Pattern Recognition

Hierarchical clustering
Face Detection

Department of Intelligent Computer Systems
Faculty of ICT
UNIVERSITY OF MALTA
L-Università ta' Malta

Dr. George Azzopardi
george.azzopardi@um.edu.mt

October 2016

Erasmus+ Agreement

- UG/PG/PhD

University of Malta

Established in 1769

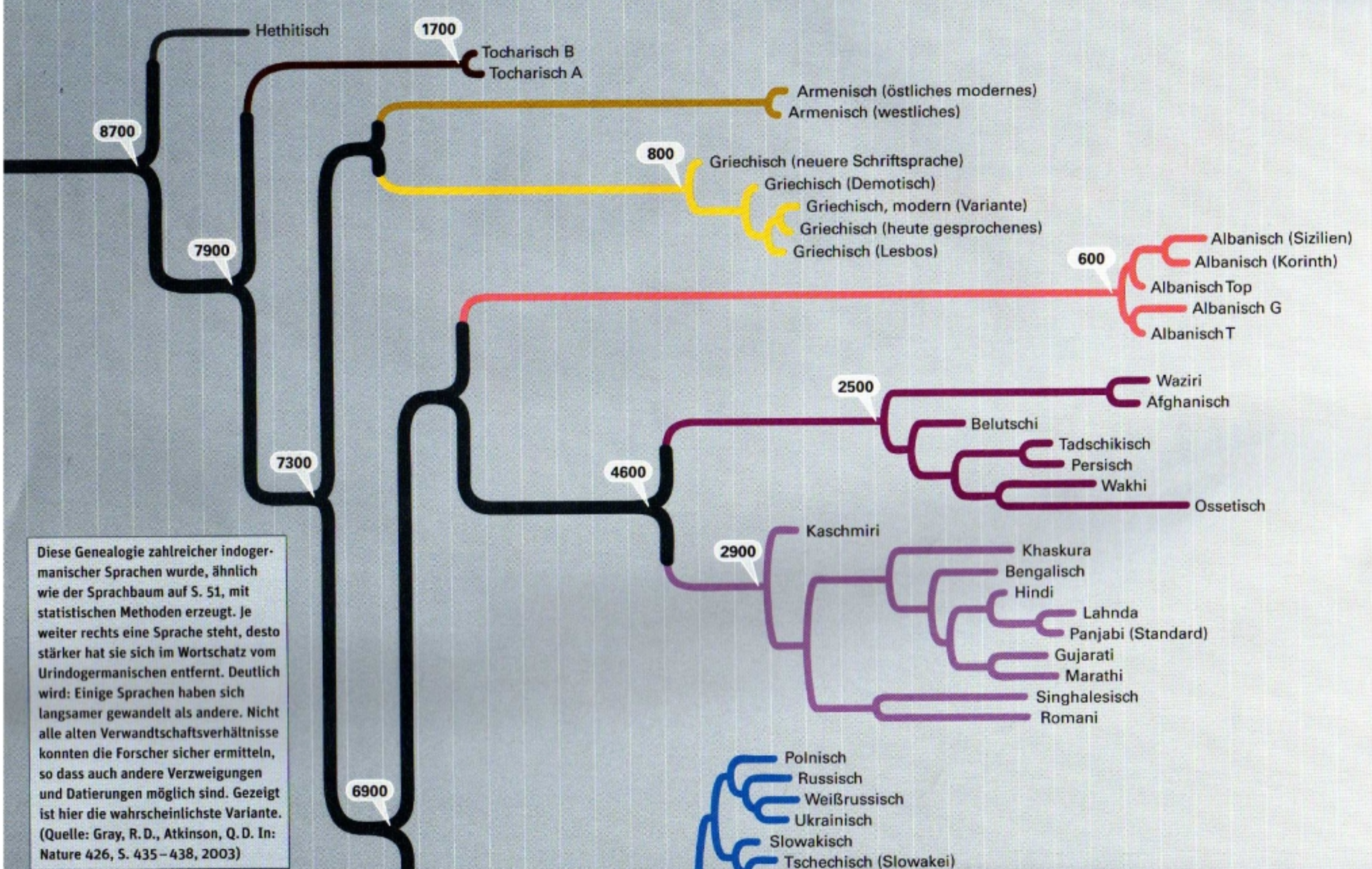
Student House

ICT Faculty

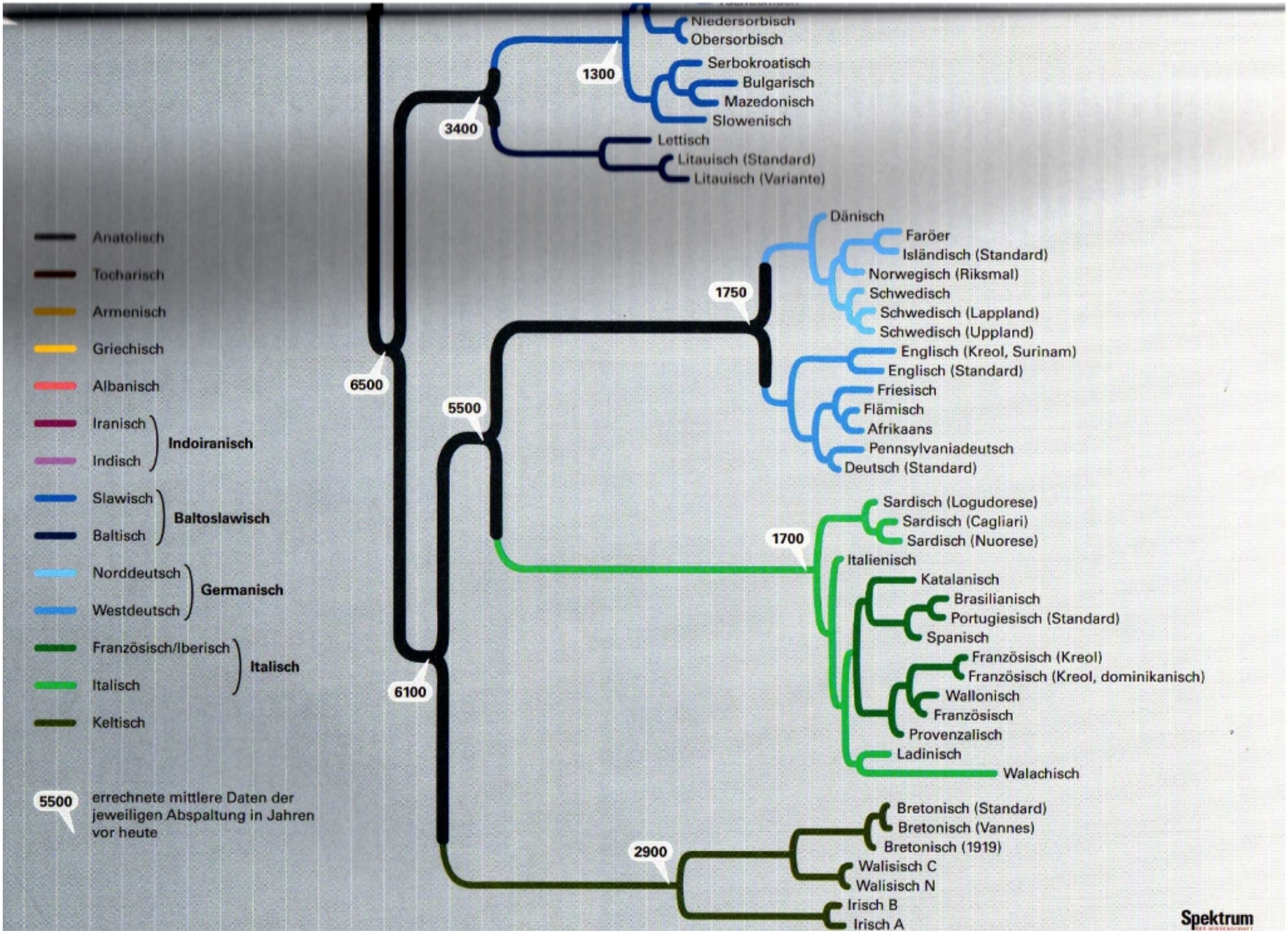
Part I

Hierarchical Clustering

STAMMBAUM DER INDOGERMANISCHEN SPRACHEN



Diese Genealogie zahlreicher indogermanischer Sprachen wurde, ähnlich wie der Sprachbaum auf S. 51, mit statistischen Methoden erzeugt. Je weiter rechts eine Sprache steht, desto stärker hat sie sich im Wortschatz vom Urindogermanischen entfernt. Deutlich wird: Einige Sprachen haben sich langsamer gewandelt als andere. Nicht alle alten Verwandtschaftsverhältnisse konnten die Forscher sicher ermitteln, so dass auch andere Verzweigungen und Datierungen möglich sind. Gezeigt ist hier die wahrscheinlichste Variante. (Quelle: Gray, R. D., Atkinson, Q. D. In: Nature 426, S. 435–438, 2003)


Niedersorbisch
Obersorbisch
1300
Serbokroatisch
Bulgarisch
Mazedonisch
Slowenisch
3400
Lettisch
Litauisch (Standard)
Litauisch (Variante)
Dänisch
Faröer
Isländisch (Standard)
Norwegisch (Riksmal)
Schwedisch
Schwedisch (Lappland)
Schwedisch (Uppland)
1750
Englisch (Kreol, Surinam)
Englisch (Standard)
Friesisch
Flämisch
Afrikaans
Pennsylvaniadeutsch
Deutsch (Standard)
6500
5500
Sardisch (Logudorese)
Sardisch (Cagliari)
Sardisch (Nuorese)
1700
Italienisch
Katalanisch
Brasilianisch
Portugiesisch (Standard)
Spanisch
Französisch (Kreol)
Französisch (Kreol, dominikanisch)
Wallonisch
Französisch
Provenzalisch
Ladinisch
Walachisch
6100
Bretonisch (Standard)
Bretonisch (Vannes)
Bretonisch (1919)
2900
Walisisch C
Walisisch N
Irisch B
Irisch A
Anatolisch
Tocharisch
Armenisch
Griechisch
Albanisch
Iranisch
Indisch
Indoiranisch
Slawisch
Baltisch
Baltoslawisch
Norddeutsch
Westdeutsch
Germanisch
Französisch/Iberisch
Italisch
Italisch
Keltisch
5500
errechnete mittlere Daten der jeweiligen Abspaltung in Jahren vor heute
Spektrum

How can we build such diagrams (called dendrograms) in which the objects cluster in groups of different size at different levels?

An example follows.

Dissimilarity between words

Let
S_1 = {a, e, i, o, u, y}

S_2 = {b, c, d, f, g, h, j, k, l, m, n, p, q, r, s, t, v, w, x, z}

We define the dissimilarity d(x,x') between two characters x and x' as follows (this is called edit distance or Levinstein distance):

d(x,x') = 0 if x == x'
= 1 if x != x' and both in S_1
= 2 if x != x' and both in S_2
= 5 if x != x' and in different sets
= 7 if x != empty and x' == empty OR
x == empty and x' != empty

Dissimilarity between words

Next we define the dissimilarity of two words as follows:

WordDistance = Σ charDist(word1(i),word2(i))

- Example 1:
  distance('boy' , 'bay')
  = 0 + 1 + 0 = 1

$$d(x,x') = \begin{cases} 0 & \text{if } x == x' \\ 1 & \text{if } x \mathrel{!=} x' \text{ and both in S\_1} \\ 2 & \text{if } x \mathrel{!=} x' \text{ and both in S\_2} \\ 5 & \text{if } x \mathrel{!=} x' \text{ and in different sets} \\ 7 & \text{if } x \mathrel{!=} \text{empty and } x' == \text{empty OR} \\ & \quad x == \text{empty and } x' \mathrel{!=} \text{empty} \end{cases}$$

- Example 2:
  distance('boss' , 'bayes')
  = 0 + 1 + 5 + 5 + 7 = 18

Dissimilarity properties

- non-negativity $d(\mathrm{x}, \mathrm{x}') \geq 0$
- reflexivity $d(\mathrm{x}, \mathrm{x}') = 0 \quad \text{if and only if} \quad \mathrm{x} = \mathrm{x}'$
- symmetry $d(\mathrm{x}, \mathrm{x}') = d(\mathrm{x}', \mathrm{x})$

Dissimilarity matrix

	Baby	Day	Disc	Human	Mucus	Music	Mainly	People
Baby								
Day								
Disc								
Human								
Mucus								
Music								
Mainly								
People								

Dissimilarity matrix

	Baby	Day	Disc	Human	Mucus	Music	Mainly	People
Baby		12	10	11	11	11	22	23
Day			11	18	18	18	18	19
Disc				15	15	13	20	20
Human					7	7	20	20
Mucus						5	18	20
Music							18	20
Mainly								7
People								

Agglomerative clustering

Starting from individual observations, produce a sequence of clusters of increasing size.

Possible ways of defining cluster dissimilarity:

(a) By the two nearest objects in the clusters (single linkage)

$$d_{\min}(D_i, D_j) = \min_{\mathrm{x} \in D_i, \mathrm{x}' \in D_j} \| \mathrm{x} - \mathrm{x}' \|$$

(b) By the two farthest objects in the clusters (complete linkage)

$$d_{\max}(D_i, D_j) = \max_{\mathrm{x} \in D_i, \mathrm{x}' \in D_j} \| \mathrm{x} - \mathrm{x}' \|$$

(c) By the average distance (average linkage)

$$d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x' \in D_j} \| \mathrm{x} - \mathrm{x}' \|$$

(d)

$$d_{mean}(D_i, D_j) = \| \mathrm{m}_i - \mathrm{m}_j \|$$

Dissimilarity matrix

	Baby	Day	Disc	Human	(Mucus, Music)$_5$	Mainly	People
Baby		12	10	11	11	22	23
Day			11	18	18	18	19
Disc				15	13	20	20
Human					7	20	20
(Mucus, Music)$_5$						18	20
Mainly							7
People							

Cluster dissimilarity matrix

	Baby	Day	Disc	((Mucus,Music)$_5$ Human)$_7$	(Mainly,People)$_7$
Baby		12	10	11	22
Day			11	18	18
Disc				13	20
((Mucus,Music)$_5$ Human)$_7$					18
(Mainly, People)$_7$					

Cluster dissimilarity matrix

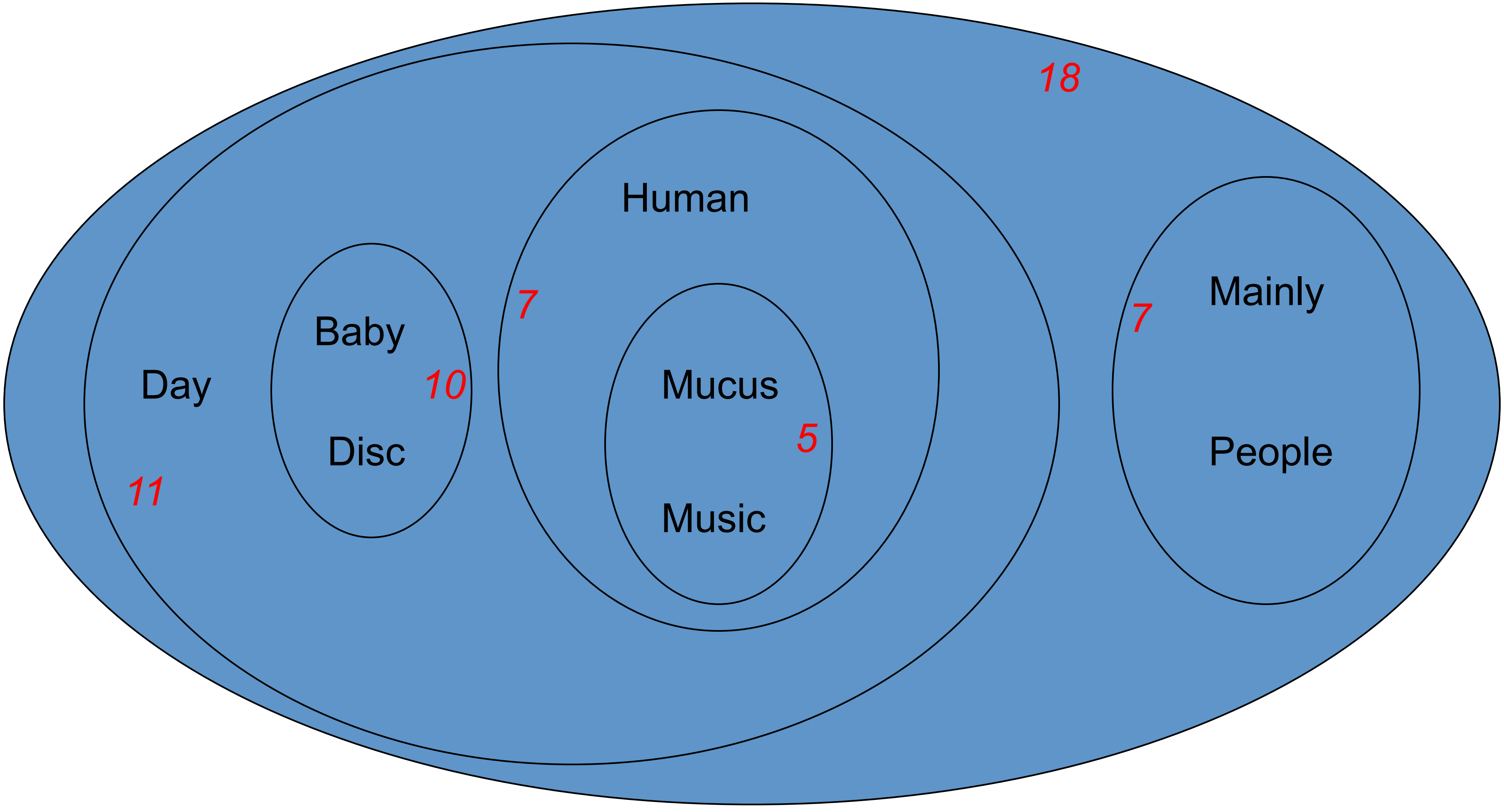
	$(Baby,Disc)_{10}$	Day	$((Mucus,Music)_5\ Human)_7$	$(Mainly, People)_7$
$(Baby,Disc)_{10}$		11	11	20
Day			18	18
$((Mucus,Music)_5\ Human)_7$				18
$(Mainly,People)_7$				

