From Transformation-Based Dimensionality Reduction to Feature Selection

Mahdokht Masaeli MASAELI.M@NEU.EDU
Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA

Glenn Fung GLENN.FUNG@SIEMENS.COM
Computer Aided Diagnosis and Therapy, Siemens Medical Solutions, USA

Jennifer G. Dy JDY@ECE.NEU.EDU
Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA

Abstract

Many learning applications are characterized by high dimensions. Usually not all of these dimensions are relevant and some are redundant. There are two main approaches to reduce dimensionality: feature selection and feature transformation. When one wishes to keep the original meaning of the features, feature selection is desired. Feature selection and transformation are typically presented separately. In this paper, we introduce a general approach for converting transformation-based methods to feature selection methods through ℓ_1/ℓ_∞ regularization. Instead of solving feature selection as a discrete optimization, we relax and formulate the problem as a continuous optimization problem. An additional advantage of our formulation is that our optimization criterion optimizes for feature relevance and redundancy removal automatically. Here, we illustrate how our approach can be utilized to convert linear discriminant analysis (LDA) and the dimensionality reduction version of the Hilbert-Schmidt Independence Criterion (HSIC) to two new feature selection algorithms. Experiments show that our new feature selection methods out-perform related state-of-the-art feature selection approaches.

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

1. Introduction

Many applications are characterized by high-dimensional data, where not all of the features are important. Dimensionality reduction can be achieved either by feature selection or transformation to a low dimensional space. Feature selection also known as variable selection is the problem of selecting a subset of the original features. In contrast to transformation-based methods which allow modification of the input features to a new feature space; in feature selection, the original representation of the variables is not changed. Feature selection is typically preferred over transformation when one wishes to keep the original meaning of the features and wishes to determine which of those features are important. Moreover, once features are selected, only these features need to be calculated or collected; whereas, in transformation-based methods all input features are still needed to obtain the reduced dimension.

Feature selection algorithms (Kohavi & John, 1997; Guyon & Elisseeff, 2003; Yu & Liu, 2004) can be organized into three main families: filter, wrapper, and embedded methods. These three basic families differ in how the learning algorithm is incorporated in evaluating and selecting features. In filter methods (Kira & Rendell, 1992; Yu & Liu, 2004; Peng et al., 2005; He et al., 2006; Zhao & Liu., 2007; Song et al., 2007), features are pre-selected without running the learning algorithm and are evaluated only through the intrinsic properties of the data. Wrapper methods (Kohavi & John, 1997; Guyon et al., 2002) select features by "wrapping" the search around the learning algorithm and evaluate feature subsets based on the learning performance of the classifier in each candidate feature subset. Embedded meth-

ods (Vapnik, 1998; Zhu et al., 2003) incorporate feature search and the learning algorithm (e.g., classifier) into a single optimization problem formulation. Contrary to filter methods, wrapper and embedded methods select features specific to the classifier. Hence, they are most likely to be more accurate than filter methods on a particular classifier, but the features they choose may not be appropriate for other classifiers. Another limitation of wrapper methods is that wrappers are computationally expensive because they need to train and test the classifier for each feature subset candidate, which can be prohibitive when working with high-dimensional data (such as text, image, gene). Thus, the recent interest in developing fast filter methods (Yu & Liu, 2004; Peng et al., 2005; He et al., 2006; Zhao & Liu., 2007; Song et al., 2007).

Filter methods evaluate features based on some criterion. The goal is to select a subset of features that optimizes this criterion. An exhaustive search of 2^d possible feature subsets (where d is the number of features) is computationally impractical. Heuristic search strategies such as greedy approaches (e.g., sequential forward/backward search (Kittler, 1978)) are commonly used (Song et al., 2007) but can lead to local optima. Random search methods, such as genetic algorithms, add some randomness in the search procedure to help escape from local optima. Exhaustive, greedy and random searches are subset search methods because they evaluate each candidate subset. In some cases when the dimensionality is very high, one can only afford an individual search. Individual search methods evaluate each feature individually according to a criterion (Guyon & Elisseeff, 2003; He et al., 2006; Zhao & Liu., 2007). They then select features, which either satisfy a condition or are top-ranked. The problem with individual search techniques is that they ignore feature interaction and dependencies. In (Yu & Liu, 2004), they select relevant features individually and then add a separate redundancy removal step to account for linear correlation between features.

Feature selection is an NP-hard optimization problem over a discrete space. Transformation-based dimensionality reduction methods, on the other hand, is as an optimization problem over a continuous feature space. Inspired by shrinkage techniques (Tibshirani, 1996; Yuan & Lin, 2006; Donoho, 2004), instead of solving feature selection as a discrete optimization, we relax and formulate the problem as a continuous optimization problem. However, contrary to least absolute shrinkage and selection operator (LASSO) (Tibshirani, 1996; Yuan & Lin, 2006) where variable selection is based on regression, our approach is based on transformation-based dimensionality reduction criteria. Research on feature selection and transformation methods are typically presented separately. In this paper, we show a general approach for converting transformation-based methods to filter feature selection methods; thereby, inheriting the rich source of transformation dimensionality reduction algorithms in the literature. In feature selection, it is important to select relevant features and to remove the redundant ones (Kohavi & John, 1997; Yu & Liu, 2004; Peng et al., 2005). Another benefit of our feature selection based on transformation approach is that we optimize both simultaneously in our formulation. Although the formulation in this paper can be applied to either supervised or unsupervised transformation methods, due to clarity and space limitation reasons, we focus this paper to the supervised case. Extension to the unsupervised case will be explored in an extended version of this paper. More specifically, we show how our approach can be utilized to convert linear discriminant analysis (LDA) (Fukunaga, 1990) and the dimensionality reduction version of the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) to two new continuous filter feature selection algorithms, we call linear discriminant feature selection (LDFS) and Hilbert-Schmidt feature selection (HSFS) respectively.

This paper is organized as follows. In Section 2, we provide our general formulation for converting transformation-based dimensionality reduction algorithms to do feature selection. In Section 3, we describe how to optimize our general formulation. Then, in Section 4, we show how to build a feature selection algorithm with a linear transformation-based algorithm, in particular LDA, using our formulation; and, in Section 5, we illustrate our formulation on a non-linear HSIC criterion. We report and discuss our experimental results in Section 6. Finally, we summarize and conclude in Section 7.

2. Transformation-Based Feature Selection

In this work, we propose a general approach for translating transformation-based dimensionality reduction methods to solve the feature selection problem. Let the original data be $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T$ with n samples and each sample $\mathbf{x}_i \in \mathbb{R}^d$, where X is an $n \times d$ matrix. Transformation-based methods can be either linear or non-linear.

In linear transformation-based methods (such as, LDA), the goal is to find a transformation matrix, $W = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q] \in \mathbb{R}^{d \times q}$ to transform the origi-

nal high-dimensional data $\mathbf{x_i}$ into a lower-dimensional form $\mathbf{y}_i \in \mathbb{R}^q$, where $q \ll d$, $\mathbf{y}_i = W^T\mathbf{x}_i$. W is selected such that some criterion, $\mathcal{J}(XW)$ is optimized.

In general non-linear transformation-based methods (such as, kernel LDA (Mika et al., 1999)), the goal is to find a non-linear mapping to a lower-dimensional $\mathbf{y}_i \in \mathbb{R}^q$, $\mathbf{y}_i = \Phi(\mathbf{x}_i)$ that optimizes some criterion $\mathcal{J}(\Phi(X))$. We do not operate directly on these types of transformation, instead we work on subspace versions of these algorithms. The idea is to find a lower dimensional central subspace of X (XW) that captures the same information with respect to the target as that of the full dimensional X. This can be expressed as finding W that optimizes a criterion of the form $\mathcal{J}(\Phi(XW))$. An example of a central subspace method is kernel dimensionality reduction (Fukumizu et al., 2004). The advantage of working on the central subspace is that the data may contain dimensions irrelevant to the task, applying $\Phi(XW)$ avoids those noisy subspaces. The other reason is that the projection W operates on are the original features X rather than on the non-linearly transformed features $\Phi(X)$, which makes it more amenable for interpreting which of the original features are important.

General Formulation. To explain our general formulation, we find it useful to define data $X_{n\times d}$ in terms of its features (columns), $\mathbf{f}_j$, as well as its samples (rows), $\mathbf{x}_i$, $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_d]$. In feature selection, it is important to have information about the original data space. Let W be a transformation matrix of size $d \times q$ (in our experiments, we set $q = d$). Given a transformation dimensionality reduction method with criterion $\mathcal{J}(XW)$, to perform feature selection, we optimize the following modified criterion with respect to W:

$$\min_{W\in\mathbb{R}^{d\times q}} \mathcal{J}(XW) + \lambda \sum_{j=1}^{d} ||\mathbf{w}_j||_\infty, \qquad (1)$$

where $\mathbf{w}_j$ is the j-th row of matrix W and $||\mathbf{w}_j||_\infty$ stands for the infinity norm of $\mathbf{w}_j$, and λ is a regularization parameter.

To understand why this optimization formulation leads to a feature selection solution, let us first describe how the structure of W should be to achieve feature selection. Let w_{jk} be the elements of transformation matrix W. If feature $\mathbf{f}_j$ is not selected by the algorithm, all the elements of the the j-th row of W should be zero, $\forall k, w_{jk} = 0$. Thus, feature $\mathbf{f}_j$, the j-th column of X will not contribute to the criterion $\mathcal{J}(XW)$. If feature $\mathbf{f}_j$ is selected by the algorithm, it means that at least one of the elements of the j-th row of W is nonzero for the feature $\mathbf{f}_j$ to contribute to the criterion, $\mathcal{J}(XW)$. Therefore, forcing W to have more zero rows can be interpreted as selecting less features. Using this idea, we enforce sparsity on the rows of W by adding an ℓ_1/ℓ_∞ regularization term to our criterion $\mathcal{J}(XW)$. ℓ_∞ norm of a vector $\mathbf{w}_j$ is the maximum of the absolute value of the elements of $\mathbf{w}_j$. The ℓ_1 norm induces sparsity (Tibshirani, 1996; Donoho, 2004). Basically, we are inducing sparsity on the maximum absolute value (the bounding hypercube) of the elements of each row; thereby, pushing all the elements of each row to zero.

Sparse dimensionality reduction methods, such as sparse principal component analysis (Zou et al., 2006) and sparse LDA (Moghaddam et al., 2006), only apply an ℓ_1 norm regularization on W. This would set individual sparsity on the elements of W but would not necessarily achieve feature selection. In order to impose sparsity on W and reach the desired configuration to decide what features to keep, an ℓ_1/ℓ_∞ is added where λ is a regularization coefficient. In general, one can set the regularizer term to be in the form of ℓ_1/ℓ_p with $1 \leq p \leq \infty$. Choosing $p = 1$ would result in individual sparsity patterns for each row of W. In the case of using transformation matrix W as a means for feature selection, however, we need to push for all-zero rows in W to decide to remove the corresponding features. Therefore, having individual sparsity patterns for each row is not suitable. Increasing p increases the sparsity sharing between the elements in each row. Because we need the zeros to be shared for elements in each row of W corresponding to a specific feature, we set our regularizer to be $p = \infty$, which promises full sharing of the elements. The general combination of norms ℓ_m/ℓ_p type of regularizer is a case of block-norm regularization, such as GroupLASSO (Yuan & Lin, 2006). However, our focus is different from their's in the sense that they are concerned with grouping predictor variables in regression, whereas our focus is to group the coefficients of rows in a transformation matrix.

LASSO (Tibshirani, 1996) selects variables by adding an ℓ_1 norm regularizer on the predictor weights. Unlike LASSO, our formulation learns a transformation matrix W that can create q new features that optimizes a criterion. However, unlike transformation methods, W only has nonzero weights on the selected original features (rows); thus, performing feature selection. Interestingly, assuming the same objective criterion, a LASSO-type or ℓ_1 penalty is a degenerate version of our formulation which selects the input features for generating only one new feature (i.e., $q = 1$) that optimizes the objective.

The regularization parameter λ controls the trade-off between the criterion $\mathcal{J}(XW)$ and sparsity. Increasing λ means forcing more rows to be zero which will result in removing more features. The extreme case, where $\lambda = 0$, results in selecting all of the features. Conversely, for very large λ, no features are selected ($W \equiv 0$). Therefore, ranging λ from zero to infinity can be interpreted as ranging the number of selected features from d to zero.

To solve the feature selection problem, it is important to find the most relevant features and remove redundancy. Note that feature selection can also be achieved by simply limiting W of size $d \times d$ to be a diagonal matrix and adding an ℓ_1 norm penalty. However, this does not take redundancy among the features into account. The formulation we provide is more general. The diagonal W case can be derived from our general formulation as a special case. Moreover, by allowing the off-diagonal elements of W to be nonzero, we do not only select the features but also learn the linear transformations using only these features that can lead to the optimal criterion value. We thus consider interactions among the features into account in this way. Our formulation also automatically removes redundancy. Since we consider the criterion value of the linear subspace in which our selected features reside, features that are correlated to the selected features will not decrease $\mathcal{J}(XW)$ but will increase the ℓ_1/ℓ_∞ norm, thereby these redundant features are automatically removed. Our general formulation thus takes both feature relevance and redundancy into account.

3. Optimization Algorithm

To optimize for the ℓ_∞ norm, we utilize a vector of dummy variables, $\mathbf{s} = [s_j]$ to represent the maximum absolute value of the elements of rows $\mathbf{w}_j$, $j = 1, \ldots, d$. This means that the absolute value of every element in row j, $|w_{jk}|$ should be smaller than or equal to s_j and that all s_j should be non-negative. Hence, Formulation (1) can be rewritten as the following formulation:

$$\begin{aligned} \min_{W,\mathbf{s}} \quad & \mathcal{J}(XW) + \lambda \sum_{j=1}^{d} s_j \\ \text{s.t.} \quad & |w_{jk}| \leq s_j, \\ & s_j \geq 0, \forall j, k \end{aligned} \tag{2}$$

This is equivalent to finding a solution to the following linear constraint problem:

$$\begin{aligned} \min_{W,\mathbf{s}} \quad & \mathcal{J}(XW) + \lambda \sum_{j=1}^{d} s_j \\ \text{s.t.} \quad & -s_j \leq w_{jk} \leq s_j \\ & s_j \geq 0, \forall j, k \end{aligned} \tag{3}$$

which we solved using a Quasi-Newton method as implemented in Mark Schmidt's optimization toolbox.[1] We used part of the toolbox that implements a two-metric projection method for optimization with box constraints, where limited-memory BFGS (L-BFGS (Nocedal & Wright, 2003)) updates are used (no Hessian information required) in computing the step direction, and a backtracking line search is used to find a step satisfying an Armijo condition that guaranties improvement at every iteration. Since computing and storing the Hessian are both computationally expensive, the L-BFGS algorithm is a good candidate to do the optimization, avoiding high computational cost while achieving super-linear convergence. The algorithm only needs the value of the function and its derivative at each point to approximate the Hessian (via low-rank updates) and to find the updates of the solution at each iteration.

4. Linear Discriminant Feature Selection

A popular supervised dimensionality reduction method is linear discriminant analysis (LDA) (Fukunaga, 1990). LDA finds the optimal linear transformation W, which minimizes the within-class distance and maximizes the between-class distance simultaneously. The criterion $\mathcal{J}(XW)$ it optimizes is:

$$\mathcal{J}(XW) = -\text{LDA}(XW) = -\frac{W^T S_B W}{W^T S_W W} \tag{4}$$

where S_B is the between class scatter matrix and S_W is the within class scatter defined by:

$$\begin{aligned} S_B &= \sum (\mu_c - \bar{x})(\mu_c - \bar{x})^T \\ S_W &= \sum_{c}^{c} \sum_{x_i \in c} (x_i - \mu_c)(x_i - \mu_c)^T \end{aligned} \tag{5}$$

in which $\bar{x}$ is the mean of the data points X, and μ_c is the mean of the data points that belongs to class c.

Plugging $-\text{LDA}(XW)$ to Formulation (3), we introduce a new convex feature selection algorithm, *Linear Discriminant Feature Selection(LDFS)* based on LDA:

$$\min_{W \in R^{d \times q}} -\frac{W^T S_B W}{W^T S_W W} + \lambda \sum_{j=1}^{d} ||\mathbf{w}_j||_\infty \tag{6}$$

Since we only need $\mathcal{J}(XW)$ and its gradient to optimize using the two-metric projection L-BFGS method,

[1] http://www.cs.ubc.ca/~schmidtm/Software/minFunc

here we show the expansion of $\nabla\mathcal{J}(XW)$.

$$\nabla\mathcal{J}(XW) = -2\frac{S_B W}{W^T S_W W} + (\frac{S_W W}{W^T S_W W})\frac{W^T S_B W}{W^T S_W W} \tag{7}$$

5. Hilbert-Schmidt Feature Selection

In this second example, we show how a non-linear dimensionality reduction based on the central subspace can be converted to a feature selection algorithm. In particular, we apply it to the Hilbert-Schmidt Independence Criterion (HSIC). Recently, (Gretton et al., 2005) suggest a way for measuring dependence between variables without explicitly estimating the joint distribution of the random variables. They measure dependence by mapping variables in reproducing kernel Hilbert spaces (RKHS) to test high order moments which enables them to measure the non-linear dependence between variables without the need for knowing the probability distributions of these variables. HSIC is a criterion that belongs to this class of dependence criterion. This criterion is based on a cross-covariance operator on RKHS to test for dependence.

We define a mapping $\phi(x) \in \mathcal{F}$ from every x in the sample domain, $\mathcal{X}$, to the RKHS $\mathcal{F}$. $\mathcal{F}$ is called an RKHS if the inner product of the mappings of the data points can be represented by a kernel function, $k(x, x^T) := \langle \phi(x), \phi(x^T) \rangle$. Another RKHS, $\mathcal{G}$, can be defined on another data space, $\mathcal{Y}$, similarly with the mapping $\psi(y)$. To represent the dependency between the two introduced domains, a cross-covariance operator is defined as

$$\mathcal{C}(x, y) = \mathcal{E}_{xy}[(\phi(x) - \mu_x) \otimes (\psi(y) - \mu_y)]$$

where $\otimes$ is the tensor product. The HSIC criterion is defined as the square of the Hilbert-Schmidt norm of $\mathcal{C}(x, y)$, $HSIC(\mathcal{P}_{x,y}, \mathcal{F}, \mathcal{G}) = ||\mathcal{C}_{xy}||^2_{HS}$.

However, we do not have the distribution $\mathcal{P}_{x,y}$. Given n observations $\mathcal{Z} := \{(x_1, y_1), ..., (x_n, y_n)\}$ from $\mathcal{P}_{x,y}$, HSIC can be empirically estimated as:

$$\begin{aligned} &\text{HSIC}^2(\mathcal{Z}, \mathcal{F}, \mathcal{G}) := \frac{1}{n^2} tr(KHLH) \\ &\text{s.t.} \\ &H, K, L \in R^{n \times n}, \\ &K_{ij} := k(x_i, x_j), L_{ij} := l(y_i, y_j) \\ &H = I - \tfrac{1}{n} 1_n 1_n^T. \end{aligned} \tag{8}$$

For simplicity, let us use the notation $\text{HSIC}(X, Y) = \text{HSIC}^2(\mathcal{Z}, \mathcal{F}, \mathcal{G})$. Here, we use $-\text{HSIC}(XW, Y)$ as $\mathcal{J}(XW, Y)$ to measure the dependency between the transformed X and class labels, Y. The Hilbert-Schmidt Feature Selection (HSFS) objective is now:

$$\min_{W \in R^{d \times q}} -\text{HSIC}(XW, Y) + \lambda \sum_{j=1}^{d} ||\mathbf{w}_j||_\infty \tag{9}$$

In this paper, we use a Gaussian kernel as the kernel function in HSIC. The kernel function in the projected space is then, $k_{ij} = exp[\frac{-1}{2\sigma^2} \mathbf{w}_l^T R_{ij} \mathbf{w}_l]$, where $R_{ij} = (x_i - x_j)(x_i - x_j)^T$, where x_i is the i-th observation and $\mathbf{w}_l$ is a row of the W matrix. Next, we explicity show the derivatives needed to solve Formulation (9) using the two-metric projection L-BFGS method. The gradient of $\mathcal{J}(XW, Y)$ is:

$$\begin{aligned} \nabla\mathcal{J}(XW, Y) &= -\frac{1}{n^2} \frac{\partial tr(KHLH)}{\partial W} \\ &= -\frac{1}{n^2} \left[\frac{\partial tr(KHLH)}{\partial \mathbf{w}_1} \cdots \frac{\partial tr(KHLH)}{\partial \mathbf{w}_k} \right] \end{aligned} \tag{10}$$

The derivative with respect to each row is:

$$\frac{\partial tr(KHLH)}{\partial \mathbf{w}_l} = vec(HLH)(\frac{\partial K}{\partial \mathbf{w}_l})^T \tag{11}$$

where $\frac{\partial K}{\partial \mathbf{w}_l} = \left[\frac{\partial k_{11}}{\partial \mathbf{w}_l} \cdots \frac{\partial k_{1n}}{\partial \mathbf{w}_l} \frac{\partial k_{21}}{\partial \mathbf{w}_l} \cdots \frac{\partial k_{nn}}{\partial \mathbf{w}_l} \right]$. For a Gaussian kernel, the derivative with respect to $\mathbf{w}_l$ is $\frac{\partial k_{ij}}{\partial \mathbf{w}_l} = -\frac{k_{ij}}{\sigma^2} \mathbf{w}_l^T R, \forall i, j$.

Note that we use HSIC as the example here, rather than KDR (Fukumizu et al., 2004), which belongs to the same family as HSIC because the HSIC criterion has been applied to perform feature selection in (Song et al., 2007). However, they applied heuristic greedy search techniques, sequential forward and backward search as the search technique. Here, we have a filter method which uses the same criterion, considers the whole set of features and does the feature selection as a relaxed continuous optimization problem. We can thus compare the performance of our formulation as a search technique compared to sequential search using the same criterion.

6. Experiments

In this section, we evaluate the performance of our two new feature selection algorithms, LDFS (LDA-ℓ_1/ℓ_∞) and HSFS (HSIC-ℓ_1/ℓ_∞), compared to the following state-of-the-art feature selection algorithms: FOHSIC (Song et al., 2007) which applies sequential forward search with the HSIC criterion, Support Vector Machine (SVM) Recursive Feature Elimination (RFE) (Guyon et al., 2002) which is a wrapper feature selection method around an SVM classifier with backward elimination where features with the lowest

Table 1. 5-fold Cross-Validated Classification Errors on Real Data

	WINE	GLASS	ARCENE	GISETTE	DEXTER	MADELON
NUM. OF ORIGINAL FEATURES	13	10	10000	5000	20000	500
NUM. OF SELECTED FEATURES	2	2	10	6	20	10
HSFS (HSIC-ℓ_1/ℓ_∞)	**4.58**	**24.8**	**11.99**	**14.97**	**26.54**	**24.72**
FOHSIC	4.96	25.11	13.56	21.11	29.93	26.55
ℓ_1-SVM	6.89	41.3	23.52	21.15	35.23	35.22
SVM-RFE	8.94	45.32	23.76	18.73	35.08	39.92
RELIEF	8.98	48.93	21.94	19.92	42.9	33.42
LDFS (LDA-ℓ_1/ℓ_∞)	8.96	47.93	22.11	37.22	43.85	34.74
LS	9.24	52.26	28.49	46.86	45.55	37.22

SVM squared weights are recursively eliminated, ℓ_1-SVM (Zhu et al., 2003) which is an embedded sparse method that performs feature selection by utilizing an ℓ_1 regularizer on a linear SVM formulation rather than the standard ℓ_2 margin, Relief (Kira & Rendell, 1992) which is a filter feature selection method that evaluates features based on how well the feature differentiates between neighboring instances from different classes versus from the same class, and the Laplacian score (LS) (He et al., 2006) which is an individual search-based filter feature selection approach that utilizes the Fisher criterion. The algorithms were implemented using the Spider machine learning toolbox.[2] We ran experiments on six data sets: glass, wine, Arcene, Gisette, Dexter and Madelon. Glass and wine are small data sets from the UCI repository (Blake & Merz, 1998); Arcene with $10,000$ features, Gisette with $5,000$, Dexter with $20,000$ and Madelon with 500 features are from the NIPS 2003 feature selection challenge.[3] These data sets provide us with a wide range of feature dimensions. For datasets with more than 1000 samples, we randomly subsampled 1000 instances.

We compare the results of HSFS and LDFS versus the other methods by showing plots of 5-fold cross-validated errors when using a SVM classifier with the features selected by these methods versus the number of selected features in Figures 1 to 5. We ran SVM with a Gaussian kernel with width 1 using LIBSVM.[4] The SVM regularization parameter is tuned via cross-validation. For our methods, we control the number of selected features through λ. Since λ is the parameter that tunes the sparsity of W and therefore the number of selected features, in our algorithm λ is increased until the desired number of rows of W (i.e., the number of discarded features) are close to zero (the maximum value of the row is less than 0.01). We also provide a "snap shot" of these errors in Table 1 by reporting the 5-fold cross-validated errors for a fixed number of selected features. The total number of features and number of selected features for every data set is also included in the table. For every dataset, the best performing algorithm is highlighted in bold font.

These results show that the proposed HSFS algorithm is consistently the best in all cases. It is followed by FOHSIC which optimizes the same HSIC criterion but utilizes a greedy sequential forward search strategy. Notice in Figures 3 and 4, as a greedy strategy, FOHSIC did better than HSFS for 4 features, but then performed worse than HSFS later when we keep more features (6 to 10). RELIEF, SVM-RFE and ℓ_1-SVM performed worse than the two HSIC-based methods. The Laplacian score optimizes the same criterion as LDA. Since the LDA criterion only captures linear discriminant relationships, whereas HSIC can capture nonlinear dependencies with the class labels, the performance of the LDA-based methods (LDFS and Laplacian score) are poorer compared to the two HSIC-based methods. However, comparing the two LDA-based feature selection approaches, the proposed LDFS method performs consistently much better than its counterpart, the Laplacian score. The reason is that in the Laplacian score algorithm, features are ranked one by one neglecting the relation between the sets of features; but in LDFS, the interactions among the whole set of features are considered. LDFS takes both feature relevance and redundancy into account.

Time Complexity. Our formulation optimizes for W which has d^2 parameters, where d is the number of original features. The computational complexity of an iteration for the quasi-Newton method we are using is $O(m^2)$, (with superlinear convergence) where m is the number of parameters. The computational complexity of our HSFS method is thus $O(d^4)$. However, in general we can set W to be $d \times q$, where $q << d$, which

[2] http://www.kyb.tuebingen.mpg.de/bs/people/spider
[3] http://www.nipsfsc.ecs.soton.ac.uk/datasets/
[4] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

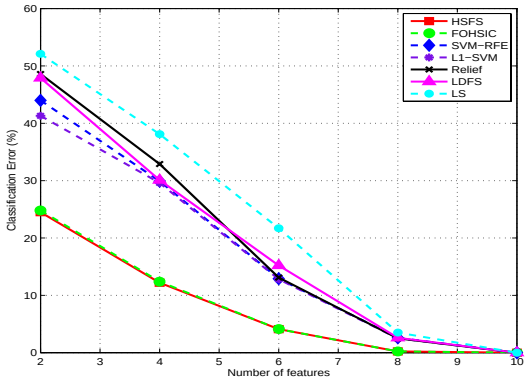

Figure 1. 5-fold cross-validated error as a function of the number of features selected for the Glass data.

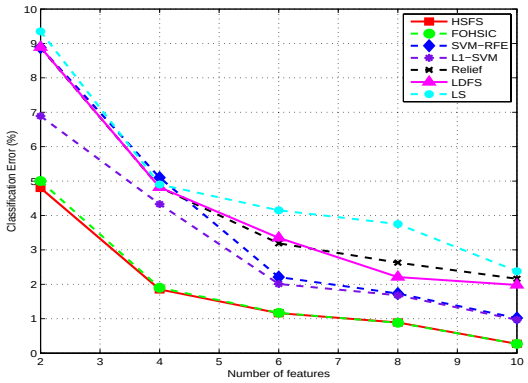

Figure 2. 5-fold cross-validated error as a function of the number of features selected for the Wine data.

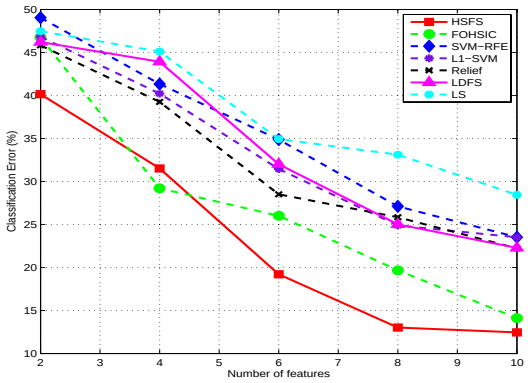

Figure 3. 5-fold cross-validated error as a function of the number of features selected for the Arcene data.

leads to a reduced complexity $O(d^2q^2)$. In comparison, FOHSIC has computational complexity $O(d^3n^2)$,

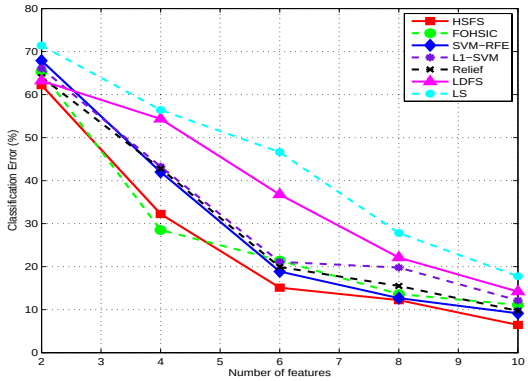

Figure 4. 5-fold cross-validated error as a function of the number of features selected for the Gisette data.

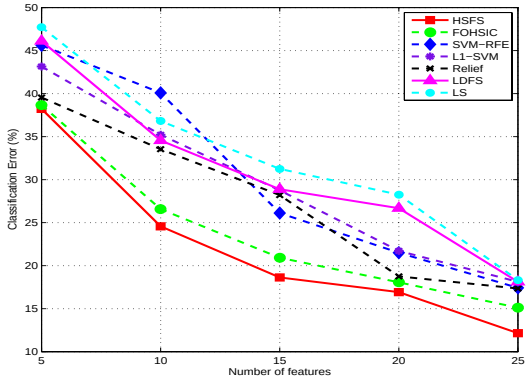

Figure 5. 5-fold cross-validated error as a function of the number of features selected for the Madelon data.

where n is the number of samples.

7. Conclusion

Feature selection is an NP-hard combinatorial optimization problem, whereas feature transformation is an optimization over continuous space. In this paper, we relaxed the feature selection problem into a continuous optimization problem and showed how to convert a general transformation-based dimensionality reduction algorithm into a feature selection formulation. In particular, we showed how our formulation can be utilized to convert linear discriminant analysis (LDA) (Fukunaga, 1990) and the dimensionality reduction version of the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) to two new continuous filter feature selection algorithms. Experiments on real data show that our continuous optimization formulation led to better performance compared

to their discrete search counterparts. Moreover, the ability of our approach to optimize for both relevance and redundancy removal simultaneously, enabled it to outperform competing feature selection methods. Note that the formulation in this paper can be applied to either supervised or unsupervised transformation methods. Because of space limitations, we focused this paper to the supervised case. In (Masaeli et al., 2010), we show how this can be applied to principal component analysis. We will explore other unsupervised dimensionality reduction methods in a future extension of this work.

Acknowledgments

We thank the support from NSF IIS-0915910 and NSF Career IIS-0347532.

References

Blake, C.L. and Merz, C.J. UCI repository of machine learning databases. In *http://www.ics.uci.edu/∼mlearn/MLRepository.html*, 1998.

Donoho, D. For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution. Technical report, Statistics Department, Stanford University, 2004.

Fukumizu, K., Bach, F., and Jordan, M. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

Fukunaga, K. *Statistical Pattern Recognition (second edition)*. Academic Press, San Diego, CA, 1990.

Gretton, A., Bousquet, O., Smola, A., and Scholkopf, B. Measuring statistical dependence with hilbert-schmidt norms. In *Int'l Conf. on Algorithmic Learning Theory*, pp. 63–77, 2005.

Guyon, I. and Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

He, X., Cai, D., and Niyogi, P. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*, pp. 507–514, 2006.

Kira, K. and Rendell, L. A. A practical approach to feature selection. In *Int'l Conf. in Machine Learning*, pp. 249–256, 1992.

Kittler, J. Feature set search algorithms. In *Pattern Recognition and Signal Processing*, pp. 41–60, 1978.

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

Masaeli, M., Yan, Y., Cui, Y., Fung, G., and Dy, J. G. Convex principal feature selection. In *SIAM Int'l Conf. on Data Mining*, pp. 619–628, 2010.

Mika, S., Ratsch, G., Western, J., Scholkoph, B., and Muller, K.R. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX*, pp. 41–48, 1999.

Moghaddam, B., Weiss, Y., and Avidan, S. Generalized spectral bounds for sparse LDA. In *Int'l Conf. on Machine Learning*, pp. 641–648, 2006.

Nocedal, J. and Wright, S. *Numerical Optimization (2nd ed.)*. Springer-Verlag, Berlin, New York, 2003.

Peng, H., Long, F, and Ding, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pp. 1226–1238, 2005.

Song, L., Smola, A. J., Gretton, A., Borgwardt, K. M., and Bedo, J. Supervised feature selection via dependence estimation. In *Int'l Conf. on Machine Learning*, pp. 823–830, 2007.

Tibshirani, R. Regression shrinkage and selection via the lasso. *J. Royal Stat. Soc. B*, 58(1):267–288, 1996.

Vapnik, V. *Statistical Learning Theory*. Wiley, New York, 1998.

Yu, L. and Liu, H. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

Yuan, M. and Lin, Y. Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. B*, 1(68):49–67, 2006.

Zhao, Z. and Liu., H. Spectral feature selection for supervised and unsupervised learning. In *Int'l Conf. on Machine Learning*, pp. 1151–1157, 2007.

Zhu, Ji, Rosset, Saharon, Hastie, Trevor, and Tibshirani, Robert. 1-norm support vector machines. In *Advances in Neural Info. Proc. Systems*, 2003.

Zou, H., Hastie, T., and Tibshirani, R. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):262–286, 2006.