JMLR: Workshop and Conference Proceedings 13: 31-46
2nd Asian Conference on Machine Learning (ACML2010), Tokyo, Japan, Nov. 8–10, 2010.

Ellipsoidal Support Vector Machines

Michinari Momma * michinari.momma@sas.com
SAS Institute Japan

Kohei Hatano hatano@inf.kyushu-u.ac.jp
Kyushu University

Hiroki Nakayama h-nakayama@cj.jp.nec.com
NEC Corporation

Editor: Masashi Sugiyama and Qiang Yang

Abstract

This paper proposes the ellipsoidal SVM (e-SVM) that uses an ellipsoid center, in the version space, to approximate the Bayes point. Since SVM approximates it by a sphere center, e-SVM provides an extension to SVM for better approximation of the Bayes point. Although the idea has been mentioned before (Ruján (1997)), no work has been done for formulating and kernelizing the method. Starting from the maximum volume ellipsoid problem, we successfully formulate and kernelize it by employing relaxations. The resulting e-SVM optimization framework has much similarity to SVM; it is naturally extendable to other loss functions and other problems. A variant of the sequential minimal optimization is provided for efficient batch implementation. Moreover, we provide an online version of linear, or primal, e-SVM to be applicable for large-scale datasets.

Keywords: Bayes point machines, Support vector machines, Pegasos

1. Introduction

The most common interpretation of the support vector machines (SVMs) (Vapnik (1996); Schölkopf and Smola (2001)) is that SVM separates positive and negative examples by maximizing the margin that is the Euclidean distance between supporting hyperplanes of both examples. Another interpretation comes from a concept called the version space. The version space is a space of consistent hypotheses, or models with no error. SVM maximizes the inscribing hypersphere to find the center that is the SVM weight vector $\boldsymbol{w}$. Given the version space, the "sphere center" completely characterizes the SVM model. The Bayes point is a point through which all hyperplanes bisect the version space by half, and is shown to have better generalization ability theoretically and empirically (Herbrich et al. (2001); Ruján (1997)).

Attempts to approximately find the Bayes point have been done since the early studies of the version space and the Bayes point. SVM can be considered as an example. The Bayes point machines (BPM) (Herbrich et al. (2001)) uses a kernel billiard algorithm to find the center of mass in the version space. The analytic center machines (ACM) (Trafalis and Malyscheff (2002)) approximate the Bayes point by analytic points of linear constraints.

*. This work was done while the autor was at NEC Corporation

The idea of using an ellipsoid rather than a sphere has been mentioned in (Rujásn (1997)), although it was neither formulated nor implemented because of its projected high computational cost $O(n^{3.5})$. Then a billiard algorithm including BPM has been developed to alleviate the computational challenge. However, as we have seen in the history of SVM, seemingly expensive problem can be made efficient by exploiting special structures in the problem. Sequential minimal optimization (SMO) or decomposition methods are notable examples of such algorithms (Keerthi et al. (2001); Chen et al. (2005)). Furthermore, recent development of large scale linear SVMs (Shalev-Shwartz et al. (2007); Hsieh et al. (2008)) impressively improved the scalability of the quadratic optimization into practically linear order. Learning from the experience, we are encouraged to develop and study the method of ellipsoidal approximation to BPM, which we refer to as the ellipsoidal SVM (e-SVM).

The e-SVM formulation is based on that of SVMs. Advantages in formulating in such a way include possible adaptation of theoretical characterization and optimization methods developed for SVM and extensions to different loss functions. These advantages would not be realized if we stick to BPM that has to rely on sampling techniques that scale poorly on a large scale dataset; In BPM, even the soft boundary formulation is nontrivial and the kernel regularization is used after all.

As an attempt to solving the challenging the kernel batch e-SVM problem efficiently, we adopt the sequential minimal optimization (SMO). The modified SMO algorithm indeed shares many convenient features with that for SVM, such as the closed-form solution for the minimal problem, Karush-Kuhn-Tucker (KKT) condition violation check, etc. Although there should exist faster algorithms to solve depending on the type of problems, we decide to start from the simpler SMO algorithm and study how e-SVM compares against BPM and SVM.

Furthermore, we develop a stochastic gradient based method for solving online *linear*, or primal, e-SVM problem using the Online Convex Optimization (OCO) framework. OCO is initiated by Zinkevich (Zinkevich (2003)). OCO deals with the following online learning protocol between the learner and the adversary; At each trial t, the learner predicts a point $\boldsymbol{x}_t \in \mathcal{X}$, where $\mathcal{X}$ is a fixed bounded subset of $\mathbb{R}^n$. Then the adversary gives a convex function $f_t : \mathcal{X} \to \mathbb{R}$ and the learner incurs the loss $f_t(\boldsymbol{x}_t)$. The goal of the learner is, after T trials, to minimize the *regret*: $\sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \inf_{\boldsymbol{x} \in \mathcal{X}} \sum_{t=1}^{T} f_t(\boldsymbol{x})$. This framework captures other existing framework such as online learning with experts (Littlestone and Warmuth (1994); Vovk (1990)) from the viewpoint of convex optimization. OCO has been studied extensively these days. A popular application of OCO is Pegasos (Shalev-Shwartz et al. (2007); Shalev-Shwartz and Srebro (2008)) and is a state-of-the-art stochastic gradient descent solver for SVMs. The OCO framework is adopted in developing an online algorithm of e-SVM; Our algorithm outputs an approximation of the underlying problem with expected error is less than ε in $\tilde{O}(\frac{n \ln \frac{R}{\nu} + \frac{1}{\nu^2}}{\varepsilon})$ steps, where R is the maximum 2-norm of instances and ν is a parameter. Like Pegasos, the algorithm is efficient in terms of ε: the number of iteration is $\tilde{O}(\frac{1}{\varepsilon})$, neglecting other terms.

Section 2 formulates the e-SVM optimization problem. Section 3 describes the SMO algorithm adapted for the kernel e-SVM problem. Section 4 develops an online algorithm for linear e-SVM. Section 5 gives experimental results. Section 6 concludes the paper.

Notation: Throughout the paper, we assume that m data points $\boldsymbol{x}_i$ in n-dimensional space and the corresponding (target) label $y_i \in \{-1, 1\}$ are given. The bold small letters represent vectors and the capital letters represent matrices. The vector/matrix transpose is ^{T}. The kernel matrix is given by $\boldsymbol{K}$ with K_{ij} as its element. tr$A$ denotes the trace of a matrix $\boldsymbol{A}$. "s.t." in optimization problems means "subject to". I is an index set of m data points: $I \in \{1, \dots, m\}$. $\|\boldsymbol{A}\|_2$ denotes the matrix 2-norm and $\|\boldsymbol{x}\|_2$ the L2-norm of a vector $\boldsymbol{x}$.

2. Ellipsoidal support vector machine formulations

Beginning from reviewing the SVM formulation, we develop e-SVM problems by modifying it step-by-step.

The version space is a space of error zero models. For linear models, it is the error-zero subspace of weight vectors $\boldsymbol{w}$. The data points are considered as hyperplanes and the classification constraints are the feasible region that is a polyhedron. The problem of finding a maximum hypersphere inside the polyhedron can be formulated as follows:

$$\max_{\rho, \boldsymbol{w}, b} \quad \rho \quad \text{s.t.} \quad \frac{y_i \left(\boldsymbol{x}_i^T \boldsymbol{w} + b\right)}{\|\boldsymbol{x}_i\|_2} \geq \rho, \quad \|\boldsymbol{w}\|_2 \leq 1, \; i \in I$$

which corresponds to maximization of the minimum distance between the center and the hyperplanes, in the absence of the bias b. By allowing errors in the above problem, we can get a soft-margin version of the above problem.

$$\min_{\rho, \boldsymbol{w}, b, \boldsymbol{\zeta}} \quad -m\rho + 1/\nu \sum_{i=1}^{m} \zeta_i \quad \text{s.t.} \quad y_i \left(\boldsymbol{x}_i^T \boldsymbol{w} + b\right) + t_i^2 \zeta_i \geq t_i^2 \rho, \; \|\boldsymbol{w}\|_2 \leq 1, i \in I \tag{1}$$

where t_i is defined to be $\|\boldsymbol{x}_i\|_2$ and $\nu > 0$ is a given constant. Note in the special case with $t_i = 1$, Problem 1 becomes identical to the ν-SVM formulation.

