Tensor Methods for Machine Learning, Computer Vision, and Computer Graphics

Part I: Factorizations and Statistical Modeling/Inference

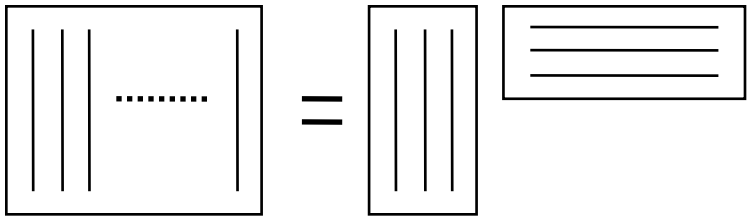
Amnon Shashua

School of Computer Science & Eng.
The Hebrew University

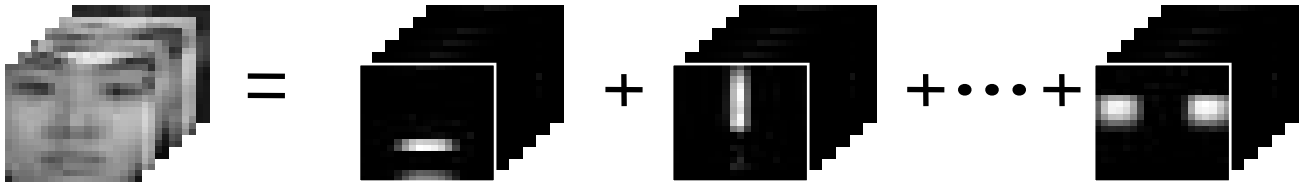
Factorizations of Multi-Dimensional Arrays

Normally: factorize the data into a lower dimensional space in order to describe the original data in a concise manner.

$G \in R^{d_1 \times d_2}$ $U \in R^{d_1 \times k}$ $V \in R^{k \times d_2}$



$$G = \sum_{j=1}^{k} u^j \otimes v^j$$



$$G = \sum_{j=1}^{k} u_1^j \otimes \ldots \otimes u_n^j$$

Focus of Lecture:

Factorization of empirical joint distribution ⟷ Latent Class Model

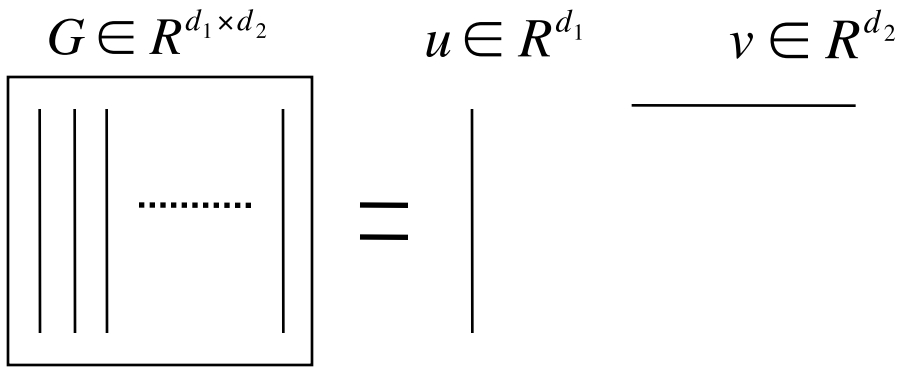
Factorization of symmetric forms ⟷ Probabilistic clustering.

Factorization of partially-symmetric forms ⟷ Latent clustering

N-way array decompositions

A rank=1 matrix G is represented by an outer-product of two vectors:

$$G_{ij} = u_i v_j \qquad G = uv^T = u \otimes v$$



A rank=1 tensor (n-way array) $G \in R^{d_1 \times d_2 \times \ldots \times d_n}$ is represented by an outer-product of n vectors $u_1, \ldots, u_n$

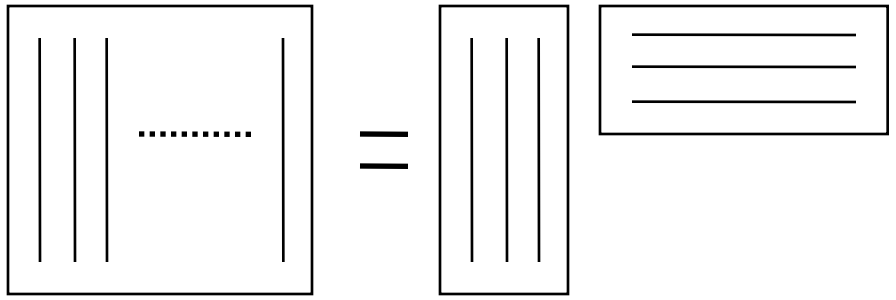
$$G_{i_1, i_2, \ldots, i_n} = u_{1,i_1} u_{2,i_2} \ldots u_{n,i_n} \qquad G = u_1 \otimes u_2 \otimes \ldots \otimes u_n$$

N-way array decompositions

A matrix G is of (at most) rank=k if it can be represented by a sum of k rank-1 matrices:

$$G = \sum_{j=1}^{k} u^j \otimes v^j$$

$G \in R^{d_1 \times d_2}$ $U \in R^{d_1 \times k}$ $V \in R^{k \times d_2}$

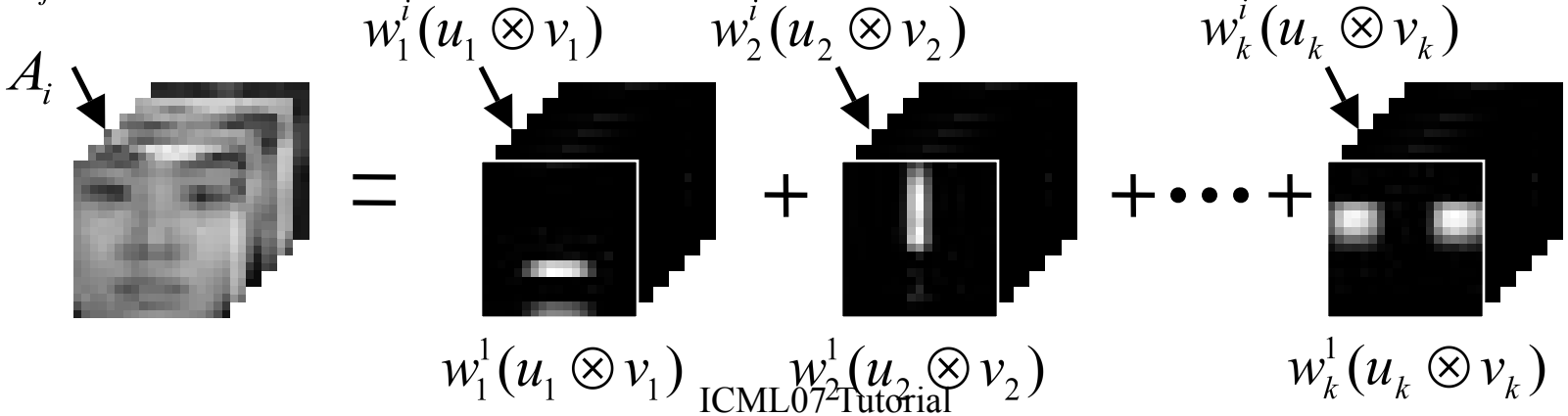


A tensor G is (at most) rank=k if it can be represented by a sum of k rank-1 tensors:

$$G = \sum_{j=1}^{k} u_1^j \otimes \ldots \otimes u_n^j$$

Example:

$$G = \sum_{j=1}^{k} u^j \otimes v^j \otimes w^j$$



N-way array Symmetric Decompositions

A symmetric rank=1 matrix G:

$$G_{ij} = u_i u_j \qquad G = uu^\top = u \otimes u$$

A symmetric rank=k matrix G:

$$G = \sum_{j=1}^{k} u_j \otimes u_j = \sum_{j=1}^{k} u_j u_j^\top = UU^\top$$

A super-symmetric rank=1 tensor (n-way array) is represented by an outer-product of n copies of a single vector

$$G \in R^D, \quad D = [m] \times \cdots [m] = [m]^{\times n}$$
$$u \in R^m$$

$$G_{i_1,\ldots,i_n} = u_{i_1} \cdots u_{i_n} \qquad G = u \otimes \cdots \otimes u = u^{\otimes n}$$

A super-symmetric tensor described as sum of k super-symmetric rank=1 tensors:

$$G = \sum_{j=1}^{k} u_j \otimes u_j \otimes \cdots \otimes u_j = \sum_{j=1}^{k} u_j^{\otimes n}$$

is (at most) rank=k.

General Tensors

Latent Class Models

Reduced Rank in Statistics

Let X_1 and X_2 be two random variables taking values in the sets $[d_i] = \{1, \ldots, d_i\}$

The statement X_1 is independent of X_2, denoted by $X_1 \perp X_2$ means:

$$P(X_1, X_2) = P(X_1)P(X_2)$$

$P(X_1, X_2)$ is a 2D array (a matrix) $G_{ij} = P(X_1 = i, X_2 = j)$

$P(X_1)$ is a 1D array (a vector) $u_i = P(X_1 = i)$

$P(X_2)$ is a 1D array (a vector) $v_j = P(X_2 = j)$

$X_1 \perp X_2$ means that $G_{ij} = u_i v_j$ is a rank=1 matrix $G = u \otimes v$

Reduced Rank in Statistics

Let $X_1,\ldots,X_n$ be random variables taking values in the sets $[d_i]=\{1,\ldots,d_i\}$

The statement $X_1\perp\ldots.\perp X_n$ means:

$$P(X_1,\ldots,X_n)=P(X_1)\cdots P(X_n)$$

$P(X_1,\ldots,X_n)$ is a n-way array (a tensor) $G_{i_1,i_2,\ldots,i_n}=P(X_1=i_1,\ldots,X_n=i_n)$

$P(X_j)$ is a 1D array (a vector) $\vec{u}_j$ whose entries $u_{j,i}=P(X_j=i)$

$X_1\perp\ldots.\perp X_n$ means that $G_{i_1,\ldots,i_n}=u_{1,i_1}u_{2,i_2}\cdots u_{n,i_n}$ is a rank=1 tensor

$$G=\vec{u}_1\otimes\vec{u}_2\otimes\ldots\otimes\vec{u}_n$$

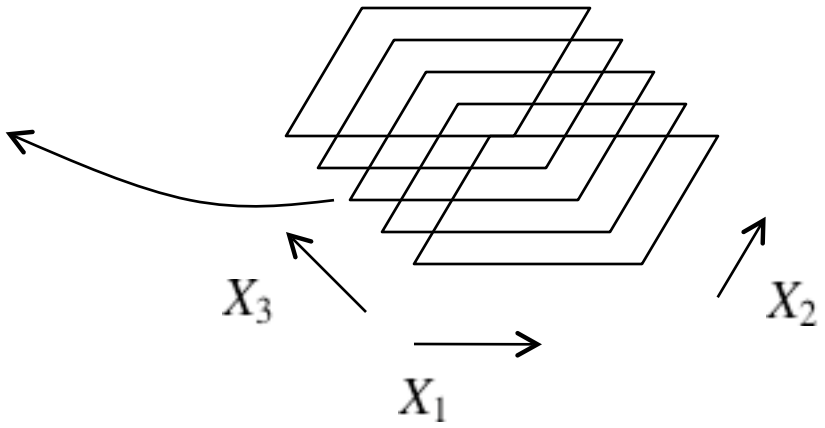
Reduced Rank in Statistics

Let X_1, X_2, X_3 be three random variables taking values in the sets $[d_i] = \{1, \ldots, d_i\}$

The conditional independence statement $X_1 \perp X_2 \mid X_3$ means:

$$P(X_1, X_2 \mid X_3 = i) = P(X_1 \mid X_3 = i) P(X_2 \mid X_3 = i)$$

Slice $X_3 = i$ is a rank=1 matrix



Reduced Rank in Statistics: Latent Class Model

Let $X_1,\ldots,X_n$ be random variables taking values in the sets $[d_i]=\{1,\ldots,d_i\}$

