Journal of Machine Learning Research 13 (2012) 2923-2954 Submitted 5/11; Revised 2/12; Published 10/12

Efficient Methods for Robust Classification Under Uncertainty in Kernel Matrices

Aharon Ben-Tal ABENTAL@IE.TECHNION.AC.IL
William Davidson Faculty of Industrial Engineering and Management,
Technion- Israel Institute of Technology
Technion City, Haifa 32000, Israel

Sahely Bhadra SAHELY@CSA.IISC.ERNET.IN
Chiranjib Bhattacharyya CHIRU@CSA.IISC.ERNET.IN
Department of Computer Science and Automation,
Indian Institute of Science, Bangalore -560012
Karnataka, India

Arkadi Nemirovski NEMIROVS@ISYE.GATECH.EDU
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology, Georgia 30332-0205
Atlanta, Georgia, USA

Editor: John Shawe-Taylor

Abstract

In this paper we study the problem of designing SVM classifiers when the kernel matrix, **K**, is affected by uncertainty. Specifically **K** is modeled as a positive affine combination of given positive semi definite kernels, with the coefficients ranging in a norm-bounded uncertainty set. We treat the problem using the Robust Optimization methodology. This reduces the uncertain SVM problem into a deterministic conic quadratic problem which can be solved in principle by a polynomial time Interior Point (IP) algorithm. However, for large-scale classification problems, IP methods become intractable and one has to resort to first-order gradient type methods. The strategy we use here is to reformulate the robust counterpart of the uncertain SVM problem as a saddle point problem and employ a special gradient scheme which works directly on the convex-concave saddle function. The algorithm is a simplified version of a general scheme due to Juditski and Nemirovski (2011). It achieves an $O(1/T^2)$ reduction of the initial error after T iterations. A comprehensive empirical study on both synthetic data and real-world protein structure data sets show that the proposed formulations achieve the desired robustness, and the saddle point based algorithm outperforms the IP method significantly.

Keywords: robust optimization, uncertain classification, kernel functions

1. Introduction

The Support Vector Machine(SVM) formulation (Vapnik, 1998) learns a classifier of the form

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{1}$$

from a training data set $D = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{X}, y_i \in \{1,-1\}\ i = 1,\ldots,n\}$. The coefficients, α, are determined by solving

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

where $S_n = \{\alpha | 0 \leq \alpha_i \leq C,\ \sum_{i=1}^n \alpha_i y_i = 0\}$ and $Y = diag(y_1,\ldots,y_n)$. Each entry of the matrix $\mathbf{K}$, is defined by $\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, is a kernel function and it defines a dot product in an associated *Reproducing Kernel Hilbert Space* (Mercer, 1909; Shawe-Taylor and Cristianini, 2000). As a consequence of $K(\cdot,\cdot)$ being a dot product, the matrix $\mathbf{K}$ needs to be positive semi-definite (see, e.g., Shawe-Taylor and Cristianini, 2000) for any positive integer n.

Observations emanating from real world data are often plagued by uncertainty. The problem of designing classifiers for uncertain observations remain an interesting open problem and has gained considerable interest in the recent past. Previous attempts (Ghaoui et al., 2003; Bhattacharyya et al., 2004; Shivaswamy et al., 2006; Bhadra et al., 2009; Ben-Tal et al., 2011) at designing robust classifiers have been limited to the case of linear classification where the uncertainty is specified over an explicitly stated feature map.

Consider the problem of automated protein structure classification, an important problem of Computational Biology, where no such feature map is available. Protein Structures are specified by a set of 3D coordinates and it is possible to design kernel functions for protein structures based on the coordinates (Qiu et al., 2007; Bhattacharya et al., 2007). Unfortunately the coordinates are not known precisely and this makes the kernel values uncertain. Motivated by this problem (Bhadra et al., 2010) initiated a study of designing robust classifiers when the entries of the kernel matrix are independently distributed random variables (a somewhat problematic assumption). The approach, based on Chance-Constraints (probabilistic) formalism, leads to a non-convex problem which may result in an invalid (i.e., indefinite) kernel matrix.

In this paper we propose a Robust Optimization(RO) approach which overcomes the above drawbacks. The approach employs a geometric description of uncertainty instead of the probabilistic description used earlier (Bhadra et al., 2010). The uncertainty in the kernel matrix $\mathbf{K}$ is modeled by a bounded convex set, which encompasses several possible realizations of $\mathbf{K}$. This new approach results first in a robust counterpart of the uncertain SVM which can be cast as a Conic Quadratic (CQ) problem. Such problems can be solved in polynomial time by Interior Point (IP) algorithm. However for large-scale problems IP methods become intractable. Our main contribution here is to reformulate the robust counterpart as a saddle point problem. Due to favorable conditions satisfied by the saddle function one can in principle refer to a gradient-based general scheme introduced in (Juditski and Nemirovskii, 2011) for solving such saddle point problems. Using this scheme we propose an algorithm, which has a much more simplified analysis, and achieves the same efficiency estimate, namely it achieves the $O(1/T^2)$ reduction in the initial error after T iterations. Experimental results performed on synthetic data, as well as real-world protein structure data sets, show that the saddle-point based algorithm outperforms the IP method considerably. We further conduct detailed experimental evaluation to test the robustness and scalability of the obtained classifiers.

The paper is structured as follows. To motivate the paper we start with a brief discussion on issues underlying protein structure classification and kernel based classifiers in Section 2. In Section 3 we review the formulation in Bhadra et al. (2010) and identify the key shortcomings of the approach. The RO approach for designing robust SVMs is discussed in Section 4. The RO approach leads to a minimax problem. In Section 5 we present the saddle point algorithm and discuss

its application to the minimax problem. In Section 6 we prepare the ground for a comprehensive computational study by introducing various prediction rules and related error metrics. The results of the computational study are described in Section 7.

1.1 Notation

The space of symmetric positive semi-definite $n \times n$ matrices will be denoted as $\mathcal{S}_+^n$. Let $A * B$ denote the the Hadamard prodcuct of two matrices $(A*B)_{ij} = A_{ij}B_{ij}$ where A and B are two square matrices. Frobenius norm of matrix A will be denoted as $\|A\|_F = \sqrt{\sum_{ij} A_{ij}^2}$. Let $\mathbf{1}_A$ be the indicator function for the event A. The uniform random variate will be represented as $U(a,b)$ $(a < b)$. We denote $diag(x_1,\ldots,x_n)$ to be a $n \times n$ diagonal matrix whose ith diagonal entry is x_i.

2. Motivation: Uncertain Kernels and Automated Protein Structure Classification

Classification of protein structures into various classes like families, superfamilies etc remains an important research challenge in computational biology (see Holm and Sander, 1996 for an introduction). Kernel based classifiers are becoming increasingly popular (Qiu et al., 2007; Bhattacharya et al., 2007), for addressing this problem.

Usually a protein structure is specified by the positions of alpha carbon (C^α) atoms. A formal description of C^α atoms and protein structures is beyond the scope of the paper and we refer the reader to Branden and Tooze (1999) for an introduction. In the sequel we will denote protein structure by a set

$$P = \{\mathbf{c}_i \in \mathbb{R}^3 | i = 1, \cdots, s\}, \tag{3}$$

where each C^α atom is determined by spatial coordinates $\mathbf{c}_i = \{\mathbf{c}_{i1}, \mathbf{c}_{i2}, \mathbf{c}_{i3}\}$ obtained by X-ray crystallography. Automated classification of such structures is an extremely useful and challenging problem in computational biology. In the recent past kernel based methods (Qiu et al., 2007; Bhattacharya et al., 2007) have emerged as an interesting alternative to this problem.

Biologists often determine the similarity between a pair of structures by first computing an *alignment* and then measuring the quality of the *alignment* by root mean square deviation(RMSD). We do not formally define the notion of alignment and RMSD in this paper but the refer the interested reader to Shindyalov and Bourne (1998) and Holm and Sander (1996) for an introduction. Though computing structural alignment is an intractable problem there are several heuristic algorithms like DALI (Holm and Sander, 1996), CE (Shindyalov and Bourne, 1998) etc, which works well in practice. Existing literature (Qiu et al., 2007; Bhattacharya et al., 2007) on kernel design rely on structural alignments computed by such programs.

All such procedures implicitly assume that the protein structures are specified exactly, that is, the location of the atoms constituting the structure is known precisely. Unfortunately in reality, the coordinates, $\mathbf{c}_i$, are difficult to determine with exact precision and is highly dependent on the *resolution* of X-ray diffraction experiment.[1] For a protein structure P, the resolution information r, specifies the error in each coordinate. More formally the position of the ith atom in a protein structure P (see (3)) could be anywhere in the uncertainty box $\{\mathbf{c} | \|\mathbf{c} - \bar{\mathbf{c}}_i\|_\infty \leq r\}$, around the value $\bar{\mathbf{c}}_i$. For any $r > 0$ one can now define the uncertainty set $U(P)$ for any P as follows

$$U(P) = \{R | R = \{\mathbf{z}_1, \ldots, \mathbf{z}_s\} \ \|\mathbf{z}_i - \bar{\mathbf{c}}_i\|_\infty \leq r, \ \mathbf{z}_i \in \mathbb{R}^3, \ i = 1, \ldots, s\}. \tag{4}$$

1. See `http://www.rcsb.org/pdb/` for examples.

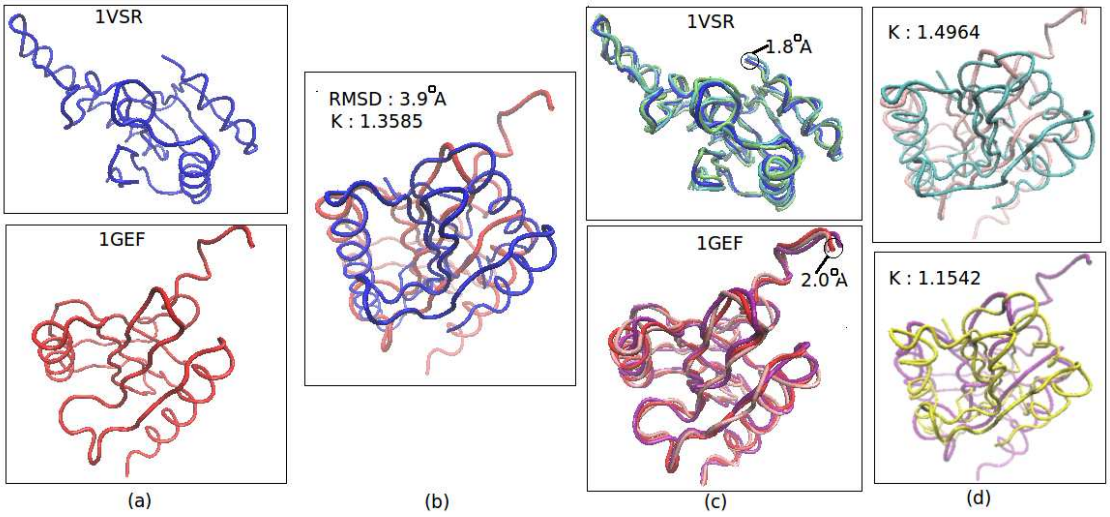

Figure 1: (a) Pictorial presentation of C^{α} atoms of protein **d1vsra1**(top) and **d1gefa1**(bottom). (b) Structural alignment between them. (c) Possible perturbation within resolution limit. (d) Alignment among perturbed structures

Furthermore we would refer to

$$\bar{P} = \{\bar{\mathbf{c}}_i \in \mathbb{R}^3 | i = 1, \ldots, n\}$$

as the *nominal* structure and $U(P)$ as the uncertainty set associated with it. The set $U(P)$ characterizes all alternative structures, including $\bar{P}$ for a given value of r.

The structural alignment between P and P' in presence of uncertainty sets $U(P)$ and $U(P')$ is not defined anymore. Even when r is small, the alignment scores between two *nominal* structures, $\bar{P}$ and $\bar{P}'$ can differ significantly from the alignment scores between an arbitrary R and R' where $R \in U(P), R' \in U(P')$. This difference in alignment scores leads to uncertain kernel values.

For example, consider two proteins[2] **d1vsra1**(denote it by P) and **d1gefa1**(denote it by P') belonging to protein superfamily **Restriction endonuclease-like**. The value of r for P is $1.8\mathring{A}$ and for P' it is $2.0\mathring{A}$ respectively. The program DALI computes a structural alignment with RMSD of 3.7 Å between these two structures. Figure 1(a) shows pictorial presentation of C^{α} atoms of these two proteins while Figure 1(b) shows structural alignment between them. If one ignores the uncertainty one obtains a kernel value of 1.3585, using the kernel function described in Bhattacharya et al. (2007). On randomly sampled structures, from the corresponding uncertainty box (4) we observe that the kernel value ranged from $1.1542 (= K_{min}) \leq K(P,P') \leq 1.4964 (= K_{max})$, see Figure 1(c,d). This variability is indeed substantial. In superfamily **Restriction endonuclease-like**, more than 60% of the kernel values, computed between any pair of nominal protein structures, lie between K_{min} and K_{max}.

2. One should refer to them as SCOP domains. But to lighten the discussion on the biology side we refer to them as proteins.

This demonstrates that accounting for resolution information leads to considerable uncertainty in kernel values. There is clearly a need for designing classifiers which can withstand this variation in kernel values. This is an extremely challenging problem which has not been well studied in the literature. Recently (Bhadra et al., 2010) initiated a study of this problem in a probabilistic setting. In the following section we review this work, to identify key shortcomings and subsequently propose a robust optimization procedure to address them.

3. Related Work

In Bhadra et al. (2010), the uncertainty is modeled by independent noise in each of the entries of the kernel matrix $\mathbf{K}$. The uncertain event $\left(\alpha^\top Y(\mathbf{K})Y\alpha \leq t\right)$ is then required to occur with high probability. This results in the following *chance constraint* problem:

$$p^* = \max_{t,\alpha\in S_n} \alpha^\top e - \frac{1}{2}t \tag{5}$$

$$\text{s.t. } \mathit{Prob}\left(\alpha^\top Y(\overline{\mathbf{K}}+Z)Y\alpha \leq t\right) \geq 1-\varepsilon \tag{6}$$

where $\varepsilon < 0.5$, and where Z is a random matrix variate.