Cluster dissimilarity matrix

	(Day, (Baby, Disc)$_{10,}$ ((Mucus, Music)$_5$ Human)$_7$)$_{11}$	(Mainly,People)$_7$
(Day, (Baby, Disc)$_{10,}$ ((Mucus, Music)$_5$ Human)$_7$)$_{11}$		18
(Mainly,People)$_7$		

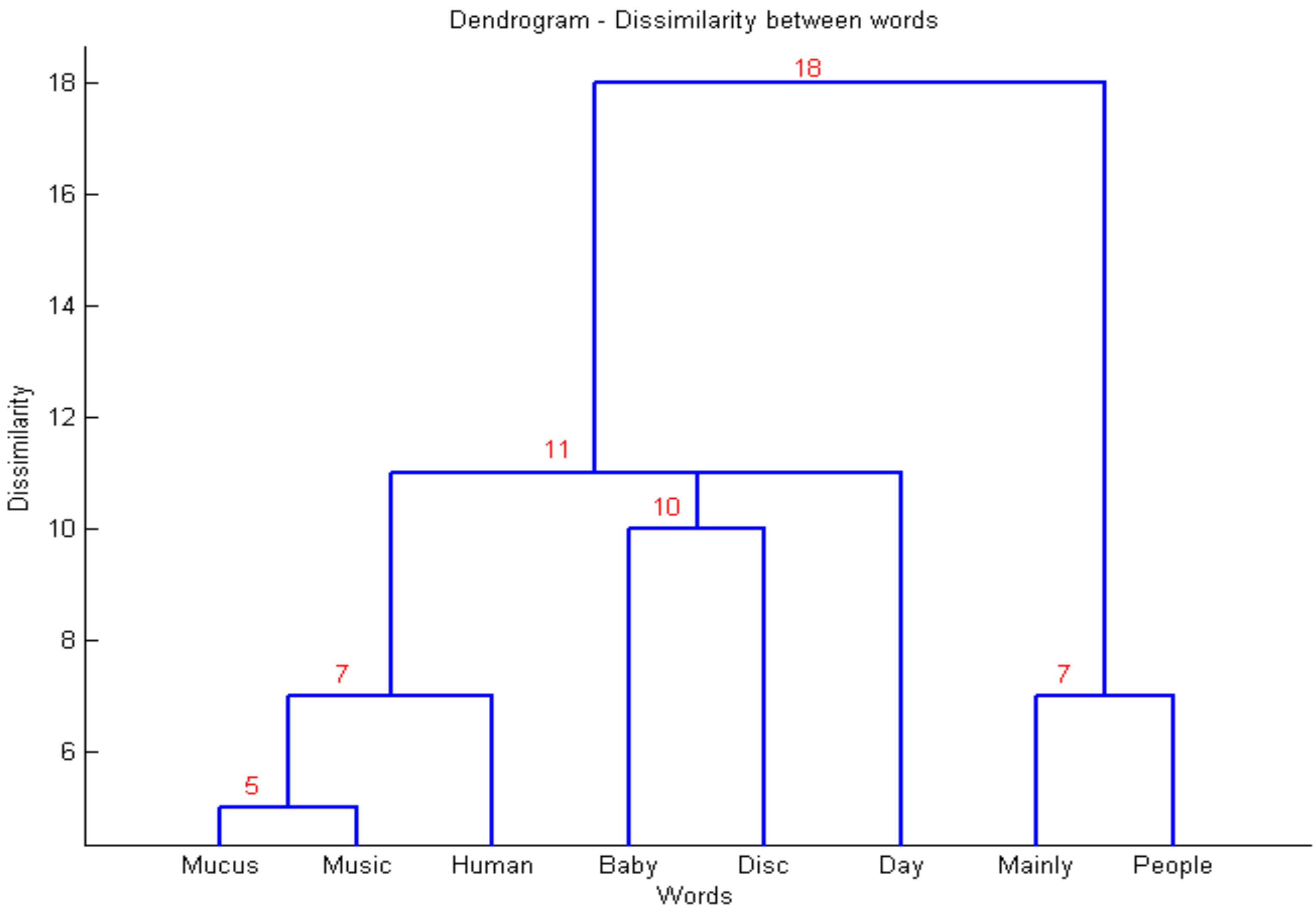
Venn diagram

$((((\text{Baby, Disc})_{10,}\ ((\text{Mucus, Music})_5, \text{Human})_7)_{11}, \text{Day})_{12}, (\text{Mainly,People})_7)_{18}$



Dendrogram

Representation of the clustering hierarchy



Dendrogram in Matlab (example)

Example: Let x_i, i = 1 … 10, be n=10 feature vectors representing 10 objects. We will make a dendrogram for this set.

1. Put x_i, i = 1 … 10, to become the rows of a matrix X.

   (E.g. `X = rand(10,2);` (10 2-dim feature vectors)

2. Compute the pairwise distances of all observations in matrix X using a given distance metric: `Y = pdist(X,'cityblock');`

   (Y is a row vector that includes the off-diagonal elements of a pair-wise distance matrix. It has n(n-1)/2 elements.)

3. Using the pair-wise distances, compute a (n-1)x3 matrix Z that represents a hierarchical binary cluster tree:

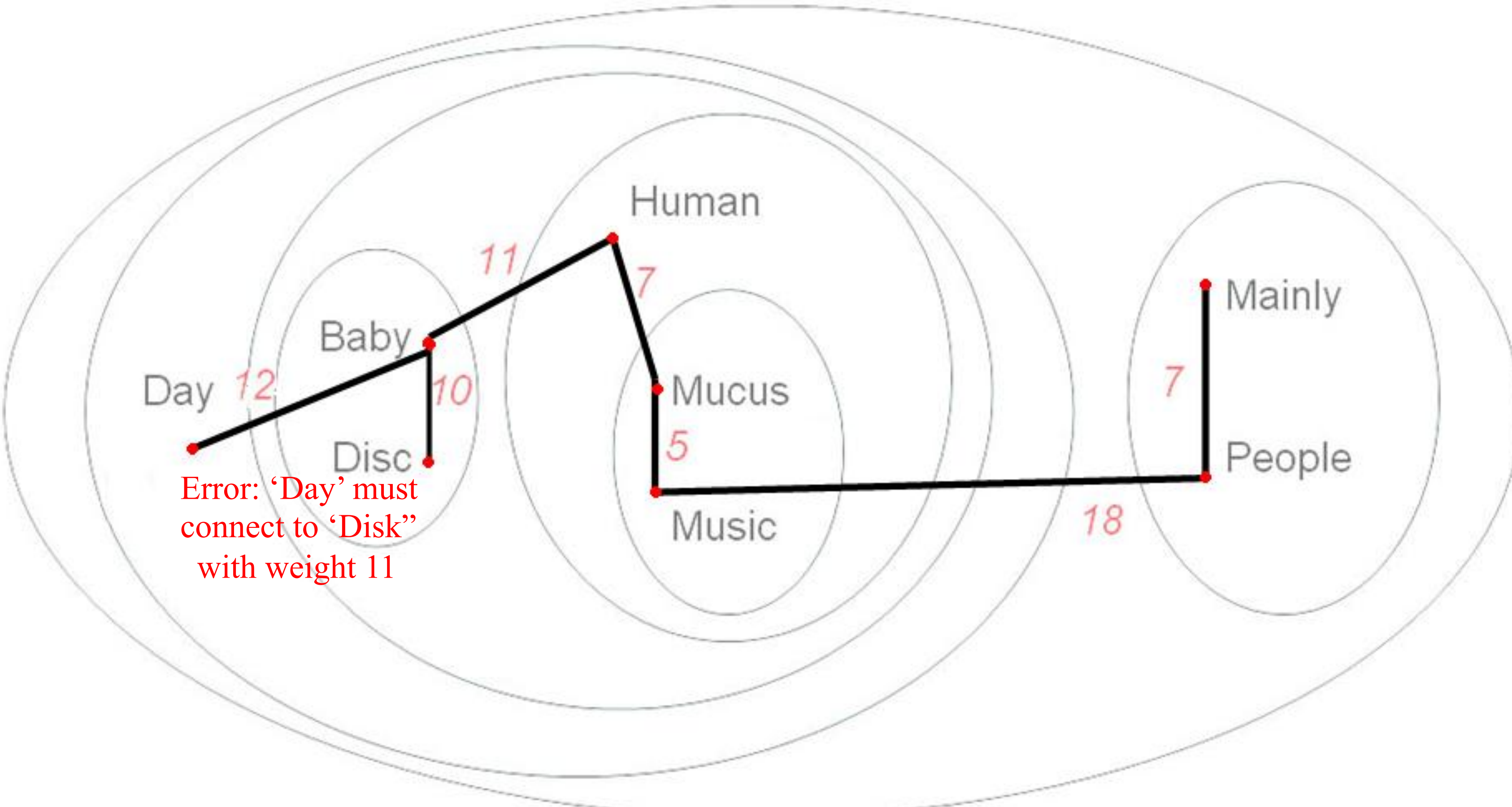
   `Z = linkage(Y,'average'); % ‘average’ – type of linkage used`

4. Compute and plot a dendrogram using

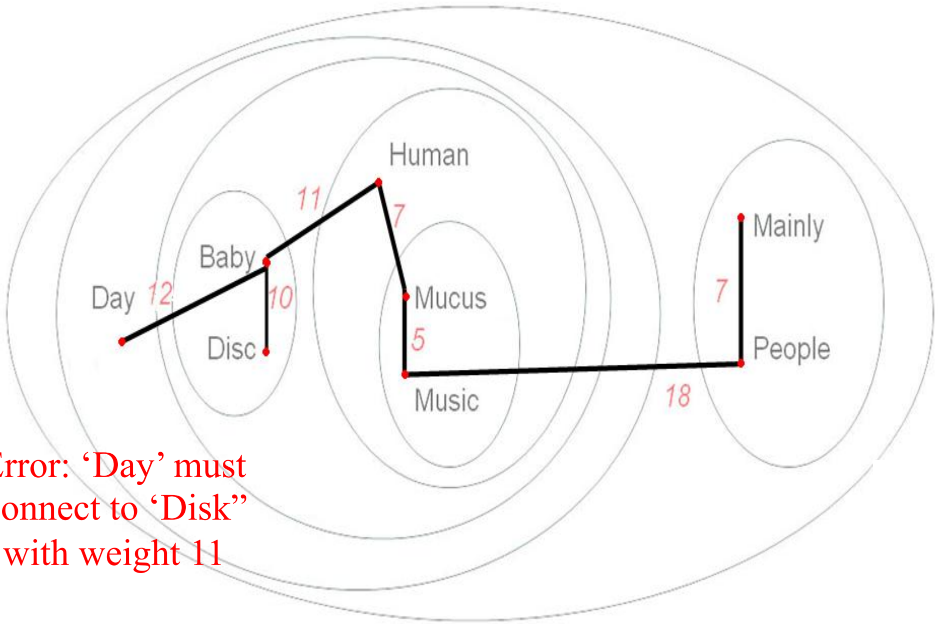
   `[H, T] = dendrogram(Z);  (T – nx1; H – vector of line handles)`

Minimal spanning tree

- Spanning tree is a weighted graph connecting all vertices without loops
- Minimum Spanning Tree (MST) is a spanning tree with minimum total weight
- If weights on edges are unique there is a unique MST
- If weights are derived from dissimilarity matrix *P(x,y) MST is identical to* the hierarchy defined by the *single linkage*

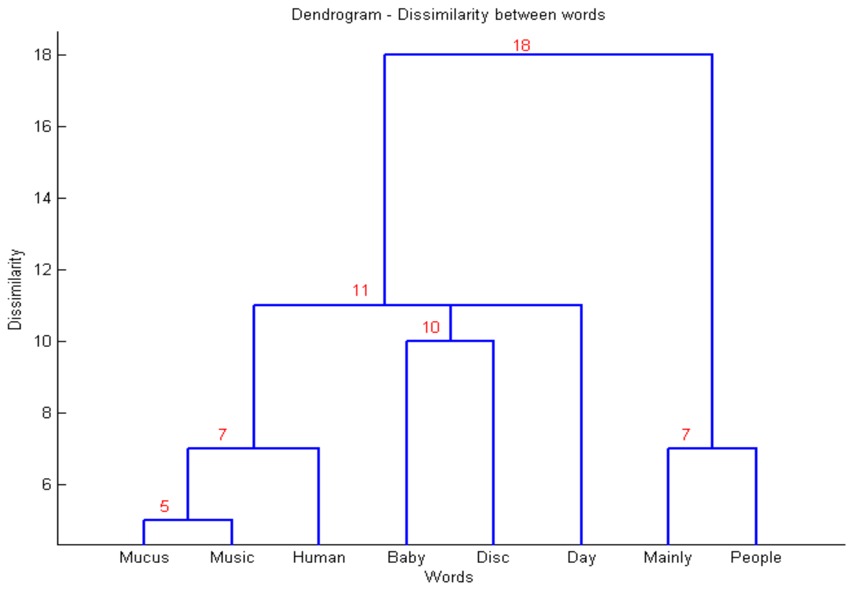


How to derive a dendrogram using a minimal spanning tree



Error: ‘Day’ must connect to ‘Disk” with weight 11

The connection lengths of the minimal spanning tree correspond with the dissimilarity values of the junctions of the dendrogram