Let Y be a "hidden" random variable taking values in the set $\{1,\ldots,k\}$

The "observed" joint probability n-way array is:

$$P(X_1,\ldots,X_n)=\sum_{j=1}^{k}P(X_1,\ldots,X_n,Y=j)=\sum_{j=1}^{k}P(X_1,\ldots,X_n\mid Y=j)P(y=j)$$

A statement of the form $X_1\perp X_2\perp\ldots\perp X_n\mid Y$ translates to the algebraic statement

About the n-way array $P(X_1,\ldots,X_n)$ having tensor-rank equal to k

Reduced Rank in Statistics: Latent Class Model

$P(X_1,....,X_n)$ is a n-way array $G \geq 0$ $\|G\|_1 = 1$

$P(X_i \mid Y = j)$ is a 1D array (a vector) u_i^j $\|u_i^j\|_1 = 1$

$P(X_1,...,X_n \mid Y = j)$ is a rank-1 n-way array $\otimes_{i=1}^n u_i^j$

$P(Y)$ is a 1D array (a vector) σ $\|\sigma\|_1 = 1$

$$\min_{u_i^j,\sigma} \|G - \sum_{j=1}^{k} \sigma_j \otimes_{i=1}^n u_i^j\|^2 \qquad s.t \quad u_i^j \geq 0,\ \sigma \geq 0,\ \|u_i^j\|_1 = 1,\ \|\sigma\|_1 = 1$$

Reduced Rank in Statistics: Latent Class Model

$$\min_{u_i^j,\sigma} \|G - \sum_{j=1}^{k} \sigma_j \otimes_{i=1}^{n} u_i^j\|^2 \qquad s.t \quad u_i^j \geq 0,\ \sigma \geq 0,\ \|u_i^j\|_1 = 1,\ \|\sigma\|_1 = 1$$

for n=2:

$$\min_{U,D,V} \|G - UDV^\top\|_F^2 \qquad s.t. \quad U^\top 1 = 1,\ V1 = 1,\ 1^\top D1 = 1$$

$$U \geq 0,\ D = diag(\sigma_1, ..., \sigma_k) \geq 0,\ , V \geq 0$$

$$U = [u_1, ..., u_k] \qquad V = \begin{bmatrix} v_1^\top \\ . \\ . \\ . \\ v_k^\top \end{bmatrix} \qquad UDV^\top = \sum_{j=1}^{k} \sigma_j u_j v_j^\top$$

Reduced Rank in Statistics: Latent Class Model

$$\min_{U,D,V} \|G - UDV^\top\|_F^2 \qquad s.t. \quad U^\top 1 = 1,\ V1 = 1,\ 1^\top D1 = 1$$

$$U \geq 0,\ D = diag(\sigma_1, ..., \sigma_k) \geq 0,\ , V \geq 0$$

$$G_{ij} = P(X_1 = a_i, X_2 = b_j)$$

$$U = [P(X_1 \mid Y = y_1), ..., P(X_1 \mid Y = y_k)]$$

$$V = \begin{bmatrix} P(X_2 \mid Y = y_1)^\top \\ \cdot \\ \cdot \\ \cdot \\ P(X_2 \mid Y = y_k)^\top \end{bmatrix} \qquad D = \begin{bmatrix} P(Y = y_1) & & \\ & \cdot & \\ & & \cdot \\ & & P(Y = y_k) \end{bmatrix}$$

Reduced Rank in Statistics: Latent Class Model

$$\min_{U,D,V} loss(G, UDV^\top) \quad s.t. \quad U^\top 1 = 1,\ V1 = 1,\ 1^\top D1 = 1$$
$$U \geq 0,\ D = diag(\sigma_1, ..., \sigma_k) \geq 0,\ , V \geq 0$$

when loss() is the relative-entropy:

$$loss(x, y) = RE(x, y) = \sum_i x_i \ln \frac{x_i}{y_i}$$

then the factorization above is called pLSA (Hofmann 1999)

Typical Algorithm: Expectation Maximization (EM)

Reduced Rank in Statistics: Latent Class Model

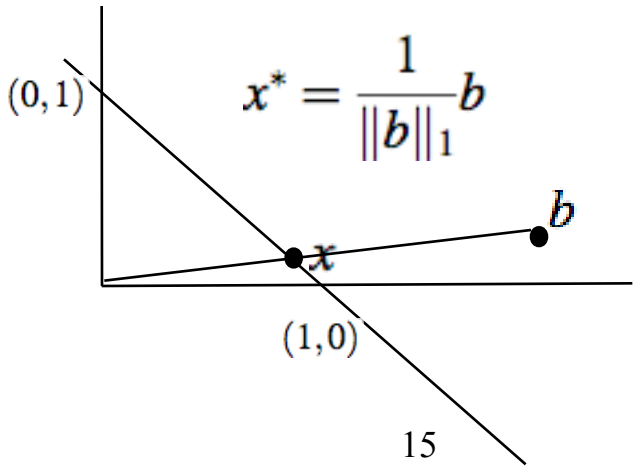
The Expectation-Maximization algorithm introduces auxiliary tensors W_j and alternates among the three sets of variables: $W_j,\ \sigma,\ u_i^j$

$$\min_{u_i^j,\sigma,W^j} \sum_{j=1}^{k} RE(W^j \circ G \mid\mid \sigma_j \otimes_i u_i^j) \quad s.t \quad u_i^j \geq 0,\ \sigma \geq 0,\ \|u_i^j\|_1 = 1,\ \|\sigma\|_1 = 1$$
$$W^j \geq 0,\ \sum_j W^j = 1$$

$(A \circ B)_i = A_i B_i$ Hadamard product.

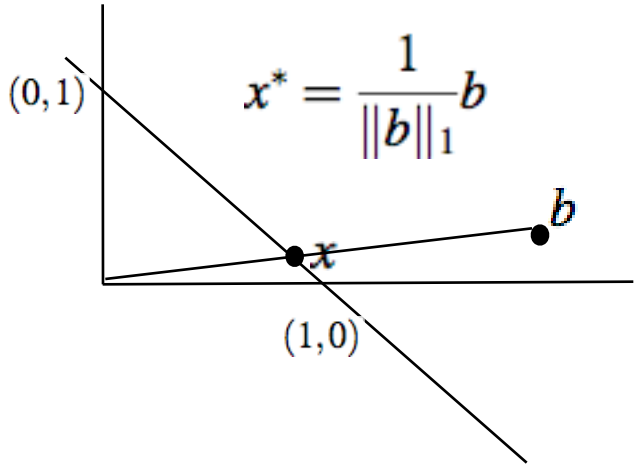
reduces to repeated application of "projection onto probability simplex"

$$\min_x RE(x \mid\mid b) \quad s.t. \quad x \geq 0,\ \sum_i x_i = 1$$

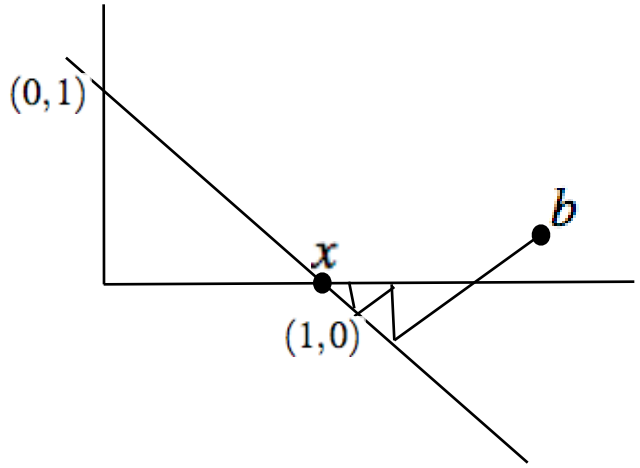


Reduced Rank in Statistics: Latent Class Model

An L2 version can be realized by replacing RE in the projection operation:



$$\min_x RE(x \mid\mid b) \quad s.t. \quad x \geq 0, \sum_i x_i = 1$$



$$\min_x \|x - b\|_2^2 \quad s.t. \quad x \geq 0, \sum_i x_i = 1$$

Reduced Rank in Statistics: Latent Class Model

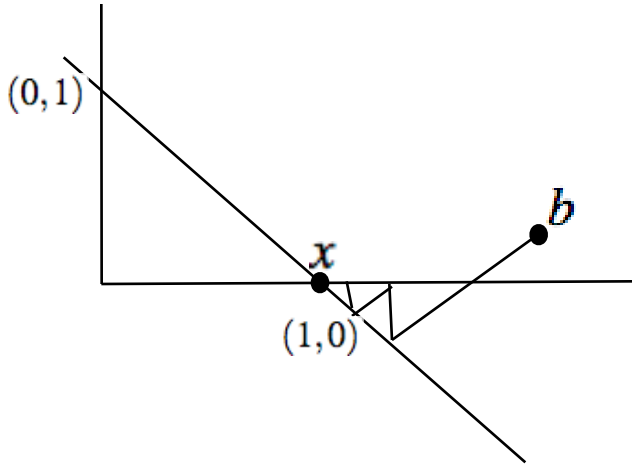
Let $G^t = G - \sum_{j \neq t} \sigma_j u_j v_j^\top$ and we wish to find $u_t \geq 0, \|u_t\|_1 = 1$

which minimizes: $\|G^t - \sigma_t u_t v_t^\top\|_F^2$

$$argmin_{u_t} \|G^t - \lambda_t u_t v_t^\top\|_F^2$$

$$= argmin_{u_t} \lambda_t^2 \|u_t\|^2 \|v_t\|^2 - 2\sigma_t u_t^\top G^t v_t$$

$$= argmin_{u_t} \|\frac{G^t v_t}{\sigma_t \|v_t\|^2} - u_t\|^2$$



$$\min_x \|x - b\|_2^2 \quad s.t. \quad x \geq 0, \sum_i x_i = 1$$

Reduced Rank in Statistics: Latent Class Model

To find $\sigma_1, ..., \sigma_k$ given $u_i,\ v_i,\ i = 1, ..., k$, we need to solve a convex program:

$$\min_{\sigma} \|A\sigma - b\|^2 \ \ s.t\ \ \sigma \geq 0,\ \|\sigma\|_1 = 1$$

$$A = [A_1, ..., A_k]$$

$$b = vec(G) \qquad A_t = vec(u_t v_t^\top)$$

Reduced Rank in Statistics: Latent Class Model

An equivalent representation:

$$\min_{U,V} \; loss(\bar{G},\; UV) \qquad s.t. \quad U \geq 0,\; V \geq 0,\; U^\top 1 = V^\top 1 = 1$$

Non-negative Matrix Factorization

$$\bar{G}_{ij} = P(X_1 = a_i \mid X_2 = b_j) \qquad \bar{G}^\top 1 = 1$$ (normalize columns of G)

