Feature Selection for Unsupervised and Supervised Inference: the Emergence of Sparsity in a Weighted-based Approach*

Lior Wolf and Amnon Shashua
School of Computer Science and Engineering,
The Hebrew University,
Jerusalem 91904, Israel
e-mail: {shashua,lwolf}@cs.huji.ac.il

Abstract

The problem of selecting a subset of relevant features in a potentially overwhelming quantity of data is classic and found in many branches of science. Examples in computer vision, text processing and more recently bio-informatics are abundant. In text classification tasks, for example, it is not uncommon to have 10^4 to 10^7 features of the size of the vocabulary containing word frequency counts, with the expectation that only a small fraction of them are relevant. Typical examples include the automatic sorting of URLs into a web directory and the detection of spam email.

In this work we present a definition of "relevancy" based on spectral properties of the Laplacian of the features' measurement matrix. The feature selection process is then based on a continuous ranking of the features defined by a least-squares optimization process. A remarkable property of the feature relevance function is that sparse solutions for the ranking values naturally emerge as a result of a "biased non-negativity" of a key matrix in the process. As a result, a simple least-squares optimization process converges onto a sparse solution, i.e., a selection of a subset of features which form a local maxima over the relevance function. The feature selection algorithm can be embedded in both unsupervised and supervised inference problems and empirical evidence show that the feature selections typically achieve high accuracy even when only a small fraction of the features are relevant.

1. Introduction

As visual recognition, text classification, speech recognition and more recently bio-informatics aim to address larger and more complex tasks the problem of focusing on the most relevant information in a potentially overwhelming quantity of data has become increasingly important. Examples from computer vision, text processing and Genomics are abundant. For instance, in visual recognition the pixel values themselves often form a highly redundant set of features; methods using an "over-complete" basis of features for recognition are gaining popularity [23], and recently methods relying on abundance of simple efficiently computable features of which only a fraction of are relevant were proposed for face detection [30] — and these are only few examples from the visual recognition literature. In text classification tasks it is not uncommon to have 10^4 to 10^7 features of the size of the vocabulary containing word frequency counts, with the expectation that only a small fraction of them are relevant [18]. Typical examples include the automatic sorting of URLs into a web directory and the detection of spam email. In Genomics, a typical example is gene selection from micro-array data where the features are gene expression coefficients corresponding to the abundance of cellular mRNA taken from sample tissues. Typical applications include separating tumor from normal cells or discovery of new subclasses of Cancer cells based on the gene expression profile. Typically the number of samples (expression patterns) is less than 100 and the number of features (genes) in the raw data ranges from 5000 to 50000. Among the overwhelming number of genes only a small fraction is relevant for the classification of tissues whereas the expression level of many other genes may be irrelevant to the distinction between tissue classes — therefore, identifying highly relevant genes from the data is a basic problem in the analysis of expression data.

From a practical perspective, large amounts of irrelevant features affects learning algorithms at three levels. First, most learning problems do not scale well with the growth of irrelevant features — in many cases the number of training examples grows exponentially with the number of irrelevant features [16]. Second, is a substantial degradation of classification accuracy for a given training set size [1, 13]. The accuracy drop affects also advanced learning algorithms that generally scale well with the dimension of the feature space such as the Support Vector Machines

*Submitted to Journal of Machine Learning Research (JMLR), Sep. 2003. Short version of this paper was presented at the ICCV, Nice France, Oct. 2003.

(SVM) as recently observed in [32]. The third aspect has to do with the run time of the learning algorithm on test instances. In most learning problems the classification process is based on inner-products between the features of the test instance and stored features from the training set, thus when the number of features is overwhelmingly large the run-time of the learning algorithm becomes prohibitively large for real time applications, for example. Another practical consideration is the problem of determining how many relevant features to select. This is a difficult problem which is hardly ever addressed in the literature and consequently it is left to the user to choose manually the number of features. Finally, there is an issue of whether one is looking for the *minimal* set of (relevant) features, or simply a possibly redundant but relevant set of features.

The potential benefits of feature selection include, first and foremost, better accuracy of the inference engine and improved scalability (defying the curse of dimensionality). Secondary benefits include better data visualization and understanding, reduce measurement and storage requirements, and reduce training and inference time. Blum and Langley [4] in a survey article distinguish between three types of methods: *Embedded*, *Filter* and *Wrapper* approaches. The filter methods apply a preprocess which is independent of the inference engine (a.k.a the predictor or the classification/inference engine) and select features by ranking them with correlation coefficients or make use of mutual information measures. The Embedded and Wrapper approaches construct and select feature subsets that are useful to build a good predictor. The issue being the notion of *relevancy*, i.e., what constitutes a good set of features. The modern approaches, therefore, focus on building feature selection algorithms in the context of a *specific* inference engine. For example, [32, 5] use the Support Vector Machine (SVM) as a subroutine (wrapper) in the feature selection process with the purpose of optimizing the SVM accuracy on the resulting subset of features. These wrapper and embedded methods in general are typically computationally expensive and often criticized as being "brute force". Further details on relevancy versus usefulness of features and references to historical and modern literature on feature selection can be found in the survey papers [4, 15, 11].

In this paper the inference algorithm is not employed directly in the feature selection process but instead general properties are being gathered which indirectly indicate whether a feature subset would be appropriate or not. Specifically, we use clustering as the predictor and use spectral properties of the candidate feature subset to guide the search. This leads to a "direct" approach where the search is conducted on the basis of optimizing desired spectral properties rather than on the basis of explicit clustering and prediction cycles. The search is conducted by the solution of a least-squares optimization function using a weighting scheme for the ranking of features. *A remarkable property of the energy function is that "sparse" solutions for the weights naturally emerge as a result of a "biased non-negativity" of a key matrix in the process*. The algorithm, called $Q-\alpha$, is iterative, very efficient and achieves remarkable performance on a variety of experiments we have conducted.

There are many benefits of our approach: First, we avoid the expensive computations associated with Embedded and Wrapper approaches, yet still make use of a predictor to guide the feature selection. Second, the framework can handle both unsupervised and supervised inference within the same framework and handle any number of classes. In other words, since the underlying inference is based on clustering class labels are not necessary, but on the other hand, when class labels are provided they can be used by the algorithm to provide better feature selections. Third, the algorithm is couched within a least-squares framework — and least-squares problems are the best understood and easiest to handle. Finally, the performance (accuracy) of the algorithm is remarkable.

2 Algebraic Definition of Relevancy

A key issue in designing a feature selection algorithm in the context of an inference is defining the notion of relevancy. Definitions of relevancy proposed in the past [4, 15] lead naturally to a explicit enumeration of feature subsets which we would like to avoid. Instead, we take an algebraic approach and measure the relevance of a subset of features against its influence on the cluster arrangement of the data points with the goal of introducing an energy function which receives its optimal value on the desired feature selection. We will consider two measures of relevancy based on spectral properties where the first is based on the Standard spectrum and the second on the Laplacian spectrum.

2.1 The Standard Spectrum

Consider a $n \times q$ data set M consisting of q samples (columns) over n-dimensional feature space R^n representing n features $x_1, ..., x_n$ over q samples. Let the row vectors of M be denoted by $\mathbf{m}_1^\top, ..., \mathbf{m}_n^\top$ pre-processed such that each row is centered around zero and is of unit L_2 norm $\|\mathbf{m}_i\| = 1$. Let $\mathcal{S} = \{x_{i_1}, ..., x_{i_l}\}$ be a subset of (relevant) features from the set of n features and let $\alpha_i \in \{0, 1\}$ be the indicator value associated with feature x_i, i.e., $\alpha_i = 1$ if $x_i \in \mathcal{S}$ and zero otherwise. Let A_s be the corresponding *affinity* matrix whose (i, j) entries are the inner-product between the i'th and j'th data points restricted to the selected coordinate features, i.e., $A_s = \sum_{i=1}^{n} \alpha_i \mathbf{m}_i \mathbf{m}_i^\top$ where $\mathbf{m}_i \mathbf{m}_i^\top$ is the rank-1 matrix defined by the outer-product between $\mathbf{m}_i$ and itself. Finally, let Q_s be a $q \times k$ matrix whose

columns are the first k eigenvectors of A_s associated with the leading (highest) eigenvalues $\lambda_1 \geq ... \geq \lambda_k$.

We define "relevancy" as directly related to the clustering quality of the data points restricted to the selected coordinates. In other words, we would like to measure the quality of the subset $\mathcal{S}$ in terms of cluster coherence of the first k clusters, i.e., we make a direct linkage between cluster coherence of the projected data points and relevance of the selected coordinates.

We measure cluster coherence by analyzing the (standard) spectral properties of the affinity matrix A_s. Considering the affinity matrix as representing weights in an undirected graph, it is known that maximizing the quadratic form $\mathbf{x}^\top A_s \mathbf{x}$ where $\mathbf{x}$ is constrained to lie on the standard simplex ($\sum x_i = 1$ and $x_i \geq 0$) provides the identification of the maximal *clique* of the (unweighted) graph [22, 9], or the maximal "dominant" subset of vertices of the weighted graph [24]. Likewise there is evidence (motivated by finding cuts in the graph) that solving the quadratic form above where $\mathbf{x}$ is restricted to the unit sphere provides cluster membership information (cf. [20, 31, 25, 28, 6, 7]). In this context, the eigenvalue (the value of the quadratic form) represents the cluster coherence. In the case of k clusters, the highest k eigenvalues of A_s represent the corresponding cluster coherences and the components of an eigenvector represent the coordinate (feature) participation in the corresponding cluster. The eigenvalues decrease as the interconnections of the points within clusters get sparser (see [26]). Therefore, we define the relevance of the subset $\mathcal{S}$ as:

$$\begin{aligned} rel(\mathcal{S}) &= trace(Q_s^\top A_s^\top A_s Q_s) \\ &= \sum_{r,s} \alpha_{i_r}\alpha_{i_s}(\mathbf{m}_{i_r}^\top \mathbf{m}_{i_s})\mathbf{m}_{i_r}^\top Q_s Q_s^\top \mathbf{m}_{i_s} \\ &= \sum_{j=1}^{k} \lambda_j^2, \end{aligned}$$

where λ_j are the leading eigenvalues of A_s. Note that the proposed measure of relevancy handles interactions among features up to a second order. To conclude, achieving a high score on the combined energy of the first k eigenvalues of A_s indicate (although indirectly) that the q input points projected onto the l-dimensional feature space are "well clustered" and that in turn suggests that $\mathcal{S}$ is a relevant subset of features.

