Recognition/classification

1. Introduction
2. Supervised learning
3. SVM classifiers
4. **Datasets and evaluation**

Datasets for learning/testing

- How to define a category ?
  - Bicycle
  - Paintings with women
  - Portraits

  …

  Concepts, semantics, ontologies …

Image/video datasets for training/testing

CalTech 101

TRECVID

- Choice of the categories (objects, concepts)
  - Number of categories
  - Number of images per category
  - Hierarchical structure ?
- Mono-label/multi-labels
- Pre-processing
  - Color, resolution, centered …

Example: ImageNet dataset

- Large Scale Visual Recognition Challenge (ILSVRC)
  - 1,2 Million images, 1000 classes

- Paper:
  - ImageNet: A Large-Scale Hierarchical Image Database, Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei, CVPR 2009

Classes of ImageNet

- Based on WordNet
  - Each node is depicted by images
- A knowledge ontology
  - Taxonomy
  - Partonomy

- Website:

IMAGENET

14,197,122 images, 2 841 synsets indexed

Explore Download Challenges Publications CoolStuff About

Not logged in. Login I Signup

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently we have an average of over five hundred images per node. We hope ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.

Click here to learn more about ImageNet, Click here to join the ImageNet mailing list.

Constructing ImageNet

- 2-step process

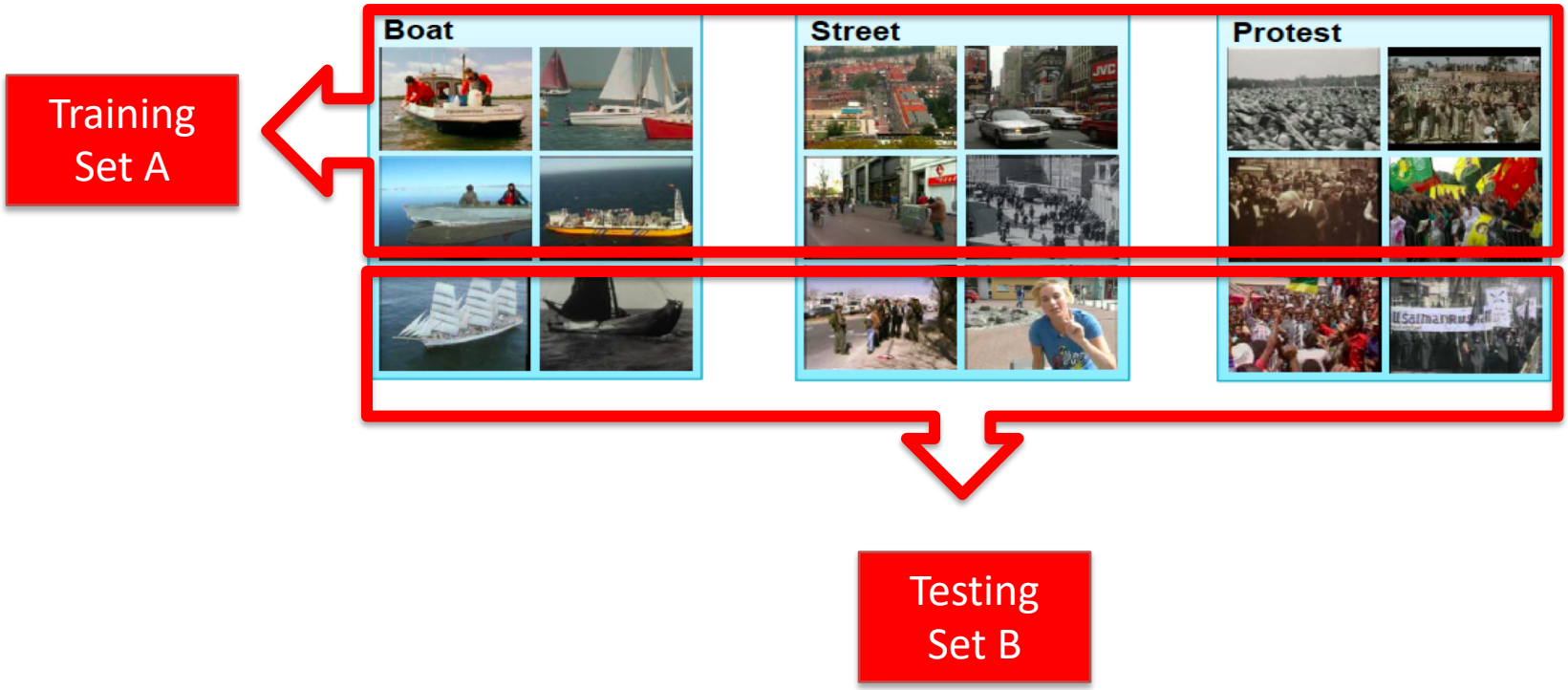
Step 1 :
Collect candidate
images Via the Internet

Step 2 :
Clean up candidate
Images by humans

Benchmarks and evaluation

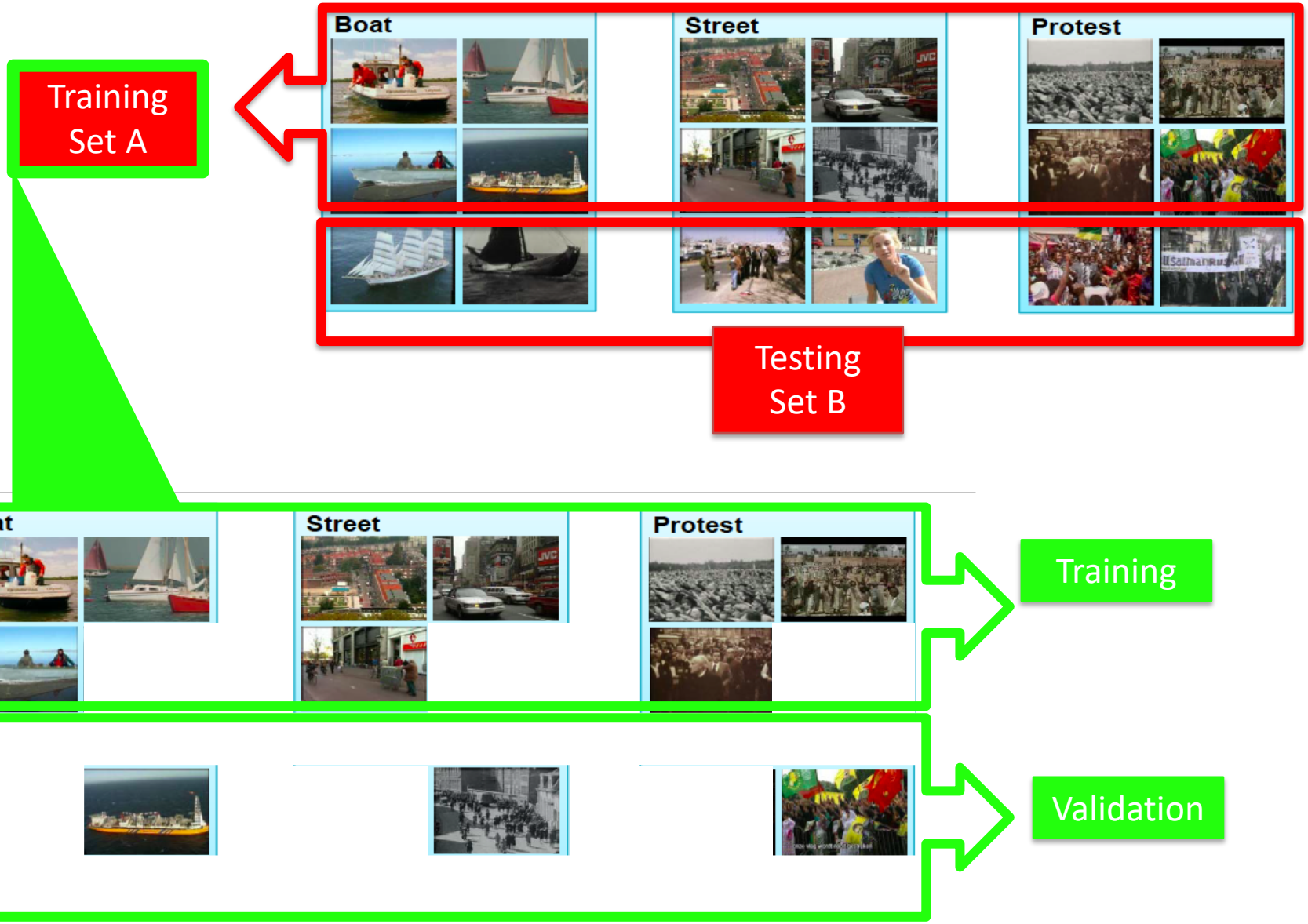
- Train / test / validation sets
  - Cross-validation
  - Learning hyper parameters
- Evaluation
  - Test Error
  - Accuracy, MAP, confusion matrix, Per-class averaging
  - Significance of the comparison, statistical tests, …
- Dataset building, concepts and semantics
  - Data pre-processing, data augmentation

Image/video datasets for training/testing



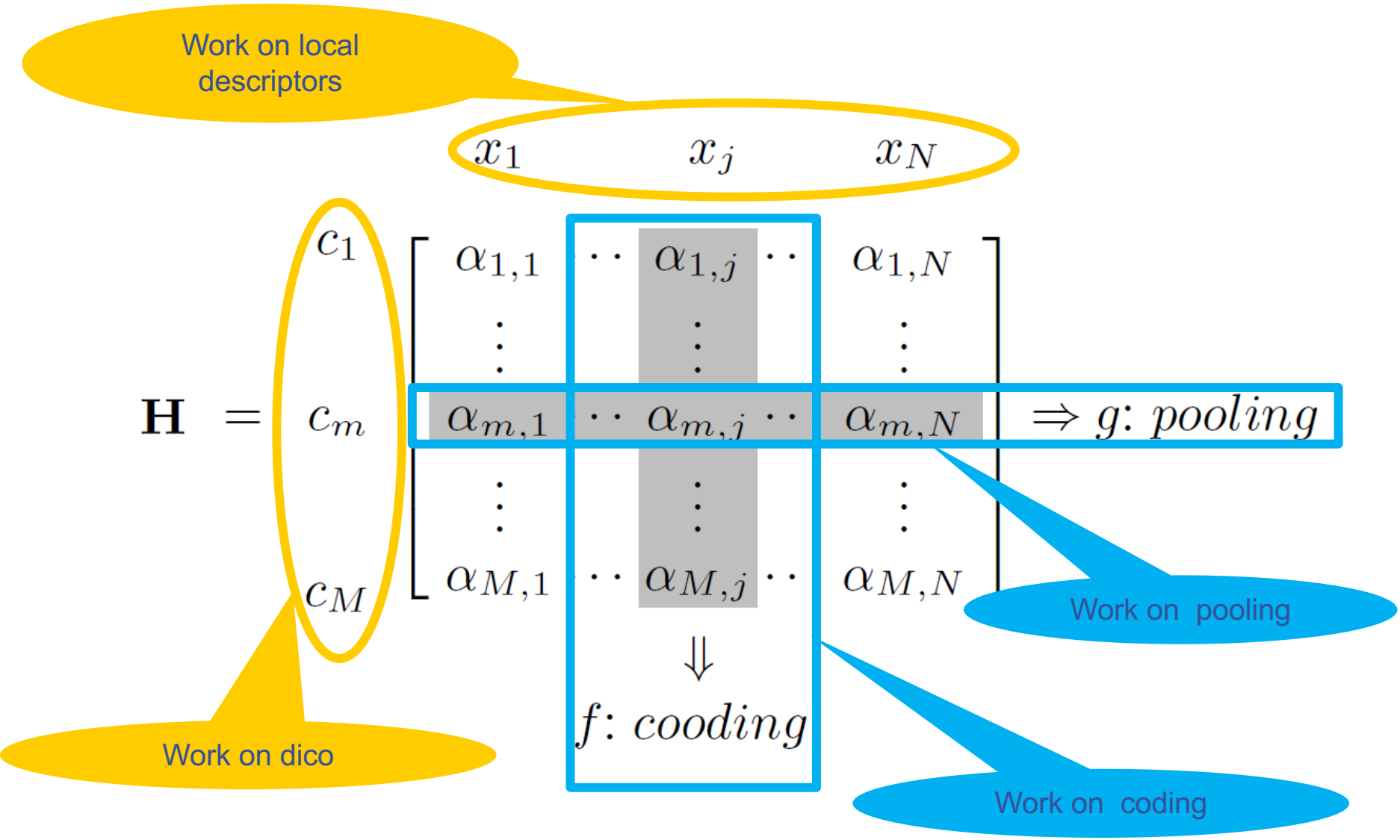
- Training classifiers on A
- Testing on B: error evaluation
- A and B disjoints!

Training: Cross-validation



Extra:

Beyond BoW representation

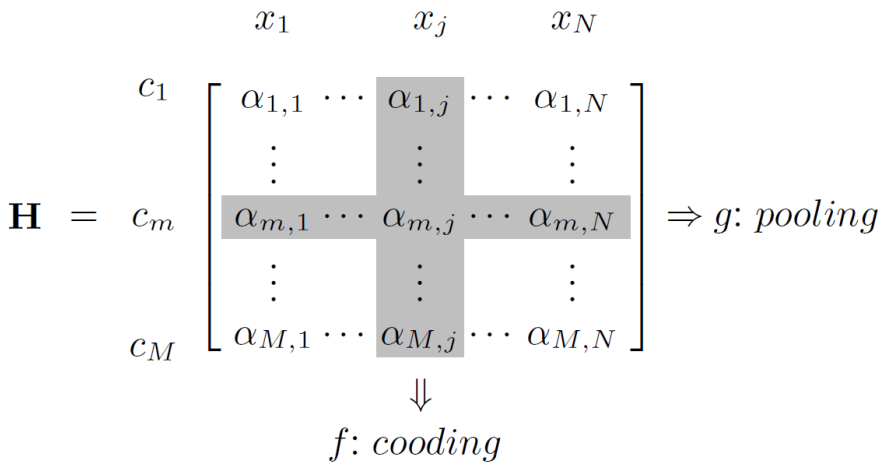

Work on local descriptors
x_1 x_j x_N
$\mathbf{H} =$
c_1
c_m
c_M
$\alpha_{1,1}$ $\alpha_{1,j}$ $\alpha_{1,N}$
$\alpha_{m,1}$ $\alpha_{m,j}$ $\alpha_{m,N}$
$\alpha_{M,1}$ $\alpha_{M,j}$ $\alpha_{M,N}$
$\Rightarrow g$: pooling
$\Downarrow$
f: cooding
Work on dico
Work on pooling
Work on coding

Pooling: Aggregating projections => global image index

Sum pooling alternative:

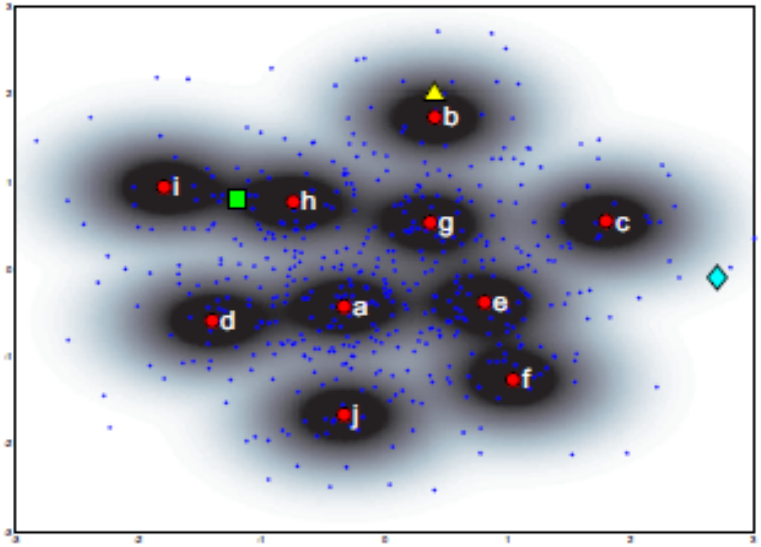
- **Max pooling** : keep the max value for the projection for each visual word
  - Relevant for sparse / soft coding: limit noise effect
  - (Partially) Justify by bio-inspired models (cortex)

$$z_m = g(\alpha_m) = \max_{j=1..N} \alpha_{m,j}$$



Coding: Projection =>dictionary

- **soft assignment**
  - Kernel codebook : absolute weight
  - Uncertainty: relative weight
  - Plausibility: absolute weight to 1-nn



Visual Word Ambiguity

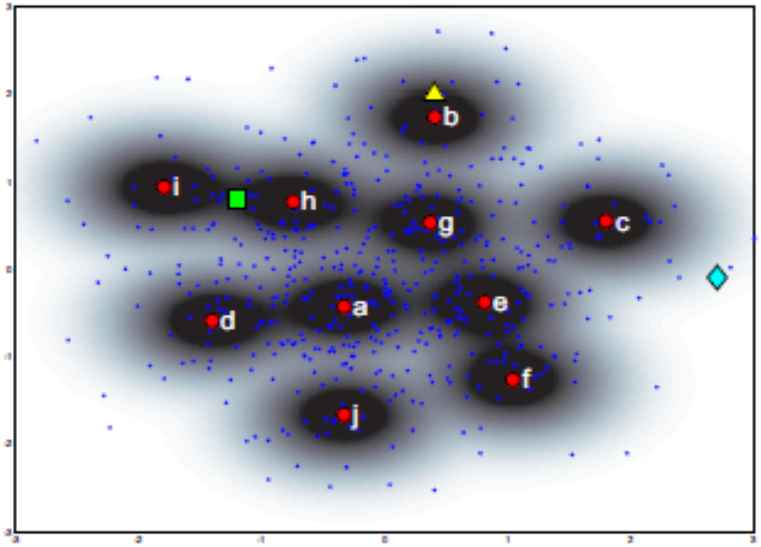
J.C. van Gemert, C.J. Veenman, A.W.M. Smeulders, J.M. Geusebroek

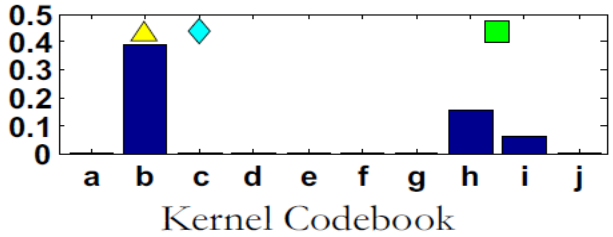
PAMI 2010

Soft Coding :kernel

$$f_{Kernel}(x_j)[m] = K(d(x_j, c_m))$$

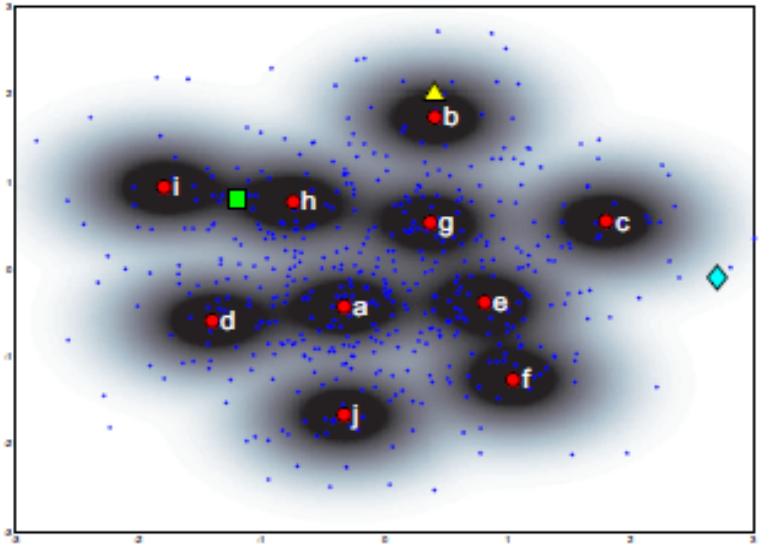
Ex: K(x)=exp(-ax)

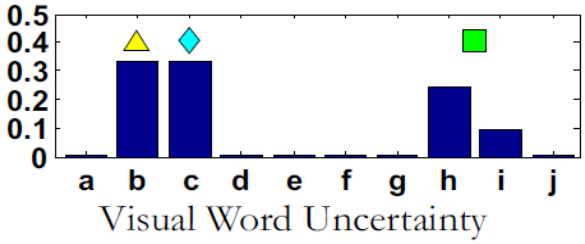




Soft Coding : uncertainty

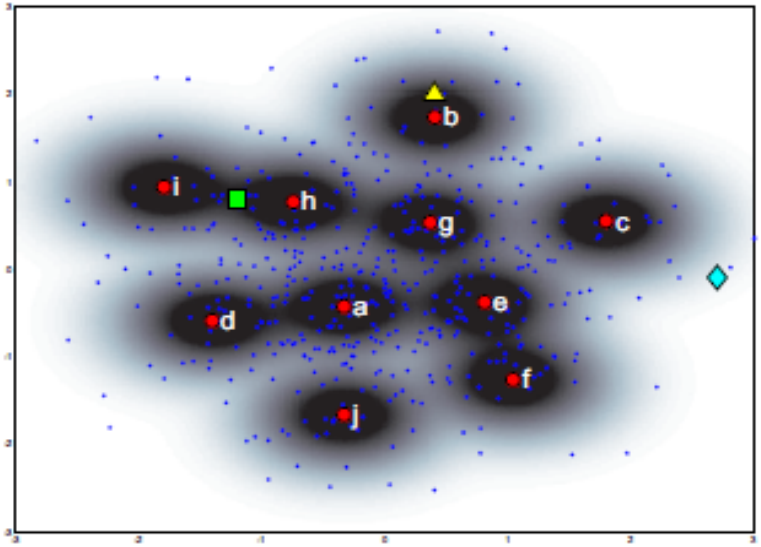
$$f_{Unc}(x_j)[m] = \frac{K(d(x_j, c_m))}{\sum_{k=1}^{M} K(d(x_j, c_k))}$$

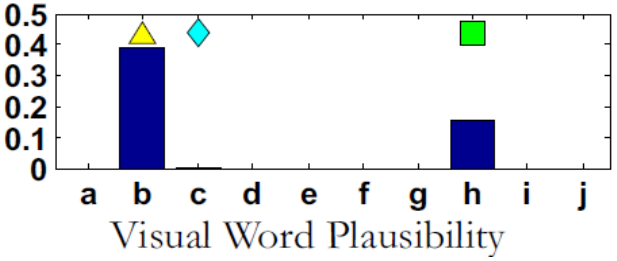




Soft Coding : plausibility

$$f_{Plau}(x_j)[m] = \begin{cases} K(d(x_j, c_m)) & \text{if } m = \underset{k \in \{1;M\}}{\operatorname{argmin}} \|x_j - c_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

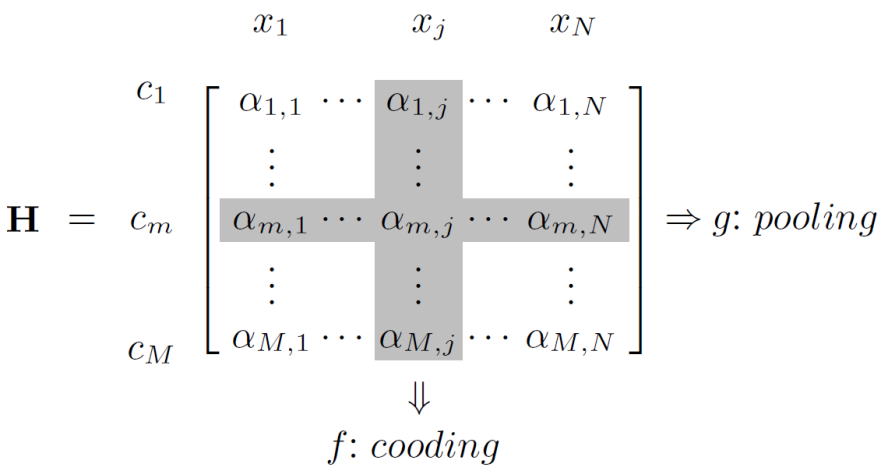




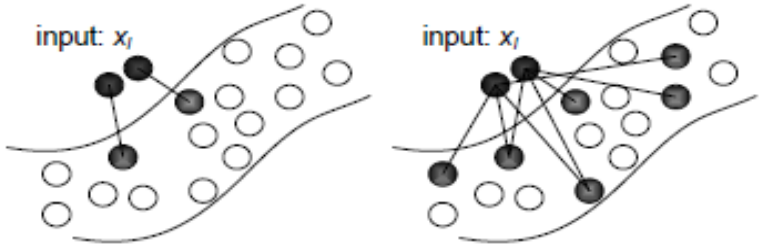
Soft Coding

- Soft vs hard assignement/coding
  - Not a so big gain soft / hard
  - Uncertainty certainly the best strategy

- Semi-soft : excellent tradeoff



Sparse Coding



- Other approach: **sparse coding**
  - Approximation of each local feature x_i (SIFT) as a lin. combination of a subset of words from the dictionary: $x_i \sim C\alpha_i$
    - α_i weight vectors, C matrix of vectors of the dictionary

$$\alpha_i = \underset{\alpha}{argmin} \quad L(\alpha, C) \triangleq \|x_i - C\alpha\|_2^2$$

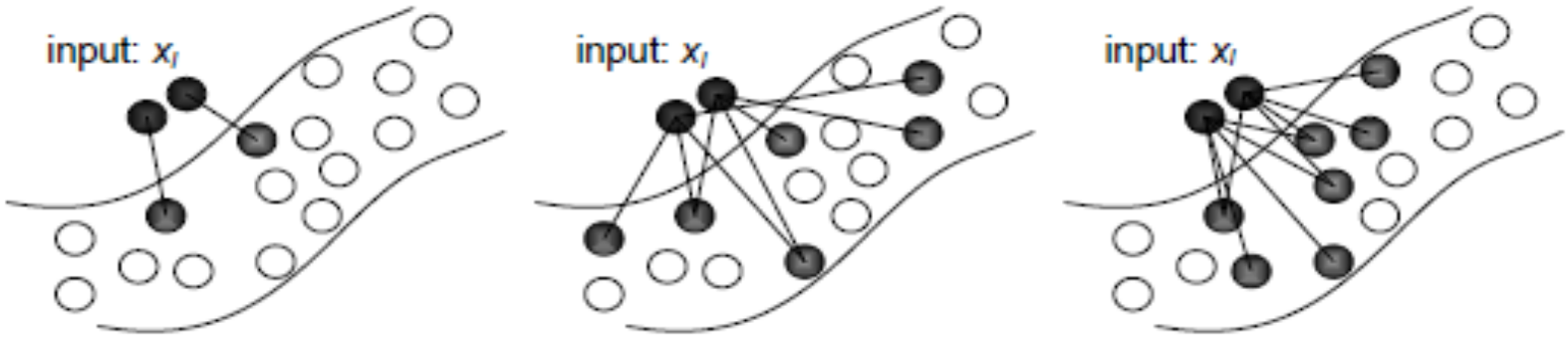
    - Pb: not sparse, many irrelevant values in M
    - Each x_i should be represented using only a small nb of visual words => sparsity

  - Sparse but no locality

$$\alpha_i = \underset{\alpha}{argmin} \quad L(\alpha, C) \triangleq \|x_i - C\alpha\|_2^2 + \lambda\|\alpha\|_1$$

Sparse Coding

- Sparse coding vs VQ (hard assignment)
  - VQ: hard coding
  - SC : Sparse Coding : most of α_i=0
  - LLC : Local Linear Coding : words representing the feature must be close (locality)



VQ SC LLC

- Are these criteria minimizing reconstruction error relevant for image classification purpose?

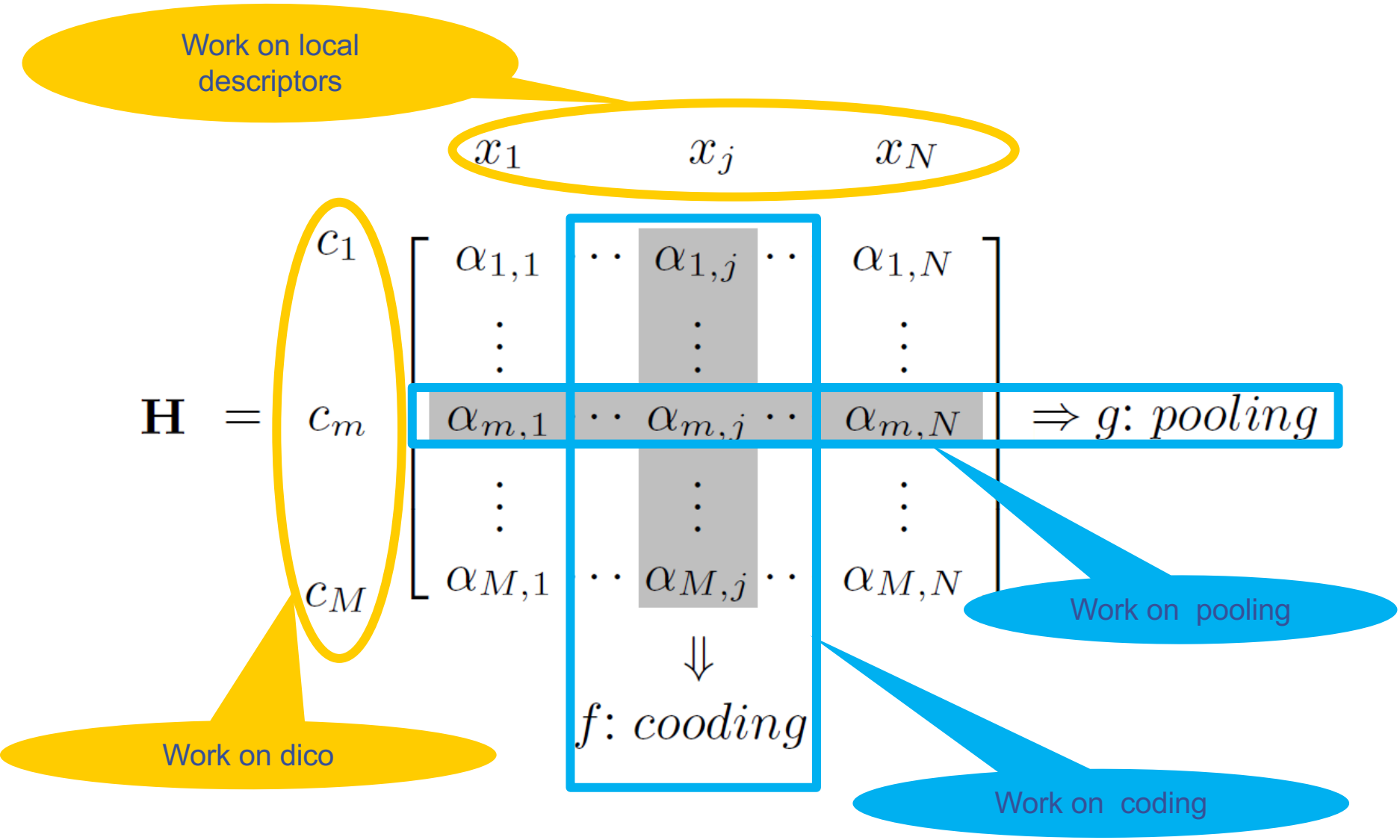
Aggregating projections => global image index

Where we are:

- Better represent coding/pooling association: work on the whole matrix of clusters/descriptors dependency => combine spatial pooling and sparse coding

Next:

- Work on new descriptors (bio inspired, learned)
- Dictionaries
  - Train the dico (supervised training)
  - Avoiding dico/clustering
    - Kernel similarity on bag of local features
- Exploit spatial image information


Work on local descriptors
x_1 x_j x_N
$\mathbf{H} =$
c_1
c_m
c_M
$\alpha_{1,1}$ $\alpha_{1,j}$ $\alpha_{1,N}$
$\alpha_{m,1}$ $\alpha_{m,j}$ $\alpha_{m,N}$
$\alpha_{M,1}$ $\alpha_{M,j}$ $\alpha_{M,N}$
$\Rightarrow g$: pooling
$\Downarrow$
f: cooding
Work on dico
Work on pooling
Work on coding

Invariance/robustness in BoW pipeline

Stability of the representation:

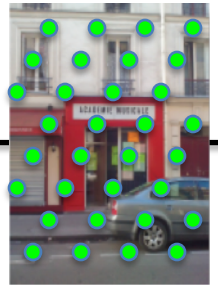
- Small deformations/transformations in the input space => similar representations
- Large (or unexpected) transformations in the input space => very dissimilar representations

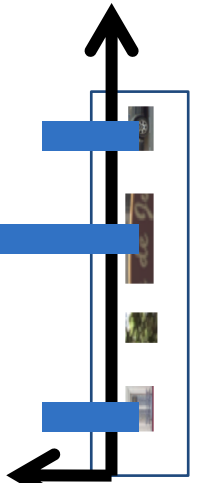
Invariance/robustness in BoW pipeline

Detection

Descriptor

Coding/pooling





Translation, flip, rotation, scale changes

Depends on detectors

Regular dense (multi-scale) Grid

Robust to translation, scale

Depends on descriptors

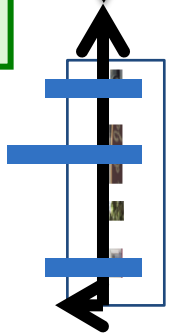
Sift : invariance to local rotation, to radiometric change

Depends on dico (=> kmeans)

Robust to noise

Pooling: Invariant to any shift, permutation in the bag

= ?



Beyond BoW

- **SPM: Spatial Pyramid (Lazebnik et al)**

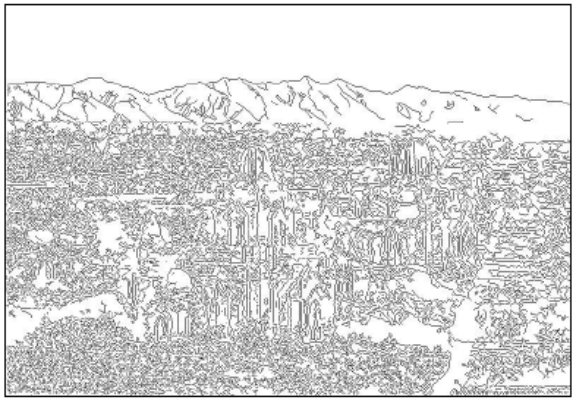
  Geometry in BoW: Pyramid in image space

- Pyramid Match Kernel (Grauman et al)

  Pyramid in feature space: Kernel similarity

SPM Algorithm

1. Extract interest point descriptors (dense scan)
2. Construct visual word dictionary
3. Build spatial histograms
4. Train an SVM



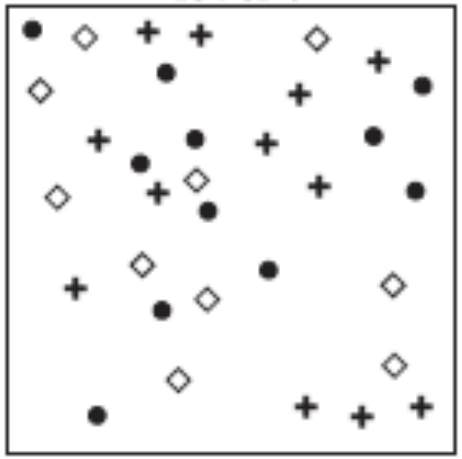
Weak (edge orientations)

OR

Strong (SIFT)

Algorithm

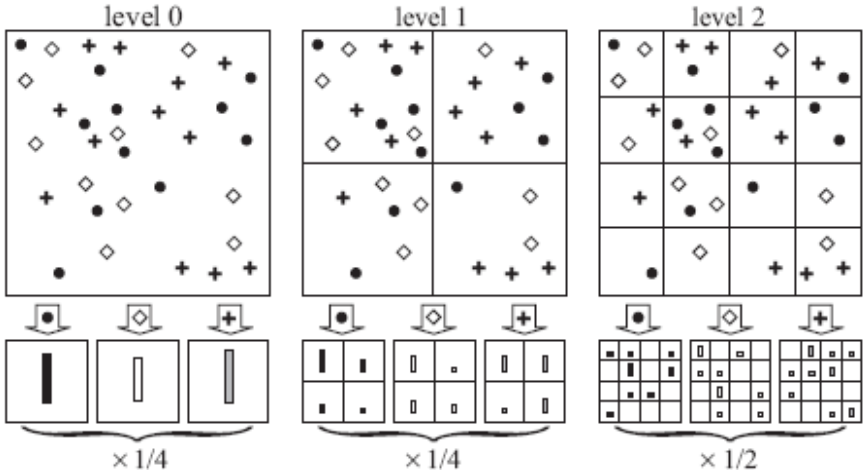
1. Extract interest point descriptors (dense scan)
2. Construct visual word dictionary
3. Build spatial histograms
4. Train an SVM



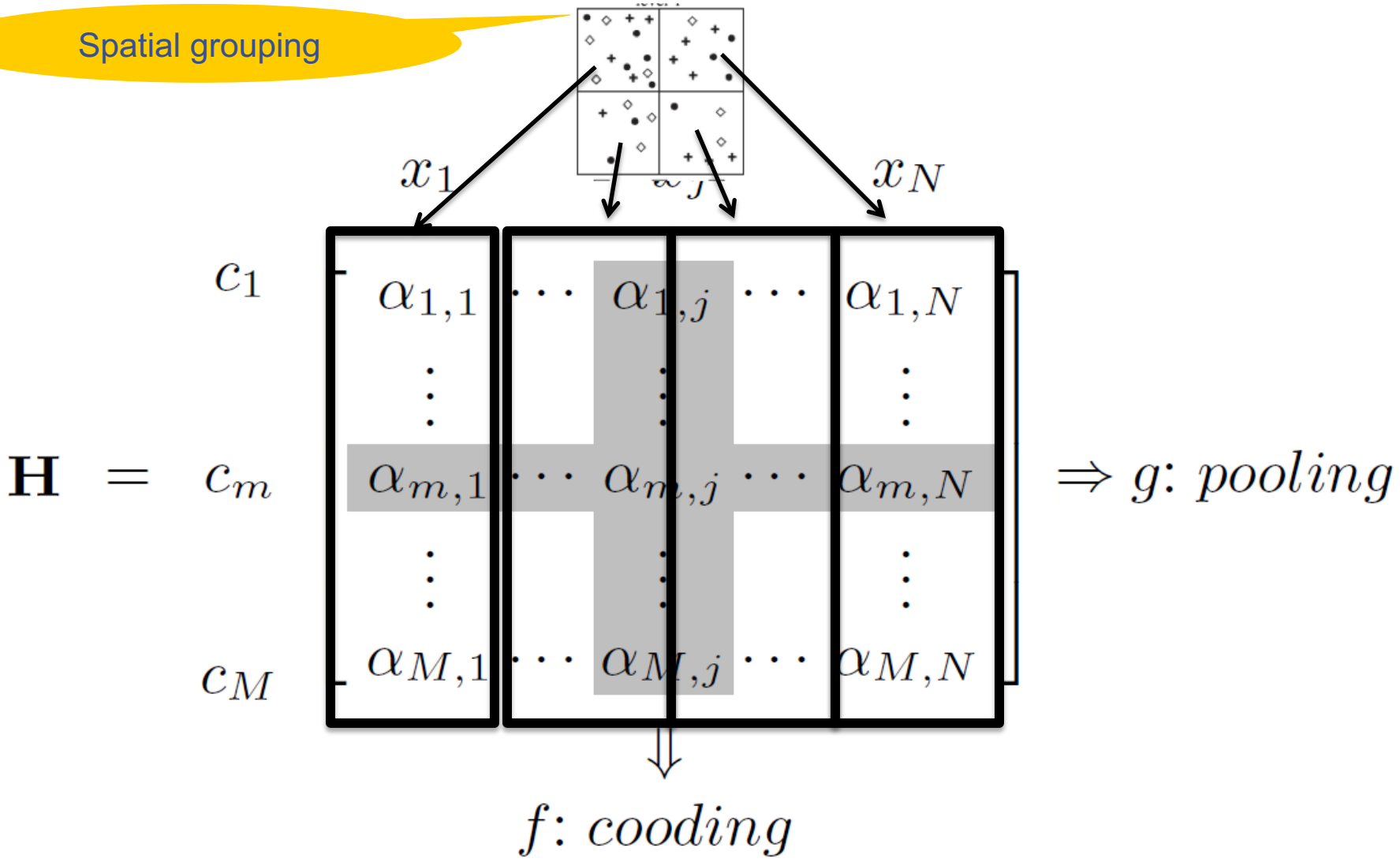
- Vector quantization
- Usually K-means clustering
- Vocabulary size (16 to 400)

Algorithm

1. Extract interest point descriptors (dense scan)
2. Construct visual word dictionary
3. Build spatial histograms
4. Train an SVM (with specific kernels)



Spatial grouping



=> Break global invariance because of fixed pyramid

- ***Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, Lazebnik et al, CVPR 06***

  Pyramid in image space, quantize features

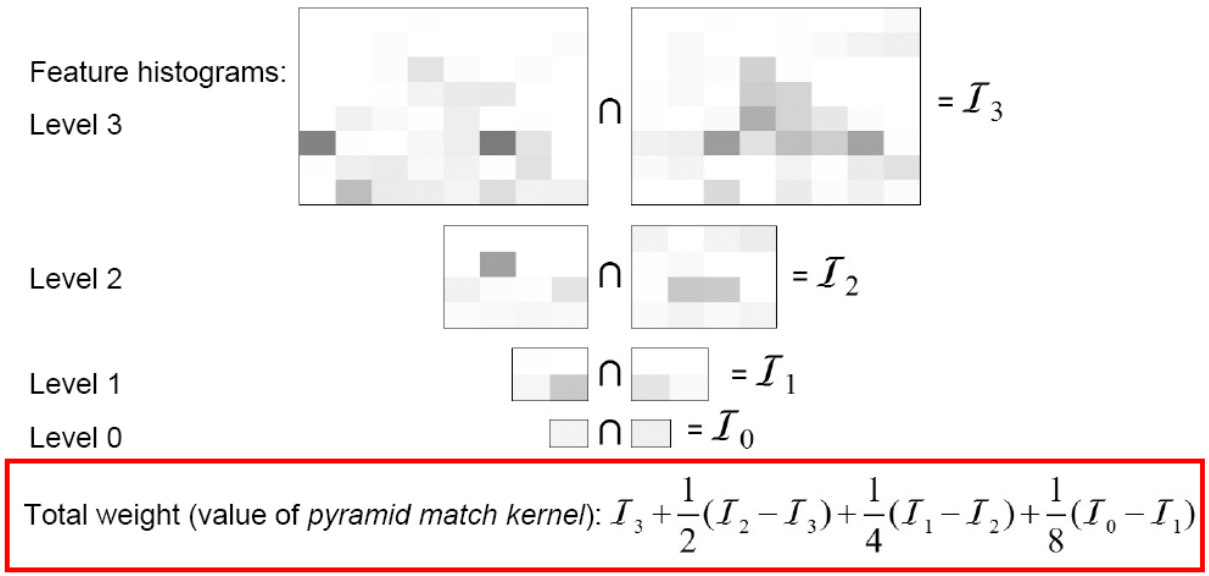
  $\Rightarrow$**Limit the global invariance:**

  S( , ) small

Algorithm

1. Extract interest point descriptors (dense scan)
2. Construct visual word dictionary
3. Build spatial histograms
4. Train an SVM

Similarity

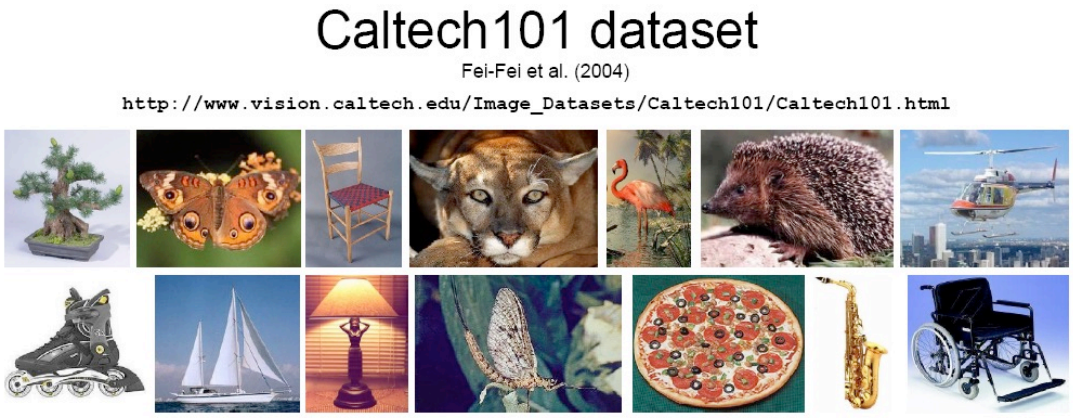


Algorithm

1. Extract interest point descriptors (dense scan)
2. Construct visual word dictionary
3. Build spatial histograms
4. **Train an SVM**

… Based on the kernel Similarity PMK

SPM Article: Results



- 3 Datasets
  - Nb images
  - Nb classes
- SVM multiclass !?!
- Eval protocol:
  - Train/test/val
    - 10 folds => average+standard deviation
    - Average per class
  - Nb of images per class in train (from 5 to 30)
- Parameter optimization
- Comparison to others

p4 Expe from SPM Article

5. Experiments

In this section, we report results on three diverse datasets: fifteen scene categories [4], Caltech-101 [3], and Graz [14]. We perform all processing in grayscale, even when color images are available. All experiments are repeated ten times with different randomly selected training and test images, and the average of per-class recognition rates[2] is recorded for each run. The final result is reported as the mean and standard deviation of the results from the individual runs. Multi-class classification is done with a support vector machine (SVM) trained using the one-versus-all rule: a classifier is learned to separate each class from the rest, and a test image is assigned the label of the classifier with the highest response.

[2]The alternative performance measure, the percentage of all test images classified correctly, can be biased if test set sizes for different classes vary significantly. This is especially true of the Caltech-101 dataset, where some of the "easiest" classes are disproportionately large.

p4 Expe from Gemert's Article

A. Experimental Setup

To obtain reliable results, we repeat the experimental process 10 times. We select 10 random subsets from the data to create 10 pairs of train and test data. For each of these pairs we create a codeword vocabulary on the train set. The exact same codeword vocabulary is used by both the codebook and the codeword ambiguity approaches to describe the train and the test set. For classification, we use an SVM with a histogram intersection kernel. Specifically, we use libSVM, and use the built in one-versus-one approach for multi-class classification. We use 10-fold cross-validation on the train set to tune parameters of the SVM and the size K_σ of the codebook kernel. The classification rate we report is the average of the per-category recognition rates which in turn are averaged over the 10 random test sets.

For image features we follow Lazebnik *et al.* [14], and use a

Multi-class SVM

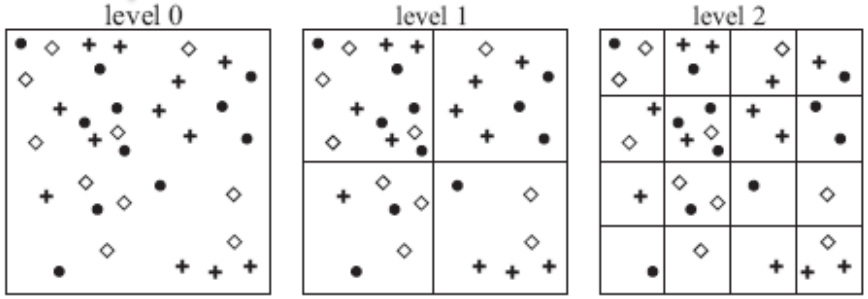
… By combining multiple two-class SVMs!

- One vs. All
  - Training: learn an SVM for each class vs. all others grouped in 1 class
  - 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

SPM Article: Results on Caltech101

Multi-class classification results (30 training images per class)

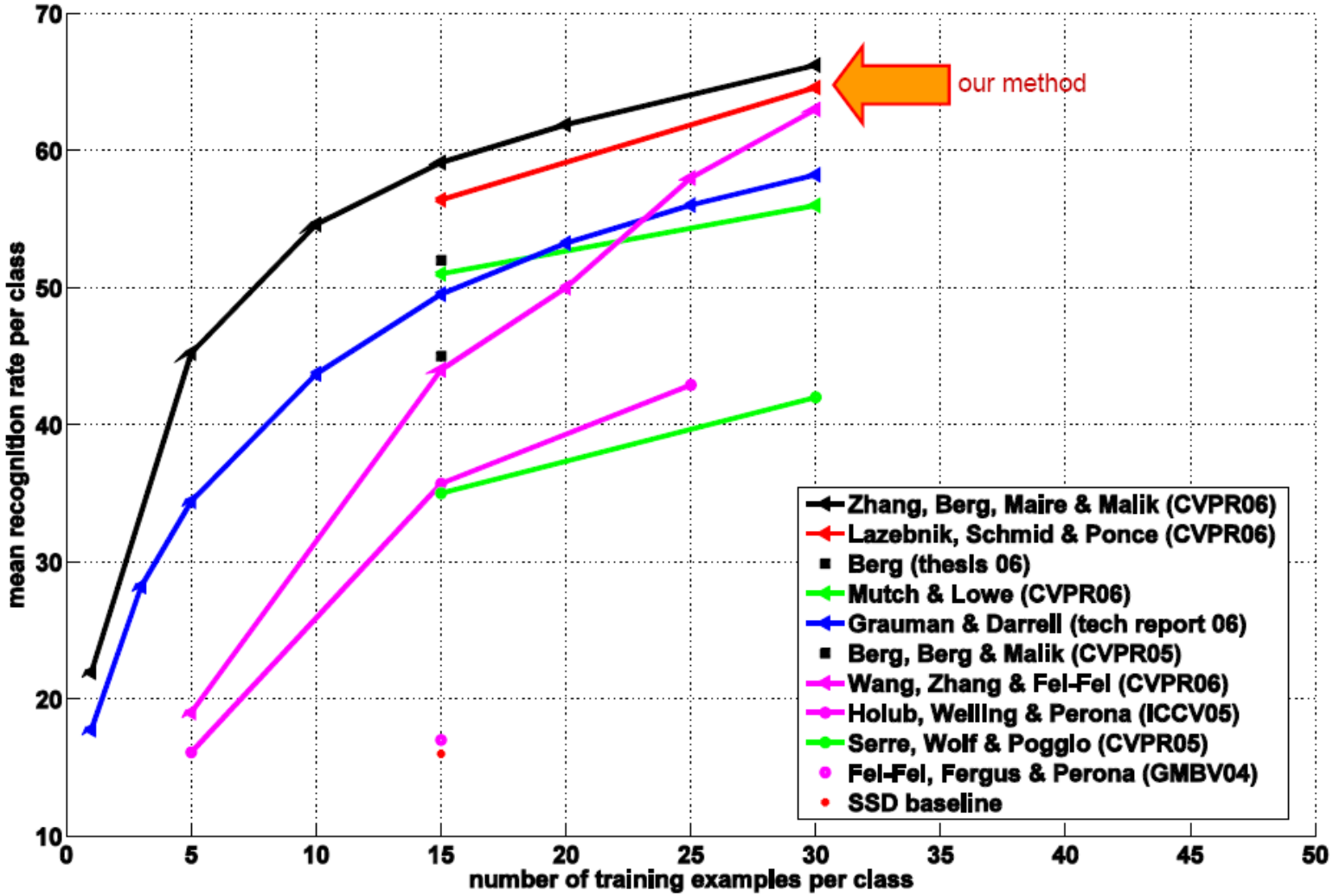
	Weak features (16)		Strong features (200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7



SPM Article

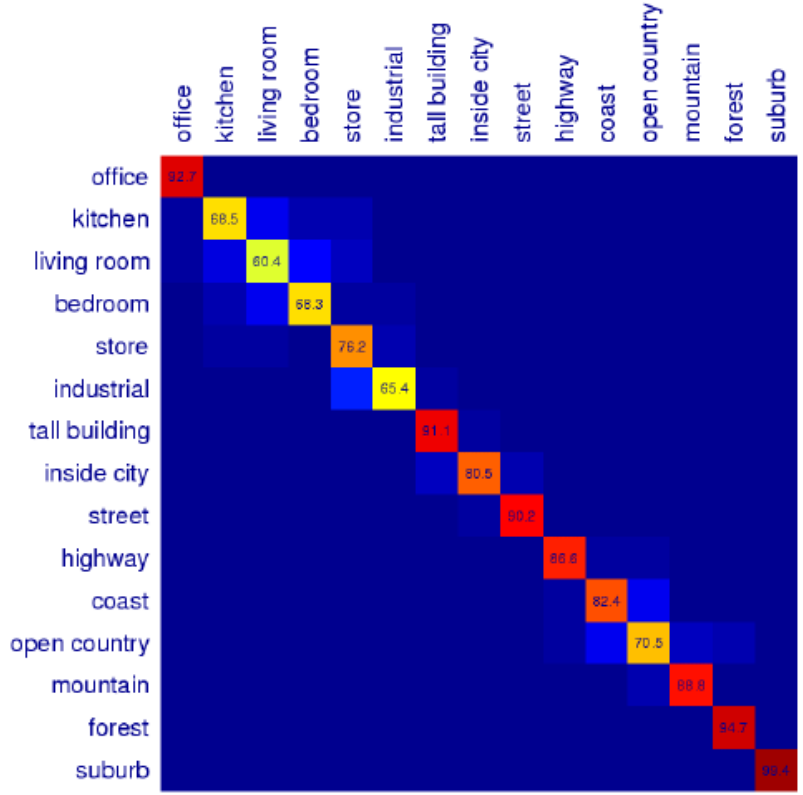
Caltech101 comparison

Zhang, Berg, Maire & Malik, 2006



SPM Article

Scene category confusions



Difficult indoor images

kitchen

living room

bedroom

SPM Article

Caltech101 challenges

Top five confusions

class 1 / class 2	class 1 mis-classified as class 2	class 2 mis-classified as class 1
ketch / schooner	21.6	14.8
lotus / water lily	15.3	20.0
crocodile / crocodile head	10.5	10.0
crayfish / lobster	11.3	9.1
flamingo / ibis	9.5	10.4

Easiest and hardest classes


minaret (97.6%)


windsor chair (94.6%)


joshua tree (87.9%)


okapi (87.8%)


cougar body (27.6%)


beaver (27.5%)


crocodile (25.0%)


ant (25.0%)

- Sources of difficulty: lack of texture, camouflage, “thin” objects, highly deformable shape

SPM Article

PMK/SIFT Best Categories (1-5)

100%

100%

99.7%

99.1%

98.2%

SPM Article

PMK/SIFT Best Categories (6-10)

97.7%

97.4%

95.7%

95.3%

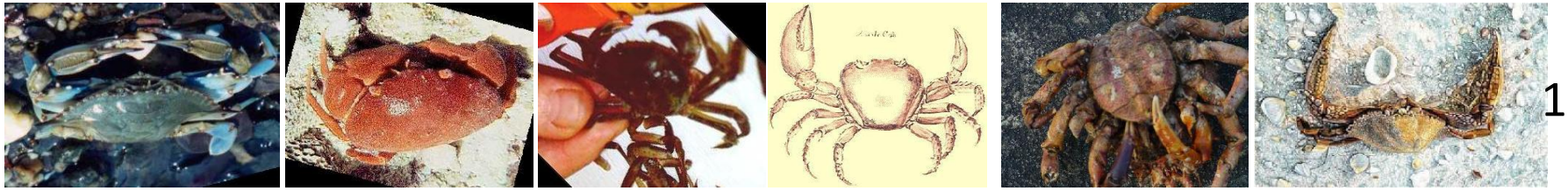
95.2%

SPM Article

PMK/SIFT 5 Worst Categories

7.7%

11.2%



11.5%

11.8%

12.3%

SPM Article

PMK/SIFT Most Confused Category Pairs







schooner

A fore-and-aft rigged sailing vessel having at least two masts, with a foremast that is usually smaller than the other masts.



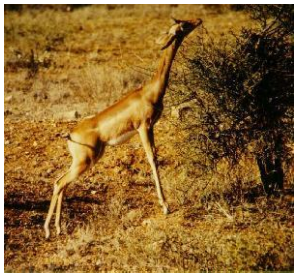


ketch

A two-masted fore-and-aft-rigged sailing vessel with a mizzenmast stepped aft of a taller mainmast but forward of the rudder.

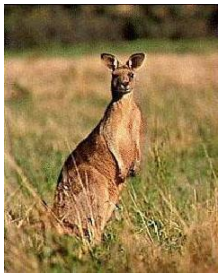
SPM Article

PMK/SIFT Most Confused Category Pairs






Gerenuk (**antilope girafe** ou **gérénuk)**

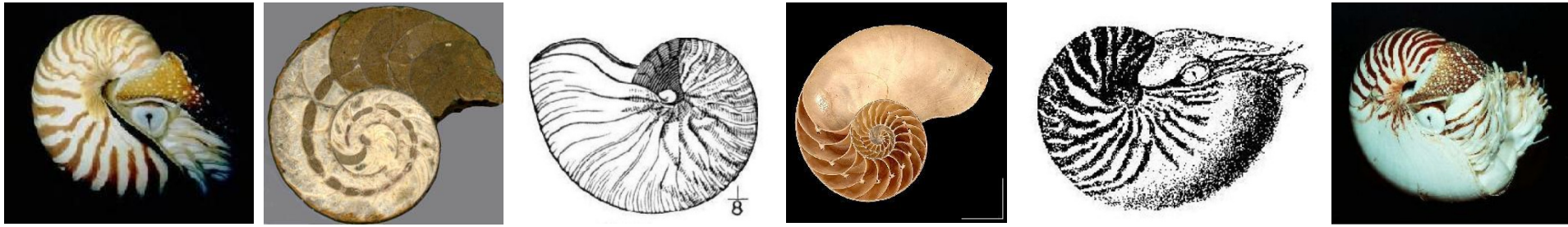




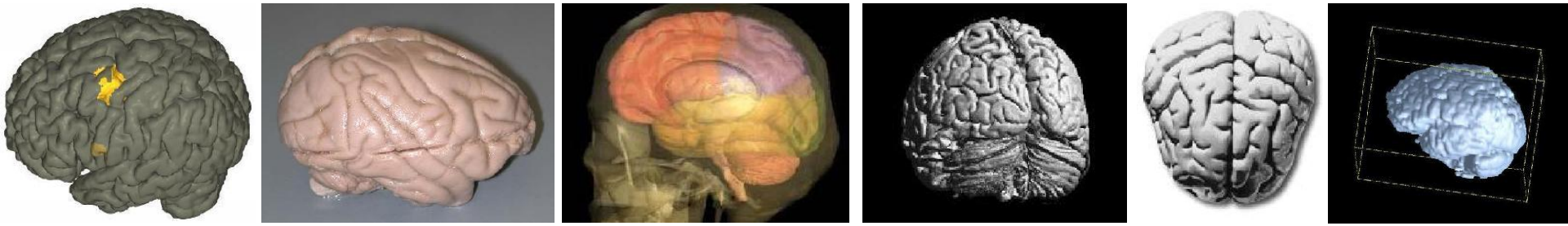


kangaroo

PMK/SIFT Most Confused Category Pairs



nautilus



brain

Beyond BoW

- Spatial Pyramid (Lazebnik et al)

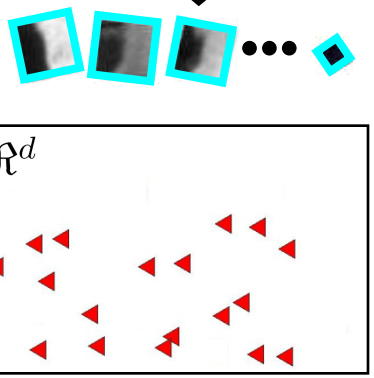
  Geometry in BoW: Pyramid in image space

- **Pyramid Match Kernel (Grauman et al)**

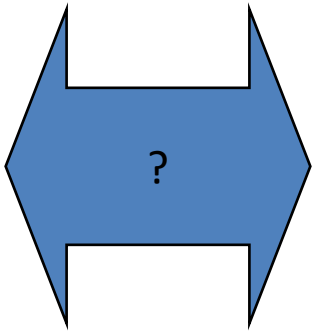
  Pyramid in feature space: Kernel similarity

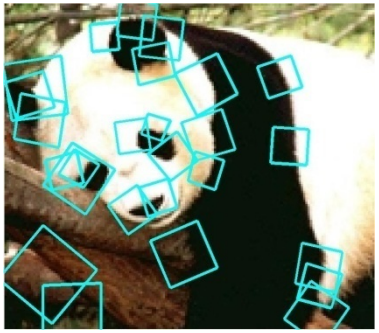
How to Compare Sets of Features?

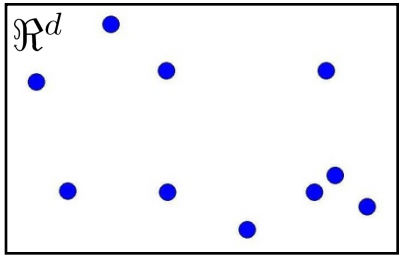
- Each instance is unordered set of vectors
- Varying number of vectors per instance



$$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$$

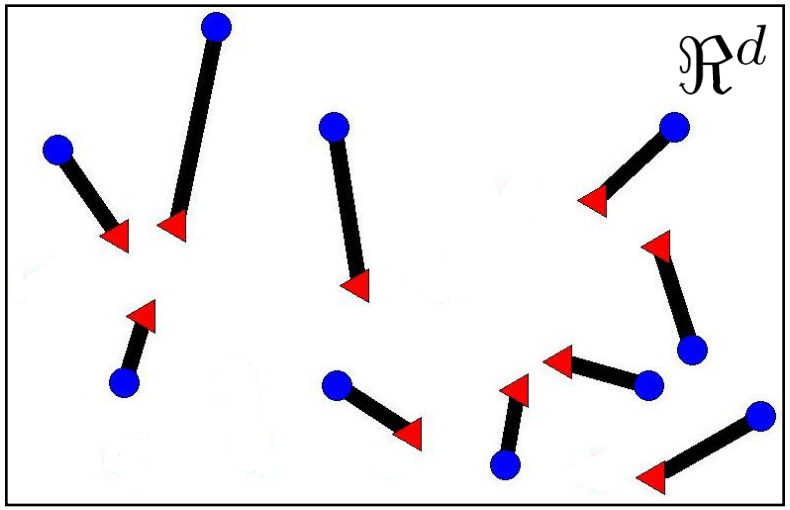






$$\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$$

Correspondence-Based Match



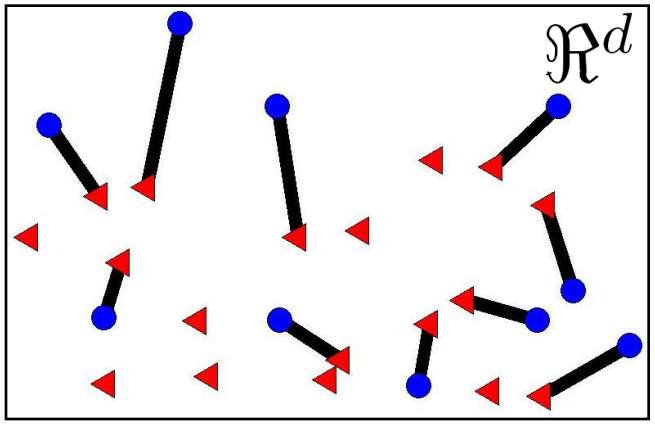
Explicit search for correspondences…

$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

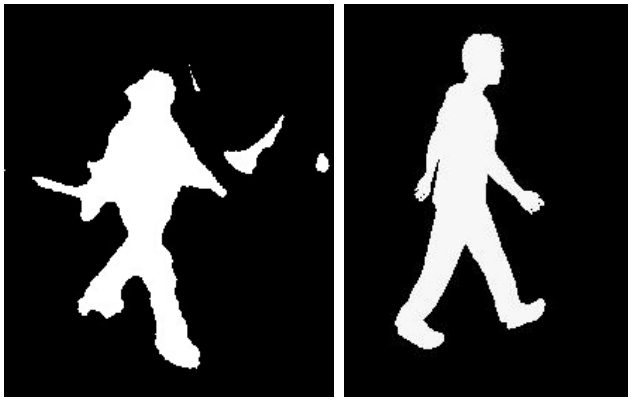
[Wallraven et al., Lyu, Boughhorbel et al., Belongie et al., Rubner et al., Berg et al., Gold & Rangarajan, Shashua & Hazan,…]

Partial Matching

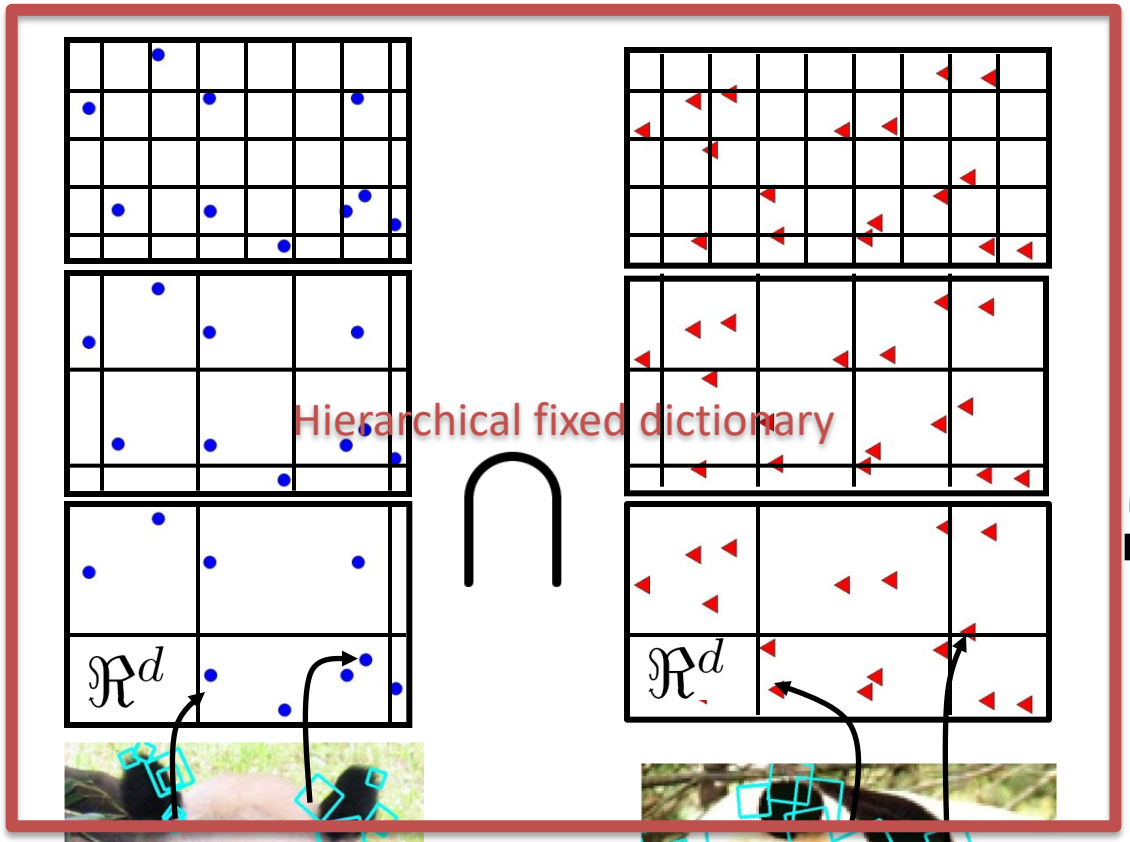
Compare sets by computing a *partial matching* between their features.

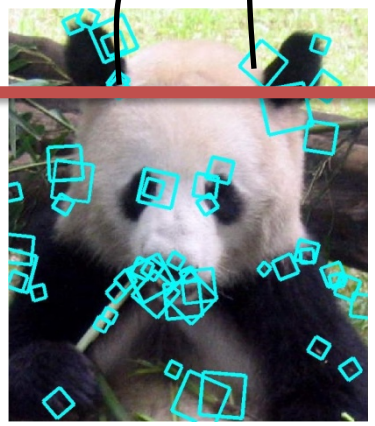


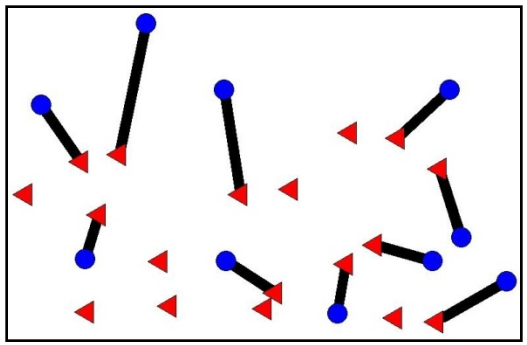
$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$



Pyramid Match



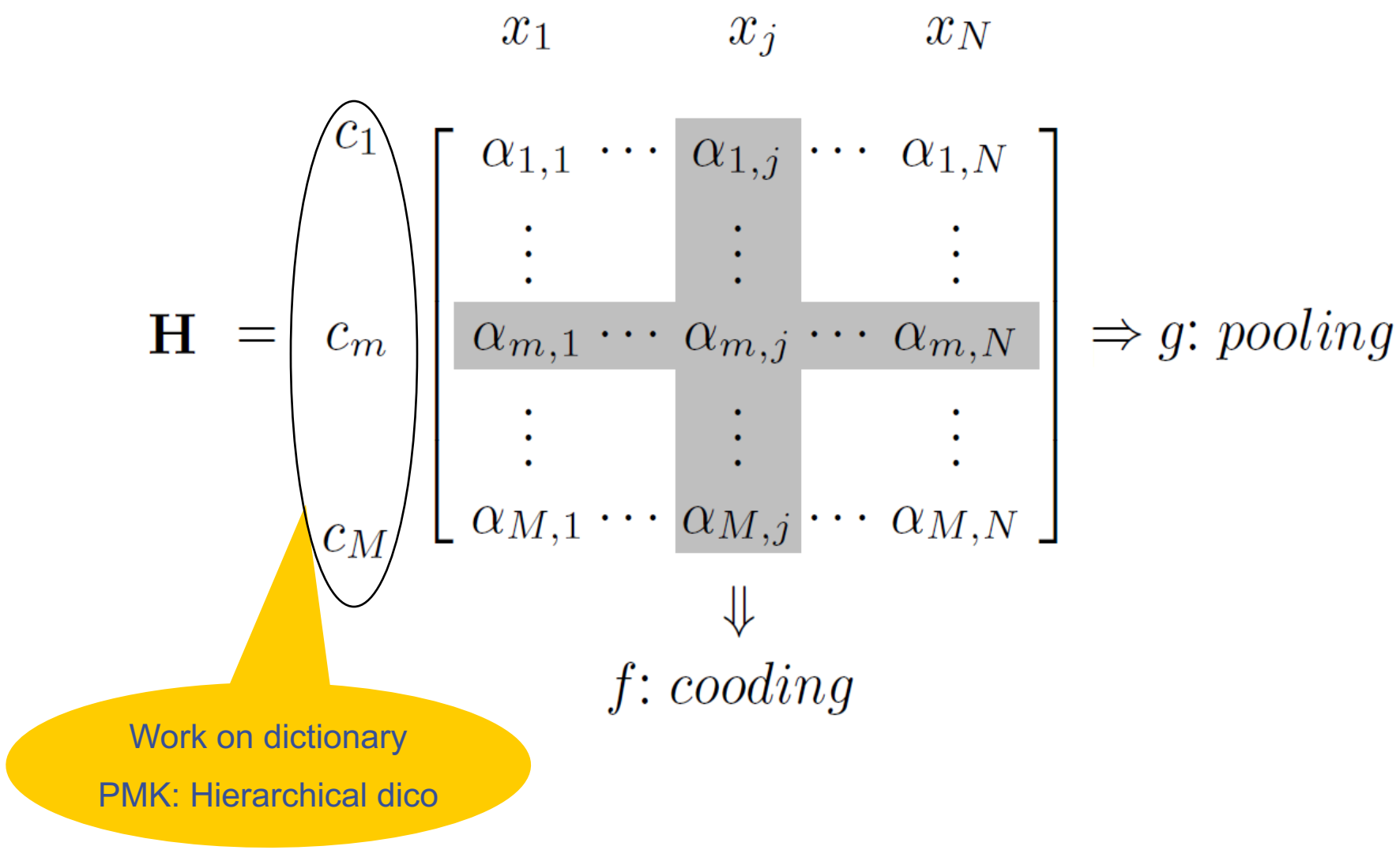




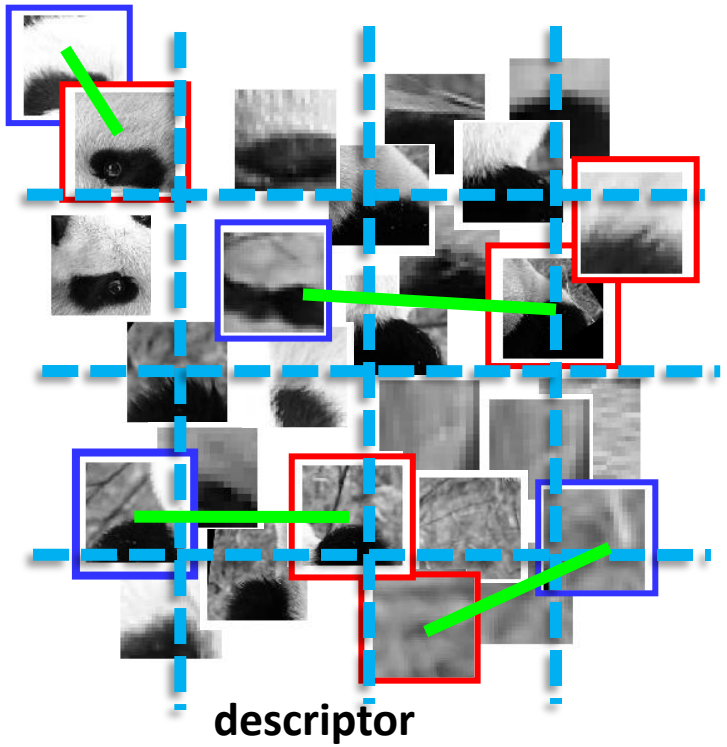
optimal partial matching

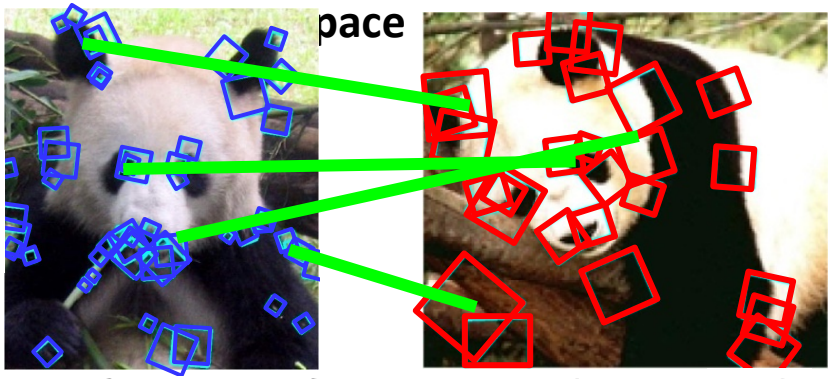
$$\max_{\pi:\mathbf{X}\to\mathbf{Y}} \sum_{\mathbf{x}_i\in\mathbf{X}} \mathcal{S}(\mathbf{x}_i, \pi(\mathbf{x}_i))$$

$$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\} \qquad \mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$$


x_1 x_j x_N
c_1
c_m
c_M
$\mathbf{H} =$
$\alpha_{1,1} \cdots \alpha_{1,j} \cdots \alpha_{1,N}$
$\alpha_{m,1} \cdots \alpha_{m,j} \cdots \alpha_{m,N}$
$\alpha_{M,1} \cdots \alpha_{M,j} \cdots \alpha_{M,N}$
$\Rightarrow g$: $pooling$
$\Downarrow$
f: $cooding$
Work on dictionary
PMK: Hierarchical dico

Pyramid match: main idea

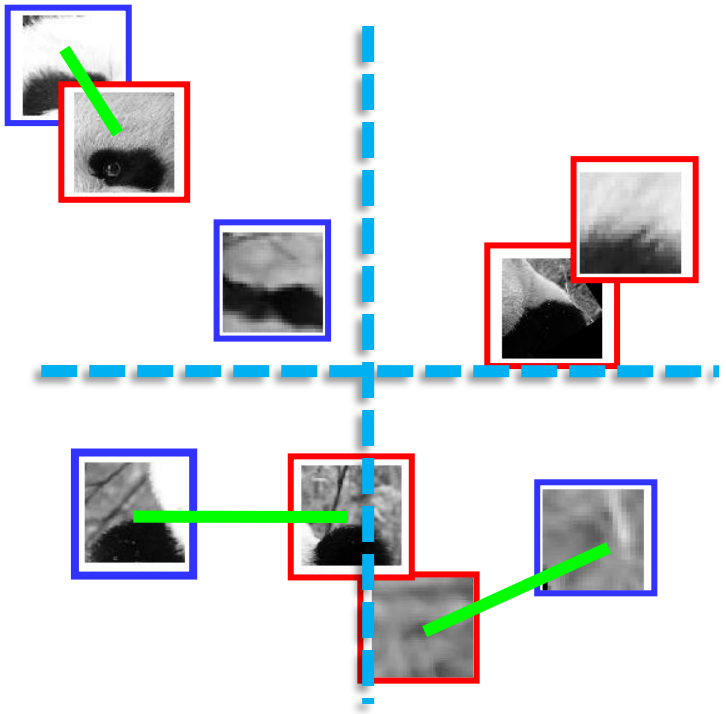


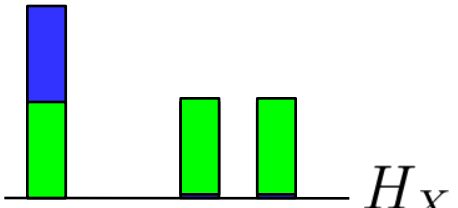


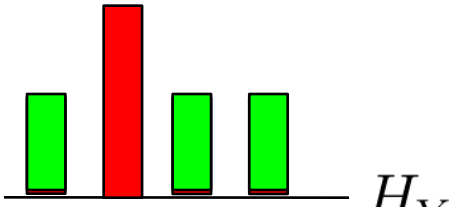
Feature space partitions serve to “match” the local descriptors within successively wider regions.

Pyramid match: main idea

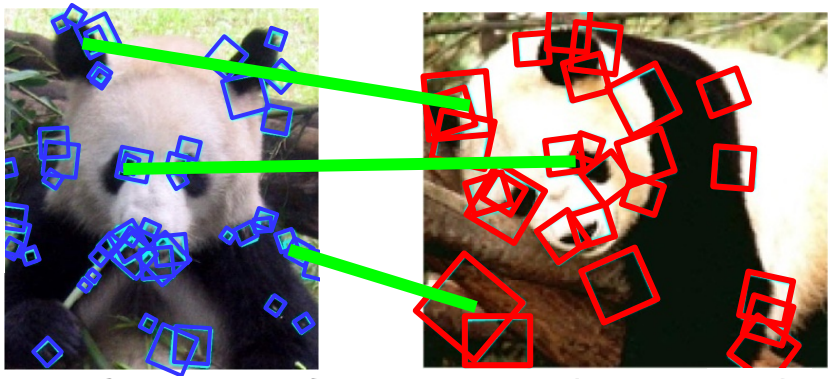
$\Re^d$







$$\mathcal{I}(H_X, H_Y) = \sum_j \min\left(H_X(j), H_Y(j)\right)$$
$$= 3$$



$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Histogram intersection counts number of possible matches at a given partitioning.

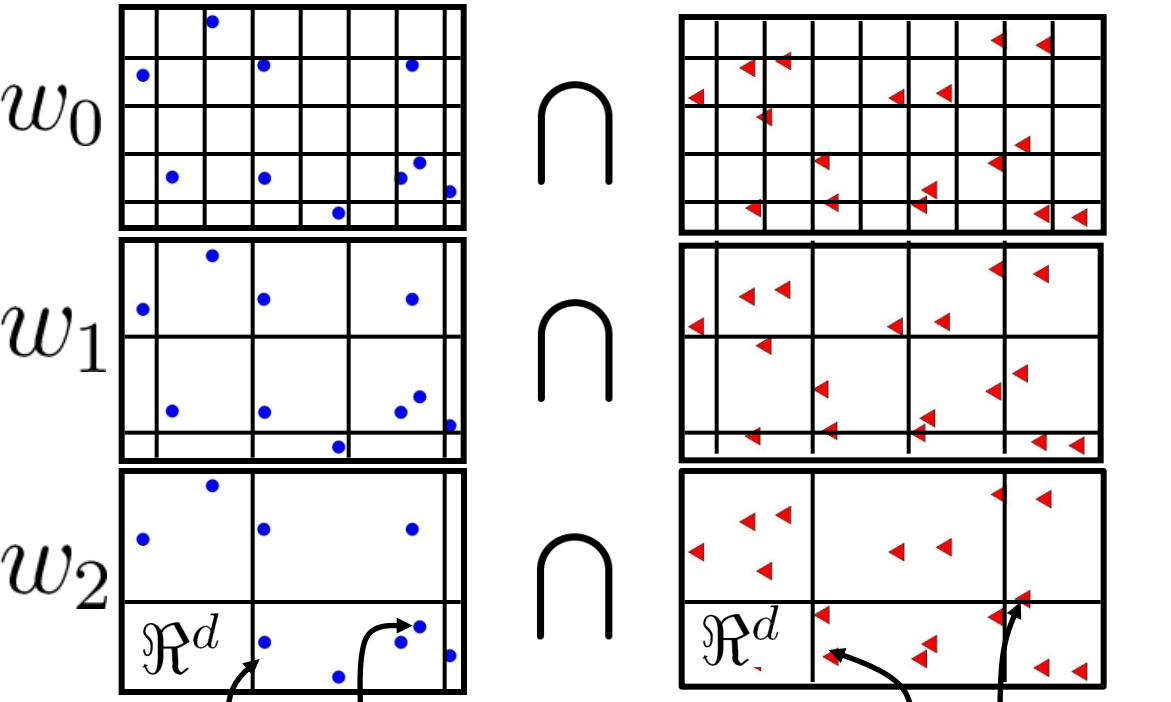
Pyramid match kernel

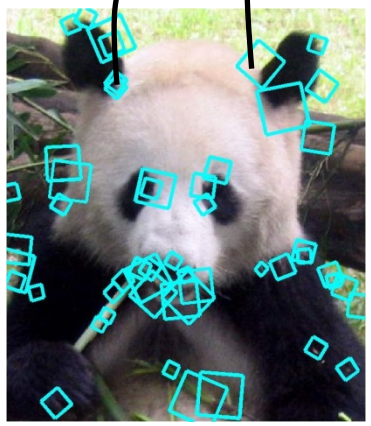
$$K_\Delta(X,Y) = \sum_{i=0}^{L} \underbrace{2^{-i}}_{\text{measures difficulty of a match at level } i} \underbrace{\mathcal{I}\left(H_X^{(i)}, H_Y^{(i)}\right) - \mathcal{I}\left(H_X^{(i-1)}, H_Y^{(i-1)}\right)}_{\text{number of newly matched pairs at level } i}$$

- For similarity, weights inversely proportional to bin size (or may be learned)
- Normalize these kernel values to avoid favoring large sets

Rq: Back to the SPM article: we have the explanation of the SPM kernel too !

Pyramid match kernel

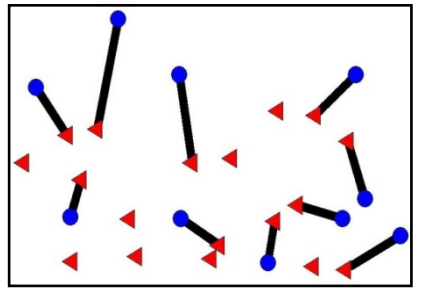




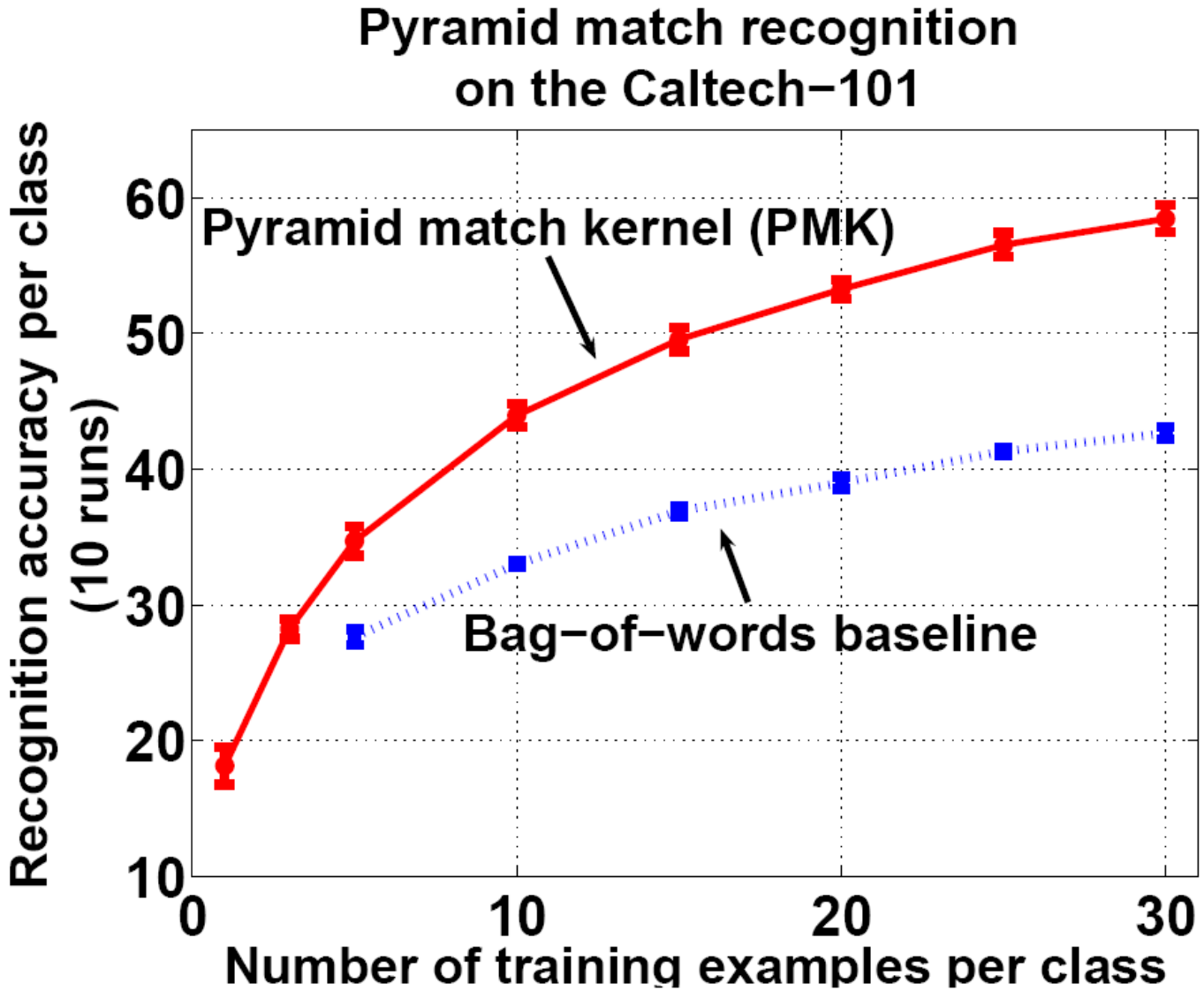
$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$ $\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$

Optimal match: $O(m^3)$

Pyramid match: $O(mL)$

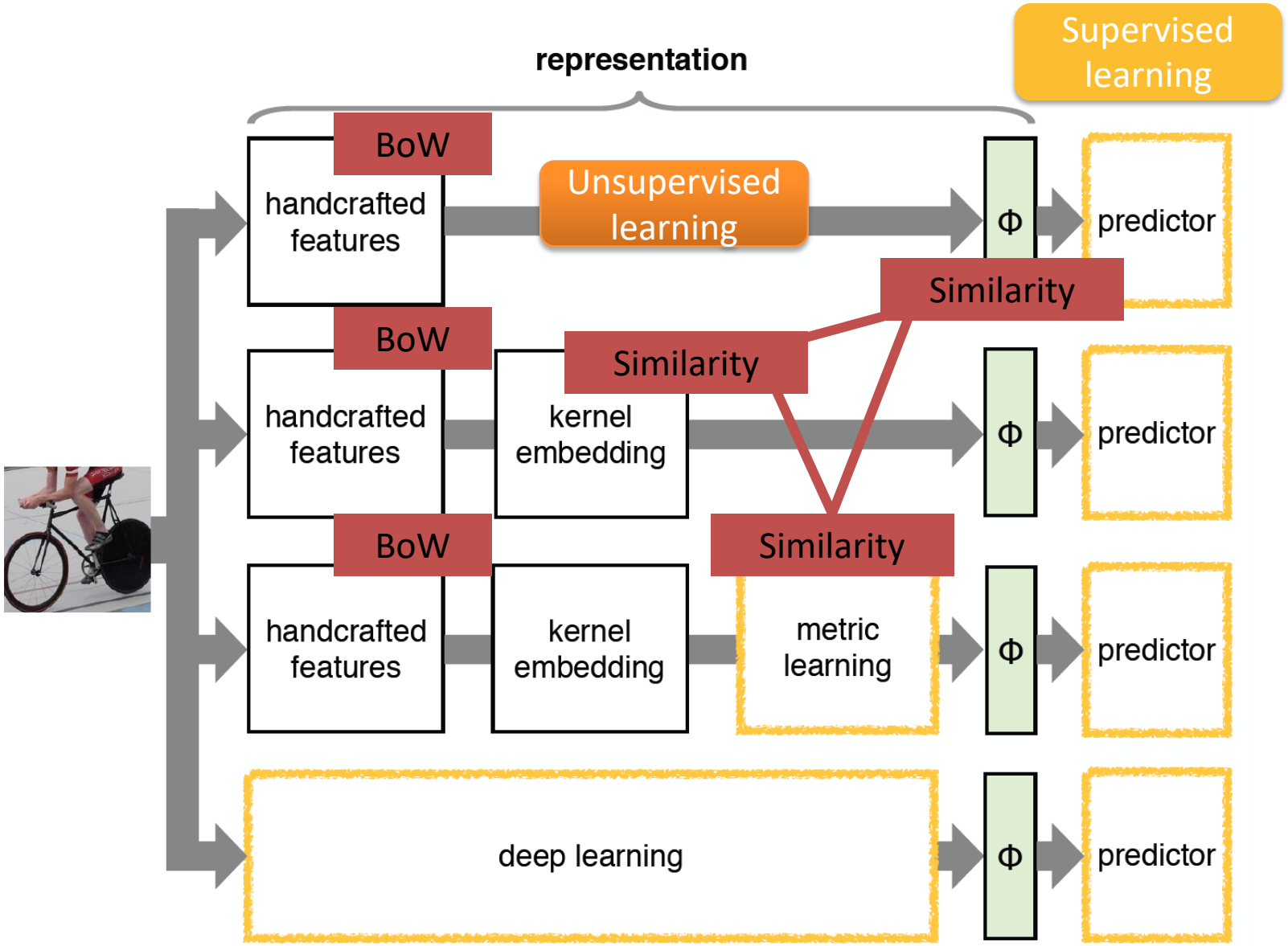


optimal partial matching


Pyramid match recognition
on the Caltech−101
Recognition accuracy per class
(10 runs)
Pyramid match kernel (PMK)
Bag−of−words baseline
60
50
40
30
20
10
0
10
20
30
Number of training examples per class

Bow and beyond / similarity

- A Full chain of process from data to labels
  - Geometry via spatial grids and pyramids
  - Coding/pooling: unsupervised learning, one step to hierarchical/deep learning representations
  - Similarity: the kernel side :


representation
Supervised learning
BoW
handcrafted features
Unsupervised learning
Φ
predictor
Similarity
BoW
handcrafted features
kernel embedding
Similarity
Φ
predictor
BoW
Similarity
handcrafted features
kernel embedding
metric learning
Φ
predictor
deep learning
Φ
predictor