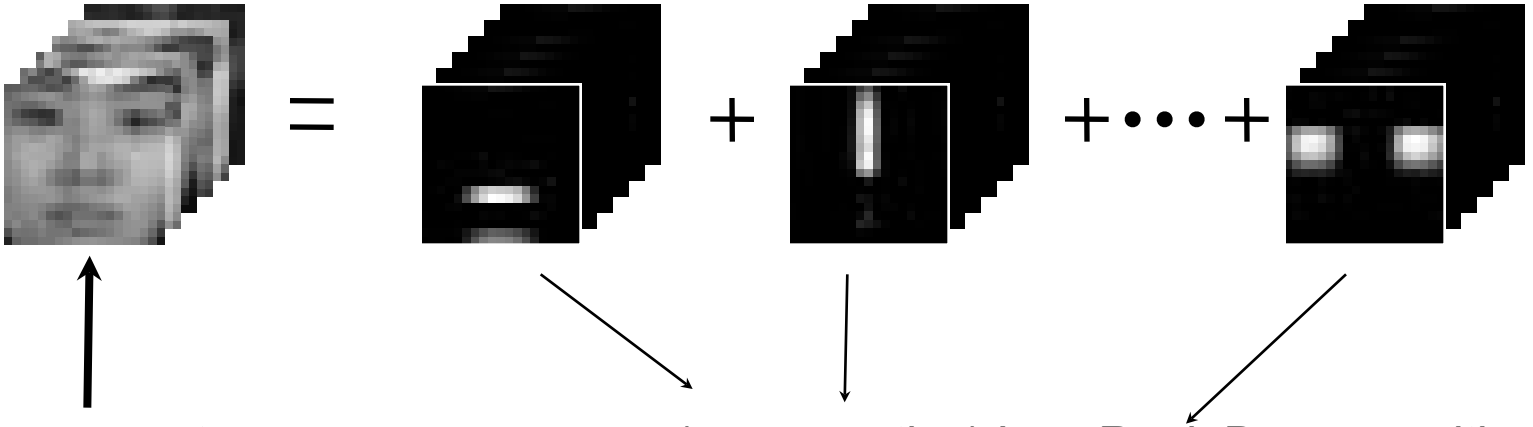
$$P(X_1 \mid X_2) = \sum_j P(X_1, Y = y_j \mid X_2) = \sum_j P(X_1 \mid Y = y_j, X_2) P(Y = y_j | X_2)$$
$$= \sum_j P(X_1 \mid Y = y_j) P(Y = y_j | X_2)$$

$$U = [P(X_1 \mid Y = y_1), ..., P(X_1 \mid Y = y_k)]$$

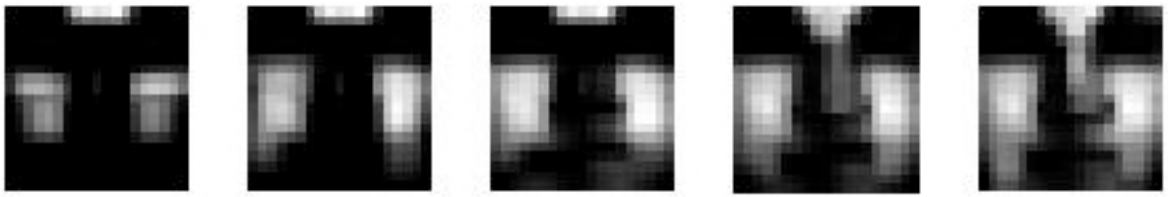
$$V = [P(Y|X_2 = b_1), ..., P(Y|X_2 = b_{d_2})]$$

(non-negative) Low Rank Decompositions

$$\min_{u_i^j,\sigma} \|G - \sum_{j=1}^{k} \sigma_j \otimes_{i=1}^{n} u_i^j\|^2 \qquad s.t \quad u_i^j \geq 0,\ \sigma \geq 0,\ \|u_i^j\|_1 = 1,\ \|\sigma\|_1 = 1$$



The rank-1 blocks tend to represent local parts of the image class



4 factors 8 factors 12 factors 16 factors 20 factors

Hierarchical build-up of "important" parts

(non-negative) Low Rank Decompositions

$$\min_{u_i^j,\sigma} \|G - \sum_{j=1}^{k} \sigma_j \otimes_{i=1}^{n} u_i^j\|^2$$

Example: The swimmer

Sample set (256)

Non-negative
Tensor Factorization

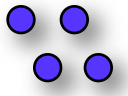
NMF

Super-symmetric Decompositions
Clustering over Hypergraphs

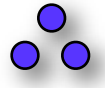
Matrices

Clustering data into k groups: Pairwise Affinity

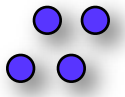
$x_1, \ldots x_m \in R^d$ input points



$K_{ij} = e^{-\|x_i - x_j\|^2/\sigma^2}$ input (pairwise) affinity value



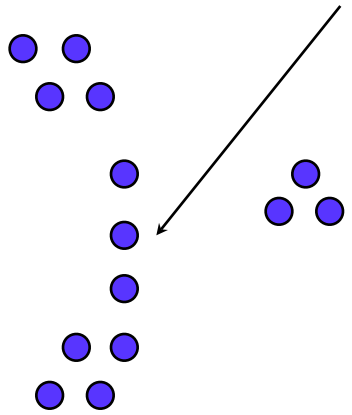
interpret K_{ij} as "the probability that x_i and x_j are clustered together"



$y_1, \ldots, y_m \in \{1, \ldots, k\}$ unknown class labels

Clustering data into k groups: Pairwise Affinity

k=3 clusters, this point does not belong to any cluster in a “hard” sense.



$X = x_1, \ldots x_m \in R^d$ input points

$K_{ij} = e^{-\|x_i - x_j\|^2/\sigma^2}$ input (pairwise) affinity value

interpret K_{ij} as “the probability that x_i and x_j are clustered together”

$y_1, \ldots, y_m \in \{1, \ldots, k\}$ unknown class labels

A probabilistic view:

$G_{ij} = P(y_i = j \mid X)$ probability that x_i belongs to the j’th cluster

note: $G \geq 0, \quad G1 = 1$

What is the (algebraic) relationship between the input matrix K and the desired G?

Clustering data into k groups: Pairwise Affinity

Assume the following conditional independence statements:

$$y_1 \perp \ldots \perp y_m \mid x_1, \ldots, x_m$$

$$K_{ij} = \sum_{r=1}^{k} P(y_i = r,\ y_j = r \mid X)$$

$$= \sum_{r=1}^{k} P(y_i = r \mid X) P(y_j = r \mid X)$$

$$= \sum_{r=1}^{k} G_{ir} G_{jr}$$

$$K = GG^{\top}, \quad G \geq 0, \quad G1 = 1$$

Probabilistic Min-Cut

$$K = GG^{\top}, \quad G \geq 0, \quad G1 = 1$$

A "hard" assignment requires that: $G^{\top}G = D = diag(n_1, \ldots, n_k)$

where $n_1, \ldots, n_k$ are the cardinalities of the clusters

Proposition: the feasible set of matrices G that satisfy $G \geq 0,\ G1 = 1,\ G^{\top}G = D$ are of the form:

$$G_{ij} = \begin{Bmatrix} 1 & x_i \in \psi_j \\ 0 & otherwise \end{Bmatrix}$$

$$\max_{G} tr(G^{\top}KG)\ s.t \quad G \geq 0,\ G1 = 1,\ G^{\top}G = D$$

equivalent to: $\max_{\psi_1,\psi_2} \sum_{(i,j)\in\psi_1} K_{ij} + \sum_{(i,j)\in\psi_2} K_{ij}$ → $\min_{\psi_1,\psi_2} \sum_{i\in\psi_1, j\in\psi_2} K_{ij}$

Min-cut formulation

Relation to Spectral Clustering

$$K = GG^\top, \quad G \geq 0, \quad G1 = 1$$

Add a "balancing" constraint: $G^\top 1 = (n/k)1$

put together $G1 = 1, \quad G^\top 1 = 1$ means that $GG^\top 1 = (n/k)1$

also, $GG^\top 1 = (n/k)1$ and $G^\top G = D$ means that $D = (n/k)I$

$$\max_G tr(G^\top KG) \; s.t \quad G \geq 0, \; GG^\top 1 = \frac{m}{k}1, \; G^\top G = \frac{m}{k}I$$

n/k can be dropped.

$$\max_G tr(G^\top KG) \; s.t \quad G \geq 0, \; GG^\top 1 = 1, \; G^\top G = I$$

Relax the balancing constraint by replacing K with the "closest" **doubly stochastic** matrix K' and **ignore** the non-negativity constraint:

$$\max_G tr(G^\top K'G) \; s.t \quad G^\top G = I$$

Relation to Spectral Clustering

$$\max_G tr(G^\top K' G) \; s.t \quad G^\top G = I$$

Let $D = diag(K1)$

Then: $K' = K - D + I$ is the closest D.S. in L1 error norm. → **Ratio-cuts**

$K' = D^{-1/2} K D^{-1/2}$ **Normalized-Cuts**

Proposition: iterating $K^{(t+1)} = D^{-1/2} K^{(t)} D^{-1/2}$ with $D = diag(K^{(t)}1)$ converges to the closest D.S. in KL-div error measure.

New Normalization for Spectral Clustering

$$\max_{G} tr(G^{\top} K' G) \; s.t \quad G^{\top} G = I$$

where

$$K' = argmin_{F} \|K - F\|_F^2 \;\; s.t. \;\; F \geq 0, \; F1 = 1, \; F = F^{\top}$$

we are looking for the closest doubly-stochastic matrix in least-squares sense.

New Normalization for Spectral Clustering

$$K' = argmin_F \|K - F\|_F^2 \;\; s.t. \;\; F \geq 0,\; F1 = 1,\; F = F^\top$$

to find K' as above, we break this into two subproblems:

$$P_1(X) = argmin_F \|X - F\|_F^2 \;\; s.t. \;\; F1 = 1,\; F = F^\top$$

$$P_2(X) = argmin_F \|X - F\|_F^2 \;\; s.t. \;\; F \geq 0$$

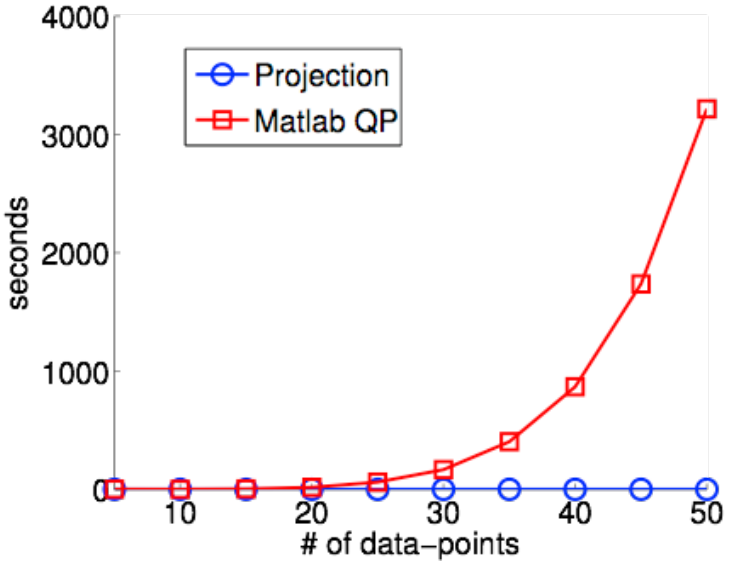
$$P_1(X) = X + \left(\frac{1}{n}I + \frac{1^\top X 1}{n^2}I - \frac{1}{n}X\right)11^\top - \frac{1}{n}11^\top X$$

$$P_2(X) = th_{\geq 0}(X)$$

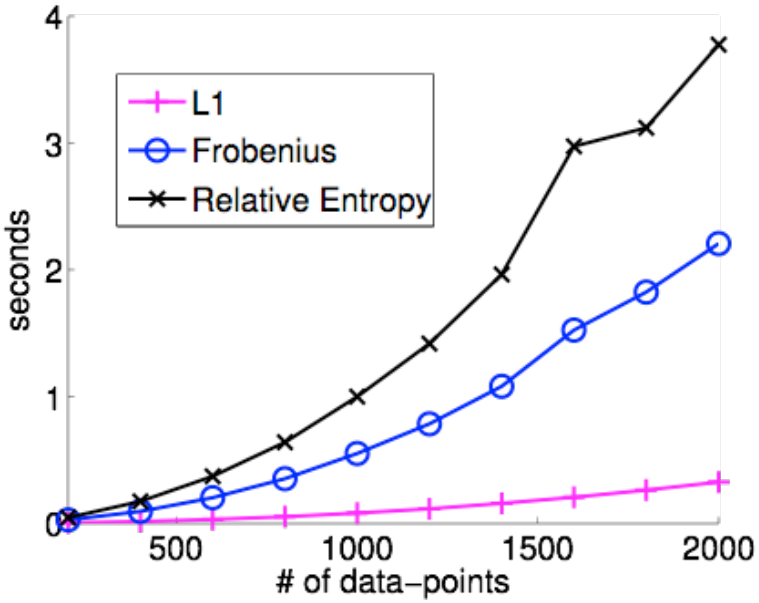
use the Von-Neumann successive projection lemma:

$$K' = P_1 P_2 P_1 P_2 \ldots P_1(K)$$

New Normalization for Spectral Clustering



successive-projection vs. QP solver



running time for the three normalizations

New Normalization for Spectral Clustering

Dataset	Kernel	Clusters	Size	Dim.	Lowest Error Rate				
					L1	Frobenius	RE	NCuts	None
SPECTF heart	RBF	2	349	44	27.5	**19.2**	27.5	27.5	29.5
Pima Indians Diabetes	RBF	2	768	8	36.2	35.2	**34.9**	35.2	35.4
Wine	RBF	3	178	13	38.8	**27.0**	34.3	29.2	27.5
SpamBase	RBF	2	4601	57	36.1	**30.3**	37.7	31.8	30.4
BUPA liver disorders	Poly	2	345	6	**37.4**	**37.4**	41.7	41.7	**37.4**
WDBC	Poly	2	569	30	18.8	**11.1**	37.4	37.4	18.8

UCI Data-sets

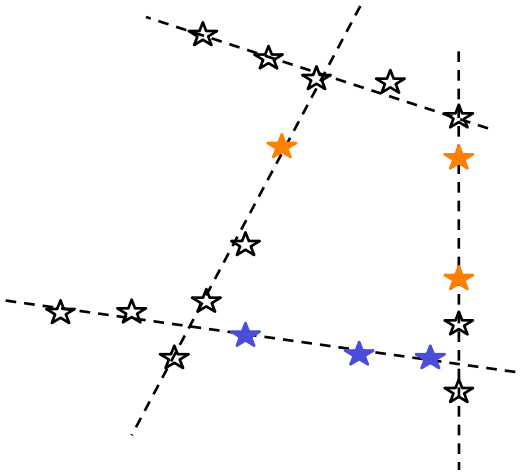
Dataset	Kernel	Clusters	Size	Dim.	Lowest Error Rate				
					L1	Frobenius	RE	NCuts	None
AML/ALL Leukemia	Poly	2	72	5	27.8	**16.7**	36.1	38.9	30.6
Lung	Poly	2	181	5	15.5	**9.9**	16.6	15.5	15.5
Prostate	RBF	2	136	5	40.4	**19.9**	43.4	40.4	40.4
Prostate Outcome	RBF	2	21	5	28.6	**4.8**	23.8	28.6	28.6

Cancer Data-sets

Super-symmetric Decompositions
Clustering over Hypergraphs

Tensors

Clustering data into k groups: Beyond Pairwise Affinity



A model selection problem that is determined by n-1 points can be described by a factorization problem of n-way array (tensor).

Example: clustering m points into k lines

$$D = \{x_1, \ldots, x_m\}$$

Input: K_{i_1,i_2,i_3} = the probability that $x_{i_1}, x_{i_2}, x_{i_3}$ belong to the same model (line).

Output: $g_{rs} = Pr(y_s = r \mid D)$ the probability that the point x_s belongs to the r'th model (line) $G = [g_1, .., g_k]$

Under the independence assumption: $y_1 \perp \ldots \perp y_m \mid x_1, \ldots, x_m$

$$P(y_{i_1} = r, y_{i_2} = r, y_{i_3} = r \mid D) = P(y_{i_1} = r \mid D) P(y_{i_2} = r \mid D) P(y_{i_3} = r \mid D)$$

$$K_{i_1,i_2,i_3} = \sum_{r=1}^{4} P(y_{i_1} = r \mid D) P(y_{i_2} = r \mid D) P(y_{i_3} = r \mid D) = \sum_{r=1}^{4} g_{r,i_1} g_{r,i_2} g_{r,i_3}$$

$$\rightarrow \quad K = \sum_{r=1}^{\kappa} g_r \otimes g_r \otimes g_r$$ is a 3-dimensional super-symmetric tensor of rank=4

Clustering data into k groups: Beyond Pairwise Affinity

General setting: clusters are defined by n-1 dim subspaces, then for each n-tuple of points $x_{i_1},\ldots,x_{i_n}$ we define an affinity value $K_{i_1,\ldots,i_n} = e^{-\Delta}$ where Δ is the volume defined by the n-tuple.

Input: $K_{i_1,\ldots,i_n}$ the probability that $x_{i_1},\ldots,x_{i_n}$ belong to the same cluster

Output: $g_{rs} = P(y_s = r \mid D)$ the probability that the point x_s belongs to the r'th cluster

Assume the conditional independence: $y_1 \perp \ldots \perp y_m \mid x_1,\ldots,x_m$

$$K_{i_1,\ldots,i_n} = \sum_{r=1}^{k} P(y_{i_1} = r \mid D) \cdots P(y_{i_n} = r \mid D) = \sum_{r=1}^{k} g_{r,i_1} \cdots g_{r,i_n}$$

→ $K = \sum_{r=1}^{k} g_r^{\otimes n}$ is a n-dimensional super-symmetric tensor of rank=k

Clustering data into k groups: Beyond Pairwise Affinity

Hyper-stochastic constraint: under balancing requirement

K is (scaled) hyper-stochastic:

$$\sum_{i_1,..,i_{j-1},i_{j+1},...,i_n} K_{i_1,...,i_n} = \left(\frac{m}{k}\right)^{n-1} 1, \quad j = 1,...,n$$

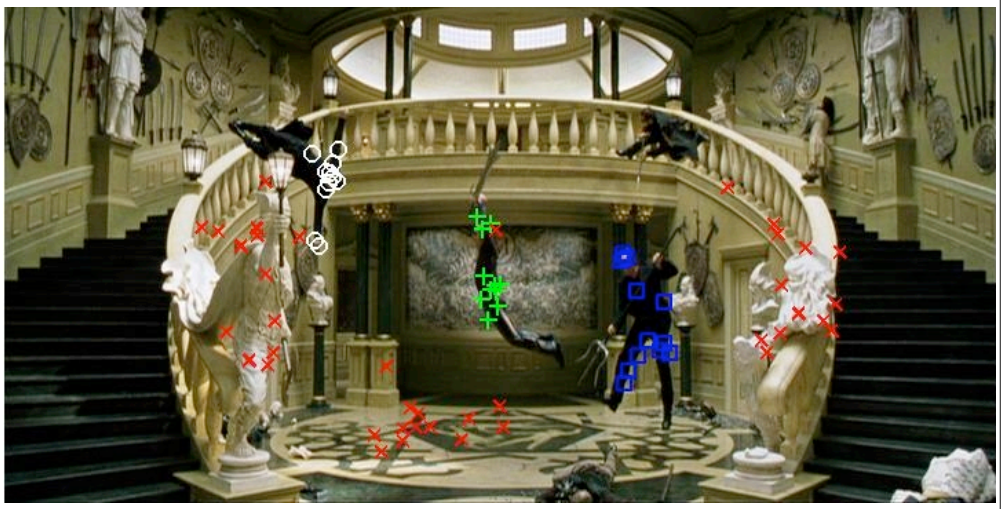
Theorem: for any non-negative super-symmetric tensor $K^{(0)}$, iterating

$$K^{(t+1)}_{i_1,...,i_n} = \frac{K^{(t)}_{i_1,...,i_n}}{(a_{i_1} \cdots a_{i_n})^{1/n}} \qquad a_i = \sum_{i_2,...,i_n} K_{i,i_2,...,i_n}, \quad i = 1,...,m$$

converges to a hyper-stochastic tensor.

Model Selection

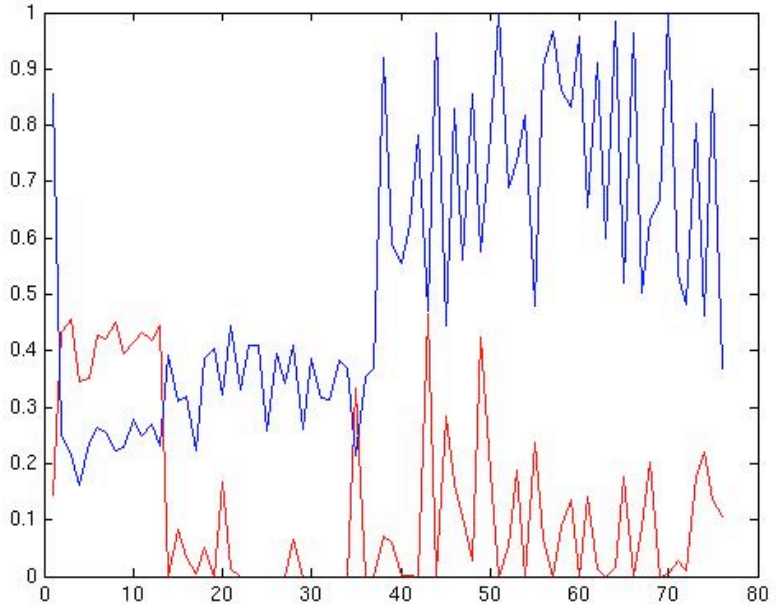
Example: **multi-body segmentation**

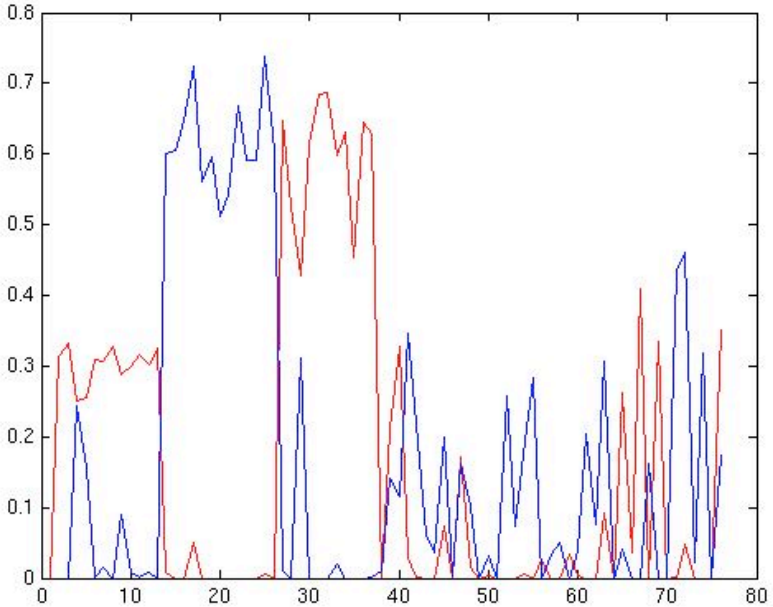


9-way array, each entry contains
The probability that a choice of 9-tuple of points arise from the same model.

$$p'^T Fp = 0$$

Probability: $e^{-(p'^T Fp)^2}$

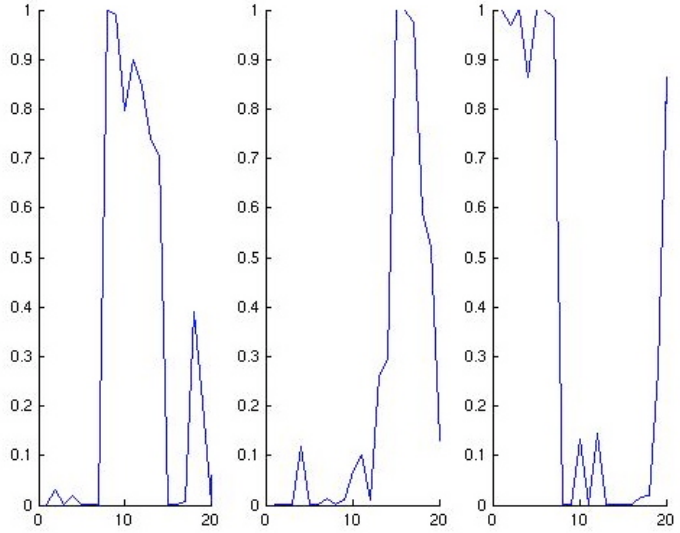




Model Selection

Example: visual recognition under changing illumination

4-way array, each entry contains
The probability that a choice of 4 images
Live in a 3D subspace.



Partially-symmetric Decompositions
Latent Clustering

Latent Clustering Model

$x_1, \ldots, x_m$ collection of data points

$Y = \{y_1, \ldots, y_k\}$ context states

$P(x_r, x_s \mid y_j)$ probability that x_r, x_s are clustered together given that $Y = y_j$

$$P(x_r, x_s) = \sum_{j=1}^{k} P(x_r, x_s \mid y_j) P(y_j)$$

small high

i.e., the vector $u_j = P(x_r, x_s, y_j)$ has a low entropy meaning that the contribution of the context variable to the pairwise affinity is far from uniform.

→ pairwise affinities do not convey sufficient information.

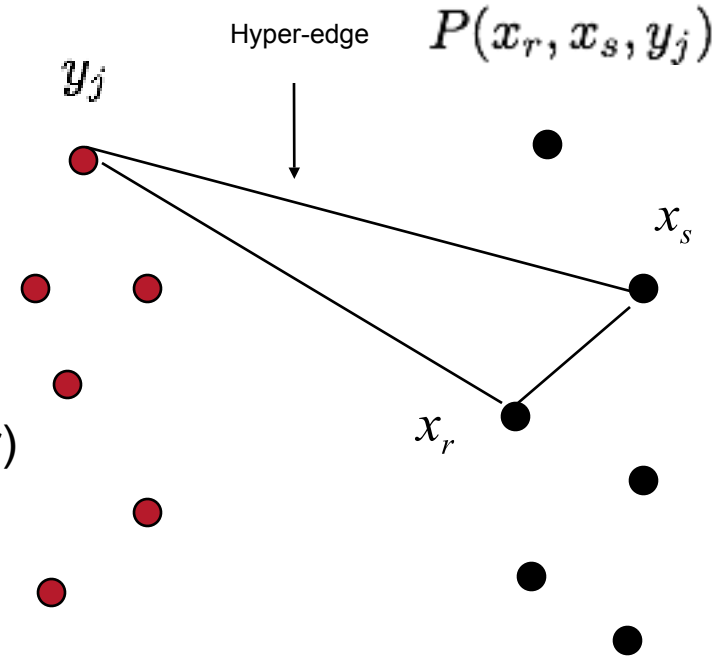
$K_{r,s,j} = P(x_r, x_s, y_j)$ input triple-wise affinities

Latent Clustering Model

Hypergraph representation G(V,E,w)

$$V = \{x_1, ..., x_m, y_1, ..., y_k\}$$

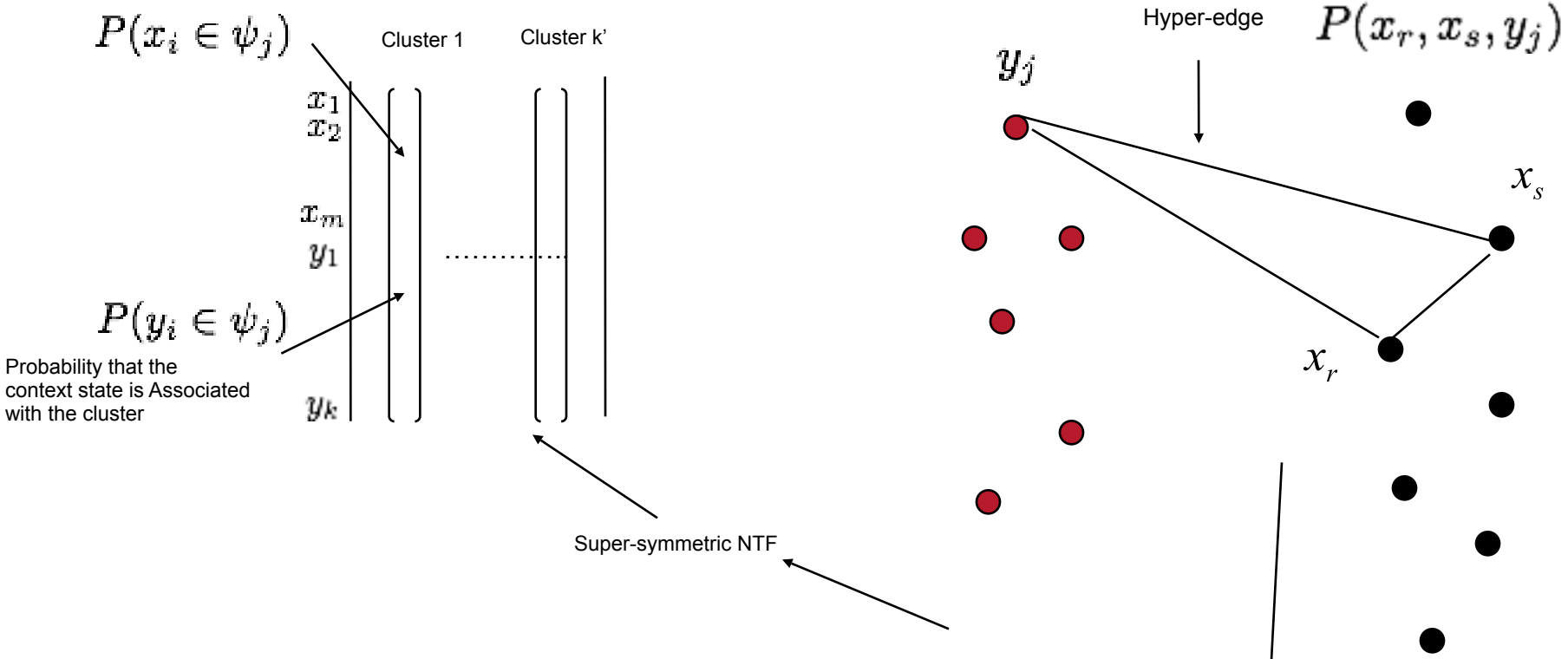
E defined over triplets $\{v_{i_1}, v_{i_2}, v_{i_3}\}$ representing two data points and one context state (any order)



$$w_{i_1,i_2,i_3} = \left\{ \begin{array}{cc} K_{i_1,i_2,i_3} & if\ \{v_{i_1}, v_{i_2}, v_{i_3}\} \in E \\ undertermined & otherwise \end{array} \right\} \qquad 1 \leq i_1, i_2, i_3 \leq m + k$$

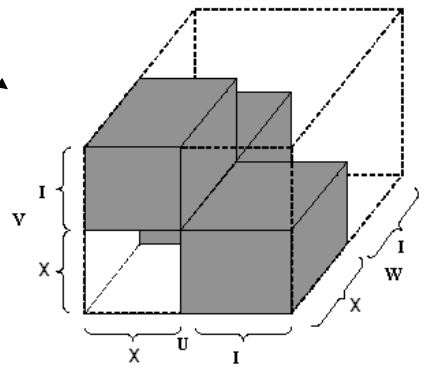
"undetermined" means we have no information, i.e, we do not have access to the probability that three data points are clustered together...

Latent Clustering Model



Hypergraph clustering would provide means to cluster together points whose pairwise affinities $P(x_r, x_s)$ are low, but there exists a subset of context states for which $P(x_r, x_s, y_j)$ is high.

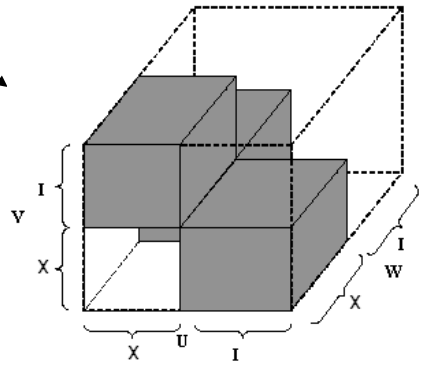
Multiple copies of this block
All the remaining entries are undefined



In that case, a cluster would be formed containing the data points ***and*** the subset of context states.

Latent Clustering Model

Multiple copies of this block
All the remaining entries are undefined

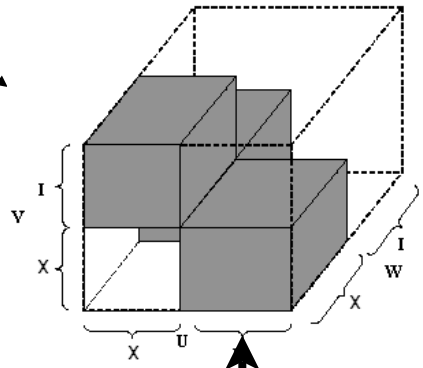


$$W = \sum_{r=1}^{q} g_r \otimes g_r \otimes g_r$$

$$= \sum_{r=1}^{q} \begin{pmatrix} h_r \\ u_r \end{pmatrix} \otimes \begin{pmatrix} h_r \\ u_r \end{pmatrix} \otimes \begin{pmatrix} h_r \\ u_r \end{pmatrix}$$

Latent Clustering Model

Multiple copies of this block
All the remaining entries are undefined



$$W = \sum_{r=1}^{q} g_r \otimes g_r \otimes g_r$$

$$= \sum_{r=1}^{q} \begin{pmatrix} h_r \\ u_r \end{pmatrix} \otimes \begin{pmatrix} h_r \\ u_r \end{pmatrix} \otimes \begin{pmatrix} h_r \\ u_r \end{pmatrix}$$

$$K = \sum_{r} u_r \otimes h_r \otimes h_r$$

$P(y \in \psi_r)$ $P(x \in \psi_r)$

Latent Clustering Model: Application

$x_1, \ldots, x_m$ collection of image fragments from a large collection of images

$I_1, \ldots, I_n$ unlabeled images holding k object classes, one object per image

Available info: # of obj classes k, each image contains only one instance from one (unknown) object class.

Clustering Task: associate (probabilistically) fragments to object classes.

Challenges:

- Clusters and obj classes are not the same thing. Generally, # of clusters is larger than # of obj classes. Also, the same cluster may be shared among a number of different classes.
- The probability $P(x_r, x_s)$ that two fragments should belong to the same cluster may be low only because they appear together in a small subset of images

$$\underbrace{P(x_r, x_s)}_{\text{small}} = \sum_{j=1}^{n} \underbrace{P(x_r, x_s / I_j)}_{\text{high}} P(I_j)$$

Images Examples

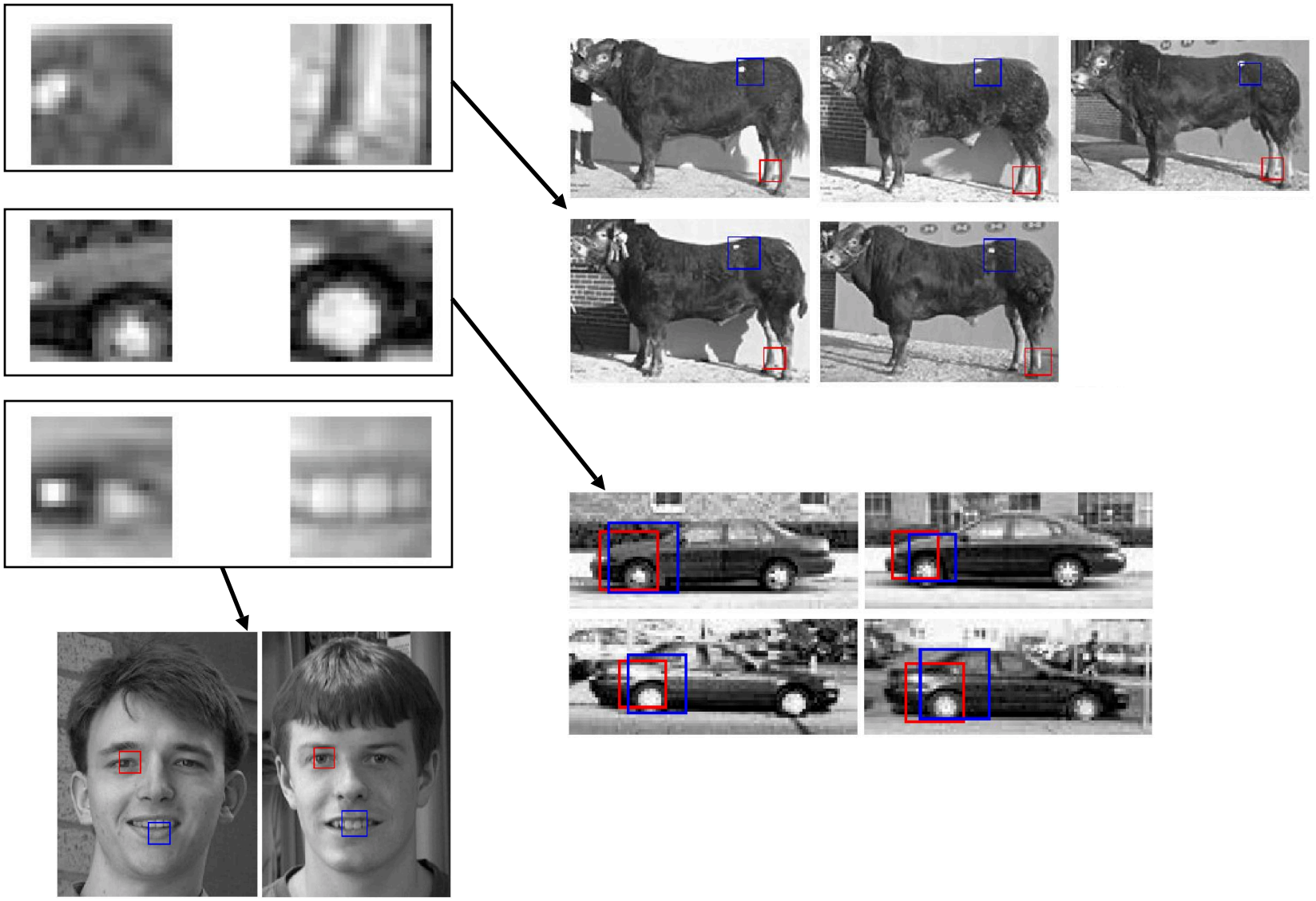
- 3 Classes of images:
- Cows

- Faces

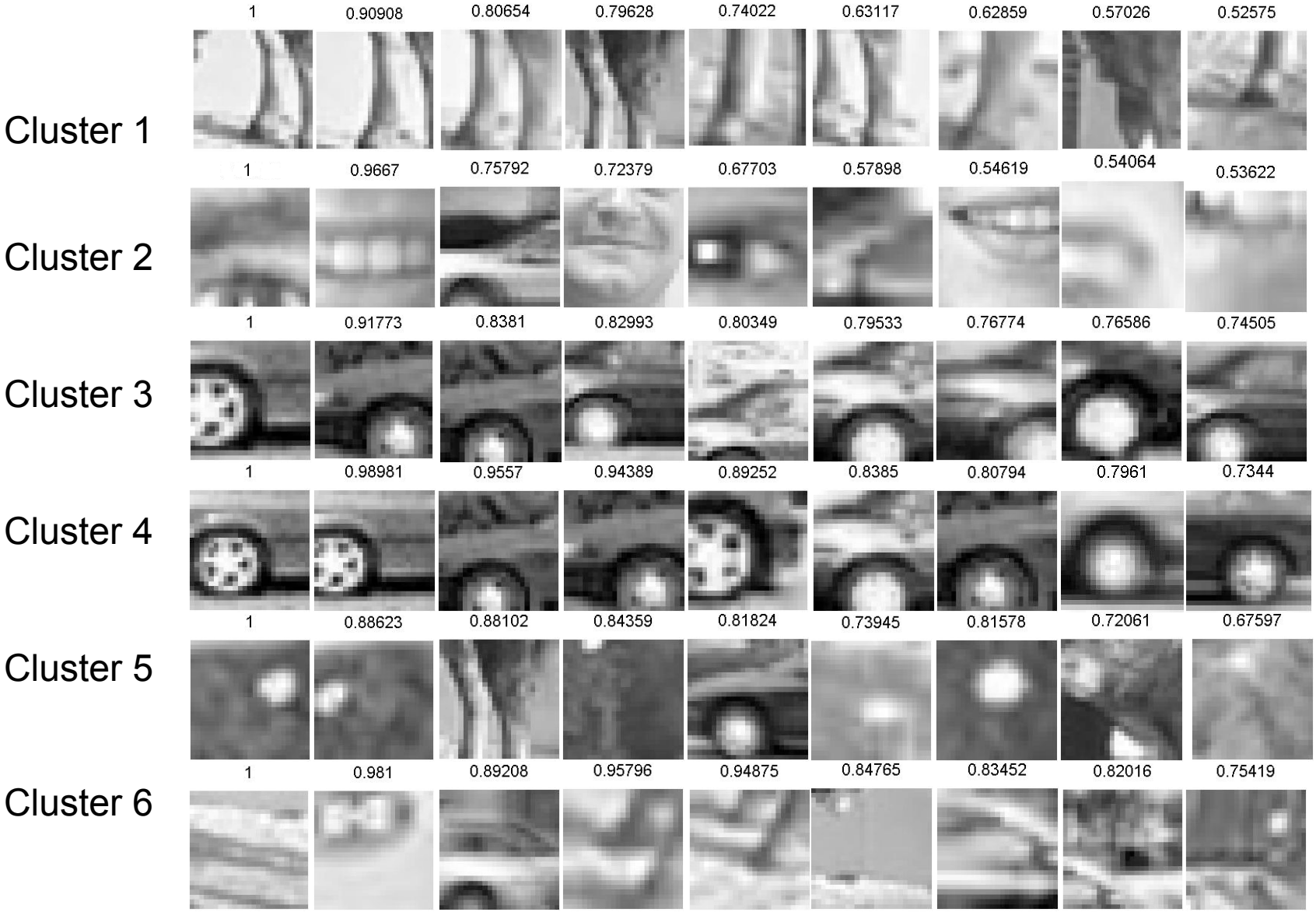
- Cars

Fragments Examples

Examples of Representative Fragments



Leading Fragments Of Clusters



Examples Of The Resulting Matrix G Rows

0.0129	0.0012	0.0102	0.0965	1	0	0	0	0
0.0008	0.0854	0.0056	0.0002	0	0.5049	0.9667	0	0.0233
0	0.0008	0.0001	0.0026	0	0	0	0.9177	0
0	0.0001	0	0	0	0	0	0.9439	0.6982
0.0049	0.0025	0.0061	0.7395	0.1594	0	0	0	0
0.0052	0.0108	0	0	0.1524	0	0	0	0.0007

Examples Of The Resulting Matrix G Rows









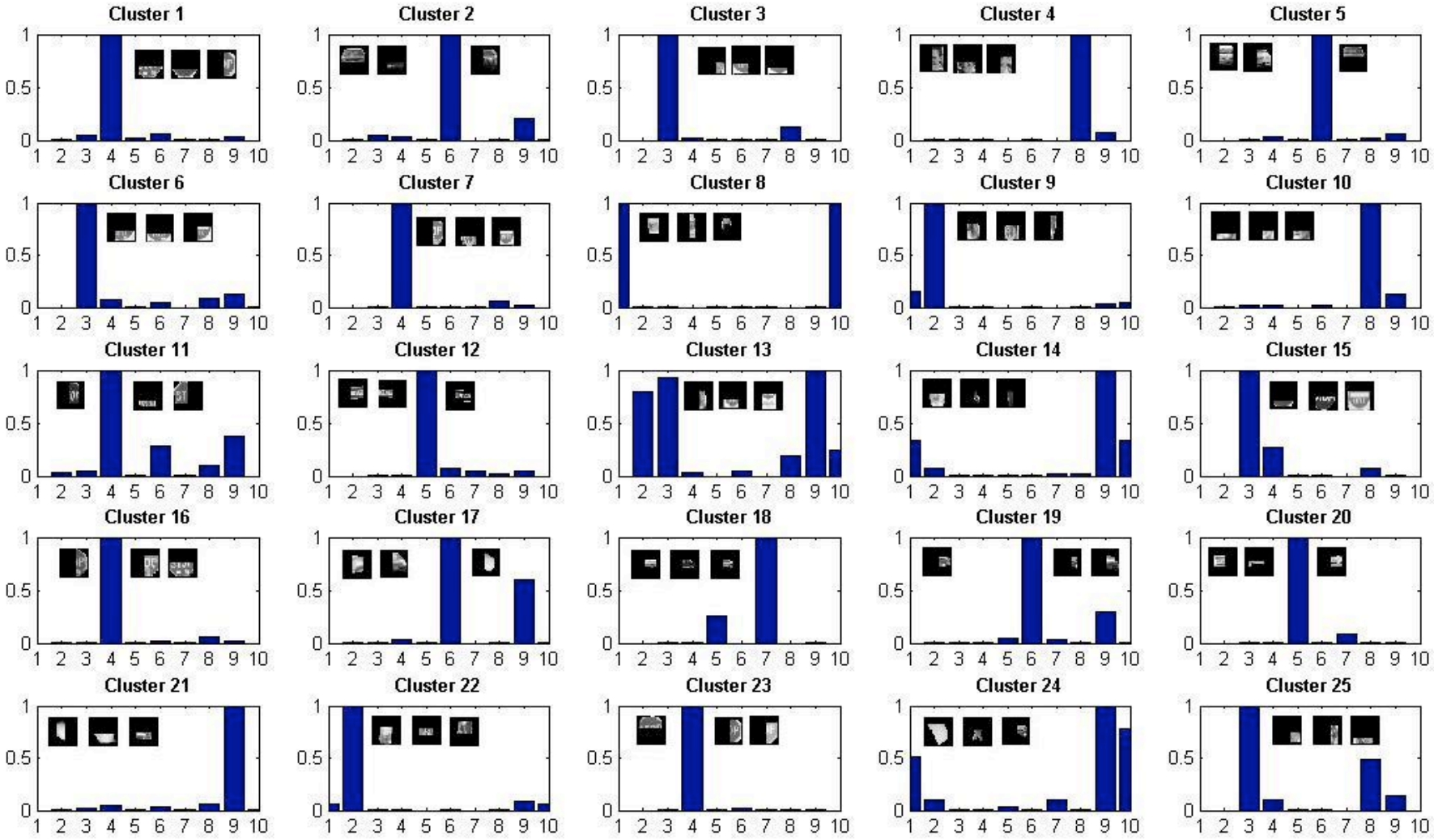
0.8776	0.0377	0.0819	0.0041	0.0054	0.0189	0	0	0.0011
0.0092	0.0192	0.0128	1.0000	0.2775	0.9007	0.0061	0.0046	0.0152
0.0064	0.0085	0.0016	0.0108	0.0102	0.0014	0.0726	0.9770	1.0000
0.0094	0.0014	0.0041	0	0.0141	0.0033	1.0000	0.1056	0.0304
0.4252	0.9024	0.4162	0.0442	0.0201	0.0082	0.0507	0.0254	0.0020
0	0.2835	0.0258	0.0329	0.0136	0.0091	0.0015	0.0023	0.0031

Handling Multiple Object Classes

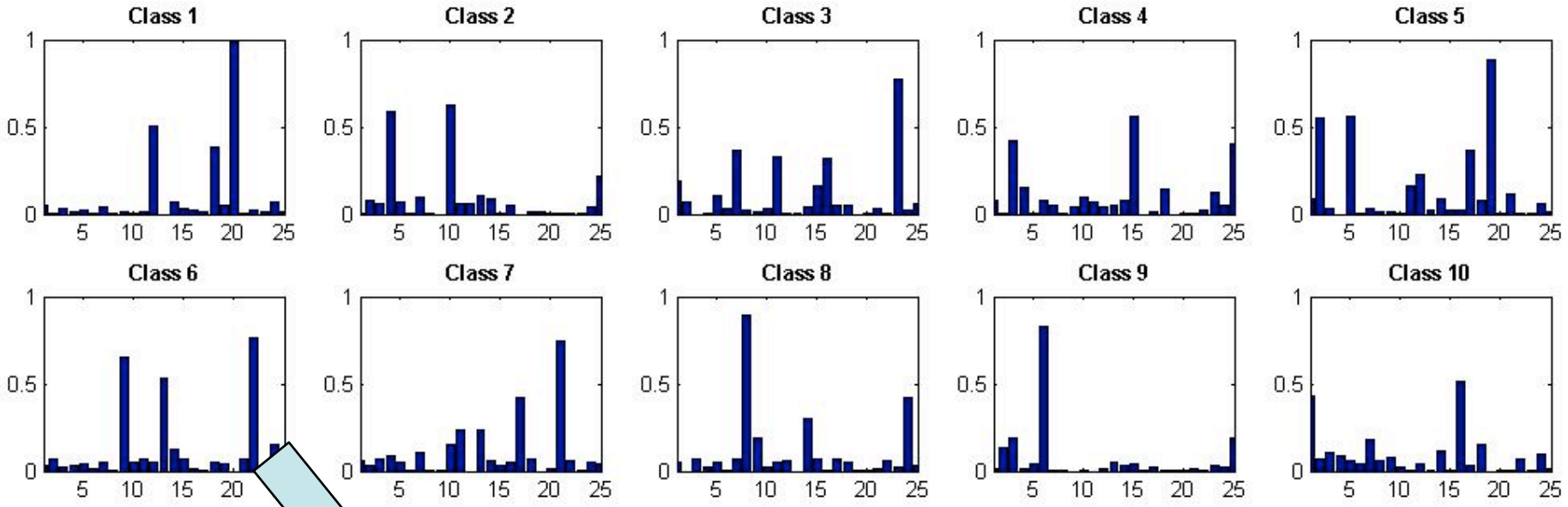
- Unlabeled set of segmented images, each containing an instance of some unknown object class from a collection of 10 classes:
  - (1) Bottle, (2) can, (3) “do not enter” sign,(4) “stop” sign, (5) “one way” sign,(6) frontal view car,(7) side view car,(8) face,(9) computer mouse, (10) pedestrian

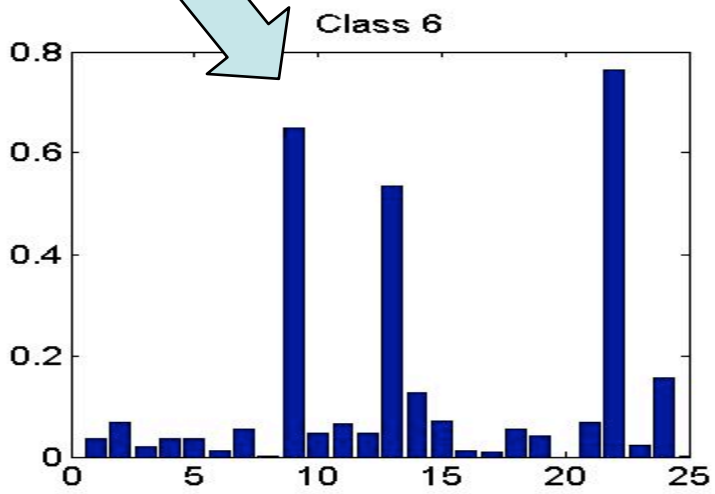
dataset adapted from Torralba,Murphey and Freeman, CVPR04

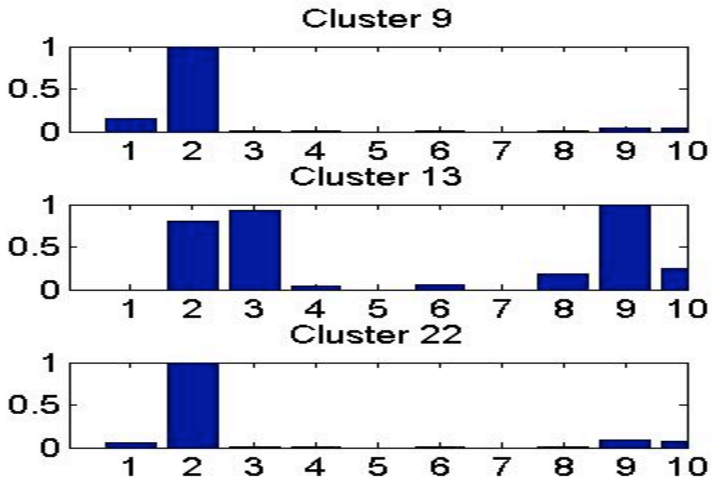
Clustering Fragments



From Clusters to Classes







Results

- A location which is associated with fragments voting consistently for the same object class will have high value in the corresponding voting map.
- Strongest hot-spots (local maximums) in the voting maps show the most probable locations of objects.

Results

Applications of NTF
local features for object class recognition

Object Class Recognition using Filters

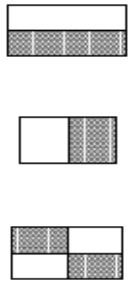
Goal: Find a good and efficient collection of filters that captures the essence of an object class of images.

Two main approaches:

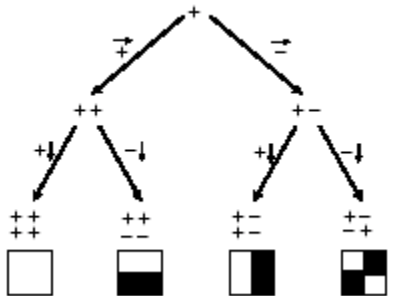
1) Use a large pre-defined bank of filters which:

- Rich in variability.
- Efficiently convolved with an image.

Viola-Jones



Hel-Or



2) Use filters as basis vectors spanning a low-dimensional vector space which best fits the training images.

- PCA, HOSVD – "holistic"

- NMF – "sparse"

The classification algorithm selects a subset of filters which is most discriminatory.

Object Class Recognition using NTF-Filters

The optimization scheme:

1) Construct a bank of filters based on Non-negative Tensor Factorization.

2) Consider the filters as weak learners and use AdaBoost.

Bank of Filters:

- We perform an NTF of k factors to approximate the set of original images.
- We perform an NTF of k factors to approximate the set of inverse images.

There are k^2 filters → Every filter is a pair of original/inverse factor.

We take the difference between the two convolutions (original - inverse).

Original/Inverse pair for face recognition →

Object Class Recognition using NTF-Filters

The optimization scheme:

1) Construct a bank of filters based on Non-negative Tensor Factorization.

2) Consider the filters as weak learners and use AdaBoost.

There are k^2 original/inverse pairs of weak learners:

- We ran AdaBoost to construct a classifier of 50 original/inverse NTF pairs.

For comparison we ran AdaBoost on other sets of weak learners:

- AdaBoost classifier constructed from 50 NMF-weak learners.
- AdaBoost classifier constructed from 50 PCA-weak learners.
- AdaBoost classifier constructed from 200 VJ-weak learners.

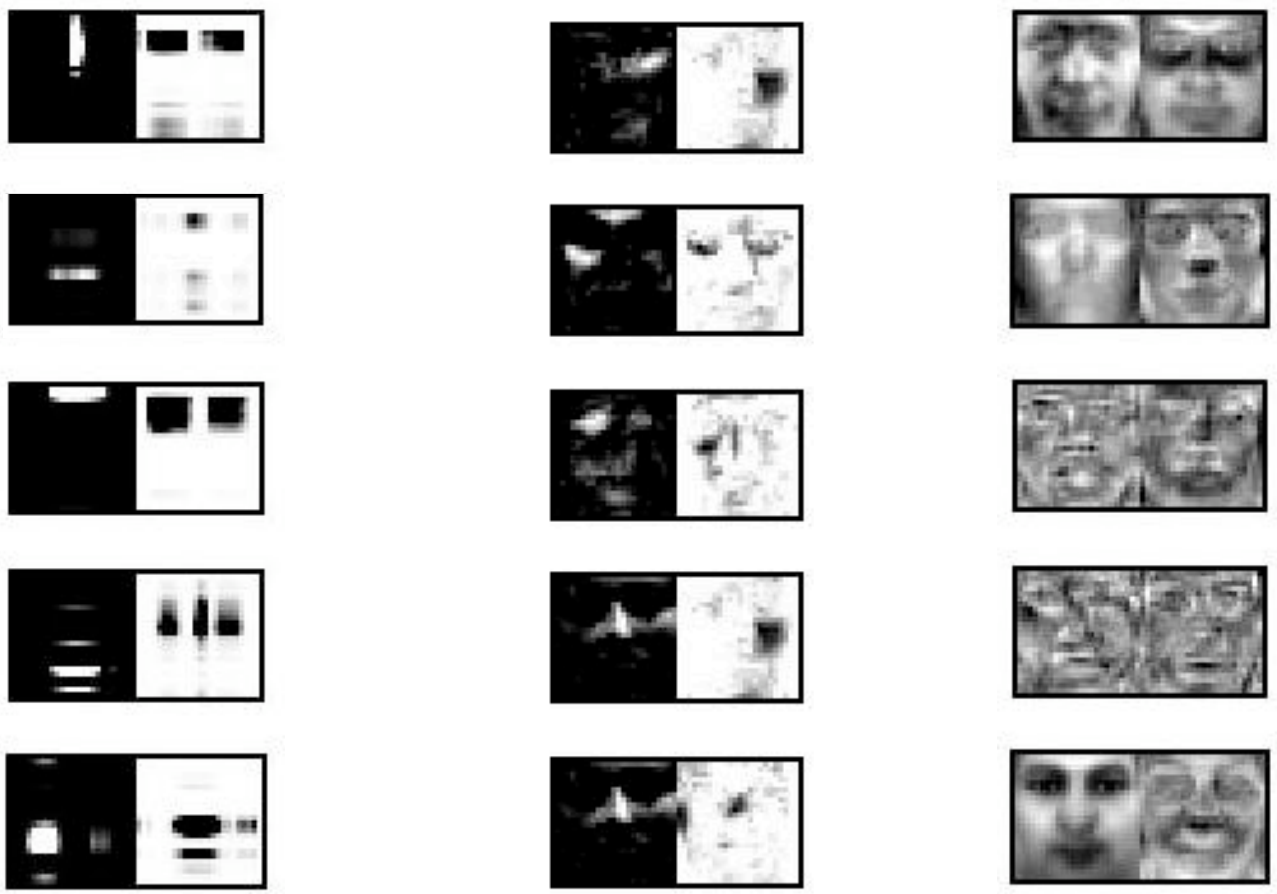
→ An NTF-Filter contains 40 multiplications. An NMF / PCA filter contains about 400 multiplications, yet we have comparable results using the same number of filters.

→ A Viola-Jones weak learner is simpler therefore we used more weak-learners in the AdaBoost classifier.

