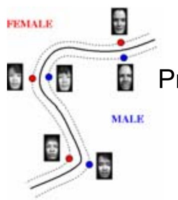
Lecture 18: Recognition IV



Thursday, Nov 15

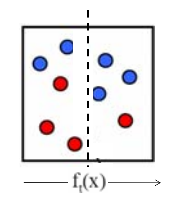
Prof. Kristen Grauman

Outline

- Discriminative classifiers
  - SVMs
- Learning categories from weakly supervised images
  - Constellation model
- Shape matching
  - Shape context, visual CAPTCHA application

Recall: boosting

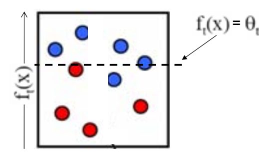
- Want to select the single feature that best separates positive and negative examples, in terms of weighted error.



Each dimension: output of a possible rectangle feature on faces and non-faces.

Recall: boosting

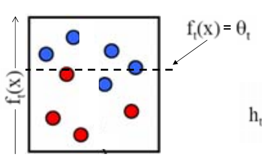
- Want to select the single feature that best separates positive and negative examples, in terms of weighted error.



Each dimension: output of a possible rectangle feature on faces and non-faces.

Recall: boosting

- Want to select the single feature that best separates positive and negative examples, in terms of weighted error.



Optimal threshold that results in minimal misclassifications

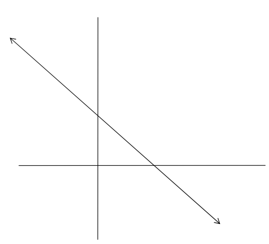
$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

Image subwindow

Each dimension: output of a possible rectangle feature on faces and non-faces.

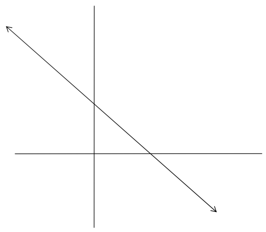
Notice that any threshold giving same error rate would be equally good here.

Lines in R^2



$$ax + by + d = 0$$

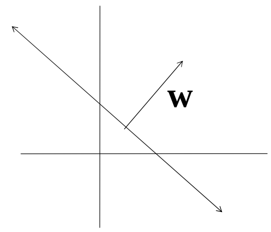
Lines in R²



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + d = 0$$

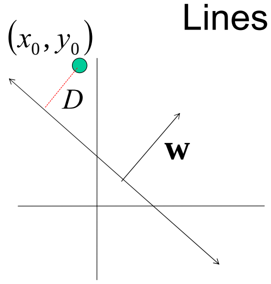
Lines in R²



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + d = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

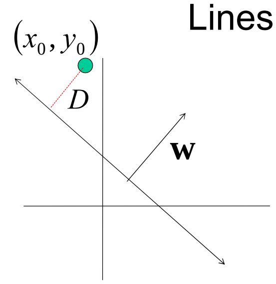
Lines in R²



Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + d = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

Lines in R²



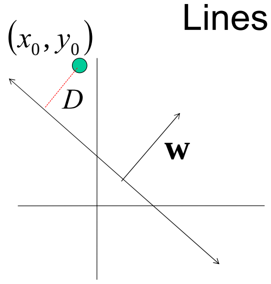
Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + d = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|ax_0 + by_0 + d|}{\sqrt{a^2 + b^2}}$$

distance from point to line

Lines in R²



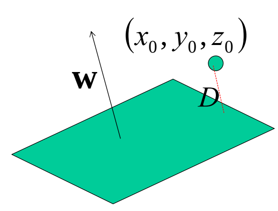
Let $\mathbf{w} = \begin{bmatrix} a \\ b \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + by + d = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|ax_0 + by_0 + d|}{\sqrt{a^2 + b^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + d}{|\mathbf{w}|}$$

distance from point to line

Planes in R³



Let $\mathbf{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$ax + by + cz + d = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + d}{|\mathbf{w}|}$$

distance from point to plane

Hyperplanes in R^n

Hyperplane H is set of all vectors $\mathbf{x} \in R^n$ which satisfy:

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b = 0$$

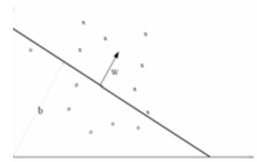
$$\updownarrow$$

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$$

$$D(H, \mathbf{x}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + b}{|\mathbf{w}|}$$ distance from point to hyperplane

Support Vector Machines (SVMs)

- Discriminative classifier based on *optimal separating hyperplane*
- What hyperplane is optimal?

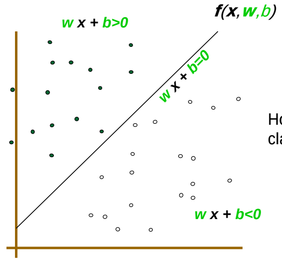


Linear Classifiers

$x \rightarrow \boxed{f} \rightarrow y^{est}$

- denotes +1
- denotes -1

$f(x,w,b) = sign(w\ x + b)$



How would you classify this data?

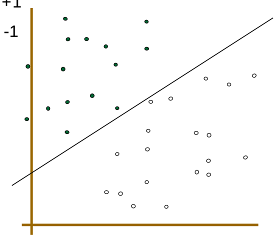
Slides from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Linear Classifiers

$x \rightarrow \boxed{f} \rightarrow y^{est}$

- denotes +1
- denotes -1

$f(x,w,b) = sign(w\ x + b)$



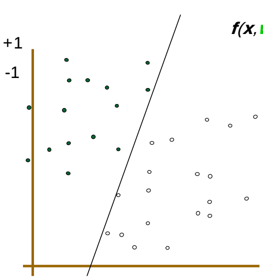
How would you classify this data?

Linear Classifiers

$x \rightarrow \boxed{f} \rightarrow y^{est}$

- denotes +1
- denotes -1

$f(x,w,b) = sign(w\ x + b)$



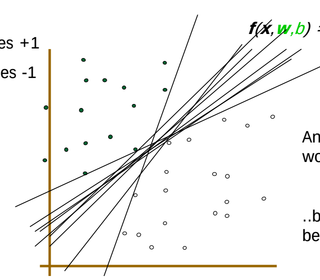
How would you classify this data?

Linear Classifiers

$x \rightarrow \boxed{f} \rightarrow y^{est}$

- denotes +1
- denotes -1

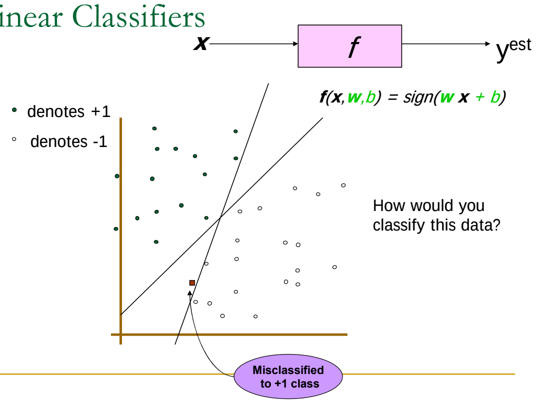
$f(x,w,b) = sign(w\ x + b)$



Any of these would be fine..

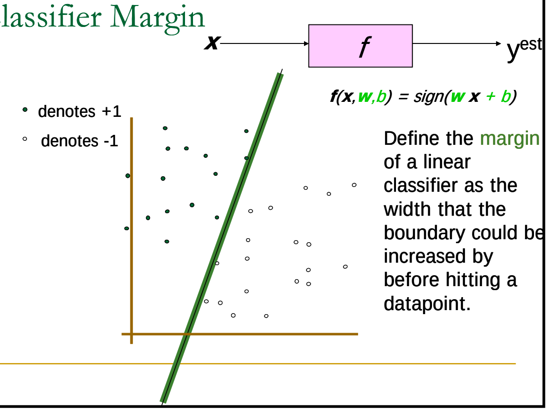
..but which is best?

Linear Classifiers



How would you classify this data?

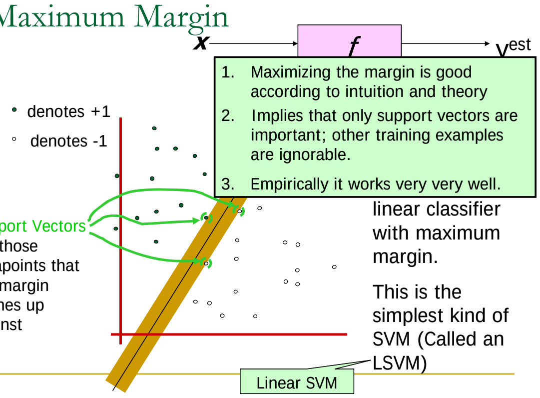
Classifier Margin



Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin

1. Maximizing the margin is good according to intuition and theory
2. Implies that only support vectors are important; other training examples are ignorable.
3. Empirically it works very very well.

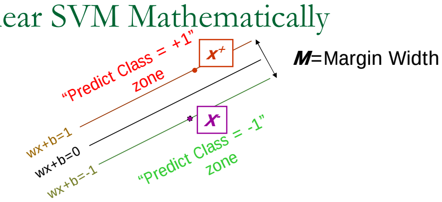


Support Vectors are those datapoints that the margin pushes up against

The maximum margin linear classifier is the linear classifier with maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Linear SVM Mathematically



For the support vectors, distance to hyperplane is 1 for a positives and -1 for negatives.

$$\frac{\mathbf{w}^T\mathbf{x}+b}{|\mathbf{w}|} = \frac{\pm 1}{|\mathbf{w}|} \qquad M = \left|\frac{1}{|\mathbf{w}|} - \frac{-1}{|\mathbf{w}|}\right| = \frac{2}{|\mathbf{w}|}$$

Question

- How should we choose values for **w,b?**

1. want the training data separated by the hyperplane so it classifies them correctly

2. want the margin width *M* as large as possible

Linear SVM Mathematically

- Goal: **1) Correctly classify all training data**

$$wx_i + b \geq 1 \quad \text{if } y_i = +1$$
$$wx_i + b \leq 1 \quad \text{if } y_i = -1$$
$$y_i(wx_i + b) \geq 1 \quad \text{for all } i$$

2) Maximize the Margin $M = \frac{2}{|w|}$

same as minimize $\frac{1}{2}w^t w$

- **Formulated as a Quadratic Optimization Problem, solve for w and b:**

- Minimize $\Phi(w) = \frac{1}{2}w^t w$

  subject to $y_i(wx_i + b) \geq 1 \quad \forall i$

The Optimization Problem Solution

- Solution has the form (omitting derivation):

$$\boxed{\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i} \qquad b = y_k - \mathbf{w}^T\mathbf{x_k} \text{ for any } \mathbf{x_k} \text{ such that } \alpha_k \neq 0}$$

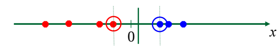
- Each non-zero α_i indicates that corresponding $\mathbf{x_i}$ is a support vector.
- Then the classifying function will have the form:

$$\boxed{f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^T\mathbf{x} + b}$$

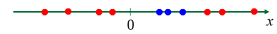
- Notice that it relies on an *inner product* between the test point **x** and the support vectors $\mathbf{x_i}$
- Solving the optimization problem also involves computing the inner products $\mathbf{x_i}^T\mathbf{x_j}$ between all pairs of training points.

Non-linear SVMs

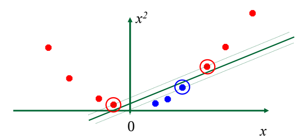
- Datasets that are linearly separable with some noise work out great:



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

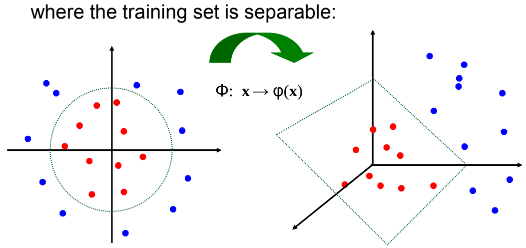


- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: Feature spaces

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



The "Kernel Trick"

- The linear classifier relies on dot product between vectors $K(x_i,x_j)=x_i^Tx_j$
- If every data point is mapped into high-dimensional space via some transformation Φ: x → φ(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.
- Example:

2-dimensional vectors $x=[x_1 \ \ x_2]$; let $K(x_i,x_j)=(1 + x_i^Tx_j)^2$

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

$$K(x_i,x_j)=(1 + x_i^Tx_j)^2,$$
$$= 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$$
$$= [1 \ \ x_{i1}^2 \ \ \sqrt{2}\, x_{i1}x_{i2} \ \ x_{i2}^2 \ \ \sqrt{2}x_{i1} \ \ \sqrt{2}x_{i2}]^T [1 \ \ x_{j1}^2 \ \ \sqrt{2}\, x_{j1}x_{j2} \ \ x_{j2}^2 \ \ \sqrt{2}x_{j1} \ \ \sqrt{2}x_{j2}]$$
$$= \varphi(x_i)^T\varphi(x_j), \quad \text{where } \varphi(x) = [1 \ \ x_1^2 \ \ \sqrt{2}\, x_1x_2 \ \ x_2^2 \ \ \sqrt{2}x_1 \ \ \sqrt{2}x_2]$$

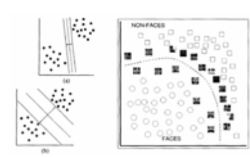
Examples of General Purpose Kernel Functions

- Linear: $K(\mathbf{x_i},\mathbf{x_j})= \mathbf{x_i}^T\mathbf{x_j}$
- Polynomial of power p: $K(\mathbf{x_i},\mathbf{x_j})= (1+ \mathbf{x_i}^T\mathbf{x_j})^p$
- Gaussian (radial-basis function network):

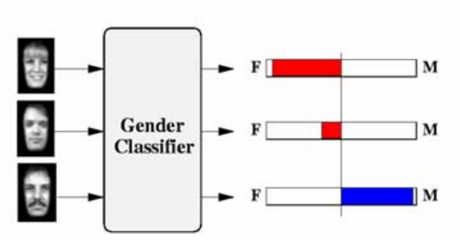
$$K(\mathbf{x_i},\mathbf{x_j}) = \exp\left(-\frac{\|\mathbf{x_i}-\mathbf{x_j}\|^2}{2\sigma^2}\right)$$

SVMs for object recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples, identify support vectors.
4. Compute kernel values between new inputs and support vectors to classify.



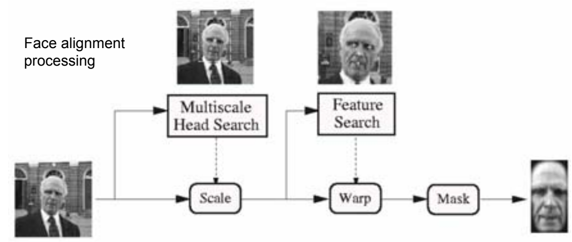
Example: learning gender with SVMs



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

Moghaddam and Yang, Face & Gesture 2000.

Face alignment processing



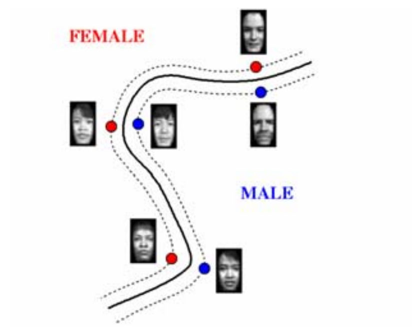
Processed faces

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

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.

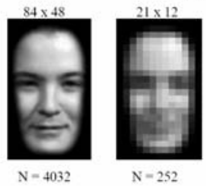
Gender perception experiment: How well can humans do?

- 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 →



Results →

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

$\sigma = 3.7\%$

Moghaddam and Yang, Face & Gesture 2000

Human vs. Machine

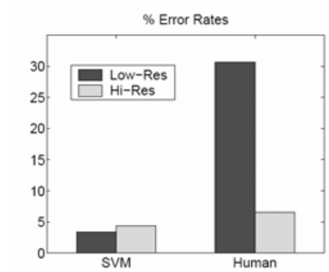
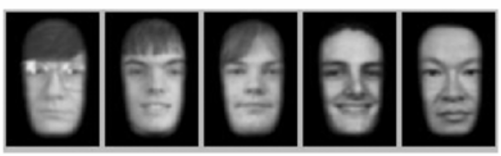


Figure 6. SVM vs. Human performance

- SVMs perform better than any single human text subject

Hardest examples for humans



Top five human misclassifications

True classification: F, M, M, F, M

Moghaddam and Yang, Face & Gesture 2000.

Summary: SVM classifiers

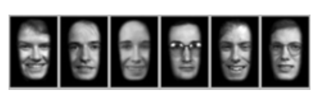
- Discriminative classifier
- Effective for high-dimesional data
- Flexibility/modularity due to kernel
- Very good performance in practice, widely used in vision applications

Outline

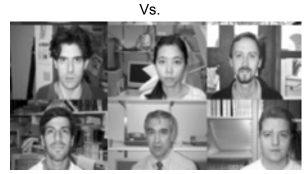
- Discriminative classifiers
  - SVMs
- Learning categories from weakly supervised images
  - Constellation model
- Shape matching
  - Shape context, visual CAPTCHA application

Weak supervision

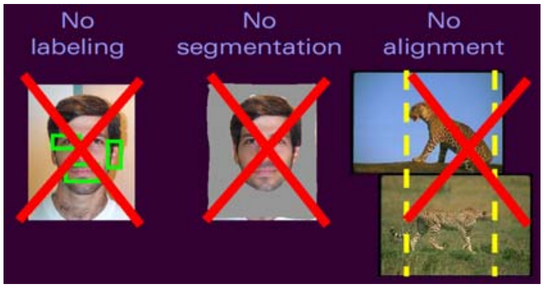
- How can we learn object models in the presence of clutter?



Vs.



Goal



Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

Weak supervision

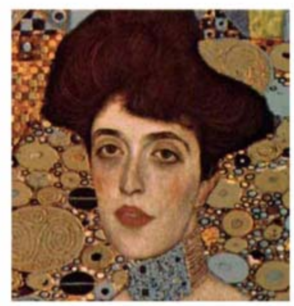
- Questions:
  – What about categories where an iconic “template” representation is infeasible?
  – What is the object to be recognized / the part of the image we want to build a model for?
  – For that object, what parts are distinctive or things that can be reliably detected in different instances?

Weber, Welling, Perona. Unsupervised Learning of Models for Recognition, ECCV 2000.

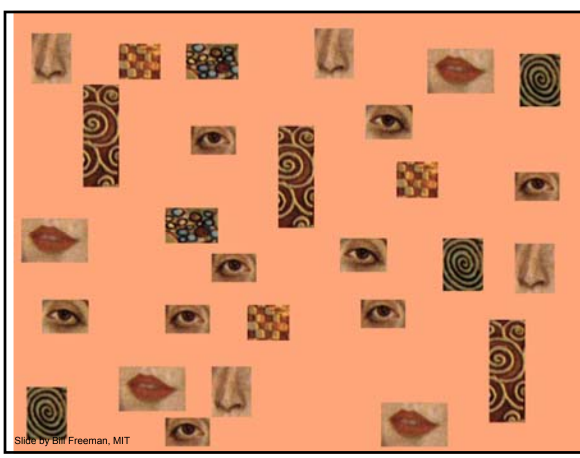
Fig. 1. Which objects appear consistently in the left images, but not on the right side? Can a machine learn to recognize instances of the two object classes (*faces* and *cars*) without any further information provided?

Weber, Welling, Perona., 2000.

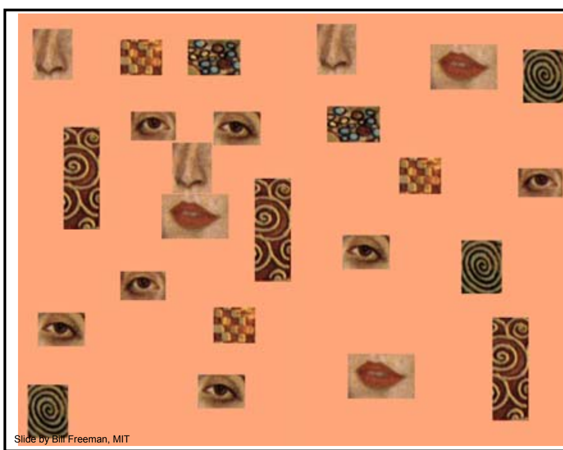
What are the features that let us recognize that this is a face?



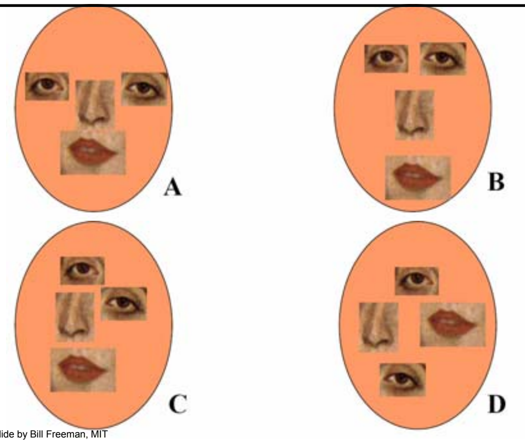
Slide by Bill Freeman, MIT



Slide by Bill Freeman, MIT

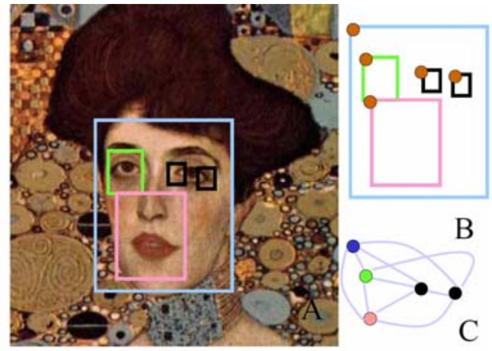


Slide by Bill Freeman, MIT



Slide by Bill Freeman, MIT

Part-based models



Slide by Bill Freeman, MIT

Part-based models

Slide by Fei-Fei Li, 2003.

One possible constellation model

- Model class with joint probability density function on shape and appearance

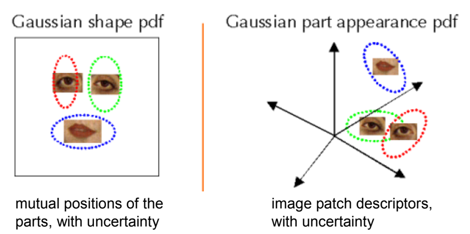


Figure from Rob Fergus

Unsupervised learning of part-based models

Main idea:

- Use interest operator to detect small highly textured regions (on both fg and bg)
  - If training objects have similar appearance, these regions will often be similar in different training examples
- Cluster patches: large clusters used to select candidate fg parts
- Choose most informative parts while simultaneously estimating model parameters
  - Iteratively try different combinations of a small number of parts and check model performance on validation set to evaluate quality

Weber, Welling, Perona, ECCV 2000.

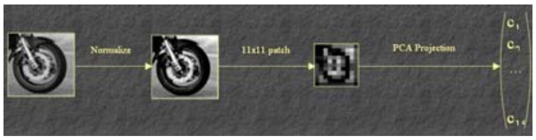
Representation

- Use a scale invariant, scale sensing feature keypoint detector (like the first steps of Lowe's SIFT).

From: Rob Fergus http://www.robots.ox.ac.uk/%7Efergus/

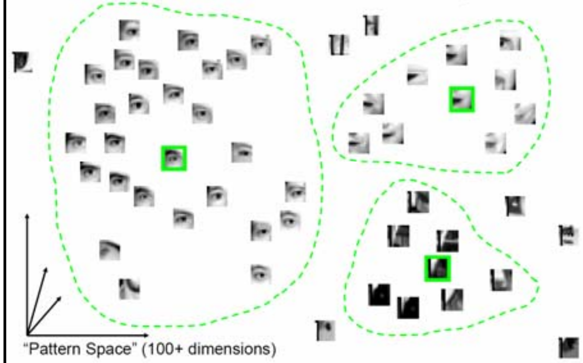
Features Keys

- A direct appearance model is taken around each located key. This is then normalized to an 11x11 window. PCA further reduces these features.



From: Rob Fergus http://www.robots.ox.ac.uk/%7Efergus/

Unsupervised detector training - 2



Slide by Bill Freeman, MIT

Candidate parts

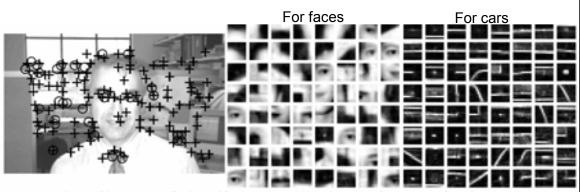
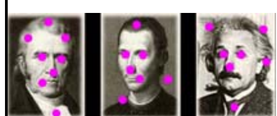


Fig. 3. Points of interest (left) identified on a training image of a human face in cluttered background using Förstner's method. Crosses denote corner-type patterns while circles mark circle-type patterns. A sample of the patterns obtained using k-means clustering of small image patches is shown for faces (center) and cars (right). The car images were high-pass filtered before the part selection process. The total number of patterns selected were 81 for faces and 80 for cars.

At this point, parts appear in both background and foreground of training images.

Weber, Welling, Perona. Unsupervised Learning of Models for Recognition, 2000.

Model learning



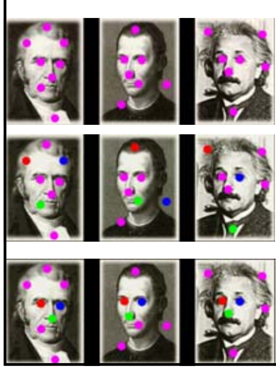
Which of the candidate parts define the class, and in what configuration?

Let's assume:

- We know number of parts that define the model (and can keep it small).
- Object of interest is only consistent thing somewhere in each training image.

Images from Rob Fergus

Model learning



Which of the candidate parts define the class, and in what configuration?

Initialize model parameters randomly.

Iterate while fit improves:

1. Find best assignment in the training images given the parameters
2. Recompute parameters based on current features

Recognition

- Given a model defining the object class and a model for "background", compute likelihood ratio to make Bayesian decision:

Identified in new image:

- X: locations
- S: scales
- A: appearances

$$R = \frac{p(\text{Object}|\mathbf{X},\mathbf{S},\mathbf{A})}{p(\text{No object}|\mathbf{X},\mathbf{S},\mathbf{A})}$$

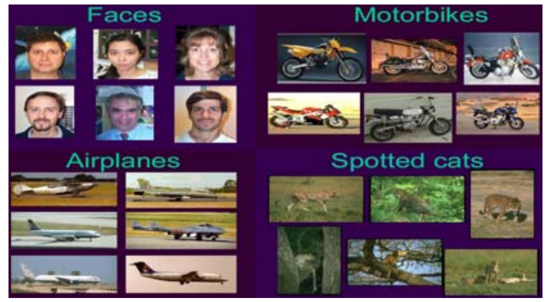
Recognition

- Given a model defining the object class and a model for "background", compute likelihood ratio to make Bayesian decision:

X: locations
S: scales
A: appearances

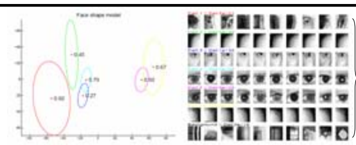
$$
\begin{aligned}
R &= \frac{p(\text{Object}|\mathbf{X},\mathbf{S},\mathbf{A})}{p(\text{No object}|\mathbf{X},\mathbf{S},\mathbf{A})} \\
&= \frac{p(\mathbf{X},\mathbf{S},\mathbf{A}|\text{Object})\;p(\text{Object})}{p(\mathbf{X},\mathbf{S},\mathbf{A}|\text{No object})\;p(\text{No object})}
\end{aligned}
$$

Example: data from four categories



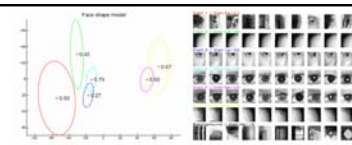
Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

Face model



Appearance: 10 patches closest to mean for each part

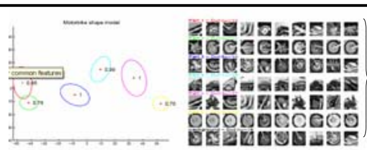
Face model



Appearance: 10 patches closest to mean for each part

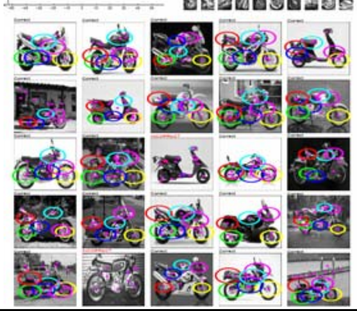
Recognition results

Motorbike model

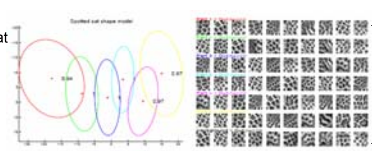


Appearance: 10 patches closest to mean for each part

Recognition results



Spotted cat model



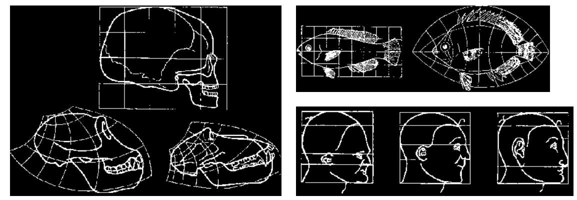
Appearance: 10 patches closest to mean for each part

Recognition results

Outline

- Discriminative classifiers
  - SVMs
- Learning categories from weakly supervised images
  - Constellation model
- Shape matching
  - Shape context, visual CAPTCHA application

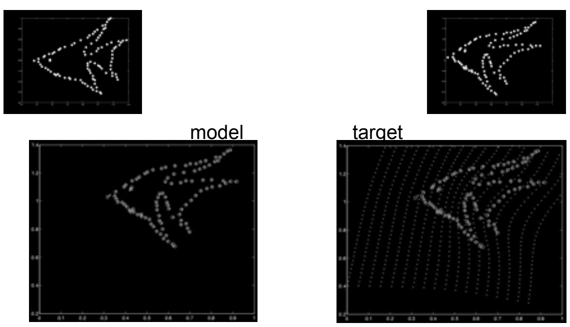
Shape and biology



- D'Arcy Thompson: *On Growth and Form*, 1917
  - studied transformations between shapes of organisms

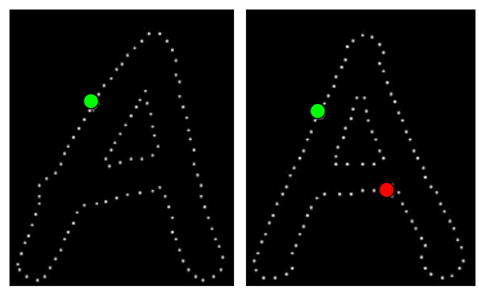
Slides adapted from Belongie, Malik, & Puzicha, Matching Shapes, ICCV 2001.
www.eecs.berkeley.edu/Research/Projects/CS/vision/shape/belongie-iccv01

Shape matching for recognition



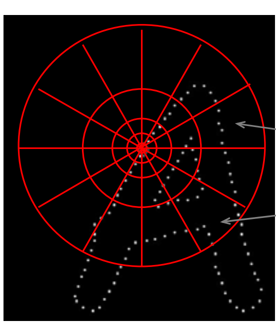
model target

Comparing shapes



What points on these two sampled contours are most similar?

Shape context descriptor



Count the number of points inside each bin, e.g.:

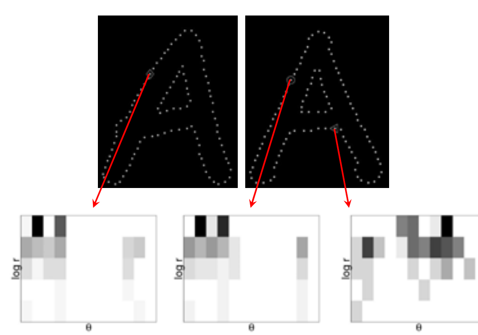
Count = 4

⋮

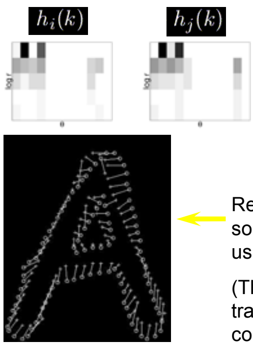
Count = 10

Compact representation of distribution of points relative to each point

Shape context descriptor



Comparing shape contexts



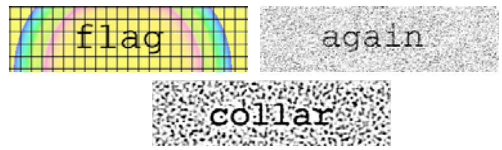
$$C_{ij} = \frac{1}{2}\sum_{k=1}^{K}\frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)}$$

Recover correspondences by solving for least cost assignment, using costs C_{ij}

(Then estimate a parameterized transformation based on these correspondences.)

CAPTCHA's

- CAPTCHA: Completely Automated Turing Test To Tell Computers and Humans Apart
- Luis von Ahn, Manuel Blum, Nicholas Hopper and John Langford, CMU, 2000.
- www.captcha.net

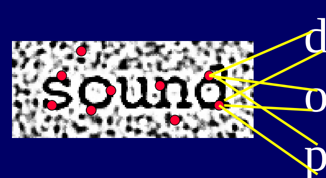


Shape matching application: breaking a visual CAPTCHA

- Use shape matching to recognize characters, words in spite of clutter, warping, etc.

Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA, by G. Mori and J. Malik, CVPR 2003

Fast Pruning: Representative Shape Contexts

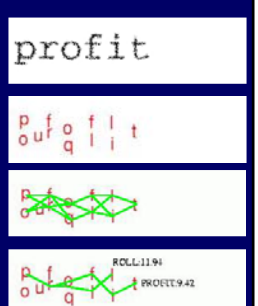


- Pick k points in the image at random
  - Compare to all shape contexts for all known letters
  - Vote for closely matching letters
- Keep all letters with scores under threshold

University of California
Berkeley
Slides by Greg Mori, CVPR 2003
Computer Vision Group

Algorithm A: bottom-up

- Look for letters
  - Representative Shape Contexts
- Find pairs of letters that are "consistent"
  - Letters nearby in space
- Search for valid words
- Give scores to the words



University of California
Berkeley
Computer Vision Group

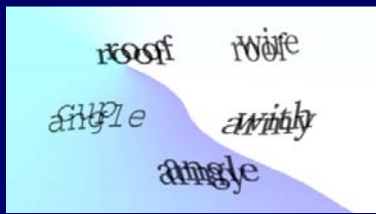
EZ-Gimpy Results with Algorithm A

- 158 of 191 images correctly identified: 83%
  - Running time: ~10 sec. per image (MATLAB, 1 Ghz P3)

horse
spade
smile
join
canvas
here

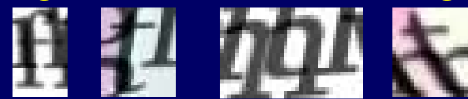
University of California
Berkeley
Computer Vision Group

Gimpy



- Multiple words, task is to find 3 words in the image
- Clutter is other objects, not texture

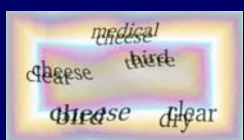
Algorithm B: Letters are not enough



- Hard to distinguish single letters with so much clutter
- Find words instead of letters
  - Use long range info over entire word
  - Stretch shape contexts into ellipses

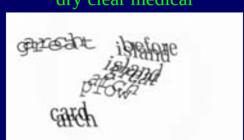
- Search problem becomes huge
  - # of words 600 vs. # of letters 26
  - Prune set of words using opening/closing bigrams

Results with Algorithm B

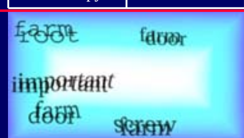


dry clear medical

# Correct words	% tests (of 24)
1 or more	92%
2 or more	75%
3	33%
EZ-Gimpy	92%



card arch plate



door farm important

Coming up

- Face images
- For next week:
  - Read Trucco & Verri handout on Motion
- Problem set 4 due 11/29

References

- Unsupervised Learning of Models for Recognition, by M. Weber, M. Welling and P. Perona, ECCV 2000.
- Towards Automatic Discovery of Object Categories, by M. Weber, M. Welling and P. Perona, CVPR 2000.
- Object Class Recognition by Unsupervised Scale-Invariant Learning, by Fergus, Perona, and Zisserman, CVPR 2003.
- Matching Shapes, by S. Belongie, J. Malik and J. Puzicha, ICCV 2001.
- Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA, by G. Mori and J. Malik, CVPR 2003.
- Learning Gender with Support Faces, by Moghaddam and Yang, TPAMI, 2002.
- SVM slides from Andrew Moore, CMU