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}) \rangle$$

$$K(\mathbf{x}_i, \mathbf{x}) = \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}) \rangle$$

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Nonlinear SVMs

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

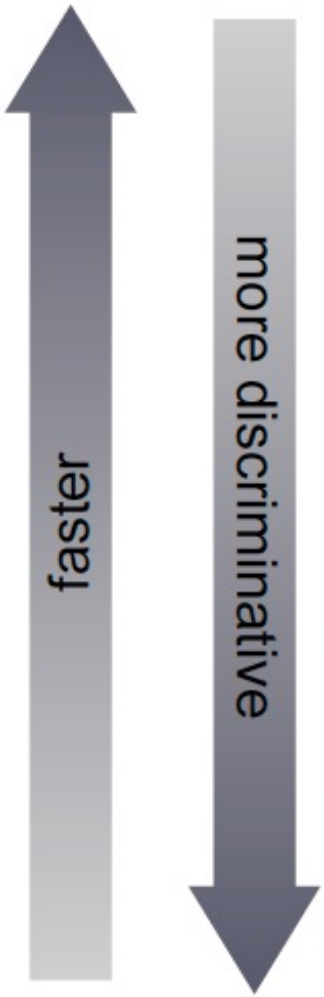
$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\Psi}(\mathbf{x}_i) \cdot \boldsymbol{\Psi}(\mathbf{x}_j)$$

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

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

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

Common kernels



linear

$$K(\mathbf{x}, \mathbf{y}) = \sum_{l=1}^{D} \mathbf{x}_l \mathbf{y}_l$$

[Zhang et al. 2007]
[Vedaldi et. al 2010]

Fast kernel and distance matrices with `vl_alldist`

additive

$$K(\mathbf{x}, \mathbf{y}) = \sum_{l=1}^{D} k(\mathbf{x}_l, \mathbf{y}_l)$$

additive RBF

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2} \sum_{l=1}^{D} d^2(\mathbf{x}_l, \mathbf{y}_l)\right)$$

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Speeding-up non-linear SVMs

- Exact feature map is usually: $\Psi(\mathbf{x})$ $K(\mathbf{x}, \mathbf{y}) = \langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle$

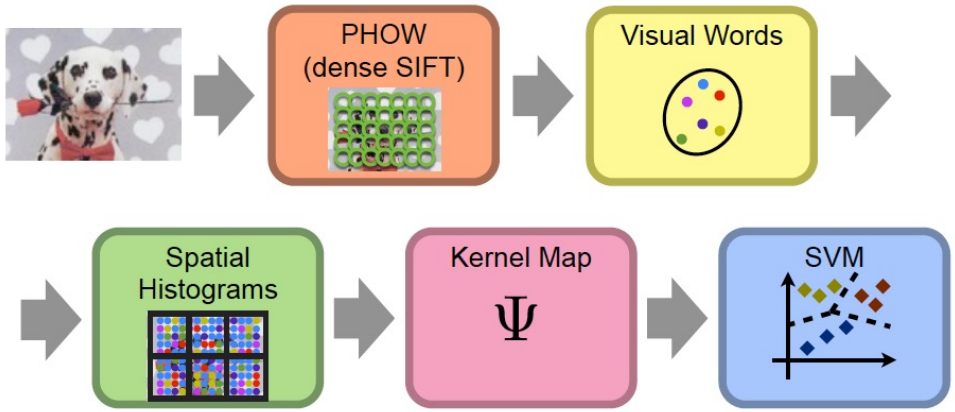
hard to compute **high dimensional**

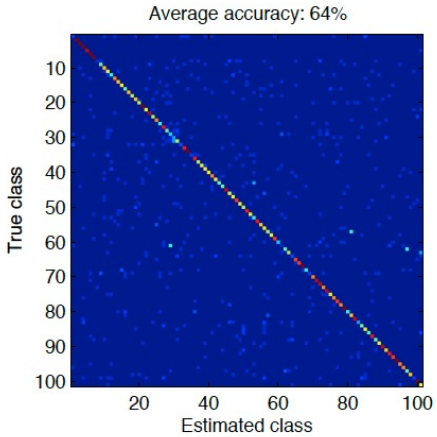
- • Seek for approximations $\widehat{\Psi}(\mathbf{x})$ $K(\mathbf{x}, \mathbf{y}) \approx \langle \widehat{\Psi}(\mathbf{x}), \widehat{\Psi}(\mathbf{y}) \rangle$
- - low dimensional
- - efficient computation
- - good approximation quality

Algorithm: SVMs for image classification

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

Implementation of the object recognition by Bag-of-Words in VL-Feat





- **PHOW features**
  - Fast dense SIFT
- **Visual Words**
  - Elkan k-means
- **Spatial Histograms**
  - Convenience functions
- **Linear SVM**
  - PEGASOS
- **X^2 SVM**
  - Homogeneous kernel map

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

Unranked retrieval evaluation: Precision and Recall

- **Precision**: fraction of retrieved objects that are relevant = P(relevant|retrieved)

- **Recall**: fraction of relevant objects that are retrieved = P(retrieved|relevant)

	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn

- Precision P = tp/(tp + fp)
- Recall R = tp/(tp + fn)

Should we instead use the accuracy measure for evaluation?