Problem (5) is hard to solve since typically the feasible set is non-convex. The key result is the derivation of a lower bound on p^* under probabilistic assumptions on the entries of Z

Theorem 1 ***[Bhadra et al., 2010]*** *Let Z be an $n \times n$ random symmetric matrix with independent entries, Z_{ij}, each having finite support, $P(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1$ and $E(Z_{ij}) = 0$. Then the chance constraint in* (5) *is satisfied at any pair (α, t) which is feasible for the constraint*

$$\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \sqrt{2\log(1/\varepsilon)}\|\beta' * (Y\alpha\alpha^\top Y)\|_F \leq t$$

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

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

Proof See Bhadra et al. (2010). ■

Theorem 1 is used to replace (5) by the following problem, whose optimal value lower-bounds p^*. Specifically $p^* > \hat{p}$, where

$$\boxed{\begin{aligned} &\hat{p} = \max_{t,\alpha\in S_n} \frac{1}{2}t - \sum_i \alpha_i \\ &\text{s.t. } \sum_{ij} y_i y_j \alpha_i \alpha_j \overline{K}_{ij} + \kappa\sqrt{\sum_{ij}\beta_{ij}\alpha_i^2\alpha_j^2} \leq t \end{aligned}} \tag{7}$$

where, $\kappa = \sqrt{2\log(1/\varepsilon)}$ and $\beta_{ij} = \beta'^2_{ij}$. Any solution of the above problem is guaranteed to satisfy the chance constraint of problem (5).

This approach suffers from several drawbacks. First unless the matrix β is psd, problem (7) is not necessarily convex. Indeed in Bhadra et al. (2010) a locally optimal Quasi newton procedure

was suggested for solving this problem. In the sequel **RSVM** will denote the solution of (7) by the Quasi newton procedure. The second most important drawback is that the constraint (6) does not define a valid model of uncertainty.

To constitute a valid characterization of uncertainty set the constraint (6) needs to be modified as follows

$$Prob\left(\alpha^\top Y(\overline{\mathbf{K}}+Z)Y\alpha \le t\right) \ge 1-\varepsilon,\ \ \overline{\mathbf{K}}+Z \in \mathcal{S}_n^+$$

The formulation (5) solves a relaxed version of the above problem by ignoring the psd requirement. As a consequence the resultant optimization problem becomes non-convex. Thirdly, the assumption that entries of Z are independently distributed is extremely unrealistic; often the uncertainty in the entries are due to uncertainty in the observations hence $K(\mathbf{x}_i,\mathbf{x}_j)$ is seldom independent of $K(\mathbf{x}_i,\mathbf{x}_l)$ for distinct i,j,l.

In this work we pursue a RO methodology where the uncertainty is described by a geometric set. This allows us to alleviate the drawbacks associated with the probabilistic model. In the next section we describe a RO procedure for designing robust classifiers.

4. Affine Uncertainty Set Model for Uncertain Kernel Matrices

In this section we introduce an uncertainty set over psd matrices and study the resultant robust SVM problem using an RO approach.

4.1 Robust Optimization

Consider an uncertain optimization problem (8) where $f, g_i : S \subset (\mathbb{R}^n \times \mathbb{R}^k) \Rightarrow \mathbb{R}$

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x},\Psi) \tag{8}$$

$$g_i(\mathbf{x},\Psi) \le 0\ i=1,\ldots,m$$

where $\Psi \in \mathbb{R}^k$ is a vector of uncertain parameters. The UOP is in fact a *family* of problems -one for each realization of Ψ. In the RO framework the information related to Ψ is modelled as a *geometric* uncertainty set $\mathcal{E} \subset \mathbb{R}^k$ and the family of problems, (8) is replaced by its *robust counterpart*:

$$r^* = \min_{\mathbf{x}} \max_{\Psi\in\mathcal{E}}\ f(\mathbf{x},\Psi) \tag{9}$$

$$g_i(\mathbf{x},\Psi) \le 0\ \forall \Psi \in \mathcal{E}\ i=1,\ldots,m.$$

A solution of (9) is feasible to (8) for any realization of $\Psi \in \mathcal{E}$ and the objective function is guaranteed to be no worse than r^*. The uncertainty set $\mathcal{E}$ is typically a polytope or ellipsoid or intersection of such sets. These sets yield useful models of uncertainty, which lead to tractable optimization problems (Ben-Tal et al., 2009). A general representation of $\mathcal{E}$ is as follows

$$\mathcal{E} = \{\Psi = \bar{\Psi} + \sum_{i=1}^{L} \eta_i \Psi^i | \|\eta\| \le \rho\}$$

where $\bar{\Psi}$ is the nominal value of the uncertain vector Ψ, the vectors Ψ^i are possible scenarios of it, and η is a perturbation vector. The norm is suitably defined to capture the geometry of the set. As an example, Consider the ellipsoidal set

$$\mathcal{E}_{ellipsoid} = \{\Psi | (\Psi-\bar{\Psi})^\top Q(\Psi-\bar{\Psi}) \le \rho\}$$

where $Q \in \mathcal{S}_+^n$ and is positive definite. It is easily seen that the set can be represented by $\mathcal{E}$ with Ψ^i being the columns of $Q^{-\frac{1}{2}}$ and where $\|\cdot\|$ is the L_2 norm.

In general the RC of an UOP may have infinite number of constraints and is often NP hard. However in several important cases it reduces to a polynomially solvable convex optimization problem. We refer the reader to Ben-Tal et al. (2009) for a comprehensive treatment of RO problems.

4.2 Affine Uncertainty Set Model

Recall the setup for the problem, there is a black box which when presented with a pair of observations $z, z' \in \mathcal{X}$ computes the kernel value $K(z,z')$. We assume that if z and z' are noisy observations with uncertainty sets $U(z)$ and $U(z')$ with nominal values z_{nom} and z'_{nom} respectively then $\overline{K}(z,z') = K(z_{nom}, z'_{nom})$, defines a kernel function and will be called the *nominal kernel*. The difference between actual and the nominal kernel is expressed by a linear combination of known L kernel functions, $K_l, l = 1,\ldots,L$ evaluated at points z, z', as follows:

$$K(z,z') - \overline{K}(z,z') = \sum_{l=1}^{L} \eta_l K_l(z,z').$$

When there is no uncertainty $K(z,z') = \overline{K}(z,z')$ and $\eta = 0$. The value of $K(z,z')$ lies in the uncertainty set

$$\{\overline{K}(z,z') + \sum_{l=1}^{L} \eta_l K_l(z,z') | \|\eta\|_p \leq \kappa,\ \eta_l \geq 0\ \forall l = 1,\ldots,L\} \tag{10}$$

where $\|\eta\|_p, p \geq 1$ denotes the l_p norm on η. The constraint $\eta_l \geq 0$ is needed to ensure that each element in the set represents a valid kernel evaluation. The quantity κ measures the quality of approximation and hence the uncertainty. If $\kappa = 0$ then we have no uncertainty. As κ increases the uncertainty set increases. In the sequel we will refer to $\overline{K}, K_l$ as base kernels.

We impose the uncertainty set (10) to all examples of interest which immediately leads to the following model of uncertainty on the kernel matrix corresponding to the training set,

$$\mathcal{E}(\kappa) = \{\mathbf{K} = \overline{\mathbf{K}} + \sum_{l=1}^{L} \eta_l \mathbf{K}_l,\ \|\eta\|_p \leq \kappa\, \eta_l \geq 0, l = 1,\ldots,L\}. \tag{11}$$

The matrices $\overline{\mathbf{K}}, \mathbf{K}_l \in \mathcal{S}_n^+$ are obtained by evaluating the known kernel functions $\overline{K}, K_l$ on the training set. As any $\mathbf{K} \in \mathcal{E}(\kappa)$ is always positive semi-definite, the set $\mathcal{E}(\kappa)$ defines a valid model for describing uncertainty in psd matrices. In a later subsection we will discuss the relevance of this setup to protein structure classification problem.

The Robust SVM problem (2) with uncertain $\mathbf{K}$, as characterized in (11), can now be cast as follows

$$\max_{\alpha \in S_n} \min_{\mathbf{K} \in \mathcal{E}(\kappa)} \quad -\frac{1}{2}\alpha^\top Y \mathbf{K} Y \alpha + \alpha^\top e$$

or more explicitly

$$\max_{\alpha \in S_n} \quad \min_{\|\eta\|_p \leq \kappa} \quad -\frac{1}{2}\alpha^\top Y \overline{\mathbf{K}} Y \alpha - \frac{1}{2}\alpha^\top Y \sum_{l=1}^{L} (\eta_l \mathbf{K}_l) Y \alpha + \alpha^\top e \quad . \tag{12}$$

Note that in the latter problem the constraint $\eta \geq 0$ is dropped. Indeed if we define $a_l = \alpha^\top Y \mathbf{K}_l Y \alpha$ then $a_l \geq 0$ as $\mathbf{K}_l \in \mathcal{S}_n^+$. The optimal η is the solution of $\max_{\eta \in B_p} a^\top \eta$ where $B_p(\kappa) = \{\eta | \|\eta\|_p \leq \kappa\}$.

which occurs at $\eta_l = \kappa \frac{a_l^{q-1}}{\|a\|_q^{q-1}} \geq 0$ where $\|\cdot\|_q$ is the dual norm of $\|\cdot\|_p$, with $q = \frac{p}{p-1}$ for any $p > 1$. For $p = 1$ one needs to observe that optimality is achieved at $\eta_l \geq 0$. Indeed there exists some l such that $a_l = \|a\|_\infty$. The optimal η is given by the condition that $\sum_{l:a_l=\|a\|_\infty} \eta_l = 1$ and $\eta_l \geq 0$. If $a_l < \|a\|_\infty$ then η_l is strictly 0. At optimality $\eta_l \geq 0$, and hence the resulant optimal kernel lies in $\mathcal{E}(\kappa)$.

Before proceeding further it might be useful to discuss the Computation of b. Recall that one also needs to compute b in (1). The choice of b is governed by the following procedure. For a given $\overline{\mathbf{K}}, \mathbf{K}_1, \ldots, \mathbf{K}_L$ let the optimal solution of (12) be α^* and η^*. This η^* can be viewed as defining an effective kernel $\mathbf{K} = \overline{\mathbf{K}} + \sum_{l=1}^{L} \eta_l^* \mathbf{K}_l$ and hence b can be computed as

$$b = \frac{1}{\#\, SV} \sum_{j \in SV} [y_j - \sum_i y_i \alpha_i^* K_{ij}] \;\; SV = \{i | \alpha_i > 0\}.$$

For $p = 2$ the problem (12) can be solved as a Second Order Cone Program(SOCP)[3]

$$\begin{aligned} \min_{\alpha \in S_n, a, t} \quad & \tfrac{1}{2} t + \tfrac{1}{2} \alpha^\top Y \overline{\mathbf{K}} Y \alpha - \alpha^\top e \\ s.t \quad & \|a\|_2 \leq t \\ & \alpha^\top Y \mathbf{K}_l Y \alpha \leq a_l, \;\; \forall l = 1, \ldots, L. \end{aligned} \tag{13}$$

SOCP problems such as (13) can be solved by Interior point(IP) algorithms (e.g., CPLEX, MOSEK, Sedumi) and will be denoted by **Uncertainty-Set SVM ($\mathbf{USSVM_{SOCP}}$)**.

Generic IP solvers are inadequate for large scale classification problems. Instead, we demonstrate here that the minimax reformulation, (12), admits algorithms which are better suited to large scale problems. For simplicity of exposition we will consider the case $p = 2$. The results can be easily extended for the general case, $p > 1$. Before we discuss the algorithmic aspects it maybe useful to discuss the computation of the base kernels.

4.3 Evaluation of Base Kernels

Recall that in the protein structure classification problem each observation is specified by a $(\bar{P}, U(P), y)$, where P(see (3)) is the nominal structure, $U(P)$(see (4)) is the uncertainty set specified by the resolution and y is the label. We discuss this problem in a formal setup and motivate the uncertainty set described in (10).

To closely parallel the protein structure classification setting we consider the following setup. Given a data set $D = \{(\bar{\mathbf{x}}_i, U_i) | \bar{\mathbf{x}}_i \in U_i, U_i \in \mathcal{X}, i = 1, \ldots, m\}$ where an observation, $\mathbf{x}_i$, is not directly specified, instead a nominal value, $\bar{\mathbf{x}}_i$, and an uncertainty set U_i are given. When there is no uncertainty the set U_i reduces to only $\bar{\mathbf{x}}_i$. We are also given a kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. We make no assumptions about the functional form of $K(\cdot, \cdot)$, and we assume that there is a black box which when presented with any pair $z, z' \in \mathcal{X}$ returns a value $K(z, z')$.

Let us now consider the computation of $K(\mathbf{x}_i, \mathbf{x}_j)$. Since $\mathbf{x}_i, \mathbf{x}_j$ are uncertain the precise value of $K(\mathbf{x}_i, \mathbf{x}_j)$ is not known but it for sure lies in the set

$$\mathcal{U}(\mathbf{x}, \mathbf{x}') = \{K(z, z') | z \in U(\mathbf{x}), z' \in U(\mathbf{x}')\}.$$

3. This is also true for any $(p > 1)$ because any p-norm can be represented by conic quadratic inequalities. For a more detailed discussion on this issue see Ben-Tal and Nemirovski (2001).

