Combining Variable Selection with Dimensionality Reduction

Lior Wolf & Stanley Bileschi

AI Memo 2005-009
CBCL Memo 247

March 2005

Abstract

This paper bridges the gap between variable selection methods (e.g., Pearson coefficients, KS test) and dimensionality reduction algorithms (e.g., PCA, LDA). Variable selection algorithms encounter difficulties dealing with highly correlated data, since many features are similar in quality. Dimensionality reduction algorithms tend to combine all variables and cannot select a subset of significant variables.

Our approach combines both methodologies by applying variable selection followed by dimensionality reduction. This combination makes sense only when using the same utility function in both stages, which we do. The resulting algorithm benefits from complex features as variable selection algorithms do, and at the same time enjoys the benefits of dimensionality reduction.[1]

[1]This report describes research done at the Center for Biological & Computational Learning, which is in the McGovern Institute for Brain Research at MIT, as well as in the Dept. of Brain & Cognitive Sciences, and which is affiliated with the Computer Sciences & Artificial Intelligence Laboratory (CSAIL).

This research was sponsored by grants from: Office of Naval Research (DARPA) Contract No. MDA972-04-1-0037, Office of Naval Research (DARPA) Contract No. N00014-02-1-0915, National Science Foundation (ITR/SYS) Contract No. IIS-0112991, National Science Foundation (ITR) Contract No. IIS-0209289, National Science Foundation-NIH (CRCNS) Contract No. EIA-0218693, National Science Foundation-NIH (CRCNS) Contract No. EIA-0218506, and National Institutes of Health (Conte) Contract No. 1 P20 MH66239-01A1.

Additional support was provided by: Central Research Institute of Electric Power Industry (CRIEPI), Daimler-Chrysler AG, Compaq/Digital Equipment Corporation, Eastman Kodak Company, Honda R&D Co., Ltd., Industrial Technology Research Institute (ITRI), Komatsu Ltd., Eugene McDermott Foundation, Merrill-Lynch, NEC Fund, Oxygen, Siemens Corporate Research, Inc., Sony, Sumitomo Metal Industries, and Toyota Motor Corporation.

1 Introduction

The question of which kind of measurements to use is fundamental to the design of any visual learning system, whether it is a supervised system or not. Since gray level features are sometimes limited in learning objects, e.g. faces [18], often other, more informative variables are used. Examples include wavelets [16], wavelet-like features [22] and statistics of pixels around interest points [7, 12].

These highly informative measurements can be exploited in the supervised case by feeding them directly into a classifier, as done in the above citations. Alternatively, one can compute the mutual information (or other scores) between the labels and single variables [21] or subsets of the variables [17]. These methods and other variable selection methods benefit from having a set of informative features. However, dimensionality reduction algorithms, such as PCA and FDA, do not, in general, benefit from using informative, uncorrelated features. A wavelet transform, for example, can be seen as a rotation in the input space [10], which no distance based algorithm (e.g., PCA, LDA, ISOMAP, LLE) benefits from.

On the other hand, while dimensionality reduction algorithms do well on sets of correlated features, variable selection methods perform poorly. They fail to pick relevant variables, because the score they assign to correlated features is too similar, and none of the variables is strongly preferred over another.

Hence, variable selection and dimensionality reduction algorithms have complementary advantages and disadvantages. Dimensionality reduction algorithms thrive on correlation between variables but fail to select informative features from a set of more complex features. Variable selection algorithms fail when all the variables are correlated but do well with informative variables. The scheme we propose is simple: first extract informative features from the data. Then, apply a variable selection algorithm. Last, apply dimensionality reduction to extract the most informative directions. The key point here is to use the same optimization function at all stages. This guarantees that the maximization of the dimensionality reduction stage is not undermined by the feature selection stage but instead benefits. When combining a random pick of a feature selection algorithm with a dimensionality reduction algorithm, the resulting optimization is not clear. We show in the experiments that in this case the results are considerably worse.

The combined optimization function searches for the variable selection that maximizes the information content of the data's most informative directions. We use a simple algebraic definition, which similar to principle component analysis (PCA), uses the variance of a direction as its information content score. Instead of strict (binary) variable selection, which makes our optimization an intractable enumeration problem, we suggest a variable weighting scheme.

2. Motivation

We find that the role of feature selection is not well understood, even by some very experienced researchers. For example, many people believe that SVM will always pick a hyperplane which uses only the relevant features because "it chooses the best hyperplane". A more sophisticated claim would say that the generalization ability of the SVM depends on the margin, which does not change with the addition of irrelevant features. These claims ignore, however, the dependence of the SVM performance on the radius of the data, which increases with the number of features. More background on variable selection in the supervised setting can be found in [3, 23].

Since there are fewer algorithms for feature selection in the unsupervised setting, this case is even less understood. In Fig. 1, we demonstrate the importance of feature selection when combined with PCA. The data was generated by two 3D Gaussians with random parameters. Additional 200 dimensions were generated in a similar manner. These dimensions were each permuted separately, to remove any class membership information in those dimensions[2].

As can be seen from Fig. 1, PCA does not perform well in the presence of the irrelevant features, i.e it fails to separate the clusters. An application of the simple Algo. 2 presented below solves the problem, and the good separation between the clusters is clear. This was done without using any label information.

The loss of rotation invariance. The algorithms we use in this work weight each feature separately. They are not invariant to the change of coordinates. This might seem troubling at first, as PCA and LDA are both *kernel methods* (methods that depend only on the similarity matrix (*kernel*) between every two data points) and enjoy invariance to unitary transformations of the data. However, in the context of feature selection, it can be shown that rotation invariance is not a desirable property.

For the supervised case, Ng [15] has shown that for any rotationally invariant classification algorithm, there exist at least one classification problem that is solvable by simply thresholding one variable, but which cannot be learned by this algorithm with a reasonably sized dataset: The required size will be linear in the total number of features, while some feature selection algorithms will need only the logarithm of it. In the unsupervised case, the situation is similar. If an algorithm is rotationally invariant, then all of the information except the data's dimension is contained in the distances between every two points (i.e, in the kernel matrix).

[2]Matlab: $M = [rmvrnd(rand(203,1)-0.5, diag(rand(203,1)+.5,50))', mvnrnd(rand(203,1) - 0.5, diag(rand(203,1) + .5,50))'];fori = 4:203,\ M(i,:) = M(i, randperm(100));$

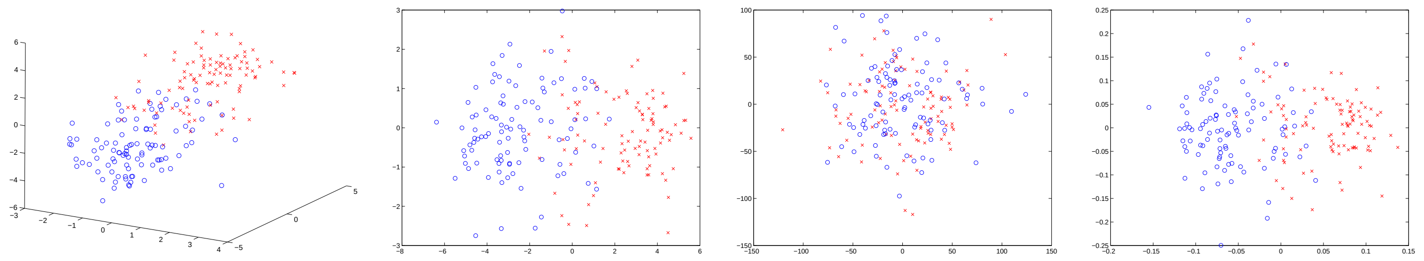

Figure 1: This figure demonstrates the importance of feature selection in the unsupervised setting, and that even though our algorithms and PCA use similar optimization functions, they are very different in their capabilities of dealing with irrelevant variables: (a) Three relevant dimensions out of 203. The rest of the dimensions are similar but were permuted to remove any class membership information; (b) The first 2 principle components of only the 3 relevant dimensions; (c) PCA of the whole 203 dimensions; (d) The results of applying PCA after weighting according to the α weights recovered by Algo. 1.

However, the kernel matrix is corrupted by the irrelevant features. In fact, for a linear kernel, where the similarity between every two points is just the dot product, the kernel matrix is the sum of the kernel matrix of the relevant features, and of that of the irrelevant features. Decomposing the kernel back to the two matrices is almost impossible when dealing with a large number of irrelevant features.

Using random matrix theory, we can try estimate the severity of the situation. Let our q data points be represented as the columns of the matrix $M = [M1^\top, M2^\top]^\top$, where $M1$ is an $n_1 \times q$ matrix containing the n_1 relevant variables of our data, and $M2$ is an $n_2 \times q$ matrix containing the irrelevant data. Assume further that every feature (row of the matrix M) is normalized to have a variance of 1. The kernel matrix $A = M^\top M$, is the sum of $A1 = M_1^\top M_1$ and $A_2 = M_2^\top M_2$ (throughout this work A will denote such a kernel matrix of the form $A = M^\top M$, and B will denote its dual – the covariance matrix $B = MM^\top$. In many cases we will use the fact that they share the same eigenvalues to switch between the two).

In what is the ideal situation for most spectral clustering algorithms, the relevant data is composed out of two equally sized very tight clusters. These clusters would be centered around two points positioned as far as possible from each other in $\Re^{n_1}$, e.g, one cluster might contain $q/2$ points equal to $[1, 1, ..., 1]^\top/sqrt(q)$, and the other cluster might contain $q/2$ copies of the point $[-1, -1, ..., -1]^\top/sqrt(q)$. The kernel matrix of the relevant data A_1 will be a rank 1 matrix with an eigenvalue n_1.

Assume now that in our ideal situation, the irrelevant variables (the elements of the matrix M_2) are i.i.d zero mean Gaussians with variance $1/q$ (so the variance of each row has an expectation of 1). The matrix A_2 here is called a *Wishart matrix*, and its first eigenvalue distributes with a mean of $n_2/q^2(\sqrt{n_2} + \sqrt{q})^2$ [8].

Let v_1 be the only eigenvector of A_1 with a positive eigenvalue. Rotating M_2 by multiplying it from the right with a $q \times q$ rotation matrix R, would not change its distribution (follows from the basic properties of Gaussians). Hence $v_1^\top A_2 v_1$ has the same distribution as $e_1^\top A_2 e_1$, e_1 being a vector of 1 followed by $q - 1$ zeros. Thus, $v_1^\top A_2 v_1$ has the same distribution as the top left element of A_2. This element is just the square norm of a vector of n_2 random Gaussians with mean zero and variance $1/q$, and distributes according to a Chi-squared distribution around a mean of n_2/q^2.

Since $v_1^\top A v_1 = v_1^\top A_1 v_1 + v_1^\top A_2 v_1$, it distributes around a mean of $n_1 + n_2/q^2$. Similarly, let u_1 be the first eigenvector of A_2. Then $u_1^\top A u_1 \geq u_1^\top A_2 u_1$, which concentrates around $n_2/q^2(\sqrt{n} + \sqrt{q})^2$. When $n_1 + n_2/q^2 << n_2/q^2(\sqrt{n_2} + \sqrt{q})^2$, v_1 is not distinguishable from u_1 and any unsupervised kernel method would fail.

The situation is worse in less ideal cases, where the real data is noisy, and the irrelevant data has a non-random structure. When the number of irrelevant features grows, rotationally invariant algorithms will fail since they only see the matrix A. Non rotationally invariant algorithms can identify a group of features that lead to a decomposition $A = A_1 + A_2$ such that A_1 **is not** likely to be a random matrix and A_2 **is** likely to be a random matrix.

3. Weighting Variables for Unsupervised Dimensionality Reduction

Given a dataset where each example is a vector in $\Re^n$, our goal is to weigh the variables such that the most informative directions in the reweighed data contain as much information as possible. Similarly to the PCA approach, we use the variance along this direction as a score for its information content. We combine these scores for the dataset's k most informative directions to obtain the information content of the whole dataset.

Let M be our data matrix with n rows, each representing a variable, and q data samples as columns. We then weigh each variable i by a positive weight $\sqrt{\alpha}_i$ [3]. Let $D = diag(\sqrt{\alpha})$ be the diagonal weighting matrix and let $\tilde{M} = DM$ be our reweighed data matrix.

A projection of the weighted data onto a direction vector w (a vector on the n-dim sphere) produces a new variable $v = w^T \tilde{M}$ with variance $\sum_{i=1}^{k}(v(i) - \bar{v})^2$. Since we are only concerned with the variances, we can subtract the average of each variable from the data matrix and assume that the mean of each row in M is 0. Therefore, the means of each row in $\tilde{M}$ are also zero, as well as the mean of v. Hence, the variance of the new variable v is given by the simple dot product $v^T v$.

We wish to find a norm 1 weight vector α and k orthonormal directions $W = [w_1|w_2|w_3|...|w_k]$ which maximize the score: $||(var(v_1), var(v_2), ..., var(v_k))||_2$, where $v_i = w_i^T \tilde{M}$, $i = 1..k$ are the projections of the weighted data onto those directions.

This is equivalent to maximizing $\sum_{i=1}^{k}(var(w_i^T DM))^2 = \sum_{i=1}^{k}(w_i^T DMM^T Dw_i)^2 = = \sum_{i=1}^{k} w_i^T(DMM^T D)(DMM^T D)w_i$. Setting the matrix $B_\alpha = DMM^T D$, this amounts to maximizing trace$(W^T B_\alpha B_\alpha W) = ||W^T B_\alpha||_F^2$.

This trace is maximized for any fixed α when W is just the first k columns of U, and Udiag$([\sigma_1, \sigma_2, ..., \sigma_n])U$ is the SVD of the matrix B_α [4][9] . The values of the maximization function would in that case be: $\sum_{i=1}^{k} \sigma_i^2$.

The singular values of the positive definite matrix $B_\alpha = \tilde{M}\tilde{M}^T$ are just its eigenvalues. These eigenvalues are the squares of the singular values of the matrix $\tilde{M}$, which are also the square roots of the eigenvalues of $A_\alpha = \tilde{M}^T\tilde{M} = M^T diag(\alpha)M$.

This relation between the right and left singular values of the matrix $\tilde{M}$ is similar to the use of the kernel trick for many algorithms. The pairs of matrices A_α/B_α and Q/W are pairs of duals for this optimization problem. By this duality we can maximize the expressions involving B_α without having to deal with the square roots in the matrix D.

From the above discussion, the maximization of trace$(W^T B_\alpha B_\alpha W)$ subject to W being orthonormal and α having a norm of 1, is equivalent to the maximization of trace$(Q^T A_\alpha A_\alpha Q)$ for an orthonormal Q and the same α. An algorithm for this optimization problem was suggested in [24], where the problem of finding a set of features which is optimal for clustering into k clusters was considered. This algorithm is termed the $Q - \alpha$ algorithm, and is embedded in step two of the algorithm below.

Algorithm 1 (Weighting for PCA) *Let N be an $n \times q$ input matrix. Perform the following steps:*

```
1. Center the data to have a mean of $0$ and normalize it to have
   a norm of $1$ in each variable. Let $M$ be the normalized data
   matrix, with rows $\mathbf{m}_1^\top, ..., \mathbf{m}_n^\top$.
2. Let $Q^{(0)}$ be some orthonormal $q \times k$ matrix. Perform the
   following steps with iterations index $r = 1, 2, ...$:
     (a) Let $G^{(r)}$ be a matrix whose $(i, j)$ components are
         $(\mathbf{m}_i^\top \mathbf{m}_j)\mathbf{m}_i^\top Q^{(r-1)} {Q^{(r-1)}}^\top \mathbf{m}_j$.
     (b) Let $\alpha^{(r)}$ be the leading eigenvector of $G^{(r)}$.
     (c) Let $A^{(r)} = \sum_{i=1}^{n} \alpha_i^{(r)} \mathbf{m}_i \mathbf{m}_i^\top$.
     (d) Let $Q^{(r)}$ be the first $k$ eigenvectors of $A_\alpha$.
     (e) Increment index $r$ and go to step a.
3. Reweight the data $\tilde{M} = diag(\sqrt{\alpha})M$.
4. Compute the Principle Components $W$ of the matrix $\tilde{M}$.
```

From the discussion in [24], the resulting weight vector α would be sparse and composed out of positive elements. Note that although our goal is somewhat different than the one suggested there (increasing information content, and not supporting k good clusters), we get a similar algorithm. The main difference is the centering of the data, which seems to have a large significance in practice.

3.1. A parameter free algorithm

Algo. 1 above is an iterative algorithm which converges to a local maxima. This iterative nature allows it to search for a subset of features among a large amount of irrelevant data. However, sometimes we might prefer a non-iterative algorithm which is guaranteed to return the same solution in every run. Also, in many situations we are willing to sacrifice some accuracy of the returned weights for the ability to work faster with many more variables. The algorithm presented next is an approximation of Algo. 1 which archives both goals, and is also parameter free.

The basic idea behind the approximation algorithm is simple. When we optimize for second order expressions like $\sum_{i=1}^{k}(var(w_i^T DM))^2$, those components with the largest values are going to dominate the expression. For that reason, the result is not going to change much whether we are optimizing the first k most informative directions, or any $l > k$ most informative directions. The score we maximize is probably going to remain high for successful weighting and low for unsuccessful ones. Moreover, most of the time we do not have any information about the correct parameter k in advance, and we might want to increase the information content in all possible directions.

Based on these intuitions we explore the case where $k = min(q, n)$. Since the trace of a matrix is the sum

[3]The representation of the vector α as the vector of the element wise squares of the actual weights ($\sqrt{\alpha}$) is done in order to make the resulting optimization problem bilinear.

[4]W can be any other orthonormal basis of this k-dim subspace as well.

of its eigenvalues, the optimization problem of maximizing trace$(Q^T A_\alpha A_\alpha Q)$ becomes maximizing trace$(A_\alpha A_\alpha)$ = trace$(M^T \text{diag}(\alpha) M M^T \text{diag}(\alpha) M)$. This expression is bilinear in α and can be written as maximizing $\alpha^T H \alpha$, where H_{ij} = trace$(m_i m_i^T m_j m_j^T)$ = $(m_i^T m_j)$trace$(m_i m_j^T)$ = $(m_i^T m_j)^2$.

Algorithm 2 (Parameter Free Weighting for PCA) *Let M be an $n \times q$ normalized and centered input data matrix. Perform the following steps*

1. *Let $H = (MM^T).^2$, where the $.^2$ means taking the square of each element separately.*
2. *Let α be the leading eigenvector of H.*
3. *Re-weight the data and apply PCA on $\tilde{M} = diag(\sqrt{\alpha})M$*

We implement both steps of the algorithm in a parallel scheme. In each stage, only a part of the matrix H is loaded into memory. This enables us to handle up to 10^5 features and hundreds of data points, while still maintaining a reasonable running time.

The results of Algo. 2 are less sharp than those of Algo. 1. The ratio of the average weights between the relevant and irrelevant features is lower with parameter free algorithm. An example is shown in Fig. 2. In this experiment, the first 3 variables are chosen from 4 multivariate normal distributions with diagonal covariance matrices. The remainder of the 200 variables are selected from the same distributions but are each permuted independently. In this way, the remaining 200 features give no information about the underlying cluster from which the data points stem. As it follows, the assignment of weights by taking the first PCA of the data is uninformative for feature selection purposes. Algo. 1 assigns very high weights to the relevant variables, and much less to the rest. Algo. 2 successfully detects the relevant variables, but assigns somewhat higher weights to the irrelevant variables.

4. Supervised variable selection for FDA

We next describe how to handle the supervised case. In [24] a different solution is suggested using a kernel as close as possible to the ideal block diagonal kernel. Here we take a different approach, viewing the matrix A_α as the kernel for a Kernel Fisher's Discriminant (KFD) analysis [14]. The kernel case of Fisher Discriminant Analysis (FDA) was treated in the past for two class classification and for regression. We extend it to the multi-class case, then match it with the appropriate variable selection. Due to space constraints, the derivations are moved to the appendix.

Algorithm 3 (Supervised Variable Selection) *Let M be an $n \times q$ input data matrix containing samples from l classes. Let $M^{(i)}$ be an $n \times q_i$ matrix containing only those samples taken from class i. Perform the following steps*

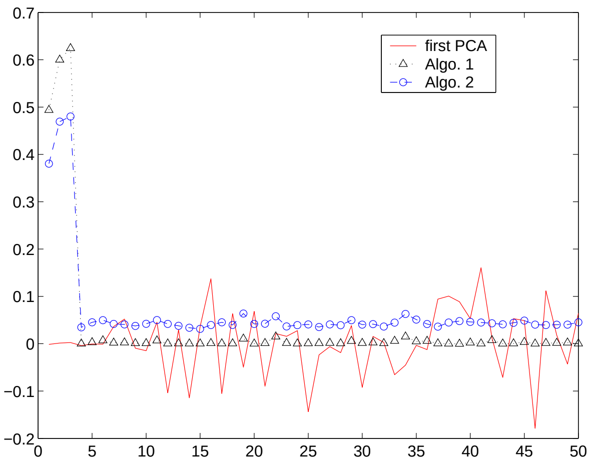

Figure 2: Comparison of on a synthetic dataset where the first 3 variable are relevant. Only the first 50 variables are shown. Algo. 1 produces sharper results than Algo. 2. The first PCA does not detect the relevant variables.

1. *Perform an eigen-decomposition of the matrix $A = M^T M = EDE^T$*
2. *Compute the means $\mu = \frac{1}{q} M 1_q$, $\mu^{(i)} = \frac{1}{q_i} M^{(i)} 1_{q_i}$*
3. *Let $H = \sum_{i=1}^{l} q_i((MED^{-1}E^T M^T). * (\mu\mu^T - \mu^{(i)}\mu^T - \mu\mu^{(i)T} + \mu\mu^T))$, where the $.*$ means multiplying each element separately*
4. *Let $G = \sum_{i=1}^{l} q_i((MED^{-1}E^T M^T). * (M^{(i)} M^{(i)T} - q_i \mu^{(i)} \mu^{(i)T}))$*
5. *Let α be the leading eigenvector of $H - \lambda G$*

5. Analysis

The choice of the optimization criteria. The optimization criteria used for the weighting is based on a simple algebraic definition of an informative score of a data matrix. It gives rise to a simple quadratic form which is easy to solve. Described using the eigenvalues $\sigma_1, \sigma_2, \ldots$ of the matrix B_α, the optimization criteria simply becomes $\sum_{i=1}^{k} \sigma_i^2$. This utility function is closely related to the square loss function in the supervised case. In general, we can try to optimize similar utility functions $\sum_{i=1}^{k} f(\sigma_i)$, where f is any (monotonic) function.

For simplicity, we are going to refer in the discussion below only to the case where $k = min(q, n)$ (the case of Algo. 2). If F is the matrix version of the function f then it is easy to show that trace$(F(B)) = \sum_i f(\sigma_i)$. The resulting optimization problem might be hard to solve directly. However, it can be approximated using the Taylor series of the function f up to any order. If there are not too many variables, one can use quadratic programming in order to solve these approximations.

Why not use $f(x) = x$ **?** There is a special case where the function f is the identity function. This choice of f would result in a weighting scheme that simply tries to maximize the trace of the kernel matrix A_α. The problem would transform into max $\alpha^T u$ where each element of the vector u is just u_i = trace$(m_i m_i^T) = m_i^T m_i$. The maximum is obtained when $\alpha = \frac{u}{\sqrt{u^T u}}$.

However, kernels with a large trace (such as those returned by the previous optimization criteria) are bad for generalization [4] (this is not obvious without reading the reference. As an intuition: for kernels with large elements along the diagonal, each point is separated from all other points). Since generalization is crucial for unsupervised feature selection, just as it is in the supervised case, we would like to achieve good clustering, but we would not like to use a kernel A_α that supports any random clustering.

Note that in both Algo. 1 and Algo. 2, the trace of the kernel (A_α) is controlled by the constraint on α. The trace of the kernel is the same as the trace of the covariance matrix B_α. The covariance matrix is a sum of rank-1 matrices, each with a trace of 1 (all features are normalized to have a norm of 1) but weighted by a corresponding weight α_i^2. Since the trace operator is linear, the trace of the covariance matrix is the same as the norm of the vector α, which is constrained to be 1.

Why use $f(x) = x^2$ **?** Recall from Sec. 2 that the main clue to help us identify whether a kernel matrix A is good is that A should not seem random. Hence, we can turn to random matrix theory to help us define a good kernel. The *Gaussian Ensemble* (GE) is probably the most basic symmetric random matrix. It is defined as a family of real symmetric $n \times n$ matrices A such that the upper triangular elements are independent Gaussians with zero-mean and variance 1 along the diagonal, and variance $1/2$ elsewhere (Matlab: `A = randn(n); A = (A+A')/2;`). Given a matrix A we can verify its probability to be in this family simply by computing the product of all the $(n^2 + n)/2$ independent Gaussians, i.e., $P(A \in GE) \sim \Pi_{i<j} e^{-A_{ij}/4} \Pi_i e^{-A_{ii}/2} = \Pi_{i,j} e^{-A_{ij}/2} = e^{||A||_F^2/2}$. Hence, the probability of a matrix A being random is proportional to the exponent of the square of its Frobenius norm. Recall that $||A||_F^2$ = trace$(AA^\top)$, which is the sum of the eigenvalues of $AA^\top$, which for a symmetric matrix, is the sum of the squares of its eigenvalues. Hence, a matrix with a high sum of squared eigenvalues is not likely to be a random GE matrix. This gives a direct motivation to the use of this score. In order to handle the $k < n$ case of Algo. 1, we add the assumption, common to spectral techniques, that the information is contained only in the matrix elements which are spanned by the first few eigenvectors of it. Hence, if Q is the $n \times k$ matrix holding the eigenvectors of A, and recall that $QQ^\top$ is the projection matrix to the linear subspace spanned by the columns of Q, then if $QQ^\top A$ has a high score it is unlikely to be random. In this case, the score is only approximated, since the elements of $QQ^\top A$ are not independent, even for a random matrix.

6. Experiments

Face Recognition. We first applied our methods to perform face recognition. We used two publicly available datasets: the YALE dataset [25] and the AR dataset [13]. The YALE dataset contains 15 different persons, each one photographed 11 times under different illumination, under different expression and with or without glasses. For the AR dataset we used only the 50 males, each having 14 images. For both datasets we created 20 instances of similar experiments, where 3 images per person were picked randomly to be the training set for that person. The rest served as test set.

We compared several algorithms: The eigenface method [20] which uses PCA to reduce the dimension of the face images; the fisherface method [2] which applies multi-class Fisher discriminant analysis to face images after they have been reduced in dimension using PCA; Both methods after applying conventional feature selection algorithms; an application of eigenfaces after variable weighting using Algo. 2; and application of fisherfaces after variable weighting using Algo. 2. All methods were tested twice: for gray level images and for wavelets. As expected, Algo. 2 and all other 3 feature selection algorithms did not work well when dealing with gray level values directly, and those results are omitted from the table. When applying eigenfaces and fisherfaces directly to the data without any feature selection, gray level performed better than haar wavelets.

We also compared the results with similar variants using Algo. 1 and 3, which for these datasets did not perform better than Algo. 2. This was probably due to the large number of classes (person identities) used in each experiment. All of these are omitted from the table below.

The values in each cell in Table 1 are the average recognition rate (in percent) over 20 runs, and the standard deviation of the results. Results are shown for all methods for the PCA dimension which gave the best results. For feature selection (FS) we show the best out of three *supervised* methods (Pearson coefficients, Fisher criterion score, and the Kolmogorov-Smirnov test) and for the best percentile of kept features. It turns out that for these datasets the Fisher criterion score did the best out of the three, and this performance occurred when keeping 40-50% of the features.

As shown in Table 1, the use of wavelets together with Algo. 2 improves the results both for eigenfaces and for fisherfaces. For the YALE dataset, running Fisher using Algo. 2 with weighting of wavelet features performed better than the Fisher results on gray levels for 75% of the trials,

DataSet	PCA	PCA + Haar	FS + PCA	Algo. 2 +PCA	Fisher	Fisher + Haar	FS + Fisher	Algo. 2 +Fisher
Yale	70.0 $\pm$ 2.5	64.5 $\pm$ 2.2	67.6 $\pm$ 4.9	**81.2** $\pm$ 2.4	83.4 $\pm$ 3.2	81.8 $\pm$ 3.1	83.4 $\pm$ 2.7	**88.2** $\pm$ 2.4
AR	62.9 $\pm$ 2.4	32.2 $\pm$ 1.6	62.9 $\pm$ 4.4	**70.6** $\pm$ 1.8	91.9 $\pm$ 1.3	90.4 $\pm$ 1.8	90.9 $\pm$ 1.6	**94.6** $\pm$ 0.8

Table 1: Face recognition precision on two databases, Yale and AR. Results indicate percentage of test examples correctly identified plus or minus the standard deviation across 20 independent trials. In each trial 3 images per person served as training images, and the rest as testing. Results are recorded for seven algorithms: PCA is an application of eigenfaces; PCA + Haar is an application of PCA on Haar wavelet; FS + PCA is the best filtering method out of Pearson coefficient, Fisher score, and KS test applied to wavelet coefficients followed by eignefaces; Algo. 2 + PCA is an application of eigenfaces after re-weighting the haar wavelet data according to Algo. 2. The rest of the columns report results for similar experiments but with Fisherfaces instead of eigenfaces.

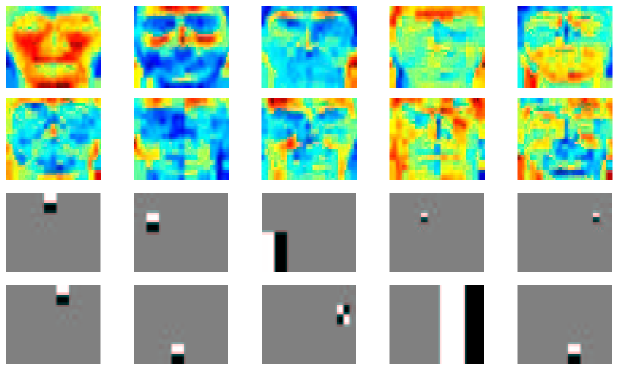

Figure 3: Top two rows show the first 10 principle components of the wavelets weighted by Algo. 2 on the YALE dataset. Bottom rows show the first 10 wavelet features picked by Algo. 2 on the same experiment.

and performed the same for 15%. For the AR dataset, the method using wavelets+Algo. 2+Fisher gave the best performance for all trials.

Figure 3 shows the first principle components of the wavelets for a specific YALE dataset experiment. The wavelet features have been weighted according to Algo. 2 and then the principle components were computed. Also shown are the wavelets chosen by this algorithm. The algorithm tends to choose large/medium sized wavelets.

Car Detection. We applied Algo. 2 to a car detection in static images task. We used the car images database from UIUC [1]. The goal is to locate the cars accurately in a number of large grayscale images. The test images vary in size, and may contain multiple cars. The training data consists of 500 car images and 550 non-car images.

We compared four sets of features: the gray level values, Haar wavelets, Haar wavelets weighted by Algo. 2, and Haar wavelets weighted by Algo. 3. We transformed each training image to the feature space, and learned a template using a linear Support Vector Machine [5]. Using the fact that the transformation to the wavelet feature space was linear, as well as the variable weighting, we were able to transform the learned template in features space back into an image, as shown in Fig. 4. We searched for this template using normalized cross correlation in the test images, and then used the same neighborhood suppression as described in [1], yielding a set of car detections weighted by strength of correlation. When compared against the ground truth locations included alongside the test data, one is able to construct a Precision-Recall (PR) curve.

[5]For Algo. 2 we also tried LDA, and received almost identical results

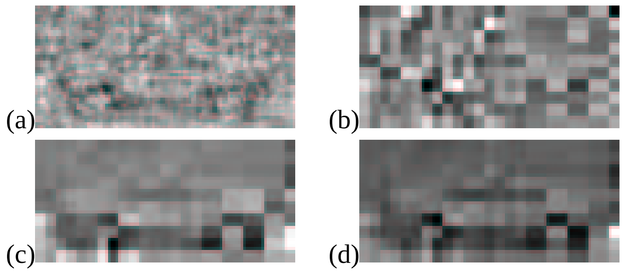

Figure 4: Template for car detection learned via SVM from four different feature sets. (a) histogram equalized gray levels (b) Haar wavelets (c) Algo. 2 on Haar wavelets (d) Algo. 3 on Haar wavelets.

The PR curves we calculated are presented in Fig. 5. As shown in the graph, SVM on gray levels performs at about the same level as the component based detector presented in [1]. SVM on wavelet features does worse and SVM on Algo. 2 weighted wavelet features does better. SVM on Algo. 3 weighted wavelet features does better than all other methods. Weights returned by Algo. 3 were extremely sparse, with over 90% of the total magnitude of the weights assigned to only 25 of the wavelets. Thus Algo. 3 combined with SVM enables the control of the run-time of the learned classifier much like Boosting does.

Place recognition. For this experiment we used the data collected by Torralba et. al.[19], in order to compare supervised learning algorithms. Given an image, the goal is to classify it into one of ten categories: conference-room, corridor, elevator lobby, inside elevator, kitchen, lab, office, open area, plaza and street. The data consists of 50,757 frames collected over 17 sequences using a wearable camera. While this method achieves good identification results

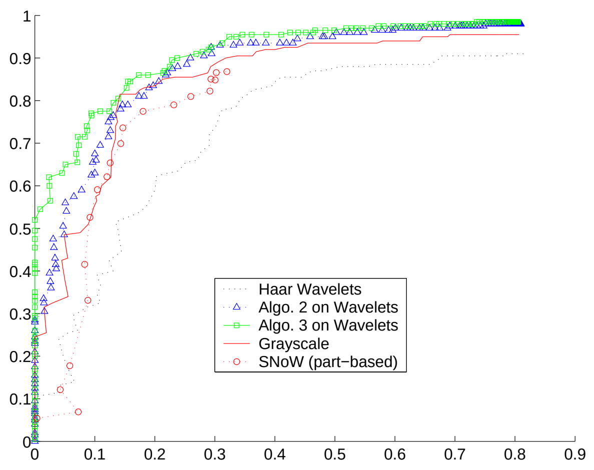

Figure 5: Precision-Recall curve for the UIUC car dataset. SVM templates are learned using gray levels, Haar wavelets, Algo. 2 weighted wavelets and Algo. 3 weighted wavelets.

by using temporal information, the task of identifying a single frame is quite difficult. For example, it is not easy to distinguish a lab from an office or a conference room. Each frame in the dataset was represented by a vector of 384 dimensions, consisting of the output of steerable filters applied to the input 120×160 image at several scales. This representation, together with the frame annotation, was made publicly available by the authors of [19].

We conducted repeated one-vs-all experiments. In each experiment 100 random examples of one place catagory served as the set of positive training examples, and 100 random examples from the rest of the frames served as negative examples. The results were then tested on a much larger set of testing examples, which contained equal numbers of positive and negative examples. Each such one-vs-all experiment was repeated three times.

In [19] the authors used the gentle boost algorithm on top of PCA. In our experiments, SVM always outperfoms the boosting algorithm. Moreover, in all of our experiments, PCA did not help at all (not even for boosting), hence we used the original 384 dimensional data. The results we obtained are 26.67% error for an 80 dimensional PCA followed by a linear SVM, 25.10% for linear SVM, and 23.17% for Algo. 2 weighting of the features followed by an SVM. The results improve to 22.83% error when using Algo. 3 weighting and then SVM.

7. Conclusions

We presented variable weighting algorithms that were designed to preprocess the data before applying dimensionality reduction algorithms. In their parameter-free form (Algo. 2, and Algo. 3), they are very simple to implement, and can be applied to tens of thousands of features. We show that using these algorithms we can effectively apply dimensionality reduction algorithms to datasets of uncorrelated variables, benefiting from both worlds of dimensionality reduction and feature selection.

References

[1] S. Agarwal and D. Roth. Learning a Sparse Representation for Object Detection. In *ECCV*, 2002.

[2] P.N. Belhumeur, J.P. Hespanha and D.J. Kreigman. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *PAMI* 19(7), 1992.

[3] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *AI*, 97(1-2), 1997.

[4] O. Bousquet, D.J.L. Herrmann. On the Complexity of Learning the Kernel Matrix. In *NIPS*, 2003.

[5] F.R.K. Chung. *Spectral Graph Theory*. AMS, 1998.

[6] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, NY, 1972.

[7] R. Fergus, P. Perona and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *CVPR*, 2003.

[8] S. Geman. A limit theorem for the norm of random matrices. *Ann. Probab 8*, 1980.

[9] G. Golub and C. Van Loan. *Matrix computations*, 1989.

[10] T. Hastie, R. Tibshirani and J. Friedman. *The Elements of Statistical Learning* Springer, 2001.

[11] T. Jebara. Convex Invariance Learning. In *AISTAT*, 2003.

[12] D.G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60, 2 (2004), pp. 91-110.

[13] A.M. Martinez and R. Benavente. The AR face database. Tech. Rep. 24, CVC, 1998.

[14] S. Mika, G. Ratsch, J. Weston, B. Schoelkopf and K.R. Muller. Fisher Discriminant Analysis with Kernels. Neural Networks for Signal Processing IX, 1999

[15] A.Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. *ICML*, 2004.

[16] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna and T. Poggio. Pedestrian Detection Using Wavelet Templates. In *CVPR 1997*.

[17] H. Schneiderman. Learning Statistical Structure for Object Detection. *CAIP,* 2003.

[18] K.K. Sung and T. Poggio. Finding Human Faces with a Gaussian Mixture Distribution-based Face Model. *ACCV* 1995.

[19] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin. Context-based vision system for place and object recognition, In *ICCV,* 2003.

[20] M. Turk and Pentland. Face Recognition Using Eigenfaces. In *ICPR,* 1991.

[21] S. Ullman, M. Vidal-Naquet and E. Sali Visual features of intermediate complexity and their use in classification. *Nature Neuroscience, 2002*

[22] P. Viola, and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR,* 2001.

[23] Weston, J., S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio and V. Vapnik. Feature Selection for SVMs. *NIPS,* 2001.

[24] L. Wolf and A. Shashua. Direct feature selection with implicit inference. In *ICCV,* 2003

[25] Yale University Face Database. Available at http://cvc.yale.edu/projects/yalefaces/yalefaces.html

A. Derivation of multi-class kernel FDA, and Alg. 3

Consider q data points $M_1, M_2, ..., M_q$ with mean μ which are assigned to l classes. Each class C_i contains q_i examples with mean μ_i. The FDA ([6]) examines orthogonal directions which maximize the inter-class variance, while minimizing each inner-class variance. This is done by maximizing the ratio $\frac{w^T S_B w}{w^T S_W w}$, where $S_B = \sum_{i=1}^{l} q_i(\mu_i - \mu)(\mu_i - \mu)^T$ is the between class scatter matrix, and $S_W = \sum_{i=1}^{l}\sum_{N \in C_i}(N-\mu_i)(N-\mu_i)^T$ is the within class scatter matrix. This maximization problem is solved by considering the largest generalized eigenvectors of the pair (S_B, S_W). It is not difficult to show that each of the directions w returned by the FDA is a linear combination $w = \sum_{i=1}^{n}\gamma_i M_i$.

Let $M^{(i)}$, $i = 1..l$ be the matrix composed only of measurements arising from class C_i. Let $A = M^T M$, $A^{(i)} = M^T M^{(i)^T}$ ($q \times q_i$ matrix), and let 1_p be a vector of p ones. Substituting the expression for w as a linear combination $w = M\gamma$ in the ratio to be maximized, we get: $\frac{w^T S_B w}{w^T S_W w} = \frac{\gamma^T R \gamma}{\gamma^T S \gamma}$ where $R = \sum_{i=1}^{l} q_i(\frac{1}{q_i}A^{(i)}1_{q_i} - \frac{1}{q}A1_q)(\frac{1}{q_i}A^{(i)}1_{q_i} - \frac{1}{q}A1_q)^T$ and $S = \sum_{i=1}^{l} q_i A^{(i)}(I_{q_i} - \frac{1}{q_i}1_{q_i}1_{q_i}^T)A^{(i)T}$.

This derivation is similar to [14], except that we handle multiple classes, and therefore need to extract more than one direction w. Let these directions be noted as $w^{(i)}$, $i = 1..l$. Note that there is no one-to-one correspondence between the directions and the classes. Similar to FDA, we constrain every two extracted directions $w^{(i)}$, $w^{(j)}$ by $w^{(i)T}w^{(j)} = \gamma^{(i)T}A\gamma^{(j)} = \delta_{ij}$. To make the form of the constraint simpler, we transform our coordinate system. Let $A = EDE^T$ be an eigenvalue decomposition of A, containing only eigenvectors with non-zero eigenvalues. Define $\phi^{(i)} = D^{1/2}E^T\gamma^{(i)}$. The matrices S and R are spanned by the column space of the matrix A. Therefore, out of all possible solutions to this linear equation $\phi^{(i)} = D^{1/2}E^T\gamma^{(i)}$, the one which maximizes our criteria $\frac{\gamma^T R\gamma}{\gamma^T S\gamma}$ is given by $\gamma^{(i)} = ED^{-1/2}\phi^{(i)}$. We therefore maximize $\frac{\phi^T D^{-1/2}E^T RED^{-1/2}\phi}{\phi^T D^{-1/2}E^T SED^{-1/2}\phi}$ subject to the directions ϕ being orthogonal, i.e $\phi^{(i)T}\phi^{(j)} = \delta_{ij}$

Variable selection is done by replacing the kernel matrix A with the matrix $A_\alpha = \sum_{i=1}^{n}\alpha_i \mathbf{m}_i\mathbf{m}_i^\top$, and the matrices $A^{(k)}$ with the matrices $A_\alpha^{(k)} = \sum_{i=1}^{n}\alpha_i\mathbf{m}_i\mathbf{m}_i^{(k)\top}$, where $\mathbf{m}_i^{(k)}$ is a vector containing the values of the i^{th} variable, but only for the points that belong to class C_k. With this replacement, based on the formulas for S and R, both the numerator and the denominator are linear in α. However the ratio is not. We therefore investigate another way to maximize the numerator and minimize the denominator. We maximize: $(\phi^T D^{-1/2}E^T RED^{-1/2}\phi) - \lambda(\phi^T D^{-1/2}E^T SED^{-1/2}\phi)$. Note that an optimization function which is similar in spirit was used by Jebara [11] in the context of invariance learning.

The optimization problem becomes bi-linear with respect to α, and variable selection algorithms similar to the ones above can be derived. For simplicity, we just show the analog of Algo. 2, created by maximizing $trace((\phi^T D^{-1/2}E^T RED^{-1/2}\phi) - \lambda(\phi^T D^{-1/2}E^T SED^{-1/2}\phi))$ over α. Algo. 3 above is simply achieved by rearranging the terms of the maximization problem, such that the bilinear nature is apparent.