	5	6	15	10	14	11	8	9	13	17	20	21	22	23	30	28	27	29	26	19	18	24	3	4	2	1	7	16	12	25
5	100.00																													
6	92.30	100.00																												
15	88.77	84.75	100.00																											
10	88.71	69.13	79.81	100.00																										
14	79.03	65.89	80.10	97.50	100.00																									
11	82.75	70.31	80.47	97.39	96.66	100.00																								
8	79.70	69.39	76.50	96.90	94.42	93.27	100.00																							
9	83.19	75.79	80.62	95.21	90.94	94.23	93.55	100.00																						
13	72.20	56.99	74.25	92.24	94.55	90.09	90.61	83.03	100.00																					
17	79.03	72.23	87.49	92.56	93.26	89.96	88.96	88.56	87.20	100.00																				
20	77.33	71.50	87.70	87.97	89.55	85.96	84.93	83.09	83.96	96.79	100.00																			
21	73.95	59.95	74.09	93.21	95.24	90.46	91.77	85.71	91.93	94.01	92.07	100.00																		
22	72.96	61.00	77.41	91.74	93.56	88.46	91.12	85.09	90.14	92.85	94.15	98.13	100.00																	
23	73.03	66.25	78.22	88.61	89.04	83.23	92.42	84.22	83.95	89.57	89.92	91.71	94.03	100.00																
30	69.23	55.35	68.07	83.91	86.79	81.05	85.51	74.22	88.59	81.31	82.29	88.63	88.03	85.80	100.00															
28	69.03	55.77	66.41	84.97	86.74	81.05	85.43	77.16	84.95	83.05	83.43	90.55	89.01	87.63	95.44	100.00														
27	66.44	52.65	67.03	85.95	87.94	81.96	87.03	76.75	88.79	84.41	84.77	92.77	93.73	90.29	95.93	96.09	100.00													
29	65.15	52.39	67.21	84.05	86.25	79.01	88.05	74.54	87.77	81.41	81.72	89.23	90.75	90.66	96.73	94.77	98.03	100.00												
26	67.31	56.11	69.93	87.92	88.79	82.32	90.94	80.51	86.13	86.20	85.01	92.77	94.37	96.10	90.99	93.99	96.65	96.07	100.00											
19	70.13	54.49	69.54	83.95	87.94	86.00	75.60	76.05	84.22	88.55	87.09	90.56	85.91	71.01	76.19	79.09	76.81	70.05	73.30	100.00										
18	76.03	61.02	75.29	87.26	91.92	89.09	81.49	70.97	91.21	89.63	88.95	92.11	88.00	78.93	84.07	84.21	82.99	79.40	79.05	96.16	100.00									
24	75.67	63.60	79.33	84.32	89.00	86.30	79.50	76.61	85.53	87.42	90.30	89.74	89.75	82.47	85.01	85.09	84.51	81.44	82.41	89.71	95.23	100.00								
3	75.82	63.82	54.99	71.01	71.30	68.11	76.21	69.01	69.08	65.82	63.11	69.17	67.82	71.97	67.38	67.29	67.15	68.61	69.98	55.17	64.78	61.26	100.00							
4	86.49	70.65	62.46	78.49	77.49	77.47	81.60	75.51	78.29	71.91	70.07	75.71	73.91	73.71	75.06	72.92	73.71	73.57	72.29	65.93	74.16	89.89	87.77	100.00						
2	75.51	55.63	57.59	77.13	79.05	79.75	73.06	70.31	79.36	71.31	69.19	76.63	72.11	64.25	73.27	73.07	69.59	67.09	66.14	80.17	85.42	78.60	77.32	82.47	100.00					
1	81.30	66.01	53.48	68.01	69.14	71.20	69.91	69.13	64.24	62.00	60.73	65.26	63.07	62.29	62.93	62.30	60.43	59.91	60.22	62.44	67.35	64.85	73.37	83.41	75.60	100.00				
7	74.64	61.43	53.66	70.83	67.25	76.41	65.93	75.03	62.45	60.68	56.37	58.81	55.47	46.64	49.89	51.11	46.69	42.80	44.50	69.07	67.83	61.49	49.28	65.71	70.14	72.59	100.00			
16	47.19	41.49	51.61	49.99	56.32	55.03	41.19	44.52	47.86	54.84	55.39	50.94	47.75	40.69	46.55	47.47	41.90	38.54	39.21	60.15	58.92	55.93	34.52	38.93	53.17	36.56	39.56	100.00		
12	41.18	37.12	41.19	52.61	50.93	50.55	53.79	48.02	46.61	47.57	44.44	47.23	45.06	48.36	43.84	43.67	43.84	45.37	46.43	39.60	44.02	44.00	40.25	41.25	37.51	33.57	29.94	18.81	100.00	
25	41.07	54.47	43.21	26.22	25.33	22.43	28.37	30.05	22.06	29.80	28.18	24.09	27.03	32.42	22.43	25.49	26.60	27.61	30.42	12.77	17.57	19.90	22.85	21.05	8.33	16.05	6.01	8.53	17.19	100.00

- Matrix showing similarity value between the black pepper cultivars (numbers on the left and right border correspond to the serial number as in table 1, indicating the cultivars). http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1948014

UPGMA dendrograms. A: black pepper cultivars using AFLP analysis along with digital fingerprint profile. B: the clustering pattern obtained for the major cultivars of black pepper.
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1948014

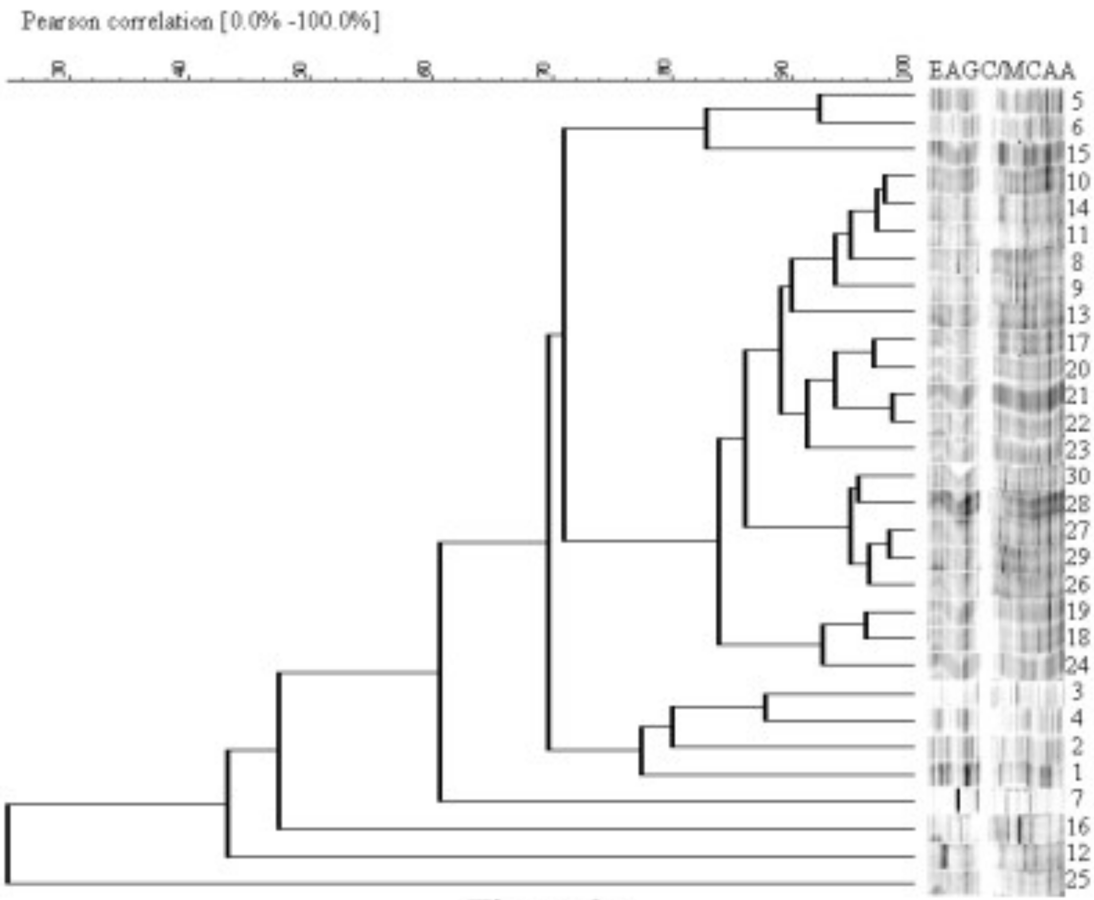
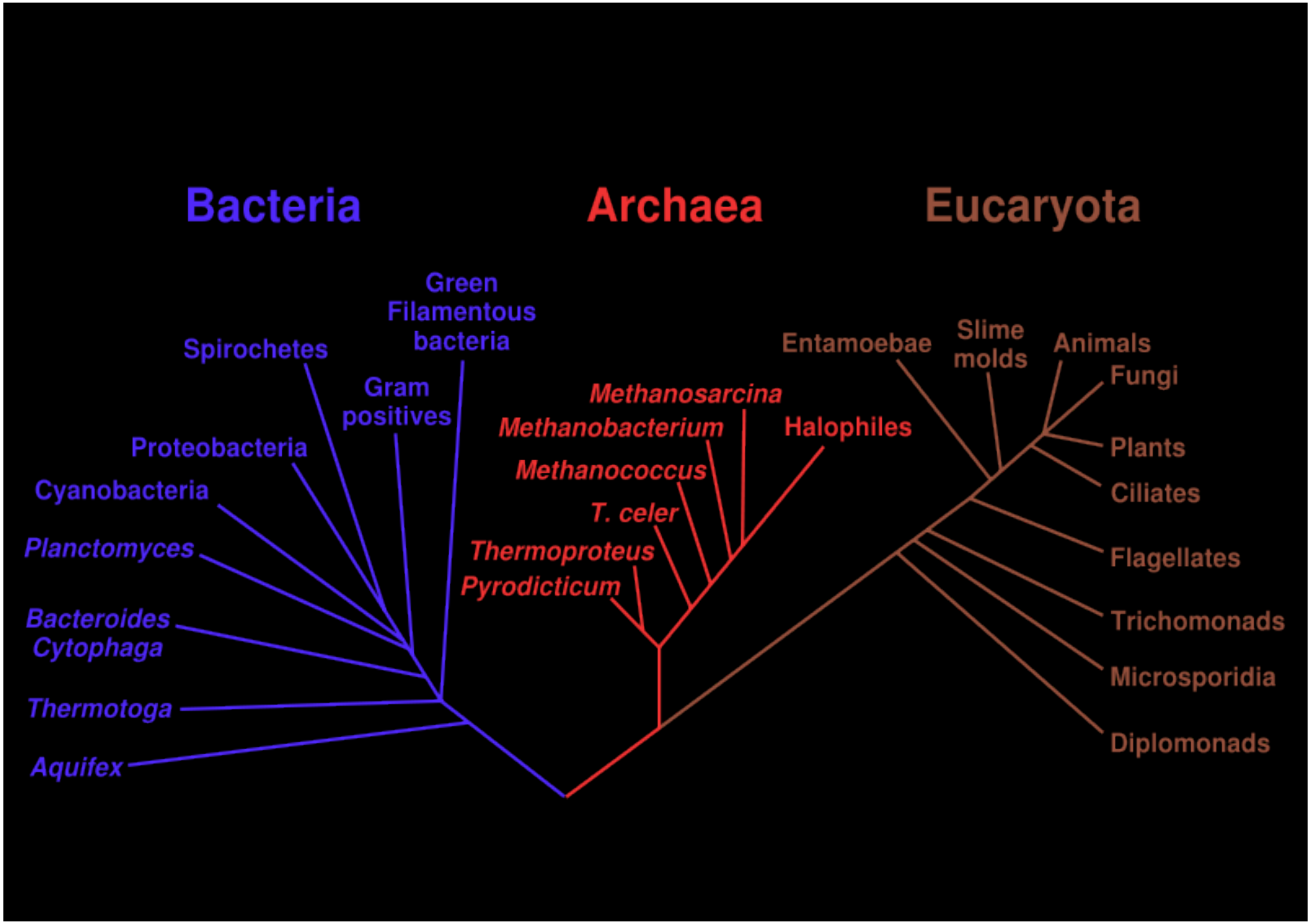
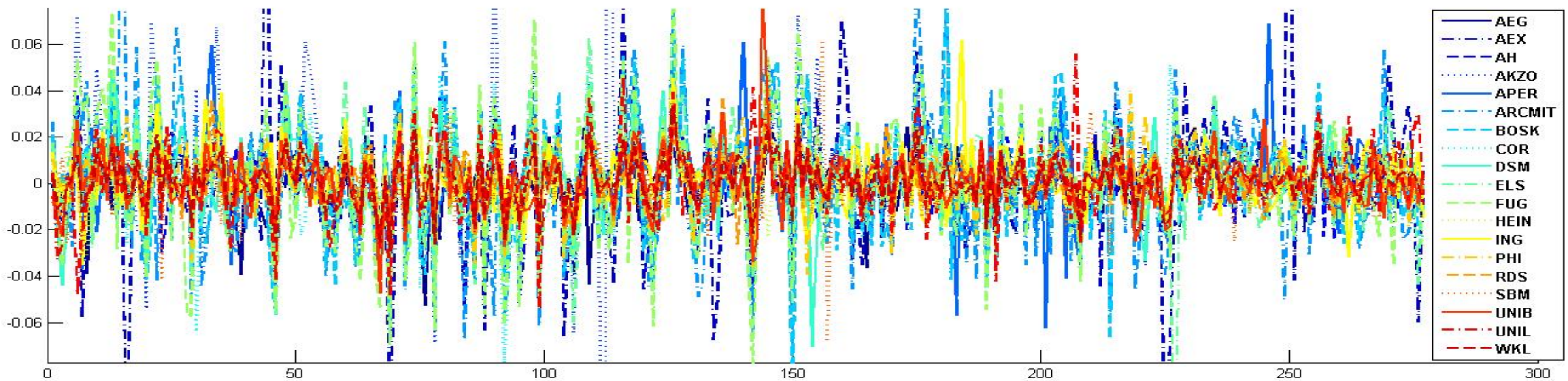


Figure 3a

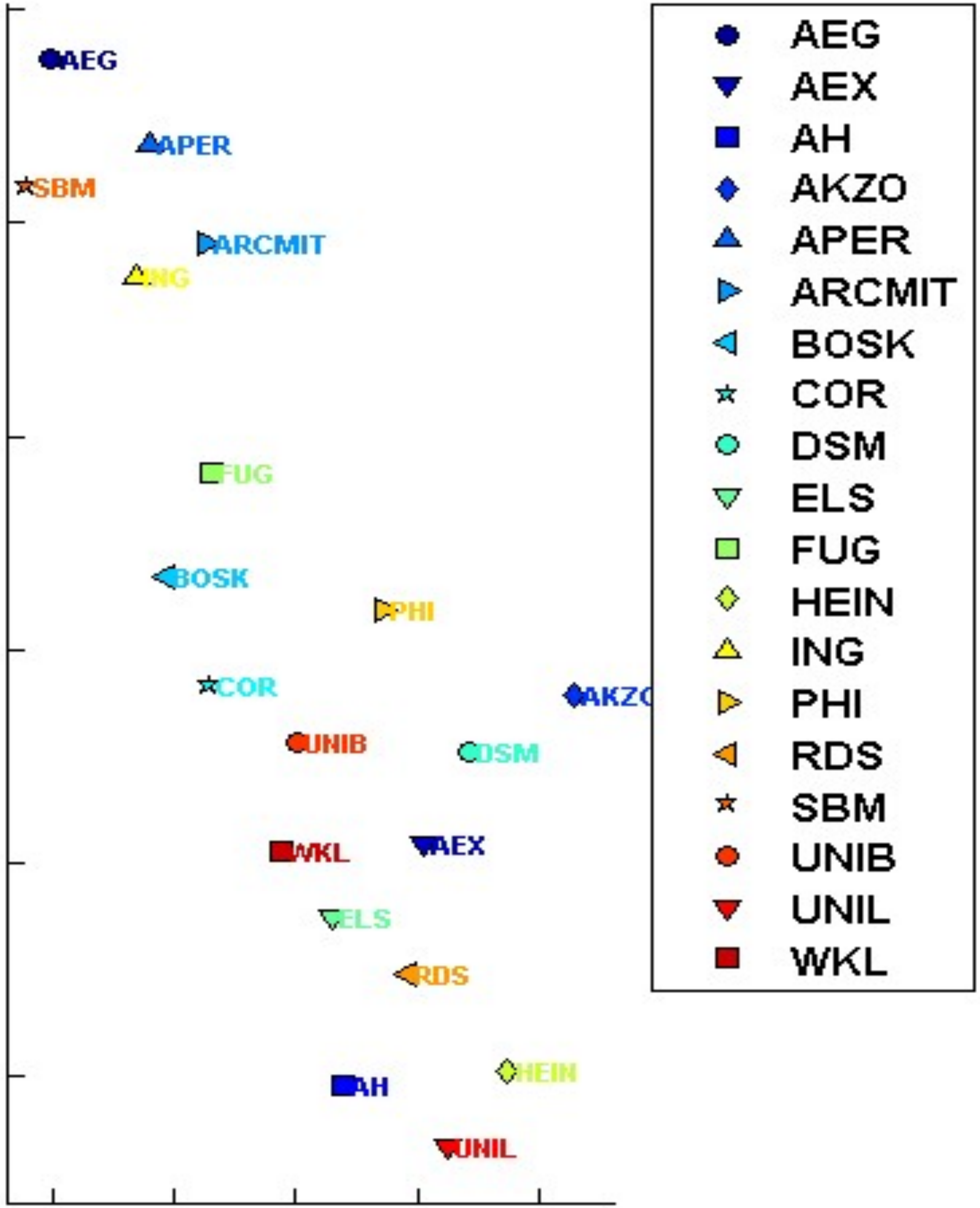
http://en.wikipedia.org/wiki/Image:Phylogenetic_tree.svg



