Ellipsoidal Kernel Machines

Pannagadatta K. Shivaswamy
Department of Computer Science
Columbia University, New York, NY 10027
pks2103@cs.columbia.edu

Tony Jebara
Department of Computer Science
Columbia University, New York, NY 10027
jebara@cs.columbia.edu

Abstract

A novel technique is proposed for improving the standard Vapnik-Chervonenkis (VC) dimension estimate for the Support Vector Machine (SVM) framework. The improved VC estimates are based on geometric arguments. By considering bounding ellipsoids instead of the usual bounding hyperspheres and assuming gap-tolerant classifiers, a linear classifier with a given margin is shown to shatter fewer points than previously estimated. This improved VC estimation method directly motivates a different estimator for the parameters of a linear classifier. Surprisingly, only VC-based arguments are needed to justify this modification to the SVM. The resulting technique is implemented using Semidefinite Programming (SDP) and is solvable in polynomial time. The new linear classifier also ensures certain invariances to affine transformations on the data which a standard SVM does not provide. We demonstrate that the technique can be kernelized via extensions to Hilbert spaces. Promising experimental results are shown on several standardized datasets.

1 INTRODUCTION

Binary classification plays a central role in machine learning and is relevant to many applied domains. Given a training set $(\mathbf{x}_i, y_i)_{i=1}^n \in \mathbb{R}^m \times \{\pm 1\}$, the objective of binary classification is to learn a function $f : \mathbb{R}^m \to \{\pm 1\}$. The Support Vector Machine (SVM) (Vapnik, 1995) has obtained considerable interest in the machine learning community over the past decade for solving such classification problems well. In its simplest form, an SVM learns a linear function $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ with $\mathbf{w} \in \mathbb{R}^m, b \in \mathbb{R}$ whose sign agrees with the labels of the training samples while maximizing the so-called margin (the distance between the hyperplanes $\mathbf{w}^\top \mathbf{x} + b = 1$ and $\mathbf{w}^\top \mathbf{x} + b = -1$ that also agree with the labelling of the data). The success of SVMs is frequently ascribed to their firm foundation in learning theory. For example, Vapnik-Chervonenkis (VC) theory (Vapnik, 1995) suggests two conditions for the success of a learning machine: low error on training data and small VC dimension (Burges, 1998). The argument for small VC dimension is realized for SVMs due to their maximization of margins. However, this argument assumes that the data is constrained to lie within a sphere since the margin is meaningful only relative to the diameter of this sphere. This so-called *gap-tolerant classifier* with a given margin frequently has a lower VC dimension than unconstrained linear classifiers. Gap-tolerant classifiers work by maximizing the margin for data lying in a sphere of diameter at most D and have a VC dimension of at most $\min\{\lceil \frac{D^2}{M^2} \rceil, m\} + 1$, where M is the minimum margin of the class of classifiers. In this article, we show that this bound on VC dimension can be further improved by considering gap-tolerant classifiers within ellipsoids instead of simple hyperspheres.

There have been several prior efforts to improve SVMs. These methods sometimes modify the basic SVM linear classifier without making any other assumptions about the learning problem. However, some methods enrich the linear classifier with additional flexibility or domain knowledge. Methods that only modify the basic linear classifier may use a different line of reasoning as a surrogate to the standard VC dimension generalization arguments. For example, the Bayes Point Machine (Herbrich et al., 2001) approximates the Bayes-optimal decision by the center of mass of the version space and averages over the set of linear classifier parameters that agree with the training data. Similarly, the Relevance Vector Machine (Tipping, 2001) follows a Bayesian treatment of a generalized linear model of the same functional form as an SVM. Different norms and different penalties have also been explored; the l_1

norm SVM (Zhu et al., 2004) minimizes the l_1 norm of $\mathbf{w}$ rather than the standard l_2 norm. While the standard SVM uses a linear penalty, SVMs with quadratic penalties have also been proposed. In addition to basic reformulations, there have been efforts to specialize SVMs to particular situations. The Minimax Probability Machine extends the SVM to the cases where the data is specified only in terms of a distribution (Lanckriet et al., 2002). There have also been several efforts to make SVMs invariant and robust by explicitly accounting for input uncertainty (Shivaswamy et al., 2006), transduction (Vapnik, 1995), unknown transformations on inputs (Shivaswamy & Jebara, 2006), kernel selection and feature selection (Weston et al., 2000).

Unfortunately, relatively little work directly addresses improving the base SVM classifier solely by optimizing the VC bounds that motivated it in the first place. This article takes this approach and improves SVMs without exploiting additional domain knowledge, norms, and so forth. The article simply seeks tighter bounds on the VC dimension to improve the linear decision boundary of the SVM. Thus, this improvement in SVMs is a general one and should complement other SVM extensions nicely. This article proposes the Ellipsoidal Kernel Machine (EKM for short), which is still a large margin linear classifier like the SVM but instead estimates ellipsoidal bounds on the data to properly optimize margin in the *correct* direction. This article shows that SVMs and VC dimension estimates based solely on a hypersphere model of the data distribution can lead to suboptimal linear decision boundaries from both an empirical and a VC dimension perspective.

