Machine Learning using Hyperkernels

Cheng Soon Ong CHENG.ONG@ANU.EDU.AU
Alexander J. Smola ALEX.SMOLA@ANU.EDU.AU
Machine Learning Group, RSISE, Australian National University, Canberra, ACT 0200, Australia

Abstract

We expand on the problem of learning a kernel via a RKHS on the space of kernels itself. The resulting optimization problem is shown to have a semidefinite programming solution. We demonstrate that it is possible to learn the kernel for various formulations of machine learning problems. Specifically, we provide mathematical programming formulations and experimental results for the C-SVM, ν-SVM and Lagrangian SVM for classification on UCI data, and novelty detection.

1. Introduction

Kernel Methods have been highly successful in solving various problems in machine learning. The algorithms work by mapping the inputs into a feature space, and finding a suitable hypothesis in this new space. In the case of the Support Vector Machine, this solution is the hyperplane which maximizes the margin in the feature space. The feature mapping in question is defined by a kernel function, which allows us to compute dot products in feature space using only the objects in the input space.

Recently, there have been many developments regarding learning the kernel function itself (Bousquet & Herrmann, 2003; Crammer et al., 2003; Cristianini et al., 2002; Lanckriet et al., 2002; Momma & Bennett, 2002; Ong et al., 2003). In this paper, we extend the hyperkernel framework introduced in Ong et al. (2003), which we review in Section 2. In particular, the contributions of this paper are:

- a general class of hyperkernels allowing automatic relevance determination (Section 3),
- explicit mathematical programming formulations of the optimization problems (Section 4),
- implementation details of various SVMs and Alignment (Section 5)
- and further experiments on binary classification and novelty detection (Section 6).

At the heart of the strategy is the idea that we learn the kernel by performing the kernel trick on the space of kernels, hence the notion of a *hyper*kernel.

2. Hyper-RKHS

As motivation for the need for such a formulation, consider Figure 1, which shows the separating hyperplane and the margin for the same dataset. Figure 1(a) shows the training data and the classification function for a support vector machine using a Gaussian RBF kernel. The data has been sampled from two Gaussian distributions with standard deviation 1 in one dimension and 1000 in the other. This difference in scale creates problems for the Gaussian RBF kernel, since it is unable to find a kernel width suitable for both dimensions. Hence, the classification function is dominated by the dimension with large variance. The traditional way to handle such data is to normalize each dimension independently.

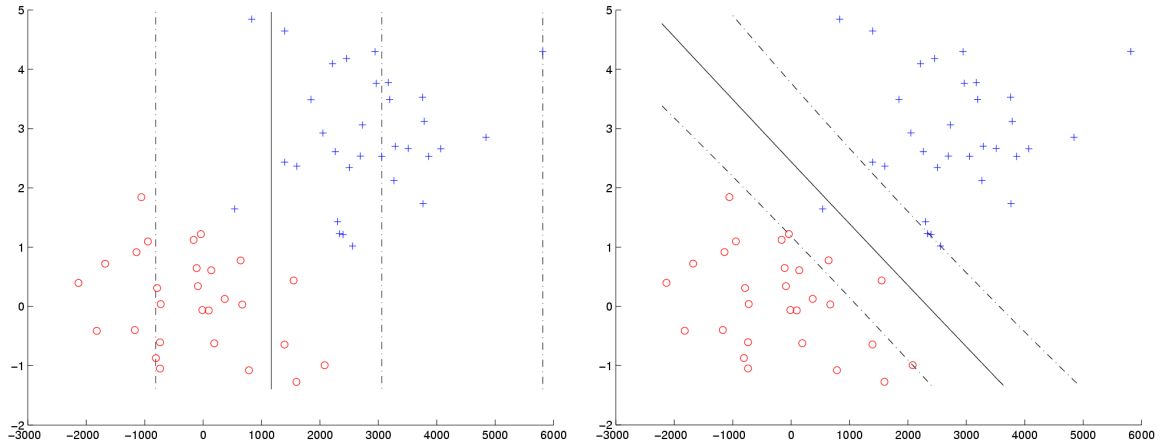
Instead of normalizing the input data, we make the kernel adaptive to allow independent scales for each dimension. This allows the kernel to handle unnormalized data. However, the resulting kernel would be difficult to tune by cross validation as there are numerous free variables (one for each dimension). We 'learn' this kernel by defining a quantity analogous to the risk functional, called the quality functional, which measures the 'badness' of the kernel function. The classification function for the above mentioned data is shown in Figure 1(b). Observe that it captures the scale of each dimension independently.

We review the definitions from (Ong et al., 2003). Given a set of input data, X, and their associated labels[1] , Y, and a class of kernels $\mathcal{K}$, we would like to select the best kernel $k \in \mathcal{K}$ for the problem.

Definition 1 (Empirical Quality Functional) *Given a kernel k, and data X, Y, we define $Q_{\text{emp}}(k, X, Y)$ to be an* empirical quality functional *if it depends on k only via $k(x_i, x_j)$ where $x_i, x_j \in X$ for $1 \leqslant i, j \leqslant m$.*

[1] only for supervised learning

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.



(a) Standard Gaussian RBF kernel (b) RBF-Hyperkernel with adaptive widths

Figure 1. For data with highly non-isotropic variance, choosing one scale for all dimensions leads to unsatisfactory results. Plot of synthetic data, showing the separating hyperplane and the margins given for a uniformly chosen length scale (left) and an automatic width selection (right).

