High-Dimensional Feature Selection by Feature-Wise Kernelized Lasso

Makoto Yamada[1], Wittawat Jitkrittum[2], Leonid Sigal[3], Eric P. Xing[4], and Masashi Sugiyama[2]
[1]Yahoo! Labs, 701 1st Ave., Sunnyvale, CA, 94089, USA
[2]Tokyo Institute of Technology, 2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan
[3]Disney Research Pittsburgh, 4720 Forbes Ave., Pittsburgh, PA 15213
[4]Carnegie Mellon University, Pittsburgh, PA, 15213
makotoy@yahoo-inc.com, wittawatj@gmail.com,
lsigal@disneyresearch.com, epxing@cs.cmu.edu, sugi@cs.titech.ac.jp

August 22, 2013

Abstract

The goal of supervised feature selection is to find a subset of input features that are responsible for predicting output values. The *least absolute shrinkage and selection operator* (Lasso) allows computationally efficient feature selection based on linear dependency between input features and output values. In this paper, we consider a *feature-wise* kernelized Lasso for capturing non-linear input-output dependency. We first show that, with particular choices of kernel functions, non-redundant features with strong statistical dependence on output values can be found in terms of kernel-based independence measures such as the Hilbert-Schmidt independence criterion (HSIC). We then show that the globally optimal solution can be efficiently computed; this makes the approach scalable to high-dimensional problems. The effectiveness of the proposed method is demonstrated through feature selection experiments for classification and regression with thousands of features.

1 Introduction

Finding a subset of features in high-dimensional supervised learning is an important problem with many real-world applications such as gene selection from microarray data (Xing *et al.*, 2001; Ding and Peng, 2005; Suzuki *et al.*, 2009; Huang *et al.*, 2010), document categorization (Forman, 2008), and prosthesis control (Shenoy *et al.*, 2008).

1.1 Problem Description

Let $\mathcal{X}(\subset \mathbb{R}^d)$ be the domain of input vector $\boldsymbol{x}$ and $\mathcal{Y}(\subset \mathbb{R})$ be the domain of output data[1] y. Suppose we are given n independent and identically distributed (i.i.d.) paired samples,

$$\{(\boldsymbol{x}_i, y_i) \mid \boldsymbol{x}_i \in \mathcal{X}, \ \ y_i \in \mathcal{Y}, \ i = 1, \dots, n\},$$

drawn from a joint distribution with density $p_{\mathrm{x,y}}(\boldsymbol{x}, y)$. We denote the original data by

$$\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n] \in \mathbb{R}^{d \times n},$$
$$\boldsymbol{y} = [y_1, \dots, y_n]^\top \in \mathbb{R}^n,$$

where $^\top$ denotes the transpose.

The goal of supervised feature selection is to find m features ($m < d$) of input vector $\boldsymbol{x}$ that are responsible for predicting output y.

---

[1] $\mathcal{Y}$ could be either continuous (i.e., regression) or categorical (i.e., classification). Structured outputs can also be handled in our proposed methods.

1.2 Lasso

The *least absolute shrinkage and selection operator* (Lasso) (Tibshirani, 1996) allows computationally efficient feature selection based on the assumption of linear dependency between input features and output values.

The Lasso optimization problem is given as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{X}^\top \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1,$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_d]^\top$ is a regression coefficient vector, α_k denotes the regression coefficient of the k-th feature, $\|\cdot\|_1$ and $\|\cdot\|_2$ are the ℓ_1- and ℓ_2-norms, and $\lambda > 0$ is the regularization parameter. The ℓ_1-regularizer in Lasso tends to produce a sparse solution, which means that the regression coefficients for irrelevant features become zero. Lasso is particularly useful when the number of features is larger than the number of training samples (Tibshirani, 1996). Furthermore, various optimization software packages were developed for efficiently computing the Lasso solution (Boyd and Vandenberghe, 2004; Daubechies *et al.*, 2004; Combettes and Wajs, 2005; Kim *et al.*, 2007; Yin *et al.*, 2008; Wright *et al.*, 2009; Tomioka *et al.*, 2011).

However, a critical limitation of Lasso is that it cannot capture non-linear dependency.

1.3 Instance-Wise Non-Linear Lasso

To handle non-linearity, the *instance-wise* non-linear Lasso was introduced (Roth, 2004), where the original instance $\boldsymbol{x}$ is transformed by a non-linear function $\boldsymbol{\psi}(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d'}$. Then the Lasso optimization problem is expressed as

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n} \quad \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1,$$

where $A_{i,j} = \boldsymbol{\psi}(\boldsymbol{x}_i)^\top \boldsymbol{\psi}(\boldsymbol{x}_j) = A(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_n]^\top$ is a regression coefficient vector, and β_j is a coefficient of the j-th basis $A(\boldsymbol{x}, \boldsymbol{x}_j)$.

The instance-wise non-linear Lasso gives a sparse solution in terms of instances, but not features. Therefore, it cannot be used for feature selection.

1.4 Feature-Wise Non-Linear Lasso (Feature Vector Machine)

To obtain sparsity in terms of features, the *feature-wise* non-linear Lasso was proposed (Li *et al.*, 2006).

The key idea is to apply a non-linear transformation in a feature-wise manner, not in an instance-wise manner. More specifically, let us represent the sample matrix $\boldsymbol{X}$ in a feature-wise manner as

$$\boldsymbol{X} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]^\top \in \mathbb{R}^{d \times n},$$

where $\boldsymbol{u}_k = [x_{k,1}, \ldots, x_{k,n}]^\top \in \mathbb{R}^n$ is the vector of the k-th feature for all samples. Then the feature vector $\boldsymbol{u}_k$ and the output vector $\boldsymbol{y}$ are transformed by a non-linear function $\boldsymbol{\phi}(\cdot) : \mathbb{R}^n \to \mathbb{R}^p$. The Lasso optimization problem in the transformed space is given as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2} \|\boldsymbol{\phi}(\boldsymbol{y}) - \sum_{k=1}^d \alpha_k \boldsymbol{\phi}(\boldsymbol{u}_k)\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1, \tag{1}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_d]^\top$ is a regression coefficient vector and α_k denotes the regression coefficient of the k-th feature. By using the kernel trick (Schölkopf and Smola, 2002), Eq.(1) was shown to be equivalently expressed as the following quadratic programming (QP) problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2} \boldsymbol{\alpha}^\top \boldsymbol{D} \boldsymbol{\alpha},$$

$$\text{s.t.} \quad \forall k, \quad |\boldsymbol{\alpha}^\top \boldsymbol{d}_k - D(\boldsymbol{u}_k, \boldsymbol{y})| \leq \frac{\lambda}{2}, \tag{2}$$

Table 1: Feature selection methods.

Method	Dependency	Optimization	Primal/Dual	Scalability w.r.t. # of features	Structured output
Lasso	Linear	**Convex**	**Primal**	**Highly scalable**	Not available
mRMR	**Non-linear**	Greedy	—	Scalable	**Available**
Greedy HSIC	**Non-linear**	Greedy	—	Scalable	**Available**
HSFS	**Non-linear**	Non-convex	—	Not scalable	**Available**
FVM	**Non-linear**	Non-convex[†]	Dual	Not scalable	**Available**
QPFS/KTA	**Non-linear**	Non-convex[†]	Dual	Not scalable	**Available**
SpAM	Additive non-linear	**Convex**	**Primal**	Scalable	Not available
Proposed	**Non-linear**	**Convex**	**Primal**	**Highly scalable**	**Available**

[†]In practice, positive constants may be added to the diagonal elements of the Hessian matrix to guarantee the convexity, although the validity of selected features by this modification is not statistically clear.

where $D_{k,l} = \phi(\boldsymbol{u}_k)^\top \phi(\boldsymbol{u}_l) = D(\boldsymbol{u}_k, \boldsymbol{u}_l)$ and $\boldsymbol{D} = [\boldsymbol{d}_1, \ldots, \boldsymbol{d}_d]$. This formulation is called the *feature vector machine* (FVM). Note, since FVM uses $d \times d$ dimensional Hessian matrix $\boldsymbol{D}$, it is especially useful when the number of training samples n is much bigger than that of features d.

In the original FVM, *mutual information* (Cover and Thomas, 2006) was used as the kernel function $D(\boldsymbol{u}, \boldsymbol{u}')$. However, the matrix $\boldsymbol{D}$ obtained from mutual information is not necessarily positive definite (Seeger, 2002), and thus the objective function Eq.(2) can be non-convex. Furthermore, when the number of training samples is smaller than that of features (which is often the case in high-dimensional feature selection scenarios), the matrix $\boldsymbol{D}$ is singular. This can cause numerical instability. Another restriction of FVM is that, irrespective of regression or classification, output $\boldsymbol{y}$ should be transformed by the same non-linear function $\phi(\cdot)$ as feature vector $\boldsymbol{u}$. This highly limits the flexibility of capturing non-linear dependency. Finally, it is not statistically clear what kind of features are found by this FVM formulation.

1.5   Contribution of This Paper

To overcome the limitations of FVM, we propose an alternative feature-wise non-linear Lasso. More specifically, we propose to use particular forms of *universal reproducing kernels* (Steinwart, 2001) as feature and output transformations, and solve the optimization problem in the *primal* space.

An advantage of this new formulation is that the global optimal solution can be computed efficiently. Thus, it is scalable to high-dimensional feature selection problems. To the best of our knowledge, this is the first convex feature selection method that is able to deal with high-dimensional non-linearly related features. Furthermore, this new formulation has a clear statistical interpretation that non-redundant features with strong statistical dependence on output values are found via kernel-based independence measures such as the *Hilbert-Schmidt independence criterion* (HSIC) (Gretton *et al.*, 2005) and the criterion based on the *normalized cross-covariance operator* (NOCCO) (Fukumizu *et al.*, 2008). Thus, the proposed methods can be regarded as a *minimum redundancy maximum relevance* based feature selection method (Peng *et al.*, 2005). In addition, the proposed methods are simple to implement, which is a highly preferable property for practitioners.

We also discuss the relation between the proposed method and existing feature selection approaches such as *minimum redundancy maximum relevance* (mRMR) (Peng *et al.*, 2005), HSIC-based greedy feature selection (Song *et al.*, 2012), *quadratic programming feature selection* (QPFS) (Rodriguez-Lujan *et al.*, 2010), *kernel target alignment* (KTA) (Shawe-Taylor and Kandola, 2002; Cortes *et al.*, 2012), Hilbert-Schmidt Feature Selection (HSFS) (Masaeli *et al.*, 2010), and *sparse additive models* (SpAM) (Ravikumar *et al.*, 2009; Liu *et al.*, 2009; Raskutti *et al.*, 2012). See Table 1 for the summary of feature selection methods.

Through experiments on real-world feature selection problems, we show that the proposed methods compare favorably with existing feature selection methods.

2 Proposed Methods

In this section, we propose alternative implementations of the non-linear feature-wise Lasso.

2.1 HSIC Lasso

We propose a feature-wise non-linear Lasso of the following form, which we call the *HSIC Lasso*[2]:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2}\|\bar{\boldsymbol{L}} - \sum_{k=1}^d \alpha_k \bar{\boldsymbol{K}}^{(k)}\|_{\text{Frob}}^2 + \lambda\|\boldsymbol{\alpha}\|_1,$$
$$\text{s.t.} \quad \alpha_1, \dots, \alpha_d \geq 0, \tag{3}$$

where $\|\cdot\|_{\text{Frob}}$ is the Frobenius norm, $\bar{\boldsymbol{K}}^{(k)} = \boldsymbol{\Gamma}\boldsymbol{K}^{(k)}\boldsymbol{\Gamma}$ and $\bar{\boldsymbol{L}} = \boldsymbol{\Gamma}\boldsymbol{L}\boldsymbol{\Gamma}$ are centered Gram matrices, $K_{i,j}^{(k)} = K(x_{k,i}, x_{k,j})$ and $L_{i,j} = L(y_i, y_j)$ are Gram matrices, $K(x, x')$ and $L(y, y')$ are kernel functions, $\boldsymbol{\Gamma} = \boldsymbol{I}_n - \frac{1}{n}\boldsymbol{1}_n\boldsymbol{1}_n^\top$ is the centering matrix, $\boldsymbol{I}_n$ is the n-dimensional identity matrix, and $\boldsymbol{1}_n$ is the n-dimensional vector with all ones. Note that we employ non-negativity constraint for $\boldsymbol{\alpha}$ so that meaningful features are selected (see Section 2.2 for details). In addition, since we use the output Gram matrix $\boldsymbol{L}$ to select features in HSIC Lasso, we can naturally incorporate structured outputs via kernels. Moreover, we can perform feature selection even if the training data set consists of input $\boldsymbol{x}$ and its affinity information $\boldsymbol{L}$ such as link structures between inputs.

Differences from the original formulation (1) are that we allow the kernel functions K and L to be different and the non-negativity constraint is imposed. The first term in Eq.(3) means that we are regressing the output kernel matrix $\bar{\boldsymbol{L}}$ by a linear combination of feature-wise input kernel matrices $\{\bar{\boldsymbol{K}}^{(k)}\}_{k=1}^d$.

2.2 Interpretation of HSIC Lasso

Here, we show that HSIC Lasso can be regarded as a *minimum redundancy maximum relevancy* (mRMR) based feature selection method (Peng *et al.*, 2005), which is a popular feature selection strategy in machine learning and artificial intelligence communities.

The first term in Eq.(3) can be rewritten as

$$\frac{1}{2}\|\bar{\boldsymbol{L}} - \sum_{k=1}^d \alpha_k \bar{\boldsymbol{K}}^{(k)}\|_{\text{Frob}}^2 = \frac{1}{2}\text{HSIC}(\boldsymbol{y}, \boldsymbol{y}) - \sum_{k=1}^d \alpha_k \text{HSIC}(\boldsymbol{u}_k, \boldsymbol{y})$$
$$+ \frac{1}{2}\sum_{k,l=1}^d \alpha_k \alpha_l \text{HSIC}(\boldsymbol{u}_k, \boldsymbol{u}_l), \tag{4}$$

where $\text{HSIC}(\boldsymbol{u}_k, \boldsymbol{y}) = \text{tr}(\bar{\boldsymbol{K}}^{(k)}\bar{\boldsymbol{L}})$ is a kernel-based independence measure called the (empirical) *Hilbert-Schmidt independence criterion* (HSIC) (Gretton *et al.*, 2005) and $\text{tr}(\cdot)$ denotes the trace. $\text{HSIC}(\boldsymbol{y}, \boldsymbol{y})$ is a constant and can be ignored. HSIC always takes a non-negative value, and is zero if and only if two random variables are statistically independent when a *universal reproducing kernel* (Steinwart, 2001) such as the Gaussian kernel is used. Note that the empirical HSIC asymptotically converges to the true HSIC with $O(1/\sqrt{n})$ (see Theorem 3 in Gretton *et al.* (2005)). In addition, HSIC can be regarded as the centered version of the *kernel target alignment* (KTA) (Shawe-Taylor and Kandola, 2002).

If the k-th feature $\boldsymbol{u}_k$ has high dependence on output $\boldsymbol{y}$, $\text{HSIC}(\boldsymbol{u}_k, \boldsymbol{y})$ takes a large value and thus α_k should also take a large value so that Eq.(3) is minimized. On the other hand, if $\boldsymbol{u}_k$ is independent of $\boldsymbol{y}$, $\text{HSIC}(\boldsymbol{u}_k, \boldsymbol{y})$ is close to zero and thus such α_k tends to be eliminated by the ℓ_1-regularizer. This means that relevant features that have strong dependence on output $\boldsymbol{y}$ tend to be selected by HSIC Lasso.

Furthermore, if $\boldsymbol{u}_k$ and $\boldsymbol{u}_l$ are strongly dependent (i.e., redundant features), $\text{HSIC}(\boldsymbol{u}_k, \boldsymbol{u}_l)$ takes a large value and thus either of α_k and α_l tends to be zero. This means that redundant features tend to be eliminated by HSIC Lasso.

---

[2]A MATLAB® implementation of the proposed algorithm is available from `http://www.makotoyamada-ml.com/hsiclasso.html`.

Overall, HSIC Lasso tends to find non-redundant features with strong dependence on output $\boldsymbol{y}$, which is the idea of *minimum redundancy maximum relevancy* (mRMR) based feature selection methods (Peng *et al.*, 2005). This is a preferable property in feature selection.

Note that, it is possible to remove the non-negativity constraint in Eq.(3) and select features that have non-zero coefficients $\boldsymbol{\alpha}$. However, if we allow negative values in $\boldsymbol{\alpha}$, it is hard to interpret selected features. Indeed, interpretability is one of important properties in feature selection, and thus we include the non-negativity constraint for HSIC Lasso.

2.3 Kernel Selection

In theory, a universal kernel such as the Gaussian kernel or the Laplace kernel permits HSIC to detect dependence between two random variables (Gretton *et al.*, 2005). Moreover, it has been proposed to use the delta kernel for multi-class classification problems (Song *et al.*, 2012). Thus, in this paper, we employ the Gaussian kernel for inputs. For output kernels, we use the Gaussian kernel for regression cases and the delta kernel for classification problems.

For input x, we first normalize the input x to have unit standard deviation, and we use the Gaussian kernel:

$$K(x, x') = \exp\left(-\frac{(x - x')^2}{2\sigma_{\mathrm{x}}^2}\right),$$

where σ_{x} is the Gaussian kernel width.

In regression scenarios (i.e., $y \in \mathbb{R}$), we normalize an output y to have unit standard deviation, and we use the Gaussian kernel:

$$L(y, y') = \exp\left(-\frac{(y - y')^2}{2\sigma_{\mathrm{y}}^2}\right),$$

where σ_{y} is the Gaussian kernel width.

In classification scenarios (i.e., y is categorical), we use the delta kernel for y,

$$L(y, y') = \left\{ \begin{array}{ll} 1/n_y & \text{if } y = y', \\ 0 & \text{otherwise,} \end{array} \right.$$

where n_y is the number of samples in class y. Note that it is also possible to use the Gaussian kernel in classification scenarios, but it tends to perform poorly (see Figure 4).

2.4 Computational Properties of HSIC Lasso

An important computational property of HSIC Lasso is that the first term in Eq.(3) can be rewritten as

$$\frac{1}{2}\|\mathrm{vec}(\bar{\boldsymbol{L}}) - [\mathrm{vec}(\bar{\boldsymbol{K}}^{(1)}), \ldots, \mathrm{vec}(\bar{\boldsymbol{K}}^{(d)})]\boldsymbol{\alpha}\|_2^2,$$

where $\mathrm{vec}(\cdot)$ is the vectorization operator. This is the same form as plain Lasso with n^2 samples and d features.

If $d > n^2$ (i.e., high-dimensional feature selection from a small number of training samples), the Lasso optimization technique called the *dual augmented Lagrangian* (DAL)[3] was shown to be computationally highly efficient (Tomioka *et al.*, 2011). Because DAL can also incorporate the non-negativity constraint without losing its computational advantages, we can directly use DAL to solve our HSIC Lasso problem. In contrast, when $n^2 \geq d$, we may use the either cKTM or FVM (i.e., dual) formulation.

If the number of samples n is relatively large, the Gaussian kernel computation in HSIC Lasso is expensive. Therefore, the overall computation cost of HSIC Lasso is high. In addition, since naive implementation of HSIC Lasso requires $n^2 d$ memory space, it is not practical if both d and n are large (e.g., $d > 10000$ and $n > 1000$). Here, we propose a table lookup approach to reduce the computation time and memory size.

---

[3]http://www.ibis.t.u-tokyo.ac.jp/ryotat/dal/

More specifically, based on the fact that the Gaussian kernel values depend only on the difference of two input values, we normalize every feature x to have mean zero and unit standard deviation and discretize the difference of two input values into B values (we use $B = 4096$ in our implementation). Then, we prepare in advance a lookup table of B elements that contain Gaussian kernel values and refer to these values when we compute the Gaussian kernels. The centered Gram matrix $\bar{K}$ can be rewritten as

$$\bar{K} = (I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)K(I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)$$
$$= K - \mathbf{1}m^\top - m\mathbf{1}^\top + s\mathbf{1}\mathbf{1}^\top,$$

where $m = \frac{1}{n}K\mathbf{1} \in \mathbb{R}^n$ and $s = \frac{1}{n^2}\mathbf{1}^\top K\mathbf{1} \in \mathbb{R}$. Thus, to compute $\bar{K}$, we only need to store m and s. Namely, the required memory size for the table lookup based approach is $O(dn + B)$, which is much smaller than the memory size required for the naive implementation, $O(dn^2)$.

Another approach to deal with large sample size would be using stability selection which consists in running HSIC Lasso many times with subsampling and computing the number of times each feature is selected across the runs (Meinshausen and Bühlmann, 2010; Bach, 2008).

2.5 Variation: NOCCO Lasso

Instead of $\bar{K}^{(k)}$ and $\bar{L}$, let us use $\widetilde{K}^{(k)} = \bar{K}^{(k)}(\bar{K}^{(k)} + \epsilon n I_n)^{-1}$ and $\widetilde{L} = \bar{L}(\bar{L} + \epsilon n I_n)^{-1}$, where $\epsilon > 0$ is a regularization parameter. Then our optimization problem is expressed as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2}\sum_{k,l=1}^d \alpha_k\alpha_l \mathrm{D}^{\mathrm{NOCCO}}(\boldsymbol{u}_k, \boldsymbol{u}_l) - \sum_{k=1}^d \alpha_k \mathrm{D}^{\mathrm{NOCCO}}(\boldsymbol{u}_k, \boldsymbol{y}) + \lambda\|\boldsymbol{\alpha}\|_1,$$
$$\text{s.t.} \quad \alpha_1, \ldots, \alpha_d \geq 0,$$

where $\mathrm{D}^{\mathrm{NOCCO}}(\boldsymbol{u}_k, \boldsymbol{y}) = \mathrm{tr}(\widetilde{K}^{(k)}\widetilde{L})$ is the kernel-based dependence measure based on the (empirical) *normalized cross-covariance operator* (NOCCO) (Fukumizu *et al.*, 2008). We call this formulation the *NOCCO Lasso*.

Because $\mathrm{D}_{\mathrm{NOCCO}}$ was shown to be asymptotically independent of the choice of kernels, NOCCO Lasso is expected to be less sensitive to the kernel parameter choice than HSIC Lasso, although ϵ needs to be tuned in practice.

2.6 Other Types of Regularizers

The proposed method is amenable to most of the popular regularizers such as group-lasso and elastic-net regularizers (Meier *et al.*, 2008; Zou and Hastie, 2005), as well as to other feature selection problems. For example, the group-lasso regularizer can be easily incorporated into our framework as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2}\|\bar{L} - \sum_{k=1}^d \alpha_k \bar{K}^{(k)}\|_{\mathrm{Frob}}^2 + \lambda\sum_{g=1}^G \|\boldsymbol{\alpha}_g\|_2,$$
$$\text{s.t.} \quad \alpha_1, \ldots, \alpha_d \geq 0,$$

where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1^\top, \ldots, \boldsymbol{\alpha}_G^\top]^\top$, $\boldsymbol{\alpha}_g$ is the gth group of variables, and G is the number of groups. This group-lasso problem can also be efficiently solved by the DAL package with the non-negativity constraint.

2.7 Relation to Two-Stage Multiple Kernel Learning

The proposed method is closely related to the two-stage *multiple kernel learning* (MKL) method called centered kernel target alignment (cKTA) (Cortes *et al.*, 2012), which has been originally proposed for learning a kernel Gram matrix (not used for feature selection problems).

If we adopt cKTA for supervised feature selection problems, the optimization problem of cKTA can be written as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \quad \frac{1}{2} \sum_{k,l=1}^{d} \alpha_k \alpha_l \mathrm{HSIC}(\boldsymbol{u}_k, \boldsymbol{u}_l) - \sum_{k=1}^{d} \alpha_k \mathrm{HSIC}(\boldsymbol{u}_k, \boldsymbol{y})$$

$$\text{s.t.} \quad \alpha_1, \ldots, \alpha_d \geq 0.$$

Differences from the HSIC Lasso are that cKTA solve the optimization problem in the *dual* space and does not have the ℓ_1 regularization term.

An advantage of cKTA is that feature selection can be performed just by solving a non-negative least-squares problem. Moreover, since cKTA has $d \times d$ dimensional Hessian matrix ($\boldsymbol{D}_{kl} = \mathrm{HSIC}(\boldsymbol{u}_k, \boldsymbol{u}_l)$), cKTA is computationally efficient for feature selection problems with large n and small d. However, the Hessian matrix $\boldsymbol{D}$ is not necessarily positive definite (Seeger, 2002) and is singular in high-dimensional problems. More specifically, $\boldsymbol{D}$ can be written as

$$\boldsymbol{D} = [\mathrm{vec}(\bar{\boldsymbol{K}}^{(1)}), \ldots, \mathrm{vec}(\bar{\boldsymbol{K}}^{(d)})]^\top [\mathrm{vec}(\bar{\boldsymbol{K}}^{(1)}), \ldots, \mathrm{vec}(\bar{\boldsymbol{K}}^{(d)})],$$

and $\boldsymbol{D}$ is singular when $d > n^2$ (i.e., high-dimensional feature selection from a small number of training samples). Thus, solving the non-negative least-squares problem in high-dimensional feature selection problems can be cumbersome in practice.

For the above reason, the proposed feature selection method is more suited than cKTA for high-dimensional feature selection problems. In contrast, if we want to solve a small d and large n feature selection problem, cKTA is more suited than the HSIC Lasso.

3 Existing Methods

In this section, we review existing feature selection methods and discuss their relation to the proposed approach. See Table 1 for the summary of feature selection methods.

3.1 Minimum Redundancy Maximum Relevance (mRMR)

Minimum redundancy maximum relevance (mRMR) (Peng *et al.*, 2005) is a mutual information based feature selection criterion.

Let $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m]^\top \in \mathbb{R}^{m \times n}$ be a sub-matrix of $\boldsymbol{X} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]^\top \in \mathbb{R}^{d \times n}$, where m features are extracted from d features. Then mRMR for $\boldsymbol{V}$ is defined as follows:

$$\mathrm{mRMR}(\boldsymbol{V}) = \frac{1}{m} \sum_{k=1}^{m} \widehat{\mathrm{MI}}(\boldsymbol{v}_k, \boldsymbol{y}) - \frac{1}{m^2} \sum_{k,l=1}^{m} \widehat{\mathrm{MI}}(\boldsymbol{v}_k, \boldsymbol{v}_l), \tag{5}$$

where $\widehat{\mathrm{MI}}(\boldsymbol{v}, \boldsymbol{y})$ is an empirical approximator of mutual information given as

$$\widehat{\mathrm{MI}}(\boldsymbol{v}, \boldsymbol{y}) = \iint \widehat{p}_{\mathrm{v,y}}(\boldsymbol{v}, \boldsymbol{y}) \log \frac{\widehat{p}_{\mathrm{v,y}}(\boldsymbol{v}, \boldsymbol{y})}{\widehat{p}_{\mathrm{v}}(\boldsymbol{v}) \widehat{p}_{\mathrm{y}}(\boldsymbol{y})} \mathrm{d}\boldsymbol{v} \mathrm{d}\boldsymbol{y}.$$

$\widehat{p}_{\mathrm{v,y,}}(\boldsymbol{v}, \boldsymbol{y})$ denotes a Parzen window estimator of the joint density of $\boldsymbol{v}$ and $\boldsymbol{y}$, and $\widehat{p}_{\mathrm{v}}(\boldsymbol{v})$ and $\widehat{p}_{\mathrm{y}}(\boldsymbol{y})$ denotes Parzen window estimators of marginal densities of $\boldsymbol{v}$ and $\boldsymbol{y}$, respectively.

The first term in mRMR measures the dependency between chosen feature $\boldsymbol{v}_k$ and output $\boldsymbol{y}$, while the second term is a penalty for selecting redundant features. Thus, mRMR finds non-redundant features with strong dependence on outputs. A very fast implementation of mRMR available, and thus, it can deal with high-dimensional feature selection problems.

mRMR-based feature selection is performed by finding a sub-matrix $\boldsymbol{V}$ that maximizes Eq.(5). However, since there are 2^d possible feature subsets, the brute force approach is computationally intractable. Hence,

greedy search strategies such as forward selection/backward elimination are used in practice (Peng *et al.*, 2005). However, the greedy approaches tend to produce a locally optimal feature set.

Another potential weakness of mRMR is that mutual information is approximated by Parzen window estimation— Parzen window based mutual information estimation is unreliable when the number of training samples is small (Suzuki *et al.*, 2009).

3.2 Greedy Feature Selection with HSIC

The HSIC-based feature selection criterion (Song *et al.*, 2012) is defined as

$$\text{tr}(\bar{\boldsymbol{M}}\bar{\boldsymbol{L}}), \tag{6}$$

where $\bar{\boldsymbol{M}} = \boldsymbol{\Gamma}\boldsymbol{M}\boldsymbol{\Gamma}$ is a centered Gram matrix, $M_{i,j} = M(\boldsymbol{v}_i, \boldsymbol{v}_j)$ is a Gram matrix, $M(\boldsymbol{v}, \boldsymbol{v}')$ is a kernel function, and $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \boldsymbol{V} \in \mathbb{R}^{m \times n}$.

HSIC-based greedy feature selection is performed by finding a sub-matrix $\boldsymbol{V}$ that maximizes Eq.(6). An advantage of HSIC-based feature selection is its simplicity; it can be implemented very easily. However, since the maximization problem Eq.(6) is NP-hard, forward selection/backward elimination strategies are used for finding a locally optimal solution in practice (Song *et al.*, 2012).

3.3 Hilbert-Schmidt Feature Selection (HSFS)

Hilbert-Schmidt feature selection (HSFS) (Masaeli *et al.*, 2010) is defined as

$$\min_{\boldsymbol{W} \in \mathbb{R}^{d \times d}} \quad -\text{HSIC}(\boldsymbol{W}\boldsymbol{X}, \boldsymbol{y}) + \lambda \sum_{j=1}^{d} \|\boldsymbol{w}_j\|_{\infty},$$

where $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d]$ is a transformation matrix, $\lambda > 0$ is the regularization parameter, and $\|\cdot\|_{\infty}$ is the ℓ_{∞}-norm.

HSFS can be regarded as a continuously relaxed version of the HSIC-based feature selection (Song *et al.*, 2012). Thanks to this continuous formulation, the HSFS optimization problem can be solved by limited-memory BFGS (L-BFGS) (Nocedal and Wright, 2003). However, since HSFS is a non-convex method, restarting from many different initial points would be necessary to select good features, which is computationally expensive. Moreover, HSFS attempts to optimize a projection matrix which has d^2 parameters. That is, following the original HSFS implementation based on a Quasi-Newton method, the total computational complexity of HSFS is $O(d^4)$, which can be unacceptably large in high-dimensional feature selection problems. To reduce the computational cost, it may be able to approximately solve the HSFS optimization problem by reducing the size of the transformation matrix to $d \times q$ for $q \ll d$. However, this approximation leads to an additional tuning parameter that can not be chosen objectively.

3.4 Quadratic Programming Feature Selection (QPFS)

Quadratic programming feature selection (QPFS) (Rodriguez-Lujan *et al.*, 2010) tries to find features by solving a QP problem.

The QPFS optimization problem is defined as

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \quad \frac{(1-\gamma)}{2}\boldsymbol{\theta}^{\top}\boldsymbol{D}\boldsymbol{\theta} - \gamma\boldsymbol{\theta}^{\top}\boldsymbol{d},$$

$$\text{s.t.} \quad \sum_{k=1}^{d} \theta_k = 1, \quad \theta_1, \ldots, \theta_d \geq 0,$$

where $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_d]^{\top}$, $D_{k,l} = D(\boldsymbol{u}_k, \boldsymbol{u}_l)$, $\boldsymbol{d} = [D(\boldsymbol{u}_1, \boldsymbol{y}), \ldots, D(\boldsymbol{u}_d, \boldsymbol{y})]^{\top}$, $D(\cdot)$ is a dependency measure, and $\gamma \in [0, 1]$ is a tuning parameter. In QPFS, an empirical estimator of mutual information is used as a dependency

measure. Note that if we employ HSIC as a dependency measure in QPFS and remove the sum-to-one constraint, QPFS is equivalent to the centered KTA (cKTA) (Cortes *et al.*, 2012), which is a multiple kernel learning method and has been originally proposed for learning a kernel matrix (see Section 2.7 for details).

Similar to cKTA, an advantage of QPFS is that feature selection can be performed just by solving a QP problem. Moreover, since QPFS has $d \times d$ dimensional Hessian matrix, QPFS is computationally efficient for feature selection problems with large n and small d. However, the Hessian matrix $\boldsymbol{D}$ is not necessarily positive definite (Seeger, 2002) and is singular in high-dimensional problems.

3.5   Sparse Additive Models (SpAM)

The sparse additive models (SpAM) is useful for high-dimensional feature selection (Ravikumar *et al.*, 2009; Liu *et al.*, 2009; Raskutti *et al.*, 2012; Suzuki and Sugiyama, 2013).

The SpAM optimization problem can be expressed as

$$\min_{\boldsymbol{\beta}_1,\ldots,\boldsymbol{\beta}_d \in \mathbb{R}^n} \quad \|\boldsymbol{y} - \sum_{k=1}^{d} \boldsymbol{K}^{(k)} \boldsymbol{\beta}_k\|_2^2 + \lambda \sum_{k=1}^{d} \sqrt{\frac{1}{n} \|\boldsymbol{K}^{(k)} \boldsymbol{\beta}_k\|_2^2},$$

where $\boldsymbol{\beta}_k = [\beta_{k,1}, \ldots, \beta_{k,n}]^\top, k = 1, \ldots, d$ are regression coefficient vectors, $\beta_{k,j}$ is a coefficient for $[K(x_{k,1}, x_{k,j}), \ldots, K(x_{k,n}, x_{k,j})]^\top$, and $\lambda > 0$ is a regularization parameter. This problem can be efficiently solved by the *back-fitting* algorithm (Ravikumar *et al.*, 2009). Note that SpAM is closely related to the *hierarchical multiple kernel learning* (Bach, 2009), which employs a sparse additive model with an alternative sparsity-inducing regularizer.

An advantage of SpAM is that it is a convex method and can be efficiently optimized by the backfitting algorithm. Moreover, statistical properties of the SpAM estimator are well studied (Ravikumar *et al.*, 2009). A potential weakness of SpAM is that it can only deal with additive models. That is, if data follows a non-additive model, SpAM may not work well (see Figure 1(b)). Another weakness of SpAM is that it needs to optimize nd variables, while the proposed methods contain only d variables. Thus, SpAM optimization tends to be computationally more expensive than the proposed methods. Finally, an output y should be a real number in SpAM, meaning that SpAM cannot deal with structured outputs such as multi-label and graph data.

4   Experiments

In this section, we experimentally investigate the performance of the proposed and existing feature selection methods using synthetic and real-world datasets.

4.1   Setup

We compare the performance of the proposed methods with mRMR (Peng *et al.*, 2005), QPFS (Rodriguez-Lujan *et al.*, 2010), cKTA (Cortes *et al.*, 2012), forward selection with HSIC (FHSIC), FVM (Li *et al.*, 2006), and SpAM[4] (Ravikumar *et al.*, 2009). Note that, since it has been reported that the performance of FHSIC is comparable to HSFS and HSFS is computationally expensive for high-dimensional data, we decided to only compare the proposed method to FHSIC. For FVM, QPFS, and mRMR, the C++ implementation of a mutual information estimator[5] is used. Then, a QP solver SeDuMi[6] is used to solve the FVM and QPFS optimization problems. We observed that the matrices $\boldsymbol{D}$ in FVM, QPFS, and cKTA tend not to be positive definite. In our experiments, we added a small constant to the diagonal elements of $\boldsymbol{D}$ so that the objective function becomes convex. For all experiments, we set $\lambda = 1$ in FVM, $\gamma = 0.5$ in QPFS and cKTA, and $\epsilon = 10^{-3}$ in NOCCO Lasso. For proposed methods, we experimentally use $\sigma_{\mathrm{x}} = 1$.

---

[4]We thank the authors of (Ravikumar *et al.*, 2009) for providing us the code used in their paper.

[5]http://penglab.janelia.org/proj/mRMR/

[6]http://sedumi.ie.lehigh.edu/

Table 2: Summary of real-world datasets.

Type	Dataset	# features (d)	# samples (n)	# Classes
Image	AR10P	2400	130	10
	PIE10P	2400	210	10
	PIX10P	10000	100	10
	ORL10P	10000	100	10
Microarray	TOX	5748	171	4
	CLL-SUB	11340	111	3

4.2 Synthetic Datasets

First, we illustrate the behavior of the proposed HSIC Lasso and NOCCO Lasso using the following two synthetic datasets:

(a) Data1 (Additive model):

$$Y = -2\sin(2X_1) + X_2^2 + X_3 + \exp(-X_4) + E,$$

where $(X_1, \ldots, X_{256})^\top \sim N(\mathbf{0}_{256}, \mathbf{I}_{256})$ and $E \sim N(0, 1)$. Here, $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the multi-variate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

(b) Data2 (Non-additive model):

$$Y = X_1 \exp(2X_2) + X_3^2 + E,$$

where $(X_1, \ldots, X_{1000})^\top \sim N(\mathbf{0}_{1000}, \mathbf{I}_{1000})$ and $E \sim N(0, 1)$.

Figure 1 shows the feature selection accuracy of each method over 30 runs as functions of the number of samples, where the accuracy is measured by the fraction of correctly selected features under the assumption that the number of true features is known. As the figure clearly shows, the proposed HSIC Lasso and NOCCO Lasso methods select good features in both additive and non-additive model cases. SpAM also works very well for Data1, but it performs poorly for Data2 because the additivity assumption is violated in Data2. QPFS and FVM behave similarly, but they tend to be outperformed by the proposed methods.

Next, we compare the computation time of each method. Here, we change the number of features in Data2 to $d = 100, 200, \ldots, 1000$, while we fix the number of samples to $n = 100$. Figure 1-(c) shows the average computation time for each method over 30 runs. As can be observed, the computation time of HSIC Lasso and NOCCO Lasso increases mildly with respect to the number of features compared to that of SpAM, FVM, and QPFS. Figure 1-(d) shows the average computation time of HSIC Lasso for 30 runs. In this experiment, we use the table lookup trick for HSIC Lasso to deal with a relatively large number of samples. This shows that the lookup trick allows us to handle relatively large datasets.

4.3 Real-World Datasets

Next, we compare the performance of feature selection methods using real-world datasets.

4.3.1 Multi-Class Classification

We use four image datasets and two microarray datasets[7]. Detailed information of the datasets is summarized in Table 2.

In this experiment, we use 80% of samples for training and the rest for testing. We repeat the experiments 100 times by randomly shuffling training and test samples, and evaluate the performance of feature selection methods by the average classification accuracy. We use multi-class ℓ_2-regularized kernel logistic regression (KLR) (Hastie *et al.*, 2001; Yamada *et al.*, 2010) with the Gaussian kernel for evaluating the classification accuracy when

---

[7]http://featureselection.asu.edu/datasets.php

(a) Data1 (Additive model)

(b) Data2 (Non-additive model)

(c) Computation time for Data2

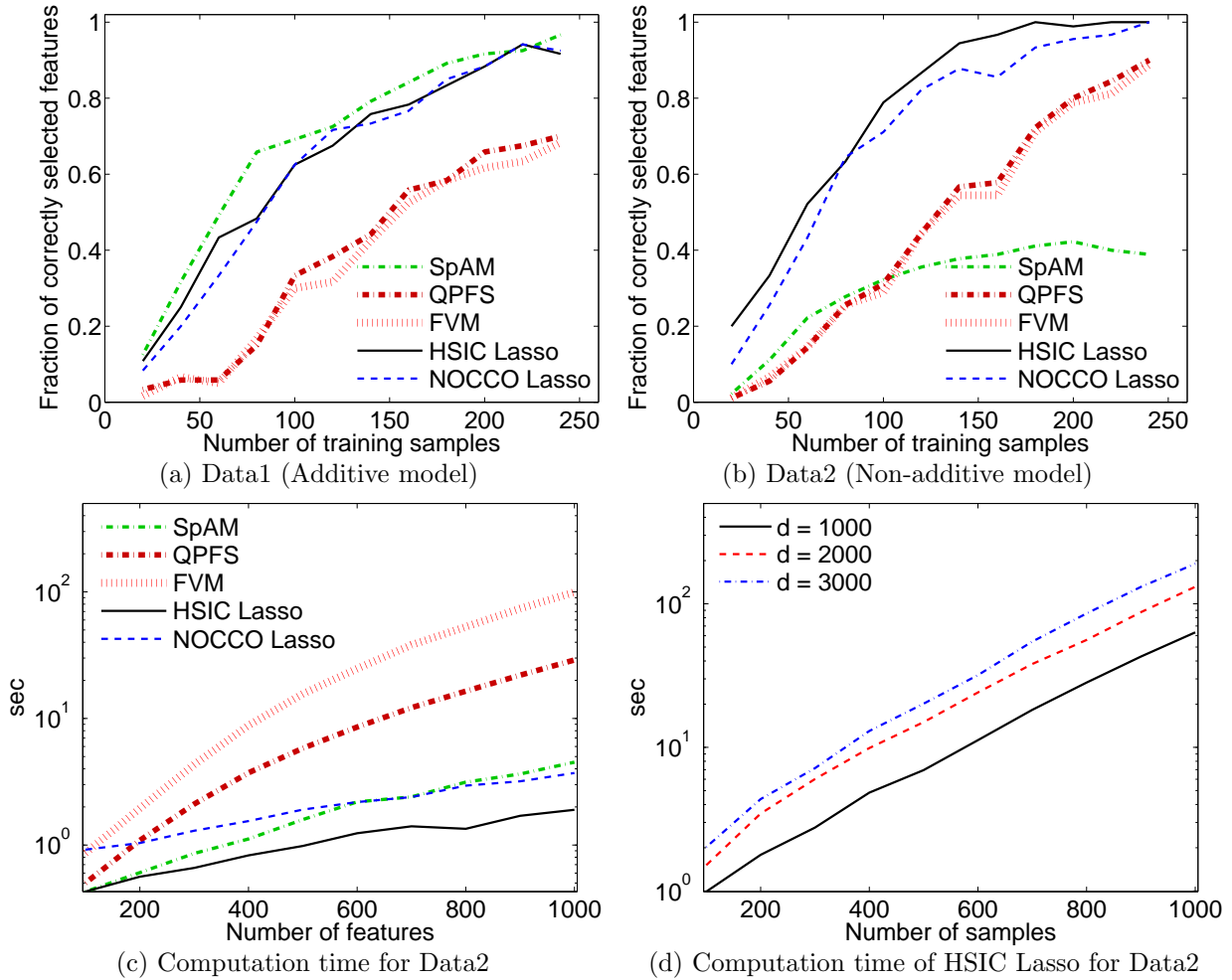(d) Computation time of HSIC Lasso for Data2

Figure 1: (a),(b): Feature selection results for artificial datasets over 30 runs. The horizontal axis denotes the number of training samples, and the vertical axis denotes the fraction of correctly selected features. In HSIC Lasso and NOCCO Lasso, the regularization parameter λ is set so that the number of non-zero coefficients is in $\{d^*, d^* + 1, \ldots, d^* + 10\}$ where d^* is the number of true features. Then, we use top d^* features by ranking regression coefficients. In QPFS, FVM, and SpAM, we use top d^* features by ranking coefficients. (c): Comparison of computation time for Data2. The horizontal axis denotes the number of entire features d, and the vertical axis denotes the computation time in log-scale. (d): Comparison of computation time for Data2 for HSIC Lasso. The horizontal axis denotes the number of training samples n, and the vertical axis denotes the computation time in log-scale.

the top $m = 10, 20, \ldots, 50$ features selected by each method are used. In this paper, we first choose 50 features and then use top $m = 10, 20, \ldots, 50$ features having the largest absolute regression coefficients. In KLR, all tuning parameters such as the Gaussian width and the regularization parameter are chosen based on 3-fold cross-validation.
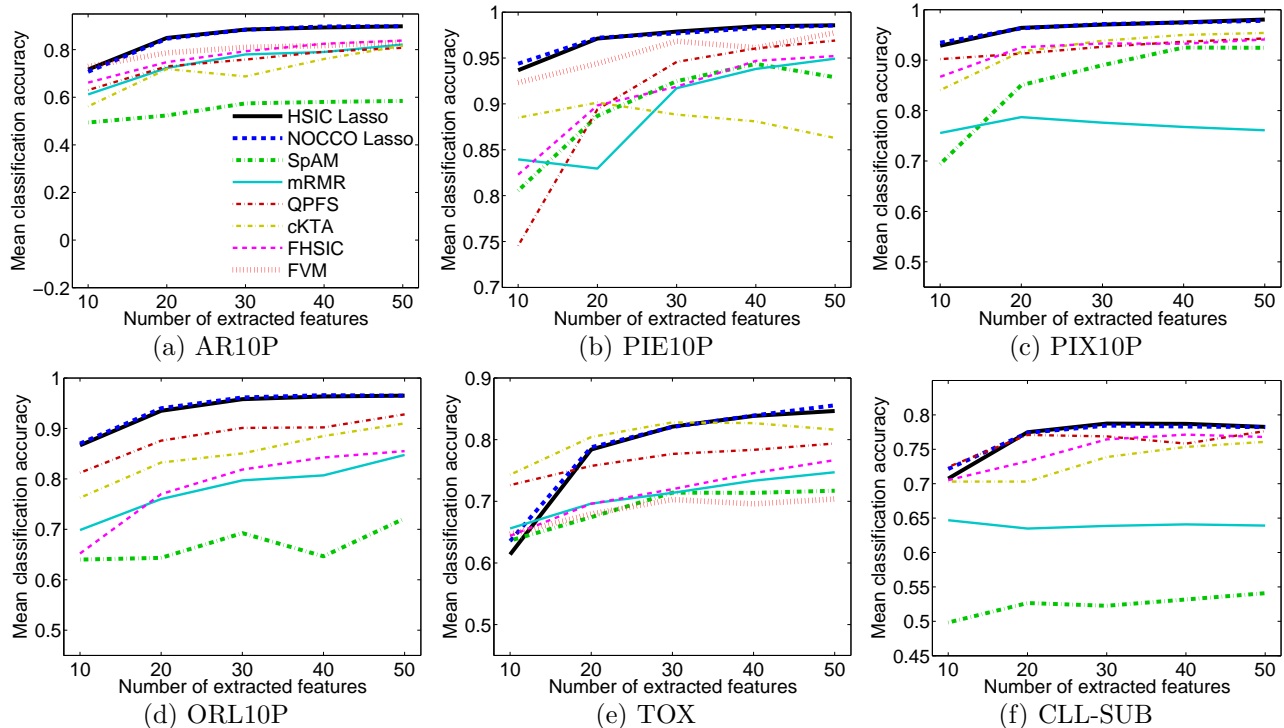
Figure 2: Mean classification accuracy for real-world data. The horizontal axis denotes the number of selected features, and the vertical axis denotes the mean classification accuracy.

We also investigate the *redundancy rate* (RED) (Zhao *et al.*, 2010)[8],

$$\text{RED} = \frac{1}{m(m-1)} \sum_{\boldsymbol{u}_k, \boldsymbol{u}_j, k > l} |\rho_{k,l}|,$$

where $\rho_{k,l}$ is the correlation coefficient between the k-th and l-th features. A large RED score indicates that selected features are more strongly correlated to each other, that is, many redundant features are selected. Thus, as a feature selection method, a small redundancy rate is preferable.

Results: Figure 2 shows the mean classification accuracy over 100 runs as functions of the number of selected features. Table 3 shows the average classification accuracy rates for the top $m = 50$ features selected by each method. In this experiment, since the computation cost of FVM was too high for datasets with a large number of features, we only included the FVM results for the datasets with a small number of features (i.e., AR10P, PIE10P, and TOX). The graphs in Figure 2 clearly show that HSIC Lasso and NOCCO Lasso compare favorably with existing methods for the image datasets (i.e., AR10P, PIE10P, PIX10P, and ORL10P) in terms of the classification accuracy, and they are comparable to existing methods for the microarray datasets (i.e., TOX and CLL-SUB).

Table 4 shows the RED values for the top $m = 50$ features selected by each method. As can be observed, HSIC Lasso and NOCCO Lasso tend to have smaller RED values, and thus they select less redundant features.

Role of the Gaussian Width and the Output Kernel: In the proposed methods, the Gaussian width and the output kernel $L(y, y')$ must be chosen manually. We carried out a set of experiments to show the sensitivity

---

[8]The original redundancy rate was defined with a plain correlation coefficient (Pearson, 1920), not the absolute correlation coefficient (Zhao *et al.*, 2010). However, this is not appropriate as an error metric because negative correlation decreases RED. For this reason, we decided to use the absolute correlation coefficient.

Table 3: Mean classification accuracy (with standard deviations in brackets) for real-world data.

Dataset	HSIC Lasso	NOCCO Lasso	SpAM	FVM	mRMR	QPFS	cKTA	FHSIC
AR10P	**.848** (.111)	.846 (.111)	.551 (.109)	.795 (.121)	.745 (.136)	.743 (.137)	.709 (.207)	.773 (.122)
PIE10P	.971 (.032)	**.972** (.031)	.898 (.109)	.955 (.062)	.895 (.118)	.952 (.067)	.884 (.218)	.908 (.091)
PIX10P	**.964** (.043)	**.964** (.042)	.857 (.146)	— (—)	.769 (.124)	.924 (.067)	.920 (.090)	.920 (.100)
ORL10P	.938 (.068)	**.941** (.066)	.669 (.120)	— (—)	.782 (.138)	.884 (.096)	.848 (.140)	.788 (.132)
TOX	.781 (.119)	.788 (.113)	.691 (.087)	.686 (.085)	.709 (.084)	.769 (.077)	**.804** (.110)	.715 (.087)
CLL-SUB	.768 (.087)	**.769** (.084)	.524 (.112)	— (—)	.640 (.098)	.760 (.116)	.709 (.104)	.732 (.141)

Table 4: Mean redundancy rate (with standard deviations in brackets) for real-world data.

| Dataset	HSIC Lasso	NOCCO Lasso	SpAM	FVM	mRMR	QPFS	cKTA	FHSIC
AR10P	.196 (.028)	**.195** (.028)	.255 (.036)	.260 (.039)	.268 (.038)	.217 (.050)	.235 (.034)	.350 (.091)
PIE10P	**.135** (.014)	.139 (.017)	.250 (.042)	.193 (.029)	.225 (.036)	.183 (.026)	.155 (.021)	.285 (.059)
PIX10P	**.177** (.023)	**.174** (.023)	.388 (.105)	— (—)	.200 (.066)	.286 (.057)	.198 (.036)	.348 (.064)
ORL10P	**.192** (.026)	**.191** (.025)	.300 (.047)	— (—)	.294 (.095)	.204 (.032)	.191 (.034)	.225 (.045)
TOX	.382 (.027)	.381 (.027)	.391 (.028)	.422 (.031)	.386 (.032)	.384 (.028)	**.371** (.040)	.396 (.036)
CLL-SUB	.344 (.034)	.345 (.034)	.403 (.058)	— (—)	.328 (.039)	.322 (.033)	**.281** (.050)	.352 (.061)
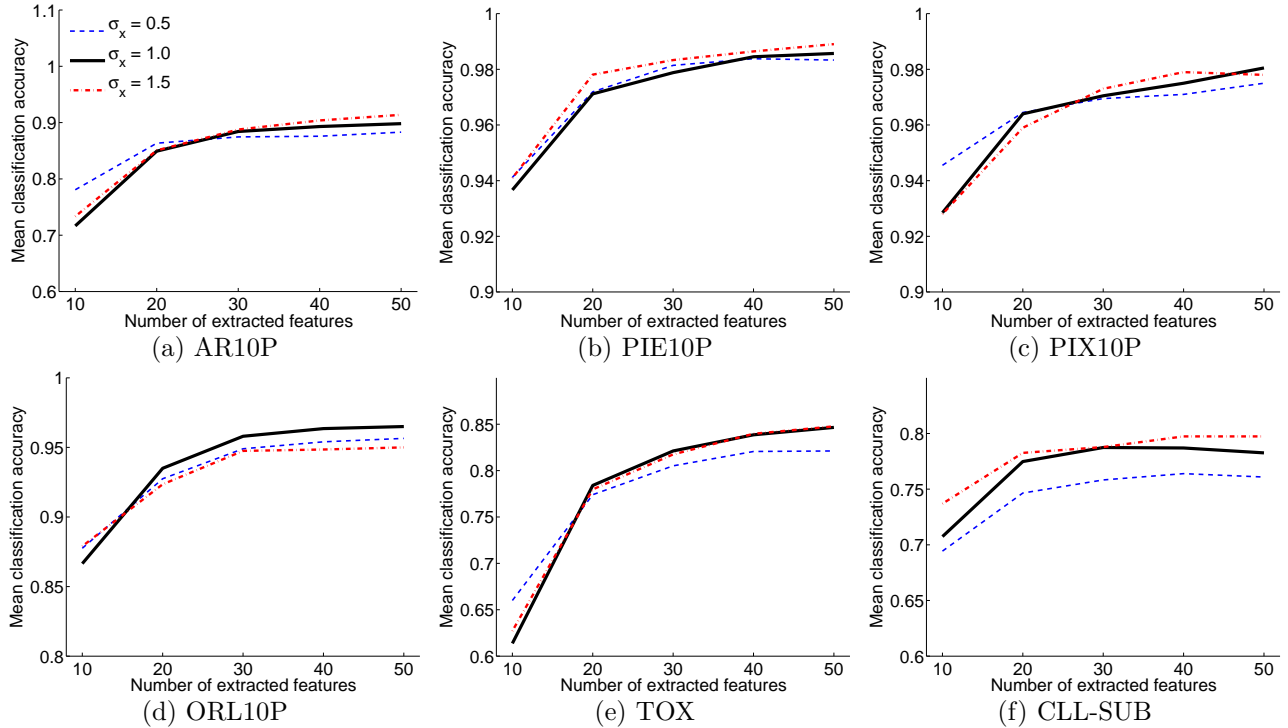
Figure 3: Mean classification accuracy of HSIC Lasso with different Gaussian widths. The horizontal axis denotes the number of selected features, and the vertical axis denotes the mean classification accuracy.

of choosing the Gaussian width and the output kernel in Figures 3 and 4. Note that, since the performance of HSIC Lasso and NOCCO Lasso are comparable, we here only evaluate HSIC Lasso. As can be seen in Figure 3,
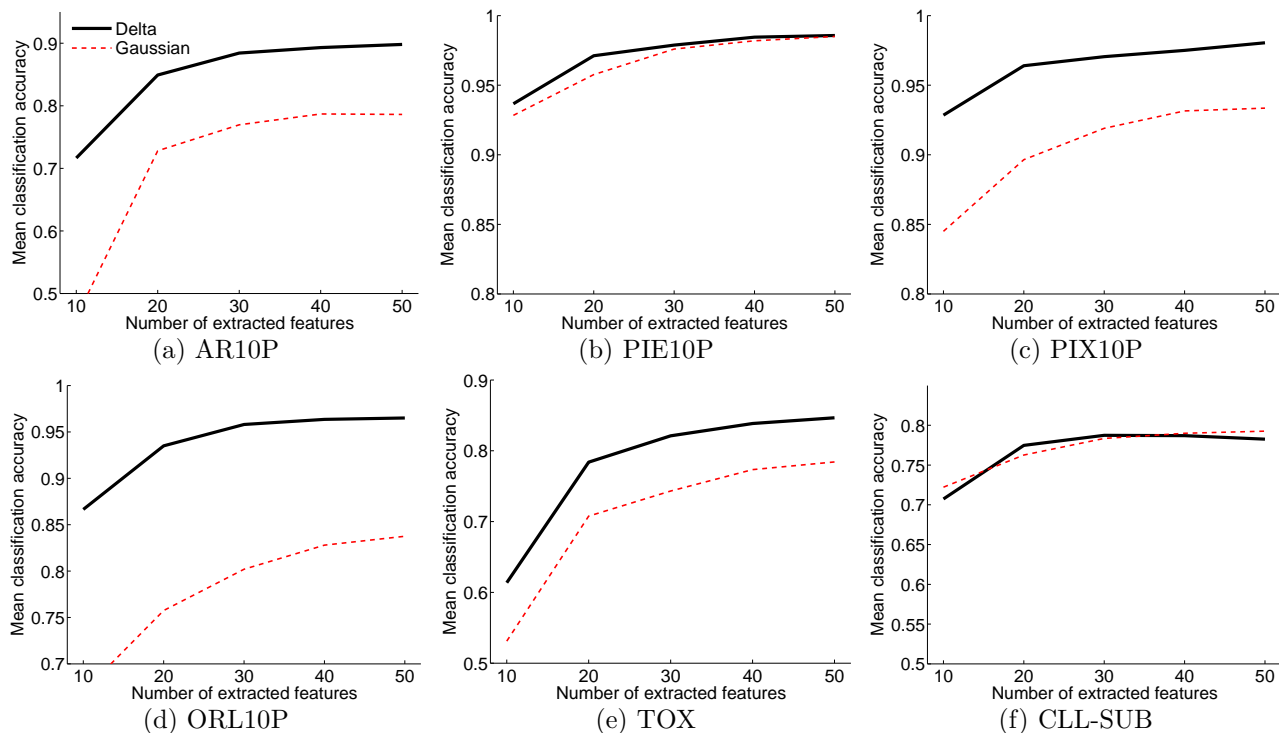
Figure 4: Mean classification accuracy of HSIC Lasso with different output kernels. Here, we use the delta kernel and the Gaussian kernel. The horizontal axis denotes the number of selected features, and the vertical axis denotes the mean classification accuracy.

the proposed method is not so sensitive to the Gaussian width. From the output kernel comparison in Figure 4, we found that HSIC Lasso with delta kernel clearly outperforms that with Gaussian kernel. Thus using different input and output kernels is important for feature selection in classification scenarios.

4.3.2   High-Dimensional Regression

We also evaluate our proposed method with the Affymetric GeneChip Rat Genome 230 2.0 Array data set (Scheetz *et al.*, 2006). The data set consists of 120 rat subjects with 31098 genes which were measured from eye tissue. Similar to Huang *et al.* (2010), we focus on finding genes that are related to the TRIM32 gene, which was recently found to cause the Bardet-Biedl syndrome. Note that, since TRIM32 takes real values, this is a regression problem.

In this experiment, we use 80% of samples for training and the rest for testing. We repeat the experiments 100 times by randomly shuffling training and test samples, and evaluate the performance of feature selection methods by the mean squared error. In addition, we use the correlation coefficient between the predicted and the true TRIM32 values, which is a popular performance metric in biology community. We use kernel regression (KR) (Schölkopf and Smola, 2002) with the Gaussian kernel for evaluating the mean squared error and the mean correlation when the top $m = 10, 20, \ldots, 50$ features selected by each method are used. We first choose 50 features and then use top $m = 10, 20, \ldots, 50$ features having the largest absolute regression coefficients. In KR, all tuning parameters such as the Gaussian width and the regularization parameter are chosen based on 3-fold cross-validation.

Results: Figure 5 shows the mean squared error and the mean correlation coefficient over 100 runs as functions of the number of selected features. As can be observed, the proposed HSIC Lasso and NOCCO Lasso compare favorably with existing methods.
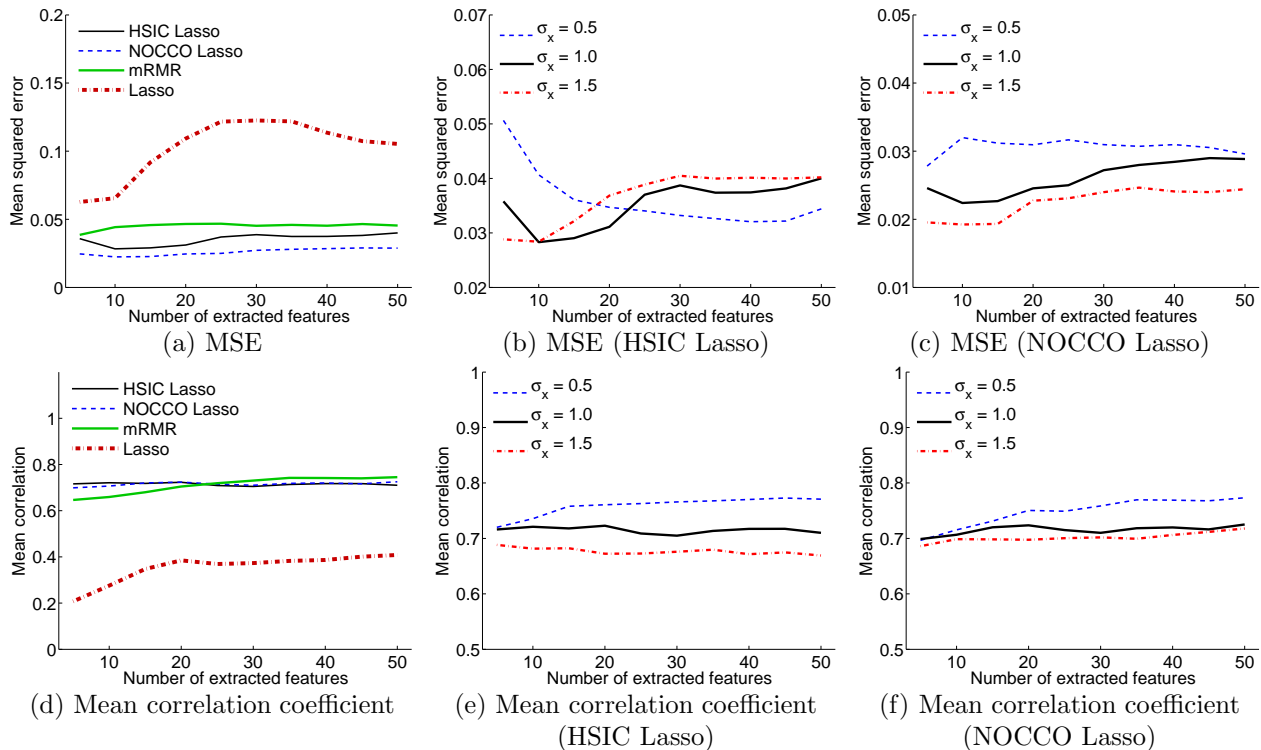
Figure 5: (a): Mean squared error for the biology data. The horizontal axis denotes the number of selected features, and the vertical axis denotes the mean squared error. Here, we use the kernel parameter $\sigma_x = 1.0$ for both HSIC Lasso and NOCCO Lasso. (b),(c): Mean squared error of HSIC Lasso and NOCCO Lasso with respect to different kernel parameters. (d): Mean correlation for the biology data. The horizontal axis denotes the number of selected features, and the vertical axis denotes the mean correlation. (e),(f): Mean correlation coefficient of HSIC Lasso and NOCCO Lasso with respect to different kernel parameters. The average redundancy rate of HSIC Lasso, NOCCO Lasso, mRMR, and Lasso are 0.44, 0.45, 0.42, and 0.43.

5   Conclusion

In this paper, we proposed novel non-linear feature selection methods called HSIC Lasso and NOCCO Lasso. In the proposed methods, global optimal solutions can be obtained by solving a Lasso optimization problem with a non-negativity constraint, which can be efficiently performed by the dual augmented Lagrangian algorithm (Tomioka *et al.*, 2011). Furthermore, the proposed methods have clear statistical interpretation that non-redundant features with strong statistical dependence on output values can be found via kernel-based independence measures (Gretton *et al.*, 2005; Fukumizu *et al.*, 2008). We applied the proposed methods to real-world image and biological feature selection tasks, and experimentally showed that they are promising.

The usefulness of the proposed method will be further investigated on more real-world applications such as computer vision, bioinformatics, and speech and signal processing in the future work. Moreover, extending the proposed model to multi-task learning and prediction and investigating theoretical properties of the proposed formulation are important issues to be investigated.

Acknowledgments

References

Bach, F. (2008). Bolasso: Model consistent lasso estimation through the bootstrap. In *International Conference on Machine learning (ICML)*, pages 33–40. ACM.

Bach, F. (2009). Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 105–112.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge.

Combettes, P. L. and Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, **4**(4), 1168–1200.

Cortes, C., Mohri, M., and Rostamizadeh, A. (2012). Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research (JMLR)*, **13**, 795–828.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2nd edition.

Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, **LVII**(11), 1413–1457.

Ding, C. and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics an Computaitonal Biology*, **3**(2), 185–205.

Forman, G. (2008). Bns feature scaling: An improved representation over tf-idf for svm text classification. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 263–270.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. (2008). Kernel measures of conditional dependence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, pages 489–496, Cambridge, MA. MIT Press.

Gretton, A., Bousquet, O., Smola, A., and Schölkopf, B. (2005). Measuring statistical dependence with Hilbert-Schmidt norms. In *Algorithmic Learning Theory (ALT)*, pages 63–77. Springer.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York.

Huang, J., Horowitz, J. L., and Wei, F. (2010). Variable selection in nonparametric additive models. *Annals of statistics*, **38**(4), 2282.

Kim, S.-J., Koh, K., Lustig, M., Boyd, S., and Gorinvesky, D. (2007). An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, **1**(4), 606–617.

Li, F., Yang, Y., and Xing, E. (2006). From lasso regression to feature vector machine. In *Advances in Neural Information Processing Systems (NIPS)*, pages 779–786.

Liu, H., Lafferty, J., and Wasserman, L. (2009). Nonparametric regression and classification with joint sparsity constraints. In *Advances in Neural Information Processing Systems (NIPS)*, pages 969–976.

Masaeli, M., Fung, G., and Dy, J. G. (2010). From transformation-based dimensionality reduction to feature selection. In *International conference on Machine learning (ICML)*, pages 751–758.

Meier, L., Van De Geer, S., and Bühlmann, P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **70**(1), 53–71.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **72**(4), 417–473.

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

Pearson, K. (1920). Notes on the history of correlation. *Biometrika*, **13**(1), 25–45.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**, 1226–1237.

Raskutti, G., Wainwright, M., and Yu, B. (2012). Minimax-optimal rates for sparse additive models over kernel classes via convex programming. *Journal of Machine Learning Research (JMLR)*, **13**, 389–427.

Ravikumar, P., Lafferty, J., Liu, H., and Wasserman, L. (2009). Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **71**(5), 1009–1030.

Rodriguez-Lujan, I., Huerta, R., Elkan, C., and Cruz, C. S. (2010). Quadratic programming feature selection. *Journal of Machine Learning Research (JMLR)*, **11**, 1491–1516.

Roth, V. (2004). The generalized Lasso. *IEEE Transactions on Neural Networks*, **15**(1), 16–28.

Scheetz, T. E., Kim, K.-Y. A., Swiderski, R. E., Philp, A. R., Braun, T. A., Knudtson, K. L., Dorrance, A. M., DiBona, G. F., Huang, J., Casavant, T. L., *et al.* (2006). Regulation of gene expression in the mammalian eye and its relevance to eye disease. *Proceedings of the National Academy of Sciences*, **103**(39), 14429–14434.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. MIT Press, Cambridge, MA.

Seeger, M. (2002). Covariance kernels from Bayesian generative models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 905–912.

Shawe-Taylor, N. and Kandola, A. (2002). On kernel target alignment. In *Advances in Neural Information Processing Systems (NIPS)*, pages 367–373.

Shenoy, P., Miller, K. J., Crawford, B., and Rao, R. N. (2008). Online electromyographic control of a robotic prosthesis. *IEEE Transactions on Biomedical Engineering*, **55**(3), 1128–1135.

Song, L., Smola, A., Gretton, A., Bedo, J., and Borgwardt, K. (2012). Feature selection via dependence maximization. *The Journal of Machine Learning Research*, **13**, 1393–1434.

Steinwart, I. (2001). On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research (JMLR)*, **2**, 67–93.

Suzuki, T. and Sugiyama, M. (2013). Fast learning rate of multiple kernel learning: Trade-off between sparsity and smoothness. *The Annals of Statistics*. to appear.

Suzuki, T., Sugiyama, M., Kanamori, T., and Sese, J. (2009). Mutual information estimation reveals global associations between stimuli and biological processes. *BMC Bioinformatics*, **10**(S52).

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, **58**(1), 267–288.

Tomioka, R., Suzuki, T., and Sugiyama, M. (2011). Super-linear convergence of dual augmented Lagrangian algorithm for sparsity regularized estimation. *Journal of Machine Learning Research (JMLR)*, **12**, 1537–1586.

Wright, S. J., Nowak, R. D., and Figueiredo, M. A. T. (2009). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, **57**(7).

Xing, E. P., Jordan, M. I., Karp, R. M., *et al.* (2001). Feature selection for high-dimensional genomic microarray data. In *International Conference on Machine Learning (ICML)*, pages 601–608.

Yamada, M., Sugiyama, M., and Matsui, T. (2010). Semi-supervised speaker identification under covariate shift. *Signal Processing*, **90**(8), 2353–2361.

Yin, W., Osher, S., Goldfarb, D., and Darbon, J. (2008). Bregman iterative algorithms for L1-minimization with applications to compressed sensing. *SIAM Journal of Imaging Sciences*, **1**(1), 143–168.

Zhao, Z., Wang, L., and Li, H. (2010). Efficient spectral feature selection with minimum redundancy. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 673–678.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**(2), 301–320.