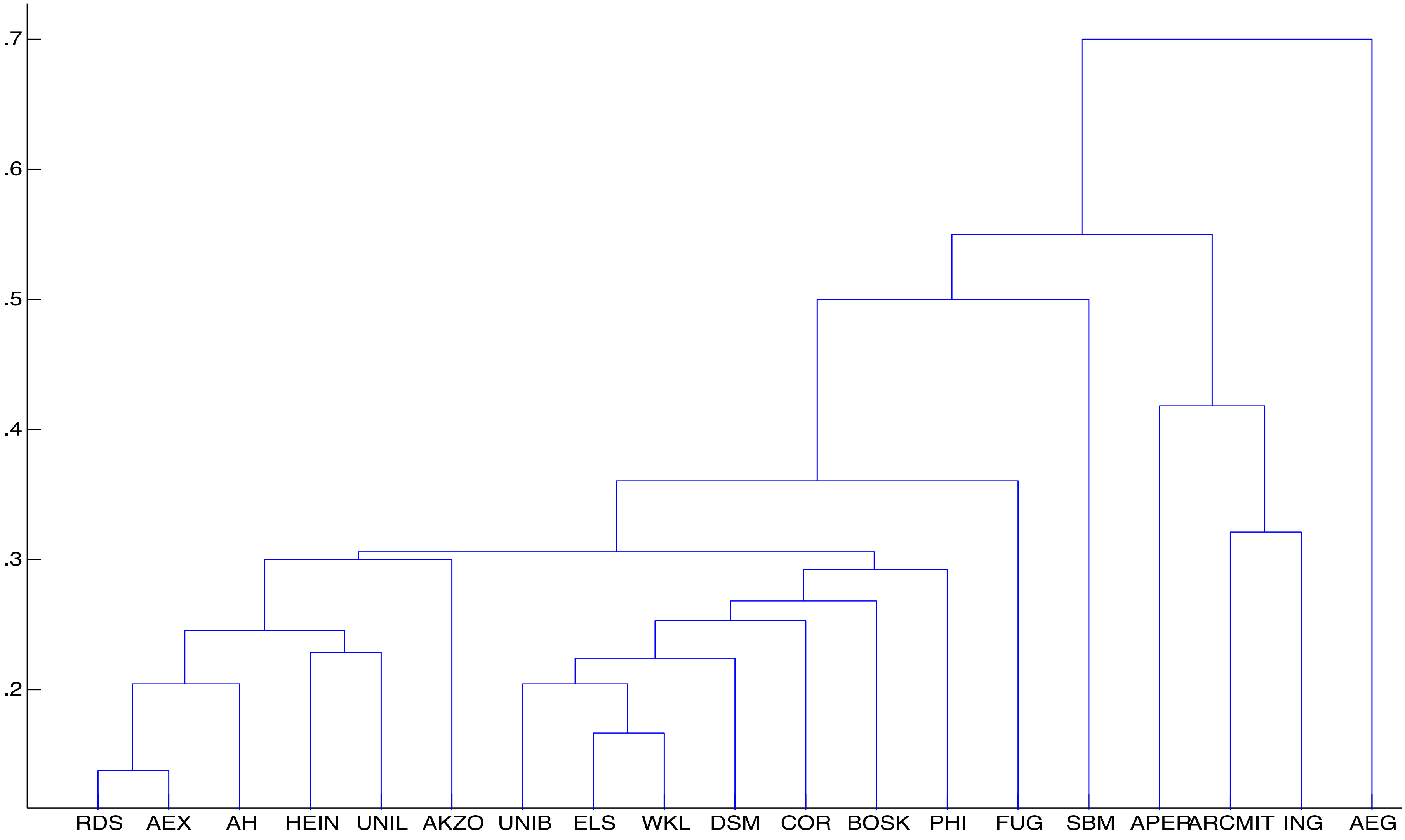
Example: stocks



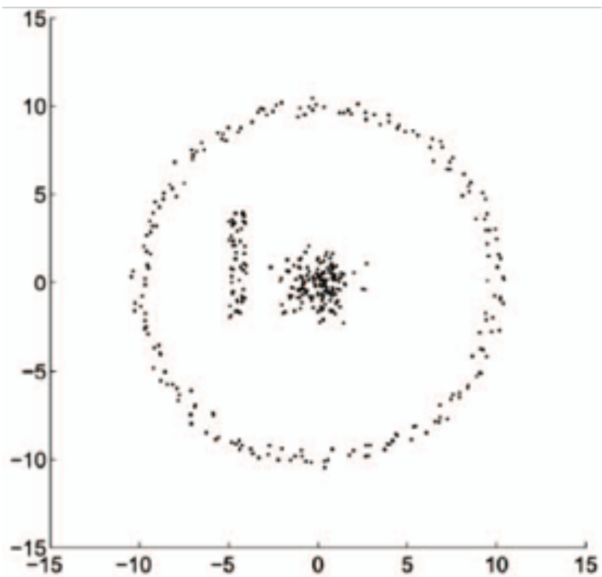
Stocks that change similarly are mapped near each other in a 2D space

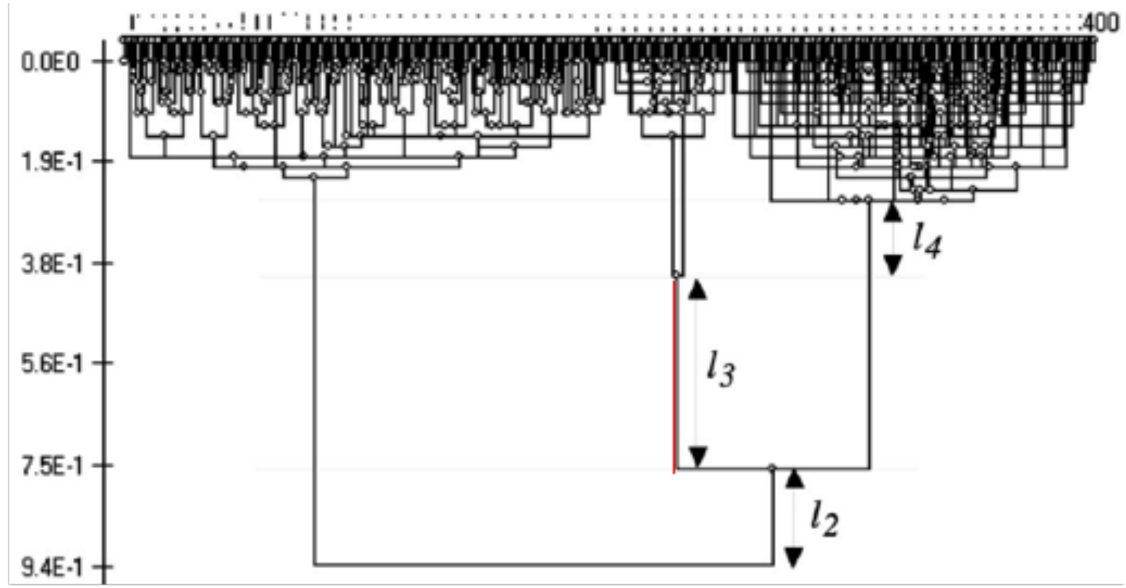


Hierarchical clustering of stocks

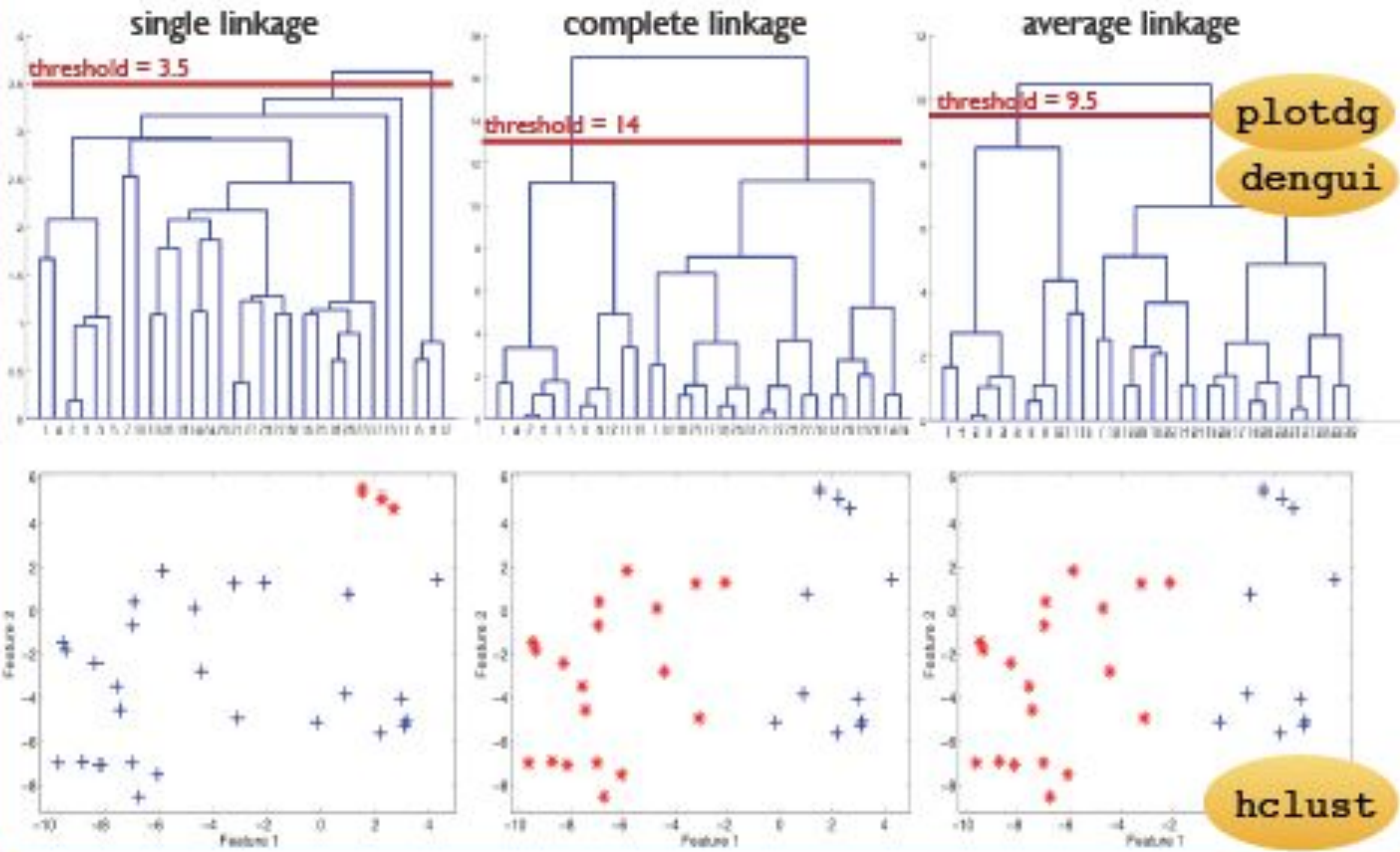


Another example





- Clustering solutions obtained by different hierarchical algorithms:



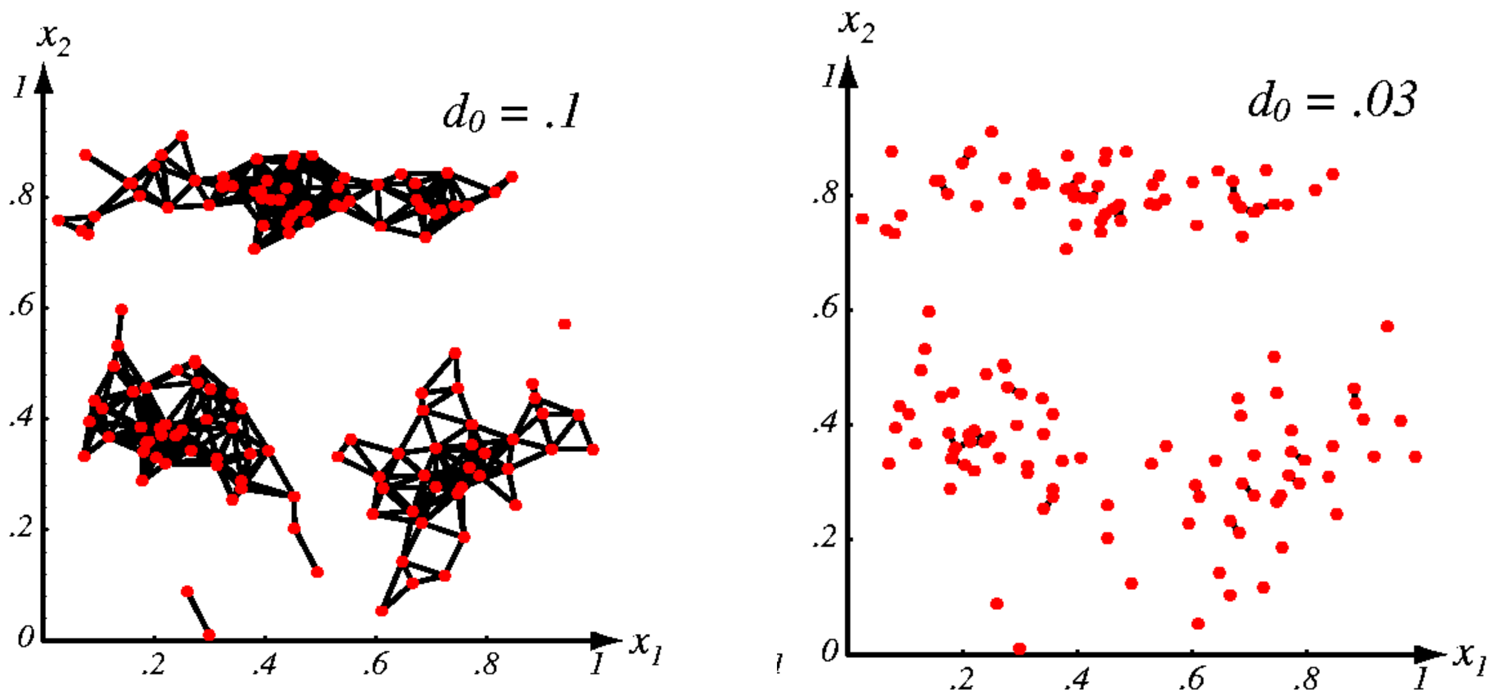
Induced metrics

Generalized distance function $d(\mathrm{x}, \mathrm{x}')$ - the lowest dissimilarity value for which x and x' are in the same cluster.

Properties:

- non-negativity $\quad d(\mathrm{x}, \mathrm{x}') \geq 0$
- reflexivity $\quad d(\mathrm{x}, \mathrm{x}') = 0 \quad \text{if and only if} \quad \mathrm{x} = \mathrm{x}'$
- symmetry $\quad d(\mathrm{x}, \mathrm{x}') = d(\mathrm{x}', \mathrm{x})$
- triangle inequality $\quad d(\mathrm{x}, \mathrm{x}') + d(\mathrm{x}', \mathrm{x}'') \geq d(\mathrm{x}, \mathrm{x}'')$

Graph-theoretical methods



The clusters are the connected components of a graph.

from Duda, Hart, Stork (2001) Pattern classification

Graph-theoretical methods

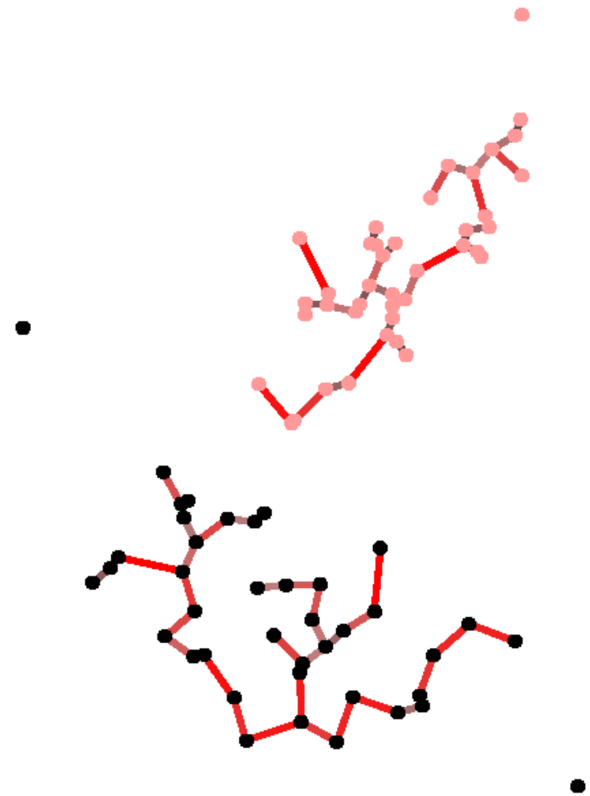
For arbitrary similarity measures, one can define a *similarity matrix* $S(x_i,x_j)$ *and define an adjacency matrix:*

$$s_{ij} = \begin{cases} 1, \text{ if } s(x_i, x_j) > s_0 \\ 0, \text{ otherwise} \end{cases}$$

This matrix induces a *similarity graph,* nodes correspond to points x_i and x_j, edge joins the nodes if $s_{ij} = 1$.

The clusters are the connected components of this graph.