If there is no uncertainty then the set $\mathcal{U}(\mathbf{x},\mathbf{x}')$ is a singelton, $K(\bar{\mathbf{x}},\bar{\mathbf{x}}')$. However as the functional form of K is not known, it is not clear how to characterize the elements of $\mathcal{U}$ which is amenable to the RO procedure. We propose to circumvent this problem by building an alternate description of the set $\mathcal{U}$ using a sampling procedure. We make L independent draws from the uncertainty sets. In the lth draw we obtain $O^l = \{z_1^l, \ldots, z_m^l\}$ where z_i^l is an independent and uniform draw from U_i. For a given O^l we invoke the kernel function K to obtain an $m \times m$ kernel matrix K_l where $K_l(\mathbf{x}_i,\mathbf{x}_j) = K(z_i^l, z_j^l)$. Once K_l are determined we propose to approximate $\mathcal{U}$ by uncertainty set described in (10).

5. An Algorithm for a Special Class of Convex-Concave Saddle Point Problems

In this section we describe a novel algorithm, which is essentially a special case of an algorithm presented in Juditski and Nemirovski (2011), for a class of convex-concave saddle point problems. The algorithm is iterative in nature, requires only first order information, and has $O(1/T^2)$ convergence where T is the total number of iterations. The proposed algorithm applies generally, more specifically we show that it can be used to solve (12) and the convex version of (7).

5.1 Assumptions

Let U be a closed convex set in Euclidean space E, $\|\cdot\|$ be a norm on E, and $\omega(u) : U \to \mathbf{R}$ be a function. We say that $\omega(\cdot)$ is a distance-generating function (d.-g.f.) for U compatible with $\|\cdot\|$, if ω is convex and continuous on U, admits a continuous on the set $U^o = \{u : \partial\omega(\cdot) \neq \emptyset\}$ selection of $\omega'(u) \in \partial\omega(u)$ and is strongly convex, modulus 1 w.r.t. $\|\cdot\|$:

$$\langle \omega'(u) - \omega'(u'), u - u' \rangle \geq \|u - u'\|^2 \ \forall u, u' \in U^o.$$

Assume that

- X is a closed and bounded convex subset in a Euclidean space $\mathcal{X}$, and Y is a closed convex subset in a Euclidean space $\mathcal{Y}$
- $\mathcal{X}$, $\mathcal{Y}$ are equipped with norms $\|\cdot\|_{\mathcal{X}}$, $\|\cdot\|_{\mathcal{Y}}$, the conjugate norms being $\|\cdot\|_{\mathcal{X},*}$, $\|\cdot\|_{\mathcal{Y},*}$;
- X is equipped with a d.-g.f. $\omega_X(\cdot)$ compatible with $\|\cdot\|_{\mathcal{X}}$, and $\mathcal{Y}$ is equipped with a d.-g.f. $\omega_{\mathcal{Y}}(\cdot)$. We denote by x_ω the minimizer of $\omega_X(\cdot)$ on X (note that $x_\omega \in X^o$) and set

$$\Omega_X = \max_{x \in X} \omega_X(x) - \min_{x \in X} \omega_X(x).$$

  We assume that the minimizer of $\omega_{\mathcal{Y}}(\cdot)$ is the origin in $\mathcal{Y}$, and set

$$\Omega_{\mathcal{Y}} = \max_{\|y\|_{\mathcal{Y}} \leq 1} \omega_{\mathcal{Y}}(y) - \min_{y \in \mathcal{Y}} \omega_{\mathcal{Y}}(y).$$

We are interested in solving a saddle point problem

$$\mathrm{SadVal} = \min_{x \in X} \max_{y \in Y} \phi(x,y), \tag{14}$$

where $\phi(\cdot,\cdot)$ satisfies the following assumptions:

A.1. $\phi(x,y) : Z := X \times Y \to \mathbf{R}$ is continuously differentiable function which is convex in $x \in X$ and is strongly concave, modulus $\theta > 0$ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, in $y \in Y$, that is,

$$\langle \phi'_y(x,y') - \phi'_y(x,y), y - y' \rangle \geq \theta \|y - y'\|^2_{\mathcal{Y}} \ \forall (x \in X, y, y' \in Y).$$

A.2. $\nabla\phi(\cdot,\cdot)$ is Lipschitz continuous on $Z = X \times Y$

A.3. $\phi(x,y)$ is affine in x: $\phi(x,y) = \langle x, a(y) \rangle + g(y)$.

In the sequel, we set

$$\overline{\phi}(x) = \max_{y \in Y} \phi(x,y), \quad \underline{\phi}(y) = \min_{x \in X} \phi(x,y),$$

so that $\overline{\phi}$ is a continuous convex function on X, $\underline{\phi}$ is a continuous strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, function on Y, and

$$\min_{x \in X} \overline{\phi}(x) = \text{SadVal} = \max_{y \in Y} \underline{\phi}(y)$$

and the set of saddle points of ϕ on $X \times Y$ is $X_* \times \{y_*\}$, where $X_* = \text{Argmin}_X \overline{\phi}$, and $y_* = \text{Argmax}_Y \underline{\phi}$. Finally, we denote by $\varepsilon_{\text{sad}}(z)$, $z \in Z$, the natural saddle point proximity measure:

$$\varepsilon_{\text{sad}}(x,y) = \overline{\phi}(x) - \underline{\phi}(y) = \left[\overline{\phi}(x) - \min_X \overline{\phi}\right] + \left[\max_Y \underline{\phi} - \underline{\phi}(y)\right].$$

5.2 Fixed Step-size per Stage(FSS) Algorithm for Convex-Concave Saddle Point Problem

The MPb algorithm presented in Juditski and Nemirovski (2011) is an extremely fast algorithm for convex-concave saddle point problems. Computation proceeds in several stages, each stage consisting of multiple updates involving varying stepsizes. Here we introduce a variation of the MPb algorithm called FSS, which employs fixed stepsize at every stage of the algorithm. We prove that this apparent limitation does not harm the theoretical convergence. The proof (see Appendix A) needs milder assumptions and is much simpler than the original MPb algorithm.

We present a variation of the algorithm where the stepsizes are fixed at every stage without any loss of convergence efficency. In the following we present the Fixed stepsize per stage(FSS) version of the MPb algorithm along with convergence analysis.

5.2.1 FSS ALGORITHM

We begin by introducing some notation

$$G(x,y) = \left[G_x(x,y) := \frac{\partial \phi(x,y)}{\partial x} = a(y); G_y(x,y) := -\frac{\partial \phi(x,y)}{\partial y}\right] : Z := X \times Y \to \mathcal{Z} := \mathcal{X} \times \mathcal{Y}$$

be the monotone operator associated with the saddle point problem (14). By A.1, this operator is Lipschitz continuous, and, as we see, its x-component depends solely on y. As a result, we can specify the partial Lipschitz constants L_{xy}, L_{yy} such that

$$\begin{array}{l} \forall (x, x' \in X, y, y' \in Y): \\ \|G_x(x,y) - G_x(x,y')\|_{\mathcal{X},*} \leq L_{xy}\|y - y'\|_{\mathcal{Y}}; \ \|G_y(x,y) - G_y(x',y)\|_{\mathcal{Y},*} \leq L_{xy}\|x - x'\|_{\mathcal{X}}; \\ \|G_y(x,y) - G_y(x,y')\|_{\mathcal{Y},*} \leq L_{yy}\|y - y'\|_{\mathcal{Y}}. \end{array} \tag{15}$$

We are now ready to present FSS algorithm. Execution of the algorithm is split into *stages.* At the beginning of stage $s = 0, 1, ...$ we have at our disposal positive R_s and a point $\bar{y}_s \in Y$ such that

$$\|\bar{y}_s - y_*\|_{\mathcal{Y}} \leq R_s/2. \qquad (I_s)$$

These data define the quantities

$$\begin{array}{ll}
(a) & Z_s = \{(x;y) \in Z : \|y - \bar{y}_s\|_{\mathcal{Y}} \leq R_s\}, \\
(b) & \mathcal{L}_s = 2L_{xy}\sqrt{\Omega_X\Omega_{\mathcal{Y}}}R_s + L_{yy}\Omega_{\mathcal{Y}}R_s^2,\ \tau_s = 1/\mathcal{L}_s \\
(c) & \alpha_s = [L_{xy}\sqrt{\Omega_X\Omega_{\mathcal{Y}}}R_s]/\mathcal{L}_s, \beta_s = [L_{xy}\sqrt{\Omega_X\Omega_{\mathcal{Y}}}R_s + L_{yy}\Omega_{\mathcal{Y}}R_s^2]/\mathcal{L}_s = 1 - \alpha_s, \\
(d) & \omega_s(x,y) = \left[\frac{\alpha_s}{\Omega_X}\omega_X(x) + \frac{\beta_s}{\Omega_{\mathcal{Y}}}\omega_{\mathcal{Y}}([y - \bar{y}_s]/R_s)\right], \\
(e) & N_s = \mathrm{Ceil}\left(\frac{64L_{xy}\sqrt{\Omega_X\Omega_{\mathcal{Y}}}R_s^{-1} + 32L_{yy}\Omega_{\mathcal{Y}}}{\theta}\right).
\end{array} \qquad (16)$$

At stage s we carry out N_s steps of the following recurrence:

1. *Initialization:* We set $z_{1,s} = (x_\omega, \bar{y}_s) = \mathrm{argmin}_Z\, \omega_s(\cdot)$.

2. *Step* $t = 1, 2, ..., N_s$*:* Given $z_{t,s} \in Z^o$, we compute

$$\begin{array}{rcl}
w_{t,s} & = & \mathrm{argmin}_{u \in Z}\left\{\langle \tau_s G(z_{t,s}) - \omega_s'(z_{t,s}), u\rangle + \omega_s(u)\right\} \\
z_{t+1,s} & = & \mathrm{argmin}_{u \in Z}\left\{\langle \tau_s G(w_{t,s}) - \omega_s'(z_{t,s}), u\rangle + \omega_s(u)\right\}
\end{array} \qquad (17)$$

   and pass to step $t+1$, provided $t < N_s$. When $t = N_s$, we define the approximate solution to (14) built in course of s stages as

$$(x^s, y^s) = N_s^{-1}\sum_{t=1}^{N_s} w_{t,s}, \qquad (18)$$

   set

$$\bar{y}^{s+1} = y^s,\ R_{s+1} = R_s/2$$

   and pass to stage $s+1$.

5.2.2 PROOF OF CONVERGENCE

In this section we discuss the convergence properties of the FSS algorithm. To this end we present the following proposition.

Proposition 2 *Let assumptions A.1-3 hold and let $\bar{y}_0 \in Y$ satisfying (I_0) be given along with $R > 0$. Then, for every s, (I_s) takes place, and*

$$\varepsilon_{\mathrm{sad}}(x^s, y^s) \leq \theta R_0^2 2^{-2s-5}, \qquad (19)$$

while the total number $M_s = \sum_{i=0}^s N_i$ of steps of the algorithm needed to build (x^s, y^s) admits the bound

$$M_s \leq O(1)\left[\frac{L_{xy}\sqrt{\Omega_X\Omega_{\mathcal{Y}}}}{\theta R_0}2^s + \frac{L_{yy}\Omega_{\mathcal{Y}} + \theta}{\theta}(s+1)\right].$$

In particular, setting

$$s^* = \max\left[s : \frac{L_{xy}\sqrt{\Omega_X\Omega_\mathcal{Y}}}{L_{yy}\Omega_\mathcal{Y}+\theta}2^s \le (s+1)R_0\right],$$

we have

$$\begin{aligned} s \le s^* \Rightarrow \quad & \varepsilon_{\text{sad}}(x^s,y^s) \le \theta R_0^2 2^{-2(s+2)} \ \& \ M_s \le O(1)\tfrac{L_{yy}\Omega_\mathcal{Y}+\theta}{\theta}(s+1) \\ s > s^* \Rightarrow \quad & \varepsilon_{\text{sad}}(x^s,y^s) \le O(1)\tfrac{L_{xy}^2\Omega_X\Omega_\mathcal{Y}}{\theta M_s^2} \ \& \ M_s \le O(1)\tfrac{L_{xy}\sqrt{\Omega_X\Omega_\mathcal{Y}}}{\theta R_0}2^s. \end{aligned} \tag{20}$$

Proof See Appendix A. ■

The above proposition points us to the convergence rate of $O(1/M_s^2)$ when number of stages is high. In the following, we explain how to apply the algorithm to the problem at hand.

5.3 Application of FSS Algorithm to (12)

In this section we discuss the application of FSS algorithm for solving formulation (12). It is *strictly concave* in α, provided $\overline{\mathbf{K}}$ is positive definite, and *affine* in η. The domain of α and η are non-empty convex sets. Clearly the formulation obeys all the assumptions, A.1-3, of the FSS procedure. In order to be consistent with the notation of FSS procedure we define the following map where LHS denotes quantities involving formulation (12) and the right hand side corresponds to the saddle point procedure detailed in the previous section. In particular we use $\eta \to x$, $\alpha \to y$, $y \to \mathbf{s}$, $YK_lY \to Q_l$, $\mathbf{E}_l = \mathbb{R}^L$, leading to the following definitions.

$$\begin{aligned} Y =& \{y \in \mathbb{R}^n : C \ge y_i \ge 0, \ \sum_i \mathbf{s}_i y_i = 0\} \subset \mathbb{R}^n \\ X =& \{x \in \mathbb{R}^L : x \ge 0, \|x\|_2 \le 1\} \subset \mathbb{R}^L \\ \phi(x,y) =& -\sum_{l=1}^{L} x_l(\frac{1}{2}y^\top Q_l y) + y^\top e \\ G_x(x,y) =& -\mathbf{d}\ , \ \mathbf{d} = [d_1,\ldots,d_L]^T \ \ d_l = \frac{1}{2}y^\top Q_l y \\ G_y(x,y) =& (\sum_{l=1}^{L} x_l Q_l)y - \mathbf{e_1}\ , \ \mathbf{e_1} = [1,\ldots,1]^\top \\ \|x\|_\mathcal{X} =& \|x\|_2, \ \omega_X(x) = \frac{1}{2}\|x\|_2^2 \ \ x_\omega = 0\ , \ \Omega_X = \frac{1}{2} \\ \|y\|_\mathcal{Y} =& \|y\|_2, \ \omega_\mathcal{Y}(y) = \frac{1}{2}\|y\|_2^2, \ \ \Omega_\mathcal{Y} = \frac{1}{2} \\ L_{xy} =& 2CL_{yy}, \ \ L_{yy} = \sqrt{L}\max_l(\lambda_{\max}(Q_l)), \ \ R_0 = 2C\sqrt{n} \end{aligned}$$

Proposition 3 *Computing the right-hand sides in every step in (17) is equivalent to solving problems of the form*

$$\begin{aligned} &(a) \quad x_+ = \text{argmin}_{x\in X}\left[\tfrac{1}{2}(x-\overline{x})^T(x-\overline{x}) - p^T x\right], \\ &(b) \quad y_+ = \text{argmin}_{y\in Y}\left[\tfrac{1}{2}(y-\overline{y})^T(y-\overline{y}) - q^T y\right], \end{aligned} \tag{21}$$

where $\overline{x} \in X$, $\overline{y} \in Y$ *are solution in previous iteration and* $p = -\frac{\Omega_x \tau_s}{\alpha_s} G_x(\widetilde{x},\widetilde{y})$ *and* $q = -\frac{\Omega_y \tau_s}{\beta_s} R_s^2 G_y(\widetilde{x},\widetilde{y})$. *The point* $(\tilde{x},\tilde{y})$ *are intermediate points in* (17).

Proof See that we have $\omega_s(x,y) = \frac{\mu_s}{2}x^T x + \frac{\nu_s}{2}y^T y$ with certain positive μ_s, ν_s (see (16)), hence a computation of the type of (17), that is

$$\begin{aligned}&\{\overline{z} = (\overline{x},\overline{y}) \in Z^o, \tau > 0, G(\widetilde{x},\widetilde{y})\}\\ &\Rightarrow z_+ = (x_+,y_+) := \text{argmin}_{u\in Z}\left\{\langle \tau G(\widetilde{x},\widetilde{y}) - \omega_s'(\overline{z}), u\rangle + \omega_s(u)\right\}\end{aligned}$$

reduces to

$$\begin{aligned}x_+ &= \text{argmin}_{x\in X}\left\{x^T[\tau G_x(\widetilde{x},\widetilde{y}) - \mu_s\overline{x}] + \tfrac{\mu_s}{2}x^T x\right\}\\ &= \text{argmin}_{x\in X}\left\{-p^T x + \tfrac{1}{2}(x-\overline{x})^\top(x-\overline{x})\right\},\end{aligned}$$

and

$$\begin{aligned}y_+ &= \text{argmin}_{y\in Y}\left\{y^T[\tau G_y(\widetilde{x},\widetilde{y}) - \nu_s\overline{y}] + \tfrac{\nu_s}{2}y^T y\right\}\\ &= \text{argmin}_{y\in Y}\left\{-y^T q + \tfrac{1}{2}(y-\overline{y})^\top(y-\overline{y})\right\}.\end{aligned}$$

■

Assuming that we start with a feasible point, that is, $x^0 \geq 0$ and $x^{0^\top}x^0 = 1$, then $x^{k+1} = \frac{p+x^k}{\sqrt{p^T p + 2p^T x^k + 1}}$ The computation steps involving y is solved by projecting a vector onto the constraint set of the Dual SVM problem. The problem can be solved by a Quadratic program. However we propose a line search procedure described in Appendix B which results in considerable saving of computation time. The solution of formulation (12) by FSS procedure will be referred to as **USSVM$_{\mathbf{MN}}$**.

5.4 Application of FSS Algorithm to the Chance Constraint Setting

The FSS procedure is fairly general and also applies to formulation (7). If β is psd then the formulation (7) can be posed as a SOCP. In particular the following is true,

Theorem 4 ***[Bhadra et al., 2010]*** *If both* $\overline{\mathbf{K}}, \beta$ *are symmetric psd matrices then formulation (7) is equivalent to*

$$\begin{array}{cc}\min_{t,\nu,t',\alpha\in S_n} & \frac{1}{2}t - \sum_i \alpha_i\\ s.t. & \kappa\|\beta^{\frac{1}{2}}\nu\| \leq t - t',\\ & \|Y(\overline{\mathbf{K}})^{\frac{1}{2}}\alpha\|_2^2 \leq t',\\ & \alpha_i^2 \leq \nu_i.\end{array} \tag{22}$$

One could recast this problem as a mini-max problem and solve it using the FSS procedure.

Theorem 5 *Formulation* (22) *is equivalent to*

$$\min_{\zeta\geq 0, \sqrt{\zeta\beta^{-1}\zeta} \leq \frac{\kappa}{2}} \max_{\alpha\in S_n} -\frac{1}{2}\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \sum_{i=1}^n \alpha_i - \sum_i \zeta_i\alpha_i^2. \tag{23}$$

Proof At the optimum of (22) the constraint involving t and t' is active. Using this we eliminate both t and t' and on further dualizing one obtains,

$$\max_{\zeta\geq 0}\min_{\alpha\in S_n} \quad L(\zeta,\alpha)\left(= \frac{1}{2}\left(\alpha^\top Y\overline{\mathbf{K}}Y\alpha + \kappa\sqrt{\nu^\top\beta\nu}\right) - \sum_{i=1}^n \alpha_i + \sum_{i=1}^n \zeta_i(\alpha_i^2 - \nu_i)\right)$$

where ζ_i is the lagrange multiplier of the constraint $\alpha_i^2 \leq \nu_i$. Using first order conditions for optimality, $\frac{\partial L}{\partial \nu_i} = 0$ one obtains

$$\frac{\kappa}{2}(\nu^\top \beta \nu)^{-\frac{1}{2}} \beta \nu + \zeta = 0,$$
$$\sqrt{\zeta^\top \beta^{-1} \zeta} = \frac{\kappa}{2}.$$

Eliminating ν and noting that at optimality the constraint of (23) involving ζ is active, the proof is complete. ■

Note that the objective, in the previous theorem, is Lipschitz continuous, linear in ζ and *strongly concave* in α as long as $\overline{\mathbf{K}}$ is positive definite. Both ζ and α lie in a convex and compact set. In principle the FSS procedure applies and could be an interesting alternative to the SOCP procedure discussed in Bhadra et al. (2010).

5.5 Remarks

In this section we make some remarks on FSS algorithm and its suitability to the problem at hand.

A minimax problem $\min_{x\in X} \max_{y\in Y} \phi(x,y)$ can be approached as follows

$$\min_{x\in X} \left(g(x) = \max_{y\in Y} \phi(x,y) \right). \tag{24}$$

If $\nabla_x g$ were Lipschitz continuous, (24) could be solved at the rate $O(1/T^2)$ by the fast gradient algorithm for smooth convex minimization due to Nesterov (1983). However for our problem, g not necessarily possesses the desired smoothness. However, the situation still allows to achieve $O(1/T^2)$ convergence rate by applying the FSS algorithm to the saddle point reformulation of (24). The FSS algorithm is essentially a modification of the Mirror Prox (MP) algorithm presented in Nemirovski (2004). The prototype algorithm MP solves saddle point problem (24) with convex-concave and smooth (with Lipschitz continuous gradient) ϕ at the rate $O(1/T)$, with the hidden factor in $O(\cdot)$ depending on the distance from the starting point to the solution set of (5.1) (this distance is taken w.r.t. a norm $\|\cdot\|$ assembling $\|\cdot\|_X$, $\|\cdot\|_\mathcal{Y}$) and the Lipschitz constant of the gradient of ϕ (this constant is taken w.r.t. the conjugate norm $\|\cdot\|_*$). Now, when ϕ is strongly concave in y, the above convergence implies qualified convergence of the y-components y^t of approximate solutions to the y-component y_* of the saddle point of ϕ, so that eventually we know that $\|y^t - y_*\|_\mathcal{Y}$ is, say, twice smaller than (a priori upper bound R on) $\|y^1 - y_*\|_\mathcal{Y}$. When it happens, affinity of ϕ w.r.t. x (Assumption **A**.3) allows to rescale the problem and to restart MP *as if* we were working with a twice smaller domain than the original one, which results in $O(1/T)$ convergence *with reduced hidden factor.* In FSS, we iterate the outlined rescalings and restarts (this is where the stages come from), thus arriving at $O(1/T^2)$ convergence.

6. Prediction Rules and Error Metrics

Recall that in the classical SVM formulation the label of each observation is predicted by (1) where (α, b) is obtained by solving (2). As $K(\mathbf{x}, \mathbf{x}_i)$ is known the classifier predicts a unique label to each test example, $\mathbf{x}$. The quality of the classifier is measured by comparing the prediction with the actual label. In the problem setup considered here the kernel values are only approximately known

and hence the prediction process is not as straightforward as in the case of SVM. In this section we introduce several prediction rules and evaluation measures.

6.1 Prediction Rules

For each observation, $\mathbf{x}_t$, the values, $K(\mathbf{x}_t,\mathbf{x}_i)$, are only approximately known and lies in an uncertainty set (10). Given any (α,b), application of decision rule (1) in this setting is not clear. To this end we propose two heuristics for modelling the prediction process.

The essence of robust classification is that for any choice of $K(\mathbf{x}_t,\mathbf{x}_i)$, governed by (10), the classifer will give the same label. In other words the classifier is robust to uncertainty in the value of K.

The simplest case would be to use $K(\mathbf{x}_t,\mathbf{x}_i)=\overline{K}(\mathbf{x}_t,\mathbf{x}_i)$, which when used in conjunction with (1) gives the following labelling rule:

$$\bar{y}_t^{pr} = sign\left(\sum_i y_i\alpha_i\overline{K}(\mathbf{x}_t,\mathbf{x}_i)+b\right), \tag{25}$$

which, in the sequel, will be referred to as the *nominal rule*. A more comprehensive process of labelling would involve evaluating all possible choices of $\mathbf{K}$ and see how robust the resultant prediction is. Let $\{\eta^{t1},\ldots,\eta^{tR}\}$, be R uniformly drawn instances of $\{\eta^t\in\mathbb{R}^L|\|\eta\|_p\leq\kappa,\eta_l\geq 0\}$. Each choice of η^t generates a realization of kernel of the form

$$K_t(\mathbf{x}_t,\mathbf{x}_i)=\overline{K}(\mathbf{x}_t,\mathbf{x}_i)+\sum_{l=1}^{L}\eta_l^t K_l(\mathbf{x}_t,\mathbf{x}_i).$$

One option for arriving at a label would be to take the *majority vote* with the above kernel function,

$$y_t^{pr}=\text{sign}\left(\sum_{s=1}^{R}y_t^s\right),\ y_t^s=sign\left(\sum_i\alpha_i y_i K_{ts}(\mathbf{x}_t,\mathbf{x}_i)+b\right). \tag{26}$$

Once we have defined these two prediction rule , namely *majority vote* and the *nominal rule*, it is important to devise measures for evaluating the resultant classifiers.

6.2 Error Metrics

Consider a test data set $\mathcal{D}=\{(\mathbf{x}_t,y_t)\ t=1,\ldots,n_{tst}\}$, where, y_t is the true label for observation $\mathbf{x}_t$. We wish to measure the performance of the classifier (26) or (25) on this test data set for a given choice of (α,b).

For the nominal classifier (25) the usual $0/1$ loss works well and we define

$$\textbf{NominalErr}(NE)=\frac{\sum_{t=1}^{n_{tst}}\mathbf{1}_{(\bar{y}_t^{pr}\neq y_t)}}{n_{tst}}.$$

Similarly for the majority vote based classifier (26), we define

$$\textbf{MajorityErr}(ME)=\frac{\sum_{t=1}^{n_{tst}}\mathbf{1}_{(y_t^{pr}\neq y_t)}}{n_{tst}} \tag{27}$$

where, y_t is the true label for $\mathbf{x}_t$. However as noted before a robust classifier is expected to ensure that $y_t^s=y\ ,s=1,\ldots,R$ is equal for all s. To capture this notion of robustness, we propose another error

Formulation	Required information
Nominal – **SVM** (2)	$\overline{\mathbf{K}}$
RSVM (5)	$\overline{\mathbf{K}}$ and Support
USSVM$_{\mathbf{SOCP}}$(13) **USSVM$_{\mathbf{MN}}$**(21)	$\overline{\mathbf{K}}$ and $\mathcal{K} = \{\mathbf{K}_1, \ldots, \mathbf{K}_L\}$ (set of valid kernels)

Table 1: Summary of Various formulations and associated Information required.

measure (**RobustErr**) which counts the fraction of data points in $\mathcal{D}$ for which there is atleast one error among R observations. More precisely

$$\textbf{RobustErr}(RE) = \frac{\sum_{t=1}^{n_{tst}} \mathbf{1}_{(\exists s | y_t^s \neq y_t)}}{n_{tst}} \tag{28}$$

is a more appropriate measure than (ME) to evaluate *robustness*.

In the following section, we report experimental results for the algorithms developed in this paper and benchmark them against the state of the art with respect to the above mentioned metrics.

7. Experimental Evaluation

This section presents experimental evaluation of the formulations, namely **Nominal** – **SVM**, **USSVM$_{\mathbf{SOCP}}$**, **USSVM$_{\mathbf{MN}}$** and **RSVM**. The **Nominal** – **SVM** formulation is the usual SVM formulation (2) with the *nominal* kernel. The minimax problem (12) when solved by the FSS procedure will be referred as **USSVM$_{\mathbf{MN}}$**. The solution of the SOCP (13) will be referred as **USSVM$_{\mathbf{SOCP}}$**. Though as discussed before the setup of Bhadra et al. (2010) does not apply here but for sake of completeness we have also included a comparison with **RSVM**. A brief summary of the formulations is presented in Table 7.[4]

In particular it would be interesting to explore the following questions.

1. Comparison of **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** against the non-robust **Nominal** – **SVM**.

2. Convergence and scalability of **USSVM$_{\mathbf{MN}}$** algorithm

The section is organized as follows. We begin by a brief description of data sets in Section 7.1. A comparative study of robustness is presented in Section 7.2. The first issue is discussed in Section 7.3. The experimental verification of the convergence rate of FSS procedure is discussed in Section 7.4. Next the scalability of **USSVM$_{\mathbf{MN}}$** over **USSVM$_{\mathbf{SOCP}}$** is discussed in Section 7.5.

7.1 Data Sets

We created synthetic data sets to test the generalization and robustness properties of the proposed formulations. Additionally we also have empirically tested them on protein structure data. We describe them below.

4. Relevant data and scripts are available at `http://mllab.csa.iisc.ernet.in/~sahely/uncertainkernel.html`.

7.1.1 SYNTHETIC DATA SETS AND KERNEL FUNCTIONS

It is important to evaluate the effect of robustness on wide variety of situations. To this end we use the following data generation mechanism suited for binary classification problems.

Choose $d \sim Unif(2,100)$, where $Unif(n_1,n_2)$ is the uniform distribution over all integers from n_1 to n_2. For such a choice of d create a mixture distribution consisting of 4 Gaussian distributions, $N(\mu,\Sigma)$, with diagonal covariance matrix. The mean of each Gaussian distribution, $\mu \in \mathbb{R}^d$, is determined by independently choosing $\mu_j \sim Uniform(-5,5)$ $j = 1,\ldots,d$. Each diagonal entry of the diagonal matrix Σ is independently drawn from $Uniform(0,5)$. We assigned labels to the centers of each Gaussian distribution according to the $sign(w^\top x)$ where $w \in \mathbb{R}^d$ is a random vector with $\|w\| = 1$. A data set of $2N$ points was generated as follows. First a set of N points corresponding to the positive class was generated by sampling N observations from the Gaussian mixture distribution consisting of positively labeled mixture components. The set of N points were generated from negatively labeled mixture components.

We study the problem of robust classification when the kernel values are not available but are governed by (10). Next we describe the construction of **base** kernels needed in (10). A linear kernel will be very effective for any data set D, created by the data generation process described. Given $D = \{(\mathbf{x}_i, y_i)|i = 1,\ldots,N\}$ we define $\overline{\mathbf{K}} = \mathbf{x}_i^\top \mathbf{x}_j$. Furthermore, L kernels were simulated as follows; $\mathbf{K}_l = \overline{\mathbf{K}} + Z^l Z^{l\top}$, where Z^l_{ij} were generated using: a) **Gaussian (0,1)** b) **Uniform [-1,1]** c) centered **Beta (0.5,0.5)** distributions. After that, the generated values were multiplied by a random $l_{ij} \sim \mathbf{Uniform}(\mathbf{0}, \mathbf{0.05}|\overline{\mathbf{K}}_{\mathbf{ij}}|)$. This leads to L valid positive semidefinite kernels for each of the distribution, namely Gaussian, Uniform, or Beta.

We will denote by $\mathbf{D_G}(S,N,L)$, the set of S data sets, $\{D_1,\ldots,D_S\}$. Each data set was created by the data-generation mechanism discussed earlier and has N examples per class with L kernels generated by the Gaussian distribution. Similarly $\mathbf{D_U}(S,N,L)$ and $\mathbf{D_B}(S,N,L)$ will correspond to the Uniform and Beta distribution.

7.1.2 RESOLUTION-AWARE PROTEIN STRUCTURE CLASSIFICATION

We have used a data set based on the SCOP (Murzin et al., 1995) 40% sequence non-redundant data set taken from Bhadra et al. (2010). The data set has 15 classes (SCOP superfamilies), having 10 structures each. The names of these superfamilies are reported in Appendix D. To study the effect of robustness we studied the classification problem on all possible pairs, which gave rise to 105 data sets in total. Each data set D can be thought of $D = \{P_i, y_i, r_i|i = 1,\ldots,n\}$ where P_i is the nominal structure described in (3) with label y_i. Incorporation of resolution information r_i leads to uncertainty sets $U(P_i)$ (see (4)). Using the kernel function described in Bhattacharya et al. (2007) and assuming that the resultant uncertainty in kernel values obey (10) the kernel functions $\overline{\mathbf{K}}, \mathbf{K}_l$ are computed by the procedure outlined in Section 4.3.

As described in the Section 3, the uncertainty set imposed by **RSVM** maynot always be appropriate. However we still provide a comparison to the robust formulations described in this paper for the sake of completion. In the setting of the paper a set of kernel matrices $\mathcal{K} = \{\mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_L\}$ are specified. The formulation **RSVM** needs support information, (see Table 7), which could be extracted as follows

$$\overline{\mathbf{K}} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{K}_l \;\; a_{ij} = \min_l(\mathbf{K}_{l_{ij}}) \;\; b_{ij} = \max_l(\mathbf{K}_{l_{ij}}).$$

The algorithms **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** and **RSVM** have been implemented in Matlab with the help of Sedumi[5] (Sturm, 1999). We have used libSVM[6] as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.66 GHz processors and 16GB of RAM. The **RSVM** implementation uses a Quasi Newton procedure outlined in Bhadra et al. (2010). As it often gets stuck in a local minima we have used multiple starting points. All results on **RSVM** reported here corresponds to the results of the best starting point among 100 randomly selected starting points according to **RSVM** objective function.

7.2 Comparison of Robustness

We begin by studying the effect of robustness on synthetic data. In the proposed model of uncertainty the parameter κ plays a very important role. When $\kappa = 0$, then there is no uncertainty and as it increases the uncertainty becomes more pronounced. The utility of robust formulations would become clear as κ is increased. One would like to experimentally verify the fact that indeed this is the case. To this end we conducted the following experiment.

We created data sets $\mathbf{D_G}(S,N,L), \mathbf{D_U}(S,N,L), \mathbf{D_B}(S,N,L)$ with $S = 10, N = 250, L = 200$, as described in Section 7.1. We have performed 5-fold cross-validation on all the 10 data set. Here we vary $\kappa \in \{0.1, 0.2, 0.3, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 5\}$ with $R = 100$. In Figure 2, we have plotted the RobustErr (28) averaged over all 10 data sets for various distributions and choices of κ. Though we get similar result for few different values of C here we have reported the results for the value of $C = 100$. Again $\mathbf{D_G}$ will refer to the Gaussian distribution, $\mathbf{D_U}$ refers to the uniform case, and $\mathbf{D_B}$ refers to the Beta distribution.

The results of the experiment were as follows. It can be seen from Figure 2 that at $\kappa = 0$, the RE for both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** are exactly same as that of **Nominal** $-$ **SVM**. It confirms the fact that at $\kappa = 0$, **USSVM$_{\mathbf{SOCP}}$**(**USSVM$_{\mathbf{MN}}$**) is equivalent to **Nominal** $-$ **SVM**, as there is no uncertainty. Figure 2 shows that, with the increase of uncertainty in the test examples, the RobustErr(28) for **Nominal** $-$ **SVM** increases substantially when compared to **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** on all the 3 data sets. This shows that, non-robust classifiers, for example, SVM, are unable to handle uncertainty compared to the proposed robust classifiers. Also as expected both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** are equivalent and so on the test one they exhibit similar performance.

7.3 Comparison of Generalization Error

In this section we compare the error measures, RE (28) and ME (27).

We again use the same data sets described in the previous subsection. For all the metrics, we have performed 5-fold cross-validation on all the 10 data sets corresponding to each distribution. The hyper-parameters (C and ε) for each classifier, were chosen using a grid search mechanism from the set $C = \{0.1,\ 1,\ 5,\ 10,\ 50,\ 100,\ 200,\ 500\}$ and $\varepsilon = \{0.05 + 0.05step | step = 0, \ldots, 9\}$. For each metric, the cross-validation accuracy, $100(1 - ErrorMeasure)\%$, averaged over 10 data sets for various distributions, are reported in Figure 3. Note that $\mathbf{D_G}$ refers to the Gaussian case, $\mathbf{D_U}$ refers to Uniform case and $\mathbf{D_B}$ refers to the Beta case. The parameter κ was set to 1.

The results were as follows. All the formulations achieved an accuracy of 90% when NE was used as a error measure. From Figure 3 we see that both **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** beats **RSVM** in terms of RE indicating that **RSVM** is not well suited for the uncertainty sets considered

5. Sedumi can be found at `http://sedumi.ie.lehigh.edu/`.
6. LibSVM can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

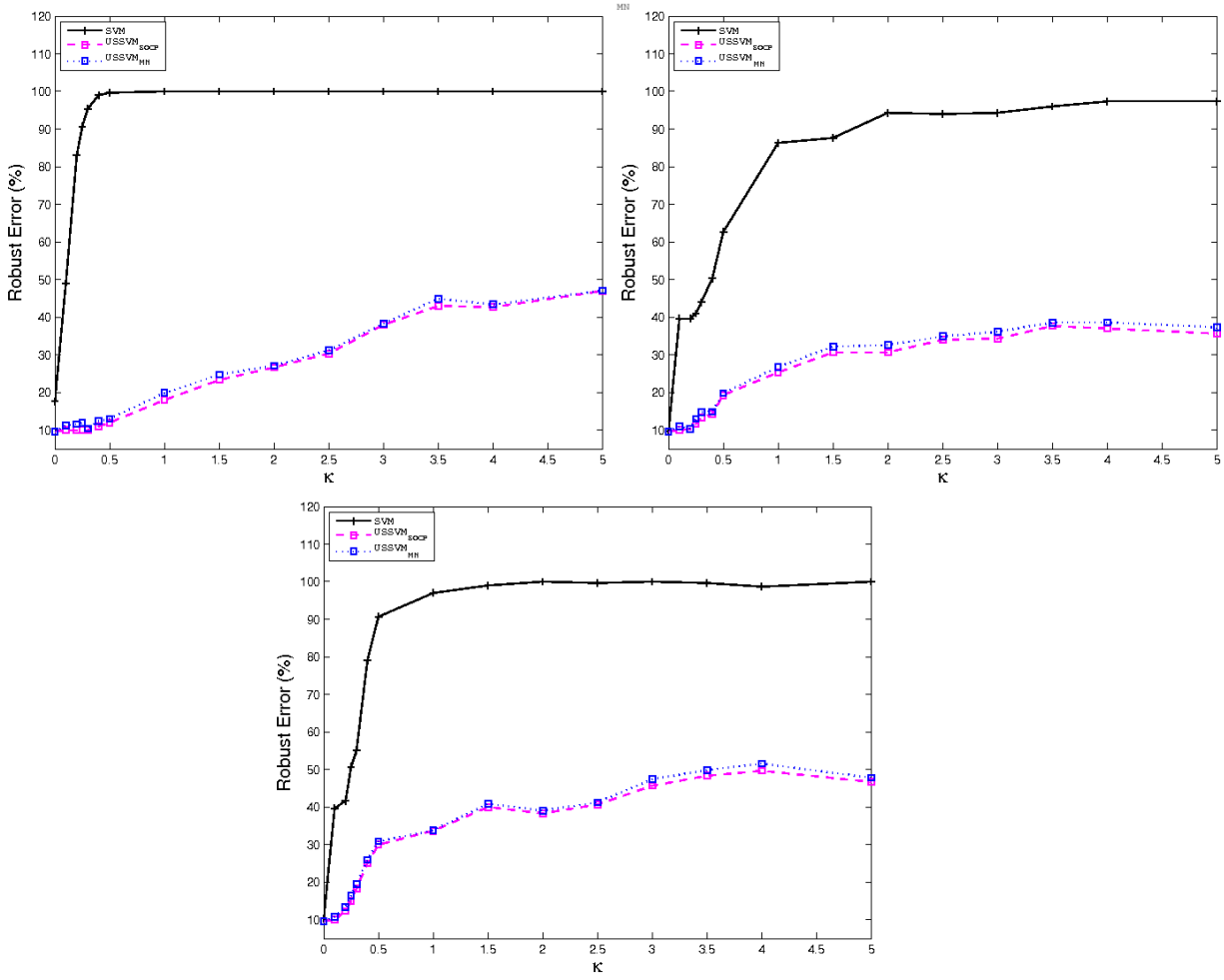

Figure 2: Plot of RE for distributions (clockwise starting from top left)$\mathbf{D_G}, \mathbf{D_U}, \mathbf{D_B}$ with varying κ. Formulations compared are $\mathbf{USSVM_{MN}}, \mathbf{USSVM_{SOCP}}$ and **Nominal** – **SVM**. The legend SVM refers to **Nominal** – **SVM**

here. When we use ME, which is not as conservative as RE the gap narrows. This experiment demonstrates that in the presence of uncertainty the performance of extremely accurate classifiers suffer drastically but the proposed robust formulations fare much better in handling uncertainty. In addition, Figure 4 shows that the average training time of $\mathbf{USSVM_{SOCP}}$ is same as that of **RSVM** but $\mathbf{USSVM_{MN}}$ is 10 times faster than both of them, even for these small scale data sets (200 training datapoints per class).

7.4 Verification of Convergence of FSS Algorithm

In this section, we have experimentally verified that the proposed saddle point based algorithm has $O(\frac{1}{M_s^2})$ convergence rate (see (20)). Recall that M_s is the actual number of steps, which one can consider as iterations.

