Robust Formulations for Handling Uncertainty in Kernel Matrices

Sahely Bhadra[1] SAHELY@CSA.IISC.ERNET.IN
Sourangshu Bhattacharya[2] SOURANGB@YAHOO-INC.COM
Chiranjib Bhattacharyya[1] CHIRU@CSA.IISC.ERNET.IN
Aharon Ben-tal[3] BENTAL@IE.TECHNION.AC.IL
[1] Department of CSA, Indian Institute of Science, INDIA
[2] Yahoo! Labs, Bangalore, INDIA
[3] Faculty of Industrial Engg. and Management, Technion, Haifa, ISRAEL

Abstract

We study the problem of uncertainty in the entries of the Kernel matrix, arising in SVM formulation. Using Chance Constraint Programming and a novel large deviation inequality we derive a formulation which is robust to such noise. The resulting formulation applies when the noise is Gaussian, or has finite support. The formulation in general is non-convex, but in several cases of interest it reduces to a convex program. The problem of uncertainty in kernel matrix is motivated from the real world problem of classifying proteins when the structures are provided with some uncertainty. The formulation derived here naturally incorporates such uncertainty in a principled manner leading to significant improvements over the state of the art.

1. Introduction

Given a dataset $D = \{(x_i, y_i) | i = 1, \ldots, n\}$ the SVM dual formulation (Vapnik, 1998) can be written as:

$$\max_{\alpha \in S_n, t} \quad \alpha^\top e - \tfrac{1}{2}t \qquad \text{s.t.} \quad \alpha^\top Y \mathbf{K} Y \alpha \leq t \tag{1}$$

where $S_n = \{\alpha | 0 \leq \alpha_i \leq C, \; \sum_{i=1}^n \alpha_i y_i = 0\}$ and $Y = diag(y_i)$. The kernel matrix, $\mathbf{K}$, is a $n \times n$ matrix, where K_{ij} can be understood as dot product between implicitly defined feature map over examples $x_i, x_j \in \mathcal{X}$. In this paper, we study the problem of designing robust classifiers when the entries of $\mathbf{K}$ are uncertain. To the best of our knowledge there is no such study of this important problem in the existing literature.

We treat the impact of uncertainty in individual examples as an additive uncertainty Z. We consider the following chance constraint setting:

$$\max_{t, \alpha \in S_n} \alpha^\top e - \frac{1}{2}t \tag{2}$$

$$\text{s.t. } \mathit{Prob}\left(\alpha^\top Y(\overline{\mathbf{K}} + Z)Y\alpha \leq t\right) \geq 1 - \epsilon \tag{3}$$

where $\epsilon < 0.5$. In this setting the inequality (3), ensures that the event $\alpha^\top Y(\overline{\mathbf{K}} + Z)Y\alpha \leq t$ holds with high probability $(1 - \epsilon)$ for any instantiation of the random variate Z. It is assumed that $\overline{\mathbf{K}}$ is a specified kernel matrix, and is symmetric, positive semidefinite. The random matrix $\overline{\mathbf{K}} + Z$ is not necessarily psd and symmetric.

Optimization problems involving chance constraints often turn out to be NP-hard and is an active area of study (Nemirovski & Shapiro, 2006; Ben-Tal & Nemirovski, 2007). Chance constraints were previously used in handling uncertainty in the context of linear classifiers (Ghaoui et al., 2003; Bhattacharyya et al., 2004; Shivaswamy et al., 2006). Assuming a full knowledge of Covariance structure of the data uncertainty and using Chebychev inequality they (Bhattacharyya et al., 2004; Shivaswamy et al., 2006) formulated the problem as a Second Order Cone Program(SOCP). Instead of using a full covariance matrix, which is difficult to estimate, an alternative based only on the support information was proposed in (Ghaoui et al., 2003). However the application of these methods to (3) is not straightforward and requires further investigation.

The problem studied here is motivated from the problem of classifying protein structures where kernel methods have been highly successful (Qiu et al., 2007; Bhattacharya et al., 2007). They designed kernels are

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010. Copyright 2010 by the author(s)/owner(s).

based on similarity scores, like Root mean square deviation(RMSD) obtained from structural alignment algorithms e.g. DALI(Holm & Sander, 1996). Existing methods assume that protein structures are determined exactly, without any uncertainty. However in reality, coordinates of atoms of protein structures are determined with uncertainty, governed by the *resolution* of X-ray diffraction experiment. [1] When the uncertainty becomes comparable to RMSD then the similarity scores becomes suspect. For example, consider the two SCOP domains **d1biaa1** and **d1repc1** belonging to different families, but same superfamily: **Winged helix DNA-binding domain**. The structures for these have been determined at resolutions $2.3\mathring{A}$ and $2.6\mathring{A}$ and Dali gives a structural alignment with RMSD 2.2 between these domains. So, the uncertainty in the kernel value for these structures is higher than the scores themselves, which could be detrimental to discriminating between the two classes.

We study the problem of solving (2) assuming that Z_{ij} are centered and independent. We study the two cases namely a.) Z_{ij} is Gaussian, and b.) Z_{ij} has finite support. For case of Gaussian distribution we derive a novel formulation which can be interpreted as a robust version of SVM. A major contribution of this paper is a novel large deviation inequality which applies to (3) which applies to the finite support case. Using this inequality, we derive a formulation similar to Gaussian case for solving (2). The optimization problem in a general setting turns out to be instance of non-convex program. However under certain assumptions the problem can be solved as a convex conic quadratic. It is interesting to note that under i.i.d assumption the formulation reduces to a SVM with a modified kernel function. Extensive experimentation on synthetic datasets show that current formulation is more robust than standard SVM. The formulation developed here incorporates resolution information available in protein structures in a principled way yielding to substantially better design of classifiers. Experimental results show that this resulted in significant improvements in classification accuracy over existing state of the art methods and their obvious extensions. Also, as expected, the new classifiers are more robust than existing ones.

