A projection pursuit framework for supervised dimension reduction of high dimensional small sample datasets

Neurocomputing, Amsterdam, v. 149, p. 767-776, Feb. 2015
http://www.producao.usp.br/handle/BDPI/50247

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
A Projection Pursuit framework for supervised dimension reduction of high dimensional small sample datasets

Soledad Espezua\textsuperscript{a,}\textsuperscript{*}, Edwin Villanueva\textsuperscript{a}, Carlos D. Maciel\textsuperscript{b}, André Carvalho\textsuperscript{a}

\textsuperscript{a} Department of Computer Science, IC\textsuperscript{MC} - USP, University of São Paulo, Brazil
\textsuperscript{b} Department of Electrical Engineering, São Carlos School of Engineering, University of São Paulo, Brazil

\begin{abstract}

The analysis and interpretation of datasets with large number of features and few examples has remained as a challenging problem in the scientific community, owing to the difficulties associated with the curse-of-the-dimensionality phenomenon. Projection Pursuit (PP) has shown promise in circumventing this phenomenon by searching low-dimensional projections of the data where meaningful structures are exposed. However, PP faces computational difficulties in dealing with datasets containing thousands of features (typical in genomics and proteomics) due to the vast quantity of parameters to optimize. In this paper we describe and evaluate a PP framework aimed at relieving such difficulties and thus ease the construction of classifier systems. The framework is a two-stage approach, where the first stage performs a rapid compaction of the data and the second stage implements the PP search using an improved version of the SPP method (Guo et al., 2000,[32]). In an experimental evaluation with eight public microarray datasets we showed that some configurations of the proposed framework can clearly overtake the performance of eight well-established dimension reduction methods in their ability to pack more discriminatory information into fewer dimensions.

\end{abstract}

\section{Introduction}

In the last few decades we have witnessed a rapid development and refinement of data acquisition technologies in several science and industrial areas [1]. This has led to the emergence of high-throughput technologies that are capable of generating datasets with the number of features \textit{(p)} far greater than the number of examples \textit{(n)}, the so-called \textit{large p small n} datasets. A representative example of these technological developments is the microarray technology [2], which has made possible the measurement of expression levels of thousands of genes in a relatively rapid and economic way, leading to significant advances in the understanding of severe diseases, like cancer, and raising hopes on possible cures [3,4].

Though the collection of \textit{large p small n} datasets is nowadays a common practice in many fields, their analysis and interpretation is still a challenging task [5,6,1]. This difficulty is mainly originated by the so-called “curse of dimensionality” phenomenon, inherent in such a kind of data [7]. This phenomenon states that as the dimensionality increases, the corresponding space becomes emptier and the data points tend to be equidistant. This generates detrimental impacts in most machine-learning and pattern-recognition methods (including model-estimation instability, model over fitting and local convergence), compromising the generalization performance and reliability of such methods [5,6].

A common approach to circumvent the curse of dimensionality is by reducing it [6]. Two kinds of methods exist for this task: feature selection (FS) [8,9] and feature extraction (FE) [10,11]. The former methods try to find small subsets of original features that are relevant to the intended analysis. The latter methods reduce the dimensionality by building new features from combinations (linear or nonlinear) of the original features. FS has the benefit of keeping the original feature meaning, facilitating the interpretability by the domain expert [9]. However, it has been said [12] that FE is preferable over FS when the final goal is an accurate system for classifying new examples and interpretability is not as important. This is because FE is not tied to the original feature space, providing greater chances of finding more useful representations for the desired task [12].

Projection pursuit (PP) [13,14] is a FE method that has been successfully applied in several domains for both supervised and unsupervised analyses (e.g. [15–18]). PP seeks low-dimensional linear projections of the data that expose interesting aspects of them. To this end, a measure of “interestingness” is employed, which is known as \textit{projection pursuit index} (PP index). A key advantage of PP is its flexibility to fit different pattern recognition tasks, depending on the PP index used. For example, PP can be
used to perform clustering analysis [19,20], classification [21–24], regression analysis [25] and density estimation [26] (some reviews of PP indexes can be found in [21,27,28]). Another advantage of PP is its out-of-sample mapping capability, that is, the possibility to map new examples in the projection space after the construction of it.