Q_{emp} tells us how well matched k is to a specific dataset X, Y. Examples of such functionals include the Kernel Target Alignment, the regularized risk and the negative log-posterior. However, if provided with a sufficiently rich class of kernels $\mathcal{K}$ it is in general possible to find a kernel that attains the minimum of any such Q_{emp} regardless of the data (see (Ong et al., 2003) for examples). Therefore, we would like to somehow control the complexity of the kernel function. We achieve this by using the kernel trick again on the space of kernels.

Definition 2 (Hyper RKHS) *Let $\mathcal{X}$ be a nonempty set and denote by $\underline{\mathcal{X}} := \mathcal{X} \times \mathcal{X}$ the compounded index set. The Hilbert space $\underline{\mathcal{H}}$ of functions $k : \underline{\mathcal{X}} \to \mathbb{R}$, endowed with a dot product $\langle \cdot, \cdot \rangle$ (and the norm $\|k\| = \sqrt{\langle k, k \rangle}$) is called a Hyper Reproducing Kernel Hilbert Space if there exists a hyperkernel $\underline{k} : \underline{\mathcal{X}} \times \underline{\mathcal{X}} \to \mathbb{R}$ with the following properties:*

1. *$\underline{k}$ has the reproducing property*
$$\langle k, \underline{k}(\underline{x}, \cdot) \rangle = k(\underline{x}) \text{ for all } k \in \underline{\mathcal{H}}; \qquad (1)$$
in particular, $\langle \underline{k}(\underline{x}, \cdot), \underline{k}(\underline{x}', \cdot) \rangle = \underline{k}(\underline{x}, \underline{x}')$.
2. *$\underline{k}$ spans $\underline{\mathcal{H}}$, i.e. $\underline{\mathcal{H}} = \overline{\text{span}\{\underline{k}(\underline{x}, \cdot) | \underline{x} \in \underline{\mathcal{X}}\}}$ where $\overline{X}$ is the completion of the set X.*
3. *For any fixed $\underline{x} \in \underline{\mathcal{X}}$ the hyperkernel $\underline{k}$ is a kernel in its second argument, i.e. for any fixed $\underline{x} \in \underline{\mathcal{X}}$, the function $k(x, x') := \underline{k}(\underline{x}, (x, x'))$ with $x, x' \in \mathcal{X}$ is a kernel.*

What distinguishes $\underline{\mathcal{H}}$ from a normal RKHS is the particular form of its index set ($\underline{\mathcal{X}} = \mathcal{X}^2$) and the additional condition on $\underline{k}$ to be a kernel in its second argument for any fixed first argument. This condition somewhat limits the choice of possible kernels, on the other hand, it allows for simple optimization algorithms which consider kernels $k \in \underline{\mathcal{H}}$, which are in the convex cone of $\underline{k}$. Analogous to the regularized risk functional, $R_{\text{reg}}(f, X, Y) = \frac{1}{m} \sum_{i=1}^{m} l(x_i, y_i, f(x_i)) + \frac{\lambda}{2} \|f\|^2$, we regularize $Q_{\text{emp}}(k, X, Y)$.

Definition 3 (Regularized Quality Functional)
$$Q_{\text{reg}}(k, X, Y) := Q_{\text{emp}}(k, X, Y) + \frac{\lambda_Q}{2} \|k\|^2_{\underline{\mathcal{H}}} \qquad (2)$$
where $\lambda_Q > 0$ is a regularization constant and $\|k\|^2_{\underline{\mathcal{H}}}$ denotes the RKHS norm in $\underline{\mathcal{H}}$.

Minimization of Q_{reg} is less prone to overfitting than minimizing Q_{emp}, since the regularizer $\frac{\lambda_Q}{2} \|k\|^2_{\underline{\mathcal{H}}}$ effectively controls the complexity of the class of kernels under consideration (this can be derived from (Bousquet & Herrmann, 2003)). The minimizer of (2) satisfies the representer theorem:

Theorem 4 (Representer Theorem) *Denote by $\mathcal{X}$ a set, and by Q an arbitrary quality functional. Then each minimizer $k \in \underline{\mathcal{H}}$ of the regularized quality functional (2), admits a representation of the form*
$$k(x, x') = \sum_{i,j=1}^{m} \beta_{i,j} \underline{k}((x_i, x_j), (x, x')). \qquad (3)$$

This shows that even though we are optimizing over a whole Hilbert space of kernels, we still are able to

find the optimal solution by choosing among a finite number, which is the span of the kernel on the data.

Note that the minimizer (3) is not necessarily positive semidefinite. In practice, this is not what we want, since we require a positive semidefinite kernel. Therefore we need to impose additional constraints. We require that all expansion coefficients $\alpha_{i,j} \geqslant 0$. While this may prevent us from obtaining the minimizer of the objective function, it yields a much more amenable optimization problem in practice. In the subsequent derivations of optimization problems, we choose this restriction as it provides a tractable problem.

Similar to the analogy between Gaussian Processes (GP) and SVMs (Opper & Winther, 2000), there is a Bayesian interpretation for Hyperkernels which is analogous to the idea of hyperpriors. Our approach can be interpreted as drawing the covariance matrix of the GP from another GP.

3. Designing Hyperkernels

The criteria imposed by Definition 2 guide us directly in the choice of functions suitable as hyperkernels. The first observation is that we can optimize over the space of kernel functions, hence we can take large linear combinations of parameterized families of kernels as the basic ingredients. This leads to the so-called harmonic hyperkernels (Ong et al., 2003):

Example 1 (Harmonic Hyperkernel) *Denote by k a kernel with $k : \mathcal{X} \times \mathcal{X} \to [0,1]$, and set $c_i := (1-\lambda_h)\lambda_h^i$ for some $0 < \lambda_h < 1$. Then we have*

$$\begin{aligned} \underline{k}(\underline{x}, \underline{x}') &= (1-\lambda_h) \textstyle\sum_{i=0}^{\infty} \left(\lambda_h k(\underline{x}) k(\underline{x}')\right)^i \\ &= \frac{1-\lambda_h}{1-\lambda_h k(\underline{x}) k(\underline{x}')} \end{aligned} \tag{4}$$

A special case is $k(x,x') = \exp(-\sigma\|x-x'\|^2)$. Here we obtain for $\underline{k}((x,x'),(x'',x'''))$

$$\frac{1-\lambda}{1-\lambda\exp(-\sigma(\|x-x'\|^2 + \|x''-x'''\|^2))} \tag{5}$$

However, if we want the kernel to adapt automatically to different widths for each dimension, we need to perform the summation that led to (4) for each dimension in its arguments separately (similar to automatic relevance determination (MacKay, 1994)).

Example 2 (Hyperkernel for ARD) *Let $k_\Sigma(x,x') = \exp(-d_\Sigma(x,x'))$, where $d_\Sigma(x,x') = (x-x')^\top \Sigma (x-x')$, and Σ a diagonal covariance matrix. Take sums over each diagonal entry $\sigma_j = \Sigma_{jj}$ separately to obtain*

$$\begin{aligned} & \underline{k}((x,x'),(x'',x''')) \\ = \;& (1-\lambda_h) \textstyle\sum_{j=1}^{d} \sum_{i=0}^{\infty} \left(\lambda_h k_\Sigma(x,x') k_\Sigma(x'',x''')\right)^i \\ = \;& \prod_{j=1}^{d} \frac{1-\lambda_h}{1-\lambda_h \exp\left(-\sigma_j((x_j-x_j')^2 + (x_j''-x_j''')^2)\right)}. \end{aligned}$$

This is a valid hyperkernel since $k(\underline{x})$ factorizes into its coordinates. A similar definition also allows us to use a distance metric $d(x,x')$ which is a generalized radial distance as defined by (Haussler, 1999).

4. Semidefinite Programming

We derive Semidefinite Programming (SDP) formulations of the optimization problems arising from the minimization of the regularized risk functional. Semidefinite programming (Vandenberghe & Boyd, 1996) is the optimization of a linear objective function subject to constraints which are linear matrix inequalities and affine equalities. The following proposition allows us to derive a SDP from a class of general quadratic programs. It is an extension of the derivation in (Lanckriet et al., 2002) and its proof can be found in Appendix A.

Proposition 5 (Quadratic Minimax) *Let $m, n, M \in \mathbb{N}$, $H : \mathbb{R}^n \to \mathbb{R}^{m \times m}$, $c : \mathbb{R}^n \to \mathbb{R}^m$, be linear maps. Let $A \in \mathbb{R}^{M \times m}$ and $a \in \mathbb{R}^M$. Also, let $d : \mathbb{R}^n \to \mathbb{R}$ and $G(\theta)$ be a function and the further constraints on θ. Then the optimization problem*

$$\begin{array}{ll} \min_\theta \max_x & -\frac{1}{2} x^\top H(\theta) x - c(\theta)^\top x + d(\theta) \\ \textit{subject to} & H(\theta) \succeq 0 \\ & Ax + a \geqslant 0 \\ & G(\theta) \succeq 0 \end{array} \tag{6}$$

can be rewritten as

$$\begin{array}{ll} \min_{t,\theta,\gamma} & \frac{1}{2} t + a^\top \gamma + d(\theta) \\ \textit{subject to} & \gamma \geqslant 0,\ G(\theta) \succeq 0 \\ & \begin{bmatrix} H(\theta) & (A^\top \gamma - c(\theta)) \\ (A^\top \gamma - c(\theta))^\top & t \end{bmatrix} \succeq 0 \end{array} \tag{7}$$

Specifically, when we have the regularized quality functional, $d(\theta)$ is quadratic, and hence we obtain an optimization problem which has a mix of linear, quadratic and semidefinite constraints.

Corollary 6

$$\begin{array}{ll} \min_\theta \max_x & -\frac{1}{2} x^\top H(\theta) x - c(\theta)^\top x + \frac{1}{2} \theta^\top \Sigma \theta \\ \textit{subject to} & H(\theta) \succeq 0 \\ & Ax + a \geqslant 0 \\ & \theta \geqslant 0 \end{array} \tag{8}$$

can be rewritten as

$$\begin{array}{ll}\underset{t,t',\theta,\gamma}{\text{minimize}} & \frac{1}{2}t + \frac{1}{2}t' + a^\top\gamma \\ \textit{subject to} & \gamma \geqslant 0 \\ & \theta \geqslant 0 \\ & \|\Sigma^{\frac{1}{2}}\theta\| \leqslant t' \\ & \left[\begin{array}{cc} H(\theta) & (A^\top\gamma - c(\theta)) \\ (A^\top\gamma - c(\theta))^\top & t \end{array}\right] \succeq 0 \end{array} \tag{9}$$

The proof of the above is obtained immediately from Proposition 5 and introducing an auxiliary variable t' which upper bounds the quadratic term of θ.

5. Implementation Details

When Q_{emp} is the regularized risk, we obtain:

$$\min_{f\in\mathcal{H},k\in\underline{\mathcal{H}}} \frac{1}{m}\sum_{i=1}^{m} l(x_i,y_i,f(x_i)) + \frac{\lambda}{2}\|f\|^2_{\mathcal{H}} + \frac{\lambda_Q}{2}\|k\|^2_{\underline{\mathcal{H}}} \tag{10}$$

Comparing the objective function in (8) with (10), we observe that $H(\theta)$ and $c(\theta)$ are linear in θ. Let $\theta' = \varepsilon\theta$. As we vary ε the constraints are still satisfied, but the objective function scales with ε. Since θ is the coefficient in the hyperkernel expansion, this implies that we have a set of possible kernels which are just scalar multiples of each other. To avoid this, we add an additional constraint on θ which is $\mathbf{1}^\top\theta = 1$. This breaks the scaling freedom of the kernel matrix. As a side-effect, the numerical stability of the SDP problems improves considerably.

We give some examples of common SVMs which are derived from (10). The derivation is basically by application of Corollary 6. We derive the corresponding SDP for the case when Q_{emp} is a C-SVM (Example 3).

Derivations of the other examples follow the same reasoning, and are omitted. In this subsection, we define the following notation. For $p,q,r \in \mathbb{R}^n, n \in \mathbb{N}$ let $r = p \circ q$ be defined as element by element multiplication, $r_i = p_i \times q_i$. The pseudo-inverse (or Moore-Penrose inverse) of a matrix K is denoted $K^\dagger$. Define the hyperkernel Gram matrix $\underline{K}$ by $\underline{K}_{ijpq} = \underline{k}((x_i,x_j),(x_p,x_q))$, the kernel matrix $K = \text{reshape}(\underline{K}\beta)$ (reshaping a m^2 by 1 vector, $\underline{K}\beta$, to a m by m matrix), $Y = diag(y)$ (a matrix with y on the diagonal and zero everywhere else), $G(\beta) = YKY$ (the dependence on β is made explicit), $\mathbf{I}$ the identity matrix and $\mathbf{1}$ a vector of ones.

The number of training examples is assumed to be m. Where appropriate, γ and χ are Lagrange multipliers, while η and ξ are vectors of Lagrange multipliers from the derivation of the Wolfe dual for the SDP, β are the hyperkernel coefficients, t_1 and t_2 are the auxiliary variables.

Example 3 (Linear SVM (C-style)) *A commonly used support vector classifier, the C-SVM (Bennett & Mangasarian, 1992; Cortes & Vapnik, 1995) uses an ℓ_1 soft margin, $l(x_i,y_i,f(x_i)) = \max(0,1-y_if(x_i))$, which allows errors on the training set. The parameter C is given by the user. Setting the quality functional $Q_{\text{emp}}(k,X,Y) = \min_{f\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^m l(x_i,y_i,f(x_i)) + \frac{1}{2C}\|w\|^2_{\mathcal{H}}$, the resulting SDP is*

$$\begin{array}{ll}\underset{\beta,\gamma,\eta,\xi}{\text{minimize}} & \frac{1}{2}t_1 + \frac{C}{m}\xi^\top\mathbf{1} + \frac{C\lambda_Q}{2}t_2 \\ \textit{subject to} & \eta \geqslant 0, \xi \geqslant 0, \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \left[\begin{array}{cc} G(\beta) & z \\ z^\top & t_1 \end{array}\right] \succeq 0, \end{array} \tag{11}$$

where $z = \gamma y + \mathbf{1} + \eta - \xi$.

The value of α which optimizes the corresponding Lagrange function is $G(\beta)^\dagger z$, and the classification function, $f = sign(K(\alpha\circ y) - b_{offset})$, is given by $f = sign(KG(\beta)^\dagger(y\circ z) - \gamma)$.

Proof [Derivation of SDP for C-SVM] We begin our derivation from the regularized quality functional (10). Dividing throughout by λ and setting the cost function to the ℓ_1 soft margin loss, that is $l(x_i,y_i,f(x_i)) = \max(0,1-y_if(x_i))$ we get the following equation.

$$\begin{array}{ll}\min\limits_{k\in\underline{\mathcal{H}}}\min\limits_{f\in\mathcal{H}_k} & \frac{1}{m\lambda}\sum\limits_{i=1}^{m}\zeta_i + \frac{1}{2}\|f\|^2_{\mathcal{H}_k} + \frac{\lambda_Q}{2\lambda}\|k\|^2_{\underline{\mathcal{H}}} \\ \text{subject to} & y_if(x_i) \geqslant 1-\zeta_i \\ & \zeta_i \geqslant 0 \end{array} \tag{12}$$

Recall the form of the C-SVM,

$$\begin{array}{ll}\min\limits_{w,\zeta} & \frac{1}{2}\|w\|^2 + \frac{C}{m}\sum_{i=1}^m\zeta_i \\ \text{subject to} & y_i(\langle x_i,w\rangle + b) \geqslant 1-\zeta_i \\ & \zeta_i \geqslant 0 \text{ for all } i = 1,\ldots,m \end{array}$$

and its dual,

$$\begin{array}{ll}\max\limits_{\alpha\in\mathbb{R}^m} & \sum_{i=1}^m\alpha_i - \frac{1}{2}\sum_{i=1}^m\alpha_i\alpha_jy_iy_jk(x_i,x_j) \\ \text{subject to} & \sum_{i=1}^m\alpha_iy_i = 0 \\ & 0 \leqslant \alpha_i \leqslant \frac{C}{m} \text{ for all } i = 1,\ldots,m. \end{array}$$

By considering the optimization problem dependent on f in (12), we can use the derivation of the dual problem of the standard C-SVM. Observe that $C = \lambda^{-1}$, and

we can rewrite $\|k\|^2_{\underline{\mathcal{H}}} = \beta^\top \underline{K}\beta$ due to the representer theorem. Substituting the dual C-SVM problem into (12), we get the following matrix equation,

$$\begin{array}{ll} \min\limits_{\beta}\max\limits_{\alpha} & \mathbf{1}^\top\alpha - \frac{1}{2}\alpha^\top G(\beta)\alpha + \frac{C\lambda_Q}{2}\beta^\top\underline{K}\beta \\ \text{subject to} & \alpha^\top y = 0 \\ & 0 \leqslant \alpha_i \leqslant \frac{C}{m} \text{ for all } i = 1,\ldots,m \\ & \beta_i \geqslant 0 \end{array} \tag{13}$$

This is of the quadratic form of Corollary 6 where $x = \alpha$, $\theta = \beta$, $H(\theta) = G(\beta)$, $c(\theta) = -\mathbf{1}$, $\Sigma = C\lambda_Q\underline{K}$, the constraints are $A = \left[\begin{array}{cccc} y & -y & \mathbf{I} & -\mathbf{I}\end{array}\right]^\top$ and $a = \left[\begin{array}{cccc} 0 & 0 & \mathbf{0} & \frac{C}{m}\mathbf{1}\end{array}\right]^\top$. Applying Corollary 6, we obtain the SDP in Example 3. To make the different constraints explicit, we replace the matrix constraint $Ax + a \geqslant 0$ and its associated Lagrange multiplier γ with three linear constraints. We use γ as the Lagrange multiplier for the equality constraint $\alpha^\top y = 0$, η for $\alpha \geqslant 0$, and ξ for $\alpha \leqslant \frac{C}{m}\mathbf{1}$. ■

Example 4 (Linear SVM (ν-style)) *An alternative parameterization of the ℓ_1 soft margin was introduced by (Schölkopf et al., 2000), where the user defined parameter $\nu \in [0,1]$ controls the fraction of margin errors and support vectors. Using ν-SVM as Q_{emp}, that is, for a given ν, $Q_{\text{emp}}(k,X,Y) = \min_{f\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^m \zeta_i + \frac{1}{2}\|w\|^2_{\mathcal{H}} - \nu\rho$ subject to $y_i f(x_i) \geqslant \rho - \zeta_i$ and $\zeta_i \geqslant 0$ for all $i = 1,\ldots,m$. The corresponding SDP is given by*

$$\begin{array}{ll} \underset{\beta,\gamma,\eta,\xi,\chi}{\text{minimize}} & \frac{1}{2}t_1 - \chi\nu + \xi^\top\frac{1}{m} + \frac{\lambda_Q}{2}t_2 \\ \textit{subject to} & \chi \geqslant 0, \eta \geqslant 0, \xi \geqslant 0, \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \left[\begin{array}{cc} G(\beta) & z \\ z^\top & t_1 \end{array}\right] \succeq 0 \end{array} \tag{14}$$

where $z = \gamma y + \chi\mathbf{1} + \eta - \xi$.

The value of α which optimizes the corresponding Lagrange function is $G(\beta)^\dagger z$, and the classification function, $f = sign(K(\alpha\circ y) - b_{offset})$, is given by $f = sign(KG(\beta)^\dagger(y\circ z) - \gamma)$.

Example 5 (Quadratic SVM) *Instead of using an ℓ_1 loss class, (Mangasarian & Musicant, 2001) uses an ℓ_2 loss class,*

$$l(x_i,y_i,f(x_i)) = \left\{\begin{array}{ll} 0 & \text{if } y_i f(x_i) \geqslant 1 \\ (1-y_i f(x_i))^2 & \text{otherwise} \end{array}\right. ,$$

and regularized the weight vector as well as the bias term, that is the empirical quality functional is set to $Q_{\text{emp}}(k,X,Y) = \min_{f\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^m \zeta_i^2 + \frac{1}{2}(\|w\|^2_{\mathcal{H}} + b^2_{offset})$ subject to $y_i f(x_i) \geqslant 1 - \zeta_i$ and $\zeta_i \geqslant 0$ for all $i = 1,\ldots,m$. This is also known as the Lagrangian SVM. The resulting dual SVM problem has fewer constraints, as is evidenced by the smaller number of Lagrange multipliers needed in the SDP below.

$$\begin{array}{ll} \underset{\beta,\eta}{\text{minimize}} & \frac{1}{2}t_1 + \frac{\lambda_Q}{2}t_2 \\ \textit{subject to} & \eta \geqslant 0, \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \left[\begin{array}{cc} H(\beta) & (\eta+\mathbf{1}) \\ (\eta+\mathbf{1})^\top & t_1 \end{array}\right] \succeq 0 \end{array} \tag{15}$$

where $H(\beta) = Y(K + \mathbf{1}_{m\times m} + \lambda m\mathbf{I})Y$, and $z = \gamma\mathbf{1} + \eta - \xi$.

The value of α which optimizes the corresponding Lagrange function is $H(\beta)^\dagger(\eta+\mathbf{1})$, and the classification function, $f = sign(K(\alpha\circ y) - b_{offset})$, is given by $f = sign(KH(\beta)^\dagger((\eta+\mathbf{1})\circ y) + y^\top(H(\beta)^\dagger(\eta+\mathbf{1})))$.

Example 6 (Single class SVM) *For unsupervised learning, the single class SVM computes a function which captures regions in input space where the probability density is in some sense large (Schölkopf et al., 2001). The quality functional $Q_{\text{emp}}(k,X,Y) = \min_{f\in\mathcal{H}} \frac{1}{\nu m}\sum_{i=1}^m \zeta_i + \frac{1}{2}\|w\|^2_{\mathcal{H}} - \rho$ subject to $f(x_i) \geqslant \rho - \zeta_i$, and $\zeta_i \geqslant 0$ for all $i = 1,\ldots,m$, and $\rho \geqslant 0$. The corresponding SDP for this problem, also known as novelty detection, is shown below.*

$$\begin{array}{ll} \underset{\beta,\gamma,\eta,\xi}{\text{minimize}} & \frac{1}{2}t_1 + \xi^\top\frac{1}{\nu m} - \gamma + \frac{\lambda_Q}{2\nu}t_2 \\ \textit{subject to} & \eta \geqslant 0, \xi \geqslant 0, \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \left[\begin{array}{cc} K & z \\ z^\top & t_1 \end{array}\right] \succeq 0 \end{array} \tag{16}$$

where $z = \gamma\mathbf{1} + \eta - \xi$, and $\nu \in [0,1]$ a user selected parameter controlling the proportion of the data to be classified as novel.

The score to be used for novelty detection is given by $f = K\alpha - b_{offset}$, which reduces to $f = \eta - \xi$, by substituting $\alpha = K^\dagger(\gamma\mathbf{1} + \eta - \xi)$, $b_{offset} = \gamma\mathbf{1}$ and $K = reshape(\underline{K}\beta)$.

Example 7 (ν-Regression) *We derive the SDP for ν regression (Schölkopf et al., 2000), which automatically selects the ε insensitive tube for regression. As in the ν-SVM case in Example 4, the user defined parameter ν controls the fraction of errors and support vectors. Using the ε-insensitive loss, $l(x_i,y_i,f(x_i)) = \max(0, |y_i - f(x_i)| - \varepsilon)$, and the ν-parameterized quality functional, $Q_{\text{emp}}(k,X,Y) = \min_{f\in\mathcal{H}} C\left(\nu\varepsilon + \frac{1}{m}\sum_{i=1}^m(\zeta_i + \zeta_i^*)\right)$ subject to $f(x_i) - y_i \leqslant \varepsilon - \zeta_i$, $y_i - f(x_i) \leqslant \varepsilon - \zeta_i^*$, $\zeta_i^{(*)} \geqslant 0$ for all*

Data	C-SVM	ν-SVM	Lag-SVM	Other	CV Tuned SVM
syndata	2.8±2.2	**1.2±1.3**	2.5±2.4	NA	15.2±3.8
pima	24.5±1.6	28.7±1.5	23.7±1.7	**23.5**	24.8±1.9
ionosph	7.3±1.9	7.4±1.7	7.1±2.0	**5.8**	6.8±1.7
wdbc	2.8±0.7	4.1±1.7	**2.5±0.6**	3.2	7.0±1.5
heart	19.7±2.7	19.5±2.1	19.8±2.4	**16.0**	23.8±3.2
thyroid	6.6±3.6	9.0±4.3	5.5±3.4	**4.4**	5.2±3.3
sonar	15.2±3.2	15.7±3.9	**14.9±3.4**	15.4	15.8±3.6
credit	14.8±1.7	**13.8±1.1**	15.3±1.8	22.8	24.3±1.9
glass	**5.2±2.3**	7.7±3.3	**5.2±1.5**	NA	6.0±1.7

Table 1. Hyperkernel classification: Test error and standard deviation in percent.The second, third and fourth columns show the results of the hyperkernel optimizations of C-SVM (Example 3), ν-SVM (Example 4) and Lagrangian SVM (Example 5) respectively. The results in the fifth column shows the best results from (Freund & Schapire, 1996; Rätsch et al., 2001; Meyer et al., 2003). The rightmost column shows a C-SVM tuned in the traditional way. A Gaussian RBF kernel was tuned using 10-fold cross validation on the training data, with the best value of C shown in brackets. A grid search was performed on (C, σ). The values of C tested were $\{10^{-1}, 10^0, \ldots, 10^5\}$. The values of the kernel width, σ, tested were between 10% and 90% quantile of the distance between a pair of sample of points in the data. These quantiles were estimated by a random 20% sample of the training data.

$i = 1, \ldots, m$ *and* $\varepsilon \geqslant 0$. *The corresponding SDP is*

$$\begin{array}{ll} \underset{\beta,\gamma,\eta,\xi,\chi}{\text{minimize}} & \frac{1}{2}t_1 + \frac{\chi\nu}{\lambda} + \xi^\top \frac{1}{m\lambda} + \frac{\lambda_Q}{2\lambda}t_2 \\ \textit{subject to} & \chi \geqslant 0, \eta \geqslant 0, \xi \geqslant 0, \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \begin{bmatrix} F(\beta) & z \\ z^\top & t_1 \end{bmatrix} \succeq 0 \end{array}, \tag{17}$$

where $z = \begin{bmatrix} -y \\ y \end{bmatrix} - \gamma \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix} + \eta - \xi - \chi \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix}$ *and* $F(\beta) = \begin{bmatrix} K & -K \\ -K & K \end{bmatrix}$.

The Lagrange function is minimized for $\alpha = F(\beta)^\dagger z$, *and substituting into* $f = K\alpha - b_{offset}$, *we obtain the regression function* $f = \begin{bmatrix} -K & K \end{bmatrix} F(\beta)^\dagger z - \gamma$.

Example 8 (Kernel Target Alignment) *For the Alignment approach (Cristianini et al., 2002),* $Q_{\text{emp}} = y^\top K y$, *we directly minimize the regularized quality functional, obtaining the following optimization problem,*

$$\begin{array}{ll} \underset{\beta}{\text{minimize}} & \frac{1}{2}t_1 + \frac{\lambda_Q}{2}t_2 \\ \textit{subject to} & \beta \geqslant 0 \\ & \|\underline{K}^{\frac{1}{2}}\beta\| \leqslant t_2 \\ & \begin{bmatrix} K & y \\ y^\top & t_1 \end{bmatrix} \succeq 0 \end{array} \tag{18}$$

Note that for the case of Alignment, Q_{emp} *does not provide a direct formulation for the hypothesis function, but instead, it determines a kernel matrix* K. *This kernel matrix,* K, *can be utilized in a traditional SVM to obtain a classification function.*

6. Experiments

We used data from the UCI repository for our experiments. Where the data was numerical, we *did not perform any preprocessing* of the data. Boolean attributes were converted to {-1,1}, and categorical attributes were arbitrarily assigned an order, and numbered {1, 2,...}. The hyperkernel used was as in Example 2. This scaling freedom means that we did not have to normalize data to some arbitrary distribution. Similar to Ong et al. (2003), we used a low rank decomposition (Fine & Scheinberg, 2000; Zhang, 2001) for the hyperkernel matrix.

6.1. Classification Experiments

A set of synthetic data sampled from two Gaussians was created, a sample of which is illustrated in Figure 1. The rest of the datasets were UCI datasets for binary classification tasks. The datasets were split into 10 random permutations of 60% training data and 40% test data. We deliberately did not attempt to tune parameters and instead made the following choices uniformly for all datasets:

- The kernel width σ was set to 50 times the 90% quantile of the value of $|x_i - x_j|$ over all the training data, which ensures sufficient coverage.
- λ was adjusted so that $\frac{1}{\lambda m} = 100$ (that is $C = 100$ in the Vapnik-style parameterization of SVMs). This has commonly been reported to yield good results.
- ν was set to 0.3. While this is clearly suboptimal for many datasets, we decided to choose it beforehand to avoid having to change *any* parameter. We could

use previous reports on generalization performance to set ν to this value for better performance.

- λ_h for the Gaussian Harmonic Hyperkernel was chosen to be 0.6 throughout, giving adequate coverage over various kernel widths in (4) (small λ_h focus almost exclusively on wide kernels, λ_h close to 1 will treat all widths equally).
- The hyperkernel regularization was set to $\lambda_Q = 1$.

We observe (Table 1) that our method achieves state of the art results for all the datasets, except the "heart" dataset. We also achieve results much better than previously reported for the "credit" dataset. Comparing the results for C-SVM and Tuned SVM, we observe that our method is always equally good, or better than a C-SVM tuned using 10-fold cross validation.

6.2. Novelty Detection Experiments

To demonstrate that we can solve problems other than binary classification using the same framework, we performed novelty detection. We apply the singleclass support vector machine (Example 6) to detect outliers in the USPS data. A subset of 300 randomly selected USPS images for the digit '5' were used for the experiments. The parameter ν was set to 0.1 for these experiments, hence selecting up to 10% of the data as outliers. The *rest of the parameters were the same* as in the previous section. Since there is no quantitative method for measuring the performance of novelty detection, we cannot directly compare our results with the traditional single class SVM. We can only subjectively conclude, by visually inspecting a sample of the digits, that our approach works for novelty detection of USPS digits. Figure 2 shows a sample of the digits. We can see that the algorithm identifies 'novel' digits, such as in the top row of Figure 2. The bottom row shows a sample of digits which have been deemed to be 'common'.

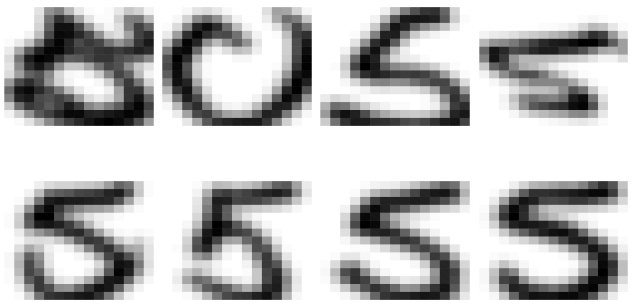


Figure 2. Top: Images of digit '5' considered novel by algorithm; Bottom: Common images of digit '5'

7. Discussion and Conclusion

We have shown that it is possible to define a convex optimization problem which learns the best kernel given the data. The resulting problem, which has a Bayesian interpretation, is expressed as a SDP. Since we can optimize over the whole class of kernel functions, we can define more general kernels which may have many free parameters, without overfitting. The experimental results on classification and novelty detection demonstrate that it is possible to achieve the state of the art, and in certain cases (such as the credit data) improve the accuracy significantly.

This approach makes support vector based estimation approaches more automated. Parameter adjustment is less critical compared to the case when the kernel is fixed. Future work will focus on deriving improved statistical guarantees for estimates derived via hyperkernels which match the good empirical performance.

Acknowledgements This work was supported by a grant of the Australian Research Council. The authors would like to thank Laurent El Ghaoui, Michael Jordan, John Lloyd, Robert Williamson and Daniela Pucci de Farias for their helpful comments and suggestions. The authors also thank Alexandros Karatzoglou for his help with SVLAB.

References

Albert, A. (1969). Conditions for positive and nonnegative definiteness in terms of pseudoinverses. *SIAM Journal on Applied Mathematics*, *17*, 434–440.

Bennett, K. P., & Mangasarian, O. L. (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, *1*, 23–34.