Graph-theoretical methods



Example of connected components

from Duda, Hart, Stork (2001) Pattern classification

Graph-theoretical methods

Divisive methods based on a minimal spanning tree:

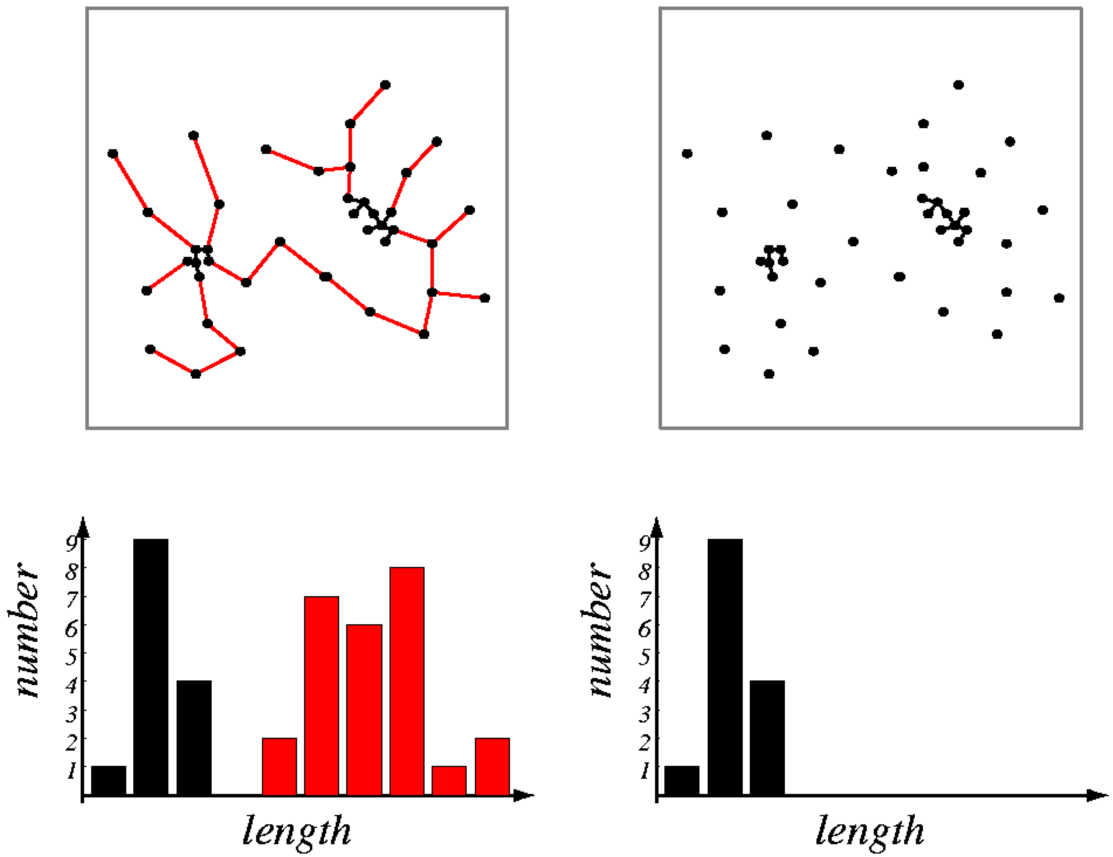
- Split the tree by removing the longest edge. Result: two clusters. Remove the next longest edge. Result: three clusters. Etc.

- Remove inconsistent edges (e.g. length exceeds twice the average length of the other edges incident on a node).

Graph-theoretical methods

Example of splitting a minimum spanning tree by removing two inconsistent edges.

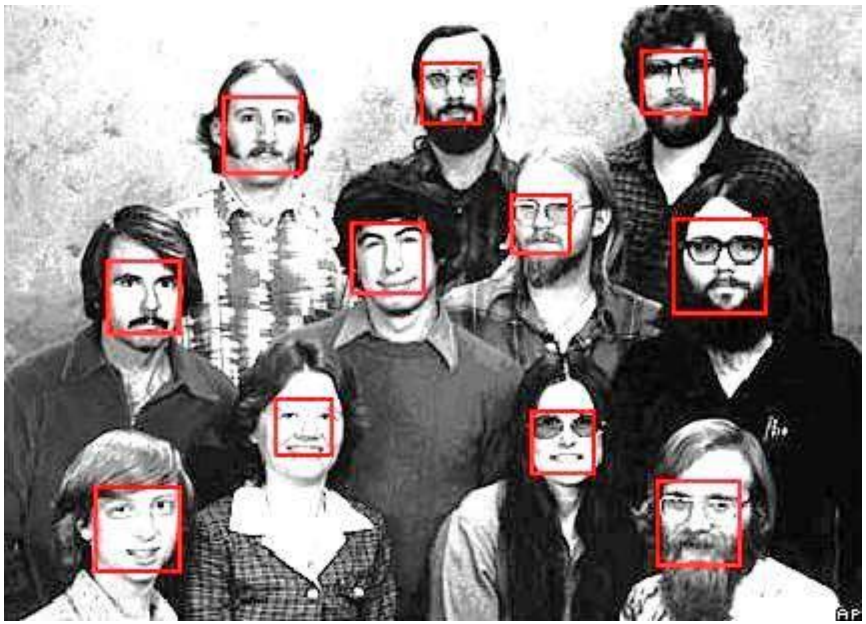
Graph-theoretical methods



Example of using the edge length histogram of a minimal spanning tree to identify two natural clusters – all edges above a certain threshold length are removed.

from Duda, Hart, Stork (2001) Pattern classification

Part 2 - Face detection



Many slides are with courtesy from from P. Viola

Why is face detection important?

- Automatic focus in digital cameras
- Interactive gaming
- Security
- Biometrics
- Further face analysis (e.g. gender, age, emotions)
- Augmented reality
- Special effects
- Etc …

Robust real-time face detection
> 10,000 citations

Paul Viola, Michael J. Jones

Article

DOI:
10.1023/B:VISI.0000013087.49260.fb

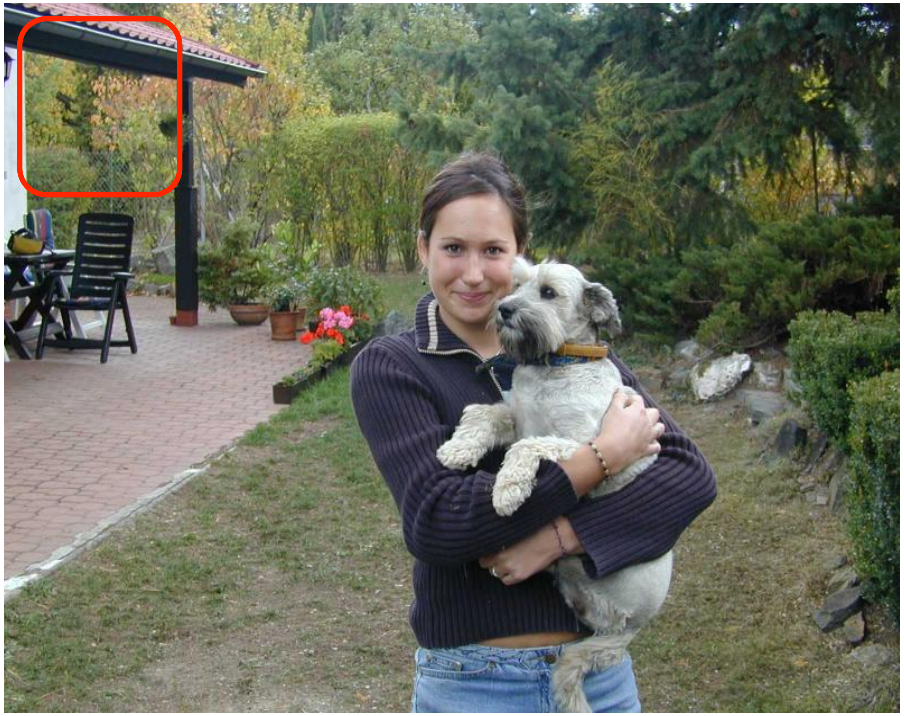
Cite this article as:
Viola, P. & Jones, M.J. International Journal of Computer Vision (2004) 57: 137. doi:10.1023/B:VISI.0000013087.49260.fb

Abstract

This paper describes a face detection framework that is capable of processing images extremely rapidly while achieving high detection rates. There are three key contributions. The first is the introduction of a new image representation called the "Integral Image" which allows the features used by our detector to be computed very quickly. The second is a simple and efficient classifier which is built using the AdaBoost learning algorithm (Freund and Schapire, 1995) to select a small number of critical visual features from a very large set of potential features. The third contribution is a method for combining classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising face-like regions. A set of experiments in the domain of face detection is presented. The system yields face detection performance comparable to the best previous systems (Sung and Poggio,

Face detection

- Basic idea: slide a window across image and evaluate a face model at every location



Challenges of face detection

- Sliding window detector must evaluate tens of thousands of location/scale combinations

- Faces are rare: 0–10 per image
  - For computational efficiency, we should try to spend as little time as possible on the non-face windows
  - A megapixel image has ~10^6 pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image, our false positive rate has to be less than 10^{-6}

The Viola/Jones Face Detector

- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - *Integral images* for fast feature evaluation
  - *Boosting* for feature selection
  - *Attentional cascade* for fast rejection of non-face windows

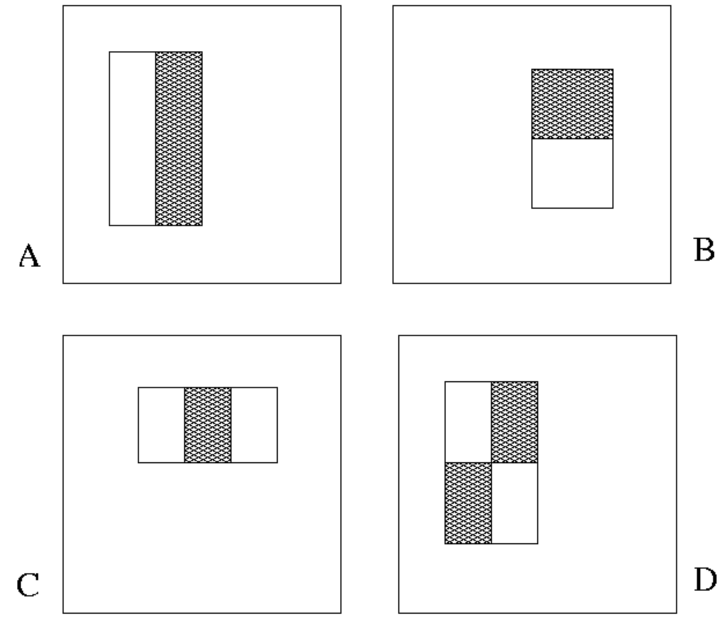
P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR 2001.

P. Viola and M. Jones. *Robust real-time face detection.* IJCV 57(2), 2004.

Image Features

"Rectangle filters"

aka Haar-like features

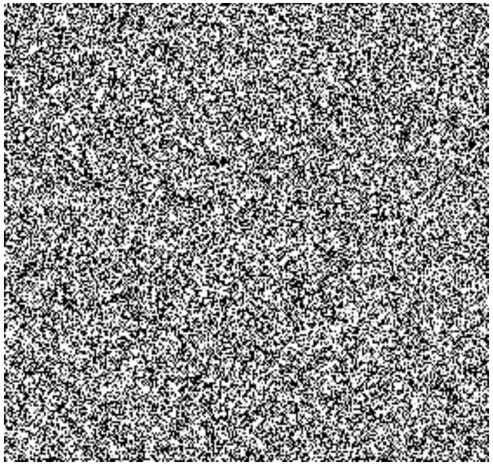


Value =

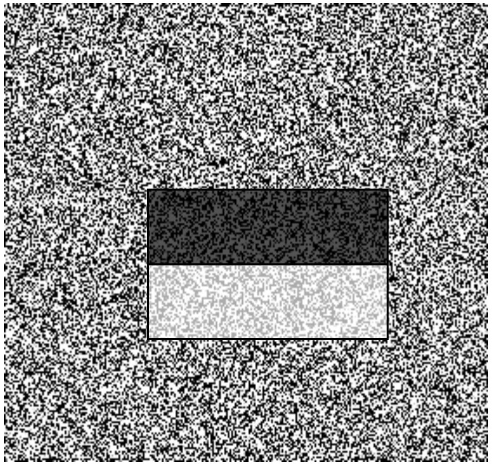
∑ (pixels in white area) –
∑ (pixels in black area)

Example

Source

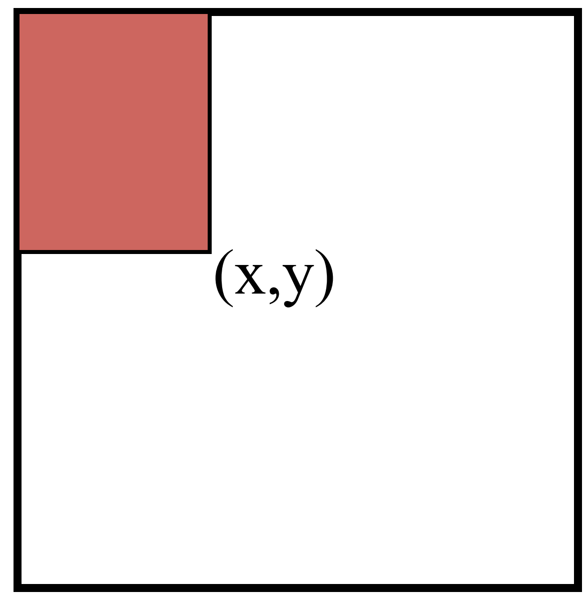


Result

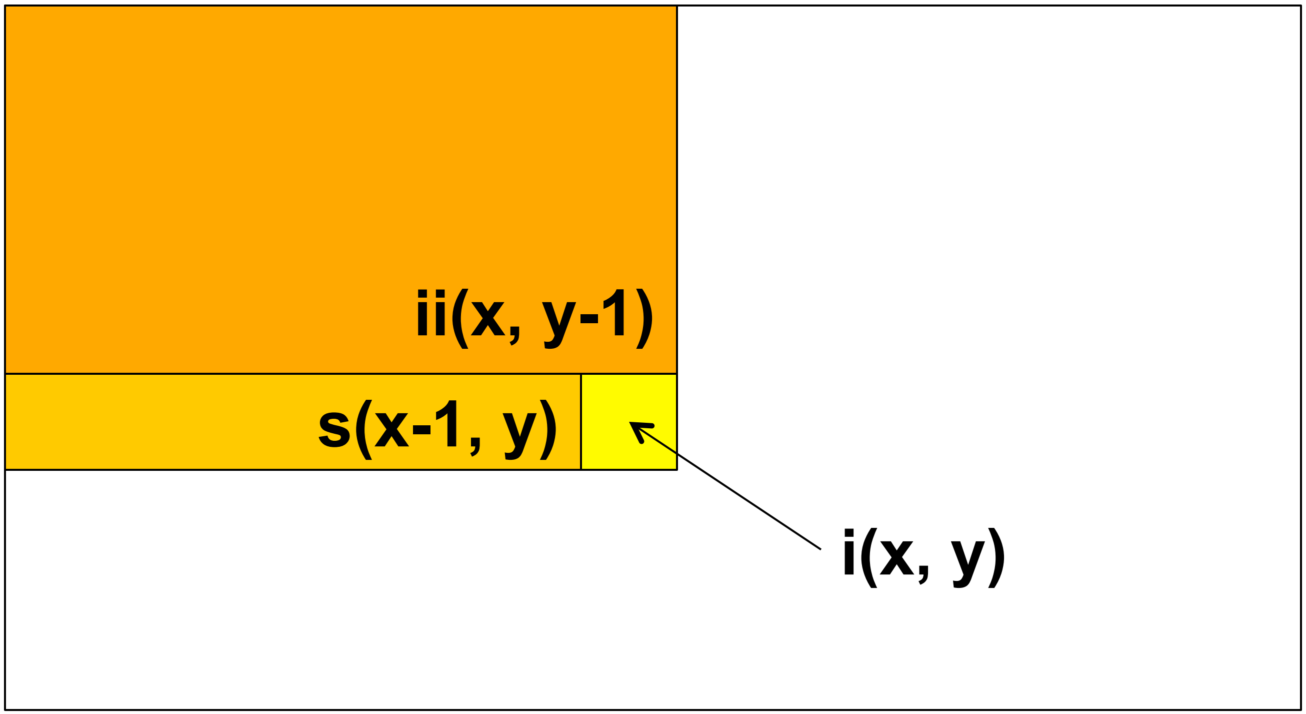


Fast computation with integral images

- The *integral image* computes a value at each pixel (x,y) that is the sum of the pixel values above and to the left of (x,y), inclusive
- This can quickly be computed in one pass through the image



Computing the integral image



- Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$
- Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$

