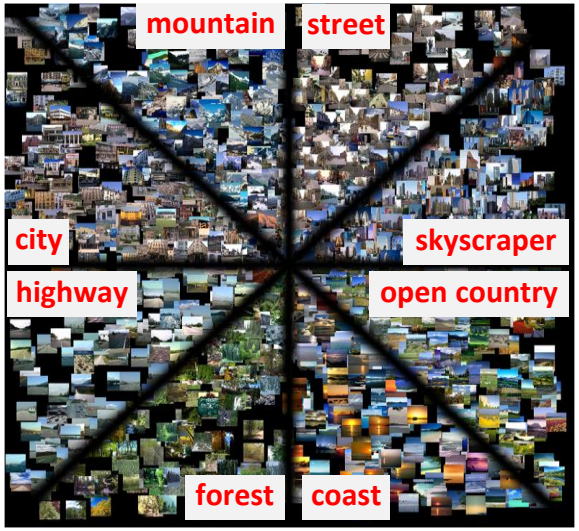
Master in Computer Vision *Barcelona*

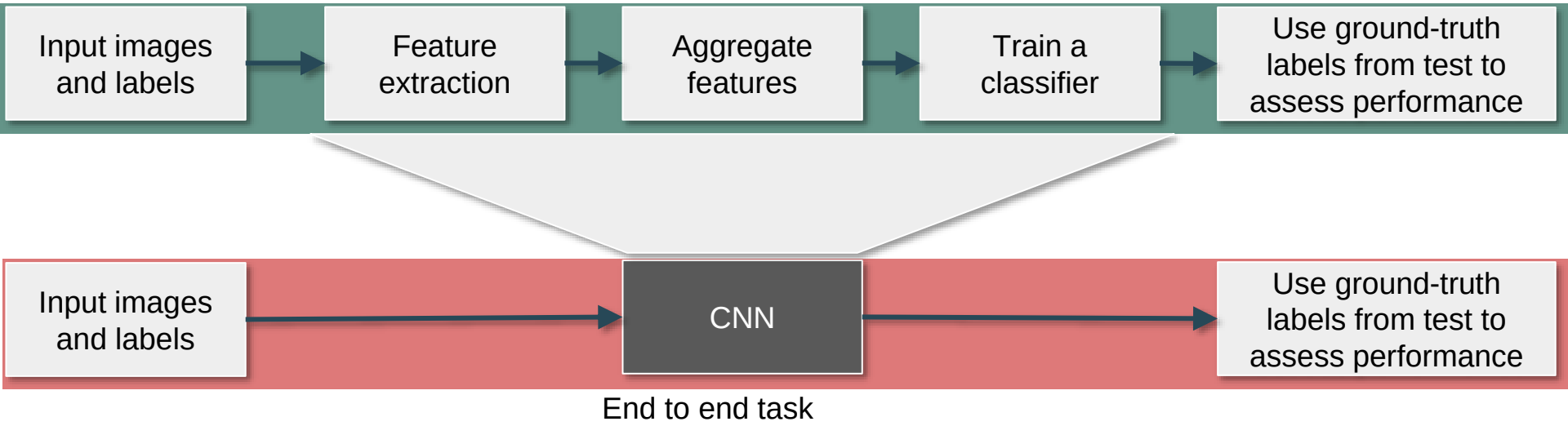
Module 3: Machine learning for computer vision

Project: Image Classification

Lecturer: Ramon Baldrich, ramon.baldrich@uab.cat

The aim of this module is to learn the techniques for category classification: handcrafted and learned.





End to end task

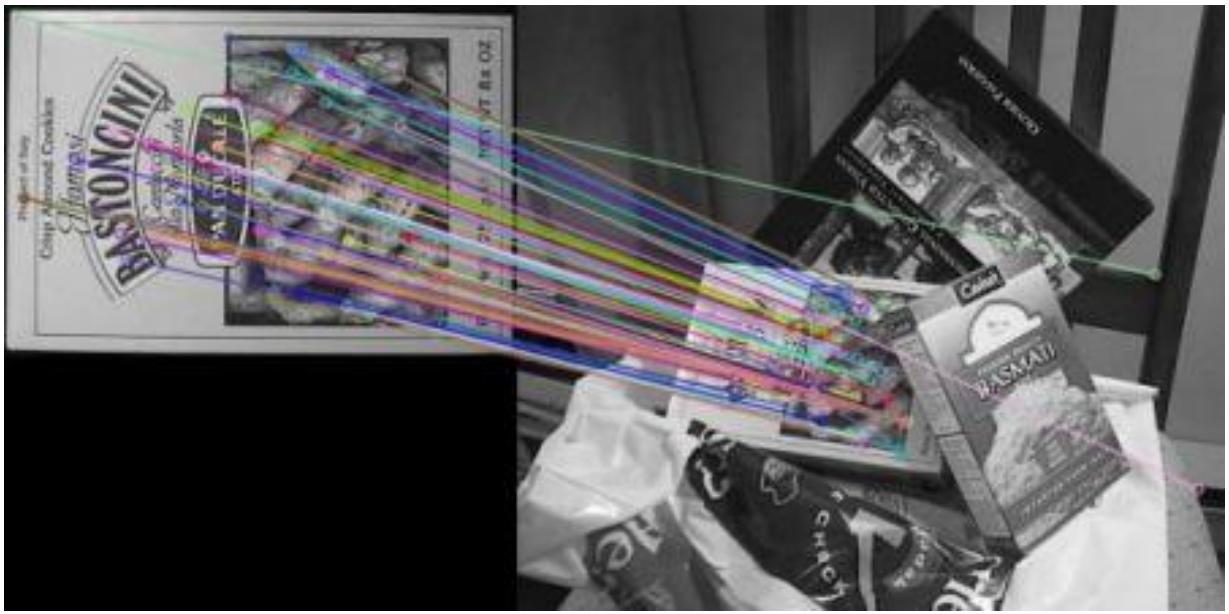
Preamble: Local descriptors

Local descriptors

- keypoint detection
- local description with strong invariance

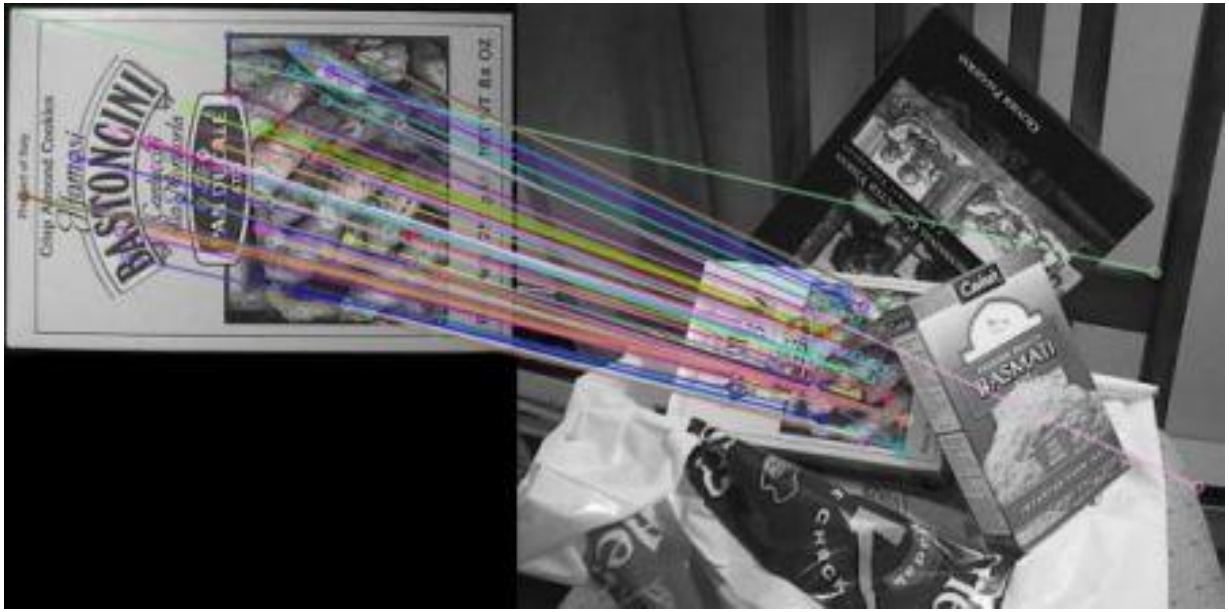
Object detection, localization, recognition, matching…

- Pair template objects within clutter environments



Preamble: Local descriptors

- SIFT (D. Lowe ICCV99, IJCV04)
- SURF,
- KAZE,
- BRIEF,
- BRISK,
- ORB...



Preamble: Local descriptors

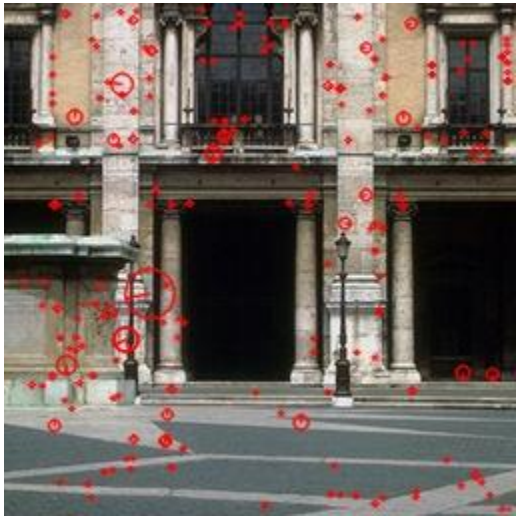
Can we use such local features for image categorization?

Preamble: Local descriptors

Use of local features (e.g. SIFT) for image categorization

Robust local features:

- Scale
- Viewpoint
- Partial occlusions
- Noise

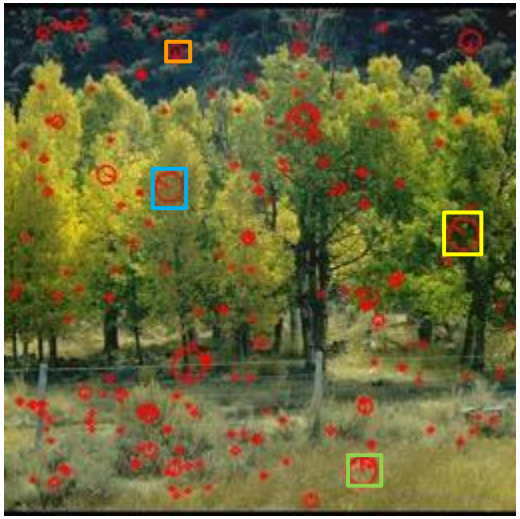


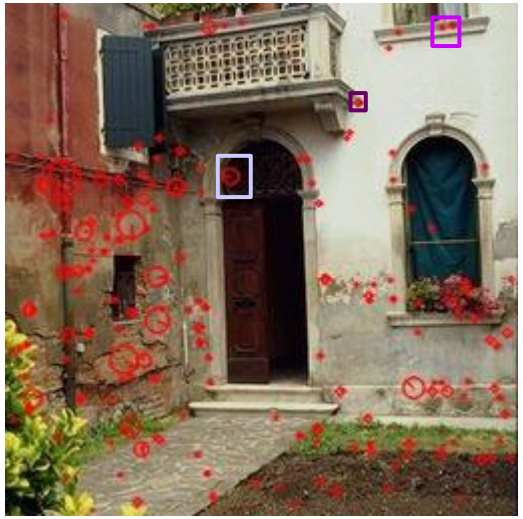
Preamble: Local descriptors

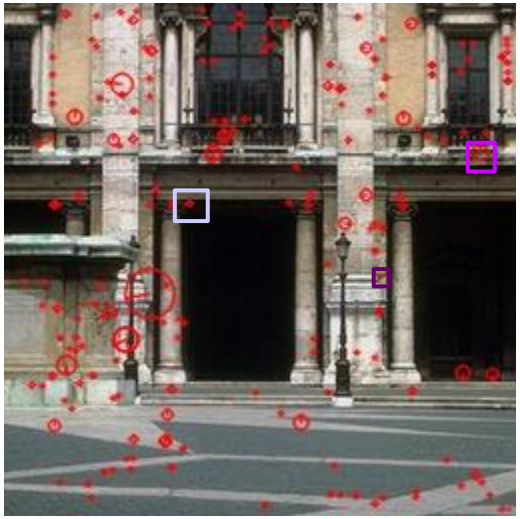
Use of local features for image categorization

Basic assumption:

- Images of the same class have similar local descriptors





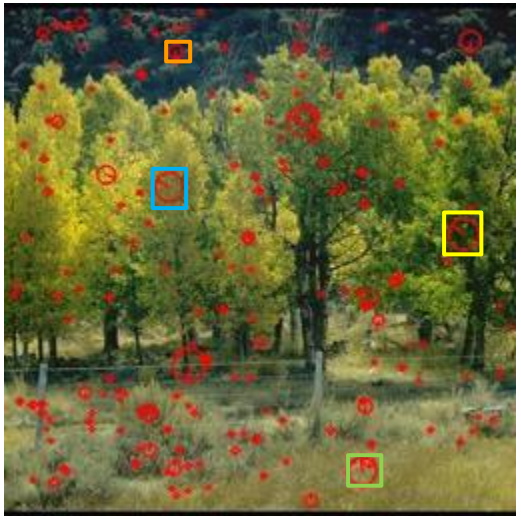


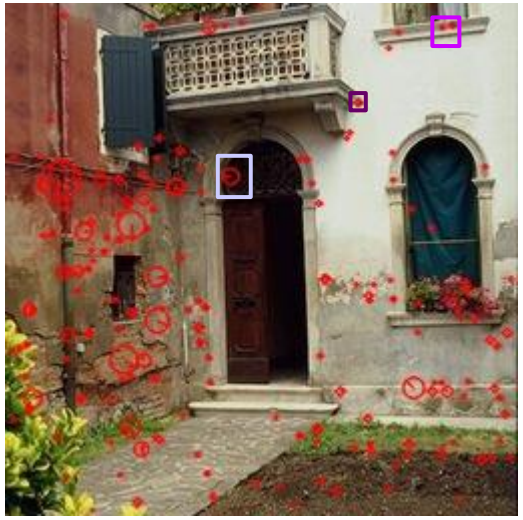
Preamble: Local descriptors

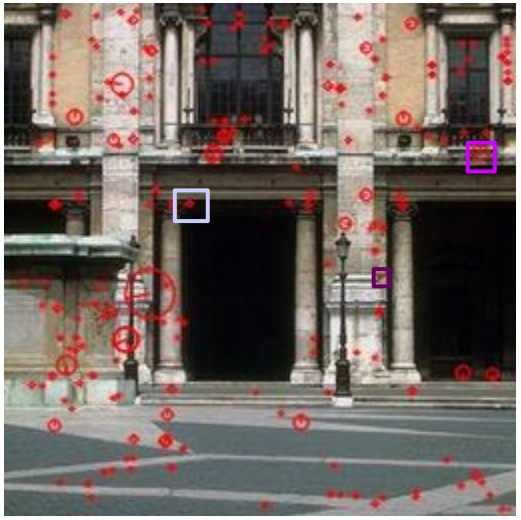
Use of local features for image categorization

Basic assumption:

- Images of the same class have similar local descriptors
- Images of different classes have different local descriptors





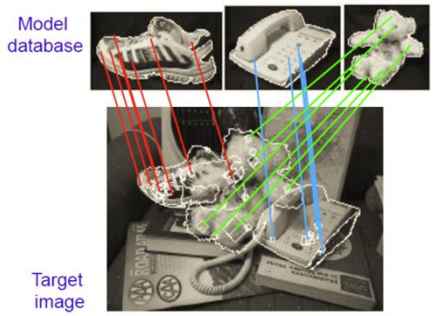


Motivation

Local features are **well suited** for **image categorization**

A **generic approach** could be:

1. Local feature extraction and description (ex. SIFT)
2. Matching local features based on similarity of local appearance
   - For every keypoint in one image find the closest keypoint (in the feature space) in the other image
   - Verify matches based on semi-local/global geometric relations



[D. Lowe, 1999]

D. Lowe. *Object Recognition from Local Scale-Invariant Features*. ICCV 1999

Motivation

Local features are **well suited** for **image categorization**

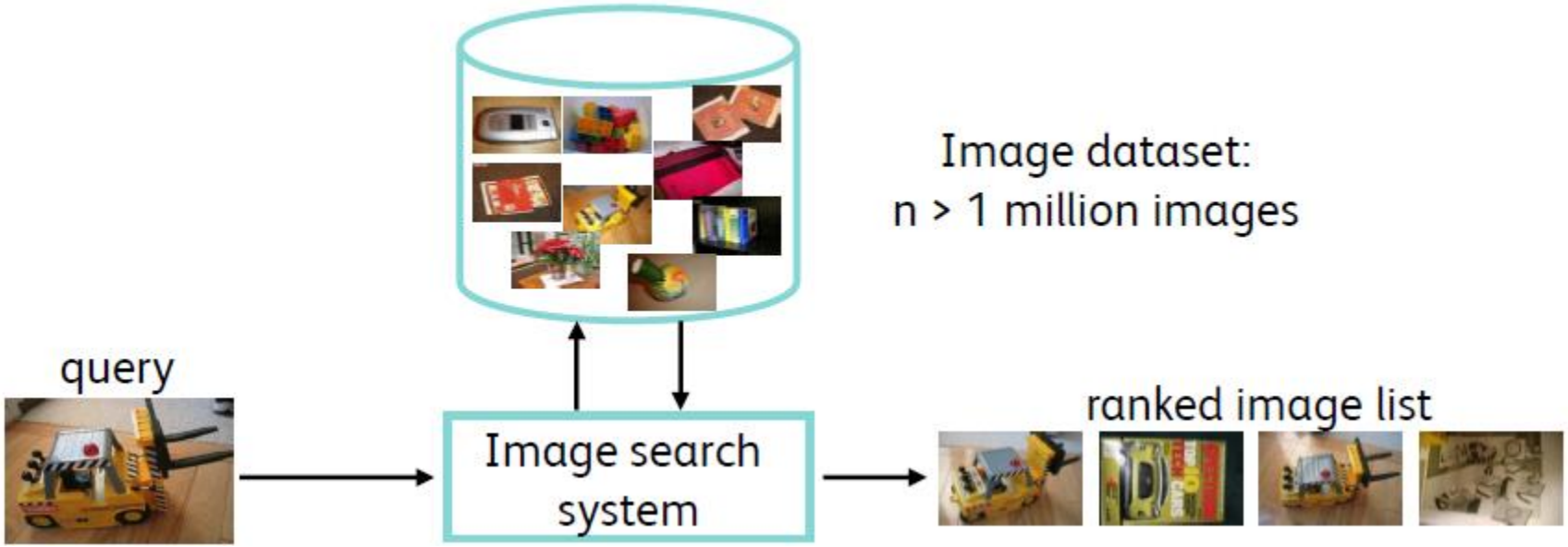
A **generic approach** could be:

1. Local feature extraction and description (ex. SIFT)
2. Matching local features based on similarity of local appearance
   - For every keypoint in one image find the closest keypoint (in the feature space) in the other image
   - Verify matches based on semi-local/global geometric relations

but…

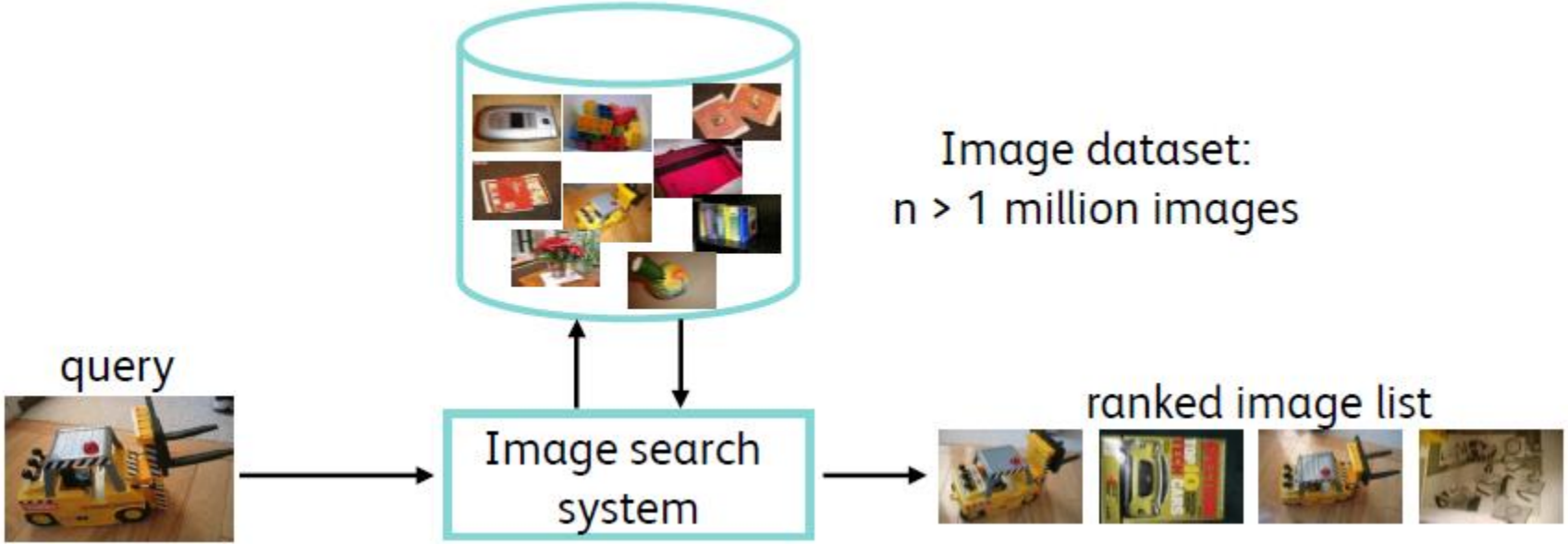
- Difficult to scale with the number of classes
- Computationally expensive
- Not well suited for applying machine learning

Let's do some numbers...



Slide credit: F. Perronin LSVR Tutorial CVPR13

Let's do some numbers...



- An image is described by m=1000 SIFT descriptors (d=128)
  - n*m= 1 billion descriptors to index
- Database representation: 128 GB RAM
- Search m^2 x n x d elementary operations!

Slide credit: F. Perronin LSVR Tutorial CVPR13

Motivation

Local features are **well suited** for **image categorization**

A **generic approach** could be:

1. Local feature extraction and description (ex. SIFT)
2. Matching local features based on similarity of local appearance
   - For every keypoint in one image find the closest keypoint (in the feature space) in the other image
   - Verify matches based on semi-local/global geometric relations

but…

- Difficult to scale with the number of classes
- Computationally expensive
- Not well suited for applying machine learning

… therefore, **better** if we can obtain a **global image representation** from the set of local features

Any thoughts??

Can we use such local features for image categorization?
So that:

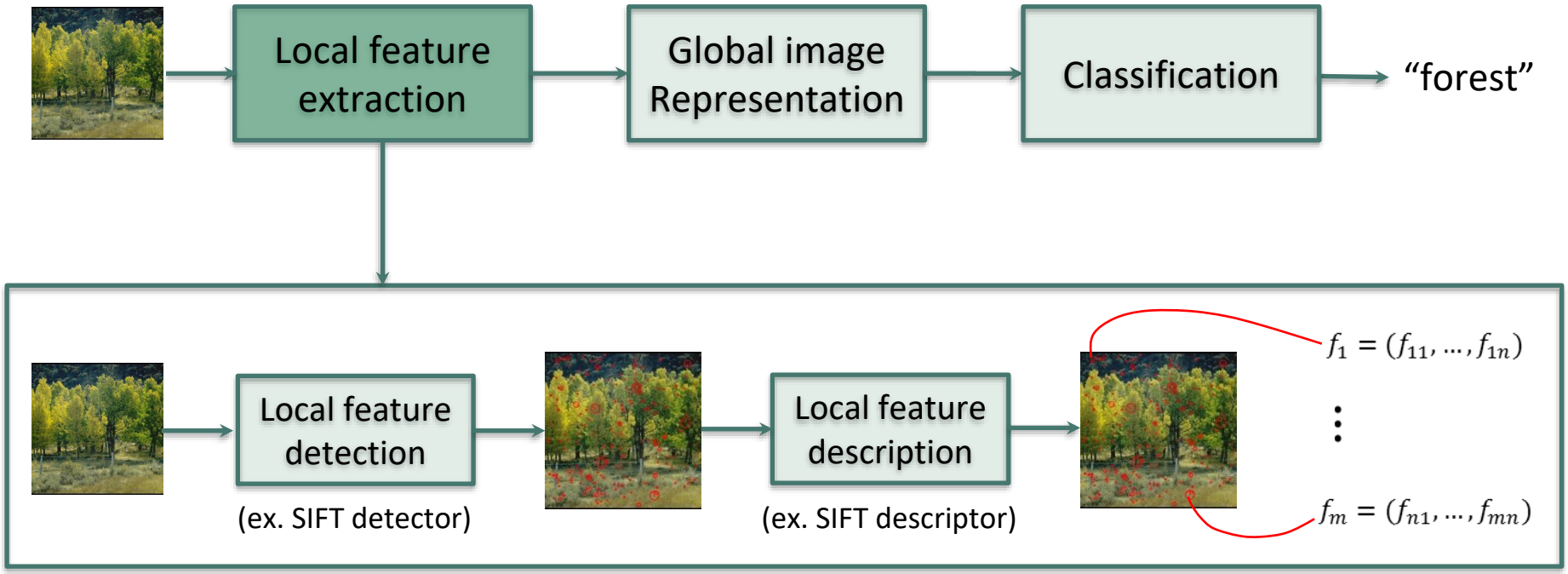
- keep discriminative power,
- we can scale,
- can apply statistical classifiers,
- ...

How can we go from a set of local descriptors to a single fixed-length global representation for each image?

Pipeline for Image Categorization

Local feature extraction → Global image Representation → Classification → “forest”

Pipeline for Image Categorization



Pipeline for Image Categorization

Local feature extraction

Global image Representation

Classification

"forest"

$f_1 = (f_{11}, \ldots, f_{1n})$

$\vdots$

$f_m = (f_{n1}, \ldots, f_{mn})$

Local feature aggregation

(ex. BoW framework)

$x = (x_1, \ldots, x_k)$

Pipeline for Image Categorization

Local feature extraction → Global image Representation → Classification → "forest"

Training

Training set

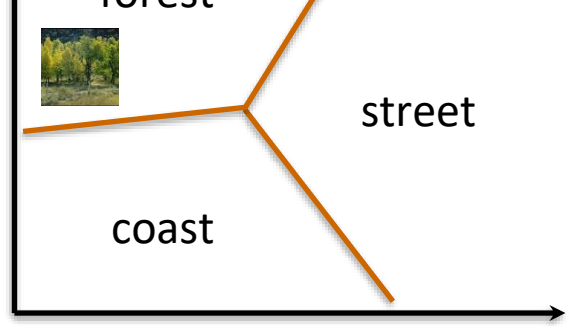
$x = (x_1, \ldots, x_k)$

$x = (x_1, \ldots, x_k)$

$x = (x_1, \ldots, x_k)$

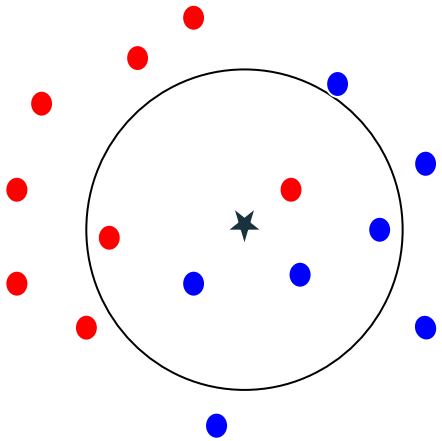
Test

$x = (x_1, \ldots, x_k)$



Pipeline for Image Categorization

Local feature extraction → Global image Representation → Classification → “forest”



k-NN classifier (for this first session), later on we will move to more powerful statistical classifiers (e.g. SVMs)

Image representation: The Bag of Words model

Local feature extraction

Global image Representation

Classification

"forest"

$f_1 = (f_{11}, \dots, f_{1n})$

$\vdots$

$f_m = (f_{n1}, \dots, f_{mn})$

Local feature aggregation

(ex. BoW framework)

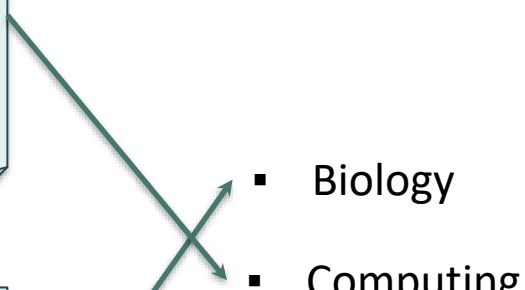
$x = (x_1, \dots, x_k)$

The Bag of Words model

Inspiration: document categorization

Just as humans use our eyes and brains to understand the world around us, computer vision tries to produce the same effect so that computers can perceive and understand an image or sequence of images and act as appropriate in a given situation. Data acquisition is achieved by various means such as image sequences, viewed from various video cameras, or multidimensional data from a medical scanner.

The sense of sight or vision is ensured by a receptor organ, the eye; a membrane, the retina, these receive the light impressions and transmit them to the brain through the optic pathways. The eye is a paired organ located in the orbital cavity. It is protected by the eyelids and by the secretion of the lacrimal gland. It is mobilized by a group of extrinsic muscles commanded by the motor nerves of the eye.



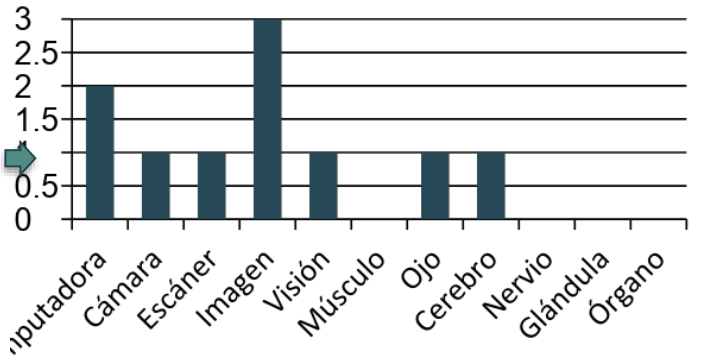
The Bag-of-Words model

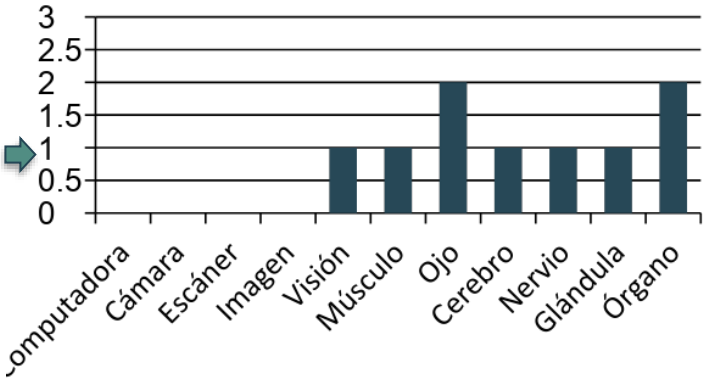
Inspiration: document categorization

Just as humans use our eyes and brains to understand the world around us, computer vision tries to produce the same effect so that computers can perceive and understand an image or sequence of images and act as appropriate in a given situation. Data acquisition is achieved by various means such as image sequences, viewed from various video cameras, or multidimensional data from a medical scanner.

The sense of sight or vision is ensured by a receptor organ, the eye; a membrane, the retina, these receive the light impressions and transmit them to the brain through the optic pathways. The eye is a paired organ located in the orbital cavity. It is protected by the eyelids and by the secretion of the lacrimal gland. It is mobilized by a group of extrinsic muscles commanded by the motor nerves of the eye.

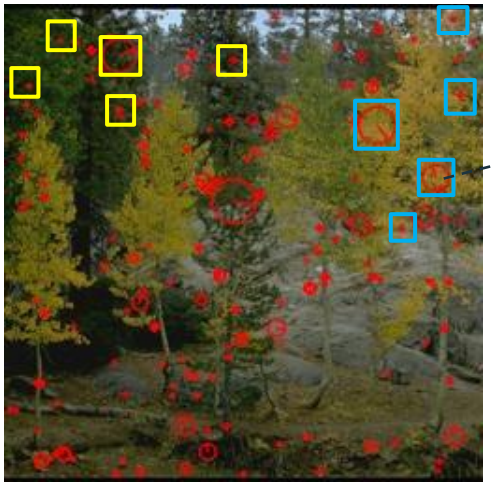
Histogram of representative words (Bag of Words)



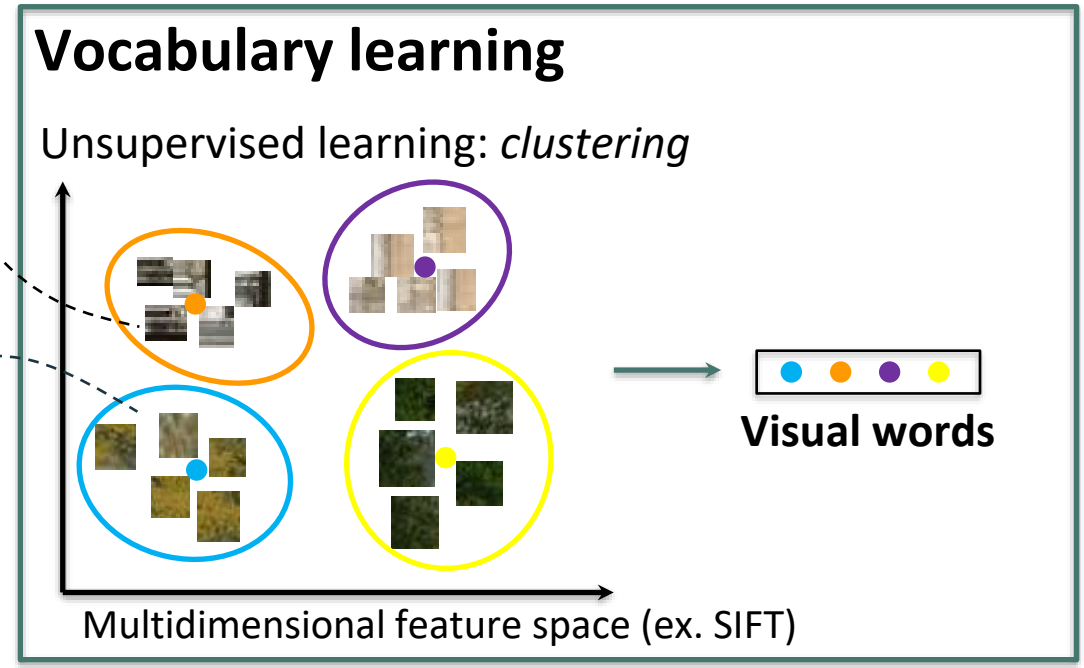


The Bag of Words model

Adapting the model to visual recognition: *Bag of Visual Words*



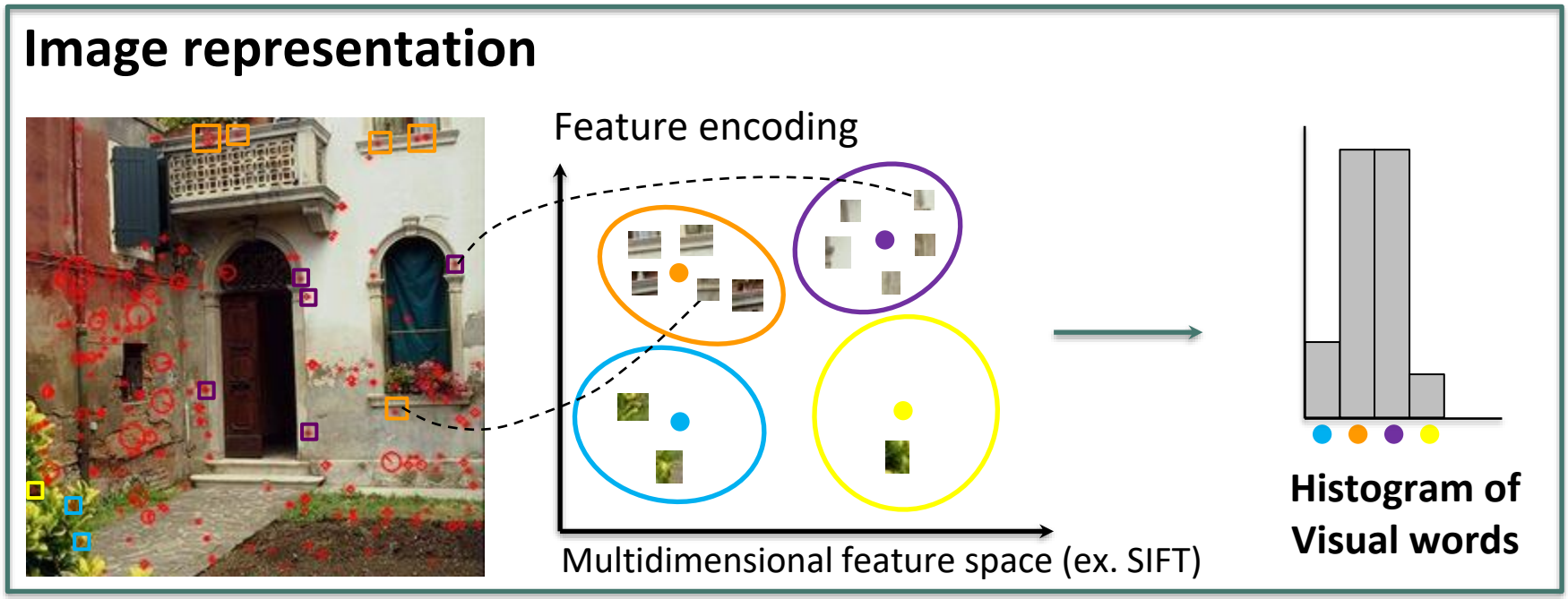
- We do not have a predefined set of relevant visual features
- We must Identify relevant common visual features: *visual words*



The Bag of Words model

Adapting the model to visual recognition: *Bag of Visual Words*

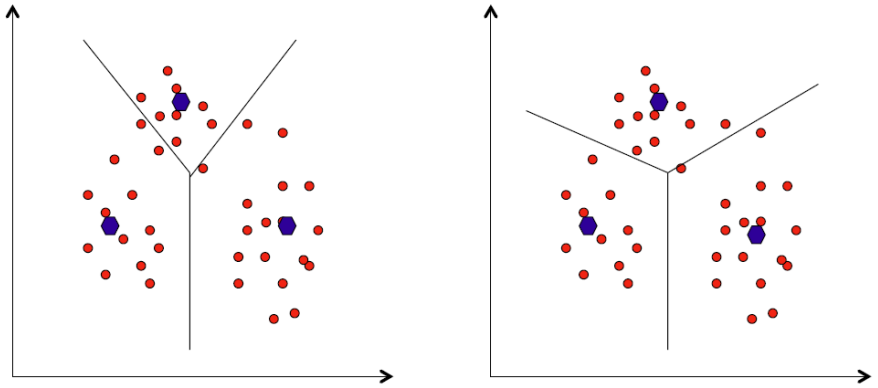
- Every local feature in the image can be assigned to one visual word
- Image representation: histogram of *visual words*



Vocabulary learning

k-means algorithm (M 1 – Lecture 10)

- K-means algorithm: example (II)

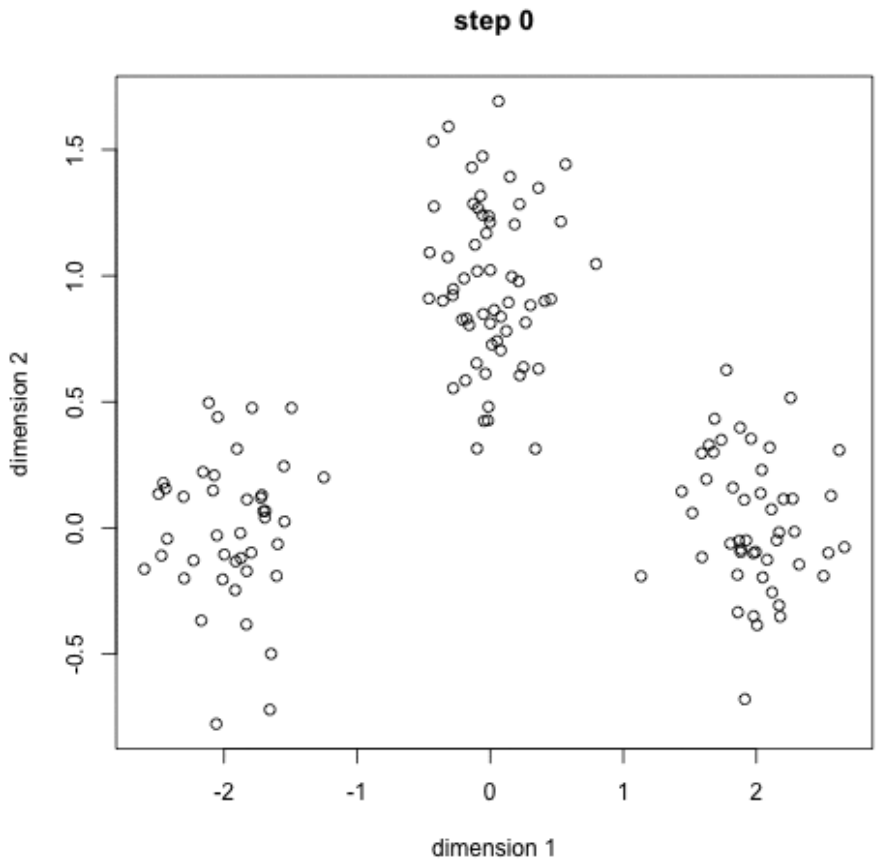


1. Initialize K classes. Compute the centers of each class
2. For each point:
   a. Compute the distances between the point and the class centers
   b. Assign the point to the closest class
3. Update the class centers
4. Repeat 2 & 3 until no change (in assignments or center values) is observed.

Max Lloyd algorithm

Vocabulary learning

k-means algorithm (M 1 – Lecture 10)



1. Initialize K classes. Compute the centers of each class
2. For each point:
   a. Compute the distances between the point and the class centers
   b. Assign the point to the closest class
3. Update the class centers
4. Repeat 2 & 3 until no change (in assignments or center values) is observed.

Max Lloyd algorithm

BoVW recap

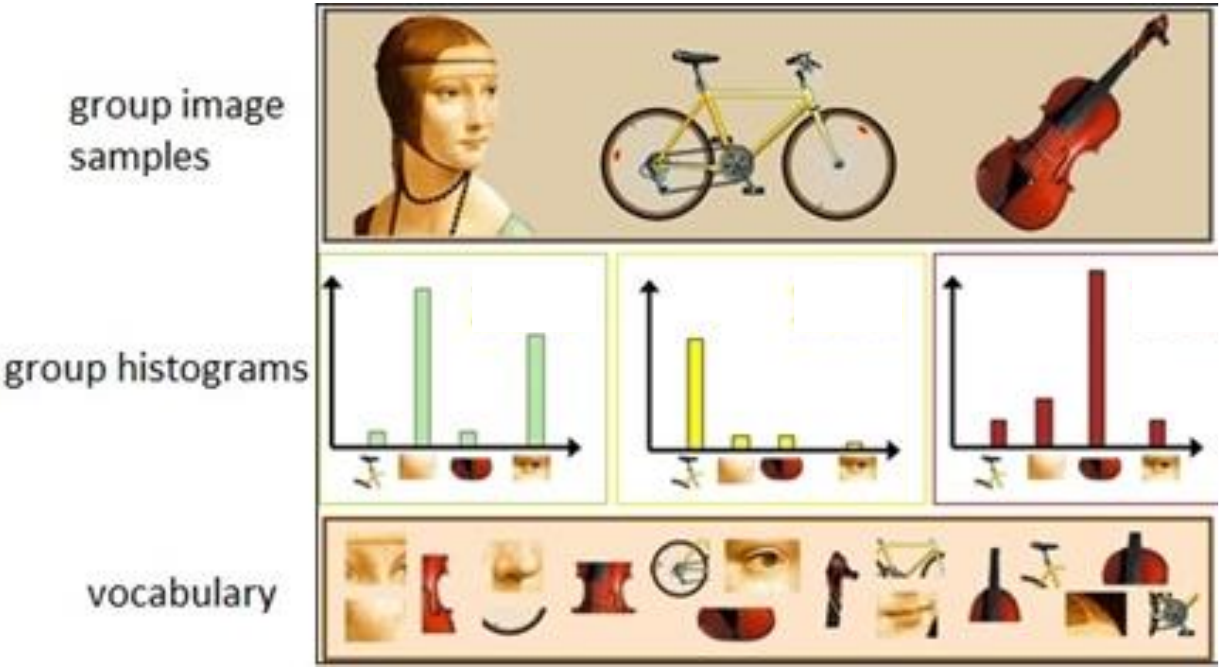
group image samples

BoVW recap

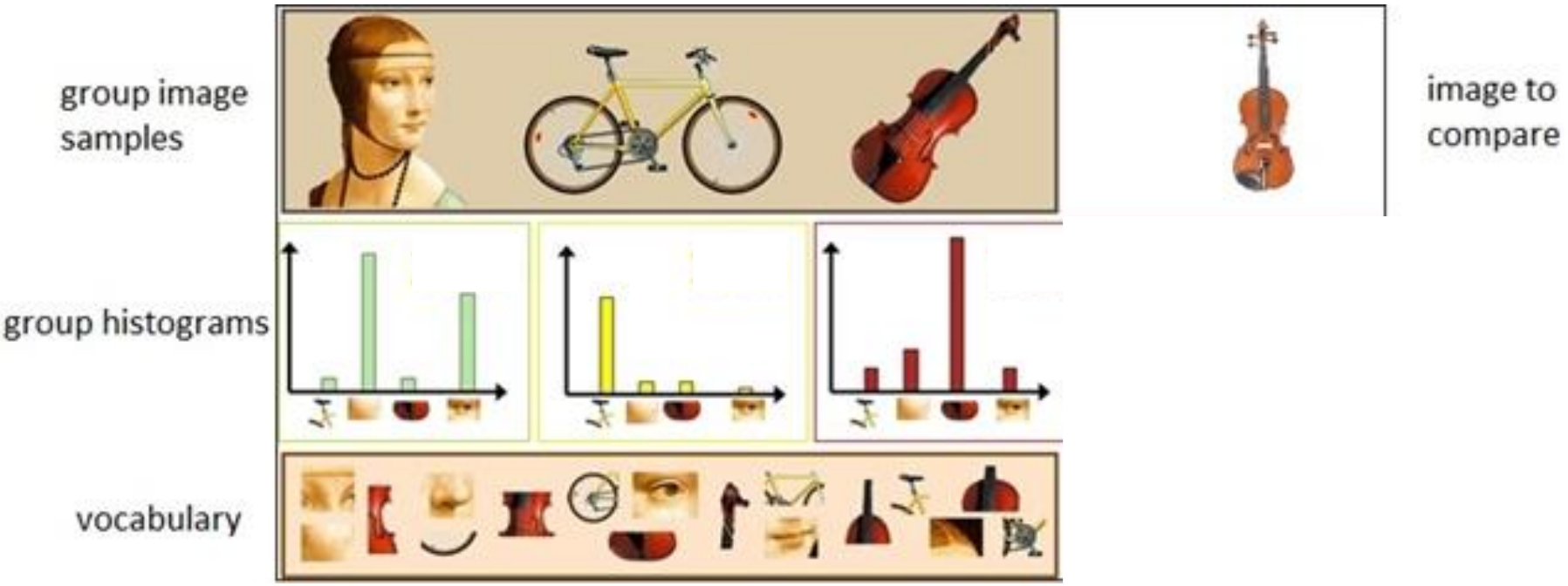
group image samples

vocabulary

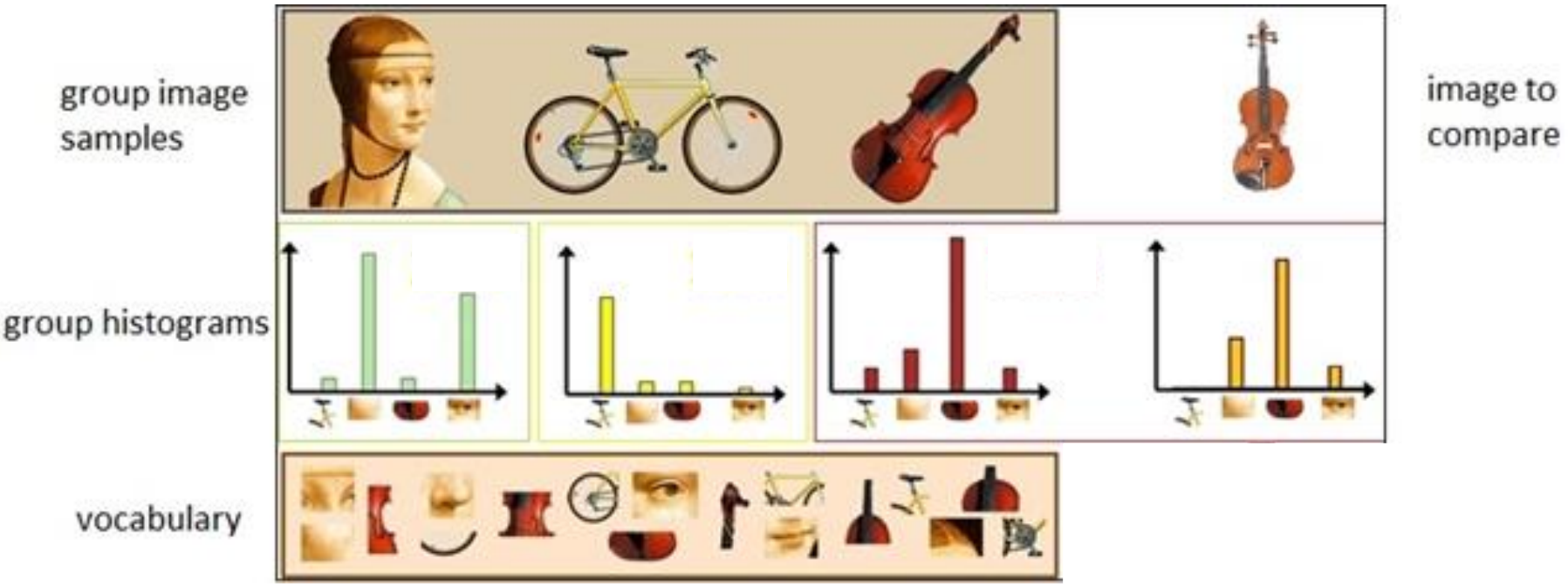
BoVW recap



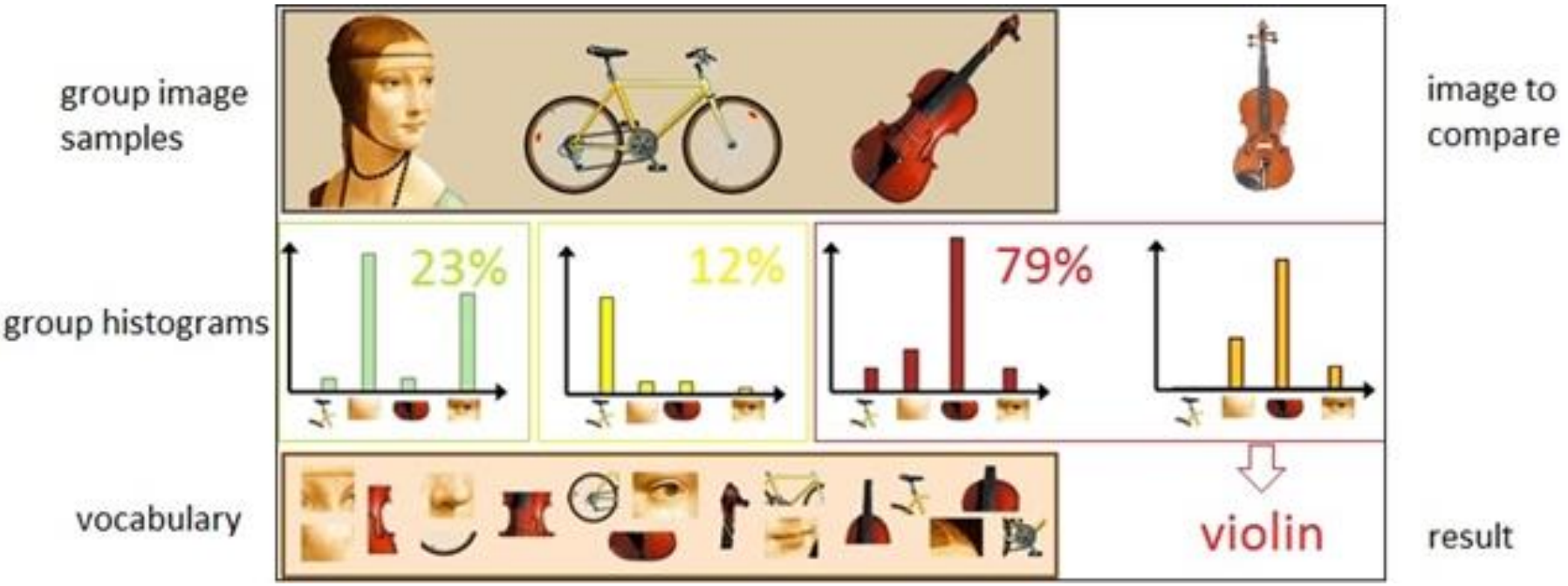
BoVW recap


group image samples
image to compare
group histograms
vocabulary

BoVW recap


group image samples
image to compare
group histograms
vocabulary

BoVW recap


group image samples
image to compare
group histograms
23%
12%
79%
vocabulary
violin
result

WAKE UP!! Surprise test!!

- Would it be a good idea to use the BoVW framework as explained using the ORB local descriptors? Why?

WAKE UP!! Surprise test!!

- Would it be a good idea to use the BoVW framework as explained using the ORB local descriptors? Why?

- What are the effects of choosing a too low or too high value for *k*?

WAKE UP!! Surprise test!!

- Would it be a good idea to use the BoVW framework as explained using the ORB local descriptors? Why?

- What are the effects of choosing a too low or too high value for *k*?

- Is there any way of performing well (i.e. have a good description) in images where the keypoint detector step tend to perform poorly (e.g. low textures, or repetitive patterns)?

WAKE UP!! Surprise test!!

- Would it be a good idea to use the BoVW framework as explained using the ORB local descriptors? Why?

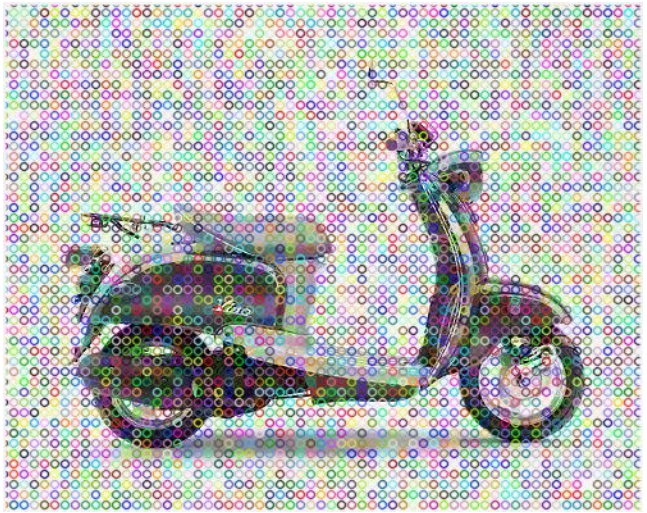
- What are the effects of choosing a too low or too high value for *k*?

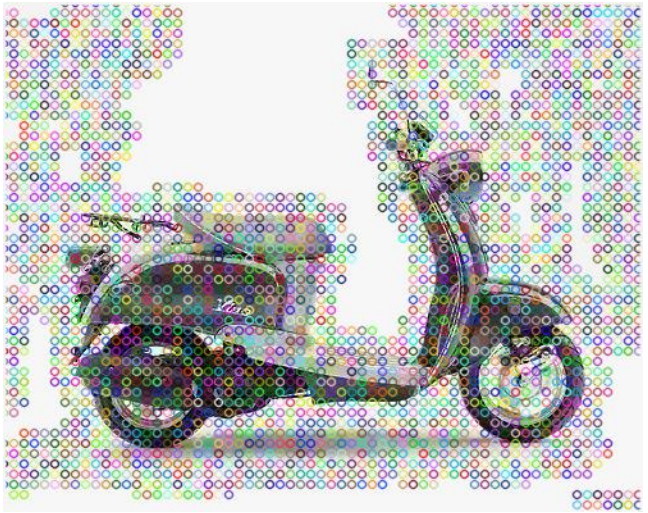
- Is there any way of performing well (i.e. have a good description) in images where the keypoint detector step tend to perform poorly (e.g. low textures, or repetitive patterns)?

- What kind of information is lost when using the BoVW framework compared to local keypoint matching? is there any way of include it in a coarse manner?

Beyond BoVW: Dense SIFT

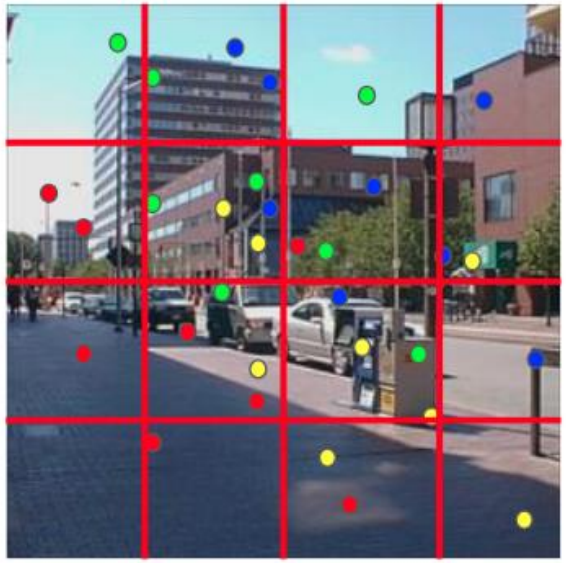
- Is there any way of performing well (i.e. have a good description) in images where the keypoint detector step tend to perform poorly (e.g. low textures, or repetitive patterns)?





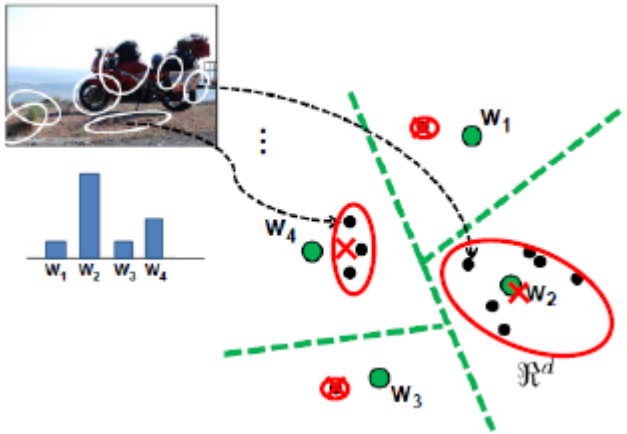
Beyond BoVW: Spatial Pyramids

- What kind of information is lost when using the BoVW framework compared to local keypoint matching? is there any way of include it in a coarse manner?



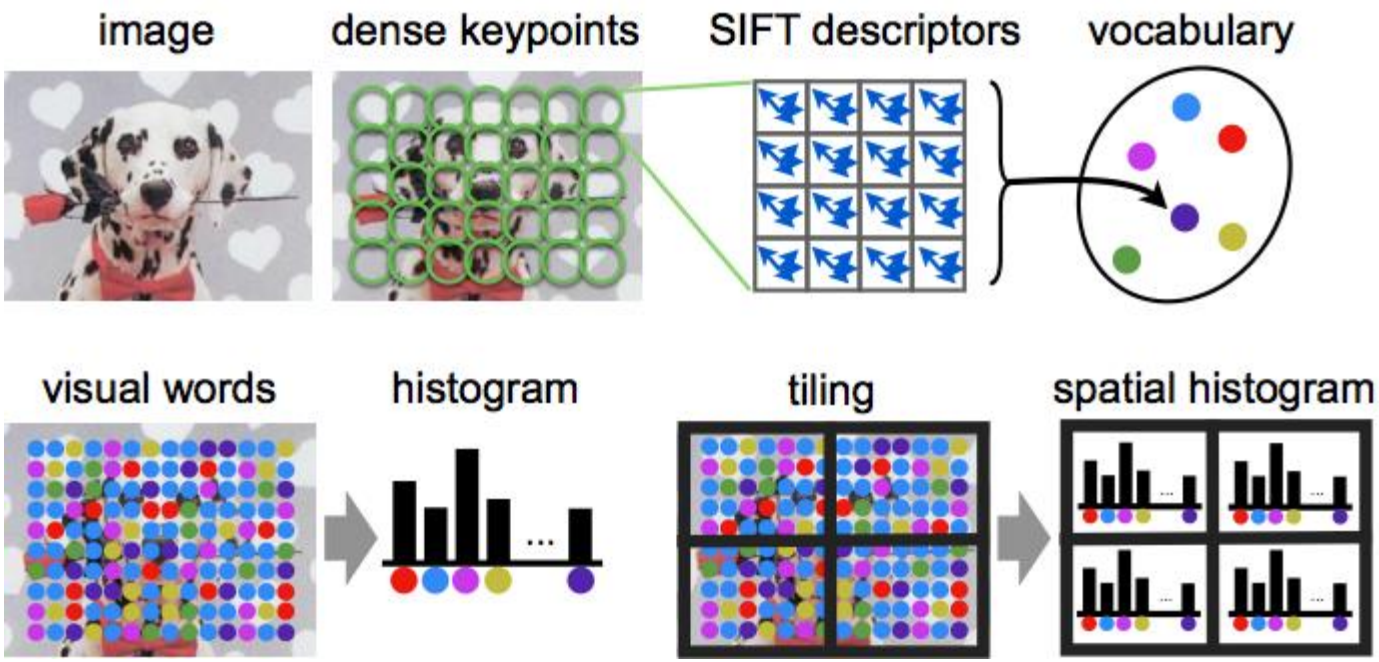
Beyond BoVW: Fisher Vectors

- BoVW is only counting the number of local descriptors assigned to each Voronoi cell
- Why not including higher order statistics?
  - Mean of local descriptors
  - Co-variance of local descriptors
- FV is typically 2 x D x k dimensional

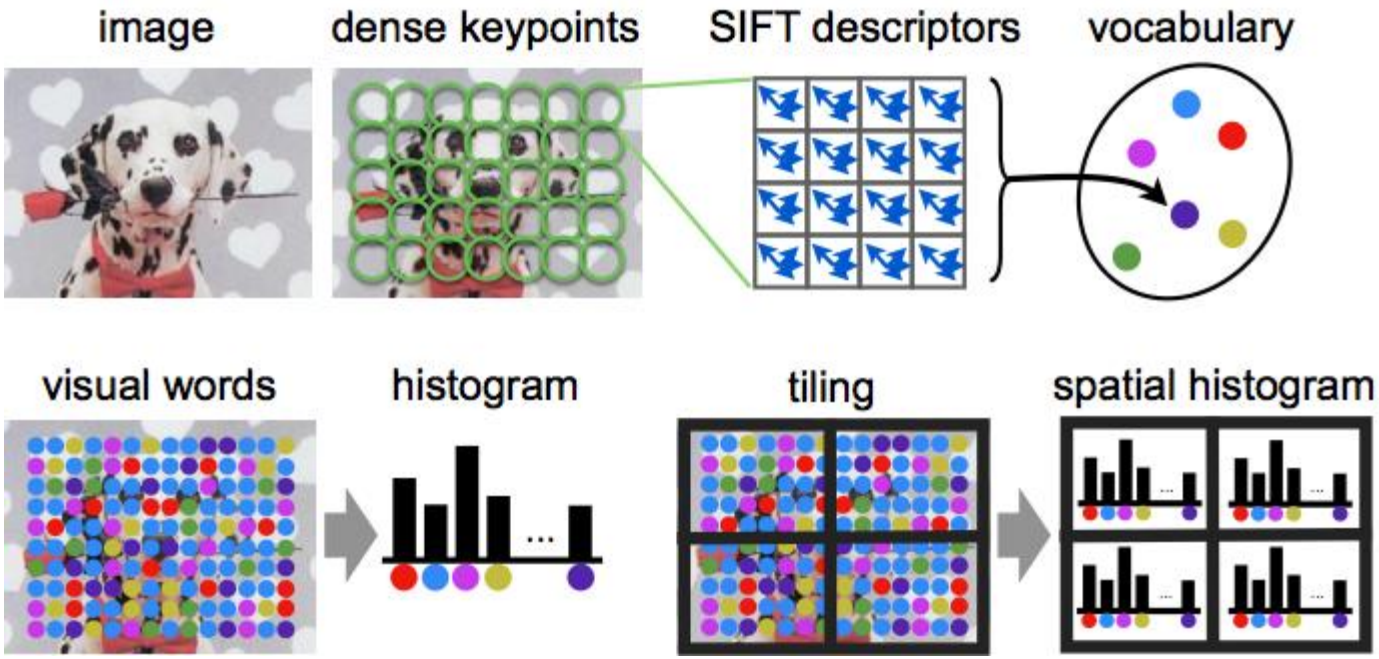


Slide creditF. Perroninn. Features for Large-Scale Visual Recognition

Everything together

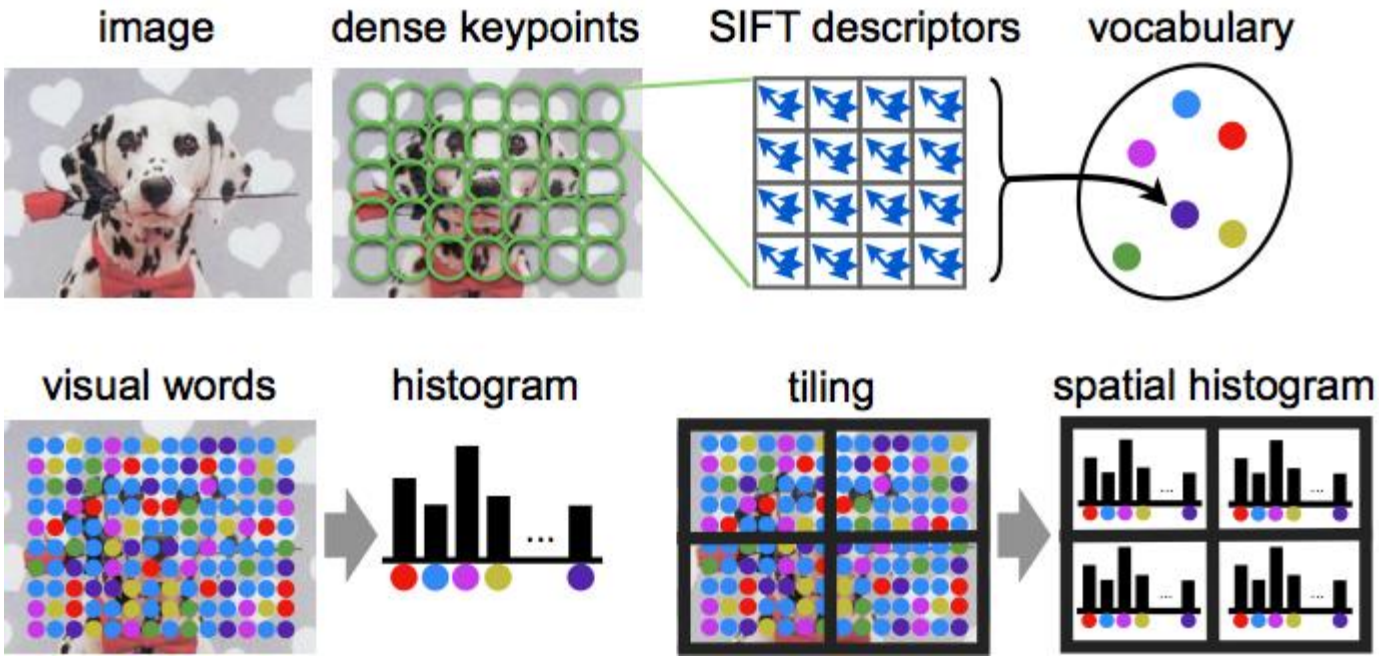


Everything together



- Image 512 x 512
- Dense SIFT extracted every 2 pixels, at 4 different scales
  - 256 x 256 x 4 = 262.144 descriptors per image…
- If we have 1M images in the dataset, how to compute the vocabulary?

Everything together



- k = 2048
- 1 level Spatial Pyramid
- BoVW dimension: 2048 x 5 = 10.240
- FV dimension: 2 x 128 x 2048 x 5 = 2.621.440
  2 x 128 x 32 x 5 = 40.960 (reducing k)
  2 x 64 x 32 x 5 = 20.480 (reducing SIFT dims)

Dimensionality Reduction

Goal: represent samples with fewer features

- Try to preserve as much **structure** in the data as possible

Feature selection

- Pick a subset of the original dimensions
- E.g. using information gain to decide which features to pick
- You are throwing out some of the features

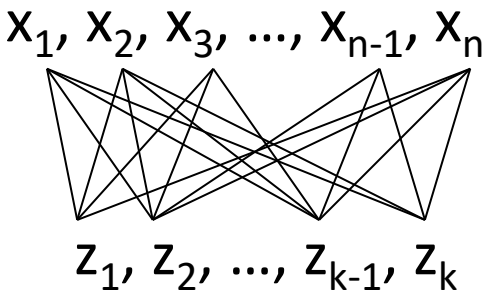
$x_1, x_2, x_3, \ldots, x_{n-1}, x_n$

⇩

$x_1, \mathbf{x_2}, x_3, \ldots, x_{n-1}, \mathbf{x_n}$

Feature extraction

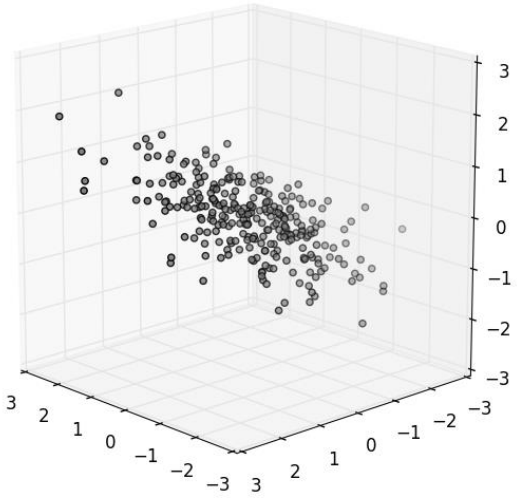
- Construct a new set of *k* features (with $k < n$) combining existing ones
- The i^{th} feature given by: $z_i = f(x_1, x_2, \ldots, x_n)$
- The easiest way is by linearly combining the original features

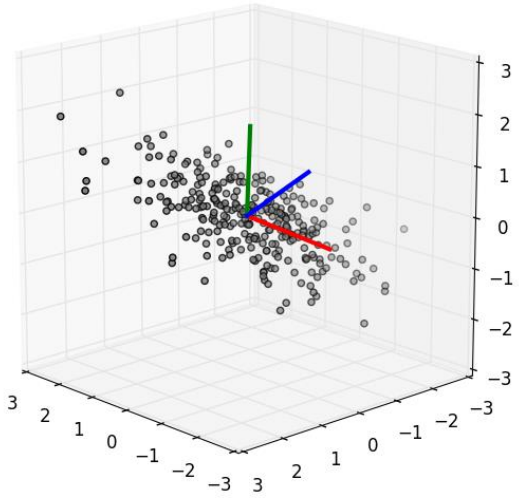


Principal Component Analysis

PCA defines a set of principal components (a new set of dimensions, a new set of features)

- 1st dimension: direction of the greatest variability in the data
- 2nd dimension: perpendicular to the 1st, greatest variability of what's left to explain
- 3rd dimension: perpendicular to all the previous ones, greatest variability of what's left to explain
- … and so on until n (the original dimensionality)