The paper is organized as follows: The main contributions are described in section 2. Section 3 presents algorithms to solve such problems, and section 4 discusses metrics for measuring the performance of resultant classifiers. Section 5 reports experimental results.

2. Robust formulations for handling uncertainty in Kernel matrices

In this section we study (3) when Z_{ij} is independent with zero mean. We begin the study by assuming that the entries are Gaussian distributed and in subsection 2.2 we consider the more general case when the entries have finite support. To this end we derive a large deviation inequality on the inner product of a matrix with Z which is later used in (3).

Notation: We denote the Hadamard product of $A, B \in \mathbb{R}^{n\times n}$ by $(A * B)$ a $n \times n$ matrix with entries $(A*B)_{ij} = a_{ij}b_{ij}$. The frobenius norm of $C \in \mathbb{R}^{n\times n}$, is given by $\|C\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^n c_{ij}^2}$. Trace of square matrix A is denoted by $Tr(A)$.

2.1. Kernel matrix is Gaussian distributed

We begin with the following Lemma.

Lemma 1. *Let Z be an $n \times n$ random matrix with entries Z_{ij} independently distributed as $Z_{ij} \sim N(0, \sigma_{ij}^2)$. For every $W, A \in \mathbb{R}^{n\times n}$ the constraint*

$$Prob(Tr\{(Z+W)A\} \geq t) \leq \epsilon \tag{4}$$

is satisfied if the following holds.

$$Tr(WA) \leq t + \Phi^{-1}(\epsilon)\|\mathbf{\Sigma} * A\|_F \tag{5}$$

where $\mathbf{\Sigma}_{ij} = \sigma_{ij}$.

Proof. Observe that $Tr(ZA) \sim N(0, \|\mathbf{\Sigma} * A\|_F^2)$. This is true because $Tr(ZA)$ can be written as a weighted sum of independently distributed normal random variables. Using a standardized normal random variate, $U \sim N(0,1)$, one can write $Tr(ZA) = U\|\mathbf{\Sigma} * A\|_F$. Using the CDF of U, defined as $Prob(U \leq u) = \Phi(u) = \int_{-\infty}^{u} \frac{1}{\sqrt{2\pi}} e^{\frac{-s^2}{2}} ds$ we get,

$$\begin{aligned} &Prob\,(Tr\{(Z+W)A\} \geq t) \\ &\quad = Prob\,(U\|\mathbf{\Sigma} * A\|_F \geq t - Tr(WA)) \\ &\quad\quad = \Phi(-u) \text{ where } u = \frac{t - Tr(WA)}{\|\mathbf{\Sigma} * A\|_F} \end{aligned} \tag{6}$$

The second equality follows because U is standard normal. This derivation shows that equation (4) can be restated as $\Phi(-u) \leq \epsilon$ Noting that Φ is an increasing function of its argument one obtains $u \geq -\Phi^{-1}(\epsilon)$ where Φ^{-1} is the inverse function of Φ. Substituting the value of u completes the proof of the theorem. □

A direct application of the above Lemma leads to the following theorem, which is the first result of the paper.

[1] http://www.rcsb.org/pdb/

Theorem 1. *Let Z be an $n \times n$ matrix whose entries are independently distributed with entries $Z_{ij} \sim N(0, \sigma_{ij}^2)$. Let $\mathbf{K} = \overline{\mathbf{K}} + Z$ be a noise corrupted matrix where $\overline{\mathbf{K}}$ is $n \times n$ known kernel matrix. For such a $\mathbf{K}$ the constraint (3) in the formulation (2) is satisfied if the following holds.*

$$\sum_{ij} y_i y_j \alpha_i \alpha_j \overline{K}_{ij} - \Phi^{-1}(\epsilon) \|\mathbf{\Sigma} * (\alpha\alpha^\top)\|_F \leq t \tag{7}$$

Proof. Substituting $W = \overline{\mathbf{K}}$ and $A_{ij} = \alpha_i y_i \alpha_j y_j$ in Lemma 1 proves the theorem □

This theorem points to a deterministic equivalent to the problem stated in (2). Note that for cases of interest $\epsilon < 0.5$, which implies that $\Phi^{-1}(\epsilon) < 0$. In a later section we will discuss algorithmic approaches for solving such programs.

2.2. Uncertainty with finite support

In this section we study the case where uncertainty in the kernel entries has finite support. We state and prove a novel large deviation inequality, and exploit it to obtain a deterministic constraint similar to the one obtained in the Gaussian case (see Theorem 1).

We begin by proving a novel large deviation inequality,

Theorem 2. *Let Z be a $n \times n$ random matrix with entries Z_{ij} independently distributed along with finite support, $Prob(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1$, and $E(Z_{ij}) = 0$. For every $V \in \mathbb{R}^{n \times n}$, and $u > 0$*

$$Prob(Tr(ZV) \geq u) \leq exp\left\{-\frac{1}{2}\frac{u^2}{\|\beta' * V\|_F^2}\right\} \tag{8}$$

$l_{ij} = \frac{b_{ij} - a_{ij}}{2}$, $c_{ij} = \frac{b_{ij} + a_{ij}}{2}$, $\hat{\mu}_{ij} = -\frac{c_{ij}}{l_{ij}}$, $\beta_{ij} = l_{ij}^2 \gamma_{ij}^2$, $\beta'_{ij} = \beta_{ij}^{\frac{1}{2}}$

$$\text{and } \gamma_{ij} = \min\{\sigma \geq 0 \mid \tfrac{\sigma^2}{2} z^2 + \hat{\mu}_{ij} z - \log(\cosh(z) + \hat{\mu}_{ij}\sinh(z)) \geq 0,\ \forall\ z \in \mathbb{R}.\} \tag{9}$$

Proof. As a consequence of Markov inequality and independence of entries of Z, the following holds $\forall\ s \geq 0$.

$$Prob\,(Tr(ZV) \geq u) \leq e^{\{-su\}} \prod_{i,j} E\left(e^{\{sV_{ij}Z_{ij}\}}\right) \tag{10}$$

Exploiting the convexity of the function, e^x, one can upperbound the moment generating function of Z_{ij}. More importantly for any $r \in \mathbb{R}$ the following is true.

$$E(e^{rZ_{ij}}) \leq \frac{b_{ij}}{b_{ij} - a_{ij}} e^{ra_{ij}} - \frac{a_{ij}}{b_{ij} - a_{ij}} e^{rb_{ij}} = e^{rc_{ij} + f_{ij}(rl_{ij})} \tag{11}$$

where, $f_{ij}(\omega) = \log(cosh(\omega) + \hat{\mu}_{ij} sinh(\omega))$. By using Taylor expansion around 0 we obtain the following bound $f_{ij}(\omega) \leq \frac{1}{2}\omega^2 + \hat{\mu}_{ij}\omega$. which could be further tightened by considering $f_{ij}(\omega) \leq \frac{1}{2}\gamma_{ij}^2\omega^2 + \hat{\mu}_{ij}\omega$ where γ_{ij} is given in (9).

Substituting $\omega = rl_{ij}$ the bound (11) can be written as $logE(e^{rZ_{ij}}) \leq r(c_{ij} + l_{ij}\hat{\mu}_{ij}) + \frac{1}{2}r^2\gamma_{ij}^2 l_{ij}^2 = \frac{1}{2}r^2\gamma_{ij}^2 l_{ij}^2$. This bound holds for all r, and for the problem at hand by putting $r = sV_{ij}$ we get $E(e^{sV_{ij}Z_{ij}}) \leq \frac{1}{2}s^2 V_{ij}^2 \gamma_{ij}^2 l_{ij}^2$. In light of this result the relation (10) can now be upperbounded as follows

$$\log\left[Prob\,(Tr(ZV) \geq u)\right] \leq \min_{s \geq 0} - su + \frac{s^2}{2}\|\beta' * V\|_F^2 = -\frac{1}{2}\frac{u^2}{\|\beta' * V\|_F^2}$$

The proof is completed by noting that minimization is attained at $s^* = \frac{u}{\|\beta' * V\|_F}$, obtained by solving an univariate quadratic optimization problem. □

The values γ_{ij} (9) can be calculated numerically and is not presented here because of space constraints.

The inequality is of independent interest, but we do not study it further. Instead we apply the inequality to the problem at hand. Consider the theorem

Theorem 3. *Let Z be a $n \times n$ matrix whose entries are independently distributed, given that $P(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1$ and $E(Z_{ij}) = 0$. Let $\mathbf{K} = \overline{\mathbf{K}} + Z$ be a noise corrupted matrix where $\overline{\mathbf{K}}$ is $n \times n$ specified kernel matrix. For such a $\mathbf{K}$ the constraint (3) in formulation (2) is satisfied if the following holds.*

$$\sum_{ij} y_i y_j \alpha_i \alpha_j \overline{K}_{ij} + \sqrt{2\log(1/\epsilon)}\sqrt{\sum_{ij} \beta_{ij}\alpha_i^2\alpha_j^2} \leq t \tag{12}$$

where β_{ij} is defined as in Theorem 2.

Proof. We begin by noting that the constraint (3) can be stated as

$$Prob(Tr\{(Z + \overline{\mathbf{K}})V\} \geq t) \leq \epsilon \tag{13}$$

where $V_{ij} = \alpha_i y_i \alpha_j y_j$. A necessary condition for satisfying the above inequality can be obtained by exploiting the large deviation inequality stated in Theorem 2. In particular a direct application of the bound yields the following constraint

$$\sqrt{2\log(1/\epsilon)}\|\beta' * V\|_F \leq t - Tr(\overline{\mathbf{K}}V) \tag{14}$$

Substituting V in the above equation proves the theorem □

2.3. A deterministic optimization problem

In light of the Theorem 1 and Theorem 3 one can motivate the following deterministic counterpart of (2).

$$\boxed{\begin{aligned} &\min_{t,\alpha\in S_n} \frac{1}{2}t - \sum_i \alpha_i \\ &\text{s.t. } \sum_{ij} y_i y_j \alpha_i \alpha_j \overline{K}_{ij} + \kappa\sqrt{\sum_{ij}\beta_{ij}\alpha_i^2\alpha_j^2} \leq t \end{aligned}} \qquad (15)$$

where, $\kappa = \sqrt{2\log(1/\epsilon)}$ when uncertainty has finite support. This formulation is robust to uncertainty in the kernel entries and will be referred as Robust SVM (**RSVM**). In case of Gaussian uncertainty, $\kappa = -\Phi^{-1}(\epsilon)$ and $\beta_{ij} = \sigma_{ij}^2$ and this formulation will be referred as Robust SVM for Gaussian distribution ($\mathbf{RSVM^{(g)}}$).

3. Algorithms for solving the robust formulation

In this section we consider algorithms for solving (15). In general these problems are instances of non-convex programs. Here we observe that in several cases of interest the problems can be reduced to convex conic quadratic programs.

Case 1 - β is rank one: For this case the formulation is equivalent of solving SVM. It is interesting to note that this case arises when the uncertainty is independent and identically distributed (i.i.d.).

Theorem 4. *Let β be a rank one matrix, i.e. $\beta_{ij} = \rho_i\rho_j$ where ρ is a vector with non-negative components. The formulation (15) is equivalent to a SVM with kernel $\overline{\mathbf{K}} + \kappa diag(\rho)$.*

Proof. Noting that, $\sqrt{\sum_{ij}\beta_{ij}\alpha_i^2\alpha_j^2} = \sum_i \rho_i\alpha_i^2$ and eliminating t the result follows. By design $\beta_{ij} \geq 0$ which implies that $\rho_i > 0$ (Minc, 1970) □

We will denote the corresponding formulation by $\mathbf{RSVM_{QP}}$ ($\mathbf{RSVM_{QP}^{(g)}}$ for Gaussian uncertainty).

As a corollary to the previous theorem one can prove

Theorem 5. *Let entries of the random matrix Z be i.i.d. and $\mathbf{K} = \overline{\mathbf{K}} + Z$ In such a case formulation (15) is equivalent of solving a SVM with Kernel matrix* $\mathbf{K} = \overline{\mathbf{K}} + \sigma\kappa\mathbf{I}$

Proof. Note that for i.i.d. case $\rho_i = \sigma$ in the previous theorem and the proof follows. □

Case 2 - β is psd: If the matrix β is positive semidefinite then the formulation can be posed as SOCP. To this end consider the following theorem.

Theorem 6. *If both $\overline{\mathbf{K}}, \beta$ are symmetric psd matrices then the following formulation is equivalent to formulation (15).*

$$\begin{array}{cc} \min_{t,\nu,t',\alpha\in S_n} & \frac{1}{2}t - \sum_i \alpha_i \\ s.t. & \kappa\|\beta^{\frac{1}{2}}\nu\| \leq t - t' \\ & \|Y(\overline{\mathbf{K}})^{\frac{1}{2}}\alpha\|_2^2 \leq t' \\ & \alpha_i^2 \leq \nu_i \end{array} \qquad (16)$$

Proof. As $\beta, \overline{\mathbf{K}}$ are psd matrices their matrix square roots, $\beta^{\frac{1}{2}}, \overline{\mathbf{K}}^{\frac{1}{2}}$, exist. At optimality $\alpha_i^2 = \nu_i$ and the theorem follows. □

Note that this formulation is second order cone representable and hence can be solved as an Second Order Cone Program (SOCP). This will be denoted by $\mathbf{RSVM_{SOCP}}$ ($\mathbf{RSVM_{SOCP}^{(g)}}$ for Gaussian uncertainty).

Case 3- The case of general β: The case of general β ($\beta_{ij} \geq 0$), is an instance of non-convex program. We do not study this setting in detail here but propose a general descent algorithm to solve this. In particular we use a modified Newton Method (Luenberger & Ye, 2008) with square penalty function leading to the following unconstrained approximation of (15).

$$\min_{\alpha} L(\alpha) \left(= f(\alpha) + \frac{P}{2}[(\sum_i \alpha_i y_i)^2 + \sum_{i:\alpha_i > C}(\alpha_i - C)^2 + \sum_{i:\alpha_i<0}\alpha_i^2] \right) \qquad (17)$$

where,

$$f(\alpha) = \tfrac{1}{2}\textstyle\sum_{ij} y_i y_j \alpha_i \alpha_j \overline{\mathbf{K}}_{ij} + \tfrac{1}{2}\kappa\sqrt{\sum_{ij}\beta_{ij}\alpha_i^2\alpha_j^2} - \sum_i \alpha_i$$

and P is a user defined cost for penalty function.

We minimize L with the Quasi Newton type method with DFP updation as follows $\alpha^{t+1} = \alpha^t - \eta^t H^t \nabla_\alpha L$ Where, η^t is step size, H^t is approximate inverse of Hessian of L obtained by DFP procedure, and $\nabla_\alpha L$ is the gradient of L w.r.t α.

This will be referred as $\mathbf{RSVM_{QN}}$ ($\mathbf{RSVM_{QN}^{(g)}}$ for Gaussian uncertainty). The algorithm suffers from the problem of local minimum. To alleviate the problem we have used multiple starting points.

The decision function for the classifier can be expressed as $f(w,b) = sign(\sum_{i\in SV} y_i\alpha_i\overline{\mathbf{K}}_{i.} + b)$, where SV is an index set of support vectors. In order to get robust performance and to reduce the effect of uncertainty in kernel, the bias b can be computed as $b = \frac{1}{\#\ SV}[\sum_{j\in SV} y_j - \sum_{i,j\in SV} y_i\alpha_i\overline{\mathbf{K}}_{ij}]$.

4. Error metrics

For each test data, P_t, for all the training data points P_i, we have the mean $\bar{K}_{ti}$ of the kernel entry K_{ti}, and either variance σ^2_{ti} (in Gaussian case) or range $[a_{ti}, b_{ti}]$ (finite support case) for the uncertainty. In both cases, we test by generating multiple samples $K^s_{ti}, s = 1, \ldots, n_s$, for each kernel entry K_{ti}, where n_s is the number of samples.

For each of these test values, $\kappa = 0$ (15) and $\bar{K}_{ti} = K^s_{ti}$. Hence, the decision function obtained from equation (15) can be written as $f^s_t = \text{sign}(\sum_{j \in SV} \alpha_j y_j K^s_{tj} + b)$. One way to assign a label y^{pr}_t to P_t is by majority votes from $f^s_t,\ s = 1, \ldots, n_s$. So, $y^{pr}_t = \text{sign}(\sum_{s=1}^{n_s} f^s_t)$.

Let $\mathcal{D} = \{P_t, t = 1, \ldots, n\}$ be a test dataset which is tested according to the majority vote scheme. The overall classification error can be calculated as:

$$\mathbf{MajErr}(ME) : \frac{\sum_{t=1}^{n} \mathbf{1}_{(y^{pr}_t \neq y_t)}}{n} \times 100 \qquad (18)$$

where, y_t is the true label for P_t.

While the above scheme is intuitive for labeling an uncertain data point, a robust classifier is expected to classify all the n_s samples generated for each uncertain data point P_t correctly. To capture this notion of robustness, we propose another error measure (**RobustErr**) which counts the fraction of data points in $\mathcal{D}$ for which all the samples are classified correctly.

$$\mathbf{RobustErr}(RE) : \frac{\sum_{t=1}^{n} \mathbf{1}_{(\exists s | f^s_t \neq y_t)}}{n} \times 100 \qquad (19)$$

We can also treat each of the samples generated from uncertain data points as individual data points, and define a standard classification error (**NomErr**), as:

$$\mathbf{NomErr}(NE) : \frac{\sum_{st} \mathbf{1}_{(f^s_t \neq y_t)}}{n n_s} \times 100 \qquad (20)$$

In the following section, we report experimental results for the techniques developed here and state of the art methods with respect to the above mentioned metrics.

5. Experiments

This section presents experimental results to compare the proposed **RSVM** (15), $\mathbf{RSVM^{(g)}}$, and Nominal-SVM (SVM with specified kernel) in terms of accuracy and robustness on the task of binary classification. All the three algorithms for RSVM: $RSVM_{QP}$, $RSVM_{SOCP}$ and $RSVM_{QN}$, were implemented in Matlab with the help of a standard QP solver and Sedumi[2]. We report results for both synthetic data and resolution aware protein structure classification problem. The results demonstrate that the proposed formulations outperform state of the art techniques with respect to both traditional error measures and new metrics defined in section 4.

5.1. Experiment with Synthetic data

We performed a thorough experimental analysis of the proposed formulations measuring its generalization performance, robustness, and performance of the bounds. For this, we created a synthetic dataset of 2 classes and 100 data points per class, using a Gaussian mixture. A linear kernel was computed for these data points. Let $K^s_{ij} = \overline{\mathbf{K}}_{ij} + Z^s_{ij}$, where $\overline{\mathbf{K}}_{ij}$ is computed from original datapoints and 100 uncertain samples Z^s_{ij} for each kernel entry K_{ij} was generated using: a) **Gaussian (0,1)** b) **Uniform [-0.5,0.5]** c) centered **Beta (0.5,0.5)** distributions and multiplied by a random l_{ij} ($l_{ij} = l_{ji}$). The support parameters are estimated as $a_{ij} = \min_s K^s_{ij}$ and $b_{ij} = \max_s K^s_{ij}$. For the Gaussian case the parameter β is estimated as $\beta_{ij} = \sigma^2_{ij} = var(K^s_{ij})$. For non-Gaussian case β is calculated from (9).

For $RSVM_{QP}$, we approximated β by $\hat{\beta} = \sqrt{v_{max}} \rho \rho^\top$, where v_{max} and ρ are principal eigenvalue and eigenvector of $\beta\beta^\top$ respectively. For $RSVM_{SOCP}$ we approximated β by $\hat{\beta} = \sum_{r, v_r\ \geq 0} v_r e_r e_r^\top$, where v_r and e_r are corresponding eigenvlaues and eigenvectors of β respectively.

5.1.1. Comparison of Generalization error

All six formulations proposed here are compared with Nominal SVM using the three metrics described in section 4. For all the metrics, we performed 5-fold cross-validation on 20 different datasets. The hyperparameters (C and/or ϵ) for each classifier, were chosen using a grid search from the set $C = \{0.1,\ 1,\ 5,\ 10,\ 50,\ 100\}$ and $\epsilon = \{0.05 : 0.05 : 0.5\}$. For each metric, the best cross-validation accuracy averaged over 20 dataset is reported in table 1.

We observe that either $\mathbf{RSVM_{SOCP}}$ or $\mathbf{RSVM^{(g)}_{SOCP}}$ perform best in terms of all the error measures, clearly demonstrating the power of the proposed methods. For generating synthetic data, we chose $l_{ij} = 0.25 * K_{ij}$, in order to have a dataset where the noise is less than the actual kernel values. Hence, the β matrix turns out to be PSD most of the times, thus leading to better performance of $\mathbf{RSVM_{SOCP}}$ or $\mathbf{RSVM^{(g)}_{SOCP}}$. $\mathbf{RSVM_{QN}}$ and $\mathbf{RSVM^{(g)}_{QN}}$ follow closely, because they get stuck at local optima. $\mathbf{RSVM_{QP}}$ and $\mathbf{RSVM^{(g)}_{QP}}$ show intermediate performance compared to SVM.

In terms of RobustErr, SVM performs very badly, showing its lack of ability to achieve robustness. Also, $\mathbf{RSVM^{(g)}}$ is found to perform better than **RSVM** when the uncertainty is Gaussian. These observations are explored in detail below.

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

Table 1. Cross-validation accuracy (%) obtained with $RSVM_{QP}^{(g)}$, $RSVM_{SOCP}^{(g)}$, $RSVM_{QN}^{(g)}$, $RSVM_{QP}$, $RSVM_{SOCP}$, $RSVM_{QN}$, Nominal SVM using NomErr (NE, 20), MajErr (ME, 18) and RobustErr (RE, 19).

	$RSVM^{(g)}$			$RSVM$			SVM
	QP	SOCP	QN	QP	SOCP	QN	
	Uniform Distribution						
ME	94.60	96.15	95.55	95.30	**96.60**	95.60	87.70
RE	52.60	93.30	92.85	74.20	**95.20**	94.60	18.65
NE	80.30	**95.94**	95.50	83.20	95.60	95.60	58.79
	Gaussian Distribution						
ME	95.45	**96.35**	95.95	94.50	95.60	95.70	71.95
RE	55.15	**93.70**	92.70	64.30	84.10	84.10	24.75
NE	74,27	**95.18**	94.92	80.45	94.80	94.65	52.69
	$\beta(0.5, 0.5)$						
ME	95.20	96.15	95.60	95.35	**96.20**	95.95	86.75
RE	47.45	91.45	91.35	75.50	**94.30**	93.85	5.20
NE	79.22	95.91	95.48	85.35	**96.20**	95.95	57.25

5.1.2. Comparison of robustness

In the proposed **RSVM** (or $\mathbf{RSVM^{(g)}}$), the effect of uncertainty in training data is controlled by ϵ and hence κ (15). Higher the value of κ, higher the effect of uncertainty. For $\kappa = 0$, **RSVM** ignores uncertainty in kernel values. A consistent reduction in uncertainty for the test data points is achieved by generating them as $K_{ij}^s = \overline{K}_{ij} + Z_{ij}^{s'}$, where $|Z_{ij}^{s'}| \leq \kappa * l_{ij}$ for uncertainty with finite support, and $Z_{ij}^{s'} \sim N(0, \kappa\sigma_{ij})$ for Gaussian uncertainty.

Figure 1 shows that, with the increase of uncertainty in test examples the RobustErr(19) for SVM increases more rapidly than that for $\mathbf{RSVM}_{SOCP}$, $\mathbf{RSVM}_{QN}$ for all types of uncertainties. This shows that non-robust classifiers, e.g. SVM, are unable to handle uncertainty, compared to the proposed robust classifiers. $\mathbf{RSVM}_{QP}$ performs comparably with SVM since the assumption of β being rank 1 does not hold for the current dataset. Hence, $\mathbf{RSVM}_{QP}$ becomes theoretically equivalent to SVM using a kernel with diagonal made heavy (see Theorem 5). Figure 2 shows that with increase in uncertainty which is i.i.d. $\mathbf{RSVM}_{QP}$ performs much better than Nominal SVM.

In Figure 1 and Figure 2 at $\kappa = 0$, RobustErr for both $RSVM$ and $RSVM^{(g)}$ are exactly same as that of SVM. It confirms the fact that at $\kappa = 0$, $RSVM^{(g)}$ and $RSVM$ are equivalent to SVM. Both $\mathbf{RSVM}_{QN}$ and $\mathbf{RSVM^{(g)}}_{QN}$ sometimes give higher error than $SOCP$ due to the solver getting stuck at local optimum.

5.1.3. Effectiveness of bound

In this section, using the same synthetic data as above, we experimentally verify the effectiveness of bounds discussed in Lemma 1 and Theorem 2, which were used to derive the RSVM formulation (15) from the chance constraint (2). For a given ϵ, we calculate $\epsilon_{effective} = \frac{\#\{K^s, s=1\dots n_s | \alpha^{*\top} Y K^s Y \alpha^* > t^*\}}{n_s}$, for the optimal values of α^* and t^* solving (15).

Figure 3 plots $\epsilon_{effective}$ vs ϵ. Ideally $\epsilon_{effective}$ should be equal to ϵ (shown as "Ideal-case" in graph). The leftmost plot shows that the Gaussian bound (Lemma 1) is much tighter than interval based bound if the uncertainty is Gaussian. In the other plots, we plot $\epsilon_{effective}$ obtained from **RSVM** for various values of ϵ, thus testing the bound in Theorem 2. The bound is very loose for the general distributions. However, we observe that for the interesting range of ϵ, $\epsilon \in [0, 0.5]$, the bound is tighter than rest of the region. The bound is very tight for small values of ϵ.

5.2. Resolution-aware protein structure classification

Here, we present experimental results which compare accuracy and robustness of the proposed **RSVM**, with state of the art methods for protein structure classification.

Dataset: We use a dataset based on SCOP (Murzin et al., 1995) 40% sequence non-redundant dataset, taken from (Bhattacharya et al., 2007). The dataset has 15 classes (SCOP superfamilies), having 10 structures each. The experimental methodology is also similar to that used in (Bhattacharya et al., 2007), e.g. using 15 "one versus all" binary classifiers, where the negative data contains 10 proteins (to keep the dataset balanced) randomly chosen from all other classes. We perform Leave-One-Out(LOO) crossvalidation here.

Let $\mathcal{D} = \{(P_i, r_i, y_i)\}$ where P_i is the set of coordinates of i^{th} protein structure obtained from Astral[3] database, r_i be the corresponding resolution information obtained from PDB, and y_i is the class label. Using resolution, we generate a set of perturbed structures $Q_i = \{P_i^1, \dots, P_i^{n_s}\}$ for each P_i as follows: for each atom p_{i_a} of P_i generate structure P_i^s with coordinates of atoms as $p_{i_a}^s = p_{i_a} + u$ and $u \sim U(\frac{-r_i}{2}, \frac{r_i}{2})$.

For any kernel K, mean kernel $\overline{K}_{ij} = E[K(p, p')]$. Also $a_{ij} = \min_{p \in Q_i, p' \in Q_j} K(p, p')$ and $b_{ij} = \max_{p \in Q_i, p' \in Q_j} K(p, p')$. For the purpose of our comparison, we have used weighted pairwise distance substructure kernel (Bhattacharya et al., 2007).

Existing techniques: Each protein structure can be viewed as a set of perturbations of the original structures. Hence, we compared $RSVM$

[3] http://astral.berkeley.edu

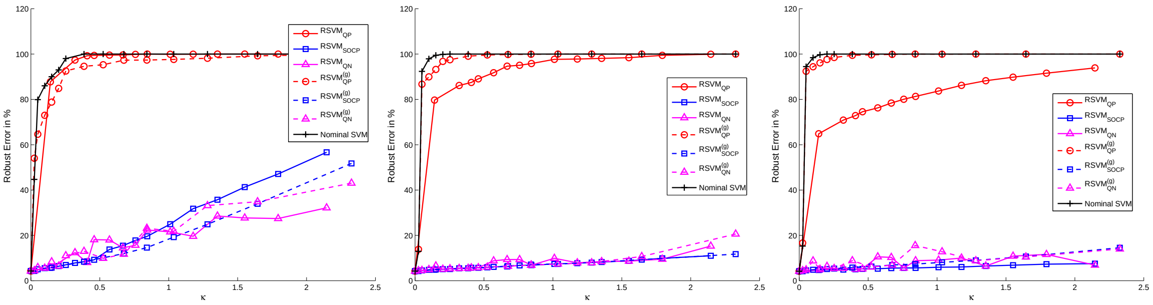

Figure 1. Robustness for **RSVM**, **RSVM$^{\mathbf{(g)}}$** and **Nominal-SVM** using (starting from left) Gaussian, Uniform and $\beta(0.5, 0.5)$ distribution for generating sample kernels. (Plot shows average error over 20 classifiers by fixing C at 10).

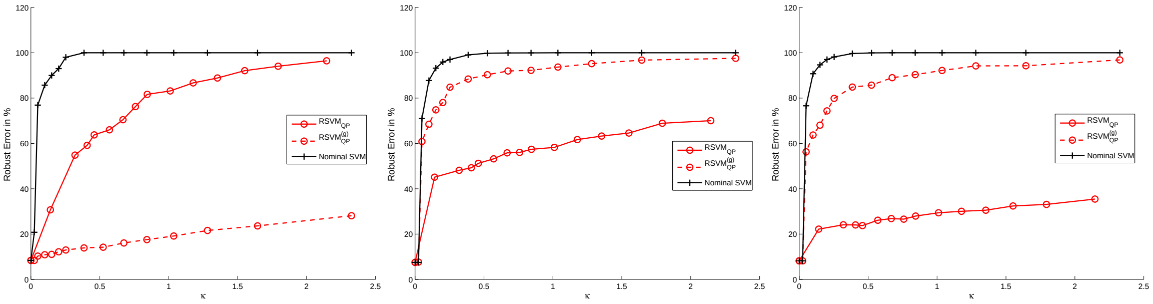

Figure 2. Robustness for **RSVM**$_{QP}$, **RSVM$^{\mathbf{(g)}}_{\mathbf{QP}}$** and **Nominal-SVM** using (starting from left) **i.i.d.** Gaussian, Uniform and $\beta(0.5, 0.5)$ distributions for generating sample kernels. (Plot shows average error over 20 classifiers by fixing C at 10).

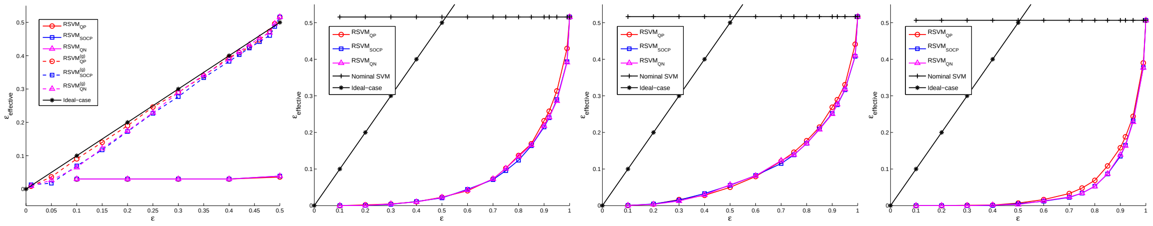

Figure 3. Verification of Bound for **RSVM** and **RSVM$^{\mathbf{(g)}}$** using (starting from left) Gaussian, Uniform, $\beta(0.5, 0.5)$ and $\beta(5, 5)$ distribution for generating sample kernels. (Plot shows average error over 20 classifiers by fixing C at 10).

with the normalized set multi-instance kernel (**MI**) (Gartner et al., 2002). For a given kernel K, normalized set kernel is defined as $K_{multi}(P_i, P_j) = \frac{\sum_{p \in Q_i, p' \in Q_j} K(p,p')}{\sqrt{(\sum_{p \in Q_i, p' \in Q_i} K(p,p'))}\sqrt{(\sum_{p \in Q_j, p' \in Q_j} K(p,p'))}}$.

We have used SVM_R(SVM considering each perturbed structure as individual data points) Nominal-SVM (SVM with kernel based on protein structure reported in PDB files), SVM_M (SVM considering $\overline{\mathbf{K}}$ as kernel) for benchmarking our result.