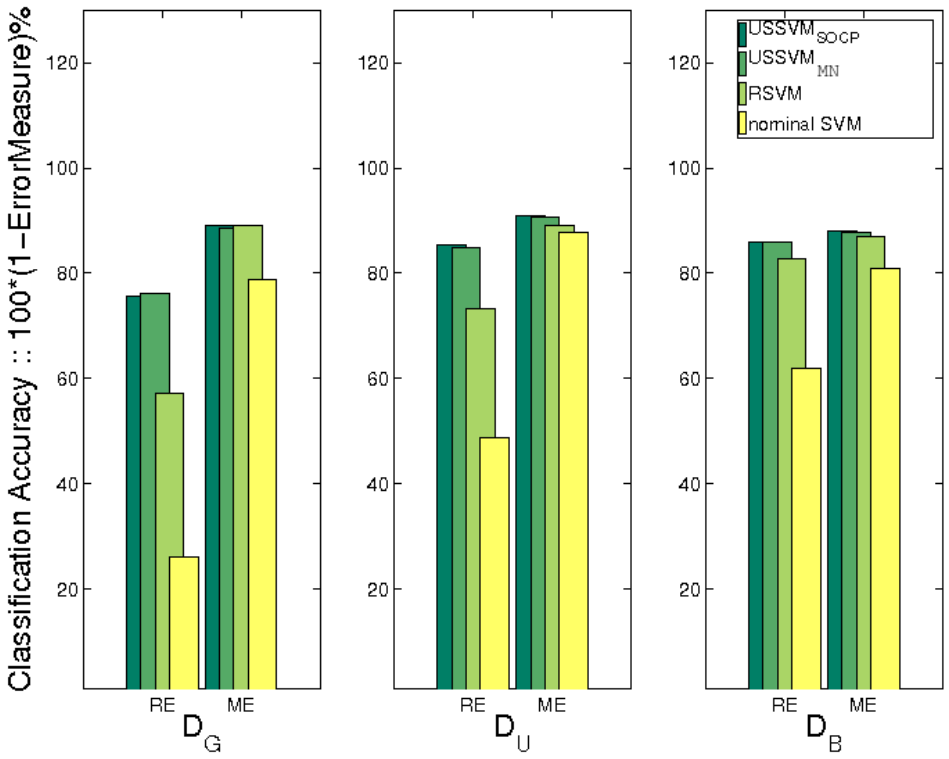

Figure 3: Cross-validation accuracy (%) obtained with $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$, **RSVM** and $\mathbf{Nominal-SVM}$ using RE (28)and ME (27). All values reported here are $100(1-Errormeasure)\%$

We report results on Data Set $\mathbf{D_U}(1,N,5)$ where $N \in \{50,500\}$, and C was chosen to be 1 for $N = 50$ and similarly for $N = 500$ it was fixed to be 10.

Figure 5 shows the convergence rate of the Saddle Point based algorithm for **USSVM** formulation. The x-axis and the y-axis denote $log_{10}(M_s)$ and $log_{10}(\varepsilon_{sad})$ respectively. All the points on the graph indicate the "end" of the s^{th} step and circled points indicate the "end" of the $s*$ step. When $s > s^*$, ideally the graph should be a straight-line with slope less than -2 and one can observe the same in Figure 5. On the other hand, for $s < s*$ rate of decrease in ε_{sad} is much slower than the case in $s > s*$.

7.5 Scalability of $\mathbf{USSVM_{MN}}$

In this section we study the relative performace of $\mathbf{USSVM_{MN}}$ versus $\mathbf{USSVM_{SOCP}}$ on large data sets. We also verify the convergence criteria of proposed $\mathbf{USSVM_{MN}}$.

In the $\mathbf{USSVM_{MN}}$ algorithm the number of stages and the number of iterations inside one stage do not depend on the number of data points, see Section 5.2. In each iteration we need to solve two gradient projection type problems. It appears that they are extremely cheap to compute; in one case there is a closed form solution, while in the other case we could solve it by a line search algorithm. Both $\mathbf{USSVM_{SOCP}}$ and $\mathbf{USSVM_{MN}}$ solves the same problem but the computation required may dif-

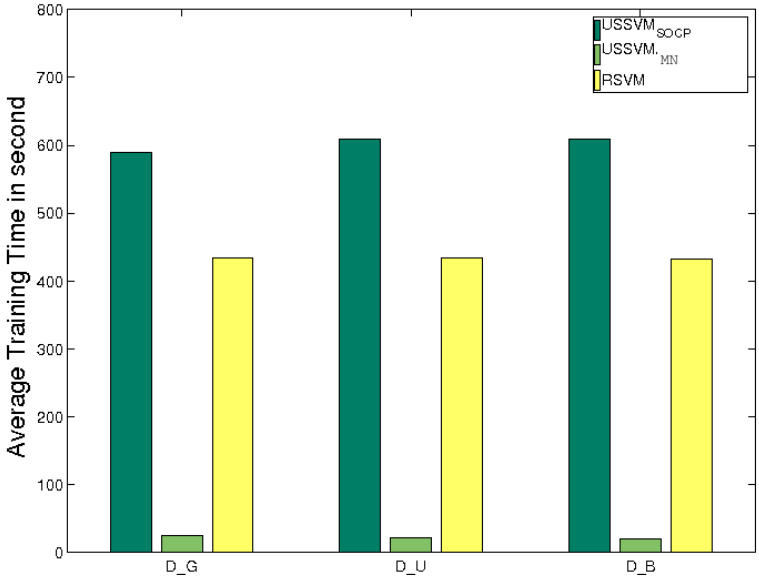

Figure 4: Average Training Time in Seconds obtained with **USSVM$_{\mathbf{SOCP}}$**, **USSVM$_{\mathbf{MN}}$**, and **RSVM**.

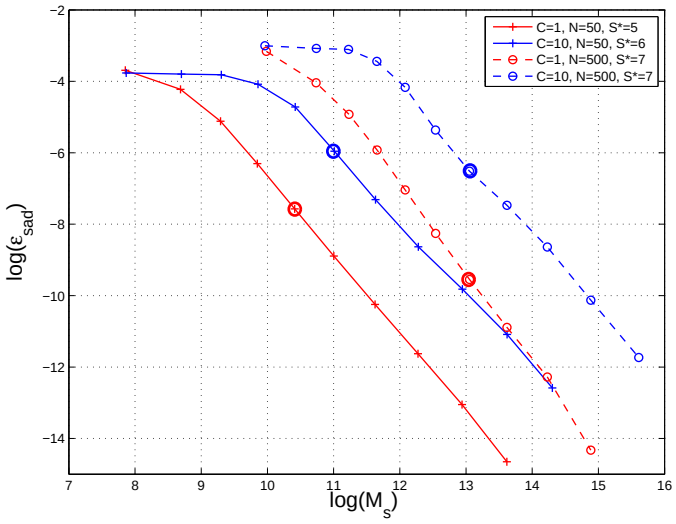

Figure 5: Rate of convergence for Saddle point based algorithm

fer significantly. To this end we have compared the training times for **USSVM$_{\mathbf{SOCP}}$** and **USSVM$_{\mathbf{MN}}$** with increasing the number of training data points (N). For this experiment, we have used following data set.

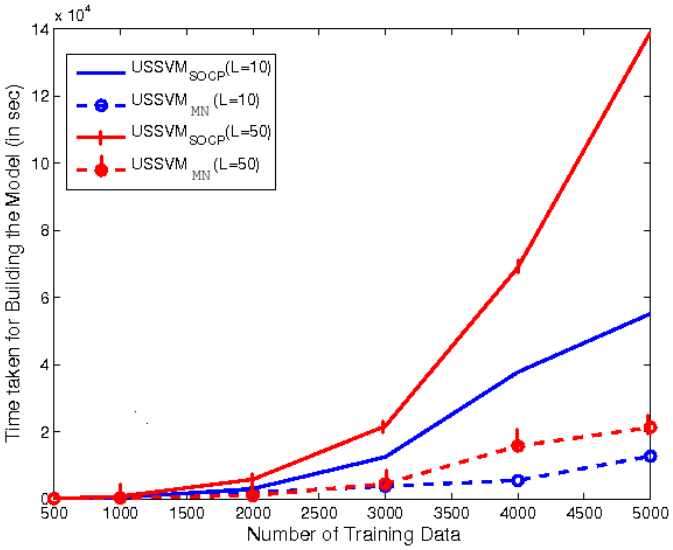

Figure 6: Training time for **USSVM$_{\mathbf{SOCP}}$ USSVM$_{\mathbf{MN}}$** with $N = [500, 1000, 2000, 3000, 4000, 5000]$ and $L = [10, 50]$

We have used synthetic data set generated similar to D_U (please see Section 7.1) with number of data points in each class are $\{250, 500, 1000, 1500, 2000, 2500\}$, where $L = \{10, 50\}$. The values of $R = 100$, $C = 10$ and $\kappa = 1$ were used.

Figure 6 shows training time (in sec) for varying N. One can observe that, with the increase of N, the training time for **USSVM$_{\mathbf{SOCP}}$** increases very steeply compared to the training time for **USSVM$_{\mathbf{MN}}$**. As expected, training time for **USSVM** in general increases with the increase of number of uncertain kernels (L). As an example, to build a robust classifier with only 3000 data points, **USSVM$_{\mathbf{SOCP}}$** needs more than **5 hours** while **USSVM$_{\mathbf{MN}}$** completes within **20 minutes**. This concludes that, to build a robust classifier with a medium scale of data (even more than 1000) the saddle point based algorithm is much more effective then a Quadratic Conic Program based formulation.

7.6 Discussion of Experimental Results

The results on the synthetic experiments show that **USSVM$_{\mathbf{SOCP}}$**, **USSVM$_{\mathbf{MN}}$** performs better than **RSVM** in terms of generalization as measured by various error measures. All the three formulations are more robust than **Nominal** $-$ **SVM**. It is also demonstrated that **USSVM$_{\mathbf{MN}}$** is much more scalable than **USSVM$_{\mathbf{SOCP}}$**. Even to build a robust classifier with 3000 data points, **USSVM$_{\mathbf{SOCP}}$** needs more than **5 hours** while **USSVM$_{\mathbf{MN}}$** completes within **20 minutes**.

7.7 Resolution-aware Protein Structure Classification

This section presents experimental results for comparing the robustness performance of the proposed **USSVM**, with the existing **RSVM** formulation.

Error Measure	$\mathbf{USSVM_{SOCP}}$	$\mathbf{USSVM_{MN}}$	**RSVM**	**Nominal − SVM**
Robust Error(RE)	25.14	24.95	20.95	10.38
Majority Error(ME)	85.70	83.36	81.12	71.01
Nominal Error(NE)	80.76	79.67	80.38	71.57

Table 2: Comparison of $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$ **RSVM** and **Nominal − SVM** using accuracy measures, $100(1-ErrorMeasure)\%$, where Error measures are defined in Section 6.2. Table shows average accuracy for all 105 **one-vs-one** classification problems

The data set is described in Section 7.1. The experimental methodology follows "one-vs-one" classification setting with all 15 classes of protein structures. Leave-One-Out (LOO) cross validation using SVM, RSVM and USSVM was performed on all 105 of such classification problem. In all cases we report accuracy, computed as $100(1-ErrorMeasure)\%$.

Let $\mathcal{D} = \{(P_i, r_i, y_i)\}$ be a protein structure data sets where P_i is the set of coordinates of i^{th} protein structure obtained from Astral[7] database, where r_i is the corresponding resolution information obtained from the PDB, and y_i is the class label. Using resolution information, we generated a set of perturbed structures $Q_i = \{P_i^1, \ldots, P_i^L\}$ for each P_i as follows. For each atom p_{i_a} of P_i generated structure P_i^s with coordinates of atoms as $p_{i_a}^l = p_{i_a} + u$ and $u \sim U(\frac{-r_i}{2}, \frac{r_i}{2})$. One can create a set of uncertain kernels, where $K(p,p')$ is a kernel function computed between two protein structures $p \in Q_i$ and $p' \in Q_j$. For our experiments, we have generated a set of kernels consisting of $L = 50$ base kernels. Denoting the kernel matrices by $\{\mathbf{K_1}, \ldots, \mathbf{K_L}\}$ the uncertainty set is defined as $\mathcal{E}(1)$ (see (11)) with $\overline{\mathbf{K}} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{K_l}$ and $\kappa = 1$. Given the base kernels the prediction is implemented, as reported in Section 6, with $R = 100$ and $\kappa = 1$. For the purpose of our comparison, we have used weighted pair-wise distance substructure kernel (Bhattacharya et al., 2007). These kernels are purely based on protein structure (specially position of c^α). Please refer to Appendix C for details. For **RSVM**, we compute the following,

$$\overline{K}_{ij} = \overline{\mathbf{K}}_{ij},\ a_{ij} = \min_{p \in Q_i, p' \in Q_j} K(p,p'), b_{ij} = \max_{p \in Q_i, p' \in Q_j} K(p,p').$$

7.7.1 RESULTS ON PROTEIN STRUCTURE CLASSIFICATION

Table 2 and Table 3 report results for **RSVM**, $\mathbf{USSVM_{MN}}$ and **Nominal − SVM** (SVM with kernels based on *nominal* protein structure reported in PDB files) using both standard and robust error measures defined in section 6.2 in the Leave-One-Out (LOO) procedure. Hyper-parameters (C and/or ε) for **RSVM** and C for $\mathbf{USSVM_{MN}}$ and **Nominal − SVM** were tuned separately using the grid search mechanism. As this is a 15 class classification problem and we followed a "one-vs-one" setting, we have reported average accuracy of all 105 classifiers (Table 2). We have also provided a list of a few individual classes (Table 3) where *USSVM* performed significantly better than RSVM. The results are presented in the form of a histogram of performance differences (%) of *USSVM* against *RSVM* and *SVM* obtained by using RE (28) in Figure 7.