MATLAB: `ii = cumsum(cumsum(double(i)), 2);`

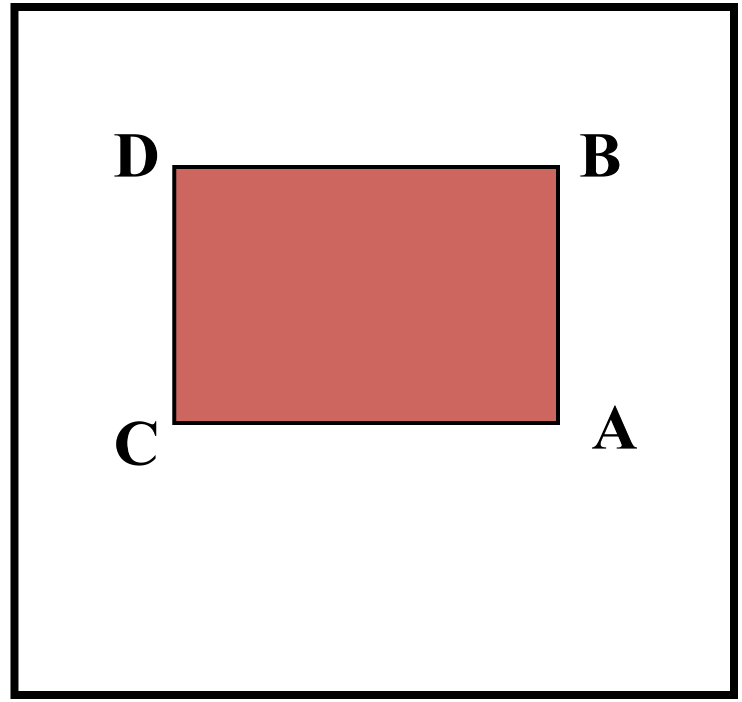
Computing sum within a rectangle

- Let A,B,C,D be the values of the integral image at the corners of a rectangle

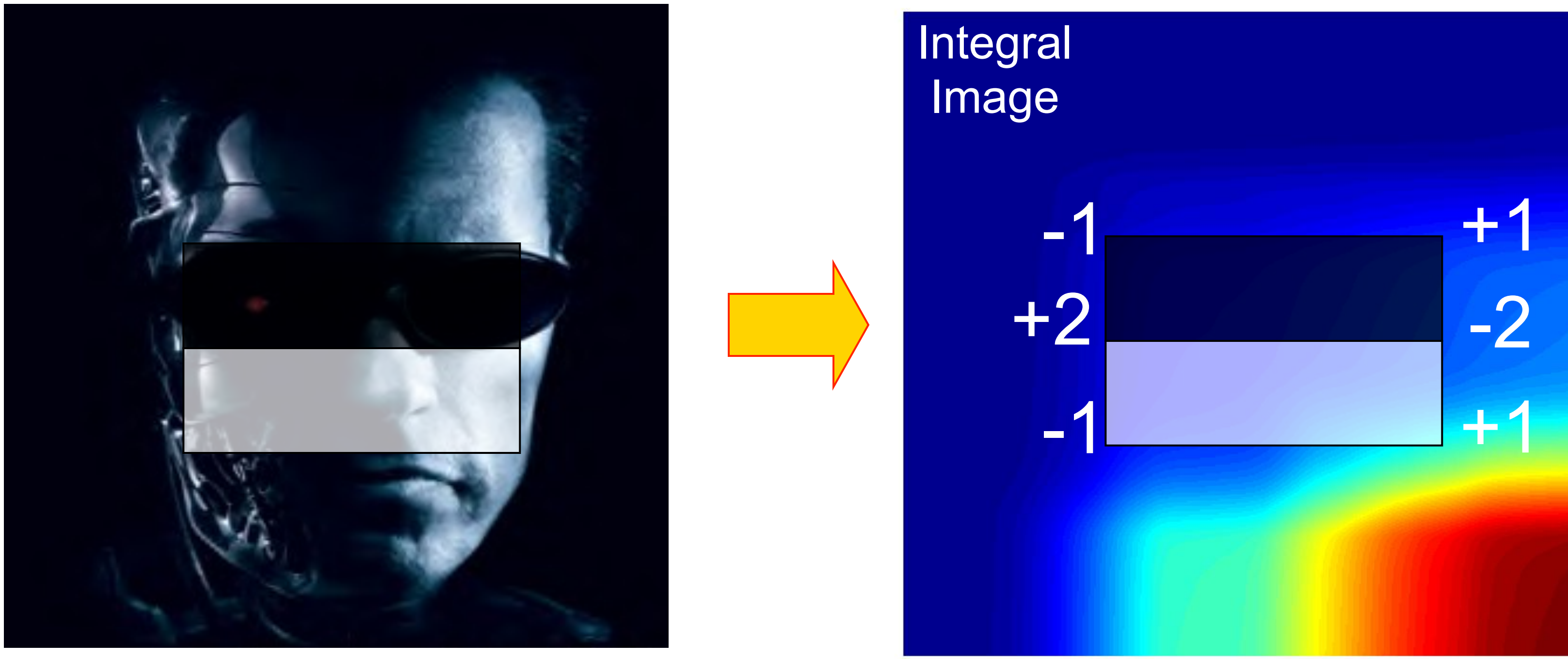
- Then the sum of original image values within the rectangle can be computed as:

  sum = A – B – C + D

- Only 3 additions are required for any size of rectangle!

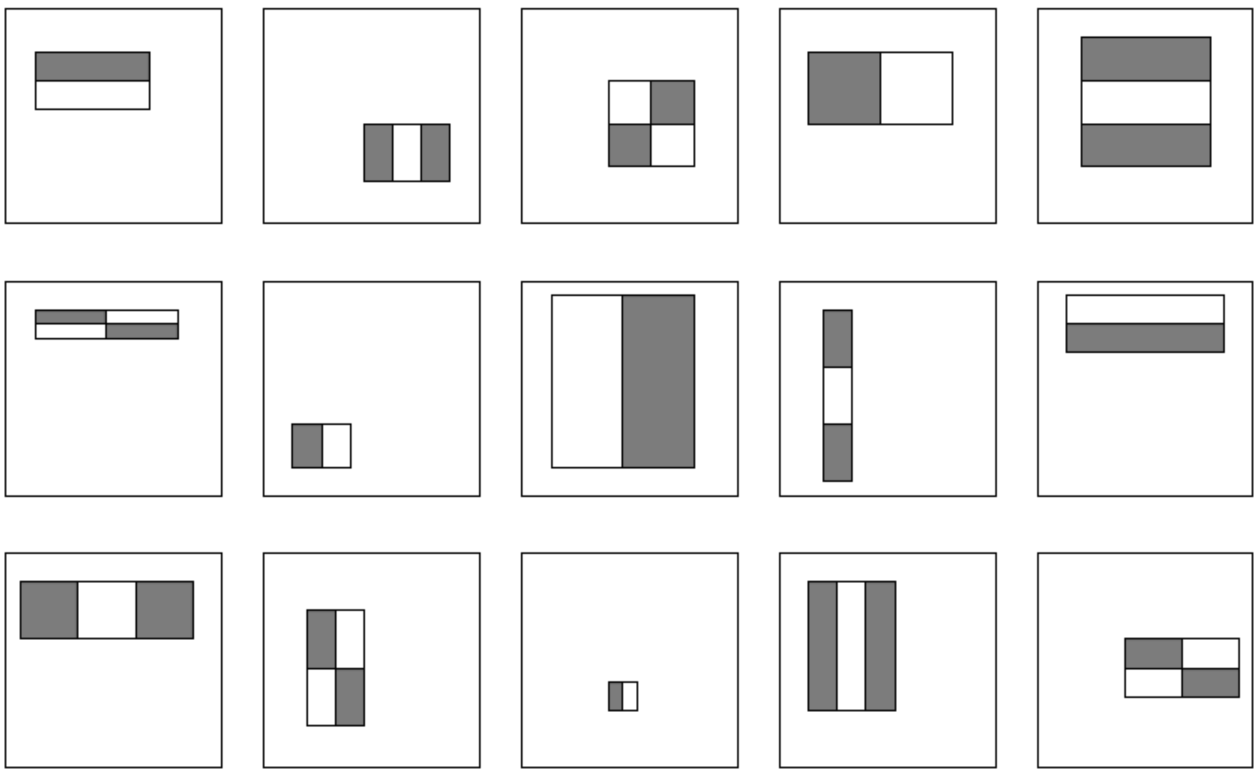


Example



Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!



Feature Discussion

- Primitive when compared with steerable filters, etc…
- Excellent for the detailed analysis of boundaries, image compression, and texture analysis.
- Sensitive to the presence of edges, bars, and other simple image structure
- Quite coarse: only three orientations (|, X, --)
- Overcomplete: 400 times, aspect ratio, location

Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!
- At test time, it is impractical to evaluate the entire feature set
- Can we create a good classifier using just a small subset of all possible features?
- How to select such a subset?

Boosting

- Boosting is a classification scheme that works by combining *weak learners* into a more accurate ensemble classifier
  - A weak learner needs only do better than chance
- Training consists of multiple *boosting rounds*
  - During each boosting round, we select a weak learner that does well on examples that were hard for the previous weak learners
  - "Hardness" is captured by weights attached to training examples

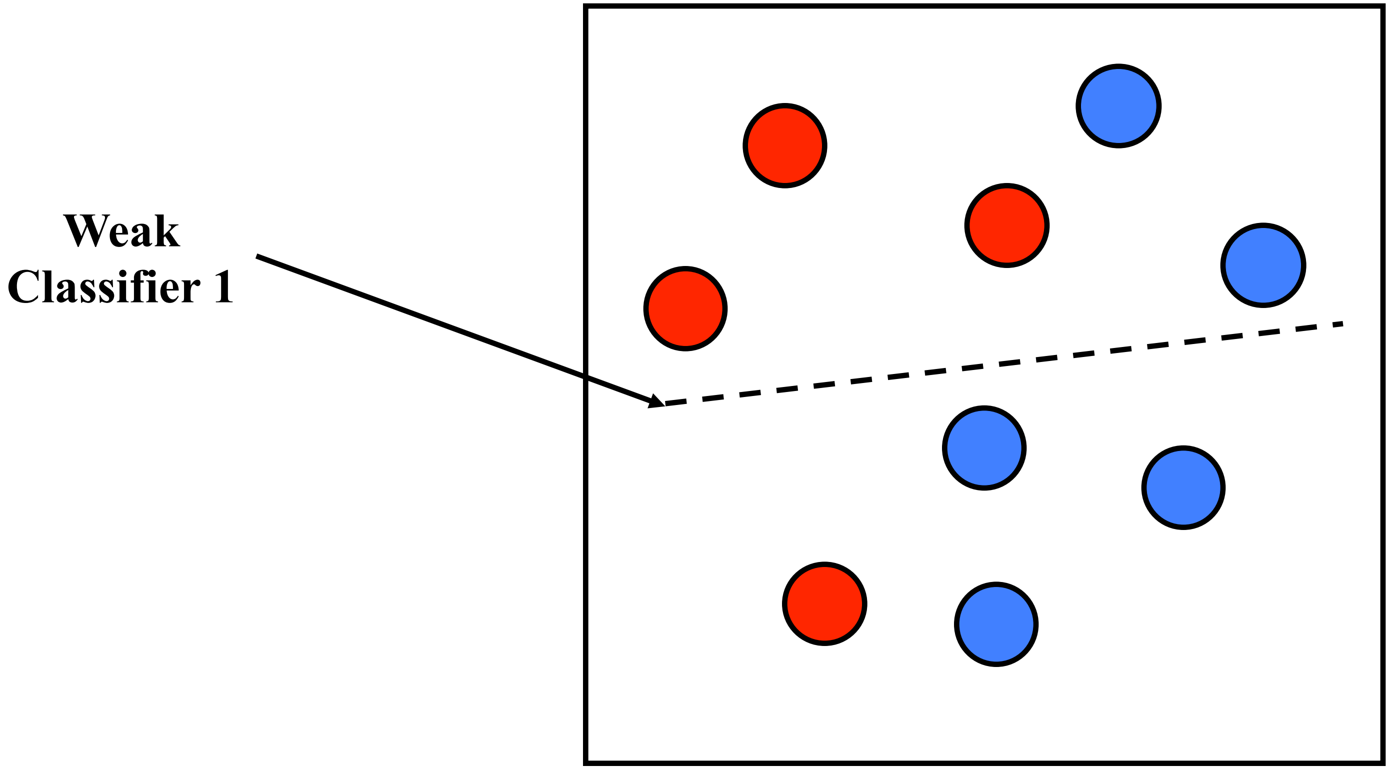
Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Training procedure

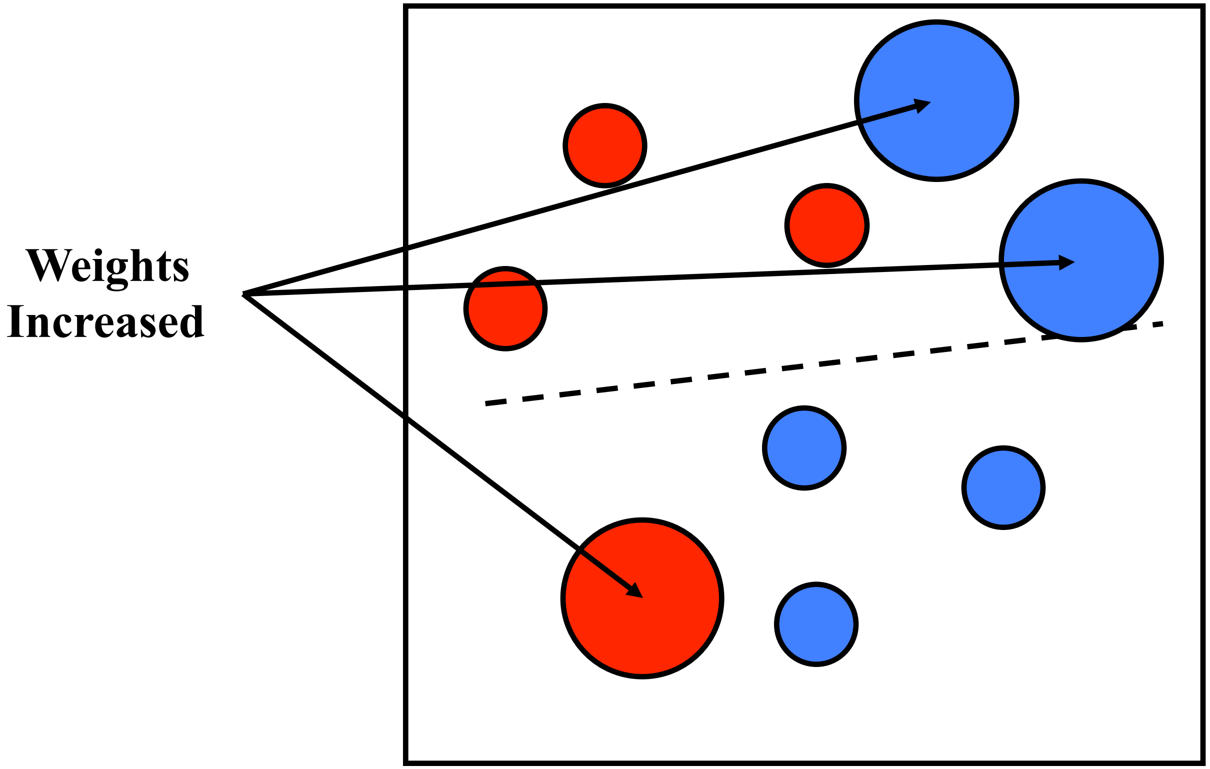
- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner that achieves the lowest *weighted* training error
  - Raise the weights of training examples misclassified by current weak learner
- Compute final classifier as linear combination of all weak learners (weight of each learner is directly proportional to its accuracy)
- Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme (e.g., AdaBoost)

Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

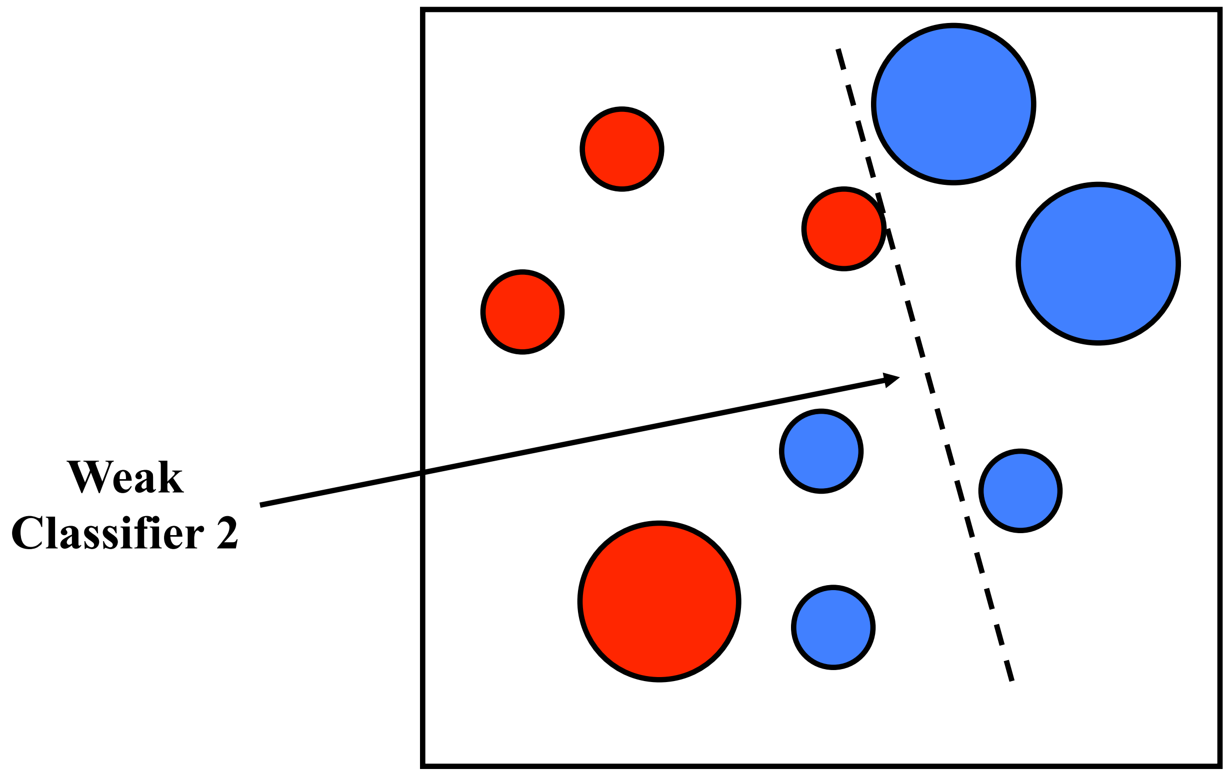
Boosting illustration



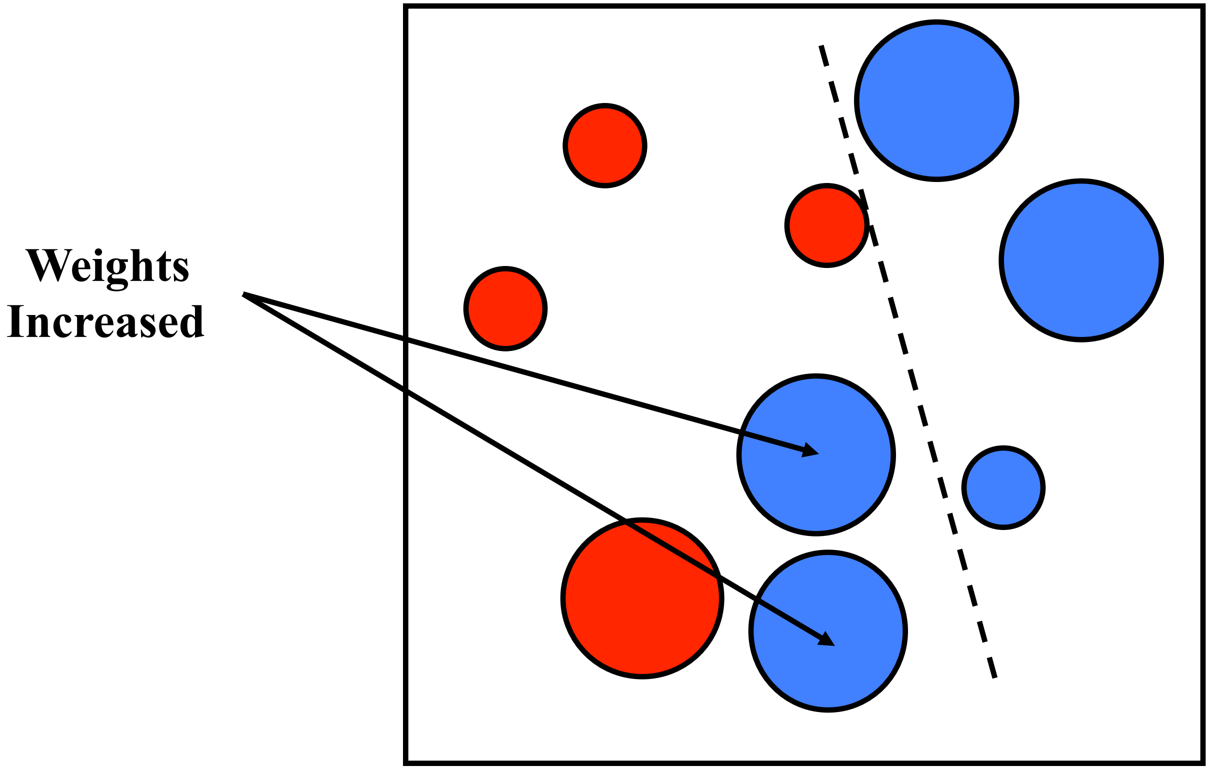
Boosting illustration



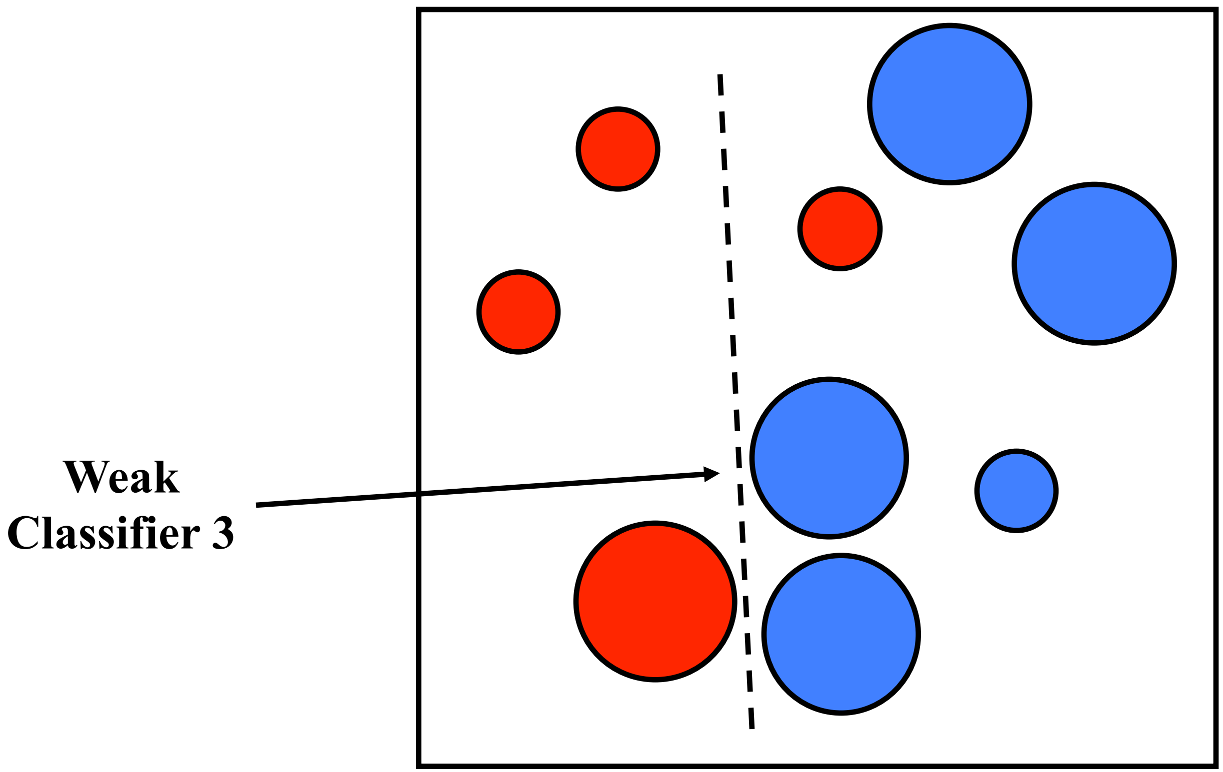
Boosting illustration



Boosting illustration

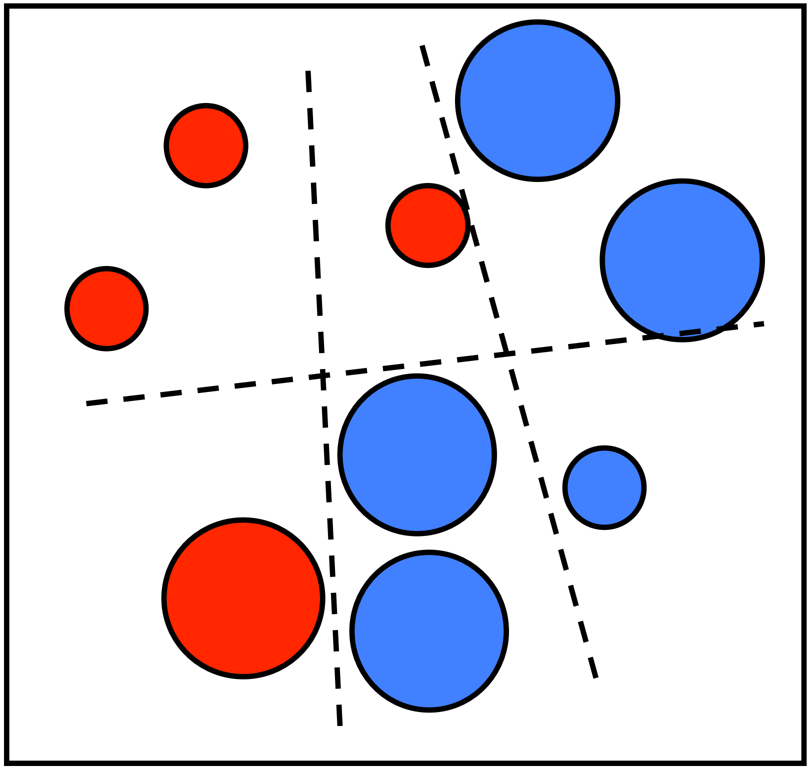


Boosting illustration



Boosting illustration

Final classifier is
a combination of weak
classifiers



Boosting - Pseudocode

- Given example images $(x_1, y_1), \dots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \dots, T$:

  1. Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
  2. Select the best weak classifier with respect to the weighted error

     $$\epsilon_t = \min_{f,p,\theta} \sum_i w_i \, |h(x_i, f, p, \theta) - y_i| \, .$$

  3. Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where f_t, p_t, and θ_t are the minimizers of ϵ_t.
  4. Update the weights:

     $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

Boosting for face detection

- Define weak learners based on rectangle features

value of rectangle feature

$$h_t(x) = \begin{cases} 1 & \text{if } f_t(x) > \theta_t \\ 0 & \text{otherwise} \end{cases}$$

threshold

window

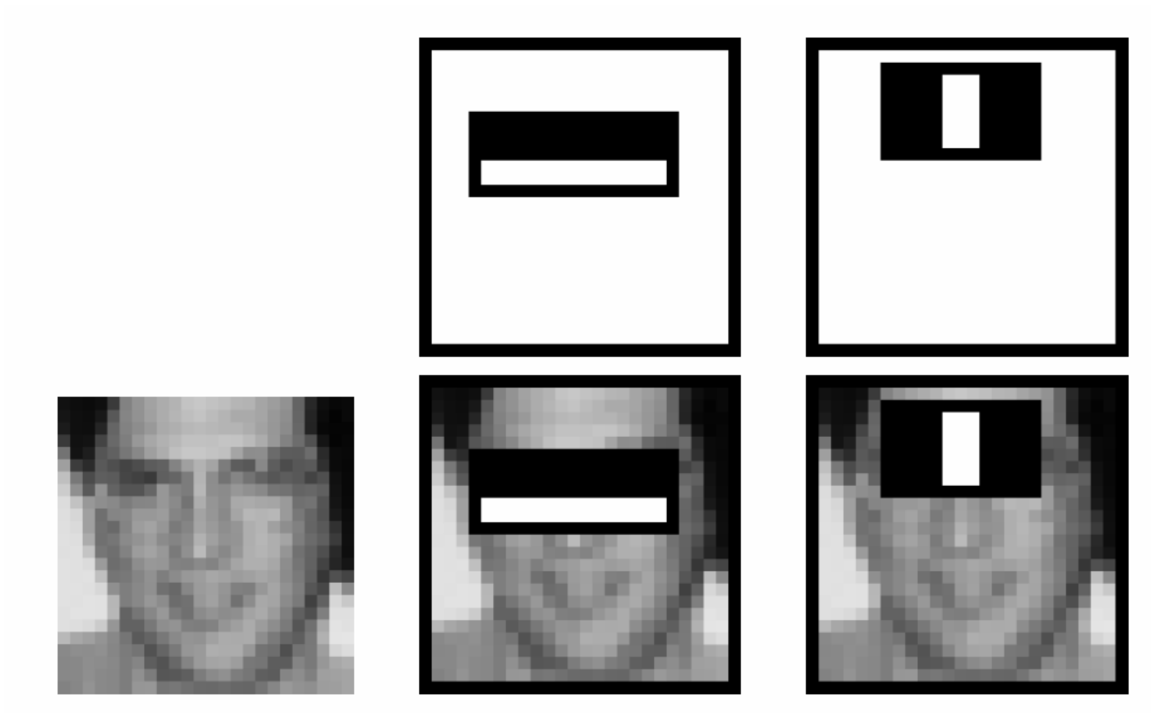
Boosting for face detection

- Define weak learners based on rectangle features

- For each round of boosting:
  - Evaluate each rectangle filter on each example
  - Select best threshold for each filter
  - Select best filter/threshold combination
  - Reweight examples

Boosting for face detection

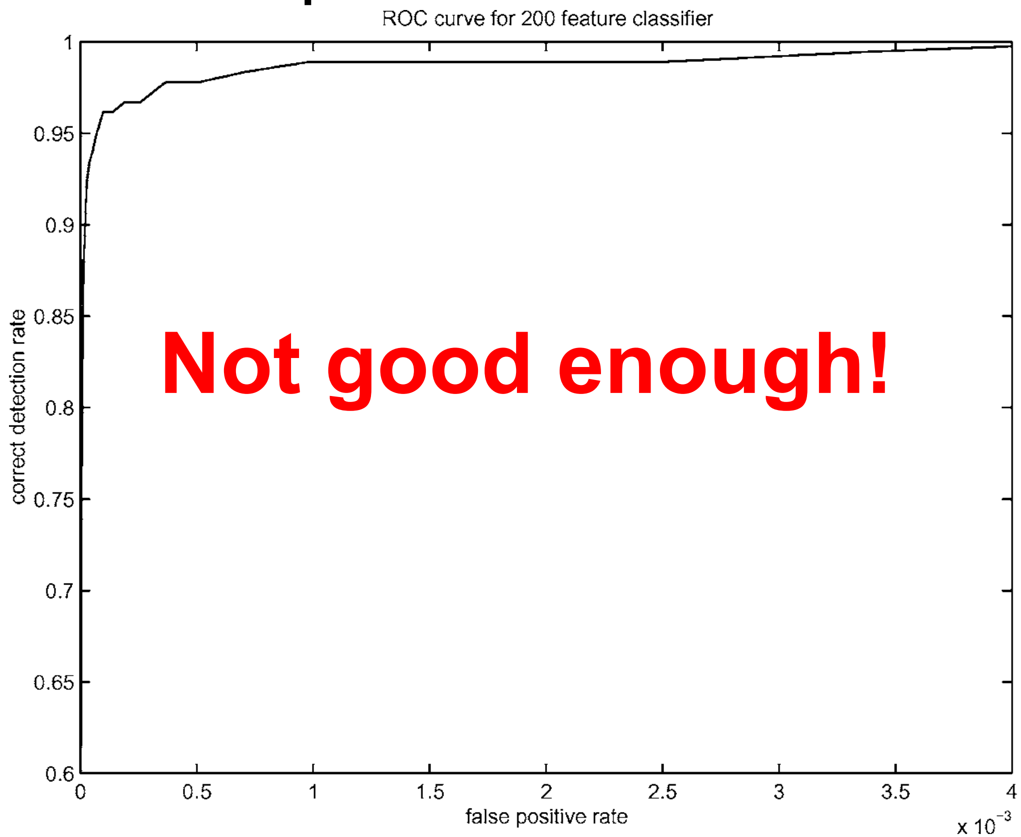
- First two features selected by boosting:



- This feature combination can yield 100% detection rate and 50% false positive rate

Boosting for face detection

- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084



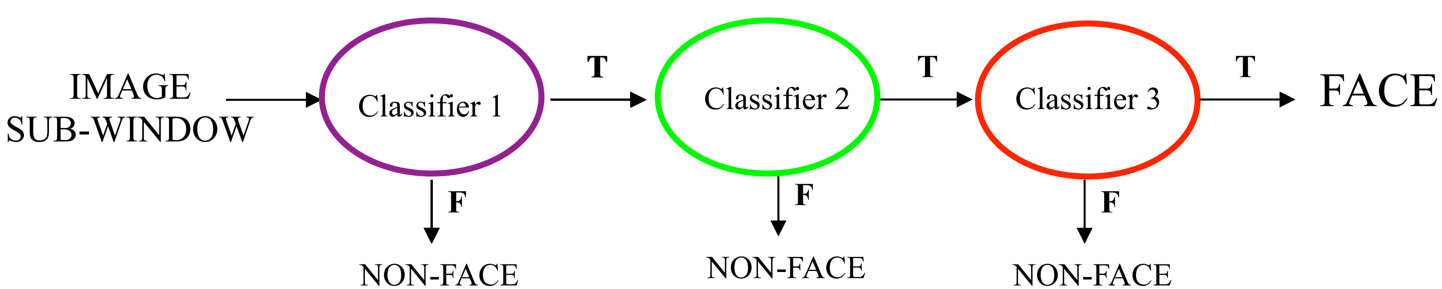
Receiver operating characteristic (ROC) curve