5.2.1. Results on protein structures

Table 2 reports results for **RSVM** and state of the art methods using both standard and robust error measures defined in section 4 using the LOO procedure. Hyper-parameters (C and/or ϵ) for each classifiers were tuned separately using a grid search. We report Total Accuracy (TA) and F1 score. All reported results are averaged over 10 different datasets, where negative dataset were selected randomly. Note that for SVM with MI kernel, one label is given to every set following (Gartner et al., 2002) method. We report this as the majority error, which will also be the Nominal error.

It is clear that $RSVM_{QN}$ performs significantly better than rest of the methods, both in terms of Nominal Accuracy (measured by NomErr) and Robustness (measured by RobustErr). This indicates that use of resolution information improves the overall classification accuracy. Moreover, very low values of accuracy

Table 2. Comparison $\mathbf{RSVM_{QP}}$, $\mathbf{RSVM_{SOCP}}$, $\mathbf{RSVM_{QN}}$, **Nominal-SVM**,$\mathbf{SVM_M}$.$\mathbf{SVM_R}$ and **MI** using accuracy measures defined in section 4

	RSVM			SVM			MI
	QP	SOCP	QN	Nominal	M	R	
	MajErr						
TA	72.67	73.56	**82.78**	62.89	71.11	71.67	72.11
F1	73.49	74.35	**82.95**	63.50	71.87	72.58	72.17
	RobustErr						
TA	27.11	50.33	**66.44**	34.56	22.00	61.56	20.11
F1	26.81	50.28	**66.36**	34.07	21.70	61.26	19.63
	NomErr						
TA	66.50	66.65	**76.00**	61.02	65.00	70.44	×
F1	65.13	65.16	**75.80**	60.86	64.48	67.58	×

corresponding to RobustErr for SVM and other competing methods suggests that the SVM classification is not robust to perturbations in coordinates of atoms within the resolution. The fact that other **RSVM** formulations perform worse than $RSVM_{QN}$ indicates that assumptions used to derive other formulations, e.g. Rank 1 or PSD, do not hold for this dataset.

In terms of RobustErr, $\mathbf{RSVM_{QP}}$ performs worse than $\mathbf{RSVM_{SOCP}}$, confirming the fact that PSD assumption is much better for robustness than rank one assumption. The simple heuristic of using all the perturbed samples ($\mathbf{SVM_R}$) performs very well in terms of robustness, which is intuitive. However, the computational complexity of SVM_R is $O(n_s^2)$ higher than others, which can be prohibitive for many cases.

6. Conclusion

We have presented an optimization problem (15), which is robust to uncertainty in the kernel matrix. The formulation applies to Gaussian uncertainty and as well as to arbitrary distributions with finite support. For the finite support case the formulation is derived from a novel large deviation inequality, stated in Theorem 2. The large deviation inequality is of independent interest and applies more generally to problems involving traces of random matrices. An interesting result is, for i.i.d uncertainty, the formulation reduces to SVM (Theorem 5). We show that for positive semidefinite β the the formulation is second-order-cone representable and can be solved by SOCP. On the real world problem of protein structure classification it yields significantly improved results.

Acknowledgments

Chiranjib Bhattacharyya and Sahely Bhadra are supported by grants from Yahoo! India. Sourangshu Bhattacharya is thankful to Yahoo! India for supporting this research. Aharon Ben-Tal's research is supprted by the MINERVA Foundation.

References

Ben-Tal, A. and Nemirovski, A. Selected Topics in Robust Convex Optimization. *Mathematical Programming*, 112(1), 2007.

Bhattacharya, S., Bhattacharyya, C., and Chandra, N. Structural alignment based kernels for protein structure classification. *ICML*, 2007.

Bhattacharyya, C., Grate, L. R., Jordan, M. I., Ghaoui, L. El, and Mian, S. I. Robust sparse hyperplane classifiers: application to uncertain molecular profiling data. *Journal of Computational Biology*, 11(6):1073–1089, 2004.

Gartner, T., Flach, P. A., Kowalczyk, A., and Smola, A. J. Multi-instance kernels. *ICML*, 2002.

Ghaoui, L. E., Lanckriet, G. R. G., and Natsoulis, G. Robust Classification with Interval Data. Technical Report UCB/CSD-03-1279, Computer Science Division, University of California, Berkeley, 2003.

Holm, L. and Sander, C. Mapping the protein universe. *Science*, 273(5275):595–602, 1996.

Luenberger, David G. and Ye, Yinyu. *Linear and nonlinear programming*. Springer, 2008.

Minc, H. On the maximal eigenvector of a positive matrix. *SIAM Journal on Numerical Analysis*, 7 (3):424–427, 1970.

Murzin, A. G., Brenner, S. E., Hubbard, T., and Chothia, C. Scop: A structural classification of proteins database for the investigation of sequences and structures. *Journal of molecular biology*, 247(4): 536–540, April 1995.

Nemirovski, A. and Shapiro, A. Convex Approximations of Chance Constrained Programs. *SIAM Journal of Optimization*, 17(4):969–996, 2006.

Qiu, J., Hue, M., B.-Hur, A., Vert, J.-P., and Noble, W. S. A structural alignment kernel for protein structures. *Bioinformatics*, 23(9):1090–1098, 2007.

Shivaswamy, P. K., Bhattacharyya, C., and Smola, A. J. Second Order Cone Programming Approaches for Handling Missing and Uncertain Data. *JMLR*, 7:1283–1314, 2006.

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