2 SVMS AND GENERALIZATION

2.1 SUPPORT VECTOR MACHINES

Recall the SVM classifier $f(x) = sign(\mathbf{w}^\top \mathbf{x} + b) \in \{\pm 1\}$ with parameters $(\mathbf{w}, b)$ estimated by the following Quadratic Programming (QP) formulation:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \text{ s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \quad \forall 1 \leq i \leq n. \tag{1}$$

The margin is the distance between the two hyperplanes $\mathbf{w}^\top \mathbf{x} + b = 1$ and $\mathbf{w}^\top \mathbf{x} + b = -1$ and is given by $M = \frac{2}{\|\mathbf{w}\|}$. The above formulation can be relaxed to nonseparable problems via the QP below: [1]:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^n \xi_i \\ &\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \ \xi_i \geq 0. \end{aligned} \tag{2}$$

[1]We drop $1 \leq i \leq n$ since it is clear from the context.

where ξ_i is the penalty for violating the constraints. The parameter C is the trade off between the margin and the penalties incurred. We next review the traditional structural risk minimization argument which originally motivated the above SVM formulation and show how some of its deficiencies can be improved.

2.2 STRUCTURAL RISK MINIMIZATION

Suppose that data $(\mathbf{x}, y)$ comes from a fixed but unknown distribution $P(\mathbf{x}, y)$. Our aim is to learn a mapping $\mathbf{x} \rightarrow y$. Consider the possibility of learning this mapping using a family of functions $f(\mathbf{x}, \alpha) \in \{-1, 1\}, \forall \mathbf{x}, \alpha$, where α parameterizes the family. For any fixed α, the risk is defined as the expected test error: $R(\alpha) = \int \frac{1}{2}|y - f(\mathbf{x}, \alpha)| dP(\mathbf{x}, y)$. Ideally, we would select α to minimize this risk. However, $R(\alpha)$ is not directly computable. Instead, consider the empirical risk, another quantity of interest which can be computed: $R_{emp}(\alpha) = \frac{1}{2n}\sum_{i=1}^n |y_i - f(\mathbf{x}_i, \alpha)|$. For any $0 \leq \eta \leq 1$, the following bound holds with probability at least $1 - \eta$ (Vapnik, 1995):

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\ln(2n/h) + 1) - \ln(\eta/4)}{n}}$$

where h is the VC dimension of the family of classifiers. Informally, the VC dimension is a measure of the complexity or capacity of the family of classifiers. Note that as the VC dimension becomes smaller, the bound on risk becomes lower. Structural risk minimization suggests selecting α to minimize the bound on the right hand side hoping that the true risk $R(\alpha)$ will also be lowered. The smaller the VC dimension, the tighter the bound. Eventually $R(\alpha)$ should be lowered and the generalization ability of the learning machine will be good. Even if the bounds are loose or vacuous, it is generally the case that any small improvements in the VC dimension will typically improve generalization. We next show how to estimate the VC dimension of SVMs to implement structural risk minimization by using gap-tolerant classifiers.

2.3 GAP-TOLERANT CLASSIFIERS

Traditionally, a gap-tolerant classifier is specified by the location and the diameter of a ball in $\mathbb{R}^m$ that encloses all data, and by two parallel hyperplanes that separate that data into two classes. The VC dimension of this family of classifiers in $\mathbb{R}^m$ with minimum margin M and maximum diameter D is upper bounded by $\min\{\lceil \frac{D^2}{M^2} \rceil, m\} + 1$ (Burges, 1998). Therefore, to reduce risk, we must drive down the VC dimension by maximizing M while keeping the empirical risk low or zero. This is precisely what the separable SVM in Equation (1) does as it minimizes $\frac{1}{2}\|\mathbf{w}\|^2 = \frac{2}{M^2}$.

Estimating the hypersphere that contains data is also solvable as a quadratic program. However, it is typically not a necessary step when learning an SVM. This computation only plays a role in evaluating the VC dimension of SVM. The hypersphere itself is sometimes necessary to compute in certain generalizations of SVMs where the data itself and its bounding hypersphere become variable (Burges, 1998; Vapnik, 1995; Weston et al., 2000). This is the case, for instance, during feature selection (Weston et al., 2000). Another theorem in (Weston et al., 2000) also relates the ratio $\frac{D^2}{M^2}$ to the generalization error:

Theorem 2.1 *If training data of size n belonging to a hyper-sphere of size R are separable with a corresponding margin M, then the expectation of the error probability has the bound $\mathbb{E}\,\mathrm{Pr}_{err} \leq \frac{1}{n}\mathbb{E}\left\{\frac{R^2}{M^2}\right\}$, where the expectation is over sets of training data of size n.*

Thus, the SVM generalization arguments depend not only on the margin but also on the diameter of the data. This leads to the question: can we leverage a richer family of gap-tolerant classifiers to improve generalization?

2.4 ELLIPSOIDAL GAP-TOLERANT CLASSIFIERS

We now extend the traditional hypersphere-bounded gap-tolerant classifier by considering ellipsoid-bounded gap-tolerant classifiers. It will be straightforward to show that the VC dimension h_E of the ellipsoid-bounded gap-tolerant classifiers is upper bounded by the previous estimate of VC dimension ($h_E \leq \min\{\lceil\frac{D^2}{M^2}\rceil, m\} + 1$) and this will eventually lead to an interesting variant of the SVM.

