Supervised object recognition, unsupervised object recognition then Perceptual organization

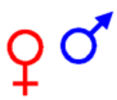
Bill Freeman, MIT

6.869 April 12, 2005

Readings

- Brief overview of classifiers in context of gender recognition:
  - http://www.merl.com/reports/docs/TR2000-01.pdf, ***Gender Classification with Support Vector Machines*** Citation: Moghaddam, B.; Yang, M-H., "Gender Classification with Support Vector Machines", *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pps 306-311, March 2000
- Overview of support vector machines—Statistical Learning and Kernel MethodsBernhard Schölkopf, ftp://ftp.research.microsoft.com/pub/tr/tr-2000-23.pdf
- M. Weber, M. Welling and P. Perona
  Proc. 6th Europ. Conf. Comp. Vis., ECCV, Dublin, Ireland, June 2000
  ftp://vision.caltech.edu/pub/tech-reports/ECCV00-recog.pdf

Gender Classification with Support Vector Machines



Baback Moghaddam

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Support vector machines (SVM's)

- The 3 good ideas of SVM's

Good idea #1: Classify rather than model probability distributions.

- Advantages:
  - Focuses the computational resources on the task at hand.
- Disadvantages:
  - Don't know how probable the classification is
  - Lose the probabilistic model for each object class; can't draw samples from each object class.

Good idea #2: Wide margin classification

- For better generalization, you want to use the weakest function you can.
  - Remember polynomial fitting.
- There are fewer ways a wide-margin hyperplane classifier can split the data than an ordinary hyperplane classifier.

Too weak

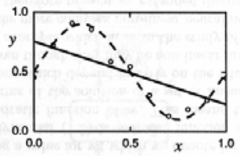


Figure 1.6. An example of a set of 11 data points obtained by sampling the function $h(x)$, defined by (1.4), at equal intervals of x and adding random noise. The dashed curve shows the function $h(x)$, while the solid curve shows the rather poor approximation obtained with a linear polynomial, corresponding to $M = 1$ in (1.2).

Bishop, neural networks for pattern recognition, 1995

Just right

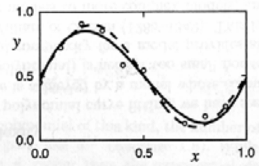


Figure 1.7. This shows the same data set as in Figure 1.6, but this time fitted by a cubic ($M = 3$) polynomial, showing the significantly improved approximation to $h(x)$ achieved by this more flexible function.

Bishop, neural networks for pattern recognition, 1995

Too strong

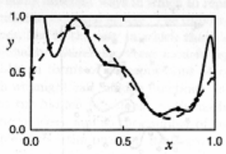


Figure 1.8. The result of fitting the same data set as in Figure 1.6 using a 10th-order ($M = 10$) polynomial. This gives a perfect fit to the training data, but at the expense of a function which has large oscillations, and which therefore gives a poorer representation of the generator function $h(x)$ than did the cubic polynomial of Figure 1.7.

Bishop, neural networks for pattern recognition, 1995

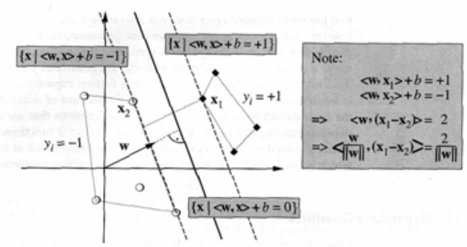


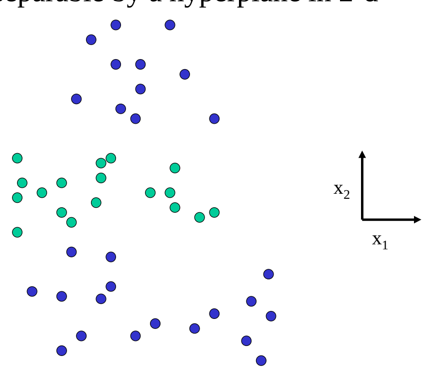
Figure 1.5 A binary classification toy problem: separate balls from diamonds. The *optimal hyperplane* (1.23) is shown as a solid line. The problem being separable, there exists a weight vector $\mathbf{w}$ and a threshold b such that $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0$ $(i = 1, \ldots, m)$. Rescaling $\mathbf{w}$ and b such that the point(s) closest to the hyperplane satisfy $|\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1$, we obtain a *canonical* form $(\mathbf{w}, b)$ of the hyperplane, satisfying $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$. Note that in this case, the *margin* (the distance of the closest point to the hyperplane) equals $1/\|\mathbf{w}\|$. This can be seen by considering two points $\mathbf{x}_1, \mathbf{x}_2$ on opposite sides of the margin, that is, $\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = 1$, $\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$, and projecting them onto the hyperplane normal vector $\mathbf{w}/\|\mathbf{w}\|$.

Learning with Kernels, Scholkopf and Smola, 2002

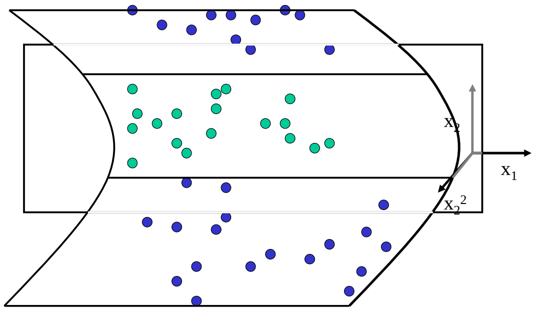
Finding the wide-margin separating hyperplane: a quadratic programming problem, involving inner products of data vectors

Good idea #3: The kernel trick

Non-separable by a hyperplane in 2-d



Separable by a hyperplane in 3-d



Embedding

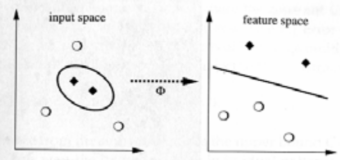


Figure 1.6 The idea of SVMs: map the training data into a higher-dimensional feature space via Φ, and construct a separating hyperplane with maximum margin there. This yields a nonlinear decision boundary in input space. By the use of a kernel function (1.2), it is possible to compute the separating hyperplane without explicitly carrying out the map into the feature space.

Learning with Kernels, Scholkopf and Smola, 2002

The kernel idea

- There are many embeddings where the dot product in the high dimensional space is just the kernel function applied to the dot product in the low-dimensional space.
- For example:
  - $K(x,x') = (<x,x'> + 1)^d$
- Then you "forget" about the high dimensional embedding, and just play with different kernel functions.