To better approximate the "center of models", an ellipsoid, instead of a hypersphere, will be used to inscribe the polyhedron. The MVIE problem is a well-known log-determinant optimization problem, see e.g. (Boyd and Vandenberghe (2004)). A representation of an ellipsoid centered at $\boldsymbol{w}$ is given by $\mathcal{E} = \{\boldsymbol{E}\boldsymbol{u} + \boldsymbol{w} \mid \|\boldsymbol{u}\|_2 \leq 1, \; \boldsymbol{E} \succeq 0\}$. Thus the constraints for SVM (1) are modified as follows:

$$y_i \left(\boldsymbol{x}_i^T \left(\boldsymbol{E}\boldsymbol{u} + \boldsymbol{w}\right) + b\right) + t_i^2 \zeta_i \geq t_i^2 \rho, \; \forall \boldsymbol{u}, \; \|\boldsymbol{u}\|_2 \leq 1 \tag{2}$$

Since Equation 2 holds for any $\boldsymbol{u}$, it suffices to use the lower bound of lhs in order to remove $\boldsymbol{u}$:

$$y_i \left(\boldsymbol{x}_i^T \left(\boldsymbol{E}\boldsymbol{u} + \boldsymbol{w}\right) + b\right) + t_i^2 \zeta_i \geq y_i \left(\boldsymbol{x}_i^T \boldsymbol{w} + b\right) - \|\boldsymbol{E}\boldsymbol{x}_i\|_2 + t_i^2 \zeta_i \geq t_i^2 \rho \tag{3}$$

where $-\frac{y_i \boldsymbol{E}\boldsymbol{x}_i}{\|\boldsymbol{E}\boldsymbol{x}_i\|_2} = \arg\min_{\boldsymbol{u}, \; \|\boldsymbol{u}\|=1} (y_i \boldsymbol{x}_i^T \boldsymbol{E}\boldsymbol{u})$ is used.

Furthermore, in order to obtain the largest ellipsoid inscribing a polyhedron, the volume of the ellipsoid should be maximized, which corresponds to maximizing the determinant of $\boldsymbol{E}$ ($|\boldsymbol{E}|$), as the volume of an ellipsoid is proportional to the determinant. In an optimization problem, log-determinant is easier to handle and thus adopted here as well. The resulting optimization problem is given as follows:

$$\begin{aligned}\min_{\boldsymbol{E},\rho,\boldsymbol{\zeta},\boldsymbol{w},b} \quad & -\lambda\left(r\log|\boldsymbol{E}|-(1-r)\mathrm{tr}\boldsymbol{E}\right)-m\rho+\frac{1}{\nu}\sum\zeta_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{x}_i^T\boldsymbol{w}+b\right)-\|\boldsymbol{E}\boldsymbol{x}_i\|_2\geq t_i^2\rho-t_i^2\zeta_i \\ & \|\boldsymbol{w}\|_2\leq 1,\ \zeta_i\geq 0,\ i\in I,\ \boldsymbol{E}\succeq 0, \end{aligned} \tag{4}$$

where $\lambda > 0$ is a trade-off parameter and r is a constant whose value takes $0 < r \leq 1$. The additional term tr$\boldsymbol{E}$ is introduced to gain numerical stability as suggested in (Dolia et al. (2006)).

Note the role of ρ and $|\boldsymbol{E}|$ as maximizing margin is similar and redundant; the determinant maximization term can subsume the linear maximization of ρ [1]. Hence, ρ is dropped from the problem hereafter, allowing us to remove λ:

$$\begin{aligned}\min_{\boldsymbol{E},\boldsymbol{\zeta},\boldsymbol{w},b} \quad & -r\log|\boldsymbol{E}|+(1-r)\mathrm{tr}\boldsymbol{E}+\frac{1}{\nu}\sum\zeta_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{x}_i^T\boldsymbol{w}+b\right)+t_i^2\zeta_i\geq\|\boldsymbol{E}\boldsymbol{x}_i\|_2 \\ & \|\boldsymbol{w}\|_2\leq 1,\ \zeta_i\geq 0,\ i\in I,\ \boldsymbol{E}\succeq 0. \end{aligned} \tag{5}$$

This MVIE problem can be solved by using existing techniques, including interior point methods or cutting plane based approaches. Here we relax the SOC constraint in Problem 5 in order to ease the high computational complexity. This change, as we shall see, plays a significant role in making the kernelization possible. As the first step, assume the matrix $\boldsymbol{E}$ is written as $\boldsymbol{E} = \boldsymbol{E}_0+\boldsymbol{B}$, where $\boldsymbol{E}_0$ is the current solution and $\boldsymbol{B}$ is a deviation from it. By the Taylor expansion, the SOC constraint is written as $\|\boldsymbol{E}\boldsymbol{x}_i\|_2 = \kappa_i + \frac{1}{\kappa_i}\boldsymbol{x}_i^T\boldsymbol{E}_0\boldsymbol{B}\boldsymbol{x}_i + O(\|\boldsymbol{B}\|_2^2)$ where κ_i is given by $\kappa_i = \|\boldsymbol{E}_0\boldsymbol{x}_i\|_2$. Using the convexity of SOC, we get the following inequality: $\|\boldsymbol{E}\boldsymbol{x}_i\|_2 \geq \kappa_i + (1/\kappa_i)\boldsymbol{x}_i^T\boldsymbol{E}_0\boldsymbol{B}\boldsymbol{x}_i$.

Now the SOC constraints are replaced by linear constraints that are much easier to handle. In the special case with $\boldsymbol{E}_0 = c\boldsymbol{I}$, $c \rightarrow +0$, the problem becomes simple and may be used as the initial problem.

$$\begin{aligned}\min_{\boldsymbol{B},\boldsymbol{\xi},\boldsymbol{w},b} \quad & -r\log|\boldsymbol{B}|+(1-r)\mathrm{tr}\boldsymbol{B}+\sum_i C_i\xi_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{x}_i^T\boldsymbol{w}+b\right)+\xi_i\geq\boldsymbol{x}_i^T\boldsymbol{B}\boldsymbol{x}_i \\ & \|\boldsymbol{w}\|_2\leq 1,\ \boldsymbol{\xi}\geq 0,\ i\in I,\ \boldsymbol{B}\succeq 0 \end{aligned} \tag{6}$$

where we define $\xi_i = \kappa_i^2\zeta_i$ and $C_i = \frac{1}{t_i^2\nu}$. This formulates the ellipsoidal support vector machines primal problem. Note the Taylor approximation gets less accurate when $\|\boldsymbol{B}\|_2$ becomes larger, which is the cost for making the formulation feasible for kernelization done in Section 2.2.

Problem 6 has some interesting similarity with other methods. By putting $\boldsymbol{B} = \Sigma^{-1}$, it can be seen as a variant of MVCE problem in which the radius in the original problem is modified to a prediction dependent constraint. Hence it can be viewed as a supervised version of (Shivaswamy and Jebara (2007); Dolia et al. (2006)); unlike EKM, e-SVM solves

[1] Our preliminary study confirmed that ρ becomes zero in most cases

the classification problem at the same time. Shivaswamy et al.'s formulation for handling missing and uncertain data (Shivaswamy et al. (2006)) looks similar to Problem 4, where the metric in margin is given by the uncertainty in the data point. In e-SVM, margin is given by the $\boldsymbol{B}$-norm, which is optimized simultaneously with the classification problem.

2.1 Dual formulation

It can be readily shown that Problem 6 is a convex optimization problem with no duality gap. Hence the complementarity can be used to solve the primal and the dual problems, just like SVMs. The Lagrangian is given as follows:

$$\begin{aligned}\mathcal{L} &= -r\log|\boldsymbol{B}| + (1-r)\,\mathrm{tr}\boldsymbol{B} + \sum_i C_i\xi_i - \sum_i \alpha_i\left(y_i\left(\boldsymbol{x}_i^T\boldsymbol{w}+b\right) - \boldsymbol{x}_i^T\boldsymbol{B}\boldsymbol{x}_i - \xi_i\right)\\ &\quad +\gamma\left(\|\boldsymbol{w}\|_2^2 - 1\right) - \boldsymbol{\pi}^T\boldsymbol{\xi} - \mathrm{tr}(\boldsymbol{B}\boldsymbol{D}),\end{aligned}$$

where $\boldsymbol{\alpha}$, γ, $\boldsymbol{\pi}$ and $\boldsymbol{D}$ are the Lagrange multipliers for the classification constraints, norm constraint on $\boldsymbol{w}$, nonnegativity on $\boldsymbol{\xi}$ and positive semidefiniteness on $\boldsymbol{B}$, respectively. The optimality condition gives the following relations[2]:

$$\boldsymbol{B}^{-1} = \frac{1}{r}\left((1-r)\boldsymbol{I} + \sum_i \alpha_i\boldsymbol{x}_i\boldsymbol{x}_i^T\right),\ \boldsymbol{D}=0,\ \boldsymbol{w} = \frac{1}{2\gamma}\sum_i \alpha_i y_i\boldsymbol{x}_i,\ \sum_i y_i\alpha_i = 0,\ C_i - \alpha_i - \pi_i = 0.$$

Thus using the above equations the dual problem is written as follows:

$$\begin{aligned}\max_{\boldsymbol{\alpha},\gamma}\quad & r\log\left|\boldsymbol{B}^{-1}\right| - \frac{1}{4\gamma}\sum_{i,j}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j - \gamma\\ \text{s.t.}\quad & \boldsymbol{B}^{-1} = \frac{1}{r}\left((1-r)\boldsymbol{I} + \sum_i \alpha_i\boldsymbol{x}_i\boldsymbol{x}_i^T\right),\ \sum_i y_i\alpha_i = 0,\ 0\le\alpha_i\le C_i,\ \gamma>0 \qquad (7)\end{aligned}$$

A pleasant surprise is that $\boldsymbol{B}^{-1}$ is **always positive definite** since $\alpha_i \ge 0$, which is a great advantage, allowing us to remove the constraint $\boldsymbol{B}^{-1}\succeq 0$ in (7).

2.2 Kernel formulation

In this subsection, we show how Problem 7 is kernelized. For notational convenience, we use the matrix notation as well as the vector notation wherever appropriate. Note Problem 7 is very similar to the SVM problems, with the only difference being the additional $r\log\left|\boldsymbol{B}^{-1}\right|$ in the objective. By the matrix determinant lemma, the following equality can be shown to hold; $\left|\boldsymbol{B}^{-1}\right| = \left|\boldsymbol{I} + \frac{\boldsymbol{A}^{1/2}\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{A}^{1/2}}{(1-r)}\right|\left|\frac{1-r}{r}\boldsymbol{I}\right|$, where $\boldsymbol{X}$ is the data matrix $\boldsymbol{X} = [\boldsymbol{x}_1\ldots\boldsymbol{x}_m]^T$ and $\boldsymbol{A}$ is a diagonal matrix whose elements are give by $\boldsymbol{A}_{i,i} = \alpha_i$. Note that the last factor is a constant so it can be ignored.

By employing the kernel defined feature mapping $\boldsymbol{x}\mapsto\phi(\boldsymbol{x})$, or $\boldsymbol{X}\boldsymbol{X}^T\mapsto\boldsymbol{K}$, we have

$$\left|\boldsymbol{I} + \tfrac{1}{(1-r)}\boldsymbol{A}^{1/2}\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{A}^{1/2}\right| \quad\mapsto\left|\boldsymbol{I} + \frac{1}{(1-r)}\boldsymbol{A}^{1/2}\boldsymbol{K}\boldsymbol{A}^{1/2}\right| = \left|\boldsymbol{I} + \frac{1}{(1-r)}\boldsymbol{K}\boldsymbol{A}\right| \qquad (8)$$

[2] The $\log|\boldsymbol{B}|$ term forces $\boldsymbol{B}$ to be full-rank. Thus $\boldsymbol{D}=0$ holds by complementarity.

The Sylvester's determinant theorem, a generalization of the Matrix determinant lemma, is used. After removing the constant terms, the kernel e-SVM optimization problem is given by

$$\begin{aligned} \max \quad & r \log \left| \boldsymbol{I} + \frac{1}{(1-r)} \sum_i \alpha_i \boldsymbol{k}_i {\boldsymbol{e}_i}^T \right| - \frac{1}{4\gamma} \sum_{i,j} y_i y_j \alpha_i \alpha_j K_{ij} - \gamma \\ \text{s.t.} \quad & \sum_i y_i \alpha_i = 0, \; 0 \le \alpha_i \le C_i, \; \gamma \ge 0, \end{aligned} \tag{9}$$

with $\boldsymbol{k}_i$ being the i-th column of the kernel matrix and $\boldsymbol{e}_i$ being a vector of zeros except for the i-th element being unity.

3. Sequential minimal optimization

Although Problem 9 can be solved by an optimization package, a customized solver should be developed to take advantage of its similarity to the familiar SVM formulation; ideally an SVM solver can be modified to handle e-SVM. For this purpose, we develop a variant of SMO for e-SVM.

The differences from the standard implementation of SMO include $\|\boldsymbol{w}\|_2$ being normalized to one, step size optimization formula, and KKT conditions. The weight normalization concerns optimization with respect to γ and can be done via the iterative projection. Step size optimization and active set selection using the KKT condition are done very similar to those for SVM. This section focuses on describing essential differences as a guide to implementation.

3.1 Optimality conditions

SMO chooses an active set, a pair of data points, to optimize at any iteration. The selection of a pair critically affects the convergence speed. We adopt the selection heuristic described in (Keerthi et al. (2001)): choose ones that violate the KKT condition most. This subsection derives the KKT condition and thus gives the criterion for choosing the active set.

First, consider the dual of (9). The Lagrangian is given by

$$\begin{aligned} \mathcal{L} \;=\; & -r \log \left| \frac{1-r}{r} \boldsymbol{I} + \frac{1}{r} \sum_i \alpha_i \boldsymbol{k}_i {\boldsymbol{e}_i}^T \right| + \frac{1}{4\gamma} \sum_{ij} y_i y_j \alpha_i \alpha_j K_{ij} + \gamma - \eta \sum_i y_i \alpha_i \\ & - \sum_i \delta_i \alpha_i + \sum_i \mu_i \left(\alpha_i - C_i \right). \end{aligned}$$

Solving the optimality conditions, we have

$$\left(F_i - \eta \right) y_i - \delta_i + \mu_i - {\boldsymbol{e}_i}^T \widetilde{\boldsymbol{B}} \boldsymbol{k}_i = 0, \; \gamma = \sqrt{\sum_{ij} y_i y_j \alpha_i \alpha_j K_{ij} / 2}. \tag{10}$$

with $F_i = \frac{1}{2\gamma} \sum K_{ij} y_j \alpha_j$ and $\widetilde{\boldsymbol{B}} = \left(\frac{1-r}{r} \boldsymbol{I} + \frac{1}{r} \sum_i \alpha_i \boldsymbol{k}_i {\boldsymbol{e}_i}^T \right)^{-1}$. Hence, by the complementarity, we have the following KKT conditions:

- For $\alpha_i = 0,\ \delta_i > 0,\ \ \mu_i = 0 \Rightarrow (H_i - \eta)\, y_i \geq 0$
- For $0 < \alpha_i < C_i,\ \delta_i,\ \ \mu_i = 0, \Rightarrow (H_i - \eta)\, y_i = 0$
- For $\alpha_i = C_i,\ \delta_i = 0,\ \ \mu_i > 0 \Rightarrow (H_i - \eta)\, y_i \leq 0$

with $H_i = F_i - y_i {\boldsymbol{e}_i}^T \widetilde{\boldsymbol{B}} \boldsymbol{k}_i$. Note the first term F_i corresponds to that in (Keerthi et al. (2001)) and the second term is newly introduced for the e-SVM problem. This means that replacing F_i by H_i suffices to establish a version of the SMO algorithm for e-SVM and can be easily integrated into an existing SVM solver.

