Representations and Kernels

Vinay P. Namboodiri

Slide credits to Kristen Grauman, Trevor Darrell, Svetlana Lazebnik, Jean Ponce, Cordelia Schmid

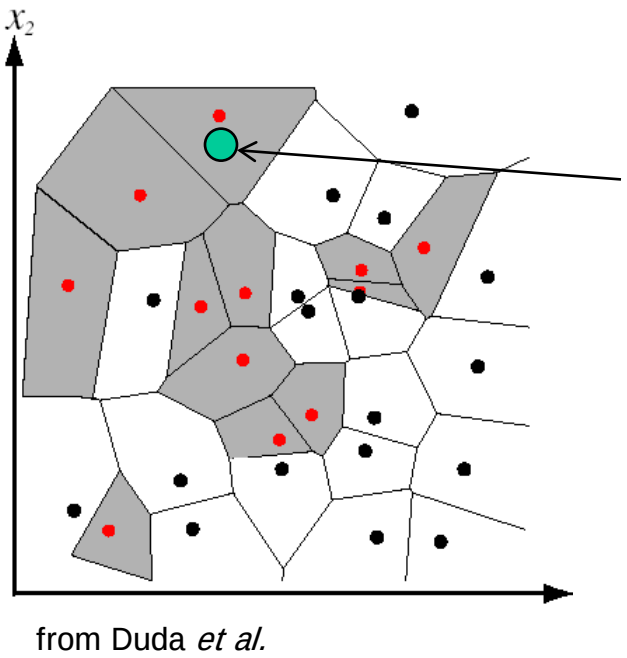
Last Class

- Representation
  - How to represent an object category
- Learning
  - How to form the classifier, given training data
- Recognition
  - How the classifier is to be used on novel data

Nearest Neighbor classification

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

Black = negative
Red = positive



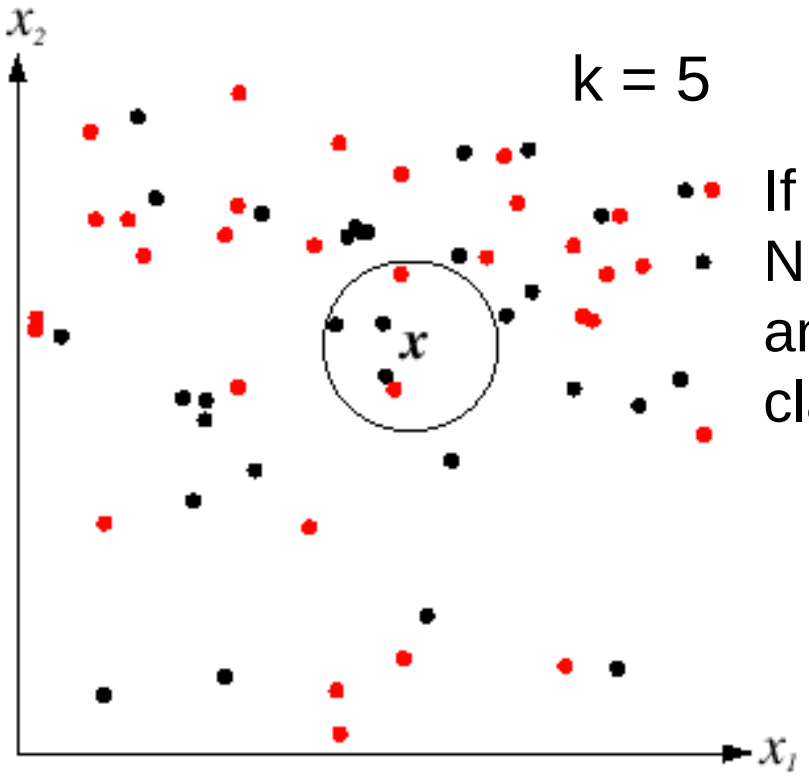
from Duda *et al.*

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

K-Nearest Neighbors classification

- For a new point, find the k closest points from training data
- Labels of the k points “vote” to classify

Black = negative
Red = positive



If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

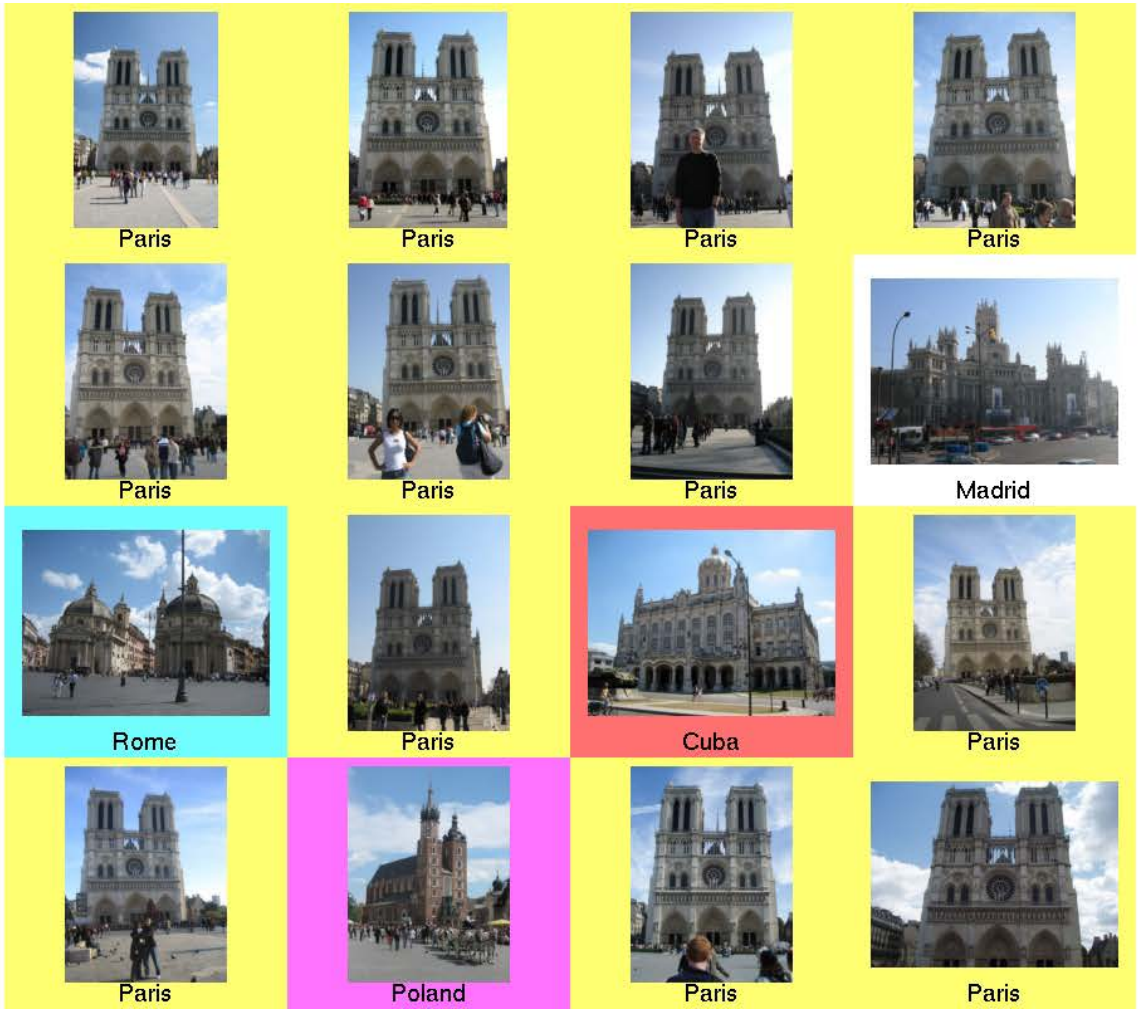
Source: D. Lowe

Where in the World?

[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

Slides: James Hays

Scene Matches



[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

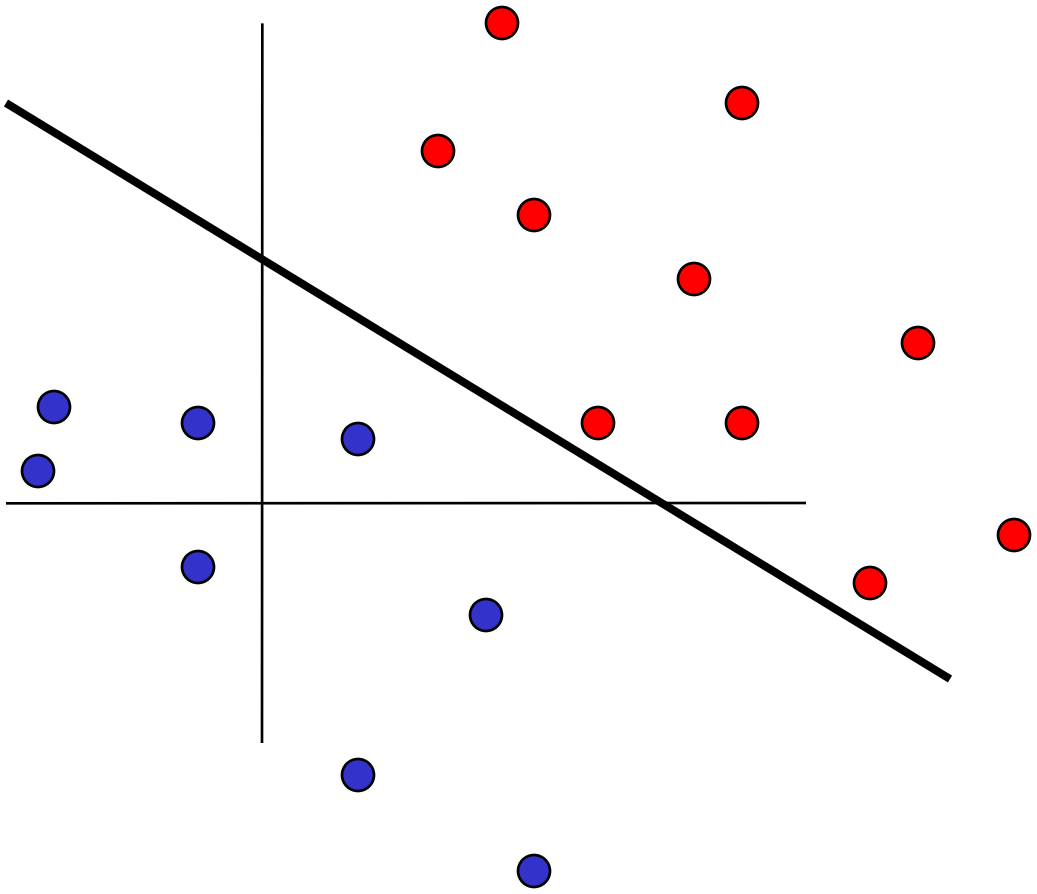
Slides: James Hays

Scene Matches

[Hays and Efros. **im2gps**: Estimating Geographic Information from a Single Image. CVPR 2008.]

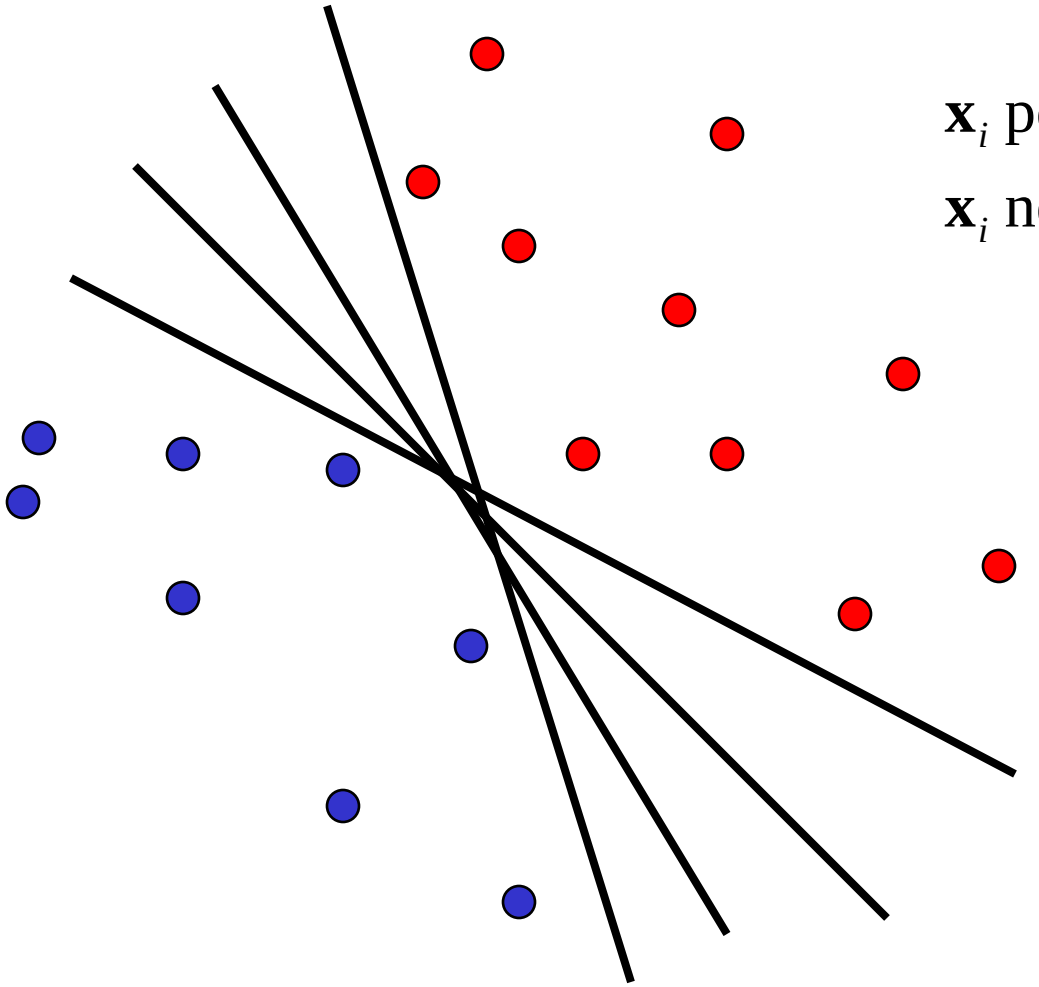
Slides: James Hays

Linear classifiers



Linear classifiers

- Find linear function to separate positive and negative examples

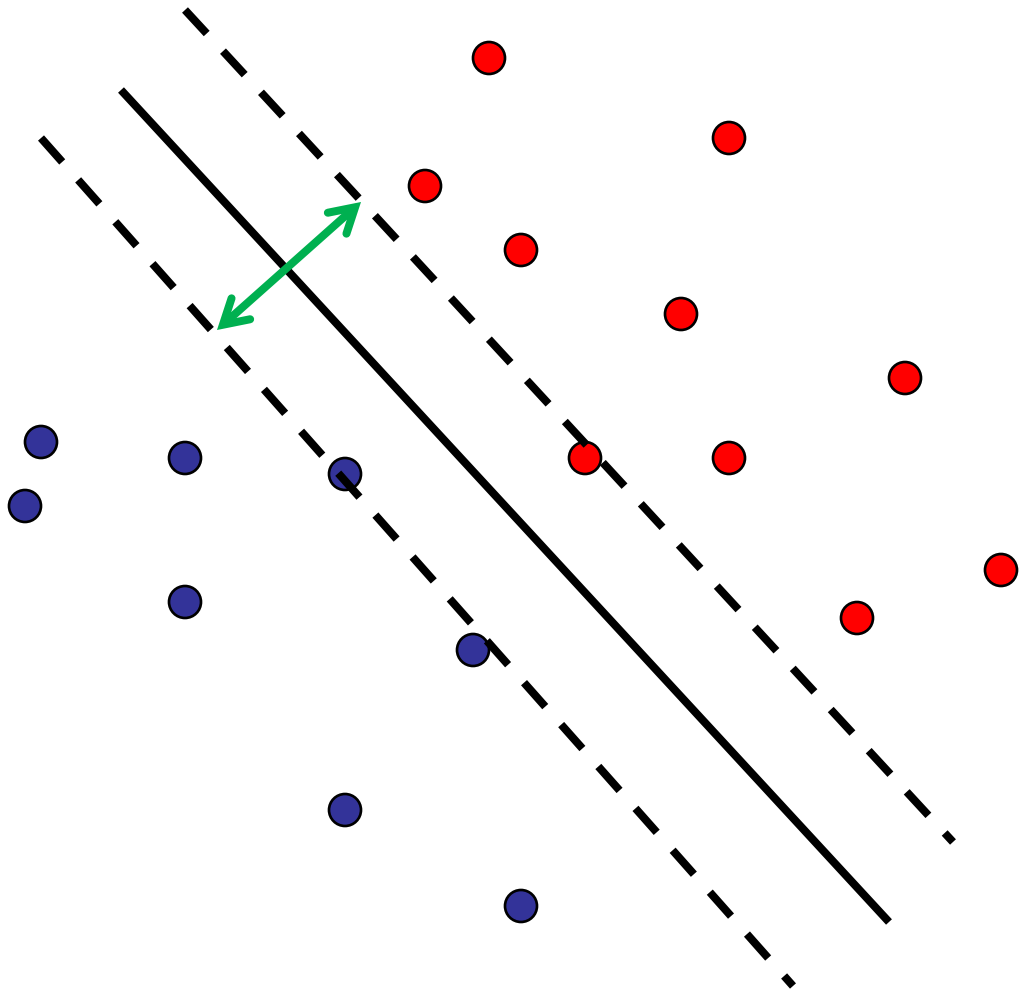


$\mathbf{x}_i$ positive: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 0$

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

Which line is best?

Support Vector Machines (SVMs)



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

Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

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

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

Quadratic optimization problem:

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

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

Finding the maximum margin line

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

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

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

- Classification function:

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

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

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

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

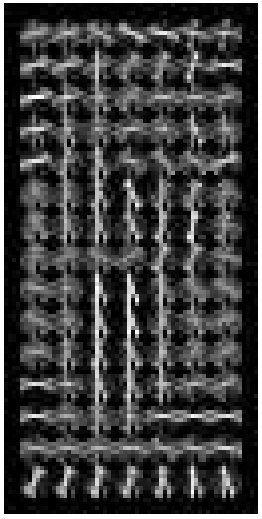
Questions

- **What if the features are not 2d?**
- What if the data is not linearly separable?
- What if we have more than just two categories?

Questions

- What if the features are not 2d?
  - Generalizes to d-dimensions – replace line with "hyperplane"
- What if the data is not linearly separable?
- What if we have more than just two categories?

Person detection with HoG's & linear SVM's



- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

Code available:
http://pascal.inrialpes.fr/soft/olt/

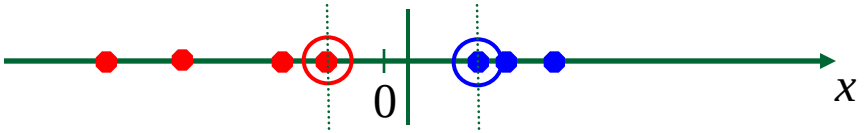
Dalal & Triggs, CVPR 2005

Questions

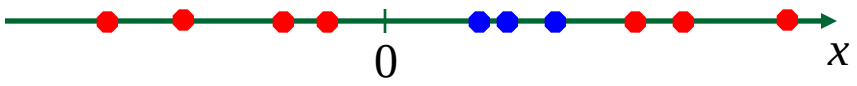
- What if the features are not 2d?
- **What if the data is not linearly separable?**
- What if we have more than just two categories?

Non-linear SVMs

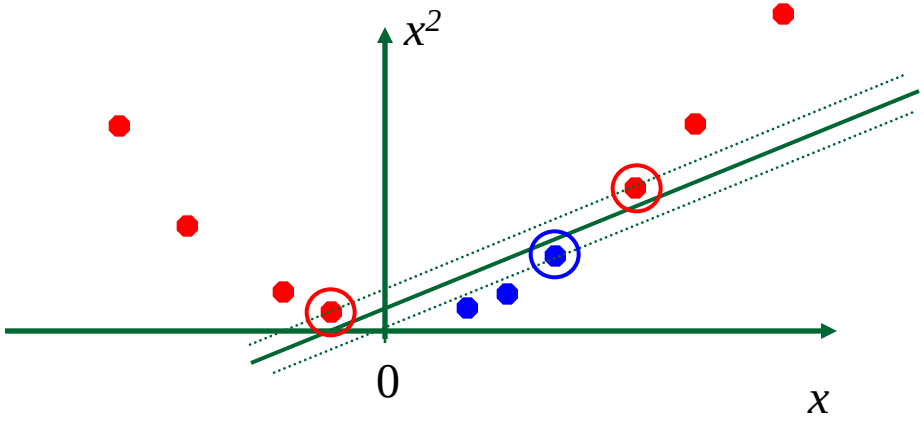
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?

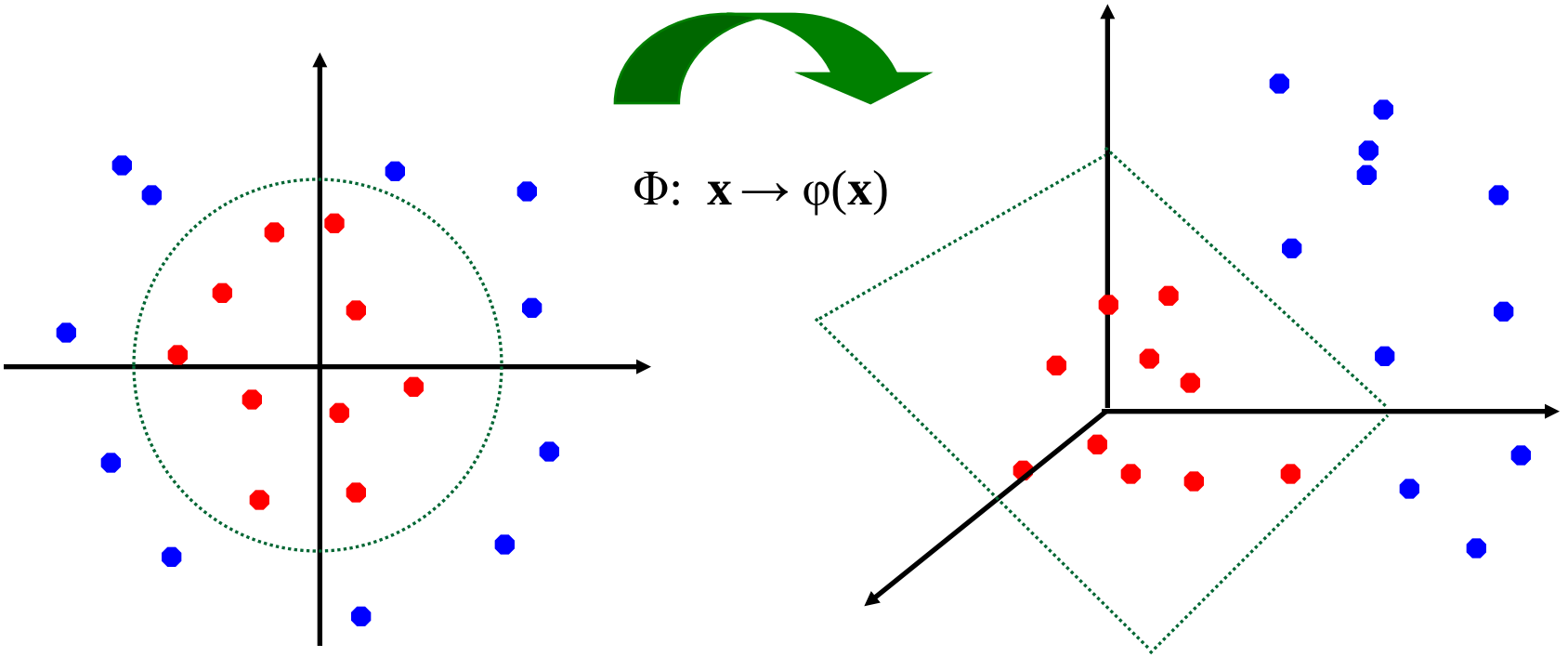


- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: feature spaces

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



Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

The "Kernel Trick"

- The linear classifier relies on dot product between vectors $K(x_i,x_j)=x_i^T x_j$

- If every data point is mapped into high-dimensional space via some transformation Φ: $x \rightarrow \varphi(x)$, the dot product becomes:

$$K(x_i,x_j)= \varphi(x_i)^T \varphi(x_j)$$

- A *kernel function* is similarity function that corresponds to an inner product in some expanded feature space.

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Example

2-dimensional vectors $x=[x_1 \;\; x_2]$;

let $K(x_i,x_j)=(1 + x_i^T x_j)^2$

Need to show that $K(x_i,x_j)= \varphi(x_i)^T \varphi(x_j)$:

$K(x_i,x_j)=(1 + x_i^T x_j)^2$,

$= 1+ x_{i1}^2 x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2} + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$

$= [1 \;\; x_{i1}^2 \;\sqrt{2}\, x_{i1}x_{i2} \;\; x_{i2}^2 \;\sqrt{2}x_{i1} \;\sqrt{2}x_{i2}]^T$

$[1 \;\; x_{j1}^2 \;\sqrt{2}\, x_{j1}x_{j2} \;\; x_{j2}^2 \;\sqrt{2}x_{j1} \;\sqrt{2}x_{j2}]$

$= \varphi(x_i)^T \varphi(x_j)$,

where $\varphi(x) = [1 \;\; x_1^2 \;\sqrt{2}\, x_1x_2 \;\; x_2^2 \;\; \sqrt{2}x_1 \;\sqrt{2}x_2]$

from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

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:

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

Examples of kernel functions

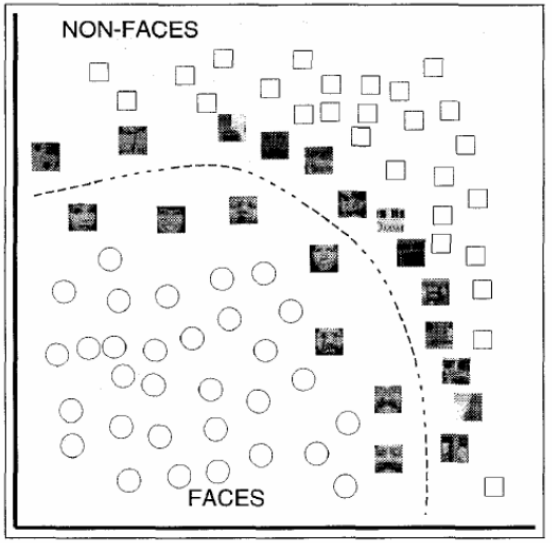
- Linear: $K(x_i, x_j) = x_i^T x_j$

- Gaussian RBF: $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$

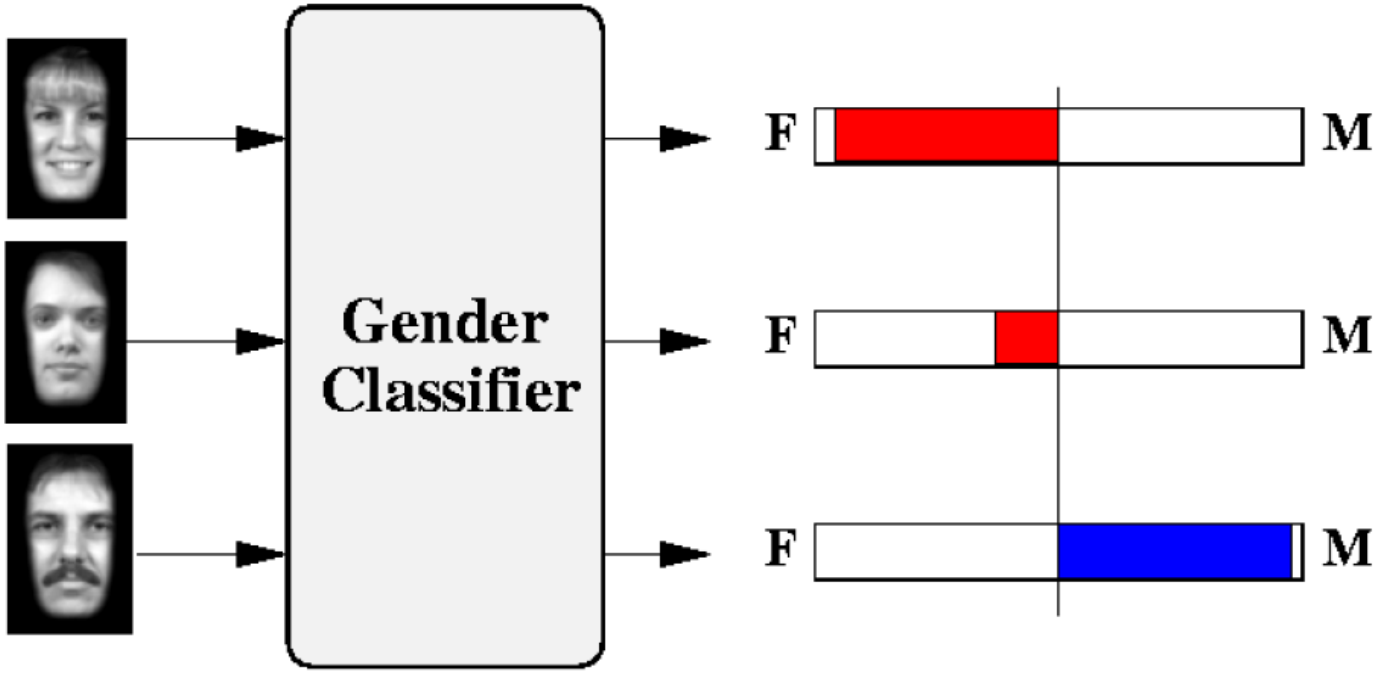
- Histogram intersection: $K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$

SVMs for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Use this “kernel matrix” to solve for SVM support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.



Example: learning gender with SVMs



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Moghaddam and Yang, Face & Gesture 2000.

Face alignment processing

Processed faces

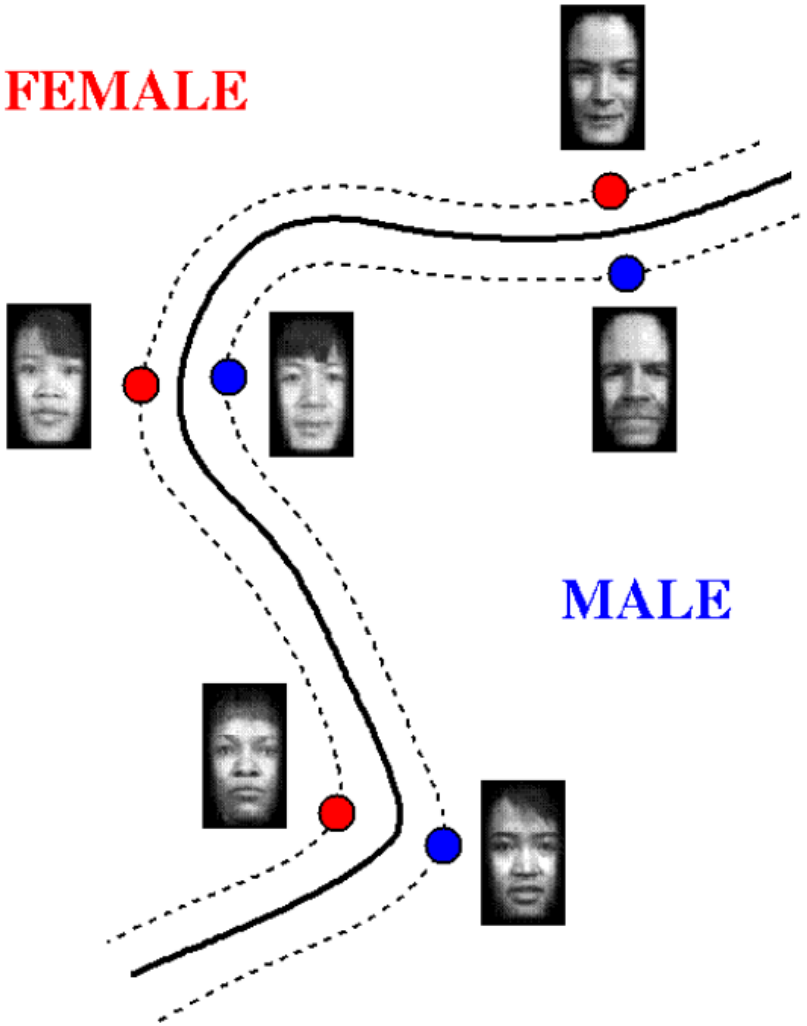
Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Learning gender with SVMs

- Training examples:
  - 1044 males
  - 713 females
- Experiment with various kernels, select Gaussian RBF

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\frac{\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2}{2\sigma^2})$$

Support Faces



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Classifier Performance

Classifier	Error Rate		
	Overall	Male	Female
SVM with RBF kernel	**3.38%**	**2.05%**	**4.79%**
SVM with cubic polynomial kernel	**4.88%**	**4.21%**	**5.59%**
Large Ensemble of RBF	5.54%	4.59%	6.55%
Classical RBF	7.79%	6.89%	8.75%
Quadratic classifier	10.63%	9.44%	11.88%
Fisher linear discriminant	13.03%	12.31%	13.78%
Nearest neighbor	27.16%	26.53%	28.04%
Linear classifier	58.95%	58.47%	59.45%

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Gender perception experiment: How well can humans do?

- Subjects:
  - 30 people (22 male, 8 female)
  - Ages mid-20's to mid-40's
- Test data:
  - 254 face images (6 males, 4 females)
  - Low res and high res versions
- Task:
  - Classify as male or female, forced choice
  - No time limit

Moghaddam and Yang, Face & Gesture 2000.

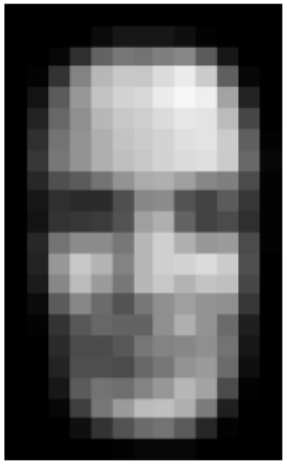
Gender perception experiment: How well can humans do?

Stimuli →

84 x 48

N = 4032

21 x 12



N = 252

Results →

High-Res	Low-Res
6.54%	30.7%
Error	Error

$\sigma = 3.7\%$

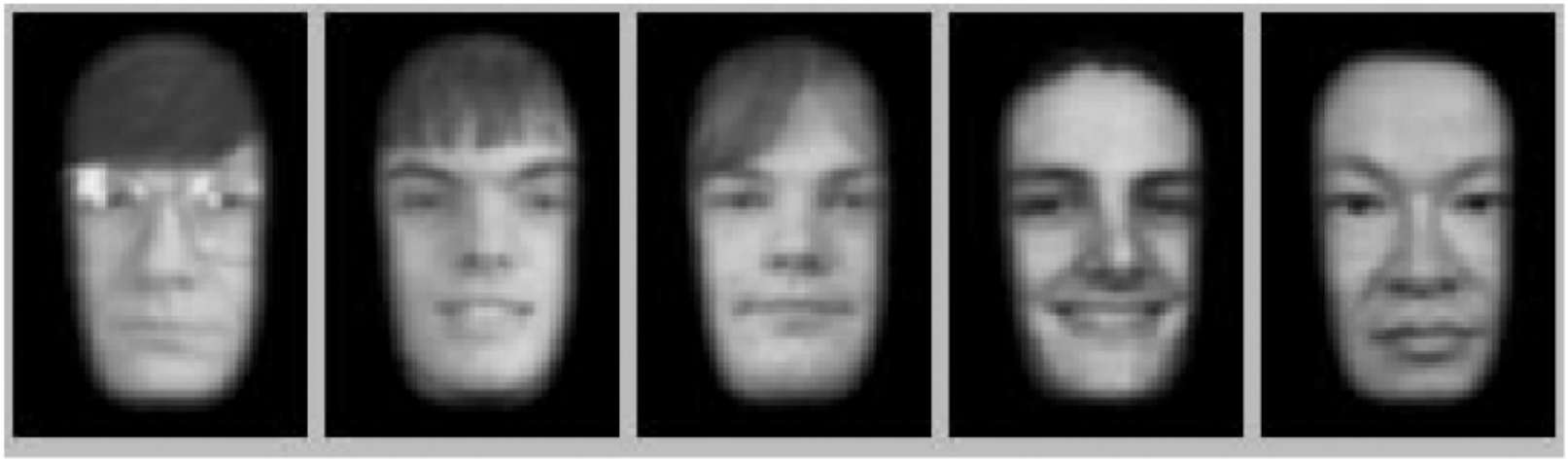
Moghaddam and Yang, Face & Gesture 2000.