Despite the aforementioned advantages, the literature shows a limited use of PP in large p small n datasets, like those generated by microarray technology. This may be due to the high computational difficulty in finding optimal projection spaces for such cases. For instance, the projection of a dataset with $p = 10k$ features (a realistic number in microarray datasets) onto a target space of dimension $m = 3$ will require the optimization of a projection matrix of $p \times m = 30k$ elements. Evidently, the problem worsens as $p$ or $m$ increase. Traditional PP optimizers based on the gradients or Newton methods [29–31,19] are usually inadequate for such a kind of data due to the vastness of possible projections and, thus, the high susceptibility to find poor local optima [14].

More global PP optimizers were described recently, including genetic algorithms (GA) [32,33], simulating annealing (SA) [21], random scan sampling (RSSA) [34] and particle swarm optimization (PSO) [35]. However, none of these works have been directly applied in dimensionalities as high as those found in microarray data, which shows the difficulty of applying PP in such scenarios.

In this paper we present a framework to facilitate the applicability of PP on large p small n datasets with the aim of classification tasks. The framework is formed by two main stages (Fig. 1): a compaction stage and a PP optimization stage. The first stage is devised to rapidly transform the original data into a less sparse representation. The second stage is the PP part, which is responsible to find optimal projections taking the compacted representation as input.

For the compaction stage we use three well-known techniques: PCA, Whitening and Partial Least Squares. For the PP stage, we adopt the Sequential Projection Pursuit (SPP) approach [32] coupled with the GA optimizer (PPGA) we described recently [33], in which a specialized crossover operator showed excellent search capabilities. An experimental study is presented over eight public microarray datasets. The evaluation systematically tested several configurations of the framework, including variations of the compaction method, the PP index function and the target dimensionality. We used the predictive accuracy of two popular classification methods (LDA and 3NN) in order to assess the quality of the tested configurations. We also compare the framework against eight well-established dimension reduction methods, including FE and FS methods.

The paper is organized as follows. Section 2 introduces some important concepts of PP, SPP, PP optimization and PP indexes used in the paper. Section 3 describes the proposed framework. Section 4 presents the experimental evaluation, including the experimental setup, results and corresponding discussion. Finally, our conclusions are presented in Section 5.

2. Projection pursuit

The projection pursuit (PP) concept was formally introduced in the paper of Friedman and Tukey [13], although the seminal ideas were originally posed by Kruskal [36]. To describe the PP concept we assume that we have a data matrix $X$ of $n \times p$ dimensions, where $n$ is the number of data examples or observations and $p$ is the number of attributes or variables. PP can be defined as the constrained optimization problem in (1), where the aim is to seek a $m$-dimensional projection space ($m < p$) (defined by the bases – columns – of $A = [a_1, ..., a_m] \in \mathbb{R}^{p \times m}$) such that the projected data points in that space maximize a pre-defined objective function $I_3$, called the projection pursuit index. This function measures the degree of interestingness of the projected data. The constraint of orthonormality in $A$ is necessary to ensure that each dimension in the target space shows different aspects of the data:

$$A^* = \arg \max_ A \{I_3(XA)\}$$

subject to $A^T \cdot A = I$. 

\[1\]

2.1. Sequential projection pursuit

Sequential projection pursuit (SPP) [32] solves the PP problem in (1) by decomposing it into a sequence of $m$ optimization problems, each computing one base in $A$.

The first base, $a_1$, is obtained by searching a $p$-dimensional unit-length vector where the projected data $Xa_1$ maximizes the one-dimensional PP index $I_3$. Once $a_1$ is found, SPP tries to remove all the information captured in that direction from the original data in order to avoid finding the same projection direction in subsequent iterations. For this task, the original SPP uses a "structure removal" procedure [14], which “Gaussianizes" the data in the found direction, as follows: $X = X - Xa_1a_1^T$. The next base $a_2$ is sought taking the updated $X$ (also called residual data) as input data, subject to the constraint that $a_2$ is orthogonal to $a_1$. The process is iteratively repeated until all $m$ bases are obtained.

2.2. PP optimization

A key component in PP is the optimization process. Early approaches in this respect were based on the gradient techniques [30,29] and Newton–Raphson [31,37,14,13], where the projections are performed in at most three dimensions for visual exploratory tasks, the so-called exploratory projection pursuit (EPP). Further developments focused on developing more global methods for PP optimization, such as random search [38,39,29], genetic algorithm (GA) [32], random scan sampling algorithm (RSSA) [34], simulated annealing (SA) [21], particle swarm optimization (PSO) [35] and tribes [40]. In a previous work [33] we describe PPGA, a GA optimizer with a specialized crossover operator that often showed to find solutions better than those found by PSO, RSSA, and SA when used inside the SPP framework, reason why it is adopted for the present work.

Another important aspect in optimizing PP is how to ensure that each resulting dimension is associated with a different and complementary aspect of the data. Many PP methodologies, including SPP, address this task by using the “structure removal" procedure. However, it has been observed [41,38] that the successive application of this procedure (as done in the original SPP) may lead to data distortions, implying that an optimum found in residual data may not be longer related to any relevant aspect of the original data. Recently, Zhang and Chan [41,28] proposed an alternative approach to structure removal, which uses the orthogonal complement space concept. In those works, the residual data is obtained projecting the current data onto the orthogonal complement of the found projection vector, which avoid data distortions and also ensures orthogonality of the projection bases.

---

1 The orthogonal complement of one vector $x \in \mathbb{R}^n$ is the vector space $y$, all of which are orthogonal to $x$. Therefore, such space can be expanded by $n - 1$ vector basis. That is, the orthogonal space of a vector $x$ n-dimensional is always dimensional size $n - 1$. 
2.3. Projection pursuit indices

The choice of the PP index is very important, since it defines what is “interesting” in the data. A great deal of research in the PP community has been centered on the construction of meaningful PP indexes for different purposes. It is possible to find PP indexes for clustering analysis [42,19,38,43,30,13], for supervised analysis [44,45,21,5] and for regression analysis [46,47]. Given that this paper is targeted to supervised analysis, we briefly describe some relevant supervised PP indexes included in the experimental evaluation of the paper. To facilitate the description we consider that each example $x_i \in X$ has associated a class label $c_i \in C = \{1, \ldots, c_{\text{max}}\}$.

Index Bhattacharya (Bat): It is based on the Bhattacharya distance between classes and uses statistics of first and second orders. The Bhattacharya index for 1D projection space$^2$ is defined as [28,5]

$$J_{\text{Bat}} = \min_{i,j \in \mathcal{C}} \left\{ \frac{1}{4} \left( \frac{\mu_i - \mu_j}{\sigma_i + \sigma_j} \right)^2 + \frac{1}{2} \left( \frac{\sigma_i + \sigma_j}{2 \sqrt{\sigma_i \sigma_j}} \right) \right\},$$

where $\mu_i$ and $\sigma_i$ denote respectively the mean and the variance of the projected examples $Xa$ of class $i$.

Index quality projected clusters (qpc): This index favors projections that allow us to find compact pure clusters of vectors separated from other clusters [24,44]. The qpc index for 1D projection space is defined as

$$J_{\text{qpc}} = \sum_{i,j} \alpha_{ij} G(x_i - x_j) a,$$

where $\alpha_{ij} > 0$ if the examples $x_i$ and $x_j$ belong to the same class ($c_i = c_j$), in other case $\alpha_{ij} < 0$. Function $G(.)$ should be localized with maximum for $x = 0$ (e.g. Gaussian function).

Index Fisher linear discriminant analysis (lda): This index was adapted from the classical LDA method [27,21] and favors linear projections with greater separation between classes (in the sense of least squares) and lower dispersion intra-classes. Eq. (4) shows the formula for the index calculation [45]:

$$J_{\text{lda}} = 1 - \frac{\|A^T W A\|}{\|A^T (W + B) A\|},$$

where $B$ is the between-class scatter matrix and $W$ is the within-class scatter matrix.

Index neighborhood components analysis (nca): This index was not proposed as such but as a cost function in the nca method [48]. It has successfully used in several applications (e.g. [49–51]), reason why it is included in the present study. The cost function is derived from a stochastic neighbor assignment scheme and is proportional to the expected number of points correctly classified under that scheme:

$$J_{\text{nca}} = \sum_{i,j \in \mathcal{D}_i} \sum_{\Omega_{ij}} p_{ij},$$

where $\Omega_{ij}$ denote the set of examples in the same class as $i$ by $\Omega_i = \{ j | c_i = c_j \}$; $p_{ij} = 0$ and $p_{ij} = \exp(-\|x_i A - x_j A\|^2)/\sum_i \exp(-\|x_i A - x_i A\|^2)$ is the probability of example $i$ selecting example $j$ as its neighbor and inheriting its class label.

---

Index locality preserving (lp): This is an unsupervised index based on the Locality Preserving Projections (LPP) method [52]. This index favors projections that concentrate the neighboring data examples together. We define here the lp index (6) as the inverse of the original LPP criterion (this was defined for minimization):

$$J_{\text{lp}} = 1/(A^T X A),$$

where $L$ is a Laplacian matrix of the k-neighborhood graph (graph obtained by linking the k nearest neighbors to each example).

3. A PP framework for supervised dimension reduction of large p small n data

We detail here the proposed framework to ease the applicability of PP in large p small n data. Fig. 2 shows the structure of this framework. Two stages compose this: the first stage implements a fast procedure to compact the data into an intermediate-dimensional representation (in the order of $n$). The second stage is a PP procedure over the compacted data, which implements an improved version of the SPP scheme. Next, we describe each framework component:

3.1. Compaction stage

The goal of compaction stage is to reduce the high $p$-dimensional space $X$ to a less sparse $q$-dimensional space $W$, where the original information is preserved as much as possible. In this paper we tested the following methods for this purpose, based on their popularity, availability of implementations and ease of computation:

- The whitening transform (Whiten) [53]: It is also called sphering, and is a popular data transformation that produces uncorrelated and normalized attributes. The effects of whitening on large p small n data were recently studied by Deng et al. [54], finding that the whitened data points lie at the vertices of a regular $(n-1)$-dimensional simplex. This means that any distance-based method fails to work in the full whitened data, since all data points are equidistant. However, the authors also show that by pruning out some irrelevant attributes (those associated with the lowest singular values) of the transformed data, it is possible to produce highly informative data for subsequent analysis. In a related study, Vicente et al. [53] found that the most influential part on the performance of ICA is the whitened transformation. Various other authors pointed the feasibility of whitening as a pre-processing step in microarray data analysis [55–61]. Whitening is performed via singular value decomposition (SVD) on the centered$^3$ data matrix $X$:

$$X = U D V^T,$$

where $U \in \mathbb{R}^{n \times n}$ is the matrix of eigenvectors of $X^T X$; $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix containing the singular values of $X^T X$ (ordered in descending order); $V \in \mathbb{R}^{n \times n}$ is the matrix of eigenvectors of $X X^T$. The whitened data $W \in \mathbb{R}^{n \times q}$ is the matrix formed by the first q columns of U. This can be expressed in terms of the input data as $W = XR$, where $R = V D^{-1} \in \mathbb{R}^{n \times q}$ is the whitening transformation matrix (also used to get the final projection matrix) and $V \in \mathbb{R}^{n \times q}$ and $D \in \mathbb{R}^{q \times q}$ are obtained by

---

$^2$ In 1D projection spaces we use a instead of A to denote the projection base.

$^3$ Data resulting of subtracting the column-mean from the original data.
taking the first $q$ columns of $V$ and the upper left $q \times q$ submatrix of $D$, respectively.

- **Principal component analysis (PCA)** [62]: It is a well-known method of dimension reduction, which is used to construct a set of orthogonal components by maximizing the variance of the linear combinations of the original predictors. Similar to Whitening, the sequence of principal components (PCs) is the reduced matrix $V \in \mathbb{R}^{p \times q}$, which can also be expressed in terms of the input data as $W = XR \in \mathbb{R}^{n \times q}$, where $R = V$ is the PCA transformation matrix.

In both PCA and Whitening the transformation matrix $R$ has the constraints $R^TR = I$ in order to ensure orthogonality. Geometrically, the PCA transformation (without pruning) represents a rotation of the original coordinate system such that the new axes are the directions of maximum variability in the original data [63]. In Whitening transformation (without pruning) the data points lie at the vertices of a regular $(n - 1)$-dimensional simplex with all data points equidistant from each other [55].

- **Partial least squares (PLS)**: PLS is a wide class of supervised methods used for modeling relations between sets of observed variables by means of latent variables (also called the latent components). It has been defined for regression analysis, classification and dimension reduction [64]. PLS can be formulated as an optimization problem that aims to find a set of optimal weights vectors $r_i$ $(i = 1, \ldots, q)$ to maximize the covariance between the response variable $Y$ and the predictor variables $X$, it can be defined as [63]

$$
\mathbf{r}_i = \text{arg max}_T \text{Cov}(W, Y),
$$

s.t. $W^T W = I$.

(8)

where $W = XR \in \mathbb{R}^{n \times q}$ represents the $n$ observations by the $q$ PLS components. The maximum number of components $q$ is at most the rank of $X$. At each step $i$ $(i = 1, \ldots, q)$, the vector $r_i \in R = \{r_1, \ldots, r_q\}$ is estimated by regression in such a way that the PLS component, $w_i = W = [w_1, \ldots, w_q]$, has maximal example covariance with the response variable $Y$, subject to being uncorrelated with all previously constructed components. When PLS is used for classification problems, the vector of class label associated at $X$ can be expressed in terms of a response data matrix: $Y \in \mathbb{R}^{n \times c}$ dividing the number of class label in $c$ columns, each being each column a binary vector with 1 in the associated class label and 0 in other case. The technique is something of a cross between multiple linear regression and PCA. As a result, the PLS components are uncorrelated and ranked in the decreasing order and can be derived from [64]:

$$
X = WR^T + \text{error}_X,
$$

(9)

$$
Y = WS^T + \text{error}_Y,
$$

(10)

where $R \in \mathbb{R}^{p \times q}$ are the matrix of predictors loading vectors (or weights) used for constructing the $w_i$. PLS components, $S = [s_1, \ldots, s_q] \in \mathbb{R}^{n \times q}$ are the response loading matrix.

We use the MATLAB® (R2009b) Statistics Toolbox for the implementation of the previous compaction methods. They return the complete set of components along with the variance explained by each component. In all compaction methods we set $q$ as the minimal number of first components in which the cumulative sum of the respective variances is over 95% of the total sum of variances. This number was experimentally verified to conserve most discriminant information and, at the same time, reduces significantly the input dimension for the PP stage.

**3.2. PP stage**

This stage is responsible for the projection search over the compacted data. We follow the SPP approach in which the projection bases are obtained one-by-one in an iterative loop. Our current implementation of SPP replaces/modifies some components of the original SPP in order to improve performance. These components are described as follows:

- **Initial population**: This component is new in SPP and is intended to improve the convergence time and fitness of the subsequent PP optimization. The recent Candidate Projection Set (CPS) method [65] is used for that purpose, which obtains candidate projection bases using class boundary information. In the experiments, the size ($w$) of the initial population is set as $w = 3q$, formed by 50% of individuals from CPS and 50% created randomly.

- **Projection pursuit genetic algorithm (PPGA)**: This is the adopted PP optimizer, taken from our previous work [33]. Unlike the original GA optimizer, which uses binary encoding and canonical operators, PPGA uses a real encoding and a specialized crossover operator (called the Inner-outer Hypercone crossover). It was showed that this operator provides a good search capability for PP optimization in high dimensionalities [33]. At each generation, an offspring population is created with crossover. Mutation is not used in PPGA, since the crossover operator has high randomness and mutation showed to slow the convergence without any apparent improvement of the results. For mating selection we use tournament selection with tournament size equal to 3, a value that was adequate in the experimentation. The population for the next generation is formed by taking the best $w$ individuals of the joint offspring and current population. PPGA ends when the difference in mean fitness between two successive generations falls within a given precision $\Delta$ or a maximum number of generations $g$ is achieved. In the experiments we use $\Delta = 1e^{-4}$ and $g = 300$.