An ellipsoid in $\mathbb{R}^m$ is defined by $\mathcal{E} = \{\mathbf{x} : (\mathbf{x} - \mu)^\top \mathbf{\Sigma^{-1}}(\mathbf{x} - \mu) \leq \mathbf{1}\}$ where $\mu \in \mathbb{R}^m$ and $\mathbf{\Sigma} \in \mathbb{R}^{m\times m}$ and $\mathbf{\Sigma} \succeq 0$[2] (positive semi-definite). This ellipsoid along with two parallel hyperplanes defines an ellipsoidal gap-tolerant classifier. In contrast, a spherical gap-tolerant classifier is the same except that it has a sphere instead of an ellipsoid. Since input dimensions in real datasets are rarely independent or isotropic, it is more likely for data to lie in an ellipsoid than in a sphere. Furthermore, the hypersphere only summarizes the data by a single scalar radius while the ellipsoid models several axes and magnitudes of variation. In addition, one can immediately reconstruct a bounding hypersphere from a bounding ellipsoid by setting the magnitudes of variation for each axis equal to the maximum magnitude. The most useful property of ellipsoidal gap-tolerant classifiers, though, is that their VC dimension is lower than the VC dimension of traditional gap-tolerant classifier.

[2] We assume that $\mathbf{\Sigma}$ is of full rank, it is possible to handle the non-full rank $\mathbf{\Sigma}$ with simple modifications.

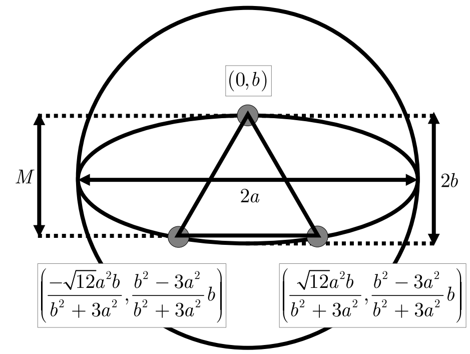

Figure 1: The maximum margin configuration of three points.

Theorem 2.2 *Consider a gap-tolerant classifier with a bounding hypersphere of radius R. Consider an ellipsoidal gap-tolerant classifier with the length of the semi-major axis (the axis along the highest eigenvalue direction) R. The VC dimension of the latter is at most the VC dimension of the former.*

Proof Without loss of generality, assume that the centroids of the sphere and the ellipsoid coincide. Suppose that for a given margin, the two hyperplanes can shatter p points in the ellipsoid, then the same p points within the sphere can also be shattered by the same margin within the hypersphere. ∎

Thus we see that the VC dimension of the ellipsoidal gap-tolerant classifier is at most the VC dimension of the spherical gap-tolerant classifier. It is also possible for the VC dimension of the ellipsoidal gap-tolerant classifier to be strictly lower than that of the spherical gap-tolerant classifier. We demonstrate this reduction in VC dimension for a 2D configuration and then provide arguments for higher dimensions as well.

Consider an axis parallel ellipse in $\mathbb{R}^2$. Let the length of the semi-major axis (along the x-axis) be a and the semi-minor axis (along the y-axis) be b, i.e., consider the ellipse defined by $\mathcal{E} = \{(x, y) : \frac{x^2}{a^2} + \frac{y^2}{b^2} \leq 1\}$. Recall that in $\mathbb{R}^2$, no more than three points can be shattered by a linear classifier. Consider placing three points on the ellipse so as to maximize the margin. In this two dimensional case, the margin over all possible binary labellings of the three points is maximized when the points are placed at vertices of the largest equilateral triangle inscribed in the ellipse. More specifically, the maximum margin setting of the three points is at $(0, b), (\frac{\sqrt{12}a^2b}{b^2+3a^2}, \frac{b^2-3a^2}{b^2+3a^2}b), (-\frac{\sqrt{12}a^2b}{b^2+3a^2}, \frac{b^2-3a^2}{b^2+3a^2}b)$. The margin at that setting is $\frac{6a^2b}{b^2+3a^2}$ which is found by projecting one of the points on the line segment connecting the other two points. The situation is depicted in Figure 1.

Now consider the sphere $x^2 + y^2 \leq a^2$. The three points that maximize the margin can simply be found by setting $b = a$ for the vertices of the above equilateral triangle. Thus, the three points are placed at $(0, a), (\frac{\sqrt{3}}{2}a, -\frac{1}{2}a), (-\frac{\sqrt{3}}{2}a, -\frac{1}{2}a)$. The maximum possible margin to shatter three points in this case is $\frac{3}{2}a$. It is obvious that $\frac{3}{2}a > \frac{6a^2b}{b^2+3a^2}$ for $a > b$. In this scenario, if the margin M allowed is such that $\frac{3}{2}a > M > \frac{6a^2b}{b^2+3a^2}$, then only two points can be shattered with the ellipsoidal gap-tolerant classifier whereas the spherical one can shatter three points. Thus, we have shown a scenario in which the VC dimension of the ellipsoidal gap-tolerant classifier is strictly less than the VC dimension of the spherical gap-tolerant classifier. Unfortunately, extending the 2D analysis above to higher dimensions is not as simple since the simplex for the highest margin is tilted and several asymmetries emerge.

The above arguments show that there is an advantage to considering the ellipsoid around the data rather than hyper sphere around that data to reduce the VC dimension of the classifiers being considered. In practice, leveraging this ellipsoid to improve our estimate of a linear classifier is straightforward. By transforming the data using the structure of the the bounding ellipsoid, we apply an affine transformation to it and stretch it such that the bounding ellipsoid has equal axis lengths in all direction. We can then estimate the large margin linear decision boundary using the SVM in this transformed space and obtain a linear classifier that minimizes our improved estimate of VC dimension. This procedure is detailed in the next Section.

3 ELLIPSOIDAL MACHINES

To make use of the ellipsoidal structure of the data, we propose estimating a bounding ellipsoid via semidefinite programming. Suppose the ellipsoid around the data $(\mathbf{x}_i)_{i=1}^n$ is characterized by $(\mathbf{x}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu) \leq 1$, the expression for points within the ellipsoid can be written as follows:

$$\begin{aligned}(\mathbf{x}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu) &= (\mathbf{x}-\mu)^\top \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{\Sigma}^{-\frac{1}{2}}(\mathbf{x}-\mu) \\ &= (\mathbf{A}\mathbf{x}-\mathbf{b})^\top(\mathbf{A}\mathbf{x}-\mathbf{b}) \leq 1\end{aligned}$$

where $\mathbf{A} = \mathbf{\Sigma}^{-\frac{1}{2}}$ and $\mathbf{b} = \mathbf{\Sigma}^{-\frac{1}{2}}\mu$. Thus, the problem of finding the minimum volume ellipsoid around $(\mathbf{x}_i)_{i=1}^n$ can be expressed as

$$\min_{\mathbf{A},\mathbf{b}} \; -\ln|\mathbf{A}| \;\; \text{s.t.} \quad \|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|^2 \leq 1, \;\; \mathbf{A} \succeq 0 \quad (3)$$

where, $|\mathbf{A}|$ denotes the determinant of the matrix $\mathbf{A}$. The smallest enclosing ellipsoid is the one that has the smallest volume yet encloses all the given points. We measure the volume of an ellipsoid via the determinant of $\mathbf{\Sigma}$. Instead of explicitly minimizing the determinant of $\mathbf{\Sigma}$, we equivalently maximize the determinant of the inversely-related matrix $\mathbf{A}$ since $\mathbf{A} = \mathbf{\Sigma}^{-\frac{1}{2}}$. After solving (3) both μ and $\mathbf{\Sigma}$ can be recovered using matrix inversion and simple algebra.

In practice, however, real world data has measurement noise and outliers so the bounding ellipsoid above is not necessarily the most reliable way to estimate the ellipsoidal gap-tolerant classifier. Just like a relaxation was used in Equation (2) to handle outliers when learning a linear classifier, we will relax the above algorithm to find an ellipsoid around data that is robust to noise and outliers. The following relaxed version of the above formulation is thus used in practice:

$$\begin{aligned}&\min_{\mathbf{A},\mathbf{b},\tau} \; -\ln|\mathbf{A}| + E\sum_{i=1}^n \tau_i \qquad (4)\\ &\text{s.t. } \|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|^2 \leq 1 + \tau_i, \; \tau_i \geq 0 \quad \mathbf{A} \succeq 0\end{aligned}$$

where τ_i is a penalty on the samples that remain outside the ellipsoid, E is a parameter that trades off the volume of the ellipsoid with the penalty on the outliers. The formulation (4) is an instance of a semidefinite program (SDP) (Boyd & Vandenberghe, 2003) which can be efficiently solved in polynomial time.

Once the bounding ellipsoid is found, each pattern is transformed using μ and $\mathbf{\Sigma}$ obtained using, $\mathbf{t}_i = \mathbf{\Sigma}^{-\frac{1}{2}}(\mathbf{x}_i - \mu)$, where $\mathbf{t}_i \in \mathbb{R}^m$. Thus the points $\mathbf{t}_i$ are now within a (relaxed) unit hypersphere centered at the origin. The modified training patterns $(\mathbf{t}_i, y_i)$ are now used for training the SVM. To test a pattern $\mathbf{x}$, compute the same transformation ($\mathbf{t} = \mathbf{\Sigma}^{-\frac{1}{2}}(\mathbf{x}-\mu)$) and then test by predicting the sign of $\mathbf{w}^\top\mathbf{t} + b$ as the label. Table 1 summarizes the EKM training algorithm which has two regularization parameters E and C corresponding to finding an ellipsoid and finding a separating hyperplane respectively.

We now note some interesting properties of the learned EKM. Consider points on the hyperplane decision boundary $\mathbf{w}^\top\mathbf{x} + b = 0$ which we obtain by solving the standard SVM formulation (2). Since $\mathbf{x} = \mu + \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{t}$, the following transformed points $\mathbf{t}$ also lie on the SVM decision boundary if they satisfy the equation, $\mathbf{w}^\top(\mu + \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{t}) + b = \mathbf{w}^\top\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{t} + \mathbf{w}^\top\mu + b = 0$. Thus, the hyperplane parameterized by $(\mathbf{w}, b)$ in the original space corresponds to the hyperplane parameterized by $(\mathbf{w}^\top\mathbf{\Sigma}^{\frac{1}{2}}, \mathbf{w}^\top\mu + b)$ in the transformed space. The margin of the SVM in the original space is $\frac{2}{\|\mathbf{w}\|}$. Thus the margin for the decision boundary under transformed data in terms of the parameters of the hyperplane

	Input : $(\mathbf{x}_i, y_i)_{i=1}^n$, Parameters: C, E.
Step 1	Solve formulation (4) using $(\mathbf{x}_i)_{i=1}^n$ with parameter E.
Step 2	Compute μ and $\mathbf{\Sigma}$ using $\mathbf{\Sigma} = \mathbf{A}^{-\frac{1}{2}}$ and $\mu = \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{b}$.
Step 3	Transform the data using $\mathbf{t}_i = \mathbf{\Sigma}^{-\frac{1}{2}}(\mathbf{x}_i - \mu)$ for all i.
Step 4	Train SVM formulation (2) with $(\mathbf{t}_i, y_i)_{i=1}^n$, parameter C, output $(\mathbf{w}, b)$

Table 1: The EKM Training Algorithm.

for the original data is given by $\frac{2}{\sqrt{\mathbf{w}^\top \mathbf{\Sigma} \mathbf{w}}}$. Hence we can directly maximize the margin for the transformed data without explicitly transforming each data point $\mathbf{x}_i \rightarrow \mathbf{t}_i$. Instead, we merely solve:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^\top \mathbf{\Sigma} \mathbf{w} + C\sum_{i=1}^n \xi_i \tag{5}$$
$$\text{s.t. } y_i(\mathbf{w}^\top(\mathbf{x}_i - \mu) + b) \geq 1 - \xi_i \qquad \xi_i \geq 0.$$

The advantage of the above formulation is that when the ellipsoid is flat (that is, $\mathbf{\Sigma}$ is not of full rank[3]) we avoid inverting it to transform the data. We use the above formulation when kernelizing the EKM method as well. When learning an SVM without considering the ellipsoidal structure, we simply minimize $\mathbf{w}^\top\mathbf{w}$. However, this is not what is minimized when the data is transformed to a sphere using the ellipsoidal structure. As shown from the analysis above, what we should minimize is $\mathbf{w}^\top\mathbf{\Sigma}\mathbf{w}$. By using $\mathbf{w}^\top\mathbf{w}$ in the objective in the original space, we only get a sub-optimal margin on the transformed data. This is another simple and intuitive motivation to use the proposed formulations independent of any learning theory arguments. Section 5.1 demonstrates this happening on a toy data set where an SVM obtains a suboptimal solution relative to transformed data even though it is linearly separable.

An important property of the EKM is that it is naturally rotation, translation and scale-invariant. The estimation of the bounding ellipsoid effectively compensates for any rotation, translation or scaling of the entire data prior to learning.In short, the EKM is affine invariant while the SVM is only rotation and translation invariant.

4 KERNELIZATION

Another reason for the success of SVMs is that they can be *kernelized*. Input data can be mapped using a function $\phi : \mathbb{R}^m \rightarrow \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space. However, $\mathcal{H}$ could be of very high dimension (possibly infinite), so it becomes very difficult to perform the mappings $\mathbf{x}_i \rightarrow \mathbf{t}_i$ explicitly. The solution to this problem is to express all computations in the previous section solely in terms of dot products (the so-called kernel trick) . A kernel function $k : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ implicitly (and more efficiently) computes the dot product of two vectors in Hilbert space. In other words, $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top\phi(\mathbf{x}_j)$. The matrix $K \in \mathbb{R}^{n\times n}$ whose entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ for all pairs of training data points is called the Gram matrix.

For simplicity, in this section we assume that the centroid of the data is at the origin (i.e., $\mu = 0$) and show how we can kernelize the problem of finding the minimum volume ellipsoid. For details on finding both the centroid and the covariance and ways to deal with rank deficcient kernel matrices, consult a forthcoming longer version of this paper. The steps in the following derivations closely resemble those in kPCA (Schölkopf et al., 1998).

We once again solve for the smallest bounding ellipsoid to obtain μ and $\mathbf{\Sigma}$ but replace any explicit evaluations of feature vectors with implicit kernel computations. We start with the following primal problem:

$$\min_{\mathbf{\Sigma}^{-1},\tau} -\ln|\mathbf{\Sigma}^{-1}| + E\sum_{i=1}^n \tau_i \tag{6}$$
$$\text{s.t. } \mathbf{x}_i^\top\mathbf{\Sigma}^{-1}\mathbf{x}_i \leq 1 + \tau_i, \quad \tau_i \geq 0 \quad \mathbf{\Sigma}^{-}1 \succeq 0.$$

Rewriting the constrained optimization as a Lagrangian, we have, $\mathcal{L}(\mathbf{\Sigma}^{-1}, \tau, \gamma, \alpha) = \ln|\mathbf{\Sigma}| + E\sum_{i=1}^n \tau_i + \sum_{i=1}^n \gamma_i(\mathbf{x}_i^\top\mathbf{\Sigma}^{-1}\mathbf{x}_i - 1 - \tau_i) - \sum_{i=1}^n \alpha_i\tau_i$, where $\alpha_i, \gamma_i \geq 0$ are Lagrange multipliers. Setting the partial derivative of $\mathcal{L}$ with respect to $\mathbf{\Sigma}^{-1}$ equal to zero, gives an expression for $\mathbf{\Sigma}$ at the optimum: $\mathbf{\Sigma} = \sum_{i=1}^n \gamma_i\mathbf{x}_i\mathbf{x}_i^\top$. The dual of (6) can be shown to be equivalent to (Kumar & Yıldırım, 2005):

$$\max_{\gamma} \ln|\sum_{i=1}^n \gamma_i\mathbf{x}_i\mathbf{x}_i^\top| - \sum_{i=1}^n \gamma_i \quad \text{s.t } 0 \leq \gamma_i \leq E. \tag{7}$$

Since $\mathbf{\Sigma} = \sum_{i=1}^n \gamma_i\mathbf{x}_i\mathbf{x}_i^\top$, the dual avoids the problem of low rank. Even if $\mathbf{\Sigma}$ is of low rank, by adding an ϵI to $\mathbf{\Sigma}$ in the objective, we can still estimate a low rank $\mathbf{\Sigma}$ using the dual. Primal (6) would have had problems in this situation as it estimates $\mathbf{\Sigma}^{-1}$. Now suppose that $\mathbf{C}$ is the equivalent matrix of $\mathbf{\Sigma}$ in the Hilbert space, we can express $\mathbf{C}$

[3]We later show how $\mathbf{\Sigma}$ can be estimated when it is not of full rank.

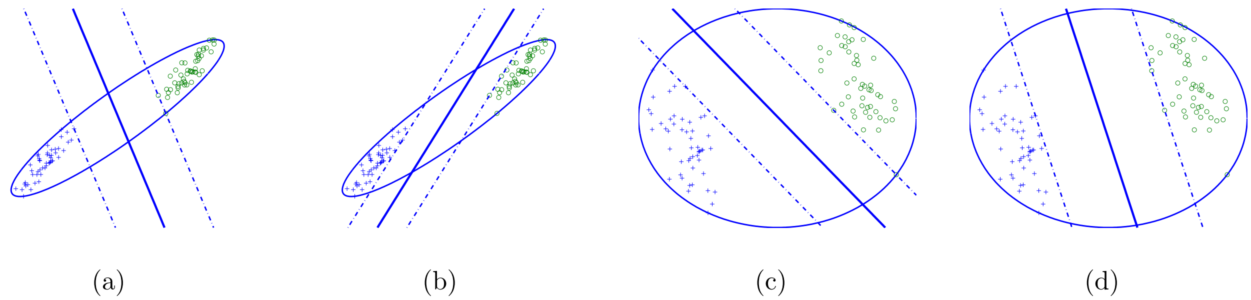

Figure 2: (a) Classical SVM solution on the data, (b) EKM solution on the data, (c) Classical SVM solution from the first plot after making the data spherical and (d) EKM solution from the second plot after making the data spherical.

as $\mathbf{C} = \sum_{i=1}^{n} \gamma_i \phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\top$. In higher dimensions, this matrix typically corresponds to a flat ellipsoid as the dimensionality of the Hilbert space could be much higher than the number of training samples. To find the minimum enclosing ellipsoid, the product of nonzero eigenvalues of $\mathbf{C}$ is to be maximized in the dual (7) although the primal (6) does the exact opposite. Consider the eigenvalue equation,

$$\mathbf{C}\mathbf{V} = \lambda \mathbf{V} \tag{8}$$

in the Hilbert space where $\mathbf{V}$ denotes an eigenvector of $\mathbf{C}$ with corresponding eigenvalue λ. Eigenvectors can be represented as $\mathbf{V} = \sum_{i=1}^{n} \nu_i \phi(\mathbf{x}_i)$ since the eigenvectors of $\mathbf{C}$ are in the span of the data (Schölkopf et al., 1998). Plugging this formula into (8) gives,

$$\sum_{i=1}^{n} \gamma_i \phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\top \sum_{j=1}^{n} \nu_j \phi(\mathbf{x}_j) = \lambda \sum_{j=1}^{n} \nu_j \phi(\mathbf{x}_j).$$

However, we only have n data points from which we can solve the above eigensystem. Multiply both sides of the above equality from the left with $\phi(\mathbf{x}_l)^\top$ to obtain

$$\sum_{i=1}^{n} \gamma_i k(\mathbf{x}_l, \mathbf{x}_i) \sum_{j=1}^{n} \nu_j k(\mathbf{x}_i, \mathbf{x}_j) = \lambda \sum_{j=1}^{n} \nu_j k(\mathbf{x}_l, \mathbf{x}_j)$$

for all $1 \leq l \leq n$. The above system can be expressed in matrix form with $\mathbf{\Gamma}$ as a diagonal matrix with γ_i as the diagonal entries:

$$K \mathbf{\Gamma} K \nu = \lambda K \nu. \tag{9}$$

The system (9) is an instance of the generalized eigenvalue problem. The above system can be solved via the following eigenvalue problem if K is invertible:

$$K^{\frac{1}{2}} \mathbf{\Gamma} K^{\frac{1}{2}} \mathbf{u} = \lambda \mathbf{u}. \tag{10}$$

The dual (7) tries to maximize the determinant of the covariance matrix. Since the above eigensystem (10) has the same eigenvalues as $\mathbf{C}$, the determinant of $K^{\frac{1}{2}} \mathbf{\Gamma} K^{\frac{1}{2}}$ can as well be maximized in Hilbert space to achieve the same objective. This motivates us to solve the following problem:

$$\max_{\tau,\gamma} \ln |K^{\frac{1}{2}} \mathbf{\Gamma} K^{\frac{1}{2}}| - \sum_{i=1}^{n} \gamma_i \quad \text{s.t. } 0 \leq \gamma_i \leq E \tag{11}$$

We can now express the term $\mathbf{w}^\top \mathbf{C} \mathbf{w}$ (the term in the objective function in Equation (5)) purely in terms of kernel functions by using $\mathbf{C} = \sum_{i=1}^{n} \gamma_i \phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^T$ and $\mathbf{w} = \sum_{i=1}^{n} \kappa_i \phi(\mathbf{x}_i)$ due to the fact the $\mathbf{w}$ is in the span of the data. The constraints in (5) are also straightforward to kernelize yielding a fully kernelized EKM algorithm. By directly solving (5) with the $\mathbf{\Sigma}$ estimated in the dual optimization of Equation (7), we also avoid ever having to invert the matrix.

5 EXPERIMENTS

5.1 ILLUSTRATION WITH A TOY DATASET

Fifty points were sampled from Gaussians $\mathcal{N}(\mu_1, \mathbf{\Sigma})$ and Gaussian $\mathcal{N}(\mu_2, \mathbf{\Sigma})$ as the patterns of each class, where the means and the covariances were set to:

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mu_2 = \begin{bmatrix} 50 \\ 25 \end{bmatrix}, \mathbf{\Sigma} = \begin{bmatrix} 17 & 15 \\ 15 & 17 \end{bmatrix}.$$

The data was then scaled so that each feature ranged between $[0, 1]$. This is the the so-called unit box constraint which is the standard way of normalizing data prior to SVM learning. The classical separable SVM was solved for this dataset and is shown in Figure 2(a). The bounding ellipsoid is shown as well but note that the data is not transformed in any manner. Figure

	Setup 1			Setup 2		
	Classical	Gaussian	Ellipsoidal	Classical	Gaussian	Ellipsoidal
Heart	$\mathbf{0.833 \pm 0.008}$	0.762 ± 0.004	$\mathbf{0.833 \pm 0.006}$	0.841 ± 0.009	0.829 ± 0.010	$\mathbf{0.851 \pm 0.007}$
Pima	0.801 ± 0.001	0.741 ± 0.001	$\mathbf{0.802 \pm 0.001}$	0.745 ± 0.002	$\mathbf{0.749 \pm 0.003}$	0.746 ± 0.002
Ion	0.842 ± 0.001	0.838 ± 0.007	$\mathbf{0.857 \pm 0.004}$	0.842 ± 0.004	0.854 ± 0.004	$\mathbf{0.862 \pm 0.002}$
Pen	0.997 ± 0.000	0.998 ± 0.000	$\mathbf{1.000 \pm 0.000}$	$\mathbf{1.000 \pm 0.000}$	0.998 ± 0.000	$\mathbf{1.000 \pm 0.000}$
Iris	$\mathbf{0.953 \pm 0.005}$	0.927 ± 0.009	$\mathbf{0.953 \pm 0.005}$	0.946 ± 0.001	$\mathbf{0.973 \pm 0.004}$	$\mathbf{0.973 \pm 0.004}$
Bupa	0.702 ± 0.004	0.664 ± 0.005	$\mathbf{0.705 \pm 0.003}$	0.688 ± 0.003	0.691 ± 0.005	$\mathbf{0.702 \pm 0.005}$
Seg	0.824 ± 0.002	0.810 ± 0.005	$\mathbf{0.828 \pm 0.003}$	0.852 ± 0.009	0.857 ± 0.007	$\mathbf{0.861 \pm 0.004}$
Sonar				$\mathbf{0.738 \pm 0.004}$	0.723 ± 0.007	0.733 ± 0.006

Table 2: Test accuracies under Setup 1 and Setup 2 on several datasets.

2(b) shows the solution obtained by the EKM , by first transforming the data to a zero centroid unit radius circle, solving an SVM and then mapping back the decision boundary obtained in the transformed space back to the original data space. Across Figure 2(a) and (b), it appears that the classical SVM has a higher margin than the EKM. However, when the data is scaled evenly by using the estimated ellipsoid to map to a unit sphere, the solution obtained by the classical SVM in the Figure 2(c) is clearly worse in terms of margin when compared to the EKM in Figure 2(d).

This toy example clearly shows that individually scaling features so that they are all on the same scale (the so-called box constraint) and then solving with an SVM does not necessarily give the best solution in terms of margin. This example shows that even when the data is linearly separable, unless the features are independent, the covariance structure of a bounding ellipsoid on the data must be exploited to get the optimal linear classifier. When the features are independent, the EKM gives the same solution as the SVM, since transforming according to the structure of the ellipsoid in this case corresponds to scaling individual features independently.

5.2 REAL WORLD DATASETS

We compared the EKM with the classical SVM on a large array of UCI datasets (D.J. Newman & Merz, 1998). The results that are presented here were produced by *fully automatic* testing procedures. Both the EKM and the classical SVM used the *same splits* for the training, cross validation and testing subsets in each experiment. We present two kinds of experimental setups.

Setup 1 Each UCI data set is randomly split into two parts with a 9:1 ratio. The smaller portion was held out for testing purposes. The bigger portion of the split was further randomly divided according to a 9:1 ratio ten times for cross validation. For the classical SVM, different values of C were attempted. The SVM was trained on the cross validation training data and evaluated on the cross validation test data. Similarly, cross validation training data was used to learn an ellipsoid for different values of the parameter E in Equation (4). We varied E to explore ellipsoids that enclosed from 65% of the training data to 100% of the training data. For each estimated ellipsoid, data is transformed into a sphere for subsequent linear decision learning. With different values of C, a classifier was learned on the transformed data. The estimated hyperplanes were evaluated using the cross validation data after transformation. From the cross validation accuracies average over ten iterations, the parameters that gave the highest accuracies were chosen for the EKM and the SVM. These parameters were used to train on the original 90% partition and to test on the originally held out test data. The entire process was repeated ten times to get the average accuracy reported here. Similarly, instead of estimating $\mathbf{\Sigma}$ from the SDP we estimated the Gaussian mean for the data and used that to transform the data. The goal was to compare the Gaussian estimate of the transformation with the SDP estimate. Test results are shown in Table 2. Results reported are average test accuracy and the variance over ten randomized iterations as mentioned above. Clearly, in most of the cases, the EKM outperforms the other two methods.

Setup 2 Although SDPs are solvable in polynomial time, in practice they are still much slower than the QP required to solve an SVM. We therefore consider a different experimental setup which avoids using the SDP too frequently. In this setup, the *entire dataset* is first used to learn different ellipsoids for different values of E. The μ and $\mathbf{\Sigma}$ values obtained were stored. From then on, the experiment methodology remained exactly the same as in Setup 1 except that for a given parameter E we simply used the stored μ and $\mathbf{\Sigma}$ for the entire dataset rather than recomputing the bounding ellipsoid for each fold. The results in this case though, are on different partitions of the data as the two experiments were conducted independently. It can be

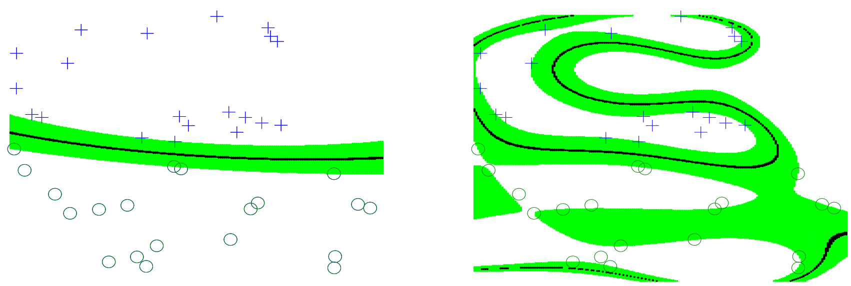

Figure 3: Kernelized SVM (left) and EKM (right) solutions.

noted that in Setup 1, we did not report results on the Sonar dataset since it was too computationally expensive to learn the bounding ellipsoid more than hundred times for each cross validation. It is reasonable to expect that the structure of the ellipsoid would remain almost the same as long as we take a significant portion of the data. Table 2 also shows equally promising results in this Setup.

Kernel Version On a small randomly generated dataset, we estimated a classical SVM as well as an EKM with a Gaussian kernel with a fixed sigma scale parameter. The EKM was estimated to bound all the training examples with an ellipsoid in Hilbert space. The resulting decision boundaries are shown in Figure 3. It can be seen that the solution can be drastically different in the kernelized version even on small datasets as the ellipsoid in very high dimensions can be quite complex.

6 CONCLUSIONS

A new technique to exploit the ellispoidal structure of the data was proposed which estimates the minimum volume bounding ellipsoid using SDP and then uses it to (implicitly) remap the data to a unit sphere prior to running an SVM. All computations were also kernelized to scale to general Hilbert spaces. Not only is this a useful preprocessing step of the data, it helps produce better VC dimension generalization arguments for the resulting linear classifier. Furthermore, experimental evidence for this classifier are promising. It is anticipated that the kernelized version of the technique would be even more successful. Future directions include exploring clustering, transduction, feature selection, and so on while exploiting the ellipsoidal structure of the data.

[3]This work was funded in part by NSF grant IIS-0347499.

References

Boyd, S., & Vandenberghe, L. (2003). *Convex optimization*. Cambridge University Press.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*, 121–167.

D.J. Newman, S. Hettich, C. B., & Merz, C. (1998). UCI repository of machine learning databases.

Herbrich, R., Graepel, T., & Campbell, C. (2001). Bayes point machines. *Journal of Machine Learning Research*, *1*, 245–279.

Kumar, P., & Yıldırım, A. (2005). Minimum volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, *126*, 1–21.

Lanckriet, G. R. G., El Ghaoui, L., Bhattacharyya, C., & Jordan, M. I. (2002). Minimax probability machine. *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press.

Schölkopf, B., Smola, A., & Müller, K. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Compuation*, *10*.

Shivaswamy, P. K., Bhattacharyya, C., & Smola, A. (2006). Second order cone programming approaches for handling missing and uncertain data. *Journal of Machine Learning Research*, *7*, 1283–1314.

Shivaswamy, P. K., & Jebara, T. (2006). Permutation invaraint support vector machines. *International Conference on Machine Learning*.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, *1*, 211–244.

Vapnik, V. (1995). *The nature of statistical learning*. New York, NY: Springer.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., & Vapnik, V. (2000). Feature selection for SVMs. *NIPS* (pp. 668–674).

Zhu, J., Rosset, S., Hastie, T., & Tibshirani, R. (2004). 1-norm support vector machines. In *Advances in neural information processing systems 16*. Cambridge, MA: MIT Press.