It is clear that **RSVM**, and **USSVM** perform significantly better than their non-robust counterparts, both in terms of Accuracy (measured by MajorityErr) and Robustness (measured by RobustErr). This result indicates that, the use of resolution information improves the overall classifica-

7. Astral can be found at `http://astral.berkeley.edu`.

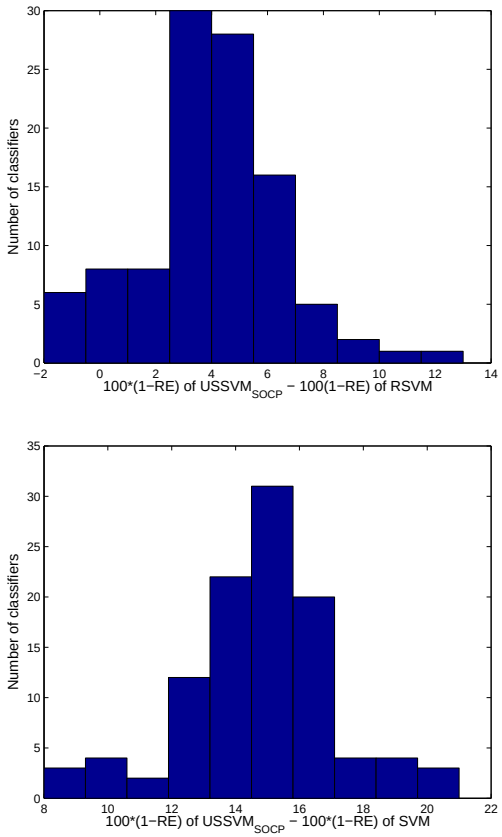

Figure 7: Histogram of performance differences (%) between $\mathbf{USSVM_{SOCP}}$ and **RSVM** is shown in the top figure. The bottom figure corresponds $\mathbf{USSVM_{SOCP}}$ and **SVM**

tion accuracy. In fact, **USSVM** even beats **RSVM** in terms of robustness. Note that, for more than 50% classification problem accuracy of $\mathbf{USSVM_{SOCP}}$ is more than 5% of that of **RSVM** in terms of Robustness. For few classes difference in accuracy was more than 10% (see Table 3). Moreover, this performance difference increases while comparing **USSVM** against **Nominal** – **SVM**. For more than 60% classification problem accuracy of $\mathbf{USSVM_{SOCP}}$ is more than 15% of that of **Nominal** – **SVM** in terms of robustness and notably for almost all the cases the margin was more than 10% in term of RobustErr.

8. Conclusion

We studied the problem of designing robust classifiers when the kernel matrices are uncertain. The chance constraint model proposed in Bhadra et al. (2010) made important progress on this problem but it had an important theoretical flaw. It did not constitute a valid model of uncertainty but instead a relaxed version of the original problem. This led to non-convexity and local minima problems. Instead of a chance constraint approach we follow the robust optimization community and advocate a geometric approach. The approach proposed here not only defines a valid model of uncertainty,

Error Measure	$USSVM_{SOCP}$	$USSVM_{MN}$	$RSVM$	**Nominal – SVM**
	c.66.1 vs c.68.1			
Robust Error	31.00	31.00	18.00	11.00
Majority Error	56.00	65.00	50.00	50.00
Nominal Error	50.00	55.00	40.00	50.00
	c.37.1 vs c.55.3			
Robust Error	23.00	23.00	12.00	6.00
Majority Error	60.00	60.00	55.00	51.00
Nominal Error	55.00	55.00	35.00	55.00
	d.58.4 vs c.108.1			
Robust Error	21.00	21.00	12.00	6.00
Majority Error	64.00	70.00	65.00	60.00
Nominal Error	60.00	65.00	60.00	55.00
	c.66.1 vs c.108.1			
Robust Error	27.00	27.00	18.00	12.00
Majority Error	58.00	55.00	60.00	50.00
Nominal Error	55.00	55.00	45.00	55.00
	c.55.1 vs c.2.1			
Robust Error	27.00	27.00	19.00	10.00
Majority Error	70.00	65.00	70.00	60.00
Nominal Error	70.00	55.00	60.00	55.00
	c.66.1 vs d.58.4			
Robust Error	33.00	33.00	25.00	18.00
Majority Error	72.00	65.00	70.00	55.00
Nominal Error	50.00	50.00	65.00	70.00
	b.18.1 vs b.80.1			
Robust Error	23.00	23.00	15.00	6.00
Majority Error	65.00	70.00	63.00	58.00
Nominal Error	60.00	60.00	60.00	50.00
	c.55.3 vs c.55.1			
Robust Error	28.00	25.00	20.00	10.00
Majority Error	88.00	70.00	80.00	65.00
Nominal Error	60.00	60.00	85.00	60.00
	c.66.1 vs d.92.1			
Robust Error	29.00	29.00	22.00	11.00
Majority Error	75.00	75.00	72.00	55.00
Nominal Error	65.00	60.00	70.00	65.00

Table 3: Comparison of $\mathbf{USSVM_{SOCP}}$, $\mathbf{USSVM_{MN}}$ **RSVM**, and **Nominal – SVM** using accuracy measures, $100(1 - ErrorMeasure)\%$, where Error measures are defined in Section 6.2. Table shows accuracy for **one-vs-one** classification problem among few classes. Description of superfamilies of SOCP are in Table 4

but leads to a tractable optimization problem namely a SOCP formulation. However SOCP formulations maynot be well suited for large scale problems. We show that the problem can be equivalently posed as a minmax problem and can be solved by saddle-point algorithm. We adapt the general purpose algorithm of Nemirovski (2004) for solving saddle point procedure to this problem. The algorithm proceeds in *stages*. We propose a novel special case of this algorithm, FSS, where the stepsize remains fixed per stage. The FSS algorithm has same order of $O(1/T^2)$ convergence, T being the number of iterates. This procedure is widely applicable which is a matter of independent study. We demonstrate its applicability to **RSVM** and to the problem at hand. The proposed algorithm, $\mathbf{USSVM_{MN}}$ combines FSS with suitable projection steps and is more scalable than the SOCP formulation, $\mathbf{USSVM_{SOCP}}$. Using a robust optimization based framework we pose the problem of classifier design as a minimax problem. The minimax procedure is solved by a novel FSS procedure which has $O(1/T^2)$ convergence. Empirical results show that $\mathbf{USSVM_{SOCP}}$ is indeed a robust alternative to uncertainty in the kernel matrices both on synthetic and real world data sets. Furthermore experimental results demonstrate that $\mathbf{USSVM_{MN}}$ indeed achieves the theoretical rate of convergence and is a scalable alternative to $\mathbf{USSVM_{SOCP}}$.

Acknowledgments

Chiranjib Bhattacharyya has been supported by a Yahoo! unrestricted Faculty gift. The authors Aharon Ben-Tal and A. Nemirovskii have been supported by the USA-Israel Bi-National Science Foundation(BSF) grant no. 2008302.

Appendix A. Proofs for the Saddle Point Algorithm

In this section we prove the convergence of FSS algorithm.

A.1 An Important Lemma

We start with the following Lemma.

Lemma 6 *Consider stage s of Algorithm FSS, and let*

$$\|(x,y)\| = \sqrt{\frac{\alpha_s}{\Omega_X}\|x\|_{\mathcal{X}}^2 + \frac{\beta_s}{\Omega_{\mathcal{Y}}R_s^2}\|y\|_{\mathcal{Y}}^2} \Rightarrow \|(\xi,\eta)\|_* = \sqrt{\frac{\Omega_X}{\alpha_s}\|\xi\|_{\mathcal{X},*}^2 + \frac{\Omega_{\mathcal{Y}}R_s^2}{\beta_s}\|\eta\|_{\mathcal{Y},*}^2}.$$

Function $\omega_s(\cdot)$ is a d.-g.f. for Z compatible with the norm $\|\cdot\|$, and

$$\begin{array}{ll}(a) & \operatorname{argmin}_Z \omega_s(\cdot) = (x_\omega, \bar{y_s}) \in Z_s \ \& \ \max_{Z_s}\omega_s(\cdot) - \min_{Z_s}\omega_s(\cdot) \le 1, \\ (b) & \forall (z,z' \in Z) : \|G(z) - G(z')\|_* \le \mathcal{L}_s\|z - z'\|. \end{array} \tag{29}$$

Proof Since $\omega_{\mathcal{Y}}(\cdot)$ is a d.-g.f. for $\mathcal{Y}$, this function is convex and continuously differentiable on the entire $\mathcal{Y}$, It follows that $\omega_s(x,y)$ is continuous convex function on Z, the set $Z^o := \{(x,y) : \partial\omega_s(z) \neq \emptyset\}$ is equal to $\{x : \partial\omega_X(x) \neq \emptyset\} \times Y$, and $\omega_s(x,y)$ admits a continuous on Z^0 selection of subgradient. All we need in order to complete the verification of the fact that ω_s is a d.-g.f. compatible with $\|\cdot\|$ is to verify that ω_s is strongly convex with modulus 1 w.r.t. the latter norm, which is immediate:

with $(x,y),(x',y') \in Z^o$ we have

$$\begin{array}{l}
\langle \omega_s'(x,y) - \omega_s'(x',y'), (x-x',y-y')\rangle \\
= \frac{\alpha_s}{\Omega_X}\langle \omega_X'(x) - \omega_X'(x'), x-x'\rangle + \frac{\beta_s}{\Omega_\mathcal{Y}} R_s^{-1}\langle \omega_\mathcal{Y}'([y-\bar{y}_s]/R_s) - \omega_\mathcal{Y}'([y'-\bar{y}_s]/R_s), y-y'\rangle \\
= \frac{\alpha_s}{\Omega_X}\langle \omega_X'(x) - \omega_X'(x'), x-x'\rangle \\
\quad + \frac{\beta_s}{\Omega_\mathcal{Y}}\langle \omega_\mathcal{Y}'([y-\bar{y}_s])/R_s) - \omega_\mathcal{Y}'([y'-\bar{y}_s]/R_s), [y-\bar{y}_s]/R_s - [y'-\bar{y}_s]/R_s\rangle \\
\geq \frac{\alpha_s}{\Omega_X}\|x-x'\|_X^2 + \frac{\beta_s}{\Omega_\mathcal{Y}}(\|y-y'\|_\mathcal{Y}^2/R_s^2) = \|(x-x',y-y')\|^2.
\end{array}$$

It remains to prove (29). Relation (29.*a*) is evident. To verify (29.*b*), let $z=(x,y), z'=(x',y') \in Z^o$, let $\Delta x = x'-x$, $\Delta y = y'-y$, $\xi = \|\Delta x\|_X\sqrt{\alpha_s/\Omega_X}$, $\eta = \|\Delta y\|_\mathcal{Y}\sqrt{\beta_s/\Omega_\mathcal{Y}}R_s^{-1}$, so that $\|z-z'\| = \|[\xi;\eta]\|_2$ by (16). We have

$$\begin{array}{l}
\|G_x(z)-G_x(z')\|_X \leq L_{xy}\|y-y'\|_\mathcal{Y},\ \|G_y(z)-G_y(z')\|_\mathcal{Y} \leq L_{xy}\|x-x'\|_X + L_{yy}\|y-y'\|_\mathcal{Y} \\
\text{[see (15)]} \\
\Rightarrow \|G(z)-G(z')\|_*^2 \leq \frac{\Omega_X}{\alpha_s}L_{xy}^2\|\Delta y\|_\mathcal{Y}^2 + \frac{\Omega_\mathcal{Y}R_s^2}{\beta_s}\left[L_{xy}\|\Delta x\|_X + L_{yy}\|\Delta y\|_\mathcal{Y}\right]^2 \\
= \frac{\Omega_X\Omega_\mathcal{Y}R_s^2L_{xy}^2}{\alpha_s\beta_s}\eta^2 + \frac{\Omega_\mathcal{Y}R_s^2}{\beta_s}\left[L_{xy}\sqrt{\Omega_X/\alpha_s}\xi + L_{yy}R_s\sqrt{\Omega_\mathcal{Y}/\beta_s}\eta\right]^2 \\
= \|M[\xi;\eta]\|_2^2, \\
M = \left[\begin{array}{c|c} & \sqrt{\frac{\Omega_X\Omega_\mathcal{Y}}{\alpha_s\beta_s}}R_sL_{xy} \\ \hline \sqrt{\frac{\Omega_X\Omega_\mathcal{Y}}{\alpha_s\beta_s}}R_sL_{xy} & \frac{\Omega_\mathcal{Y}R_s^2}{\beta_s}L_{yy}\end{array}\right] = \mathcal{L}_sN,\ N = \left[\begin{array}{c|c} & \sqrt{(1-\beta_s)/\beta_s} \\ \hline \sqrt{(1-\beta_s)/\beta_s} & (2\beta_s-1)/\beta_s\end{array}\right],
\end{array}$$

where the relations in the last line are readily given by (16). In view of this computation and the fact that $\|z-z'\| = \|[\xi;\eta]\|_2$, in order to verify (29.*b*) it suffices to show that the spectral norm of the symmetric matrix N is ≤ 1; since N is nonnegative due to $\beta_s \geq 1/2$, see (16), the latter task is exactly the same as verifying positive semidefiniteness of the matrix $I_2 - N$, which is immediate. ■

A.2 Proof of Proposition 2

Consider stage s, assuming that (I_s) take place. For $z \in Z^o$ and $\xi \in \mathcal{Z} := \mathcal{X}\times\mathcal{Y}$, let

$$\begin{array}{l}
V_z^s(u) = \omega_s(u) - \omega_s(z) - \langle \omega_s'(z), u-z\rangle : Z \to \mathbf{R}, \\
\mathrm{Prox}_z(\xi) = \mathop{\mathrm{argmin}}_{u\in Z}\{\langle \xi - \omega_s'(z), u\rangle + \omega_s(u)\} : \mathcal{Z} \to Z;
\end{array}$$

note that $\mathrm{Prox}_z(\xi)$ is well defined due to strong convexity of $\omega_s(\cdot)$. Our basic observation is as follows:

Lemma 7 Nemirovski, 2004, cf. Lemma 3.1 *Given $z \in Z^o$, $\xi,\eta \in E$, let $w = \mathrm{Prox}_z(\xi)$ and $z_+ = \mathrm{Prox}_z(\eta)$. Then for all $u \in Z$ it holds*

$$\begin{array}{ll}
\langle \eta, w-u\rangle \leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta, w-z_+\rangle - V_z^s(z_+) & (a) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta-\xi, w-z_+\rangle - V_z^s(w) - V_w^s(z_+) & (b) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \left[\frac{1}{2}\|\eta-\xi\|_*\|w-z_+\| - \frac{1}{2}\|z-w\|^2 - \frac{1}{2}\|z_+-w\|^2\right] & (c) \\
\leq V_z^s(u) - V_{z_+}^s(u) + \frac{1}{2}[\|\eta-\xi\|_*^2 - \|w-z\|^2] & (d).
\end{array} \tag{30}$$