Example kernel

$$K(x,x') = (<x,x'> + 1)^d$$

Here, the high-dimensional vector is

$$(x_1, x_2) -> (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2)$$

You can see for this case how the dot product of the high-dimensional vectors is just the kernel function applied to the low-dimensional vectors. Since all we need to find the desired hyperplanes separating the high-dimensional vectors is their dot product, we can do it all with kernels applied to the low-dimensional vectors.

$$K((x_1,x_2),(x_1',x_2')) = (x_1x_1' + x_2x_2' + 1)^2$$ kernel function applied to the low-dimensional vectors

$$= (x_1x_1')^2 + (x_2x_2')^2 + 1 + 2x_1x_1' + 2x_2x_2'$$

dot product of the high-dimensional vectors $$=< (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2), (1, \sqrt{2}x_1', x_1'^2, \sqrt{2}x_2', x_2'^2) >$$

- See also nice tutorial slides http://www.bioconductor.org/workshops/NGFN03/svm.pdf

Example kernel functions

- Polynomials
- Gaussians
- Sigmoids
- Radial basis functions
- Etc…

The hyperplane decision function

$$f(x) = \mathrm{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i (x \cdot x_i) + b\right)$$

Eq. 32 of "statistical learning and kernel methods, MSR-TR-2000-23

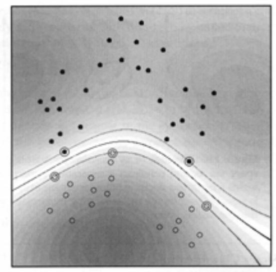
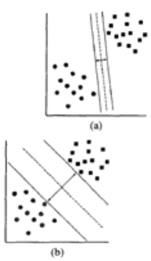
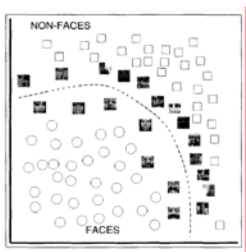


Figure 1.7 Example of an SV classifier found using a radial basis function kernel $k(x, x') = \exp(-\|x - x'\|^2)$ (here, the input space is $\mathcal{X} = [-1, 1]^2$). Circles and disks are two classes of training examples; the middle line is the decision surface; the outer lines precisely meet the constraint (1.25). Note that the SVs found by the algorithm (marked by extra circles) are not centers of clusters, but examples which are critical for the given classification task. Gray values code $|\sum_{i=1}^{m} y_i \alpha_i k(x, x_i) + b|$, the modulus of the argument of the decision function (1.35). The top and the bottom lines indicate places where it takes the value 1 (from [471]).

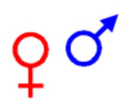
Learning with Kernels, Scholkopf and Smola, 2002

Discriminative approaches: e.g., Support Vector Machines





Gender Classification with Support Vector Machines

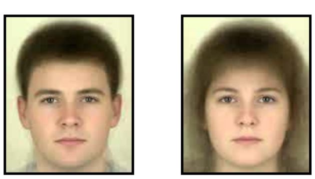


Baback Moghaddam

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

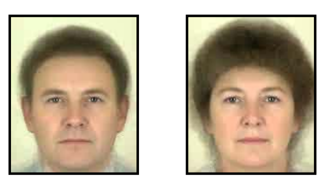
Gender Prototypes



Images courtesy of University of St. Andrews Perception Laboratory

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Gender Prototypes



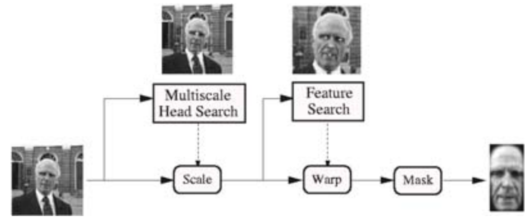
Images courtesy of University of St. Andrews Perception Laboratory

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Classifier Evaluation

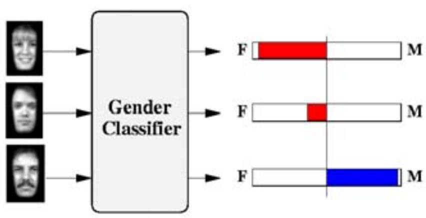
- Compare "standard" classifiers
- 1755 FERET faces
  - 80-by-40 full-resolution
  - 21-by-12 "thumbnails"
- 5-fold Cross-Validation testing
- Compare with human subjects

Face Processor

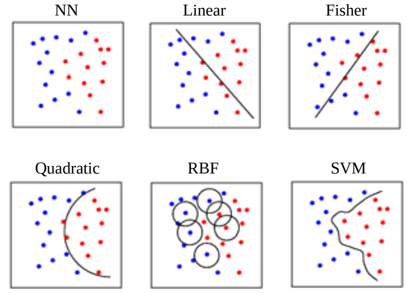


[Moghaddam & Pentland, PAMI-19:7]

Gender (Binary) Classifier



Binary Classifiers

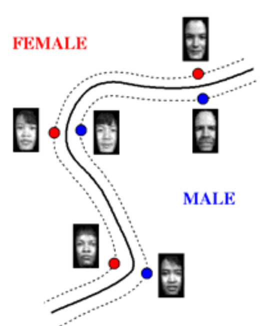


Linear SVM Classifier

- Data: $\{\mathbf{x}_i , y_i\}$ $i = 1,2,3 \ldots N$ $y_i = \{-1,+1\}$
- Discriminant: $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x} + b) > 0$
- minimize $\|\mathbf{w}\|$
- subject to $y_i\,(\mathbf{w} \cdot \mathbf{x}_i + b) > 1$ for all i
- Solution: QP gives $\{\alpha_i\}$
- $\mathbf{w}_{opt} = \Sigma\, \alpha_i\, y_i\, \mathbf{x}_i$
- $f(\mathbf{x}) = \Sigma\, \alpha_i\, y_i\, (\mathbf{x}_i \cdot \mathbf{x}) + b$

Note we just need the vector dot products, so this is easy to "kernelize".

"Support Faces"

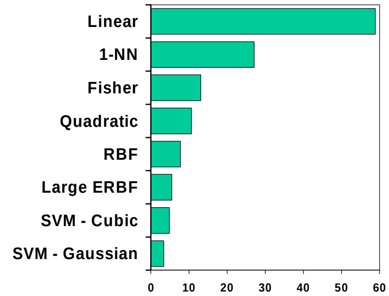


Classifier Performance

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, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Classifier Error Rates



Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Gender Perception Study

- **Mixture**: 22 males, 8 females
- **Age**: mid-20s to mid-40s
- **Stimuli**: 254 faces (randomized)
  - low-resolution 21-by-12
  - high-resolution 84-by-48
- **Task**: classify gender (M or F)
  - forced-choice
  - no time constraints

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

How would you classify these 5 faces?