Human vs. Machine

Figure 6. SVM vs. Human performance

- SVMs performed better than any single human test subject, at either resolution

Hardest examples for humans



Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.

Questions

- What if the features are not 2d?
- What if the data is not linearly separable?
- **What if we have more than just two categories?**

Multi-class SVMs

- Achieve multi-class classifier by combining a number of binary classifiers

- **One vs. all**
  - Training: learn an SVM for each class vs. the rest
  - 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

SVMs: Pros and cons

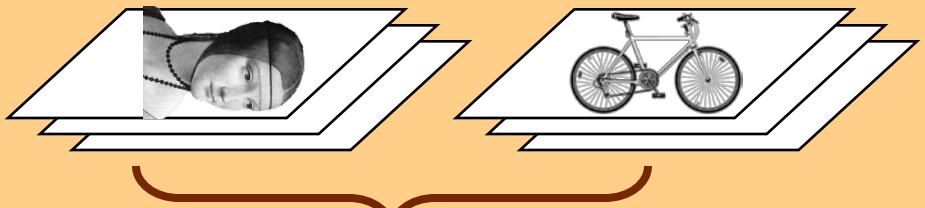
- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - Kernel-based framework is very powerful, flexible
  - Often a sparse set of support vectors – compact at test time
  - Work very well in practice, even with very small training sample sizes
- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Can be tricky to select best kernel function for a problem
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Adapted from Lana Lazebnik

Object → **Bag of ‘words’**

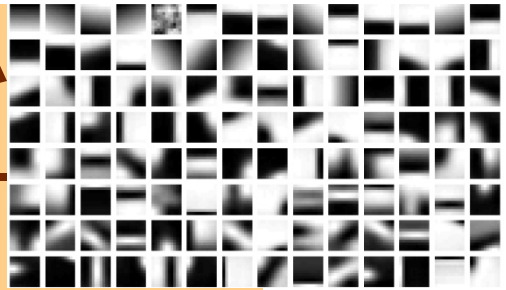
Last Class

Representation



2.

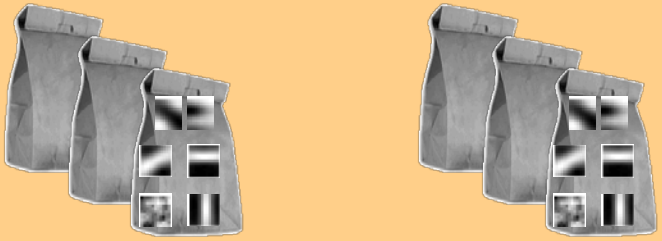
codewords dictionary



1. feature detection & representation

image representation

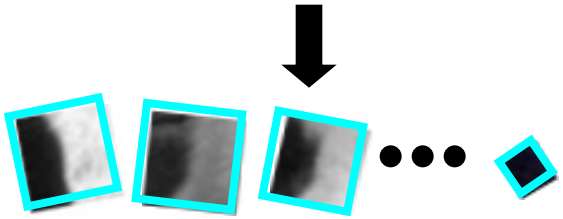
3.



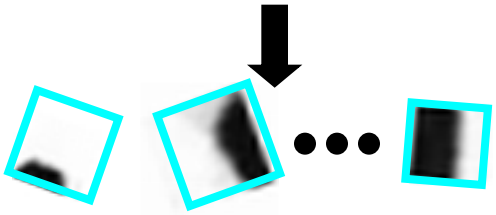
The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features

Kristen Grauman
Trevor Darrell
MIT

Sets of features

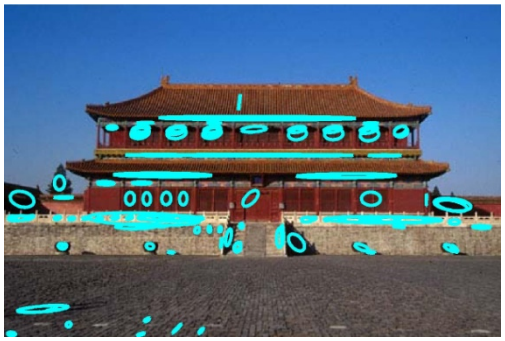


$$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$$

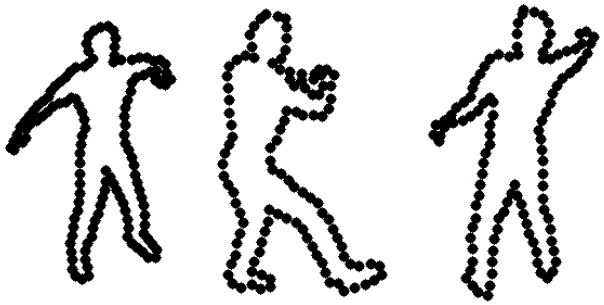


$$\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$$

Sets of features



invariant region descriptors



local shape features

examples under varying conditions

Problem

How to build a **discriminative classifier** using the set representation?

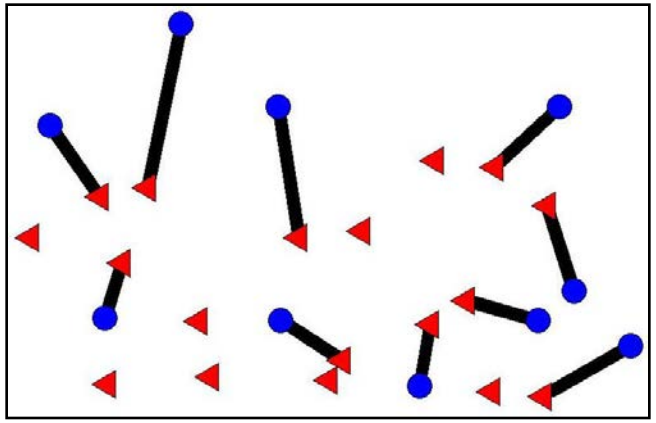
Kernel-based methods (e.g. SVM) are appealing for efficiency and generalization power…

But what is an appropriate kernel?

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

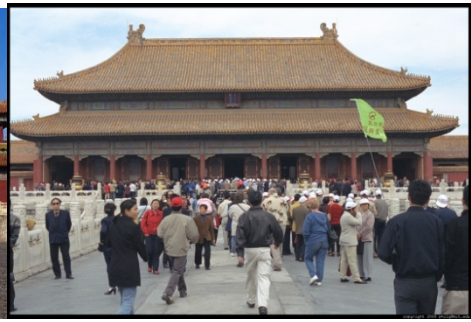
Partial matching for sets of features

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



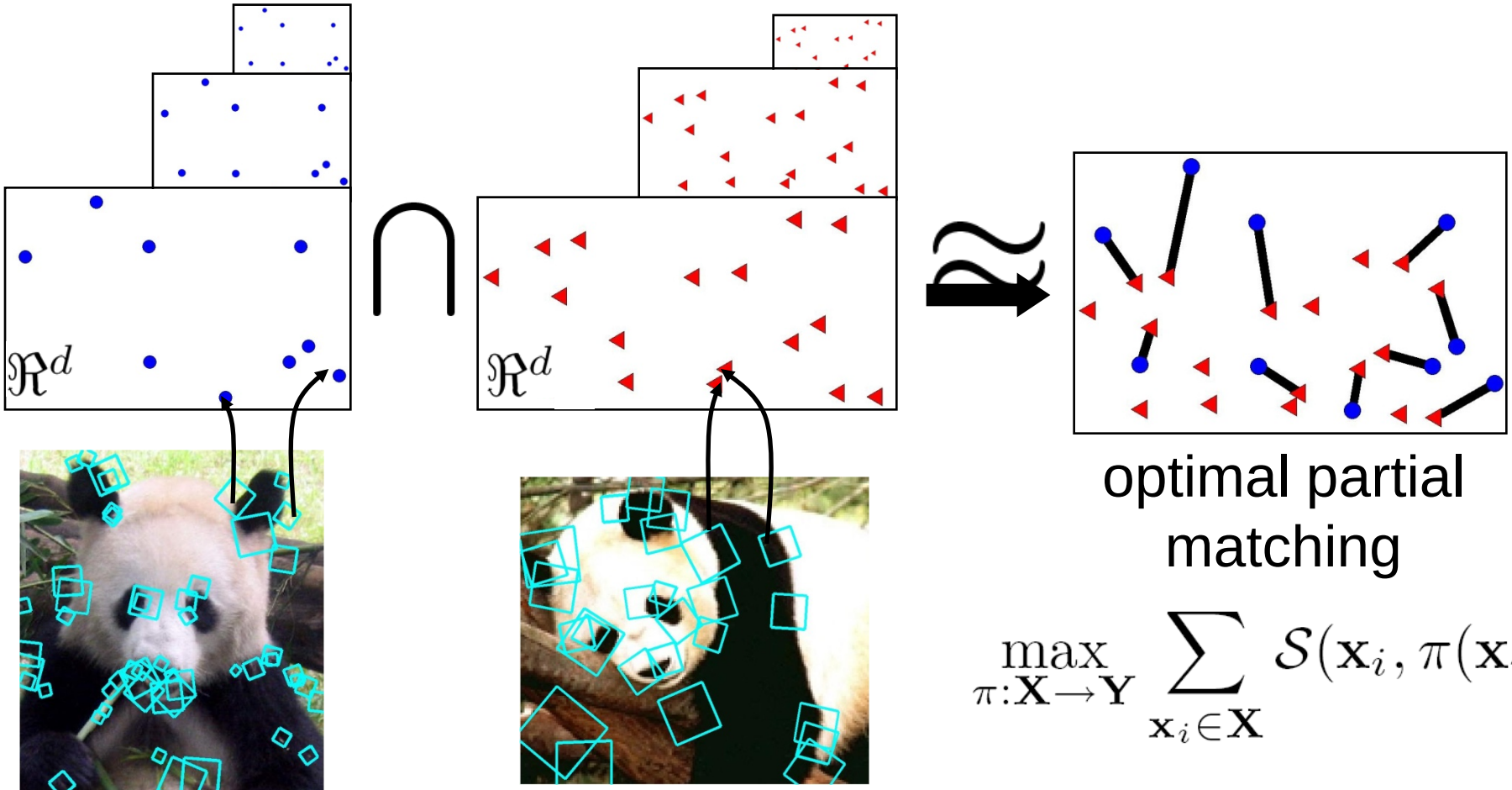




Robust to clutter, segmentation errors, occlusion…

Pyramid match



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

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

$$\vec{\mathbf{x}}_i \in \Re^d \qquad \vec{\mathbf{y}}_i \in \Re^d$$

Pyramid match overview

Pyramid match kernel measures similarity of a partial matching between two sets:

Place multi-dimensional, multi-resolution grid over point sets

Consider points matched at finest resolution where they fall into same grid cell

Approximate similarity between matched points with worst case similarity at given level

No explicit search for matches!

Pyramid match kernel

Number of newly matched pairs at level *i*

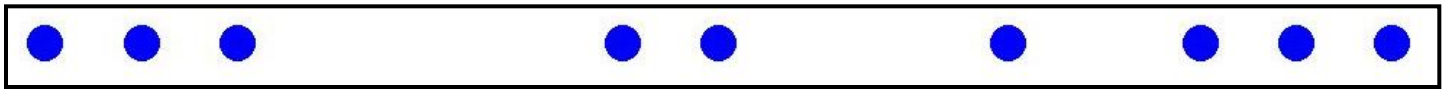
Approximate partial match similarity

$$K_\Delta = \sum_{i=0}^{L} w_i N_i$$

Measure of difficulty of a match at level *i*

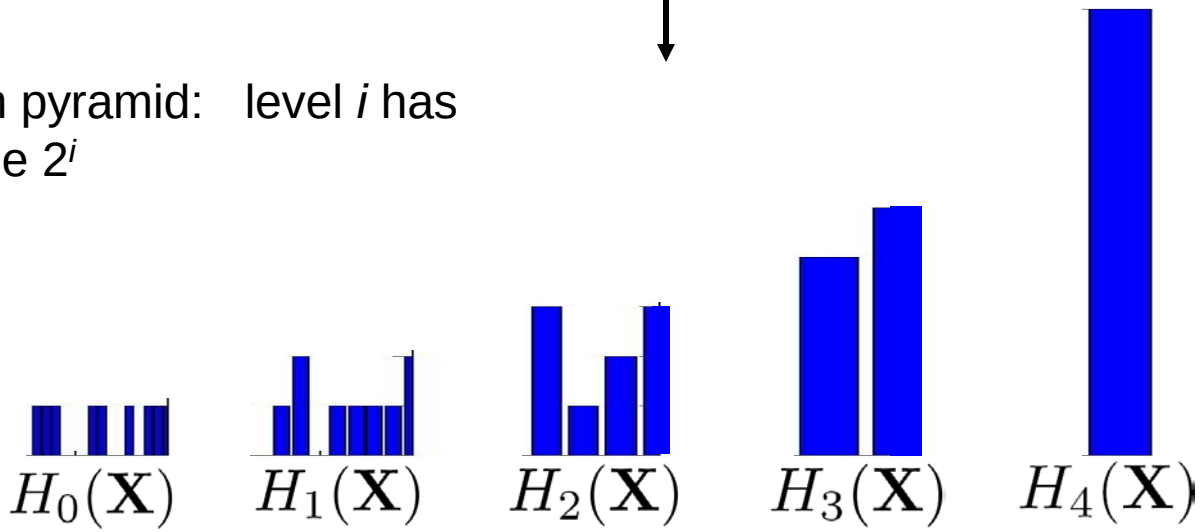
Feature extraction

$$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}, \quad \vec{\mathbf{x}}_i \in \Re^d$$



$d = 1$

Histogram pyramid: level *i* has bins of size 2^i

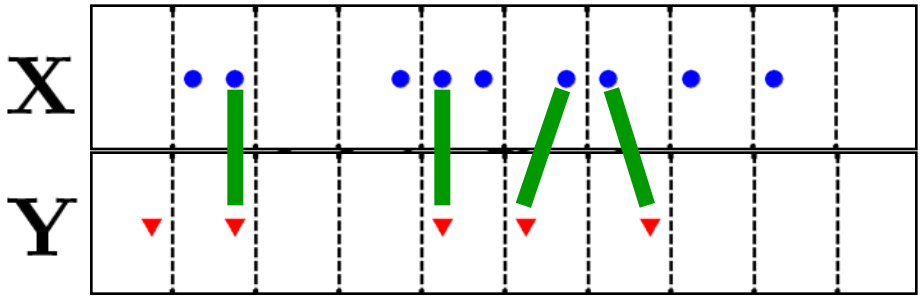


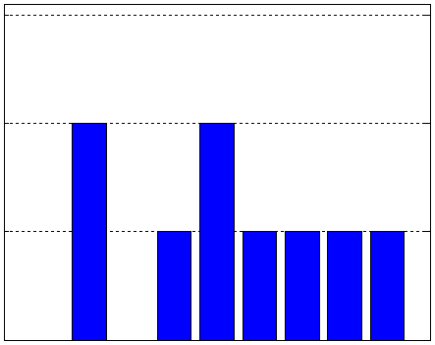
$$\Psi(\mathbf{X}) = [H_0(\mathbf{X}), \ldots, H_L(\mathbf{X})]$$

Counting matches

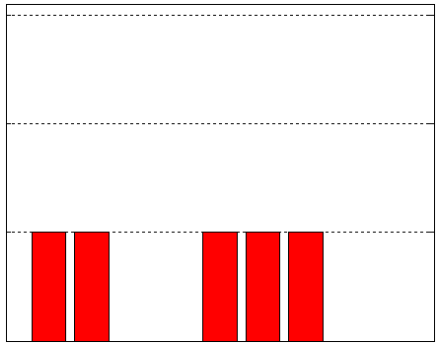
Histogram intersection

$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$

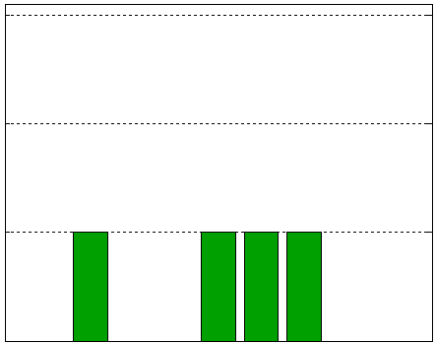




$H(\mathbf{X})$



$H(\mathbf{Y})$



$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = 4$

Counting new matches

Histogram intersection

$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$

matches at this level — matches at previous level

$$N_i = \mathcal{I}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right) - \mathcal{I}\left(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})\right)$$

Difference in histogram intersections across levels counts *number of new pairs* matched

Pyramid match kernel

histogram pyramids

$$K_\Delta\left(\overbrace{\Psi(\mathbf{X}), \Psi(\mathbf{Y})}\right) =$$

$$\sum_{i=0}^{L} \frac{1}{2^i} \Big(\underbrace{\mathcal{I}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right) - \mathcal{I}\left(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})\right)}_{\text{number of newly matched pairs at level } i} \Big)$$

measure of difficulty of a match at level *i*

- Weights inversely proportional to bin size
- Normalize kernel values to avoid favoring large sets

Efficiency

For sets with *m* features of dimension *d*, and pyramids with *L* levels, computational complexity of

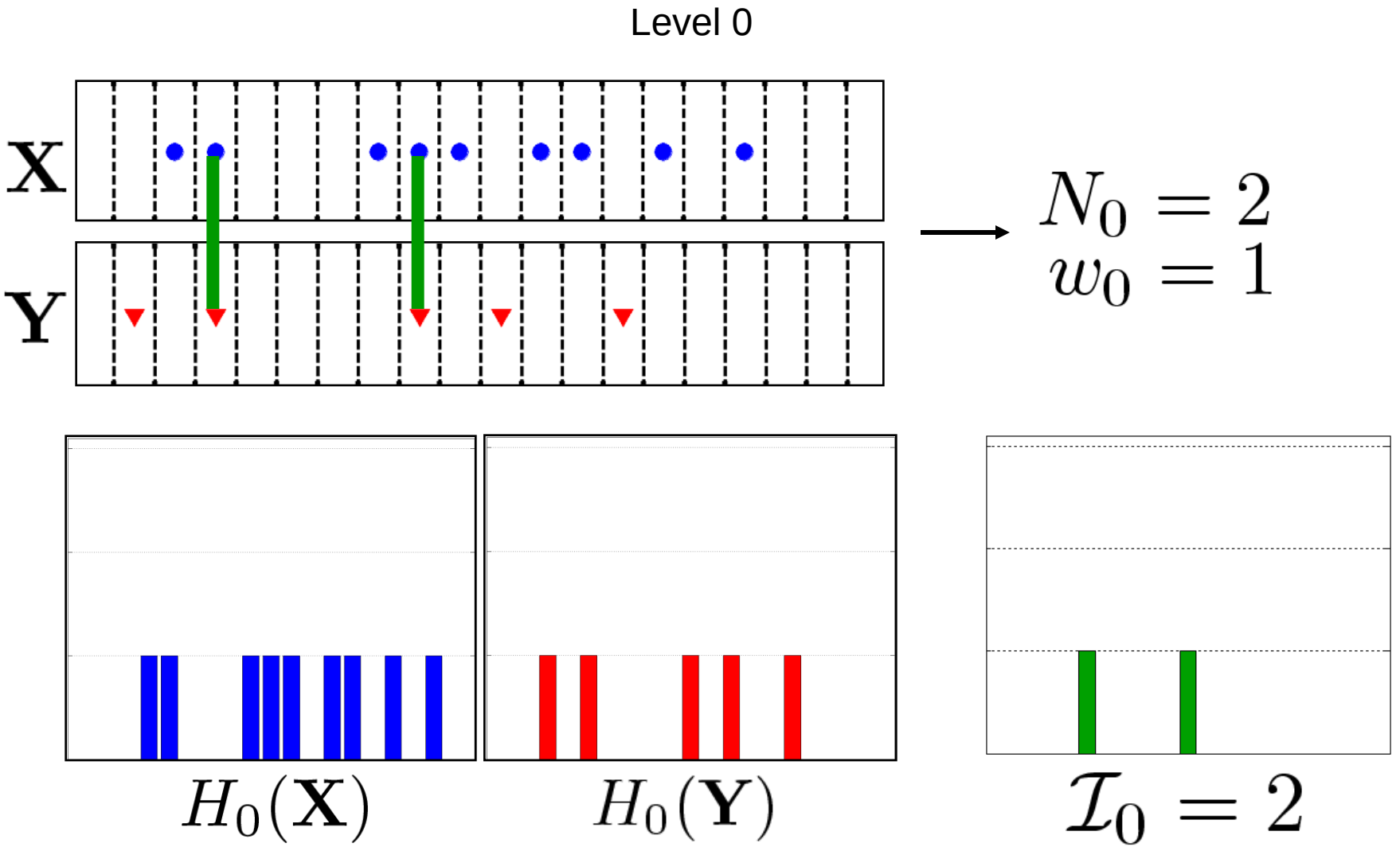
Pyramid match kernel:

$$O(dmL)$$

Existing set kernel approaches:

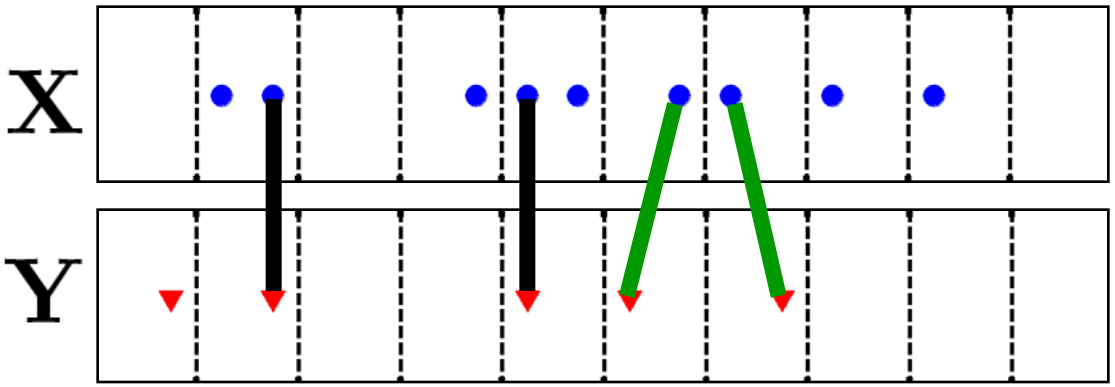
$$O(dm^3) \text{ or } O(dm^2)$$

Example pyramid match



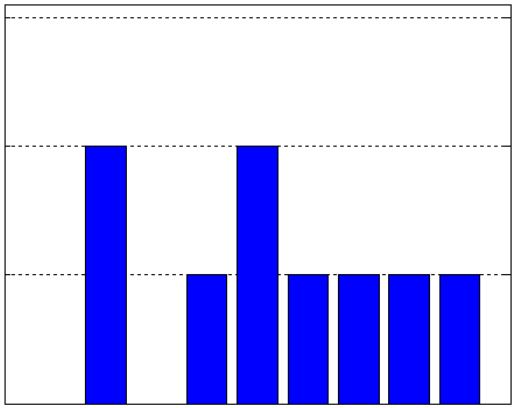
Example pyramid match

Level 1

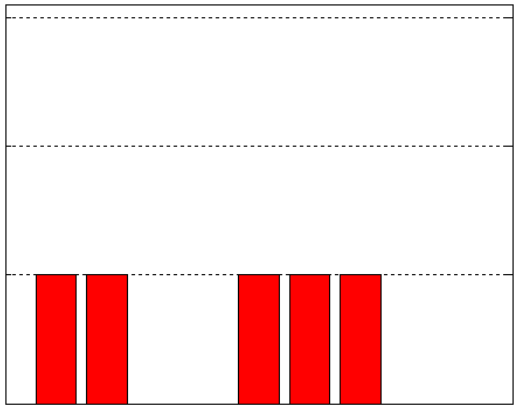


$$N_1 = 4 - 2 = 2$$

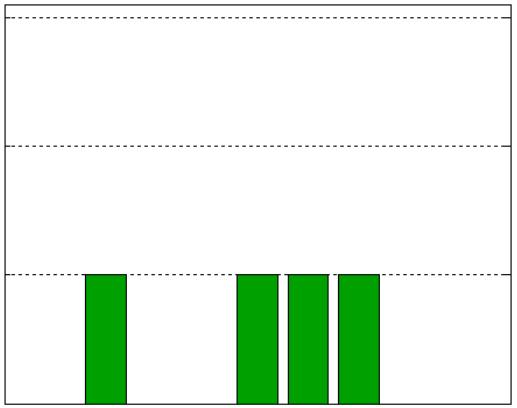
$$w_1 = \frac{1}{2}$$



$H_1(\mathbf{X})$



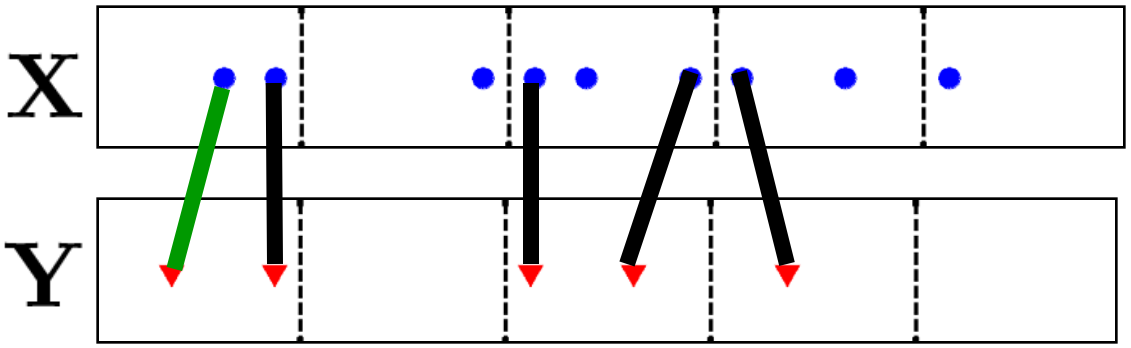
$H_1(\mathbf{Y})$



$\mathcal{I}_1 = 4$

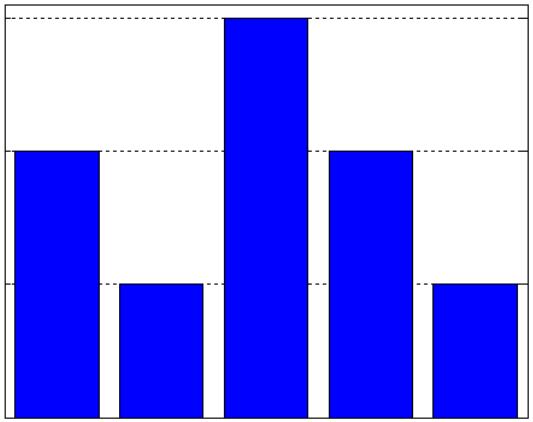
Example pyramid match

Level 2

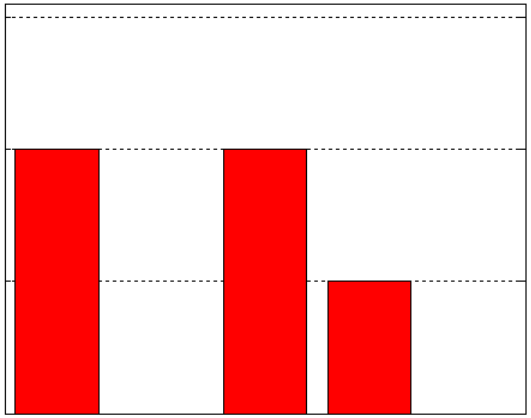


$$N_2 = 5 - 4 = 1$$

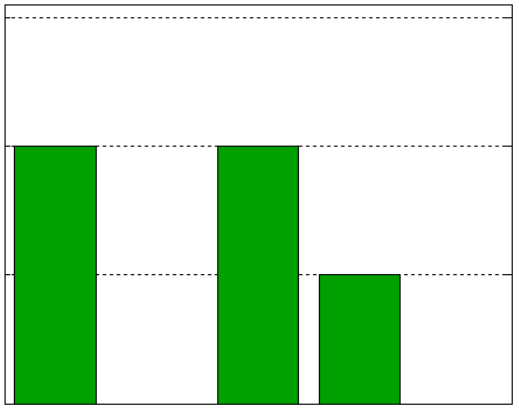
$$w_2 = \frac{1}{4}$$



$H_2(\mathbf{X})$



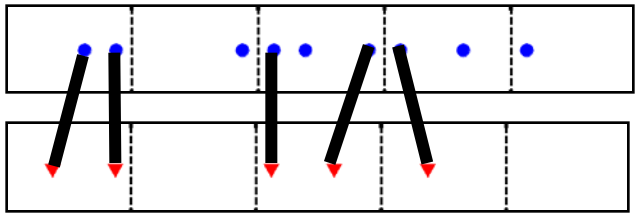
$H_2(\mathbf{Y})$



$\mathcal{I}_2 = 5$

Example pyramid match

pyramid match



$$K_\Delta = \sum_{i=0}^{L} w_i N_i$$

$$= 1(2) + \tfrac{1}{2}(2) + \tfrac{1}{4}(1) = 3.25$$

Building a classifier

Train SVM by computing kernel values between all labeled training examples

Classify novel examples by computing kernel values against support vectors

One-versus-all for multi-class classification

Convergence is guaranteed since pyramid match kernel is positive-definite.

Object recognition results

ETH-80 database 8 object classes

Features:

- Harris detector
- PCA-SIFT descriptor, *d*=10

Kernel	Complexity	Recognition rate
Match *[Wallraven et al.]*	$O(dm^2)$	84%
Bhattacharyya affinity *[Kondor & Jebara]*	$O(dm^3)$	85%
Pyramid match	$O(dmL)$	84%

Eichhorn and Chapelle 2004

Object recognition results

Caltech objects database
101 object classes

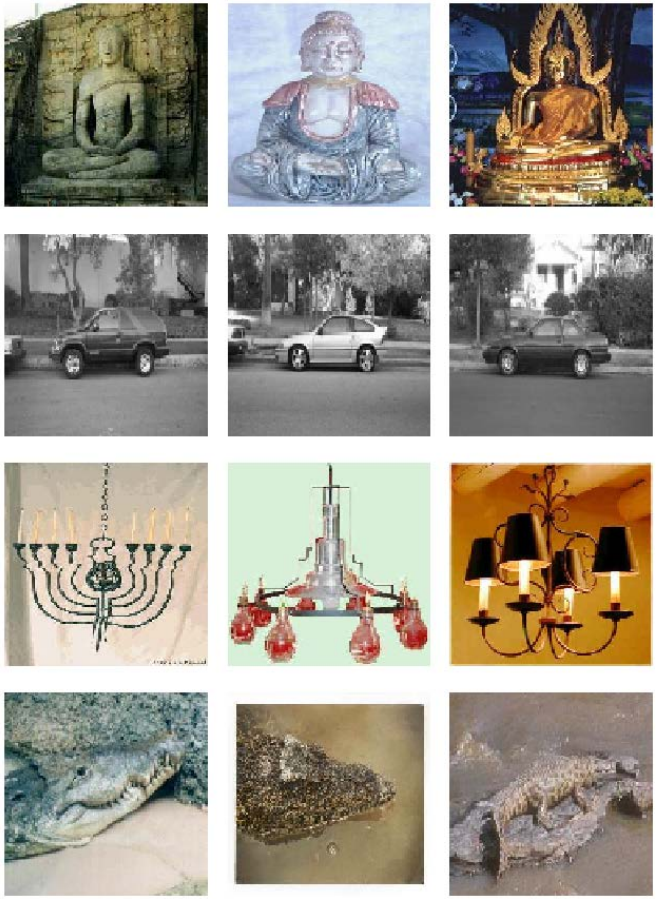
Features:

- SIFT detector
- PCA-SIFT descriptor, *d*=10

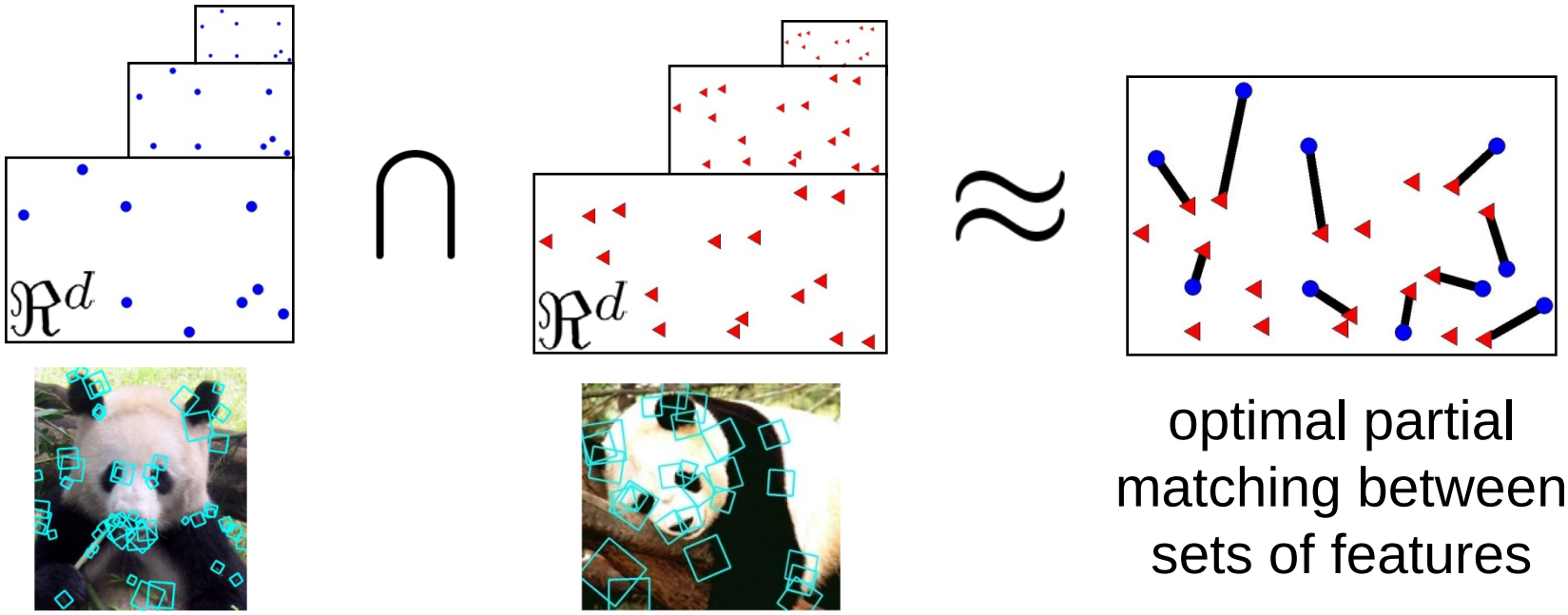
30 training images / class

43% recognition rate
(1% chance performance)

0.002 seconds per match



Summary: Pyramid match kernel



optimal partial matching between sets of features

$$K_\Delta\left(\Psi(\mathbf{X}), \Psi(\mathbf{Y})\right) = \sum_{i=0}^{L} \underbrace{\frac{1}{2^i}}_{\text{difficulty of a match at level i}} \underbrace{\left(\mathcal{I}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right) - \mathcal{I}\left(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})\right)\right)}_{\text{number of new matches at level i}}$$

Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories

To appear in CVPR 2006

Svetlana Lazebnik (slazebni@uiuc.edu)
Beckman Institute, University of Illinois at Urbana-Champaign

Cordelia Schmid (cordelia.schmid@inrialpes.fr)
INRIA Rhône-Alpes, France

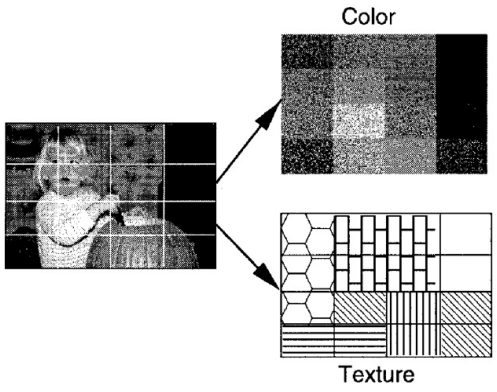
Jean Ponce (ponce@di.ens.fr)
Ecole Normale Supérieure, France

http://www-cvr.ai.uiuc.edu/ponce_grp

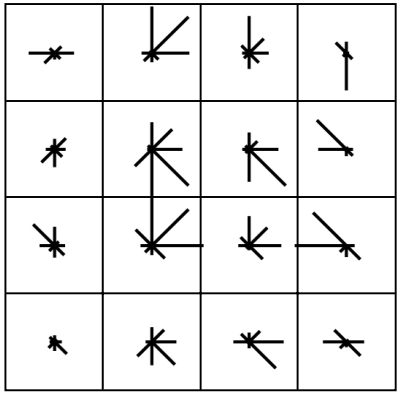
Overview

- A “pre-attentive” approach: recognize the scene as a whole without examining its constituent objects Biederman (1988), Thorpe et al. (1996), Fei-Fei et al. (2002), Renninger & Malik (2004)
- Inspiration: *locally orderless images* Koenderink & Van Doorn (1999)

- Previous work: “subdivide-and-disorder” strategy



Szummer & Picard (1997)

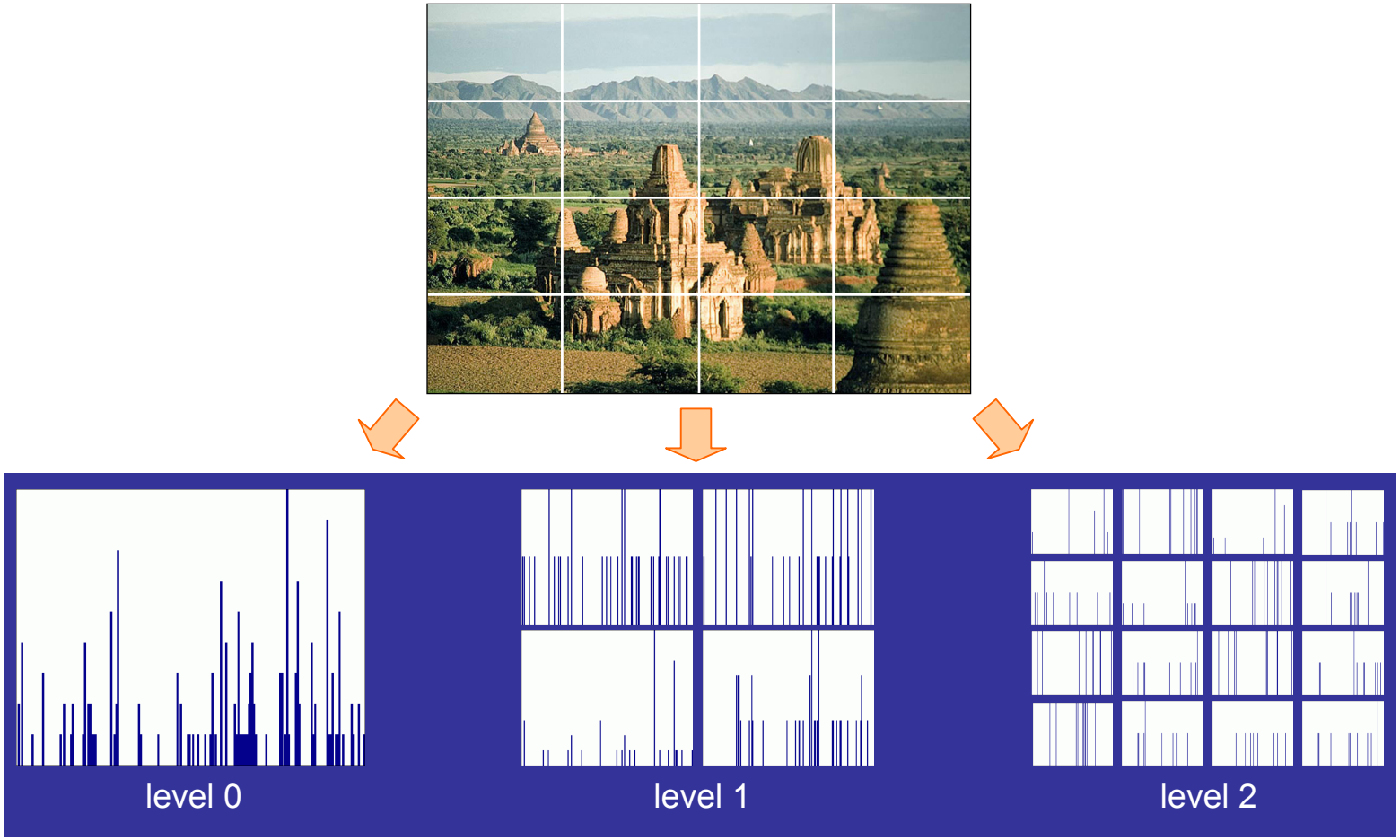


SIFT: Lowe (1999, 2004)

Gist: Torralba et al. (2003)

Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution
- Based on *pyramid match kernels* Grauman & Darrell (2005)
  - Grauman & Darrell: build pyramid in feature space, discard spatial information
  - Our approach: build pyramid in image space, quantize feature space



Pyramid matching

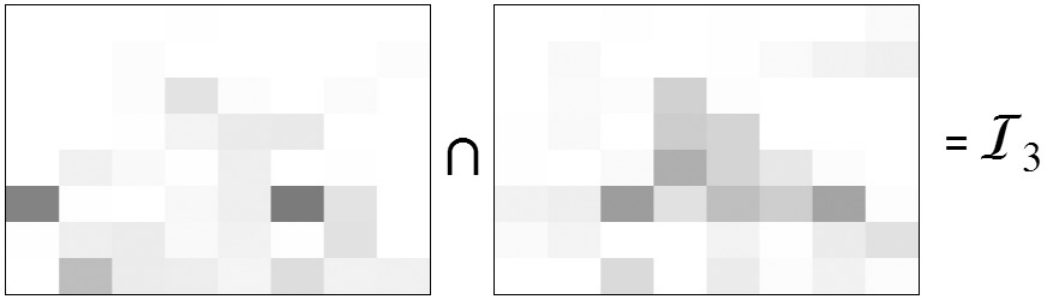
Indyk & Thaper (2003), Grauman & Darrell (2005)

Find maximum-weight matching (weight is inversely proportional to distance)

Original images

Feature histograms:

Level 3



Level 2

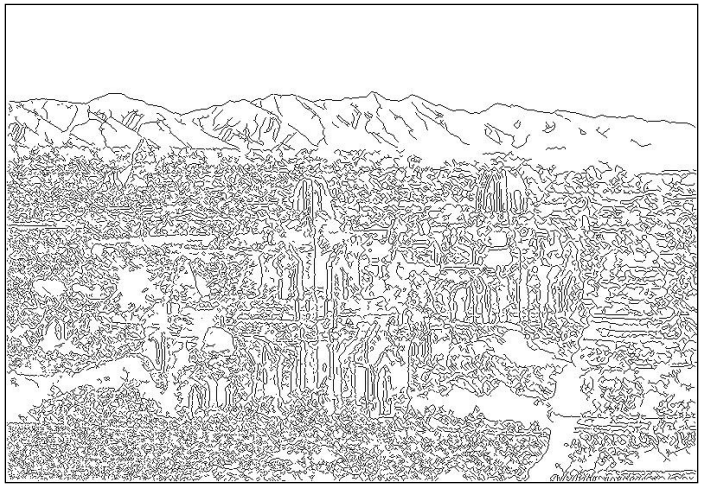
Level 1

Level 0

Total weight (value of *pyramid match kernel*): $\mathcal{I}_3 + \frac{1}{2}(\mathcal{I}_2 - \mathcal{I}_3) + \frac{1}{4}(\mathcal{I}_1 - \mathcal{I}_2) + \frac{1}{8}(\mathcal{I}_0 - \mathcal{I}_1)$

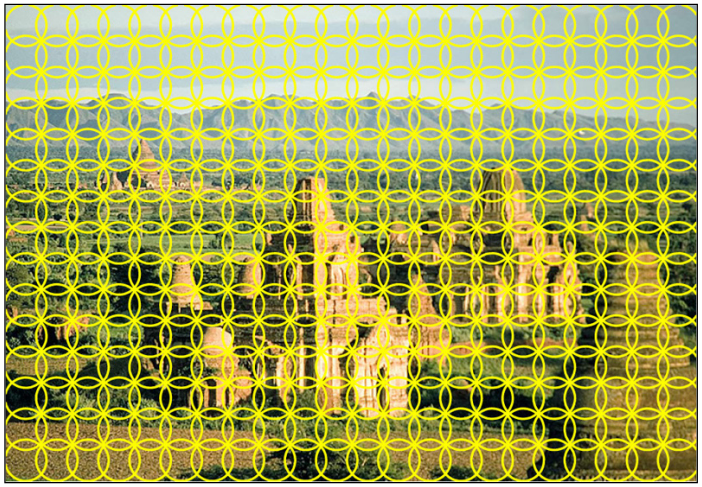
Feature extraction

Weak features



Edge points at 2 scales and 8 orientations (vocabulary size 16)

Strong features



SIFT descriptors of 16x16 patches sampled on a regular grid, quantized to form visual vocabulary (size 200, 400)

Scene category dataset

Fei-Fei & Perona (2005), Oliva & Torralba (2001)

