To appear in the *Applied Artificial Intelligence (AAI) Journal*
Vol. 00, No. 00, Month 20XX, 1–16

A randomized algorithm for the exact solution of Transductive Support Vector Machines

G. Esposito^{a*} and M. Martin^a

^a *Universitat Politècnica de Catalunya, Barcelona, Spain;*

(v1.0 released October 2014)

Random sampling is an efficient method to deal with constrained optimization problems. In computational geometry, this method has been successfully applied, through Clarkson’s algorithm (Clarkson, 1996), to solve a general class of problems called *violator spaces*. In machine learning, *TSVM* is a learning method used when only a small fraction of labeled data is available, which implies to solve a non convex optimization problem. Several approximation methods have been proposed to solve it, but they usually find suboptimal solutions. However, global optimal solution may be obtained using exact techniques, but at the cost of suffering an exponential time complexity with respect to the number of instances. In this paper, an interpretation of *TSVM* in terms of violator space is given. Hence, a randomized method is presented extending the use of exact methods now reducing the time complexity to exponential w.r.t. the number of support vectors of the optimal solution instead of exponential w.r.t. the number of instances.

Keywords: Transduction, Semi-Supervised Learning, Support Vector Machine, Violator Spaces, Branch and Bound

1. Introduction

In computational geometry, random sampling is an efficient method to deal with constrained optimization problems. Firstly one finds the optimal solution *subject to a random subset of constraints*. Likely, the expected number of constraints violating that solution is significantly smaller than the overall number of remaining constraints. Even in some lucky cases, the found solution does not violate the remaining constraints. Hence, one can exploit this property to build a simple randomized algorithm. Clarkson’s algorithm (Clarkson, 1996) is a two-staged random sampling technique able to solve linear programming problems, which can also be applied to the more general framework of violator spaces. The violator space framework has become a well-established tool in the field of geometric optimization, developing subexponential algorithms starting from a randomized variant of the simplex method. The class of violator space includes the problem of computing the minimum-volume ball or ellipsoid enclosing a given point set in R^d , the problem of finding the distance of two convex polytopes in R^d and many other computational geometry problems (Gartner, J. Matousek, and P. Skovron, 2008). Generalization to violator space problems makes it applicable to a number of non-linear and mostly geometric problems. Clarkson’s algorithm stages are based on random sampling and are conceptually very simple. Once it is shown that a particular optimization problem can be

*Corresponding author. Email: gesposit@lsi.upc.edu

regarded as violator space problem, and certain algorithmic primitives are implemented for it, Clarkson's algorithm is immediately applicable.

In machine learning, Transductive Support Vector Machine (*TSVM*) (Vapnik, 1995) extends the well know support vector machines to handle partially labeled datasets. *TSVM* learns the maximum margin hyperplane classifier using labeled training data that does not present unlabeled data inside its margin. Unfortunately, dealing with a *TSVM* implies to solve a non convex optimization problem. A wide spectrum of approximative techniques have been applied to solve the *TSVM* problem (Chapelle, 2008), but they do not guarantee finding the optimal global solution. In fact, when state-of-the-art approximative *TSVM* methods have been applied to different benchmark problems, far from optimal solutions have been found (Chapelle, 2008). Unfortunately exact methods can only be applied to small datasets due to the exponential time complexity cost with respect to the number of instances. In (J., Y., J., and O., 2008) Balcazar *et alt.* suggested that a hard margin *SVM* belongs to the class of violator space, proposing a random sampling technique for the determination of the maximum margin separating hyperplane. Note that the problem of solving a *SVM* is convex while solving a *TSVM* is a non convex problem, so they are very different in nature. In this paper we prove that the global optimal solution of a *TSVM* totally relies on the knowledge of the support vectors set, gathering a size smaller than the whole set of instances. Moreover, we also demonstrate that *TSVM* can be formulated as a violator space problem allowing the use of the Clarkson's algorithm to find its optimal global solution. Fostering the *TSVM* sparsity property, we introduce a randomized algorithm able to reduce the time complexity of exact methods, scaling now exponentially w.r.t. the number of support vectors of the optimal solution instead of exponentially w.r.t. the number of instances. Using our method one may find the *exact* solution independently on the number of instances when the problem has few support vectors. This include problems where the dimensionality in the feature space is relatively small.

2. Violator spaces and randomized algorithms

Violator space problems were introduced as an abstract framework for randomized algorithms able to solve linear programs by a variant of the simplex algorithm. In computational geometry, an example problem that can be solved using this method is the *Smallest Enclosing Ball (SEB)*. Here the goal is to compute the smallest ball enclosing a set of n points in a d dimensional space (fig. 1). In the following we introduce the main tools of the abstract framework of violators spaces in order to show how randomized methods devised in computational geometry can be also applied to solve the *TSVM* problem. Details and proofs of the reported properties can be found in (Gartner et al., 2008).

Definition 1: An abstract LP-type problem consists of a finite set H , representing the constraints of the problem, a weight function w giving for $G \subseteq H$ the cost $w(G) \in W$ of the optimum solution, W linearly ordered $(W, \leq)$. $(H, w, W, \leq)$ satisfy:

- **Monotonicity:** for all $F \subseteq G \subseteq H$ we have $w(F) \leq w(G)$ and
- **Locality:** for all $F \subseteq G \subseteq H$ and all $h \in H$ with $w(F) = w(G)$ and $w(G) < w(G \cup \{h\})$ we have $w(F) < w(F \cup \{h\})$

For a $G \subseteq H$ a *basis* is an inclusion-minimal subset $B \subseteq G$ such that $w(G) = w(B)$. The *combinatorial dimension* δ of an LP-type problem is the size of the largest basis. A *violator* of G is an additional constraint $h \in H$ such that $w(G) \neq w(G \cup h)$. An

Figure 1. Smallest Enclosing Ball: extremes (red encircled in red) are the points essential for the solution, violators (in blue) are the points lying outside the ball, a basis is a minimal set of points having the same ball

element h is an *extreme* of G if $w(G - \{h\}) \neq w(G)$. From definition h violates $G \Leftrightarrow h$ is extreme in $G \cup \{h\}$. The set of all constraints violating G is called *violator mapping*

$$V(G) := \{h \in H : w(G) < w(G \cup \{h\})\}$$

. Using the violator mapping is possible to phrase in terms of violation tests the LP-type problem neglecting the explicit evaluation of w :

Definition 2: A violator space is a pair (H, V) with H a finite set and V a mapping. (H, V) satisfy:

- **Consistency:** $G \cap V(G) = \emptyset$ holds for all $G \subset H$ and
- **Locality:** for all $F \subset G \subset H$ where $G \cap V(F) = \emptyset$ we have $V(G) = V(F)$.

Given an LP-type problem $(H, w, W, \leq)$ the pair (H, V) is a violator space which is the most general framework to which apply the randomized algorithm described later. In a violator space (H, V) of combinatorial dimension δ , in order to setup a basis evaluation algorithm we need to define the following primitive operation:

Primitive 1: (Violation test) (H, V) violator space can be implicitly defined by the primitive: given $G \subseteq H$ and $h \in H \setminus G$ decide whether $h \in V(G)$

Consider a suitable violator space and define

$$V(R) := \{h \in G \setminus R \mid w(R \cup \{h\}) \neq w(R)\}$$

the set of violators of R while

$$X(R) := \{h \in R \mid w(R \setminus \{h\}) \neq w(R)\}$$

being the set of extremes of R . The following result holds:

Lemma 2.1 (Sampling Lemma): For a set R of size r uniformly chosen at random from the set $\binom{G}{r}$ of all r -element subsets of G with $(|G| = n)$, define two random variables $V_r : R \mapsto |V(R)|$ $X_r : R \mapsto |X(R)|$ with expected values $v_r := E(V_r)$, $x_r := E(X_r)$. Now for $0 \leq r \leq n$ we have $v_r = x_{r+1} \frac{n-r}{r+1}$.

Therefore in a violator space with $|H| = n$ and combinatorial dimension δ choosing a subset R , the sampling lemma bounds the expected number of violators through $v_r \leq \delta \frac{n-r}{r+1}$.

Clarkson (Clarkson, 1996) envisaged a smart randomized algorithm able to solve violator space problems relying on the expected number of violators bounded according to the sampling lemma. In the *SEB* case, w is represented by the radius of the ball, H is the set of constraints requiring all the points inside the ball, violators are points outside the ball. It can be proved that *SEB* is a violator space problem and Clarkson's algorithm can be used to solve it. The algorithm proceeds in rounds maintaining a voting box that initially contains one voting slip per point. In each round, a set R of r voting slips is drawn at random without repetitions from the voting box and the *SEB* of the corresponding set R is computed. For each new round the number of voting slips for the *violators* points is doubled. The algorithm terminates as soon as there are no violators (no points outside the ball). If $r \approx d^2$, the expected number of rounds is $O(\log n)$ reducing a problem of size n to $O(\log n)$ problems of size $O(d^2)$.

In the Clarkson's algorithm the problem of finding a basis is solved by using a *Trivial* algorithm able to find the solution for subsets of size at most δ . Clarkson's *Basis2* algorithm calls the *Trivial* algorithm as a subroutine, increasing the probability of obtaining a base in further iterations. Consider G as a multiset with $\mu(h)$ (initially set to one) denoting the multiplicity of h . For a set $F \subset G$ the compound multiplicity of all elements of F is $\mu(F) := \sum_{h \in F} \mu(h)$. Sampling from G is done envisaging that it contains $\mu(h)$ copies of every element h . At each round, when the amount of violators is below a threshold, *Basis2* doubles the multiplicity μ (weight) of violator points, which increments the probability of selecting a base in next rounds. Convergence of *Basis2* algorithm relies on the fact that *Trivial* is correct. An iteration of the loop is *successful* if changes the weights of the elements. To estimate how many unsuccessful iterations pass between two successful ones, the sampling lemma bound can be used. Hence, it can be shown that after k δ successful iterations holds

$$2^k \leq E[\mu(B)] \leq n e^{k/3}$$

for every basis B of G with $|G| = n$ and in particular $k < 3 \log n$. Summarizing Clarkson's algorithm *Basis2* computes a basis of G with an expected number of at most $6\delta n \log n$ calls to primitive 1, and an expected number of at most $6\delta \log n$ calls to *Trivial* with sets of size $6\delta^2$.

3. Transductive Support Vector Machines

TSVM can be described as a training set of l labeled examples $\{(\mathbf{x}_i, y_i)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ with labels $y_i = \{\pm 1\}$ and $i = 1, \dots, l$ i.i.d. according to an unknown distribution $p(\mathbf{x}, y)$ together with u unlabeled examples $\{\mathbf{x}_k\}$ with $k = l + 1, \dots, n$ and $n = l + u$ the total number of examples, distributed according to some distribution $p(\mathbf{x})$. Considering $\mathbf{w}$ the vector normal to the hyperplane and the constant b , the problem can be formulated as finding the vector of labels $y_{\mathbf{u}} = \{y_{l+1}, \dots, y_n\}$ ($y_i = \{\pm 1\}$) having the maximal geometric margin with a separating hyperplane $(\mathbf{w}, b)$ solving:

$$\begin{aligned}
I(\mathbf{w}, b, y_{\mathbf{u}}) = \min_{(\mathbf{w}, b, y_{\mathbf{u}})} & \quad \frac{\|\mathbf{w}\|_2^2}{2} + C \sum_{i=1}^l \xi_i^p + C^* \sum_{k=l+1}^n \xi_k^p \\
\text{subj. to} & \quad y_i(\mathbf{w}^T \varphi(\mathbf{x}_i) + b) \geq 1 - \xi_i \quad \xi_i \geq 0 \quad 1 \leq i \leq l \\
& \quad y_k(\mathbf{w}^T \varphi(\mathbf{x}_k) + b) \geq 1 - \xi_k \quad \xi_k \geq 0 \quad l+1 \leq k \leq n
\end{aligned} \tag{1}$$

with $p = 1$ or 2 respectively for *linear* (L_1) losses and *quadratic* (L_2) losses. The first term controls the generalization capacity while the others, through the slack variables ξ_i , the number of misclassified samples. The two regularization parameters (C and C^*) should akin our confidence in the known labels. The decision function in the hypothesis space $f \in \mathcal{F}$ is represented as