Top five human misclassifications

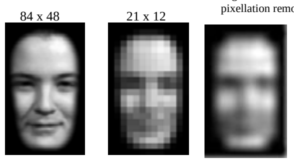
True classification: F, M, M, F, M

Human Performance

But note how the pixellated enlargement hinders recognition. Shown below with pixellation removed

Stimuli →

84 x 48 | 21 x 12



N = 4032 | N = 252

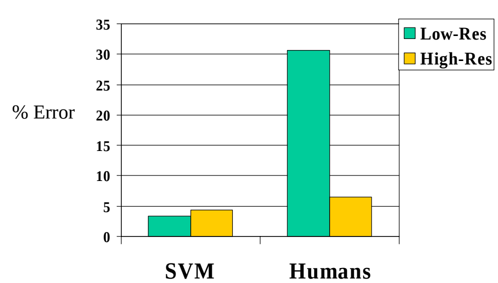
Results →

High-Res	Low-Res
6.54%	30.7%

σ = 3.7%

Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

Machine *vs.* Humans



Moghaddam, B.; Yang, M-H, "Learning Gender with Support Faces", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, May 2002

End of SVM section

6.869

Previously: Object recognition via labeled training sets.
Now: Unsupervised Category Learning
Followed by:
Perceptual organization:
- Gestalt Principles
- Segmentation by Clustering
  - K-Means
  - Graph cuts
- Segmentation by Fitting
  - Hough transform
  - Fitting

Readings: F&P Ch. 14, 15.1-15.2

Unsupervised Learning

- Object recognition methods in last two lectures presume:
  - Segmentation
  - Labeling
  - Alignment
- What can we do with unsupervised (weakly supervised) data?
- See work by Perona and collaborators
  - (the third of the 3 bits needed to characterize all computer vision conference submissions, after SIFT and Viola/Jones style boosting).

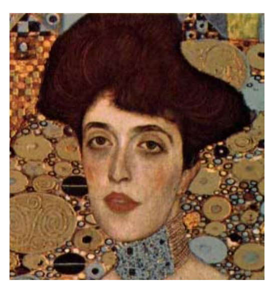
References

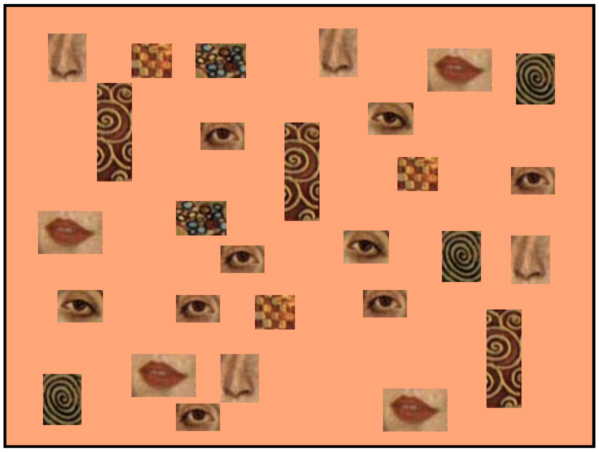
- **Unsupervised Learning of Models for Recognition**
- M. Weber, M. Welling and P. Perona
(15 pages postscript) (15 pages PDF)
Proc. 6th Europ. Conf. Comp. Vis., ECCV, Dublin, Ireland, June 2000
- **Towards Automatic Discovery of Object Categories**
- M. Weber, M. Welling and P. Perona
(8 pages postscript) (8 pages PDF)
Proc. IEEE Comp. Soc. Conf. Comp. Vis. and Pat. Rec., CVPR, June 2000

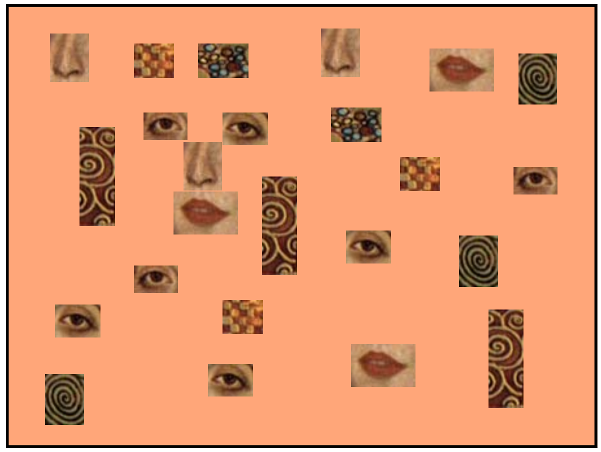
Yes, contains object — No, does not contain object

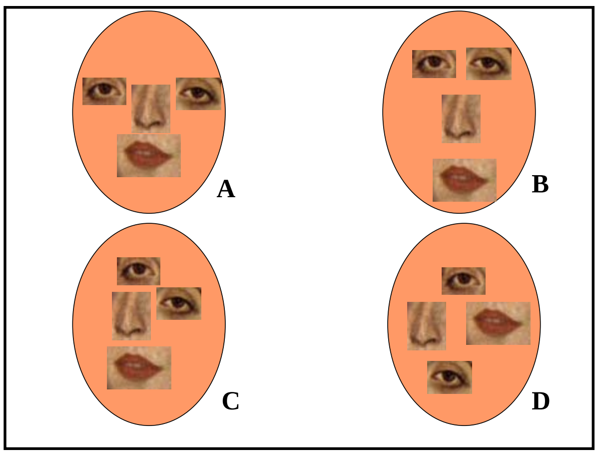
Fig. 1. Which objects appear consistently in the left images, but not on the right side? Can a machine learn to recognize instances of the two object classes (*faces* and *cars*) without any further information provided?

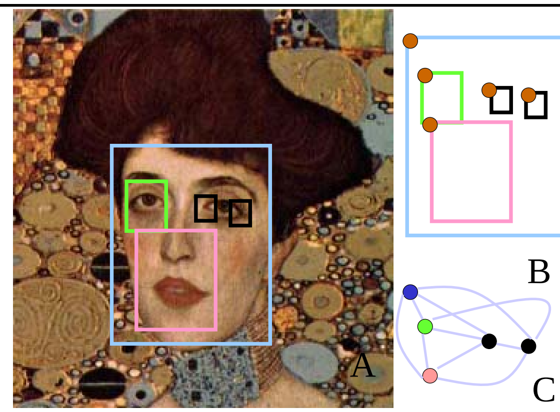
What are the features that let us recognize that this is a face?









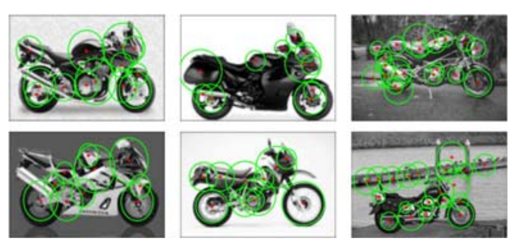


Feature detectors

- Keypoint detectors [Foerstner87]
- Jets / texture classifiers [Malik-Perona88, Malsburg91,…]
- Matched filtering / correlation [Burt85, …]
- PCA + Gaussian classifiers [Kirby90, Turk-Pentland92….]
- Support vector machines [Girosi-Poggio97, Pontil-Verri98]
- Neural networks [Sung-Poggio95, Rowley-Baluja-Kanade96]
- ……whatever works best (see handwriting experiments)

Representation

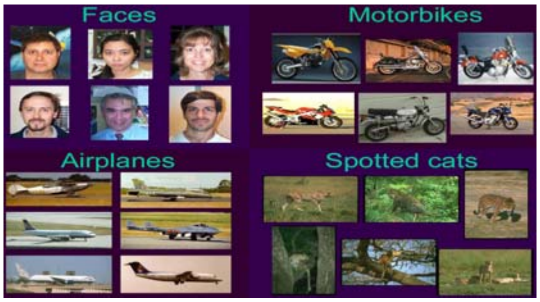
Use a scale invariant, scale sensing feature keypoint detector (like the first steps of Lowe's SIFT).



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

[Slide from Bradsky & Thrun, Stanford]

Data

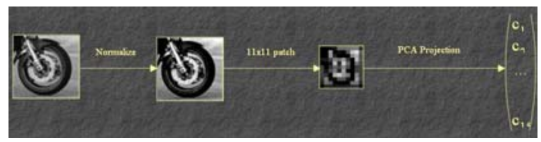


Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

[Slide from Bradsky & Thrun, Stanford]

Features for Category Learning

A direct appearance model is taken around each located key. This is then normalized by it's detected scale to an 11x11 window. PCA further reduces these features.



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

[Slide from Bradsky & Thrun, Stanford]

Unsupervised detector training - 2

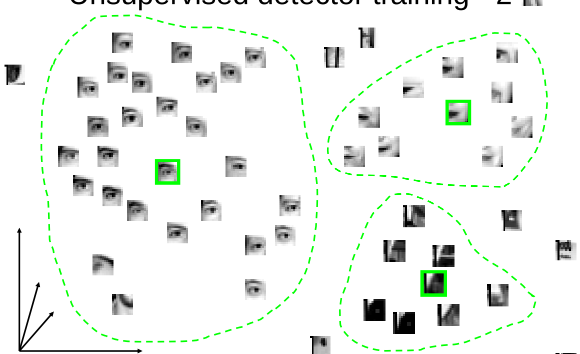
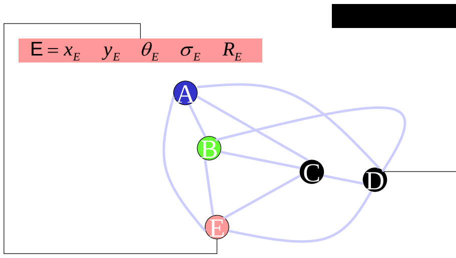


Fig. 3. Points of interest (left) identified on a training image of a human face in cluttered background using Förstner's method. Crosses denote corner-type patterns while circles mark circle-type patterns. A sample of the patterns obtained using k-means clustering of small image patches is shown for 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.



Hypothesis: H=(A,B,C,D,E)

*Probability density: P(*A,B,C,D,E*)*

Learning

- Fit with E-M (this example is a 3 part model)
- We start with the dual problem of **what** to fit and **where** to fit it.

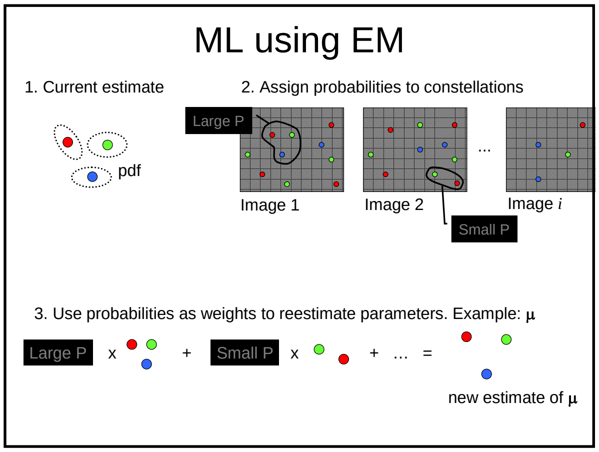
Assume that an object instance is the only consistent thing somewhere in a scene.

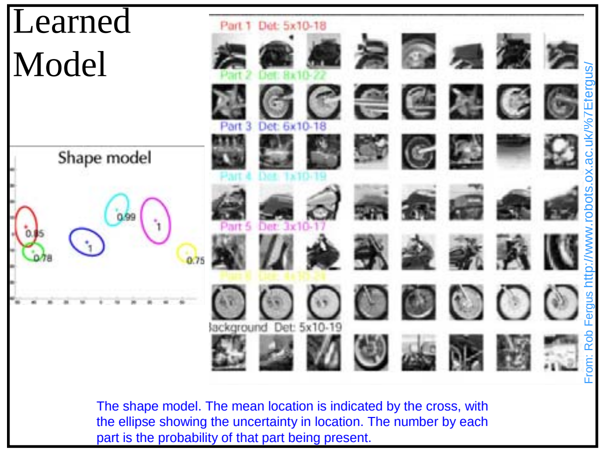
We don't know where to start, so we use the initial random parameters.

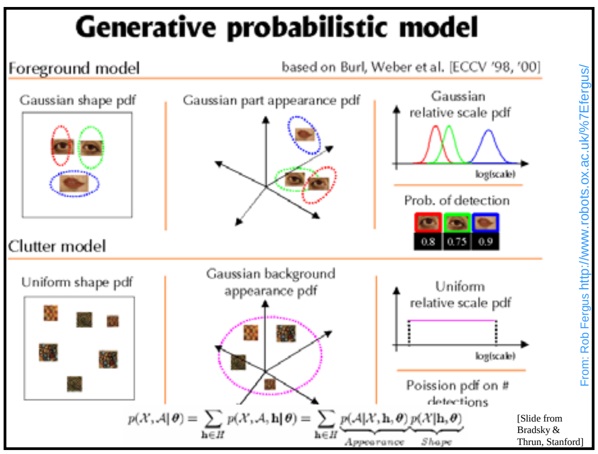
1. (M) We find the best (consistent across images) assignment given the params.
2. (E) We refit the feature detector params. and repeat until converged.
   - Note that there isn't much consistency
3. This repeats until it converges at the most consistent assignment with maximized parameters across images.

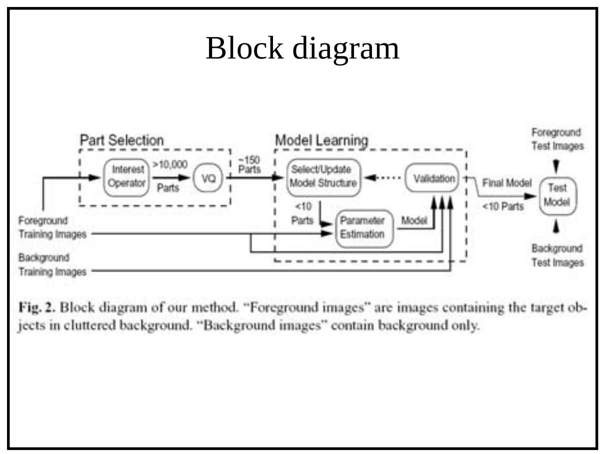
From: Rob Fergus http://www.robots.ox.ac.uk/%7Efergus/

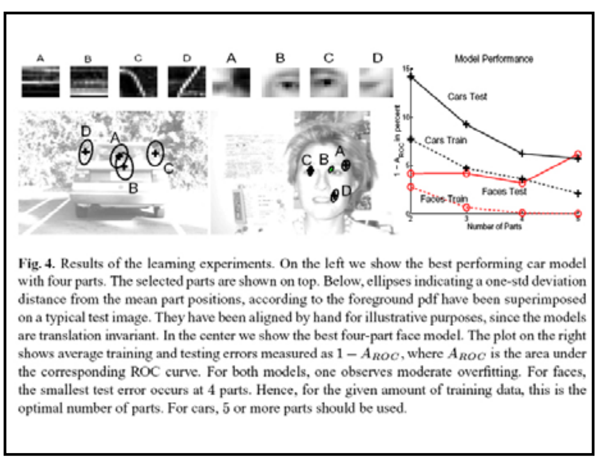
[Slide from Bradsky & Thrun, Stanford]

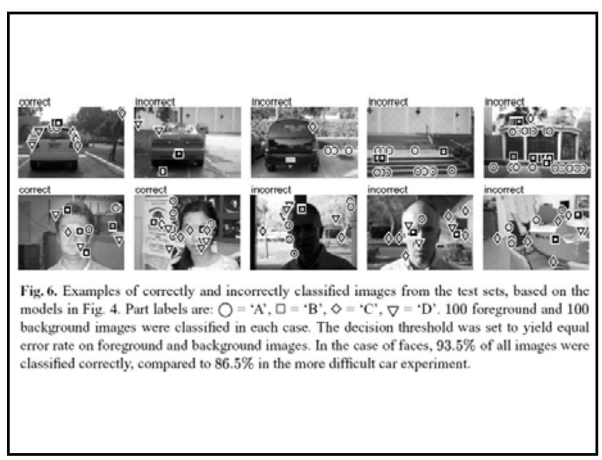

ML using EM
1. Current estimate
2. Assign probabilities to constellations
Large P
pdf
Image 1
Image 2
...
Image i
Small P
3. Use probabilities as weights to reestimate parameters. Example: μ
Large P x + Small P x + ... =
new estimate of μ


Learned Model
Part 1 Det: 5x10-18
Part 2 Det: 8x10-22
Part 3 Det: 6x10-18
Part 4 Det: 1x10-19
Part 5 Det: 3x10-17
Part 6 Det: 4x10-24
Background Det: 5x10-19
Shape model
0.85
0.78
1
0.99
1
0.75
The shape model. The mean location is indicated by the cross, with the ellipse showing the uncertainty in location. The number by each part is the probability of that part being present.
From: Rob Fergus http://www.robots.ox.ac.uk/%7Efergus/


Generative probabilistic model
Foreground model
based on Burl, Weber et al. [ECCV '98, '00]
Gaussian shape pdf
Gaussian part appearance pdf
Gaussian relative scale pdf
log(scale)
Prob. of detection
0.8 0.75 0.9
Clutter model
Uniform shape pdf
Gaussian background appearance pdf
Uniform relative scale pdf
log(scale)
Poission pdf on # detections
$p(\mathcal{X},\mathcal{A}|\theta) = \sum_{h\in H} p(\mathcal{X},\mathcal{A},h|\theta) = \sum_{h\in H} \underbrace{p(\mathcal{A}|\mathcal{X},h,\theta)}_{Appearance} \underbrace{p(\mathcal{X}|h,\theta)}_{Shape}$
[Slide from Bradsky & Thrun, Stanford]
From: Rob Fergus http://www.robots.ox.ac.uk/%7Efergus/


Block diagram
Part Selection
Model Learning
Foreground Test Images
Interest Operator
>10,000 Parts
VQ
~150 Parts
Select/Update Model Structure
Validation
Final Model
Test Model
<10 Parts
<10 Parts
Parameter Estimation
Model
Foreground Training Images
Background Training Images
Background Test Images
Fig. 2. Block diagram of our method. "Foreground images" are images containing the target objects in cluttered background. "Background images" contain background only.

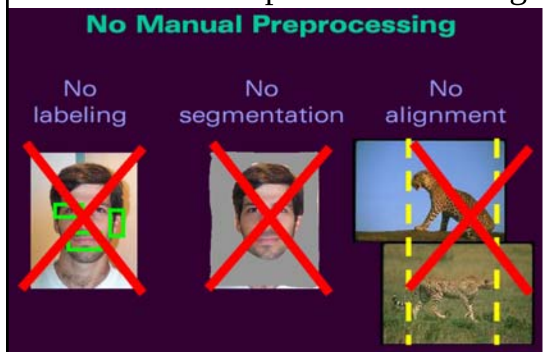

A B C D A B C D
Model Performance
Cars Test
Cars Train
Faces Test
Faces Train
Number of Parts
Fig. 4. Results of the learning experiments. On the left we show the best performing car model with four parts. The selected parts are shown on top. Below, ellipses indicating a one-std deviation distance from the mean part positions, according to the foreground pdf have been superimposed on a typical test image. They have been aligned by hand for illustrative purposes, since the models are translation invariant. In the center we show the best four-part face model. The plot on the right shows average training and testing errors measured as $1 - A_{ROC}$, where A_{ROC} is the area under the corresponding ROC curve. For both models, one observes moderate overfitting. For faces, the smallest test error occurs at 4 parts. Hence, for the given amount of training data, this is the optimal number of parts. For cars, 5 or more parts should be used.


correct
incorrect
incorrect
incorrect
incorrect
correct
correct
incorrect
incorrect
incorrect
Fig. 6. Examples of correctly and incorrectly classified images from the test sets, based on the models in Fig. 4. Part labels are: ○ = 'A', □ = 'B', ◇ = 'C', ▽ = 'D'. 100 foreground and 100 background images were classified in each case. The decision threshold was set to yield equal error rate on foreground and background images. In the case of faces, 93.5% of all images were classified correctly, compared to 86.5% in the more difficult car experiment.

Recognition

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

Result: Unsupervised Learning



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

[Slide from Bradsky & Thrun, Stanford]

Algorithm	Training Examples	Categories	Results (error)
Burl, et al. Weber, et al. Fergus, et al.	200 ~ 400	Faces, Motorbikes, Spotted cats, Airplanes, Cars	5.6 - 10 %

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

6.869

Previously: Object recognition via labeled training sets.
Previously: Unsupervised Category Learning
Now:

Perceptual organization:
- Gestalt Principles
- Segmentation by Clustering
  - K-Means
  - Graph cuts
- Segmentation by Fitting
  - Hough transform
  - Fitting

Readings: F&P Ch. 14, 15.1-15.2

Segmentation and Line Fitting

- Gestalt grouping
- K-Means
- Graph cuts
- Hough transform
- Iterative fitting

Segmentation and Grouping

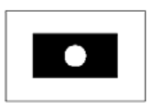
- Motivation: vision is often simple inference, but for segmentation
- Obtain a compact representation from an image/motion sequence/set of tokens
- Should support application
- Broad theory is absent at present

- Grouping (or clustering)
  - collect together tokens that "belong together"
- Fitting
  - associate a model with tokens
  - issues
    - which model?
    - which token goes to which element?
    - how many elements in the model?

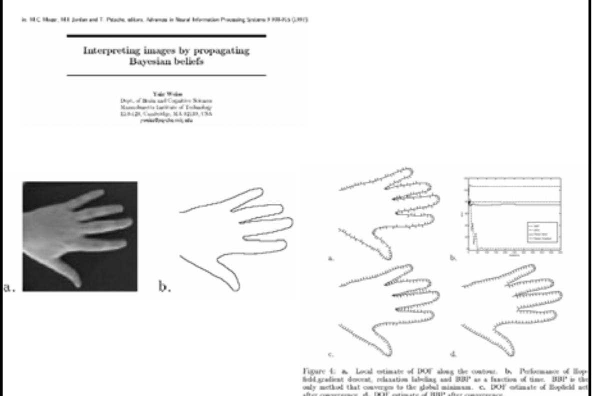
General ideas

- Tokens
  - whatever we need to group (pixels, points, surface elements, etc., etc.)
- Top down segmentation
  - tokens belong together because they lie on the same object
- Bottom up segmentation
  - tokens belong together because they are locally coherent
- These two are not mutually exclusive

Why do these tokens belong together?

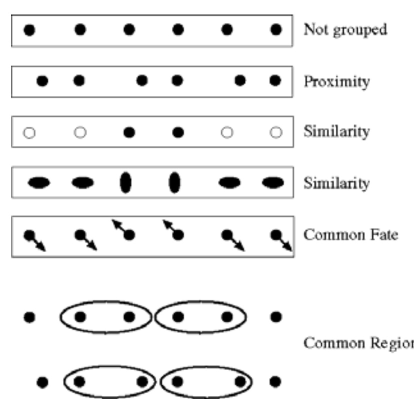


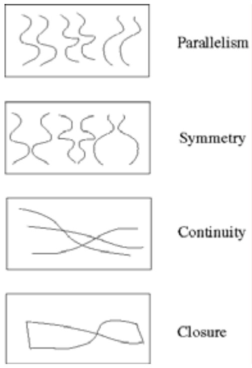
What is the figure?

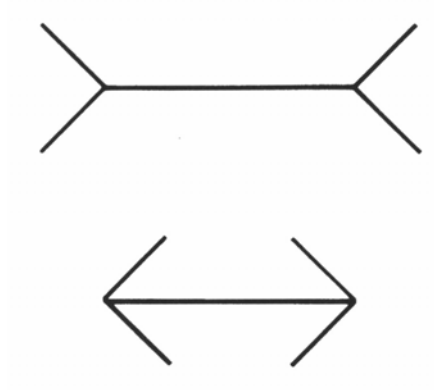


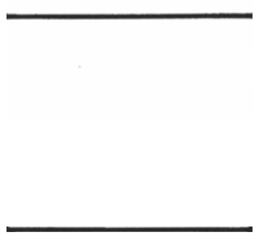
Basic ideas of grouping in humans

- Figure-ground discrimination
  - grouping can be seen in terms of allocating some elements to a figure, some to ground
  - impoverished theory
- Gestalt properties
  - A series of factors affect whether elements should be grouped together



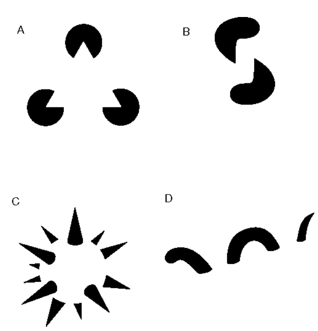






Occlusion is an important cue in grouping.

Consequence:
Groupings by Invisible Completions



* Images from Steve Lehar's Gestalt papers: http://cns-alumni.bu.edu/pub/slehar/Lehar.html

And the famous…

And the famous invisible dog eating under a tree:

- We want to let machines have these perceptual organization abilities, to support object recognition and both supervised and unsupervised learning about the visual world.

Segmentation as clustering

- Cluster together (pixels, tokens, etc.) that belong together…
- Agglomerative clustering
  - attach closest to cluster it is closest to
  - repeat
- Divisive clustering
  - split cluster along best boundary
  - repeat
- Dendrograms
  - yield a picture of output as clustering process continues

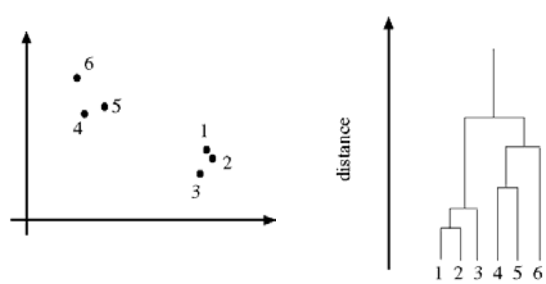
Clustering Algorithms

Algorithm 15.3: Agglomerative clustering, or clustering by merging

```
Make each point a separate cluster
Until the clustering is satisfactory
    Merge the two clusters with the
        smallest inter-cluster distance
end
```

Algorithm 15.4: Divisive clustering, or clustering by splitting

```
Construct a single cluster containing all points
Until the clustering is satisfactory
    Split the cluster that yields the two
        components with the largest inter-cluster distance
end
```



K-Means

- Choose a fixed number of clusters
- Choose cluster centers and point-cluster allocations to minimize error
- can't do this by search, because there are too many possible allocations.
- Algorithm
  - fix cluster centers; allocate points to closest cluster
  - fix allocation; compute best cluster centers
- x could be any set of features for which we can compute a distance (careful about scaling)

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

K-Means

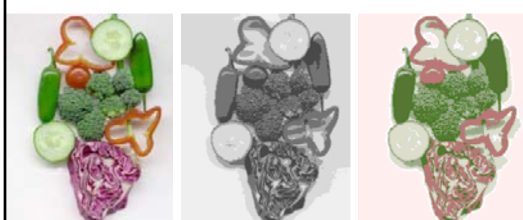
Algorithm 15.5: Clustering by K-Means

```
Choose k data points to act as cluster centers
Until the cluster centers are unchanged
    Allocate each data point to cluster whose center is nearest
    Now ensure that every cluster has at least
      one data point; possible techniques for doing this include .
      supplying empty clusters with a point chosen at random from
      points far from their cluster center.
    Replace the cluster centers with the mean of the elements
      in their clusters.
end
```

Image

Clusters on intensity (K=5)

Clusters on color (K=5)

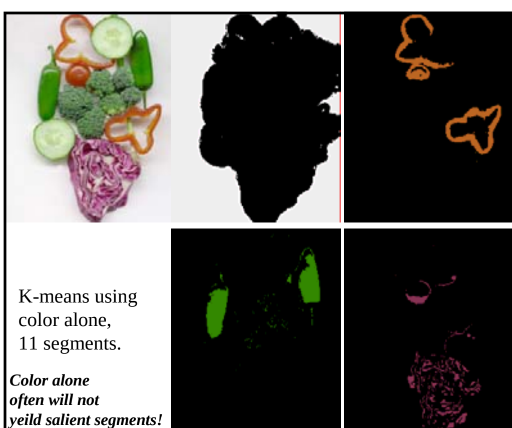


K-means clustering using intensity alone and color alone

Image

Clusters on color

K-means using color alone, 11 segments



K-means using color alone, 11 segments.

Color alone often will not yeild salient segments!

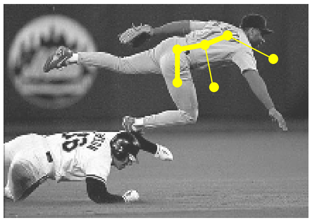
K-means using colour and position, 20 segments

Still misses goal of perceptually pleasing segmentation!

Hard to pick K…

Graph-Theoretic Image Segmentation

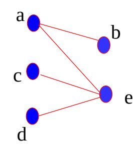
Build a weighted graph G=(V,E) from image



V: image pixels

E: connections between pairs of nearby pixels

Graphs Representations

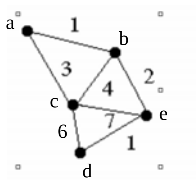


$$\begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Adjacency Matrix

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Weighted Graphs and Their Representations



$$\begin{bmatrix} 0 & 1 & 3 & \infty & \infty \\ 1 & 0 & 4 & \infty & 2 \\ 3 & 4 & 0 & 6 & 7 \\ \infty & \infty & 6 & 0 & 1 \\ \infty & 2 & 7 & 1 & 0 \end{bmatrix}$$

Weight Matrix

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Boundaries of image regions defined by a number of attributes

- Brightness/color
- Texture
- Motion
- Stereoscopic depth
- Familiar configuration

[Malik]

Measuring Affinity

Intensity

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_i^2} \right) \left(\| I(x) - I(y) \|^2 \right) \right\}$$

Distance

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_d^2} \right) \left(\| x - y \|^2 \right) \right\}$$

Color

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_t^2} \right) \left(\| c(x) - c(y) \|^2 \right) \right\}$$

Eigenvectors and affinity clusters

- Simplest idea: we want a vector a giving the association between each element and a cluster
- We want elements within this cluster to, on the whole, have strong affinity with one another
- We could maximize

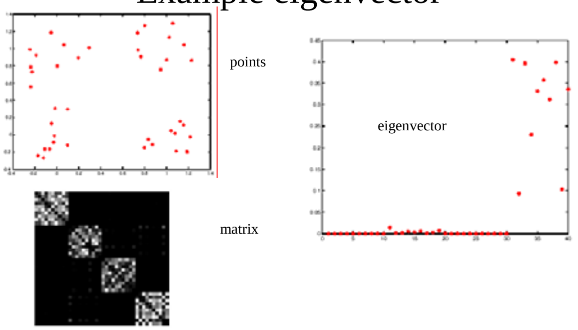
$$a^T A a$$

- But need the constraint

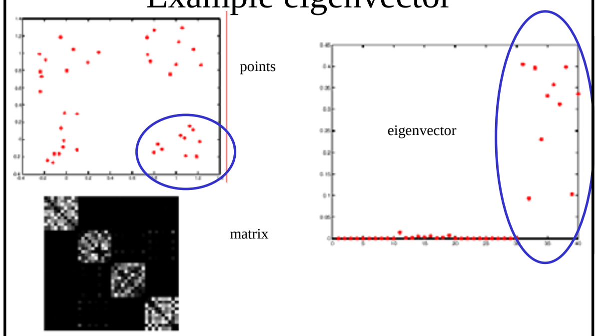
$$a^T a = 1$$

- This is an eigenvalue problem - choose the eigenvector of A with largest eigenvalue

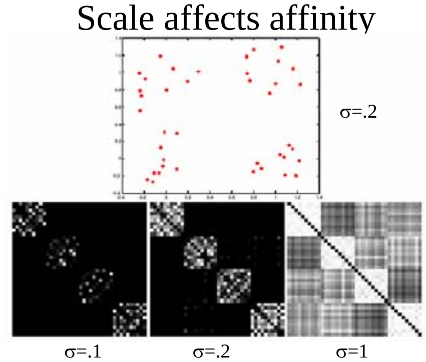
Example eigenvector



Example eigenvector



Scale affects affinity



Scale affects affinity

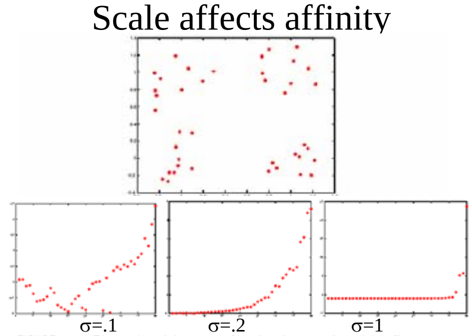
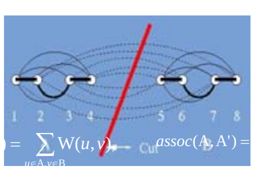


FIGURE 15.21: The number of clusters is reflected in the eigenvalues of the affinity matrix.

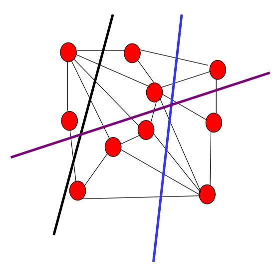
Some Terminology for Graph Partitioning

- How do we bipartition a graph:



[Malik]

Minimum Cut



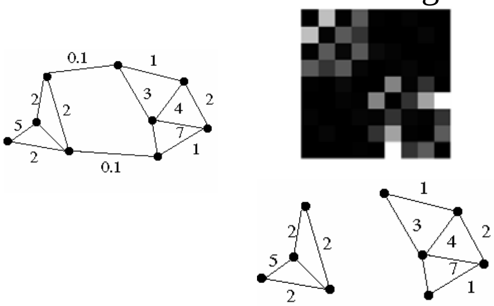
A cut of a graph *G* is the set of edges *S* such that removal of *S* from *G* disconnects *G*.

Minimum cut is the cut of minimum weight, where weight of cut <A,B> is given as

$$w(\langle A,B\rangle)=\sum_{x\in A,y\in B} w(x,y)$$

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

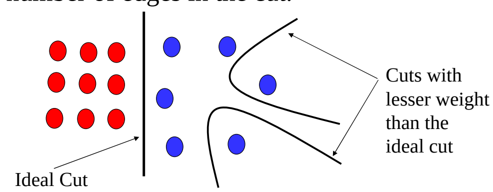
Minimum Cut and Clustering



* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Drawbacks of Minimum Cut

- Weight of cut is directly proportional to the number of edges in the cut.



* Slide from Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Normalized cuts

- First eigenvector of affinity matrix captures within cluster similarity, but not across cluster difference
- Min-cut can find degenerate clusters
- Instead, we'd like to maximize the within cluster similarity compared to the across cluster difference
- Write graph as V, one cluster as A and the other as B

- Maximize

$$\frac{\text{cut(A,B)}}{\text{assoc(A,V)}} + \frac{\text{cut(A,B)}}{\text{assoc(B,V)}}$$

where cut(A,B) is sum of weights that straddle A,B; assoc(A,V) is sum of all edges with one end in A.

I.e. construct A, B such that their within cluster similarity is high compared to their association with the rest of the graph

Solving the Normalized Cut problem

- Exact discrete solution to Ncut is NP-complete even on regular grid,
  - [Papadimitriou'97]
- Drawing on spectral graph theory, good approximation can be obtained by solving a generalized eigenvalue problem.

[Malik]

Normalized Cut As Generalized Eigenvalue problem

$$Ncut(\text{A,B}) = \frac{cut(\text{A,B})}{assoc(\text{A,V})} + \frac{cut(\text{A,B})}{assoc(\text{B,V})}$$

$$= \frac{(1+x)^T(D-W)(1+x)}{k1^T D1} + \frac{(1-x)^T(D-W)(1-x)}{(1-k)1^T D1};\ k = \frac{\sum_{x_i>0} D(i,i)}{\sum_i D(i,i)}$$

$$= \ldots$$

- after simplification, we get

$$Ncut(A,B) = \frac{y^T(D-W)y}{y^T Dy}, \quad \text{with } y_i \in \{1,-b\},\ y^T D1 = 0.$$

[Malik]

Normalized cuts

- Instead, solve the generalized eigenvalue problem

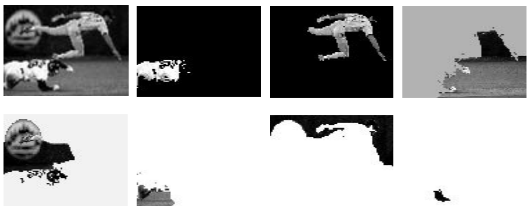
$$\max_y \left(y^T(D-W)y\right) \text{ subject to } \left(y^T Dy = 1\right)$$

- which gives

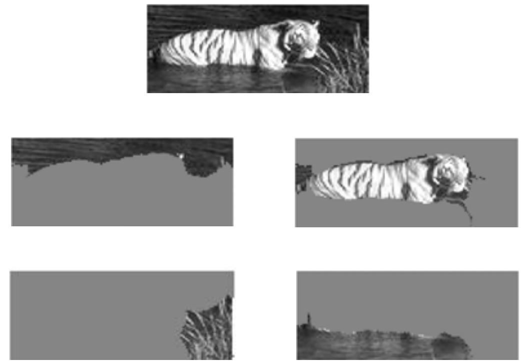
$$(D-W)y = \lambda Dy$$

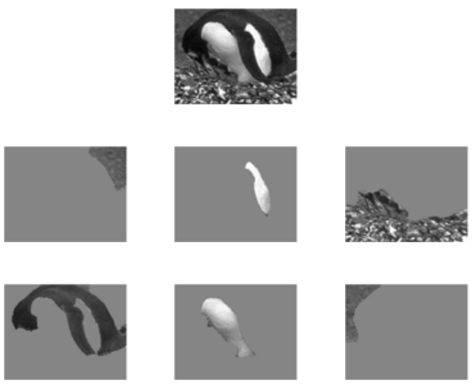
- Now look for a quantization threshold that maximizes the criterion --- i.e all components of y above that threshold go to one, all below go to -b

Brightness Image Segmentation

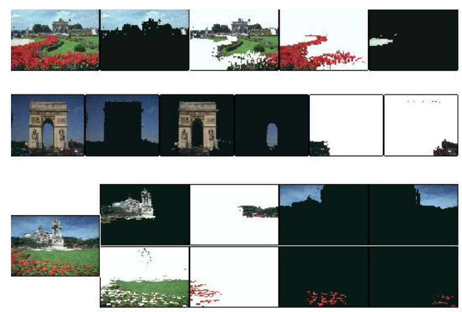


Brightness Image Segmentation



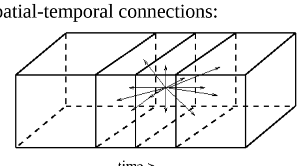


Results on color segmentation

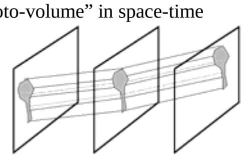


Motion Segmentation with Normalized Cuts

- Networks of spatial-temporal connections:



- Motion “proto-volume” in space-time



Comparison of Methods

Authors	Matrix used	Procedure/Eigenvectors used
Perona/ Freeman	Affinity A	1st x: $Ax = \lambda x$ Recursive procedure
Shi/Malik	D-A with D a degree matrix $D(i,i) = \sum_j A(i,j)$	2nd smallest *generalized* eigenvector $(D - A)x = \lambda Dx$ Also recursive
Scott/ Longuet-Higgins	Affinity A, User inputs k	Finds k eigenvectors of A, forms V. Normalizes rows of V. Forms Q = VV'. Segments by Q. Q(i,j)=1 -> same cluster
Ng, Jordan, Weiss	Affinity A, User inputs k	Normalizes A. Finds k eigenvectors, forms X. Normalizes X, clusters rows

Advantages/Disadvantages

- Perona/Freeman
  - For block diagonal affinity matrices, the first eigenvector finds points in the "dominant"cluster; not very consistent
- Shi/Malik
  - 2nd generalized eigenvector minimizes affinity between groups by affinity within each group; no guarantee, constraints

Advantages/Disadvantages

- Scott/Longuet-Higgins
  - Depends largely on choice of k
  - Good results
- Ng, Jordan, Weiss
  - Again depends on choice of k
  - Claim: effectively handles clusters whose overlap or connectedness varies across clusters