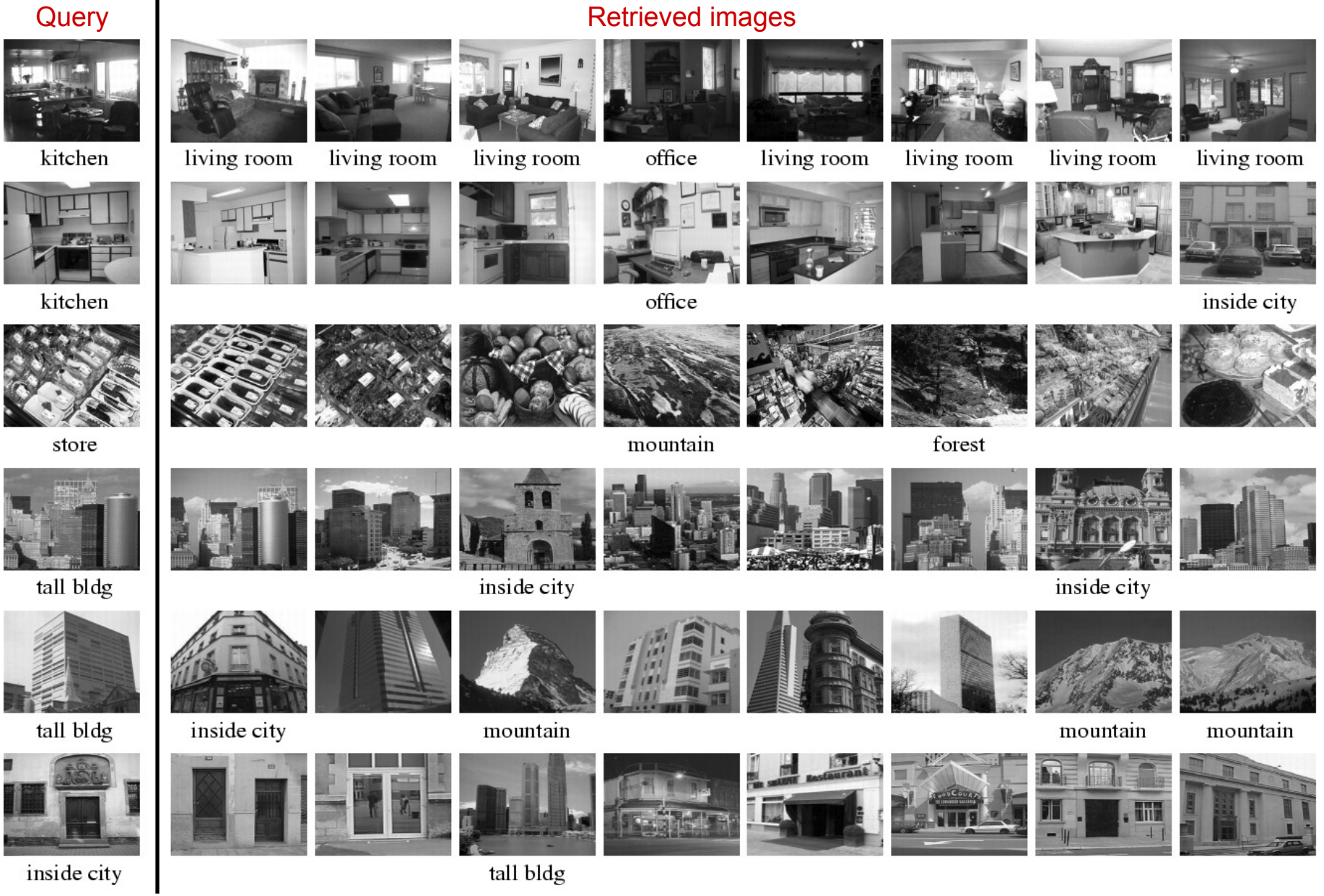
http://www-cvr.ai.uiuc.edu/ponce_grp/data

Multi-class classification results (100 training images per class)

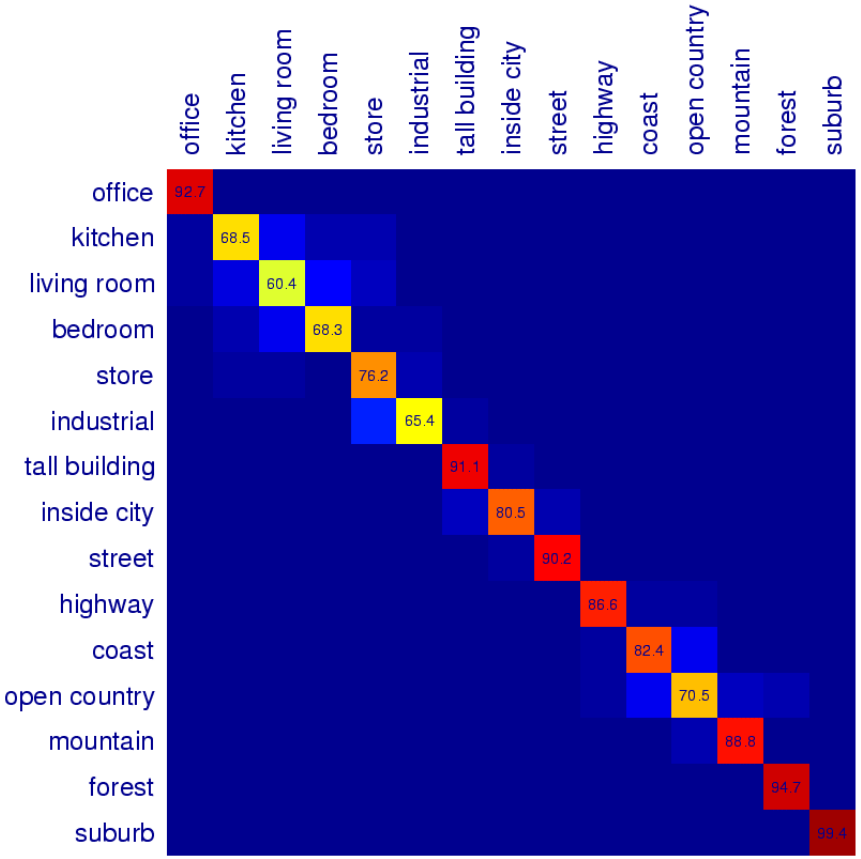
	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0 (1×1)	45.3 ±0.5		72.2 ±0.6	
1 (2×2)	53.6 ±0.3	56.2 ±0.6	77.9 ±0.6	79.0 ±0.5
2 (4×4)	61.7 ±0.6	64.7 ±0.7	79.4 ±0.3	**81.1** ±0.3
3 (8×8)	63.3 ±0.8	**66.8** ±0.6	77.2 ±0.4	80.7 ±0.3

Fei-Fei & Perona: 65.2%

Scene category retrieval



Scene category confusions



Difficult indoor images

kitchen

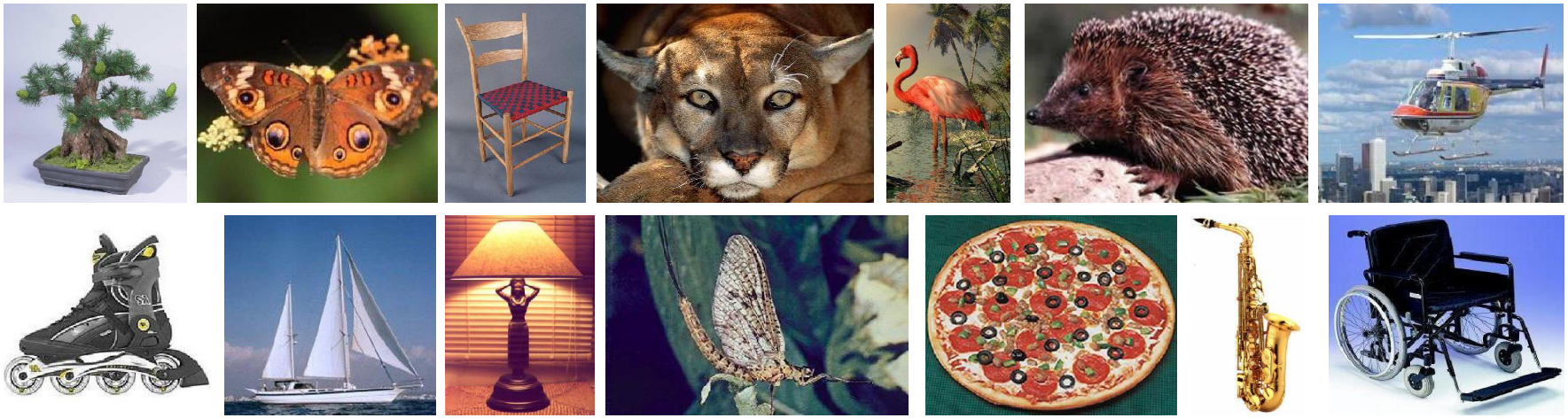
living room

bedroom

Caltech101 dataset

Fei-Fei et al. (2004)

http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html

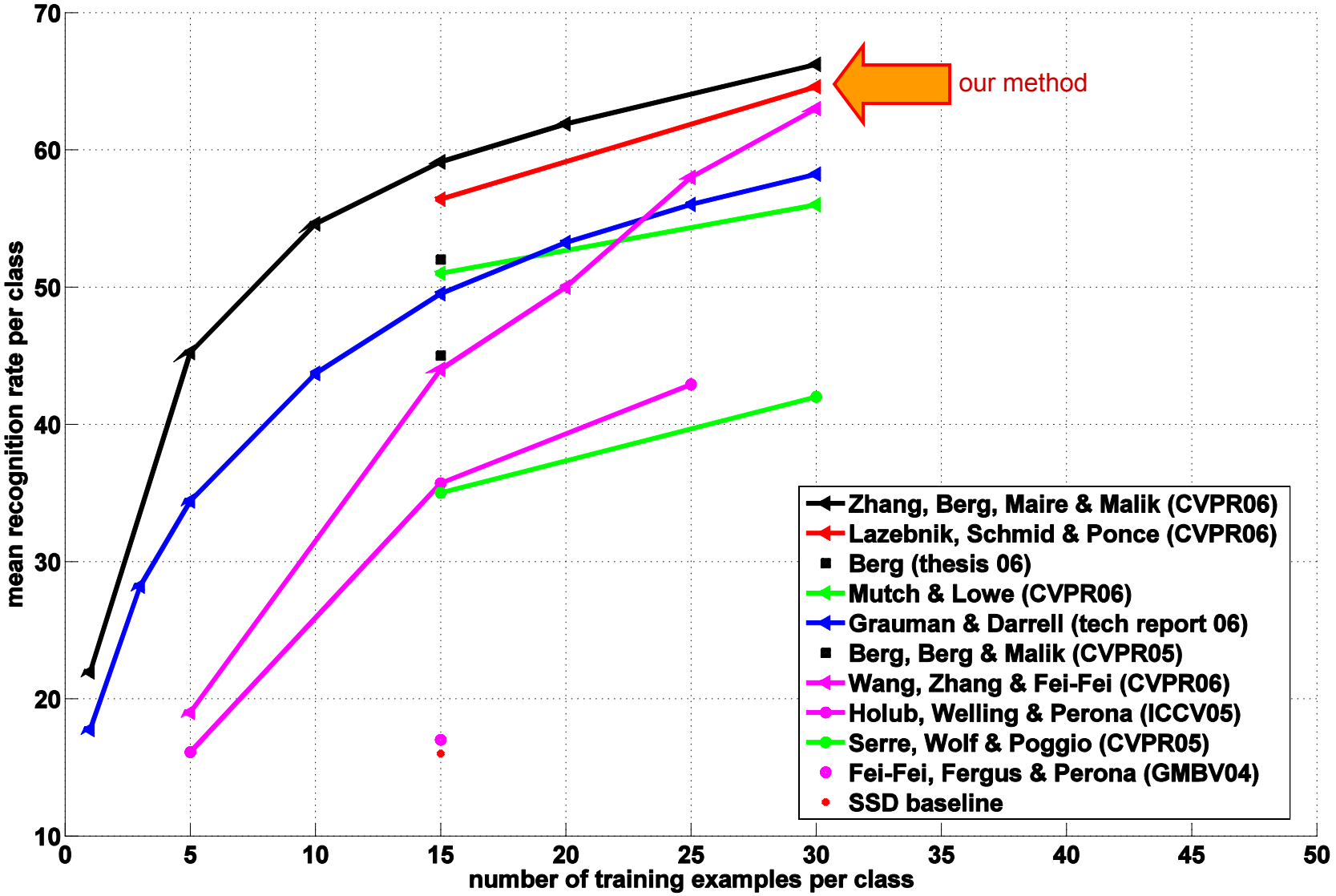


Multi-class classification results (30 training images per class)

	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

Caltech101 comparison

Zhang, Berg, Maire & Malik, 2006

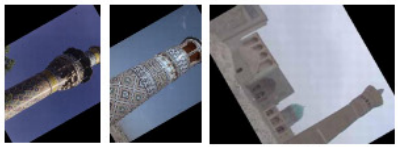


Caltech101 challenges

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

Graz dataset

Opelt et al. (2004)

`http://www.emt.tugraz.at/~pinz/data/`

bike

person

background

Detection results (100 pos./100 neg. training images)

Class	Spatial pyramid ($M = 200$) $L = 0$	Spatial pyramid ($M = 200$) $L = 2$	Opelt et al. (2004)	Zhang et al. (2005)
Bikes	82.4 ±2.0	86.3 ±2.5	86.5	**92.0**
People	79.5 ±2.3	82.3 ±3.1	80.8	**88.0**

bag-of-features methods

- Global spatial regularities (natural scene statistics) help even in databases with high geometric variability!