$$f(\mathbf{x}_i) = \mathbf{w}^T \varphi(\mathbf{x}_i) + b$$

and $y_i = \text{sign}(f(\mathbf{x}_i))$ assuming there exist a given Hilbert $\mathcal{H}$ space and a mapping $\varphi : \mathbb{R}^d \rightarrow \mathcal{H}$. The mapping sends the examples data into a *feature space* generating the kernel and satisfying the Mercer's conditions. In this work we refer to quadratic losses, since once fixed the $y_{\mathbf{u}}$, the associated Hessian matrix is positive definite bringing to an objective function unique and strictly convex. Moreover, the optimization problem is considered computationally more stable for L_2 losses. However, main results still apply for the L_1 losses case. Early on, two main strategies have been adopted to minimize $I(\mathbf{w}, b, y_{\mathbf{u}})$:

- **local approximation methods:** starting from a tentative labeling $y_{\mathbf{u}}$ they perform a local search in the space of labelings, using effective heuristics, to maximize $I(\mathbf{w}, b, y_{\mathbf{u}})$. These methods are sensible to local minima. For instance, *SVM^{light}* method (Joachims, 1999) uses a local search algorithm to find the solution which may fail in delivering a solution close to the global one.
- **exact combinatorial methods:** fixing the unlabeled vector $y_{\mathbf{u}}$ in $I(\mathbf{w}, b, y_{\mathbf{u}})$ converts the optimization over $(\mathbf{w}, b)$ into a standard *SVM*. Combinatorial methods find the global optimal solution by searching on the entire space of possible labelings of $y_{\mathbf{u}}$ the *SVM* with maximum margin.

Focusing on exact combinatorial optimization of $\mathcal{J}(y_{\mathbf{u}}) = \min_{(\mathbf{w}, b)} I(\mathbf{w}, b, y_{\mathbf{u}})$, the objective becomes to minimize $\mathcal{J}(y_{\mathbf{u}})$ over a set of binary variables. Such an optimization is non convex belonging to the computational class of NP-hard. It can be solved, for instance, by using Branch and Bound (*BB*) (Chapelle, Sindhwani, and Keerthi, 2006) or Integer Programming (*IP*) (Bennett and Demiriz, 1998), but they are computationally very demanding because of the large number of different possible labelings of unlabeled instances.

3.1. Critical analysis of TSVM exact global solution

An examination of the optimization problem shows that the optimal solution does not depend on the labelling of *all* unlabeled data, but on the labeling of the support vectors of the optimal solution due to the sparsity property of *SVMs*.

Observation 1: Assume having a set $\mathcal{L}$ of labelled and $\mathcal{R}$ unlabelled examples. Name $y_{\mathcal{R}}$ the vector of labels solving the TSVM with cost $\mathcal{J}(y_{\mathcal{R}})$ (i.e. the labelling of instances $\mathcal{R}$ with lower cost). Now consider $\text{SVM}_{y_{\mathcal{R}}}$ to be the inductive SVM with labels $(y_{\mathcal{L}} \cup y_{\mathcal{R}})$.

Therefore, adding an unlabelled point x to the set $\mathcal{L} \cup \mathcal{R}$, lying **outside** the margin of $SVM_{y_{\mathcal{R}}}$, induce the optimal labelling for the transduction problem on $\mathcal{L} \cup \mathcal{R} \cup \{x\}$ to be $y_{\mathcal{R} \cup \{x\}} = (y_{\mathcal{R}} \cup \{y_x\})$ (y_x being the label induced by $SVM_{y_{\mathcal{R}}}$ on x).

Proof. Assuming the existence of a different labelling $y'_{\mathcal{R} \cup \{x\}}$ better than $y_{\mathcal{R} \cup \{x\}}$ such that $\mathcal{J}(y'_{\mathcal{R} \cup \{x\}}) < \mathcal{J}(y_{\mathcal{R} \cup \{x\}})$ leads to a contradiction. Consequently $y_{\mathcal{R} \cup \{x\}}$ must be the optimal labelling. In fact, from the assumption we have $\mathcal{J}(y'_{\mathcal{R} \cup \{x\}}) < \mathcal{J}(y_{\mathcal{R} \cup \{x\}})$ but due to the fact that adding a point x lying **outside** the margin of $SVM_{y_{\mathcal{R}}}$ does not change the SVM solution follows $\mathcal{J}(y_{\mathcal{R} \cup \{x\}}) = \mathcal{J}(y_{\mathcal{R}})$. Finally, stems $\mathcal{J}(y'_{\mathcal{R}}) \leq \mathcal{J}(y'_{\mathcal{R} \cup \{x\}})$ because adding one point to a given labelling can only increase or leave unchanged its cost. Therefore $\mathcal{J}(y'_{\mathcal{R}}) \leq \mathcal{J}(y'_{\mathcal{R} \cup \{x\}}) < \mathcal{J}(y_{\mathcal{R} \cup \{x\}}) = \mathcal{J}(y_{\mathcal{R}})$ and consequently $\mathcal{J}(y'_{\mathcal{R}}) < \mathcal{J}(y_{\mathcal{R}})$ a contradiction, due to the hypothesis that $y_{\mathcal{R}}$ represents the optimal labelling for set $\mathcal{R}$. $\square$

Observation 2: Assume having a set $\mathcal{L}$ of labelled and $\mathcal{U}$ unlabelled examples. Consider a subset $\mathcal{R} \subset \mathcal{U}$ with a given vector of labels $y_{\mathcal{R}}$ solving the TSVM for the set $\mathcal{L} \cup \mathcal{R}$ and be $SVM_{y_{\mathcal{R}}}$ the inductive SVM with labels $(y_{\mathcal{L}} \cup y_{\mathcal{R}})$. Assume we don't have points in $\mathcal{U} \setminus \mathcal{R}$ lying **inside** the margin of $SVM_{y_{\mathcal{R}}}$. In this case, the optimal labelling for the TSVM on $\mathcal{L} \cup \mathcal{U}$ is represented by $y_{\mathcal{L} \cup \mathcal{U}} = (y_{\mathcal{L}} \cup y_{\mathcal{U}})$, $y_{\mathcal{U}}$ being the vector of labels induced by the solution of $SVM_{y_{\mathcal{R}}}$ on the set $\mathcal{U}$.

Proof. This can be proved by applying observation 1 for any point in $\mathcal{U} \setminus \mathcal{R}$. Starting with label vector $y_{\mathcal{R}}$, and noticing that any point in $\mathcal{U} \setminus \mathcal{R}$ lies outside the margin of $SVM_{y_{\mathcal{R}}}$ then iteratively applying the observation 1 we end up with the global solution for $\mathcal{L} \cup \mathcal{U}$. $\square$

Observation 3: Necessary condition to find the global optimal solution for the set of points $\mathcal{L} \cup \mathcal{U}$ using the subset $\mathcal{L} \cup \mathcal{R}$ is that $\mathcal{R}$ must contains all support vectors of the global optimal solution in $\mathcal{L} \cup \mathcal{U}$

Proof. This can be proved by applying observation 2. Note that the optimal solution for $\mathcal{L} \cup \mathcal{U}$ can be built from $\mathcal{L} \cup \mathcal{R}$. The only condition is that points in $\mathcal{U} \setminus \mathcal{R}$ do not lie inside the margin. Henceforth, there won't be support vectors in $\mathcal{U} \setminus \mathcal{R}$ being all of them in $\mathcal{R}$. $\square$

According to the observations above we may come to the following conclusion: *Given a TSVM, most of data points may be unnecessary to build the global optimal solution. The set support vectors completely define the global optimal solution (TSVM sparsity property).* This suggests that it can be feasible to obtain the TSVM optimal global solution when the set $\mathcal{L} \cup \mathcal{R}$ includes the whole set of support vectors. Therefore the question becomes how to get the *right* subset of points. We claim that the TSVM solution may be obtained working on a reduced subset of examples $\mathcal{L} \cup \mathcal{R}$ with computational leverages when $|\mathcal{R}| \ll |\mathcal{U}|$. Noticeably, for a TSVM we may show how to build a violator space solving the problem by using randomized algorithms. It is worth noting that this method might be of practical use whenever the number of support vector in the solution of the TSVM problem scales with $O(\log n)$, which is not usually the case. However, the method can still be applied when the solution has few support vectors or when the dimension of the feature space of the problem is relatively small.

4. Randomized algorithms for *TSVM*

In this section we are ready to show that a *TSVM* problem belongs to the class of violator space. Therefore, Clarkson's algorithm can be used to find its optimal global solution. In this case, the weighting function w is represented by $I(\mathbf{w}, b, y_{\mathbf{u}})$ evaluated over subset of constraints $\mathcal{L} \cup \mathcal{R} \subseteq \mathcal{L} \cup \mathcal{U}$ with combinatorial dimension depending on the number of support vectors. Given a subset of partially labeled points, *TSVM* global optimal solution (a basis) can be obtained using an exact method like *IP* or *BB* (our *Trivial* algorithm). Moreover we need to define a violation test as Clarkson's algorithm rely on the probability of selecting a basis increasing the weight of violating points. Then we will show that violators can be easily detected as the remaining points lying in the *TSVM* separating margin. *TSVM* may also arouse a formulation in terms of violator space problem. Endowing the constrained formulation of *TSVM* problem and violator space definition we formally propose:

Proposition 4.1: *Given a TSVM with constraints $H_{\mathcal{F}}$, $w_{\mathcal{F}}(G_{\mathcal{F}}) : 2^{H_{\mathcal{F}}} \rightarrow W_{\mathcal{F}}$ a mapping defined for each subset $G_{\mathcal{F}} \subseteq H_{\mathcal{F}}$ as $w_{\mathcal{F}}(G_{\mathcal{F}}) = \min_{H_{\mathcal{F}}} \mathcal{F}(G_{\mathcal{F}}) = \mathcal{F}(\mathbf{w}_g, b_g, y_g)$ and $W_{\mathcal{F}}$ bounded and linearly ordered by $\leq$, the quadruple $(H_{\mathcal{F}}, w_{\mathcal{F}}, W_{\mathcal{F}}, \leq)$ represents an associated violator space problem.*

In order to verify locality and monotonicity we need to prove the lemma:

Lemma 4.2: *Given a TSVM with constraints $H_{\mathcal{F}}$ and a subset $G_{\mathcal{F}} \subseteq H_{\mathcal{F}}$ with global optimum $\mathcal{F}(G_{\mathcal{F}})$, adding a constraint $h_{\mathcal{F}} \in H_{\mathcal{F}}$ to $G_{\mathcal{F}}$ will change the global optimum according to $\mathcal{F}(G_{\mathcal{F}}) \leq \mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}})$*

Proof. The lemma may be easily proved by contradiction. Suppose the lemma is not true then $\mathcal{F}(G_{\mathcal{F}}) > \mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}})$. Now the solution to the *TSVM* problem on the set of constraints $G_{\mathcal{F}} \cup h_{\mathcal{F}}$ is also a feasible solution for the *TSVM* problem on the set of constraints $G_{\mathcal{F}}$. Hence the hypothesis $\mathcal{F}(G_{\mathcal{F}}) > \mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}})$ contradict the minimality of the *TSVM* solution $\mathcal{F}(G_{\mathcal{F}})$. However, as we need to provide the primitive to build the violator space, we need a more constructive proof of the lemma following the incremental solution of the *TSVM*. Therefore consider the subset of constraints $G_{\mathcal{F}} \subseteq H_{\mathcal{F}}$, getting the objective $\mathcal{F}(G_{\mathcal{F}})$ means to find the global optimal solution of the *TSVM* problem using a subset of labeled and unlabeled data $\mathcal{L} \cup \mathcal{R} \subseteq \mathcal{L} \cup \mathcal{U}$. With an exact method the quest for the global optimum over the subset of constraints $G_{\mathcal{F}}$ entails that among the 2^r possible configurations of vector of labels $y_{\mathbf{r}}$, we cast the corresponding *SVM* with minimum objective $\mathcal{F}(G_{\mathcal{F}})$ which solving the dual problem can be written as $\mathcal{W}(\alpha_s, b)$. Adding another constraint $G_{\mathcal{F}} \cup h_{\mathcal{F}}$ with vector of labels $y_{\mathbf{r}+1}$ means that among 2^{r+1} possible configurations, we select the corresponding *SVM* with minimum objective $\mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}})$ which solving the dual can be expressed as $\mathcal{W}(\alpha'_s, \alpha'_h, b')$. Hence, the relationship between the two objective values can be found considering the incremental training analysis of a *SVM* as reported in (Cauvemberghs and Poggio, 2001) with some differences in our case gathered from quadratic losses.