- **Deflation/inflation**: These procedures implement an efficient way to ensure orthogonality of the projection bases and, thus,
to capture distinct and complementary aspect of the data. The original structure removal procedure of SPP is replaced by these procedures, which are based on the work of Rodriguez-Martinez et al. [28]. Deflation prepares the search space for the next iteration and is executed after PPGA completion (except when the required number of bases is attained, in which case the loop is halted). Specifically at iteration $i$, PPGA finds the basis $b_i$ from residual data matrix $Z_i^{(n \times q-i+1)}$ (at first iteration, the residual data is the original whitened data $Y$). Deflation then uses $b_i$ to compute the set of basis $Q_i^{(q-i+1 \times q-i)}$ that defines the orthogonal complement of $b_i$. Then, the current residual data are projected onto that space to get the residual data for the next iteration: $Z_i^{(n \times i)} = Z_i^{(n \times q-i+1)} Q_i^{(q-i+1 \times q-i)}$. Note that after each deflation the dimension of the residual data decreases one unit, which means that the difficulty in finding bases decreases with the advance of the process (an advantage over the original structure removal procedure, which maintains a uniform difficulty along iterations). The inflation process is performed once all $m$ bases are obtained. As each base $b_i$ is defined for its corresponding residual data (of different dimensions), it is necessary to put all the bases in the original (compacted) space. Inflation, thus, constructs the projection matrix $A$ computing each base $a_i$ ($i \geq 2$) by multiplying all matrices $Q_j$, for all $j < i$, and then the resulting matrix with $b_i$ (operation known as base inflation).

Finally, once the PP stage ends and the projection matrix $A$ is returned, the overall projection matrix $P$ that maps the input data $X$ to the target space is computed, thus $P^{(n \times m)} = RA$.

4. Experimental evaluation

This section presents the experimental evaluation conducted over the proposed framework in order to determine its suitability in classification tasks of large $p$ small $n$ data.

4.1. Experimental setup

Eight public microarray datasets were used in the evaluation (Table 1). Fifteen configurations of the framework were evaluated, corresponding to all combinations of the compaction methods (Section 3.1) with the five PP indexes of Section 2.3. As evaluation metrics, we used the predictive accuracies of two popular classification methods: Linear Discriminant Analysis (LDA) [74] and $k$-Nearest Neighborhood (K-NN) (with $k=3$) [75] over the reduced data resulting of each configuration. These methods were chosen by their popularity, simplicity, speed and few parameters to set up. Also, they are deterministic (with the same training data we get the same classifier), which means that the final classification results are only affected by the quality of the projections. The implementations used for LDA and K-NN algorithms were those available in the Matlab Bioinformatics toolbox [76]. We additionally included in the evaluation eight popular dimension reduction methods: Locally Linear Embedding (LLE) [77], Neighborhood components analysis (NCA) [48], Partial Least Squares (Pls) [64], Sliced Inverse Regression (SIR) [78], the three compaction methods used in a standalone way, and two well-known feature selection methods (T-test Modified (T-testM) [79] and ReliefF [80]). We used the following implementations for these methods: DRtoolbox [81] for LLE and NCA, Weka implementation [82] for ReliefF, Zhou’s implementation [79] for T-testM, Matlab Statistics Toolbox for Pls (using plsregress function) and PCA (using princomp function).

Fig. 3 depicts the structure of an experiment replicate executed to assess the performance of the framework’s configurations in classification tasks.

![Fig. 3. Scheme of an experiment replicate executed to assess the performance of the framework’s configurations in classification tasks.](image)