3.2 Step size computation

As explained, the KKT condition for e-SVM is easily adopted to the existing SMO algorithm. Another important piece in SMO algorithm is to find the optimal step size. The incremental step for α_i can be expressed as

$$\boldsymbol{\alpha}^{new} = \boldsymbol{\alpha}^{old} + s\left(\boldsymbol{e}_i - y_i y_j \boldsymbol{e}_j\right), \tag{12}$$

which satisfies the constraint $\sum_i y_i \alpha_i^{new} = 0$ given $\boldsymbol{\alpha}^{old}$ is a feasible solution. Consider the following objective function, $U(s)$, after removing any constant terms with respect to s:

$$U(s) = r \log \left|\widetilde{\boldsymbol{B}}^{-1}\right| - \sum_{i,j} \frac{1}{4\gamma} \alpha_i^{new} \alpha_j^{new} y_i y_j K_{ij} - \gamma. \tag{13}$$

The first term is modified using the update formula:

$$\begin{aligned} \left|\widetilde{\boldsymbol{B}}^{-1}\right| &= \left|\widetilde{\boldsymbol{B}}^{old-1} + \frac{s}{r}\left(\boldsymbol{k}_i {\boldsymbol{e}_i}^T - y_i y_j \boldsymbol{k}_j {\boldsymbol{e}_j}^T\right)\right| \\ &= \left|\boldsymbol{I} + \frac{s}{r}\begin{bmatrix} {\boldsymbol{e}_i}^T \\ -y_i {\boldsymbol{e}_j}^T \end{bmatrix} \widetilde{\boldsymbol{B}}^{old}\left[\boldsymbol{k}_i \ \ y_j \boldsymbol{k}_j\right]\right| \left|\widetilde{\boldsymbol{B}}^{old-1}\right| = \begin{vmatrix} 1 + \frac{s}{r}\omega_{ii} & \frac{s}{r} y_j \omega_{ij} \\ -\frac{s}{r} y_i \omega_{ji} & 1 - \frac{s}{r} y_i y_j \omega_{jj} \end{vmatrix} \times const \end{aligned}$$

where ω_{ij} is defined to be $\omega_{ij} = {\boldsymbol{e}_i}^T \widetilde{\boldsymbol{B}}^{old} \boldsymbol{k}_i$ and the matrix determinant lemma is used for deriving the 2nd line. The resulting matrix is merely a 2×2 matrix determinant and easily expandable.

Likewise, we can rewrite the second term in (13) as follows:

$$\sum_{i,j} \alpha_i^{new} \alpha_j^{new} y_i y_j K_{ij} \quad = \quad \sum_{i,j} \alpha_i^{old} \alpha_j^{old} y_i y_j K_{ij} - 4\gamma s y_i \left(F_i - F_j\right) - s^2 \left(K_{ii} - 2K_{ij} + K_{jj}\right).$$

Hence by putting all the pieces together, we have the following optimality condition on s.

$$\begin{aligned} \frac{\partial U(s)}{\partial s} &= r \frac{\partial \log \left|\widetilde{\boldsymbol{B}}^{-1}\right|}{\partial s} - \frac{\partial}{\partial s}\left(\frac{1}{4\gamma} \sum_{ij} \alpha_i \alpha_j y_i y_j K_{ij}\right) = 0 \\ &\Rightarrow \ \ r a_1 - a_3 + (2 r a_2 - a_1 a_3 - a_4) s - (a_2 a_3 + a_1 a_4) s^2 - a_2 a_4 s^3 = 0 \end{aligned}$$

with $a_1 = r(\omega_{ii} - y_i y_j \omega_{jj}),\ a_2 = y_i y_j(\omega_{ij}\omega_{ji} - \omega_{ii}\omega_{jj}),\ a_3 = y_i\left(F_i - F_j\right), a_4 = \frac{K_{ii} + K_{ii} - 2K_{ij}}{2\gamma}$. This is merely a cubic equation and can be solved **analytically**.

3.3 Computing $\widetilde{\boldsymbol{B}}$

At each iteration, access to $\widetilde{\boldsymbol{B}}$ is needed to calculate ω's. Specifically, the diagonal elements ω_{ii} are required for the KKT violation check and ω_{ij} as well as ω_{ii} and ω_{jj} for the step size computation concerning an update of α_i and α_j. Since we solve the dual $\boldsymbol{\alpha}$, as well as γ in SMO, $\widetilde{\boldsymbol{B}}^{-1}$ is easily obtained, but getting $\widetilde{\boldsymbol{B}}$, in a naive way, would require an inverse matrix operation that is never done in practice.

A way to efficiently computing $\widetilde{\boldsymbol{B}}$ is to employ the rank-one update of matrix inversion and factorize the matrix in the following way: $\widetilde{\boldsymbol{B}}^{new} = \widetilde{\boldsymbol{B}}^{old} + \sum_{k\in\{i,j\}} \sigma_k \boldsymbol{u}_k \boldsymbol{v}_k^T$. By using the Woodbury formula, $\widetilde{\boldsymbol{B}}$ is updated at each SMO step involving update of α_i and α_j: $\widetilde{\boldsymbol{B}}^{new} = \widetilde{\boldsymbol{B}}^{old} + \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T + \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^T$, where $\boldsymbol{u}_i = \widetilde{\boldsymbol{B}}^{old}\boldsymbol{k}_i,\ \boldsymbol{v}_i = \widetilde{\boldsymbol{B}}^{old T}\boldsymbol{e}_i,\ \omega_{ii} = \boldsymbol{k}_i{}^T\boldsymbol{v}_i,\ \sigma_i = -\frac{s}{r+s\omega_{ii}}$. $\boldsymbol{u}_j = \left(\widetilde{\boldsymbol{B}}^{old} + \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T\right)\boldsymbol{k}_j,\ \boldsymbol{v}_j = \left(\widetilde{\boldsymbol{B}}^{old} + \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T\right)^T \boldsymbol{e}_j,\ \sigma_j = \frac{s y_i y_j}{r - s y_i y_j \boldsymbol{k}_j{}^T \boldsymbol{v}_j}$. Note this decomposition formula on $\widetilde{\boldsymbol{B}}$ enables us to do an incremental update of ω:

$$\omega_{kl}^{new} = \omega_{kl}^{old} + \sigma_i \boldsymbol{u}_i{}^T \boldsymbol{e}_k \boldsymbol{v}_i{}^T \boldsymbol{k}_l + \sigma_j \boldsymbol{u}_j{}^T \boldsymbol{e}_k \boldsymbol{v}_j{}^T \boldsymbol{k}_l$$

where $\omega_{kl}^{old} = \boldsymbol{e}_k{}^T \widetilde{\boldsymbol{B}}^{old} \boldsymbol{k}_l$. We use this iterative update for diagonal ω_{ii} as they are used in any case for the KKT condition violation check. Further efficiency may be realized if exploiting caching of off-diagonal elements.

4. Online linear e-SVM

4.1 Preliminaries

For a strictly convex function of vectors $\mathcal{R}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$, *Bregman divergence* between two vectors $\boldsymbol{u}$ and $\boldsymbol{w}$ is defined as $D_{\mathcal{R}}(\boldsymbol{u}, \boldsymbol{v}) = \mathcal{R}(\boldsymbol{u}) - \mathcal{R}(\boldsymbol{v}) - \nabla\mathcal{R}(\boldsymbol{v})^T(\boldsymbol{u} - \boldsymbol{v})$. Also, for a strictly convex function of matrices, $\mathcal{R}(\boldsymbol{x}) : \mathbb{R}^{n\times n} \to \mathbb{R}$, Bregman divergence between two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is

$$D_{\mathcal{R}}(A, B) = \mathcal{R}(A) - \mathcal{R}(B) - \mathrm{tr}(\nabla\mathcal{R}(\boldsymbol{B})^T(\boldsymbol{A} - \boldsymbol{B})).$$

In particular, the Burg divergence between $\boldsymbol{A}$ and $\boldsymbol{B}$ is

$$\mathrm{tr}(\boldsymbol{A}\boldsymbol{B}^{-1}) - \ln\left|\boldsymbol{A}\boldsymbol{B}^{-1}\right| - n.$$

Burg divergence is the Bregman divergence for $\mathcal{R}(\boldsymbol{A}) = -\ln|\boldsymbol{A}|$.

4.2 Problem

Let

$$f(\boldsymbol{B}, \boldsymbol{w}) = -\ln|\boldsymbol{B}| + \frac{1}{m}\sum_i C_i \ell_i(\boldsymbol{B}, \boldsymbol{w}),$$

where $\ell_i(\boldsymbol{B}, \boldsymbol{w}) = \max(0, \boldsymbol{x}_i^T \boldsymbol{B} \boldsymbol{x}_i - y_i \boldsymbol{x}_i^T \boldsymbol{w})$, and $C_i = \frac{1}{\nu\|\boldsymbol{x}_i\|}$. Let $R = \max_i \|\boldsymbol{x}_i\|_2$. Note that $y_i \boldsymbol{x}_i^T \boldsymbol{w} \le \|\boldsymbol{x}_i\|_2 \|\boldsymbol{w}\|_2 \le R$. To make the loss ℓ meaningful, we limit the size of $\boldsymbol{x}_i^T \boldsymbol{B} \boldsymbol{x}_i$ at most R. To do this, we introduce a constraint $\mathrm{tr}\boldsymbol{B} \le 1/R$. Then,

$$\boldsymbol{x}_i^T \boldsymbol{B} \boldsymbol{x}_i = \mathrm{tr}(\boldsymbol{x}_i^T \boldsymbol{B} \boldsymbol{x}_i) = \mathrm{tr}(\boldsymbol{B} \boldsymbol{x}_i \boldsymbol{x}_i^T) \le \mathrm{tr}(\boldsymbol{B})\mathrm{tr}(\boldsymbol{x}_i \boldsymbol{x}_i^T) = \mathrm{tr}(\boldsymbol{B})\|\boldsymbol{x}_i\|_2^2 \le R,$$

Algorithm 1 Online e-SVM

```
1. Let $\boldsymbol{B}_1 = \frac{1}{nR}\boldsymbol{I}$ and $\boldsymbol{w} = \boldsymbol{0}$.
2. For $t = 1, \ldots$
    (a) Pick up $(\boldsymbol{x}_t, y_t)$ uniformly randomly from the training set.
    (b) Let $\eta_1 = \frac{1}{2}$ and $\eta_t = \frac{1}{2t}$ for $t \geq 2$.
    (c) $\boldsymbol{B}_{t+\frac{1}{2}}^{-1} = (1-\eta_t)\boldsymbol{B}_t^{-1} + \eta_t\sigma_t C_t \boldsymbol{x}_t \boldsymbol{x}_t^T$, where $\sigma_t = 1$ if $\boldsymbol{x}_{i_t}^T \boldsymbol{B}_t \boldsymbol{x}_t - y_t \boldsymbol{x}_t^T \boldsymbol{w}_t \geq 0$, and $\sigma_t = 0$, otherwise.
    (d) $\boldsymbol{B}_{t+1} = \arg\min_{\boldsymbol{B} \succeq 0, \mathrm{tr}\boldsymbol{B} \leq \frac{1}{R}} D_{\mathcal{R}}(\boldsymbol{B}, \boldsymbol{B}_{t+\frac{1}{2}})$.
    (e) $\boldsymbol{w}_{t+\frac{1}{2}} = \boldsymbol{w}_t + \eta_t \sigma_t C_t y_t \boldsymbol{x}_t$.
    (f) $\boldsymbol{w}_{t+1} = \min\left\{1, \frac{1}{\|\boldsymbol{w}_{t+\frac{1}{2}}\|}\right\} \boldsymbol{w}_{t+\frac{1}{2}}$.
```

where the first inequality follows from the fact that $\mathrm{tr}(\boldsymbol{AB}) \leq \mathrm{tr}(\boldsymbol{A})\mathrm{tr}(\boldsymbol{B})$ for positive semi-definite matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. Consider the following problem [3]

$$\min_{\boldsymbol{B}, \boldsymbol{w}} \; f(\boldsymbol{B}, \boldsymbol{w}) \;\; \text{s.t. } \boldsymbol{B} \succeq 0, \; \mathrm{tr}\boldsymbol{B} \leq 1/R, \; \|\boldsymbol{w}\|_2 \leq 1. \tag{14}$$

By using KKT conditions, the optimal solution $(\boldsymbol{B}^*, \boldsymbol{w}^*)$ has the following property:

$$B^{*-1} = \sum_{i=1}^{m} \alpha_i \boldsymbol{x}_i \boldsymbol{x}_i^T \text{ and } \boldsymbol{w}^* = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i,$$

where each α_i satisfies $0 \leq \alpha_i \leq C_i/m$.

4.3 Algorithm

Our algorithm for solving the problem (14) is based on the online convex optimization algorithm called Regularized Follow the Leader (RFTL) (Hazan (2009)). The algorithm RFTL captures many existing algorithms. Let

$$f(\boldsymbol{B}, \boldsymbol{w}, i) = -\ln|\boldsymbol{B}| + C_i \ell_i(\boldsymbol{B}, \boldsymbol{w}).$$

For any given $i_t \in \{1, ..., m\}$ at trial t, we denote $f_t(\boldsymbol{B}, \boldsymbol{w}) = f(\boldsymbol{B}, \boldsymbol{w}, i_t)$. By following the RFTL algorithm, the pseudo code for the online e-SVM is given in Algorithm 1. Note Step 2(c) can be calculated, using the Woodbury formula, as a rank-one update of $\boldsymbol{B}_t$: $\boldsymbol{B}_{t+\frac{1}{2}} = \frac{1}{\eta_t}\left(\boldsymbol{B}_t - \frac{\eta_t C_t}{1-\eta_t+\eta_t C_t} \boldsymbol{B}_t \boldsymbol{x} (\boldsymbol{B}_t \boldsymbol{x})^T\right)$.

4.4 Analysis

In this subsection, we analyze the algorithm and derive some properties. Proofs are provided in Appendix.

[3] Note that, for simplicity, we omit the bias term and the trace term and assume that the solution exists.

Let $\Xi = (\boldsymbol{B}, \boldsymbol{w})$ and $\mathcal{R}(\Xi) = -\ln|\boldsymbol{B}| + \frac{1}{2}\|\boldsymbol{w}\|^2$. Further, for $\Xi = (\boldsymbol{B}, \boldsymbol{w})$ and $\Xi' = (\boldsymbol{B}', \boldsymbol{w}')$, we denote

$$D_{\mathcal{R}}(\Xi, \Xi') = D_{\mathcal{R}}(\boldsymbol{B}, \boldsymbol{B}') + D_{\mathcal{R}}(\boldsymbol{w}, \boldsymbol{w}'),$$

where $D_{\mathcal{R}}(\boldsymbol{B}, \boldsymbol{B}') = \mathrm{tr}(\boldsymbol{B}\boldsymbol{B}'^{-1}) - \ln\left|\boldsymbol{B}\boldsymbol{B}'^{-1}\right| - n$, and $D_{\mathcal{R}}(\boldsymbol{w}, \boldsymbol{w}') = \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}'\|_2^2$, respectively. Finally, given $R > 0$, let $\mathcal{F}$ be the set of feasible region, i.e., $\mathcal{F} = \{(\boldsymbol{B}, \boldsymbol{w}) \in \mathbb{R}^{n \times n} \times \mathbb{R}^n | \ \boldsymbol{B} \succeq \boldsymbol{0}, \mathrm{tr}\boldsymbol{B} \leq R, \|\boldsymbol{w}\|_2 \leq 1\}$.

We can prove the following upper-bound of regret.

Theorem 1 *For any $T \geq 1$,*

$$\sum_{t=1}^{T} f_t(\Xi_t) - \inf_{\Xi \in \mathcal{F}} \sum_{t=1}^{T} f_t(\Xi) \leq O\left(n \ln \frac{R}{\nu}\right) + O\left((n + \frac{1}{\nu^2}) \ln T\right).$$

The final solution for use is an average over all learning steps. The following theorem states the convergence speed of the online e-SVM.

Theorem 2 *For any $T \geq 1$, let $\bar{\Xi} = (\bar{\boldsymbol{B}}, \bar{\boldsymbol{w}})$, where $\bar{\boldsymbol{B}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{B}_t$ and $\bar{\boldsymbol{w}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{w}_t$. Then,*

$$\mathbf{E}[f(\bar{\Xi})] \leq \inf_{\Xi \in \mathcal{F}} f(\Xi) + O\left(\frac{n \ln \frac{R}{\nu}}{T}\right) + O\left(\frac{(n + \frac{1}{\nu^2}) \ln T}{T}\right).$$

Therefore, we can have the following Corollary for the number of steps toward the ε-approximate solution.

Corollary 3 *After $\tilde{O}(\frac{nln\frac{R}{\nu} + \frac{1}{\nu^2}}{\varepsilon})$ steps, our algorithm outputs the final hypothesis $\bar{\Xi}$, whose expected approximation error is less than ε. w.r.t. the problem (14).*

Stopping Criterion For a practical implementation, we consider the following stopping criterion: Run the algorithm for T steps, where T is such that

$$\frac{1}{T}\left(n \ln \frac{R}{\nu} + \sum_{t=1}^{T} D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}})\right) \leq \varepsilon,$$

where ε is a precision parameter. Since the left hand side is an upperbound of loss of the final hypothesis $\bar{\Xi}$, after T steps, the algorithm outputs an ε-approximation (in terms of expectation).

5. Experimental study

5.1 Effect of Approximation on Predictive Performance

It is important to examine the quality of e-SVM solutions to understand how the approximation and relaxation used in e-SVM affect the performance. First, we examine how the relaxation in Problem 6 compares against the exact problem, Problem 5, using some benchmark datasets available in (Fan). SeDuMi (Sturm (1999)) is used for both problems in order to eliminate differences coming from different implementation. Except *splice*, where

Table 1: Comparison between Problem 5 and 6

model		splice	mushroom	a1a	a2a	a3a	w1a	w2a	w3a
exact	error rate (%)	17.37	0.81	24.09	16.24	16.06	2.92	2.91	2.85
(5)	time (sec)	525.9	2249.1	1512.9	2282.8	3823.2	3191.3	4544.4	9789.3
approx.	error rate (%)	17.67	0.97	24.09	16.25	16.22	2.96	2.97	2.98
(6)	time (sec)	320.0	807.2	612.6	966.3	1201.8	1028.3	1606.8	2411.6

Data set	SVM	BPM	e-SVM
thyroid	4.96 (.24)	**4.24 (.22)**	**4.42 (.25)**
heart	25.86 (.40)	**22.58 (.33)**	**20.87 (.32)**
diabetes	33.87 (.21)	**31.06 (.22)**	**29.68 (.24)**
wave	13.19 (.12)	**12.02 (.08)**	**11.59 (.07)**
banana	16.24 (.14)	**13.70 (.10)**	**12.76 (.08)**
wisc-bc	4.22 (.13)	**2.56 (.10)**	**3.28 (.12)**
bupa	37.04 (.39)	**34.5 (.38)**	**32.71 (.35)**
german	30.07 (.22)	**27.16 (.24)**	**26.34 (.27)**
brest	35.17 (.51)	**33.04 (.48)**	**31.96 (.51)**
sonar	**14.90 (.38)**	16.26 (.36)	16.87 (.38)
iono	7.94 (.25)	11.45 (.25)	**5.92 (.21)**

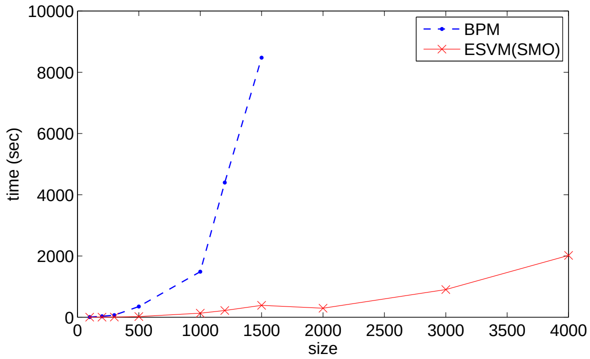

Figure 1: *left*: Error rates for hard margin/bounary classifiers. *right*: Computing time for BPM and e-SVM.

we randomly split into 3/4 for training and 1/4 for testing, the supplied test sets are used for performance evaluation. For all datasets, we use the first 30 principal components to reduce the dimensionality. The linear kernel is used as a kernel formulation is not available in Problem 5. Hyperparameters are tuned using the three-fold cross-validation (CV) inside the training set. The result is summarized in Table 1. Although the linearization used in Problem 6 seems a crude approximation, the effect is very limited, but is more computationally efficient.

5.2 Comparison against BPM

Next, we examine how the ellipsoidal approximation to the Bayes Point affect the generalization ability. To this end, we reproduce similar study as done in the original BPM paper (Herbrich et al. (2001)). Also, BPM and e-SVM are compared against SVM as a reference. A wide range of datasets in the UCI machine learning repository (Blake and Merz (1998)) are used. Both SVM and e-SVM are implemented in pure MATLAB, using the SMO. Note ν-SVM formulation is adopted in this study, since e-SVM is based on ν-SVM. BPM's implementation follow (Herbrich et al. (2001)) and is implemented in C.

For the experimental setting, 100 randomizations are done and the average error rates are reported in Figure 1 *left*. In order to evaluate significance of statistics, the paired t-tests are conducted for comparing BPM with SVM, and e-SVM with SVM. Bold numbers denote the test results being significant. For hard margin e-SVM, r is set to $1 - 10^{-6}$. The radial basis function (RBF) kernel is used for this experiment. For further details of experimental design, see (Herbrich et al. (2001)). The overall performance for BPM and e-SVM is very similar. This suggests that the approximations made to formulate e-SVM do not affect the

Table 2: Comparison between soft-margin SVM and e-SVM (%-error)

model	ionosphere	mushroom	splice	dna	letter	satimage	usps
SVM	7.41	0.74	15.31	5.48	20.82	16.20	8.13
e-SVM	7.12	0.00	14.67	4.72	20.54	13.60	6.48

classification performance for the datasets examined. In comparison with SVM, the hard boundary/margin classifiers significantly outperform those of SVM.[4]

Figure 1 *right* shows the computing time to see how e-SVM (RBF kernel) scales in comparison with BPM, using the *adult* dataset (Blake and Merz (1998)). The size of the training set is increased from 100 up to 4000. Test performance is observed to check if the there is no significant difference between the methods. Obviously, e-SVM runs much faster than BPM as the data size grows.

5.3 Comparison against soft-margin SVM

We apply bigger datasets for illustrating performance difference between SVM and e-SVM. In this experiment, we focus on soft-margin classifiers and use the linear kernel. We conduct the nested CV to tune hyperparameters and evaluate for both methods. The outer CV is set to 10-fold and the inner CV 5-fold. Table 2 shows the results. Note the datasets used are of medium size in both data points and dimensionality, as opposed to those in Table 1. As mentioned in (Herbrich et al. (2001)), advantage in BPM-like methods tend to dissipate when used in a soft-margin case. However this experiment show that e-SVM, or BPM, can out-perform SVM for most datasets as it captures some covariance structure in relatively higher dimensional space.

5.4 Large-scale datasets by online linear e-SVM

To illustrate applicability of online linear e-SVM to large scale datasets. We use the full *adult* dataset with 45,222 records in 14 dimensions, and *covtype* (Blake and Merz (1998)) with 581,012 records in 54 dimensions. We use half for training and the rest for testing. We set ε to 10^{-3} for the stopping condition. The test error for *adult* was 15.6% and *covtype* 23.4% with computing time 76 sec and 14386 sec, respectively, while SMO took 887 sec for *adult* and more than 3 days for *covtype*. [5] Note we did not handle data with sparse format. A better handle of sparse structure should reduce the computing time. At any rate, this observation shows, for large scale datasets, the online e-SVM is particularly useful for building linear e-SVM models.

6. Conclusion

In this paper, the ellipsoidal support vector machine was proposed. The formulation is based on that of the familiar SVM and familiar convex optimization methods are applied to solve kernel and primal e-SVM. The framework is flexible for possible modification of

[4] We conducted the same study with soft margin models and found that the advantage dissipates as noted in (Herbrich et al. (2001)).

[5] For reference, SVM obtained 16.2% and 23.8%, respectively. Again, e-SVM keeps advantage; Small performance lift can translate in significant error reduction for large-scale datasets.

loss functions or application to other problems. Also, by the minimum volume ellipsoid interpretation, it can be used to learn the metric guided through the maximum margin framework. None of these advantages is available in BPM and thus novel in e-SVM. e-SVM showed comparable performance with BPM, indicating the approximations in e-SVM do not affect the performance over wide variety of datasets. Online e-SVM was shown to be applicable to real large scale problems with some performance advantage.

Acknowledgments

We are deeply grateful to Professor John E. Mitchell at Rensselaer Polytechnic Institute and Dr. Ralf Herbrich at Microsoft for fruitful discussion and referees for their useful comments.

Appendix A. Proofs

First, we prove Theorem 1. As a preparation, we prove several Lemma's.

Lemma 1 (Cf. Hazan (2009)) *For any $T \geq 1$ and any $\Xi^* \in \mathcal{F}$,*

$$\sum_{t=1}^{T} f_t(\Xi_t) - \sum_{t=1}^{T} f_t(\Xi^*) \leq D_{\mathcal{R}}(\Xi^*, \Xi_1) + \sum_{t=1}^{T} \frac{1}{\eta_t} D_{\mathcal{R}}(\Xi_t, \Xi_{t+\frac{1}{2}}).$$

Proof By definition of D_{f_t}, we have

$$\begin{aligned} f(\Xi_t) - f_t(\Xi^*) &= \nabla f_t(\Xi_t)(\Xi_t - \Xi^*) - D_{f_t}(\Xi^*, \Xi_t) \\ &= \frac{1}{\eta_t}(\nabla \mathcal{R}(\Xi_t) - \nabla \mathcal{R}(\Xi_{t+\frac{1}{2}}))(\Xi_t - \Xi^*) - D_{\mathcal{R}}(\Xi^*, \Xi_t), \end{aligned}$$

where the second equation follows from the update of 2. (c) and (e) in Algorithm 1 and the fact that $D_{f_t} = D_{\mathcal{R}}$.

By using the following relationship for any $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$

$$(\boldsymbol{x} - \boldsymbol{y})(\nabla \mathcal{R}(z) - \nabla \mathcal{R}(y)) = D_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{y}) - D_{\mathcal{R}}(\boldsymbol{x}, \boldsymbol{z}) + D_{\mathcal{R}}(\boldsymbol{y}, \boldsymbol{z}),$$

we have

$$\begin{aligned} f_t(\Xi_t) - f_t(\Xi^*) &= \frac{1}{\eta_t}(D_{\mathcal{R}}(\Xi^*, \Xi_t) - D_{\mathcal{R}}(\Xi^*, \Xi_{t+\frac{1}{2}}) + D_{\mathcal{R}}(\Xi_t, \Xi_{t+\frac{1}{2}})) - D_{\mathcal{R}}(\Xi^*, \Xi_t) \\ &\leq \frac{1}{\eta_t}(D_{\mathcal{R}}(\Xi^*, \Xi_t) - D_{\mathcal{R}}(\Xi^*, \Xi_{t+1}) + D_{\mathcal{R}}(\Xi_t, \Xi_{t+\frac{1}{2}})) - D_{\mathcal{R}}(\Xi^*, \Xi_t), \end{aligned}$$

where the last inequality follows from the Pythagorean Theorem for Bregman divergences (e.g., Cesa-Bianchi and Lugosi (2006)). So we have

$$\begin{aligned} \sum_{t=1}^{T} f_t(\Xi_t) - \sum_{t=1}^{T} f_t(\Xi^*) \leq & \left(\frac{1}{\eta_1} - 1\right) D_{\mathcal{R}}(\Xi^*, \Xi_1) + \left(\frac{1}{\eta_2} - 1 - \frac{1}{\eta_1}\right) D_{\mathcal{R}}(\Xi^*, \Xi_2) \\ & + \sum_{t=2}^{T-1} \left(\frac{1}{\eta_{t+1}} - 1 - \frac{1}{\eta_t}\right) D_{\mathcal{R}}(\Xi^*, Z_t) - \frac{1}{\eta_T} D_{\mathcal{R}}(\Xi^*, \Xi_T) + \sum_{t=1}^{T} \frac{1}{\eta_t} D_{\mathcal{R}}(\Xi_t, \Xi_{t+\frac{1}{2}}) \\ \leq & D_{\mathcal{R}}(\Xi^*, \Xi_1) + \sum_{t=1}^{T} \frac{1}{\eta_t} D_{\mathcal{R}}(\Xi_t, \Xi_{t+\frac{1}{2}}), \end{aligned}$$

where the last inequality holds since the second and forth term is negative and the third term is zero. ■

Lemma 2 *For any $t \geq 1$, $D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}}) \leq 4\eta_t^2 \left(n + \frac{1}{\nu^2}\right)$.*

Proof

$$\begin{aligned}
D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}}) &= \mathrm{tr}(\boldsymbol{B}_t\boldsymbol{B}_{t+\frac{1}{2}}^{-1}) - \ln\left|\boldsymbol{B}_t\boldsymbol{B}_{t+\frac{1}{2}}^{-1}\right| - n \\
&= \mathrm{tr}((1-\eta_t)\boldsymbol{I} + \boldsymbol{B}_t\eta_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T) - \ln\left|((1-\eta_t)\boldsymbol{I} + \boldsymbol{B}_t\eta_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T\right| - n.
\end{aligned}$$

Note that, since $\left|I + \boldsymbol{u}\boldsymbol{v}^T\right| = 1 + \boldsymbol{u}^T\boldsymbol{v}$, we have

$$\begin{aligned}
\left|((1-\eta_t)\boldsymbol{I} + \eta_t\boldsymbol{B}_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T\right| &= (1-\eta_t)^n\left|(\boldsymbol{I} + \frac{\eta_t}{1-\eta_t}\sigma_t C_i\boldsymbol{B}_t\boldsymbol{x}_t\boldsymbol{x}_t^T\right| \\
&= (1-\eta_t)^n\left|1 + \frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t\right|.
\end{aligned}$$

Therefore,

$$\begin{aligned}
D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}}) &= \mathrm{tr}((1-\eta_t)\boldsymbol{I} + \eta_t B_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T) + -\ln((1-\eta_t)^n(1 + \frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t)) - n \\
&= \mathrm{tr}(-\eta_t\boldsymbol{I} + \eta_t B_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T) - n\ln(1-\eta_t) - \ln(1 + \frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t).
\end{aligned}$$

Since $-\ln(1-x) \leq x + \frac{x^2}{c(1-c)}$ for $0 \leq x \leq c < 1$,

$$D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}}) \leq \mathrm{tr}(\eta_t B_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T) + 4n\eta_t^2 - \ln(1 + \frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t).$$

Further, since $-\ln(1+x) \leq -x + x^2$ for $0 \leq x$ and $\mathrm{tr}(\boldsymbol{B}\boldsymbol{x}\boldsymbol{x}^T) = \boldsymbol{x}^T\boldsymbol{B}^T\boldsymbol{x}$,

$$\begin{aligned}
D_{\mathcal{R}}(\boldsymbol{B}_t, \boldsymbol{B}_{t+\frac{1}{2}}) &\leq \mathrm{tr}(\eta_t B_t\sigma_t C_t\boldsymbol{x}_t\boldsymbol{x}_t^T) + 4n\eta_t^2 - \frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t + \left(\frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t\right)^2 \\
&\leq +4n\eta_t^2 + \left(\frac{\eta_t}{1-\eta_t}\sigma_t C_t\boldsymbol{x}_t^T\boldsymbol{B}_t^T\boldsymbol{x}_t\right)^2 \leq 4\eta_t^2\left(n + \frac{1}{\nu^2}\right).
\end{aligned}$$

■

Lemma 3 *For any $\boldsymbol{B}^*$ such that $\boldsymbol{B}^* \succeq \boldsymbol{0}$ and $\mathrm{tr}\boldsymbol{B} \leq R$, $D_{\mathcal{R}}(\boldsymbol{B}^*, \boldsymbol{B}_1) \leq n\ln\frac{R}{\nu}$.*

Proof Let $\lambda_1, \ldots, \lambda_n$ be eigenvalues of B^{*-1}. Then, by definition of $\boldsymbol{B}^{*-1}$ and the inequality of arithmetic and geometric means,

$$D_{\mathcal{R}}(\boldsymbol{B}^*, \boldsymbol{B}_1) \leq \ln(nR)^n + \ln\left|\boldsymbol{B}^{*-1}\right| \leq -\ln(nR)^n + \ln\Pi_{i=1}^n\lambda_i = -\ln(nR)^n + \ln\left(\frac{\mathrm{tr}\boldsymbol{B}^{*-1}}{n}\right)^n.$$

Further, by definition of B^{*-1}, the r.h.s. is given as

$$-\ln(nR)^n - n\ln n + n\ln\left(\sum_i \alpha_i^* \mathrm{tr}\boldsymbol{x}_i\boldsymbol{x}_i^T\right) \leq n\ln\frac{R}{\nu},$$

where the last inequality follows from the fact that $\alpha_i \leq C_i/m$. ■

Proof of Theorem 1

Proof Note that

$$D_{\mathcal{R}}(\boldsymbol{w}_{t+\frac{1}{2}}, \boldsymbol{w}_t) = \frac{1}{2}\|\boldsymbol{w}_{t+\frac{1}{2}} - \boldsymbol{w}_t\|^2 = \frac{1}{2}\|\eta_t\sigma_t C_t y_t \boldsymbol{x}_t\|^2 \leq \frac{\eta_t^2}{2\nu^2},$$

and $D_{\mathcal{R}}(\boldsymbol{w}^*, \boldsymbol{w}_1) = \frac{1}{2}\|\boldsymbol{w}^*\|^2 \leq \frac{1}{2}$. By combining these with Lemma 1,2, and 3 and the fact that $\eta_t = 1/(2t)$, we complete the proof. ■

Proof of Theorem 2

Proof By convexity of f and linearity of expectation, we have

$$\mathbf{E}\left[f(\bar{\boldsymbol{\Xi}})\right] \leq \frac{1}{T}\mathbf{E}\left[\sum_{t=1}^{T} f_t(\boldsymbol{\Xi}_t)\right].$$

Then, by applying Theorem 1, for any $\boldsymbol{\Xi}^* \in \mathcal{F}$, the right hand side is further bounded by

$$\frac{1}{T}\mathbf{E}\left[\sum_{t=1}^{T} f_t(\boldsymbol{\Xi}^*)\right] + O\left(\frac{n\ln\frac{R}{\nu}}{T}\right) + O\left(\frac{(n+\frac{1}{\nu^2})\ln T}{T}\right). \tag{15}$$

Note that $\mathbf{E}\left[\sum_{t=1}^{T} f_t(\boldsymbol{\Xi}^*)\right] = f(\boldsymbol{\Xi}^*)$, which implies

$$\mathbf{E}\left[f(\bar{\boldsymbol{\Xi}})\right] \leq \inf_{\boldsymbol{\Xi}\in\mathcal{F}} f(\boldsymbol{\Xi}) + +O\left(\frac{n\ln\frac{R}{\nu}}{T}\right) + O\left(\frac{(n+\frac{1}{\nu^2})\ln T}{T}\right),$$

as claimed. ■

References

C. L. Blake and C. J. Merz. UCI Repository of machine learning databases, 1998. `http://www.ics.uci.edu/~mlearn/ MLRepository.html`.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, 2006.

Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Schölkopf. A tutorial on ν-support vector machines: Research articles. *Appl. Stoch. Model. Bus. Ind.*, 21(2), 2005.

A.N. Dolia, T. De Bie, C.J. Harris, J. Shawe-Taylor, and D.M. Titterington. The minimum volume covering ellipsoid estimation in kernel-defined feature spaces. In *ECML*. 2006.

Rong-En Fan. Libsvm data: Classification, regression, and multi-label. `http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/`.

Elad Hazan. A survey: The convex optimization approach to regret minimization. `http://www.cs.princeton.edu/ ehazan/papers/OCO-survey.pdf`, 2009.

Ralf Herbrich, Thore Graepel, and Colin Campbell. Bayes point machines. *JMLR*, 2001.

C. Hsieh, K. Chang, C. Lin, S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. 2008.

S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Comput.*, 13(3), 2001.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 1994.

Pál Ruján. Playing billiards in version space. *Neural Comput.*, 9(1), 1997.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2001.

Shai Shalev-Shwartz and Nathan Srebro. Svm optimization: inverse dependence on training set size. In *ICML*, 2008.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *ICML*, 2007.

P. Shivaswamy and T. Jebara. Ellipsoidal kernel machines. *AISTATS*, 2007.

Pannagadatta K. Shivaswamy, Chiranjib Bhattacharyya, and Alexander J. Smola. Second order cone programming approaches for handling missing and uncertain data. *JMLR*, 7, 2006.

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

Theodore B. Trafalis and Alexander M. Malyscheff. An analytic center machine. *Mach. Learn.*, 46 (1-3), 2002.

V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, 1996.

V. Vovk. Aggregating strategies. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory*, pages 371–386, 1990.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.