Comput Manag Sci (2009) 6:25–40
DOI 10.1007/s10287-008-0070-7

ORIGINAL PAPER

Feature selection for high-dimensional data

Augusto Destrero · Sofia Mosci · Christine De Mol · Alessandro Verri · Francesca Odone

Published online: 18 April 2008

Abstract This paper focuses on feature selection for problems dealing with high-dimensional data. We discuss the benefits of adopting a regularized approach with L_1 or L_1–L_2 penalties in two different applications—microarray data analysis in computational biology and object detection in computer vision. We describe general algorithmic aspects as well as architecture issues specific to the two domains. The very promising results obtained show how the proposed approach can be useful in quite different fields of application.

Keywords Feature selection · Regularized methods · L_1–L_2 penalties · Iterative solutions · Computational biology · Computer vision

1 Introduction

Feature selection constitutes one of the central aspects of the learning process and in the last few years this question has been studied in depth, as demonstrated by the increasing amount of publications on this topic. In particular, feature selection is crucial for applications where the number of data available is small with respect to the input space size. Examples of applications that may lead to very high-dimensional data include signal processing in general (in particular audio and image processing), data

A. Destrero (✉) · S. Mosci · A. Verri · F. Odone
DISI, Università di Genova, Via Dodecaneso 35, 16146 Genoa, Italy
e-mail: destrero@disi.unige.it

S. Mosci
DIFI, Università di Genova, Via Dodecaneso 33, 16146 Genoa, Italy

C. De Mol
Department of Mathematics and ECARES, Université Libre de Bruxelles, Campus Plaine CP217, Bd du Triomphe, 1050 Brussels, Belgium

mining and multimedia data retrieval, biology, and computer graphics. In these cases it is particularly important to extract meaningful features, discarding noise or redundant information. Obtaining a small set of useful data from high-dimensional descriptions is important for a number of reasons, firstly to cope with the curse of dimensionality; secondly, to obtain compact descriptions that allow for faster computation; thirdly to obtain hints on the underlying processes producing a given output.

In this paper, after a brief account on the state of the art on feature selection methods, we describe a regularized approach to feature selection that looks for the solution to least squares problems with sparsity enforcing constraints. We then discuss how this approach has been applied with success to two different applications: biological computing and computer vision. We describe the specificities of the examined problems and show how the general algorithm has been adapted to address such specificities.

The algorithm which we discuss and apply to two different scenarios combines the so-called elastic net strategy (Zou and Hastie 2005) and an iterative soft-thresholding algorithm (Daubechies et al. 2004). It has been first introduced in De Mol et al. (2007). Its general form leads to a nonlinear iterative algorithm the solution of which minimizes the objective function defined as the least-squares residual to which both L_1 and L_2 penalties are added. The L_2-norm helps the classifier to select all elements belonging to groups of correlated features, while the L_1-norm enforces sparsity.

From the experimental viewpoint, the first application that we analyse consists in extracting sets of meaningful features from gene expression data. Biological data are high-dimensional data, but usually very few examples (in the order of tens or hundreds) are available. In this case feature selection is particularly important, as it can lead, not only to a more effective description of the data, but also to an explicit account of the feautures relevant to the biological process. In this case a penalty term combining L_1 and L_2 type penalties is used, where L_1 enforces sparsity, while L_2 preserves correlation (De Mol et al. 2007). The second application is a classical computer vision problem: object detection in digital images. In this case the size of training data available can be relatively high, but the input space tends to be much higher, in particular when overcomplete sets of features are available. In this context the main issue is to limit the amount of redundancy derived from images autocorrelation properties. Enforcing sparsity is a way to optimize the detection process at run time. It will be shown that dealing with correlation issues is not of interest in such a case, and therfore we will apply a pure L_1 penalty.

1.1 State-of-the-art on feature selection

Feature selection is a problem of formidable complexity. Recently many learning techniques have been proposed to solve the problem of feature selection. It is certainly worth mentioning a number of trainable methods that have emerged empirically for their effectiveness (Tibshirani 1996; Weston et al. 2003; Viola and Jones 2004; Zhu et al. 2004).

A survey of the topic can be found in Guyon and Elisseeff (2003). This survey offers a taxonomy of the methods proposed so far, that we briefly describe in the following.

Filters represent a preprocessing step completely disconnected from the learning phase; some examples are ranking criteria (Forman 2003; Weston et al. 2003) where correlation, mutual information or performance of single variable classifiers are used to assign a score to each feature. Instead, algorithms where the relevance of a feature subset is determined according to prediction performance of a learning machine are known in the literature as *wrappers* methods (Kohavi and John 1997). Notice that in this case the learning machine is used as a black box which, given in input a feature subset, returns a score according to its prediction performance. Differently from wrappers and filters, where variable selection and training are two separate processes, *embedded* methods are particularly interesting because they incorporate feature selection as part of the training phase. Examples are decision trees (Breiman et al. 1984) or more recent methods using sparsity-enforcing penalties. When such penalty is defined as the L_1-norm, and the error is the residual sum of squares, an algorithm known as LASSO (Least Absolute Shrinkage and Selection Operator) has been proposed in Tibshirani (1996) to find the minimizer of the corresponding objective function. Alternative algorithms for solving this problem have also been proposed in the recent literature.

1.2 Outline of the paper

The structure of the paper is as follows. Section 2 describes the feature selection approach that we adopt and introduces the two different penalties that we will consider. Section 3 discusses implementation aspects. Sections 4 and 5 describe the two applications upon which we base our analysis: microarray data analysis and object detection. Section 6 is left for a final discussion.

2 Regularized feature selection

In this section, we describe the basic algorithm upon which our feature selection method is built. We restrict ourselves to the case of a *linear* dependence between input and output data, which means that the problem can be reformulated as the solution of the following linear system of equations:

$$y = X\beta \tag{1}$$

where $y = (y_1, \ldots, y_n)^\top$ is the $n \times 1$ vector containing output values, $X = \{X_{ij}\}, i = 1, \ldots, n; j = 1, \ldots, p$ is the $n \times p$ matrix containing the collection of features j for each datum i and $\beta = (\beta_1, \ldots, \beta_p)^\top$ the vector of the unknown weights to be estimated. The data, or *training set* may represent different realizations of the problem under analysis, for instance different patients in the microarray domain or a set of images in computer vision. The output values may be a continuous real variable (such as e.g. a measurement of some biological parameter) or a discrete class label, in the case of classification. In standard regression problems the aim is to devise a rule relating the variable or response y to the input contained in the matrix X.

Since in many practical applications X is large, the usual approaches to the solution of the algebraic system (1) turn out to be infeasible. Moreover, usually the number of features p is much larger than the training set size n, $p \gg n$, so the system is hugely underdetermined. Finally, because of the redundancy of the feature set, which is common in high-dimensional problems, we also have to deal with collinearities responsible for ill-conditioning. A classical way to remedy to these curses is provided by regularization methods.

One of the best known strategies consists in adding a quadratic penalty, namely the L_2 norm of the coefficient vector ($|\beta|_2^2 = \sum_j |\beta_j|^2$), to the loss function, i.e. to find the minimizer of the following penalized least-squares objective function

$$\Phi_{\text{ridge}}(\beta) = \left[|y - X\beta|_2^2 + \mu|\beta|_2^2 \right] \tag{2}$$

where μ is a positive parameter, called the *regularization parameter*, the role of which is to balance properly the respective influence of the data-misfit and of the penalty. This estimate is known as the *ridge regression* estimate in the statistical literature (Hoerl and Kennard 1970; Hastie et al. 2001), as the Tikhonov regularized solution in the deterministic inverse problem theory (see the general references, Engl et al. 1996 or Bertero and Boccacci 1998), and as weight decay in the Neural Networks community (Werbos 1988). Since the objective function (2) is strictly convex, it admits a unique minimizer given by

$$\beta_{\text{ridge}} = (X^T X + \mu\, Id)^{-1} X^T y \tag{3}$$

where Id represents the $(p \times p)$ identity matrix. This is a regularized (stabilized) version of the ordinary least-squares estimate.

Linear regularization methods select the relevant (stable) components of the solution and discard the others only on the basis of spectral properties of the matrix $X^T X$, independently of the data or response vector y. It is only in the choice of the regularization parameter μ, that the error level and therefore the output y is taken into account. Moreover, since the eigenvectors are linear combinations of all the components of β, such methods are unable to perform variable selection i.e. to select a few coefficients that would account for the response. These drawbacks of linear methods can be overcome when resorting to nonlinear techniques, involving thresholding or nonlinear shrinkage; such methods will be described in the remainder of the section.

2.1 L_1 penalization

A simple instance of a nonlinear regularization method is obtained when replacing in the ridge objective function (2) the quadratic L_2-penalty term $|\beta|_2^2$ by the L_1-norm of the vector of the regression coefficients: $|\beta|_1 = \sum_{j=1}^p |\beta_j|$. Clearly, such modification increases the penalty on the small coefficients (those for which $|\beta_j| < 1$) whereas it decreases the penalty on the large coefficients (those for which $|\beta_j| > 1$). As a result, it favors the recovery of a few large coefficients instead of many fairly small ones. Moreover, the L_1-penalty favors truly zero values instead of small ones, i.e. it favors *sparse* solutions. To see this, let us consider the minimization of the resulting objective function with regularization parameter τ

$$\beta_{\text{lasso}} = \text{argmin}_\beta \left[|y - X\beta|_2^2 + \tau|\beta|_1 \right] \tag{4}$$

in the special case where $p = n$ and X is the identity matrix. The minimizer is then easily seen to be

$$(\beta_{\text{lasso}})_j = S_\tau(y_j) \tag{5}$$

where S_τ is the *soft-thresholder* defined by

$$S_\tau(\gamma) = \begin{cases} (|\gamma| - \tau/2)\text{sign}(\gamma) & \text{if } |\gamma| \geq \tau/2 \\ 0 & \text{if } |\gamma| < \tau/2. \end{cases} \tag{6}$$

In the general case, however, the solution of the optimization problem is not so simple and several algorithms have been proposed in the literature to actually compute minimizers of (4). The Lasso (Least Absolute Shrinkage and Selection Operator, Tibshirani 1996) has been proposed in a statistical framework whereas the Basis Pursuit (Chen et al. 1998), based on quadratic programming with interior points was rather put in a signal processing context. More recently different algorithms for the same problem have been developed under the name Lars (Least Angle Regression) in Efron et al. (2004) and Dantzig Selector in Candes and Tao (2005).

In this work, we will adopt instead an iterative thresholding algorithm similar to the one proposed in De Mol and Defrise (2002) and Daubechies et al. (2004) and briefly described in the next Section. Before we move on to the description of the algorithm, we discuss a different type of penalty that can be useful to deal with groups of correlated features.

2.2 L_1–L_2 penalization

We consider now the case where we use both a lasso-type and a ridge-type penalty in order to select groups of correlated variables. The minimization problem is then

$$\beta_{\text{en}} = \text{argmin}_\beta \left[|y - X\beta|_2^2 + \mu|\beta|_2^2 + \tau|\beta|_1 \right]. \tag{7}$$

It has been originally proposed in Zou and Hastie (2005) under the name "Elastic Net". More precisely the estimator (7) is called the *naïve elastic net* whereas the so-called *elastic net* is simply a rescaled version obtained by multiplying the naïve estimator by $(1 + \mu)$.

With the pure L_1 penalty, corresponding here to the case $\mu = 0$, if X has a non trivial null-space, the solution to (4) may be non unique. In particular, in the case of correlated features, the Lasso estimator can produce very different solutions, either by picking one of those with a great weight or picking a few of them or else picking them all. Depending on the application of interest, as we will see in Sects. 4 and 5, this phenomenon may be rather annoying. The situation is easily understood by examining the case example where we artificially replicate some columns of the matrix X creating perfectly correlated (replicated) variables. The lasso objective function is perfectly indifferent to the way the appropriate weight is distributed among those predictors since all situations yielding the same value for the L_1-norm are perfectly equivalent.

We can put the full weight on (an arbitrary) one of these predictors, or put equal weight on all of them, or on some of them in many different ways. On the other hand, the L_2 ridge penalty forces democracy in such cases since the L_2-norm of the equal-weight configuration is smaller than that of all other configurations. However, contrary to the L_1 penalty, the pure L_2-penalty ($\tau = 0$) does not allow to perform variable selection at all, especially in the presence of noise, where we will have many small but non-zero coefficients. It is the virtue of the combination of the two penalties that allow to both select variables and, among groups of correlated variables, to force democracy (see also, Zou and Hastie 2005). Notice, however, that for prediction purposes only, different pairs of parameters will be perfectly equivalent if at least one representative from each relevant group is selected and all the irrelevant features are discarded. In fact, the surface of the generalization error versus the two parameters presents an entire plateau of global minima as it has been empirically verified in the experiments, where the generalization error is estimated through the cross-validation. Since we could do well with one representative of each group of correlated genes, using the L_1-norm penalty could be the strategy of choice, leading to the sparsest among the solutions with highest prediction power (Sect. 5). Nevertheless if we are interested in learning all the features involved in the process, the addition of a L_2-norm penalty with its grouping effect will be preferable (Sect. 4).

3 An iterative algorithm for regularized feature selection

To compute the solution of the elastic net, we adopt a procedure which is a modification of Landweber or gradient descent (Engl et al. 1996) iterative procedure used to compute the least squares solution through the successive approximations scheme

$$\beta^{(l+1)} = \beta^{(l)} + X^T y - X^T X \beta^{(l)}, \quad l = 0, 1, \ldots \tag{8}$$

With the addition of a *soft-thresholding* at each iteration we obtain the following iterative algorithm which has been demonstrated in Daubechies et al. (2004) to converge to a minimizer of the Lasso objective function (4)

$$\beta_{\text{lasso}}^{(l+1)} = \mathbf{S}_{\frac{\tau}{\mathbf{C}}}\left(\beta_{\text{lasso}}^{(l)} + \frac{1}{C}\left[X^T y - X^T X \beta_{\text{lasso}}^{(l)}\right]\right); \tag{9}$$

where the constant C is a strict upper bound for the norm of the matrix $X : \|X^T X\| < C$, and the thresholding operator acts on a vector componentwise by performing the soft-thresholding operation defined by (6) ($[\mathbf{S}_\tau(\beta)]_j = S_\tau(\beta_j)$). This operation enforces the sparsity of the regression coefficients and the number of selected features (corresponding to non-zero weights) is controlled by τ. Notice that if the null-space of X is not trivial (in particular if $n < p$), the Lasso objective function is not strictly convex, and therefore the minimizer of (4) is not necessarily unique.

Similarly, we can derive an equivalent iterative scheme for the objective function with combined penalties which is a damped and thresholded Landweber scheme

(De Mol et al. 2007):

$$\beta_{\text{en}}^{(l+1)} = \frac{1}{1+\frac{\mu}{C}} \mathbf{S}_{\frac{\tau}{C}} \left(\beta_{\text{en}}^{(l)} + \frac{1}{C} \left[X^T y - X^T X \beta_{\text{en}}^{(l)} \right] \right). \tag{10}$$

To obtain a solution to the Elastic Net (as opposed to the so called naïve one) we have to suppress the damping factor at the last iteration.

Equation (10) is a general form comprising the ones mentioned above: we recover the case of ridge regression and of the damped Landweber iteration for $\tau = 0$, whereas pure lasso and thresholded Landweber correspond to the special case $\mu = 0$. For $\tau = \mu = 0$, we get the original Landweber iteration, that converges to a minimizer of the unpenalized least-squares objective function.

We conclude by remarking that, as in all iterative processes, a stopping rule has to be defined. When the L_1–L_2 algorithm is used as predictor, such rule can be simply defined through a tolerance δ, that is the iterations stop when $|\beta^{(l+1)} - \beta^{(l)}|_2 \leq \delta |\beta^{(l)}|_2$. When the algorithm is used as feature selector, we are not interested in recovering the coefficient vector β, but rather in determining its support; as a consequence it may be the case to define a stopping rule depending on the stability of the selected features.

4 Microarray data analysis

4.1 The problem

A major application of feature selection is found in the analysis of high-throughput biomedical data, such as gene expression arrays (microarrays), where thousands of features are studied with comparably few training examples. In the context of gene expression analysis, a main target of machine learning is to identify the set of genes relevant to the underlying biological process and leading to accurate diagnosis. However the presence of correlated variables often combined with the small number of examples causes most learning algorithms to achieve high prediction rates selecting only a subset of the relevant variables. This is common in gene expression data where the input is made of groups of correlated genes and only some representatives from these groups are usually involved in the determination of the classifier. For this reason the double penalty of the L_1–L_2 algorithm is extremely helpful in gene expression analysis, being potentially capable of selecting all genes from such relevant groups. Interestingly, appropriate tuning of the two parameters allows us to extract nested sets of genes with equivalent prediction performance, but determined by a different sparsity-correlation ratio. The lists of genes obtained varying μ, which differ for the degree of correlation among genes in each list, represent a useful output for biologists and geneticists; in fact, being each list almost perfectly nested in those with higher correlation degree, the least correlated gene set can be interpreted as the smallest set of biomarkers for the problem in study, whereas the other lists represent the same basic set with the addition of correlated genes.

In the next subsections we describe a protocol for applying such learning algorithm on microarray data analysis and then present some experimental results on benchmark microarray data.

4.2 The method

In order to exploit all the potentiality of the L_1–L_2 algorithm on gene expression data we have designed a learning machine for gene selection which is a combination of the doubly penalized least squares with ridge regression (see also De Mol et al. 2007). The former is employed as gene selector, whereas the weights of the selected genes are then estimated with ridge regression. The motivation of such choice is found in the underestimating behavior of the L_1-regularization when used as a predictor. In fact, through some simulations, we observed that the algorithm often returns an output vector β which has correct support, but its coefficients are strongly underestimated. Such a behavior, which has already been mentioned in Candes and Tao (2005), provides a hint that L_1-regularization has a high accuracy as feature selector, but that it is much less accurate as a predictor.

In more details, we first determine an estimator $f_1(X) = X\beta_1$ with the L_1–L_2 algorithm on all the features, we select the dimensions (genes) corresponding to the non null coefficients of β_1 and perform ridge regression on the selected dimensions, hence determining a second estimator $f_2(X) = X\beta$. The final classifier is given by $y = \text{sign}(X\beta)$. It is interesting to observe that no prescreening is performed to reduce dimensionality of the input data, since the algorithm can easily handle high-dimensional data-sets, provided that the training set size is sufficiently small.

The learning machine described earlier depends on three free parameters, τ and μ from the L_1–L_2 algorithm as defined in the objective function (7), and another parameter λ determining the degree of regularization in ridge regression. Notice that for fixed τ and λ different choices of μ will lead to a different classifier determined by a different grouping effect, but with equal prediction power.

Optimal choices for τ and λ are determined through minimization of the generalization error of the L_1–L_2 algorithm with parameter μ set to zero. This procedure returns the minimum number of genes needed to classify the samples; in order to select also other relevant genes, which may be correlated with this basic set, we can run the learning machine with increasing values of μ hence preserving correlation. Since the generalization error cannot be empirically evaluated we determined the optimal choice of the parameters through the minimization of an estimate of the generalization error obtained via cross-validation. In order to avoid any over-fitting we follow a strict validation protocol:

1. Split of the entire data set in training and test set.
2. Set $\mu = 0$, then for different values of τ in a given range, determine with cross validation the best parameters $(\tau_{\text{opt}}, \lambda_{\text{opt}})$ leading to the smallest validation error.
3. Determine the final classifiers (one for each value of μ) by training the entire training set with L_1 parameter τ_{opt}, ridge parameter λ_{opt} and varying the $L - 2$ parameter μ.
4. Test the performance of the final classifiers.

As final result we obtain a different classifier for each value of the parameter μ. From a theoretical point of view there seems not to be a motivation for choosing one particular solution, unless we are asked to take the minimum L_0 norm—the smallest set of genes—which would lead to the choice $\mu = 0$. However, when handling microarray data, geneticists may be interested not only in the smallest group of genes capable of discriminating between the two classes but also in all the genes correlated to such group. In fact, thanks to the grouping effect, the presence of the L_2-norm in the penalization term enforces the algorithm to select not only the genes strictly necessary for good prediction (which will be selected with $\mu = 0$) but also those genes correlated with such basic group. We therefore expect each group of genes selected with fixed μ to have an almost complete enclosure in the groups selected with bigger values of μ.

4.3 Experiments

We carried out experiments on publicly available data sets concerning classification problems related to three different diseases: leukemia, lung cancer and prostate cancer. The first one is the Golub data set Golub et al. (1999)[1] which comprises expressions of 7129 probe sets for 72 patients divided in two classes according to their diagnosis, Acute Myoloid Leukemia (AML) and Acute Lymphoblastic Leukemia (ALL). The lung cancer data set Gordon et al. (2002)[2] is made of 151 patients with 12533 probe sets and each patient is either affected by malignant pleural mesothelioma (MPM) or adenocarcinoma (ADCA). The last data set Singh et al. (2002)[3] is again made of 12533 probe sets for 102 instances, tumor or normal tissue.

We run the L_1–L_2 algorithm with the protocol described above on the three datasets. Figure 1 reports results related to *step 1*: the validation error is obtained varying the L_1 parameter τ (and setting $\mu = 0$). Each value is obtained minimizing the validation error with respect to the ridge parameter λ. In Table 1 are reported results related to *step 2*: for increasing value of μ are listed the number of selected genes and the test error.

Concerning prediction accuracy we can say that on these data sets performance of the L_1–L_2 algorithm is comparable with other learning algorithms applied in the original works to the same data sets. In fact, on leukemia data set we achieve 100% accuracy on both classes, whereas in Golub original paper Golub et al. (1999) a 50-genes classifier is built which made predictions on the test set with 100% accuracy, however only 29 of the 34 test instances had strong prediction. Concerning the lung cancer data analysis in Gordon et al. (2002), different classifiers were obtained with prediction accuracy ranging from 91 to 99% to be compared with the 99% achieved with our algorithm. In the end for the prostate cancer in Singh et al. (2002), after gene ranking with variation of signal to noise metric, k-NN algorithm is run on the test set with a prediction accuracy ranging from 82.9 to 95.7% according to the number of genes (4 or 6) used in the algorithm; with L_1–L_2 algorithm we achieve 94% accuracy.

We also run the learning machine on normalized data; that is for each split we estimated the mean and the standard deviation of the data on the training set, and recentered both training and validation sets with respect to these estimates, so that for the training set we have $\sum_{i=1}^{n}(X_{tr}^{i})_j = 0$ and $\sum_{i=1}^{n}[(X_{tr}^{i})_j]^2 = 1 \; \forall j$. We observed

1 http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

2 http://www.chestsurg.org.

3 http://www-genome.wi.mit.edu/mpr/prostate.

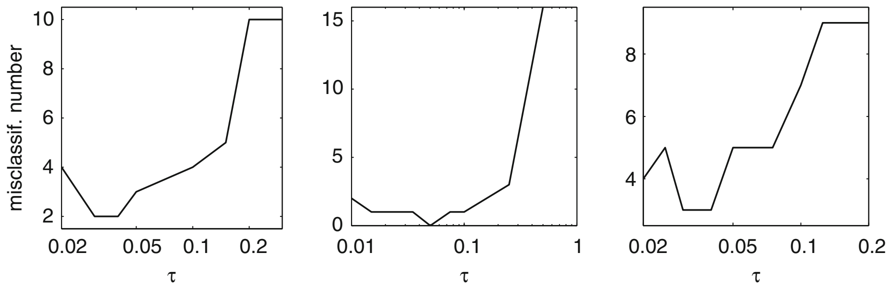

Fig. 1 Minimum cross-validation error (number of misclassified samples) with respect to ridge parameter τ for leukemia (*left*), lung cancer (*middle*) and prostate cancer (*right*)

Table 1 Results of L_1–L_2+ ridge algorithm on microarray data sets

Leukemia			Lung cancer			Prostate cancer		
μ	Number of sel. genes	Test error	μ	Number of sel. genes	Test error	μ	Number of sel. genes	Test error
0	15	1/34	0	18	1/90	0	17	6/51
0.00025	21	1/34	0.0001	18	1/90	0.0001	19	6/51
0.0005	33	1/34	0.00025	23	1/90	0.0005	21	5/51
0.001	44	1/34	0.0005	31	1/90	0.001	26	5/51
0.0025	70	0/34	0.001	37	1/90	0.005	52	3/51
0.005	94	0/34	0.0025	59	1/90	0.01	75	4/51
0.01	126	0/34	0.005	85	1/90	0.025	101	3/51
0.05	211	0/34	0.01	113	1/90	0.05	123	6/51
0.1	237	0/34	0.025	191	1/90	0.1	148	5/51

that prediction performance is lower with normalized data, and we may explain such effect by guessing a loss of information induced by across-array normalization on the data sets we examined in this work.

5 Object detection in images

5.1 The problem

Object detection amounts at finding one or more instances of an object or a class in a digital image. From the algorithmic point of view it boils down to classifying image subregions. Usually it is modeled as a binary classifier, where we assign a positive (1) or negative (−1) label to an image sub-region in accordance to the presence or absence of the object of interest. A variety of object detectors of different nature have been proposed in the literature. Here, for the sake of simplicity, we focus on the widely studied *face detection*. In the last years face detection has been boosted by the contribution of the learning from examples paradigm which helps to solve many well-known problems related to face variability in images (Yang et al. 2002; Osuna et al. 1997; Schneiderman and Kanade 2000). In particular, component-based approaches highlighted the fact that local areas are often more meaningful and more appropriate for dealing with occlusions and deformations (Mohan et al. 2001; Ullman et al. 2002).

Among the various ways of representing the image content, we consider the so called *rectangle features* that have been used with success on a number of object detection problems and on face detection in particular (Viola and Jones 2004). Rectangle features “expand” the information carried by each image in a high-dimensional dictionary of overcomplete features, describing various local patterns. The number of features obtained is related to the image dimensions, and it is usually very large. For example in a 19 x 19 pixel image, the number of rectangle features is about 64,000. Considering the amount of redundant information contained in images, it is obvious that an automatic feature selection step is necessary to obtain a small and compact set of features that capture the most meaningful patterns for a given classification problem.

In the remainder of the section we describe a procedure to extract meaningful features leading to an efficient face detection architecture. Notice that this architecture may be applied to different object detection problems, simply by changing the training set; here the main objective of feature selection is to obtain a very compact set of features that allow us to build an effective real-time face detection system. To this purpose, due to the high degree of redundancy of image features, we apply the L_1 algorithm.

5.2 The method

Our face detection machine is a combination of a L_1 feature selection and a cascade of linear SVM classifiers. The SVM cascade follows a rather standard design for real time object detection (Heisele et al. 2001; Viola and Jones 2004), while the feature selection procedure has been implemented efficiently with a split&merge approach, in order to deal with a high-dimensional matrix (Destrero et al. 2007). Thus, we look for a solution of problem (1), where matrix X is built from a training set of positive (faces) and negative (other patterns) example images, vector y represent the class labels (1 for faces -1 for non faces), and β represents the unknown weight vector. In this case the matrix X will become rather big: in the case of images it is common to have at disposal thousands of images of the same class, each of which represented by a fairly high-dimensional vector. For this reason applying the iterative algorithm described in Eq. 9 directly to the whole matrix may not be feasible on all systems: the matrix multiplication needs to be implemented carefully so that we do not keep in primary memory the entire matrix X. One possibility is to compute intermediate solutions with multiple accesses to secondary memory.

We implemented a different approach, based on resampling the features set and obtaining many smaller problems, that can be briefly described as follows: we build S feature subsets *each* time extracting m features from the *original* set of size p ($m \ll p$), we then obtain S smaller linear sub-problems of the type: $X_s \beta_s = y$ for $s = 1, \ldots, S$, where X_s is a submatrix of X containing the columns relative to the features in s; β_s is computed accordingly. As for the choice of the number S of sub-problems and their size we observe that the subset size should be big enough to be descriptive, small enough to handle the matrix easily; thus, we consider subsets of features made up with a 10% of the original size. To choose the number of sub-problems S, we rely on the binomial distribution and estimate how many extractions are needed so that each feature is extracted at least 10 times with high probability. After we build the S sub-problems we look for the S solutions running S iterative methods as in Eq. (8). At the end of the process we are left with S overlapping sets of features. The final set of features, say S_1, is obtained choosing the features that have been selected each time they appear in the subsets.

In order to obtain a smaller set of meaningful features we further apply the feature selection procedure to the set S_1 looking for a new, sparser, solution. The new data matrix is obtained by selecting from X the columns corresponding to S_1, and $y_{S_1}^{(0)}$ is initialized again to a vector of zeros. We call S_2 the set of features obtained from this further selection step. The final set of features S_3 is obtained through a further procedure that cleans the set from the presence of correlated features, applying the Spearman's test to this purpose.

The features in S_3 is used are used to build a cascade of small SVMs that is able to process video frames in real time. The cascade of classifiers analyses an image according to a standard coarse-to-fine approach. Each image patch at a certain location and scale is the input of the classifiers cascade: if the patch fails one test it is immediately discarded; if it passes all tests a face is detected. Each classifier of the cascade is built starting by three mutually distant features, training a linear SVM on such features, and adding further features until a target performance is obtained on a validation set. Target performance is chosen so that each classifier will not be likely to miss faces: we set the minimum hit rate to 99.5% and the maximum false positive rate to 50%. Assuming a cascade of 10 classifiers, we would get as overall performance: $\mathrm{HR} = 0.995^{10} \sim 0.9$ and $\mathrm{FPR} = 0.5^{10} \sim 3 \times 10^{-5}$ (Viola and Jones 2004).

5.3 Experiments

We carried out experiments to validate the quality of the selected features and, as a consequence, the quality of the face detection machine. We started off with a publicly available training set of frontal faces[4] and with a set of faces and non faces images acquired in our department.[5] We start from measurement matrix X which is $4{,}000 \times 64{,}000$, and if computed all in once would occupy about 1 Gb of memory. Therefore it is advisable to apply our split & merge procedure. As for the number of subproblems we observe that, if $S = 200$, such a probability is 99.6% which is good enough to our purposes—in practice only 256 features over 64,000 are extracted less then 10 times. Thus, we apply the various selection stages described above, tuning the parameter τ in order to achieve a compromise between the number of selected features (that we would like to be small) and the estimated error on a validation set. Table 2 reports how the error and the number of features vary while changing the parameter τ. We observe that the error is not too sensitive to the parameter choice, while the number of selected

[4] http://cbcl.mit.edu/software-datasets/FaceData2.html.

[5] This dataset contains 4,000 training and 3,000 validation images and over 20,000 test data. All images are 19 x 19 pixel. The dataset is available upon request.

Table 2 The trend of error and features number while varying the parameter τ

τ	Number of sel. features	Test error
1 × 10e-8	4,636	0.022
1 × 10e-7	4,595	0.023
1 × 10e-6	2,219	0.022
2 × 10e-6	1,922	0.022
5 × 10e-6	1,813	0.023
1 × 10e-5	1,791	0.021
2 × 10e-5	1,303	0.020
5 × 10e-5	867	0.022
1 × 10e-4	581	0.023
2 × 10e-4	247	0.027
4 × 10e-4	66	0.135
5 × 10e-4	0	–

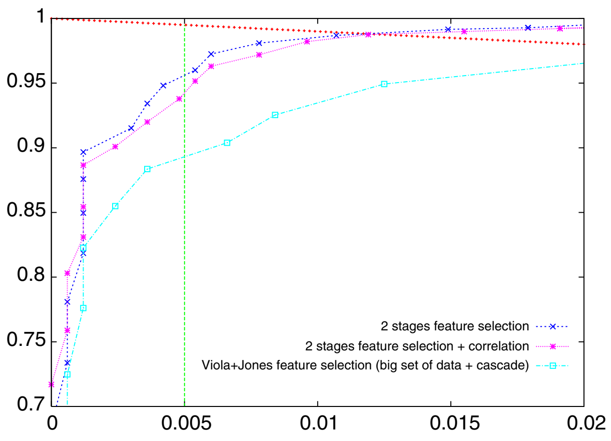

Fig. 2 ROC curve comparing the classification results obtained with our approach and with an Adaboost feature detector all trained with the dataset collected in our department

features depends critically from this choice. In this case we chose τ corresponding to an error lower than 0.1 and leading to the smaller set of features: $\tau = 2 \times 10e - 4$.

For what concerns the prediction accuracy we validated the face detector architecture on a test dataset of over 20,000 images acquired by us. Figure 2 reports a comparison between the results obtained with our method (with and without the final step that leads to the minimal set of features) and Adaboost trained on our same set of data. The ROC curves show how our approach is superior, when the size of the training set is relatively small.

Currently our face detector is running as a module of a monitoring system installed in our department (Destrero et al. 2007). Table 3 shows its detection performance in two different situations. The first row of the table refers to the results obtained on

Table 3 The performance of the face detection system on a controlled set of images (top row) and on a 5 hours unconstrained live video (bottom row)

Test data	False pos rate (%)	Hit rate (%)
Test images	0.1	94
5 h live	4×10^{-7}	76

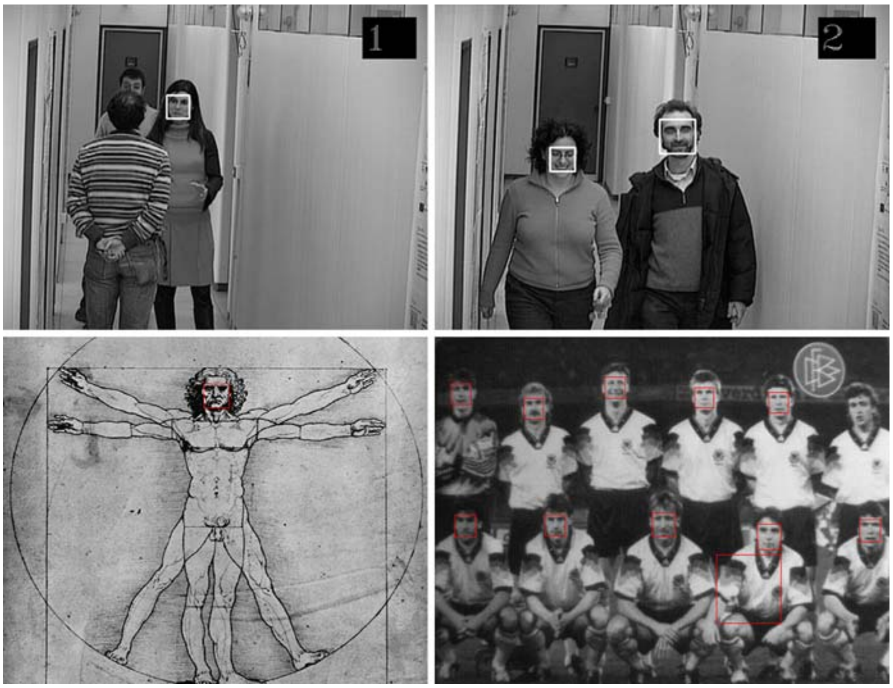

Fig. 3 Detection results on a selection of images from our monitoring system and from public datasets

images acquired in controlled conditions, while the second row refers to uncontrolled detection: we manually labeled the events occurring in a 5 h recording of a busy week day; the recording was entirely out of our control and it included changes of the scene, people stopping for impredictable time, lateral faces. Notice that at run time, the classifiers have been tuned so as to minimize the number of false positives. As a final remark we observe that the amount of data analysed is huge since, on average, the detector analyses 20,000 patches per image or frame.

We also tested our face detection method on benchmark datasets—test set A and C of MIT-CMU and the BioID dataset—obtaining results consistently above an Adaboost detector (Viola and Jones 2004) trained with our same dataset. Keeping the false positive rate fixed, the hit rate gain with our approach is on average 8% on the various test sets. Figure 3 shows a few detection examples, both obtained from our monitoring system and from running the detector on publicly available datasets. See Destrero et al. (2007) for more examples and details on the experiments.

6 Discussion

In this paper we discussed feature selection for data living in high-dimensional spaces, basing our analysis on two prototypical applications: microarray analysis and object detection. We adopted a regularized scheme to feature selection and, in particular, we considered both L_1 and mixed L_1–L_2 types of penalties. The former does not always lead to a unique solution in terms of the selected features, but it allows us to keep the number of selected features minimal; it is therefore recommended in those applications where features are not important per se, but for the information that they carry. Also it is useful in such applications where a compact description of the data is a requirement. L_1–L_2 penalty, instead, guarantees a unique solution, and guarantees that groups of correlated features are all kept in the final set with comparable weights. It may be adopted in applications were one wants to maintain all the expressive power of the original representation simply discarding meaningless features. These schemes can be applied to problems with huge underdetermination, as biological data tend to be, and, with the help of ad hoc split and merge strategies, may be used even if the matrix size grows.

From the application standpoint, the adopted scheme allowed us to achieve results in line, and sometimes superior, to state-of-the-art solutions and, at the same time, gain a deeper insight of the features which are significant for a given problem.

Acknowledgments The authors would like to thank Ernesto De Vito, Lorenzo Rosasco, Annalisa Barla for many useful discussions on these topics. This work has been partially supported by the FIRB project LEAP PRIN04PARL and by the EU Integrated Project Health-e-Child IST-2004-027749. C. De Mol acknowledges support of the ARC grant Nb 02/07-281 as well as the kind hospitality of DISI where part of this work was completed.

References

Bertero M, Boccacci P (1998) Introduction to inverse problems in imaging. Institute of Physics Publishing, Bristol and Philadelphia

Breiman L, Friedman JH, Olshen A, Stone CJ (1984) Classification and Regression Trees. Wadsworth and Brooks, Belmont

Candes E, Tao T (2005) The Dantzig selector: statistical estimation when P is much larger than N

Chen S, Donoho D, Saunders M (1998) Atomic decomposition by basis pursuit. SIAM J Sci Comput 20(1):33–61

Daubechies I, Defrise M, De Mol C (2004) An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Commun Pure Appl Math 57:1413–1457

De Mol C, Defrise M (2002) A note on wavelet-based inversion algorithms. Contemp Math 313:85–96

De Mol C, Mosci, Traskine MS, Verri A (2007) Sparsity enforcing and correlation preserving algorithm for microarray data analysis. Technical Report DISI-TR-07-04, DISI, Università di Genova

Destrero A, De Mol C, Odone F, Verri A (2007) A regularized approach to feature selection for face detection. Technical Report DISI-TR-07-01, DISI, Università di Genova

Destrero A, Odone F, Verri A (2007) A system for face detection and tracking in unconstrained environments. In: Advanced video and signal based surveillance, AVSS, London, 2007. ISBN 978-1-4244-1696-7/07

Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Ann Stat 32:407–499

Engl HW, Hanke M, Neubauer A (1996) Regularization of inverse problems. Math Appl 375

Forman G (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1306

Gordon GJ, Jensen RV, Hsiao L, Gullans SR, Blumenstock JE, Ramaswamy S, Richard WG, Sugarbaker DJ, Bueno R (2002) Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer Res 62:4963–4967

Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182

Hastie T, Tibshirani R, Friedman J (2001) The elements of statistical learning. Springer, Heidelberg

Heisele B, Serre T, Mukherjee S, Poggio T (2001) Feature reduction and hierarchy of classifiers for fast object detection in video images. In: IEEE proceedings of CVPR

Hoerl AE, Kennard R (1970) Ridge regression: biased estimation for nonorthogonal problems. Technometrics 12:55–67

Kohavi R, John G (1997) Wrappers for feature subset selection. Artif Intell 97(1-2):273–324

Mohan A, Papageorgiou C, Poggio T (2001) Example-based object detection in images by components. IEEE Trans Pattern Anal Mach Intell 23(4):349–361

Osuna E, Freund R, Girosi F (1997) Training support vector machines: an application to face detection. In: IEEE proceedings international conference on computer vision and pattern recognition (CVPR), pp 130–136

Schneiderman H, Kanade T (2000) A statistical method for 3D object detection applied to faces and cars. In: IEEE proceedings international conference on computer vision and pattern recognition (CVPR), pp 1746–1759

Singh D, et al (2002) Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1:203–209

Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc Ser B 56:267–288

Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286:531–537

Ullman S, Vidal-Naquet M, Sali E (2002) Visual features of intermediate complexity and their use in classification. Nat Neurosci 5(7):682–687

Viola P, Jones MJ (2004) Robust real-time face detection. Int J Comput Vis 57(2):137–154

Werbos P (1988) Backpropagation: past and future. In: Proceedings of the IEEE international conference on neural networks. IEEE Press, pp 343–353

Weston J, Elisseeff A, Schoelkopf B, Tipping M (2003) Use of the zero norm with linear models and kernel methods. J Mach Learn Res 3:1439–1461

Weston J, Elisseeff A, Scholkopf B, Tipping M (2003) The use of zero-norm with linear models and kernel methods. J Mach Learn Res 3:1439–1461

Yang M-H, Kriegman DJ, Ahuja N (2002) Detecting faces in images: a survey. IEEE Trans Pattern Anal Mach Intell 24(1):34–58

Zhu J, Rosset S, Hastie T, Tibshirani R (2004) 1-norm support vector machines. In: Thrun S, Saul LK, Schölkpf B (eds) Advances in neural information processing systems 16. MIT Press, Cambridge, pp 49–56

Zou Z, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc Ser B 67:301–320