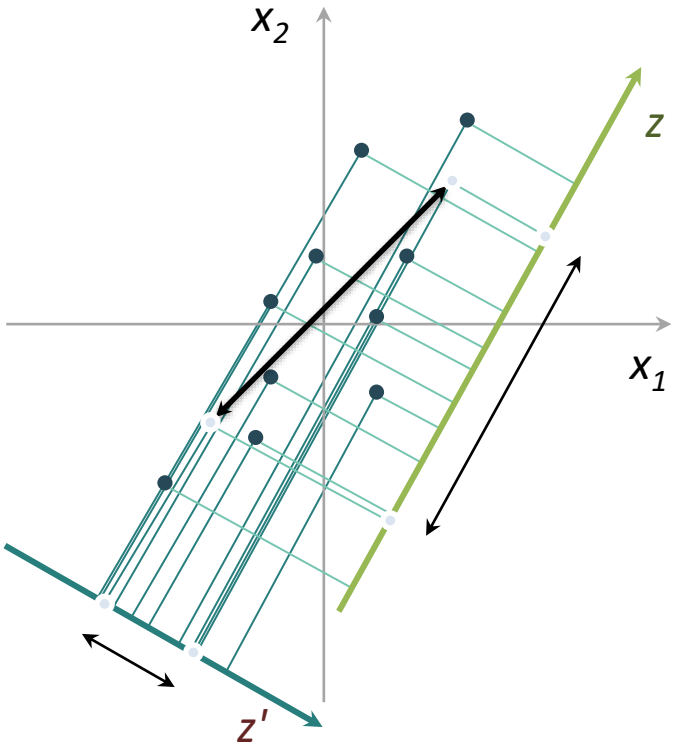
Why maximum variability

An example, reducing from $\mathbb{R}2$ to $\mathbb{R}1$

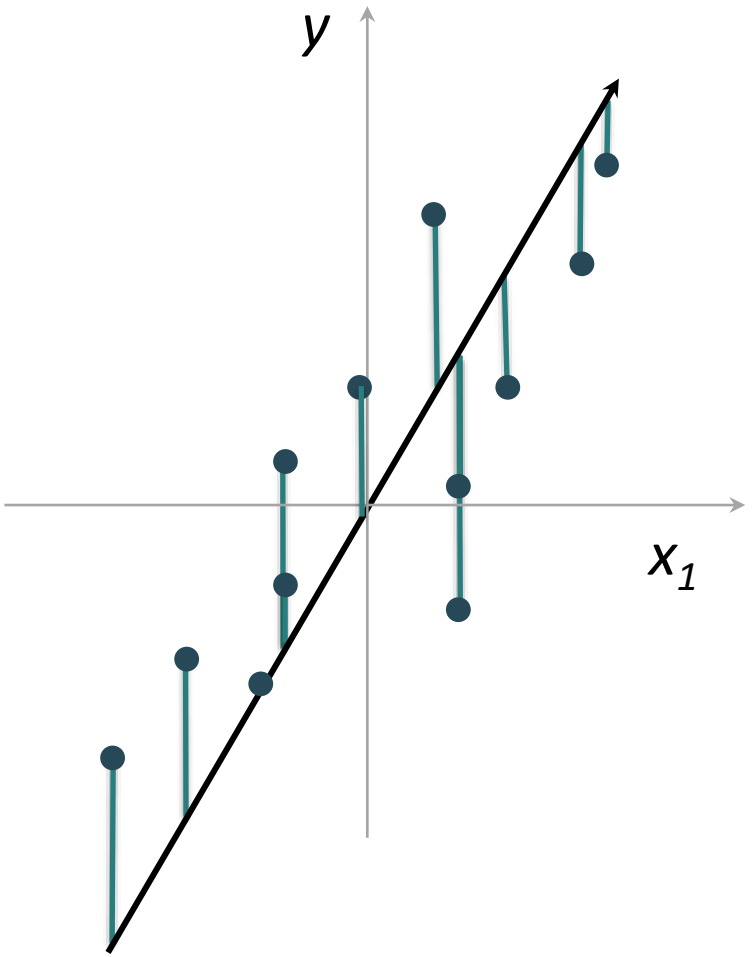
If we pick a direction *z* that maximises variability (green in the plot), it will maintain to a certain degree the relative distance between points in the original space:

If two points are far in the original (x_1, x_2)-space, they will most probably be far in the new (*z*)-space

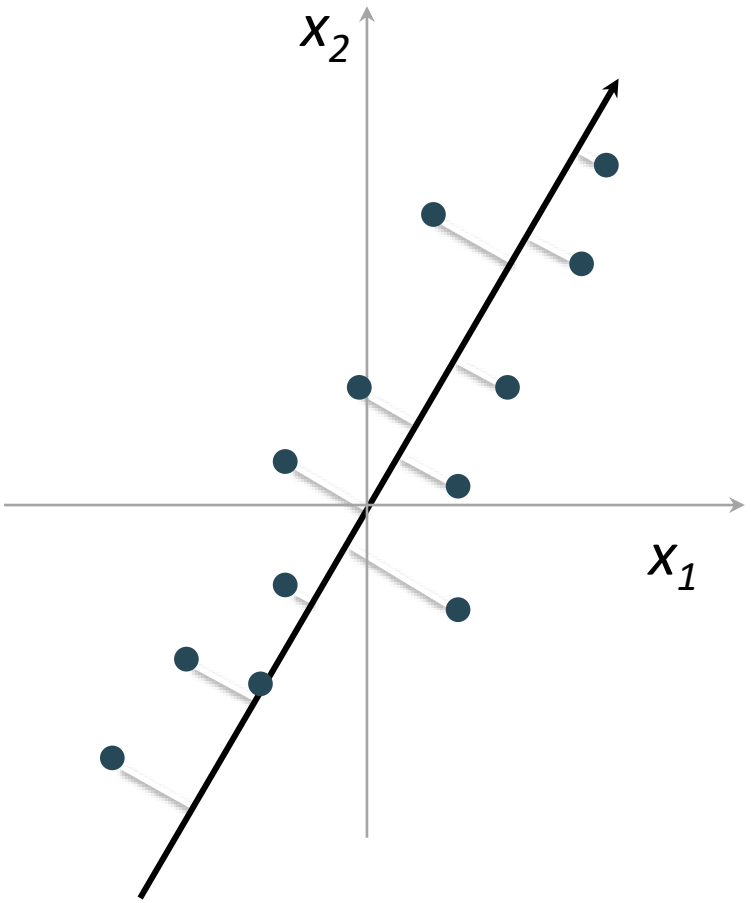
If we pick any other direction z' the relative distance between points in the original space is not preserved so well.



PCA is not linear regression



Linear regression



PCA

PCA algorithm summary

1. We start with correlated, high-dimensional data, $x \in \mathbb{R}^n$
2. Centre the points (optionally scale the features)
3. Computer the covariance matrix
4. Find the eigenvectors and the eigenvalues of the covariance matrix (e.g. using SVD)
5. Pick the $k \ll n$ eigenvectors with the highest eigenvalues
6. Project data points to the selected eigenvectors
7. Obtain uncorrelated low-dimensional data, $z \in \mathbb{R}^k$

PCA and classification

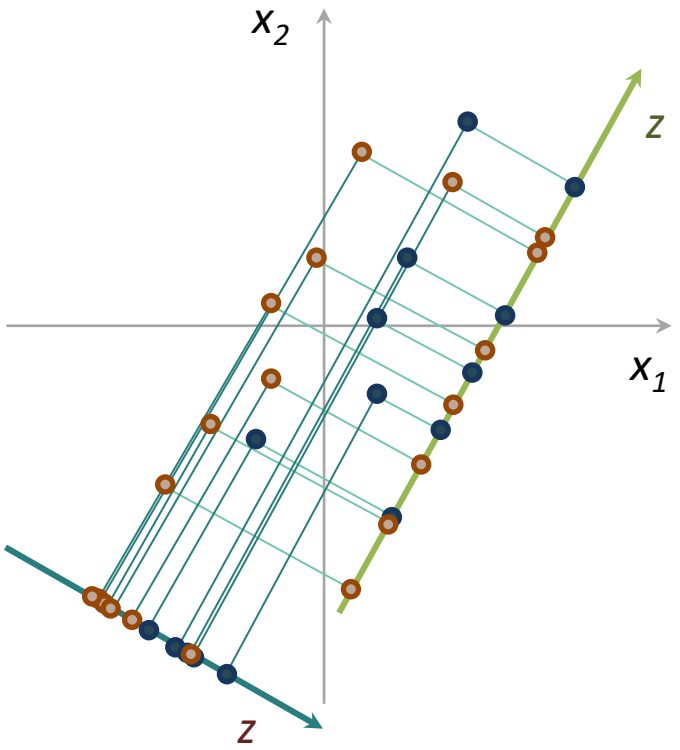
PCA can sometimes hurt instead of help, as it does not take into account the class labels

PCA is unsupervised

- Maximises overall variance along a small set of directions
- Does not know anything about class labels

Discriminative approach

- Look for a dimension that makes it easy to separate classes



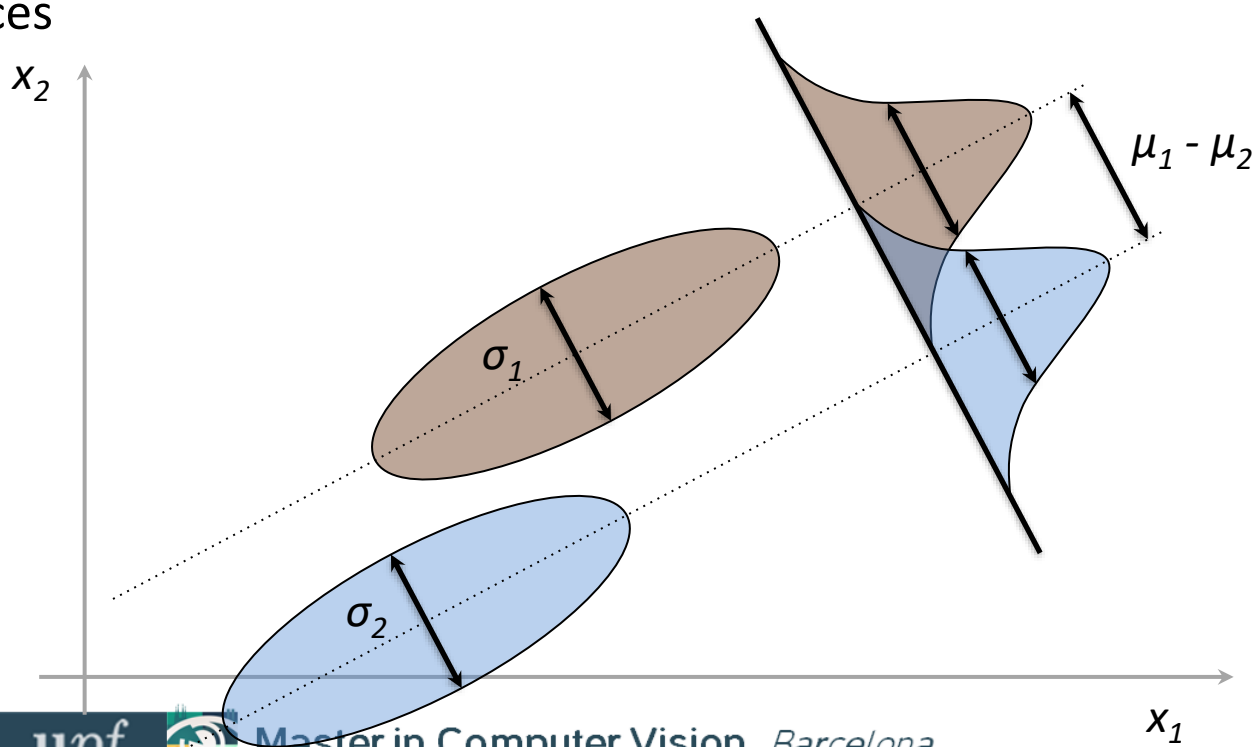
Linear Discriminant Analysis

LDA picks a new dimension that gives:

- Maximum separation between means of projected classes
- Minimum variance within each projected class

Solution: eigenvectors based on between-class and within-class covariance matrices

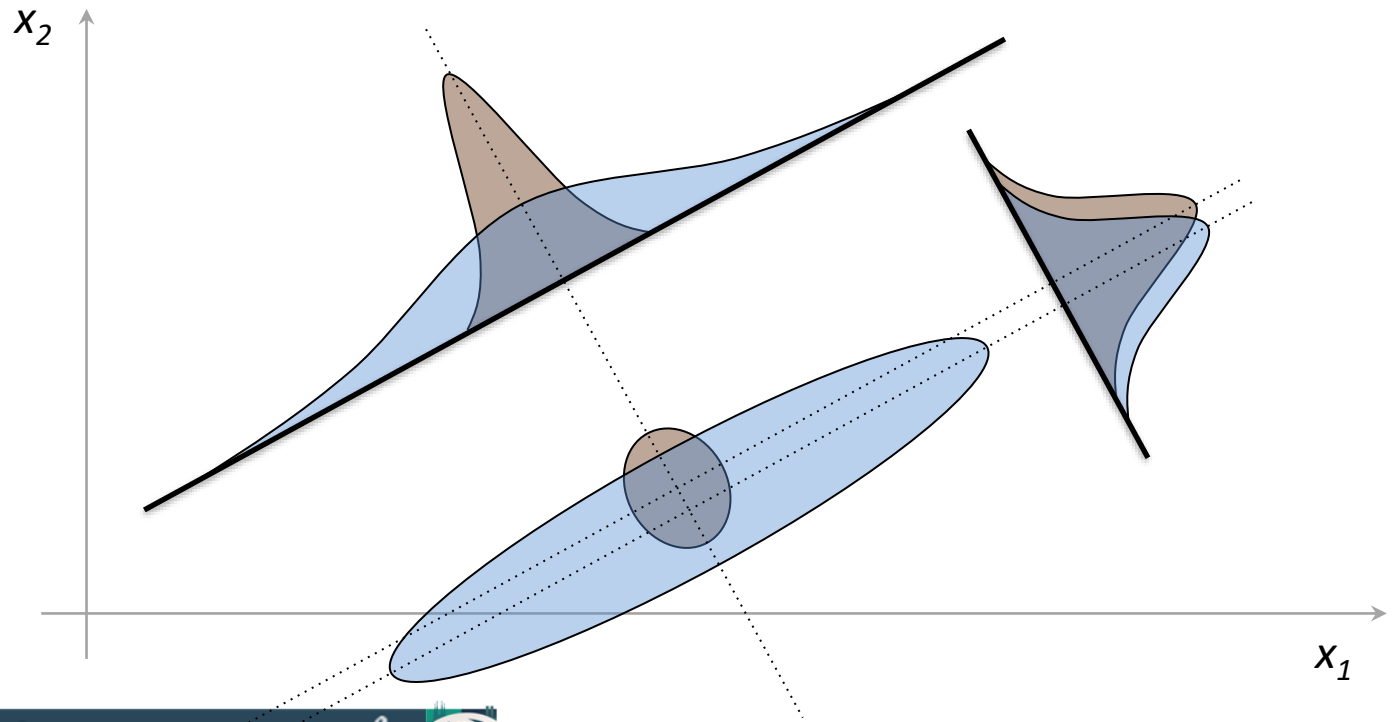
$$\max \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$$



Linear Discriminant Analysis

LDA is not guaranteed to be better for classification

- Assumes that distributions are unimodal Gaussians
- Assumes that they are separable
- Fails when the discriminatory information is not in the mean but in the variance of the data



References

- D. Lowe. *Object Recognition from Local Scale-Invariant Features*. ICCV 1999
- D. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*. IJCV 2004
- E. Nowak, F. Jurie, B. Triggs. *Sampling Strategies for Bag-of-Features Image Classification*. ECCV'06
- J. Sivic and A. Zisserman. *Video Google: A Text Retrieval Approach to Object Matching in Videos*. ICCV 2003.
- F. Perronnin, C. Dance, G. Csurka, M. Bressan. *Adapted Vocabularies for Generic Visual Categorization*, ECCV 2006.
- J. Vogel, B.Schiele. *Semantic Modeling of Natural Scenes for Content-Based Image Retrieval*. International Journal of Computer Vision, 2006.
- L. Fei-Fei and P. Perona. *A Bayesian hierarchical model for learning natural scene categories.* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2005.
- G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray. *Visual Categorization with Bags of Keypoints*. ECCV 2004

References

- J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. *Discovering object categories in image collections.* Technical Report A. I. Memo 2005-005, Massachusetts Institute of Technology, 2005.
- J. Savarese, A. Winn and T. Criminisi, *Object Categorization by Learned Universal Visual Dictionary.* CVPR 2006
- A. Vedaldi, A. Zisserman. *Efficient additive kernels via explicit feature maps.* PAMI 2011
- S. Lazebnik, C. Schmid, J. Ponce. *Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories*. CVPR 2006.
- K. Grauman, T. Darrell. *The Pyramid Match Kernel: Efficient Learning with Sets of Features*. Journal of Machine Learning Research, 2008
- N. Elfiky, F. Khan, J. Van de Weijer, J. González. *Discriminative Compact Pyramids for Object and Scene Categorization*. PR, 2011
- F. Perronin, J. Sánchez, T. Mensink. *Improving the Fisher Kernel for large-scale image classification*. ECCV 2010
- F. Perronin, C. Dance. *Fisher Kernels on Visual Vocabularies for Image Categorization.* CVPR 2007
- K. Chatfield, V. Lempitsky, A. Vedaldi, A. Zisserman. *The devil is in the details: an evaluation of recent feature encoding methods*. BMVC 2011

Toy Dataset

8 classes








Coast	Forest	Highway	Inside city	Mountain	Open country	Street	Tall building
244 train	227 train	184 train	214 train	260 train	292 train	212 train	248 train
116 test	101 test	76 test	94 test	114 test	118 test	80 test	108 test

To Do...

Improve the BoVW code with:

- Test different amounts of local features. What performs best?
- Use dense SIFT instead of detected keypoints. Conclusions?
- Test different amounts of codebook sizes *k.* What performs best?
- Test different values of *k* for the k-nn classifier. What performs best?
- Test other distances in k-nn classifier. Does that make a difference? Why?
- Play with reducing dimensionality. Conclusions?
- Cross-validate everything (topic covered on Wednesday)

Next session:

- SVM classifier.
- Linear, RBF and histogram intersection kernels.
- Spatial Pyramid.
- Fisher Vectors (OPTIONAL, check out yael library…)

Warning: provided code might not work out of the box depending on the used versions (OpenCV, numpy, sklearn…) do not panic, and read the documentation

Deliverable

- A **single Python notebook file per group** reporting all the work done,
  - with the different experiments,
  - code,
  - plots,
  - explanations, etc.
  - **EVERYTHING EXECUTED!**

- To deliver by Saturday, january, 8th @ 23:59 P.M. at Virtual Campus

Detector/Descriptor atlernatives

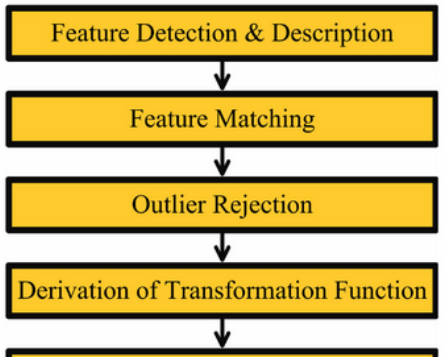
2018 International Conference on Computing, Mathematics and Engineering Technologies – iCoMET 2018

A Comparative Analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK

Shaharyar Ahmed Khan Tareen
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
sakts@live.com

Zahra Saleem
Fatima Jinnah Women University (FJWU)
Rawalpindi, Pakistan
zahse@outlook.com

***Abstract*—Image registration is the process of matching, aligning and overlaying two or more images of a scene, which are captured from different viewpoints. It is extensively used in numerous vision based applications. Image registration has five main stages: Feature Detection and Description; Feature Matching; Outlier Rejection; Derivation of Transformation Function; and Image Reconstruction. Timing and accuracy of feature-based Image Registration mainly depend on computational efficiency and robustness of the selected feature-detector-descriptor, respectively. Therefore, choice of feature-detector-descriptor is a critical decision in feature-matching applications. This article presents a comprehensive comparison of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK algorithms. It also elucidates a critical dilemma: Which algorithm is more invariant to scale, rotation and viewpoint changes? To investigate this problem, image matching has been performed with these features to match the scaled versions (5% to 500%), rotated versions (0° to 360°), and perspective-transformed versions of standard images with the original ones. Experiments have been conducted on diverse images taken from benchmark datasets:**



https://ieeexplore.ieee.org/abstract/document/8346440

- SIFT is found to be the most accurate feature-detector-descriptor for scale, rotation and affine variations (overall).
- BRISK is at second position with respect to the accuracy for scale and rotation changes.
- AKAZE's accuracy is comparable to BRISK for image rotations and scale variations in the range of **40% to 400%**. Beyond this range its accuracy decreases.
- ORB(1000) is less accurate than BRISK(1000) for scale and rotation changes but both are comparable for affine changes.
- ORB(1000) and BRISK(1000) are less accurate than ORB and BRISK (see Fig. 5 and Fig. 6). These algorithms have a trade-off between quantity of features, feature-matching cost and the required accuracy of results.
- AKAZE is more accurate than KAZE for scale and affine variations. However, for rotation changes KAZE is more accurate than AKAZE.

Hyperparamter optimization

Journal of Machine Learning Research 13 (2012) 281-305 Submitted 3/11; Revised 9/11; Published 2/12

Random Search for Hyper-Parameter Optimization

James Bergstra JAMES.BERGSTRA@UMONTREAL.CA
Yoshua Bengio YOSHUA.BENGIO@UMONTREAL.CA
Département d'Informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC, H3C 3J7, Canada

Editor: Leon Bottou

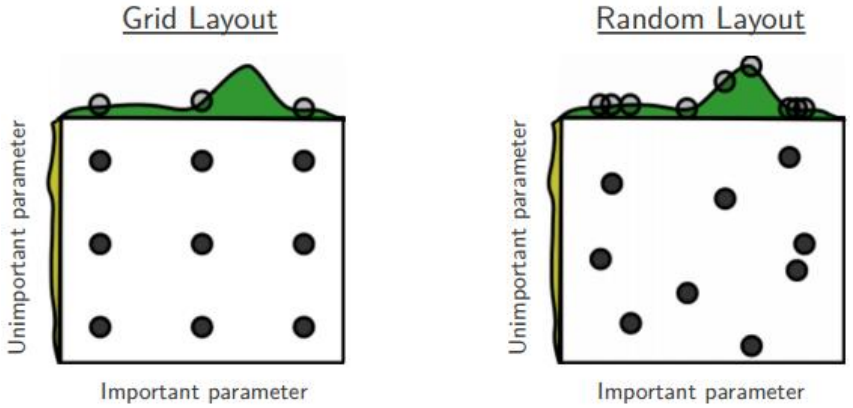
Abstract

Grid search and manual search are the most widely used strategies for hyper-parameter optimization. This paper shows empirically and theoretically that randomly chosen trials are more efficient for hyper-parameter optimization than trials on a grid. Empirical evidence comes from a comparison with a large previous study that used grid search and manual search to configure neural networks and deep belief networks. Compared with neural networks configured by a pure grid search, we find that random search over the same domain is able to find models that are as good or better within a small fraction of the computation time. Granting random search the same computational budget, random search finds better models by effectively searching a larger, less promising configuration space. Compared with deep belief networks configured by a thoughtful combination of manual search and grid search, purely random search over the same 32-dimensional configuration space found statistically equal performance on four of seven data sets, and superior performance on one of seven. A Gaussian process analysis of the function from hyper-parameters to validation set performance reveals that for most data sets only a few of the hyper-parameters really matter, but that different hyper-parameters are important on different data sets. This phenomenon makes grid search a poor choice for configuring algorithms for new data sets. Our analysis casts some light on why recent "High Throughput" methods achieve surprising success—they appear to search through a large number of hyper-parameters because most hyper-parameters do not matter much. We anticipate that growing interest in large hierarchical models will place an increasing burden on techniques for hyper-parameter optimization; this work shows that random search is a natural baseline against which to judge progress in the development of adaptive (sequential) hyper-parameter optimization algorithms.

Keywords: global optimization, model selection, neural networks, deep learning, response surface modeling

1. Introduction

The ultimate objective of a typical learning algorithm $\mathcal{A}$ is to find a function f that minimizes some expected loss $\mathcal{L}(x; f)$ over i.i.d. samples x from a natural (grand truth) distribution $\mathcal{G}_x$. A *learning algorithm* $\mathcal{A}$ is a functional that maps a data set $\mathcal{X}^{(\text{train})}$ (a finite set of samples from $\mathcal{G}_x$) to a function



Continuous hyperparameter: distribution over possible values

generate random variable

Discrete hyperparameter: list of discrete choices
random selection
(without replacement if all discrete)

Set the number of trials

Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of Machine Learning Research* 13.Feb (2012): 281-305.

```python
params = {
    "C": np.arange(0.001, 1, 0.01),
    "max_iter": np.arange(50, 550, 50)
}

lg = LogisticRegression(solver='liblinear')

lg_grid = GridSearchCV(lg, params, cv=8,
                       scoring=["accuracy", "f1_macro"], refit="accuracy", return_train_score=True)
lg_grid.fit(visual_words, train_labels)
```

Consider  https://optuna.org

An open source hyperparameter optimization framework to automate hyperparameter search