Bousquet, O., & Herrmann, D. (2003). On the complexity of learning the kernel matrix. *Advances in Neural Information Processing Systems 15*.

Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, *20*, 273–297.

Crammer, K., Keshet, J., & Singer, Y. (2003). Kernel design using boosting. *Advances in Neural Information Processing Systems 15*.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., & Kandola, J. (2002). On kernel-target alignment. *Advances in Neural Information Processing Systems 14* (pp. 367–373). Cambridge, MA: MIT Press.

Fine, S., & Scheinberg, K. (2000). *Efficient SVM training using low-rank kernel representation* (Technical Report). IBM Watson Research Center, New York.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the International Conference on Machine Learing* (pp. 148–146). Morgan Kaufmann Publishers.

Haussler, D. (1999). *Convolutional kernels on discrete structures* (Technical Report UCSC-CRL-99-10). Computer Science Department, UC Santa Cruz.

Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. (2002). Learning the kernel matrix with semidefinite programming. *Proceedings of the International Conference on Machine Learning* (pp. 323–330). Morgan Kaufmann.

MacKay, D. J. C. (1994). Bayesian non-linear modelling for the energy prediction competition. *American Society of Heating, Refrigerating and Air-Conditioning Engineers Transcations*, *4*, 448–472.

Mangasarian, O. L., & Musicant, D. R. (2001). Lagrangian support vector machines. *Journal of Machine Learning Research*, *1*, 161–177. http://www.jmlr.org.

Meyer, D., Leisch, F., & Hornik, K. (2003). The support vector machine under test. *Neurocomputing*. Forthcoming.

Momma, M., & Bennett, K. P. (2002). A pattern search method for model selection of support vector regression. *Proceedings of the Second SIAM International Conference on Data Mining*.

Ong, C. S., Smola, A. J., & Williamson, R. C. (2003). Hyperkernels. *Advances in Neural Information Processing Systems 15*.

Opper, M., & Winther, O. (2000). Gaussian processes and SVM: Mean field and leave-one-out. *Advances in Large Margin Classifiers* (pp. 311–326). Cambridge, MA: MIT Press.

Rätsch, G., Onoda, T., & Müller, K. R. (2001). Soft margins for adaboost. *Machine Learning*, *42*, 287–320.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, *13*, 1443–1471.

Schölkopf, B., Smola, A., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, *12*, 1207–1245.

Vandenberghe, L., & Boyd, S. (1996). Semidefinite programming. *SIAM Review.*, *38*, 49–95.

Zhang, T. (2001). Some sparse approximation bounds for regression problems. *Proc. 18th International Conf. on Machine Learning* (pp. 624–631). Morgan Kaufmann, San Francisco, CA.

A. Proof of Proposition 5

We prove the proposition that the solution of the quadratic minimax problem (6) is obtained by minimizing the SDP (7).

Proof Rewrite the terms of the objective function in (6) dependent on x in terms of their Wolfe dual. The corresponding Lagrange function is

$$L(x,\theta,\gamma) = -\frac{1}{2}x^\top H(\theta)x - c(\theta)^\top x + \gamma^\top(Ax+a), \quad (19)$$

where $\gamma \in \mathbb{R}^M$ is a vector of Lagrange multipliers with $\gamma \geqslant 0$. By differentiating $L(x,\theta,\gamma)$ with respect to x and setting the result to zero, one obtains that (19) is maximized with respect to x for $x = H(\theta)^\dagger(A^\top\gamma - c(\theta))$ and subsequently we obtain the dual

$$D(\theta,\gamma) = \frac{1}{2}(A^\top\gamma - c(\theta))^\top H(\theta)^\dagger(A^\top\gamma - c(\theta)) + \gamma^\top a. \quad (20)$$

Note that $H(\theta)^\dagger H(\theta)H(\theta)^\dagger = H(\theta)^\dagger$. For equality constraints in (6), such as $Bx + b = 0$, we get correspondingly free dual variables. The dual optimization problem is given by inserting (20) into (6)

$$\begin{array}{ll} \underset{\theta,\gamma}{\text{minimize}} & \frac{1}{2}(A^\top\gamma - c(\theta))^\top H(\theta)^\dagger(A^\top\gamma - c(\theta)) \\ & +\gamma^\top a + d(\theta) \\ \text{subject to} & H(\theta) \succeq 0, G(\theta) \succeq 0, \gamma \geqslant 0. \end{array} \quad (21)$$

Introducing an auxiliary variable, t, which serves as an upper bound on the quadratic objective term gives an objective function linear in t and γ. Then (21) can be written as

$$\begin{array}{ll} \underset{\theta,\gamma}{\text{minimize}} & \frac{1}{2}t + \gamma^\top a + d(\theta) \\ \text{subject to} & t \succeq (A^\top\gamma - c(\theta))^\top H(\theta)^\dagger(A^\top\gamma - c(\theta)), \\ & H(\theta) \succeq 0, G(\theta) \succeq 0, \gamma \geqslant 0. \end{array} \quad (22)$$

From the properties of the Moore-Penrose inverse, we get $H(\theta)H(\theta)^\dagger(A^\top\gamma - c(\theta)) = (A^\top\gamma - c(\theta))$. Since $H(\theta) \succeq 0$, by the Schur complement lemma (Albert, 1969), the quadratic constraint in (22) is equivalent to

$$\begin{bmatrix} H(\theta) & (A^\top\gamma - c(\theta)) \\ (A^\top\gamma - c(\theta))^\top & t \end{bmatrix} \succeq 0 \quad (23)$$

■