Rather than enumerating all possible feature subsets $\mathcal{S}$ and ranking them according to the value of $rel(\mathcal{S})$ we consider the prior weights $\alpha_1, ..., \alpha_n$ as unknown *real numbers* and define the following optimization function:

Definition 1 (Relevant Features Optimization) *Let M be an $n \times q$ input matrix with rows $\mathbf{m}_1^\top, ..., \mathbf{m}_n^\top$. Let $A_\alpha = \sum_{i=1}^n \alpha_i \mathbf{m}_i \mathbf{m}_i^\top$ for some unknown scalars $\alpha_1, ..., \alpha_n$. The weight vector $\alpha = (\alpha_1, ..., \alpha_n)^\top$ and the orthonormal $q \times k$ matrix Q are determined at the maximal point of the following optimization problem:*

$$\max_{Q,\alpha_i} trace(Q^\top A_\alpha^\top A_\alpha Q) \quad (1)$$

$$subject\ to \quad \sum_{i=1}^{n} \alpha_i^2 = 1,\ \ Q^\top Q = I$$

Note that the optimization function does not include the inequality constraint $\alpha_i \geq 0$ and neither a term for "encouraging" a sparse solution of the weight vector α — both of which are necessary for a "feature selection". As will be shown later in Section 4, the sparsity and positivity conditions are implicitly embedded in the nature of the optimization function and therefore "emerge" naturally with the optimal solution.

Note also that it is possible to maximize the gap $\sum_{i=1}^k \lambda_i^2 - \sum_{j=k+1}^q \lambda_j^2$ by defining $Q = [Q_1|Q_2]$ where Q_1 contains the first k eigenvectors and Q_2 the remaining $q-k$ eigenvectors (sorted by decreasing eigenvalues) and the criterion function (1) would be replaced by:

$$\max_{Q=[Q_1|Q_2],\alpha_i} trace(Q_1^\top A_\alpha^\top A_\alpha Q_1) - trace(Q_2^\top A_\alpha^\top A_\alpha Q_2).$$

We will describe in Section 3 an efficient algorithm for finding a local maximum of the optimization (1) and later address the issue of sparsity and positivity of the resulting weight vector α. The algorithms are trivially modified to handle the gap maximization criterion and those will not be further elaborated here. We will describe next the problem formulation using an additive normalization (the Laplacian) of the affinity matrix.

2.2 The Laplacian Spectrum

Given the standard affinity matrix A, consider the Laplacian matrix: $L = A - D + d_{max} I$ where D is a diagonal matrix $D = diag(\sum_j a_{ij})$ and d_{max} is a scalar larger or equal to the maximal element of D[1]. The matrix L normalizes A in an additive manner and there is much evidence to support such a normalization both in the context of graph partitioning [21, 12] and spectral clustering [31, 20].

It is possible to reformulate the feature selection problem (1) using the Laplacian as follows. Let $A_i = \mathbf{m}_i \mathbf{m}_i^\top$ and $D_i = diag(\mathbf{m}_i \mathbf{m}_i^\top \mathbf{1})$. We define $L_\alpha = \sum_i \alpha_i L_i$ where $L_i = A_i - D_i + d_{max} I$. We have, therefore:

$$L_\alpha = A_\alpha - D_\alpha + (\sum_i \alpha_i) d_{max} I,$$

where $D_\alpha = diag(A_\alpha^\top \mathbf{1})$. Note that since α is a unit norm vector then $\sum_i \alpha_i > 1$. The feature selection problem is identical to (1) where L_α replaces A_α.

[1] Note that in applications of algebraic graph theory the Laplacian is defined as $D - A$. The reason for the somewhat different definition is that we wish to maintain the order of eigenvectors as in those of A (where the eigenvectors associated with the largest eigenvalues come first).

3 An Efficient Algorithm

We wish to find an optimal solution for the non-linear problem (1). We will focus on the Standard spectrum matrix A_α and later discuss the modifications required for L_α. If the weight vector α is known, then the solution for the matrix Q is readily available by employing a Singular Value Decomposition (SVD) of the symmetric (and positive definite) matrix A_α. Conversely, if Q is known then α is readily determined as shown next. We already saw that

$$\begin{aligned} trace(Q^\top A_\alpha^\top A_\alpha Q) &= \sum_{i,j} \alpha_i \alpha_j (\mathbf{m}_i^\top \mathbf{m}_j) \mathbf{m}_i^\top Q Q^\top \mathbf{m}_j \\ &= \alpha^\top G \alpha \end{aligned}$$

where $G_{ij} = (\mathbf{m}_i^\top \mathbf{m}_j)\mathbf{m}_i^\top Q Q^\top \mathbf{m}_j$ is symmetric and positive definite. The optimal α is therefore the solution of the optimization problem:

$$\max_\alpha \alpha^\top G \alpha \quad subject\ to\ \alpha^\top \alpha = 1,$$

which results in α being the leading eigenvector of G, i.e., the one associated with its largest eigenvalue. A possible scheme, guaranteed to converge to a local maxima, is to start with some initial guess for α and iteratively interleave the computation of Q given α and the computation of α given Q until convergence. We refer to this scheme as the **Basic $Q-\alpha$ Method**.

A more advanced scheme with superior convergence rate and more importantly accuracy of results (based on empirical evidence) is to embed the computation of α within the "orthogonal iteration" [10] cycle for computing the largest k eigenvectors, described below:

Definition 2 (Standard Power-Embedded $Q-\alpha$ Method) *Let M be an $n \times q$ input matrix with rows $\mathbf{m}_1^\top, ..., \mathbf{m}_n^\top$, and some orthonormal $q \times k$ matrix $Q^{(0)}$, i.e., ${Q^{(0)}}^\top Q^{(0)} = I$. Perform the following steps through a cycle of iterations with index $r = 1, 2, ...$*

1. *Let $G^{(r)}$ be a matrix whose (i,j) components are $(\mathbf{m}_i^\top \mathbf{m}_j)\mathbf{m}_i^\top Q^{(r-1)} {Q^{(r-1)}}^\top \mathbf{m}_j$.*
2. *Let $\alpha^{(r)}$ be the largest eigenvector of $G^{(r)}$.*
3. *Let $A^{(r)} = \sum_{i=1}^n \alpha_i^{(r)} \mathbf{m}_i \mathbf{m}_i^\top$.*
4. *Let $Z^{(r)} = A^{(r)} Q^{(r-1)}$.*
5. *$Z^{(r)} \xrightarrow{QR} Q^{(r)} R^{(r)}$.*
6. *Increment index r and go to step 1.*

The method is very efficient and achieves very good performance (accuracy). Note that steps 4,5 of the algorithm consist of the "orthogonal iteration" module, i.e., if we were to repeat steps 4,5 *only* we would converge onto the eigenvectors of $A^{(r)}$. However, note that the algorithm does not repeat steps 4,5 in isolation and instead recomputes the weight vector α (steps 1,2,3) before applying another cycle of steps 4,5. We show below that the recomputation of α does not alter the convergence property of the orthogonal iteration scheme, thus the overall scheme converges to a local maxima:

Proposition 1 (Convergence of Power-Embedded $Q-\alpha$) *The Power Embedded $Q-\alpha$ method convergence to a local maxima of the criterion function (1).*

Proof: We will prove the claim for the case $k = 1$, i.e. the scheme optimizes over the weight vector α and the largest eigenvector $\mathbf{q}$ of A_α.

Because the computation of α is analytic (the largest eigenvector of G), it is sufficient to show that the computation of $\mathbf{q}$ monotonically increases the criterion function. It is therefore sufficient to show that:

$$\mathbf{q}^{(r)} A^2 \mathbf{q}^{(r)} \geq \mathbf{q}^{(r-1)} A^2 \mathbf{q}^{(r-1)}, \tag{2}$$

for all symmetric matrices A. Since steps 4,5 of the algorithm are equivalent to the step:

$$\mathbf{q}^{(r)} = \frac{A\mathbf{q}^{(r-1)}}{\|\mathbf{q}^{(r-1)}\|^2},$$

we can substitute the right hand side into (2) and obtain the condition:

$$\mathbf{q}^\top A^2 \mathbf{q} \leq \frac{\mathbf{q}^\top A^4 \mathbf{q}}{\mathbf{q}^\top A^2 \mathbf{q}}, \tag{3}$$

which needs to be shown to hold for all symmetric matrices A and unit vectors $\mathbf{q}$. Let $\mathbf{q} = \sum_i \gamma_i \mathbf{v}_i$ be represented with respect to the orthonormal set of eigenvectors $\mathbf{v}_i$ of the matrix A. Then, $A\mathbf{q} = \sum_i \gamma_i \lambda_i \mathbf{v}_i$ where λ_i are the corresponding eigenvalues. Since $\mathbf{q}^\top A^2 \mathbf{q} \geq \mathbf{0}$, it is sufficient to show that: $\|A\mathbf{q}\|^4 \leq \|A^2\mathbf{q}\|^2$, or equivalently:

$$(\sum_i \gamma_i^2 \lambda_i^2)^2 \leq \sum_i \gamma_i^2 \lambda_i^4. \tag{4}$$

Let $\mu_i = \lambda_i^2$ and let $f(x) = x^2$. We then have:

$$f(\sum_i \gamma_i^2 \mu_i) \leq \sum_i \gamma_i^2 f(\lambda_i^2),$$

which follows from convexity of $f(x)$ and the fact that $\sum_i \gamma_i^2 = 1$. □

A faster converging algorithm is possible by employing the "Ritz" acceleration [10] to the basic power method as follows:

Definition 3 ($Q-\alpha$ with Ritz Acceleration) *Let M be an $n \times q$ input matrix with rows $\mathbf{m}_1^\top, ..., \mathbf{m}_n^\top$, and some orthonormal $n \times k$ matrix $Q^{(0)}$, i.e., ${Q^{(0)}}^\top Q^{(0)} = I$. Perform the following steps through a cycle of iterations with index $r = 1, 2, ...$*

1. *Let $G^{(r)}, \alpha^{(r)}$ and $A^{(r)}$ be defined as in the Standard Power-Embedded $Q-\alpha$ algorithm.*
2. $Z^{(r)} = A^{(r)}Q^{(r-1)}$.
3. $Z^{(r)} \xrightarrow{QR} \bar{Q}^{(r)}R^{(r)}$.
4. *Let $\bar{G}^{(r)}$ be a matrix whose (i,j) components are $\mathbf{m}_i^\top \bar{Q}^{(r)^\top} \bar{Q}^{(r)} \mathbf{m}_j$.*
5. *Recompute $\alpha^{(r)}$ as the largest eigenvector of $\bar{G}^{(r)}$, and recompute $A^{(r)}$ accordingly.*
6. *Let $S^{(r)} = \bar{Q}^{(r)^\top} A^{(r)} \bar{Q}^{(r)}$.*
7. *Perform SVD on $S^{(r)}$: $[U^{(r)^\top} S^{(r)} U^{(r)}] = svd(S^{(r)})$.*
8. $Q^{(r)} = \bar{Q}^{(r)} U^{(r)}$.
9. *Increment index r and go to step 1.*

The $Q-\alpha$ algorithm for the Laplacian spectrum L_α follows the Standard spectrum with the necessary modifications described below.

Definition 4 (Laplacian Power-Embedded $Q-\alpha$ Method) *In addition to the definition of the Standard method, let $d_i = \max diag(\mathbf{m}_i\mathbf{m}_i^\top)$ and $L_i^{(0)} = \mathbf{m}_i\mathbf{m}_i^\top - diag(\mathbf{m}_i\mathbf{m}_i^\top \mathbf{1}) + d_i I$. Perform the following steps with index $r = 1, 2, ...$*

1. *Let $F^{(r)}$ be a matrix whose (i,j) components are $trace(Q^{(r-1)^\top} L_i^{(r-1)^\top} L_j^{(r-1)} Q^{(r-1)})$.*
2. *Let $\alpha^{(r)}$ be the largest eigenvector of $F^{(r)}$.*
3. *Let $d^{(r)} = (\max diag(\sum_{i=1}^n \alpha_i^{(r)} \mathbf{m}_i\mathbf{m}_i^\top))/(\sum_{i=1}^n \alpha_i)$*
4. *For each i let $L_i^{(r)} = \mathbf{m}_i\mathbf{m}_i^\top - diag(\mathbf{m}_i\mathbf{m}_i^\top \mathbf{1}) + d^{(r)} I$*
5. *Let $L^{(r)} = \sum_{i=1}^n \alpha_i^{(r)} L_i^{(r)}$.*
6. *Let $Z^{(r)} = L^{(r)} Q^{(r-1)}$.*
7. $Z^{(r)} \xrightarrow{QR} Q^{(r)} R^{(r)}$.
8. *Increment index r and go to step 1.*

3.1 The Supervised Case

The $Q-\alpha$ algorithms and the general approach can be extended to handle data with class labels. One of the strengths of our approach is that the feature selection method can handle both unsupervised and supervised data sets. In a nutshell, the supervised case is handled as follows. Given c classes, we are given c data matrices $M^l, l = 1, ..., c$, each of size $n \times q^l$.

Definition 5 (Supervised Relevant Features Optimization) *Let M^l be an $n \times q^l$ input matrices with rows $\mathbf{m}_1^{l\top}, ..., \mathbf{m}_n^{l\top}$. Let $A_\alpha^{gh} = \sum_{i=1}^n \alpha_i \mathbf{m}_i^g \mathbf{m}_i^{h\top}$ for some unknown scalars $\alpha_1, ..., \alpha_n$. The weight vector $\alpha = (\alpha_1, ..., \alpha_n)^\top$ and the orthonormal $q^h \times k^{gh}$ matrices Q^{gh} are determined at the maximal point of the following optimization problem:*

$$\max_{Q^{gh}, \alpha_i} \sum_l trace(Q^{ll\top} A_\alpha^{ll\top} A_\alpha^{ll} Q^{ll}) - \gamma \sum_{g \neq h} trace(Q^{gh\top} A_\alpha^{gh\top} A_\alpha^{gh} Q^{gh}) \quad (5)$$
$$subject\ to\ \sum_{i=1}^n \alpha_i^2 = 1,\ \ Q^{gh\top} Q^{gh} = I$$

Where the weight γ and the parameters k^{gh} are determined manually (see below).

The criterion function seeks a weight vector α such that the resulting affinity matrix of all the data points (sorted) would be semi-block-diagonal, i.e., high inter-class eigenvalue energy and low intra-class energy. Therefore, we would like to minimize of the intra-class eigenvalue energy $trace(Q^{gh\top} A_\alpha^{gh\top} A_\alpha^{gh} Q^{gh})$ (off-block-diagonal blocks) and maximize the inter-class eigenvalue energy $trace(Q^{ll\top} A_\alpha^{ll\top} A_\alpha^{ll} Q^{ll})$. The parameters k^{gh} control the complexity of each affinity matrix. A typical choice of the parameters would be $k^{gh} = 2$ when $g = h$, $k^{gh} = 1$ otherwise, and $\gamma = 0.5$.

The solution to the optimization function follows step-by-step the $Q-\alpha$ algorithms. At each cycle Q^{gh} is computed using the current estimates A_α^{gh} and α is optimized by maximizing the expression:

$$\sum_l \alpha^\top G^{ll} \alpha - \gamma \sum_{g \neq h} \alpha^\top G^{gh} \alpha = \alpha^\top \mathcal{G} \alpha$$

where $G_{ij}^{gh} = (\mathbf{m}_i^{g\top} \mathbf{m}_j^g) \mathbf{m}_i^{h\top} Q^{gh\top} Q^{gh} \mathbf{m}_j^h$ and $\mathcal{G} = \sum_l G^{ll} - \gamma \sum_{g \neq h} G^{gh}$. We analyze next the properties of the unsupervised $Q-\alpha$ algorithm with regard to sparsity and positivity of the weight vector α and then proceed to experimental analysis.

4 Sparsity and Positivity of α

The optimization criteria (1) is formulated as a least-squares problem and as such there does not seem to be any apparent guarantee that the weights $\alpha_1, ..., \alpha_n$ would come out *non-negative* (same sign condition), and in particular *sparse* when there exists a sparse solution (i.e., there is a relevant subset of features). These two conditions are critical for the compatibility of the algorithm for feature selection. The positivity is required for making the variables α_i serve as weights, and the sparsity for the feature selection itself — otherwise the scheme would produce some feature combination rather than feature selection.

Typically, these conditions should be specifically presented into the optimization criterion one way or the other. The possible means for doing so include introduction of inequality constraints, use of L_0 or L_1 norms, adding specific terms to the optimization function to "encourage" sparse solutions or use a multiplicative scheme of iterations which

preserve the sign of the variables throughout the iterations (for a very partial list see [23, 14, 17, 29]). It is therefore somewhat surprising, if not remarkable, that the least-squares formulation of the feature selection problem could consistently converge onto same-sign and sparse solutions.

We will first address the issue of positivity of the weight vector α by analyzing the structure of the matrix G. Specifically, since α comes out as the first eigenvector of G there is a direct relationship between the "positivity" of G, i.e., the likelihood that the entries G_{ij} are non-negative, and the positivity of α. The sparsity issue will then be addressed by first defining what we mean by a "sparse" solution. Since the optimization does not directly enforce vanishing weight variables α_i we cannot expect a true sparse solution. Nevertheless, the weights α_i tend to concentrate around a relatively small number of rows (features) and have very low values for the remaining indices. We therefore define and derive a "sparsity gap" which measures the ratio between the average value of the weights associated with relevant features and the average value of the remaining weights as a function of the ratio between the number of relevant features and the total number of features. With the risk of abusing standard terminology, we will refer to the property of having the weight vector concentrate its (high) values around a small number of coordinates as a sparsity feature — and leave the precise definition to Section 4.2.

Before we proceed with the technical issues, it is worthwhile to make a qualitative argument (which was the basis of developing this approach to begin with) as to the underlying reason for sparsity. Consider rewriting the optimization criterion (1) by an equivalent criterion:

$$\min_{\alpha,Q} \left\{ \|A_\alpha - QQ^\top A_\alpha\|_F^2 - \|A_\alpha\|_F^2 \right\} \qquad (6)$$

where $\|\cdot\|_F^2$ is the square Frobenious norm of a matrix defined as the sum of squares of all entries of the matrix. The first term of (6) measures the distance between the columns of A_α and the projection of those columns onto a k-dimensional subspace (note that $QQ^\top$ is a projection matrix). This term receives a low value if indeed A_α has a small (k) number of dominant eigenvectors, i.e., the spectral properties of the feature subset represented by A_α are indicative to a good clustering score. Since $A_\alpha = \sum_i \alpha_i \mathbf{m}_i \mathbf{m}_i^\top$ is represented by the sum of rank-1 matrices one can combine only a *small* number of them if the first term is desired to be small. The second term (which may be viewed also as a regularization term) encourages addition of more rank-1 matrices to the sum provided they are *redundant*, i.e., are already spanned by the previously selected rank-1 matrices. This emphasizes the point made in Section 2 that the feature selection scheme looks for relevant features but not necessarily the minimal set of relevant features. To summarize, from a qualitative point of view the selection of values for the weights α_i is directly related to the rank of the affinity matrix A_α which should be small if indeed A_α arises from a clustered configuration of data points. A uniform spread of values α_i would result in a high rank for A_α, thus the criteria function encourages a non-uniform (i.e., sparse) spread of weight values. This argument is presented here to facilitate clarity of the approach and should not be taken as a proof for sparsity. The positivity and sparsity issues are approached in the sequel from a different angle which provides a more analytic handle to the underlying search process than the qualitative argument above.

4.1 Positivity of α

The key for the emergence of a sparse and positive α has to do with the way the entries of the matrix G are defined. Recall that $G_{ij} = (\mathbf{m}_i^\top \mathbf{m}_j)\mathbf{m}_i^\top QQ^\top \mathbf{m}_j$ and that α comes out as the leading eigenvector of G (at each iteration). If G were to be non-negative (and irreducible), then from the Perron-Frobenious theorem the leading eigenvector is guaranteed to be non-negative (or same-sign). However, this is not the case and G in general has negative terms as well as positive ones. Nevertheless, a closer look reveals that each entry of G consists of a sum of products of three inner-products:

$$G_{ij} = \sum_{l=1}^{k} (\mathbf{m}_i^\top \mathbf{q}_l)(\mathbf{m}_j^\top \mathbf{q}_l)(\mathbf{m}_i^\top \mathbf{m}_j).$$

In general, a product of the form $f = (\mathbf{a}^\top \mathbf{b})(\mathbf{a}^\top \mathbf{c})(\mathbf{b}^\top \mathbf{c})$, where $\|\mathbf{a}\| = \|\mathbf{b}\| = \|\mathbf{c}\| = 1$ satisfies $-1 < f \leq 1$ where $f = 1$ when $\mathbf{a} = \mathbf{b} = \mathbf{c}$. Since $f > -1$ there is an asymmetry on the expected value of f, i.e., the expected values of the entries of G are biased towards a positive value — and we should expect a bias towards a positive leading eigenvector of G. In the context of deriving the probability that the leading eigenvector of G is positive we will address the following three questions:

- What is the minimal value of $f = (\mathbf{a}^\top \mathbf{b})(\mathbf{a}^\top \mathbf{c})(\mathbf{b}^\top \mathbf{c})$ when $\mathbf{a}, \mathbf{b}, \mathbf{c}$ vary over the n-dimensional unit hypersphere? We will show that the $-1/8 \leq f \leq 1$.
- Given a uniform sampling of the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ over the n-dimensional unit hypersphere, what is the mean μ and variance σ^2 of f? The result that $-1/8 \leq f \leq 1$ suggests that $\mu > 0$.
- Given that $G_{ij} \sim N(\mu, \sigma^2)$, what is the probability (as a function of n) that the first eigenvector of G is strictly non-negative (same sign)?

We will show that in the worst case of a random data matrix M, the probability of the first eigenvector α of G to be strictly non-negative rapidly approaches 1 with n. We

will then focus our attention to the issue of sparsity of the recovered weight vector α.

Proposition 2 *The minimal value of* $f = (\mathbf{a}^\top\mathbf{b})(\mathbf{a}^\top\mathbf{c})(\mathbf{b}^\top\mathbf{c})$ *where* $\mathbf{a},\mathbf{b},\mathbf{c} \in R^n$ *are defined over the unit hypersphere is* $-1/8$.

Proof: Let $\mathbf{e}_1,\mathbf{e}_2,\mathbf{e}_3 \in R^n$ be three units vectors $(1,0,...,0)$, $(0,1,0,...,0)$ and $(0,0,1,0,...,0)$. The parameterization of 3 points on the unit hypersphere takes the form:

$$[\mathbf{a},\mathbf{b},\mathbf{c}] = [\mathbf{e}_1,\mathbf{e}_2,\mathbf{e}_3]\begin{bmatrix} 1 & \cos(\beta) & \cos(\gamma_1) \\ 0 & \sin(\beta) & \sin(\gamma_1)\cos(\gamma_2) \\ 0 & 0 & sin(\gamma_1)sin(\gamma_2) \end{bmatrix} \quad (7)$$

Setting the partial derivatives of

$$f = \cos(\beta)\cos(\gamma_1)\left(\cos(\beta)\cos(\gamma_1) + \sin(\beta)\sin(\gamma_1)\cos(\gamma_2)\right) \quad (8)$$

with respect to β,γ_1,γ_2 to zero and solving for the extremum points (using a symbolic solver such as Maple) yields 36 solutions for the triplet $(\beta,\gamma_1,\gamma_2)$. When these solutions are substituted in expression (8) the values $f = \{-1/8, 0, 1\}$ appear with multiplicity $\{16, 14, 4\}$, respectively. □

Proposition 3 *The expected value of* $f = (\mathbf{a}^\top\mathbf{b})(\mathbf{a}^\top\mathbf{c})(\mathbf{b}^\top\mathbf{c})$ *where* $\mathbf{a},\mathbf{b},\mathbf{c} \in R^n$ *are uniformly sampled over the unit hypersphere is* $\mu = \frac{1}{6}$ *with a standard deviation (s.t.d)* $\sigma = \sqrt{2}\frac{1}{6}$.

Proof: Let the parameterization of 3 points on the unit hypersphere be described as in (7), where $0 \leq \beta \leq 2\pi$, $-1 \leq \cos(\gamma_1) \leq 1$ and $0 \leq \gamma_2 \leq \pi$ are sampled uniformly inside their respective interval domains. This parameterization guarantees a uniform sampling of all the unit direction triplets which is invariant to rotation. For instance, a uniform sampling of γ_1 would have resulted in a bias (at the poles) which can be fixed by sampling $\cos(\gamma_1)$ uniformly instead (as can be verified by deriving the Jacobian of the joint distribution). The expectation μ can be computed by the following integral:

$$\begin{aligned}\mu &= \frac{1}{4\pi}\int_0^\pi\int_{-1}^1\int_0^{2\pi}(\mathbf{a}^\top\mathbf{b})(\mathbf{a}^\top\mathbf{c})(\mathbf{b}^\top\mathbf{c})d\gamma_2 d(cos(\gamma_1))d\beta \\ &= \frac{1}{4\pi}\int_0^\pi\int_{-1}^1\int_0^{2\pi} cos(\beta)cos(\gamma_1)(cos(\beta)cos(\gamma_1) \\ &+ sin(\beta)\sqrt{1-cos(\gamma_1^2)}cos(\gamma_2))d\gamma_2 dcos(\gamma_1)d\beta = \frac{1}{6}\end{aligned}$$

The s.t.d of the distribution can be similarly computed with the result of $\sigma = \sqrt{2}\frac{1}{6}$. □

Each entry G_{ij} is a a sum of k such terms, each with a mean of $1/6$ and s.t.d $\sqrt{2}(1/6)$, therefore the mean of G_{ij} is $k(1/6)$ with s.t.d $\sqrt{2k}(1/6)$. In the sequel we will take the worst case where $k = 1$. Next, we address the probability that a matrix G whose entries are random variables normally distributed and i.i.d. $G_{ij} \sim N(\mu,\sigma^2)$ will have a strictly non-negative leading eigenvector. We refer to this question as the "probabilistic" version of the Perron-Frobenious theorem over random matrices.

The body of results on spectral properties of random matrices (see for example [19]) deal with the distribution of eigenvalues. For example, the corner-stone theorem known as Wigner's *semicircle* theorem [33] is about the asymptotic distribution of eigenvalues with the following result: "Given a symmetric $n \times n$ matrix whose entries are bounded independent random variables with mean μ and variance σ^2, then for any $c > 2\sigma$, with probability $1 - o(1)$ all eigenvalues except for at most $o(n)$ belong to $\Theta(\sqrt{n})$, i.e., lie in the interval $\mathcal{I} = (-c\sqrt{n}, c\sqrt{n})$."

The notation $f(n) = o(g(n))$ stands for $\lim_{n\to\infty} f(n)/g(n) = 0$, i.e., $f(n)$ becomes insignificant relative to $g(n)$ with the growth of n. This is a short-hand notation (which we will use in the sequel) to the formal statement: "$\forall\epsilon > 0$, $\exists n_0$ s.t. $\forall n > n_0$ the statement holds with probability $1 - \epsilon$."

It is also known that when $\mu = 0$ all the eigenvalues belong to the interval $\mathcal{I}$ (with probability $1 - o(1)$), while for the case $\mu > 0$ only the leading eigenvalue λ_1 is outside of $\mathcal{I}$ and

$$\lambda_1 = \frac{1}{n}\sum_{i,j} G_{ij} + \frac{\sigma^2}{\mu} + O(\frac{1}{\sqrt{n}}),$$

i.e., λ_1 asymptotically has a normal distribution with mean $\mu n + \sigma^2/\mu$ [8]. Our task is to derive the asymptotic behavior of the leading eigenvector when $\mu > 0$ under the assumption that the entries of G are normally distributed i.i.d. random variables. We will prove below the following theorem:

Theorem 1 (Probabilistic Perron-Frobenious) *Let* $G = g_{ij}$ *be a real symmetric* $n \times n$ *matrix whose entries for* $i \geq j$ *are independent identically and normally distributed random variables with mean* $\mu > 0$ *and variance* σ^2. *Then, for any* $\sigma > 0$ *there exist* n_o *such that for all* $n > n_0$ *the leading eigenvector* $\mathbf{v}$ *of* G *is positive with probability* $1 - o(1)$.

Preliminaries: Let $G = \mu J + \sigma S$ where $J = \mathbf{1}\mathbf{1}^\top$ and S_{ij} are i.i.d. sampled according to $N(0,1)$. Let $\mathbf{e} = \frac{1}{\sqrt{n}}\mathbf{1}$. and let $\mathbf{v},\mathbf{v}_2,...,\mathbf{v}_n$ and $\lambda \geq \lambda_2 \geq ... \geq \lambda_n$ be the spectrum of G. From the semicircle law [33] and from [8] it is known that $\lambda_i = \Theta(\sqrt{n})$ for $i = 2, 3..., n$.

The following auxiliary claims would be useful for proving the main theorem.

Lemma 1 (Bounds on Leading Eigenvalue) *Under the conditions of Theorem 1 above, with probability* $1 - o(1)$

the leading eigenvalue λ of G falls into the following interval:

$$\mu n - o(1) \leq \lambda \leq \mu n + \Theta(\sqrt{n}).$$

Proof: From the definition of the leading eigenvalue we have:

$$\begin{aligned} \lambda &= \max_{\|\mathbf{x}\|=1} \mathbf{x}^\top G\mathbf{x} = \mu(\sum_i x_i)^2 + \sigma \max_{\|\mathbf{x}\|=1} \mathbf{x}^\top S\mathbf{x} \\ &\leq \mu n + \Theta(\sqrt{n}) \end{aligned}$$

where from the semicircle law $\max_{\|\mathbf{x}\|=1} \mathbf{x}^\top S\mathbf{x} = \Theta(\sqrt{n})$ and from Cauchy-Schwartz inequality $(\sum_i x_i)^2 \leq n(\sum_i x_i^2) = n$. The lower bound follows from:

$$\begin{aligned} \lambda &\geq \mathbf{e}^\top G\mathbf{e} = \mu n + \sigma \mathbf{e}^\top S\mathbf{e} \\ &= \mu n + \sigma N(0,1) \geq \mu n - o(1) \end{aligned}$$

□

Lemma 2 *Under the conditions of Theroem 1 above, with probability $1 - o(1)$ we have the following bound:*

$$\sum_i v_i \geq \sqrt{n} - c \tag{9}$$

for some constant c where v_i are the entries of the leading eigenvector $\mathbf{v}$ of G.

Proof: Let $\mathbf{e} = a\mathbf{v} + \sum_{i=2}^n a_i \mathbf{v}_i$. Since the eigenvectors and $\mathbf{e}$ are of unit norm we have $a^2 + \sum_{i=2}^n a_i^2 = 1$ and w.l.o.g. we can assume $a > 0$. We have therefore $\mathbf{e}^\top G\mathbf{e} = a^2\lambda + \sum_i \lambda_i a_i^2$. Since $\lambda_i = \Theta(\sqrt{n})$ for $i = 2, ..., n$ and $a^2 + \sum_i a_i^2 = 1$ we have:

$$\mathbf{e}^\top G\mathbf{e} \leq a^2\lambda + \Theta(\sqrt{n}).$$

Using the bound derived above of $\mathbf{e}^\top G\mathbf{e} \geq \mu n - o(1)$ and Lemma 1, we have:

$$\begin{aligned} \mu n - o(1) &\leq \lambda a_1^2 + \Theta(\sqrt{n}) \\ \frac{\mu n - \Theta(\sqrt{n})}{\mu n + \Theta(\sqrt{n})} &\leq a^2 \leq a \end{aligned}$$

from which we can conclude (with further manipulation):

$$1 - \frac{2\Theta(\sqrt{n})}{\mu n} = 1 - \frac{1}{\mu\Theta(\sqrt{n})} \leq a.$$

Consider now that a is the angle between $\mathbf{e}$ and $\mathbf{v}$:

$$\frac{1}{\sqrt{n}} \sum_i v_i = \mathbf{e}^\top \mathbf{v} = a \geq 1 - \frac{1}{\mu\Theta(\sqrt{n})},$$

from which we obtain:

$$\sum_i v_i \geq \sqrt{n} - c,$$

for some constant c. □

As a result so far, we have that

$$\begin{aligned} \lambda v_i &= (G\mathbf{v})_i = \mu \sum_i v_i + \sigma(S\mathbf{v})_i \\ &\geq \mu\sqrt{n} - C + \sigma \mathbf{g}^\top \mathbf{v} \end{aligned}$$

where $C = \mu c$ is a constant $\mathbf{g}$ is some n-dimensional normally distributed i.i.d random vector. We would be done if we could show that the probability of the event $\mathbf{g}^\top \mathbf{v} > (1/\sigma)\mu\sqrt{n}$ occurs with probability $o(1)$, i.e., decays with the growth of n. The problem is that since $\mathbf{g}$ stands for a row of S and because $\mathbf{v}$ depends on S we cannot make the assumption that $\mathbf{g}$ and $\mathbf{v}$ are independent — thus a straightforward Gaussian tail bound would not be appropriate. The remainder of the proof below was contributed by Ofer Zeitouni where care is taken to decouple the dependency between $\mathbf{g}$ and $\mathbf{v}$.

Proof of Theorem 1: Let $D(c)$ be the set of vectors in R^n satisfying Lemma 2:

$$D(c) = \left\{ \mathbf{v} \in R^n \ : \ \|\mathbf{v}\| = 1, \ \sum_i v_i \geq \sqrt{n} - c \right\},$$

and let $\mathbf{g} \in R^n$ be some vector of i.i.d. random variables with standard Normal distribution $N(0,1)$. We would like to analyze the probability of the event:

$$F = \left\{ \mathbf{g}^\top \mathbf{v} \geq \frac{\mu}{\sigma}\sqrt{n} \right\} \ \ \mathbf{g} \in R^n, \ \ g_i \sim N(0,1), \ \ \mathbf{v} \in D(c).$$

In particular we would like to show that the probability $P(F)$ belongs to $o(1)$, i.e., decays with the growth of n.

Let $\mathbf{v} = \mathbf{e} + \mathbf{f}$ where $\mathbf{e} = \frac{1}{\sqrt{n}}\mathbf{1}$ was defined above and $\mathbf{f}$ is the residual. From the constraint $\|\mathbf{v}\|^2 = 1$ we obtain a constraint on $\mathbf{f}$:

$$\frac{2}{\sqrt{n}} \sum_i f_i + \sum_i f_i^2 = 0 \tag{10}$$

Given that $\mathbf{v} \in D(c)$ we obtain:

$$\sum_i v_i = \sqrt{n}\mathbf{v}^\top \mathbf{e} = \sqrt{n} + \sum_i f_i \geq \sqrt{n} - c,$$

from which obtain another constraint on $\mathbf{f}$:

$$-\sum_i f_i \leq c \tag{11}$$

Combining both constraints (10) and (11) we arrive at:

$$\|\mathbf{f}\|^2 \leq \frac{2c}{\sqrt{n}} \tag{12}$$

The expression $\mathbf{g}^\top \mathbf{v}$ can be broken down to a sum of two terms:

$$\begin{aligned} \mathbf{g}^\top \mathbf{v} &= \mathbf{g}^\top \mathbf{e} + \mathbf{g}^\top \mathbf{f} \leq o(1) + \|\mathbf{g}\|\|\mathbf{f}\| \\ &\leq o(1) + \|\mathbf{g}\| \left(\frac{\sqrt{2c}}{n^{1/4}} \right) \end{aligned}$$

Since, with probability $1 - o(1)$, $\|\mathbf{g}\| = \Theta(\sqrt{n})$, the probability that $\mathbf{g}^\top \mathbf{v} \geq \Theta(\sqrt{n})$ is proportional to the probability that $\|\mathbf{g}\| \geq n^{3/4}$ which by the Gaussian tail bound decays eponentially with the growth of n. Since the probability that each entry of $\mathbf{v}$ is negative decays exponentially, i.e., $p(v_i < 0) < e^{-Cn}$, for some constant C, then by the union-bound the union of such events $p(v_1 < 0 \cup .... \cup v_n < 0)$ is bounded from above by ne^{-Cn} which decays exponentially with the growth of n. □

Fig. 1a displays a simulation result ploting the probability of positive leading eigenvector of G (with $\mu = 1/6$ and $\sigma = \sqrt{2}/6$) as a function of n. One can see that for $n > 20$ the probability becomes very close to 1 (above 0.99). Simulations with $\mu = 0.1$ and $\sigma = 1$ show that the probability is above 0.99 starting from $n = 500$.

To summarize the positivity issue, the weight vector α comes out positive due to the fact that it is the leading eigenvector of a matrix whose entries have a positive mean (Propositions 2 and 3). Theorem 1 made the connection between matrices which have the property of a positive mean and the positivity of its leading eigenvector in a probabilistic setting.

4.2 Sparsity Gap

We move our attention to the issue of sparsity of the weight vector α. It has been observed in the past that the key for sparsity lies in the positive combination of terms (cf. [17]) — therefore there is a strong (somewhat anecdotal) relationship between the positivity of α and the sparsity feature. We will establish below the relationship between the "sparsity gap" and the fraction of relevant features $0 < p \leq 1$. We will see that the gap between the large and small values of α_i is inversely proportional to the value of p. In other words, the sparsity result is significant when the ratio between the number of relevant and irrelevant features is high.

In Theorem 1 the matrix G was modeled as a single block where each element was normally distributed $N(\mu, \sigma^2)$. We extend this model to consist of a block structured matrix which includes a block with high correlation among the corresponding features (μ is high) and blocks with low correlation ($\mu = 1/6$) representing randomly selected feature vectors:

$$G = \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix} \tag{13}$$

where A is $np \times np$ (np being the number of relevant features) with entries normally distributed $N(\mu_a, \sigma^2)$, where C is $nq \times nq$ (nq, $p+q=1$, being the number of irrelevant features) and the entries of B, C are normally distributed $N(\mu_b, \sigma^2)$, where from Proposition 3 we can assume that $\mu_b = \frac{1}{6}$. The largest eigenvector of G will also broken down into two pieces the first containing np elements and the second nq elements. The *sparsity gap* is defined below:

Definition 6 (Sparsity Gap) *Let $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)^\top$ be the largest eigenvector of G defined in (13) where $\mathbf{x}_1$ holds the first np entries of $\mathbf{x}$ and $\mathbf{x}_2$ the remaining nq entries. The sparsity gap corresponding to G is the ratio $\frac{\bar{\mathbf{x}}_1}{\bar{\mathbf{x}}_2}$ where $\bar{\mathbf{x}}_i$ is the mean of $\mathbf{x}_i$, $i = 1, 2$.*

The theorem below derives the sparsity gap as a function of p:

Theorem 2 *Let $\bar{G}$ be the 2×2 matrix defined below:*

$$\bar{G} = \begin{bmatrix} np\mu_a & nq\mu_b \\ np\mu_b & nq\mu_b \end{bmatrix}$$

and let $\bar{\mathbf{x}} = (x_1, x_2)$ be the eigenvector associated with the largest eigenvalue of $\bar{G}$. The sparsity gap corresponding to G is

$$\frac{x_1 + N(0, \frac{\sigma^2}{np})}{x_2 + N(0, \frac{\sigma^2}{nq})}.$$

Proof: Sum up the rows of the matrix equation $\lambda \mathbf{x} = G\mathbf{x}$:

$$\lambda \sum_{i=1}^{np} x_i = (np) \sum_{i=1}^{np} \mu_a x_i + (np) \sum_{i=np+1}^{n} \mu_b x_i + N(0, np\sigma^2)$$
$$\lambda \sum_{i=np+1}^{n} x_i = (nq) \sum_{i=1}^{np} \mu_b x_i + (nq) \sum_{i=np+1}^{n} \mu_b x_i + N(0, nq\sigma^2),$$

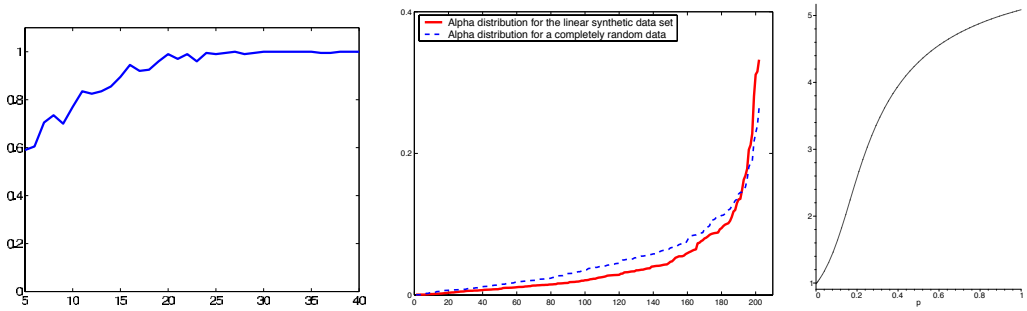
and divide the first equation by np and the second by nq. □

Note that we have omitted the higher values along the diagonal of the block A and C since they make no difference to final result. The sparsity gap asymptotes for large values of n because the normal distributions $N(0, \frac{\sigma^2}{np})$ and $N(0, \frac{\sigma^2}{nq})$ become delta functions around the origin as n increases. For example, empirical data show that $\mu_a \approx 0.85$ therefore the sparsity gap for values $\mu_a = 0.85, \mu_b = \frac{1}{6}$ and $n = 100$ is:

$$\frac{61p - 10 + \sqrt{3321p^2 - 820p + 100}}{20p}$$

Fig. 1c plots the sparsity gap as a function of p. The gap approaches the value of 5.1 when p approaches the value of 1. For example, when $p = 0.2$ (20% of the features are relevant) the sparsity gap is fairly significant and stands on 2.6.

In the next section we will present a number of experiments, both synthetic and with real data. Fig. 1b shows the weight vector α for a random data matrix M, and for a synthetic experiment (6 relevant features out of 202) described in the next section. One can clearly observe the positivity and sparsity of the recovered weight vector — even for a random matrix.



(a) (b) (c)

Figure 1: (a) Probability of positive leading eigenvector of the matrix G in simulations with $\mu = 1/6$ and $\sigma = \sqrt{2}/6$. The probability is very close to 1 starting from $n = 20$. (b) Positivity and sparsity demonstrated on the synthetic feature selection problem described in Section 6 (6 relevant features out of 202) and of a random data matrix. The alpha weight vector (sorted for display) comes out positive and sparse. (c) the plot of the theoretical sparsity gap (see text)

5 Representing Higher-order Cumulants using Kernel Methods

The information on which the $Q - \alpha$ method relies on to select features is contained in the matrix G. Recall that the criterion function underlying the $Q - \alpha$ algorithm is a sum over all pairwise feature vector relations:

$$trace(Q^\top A_\alpha^\top A_\alpha Q) = \alpha^\top G \alpha,$$

where G is defined such that $G_{ij} = (\mathbf{m}_i^\top \mathbf{m}_j)\mathbf{m}_i^\top Q Q^\top \mathbf{m}_j$. It is apparent that feature vectors interact in pairs and the interaction is *bilinear*. Consequently, cumulants of the original data matrix M which are of higher order than two are not being considered by the feature selection scheme. For example, if M were to be decorrelated (i.e., $MM^\top$ is diagonal) the matrix G would be diagonal and the feature selection scheme would select only a single feature rather than a feature subset.

In this section we employ the so called "kernel trick" to allow for cumulants of higher orders among the feature vectors to be included in the feature selection process. Kernel methods in general have been attracting much attention in the machine learning literature — initially with the introduction of the support vector machines [29] and later took a life of their own (see [27]). The common principle of kernel methods is to construct nonlinear variants of linear algorithms by substituting inner-products by nonlinear kernel functions. Under certain conditions this process can be interpreted as mapping of the original measurement vectors (so called "input space") onto some higher dimensional space (possibly infinitely high) commonly referred to as the "feature space" (which for this work is an unsuccessful choice of terminology since the word "feature" has a different meaning). Mathematically, the kernel approach is defined as follows: let $\mathbf{x}_1, ..., \mathbf{x}_l$ be vectors in the input space, say R^q, and consider a mapping $\phi(\mathbf{x}) : R^q \to \mathcal{F}$ where $\mathcal{F}$ is an inner-product space. The kernel-trick is to calculate the inner-product in $\mathcal{F}$ using a kernel function $k : R^q \times R^q \to R$, $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$, while avoiding explicit mappings (evaluation of) $\phi()$. Common choices of kernel selection include the d'th order polynomial kernels $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j + c)^d$ and the Gaussian RBF kernels $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2\sigma^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)$. If an algorithm can be restated such that the input vectors appear in terms of inner-products only, one can substitute the inner-products by such a kernel function. The resulting kernel algorithm can be interpreted as running the original algorithm on the space $\mathcal{F}$ of mapped objects $\phi(\mathbf{x})$. Kernel methods have been applied to the support vector machine (SVM), principal component analysis (PCA), ridge regression, canonical correlation analysis (CCA), QR factorization and the list goes on. We will focus below on deriving a kernel method for the $Q - \alpha$ algorithm.

5.1 Kernel $Q - \alpha$

We will consider mapping the rows $\mathbf{m}_i^\top$ of the data matrix M such that the rows of the mapped data matrix become $\phi(\mathbf{m}_1)^\top, ..., \phi(\mathbf{m}_n)^\top$. Since the entries of G consist of inner-products between pairs of mapped feature vectors, the interaction will be no longer bilinear and will contain higher-order cumulants whose nature depends on the choice of the kernel function.

Replacing the rows of M with their mapped version introduces some challenges before we could apply the kernel trick. The affinity matrix $A_\alpha = \sum_i \alpha_i \phi(\mathbf{m}_i)\phi(\mathbf{m}_i)^\top$ cannot be explicitly evaluated because A_α is defined by *outer-products* rather than inner-products of the mapped feature vectors $\phi(\mathbf{m}_i)$. The matrix Q holding the eigenvectors of A_α cannot be explicitly evaluated as well and likewise the

matrix $Z = A_\alpha Q$ (in step 4). As a result, kernelizing the $Q-\alpha$ algorithm requires one to represent α without explicitly representing A_α and Q both of which were instrumental in the original algorithm. Moreover, the introduction of the kernel should be done in such a manner to preserve the key property of the original $Q-\alpha$ algorithm of producing a sparse solution.

Let $V = MM^\top$ be the $n \times n$ matrix whose entries are evaluated using the kernel $v_{ij} = k(\mathbf{m}_i, \mathbf{m}_j)$. Let $Q = M^\top E$ for some $n \times k$ (recall k being the number of clusters in the data) matrix E. Let $D_\alpha = diag(\alpha_1, ..., \alpha_n)$ and thus $A_\alpha = M^\top D_\alpha M$ and $Z = A_\alpha Q = M^\top D_\alpha VE$. The matrix Z cannot be explicitly evaluated but $Z^\top Z = E^\top V D_\alpha V D_\alpha V E$ can be evaluated. The matrix G can be expressed with regard to E instead of Q:

$$\begin{aligned} G_{ij} &= (\phi(\mathbf{m}_i)^\top \phi(\mathbf{m}_j))\phi(\mathbf{m}_i)^\top QQ^\top \phi(\mathbf{m}_j) \\ &= k(\mathbf{m}_i, \mathbf{m}_j)\phi(\mathbf{m}_i)^\top (M^\top E)(M^\top E)^\top \phi(\mathbf{m}_j) \\ &= k(\mathbf{m}_i, \mathbf{m}_j)\mathbf{v}_i^\top EE^\top \mathbf{v}_j \end{aligned}$$

where $\mathbf{v}_1, ..., \mathbf{v}_n$ are the columns of V. Step 5 of the $Q-\alpha$ algorithm consists of a QR factorization of Z. Although Z is uncomputable it is possible to compute R and R^{-1} directly from the entries of $Z^\top Z$ without computing Q using the Kernel Gram-Schmidt described in [34]. Since $Q = ZR^{-1} = M^\top D_\alpha VER^{-1}$ the update step is simply to replace E with ER^{-1} and start the cycle again. In other words, rather than updating Q we update E and from E we obtain G and from there the newly updated α. The kernel $Q-\alpha$ is summarized below:

Definition 7 (Kernel $Q-\alpha$) *Let M be an uncomputable matrix with rows $\phi(\mathbf{m}_1)^\top, ..., \phi(\mathbf{m}_n)^\top$ where $\phi() : R^n \longrightarrow \mathcal{F}$ is a mapping from input space to a feature space and which is endowed with a kernel function $\phi(\mathbf{m}_i)^\top \phi(\mathbf{m}_j) = k(\mathbf{m}_i, \mathbf{m}_j)$. Therefore the matrix $V = MM^\top$ is a computable $n \times n$ matrix. Let $E^{(0)}$ be an $n \times k$ matrix selected such that $M^\top E^{(0)}$ has orthonormal columns. Perform the following steps through a cycle of iterations with index $r = 1, 2, ...$*

1. *Let $G^{(r)}$ be a $n \times n$ matrix whose (i,j) components are $k(\mathbf{m}_i, \mathbf{m}_j)\mathbf{v}_i^\top E^{(r-1)} {E^{(r-1)}}^\top \mathbf{v}_j$.*
2. *Let $\alpha^{(r)}$ be the largest eigenvector of $G^{(r)}$, and let $D^{(r)} = diag(\alpha_1^{(r)}, ..., \alpha_n^{(r)})$.*
3. *Let $Z^{(r)}$ be an uncomputable matrix*

   $$Z^{(r)} = (M^\top D^{(r)} M)(M^\top E^{(r-1)}) = M^\top D^{(r)} V E^{(r-1)}.$$

   Note that ${Z^{(r)}}^\top Z^{(r)}$ is a computable $k \times k$ matrix.
4. *$Z^{(r)} \xrightarrow{QR} QR$. It is possible to compute directly R, R^{-1} from the entries of ${Z^{(r)}}^\top Z^{(r)}$ without explicitly computing the matrix Q (see [34]).*
5. *Let $E^{(r)} = E^{(r-1)} R^{-1}$.*
6. *Increment index r and go to step 1.*

The result of the algorithm is the weight vector α and the design matrix G which contains all the data about the features.

6 Experiments

Synthetic Data

We compared the $Q-\alpha$ algorithm with three classical filter methods (Pearson correlation coefficients, Fisher criterion score and the Kolmogorov-Smirnoff test), standard SVM and the wrapper method using SVM of [32]. The data set we used follow precisely the one described in [32] which was designed for supervised 2-class inference. In [32] two experiments were designed, one with 6 relevant features out of 202 referred to as "linear" problem, and the other experiment with 2 relevant features out of 52 designed in a more complex manner and referred to as "non-linear" problem. In the linear data the class label $y \in \{-1, 1\}$ was drawn at equal probability. The first six features were drawn as $x_i = yN(i, 1)$, $i = 1..3$, and $x_j = N(0, 1)$, $j = 4..6$ at probability 0.7, otherwise they were drawn as $x_i = N(0, 1)$, $i = 1..3$, and $x_j = yN(i-3, 1)$, $j = 4..6$. The remaining 196 dimensions were drawn from $N(0, 20)$. The reader is referred to [32] for details of the non-linear experiment. We ran $Q-alpha$ on the two problems once with known classes (supervised version) and with unknown class labels (unsupervised version). In the supervised case the selected features were used to train an SVM and in the unsupervised case the class labels were not used for the $Q-\alpha$ feature selection but were used for the SVM training. The unsupervised test appears artificial but is important for appreciating the strength of the approach as the results of the unsupervised are only slightly inferior to the supervised test. In Fig. 2a we *overlay* the $Q-\alpha$ results (prediction error of the SVM on a testing set) on the figure obtained by [32]. The performance of the supervised $Q-\alpha$ closely agrees with the performance of the wrapper SVM feature selection of [32]. The performance of the unsupervised version does not fall much behind. Similar results were obtained for the non-linear problem but are omitted due to lack of space.

Since our method can handle more than two classes we investigated the scaling-up capabilities of the algorithm as we increase the number of classes in an unsupervised setting. For $k = 2, 3, ...$ classes we sampled k cluster centers in 3D space (3 coordinates per center) in the 3D cube where each coordinate is uniformly sampled in the interval $[-3, 3]$. Around each of the k class centers we sampled 20 points according to a normal distribution whose mean is the class center and with a unit s.t.d. for each coordinate. Taken together we have $20k$ points in 3D. We added 70 additional

coordinates sampled similarly around k centers sampled uniformly inside the 70D hypercube with edges of length 6. Each such added coordinate was permuted by a random permutation to break the correlation between the dimensions. Thus each of the $20k$ points lives in a 73-dimensional space out of which only the first three dimensions are relevant. We ran the $Q-\alpha$ algorithm on the $73 \times 20k$ data matrix and obtained the weight vector α and selected the three coordinates with the highest weights. We ran this experiment 50 times and recorded for each of the 73 coordinates the probability of being selected as one of the relevant features. The ratio of the average probability of the first three coordinates to the average probability of the remaining 70 coordinates was recorded. Ideally the ratio should be very high if the algorithm consistently succeeds in selecting the first three coordinates as the relevant ones. Fig. 2b illustrates the results of this experiment in a graph whose x-axis runs over the number of classes k and the y-axis displays the ratio score discussed above. One can see that the algorithm performed well until $k = 5$ classes (in a setting of three relevant features among 73) judging by the high ratio score. The performance degrades sharply after 5 classes as indicated by the low ratio score.

Real Image Unsupervised Feature Selection

The strength of the $Q-\alpha$ method is that it applies for unsupervised settings as well as supervised. An interesting unsupervised feature selection problem in the context of visual processing is the one of automatic selection of relevant features which discriminate among perceptual classes. Assume one is given a collection of images where some of them contain pictures of a certain object class (say, green frogs (*Rana clamitans* specie)) and other images contain pictures of a different class of objects (say, American toads) — see Fig. 3. We would like to automatically, in an unsupervised manner, select the relevant features such that a new picture could be classified to the correct class membership.

The features were computed by matching patches of equal size of 20×20 pixels in the following manner. Assuming that the object of interest lies in the vicinity of the image center, we defined 9 "template" patches arranged in a 3×3 block centered at the image. We had 27 images (18 from one class and 9 from the other), which in turn defines $27 * 9 = 243$ feature coordinates. Each image was sampled by 49 "candidate" patches (covering the entire image) where each of the 243 template patches was matched against the 49 patches in its respective image and the score of the best match was recorded in 243×27 data matrix. The matching between a pair of patches was based on L_1-distance between the respective color histograms in HSV space. The resulting α weight vector forms a feature selection from which we create a submatrix of data points

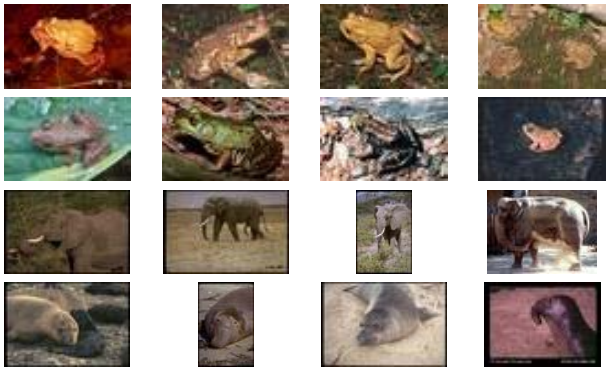


Figure 3: Image samples of several animal classes — American toad (top row) and Green frogs (*Rana clamitans*), elephants, and sea elephants. The objects appear in various positions, illumination, context and size.

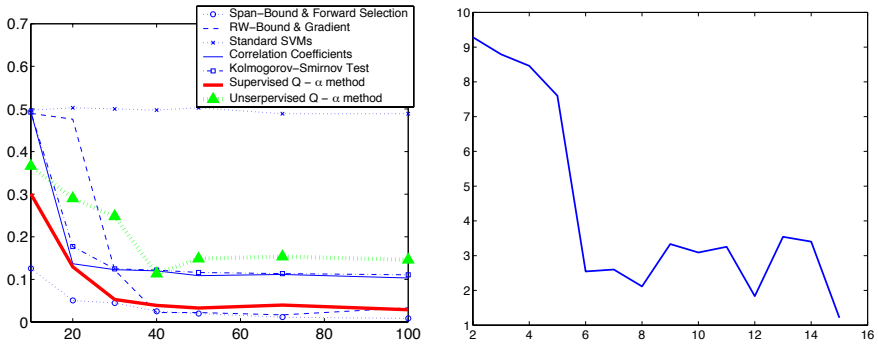
and construct its affinity matrix and the associated matrix of eigenvectors Q. The rows of the Q matrix were clustered using k-means into two clusters. A test image was classified based on distance from the cluster centroids. Performance on test images varied between 80% to 90% correct classification over many experiments over several object classes (including elephants, sea elephants, and so forth). This performance was compared to spectral clustering using all the 243 features which provided a range of 55% to 65% correct classification.

Fig. 5a and Fig. 5b show the 20 most relevant templates selected for the two classes, and Fig. 5c shows the alpha values. Note that the α weights are positive as predicted from Theorem 1 and exhibit a sharp break when the relevant features begin (sparsity).

6.1 Kernel $Q-\alpha$ Experiments

One of the possible scenarios for which a polynomial (for example) kernel is useful is when hidden variables affect the original feature measurements and thus create non-linear interactions among the feature vectors. We consider the situation in which the original measurement matrix M is multiplied, element wise, with a hidden variable matrix whose entries are ± 1. The value of the hidden state was changed randomly every 8 measurements and independently for each feature. This scheme simulates measurements taken in "sessions" where a session lasts for 8 sample data points. As a result, the expectation of the inner product between any two feature vectors is zero yet however any two feature vectors contain higher-order interactions which could come to bear using a polynomial kernel.

The kernel we used in this experiment was a sum second-order polynomial kernels each over a portion of 8 entries of



(a) (b)

Figure 2: (a) Comparison of feature selection methods following [32]. Performance curves of $Q-\alpha$ were overlaid on the figure adapted from [32]. The x-axis is the number of training points and the y-axis is the test error as a fraction of test points. The thick solid lines correspond to the $Q-\alpha$ supervised and unsupervised methods (see text for details). (b) Performance of a test with three relevant features and 70 irrelevant ones with k clusters represented by the x-axis of the graph. The y-axis represents a success score (see text for details). One can see that the unsupervised $Q-\alpha$ sustained good performance up to 5 classes.

the feature vector:

$$k(\mathbf{m}_i, \mathbf{m}_j) = \sum_k (m_i^{k\top} m_j^k)^2,$$

where m_i^k represents the k'th section of 8 successive entries of the feature vector $\mathbf{m}_i$. The original data was composed out 120 sample points with 60 coordinates out of which 12 were relevant and 48 were irrelevant. The relevant features were generated from three clusters, each containing 40 points. The points of a cluster were Normally distributed with a mean vector drawn uniformly from the unit hypercube in $\mathcal{R}^{12}$ and with a diagonal covariance matrix with entries uniformly distributed in the range [λ,2λ], where λ is a parameter of the experiment. A 2D slice out of the relevant 12 dimensions is shown in figure 4(a). The irrelevant features were generated in a similar manner, where for each irrelevant feature the sample points were permuted independently in order to break the interactions between the irrelevant features. This way it is impossible to distinguish between a single relevant feature and a single irrelevant feature.

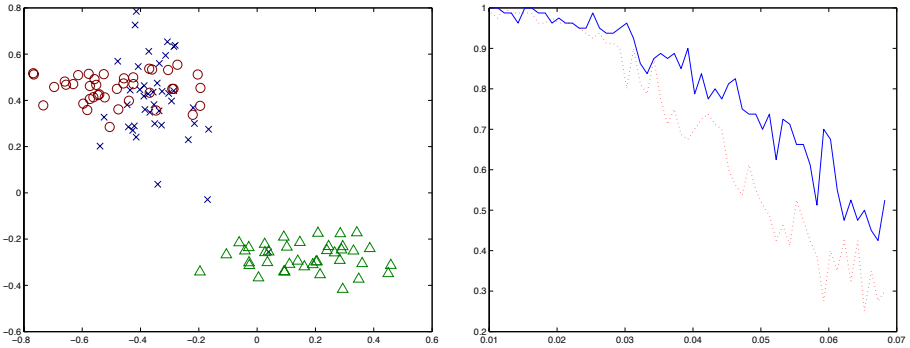
We considered an experiment to be successful if among the 12 features with the highest α values, at least 10 were from the relevant features subset. The graph in figure 4(b) shows the success rate for the kernel $Q-\alpha$ algorithm averaged over 80 runs. It also shows, for comparison, the success rate for experiments conducted by taking the square of every element in the measurements matrix followed by running the original $Q-\alpha$ algorithm. The success rate for the original $Q-\alpha$ algorithm on the unprocessed measurements was constantly zero and is not shown in the graph.

Genomics

We have tested our algorithm against the synthetic model of gene expression data ("microarrays") given in [3]. This synthetic model has 6 parameters m, a, b, e, d, s, explained below. a samples are drawn from class A, and b samples are drawn from class B. Each sample has m dimensions - em samples are drawn randomly using the distribution $N(0, s)$. The rest of the $(1-e)m$ features are drawn using either $N(\mu_A, \mu_A s)$ or $N(\mu_B, \mu_B s)$, depending on the class of the sample. The means of the distributions μ_A and μB are uniformly chosen from the interval $[-1.5d, 1.5d]$.

In [3] the parameters of the model were estimated to best fit the gene expressions of the leukemia dataset: $m = 600, a = 25, b = 47, e = 0.72, d = 555, s = 0.75$ (the leukemia dataset has over 7000 gene expressions but contains much redundancy). Similarly to [3], we varied one of the parameters m, d, e, s while fixing the other parameters to the values specified above. This enabled us to compare the performance of the $Q-\alpha$ algorithm to the performance of their Max-Surprise algorithm (MSA).

Our algorithm was completely robust to the number of features m. It always chose the correct features using as few as 5 features. MSA needed at least 250 features, since it used the redundancy in the features in order to locate the informative features. Both algorithms are invariant to the distance between the means of the distributions determined by d, and perform well for $d \in [1, 1000]$. The percentage of irrelevant features, e, can reach 95% for MSA and 99.5% for our algorithm. Such performance suggests that the data set is not very difficult.



(a) (b)

Figure 4: (a) 2D slice out of the relevant features in the original data matrix used in the synthetic experiment, showing three clusters. (b) A graph showing the success rate for the 2nd order polynomial kernel (solid blue), and for a preprocessing of the data (dashed red). The results are shown over the parameter λ specifying the variance of the original dataset (see text). The success rate of the regular $Q - \alpha$ algorithm was constantly zero and is not shown.

The parameter s effects the spread of each class. While MSA was able to handle values of s reaching 2, our algorithm was robust to s, and was at least 30 times more likely to choose a relevant feature than an irrelevant one, even for $s > 1000$.

The unsupervised analysis of real gene expression data sets is subject to future research.

Acknowledgments

We would like to thank Ofer Zeitouni, Michael Ben-Or and Alex Samorodnitsky for assistance with the proof of Theorem 1.

References

[1] H. Almuallim and T .G .Dietterich. Learning with many irrelevant features. *Proc. 9th Nat. Conf. on AI*, 1991.

[2] R. Bhatia. *Matrix Analysis*. Springer-Verlag, NY, 1996.

[3] A. Ben-Dor, N. Friedman, and Z. Yakhini. Class Discovery in Gene Expression Data. In *RECOMB*, 2001

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

[5] P.S. Bradley and O.L. Mangasarian feature selection via concave minimization and support vector machines *ICML*, 1998

[6] M. Brand and K. Huang. A unifying theorem for spectral embedding and clustering In *9th Int. Conf. on AI and Statistics*, 2002.

[7] F.R.K. Chung. *spectral graph theory*. AMS, 1998.

[8] Z. Furedi and J. Komlos. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.

[9] L.E. Gibbons, D.W. Hearn, P.M. Pardalos, and M.V. Ramana. Continuous characterizations of the maximum clique problem. *Math. Oper. Res.*, 22:754–768, 1997.

[10] G. Golub and C.V. Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.

[11] I. Guyon and A. Elissef. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 2003.

[12] K. Hall. An R-dimensional quadratic placement algorithm. *Management Science*, 17(3), 219–229, 1970.

[13] K. Kira and L. Rendell. A practical approach to feature selection. *ICML*, 1992.

[14] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *STOC*, 95

[15] R. Kohavi and G. John. Wrappers for feature selection. *Artificial Intelligence*, 97(1-2), 273–324, 1997.

[16] P. Langley and W. Iba. Average-case analysis of a nearest neighbor algorithm. In *Proceedings of the 13th Int. Conf. on Artificial Intelligence*, 1993.

[17] D.D. Lee and H.S. Seung. learning the parts of objects by non-negative matrix factorization. *Nature* 401(10), 1999

[18] D. D. Lewis. Feature selection and feature extraction for text categorization. In *Proceedings of Speech and Natural Language Workshop*, 1992.

[19] M.L. Mehta. *Random Matrices*. Academic Press, 1991.

[20] A.Y. Ng, M.I. Jordan and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. NIPS, 2001.

[21] B. Mohar. *The Laplacian Spectrum of Graphs*. Wiley, 1991.

[22] T.S. Motzkin and E.G. Straus. Maxima for graphs and a new proof of a theorem by turan. *Canadian Journal of Math.*, 17:533–540, 1965.

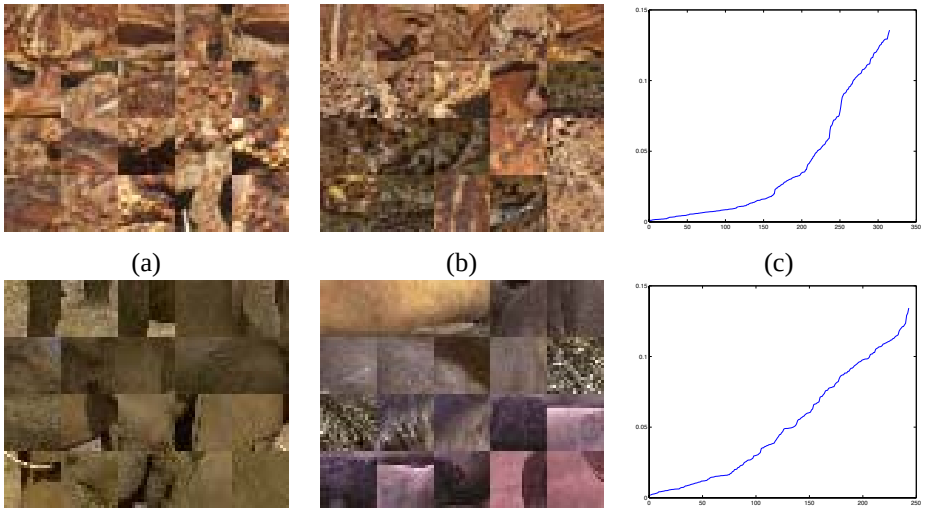


Figure 5: Unsupervised feature selection for automatic object discrimination from images. (a),(b) the first 20 features from pictures containing the American frog and the Green frog ranked by the α weight vector. (c) the (sorted) α values. (d),(e),(f) similar to the elephant and sea elephant.

[23] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(13), 1996.

[24] M. Pavan and M. Pelillo. A new graph-theoretic approach to clustering and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[25] P. Perona and W. T. Freeman. A Factorization Approach to Grouping. ECCV,1998.

[26] S. Sarkar and K.L. Boyer Quantitative measures of change based on feature organization: eigenvalues and eigenvectors In *CVIU*, 71(1), pp. 110-136, 1998.

[27] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[28] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *PAMI* 22(8), 2000.

[29] V.N. Vapnik. *The nature of statistical learning*. Springer, 2nd edition, 1998.

[30] P. Viola and M. Jones. Robust Real-time Object Detection *IJCV*, 2002.

[31] Y. Weiss. Segmentation using eigenvectors: a unifying view. *ICCV*, 1999.

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

[33] E.P. Wigner. On the distribution of the roots of certain symmetric matrices. *Ann. of Math. (2)*, 67:325–327, 1958.

[34] L. Wolf and A. Shashua. Kernel principal angles for classification machines with applications to image sequence interpretation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.