Attentional cascade

- We start with simple classifiers which reject many of the negative sub-windows while detecting almost all positive sub-windows

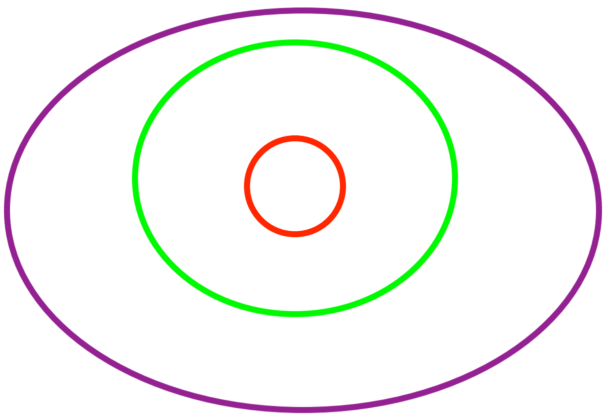
- Positive response from the first classifier triggers the evaluation of a second (more complex) classifier, and so on

- A negative outcome at any point leads to the immediate rejection of the sub-window

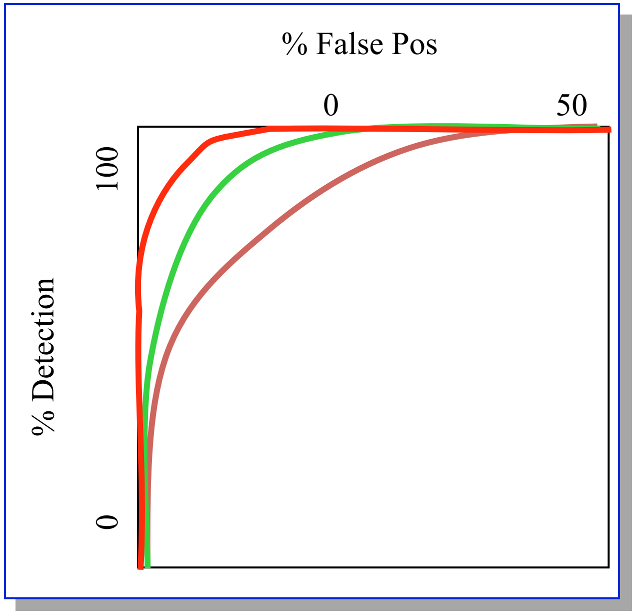


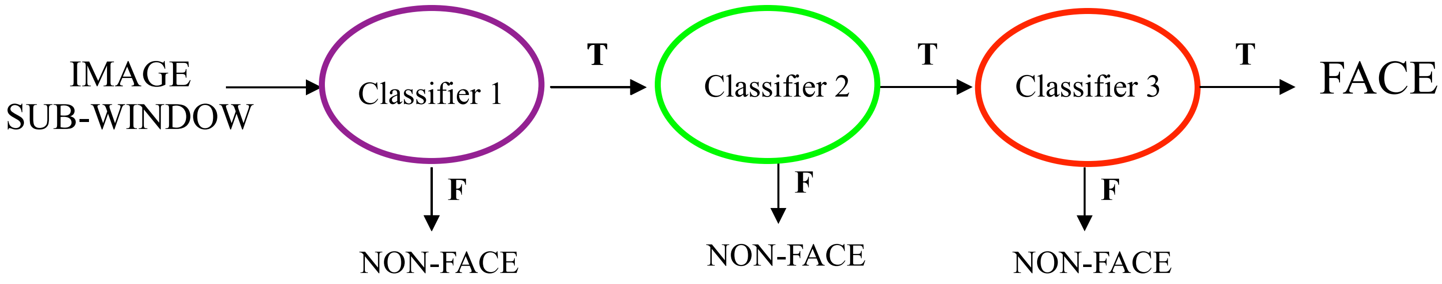
Attentional cascade

- Chain classifiers that are progressively more complex and have lower false positive rates:



Receiver operating characteristic

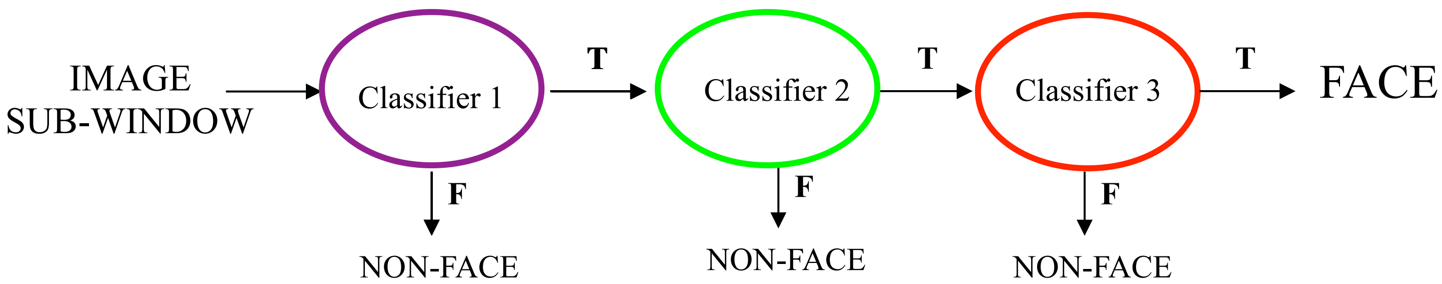




Attentional cascade

- The detection rate and the false positive rate of the cascade are found by multiplying the respective rates of the individual stages

- A detection rate of 0.9 and a false positive rate on the order of 10^{-6} can be achieved by a 10-stage cascade if each stage has a detection rate of 0.99 ($0.99^{10} \approx 0.9$) and a false positive rate of about 0.30 ($0.3^{10} \approx 6\times10^{-6}$)



Training the cascade

- Set target detection and false positive rates for each stage

- Keep adding features to the current stage until its target rates have been met

- If the overall false positive rate is not low enough, then add another stage

The implemented system

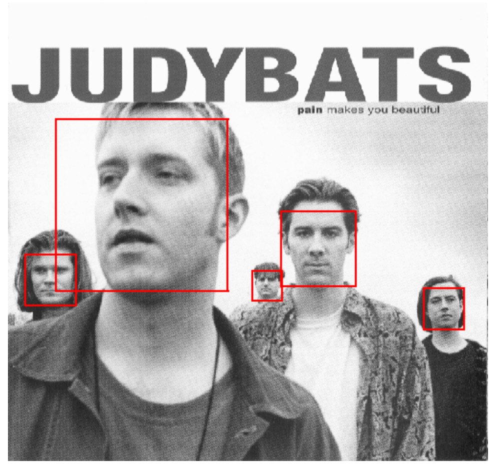
- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
  - Faces are normalized
    - Scale, translation

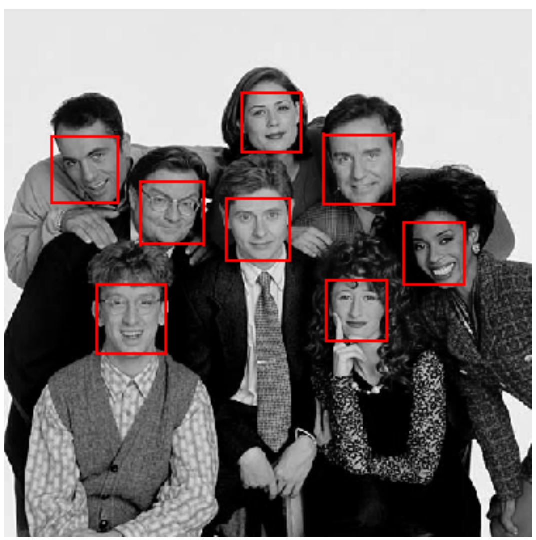
- Many variations
  - Across individuals
  - Illumination
  - Pose

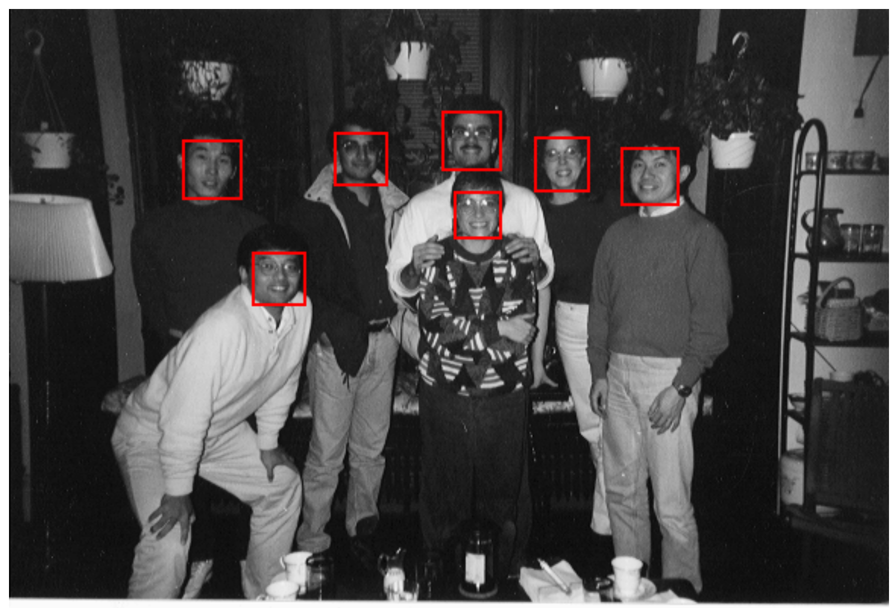
System performance

- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous detector of comparable accuracy (Rowley et al., 1998)

Output of Face Detector on Test Images

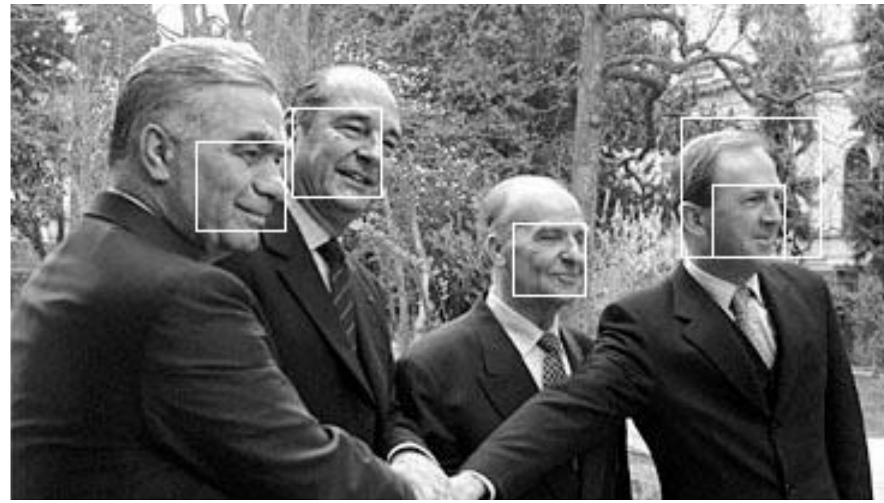






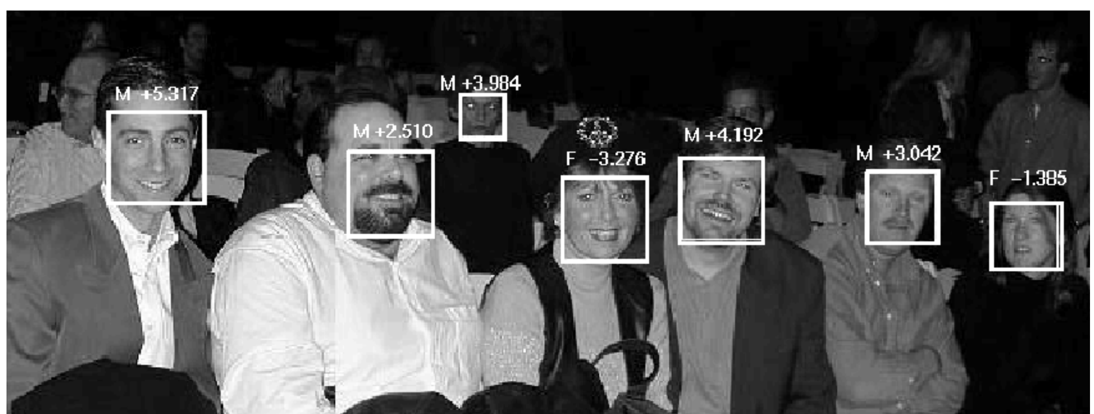
Other detection tasks

Facial Feature Localization

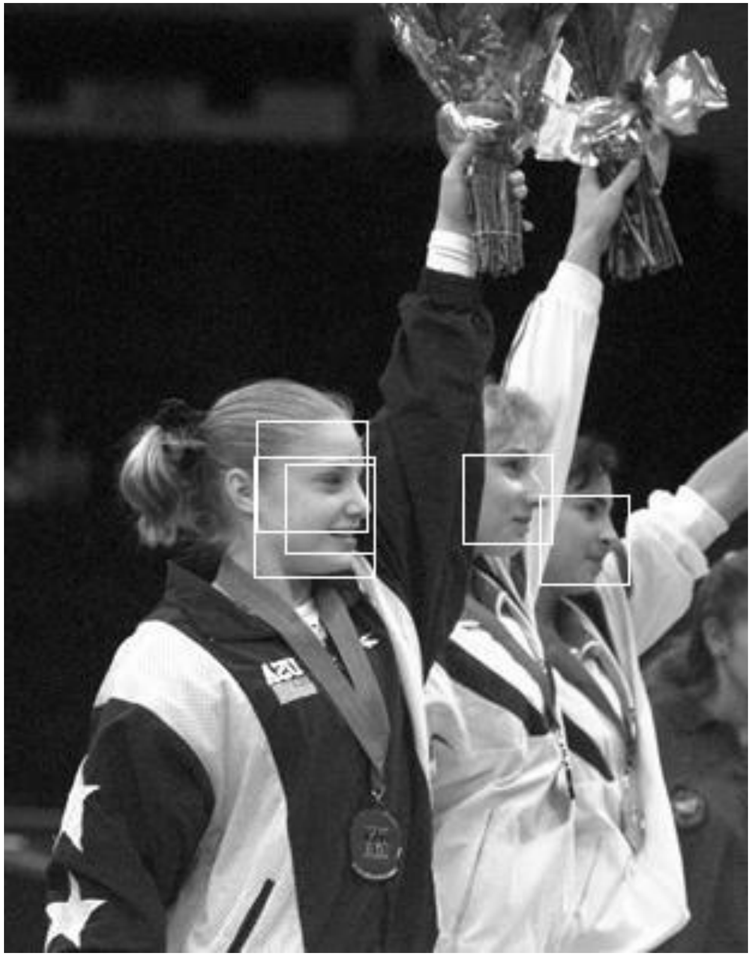


Profile Detection

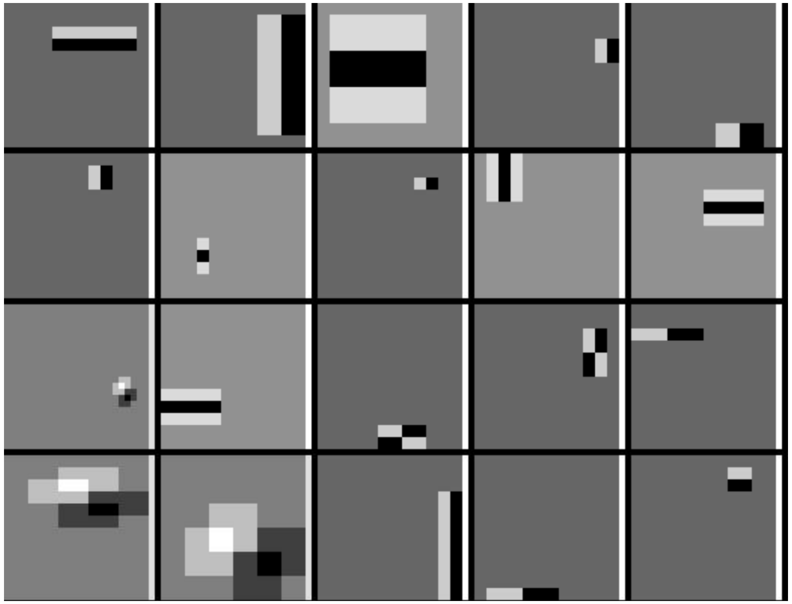
Male vs. female



Profile Detection



Profile Features



Summary: Viola/Jones detector

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows