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

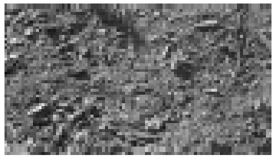
Image classification

- Given

  Positive training images containing an object class



  Negative training images that don’t



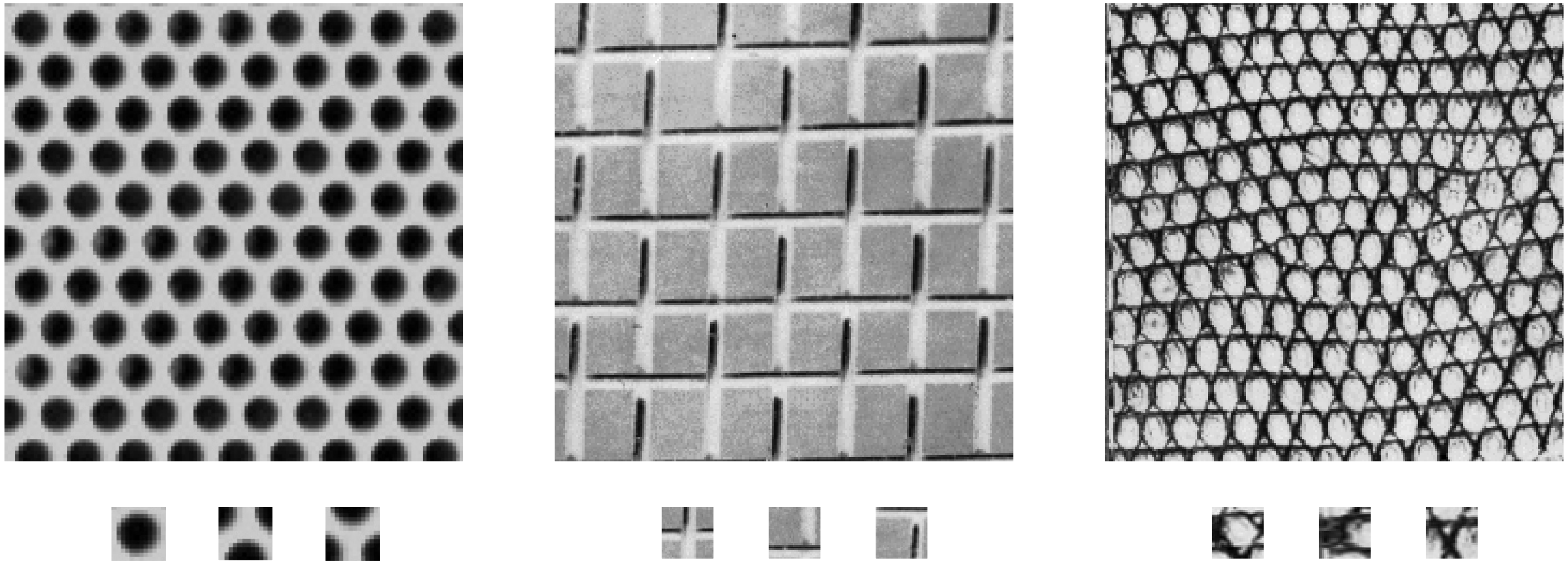
- Classify

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

?

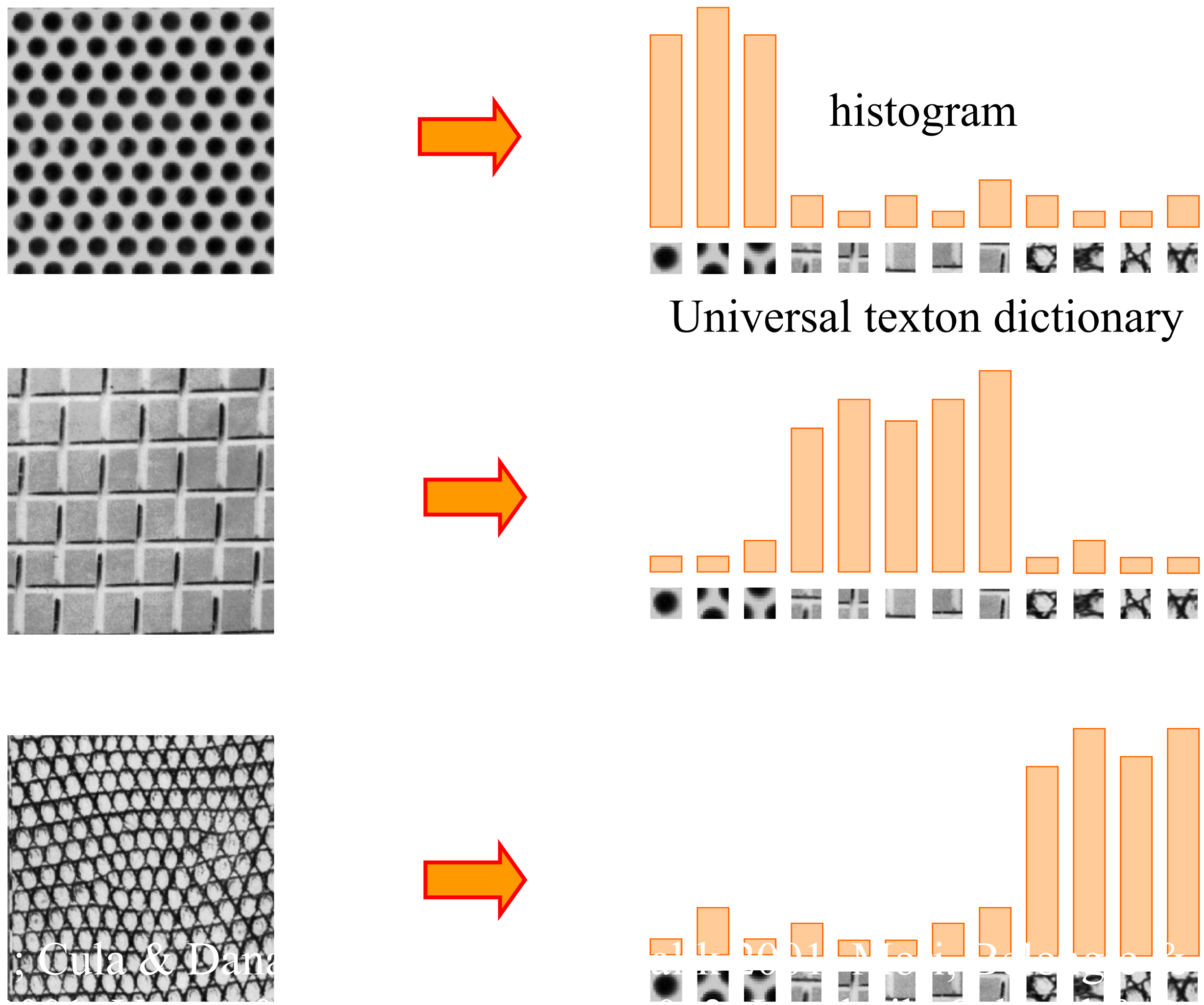
Bag-of-features for image classification

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



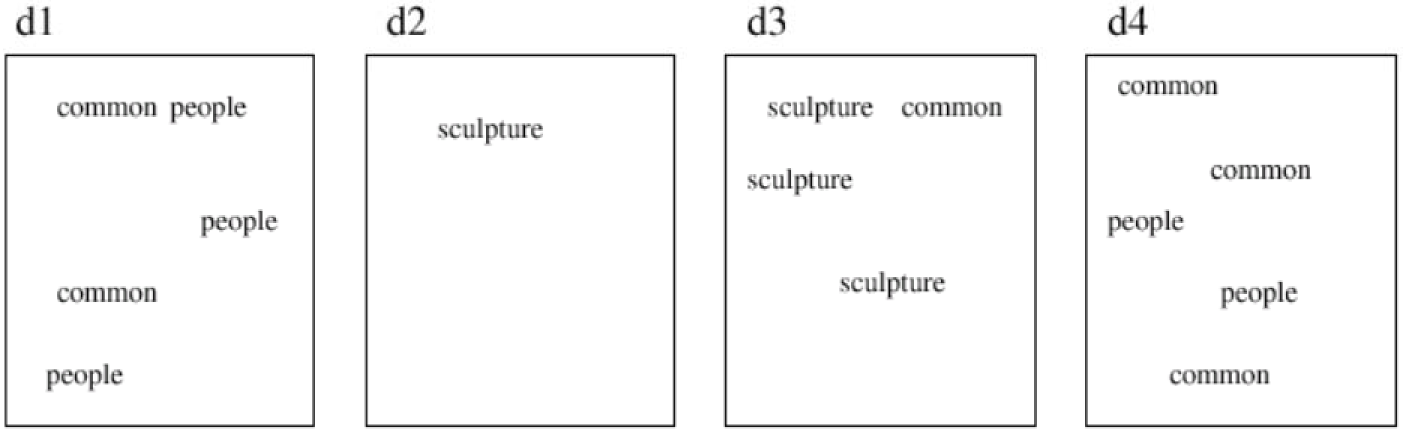
Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001
Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Texture recognition



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

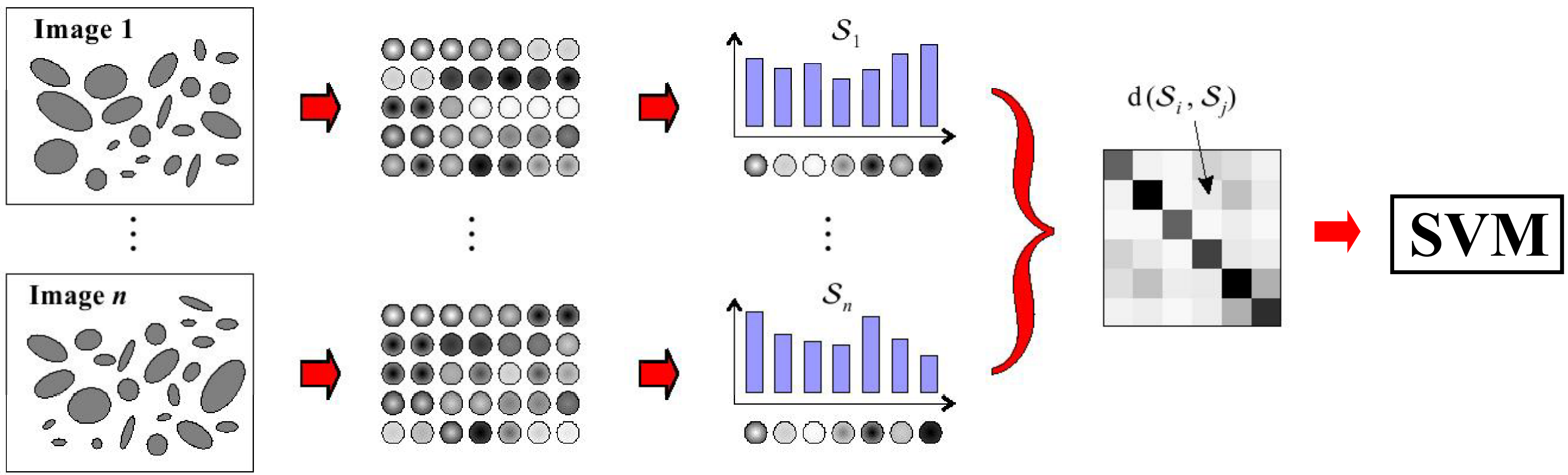
- Orderless document representation: frequencies of words from a dictionary
- Classification to determine document categories



Bag-of-words

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

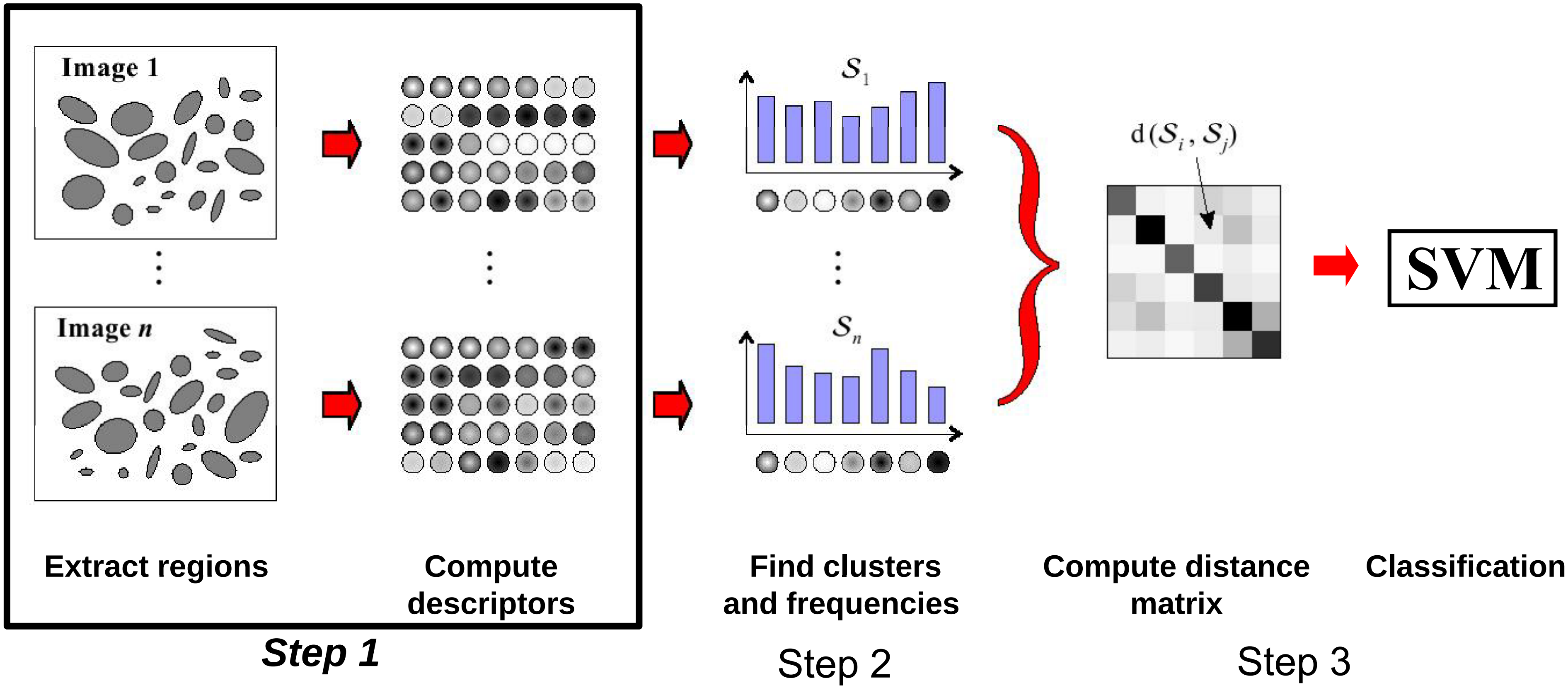
Bag-of-features for image classification



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

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

Bag-of-features for image classification

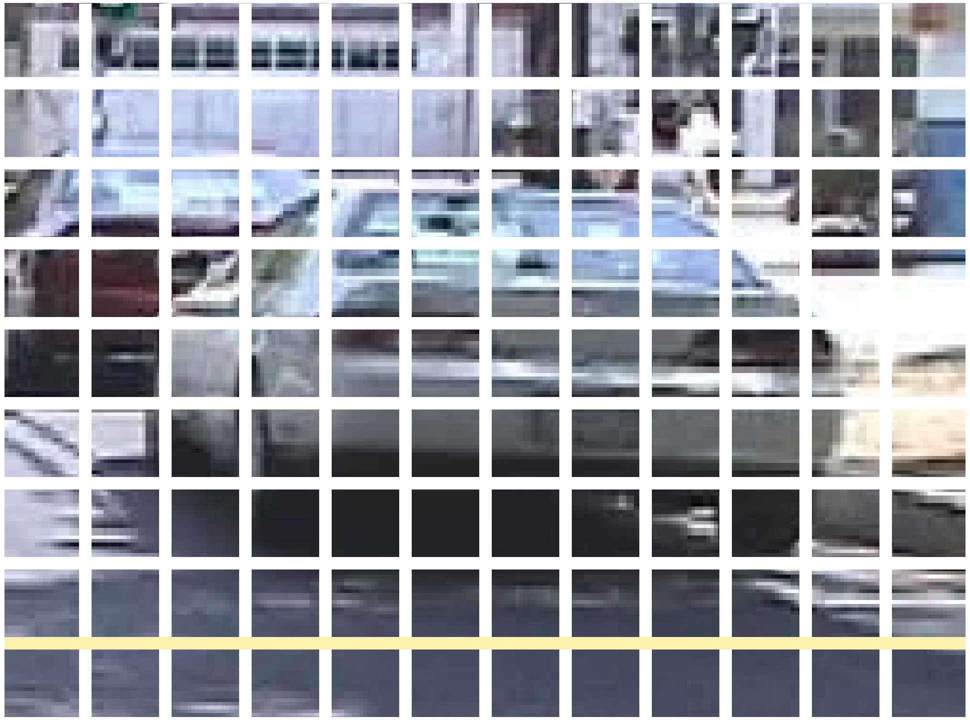


[Nowak,Jurie&Triggs,ECCV'06], [Zhang,Marszalek,Lazebnik&Schmid,IJCV'07]

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

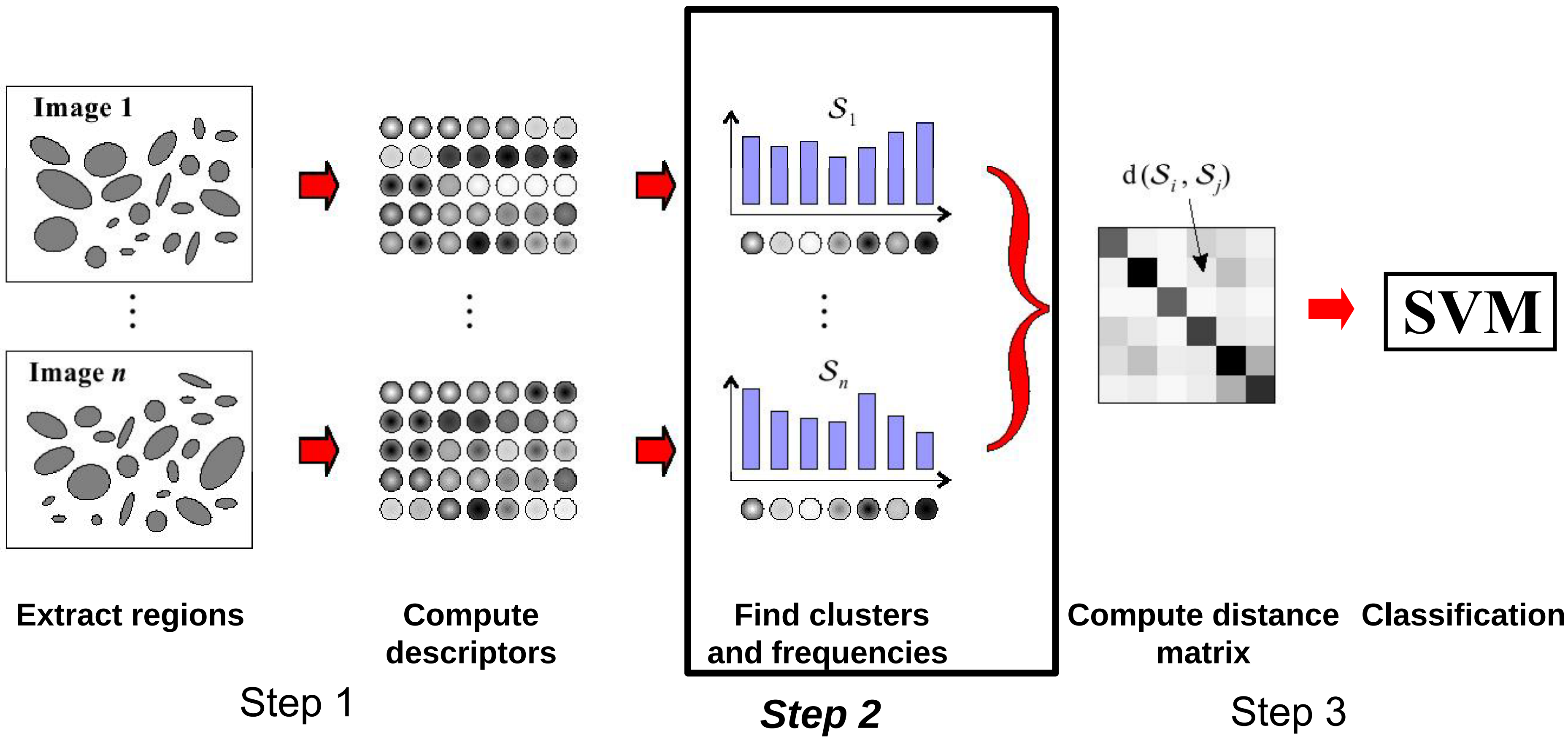


- Multi-scale dense grid: extraction of small overlapping patches at multiple scales

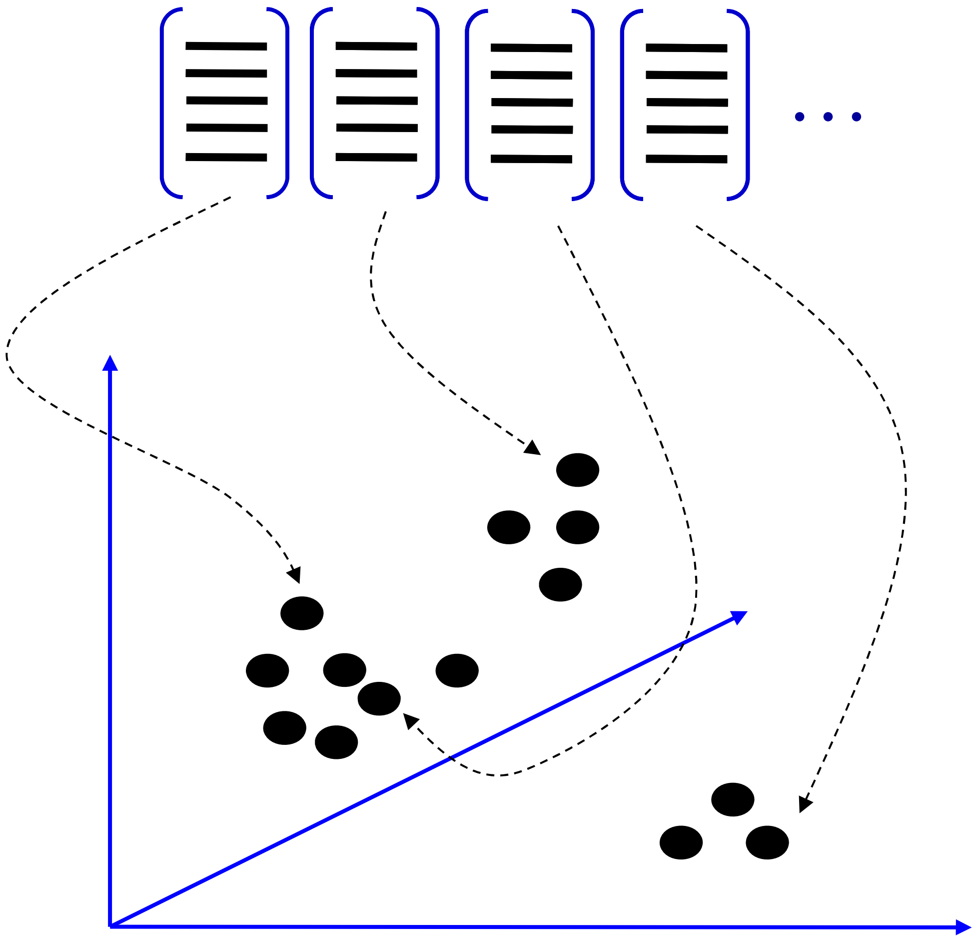
-Computation of the SIFT descriptor for each grid cells

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

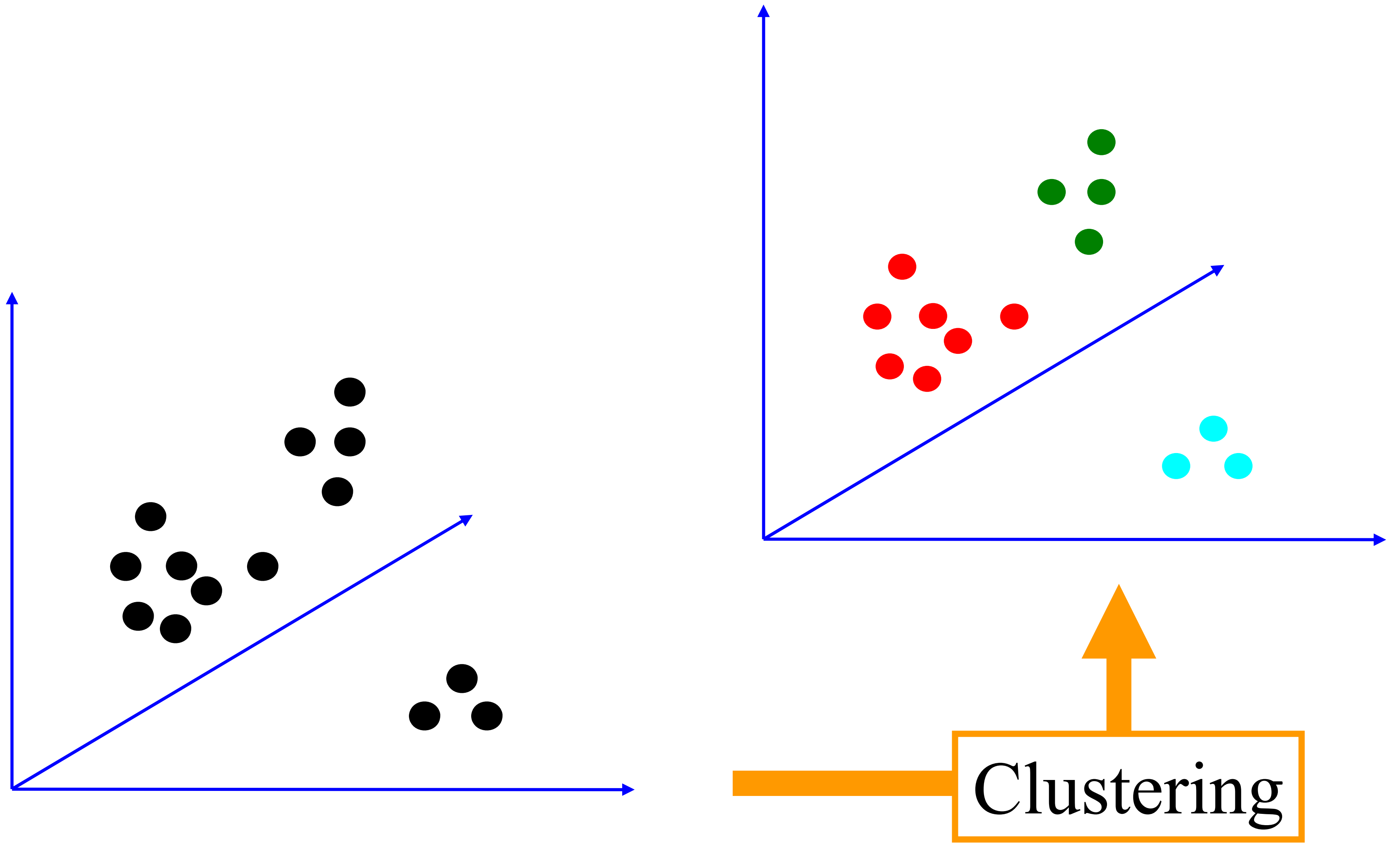
Bag-of-features for image classification



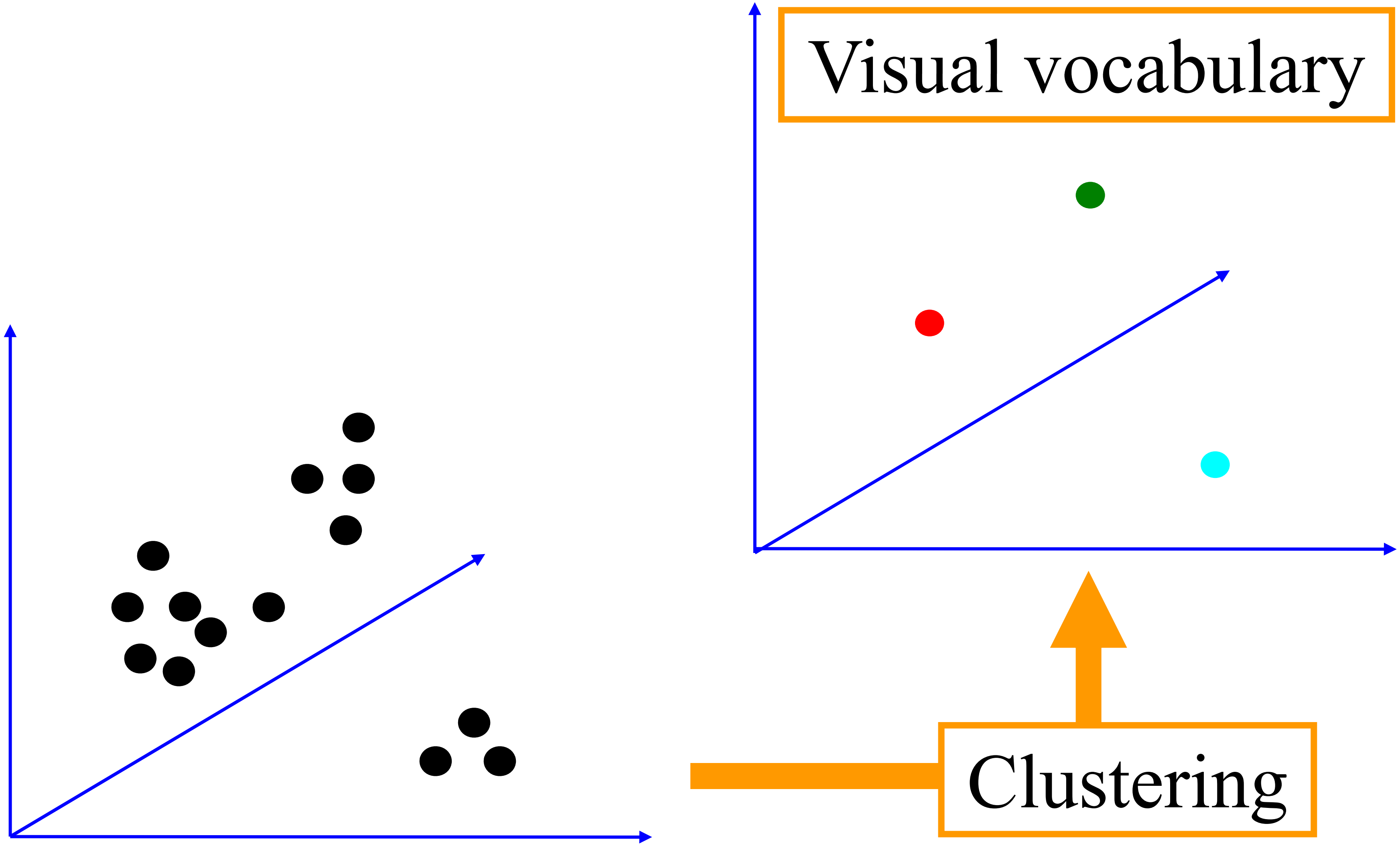
Step 2: Quantization



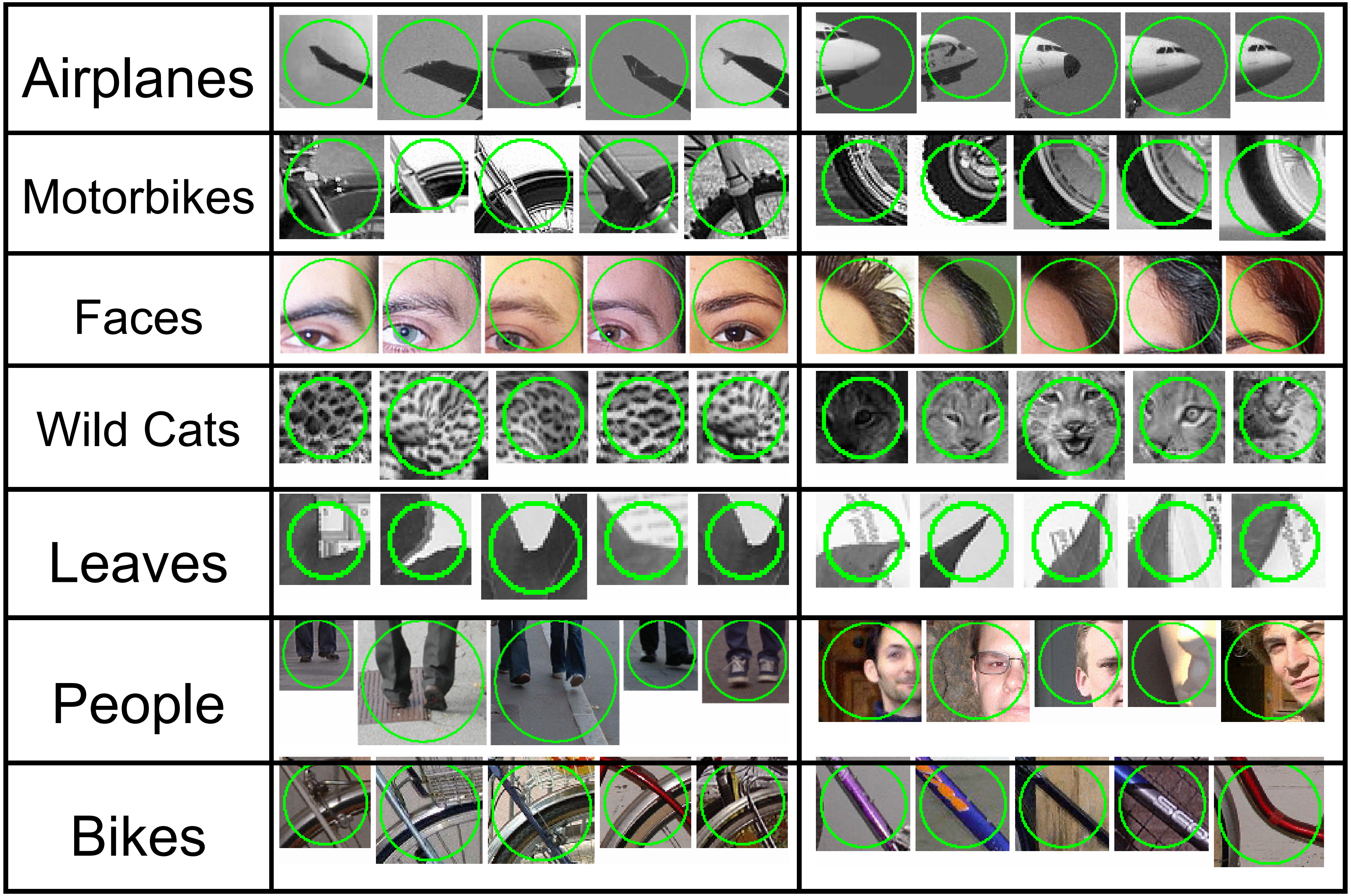
Step 2:Quantization



Step 2: Quantization



Examples for visual words



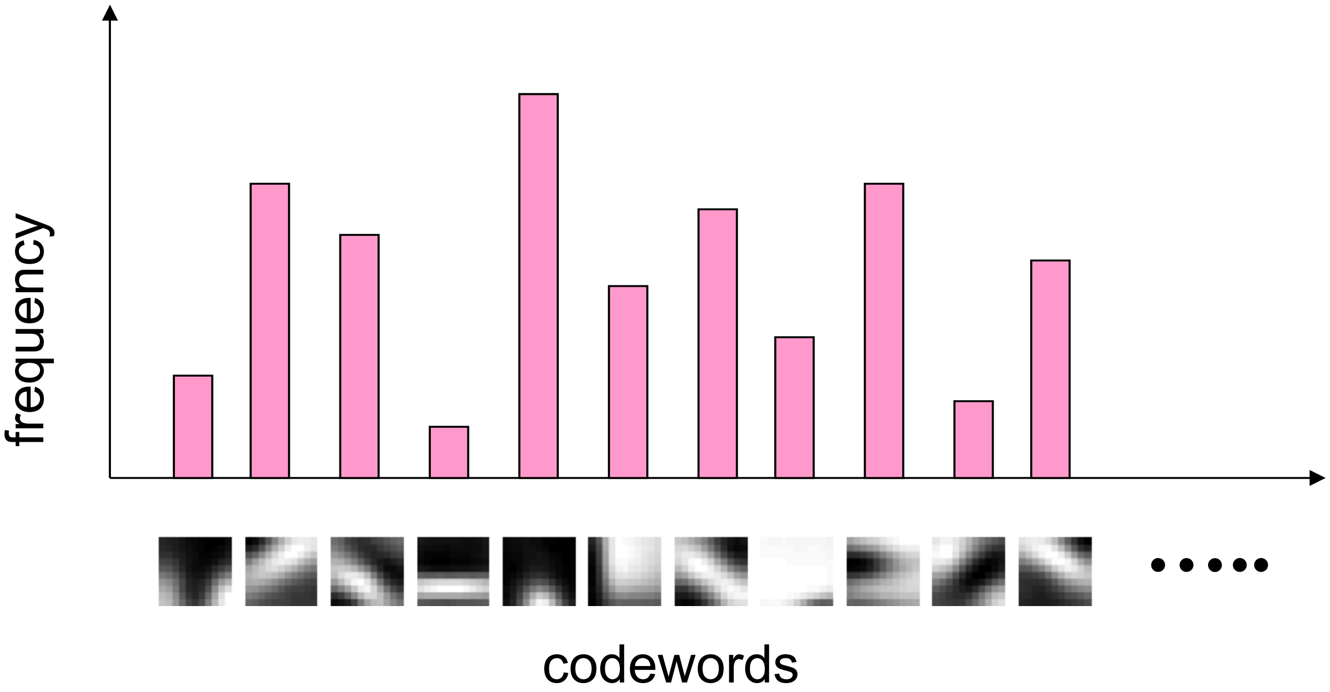
Step 2: Quantization

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

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

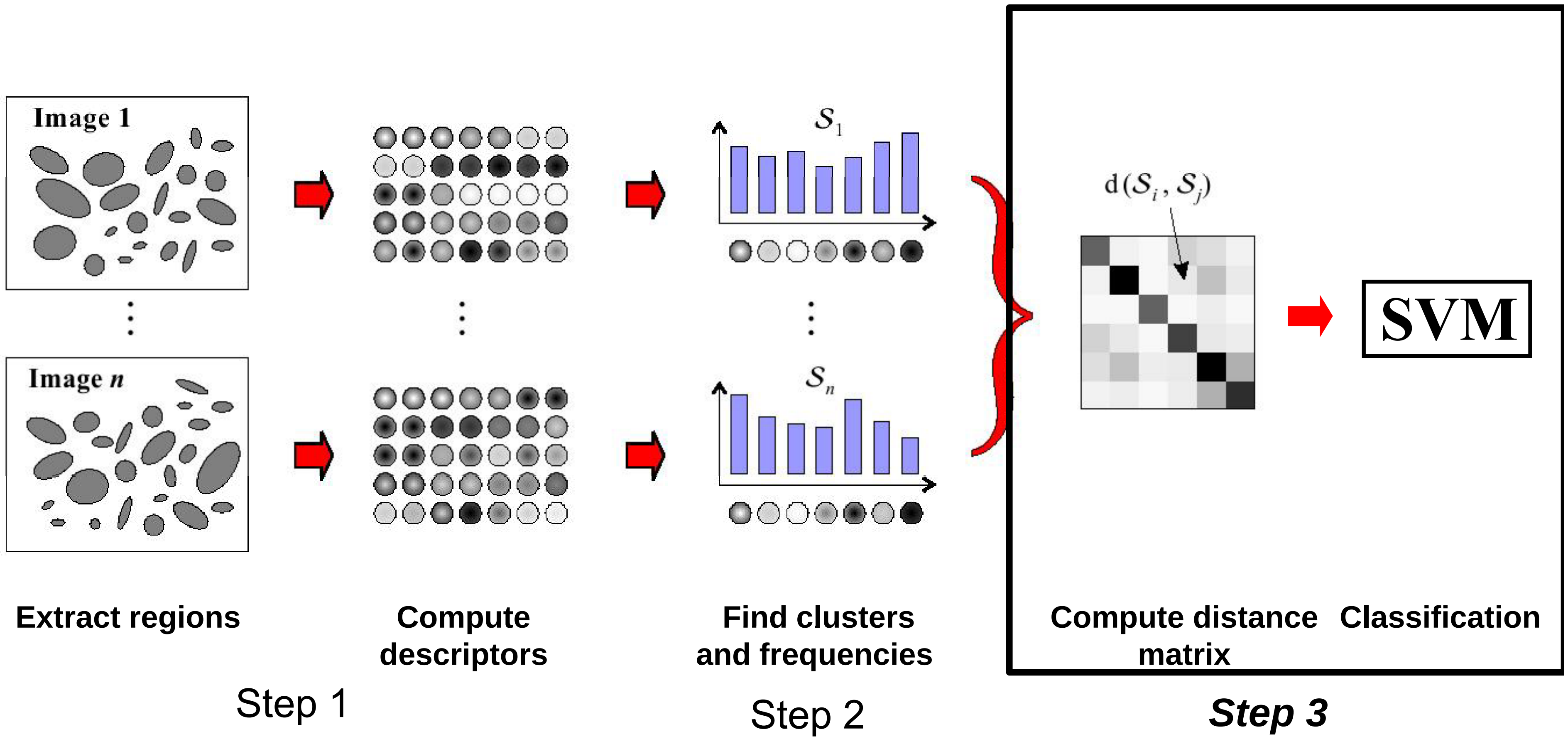


• Each image is represented by a vector, typically 1000-4000 dimension, normalization with L2 norm

• fine grained – represent model instances

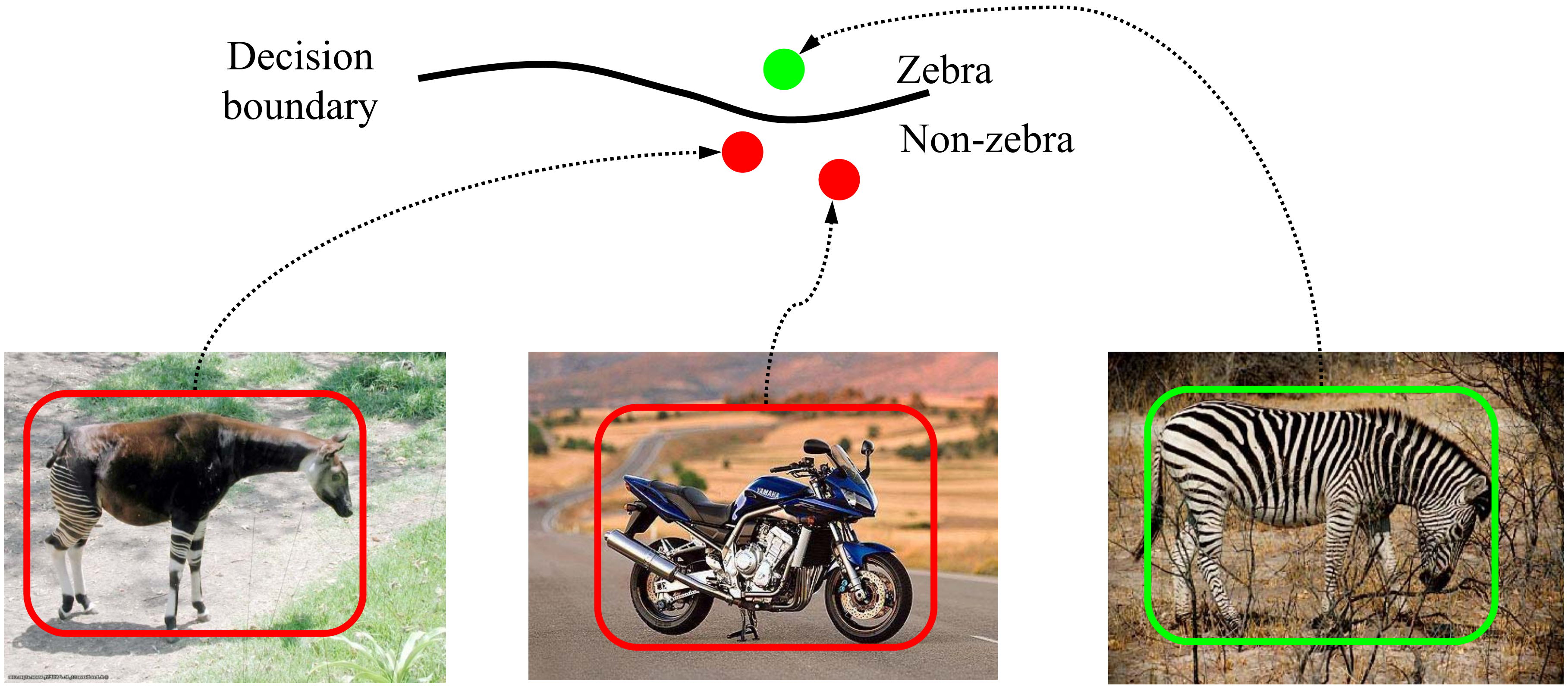
• coarse grained – represent object categories

Bag-of-features for image classification



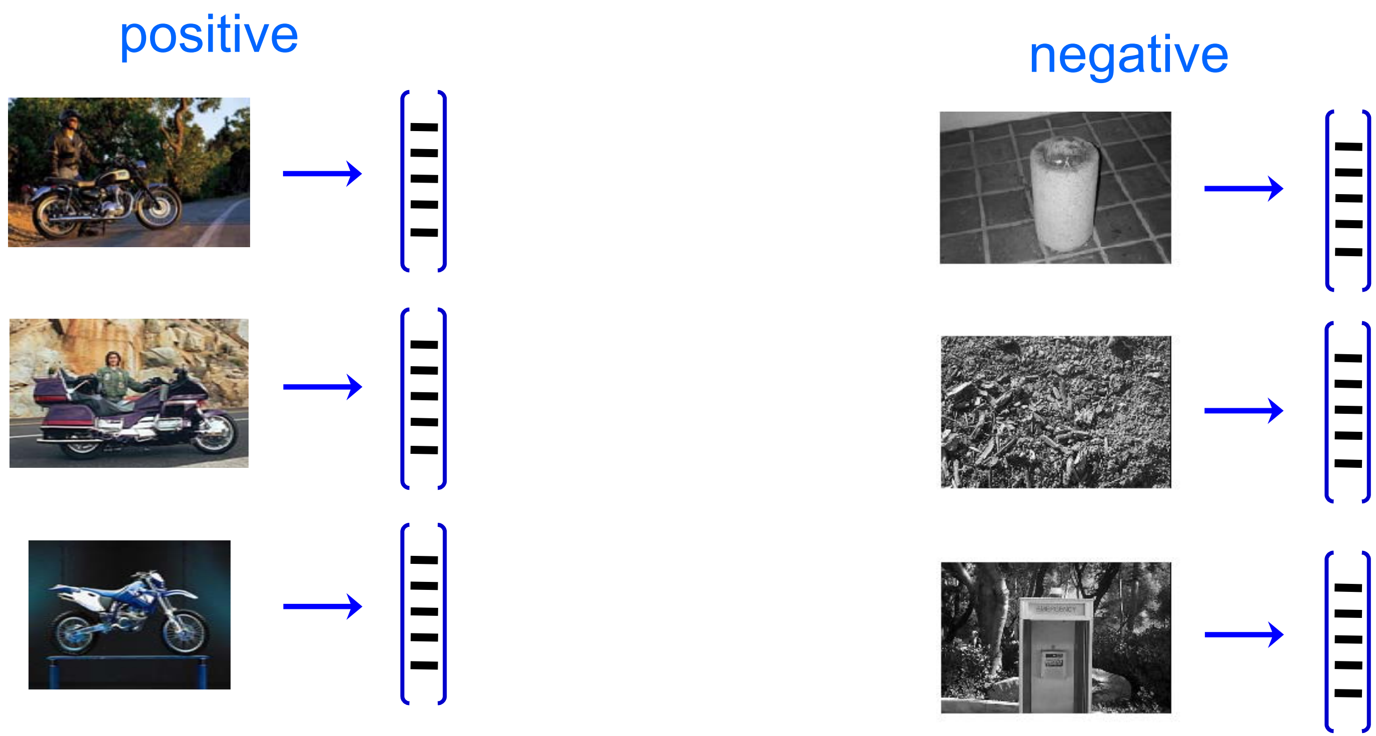
Step 3: Classification

- 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



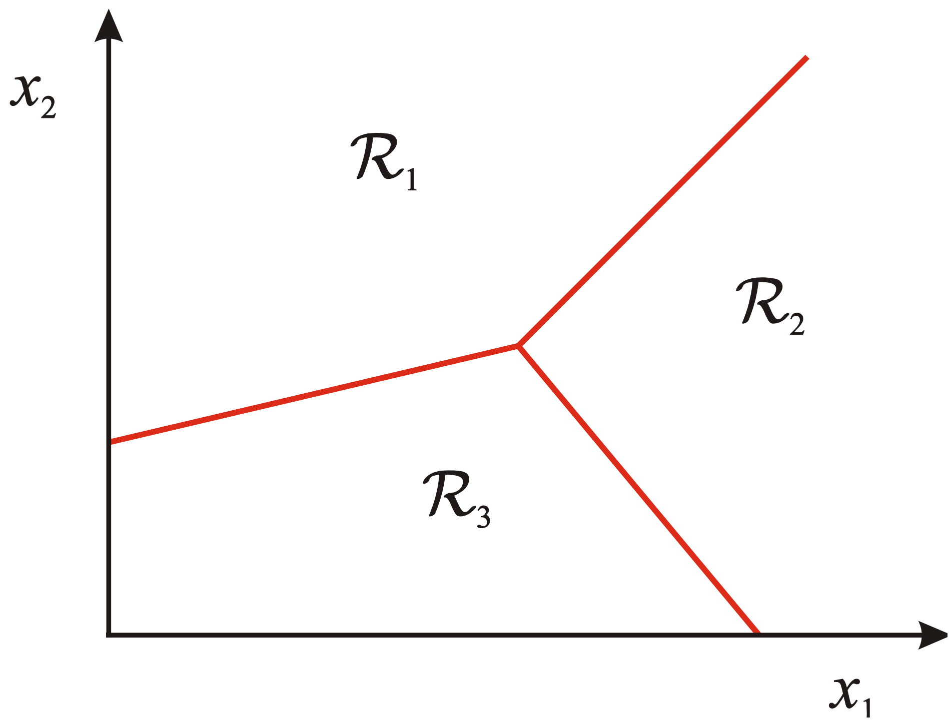
Train classifier,e.g.SVM

Classifiers

- K-nearest neighbor classifier
- Linear classifier
  - Support Vector Machine
- Non-linear classifier
  - Kernel trick
  - Explicit lifting

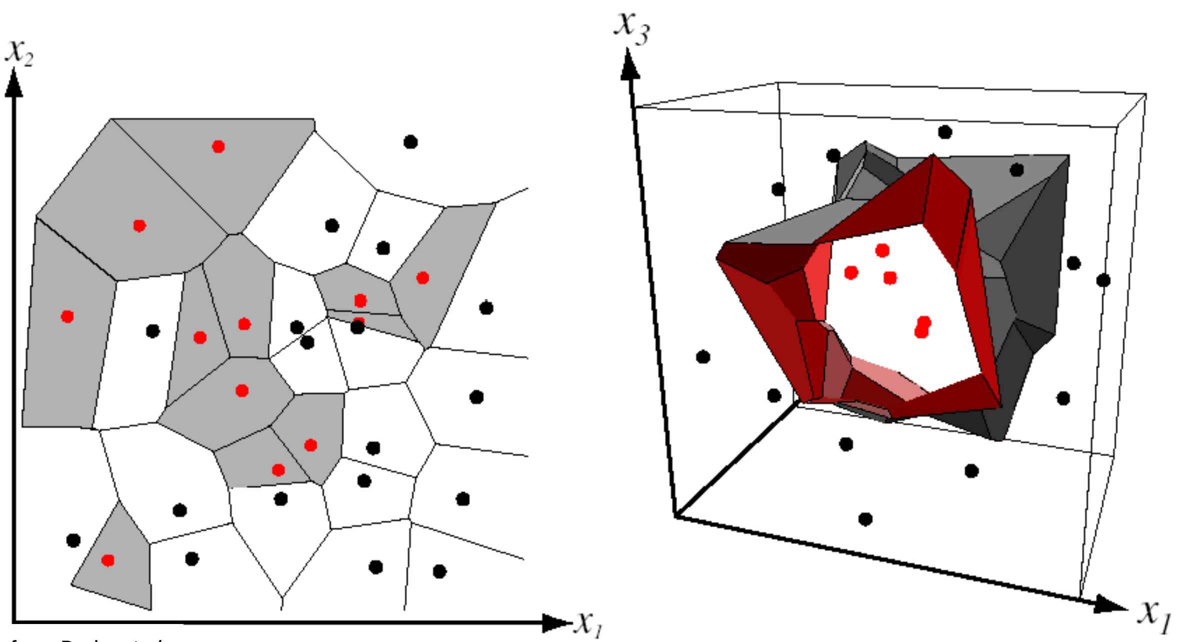
Classification

- 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

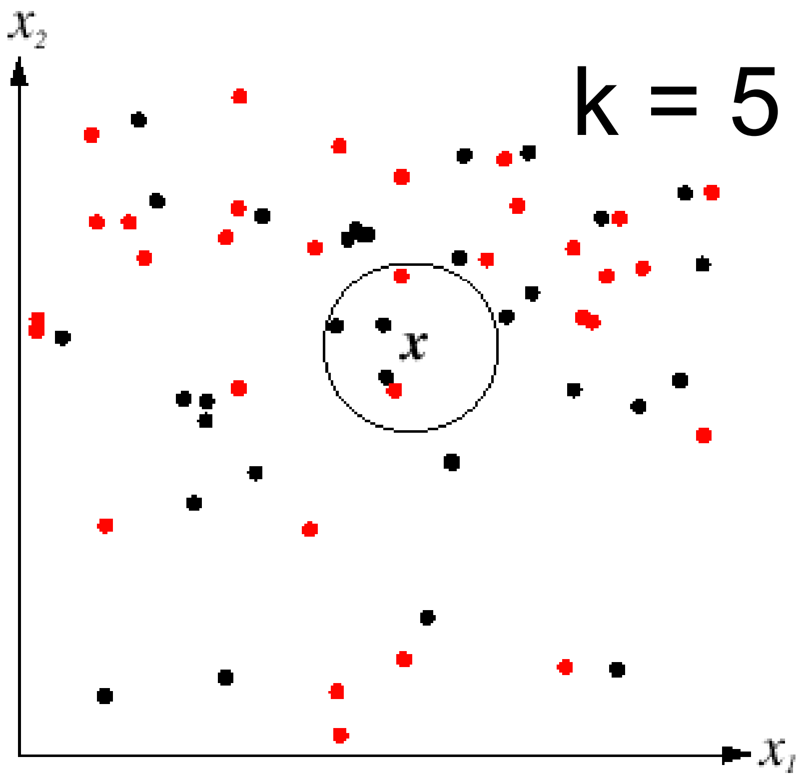


from Duda *et al.*

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

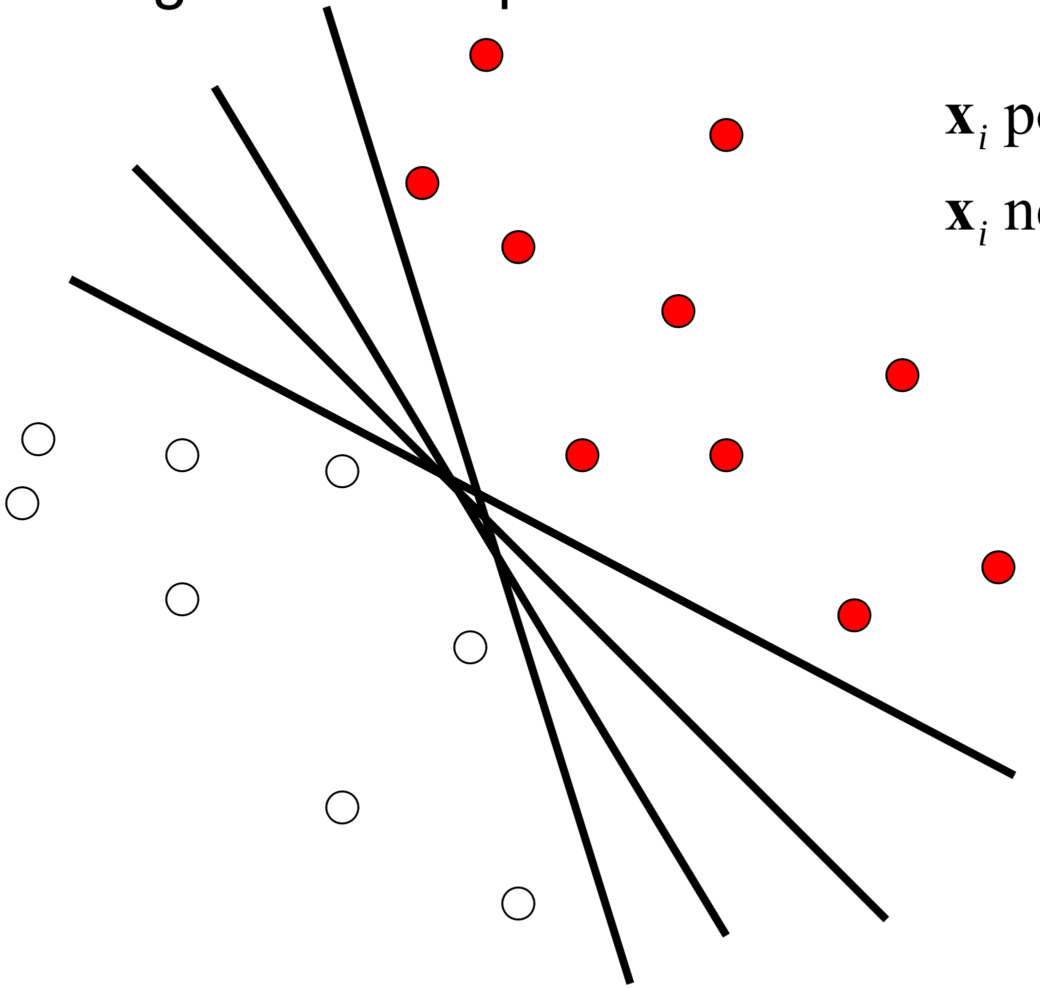
k-Nearest Neighbors

- 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



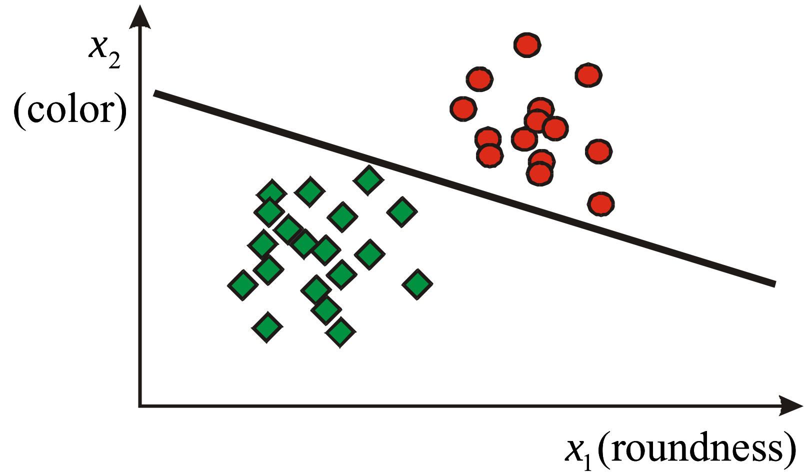
$\mathbf{x}_i$ positive: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$

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

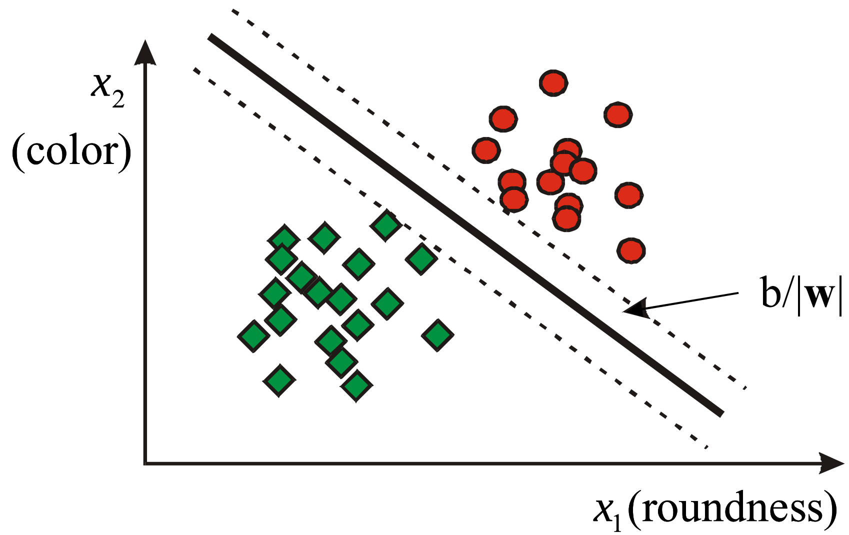
Which hyperplane is best?

Linear classifiers - margin

- Generalization is not good in this case:

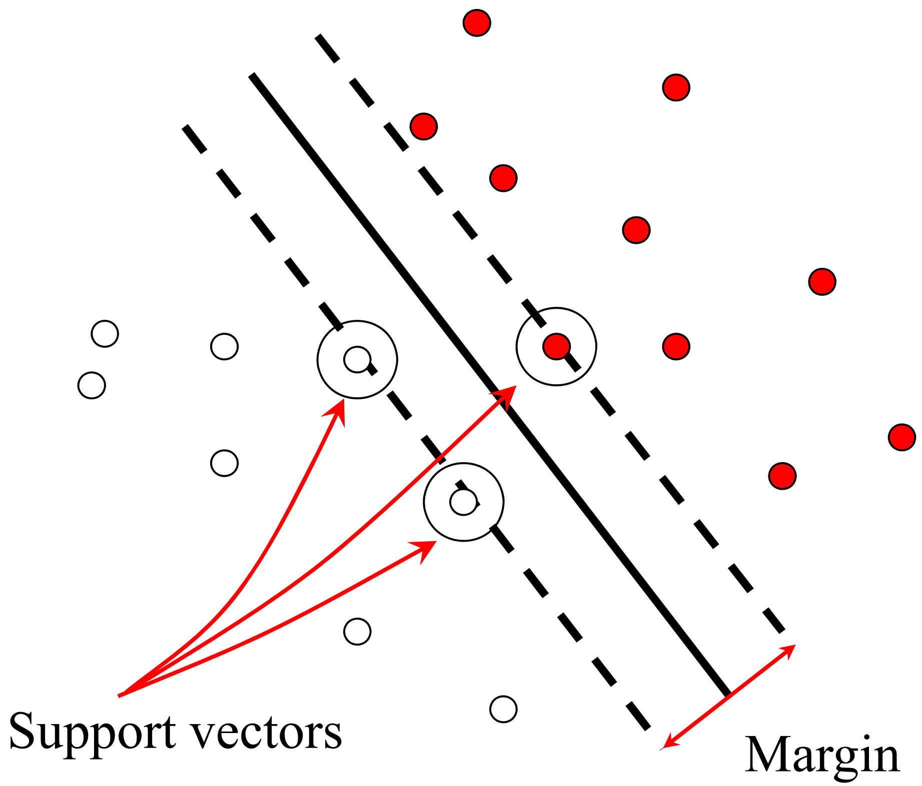


- Better if a margin is introduced:



Support vector machines

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



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

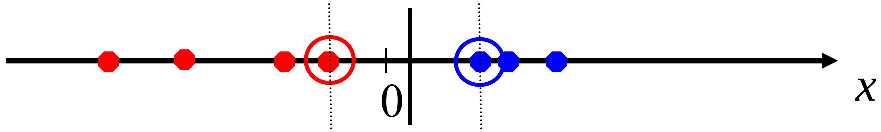
$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

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

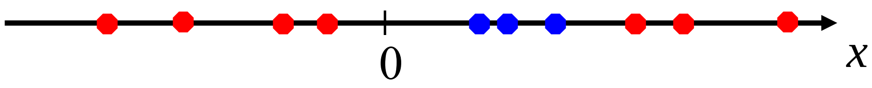
The margin is $2 / ||\mathbf{w}||$

Nonlinear SVMs

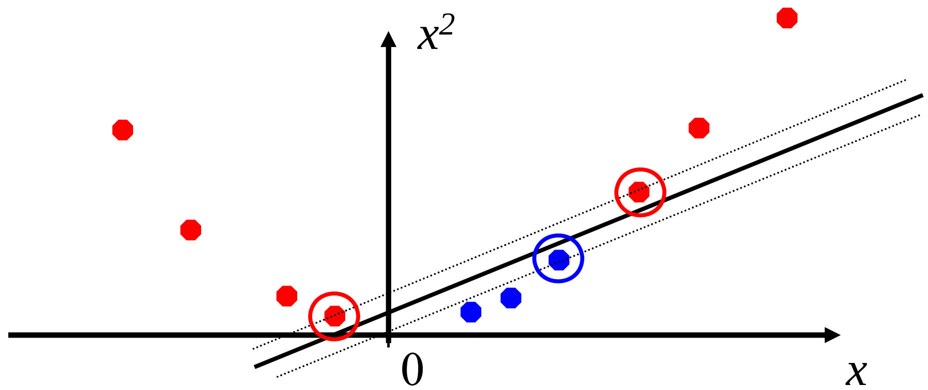
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?

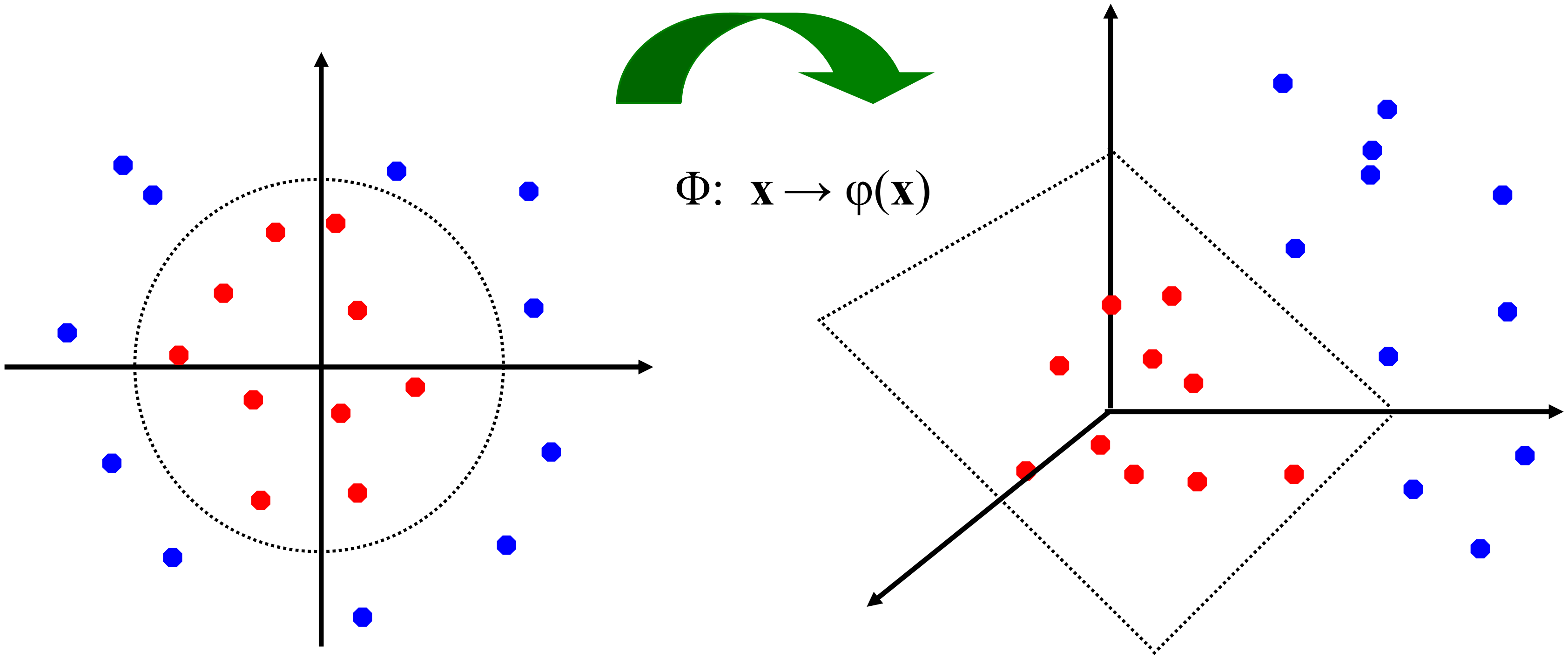


- We can map it to a higher-dimensional space:



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 $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

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

- 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, χ^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

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

- 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

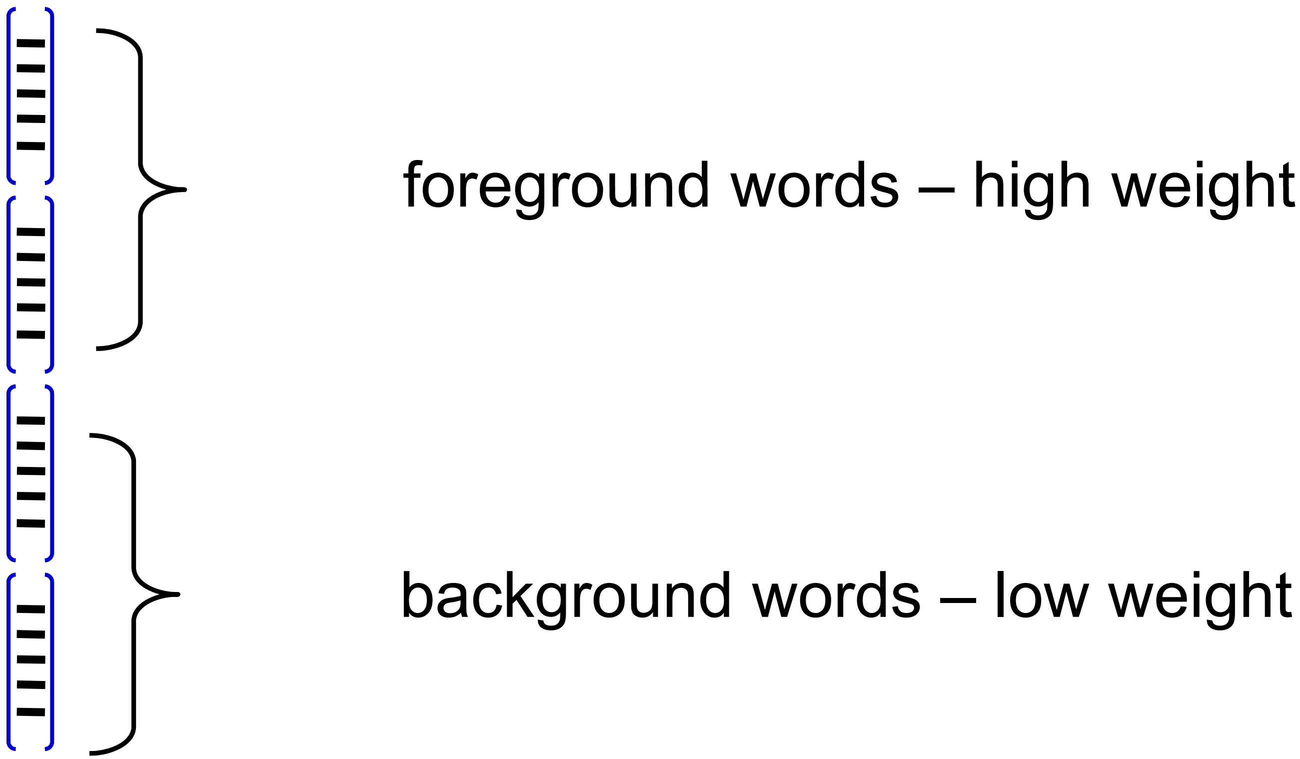
- Various direct formulations exist, but they are not widely used in practice. It is more common to obtain multi-class SVMs by combining two-class SVMs in various ways.

- 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

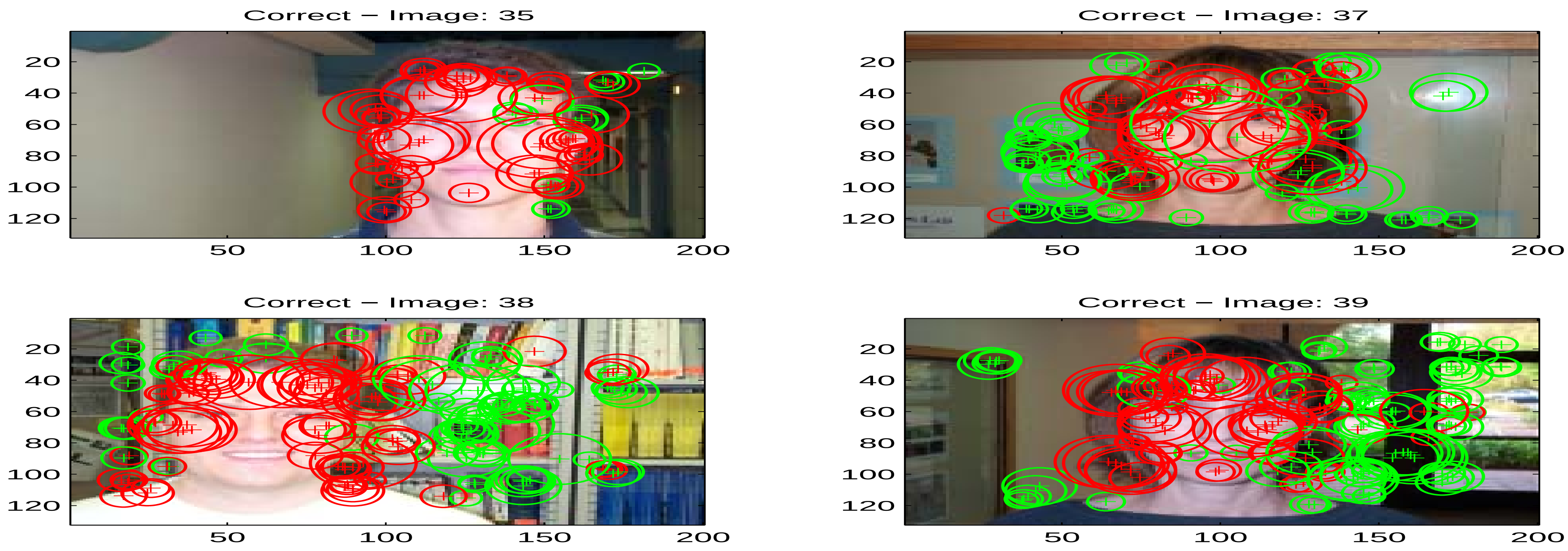
Why does SVM learning work?