Face Detection using NTF-Filters

We recovered 100 original factors and 100 inverse factors by preforming NTF on 500 faces of size 24x24.

leading filters:

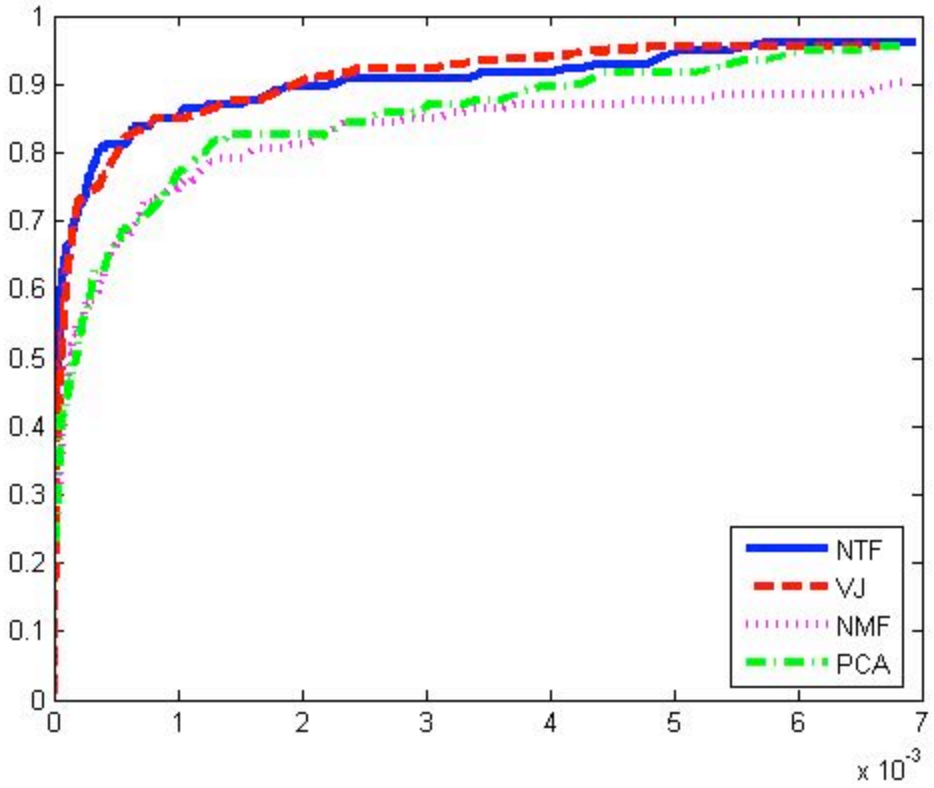


Face Detection using NTF-Filters

ROC curve: **For face recognition all the methods achieve comparable performance.**

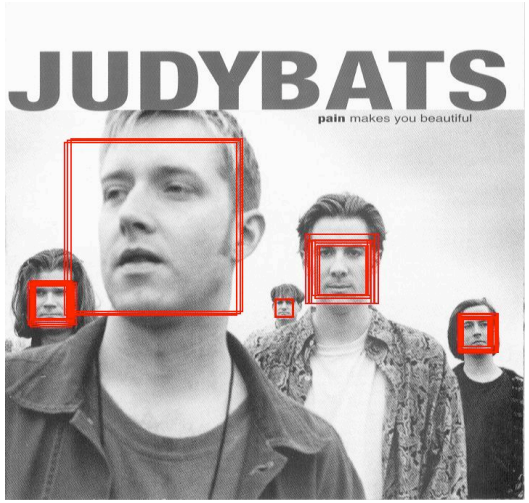
1) **Training:**
The AdaBoost was trained on 2500 faces and 4000 non-faces

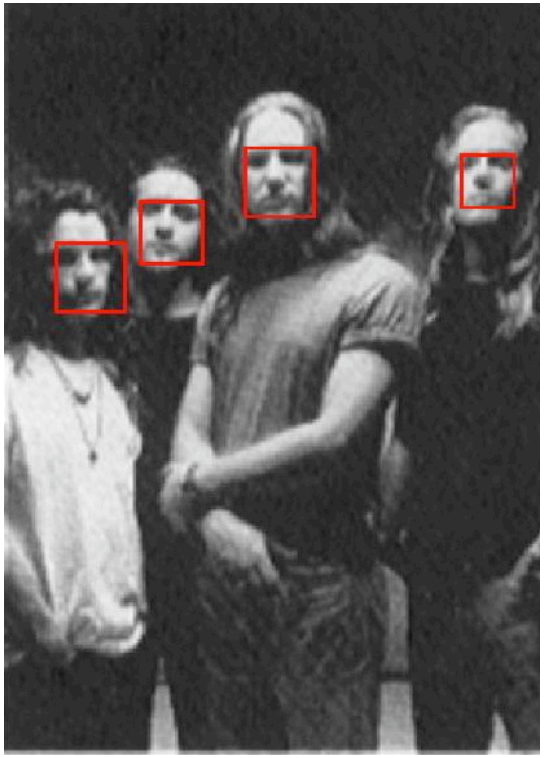
2) **Test:**
The AdaBoost was tested on the MIT Test Set, containing 23 images with 149 faces.

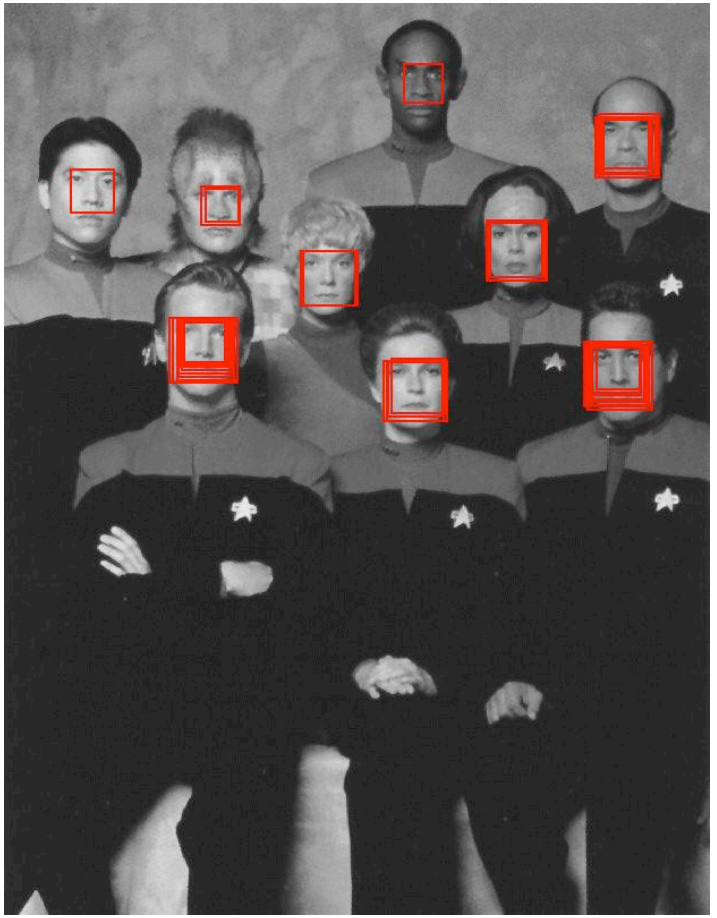


Face Detection using NTF-Filters

Results







Pedestrian Detection using NTF-Filters

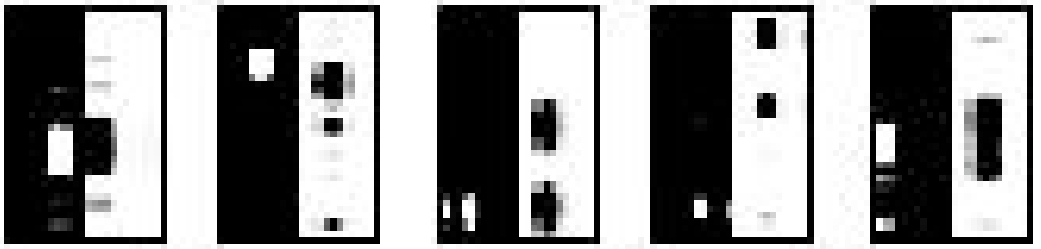
Sample of the database:

Pedestrian Detection using NTF-Filters

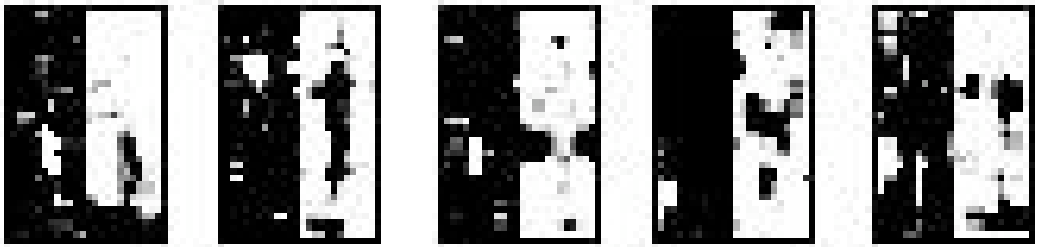
We recovered 100 original factors and 100 inverse factors by preforming NTF on 500 pedestrians of size 10x30.

leading filters:

NTF



NMF

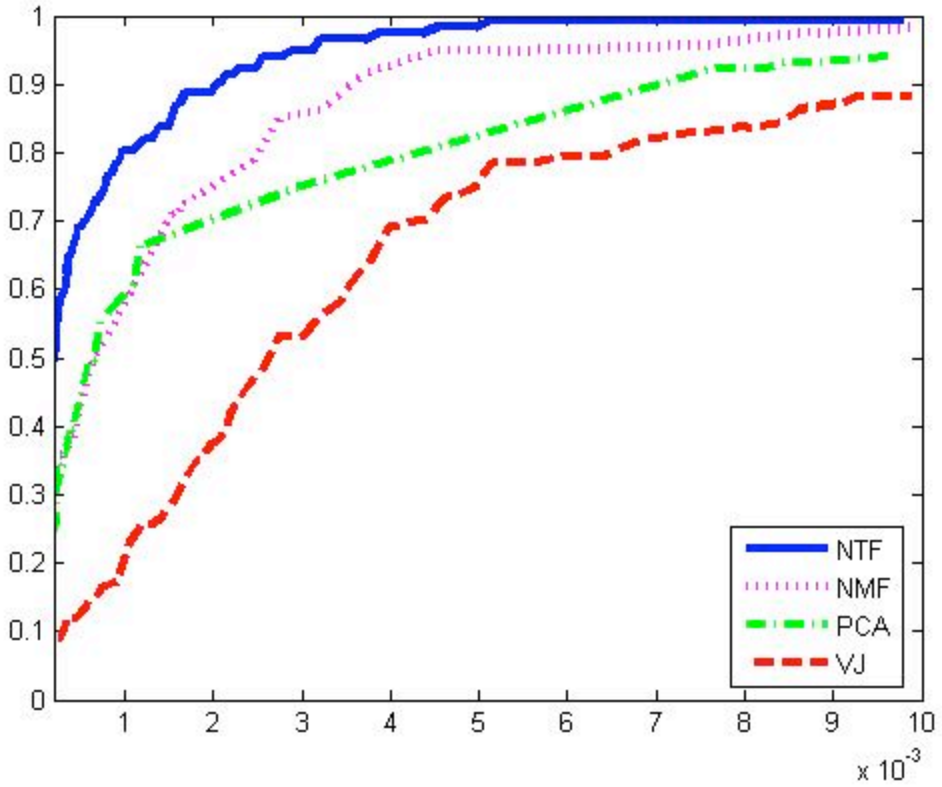


PCA

Pedestrian Detection using NTF-Filters

ROC curve: **For pedestrian detection NTF achieves far better performance than the rest.**

1) **Training:**
   The AdaBoost was trained on 4000 pedestrians and 4000 non-pedestrains.
2) **Test:**
   The AdaBoost was tested on 20 images with 103 pedestrians.



Summary

Factorization of empirical joint distribution $\longleftrightarrow$ Latent Class Model

Factorization of symmetric forms $\longleftrightarrow$ Probabilistic clustering.

Factorization of partially-symmetric forms $\longleftrightarrow$ Latent clustering

Further details in http://www.cs.huji.ac.il/~shashua

END