Partial Least Squares Regression

Bob Collins

LPAC group meeting

October 13, 2010

Cavaet

- Learning about PLS is more difficult than it should be, partly because papers describing it span areas of chemistry, economics, medicine and statistics, with little agreement on terminology.
- There are also two related but different methods called PLS, one due to Wold and Martens, and the other due to Bookstein (BPLS).
- Within the Wold family, two different algorithms PLS1 and PLS2 have arisen to handle single versus multiple dependent variables.

What is PLS Regression?

- Basically, we want to do linear regression Y = X B
- This is ill-conditioned when the features X have "colinearities" (feature matrix has less than full rank)
- Project the features into a new set of features in a lower-dimensional space. Each such "latent feature" is a linear combination of the original features.
- Do regression using the latent variables
- What distinguishes PLS from other methods (like principal components regression) is how the projection is done.

PCR vs PLS

- In particular, PCR chooses basis vectors of its low dimensional projection to describe as much as possible the variation in the data X.
- However nothing guarantees that the principal components, which “explain” X optimally, will be relevant for the prediction of Y.
- Solution: incorporate information from Y when choosing the projection. We thus choose a projection that describes as well as possible the covariation of data X and labels Y.

PLS1 versus PLS2

- PLS1 – only considers a single class label at a time, so we have a single vector of dependent variables y
- PLS2 – we have multiple class labels, so there is a whole matrix Y of dependent variables
- Possible motivations for PLS2: performing multiclass classification, using one set of latent features. Y class labels may not be independent. May just want to do some exploratory data analysis.
- However, may get better classification results if you just apply PLS1 separately to each column of Y.

Background

Consider linear regression of a dependent variable y (say class label) given a set of independent variables (features) x1, x2, ..., xm.

$$y = b_1 x_1 + b_2 x_2 + \dots + b_m x_m + e$$

Here, the bi are the unknown regression coefficients, and e is a residual error that we will want to make as small as possible.

Rewrite slightly

$$y = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} + e$$

and consider n training samples

$$\underset{n\times 1}{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}} = \underset{n\times m}{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}} \underset{m\times 1}{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}} + \underset{n\times 1}{\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}}$$

Background

Now consider this as a matrix equation

$$\underset{n\times 1}{y} = \underset{n\times m}{X}\ \underset{m\times 1}{b} + \underset{n\times 1}{e}$$

We want a least-squares solution for the unknown regression parameters b such that we minimize the sum of squared errors of the residuals in e

$$\hat{b} = (X'X)^{-1}X'y$$

minimizes $e'e$
SSQ residuals

To use this for predicting class labels y given a new set of feature measurements Xnew, we can now do

$$\hat{y} = X_{new}\,\hat{b}$$

Important note: we have assumed that vector y, and each column of X, have been centered by subtracting their mean values. We may also want to further normalize the columns xi by dividing by their standard deviations (to make scaling comparable across different features).

Background

$$\hat{b} = (X'X)^{-1} X' y$$

minimizes $e'e$
SSQ residuals

Problem: this least-squares solution is ill-conditioned if X' X does not have full rank. This can happen when there are strong correlations ("colinearities") between subsets of features that cause them to only span a lower-dimensional subspace. X' X is certainly not full rank when the number of features m exceeds the number of training samples n.

Solution: project each measurement into a lower-dimensional subspace spanned by the data. We can think of this as forming a smaller set of features, each being the linear combination of the original set of features. These new features are also called "latent" variables.

PCR – Principal Components Regression

Basic idea: Use SVD to form new latent vectors (principal components) associated with a low-rank approximation of X

First apply SVD to X

$$\underset{n\times m}{X} = \underbrace{\underset{n\times m}{U}\,\underset{m\times m}{D}}_{T}\,\underbrace{\underset{m\times m}{V}'}_{P'} = d_1 u_1 v_1' + d_2 u_2 v_2' + \ldots + d_m u_m v_m'$$

where U'U = V'V = I , and D is a diagonal matrix of singular values in descending order of magnitude d1 >= d2 >= ... >= dm

Columns of T: "principal components", "factor scores", "latent variables".
Columns of V: "loadings"

PCR – Principal Components Regression

Form a low-rank approximation of X by keeping just the first k<m principal components (the ones associated with the k largest singular values).

$$X \approx T_k P_k' = d_1 u_1 v_1' + d_2 u_2 v_2' + \dots + d_k u_k v_k'$$

We now can consider a regression problem in a lower-dimensional feature space by using the latent variables as our new features

$$y = T_k c + f$$

$$\hat{c} = (T_k' T_k)^{-1} T_k' y$$

Note that the columns of T are orthogonal to each other (recall T = U D), thus (Tk' Tk) is a diagonal matrix (values on the diagonal are the squares of the singular values), so it is really easy to solve this new regression problem.

PCR – Principal Components Regression

To use use the solution to this reduced dimension regression problem to solve the original problem of predicting class labels y given a new set of feature measurements Xnew, we can now do

$$\hat{Y} = X_{new} P_K \hat{c}$$

Xnew Pk is projection To k latent variables

because $X = T_K P_K'$

$$\Rightarrow X P_K = T_K P_K' P_K = T_K$$

Since T = X P, and P(=V) is an orthonormal matrix that performs a change of basis,, we can think of X Pk as the rotation and projection of old features X (in m-dim space) into new latent variables T (in k-dim space)

Key point: after projecting into latent variables, there is no reason we have to restrict ourselves to linear regression! We instead could just use these new features and do a nonlinear regression using SVMs, quadratic discriminant functions, or whatever we want.

Digression (but will become relevant)

Power method algorithm, for computing eigenvalues, eigenvectors.

```
%find first k largest eigenvalues and eigenvectors
Evec = [];
Eval = [];
for j=1:k
    [dummy,c] = max(max(abs(A)));    %find max norm column c
    tmp = A(:,c);
    u = tmp / sqrt(dot(tmp,tmp));
    %iterate (should use a convergence test)
    for i=1:20
        u = A' * u;
        u = u / sqrt(dot(u,u));      %unit vector
    end
    lam = u' * A * u;                %compute eigenvalue
    Evec(:,j) = u;                   %store eigenvalue/vector
    Eval(j) = lam;

    A = A - lam*u*u';                %deflation
end
```

Digression (but will become relevant)

Power-method-like algorithm for computing X = T P' (basically, SVD).

```
%find first k largest principal components vectors
Tmat = [];
Pmat = [];
for j=1:3
    [dummy,c] = max(max(abs(X)));  %find max norm column c
    t = X(:,c);
    %iterate (should use a convergence test)
    for i=1:20
        p = X' * t;
        p = p / sqrt(dot(p,p));     %right singular vector vj of UDV'
        t = X * p;                  %principal component (dj * uj) of UDV'
    end
    Tmat(:,j) = t;                  %store principal component and "loading"
    Pmat(:,j) = p;
    X = X - t*p';                   %deflation
end
```

Working Towards PLS

Recall the decomposition X = U D V’ = T P’ and that T = X P rotates and projects columns of X into a set of orthogonal columns in T, the so-called principal components or latent variables.

First, note that vectors in P (=V) are eigenvectors of X’ X

$$X'X = VDU'UDV' = VD^2V'$$

Now, if we have centered out feature measurements (columns of X) by subtracting the mean of each column, X’ X has a specific interpretation

$$\text{let } X = \left.\begin{bmatrix} f_1 & f_2 \end{bmatrix}\right\} n \text{ samples} \qquad \begin{aligned} f_1 &= a - \mu_a \\ f_2 &= b - \mu_b \end{aligned}$$

$$X'X = \begin{bmatrix} \sum_{i=1}^{n} (a-\mu_a)^2 & \sum (a-\mu_a)(b-\mu_b) \\ \sum (a-\mu_a)(b-\mu_b) & \sum (b-\mu_b)^2 \end{bmatrix}$$

Sample Covariance Matrix!

Working Towards PLS

Thus, the first k principal components maximize the ability to describe the covariance or spread of the data in X, that is Cov(X,X) = X' X. For example, the first component $t1 = X\ p1$ maximizes $cov(t1,t1) = p1\ X'\ X\ p1$.

Problem: rotation and data reduction to explain the principal variation in X is not guaranteed to yield latent features that are good for predicting y.

Solution, and the basic idea behind PLS: project to latent variables that maximize the covariation between Xand y, namely Cov(X,y).

So for the first latent vector, search for a vector $t = X\ w$ such that we

maximize $cov(Xw, Y)$ subject to $w'w = 1$

NIPALS Algorithm (PLS1)

note that unlike power method for SVD, there is no iteration to compute each principal component

10/12/2010

$$\overset{m\times 1}{w} = \overset{m\times n}{X'}\ \overset{n\times 1}{y}$$

$$\overset{m\times 1}{w} = w / \|w\|$$

w is unit vector that maximize cov(Xw, y)
i.e. $\max_{\|x\|=1} w'X'y$
i.e. maximize dot product of w and $X'y$
so w points in direction $X'y$!

$$\overset{n\times 1}{\tau} = \overset{n\times m}{X}\ \overset{m\times 1}{w}$$

new "latent" feature

$$\overset{m\times 1}{p} = \overset{m\times n}{X'}\ \overset{n\times 1}{\tau} / \tau'\tau$$

$X = \tau p'$ $\quad X' = p\tau'$
$X'\tau = p\tau'\tau$ $\quad X'\tau(\tau'\tau)^{-1} = p$

$$\overset{n\times m}{X} = \overset{n\times m}{X} - \overset{n\times 1}{\tau}\ \overset{1\times m}{p'}$$

residual in X
(unaccounted for by $\tau p'$)

$$\overset{1\times 1}{\hat{c}} = \overset{1\times n}{\tau'}\ \overset{n\times 1}{y} / \tau'\tau$$

linear regression coeff of y as a function of τ
$y = \tau c$ $\quad \tau' y = \tau'\tau c$
$c = \tau' y\,(\tau'\tau)^{-1}$

$$\overset{n\times 1}{y} = \overset{n\times 1}{y} - \overset{n\times 1}{\tau}\ \overset{1\times 1}{\hat{c}}$$

residual in y

deflation of X and y

this gives first latent variables t and u... apply again to get next ones, and so on

PLS2

PLS2 – we have multiple class labels, so there is a whole matrix Y of dependent variables and matrix B of regression coefficients.

$$\underset{n \times r}{[\,Y\,]} = \underset{n \times m}{[\,X\,]}\,\underset{m \times r}{[\,B\,]} + \underset{n \times r}{[\,E\,]}$$

We could treat this as multiple, separate PLS1 problems (and that might even be best from a classification accuracy standpoint), but if you insist on simultaneous decomposition, we can project both X and Y into latent variable spaces T and U, such that T and U are coupled, and chosen to maximize cov(X,Y) = X' Y .

$$X = TP'$$

$$Y = UQ'$$

$$U = TC$$

Then we can learn a regression function between the T and U latent variables, using linear regression or SVM or ...

PLS2 Algorithm

Let u be an arbitrary col of Y

until convergence

$$w = X'u \,/\!/\, \|X'u\|$$

u plays role of y here. It represents information from all cols of Y

$$\tau = Xw$$

same as PLS1

$$q = Y'\tau \,/\!/\, \|Y'\tau\|$$

$$u = Yq$$

do similar computation with Y, to get score vec and latent vec

$$p = X'\tau / \tau'\tau$$

$$X = X - \tau p'$$

same as PLS1

$$\hat{c} = \tau' u / \tau'\tau$$

regression coeff btw two latent vectors

$$Y = Y - \hat{c}\,\tau q'$$

residual in Y

this gives first latent variables t and u... apply again to get next ones, and so on.

Note, if Y has only 1 column, this reduces to PLS1 (q becomes 1, u becomes y)

Comparisons

	Overview (as related to SVD)	Latent variables
PLS1	w_i is left singular vector of X' y Deflate X and y	$t_i = X\, w_i$
PLS2	w_i is left singular vector of X' Y q_i is right singular vector of X' Y Deflate X and Y	$t_i = X\, w_i$ $u_i = X\, q_i$
Bookstein	svd(X' Y) = W D Q' extracts multiple left/right singular vectors simultaneously no deflation (since no iteration)	$T = X\, W$ $U = Y\, Q$

ICCV 2009

Human Detection Using Partial Least Squares Analysis

William Robson Schwartz, Aniruddha Kembhavi, David Harwood, Larry S. Davis

University of Maryland, A.V.Williams Building, College Park, MD 20742

schwartz@cs.umd.edu, anikem@umd.edu, harwood@umiacs.umd.edu, lsd@cs.umd.edu

Abstract

Significant research has been devoted to detecting people in images and videos. In this paper we describe a human detection method that augments widely used edge-based features with texture and color information, providing us with a much richer descriptor set. This augmentation results in an extremely high-dimensional feature space (more than 170,000 dimensions). In such high-dimensional spaces, classical machine learning algorithms such as SVMs are nearly intractable with respect to training. Furthermore, the number of training samples is much smaller than the dimensionality of the feature space, by at least an order of magnitude. Finally, the extraction of features from a densely sampled grid structure leads to a high degree of multicollinearity. To circumvent these data characteristics, we employ Partial Least Squares (PLS) analysis, an efficient dimensionality reduction technique, one which preserves significant discriminative information, to project the data onto a much lower dimensional subspace (20 dimensions, reduced from the original 170,000). Our human detection system, employing PLS analysis over the enriched descriptor set, is shown to outperform state-of-the-art techniques on three varied datasets including the popular INRIA pedestrian dataset, the low-resolution gray-scale DaimlerChrysler pedestrian dataset, and the ETHZ pedestrian

Figure 1. Image demonstrating the performance of our system in a complex scene. The image (689×480 pixels) is scanned at 10 scales to search for humans of multiple sizes. We achieve minimal false alarms even though the number of detection windows is 44,996 (best visualized in color).

ods consists of a generative process where detected parts of the human body are combined according to a prior human model. The second class of methods considers purely statistical analysis that combine a set of low-level features within a detection window to classify the window as containing a human or not. The method presented in this paper belongs to the latter category.

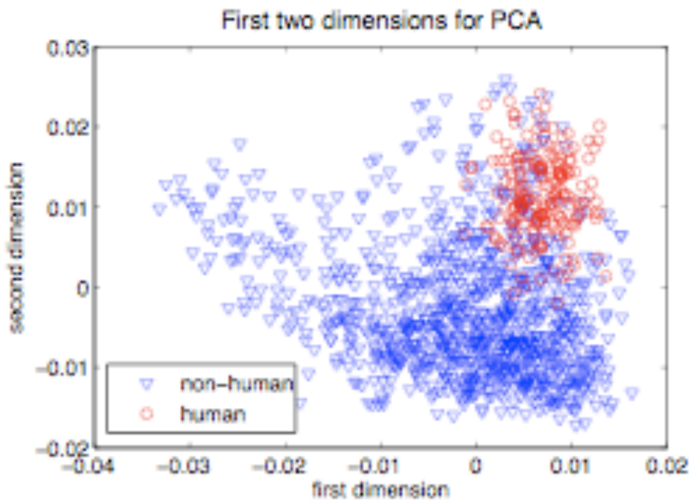
Schwartz et.al. Approach

- Sliding window approach to pedestrian detection
- Compute a feature vector within each window and try to classify it as human or non-human
- Feature vector consists of features extracted from overlapping blocks within a candidate detection window
- Features computed in each block are
  - HOG descriptors (e.g. Dalaal and Triggs)
  - texture features computed from co-occurrence matrices
  - color frequency (number of times each color channel contained the highest gradient magnitude when computing HOG features)
- Full feature vector has more than 170,000 dimensions!

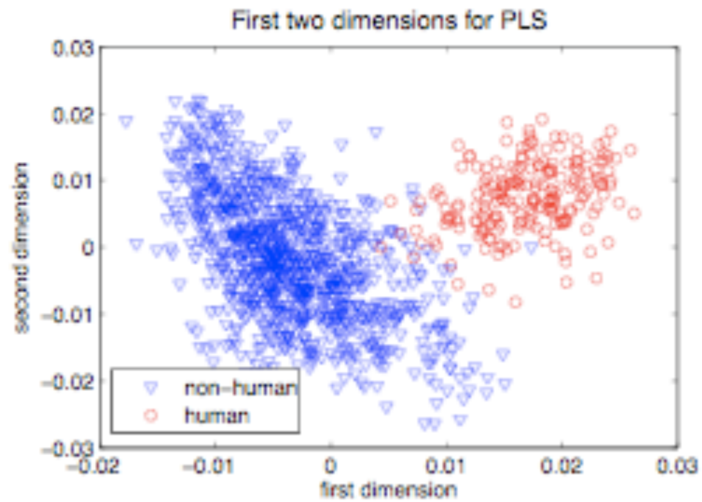
Schwartz et.al. Approach

- Use PLS1 to project the 170,000 dimensional feature space down to 20 dimensions
- Train a Quadratic Discriminant Analysis (QDA) classifier in the 20 dimensional latent space. Noted you could also use SVM, but since PLS gives good separability between classes, it is possible to use the simpler (and less expensive) classifier.
- Compared performance with other classifiers using 10-fold cross-validation.

PCA versus PLS



(a) PCA - first two dimensions

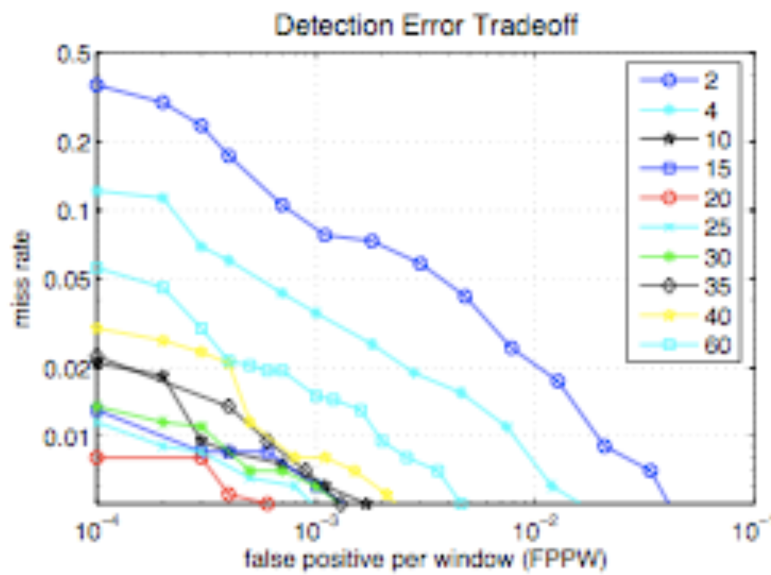


(b) PLS - first two dimensions

PLS gives better class separability for the first 2 dimensions

PCA versus PLS

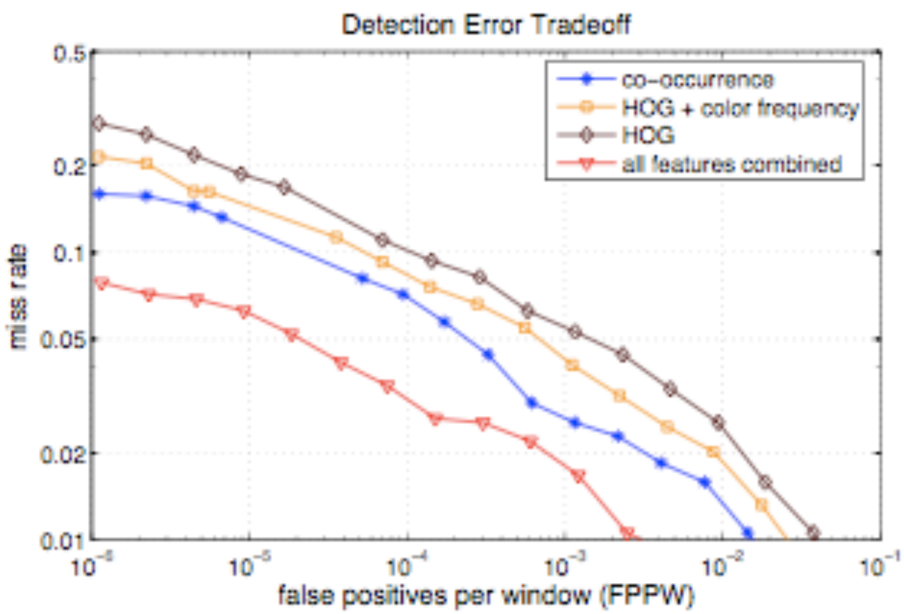
(c) PCA - cross-validation

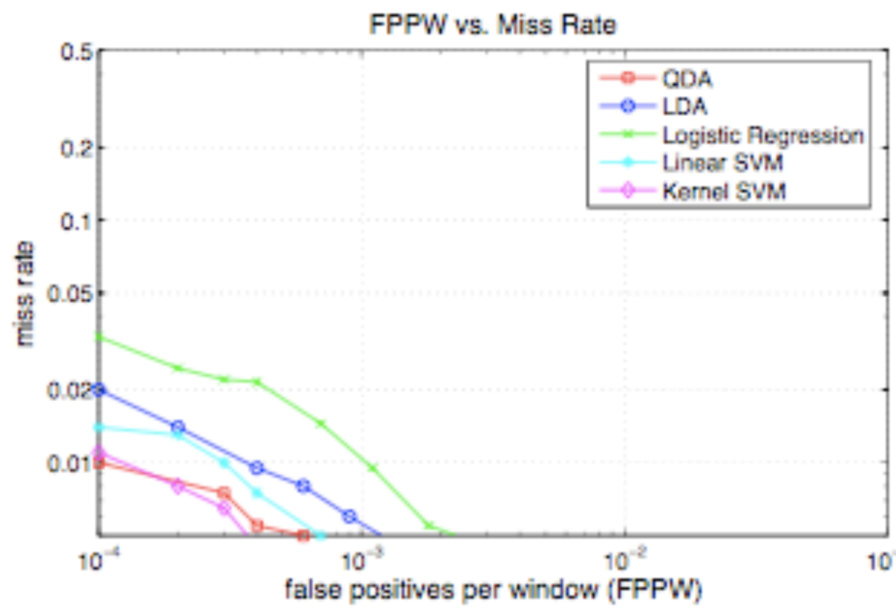


(d) PLS - cross-validation

PCA worked best with a latent space of 180 dimensions
PLS worked best with a latent space of 20 dimensions

Tuning





Using HOG + Texture + Color Frequency together did better than individually

Using Kernel SVM or QDA did best for classification after PLS reduction

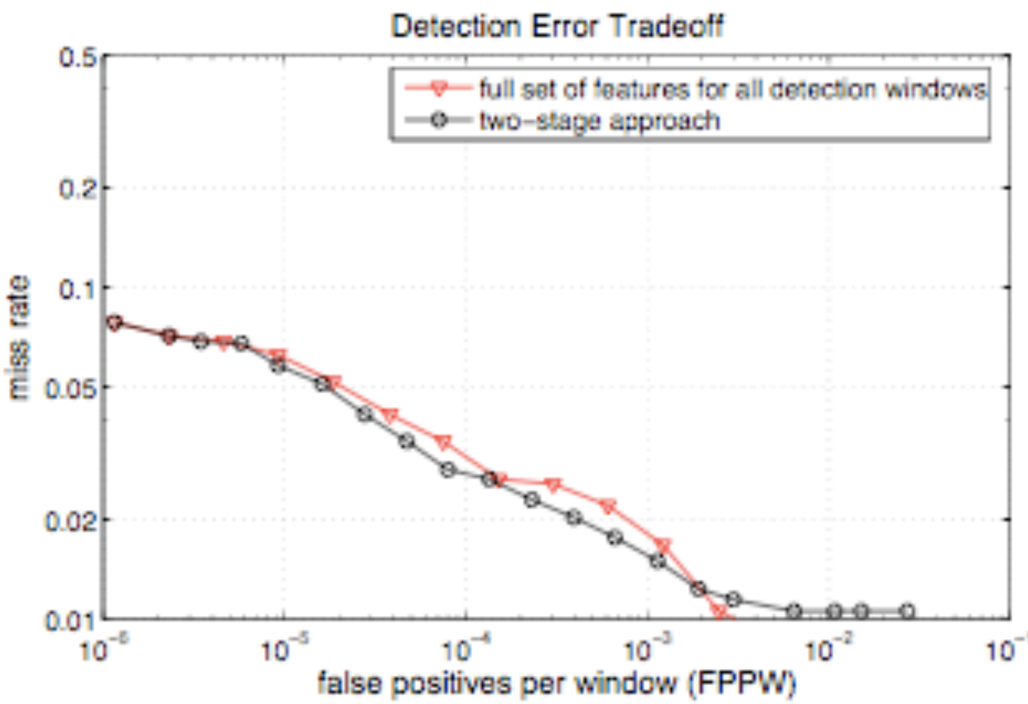
It is computational worth it...

# samples	PLS + QDA	SVM
200	23.63	131.72
600	62.62	733.63
1000	97.38	1693.50
1400	135.81	2947.51
1800	174.57	4254.63
2200	213.93	-
11370	813.03	-

Table 1. Time, in seconds, to train SVM and PLS + QDA models. The number of features per sample is 170,820. The training time increases with an increase in the number of training samples.

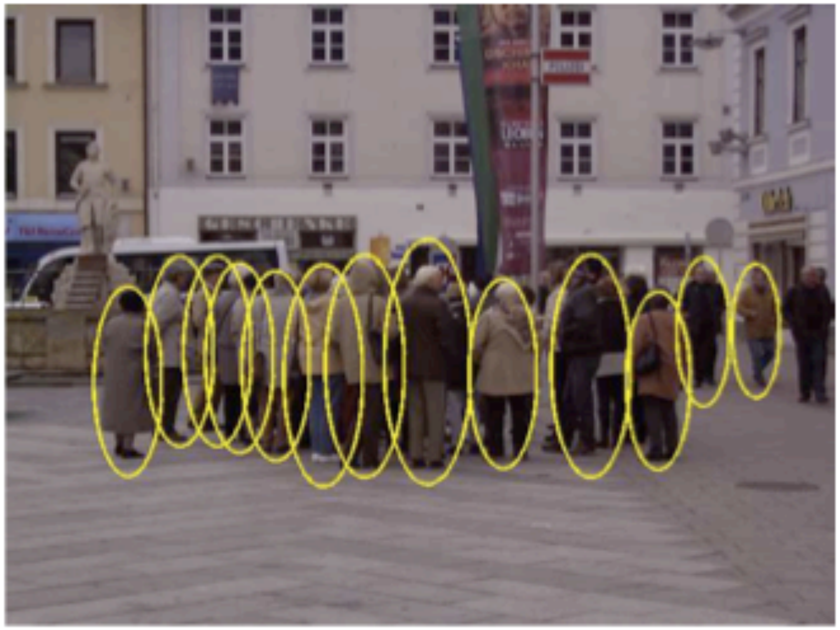
Concern with Runtime

To speed up classification during run time, they came up with a two-stage approach where they first do classification using a smaller set of features from a subset of most discriminative blocks (determined offline). Windows that pass that test are then analyzed with the full set of features.



this graph shows the 2-stage approach does not degrade overall performance

Some sample results



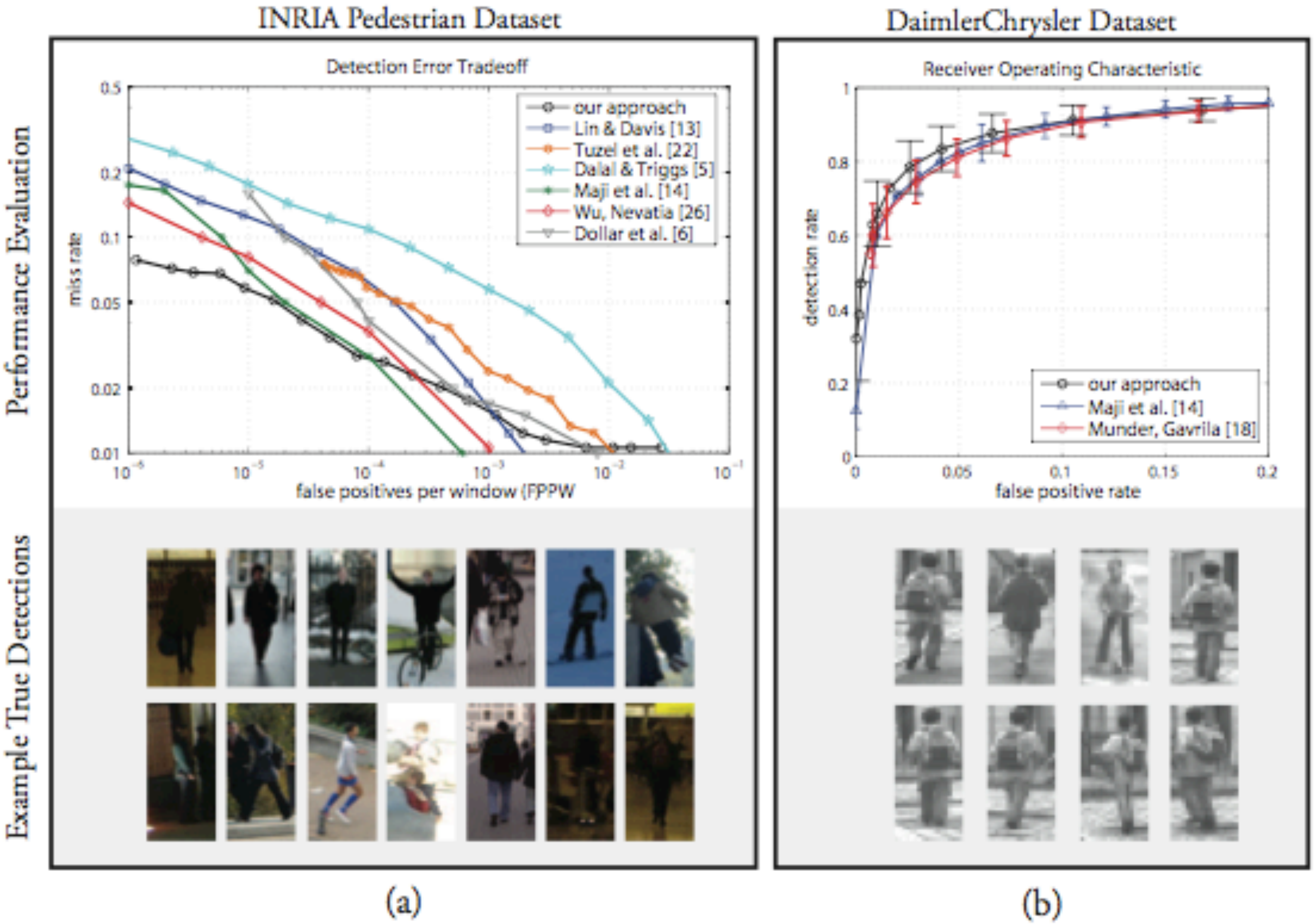
(a) 640 × 480 (41,528 det. windows)

(b) 1632 × 1224 (389,350 det. windows)

(c) 1600 × 1200 (373,725 det. windows)

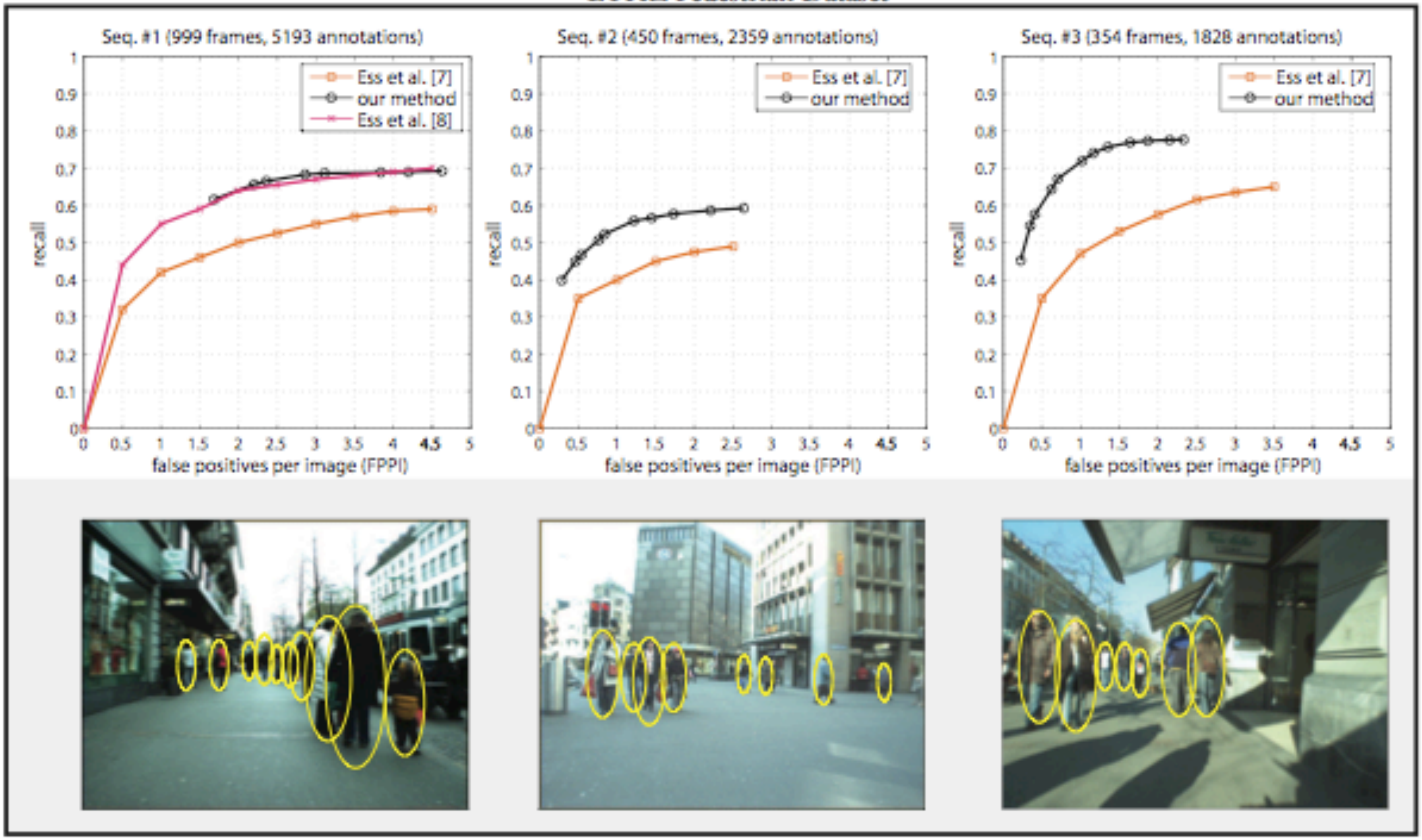
Remember to show the video

Evaluations



Evaluations

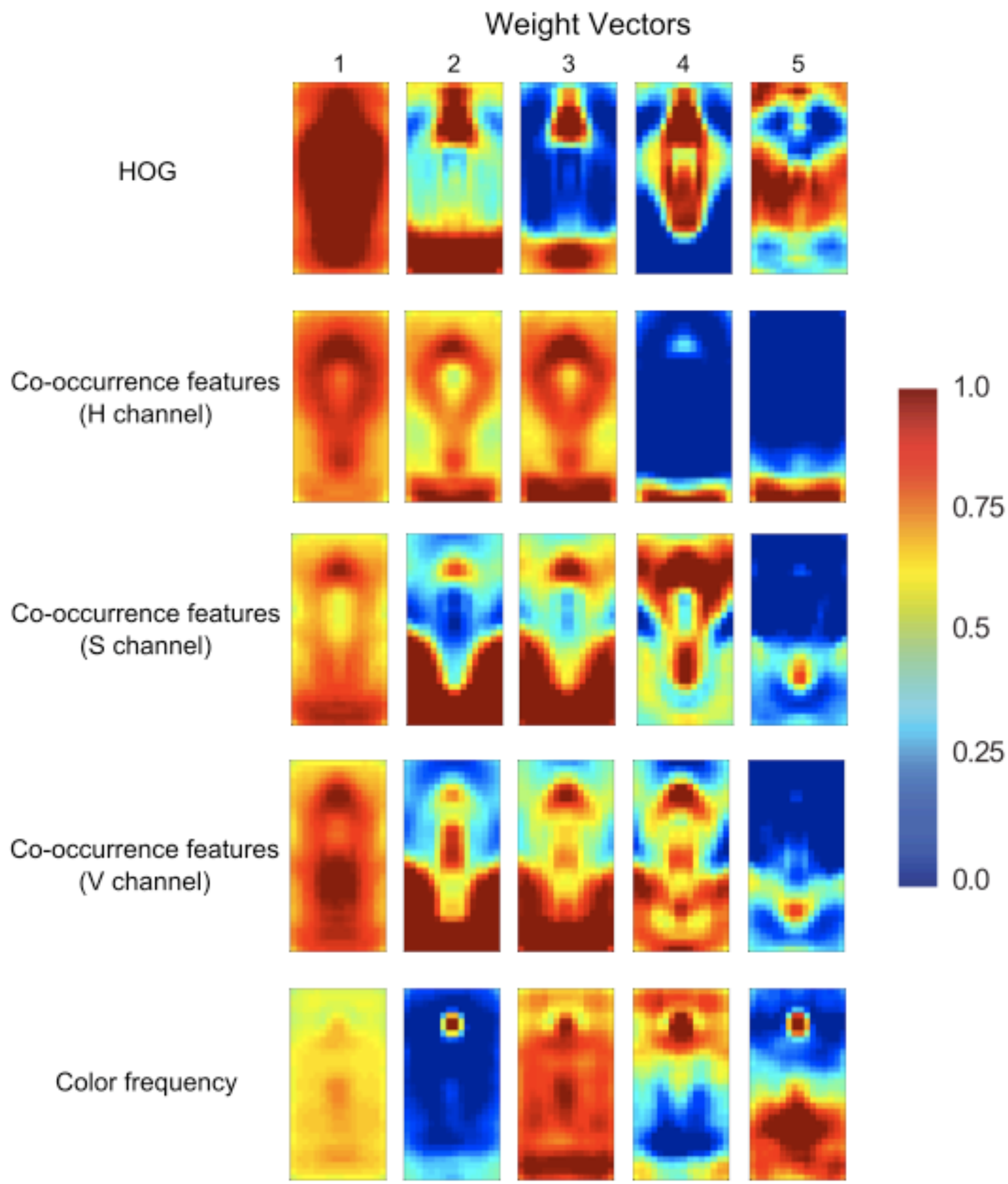
ETHZ Pedestrian Dataset



(c) (d) (e)

Analysis

plotting the set of weight vectors w (recall these are the left singular vectors of X’ y) gives some indication about what features/location contribute most to each latent variable.



NEUROIMAGE 3, 143–157 (1996)
ARTICLE NO. 0016

Spatial Pattern Analysis of Functional Brain Images Using Partial Least Squares

A. R. McINTOSH,* F. L. BOOKSTEIN,† J. V. HAXBY,‡ AND C. L. GRADY*,§

*Rotman Research Institute of Baycrest Centre, 3560 Bathurst Street, University of Toronto, Toronto, Ontario M6A 2E1, Canada; †Institute of Gerontology, University of Michigan, Ann Arbor, Michigan 48109; ‡Functional Brain Imaging Section, Laboratory of Psychology & Psychopathology, National Institute of Mental Health, National Institutes of Health, Bethesda, Maryland 20892; and §Laboratory of Neurosciences, National Institute on Aging, Bethesda, Maryland 20892

Received December 13, 1995

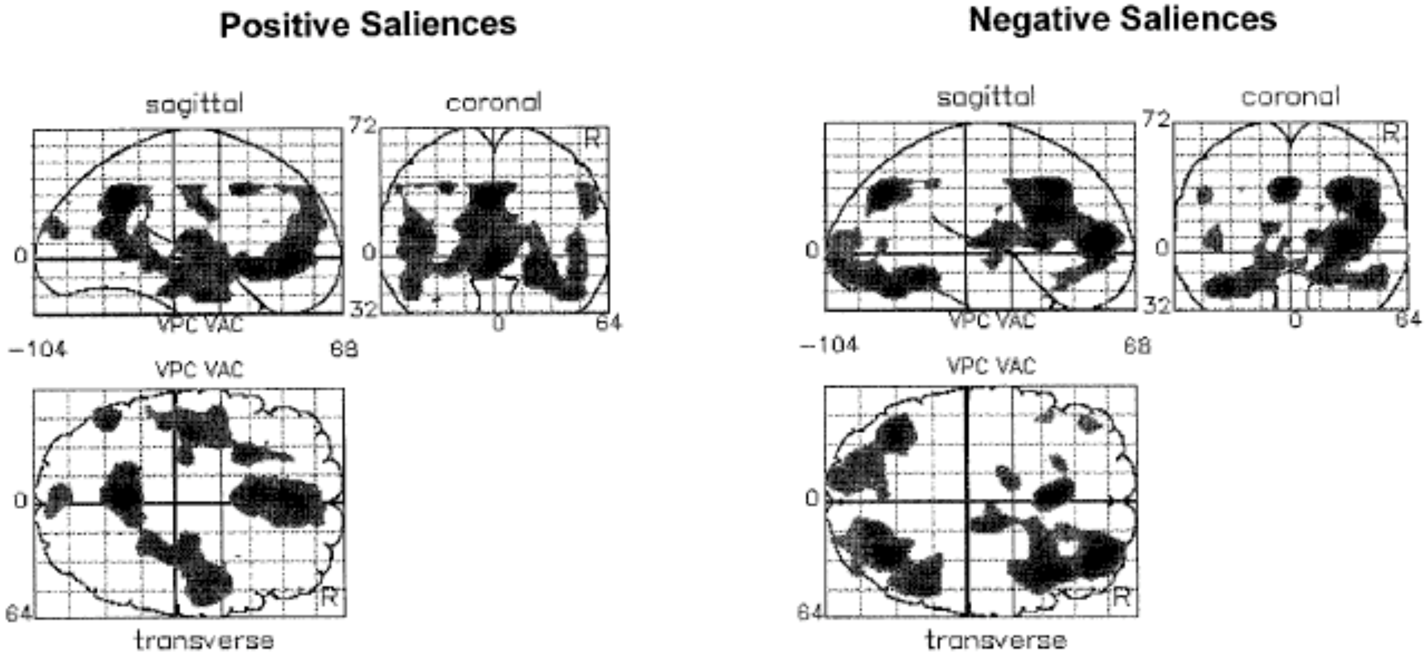
This paper introduces a new tool for functional neuroimage analysis: partial least squares (PLS). It is unique as a multivariate method in its choice of emphasis for analysis, that being the covariance between brain images and exogenous blocks representing either the experiment design or some behavioral measure. What emerges are spatial patterns of brain activity that represent the optimal association between the images and either of the blocks. This process differs substantially from other multivariate methods in that rather than attempting to predict the individual val-

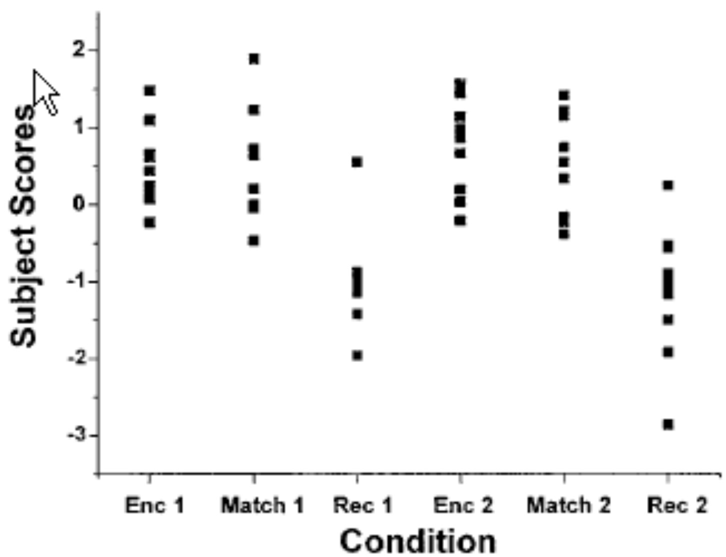
ment design or some behavioral measure. Its greatest strength is the flexible treatment of images in the context of simultaneous prediction of those images by their causes (e.g., aspects of the task design) and prediction by those images of their effects (e.g., measures of behavior). Partial least squares extracts certain features that are inaccessible by other methods, while overlooking some complexities for which other methods may be more suited.

Most of the contemporary techniques for analysis of functional neuroimaging data are variations of univari-

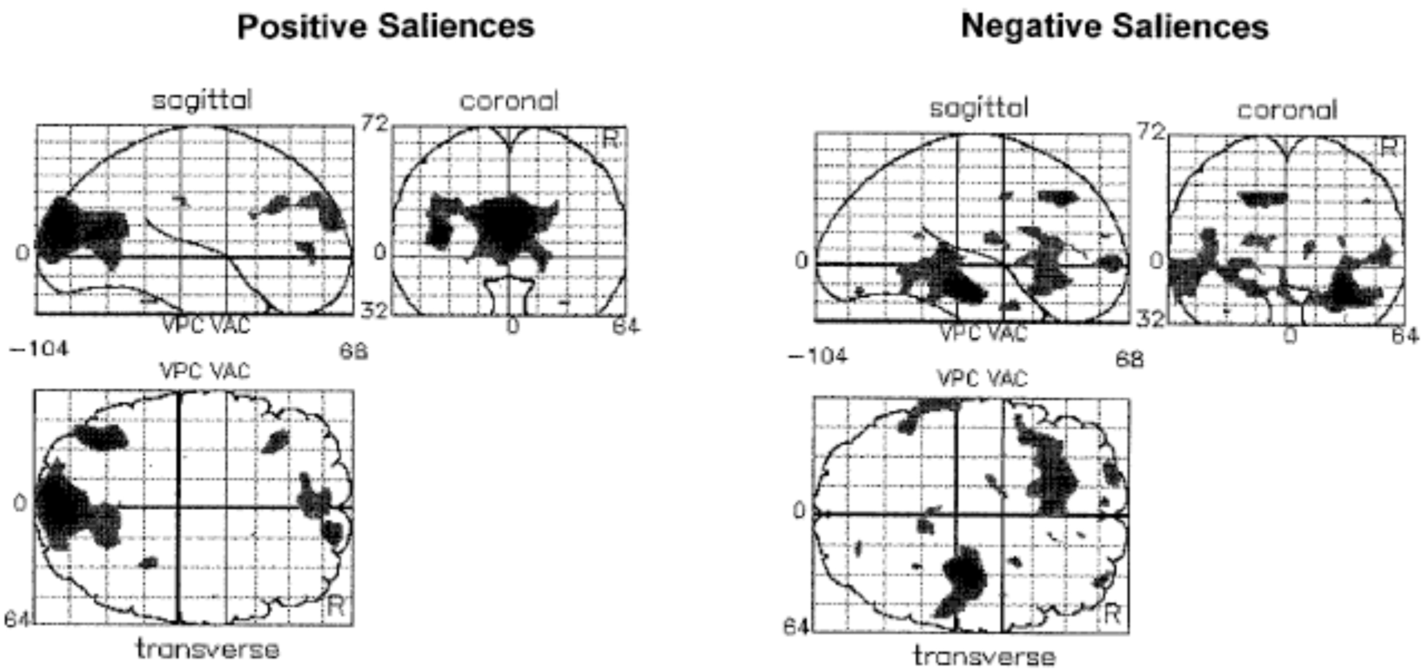
Bookstein Approach

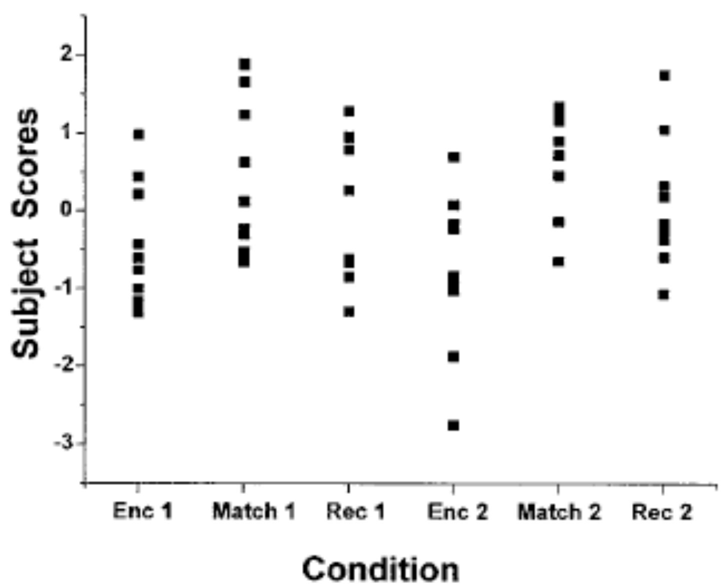
- Original variables X are the intensity values in a 3D volumetric PET scan, concatenated into a big vector
- Want to explore covariation of locations in the brain with different tasks Y
- Uses PLS (the Bookstein version!) to extract the "singular images" (weight vectors wi) from X' Y
- Then plot these with respect to 3D brain coordinates





In summary, the PLS activation analysis shows that the dominant (first) pattern distinguished recognition of faces from encoding and face matching. The singular image incorporates positive saliences for posterior and ventral anterior cingulate cortices, anterior temporal cortices, and right hippocampus, and negative saliences for right prefrontal and dorsal anterior cingulate, ventral occipital and cerebellum, and thalamus. The scores are equal for encoding and face matching, suggesting that the areas identified in the first SI do not differentiate encoding and matching. The second SI



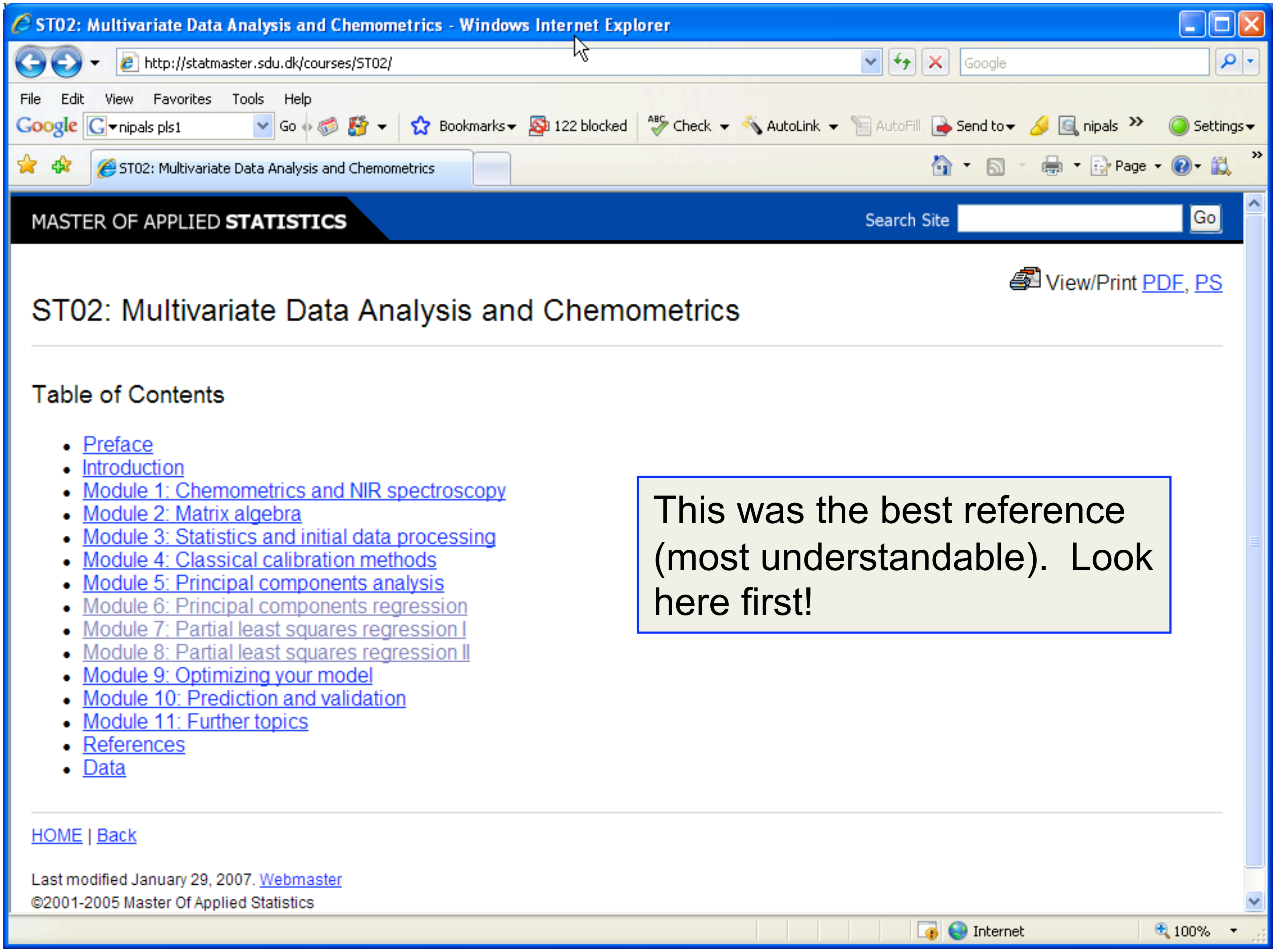


differentiate encoding and matching. The second SI distinguishes encoding from matching with positive saliences for dorsal occipital cortex and negative for ventral–anterior right parahippocampal gyrus and left prefrontal cortex. Scores in the recognition condition were most similar to those from encoding. In view of the similarity in scores for both memory conditions on the second SI, it is possible that these regions represent general memory operations. There have been suggestions that recognition of previously presented information requires reactivation of some of the same regions engaged in the initial encoding episode (Tulving and Thompson, 1973; Nyberg *et al.*, 1995). The PLS results are consistent with this possibility.

References

- Schwartz et.al., “Human Detection Using Partial Least Squares Analysis”, ICCV’09
- McIntosh et.al., “Spatial Pattern Analysis of Functional Brain Images using Partial Least Squares,” Neuroimage 3, 1996.
- Helland, “Partial Least Squares Regression and Statistical Models,” Scandinavian Journal of Statistics, Vol. 17, No. 2 (1990), pp. 97-114
- Abdi, “Partial least squares regression and projection on latent structure regression (PLS Regression)” Wires Computational Statistics, Wiley, 2010
- http://statmaster.sdu.dk/courses/ST02 [this was the best reference]

References



This was the best reference (most understandable). Look here first!