- Learns foreground and background visual words



Illustration

Localization according to visual word probability



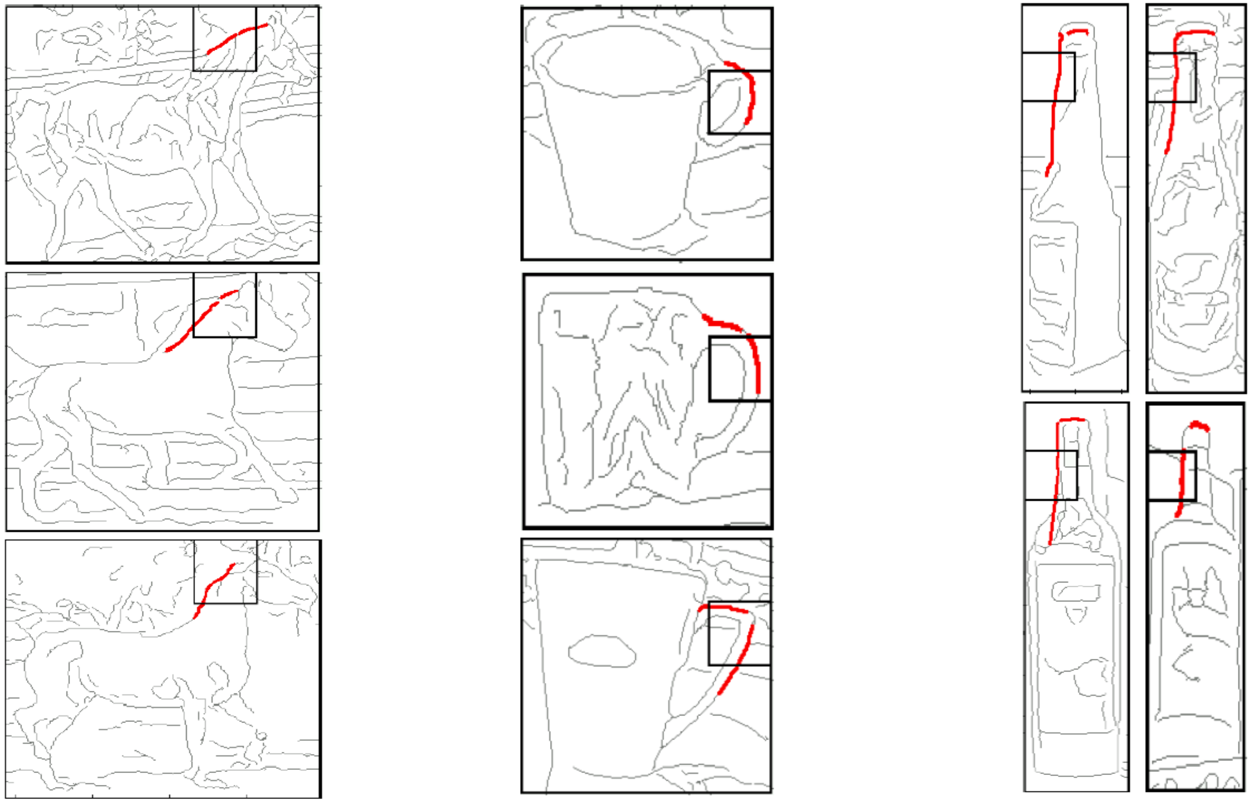
foreground word more probable

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

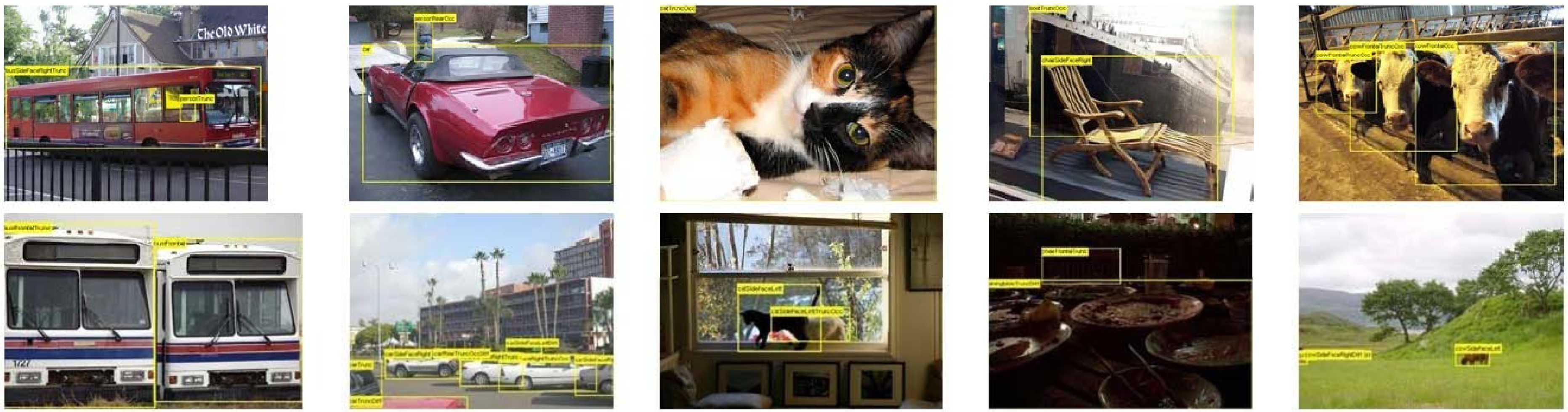
- PASCAL VOC 2007
  - Training *and* test dataset available
  - Used to report state-of-the-art results
  - Collected January 2007 from Flickr
  - 500 000 images downloaded and random subset selected
  - 20 classes
  - Class labels per image + bounding boxes
  - 5011 training images, 4952 test images

- Evaluation measure: average precision

PASCAL 2007 dataset

Aeroplane | Bicycle | Bird | Boat | Bottle

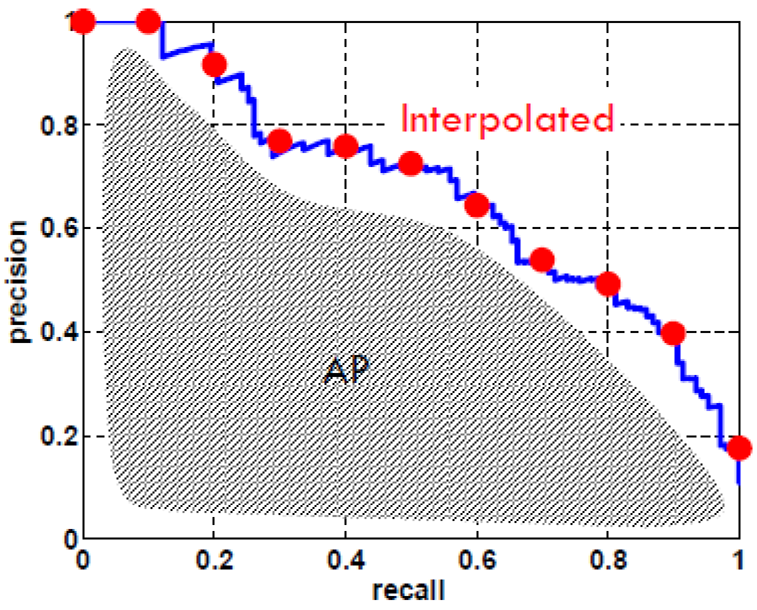
Bus | Car | Cat | Chair | Cow



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

- Dense is on average a bit better
- IP and dense are complementary, combination improves results

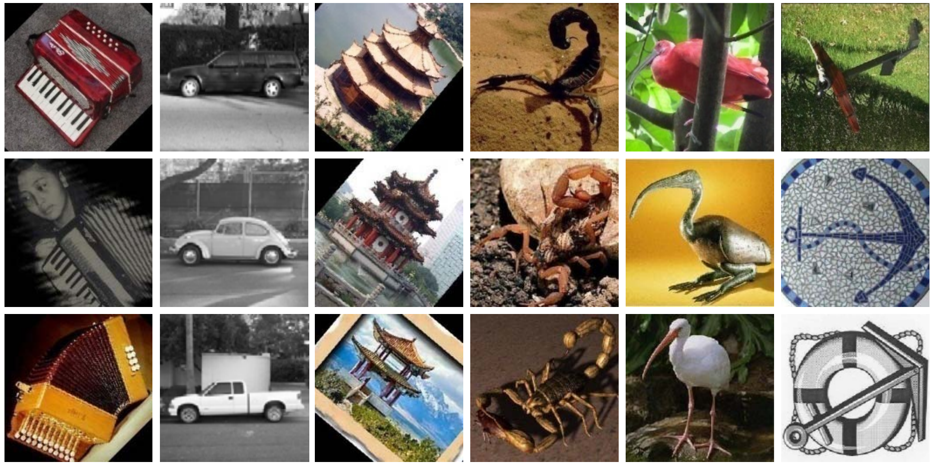
Spatial pyramid matching

- Add spatial information to the bag-of-features

- Perform matching in 2D image space

[Lazebnik, Schmid & Ponce, CVPR 2006]

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 %

Evaluation spatial pyramid

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

Evaluation spatial pyramid

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 spatial pyramid

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

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

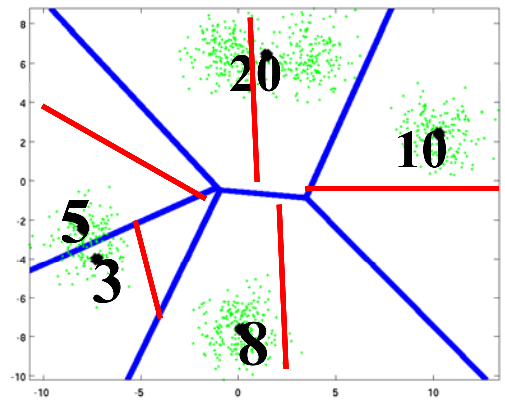
Recent extensions

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

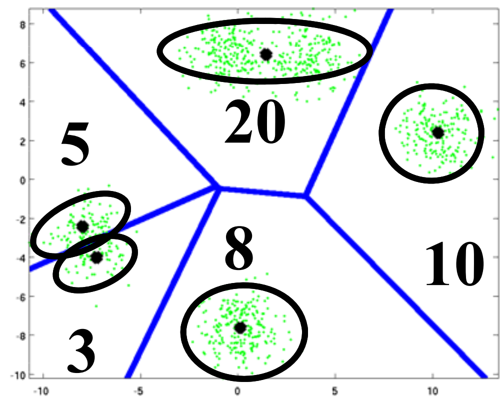
- Improving the Fisher Kernel for Large-Scale Image Classification, Perronnin et al., ECCV'10
  - More discriminative descriptor, power normalization, linear SVM

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 store: mean and variance of data in cell
  - Representation 2D times larger, at same computational cost
  - High dimensional, robust representation



Fisher vector image representation

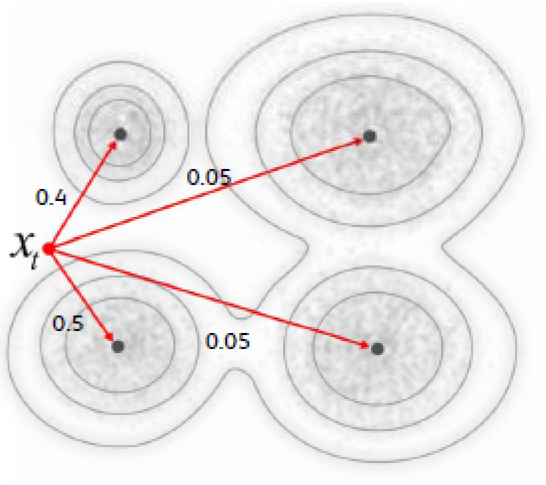
$X = \{x_t, t = 1 \ldots T\}$ is the set of T i.i.d. D-dim local descriptors (e.g. SIFT) extracted from an image:

$u_\lambda(x) = \sum_{i=1}^{K} w_i u_i(x)$ 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:

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

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



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

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 %

Large-scale image classification

- Image classification: assigning a class label to the image

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

- What makes it large-scale?
  - number of images
  - number of classes
  - dimensionality of descriptor

IMAGENET has 14M images from 22k classes

Large-scale image classification

- Image descriptors
  - Fisher vector (high dimensional)
  - Normalization: square-rooting or latent MOG+ L2 normalization

    [Image categorization using Fisher kernels of non-iid image models, Cinbis, Verbeek, Schmid, CVPR'12] [Perronnin'10]

- Classification approach
  - Linear classifiers
  - One versus rest classifier
  - Stochastic gradient descent optimization

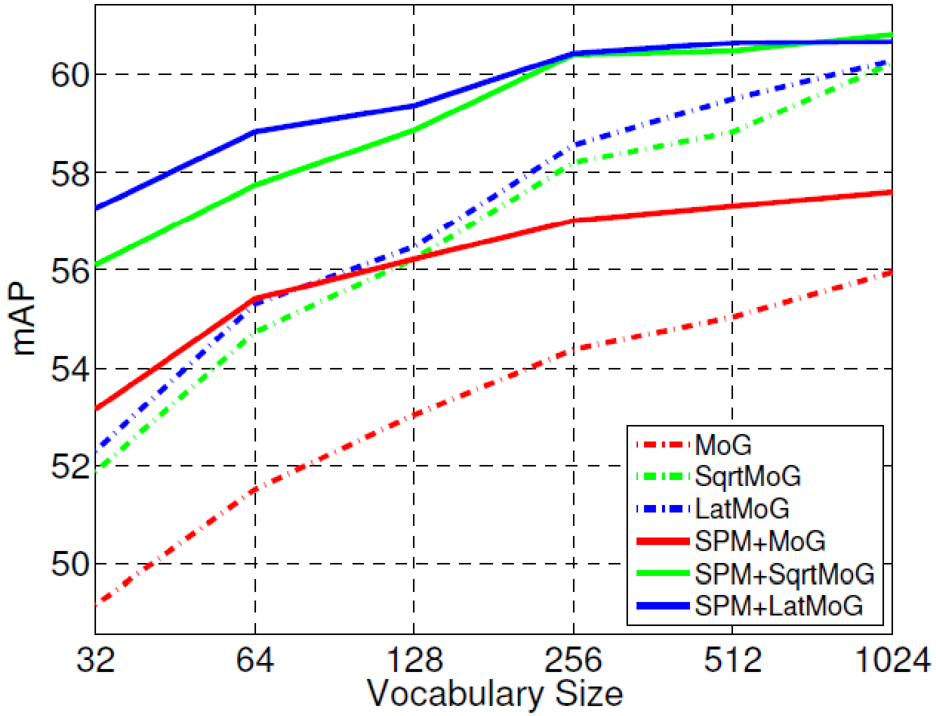
    [Towards good practice in large-scale learning for image classification, Perronnin, Akata, Harchaoui, Schmid, CVPR'12]

Evaluation image description

- Comparing on PASCAL VOC'07 linear classifiers with
  - Fisher vector
  - Sqrt transformation of Fisher vector
  - Latent GMM of Fisher vector

- Sqrt transform + latent MOG models lead to improvement

- State-of-the-art performance obtained with linear classifier



Evaluation image description

Fisher versus BOF vector + linear classifier on Pascal Voc'07

SPM	Method	64	128	256	512	1024
No	BoW	20.1	29.0	36.2	40.7	44.1
No	SqrtBoW	21.0	29.5	37.4	**41.3**	**46.1**
No	LatBoW	**22.9**	**30.1**	**38.9**	41.2	44.5
Yes	BoW	37.1	40.1	42.4	46.4	48.9
Yes	SqrtBoW	37.8	41.2	44.6	47.8	51.6
Yes	LatBoW	**39.3**	**41.7**	**45.3**	**48.7**	**52.2**

SPM	Method	32	64	128	256	512	1024
No	MoG	49.2	51.5	53.0	54.4	55.0	55.9
No	SqrtMoG	51.9	54.7	56.2	58.2	58.8	60.2
No	LatMoG	**52.3**	**55.3**	**56.5**	**58.6**	**59.5**	**60.3**
Yes	MoG	53.2	55.4	56.2	57.0	57.3	57.6
Yes	SqrtMoG	56.1	57.7	58.9	**60.4**	60.5	**60.8**
Yes	LatMoG	**57.3**	**58.8**	**59.4**	**60.4**	**60.6**	60.7

- Fisher improves over BOF
- Fisher comparable to BOF + non-linear classifier
- Limited gain due to SPM on PASCAL
- Sqrt helps for Fisher and BOF

Large-scale image classification

- Classification approach
  - One-versus-rest classifiers
  - stochastic gradient descent (SGD)
  - At each step choose a sample at random and update the parameters using a sample-wise estimate of the regularized risk

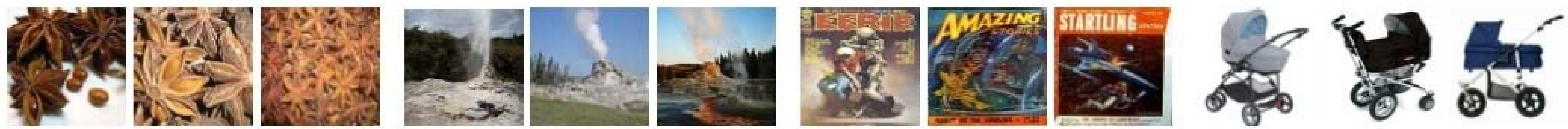
- Data reweighting
  - When some classes are significantly more populated than others, rebalancing positive and negative examples
  - Empirical risk with reweighting

$$\frac{\rho}{N_+} \sum_{i \in I_+} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) + \frac{1-\rho}{N_-} \sum_{i \in I_-} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w})$$

$\rho = 1/2$ Natural rebalancing, same weight to positive and negatives

Experimental results

- Datasets
  - ImageNet Large Scale Visual Recognition Challenge 2010 (ILSVRC)
    - 1000 classes and 1.4M images
  - ImageNet10K dataset
    - 10184 classes and ~ 9 M images



(a) Star Anise (92.45%) (b) Geyser (85.45%) (c) Pulp Magazine (83.01%) (d) Carrycot (81.48%)

(e) European gallinule (15.00%) (f) Sea Snake (10.00 %) (g) Paintbrush (4.68 %) (h) Mountain Tent (0.00%)

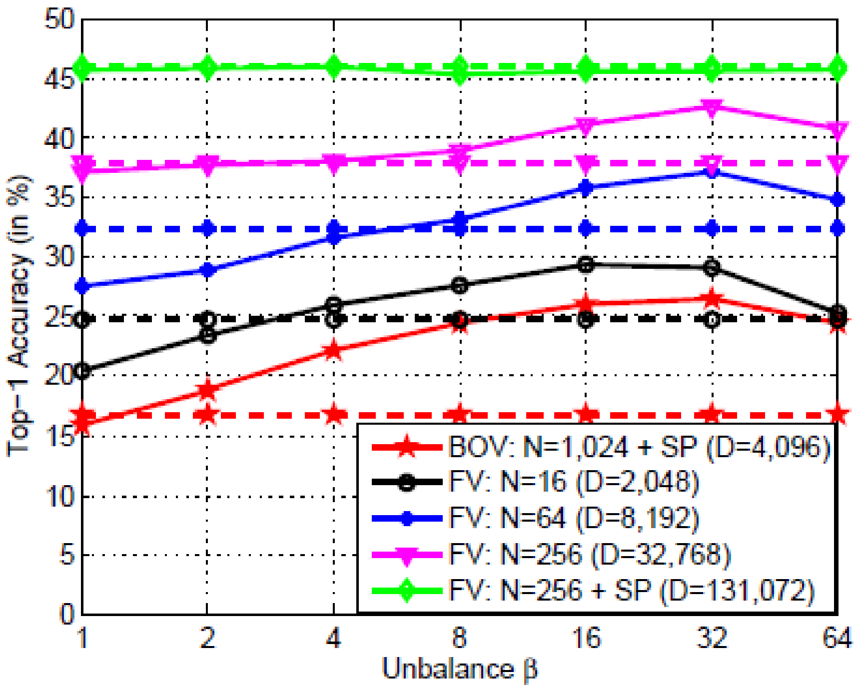
Experimental results

- Features: dense SIFT, reduced to 64 dim with PCA

- Fisher vectors
  - 256 Gaussians, using mean and variance
  - Spatial pyramid with 4 regions
  - Approx. 130K dimensions (4x [2x64x256])
  - Normalization: square-rooting and L2 norm

- BOF: dim 1024 + R=4
  - 4960 dimensions
  - Normalization: square-rooting and L2 norm

Importance of re-weighting



- Plain lines correspond to w-OVR, dashed one to u-OVR
- ß is number of negatives samples for each positive, β=1 natural rebalancing
- Results for ILSVRC 2010

- Significant impact on accuracy
- For very high dimensions little impact

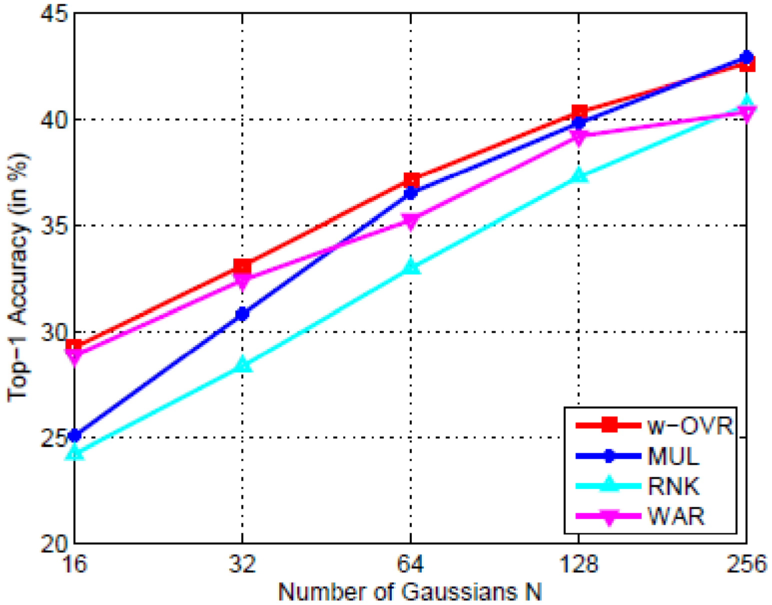
One-versus-rest works

- 256 Gaussian Fisher vector + SP with R=4 (dim 130k)
- BOF dim=1024 + SP with R=4 (dim 4000)
- Results for ILSVRC 2010
- FV >> BOF

		w-OVR
Top-1	BOV	26.4
	FV	45.7

Impact of the image signature size

- Fisher vector (no SP) for varying number of Gaussians + different classification methods, ILSVRC 2010



- Performance improves for higher dimensional vectors

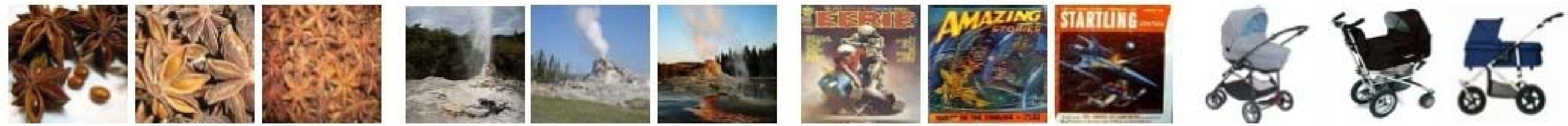
Large-scale experiment on ImageNet10k

	u-OVR	w-OVR
BOV 4K-dim	3.8	7.5
FV 130K-dim	16.7	19.1

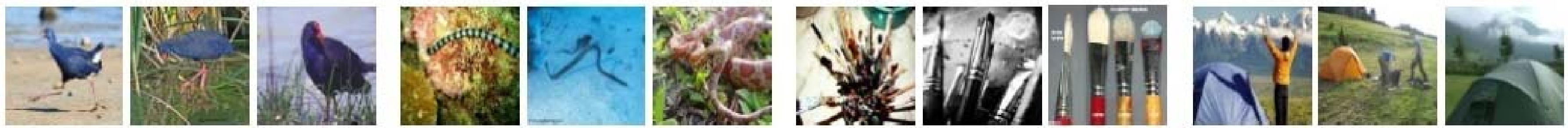
- Significant gain by data re-weighting, even for high-dimensional Fisher vectors
- w-OVR > u-OVR
- Improves over state of the art: 6.4% [Deng et. al] and WAR [Weston et al.]

Large-scale experiment on ImageNet10k

- Illustration of results obtained with w-OVR and 130K-dim Fisher vectors, ImageNet10K top-1 accuracy



(a) Star Anise (92.45%) (b) Geyser (85.45%) (c) Pulp Magazine (83.01%) (d) Carrycot (81.48%)



(e) European gallinule (15.00%) (f) Sea Snake (10.00 %) (g) Paintbrush (4.68 %) (h) Mountain Tent (0.00%)

Conclusion

- *Stochastic training:* learning with SGD is well-suited for large-scale datasets
- *One-versus-rest*: a flexible option for large-scale image classification
- *Class imbalance:* optimize the imbalance parameter in one-versus-rest strategy is a must for competitive performance