Proof. By definition of $z_+ = \mathrm{Prox}_z(\eta)$ we have $\langle \eta - \omega_s'(z) + \omega_s'(z_+), u - z_+\rangle \geq 0$; rearranging terms and taking into account the definition of $V_v^s(u)$, we get (a). By definition of $w = \mathrm{Prox}_z(\xi)$ we have $\langle \xi - \omega_s'(z) + \omega_s'(w), z_+ - w\rangle \geq 0$, whence $\langle \eta, w - z_+\rangle \leq \langle \eta - \xi, w - z_+\rangle + \langle \omega_s'(w) - \omega_s'(z), z_+ - w\rangle$; replacing the third term in the right-hand side of (a) with this upper bound and rearranging terms, we get (b). (c) follows from (b) due to the strong convexity of ω_s implying that $V_v^s(u) \geq \frac{1}{2}\|u - v\|^2$, and (d) is an immediate consequence of (c). □

Applying Lemma 7 to $z = z_{t,s}$, $\xi = \tau_s G(z_{t,s})$ (which results in $w = w_{t,s}$) and $\eta = \tau_s G(w_{t,s})$ (which results in $z_+ = z_{t+1,s}$), we obtain due to (30) for all $u \in Z$:

$$\tau_s\langle G(w_{t,s}), w_{t,s} - u\rangle \leq V^s_{z_{t,s}}(u) - V^s_{z_{t+1,s}}(u) + \underbrace{\tfrac{1}{2}\left[\tau_s^2\|G(w_{t,s}) - G(z_{t,s})\|_*^2 - \|w_{t,s} - z_{t,s}\|^2\right]}_{\delta_{t,s}}.$$

Observe that $\delta_{t,s} \leq 0$ by (29.b) and by definition of τ_s (see (16)), we arrive at

$$\tau_s\langle G(w_{t,s}), w_{t,s} - u\rangle \leq V^s_{z_{t,s}}(u) - V^s_{z_{t+1,s}}(u)\ \forall u \in Z. \tag{31}$$

Let $u \in Z_s$, and let

$$\overline{\phi}_s(x) = \max_{y \in Y:\|y - \overline{y}_s\|_Y \leq R_s} \phi(x,y).$$

Summing up (31) over $t = 1, ..., N_s$, taking into account that $V^s_{z_{1,s}}(u) \leq 1$ due to $u \in Z_s$ by (29.a), that $V_z^s(u) \geq 0$, we get

$$\forall u \in Z_s : \frac{1}{N_s}\sum_{t=1}^{N_s}\langle G(w_{t,s}), w_{t,s} - u\rangle \leq A := \frac{1}{\tau_s N_s}.$$

On the other hand, setting $w_{t,s} = (x_{t,s}, y_{t,s})$, $u = (x,y)$ and noting that $z^s = (x^s, y^s) = \frac{1}{N_s}\sum_{t=1}^{N_s} w_{t,s}$ (see (18)), we have

$$\begin{array}{ll}
\frac{1}{N_s}\sum_{t=1}^{N_s}\langle G(w_{t,s}), w_{t,s} - u\rangle & \\
= \frac{1}{N_s}\sum_{t=1}^{N_s}\left[\langle \phi_x'(x_{t,s}, y_{t,s}), x_{t,s} - x\rangle + \langle \phi_y'(x_{t,s}, y_{t,s}), y - y_{t,s}\rangle\right] & \\
\geq \frac{1}{N_s}\sum_{t=1}^{N_s}\left[[\phi(x_{t,s}, y_{t,s}) - \phi(x, y_{t,s})] + [\phi(x_{t,s}, y) - \phi(x_{t,s}, y_{t,s})]\right] & (a) \\
= \frac{1}{N_s}\sum_{t=1}^{N_s}[\phi(x_{t,s}, y) - \phi(x, y_{t,s})] & \\
\geq \phi(\frac{1}{N_s}\sum_{t=1}^{N_s} x_t, y) - \phi(x, \frac{1}{N_s}\sum_{t=1}^{N_s} y_t) = \phi(x^s, y) - \phi(x, y^s) & (b)
\end{array} \tag{32}$$

(inequalities in $(a), (b)$ are due to the convexity-concavity of ϕ), so that (32) results in $\phi(x^s, y) - \phi(x, y^s) \leq A$ for all $(x,y) \in Z_s$. Taking supremum in $(x,y) \in Z_s$, we arrive at

$$\overline{\phi}_s(x^s) - \underline{\phi}(y^s) \leq A \leq \frac{\mathcal{L}_s}{N_s}, \tag{33}$$

where the concluding inequality follows from the definition of A due to $\tau_s = 1/\mathcal{L}_s$. Observe that the left-hand side in (33) is $\geq \overline{\phi}_s(x^s) - \mathrm{SadVal}$ (due to $\underline{\phi}(y^s) \leq \mathrm{SadVal}$), while the right-hand side in (33) is $\leq \frac{\theta R_s^2}{32}$ due to (16.e). Thus, (33) implies that

$$\overline{\phi}_s(x^s) - \mathrm{SadVal} \leq \frac{\theta R_s^2}{32}. \tag{34}$$

We claim that in fact $\overline{\phi}_s(x^s) = \overline{\phi}(x^s)$. Indeed, assuming the opposite, let $Y_s = \{y \in Y : \|y - \overline{y}_s\|_{\mathcal{Y}} \leq R_s\}$, and let $\overline{y} = \mathrm{argmax}_{y \in Y_s} \phi(x^s, y)$, so that $\overline{\phi}_s(x^s) = \phi(x^s, \overline{y})$. Since $\overline{\phi}(x^s) := \max_{y \in Y} \phi(x^s, y) > \overline{\phi}_s(x^s) := \max_{y \in Y_s} \phi(x^s, y)$ and Y_s is cut off Y by the inequality $\|y - \overline{y}_s\|_{\mathcal{Y}} \leq R_s$, we have $\|\overline{y} - \overline{y}_s\|_{\mathcal{Y}} = R_s$, while by (I_s) we have $\|y_* - \overline{y}_s\|_{\mathcal{Y}} \leq R_s/2$, whence, in particular, $y_* \in Y_s$ and $\|y_* - \overline{y}_s\|_{\mathcal{Y}} \geq R_s/2$. Since the function $\phi(x^s, y)$ is strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, and attains its maximum in $y \in Y_s$ at $\overline{y}$, while $y_* \in Y_s$, we have $\phi(x^s, y_*) \leq \phi(x^s, \overline{y}) - \frac{\theta}{2}\|y_* - \overline{y}\|_{\mathcal{Y}}^2 \leq \phi(x^s, \overline{y}) - \frac{\theta}{8} R_s^2$. It follows that $\mathrm{SadVal} = \min_{x \in X} \phi(x, y_*) \leq \phi(x^s, y_*) \leq \phi(x^s, \overline{y}) - \frac{\theta}{8} R_s^2 = \overline{\phi}_s(x^s) - \frac{\theta}{8} R_s^2$. The resulting inequality contradicts (34), and this contradiction shows that in fact $\overline{\phi}_s(x^s) = \overline{\phi}(x^s)$. Thus, (33) reads

$$\overline{\phi}(x^s) - \underline{\phi}(y^s) \leq \frac{\mathcal{L}_s}{N_s} \leq \frac{\theta R_s^2}{32}, \tag{35}$$

as required in (19) (recall that by construction $R_s = 2^{-s} R_0$). Finally, (35) implies that

$$\underline{\phi}(y_*) - \underline{\phi}(y^s) = \mathrm{SadVal} - \underline{\phi}(y^s) \leq \overline{\phi}(x^s) - \underline{\phi}(y^s) \leq \frac{\theta R_s^2}{32}; \tag{36}$$

since the function $\underline{\phi}(\cdot)$ is strongly concave, modulus θ w.r.t. $\|\cdot\|_{\mathcal{Y}}$, and attains its maximum over $y \in Y$ at y_*, we have $\underline{\phi}(y_*) - \underline{\phi}(y^s) \geq \frac{\theta}{2}\|y_* - y^s\|_{\mathcal{Y}}^2$, which combines with (36) to imply that $\|y_* - y^s\|_{\mathcal{Y}} \leq R_s/4 = R_{s+1}/2$; this is nothing but (I_{s+1}). Thus, we have proved that if (I_s) takes place, then Algorithm FSS ensures (19) and (I_{s+1}). Since (I_0) holds true by assumption, we conclude that (19) and (I_s) take place for all s. All remaining claims in Proposition are now straightforward. □

Appendix B. Projection on the SVM Constraint Set

This appendix discusses the projection step encountered in Section 5.3. We consider the following problem

$$y_+ = \mathrm{argmin}_{y \in Y} \left\{ -y^T q + \frac{1}{2}(y - \overline{y})^T (y - \overline{y}) \right\},$$

where $Y = \{y \in \mathbb{R}^n | 0 \leq y_i \leq C, \sum_{i=1}^n y_i s_i = 0\}$ is the SVM constraint set and $s_i \in \{1, -1\}$. When $\overline{y} \in Y$ the optimality conditions yield

$$y_{+i} = \left\{ \begin{array}{ll} 0, & \overline{y}_i + q_i + \nu s_i \leq 0 \\ C, & \overline{y}_i + q_i + \nu s_i \geq C \\ \overline{y}_i + q_i + \nu s_i & otherwise \end{array} \right\}.$$

Furthermore y_+ should satisfy $\sum_i y_{+i} s_i = 0$. It is easy to verify that $min(-\max_{i_+}(\overline{y}_i + q_i), \min_{i_-}(\overline{y}_i + q_i) - C) \leq \nu \leq max(C - \min_{i_+}(\overline{y}_i + q_i), \max_{i_-}(\overline{y}_i + q_i))$. Since the problem is feasible there exists at least one ν for which $\sum_i y_{+i} s_i = 0$ holds. We compute this by grid search. If there are more than one solution satisfying the constraint $\sum_i y_{+i} s_i = 0$ we choose the solution which yields a lower objective.

Appendix C. Kernel Functions for Protein Structures

Experiments on protein structures have been conducted with **Weighted Pairwise Distance Substructure Kernel** described in Bhattacharya et al. (2007). To make the paper self-contained we describe the kernel function in brief, for more details please see Bhattacharya et al. (2007).

Fix a positive integer l. A substructure N_{i_a} consists of l spatially nearest residues to the ath residue of protein P_i. The substructure kernel between two substructures N_{i_a} and N_{j_b} is defined as

$$K_{pds}(N_{i_a}, N_{j_b}) = \sum_{\pi \in \prod(l)} e^{\frac{-\|d_{i_a} - \pi(d_{j_b})\|^2}{\sigma^2}}$$

where d_{i_a} denotes set of pairwise distance betwen all possible pair of residues in N_{i_a}. Hence $d_i = \{d_i^1, \ldots, d_i^m\}$ where $m = l(l-1)/2$. The distance between any two residues a and b are computed by $\|\mathbf{c}_a - \mathbf{c}_b\|$, see (3), where $\prod(l)$ denote all possible permutations of l residues in the substructure. Finally the kernel function between two protein structures is defined as

$$K(P_i, P_j) = \sum_{a,b=1}^{n_i} \sum_{c,d=1}^{n_j} K_{pds}(N_{i_a}, N_{j_c}) K_{pds}(N_{i_b}, N_{j_d}) K_{norm}(i_a, i_b, j_c, j_d)$$

where $K_{norm}(i_a, i_b, j_c, j_d) = e^{-\frac{(\|\mathbf{c}_{i_a} - \mathbf{c}_{i_b}\| - \|\mathbf{c}_{j_c} - \mathbf{c}_{j_d}\|)^2}{\sigma^2}}$.

Appendix D. List of Superfamilies

We list below the Superfamilies studied in Section 7.7.

Superfamily	Description
b.18.1	Galactose-binding domain-like
b.29.1	Concanavalin A-like lectins/glucanases
b.30.5	Galactose mutarotase-like
b.40.4	Nucleic acid-binding proteins
b.80.1	Pectin lyase-like
c.2.1	NAD(P)-binding Rossmann-fold domains
c.37.1	P-loop containing nucleoside triphosphate hydrolases
c.55.1	Actin-like ATPase domain
c.55.3	Ribonuclease H-like
c.66.1	S-adenosyl-L-methionine-dependent methyltransferases
c.68.1	Nucleotide-diphospho-sugar transferases
c.69.1	alpha/beta-Hydrolases
c.108.1	HAD-like
d.58.4	Dimeric alpha+beta barrel
d.92.1	Metalloproteases (”zincins”), catalytic domain

Table 4: List of Superfamilies

References

A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001. ISBN 0-89871-491-5.

A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

A. Ben-Tal, S. Bhadra, C. Bhattacharyya, and J. S. Nath. Chance constrained uncertain classification via robust optimization. *Math. Program.*, 127(1), 2011.

S. Bhadra, J. Saketha Nath, A. Ben-Tal, and C. Bhattacharyya. Interval data classification under partial information: A chance-constraint approach. In *PAKDD*, pages 208–219, 2009.

S. Bhadra, S. Bhattacharya, C. Bhattacharyya, and A. Ben-Tal. Robust formulations for handling uncertainty in kernel matrices. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 71–78, 2010.

S. Bhattacharya, C. Bhattacharyya, and N. Chandra. Structural alignment based kernels for protein structure classification. In *Proceedings of 24th International Conference on Machine Learning (ICML)*, pages 73–80, 2007. URL `http://doi.acm.org/10.1145/1273496.1273506`.

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

C. Branden and John Tooze. *Introduction to Protein Structure*. Garland Publishing, second edition, 1999. ISBN 0815323050.

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

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

A. Juditski and A. Nemirovski. First order methods for large-scale convex optimization. In Sra S., S. Nowozin, and S. Wright, editors, *Optimization for Machine Learning*. The MIT Press, 2011.

J. Mercer. Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations. In *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, volume 209, pages 415–446, 1909.

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

A. Nemirovski. Prox-method with rate of convergence $o(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. Optim.*, 15:229–251, 2004.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady, 27*, pages 372–376, 1983.

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

J. Shawe-Taylor and N. Cristianini. *Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Engineering*, 11(9):739–747, 1998.

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

J. F. Sturm. Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

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