Table 1

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Examples</th>
<th>Genes</th>
<th>Classes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brain Tumor 1</td>
<td>90</td>
<td>5921</td>
<td>5</td>
<td>[66]</td>
</tr>
<tr>
<td>Brain Tumor 2</td>
<td>50</td>
<td>10,368</td>
<td>4</td>
<td>[67]</td>
</tr>
<tr>
<td>Colon tumor</td>
<td>62</td>
<td>1999</td>
<td>2</td>
<td>[68]</td>
</tr>
<tr>
<td>DLBCL</td>
<td>77</td>
<td>5470</td>
<td>2</td>
<td>[69]</td>
</tr>
<tr>
<td>MLL</td>
<td>72</td>
<td>8677</td>
<td>3</td>
<td>[70]</td>
</tr>
<tr>
<td>Prostate Tumor</td>
<td>102</td>
<td>10,510</td>
<td>2</td>
<td>[71]</td>
</tr>
<tr>
<td>SRBCT</td>
<td>83</td>
<td>2039</td>
<td>4</td>
<td>[72]</td>
</tr>
<tr>
<td>TBC</td>
<td>96</td>
<td>4179</td>
<td>3</td>
<td>[73]</td>
</tr>
</tbody>
</table>
Fig. 4. Average 10-fold-CV accuracies obtained with 3NN classifiers on the eight datasets. Each figure corresponds to a different dataset and shows results for the three compaction methods (the three vertical regions) and the five PP indexes (curves) tested in the proposed framework. The abscissa axis represents the different target dimensions tested. Only Compaction curve presents the results for the corresponding compaction method without subsequent PP search. (a) Brain Tumor1 \( (n = 90, p = 5921, c = 5) \), (b) Brain Tumor2 \( (n = 50, p = 10368, c = 4) \), (c) Colon \( (n = 62, p = 1999, c = 2) \), (d) DLBCL \( (n = 77, p = 5470, c = 2) \), (e) MLL \( (n = 72, p = 8677, c = 3) \), (f) Prostate Tumor \( (n = 102, p = 10510, c = 2) \), (g) SRBCT \( (n = 83, p = 2039, c = 4) \), (h) TBC \( (n = 96, p = 4178, c = 3) \).
first target dimensions. The lower gains of the framework with respect to only Pls compaction are due to the better results of this latter (compared to PCA and Whitening compaction), since this makes use of the class information. However, as the results show, better compaction does not necessarily imply better results of the PP search. The behavior of lda index is especially interesting, since it generally manages to pack more relevant information for classification in fewer dimensions than the other PP indexes and it also tends to be stable along the target dimensionalities once it reach its maximum accuracy, which means that there is no degradation of the projections with more target dimensions once the discriminatory information is exhausted.

In order to have a more statistically meaningful picture of the previous results, we carried out a ranking analysis. This analysis consisted in ranking all the PP indexes together with the standalone compaction method (based on their associated 10-fold-CV average accuracies) for each particular combination of dataset, classification method, compaction method and target dimensionality. In cases of ties, we perform the correction suggested in [83], in which all methods in the tie are assigned a rank equivalent to the center of the positions they occupy in the ranking (for example, if two methods tied for second place, they are assigned to rank 2.5). Fig. 5 shows the average corrected rankings for each classifier, compaction method and target dimension (averaged across datasets and experiment runs). The dashed boxes in the plots frame all those PP indexes that have no significant statistical difference with respect to the best placed index in the corresponding dimension. The statistical significance was assessed following the procedure described in [83], in which the non-parametric Friedman’s test statistics is first employed to verify the existence of differences in performances of the indexes (we found that, for all dimensions, there exist significant differences at a significance level of 0.05). Next, we determined which methods are different from the top placed index (with whitening compaction), although the standalone whitening compaction tends to improve their relative positions with the increase of the target dimension. With the Pls compaction, we verify that the framework presents a significant advantage only at the first two and three target dimensions (with LDA and 3NN classifiers respectively). In the other dimensions, the standalone Pls method has no statistical difference from the best results of the framework (usually with lda index in 3NN classifier and nca index in LDA classifier). The gpc and Lp indexes always present the worst performances, very distant from the other indexes.

Fig. 6 shows the results of the ranking analysis conducted on the proposed framework and some popular dimension reduction methods. This time, each index-compaction framework configuration entered the analysis as an independent method, together with the eight dimension reduction methods indicated in Section 4.1, totaling 23 methods in the analysis. The results show that, with both classifiers and in almost all target dimensions, some configuration of the proposed framework is the best positioned in the ranking. The exception is in 10 dimensions with the LDA classifier, where the standalone Pls method is the first, but not significantly different from the configurations PCA-Bat and occasional nca (with whitening compaction), although the standalone whitening compaction tends to improve their relative positions with the increase of the target dimension. With the Pls compaction, we verify that the framework presents a significant advantage only at the first two and three target dimensions (with LDA and 3NN classifiers respectively). In the other dimensions, the standalone Pls method has no statistical difference from the best results of the framework (usually with lda index in 3NN classifier and nca index in LDA classifier). The gpc and Lp indexes always present the worst performances, very distant from the other indexes.

Finally, to get a sense of the computational cost, we included in Fig. 6 the time used to run all experiments relative to each configuration/method. It is observed that the framework is more time consuming than the other methods. This is due to the optimization stage, which is a stochastic process. Nevertheless, the best performing configuration PCA-lda has the lowest times among all configurations. In real situations, the framework would not be so delayed for a particular dataset (it is in the order of few seconds to get one dimension), since it is not necessary to perform all the experimentation presented here. In addition, it is worth to mention that the eight evaluated methods are well established and the toolboxes used for their implementations are computationally optimized.
5. Conclusion

Reducing the dimensionality of datasets with large number of features and few examples is a challenging problem. In this paper we described and evaluated a Projection Pursuit framework, which is intended to circumvent the difficulties associated with that kind of data and to facilitate the construction of classifiers. The framework is formed by two stages: the first stage performs a rapid compaction of the data, which is used by the second stage to perform a projection search, seeking to optimize a measure of interestingness (the PP index). In an experimental study, comprising eight public microarray datasets and various framework configurations (varying the compaction method, PP index and target dimensionality), we showed that the proposed framework can effectively reduce the dimensionality of datasets with large number of features into fewer dimensions. The framework, with Whitening or PCA compaction and PP indexes Ida, Bhattacharya and nca, was able to outperform eight well established dimension reduction methods in their ability to pack more discriminatory information into fewer dimensions.

We are planning to investigate more thoroughly the links between the properties of datasets and the performance of the different framework configurations, since we noted that the suitability of the configurations can vary across datasets. The aim would be the construction of a system that can select the best configuration of the framework for the problem at hand. Probably, a meta-learning approach [83] would be a good approach to this end. We also intend to apply the framework in other domains, like proteomics and astronomy datasets, where the imbalance between features and examples is even more aggravated.

Acknowledgments

We would like to thank CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) grant#151547/2013-0 and FAPESP (São Paulo Research Foundation) grant#2012/22295-0 for funding this study.

References

Soledad Espezua is currently a Postdoctoral fellow in the Department of Computer Science at the University of São Paulo, Brazil. Her major research focuses are Evolutionary Computation, Bioinformatics, Machine Learning, Data Mining and Meta-learning. She received his Ph.D. and M.Sc. degrees in electrical engineering from the University of São Paulo, Brazil, in 2008 and 2013 respectively.

Edwin Villanueva received his M.Sc. and Ph.D. degrees in electrical engineering from the University of São Paulo, Brazil, in 2007 and 2012 respectively. He is currently a Postdoctoral fellow in the Department of Computer Science at the University of São Paulo, Brazil. His main interests are Machine Learning, Data Mining, Meta-learning, Bioinformatics, Evolutionary Computation, Bio-inspired Computing, Optimization and Probabilistic Graphical Models.

Carlos D. Maciel is an Associate Professor in Statistical Signal Processing and Pattern Recognition at the Department of Electrical Engineering, University of São Paulo (USP) at São Carlos, Brazil. He received his B.Sc. from the Military Institute of Engineering (IME), Brazil, in 1989 and Ph.D. degree from the Federal University of Rio de Janeiro (UFRJ), Brazil, in 2000.

André Carvalho is a Full Professor in the Department of Computer Science at the University of São Paulo, Brazil. He received his Ph.D. in Electronics from the University of Kent, UK, in 1994. He co-authored one textbook on neural networks and one on artificial intelligence, both in Portuguese. He has several publications in books, refereed journals and conferences. He is in the editorial board and was a guest editor for international journals and general program chair of several national and international conferences. He gave invited talks and won the best paper awards in national and international conferences. He collaborates with researchers from Brazil and abroad in several research projects.