Linear Discriminant Analysis

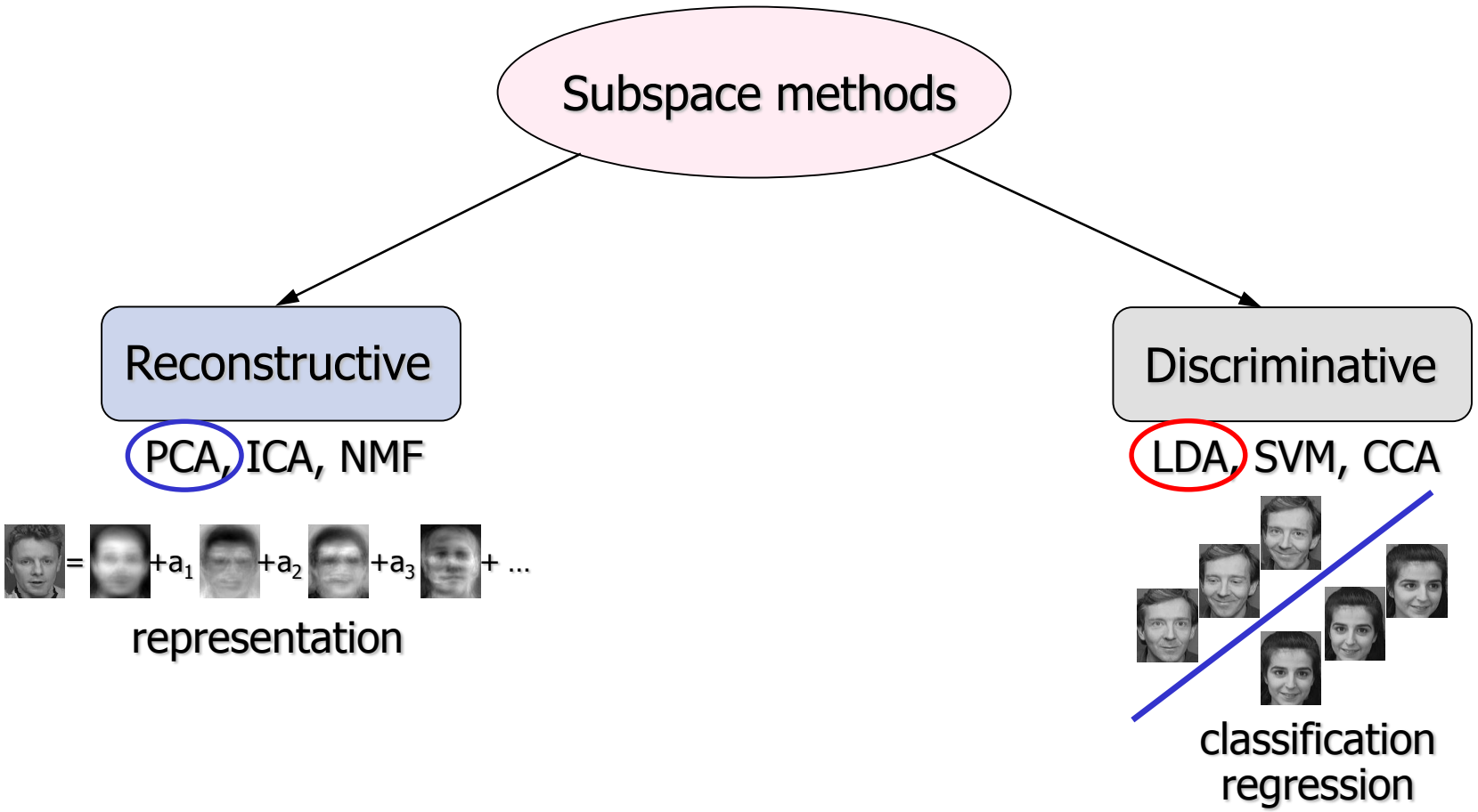
Danijel Skočaj



University of Ljubljana
Faculty of Computer and Information Science

Udine; 5, 12 December 2014

Subspace methods



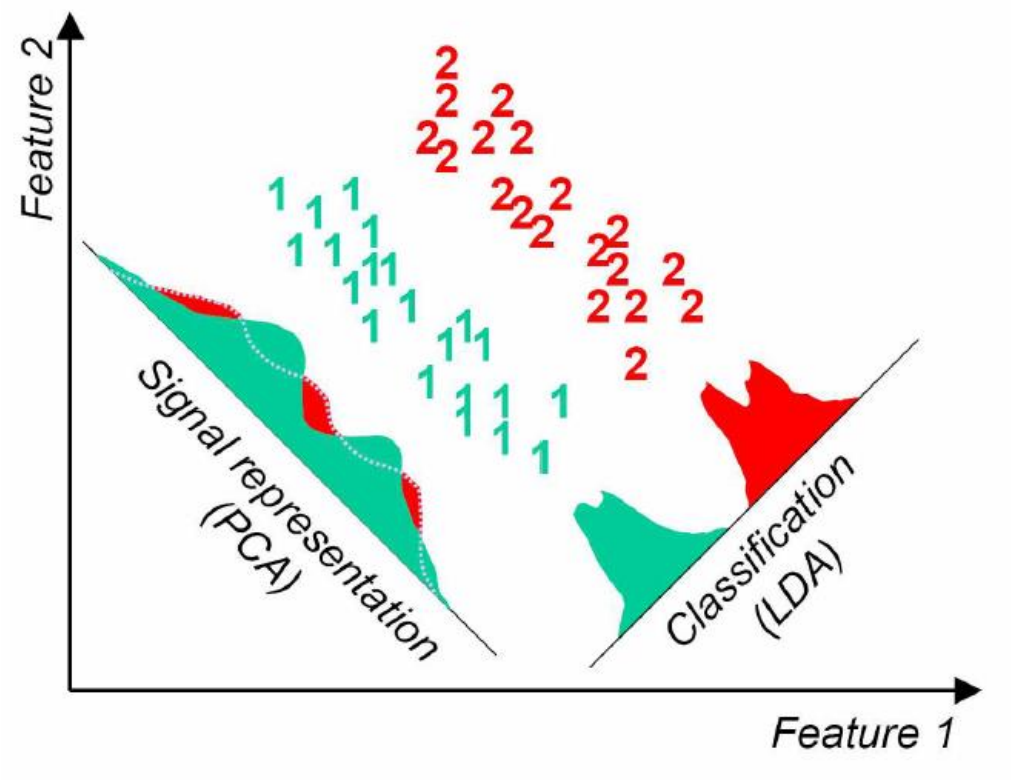
Reconstructive methods

- Goal: reconstruction
  - Good approximation of data → good reconstruction
  - Encompass variability of training data
  - General, task-independent
  - Not very compact, redundant
  - Unsupervised
  - Enables incremental updating
- PCA
  - Finds linear representations that best describes input data; model every sample as well as possible
  - Minimize reconstruction error

Discriminative methods

- Goal: discriminantion between classes
  - Separates the data
    → good classification
  - Spatially/computationally efficient
  - Task-dependent
  - Supervised, focus on specific prior knowledge
  - Effective representations
  - No reconstruction (low dimensional projections)
- LDA
  - Maximizes between class scatter
  - Minimizes within class scatter

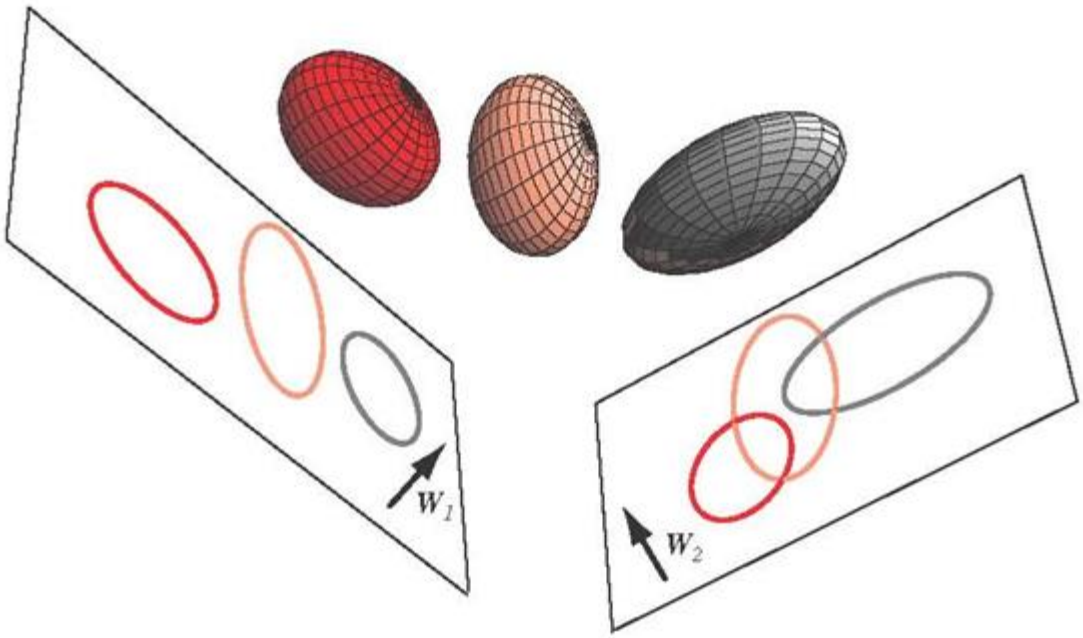
PCA vs. LDA: main idea



www.lsv.uni-saarland.de/dsp_ss05_chap11.pdf

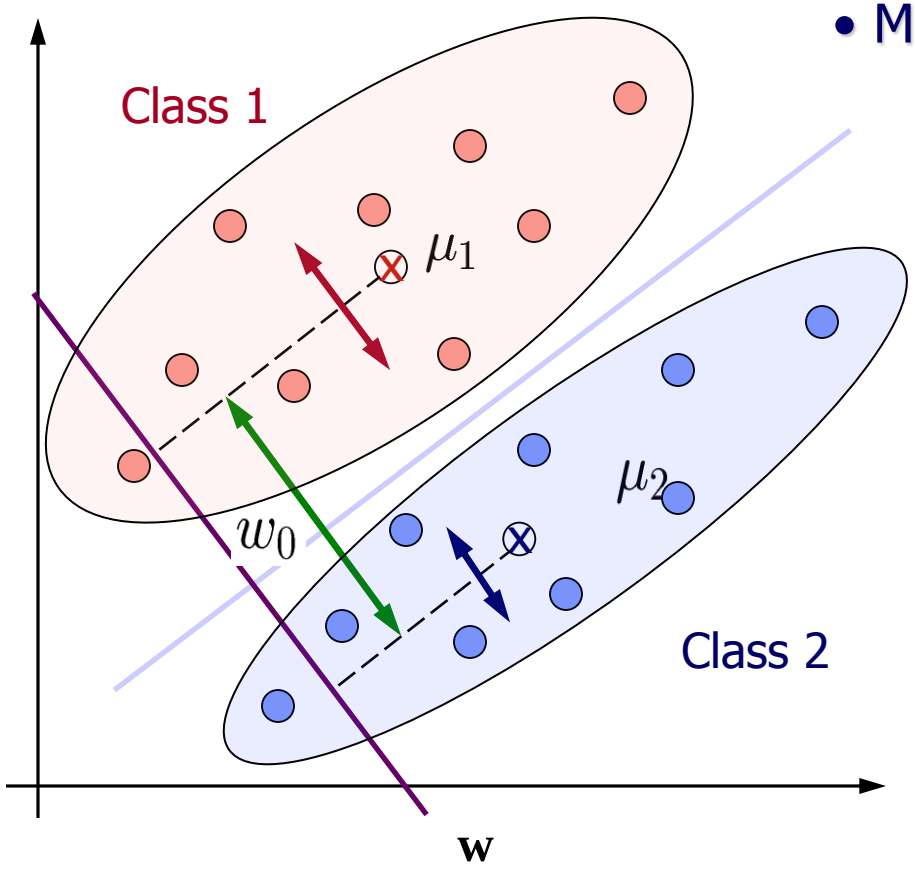
LDA: main idea

- Find a subspace that best discriminates between the classes



From R. Duda, P. Hart, D. Stork, Pattern Classification.
John Wiley & sons, 2001

Linear Discriminant Analysis (LDA)



- Maximize distance between classes.
- Minimize distance within a class.

Criterion function:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$$

S_b … between scatter matrix

S_w … within scatter matrix

w is a solution of a generalized eigenvalue problem:

$$S_b \mathbf{w} = \lambda S_w \mathbf{w}$$

Classification function:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} 0$$

Definitions

- Between-class scatter matrix

$$S_b = \sum_{c=1}^{C} p_c (\vec{\mu}_c - \vec{\mu})(\vec{\mu}_c - \vec{\mu})^t$$

- where:
  - C : number of classes
  - $\vec{\mu}$: mean of all training samples
  - $p_c = \frac{\mathrm{N}_c}{\sum_{l=1}^{C} N_l}$ - $\vec{\mu}_c = \frac{1}{\mathrm{N}_c} \sum_{i=1}^{N_c} \vec{\mathrm{x}}_{\mathrm{i},c}$
- Scatter of the class means around the total mean

Definitions

- Within class scatter-matrix

$$S_w = \sum_{c=1}^{C} p_c \Sigma_c$$

- where

$$\Sigma_c = \frac{1}{N_c} \sum_{i=1}^{N_c} (\vec{x}_{i,c} - \vec{\mu}_c)(\vec{x}_{i,c} - \vec{\mu}_c)^t$$

- Sum of class covariance matrices

Criterion function

- Maximize S_b and minimize S_w
  - Fischer criterium

  ⇨ maximize: $J(\mathbf{w}) = \dfrac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$

  - Rayleight quotient

- Solution: generalized eigenvalue problem:

$$S_b \mathbf{w} = \lambda S_w \mathbf{w}$$

Algorithm

```
function [W,Ms]=lda(X,c,n)
% X: input samples in columns, arranged by classes
% c, n: number of classes, number of samples per class
% W: LDA subspace basis vectors
% Ms: class means in the LDA subspace
SB=0; SW=0;
MM=mean(X')'; %oveall mean
for i=1:c
   Ms(:,i)=mean(X(:,(i-1)*n+1:i*n)')';  %class means
   SB=SB+n*(Ms(:,i)-MM)*(Ms(:,i)-MM)';  % between class scatter m.
   for j=1:n % within class scatter matrix
      SW=SW+(X(:,(i-1)*n+j)-Ms(:,i))*(X(:,(i-1)*n+j)-Ms(:,i))';
   end;
end;
% the solution of a generalized eigenproblem:
[W, EIGD] = eigs(inv(SW)*SB, c-1,'LM',opts);
Ms=W'*Ms;  %map means into the LDA space
```

Using LDA

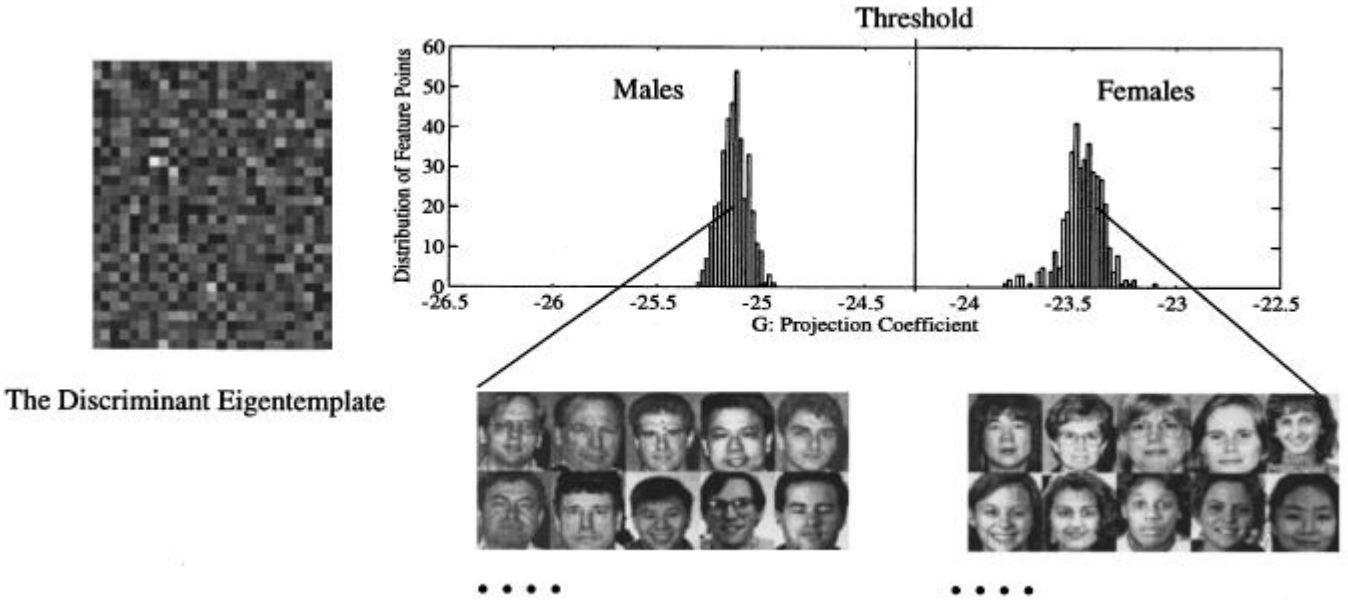
- Learning
  - Perform LDA on X, obtain W and Ms
- Recognition
  - Project sample x in LDA space W
  - Look for the nearest class mean Ms
  - Classification function: $g(\mathbf{x}) = \mathbf{W}^T(\mathbf{x} - \mu)$
- Singularity problems often occur
  - Solution: first preprocess the data using PCA (and keep all eigenvectors)

Using LDA

- Learning
  - Perform PCA on X, obtain PCA space U and coefficient vectors A (keep all coefficeints)
  - Perform LDA on A, obtain LDA space V and Ms
- Recognition
  - Project sample x in PCA space U, obtain projection a
  - Project a in LDA space V
  - Look for the nearest class mean Ms

Example

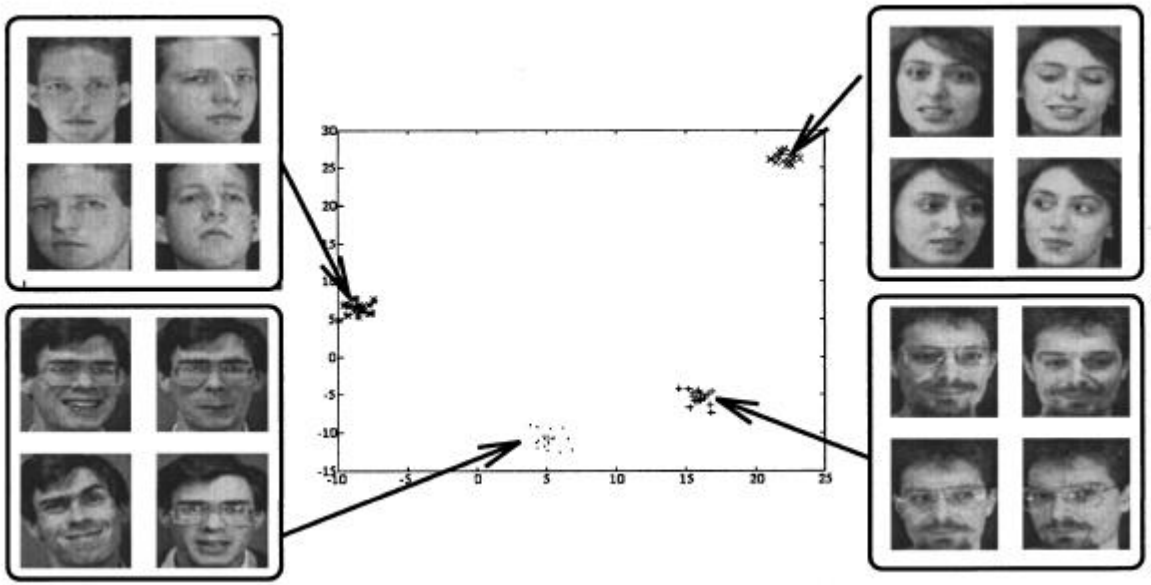
- Males vs. females
- Recognition rate:
  - Same individuals: 95%
  - New individuals: 92%



From K. Etemad, R. Chellapa, Discriminant analysis for recognition of human faces. J. Opt. Soc. Am. A,Vol. 14, No. 8, August 1997

Example

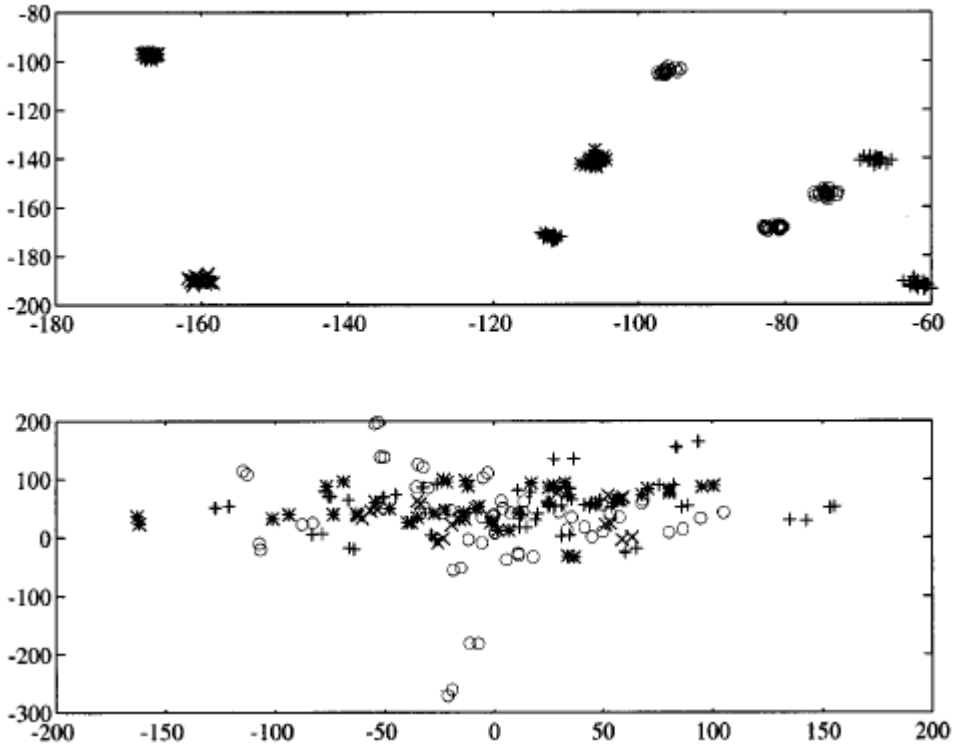
- 1500 images
- 60 individuals
- Recognition rate: 99.2%



From K. Etemad, R. Chellapa, Discriminant analysis for recognition of human faces. J. Opt. Soc. Am. A,Vol. 14, No. 8, August 1997

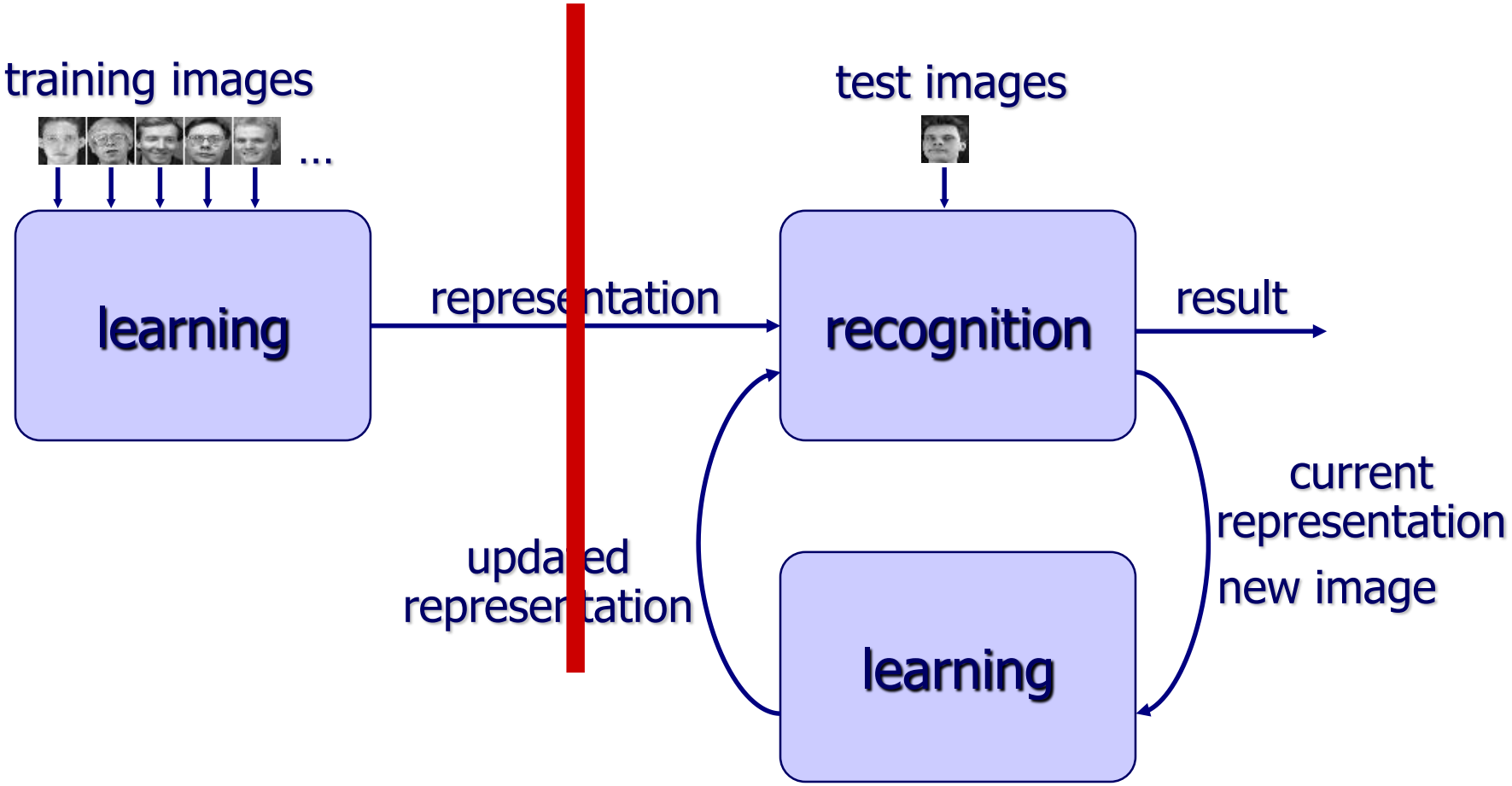
Example

- LDA vs. PCA
  - Best two-dimensional subspace



From K. Etemad, R. Chellapa, Discriminant analysis for recognition of human faces. J. Opt. Soc. Am. A,Vol. 14, No. 8, August 1997

Incremental learning

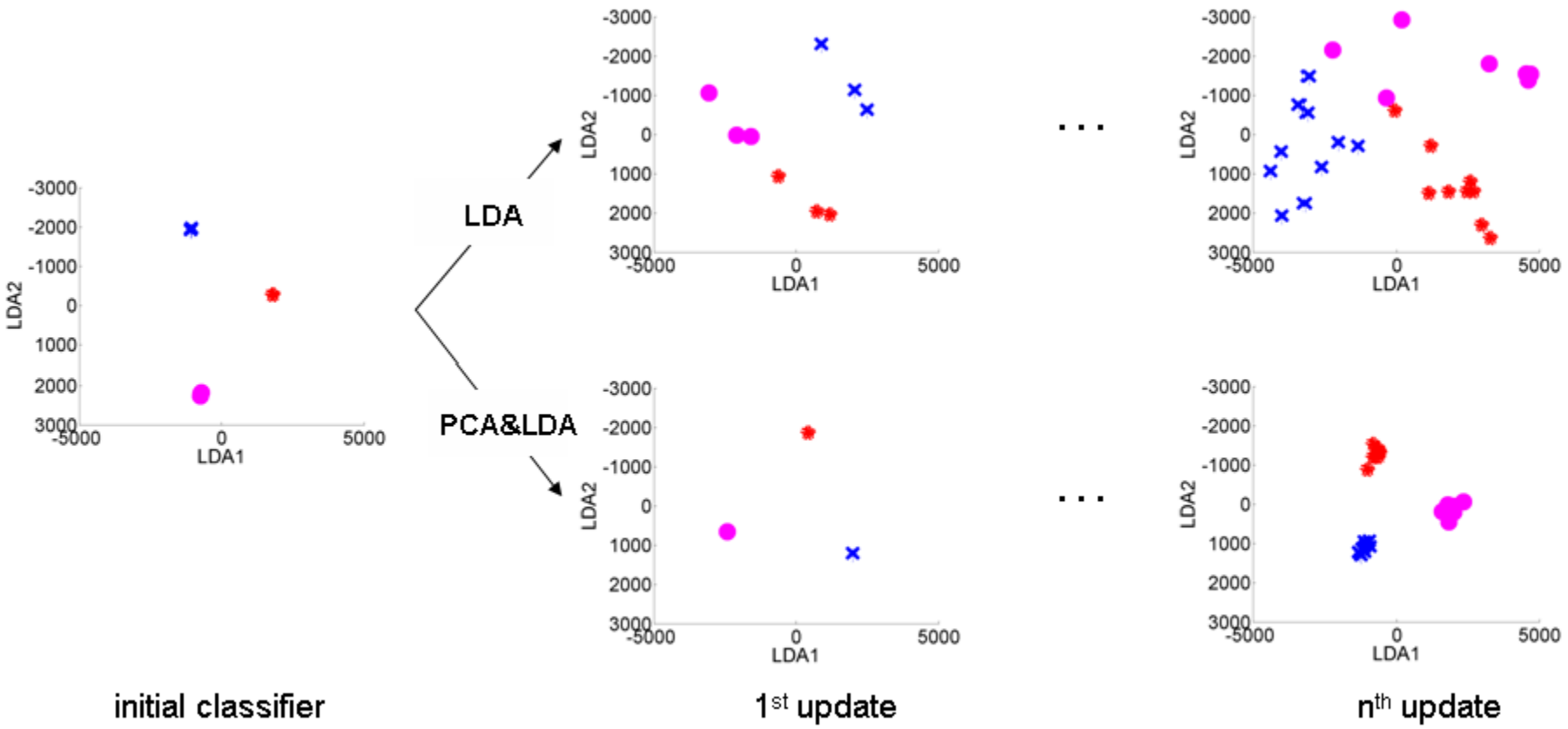


Incrementalization

- Reconstructive methods (PCA)
  - Good approximation of data
  - General, task-independent
  - Not very compact, redundant representations
  - ⇨ Enable incremental updating
- Discriminative methods (LDA)
  - Supervised, task-dependent
  - Compact representations
  - No reconstruction (low dimensional projections)
  - ⇨ Do not enable incremental updating

⇨ Combine reconstructive model and discriminative classifier

Incremental PCA and LDA



Augmenting the PCA basis

- Embed LDA learning and classification into PCA framework

$$\mathbf{X} \in \mathbf{R}^{m \times n} : PCA(\mathbf{X}) \rightarrow \mathbf{U} = \left[\begin{array}{cc} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{(n-k)}} \end{array} \right] \in \mathbf{R}^{m \times n}$$

$$\mathbf{A} = \left[\begin{array}{c} \mathbf{A}_{\underline{k}} \\ \mathbf{A}_{\underline{(n-k)}} \end{array} \right] \in \mathbf{R}^{n \times n}$$

$$LDA(\mathbf{A}) \rightarrow \mathbf{V} = \left[\begin{array}{c} \mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{(n-k)}} \end{array} \right] \in \mathbf{R}^{n \times (c-1)}$$

- First k << n principal vectors
  - Contain most visual variability
  - No guarantee that it contains most discriminative information

Augmenting the PCA basis

- Augment PCA subspace with c-1 additional basis vectors → keeps discriminative information

$$\widetilde{\mathbf{V}}_{\underline{(n-k)}} = \mathbf{V}_{\underline{(n-k)}}(\mathbf{V}_{\underline{(n-k)}}^{\top}\mathbf{V}_{\underline{(n-k)}})^{-1/2} \in \mathbf{R}^{(n-k)\times(c-1)}$$

$$\widehat{\mathbf{U}} = \left[\begin{array}{cc} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{(n-k)}}\widetilde{\mathbf{V}}_{\underline{(n-k)}} \end{array}\right] \in \mathbf{R}^{m\times(k+c-1)}$$

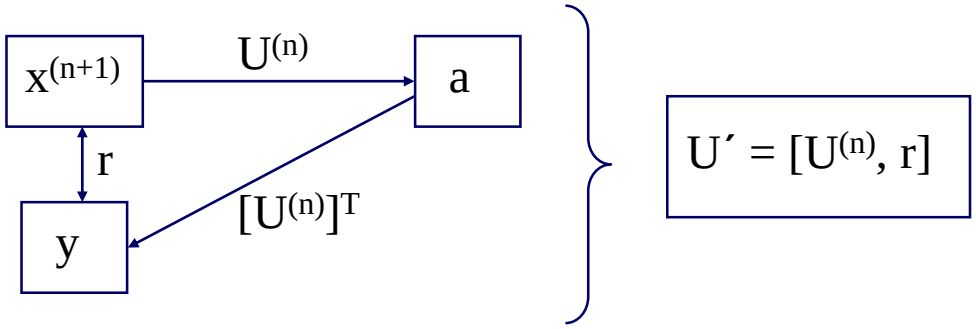
$$\widehat{\mathbf{A}} = \left[\begin{array}{c} \mathbf{A}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{(n-k)}}^{\top}\mathbf{A}_{\underline{(n-k)}} \end{array}\right] \in \mathbf{R}^{(k+c-1)\times n}$$

$$\widehat{\mathbf{V}} = \left[\begin{array}{c} \mathbf{V}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{(n-k)}}^{\top}\mathbf{V}_{\underline{(n-k)}} \end{array}\right] \in \mathbf{R}^{(k+c-1)\times(c-1)}$$

- → APCA subspace

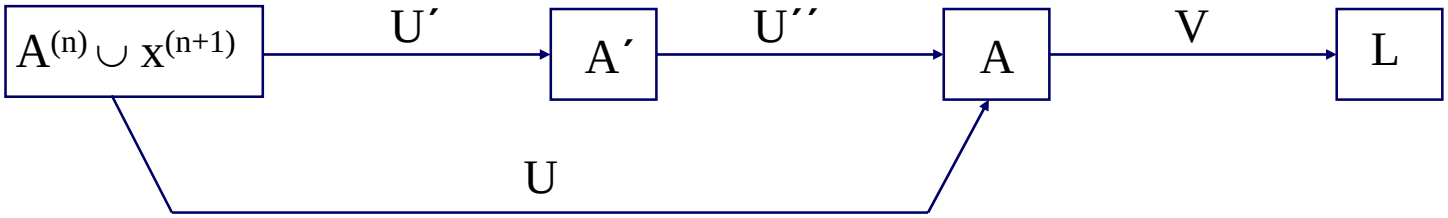
Incremental updating

1. Project new image $\mathbf{x}^{(n+1)}$ in the current eigenspace $\mathbf{U}^{(n)}$
2. Difference between the reconstruction $\mathbf{a}$ and the input image $\mathbf{x}^{(n+1)}$ is orthogonal to the current eigenspace
3. Enlarge eigenspace with normalized residual → $\mathbf{U}'$



Incremental updating

4. Perform PCA on the actual representation $\mathbf{A}'$
   → new principal axes $\mathbf{U}''$
5. Project $\mathbf{A}'$ into new eigenspace $\mathbf{U}''$ → $\mathbf{A}$
6. Rotate current basis vectors $\mathbf{U}'$ to match new basis $\mathbf{U}$
7. Update the mean
8. Perform LDA on $\mathbf{A}$ in new eigenspace → $\mathbf{V}$



Classification function: $g(\mathbf{x}) = (\mathbf{UV})^{\top}\mathbf{x}$

Incremental updating

- Model grows when adding a new image
  → truncate by one analogue to augmenting the basis

$$\mathbf{U} = \left[\mathbf{U}_{\underline{k}} \;\; \mathbf{U}_{\underline{c}} \right] \in \mathbf{R}^{m \times (k+c)}, \quad \mathbf{A} = \begin{bmatrix} \mathbf{A}_{\underline{k}} \\ \mathbf{A}_{\underline{c}} \end{bmatrix} \in \mathbf{R}^{(k+c) \times (n+1)}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{c}} \end{bmatrix} \in \mathbf{R}^{(k+c) \times (c-1)}, \quad \widetilde{\mathbf{V}}_{\underline{c}} = \mathbf{V}_{\underline{c}} (\mathbf{V}_{\underline{c}}^{\top} \mathbf{V}_{\underline{c}})^{-1/2} \in \mathbf{R}^{c \times (c-1)}$$

$$\widehat{\mathbf{U}}^{(n+1)} = \left[\mathbf{U}_{\underline{k}} \;\; \mathbf{U}_{\underline{c}} \widetilde{\mathbf{V}}_{\underline{c}} \right] \in \mathbf{R}^{m \times (k+c-1)}$$

$$\widehat{\mathbf{A}}^{(n+1)} = \begin{bmatrix} \mathbf{A}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{c}}^{\top} \mathbf{A}_{\underline{c}} \end{bmatrix} \in \mathbf{R}^{(k+c-1) \times (n+1)}$$

$$\widehat{\mathbf{V}}^{(n+1)} = \begin{bmatrix} \mathbf{V}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{c}}^{\top} \mathbf{V}_{\underline{c}} \end{bmatrix} \in \mathbf{R}^{(k+c-1) \times (c-1)}$$

- APCA subspace is small
  → update is computationally efficient

Update step

- Input: current APCA subspace, new image

1. Update the current APCA subspace with the new image.
2. Do not discard the new dimension (preserving all previously preserved reconstruction and discriminative information as well as information about new image).
3. Perform LDA on low-dimensional APCA coefficients.
4. Preserve *k* principal vectors and append *c-1* vectors, which preserve discriminative information.

- Output: updated APCA subspace, new LDA vector, updated LDA coefficients

Algorithm

Input: current augmented principal subspace: mean vector $\boldsymbol{\mu}^{(n)}$, APCA vectors $\widehat{\mathbf{U}}^{(n)}$, APCA coefficients $\widehat{\mathbf{A}}^{(n)}$, new input image $\mathbf{x}$.

Output: new augmented principal subspace: mean vector $\boldsymbol{\mu}^{(n+1)}$, APCA vectors $\widehat{\mathbf{U}}^{(n+1)}$, coefficients $\widehat{\mathbf{A}}^{(n+1)}$, new low-dimensional LDA vectors $\widehat{\mathbf{V}}^{(n+1)}$ and class centers $\boldsymbol{\nu}^{(n+1)}$.

1: Project a new image $\mathbf{x}$ into the current eigenspace: $\mathbf{a} = \widehat{\mathbf{U}}^{(n)\top}(\mathbf{x} - \boldsymbol{\mu}^{(n)})$.

2: Reconstruct the new image: $\mathbf{y} = \widehat{\mathbf{U}}^{(n)}\mathbf{a} + \boldsymbol{\mu}^{(n)}$.

3: Compute the residual vector: $\mathbf{r} = \mathbf{x} - \mathbf{y}$.

4: Append $\mathbf{r}$ as a new basis vector: $\mathbf{U}' = \begin{bmatrix} \widehat{\mathbf{U}}^{(n)} & \frac{\mathbf{r}}{\|\mathbf{r}\|} \end{bmatrix}$.

5: Determine the coefficients in the new basis: $\mathbf{A}' = \begin{bmatrix} \widehat{\mathbf{A}}^{(n)} & \mathbf{a} \\ \mathbf{0} & \|\mathbf{r}\| \end{bmatrix}$.

6: Perform PCA on $\mathbf{A}'$. Obtain the mean value $\boldsymbol{\mu}''$ and the eigenvectors $\mathbf{U}''$.

7: Project the coefficient vectors to the new basis: $\mathbf{A} = \mathbf{U}''^{\top}(\mathbf{A}' - \boldsymbol{\mu}''\mathbf{1}_{1\times n+1})$.

8: Rotate the subspace $\mathbf{U}'$ for $\mathbf{U}''$: $\mathbf{U} = \mathbf{U}'\mathbf{U}''$.

9: Perform LDA on $\mathbf{A}$. Obtain low-dimensional LDA vectors $\mathbf{V}$ and class centers $\boldsymbol{\nu}$.

10: Divide $\mathbf{U}$, $\mathbf{A}$, and $\mathbf{V}$ on submatrices: $\mathbf{U} = \begin{bmatrix} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{c}} \end{bmatrix}$, $\mathbf{A} = \begin{bmatrix} \mathbf{A}_{\underline{k}} \\ \mathbf{A}_{\underline{c}} \end{bmatrix}$, $\mathbf{V} = \begin{bmatrix} \mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{c}} \end{bmatrix}$.

11: Orthonormalize $\mathbf{V}_{\underline{c}}$: $\widetilde{\mathbf{V}}_{\underline{c}} = \mathbf{V}_{\underline{c}}(\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{-1/2}$.

12: Update the mean: $\boldsymbol{\mu}^{(n+1)} = \boldsymbol{\mu}^{(n)} + \mathbf{U}'\boldsymbol{\mu}''$.

13: Update the APCA basis: $\widehat{\mathbf{U}}^{(n+1)} = \begin{bmatrix} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{c}}\widetilde{\mathbf{V}}_{\underline{c}} \end{bmatrix}$.

14: Update the coefficients: $\widehat{\mathbf{A}}^{(n+1)} = \begin{bmatrix} \mathbf{A}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{c}}^T\mathbf{A}_{\underline{c}} \end{bmatrix}$.

15: New LDA vectors: $\widehat{\mathbf{V}}^{(n+1)} = \begin{bmatrix} \mathbf{V}_{\underline{k}} \\ (\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{1/2} \end{bmatrix}$.

16: New class centers: $\boldsymbol{\nu}^{(n+1)} = \boldsymbol{\nu}$.

Classification

- Classification is a two step procedure:
  - Project novel image into PCA space
  - Project obtained coefficient vector into LDA space

$$\hat{g}(\mathbf{x}) = (\widehat{\mathbf{U}}^{(n+1)}\widehat{\mathbf{V}}^{(n+1)})^{\top}(\mathbf{x} - \mu^{(n+1)})$$

- The classification function
  - Keeps unchanged

$$\hat{g}(\mathbf{x}) = g(\mathbf{x})$$

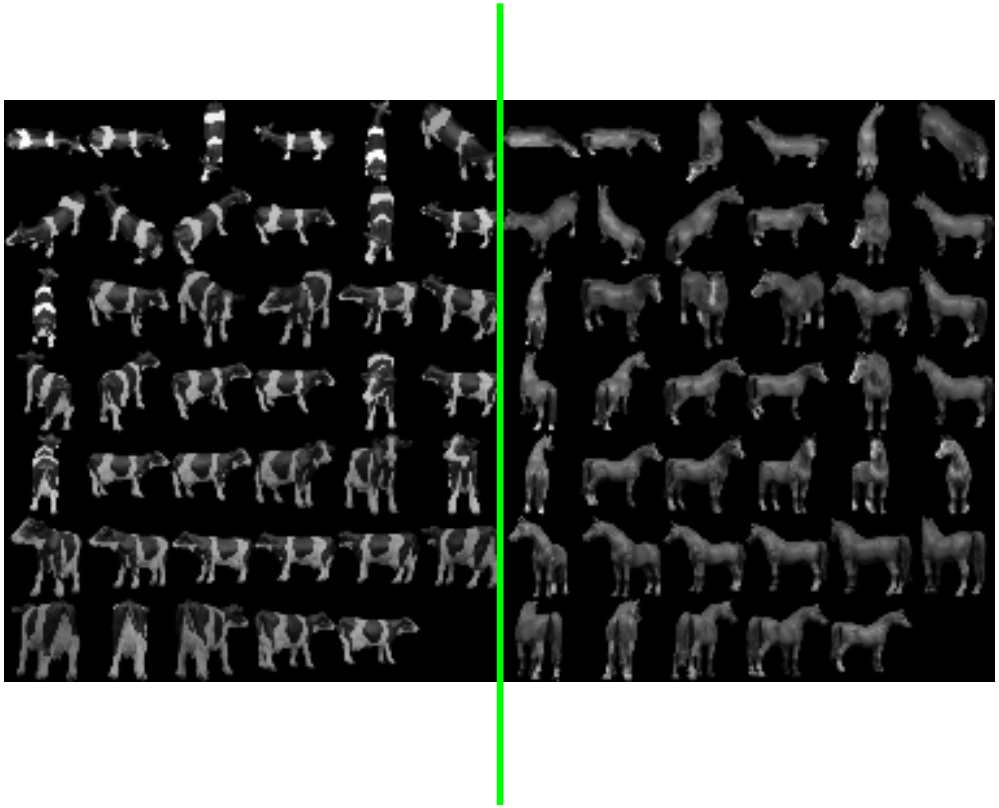
  - Preserves all discriminative information

Proof

$$
\begin{aligned}
\widehat{g}(\mathbf{x}) &= (\widehat{\mathbf{U}}^{(n+1)}\widehat{\mathbf{V}}^{(n+1)})^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left(\left[\mathbf{U}_{\underline{k}}, \mathbf{U}_{\underline{c}}\widetilde{\mathbf{V}}_{\underline{c}}\right]\begin{bmatrix} \mathbf{V}_{\underline{k}} \\ (\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{1/2} \end{bmatrix}\right)^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left[\mathbf{U}_{\underline{k}}\mathbf{V}_{\underline{k}} + \mathbf{U}_{\underline{c}}\mathbf{V}_{\underline{c}}\right]^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left(\left[\mathbf{U}_{\underline{k}}, \mathbf{U}_{\underline{c}}\right]\begin{bmatrix} \mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{c}} \end{bmatrix}\right)^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= (\mathbf{U}\mathbf{V})^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= g(\mathbf{x})\ .
\end{aligned}
$$

ILDA - Example

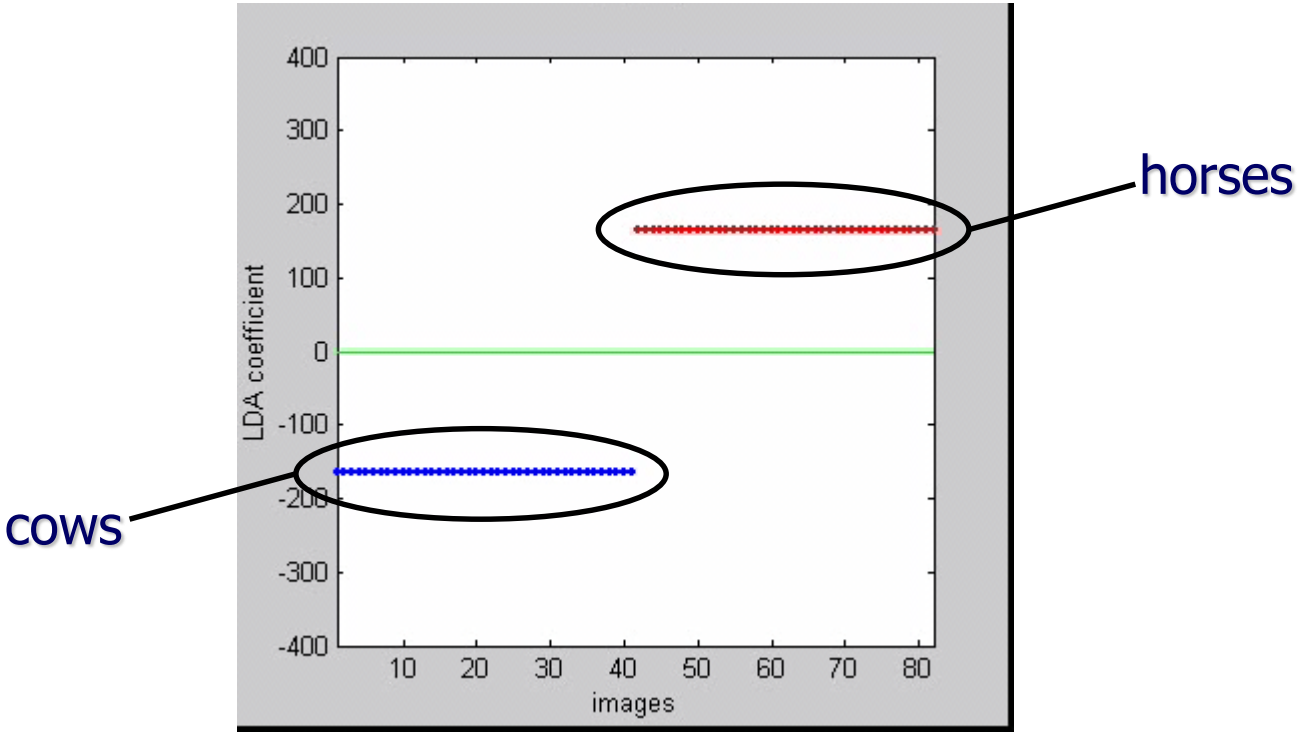
Find a classification vector, which separates cows and horses.



Two classes from CogVis-ETH-80 database.

ILDA - Example

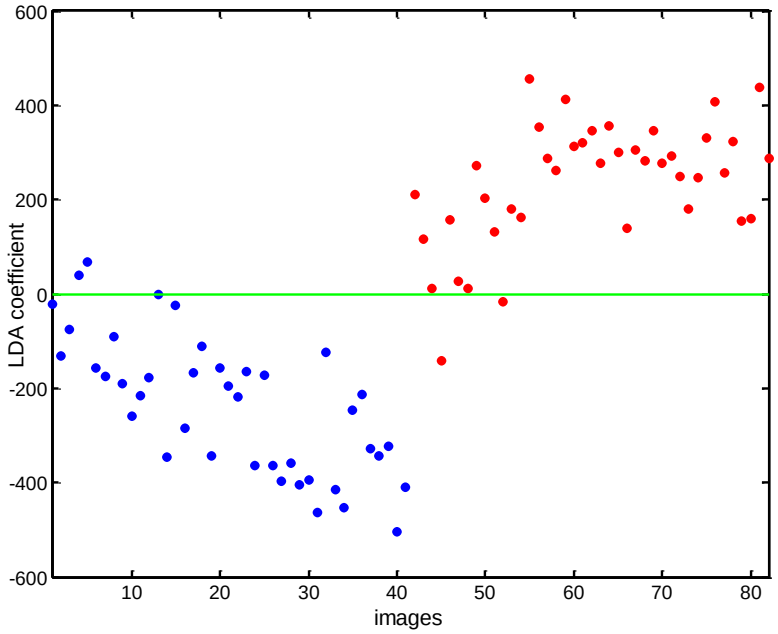
Two classes ⇨ one LDA coefficient.



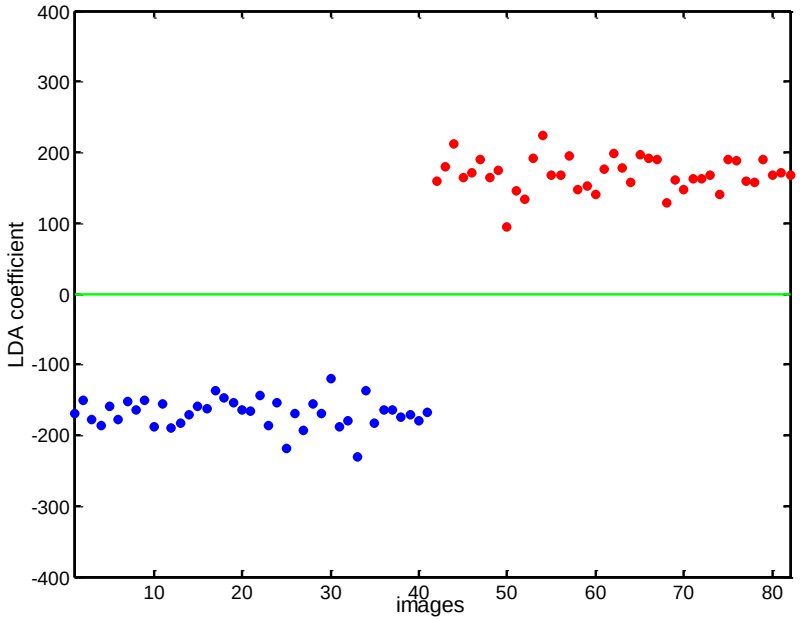
ILDA example

- ILDAonK: LDA on $k<<n-1$ principal components ⇨ some discriminative information has been lost!

- ILDAaAPCA: Estimation of coefficients in the augmented PCA basis ⇨ most of the discriminative information has been preserved!



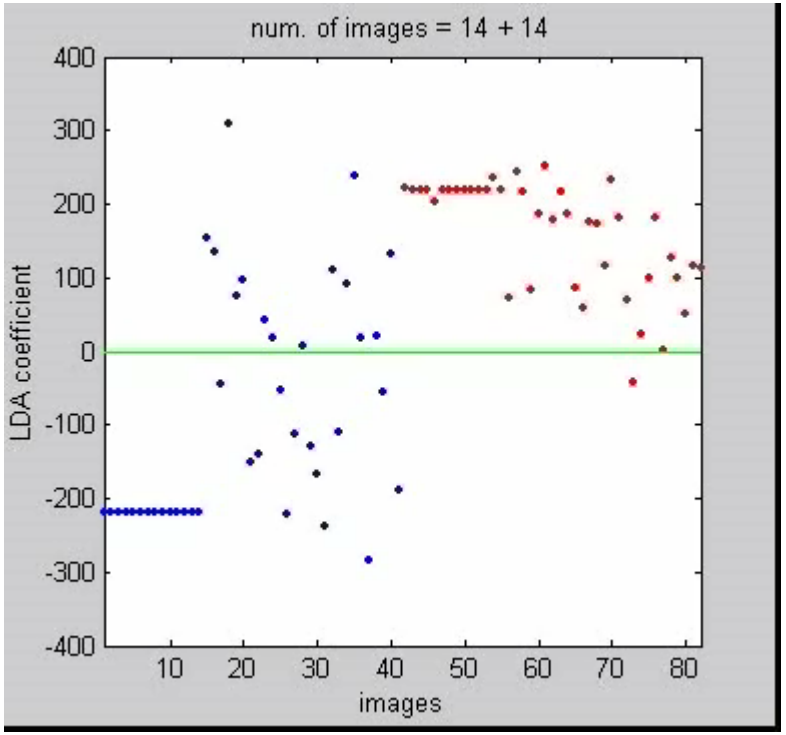
ILDAonK



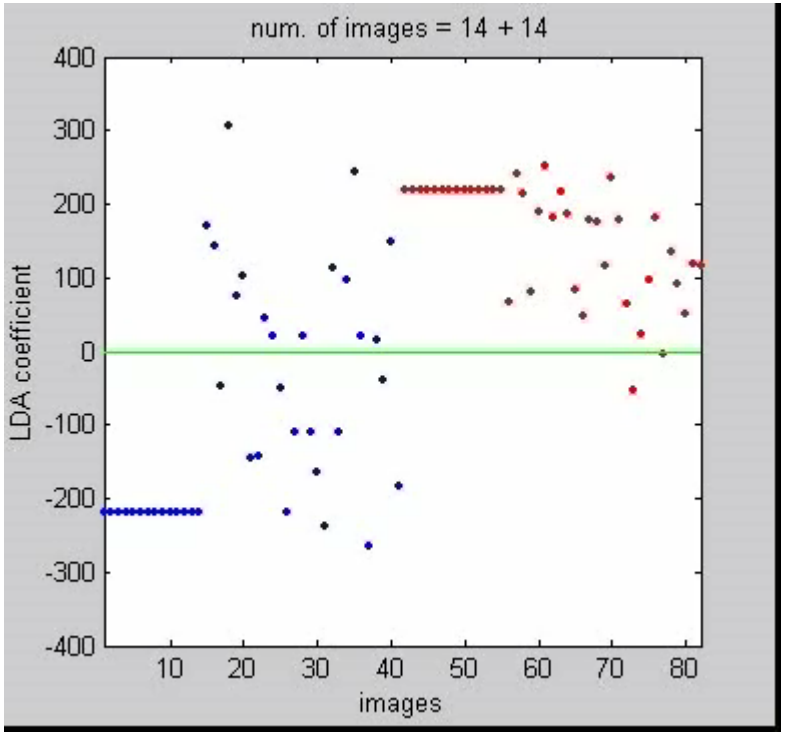
ILDAaPCA

ILDA example

- Classification of all images at each update



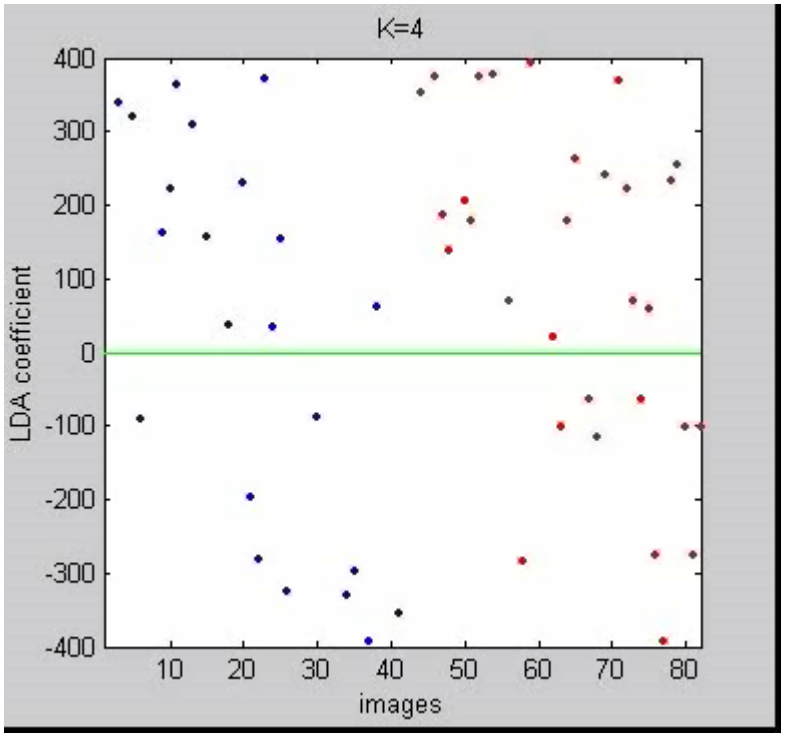
ILDAonK



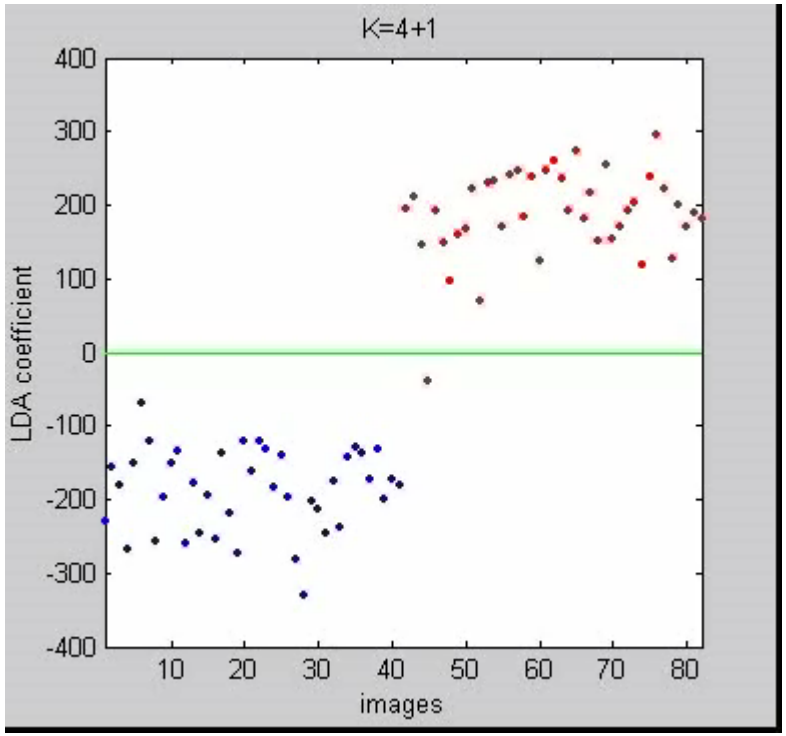
ILDAaPCA

ILDA example

- Various sizes of principal subspace



ILDAonK



ILDAaPCA

Experiments

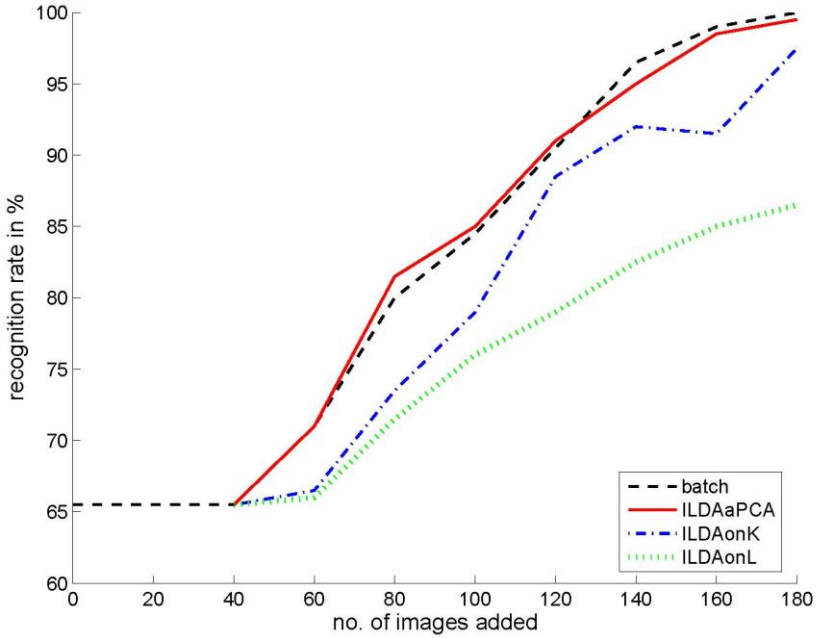
- Aim: Achieve a recognition rate as good as batch method
- Compare
  - BatchLDA: build model from scratch
  - ILDAaPCA: discriminative and reconstructive information
  - ILDAonK: only reconstructive information
  - ILDAonL: only discriminative information
- Experiments
  1. Adding new classes
  2. Adding new instances

Adding new Images (Faces)

- 20 faces, 19 views (side to front), 9 for training



- Starting model : 2 faces of each class → 40 images
- Updates: 7 faces of each class → 140 images

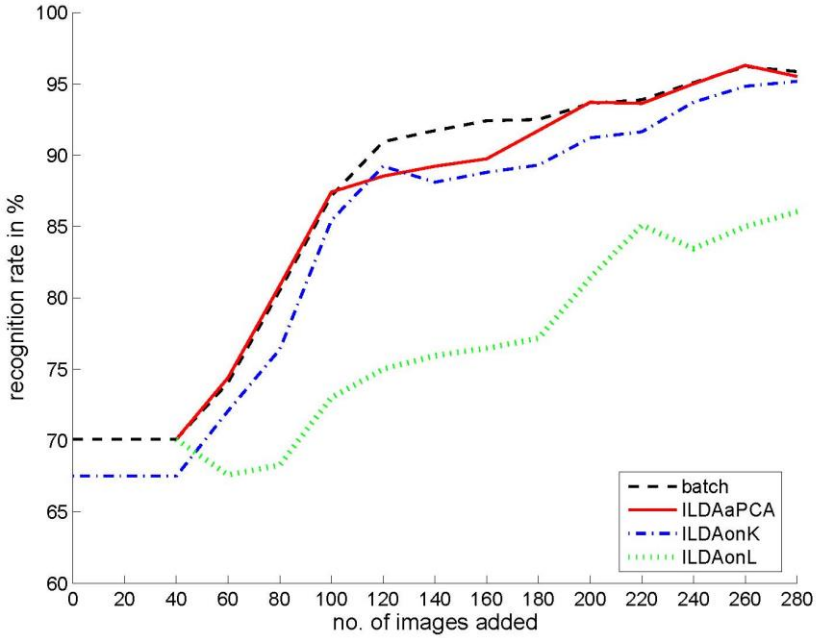


Adding new Images (COIL20)

- 20 objects, 72 views, 14 for training



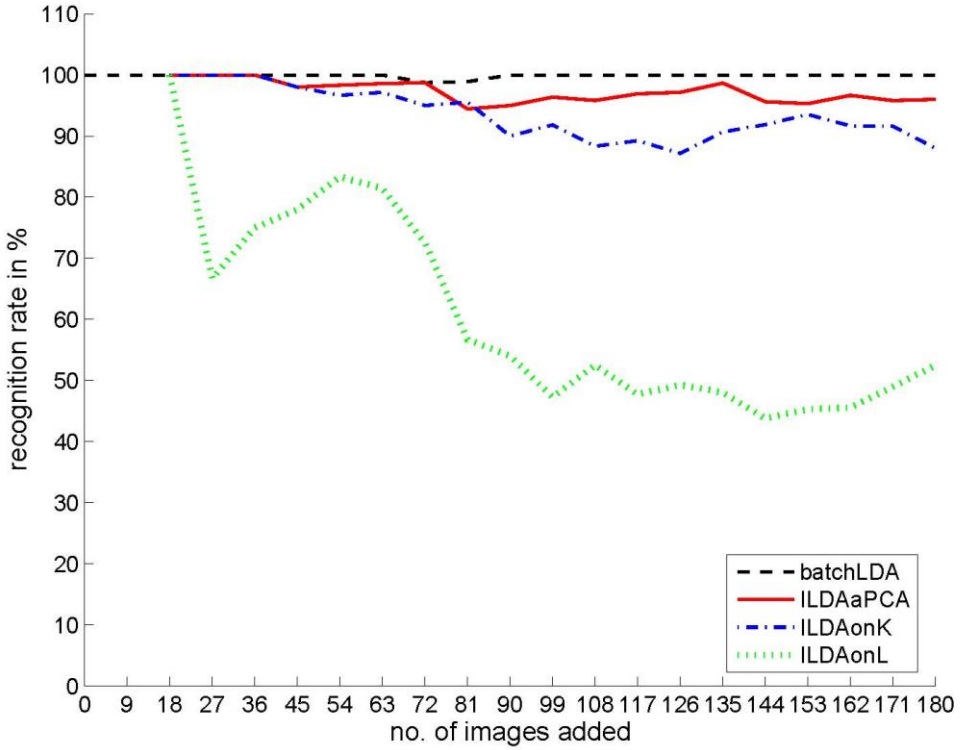
- Starting model : 2 views of each class → 40 images
- Updates: 12 views of each class → 240 images



Adding new Classes (Faces)

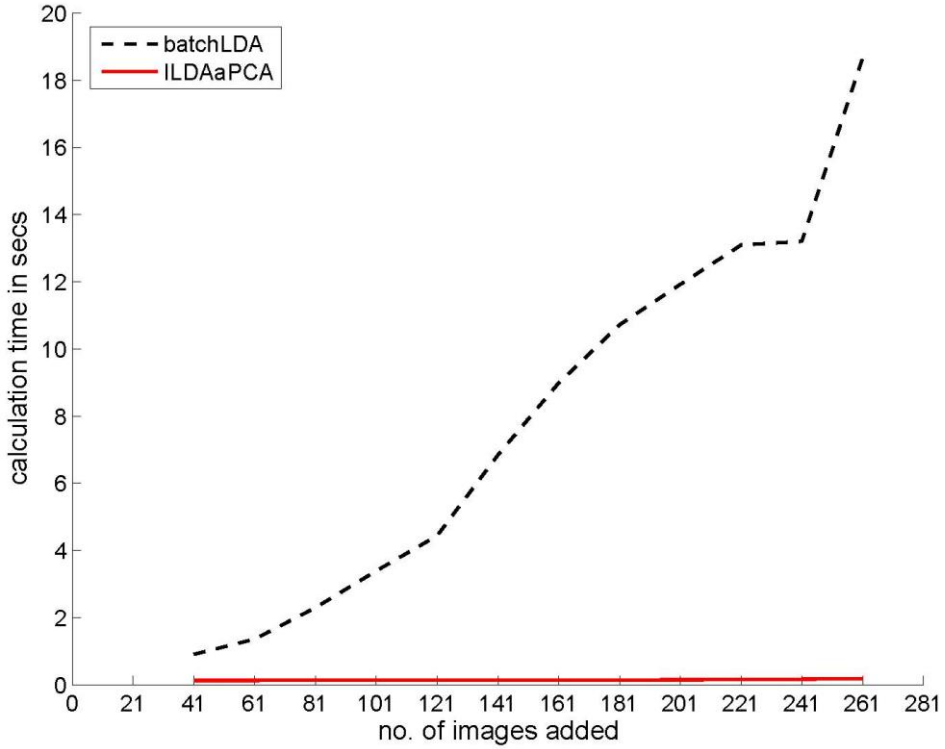
- 20 faces, 19 views (side to front), 9 for training
- Starting model : all faces of two classes → 18 images
- Updates: 18 * all faces of one class → 18*9 images





Time Factor

- COIL20
- Performed on: P4 3.2 GHz CPU, 1GB RAM
- Update from n to n+1 images



Conclusion

- ILDAaPCA combines reconstructive and discriminative methods
- Augmented PCA subspace contains
  - Reconstructive information
    → enables incremental learning
  - Discriminative information
    → enables efficient classification
- Possibilities
  - Update already learned classes
  - Introduce new classes
- Comparison to batch method
  - Similar recognition rate
  - Faster
  - Needs less storage

Literature

- M. Uray, D. Skočaj, P. Roth, H. Bischof and A. Leonardis, *Incremental LDA learning by combining reconstructive and discriminative approaches*, BMVA 2007
  - http://vicos.fri.uni-lj.si/data/publications/urayBMVC07.pdf
- S. Fidler, D. Skočaj and A. Leonardis, *Combining Reconstructive and Discriminative Subspace Methods for Robust Classification and Regression by Subsampling*, IEEE PAMI, 2006
  - http://vicos.fri.uni-lj.si/data/publications/fidlerPAMI06.pdf

- C. M. Bishop, *Pattern recognition and machine learning*, Springer, 2006
- R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification*, Wiley, 2000