- Given a query, an engine classifies each object as "Relevant" or "Nonrelevant"
- The **accuracy** of an engine: the fraction of these classifications that are correct
  - (tp + tn) / (tp + fp + fn + tn)
- **Accuracy** is a commonly used evaluation measure in machine learning classification work
- Why is this not always a very useful evaluation measure?

Precision/Recall

- You can get high recall (but low precision) by retrieving all objects for all queries!
- Recall is a non-decreasing function of the number of objects retrieved
- In a good system, precision decreases as either the number of objects retrieved or recall increases
  - This is not a theorem, but a result with strong empirical confirmation

Difficulties in using precision/recall

- Should average over large objects collection/query ensembles
- Need human relevance assessments
  - People aren't reliable assessors
- Assessments have to be binary
  - Nuanced assessments?
- Heavily skewed by collection
  - Results may not translate from one domain to another

A combined measure: F

- Combined measure that assesses precision/recall tradeoff is **F measure** (weighted harmonic mean):

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

- People usually use balanced F_1 measure
  - i.e., with $\beta = 1$ or $\alpha = ½$

Results of the Bags of words for object recognition






class	bag of features	bag of features	Parts-and-shape model
	Zhang et al. (2005)	Willamowski et al. (2004)	Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Slides mostly borrowed from Li Fei-Fei and A. Vedaldi

What about multi-class SVMs?

- Unfortunately, there is 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
  - Training: 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

Advantages and disadvantages of SVM

- The **advantages** of support vector machines are:
  - Effective in high dimensional spaces.
  - Still effective in cases where number of dimensions is greater than the number of samples.
  - Use a subset of training points in the decision function (called support vectors), so it is also memory efficient.
  - Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.
  - 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

http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation

Advantages and disadvantages of SVM

The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, the method is likely to give poor performances.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.
- 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

http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation

Advantages and disadvantages of SVM

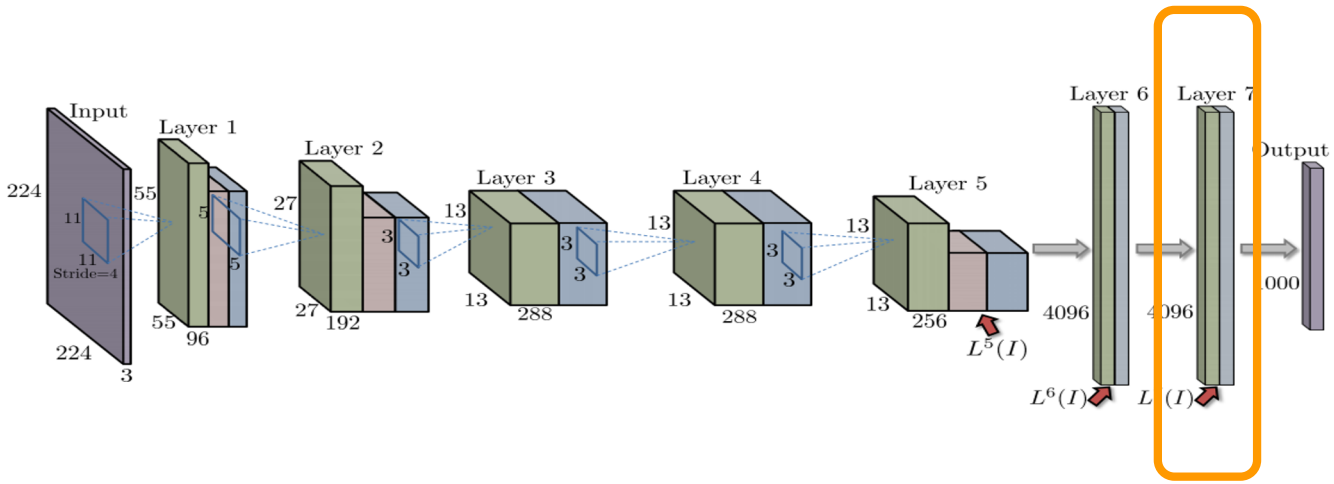
The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, the method is likely to give poor performance.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.
- 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

http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation

Laboratory

Compare BoW vs Neural networks
for object recognition
on a small dataset (Caltech101)



http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation