Simultaneous Selection of Features and Metric for Optimal Nearest Neighbor Classification

**David A. Johannsen,[1] Edward J. Wegman,[2,*]
Jeffrey L. Solka,[1] and Carey E. Priebe[3]**

[1]Naval Surface Warfare Center, Dahlgren Division,
Dahlgren, Virginia, USA
[2]Center for Computational Statistics, George Mason University,
Fairfax, Virginia, USA
[3]Department of Applied Mathematics and Statistics, Johns Hopkins
University, Baltimore, Maryland, USA

ABSTRACT

Given a set of observations in $\mathbb{R}^n$ along with provided class labels, $\mathscr{C}$, one is often interested in building a classifier that is a mapping from $\mathbb{R}^n \to \mathscr{C}$. One way to do this is using a simple nearest neighbor classifier. Inherent in the use of this classifier is a metric or pseudo-metric that measures the distance between the observations. One typically uses the L_2 metric. We examine the classification benefits of the use

of alternative Minkowski p-metrics. We also study the relationship between the selection of the p-metric and the selection of optimal classification features. We compare a simple greedy approach of Minkowski p-metric optimization followed by feature selection, the greedy method, with a simultaneous optimization of the p-metric and feature selection process. We utilize a stochastic optimization methodology to perform the simultaneous optimization.

Key Words: Minkowski p-metric; Simultaneous optimization; Cross-validated classifier; Greedy algorithm.

Mathematics Subject Classification: Primary 62H30; Secondary 62-07.

1. INTRODUCTION

Suppose that one is presented with ''high-dimensional'' training data from which one would like to build a nearest neighbor classifier. Though there are several possible ways in which to proceed, one (not unreasonable) course is to begin by selecting a metric that gives optimal nearest neighbor classifier performance. Then, one could select some ''optimal'' subset of the features using this previously selected metric. We will refer to this method as a ''greedy'' algorithm, in the sense that one first optimizes the metric and then one chooses an ''optimal'' feature set using this metric. The work that we describe here is the use of stochastic optimization techniques to perform simultaneous feature and metric selection.

This paper details our investigation of nearest neighbor cross-validated classifier performance (a surrogate for the Bayes error) as a function of feature selection and Minkowski metric. More specifically, we describe a methodology for simultaneous optimization of metric and feature selection. We also compare the performance of the classifier resulting from this simultaneous optimization with earlier work in which the metric and feature set optimizations are done separately (as seems to be the usual case, in practice).

We conducted this study of the optimization using two data sets: the Tufts University ''artificial nose'' data set (both ''raw'' and smoothed), and the Golub gene data set (both the entire data set, as well as normalized data). We remark that we have done no theoretical work at this point regarding existence of global minimums, convergence properties of the algorithms, etc. The motivation for this method of simultaneous optimization is simply the heuristic that greedy solutions are seldom global solutions.

Our approach is to pose the question of simultaneous metric and feature selection as an optimization problem. In this setting, the cross-validated classifier performance (being a mapping of the combined feature and metric space to the interval $[0, 1]$) is the objective function. We then seek a point in the combined feature and metric space that yields optimal performance of the classifier. Though we have not had the opportunity to perform a detailed theoretical analysis, it is obvious that no such optimal point need exist. This fact is immediate because the domain is not compact (in principle, the parameter for the metric lies in the half-open interval $[0, \infty)$). However, in practice (due to finite range of floating point numbers) we do not allow the parameter for the metric to become arbitrarily large, restricting the parameter to the closed interval $[1, k]$, for some integer k. Then, by the Extreme Value Theorem, we may speak of an optimal point.

With high-dimensional data, it is not possible to do classical gradient descent optimization (our original thought). This is because, for high-dimensional data, finite difference gradient estimates are too costly. To proceed, we adopted Spall's methodology for stochastic optimization (see Spall, 2003 or Spall, 1998a). Finally, we found that to perform feature selection/dimensionality reduction, it is not sufficient to optimize classifier performance, but rather an augmented function which includes a penalty term to force dimensionality reduction.

This simultaneous feature and metric selection outperformed the sequential methodology of metric selection followed by feature selection. We will describe the process of metric selection followed by feature selection by scatter in Sec. 3. We then compare the classifier performance using this (traditional) ''greedy'' approach with the cross-validated classifier performance using simultaneous selection of features and metric.

We mention that we have also considered additional methods of classifier optimization. We refer the reader to Solka and Johannsen (2003) where we studied the use of minimal spanning trees as an alternative measure of classifier performance. The number of inter-class edges in the minimal spanning tree for the training data was treated as a measure of classifier complexity. We examined nearest neighbor classifier performance as a function of Minkowski p-parameter, where the value of p was chosen that minimized this measure of the classifier complexity.

2.   DATA SETS

As we stated in the introduction, the purpose of the work described here was initial assessment of the feasibility and performance of

simultaneous optimization of metric and feature selection. To conduct this study, we implemented the algorithms and tested using two high-dimensional data sets: the ''artificial nose'' and Golub gene data sets. In this section, we briefly describe these two data sets.

2.1. "Artificial Nose" Data Set

The ''artificial nose'' data set was taken from an optical system constructed at Tufts University (White et al., 1996). The system consists of a bundle of 19 fibers. The end of each fiber is doped with a solvato-chromic dye which fluoresces in the presence of an analyte. An observation is obtained as the intensity of florescence of the dye molecules for each fiber at two wavelengths at 60 times during a 20 sec exposure. Thus, the response of the system to a ''sniff'' is a point in 2280-dimensional space:

19 fibers $\times$ 2 wavelengths $\times$ 60 samples $=$ 2280 dimensional data.

The task at hand is to identify trichloroethylene (TCE) as a component of the analyte in the presence of various confusers. (TCE, a carcinogenic industrial solvent, is of interest as a ground water contaminant.)

If one considers a single fiber/wavelength combination, a sample then consists of a time series of 60 observations of florescence intensity equally spaced over a 20 sec interval. The smoothed ''artificial nose'' data set was obtained by smoothing each of these time series (see Ramsay and Silverman, 1997). In particular, we employed polynomial smoothing splines to smooth the raw nose data. The reader is referred to Priebe et al. (2000) for more details.

2.2. Golub Gene Data Set

The second data set is the Golub gene data set (Golub and Slonin, 1999). This *Affymetrix* data set consists of expression levels of 7129 genes on a group of 72 patients all of whom are ill with leukemia. The patient population was divided between those with the acute lymphoblastic (ALL) variant, and those with the acute myeloblastic (AML) variant (47 and 25 patients, respectively). The discrimination problem is to distinguish between the ALL and AML variants of leukemia from the gene expression levels.

A variation of the full Golub data set was obtained by normalizing the data. In this procedure, the dimensionality of the data set was reduced

by retaining only the genes whose expression level is greater than 20 across all patients. This reduces the dimensionality from 7129 to 1753. Then, we normalized this reduced data. This normalization was performed by considering the data set as an n_g genes by an n_s patients data matrix. We then divided each column by its mean, and then subjected each row to a standard normalizing transformation.

3. THE GREEDY APPROACH

In this section, we describe what we refer to as the the "greedy" approach to metric and feature selection. In this setting, we began by selecting a metric which gives the optimal nearest neighbor cross-validated classifier performance using the full feature set. Then, we selected a subset of the features that gave "optimal" performance of the cross-validated nearest neighbor classifier, where interpoint distances were computed using the previously determined metric. We did not perform an exhaustive search for the optimal combination of features (clearly this is infeasible with high-dimensional data sets). Instead, we ranked the features by a surrogate measure (scatter, described below), and then added the features one at a time and computed the full cross-validated performance. After reaching the full feature set, we determined which collection of features gave the best classifier performance. This sequential optimization (first selecting optimal metric and then the optimal features) is why we refer to this method as a "greedy" algorithm.

3.1. Metric Selection

In the metric selection, we only consider Euclidean metrics. More specifically, we select the metric from the one-parameter family of Minkowski metrics. That is if $\mathbf{u}$ and $\mathbf{v}$ are vectors in $\mathbb{R}^n$, then we define the Minkowski p metric, $\rho_p(\cdot,\cdot)$ by

$$\rho_p(\mathbf{u},\mathbf{v}) = \left(\sum_{i=1}^{n} |u_i - v_i|^p \right)^{1/p}.$$

We note that for the "artificial nose" data set we must define the Minkowski metric differently. Recall, that we consider an observation to be a 38-dimensional vector each of whose entries is a 60-dimensional vector (the time-series of florescence intensities for a particular fiber at a particular wavelength). We will denote such an observation by

$\mathbf{v} \equiv v^{\phi,\lambda,t}$, where ϕ indicates the fiber, λ the wavelength, and t the time. Now, suppose that $\mathbf{u}$ and $\mathbf{v}$ are observations. We define the Minkowski p metric by the equation

$$\tilde{\rho}_p(\mathbf{u}, \mathbf{v}) = \left(\sum_{\phi=1}^{19} \sum_{\lambda=1}^{2} \left[\rho_p\left(u^{\phi,\lambda,\cdot} - v^{\phi,\lambda,\cdot} \right) \right]^p \right)^{1/p}$$

$$= \left(\sum_{\phi=1}^{19} \sum_{\lambda=1}^{2} \left(\sum_{i=1}^{60} \left| u^{\phi,\lambda,i} - v^{\phi,\lambda,i} \right|^p \right) \right)^{1/p}.$$

3.2. Feature Selection by "Scatter"

Our method for performing feature selection was based on "scatter" (derived from the Fisher linear discriminant, see Fukunaga, 1990). The basic idea is to select those dimensions in which the two classes are well separated. By this, we mean that the class means are well separated, and that each class has small intraclass variance.

We begin a more detailed discussion of the scatter. Suppose (for simplicity) that we have two classes, $\mathscr{C}_1$ and $\mathscr{C}_2$, consisting of n_1 and n_2 members, respectively. One first computes the class means,

$$m_i = \frac{1}{n_i} \sum_{x \in \mathscr{C}_i} x, \quad i = 1, 2.$$

Next, one computes the class scatter matrices

$$S_i = \sum_{x \in \mathscr{C}_i} (x - m_i)(x - m_i)^T.$$

Then, the within-class scatter matrix is expressed by

$$S_W = S_1 + S_2,$$

and the between-class scatter matrix as

$$S_B = (m_1 - m_2)(m_1 - m_2)^T.$$

In the univariate case, S_W and S_B are scalars. This is the case when considering individual dimensions (i.e., genes) in the Golub gene data. Then, the well separated dimensions are those with large value of S_B/S_W.

In the multivariate case (say, n-dimensional), S_W and S_B are $n \times n$ matrices. This is the case when considering the "artificial nose" data set, where it was natural to consider the data as a 38-dimensional vector (i.e., the 19 fibers each of two wavelengths) each of whose entries is a 60-dimensional vector (i.e., the 60 equally spaced measurements of the intensity during the "sniff"). In this case, S_B/S_W for a particular dimension (i.e., fiber/wavelength pair) is no longer meaningful. Instead, the quality of interest is large values of $\text{tr}(S_B)/\text{tr}(S_W)$, where $\text{tr}(\cdot)$ denotes the trace operator.

Finally, we note a minor point regarding the "artificial nose" data. One of the fibers at one of the two wavelengths had an identically zero response to all analytes. Thus, for this dimension, the within-class variance is zero. Thus, for this dimension, $\text{tr}(S_B)/\text{tr}(S_W)$ is not defined. Thus, this dimension was pruned from the data, resulting in 37-dimensional data.

3.3.   Results Using the "Greedy" Optimization

In Sec. 3.1, we described the family of metrics considered. Recall, that these metrics are parametrized by the (in this case integer) value p. We first selected an optimal value of p by performing an exhaustive search. That is, we specified an allowable range for p (we considered the following range of values: $p \in \{1, 2, \ldots, 50\}$). Then at each value of p in this range, we computed the full cross-validated nearest neighbor classifier performance (where interpoint distances were computed using the current value of p). We then selected the value of p that gave the best classifier performance, say $\bar{p}$.

We next determined the optimal subset of features. An exhaustive search of the feature space is clearly impossible with high-dimensional data (there is a combinatorial explosion if one wants to check all possible subsets – recall that there are 2^n subsets of a set of cardinality n). So, we adopted the surrogate measure of scatter, as described in Sec. 3.2. More specifically, we computed the scatter along each dimension. Then, we ranked the dimensions in order of decreasing scatter (i.e., from highest scatter value to lowest). Then we computed the full cross-validated nearest neighbor classifier performance (using the previously determined $\bar{p}$) using the single dimension of greatest scatter. Next, the dimension with the next greatest scatter was used to augment the feature set, and the cross-validated classifier performance was again computed. This process (of augmenting the feature set with the dimension of greatest scatter that had not previously been included) was repeated until the cross-validated

performance was computed using the full feature set. Finally, the feature set was selected that gave the best cross-validated nearest neighbor classifier performance using the $\bar{p}$ metric.

In the following figures we present plots of classifier performance as a function of the successive addition of scatter selected genes. We noticed a similarity across the two data sets. That is, with the raw data (the nonsmoothed ''artificial nose'' and the non-normalized Golub gene data sets), classifier performance initially increases, reaches some peak performance and then decreases with the addition of more features. We noted that with the smoothed data sets, classifier performance increases more or less monotonically until the full feature set is included.

Figure 1 shows the performance of the classifier on the (non-smoothed) ''artificial nose'' data set, and Fig. 2 shows the performance of the classifier on the full Golub gene data set. Note that with both of these data sets, the classifier performance initially increases with the inclusion of additional features, followed by a decrease in classifier
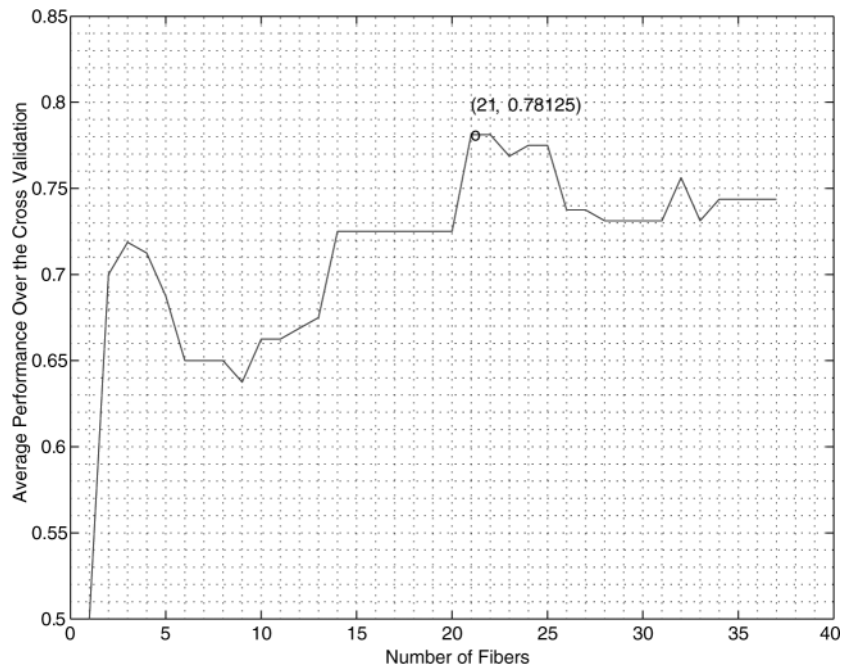

Figure 1. Performance of the cross-validated nearest neighbor classifier for the (nonsmoothed) ''artificial nose'' data set at the optimal $p = 5$.
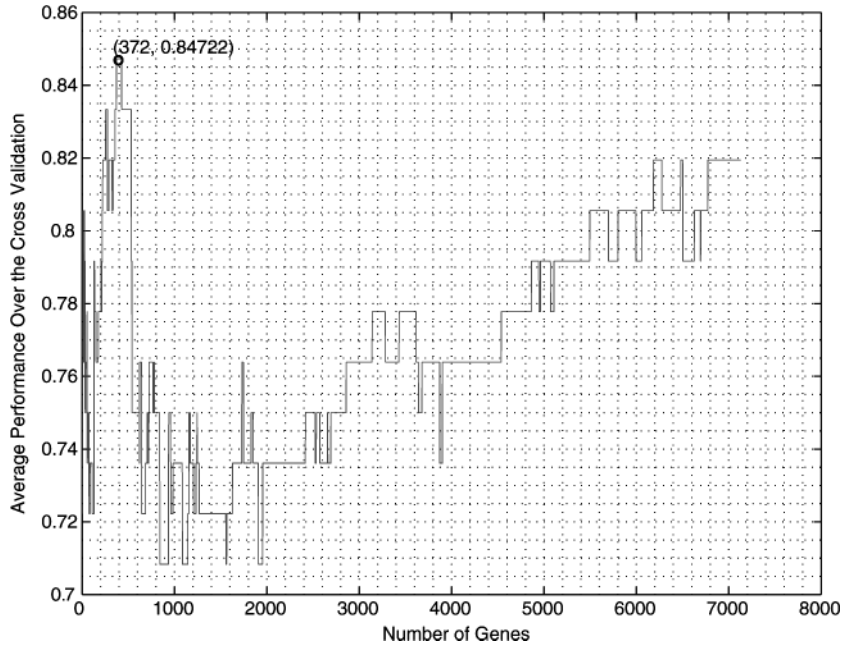
Figure 2. Performance of the cross-validated nearest neighbor classifier for the full Golub data set at the optimal $p = 4$.

performance as additional features are added. For the "artificial nose" data set, the peak classifier performance (≈ 0.78) occurs when using 21 of the 37 available wavelength/frequency combinations. The optimal Minkowski metric was $p = 5$. For the Golub gene data set, the peak classifier performance (≈ 0.85) occurs when using 372 of the 7129 available genes. The optimal Minkowski metric was $p = 4$.

Figure 3 shows the classifier performance for the smoothed artificial nose data set, and Fig. 4 shows the classifier performance for the normalized Golub gene data set. These two figures illustrate the approximately monotonic improvement in the classifier performance on the smoothed data with the addition of more features. For the smoothed artificial nose data set the peak classifier performance (≈ 0.86) occurs using all 37 available fiber/wavelength combinations and a Minkowski metric $p = 29$. For the normalized Golub gene data set, the peak classifier performance (≈ 0.86) occurs using 1698 of the 1753 available genes and a Minkowski metric $p = 4$.
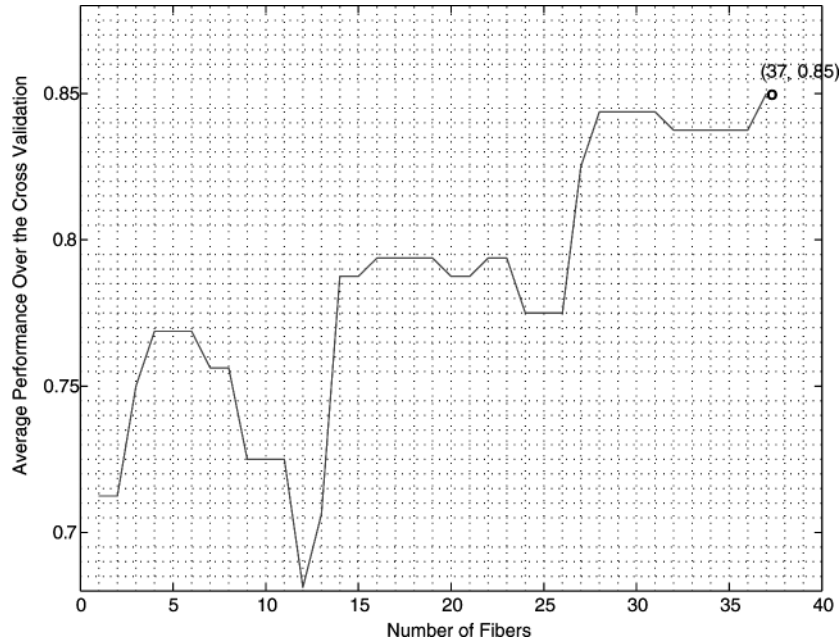
Figure 3. Performance of the cross-validated nearest neighbor classifier for the smoothed "artificial nose" data set at the optimal $p = 29$.

4. SIMULTANEOUS OPTIMIZATION

It is a general rule that, because greedy solutions consider only a small subset of all possible combinations, they are seldom globally optimal. Thus, to build truly optimal nearest neighbor classifiers, we want to perform a simultaneous optimization of the metric and feature set. Describing the method of simultaneous optimization and the results obtained using this method constitutes the material in this section. In the following section (Sec. 5), we compare the classifier performance obtained with the simultaneous metric and feature selection with that of the greedy approach.

Roughly, the idea is to treat the simultaneous optimization as a classical constrained optimization problem on $[0,1]^n \times [1, \infty)$ (actually, $[0,1]^n \times [1, k]$, some "large" k). One treats the cross-validated classifier performance as the objective function, where the performance is a function of the weights (the $[0,1]^n$ term) and the metric (the $[1, \infty)$ term). The weights (by being in $[0,1]$) perform the feature selection. The optimization proceeds by gradient descent to an optimal set of features and metric.
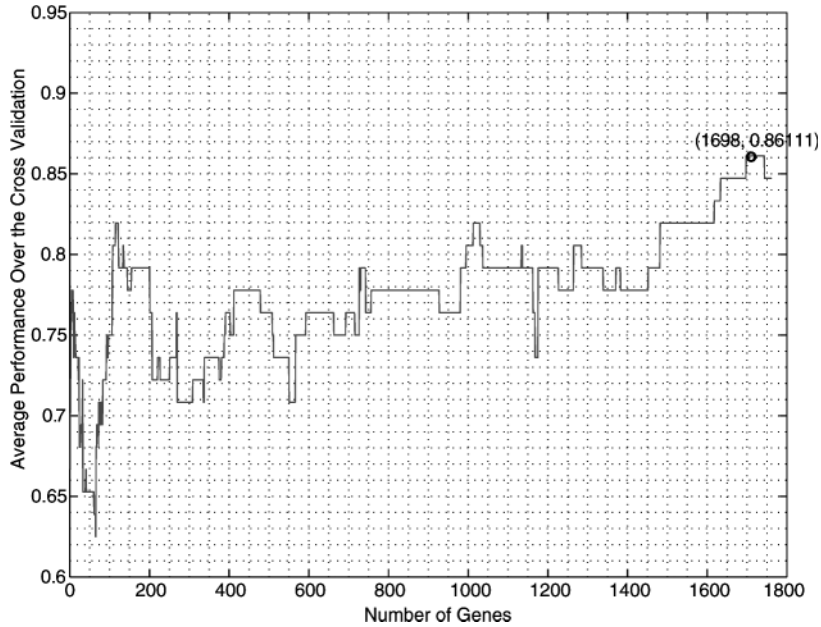
Figure 4. Performance of the cross-validated nearest neighbor classifier for the normalized Golub data set at the optimal $p = 4$.

4.1. The General Setting

We now describe the general setting for the simultaneous selection of metric and features. The idea is to treat the simultaneous $(\mathbf{w}, p)$ determination as a classical optimization problem. Denote by $\widehat{L}(\mathbf{w}, p)$ the estimated probability of misclassification (i.e., minimization of $\widehat{L}$ yields the optimal classifier), where this notation makes explicit the dependence of the classifier on the two parameters $\mathbf{w}$, and p.

The optimization problem is to determine the $\underset{(\mathbf{w},p)}{\text{argmin}}\ \widehat{L}(\mathbf{w}, p)$. Function evaluations are computed by performing cross-validated nearest neighbor classification. We note that we are now using a weighted metric, slightly different than the one given above (Sec. 3.1). We incorporate a vector of weights $\mathbf{w} \in [0,1]^n$, that serve to perform the feature selection. Thus, the metric used for the cross-validation is now given by

$$\rho_p(u, v) = \left(\sum_{i=1}^{n} (w_i |u_i - v_i|)^p \right)^{1/p}$$

for the case of the Golub gene data, and by

$$
\tilde{\rho}_p(\mathbf{u}, \mathbf{v}) = \left(\sum_{\phi=1}^{19} \sum_{\lambda=1}^{2} \left[w_{\phi,\lambda} \cdot \rho_p \left(u^{\phi,\lambda,\cdot} - v^{\phi,\lambda,\cdot} \right) \right]^p \right)^{1/p}
$$

$$
= \left[\sum_{\phi=1}^{19} \sum_{\lambda=1}^{2} \left(w_{\phi,\lambda} \left(\sum_{i=1}^{60} |u^{\phi,\lambda,i} - v^{\phi,\lambda,i}|^p \right)^{1/p} \right)^p \right]^{1/p}.
$$

in the case of the "artificial nose" data. Thus, $\widehat{L}(\mathbf{w}, p)$ is explicitly a function of $\mathbf{w} \in [0,1]^n$ and $p \in [0, \infty)$.

For true feature selection, we would like $\mathbf{w} \in \{0,1\} \times \cdots \times \{0,1\}$ (n-fold); i.e., $w_i \in \{0,1\}$ for $i = 1, \ldots, n$. In order to encourage this, we augment the objective function with the penalty term

$$
\tau(\mathbf{w}) = \mu \sum_{k=1}^{n} w_k^2 (w_k - 1)^2,
$$

where $\mu = 16/n$ is chosen so that the penalty term has approximately the same order of magnitude as the objective function (note that if $\mathbf{w} = (1/2, \ldots, 1/2)$ then $\tau(\mathbf{w}) = 1$). Figure 5 gives a plot of the penalty function in two dimensions.

Thus, the function that we sought to minimize is

$$
\psi(\mathbf{w}, p) = \widehat{L}(\mathbf{w}, p) + \tau(\mathbf{w}),
$$

subject to the constraints $\mathbf{w} \in [0,1]^n$ and $p \in [1, \infty)$, where n is the dimensionality of the data.

Now, the obvious method for finding the minimum of $\psi(\mathbf{w}, p)$ is to do a traditional gradient descent algorithm using finite difference approximation to the gradient at each iteration. Perhaps we should note that $\widehat{L} \notin \mathscr{C}^1$, but of course we may take a $\mathscr{C}^\infty$ approximation arbitrarily close (in the compact-open topology); anyway, $\widehat{L}$ is itself an estimator of the Bayes error, so we will not concern ourselves unduly with technical minutia here. There are several factors which make this traditional gradient descent approach infeasible. The principle of which is that function evaluations are computationally expensive, in that they consist of cross-validation nearest neighbor classifier performance on the training data. With high-dimensional data, finite difference gradient approximations may require as many as twice the number of function evaluations as
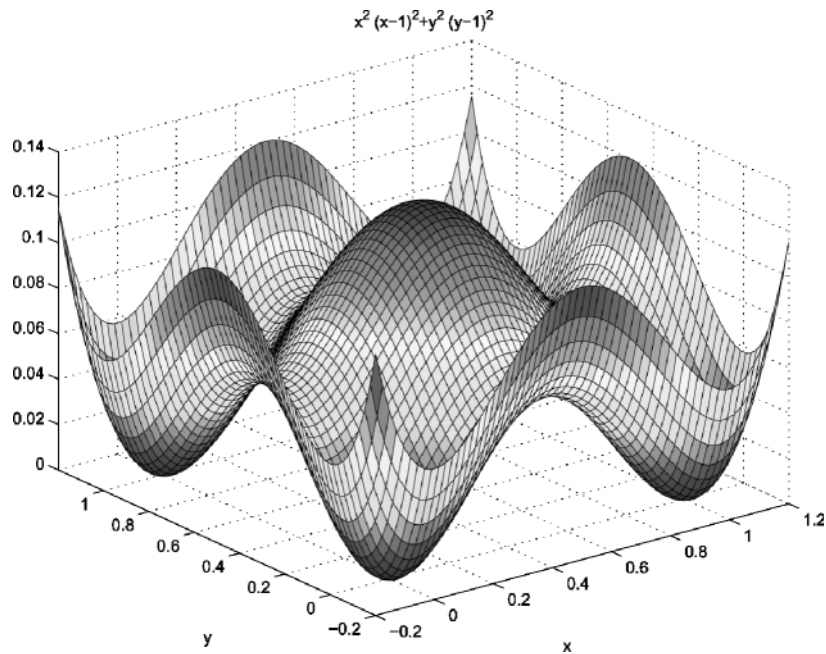
Figure 5. A plot of the penalty function in two dimensions.

the dimensionality of the data (in the case of centered difference approximations to each of the partial derivatives). Thus, centered difference approximations to $\nabla \psi(\mathbf{w}, p)$ for the Golub gene data would require 14,258 iterations of cross-validated classifier performance. Clearly, this is not computationally feasible.

We employed two strategies to determine the approximate minimum. The first strategy was to make the function evaluations less expensive. Instead of performing the full cross-validation of the classifier performance at each iteration, we performed the cross-validation only on a uniformly sampled (without replacement) subset of the training data (preserving in the subset the relative frequencies of representatives of each class present in the training data). The second modification of the naive approach was to reduce the number of function evaluations. Instead of computing a finite difference approximation to the gradient, we estimated the gradient using Spall's simultaneous perturbation stochastic approximation (SPSA) method (see Spall, 1998a or the recent Spall, 2003 for a detailed overview of the SPSA method; we present a brief description in Sec. 4.2). The chief benefit of the SPSA method is that gradient

approximations require only two function evaluations, regardless of the dimensionality of the optimization problem.

4.2. Spall's "Simultaneous Perturbation Method"

Recall that we want to minimize $\psi(\mathbf{w}, p)$, where we assume that the real-valued function is at least $\mathscr{C}^1$. Thus, we seek $(\tilde{\mathbf{w}}, \tilde{p})$ such that $\nabla\psi(\tilde{\mathbf{w}}, \tilde{p}) = \mathbf{0}$. Of course, we should also note that, due to subsampling of the training data for the cross-validation, $\psi(\mathbf{w}, p)$ is not a deterministic function. However, this indeterminacy is modeled as noise in function evaluations, a phenomenon to which the SPSA method is robust.

Like a traditional gradient descent algorithm, the SPSA method produces a sequence of iterates

$$(\mathbf{w}_{k+1}, p_{k+1}) = (\mathbf{w}_k, p_k) - a_k \hat{\mathbf{g}}_k(\mathbf{w}_k, p_k), \tag{1}$$

where $\hat{\mathbf{g}}_k(\mathbf{w}_k, p_k)$ is an estimate of the gradient $\nabla\psi(\mathbf{w}_k, p_k)$, and $\{a_k\}$ is a nonnegative gain sequence. Under appropriate conditions, this sequence converges almost surely to a local extrema.

Unlike traditional implementations of gradient descent, instead of estimating the gradient by finite difference approximations to each of the partial derivatives (which can be thought of as perturbing each dimension individually), SPSA proceeds by generating a simultaneous perturbation vector. More specifically, let $\boldsymbol{\Delta}_k = (\Delta_{k1}, \ldots, \Delta_{k(n+1)})^T$ be a vector, each component of which is sampled from a Bernoulli ± 1 distribution with probability $1/2$ for each ± 1 outcome.

The SPSA gradient estimation is computed by performing two evaluations of the objective function based on simultaneous perturbation around the current iterate, $(\mathbf{w}_k, p_k)$. That is one computes $\psi((\mathbf{w}_k, p_k) + c_k \cdot \boldsymbol{\Delta}_k)$ and $\psi((\mathbf{w}_k, p_k) - c_k \cdot \boldsymbol{\Delta}_k)$, where $\{c_k\}$ is a nonnegative gain sequence. Then the gradient is approximated as

$$\hat{\mathbf{g}}_k(\mathbf{w}_k, p_k) = \frac{\psi((\mathbf{w}_k, p_k) + c_k \cdot \boldsymbol{\Delta}_k) - \psi((\mathbf{w}_k, p_k) - c_k \cdot \boldsymbol{\Delta}_k)}{2c_k} \begin{pmatrix} (\Delta_k^1)^{-1} \\ \vdots \\ (\Delta_k^{(n+1)})^{-1} \end{pmatrix}. \tag{2}$$

Given the gradient estimate in Eq. (2), the iteration scheme of Eq. (1) was applied until convergence. Thus, the algorithm is a simple gradient descent approach, except that it uses a novel (nondeterministic) estimate of the gradient (as opposed to the usual, say, centered difference approximation).

4.3. Implementation of the SPSA Method

There there several issues which must be addressed in any implementation of the SPSA method (the reader is referred to the paper Spall, 1998b for discussion of the details necessary for implementing the SPSA method). In this section, we will discuss specification of the gain sequences, the initialization and stopping criteria, and several other miscellaneous implementation details.

In the preceding section (Sec. 4.2), we introduced two gain sequences, $\{a_k\}$ and $\{c_k\}$. The satisfactory performance of the SPSA method depends critically on these sequences. Spall specifies these sequences by

$$a_k = \frac{a}{(A + k)^{\alpha}} \quad \text{and} \quad c_k = \frac{c}{k^{\gamma}}, \tag{3}$$

where $a, c, \alpha, \gamma > 0$ and $A \geq 0$. Spall (1998b) recommends choosing $\alpha = 0.602$ and $\gamma = 0.101$. The constant c was set to be the standard deviation of the noise in the objective function, ψ. Through a rather *ad hoc* trial and error process, we determined that a value of $a = 0.75$ seemed to yield a sequence of reasonable step sizes. Finally, we simply set the "stability constant" A to be zero.

We next discuss the initialization, $(\mathbf{w}_0, p_0)$. We chose initial vector of weights to be the vector each of whose elements is $1/2$, i.e., $\mathbf{w}_0 = (0.5, \ldots, 0.5)^T$. We experimented with uniform random initialization, but felt that initialization too far from the "central hump" of the penalty term resulted in a sequence of iterates that where unable to climb from the initialization. The initial choice of the metric, p_0, was determined for each data set. In particular, the optimal value of the metric determined from the "greedy" approach was used as the initialization of the metric.

Our stopping criteria for the optimization were the standard criteria used in classical optimization. That is, we specified "small" step size (where one uses the current value of p_k for the norm) and "small" relative change in the objective function. More specifically,

$$\|(\mathbf{w}_{k+1}, p_{k+1}) - (\mathbf{w}_k, p_k)\|_{p_{k+1}} < \epsilon_1$$

and

$$\left| \frac{\psi(\mathbf{w}_{k+1}, p_{k+1}) - \psi(\mathbf{w}_k, p_k)}{\psi(\mathbf{w}_{k+1}, p_{k+1})} \right| < \epsilon_2,$$

where ϵ_1 and ϵ_2 are constants.

Of course, given that one must specify the gain sequence for the step size, $\{a_k\}$, the first criterion is essentially equivalent to specifying the number of iterations. We found the selection of the appropriate stopping criteria quite difficult and it is clear from our repeated trials that several times too few or too many iterations were performed.

We also mention several other miscellaneous implementation issues. The first issue is that the components $(\mathbf{w}, p)$ are not well scaled. From the initialization, one can only have small changes in the components of $\mathbf{w}$; i.e., $|w_k^i - w_0^i| < 1/2$. However, p has no such constraint. In order to get appreciable movement from the metric during the optimization, the vector of unknowns over which the optimization was performed was actually, $(\mathbf{w}, \log p)$; i.e., interpoint distances were computed using the exponential of the last component of the vector. Also, we note that feasibility was enforced at each update. By this we mean that after performing the update given by Eq. (1) any infeasible component was forced to be in $[0, 1]$; i.e.,

If $w_k^i < 0$, then $w_k^i = 0$

or

If $w_k^i > 1$, then $w_k^i = 1$.

Finally, due to the uncertainty in evaluations of ψ (due to the subsampling for the cross-validation), it is only with small probability that one would ever achieve $w_k^i \in \{0, 1\}$. Thus, in order to perform feature selection/dimensionality reduction, rounding was employed. We rounded at four different levels (the most aggressive of which was to round every weight to either 0 or 1), and computed final classifier performance for the full-cross validation with each of these vectors of (differently rounded) weights.

4.4. Results Using the SPSA Method

We note, first, that we performed the simultaneous optimization on only two of the data sets. These were the smoothed "artificial nose" and the normalized Golub gene data sets. This decision was made exclusively on the basis of time. Even using the SPSA method (so that only two function evaluations are required per iteration) function evaluations are still extremely computationally intensive (cross-validated nearest neighbor classifier on high dimensional data), so several days were required for each run (with termination either by meeting the stopping criteria or after a specified number of iterations).

Due to the stochastic nature of the algorithm, and the fact that one has no guarantees that one is not finding a local minimum, the optimization was repeated multiple times on each of these data sets. Also, due to the nature of numerical algorithms, no weights actually ever become 1 or 0 (unless the step results in an infeasible component which is then forced to the boundary of the feasible region). Thus, we applied various degrees of rounding to the final weight vectors.

Figure 6 shows two typical trials of the SPSA algorithm for the (smoothed) "artificial nose" data set. On the left of the figure are plots of Minkowski p vs. iteration, and on the right are plots of the full cross-validated classifier performance vs. iteration (the full cross-validation was performed every 100 iterations for the purpose of generating this plot). Figure 7 provides analogous plots for the normalized Golub gene data.

Several comments are in order. The first is that the classifier performance achieved by the simultaneous optimization generally exceeds that of the greedy approach. The second observation is that the dimensionality
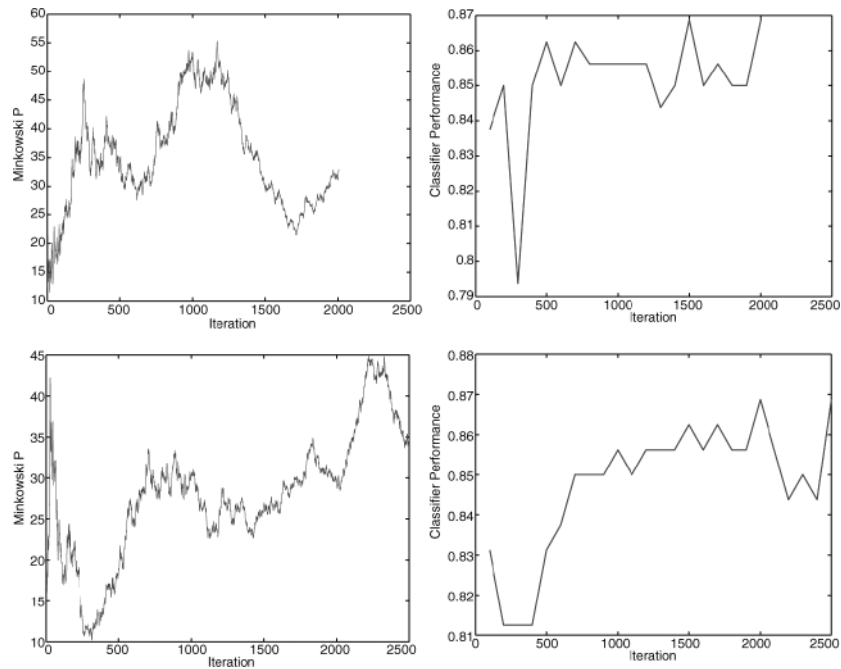

Figure 6. Two trials of the SPSA method for the smooth "artificial nose" data. Minkowski p vs. iteration (left) and full cross-validated classifier performance (right).
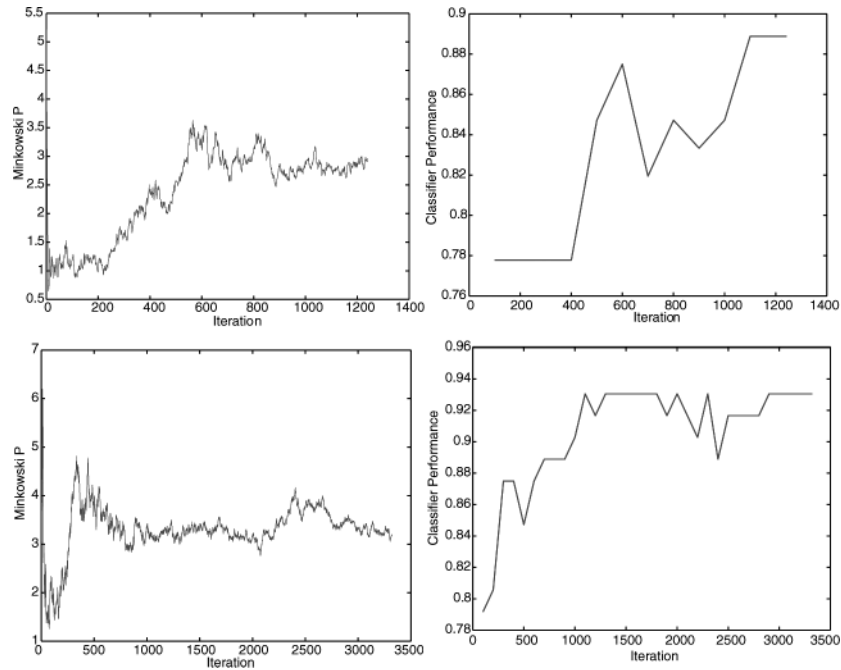
Figure 7. Two trials of the SPSA method for the normalized Golub gene data. Minkowski *p* vs. iteration (left) and full cross-validated classifier performance (right).

reduction achieved by the SPSA method seems to be to select approximately half of the features (after the weights are rounded). Finally, we note that the feasible region seems to contain numerous local minima, as repeated trials resulted in selecting several different values of the Minkowski *p*-parameter and greatly differing feature sets.

5. FINDINGS

We have described above that the simultaneous (w, p) optimization performed better than greedy approach. However, what is not clear is whether the simultaneous optimization is not just over fitting the data in order to produce a classifier which is effective. We reluctantly posit this theory, because intersections of selected features between subsequent trials of the optimization, seemed to be no more significant than one would expect at random.

We recall some facts about the hypergeometric distribution. We let N be the cardinality of the training data set. We denote by M the total number of features selected in experiment one, and by n the total number of features selected in experiment two. Finally, we denote by X the number of features selected in both experiments. Then, under the null hypothesis that the features are selected at random with uniform probability, we have the following formulae for the probability of the intersection consisting of precisely k features

$$P(X = k) = \frac{\binom{M}{k}\binom{N-M}{n-k}}{\binom{N}{n}}$$

and the expected cardinality of the intersection

$$E(X) = \frac{Mn}{N}.$$

Tables 1 and 2 contain the cardinality of the intersections of the feature sets and the expected value for the size of the intersection given

Table 1. Cardinality of intersections of selected features between trials and the predicted value (in parentheses).

Trial	Trial							
	2	3	4	5	6	7	8	9
2	19	4	11	5	11	8	9	11
	(NA)	(6.5)	(10)	(7.5)	(9.5)	(11)	(10.5)	(9)
3	4	13	7	13	6	10	8	4
	(6.5)	(NA)	(6.8)	(5.1)	(6.5)	(7.5)	(4.4)	(6.2)
4	11	7	20	8	9	12	13	10
	(10)	(6.8)	(NA)	(7.9)	(10)	(11.6)	(11.1)	(9.5)
5	5	13	8	15	7	11	9	5
	(7.5)	(5.1)	(7.9)	(NA)	(7.5)	(8.7)	(8.3)	(7.1)
6	11	6	9	7	19	12	13	8
	(9.5)	(6.5)	(10)	(7.5)	(NA)	(11)	(10.5)	(9)
7	8	10	12	11	12	22	13	9
	(11)	(7.5)	(11.6)	(8.7)	(11)	(NA)	(12.2)	(10.4)
8	9	8	13	9	13	13	21	10
	(10.5)	(4.4)	(11.1)	(8.3)	(10.5)	(12.2)	(NA)	(9.9)
9	11	4	10	5	8	9	10	18
	(9)	(6.2)	(9.5)	(7.1)	(9)	(10.4)	(9.9)	(NA)

Table 2. Cardinality of intersections of selected features between trials and the predicted value (in parentheses).

Trial	Trial									
	1	2	3	4	5	6	7	8	9	10
1	872	451	407	450	449	405	420	414	426	419
	(NA)	(443)	(430)	(450)	(436)	(425)	(429)	(423)	(437)	(438)
2	451	895	440	449	441	447	449	438	449	461
	(443)	(NA)	(441)	(462)	(448)	(437)	(440)	(435)	(449)	(450)
3	407	440	869	454	431	412	427	409	434	424
	(430)	(441)	(NA)	(449)	(435)	(424)	(427)	(422)	(436)	(437)
4	450	449	454	910	428	469	457	461	456	458
	(450)	(462)	(449)	(NA)	(455)	(444)	(448)	(442)	(456)	(457)
5	449	441	431	428	882	420	452	428	443	428
	(436)	(448)	(435)	(455)	(NA)	(431)	(434)	(428)	(442)	(443)
6	405	447	412	469	420	861	426	429	454	430
	(425)	(437)	(424)	(444)	(431)	(NA)	(423)	(418)	(432)	(433)
7	420	449	427	457	452	426	867	424	436	448
	(429)	(440)	(427)	(448)	(434)	(423)	(NA)	(421)	(435)	(436)
8	414	438	409	461	428	429	424	856	418	444
	(423)	(435)	(422)	(442)	(428)	(418)	(421)	(NA)	(429)	(430)
9	426	449	434	456	443	454	436	418	884	441
	(437)	(449)	(436)	(456)	(442)	(432)	(435)	(429)	(NA)	(444)
10	419	461	424	458	428	430	448	444	441	886
	(438)	(450)	(437)	(457)	(443)	(433)	(436)	(430)	(444)	(NA)

the null hypothesis. More specifically, the entry in the (i, j) position in the table consists of two integers (except along the diagonal). The upper number indicates the actual cardinality of the set of features that were selected in both trials i and j. The number in parentheses, is the expected cardinality of the intersection under the null hypothesis (i.e., that the SPSA method is simply selecting features by drawing sample uniformly from the feature set). Even cursory inspection of these tables, makes the null hypothesis very attractive.

A question that warrants further investigation is if there is possibly some small subset of features that occur with higher than expected probability. And if in this small number of features that are frequently selected lies most of the discriminant power. Especially with the Golub gene data, it would be very hard to detect if there were some small subset occurring more frequently than is probable than would occur under the null hypothesis.

ACKNOWLEDGMENTS

REFERENCES

Fukunaga, K. (1990). *Statistical Pattern Recognition*. 2nd ed. San Diego: Academic Press.

Golub, T. R., Slonin, D. K. (1999). Molecular classification of cancer; class discovery and class prediction by gene expression monitoring. *Science* 286:531–537.

Priebe, C. E., Marchette, D. J., Solka, J. L. (2000). On the selection of distance for a high-dimensional classification problem. In: Proceedings of the Statistical Computing Section and Section on Statistical Graphics. American Statistical Association, Alexandria, VA, pp. 58–63.

Ramsay, J. O., Silverman, B. W. (1997). *Functional Data Analysis*. New York: Springer-Verlag.

Solka, J. L., Johannsen, D. A. (2003). Classifier optimization via graph complexity (to appear). In: Proceedings of the Army Conference on Applied Statistics.

Spall, J. C. (1998a). An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins APL Tech. Digest* 19(4):817–823.

Spall, J. C. (1998b). Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Trans. Aerospace Electron. Sys.* 34(3):482–492.

Spall, J. C. (2003). *Introduction to Stochastic Search and Optimization*. Hoboken: Wiley-Interscience.

White, J., Kauer, J. S., Dickinson, T. A., Walt, D. R. (1996). Rapid analyte recognition in a device based on optical sensors and the olfactory system. *Anal. Chem.* 68:2191–2202.