The *SVM* dual problem might be obtained introducing $\alpha_i \in \mathbb{R}^+$ the Lagrangian multipliers and considering that for the optimal solution the *KKT* conditions must be satisfied. After elimination of $\mathbf{w}, \xi_i$ (note that for quadratic losses if $\alpha_i \neq 0$ then $\xi_i \neq 0$) one get the dual problem

$$\mathcal{W}(\alpha, b) = \max_{(\alpha \in \mathbb{R}^+, b \in \mathbb{R})} \sum_{i=1}^s \alpha_i - \frac{1}{2} \sum_{i,j=1}^s y_i y_j \alpha_i \alpha_j (K(\mathbf{x}_i, \mathbf{x}_j) + \frac{\delta_{ij}}{\hat{C}}) + b \sum_{i=1}^s \alpha_i y_i$$

(where $\hat{C} = C$ or C^* accordingly) and defining

$$\Omega = [\Omega_{ij}] = [y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \frac{\delta_{ij}}{\hat{C}}]$$

($\delta_{ij} = 1$ if $i = j$, 0 otherwise), the *KKT* conditions for the support vectors ($\alpha_i \neq 0$) are $\sum_{i=1}^s y_i \alpha_i = 0$ and $g_s = y_i f(\mathbf{x}_i) + \xi_i - 1 = 0$ which defining the vectors Y and α can be equivalently expressed in a more compact form as

$$\begin{bmatrix} 0 & Y^T \\ Y & \Omega \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2)$$

The *KKT* for non support vectors ($\alpha_i = 0$) can be expressed as $g_r = y_i f(\mathbf{x}_i) - 1 \geq 0$. Finally the solution of the dual problem α_s gives the maximum value of the $\mathcal{W}(\alpha_s)$ which substituting the *KKT* conditions and defining

$$\hat{\Omega}_s = \begin{pmatrix} 0 & Y_s^T \\ Y_s & \Omega_s \end{pmatrix}$$

can be written as $\mathcal{W}(\alpha_s) = \frac{1}{2} \alpha_s^T \hat{\Omega}_s \alpha_s$. In incremental training adding a new point h consists adiabatically incrementing α_h starting from $\alpha_h = 0$ and changing the other parameters until the *KKT* conditions for point h are fulfilled. The increment of α_h is done ensuring that the *KKT* conditions for the other points are maintained. In the following, $\Delta\alpha$ is defined as the difference between the new alpha value α' and the old value α . The *KKT* conditions before and after α_h update connect the variations of the support vectors $\Delta\alpha_s$, Δb and the added one $\Delta\alpha_h$ according to:

$$\begin{bmatrix} 0 \\ \Delta g_s \\ \Delta g_r \\ \Delta g_h \end{bmatrix} = \begin{bmatrix} 0 & y_s^T \\ y_s & \Omega_{ss} \\ y_r & \Omega_{rs} \\ y_h & \Omega_{hs} \end{bmatrix} \begin{bmatrix} \Delta b \\ \Delta \alpha_s \end{bmatrix} + \begin{bmatrix} y_h \\ \Omega_{hs}^T \\ \Omega_{hr}^T \\ \Omega_{hh}^T \end{bmatrix} \Delta \alpha_h$$

For support vectors $\Delta g_s = 0$ then

$$\begin{bmatrix} \Delta b \\ \Delta \alpha_s \end{bmatrix} = - \begin{bmatrix} 0 & y_s^T \\ y_s & \Omega_{ss} \end{bmatrix}^{-1} \begin{bmatrix} y_h \\ \Omega_{hs}^T \end{bmatrix} \Delta \alpha_h = -Q_s^{-1} z_h \Delta \alpha_h \quad (3)$$

gathering

$$\begin{bmatrix} \Delta g_r \\ \Delta g_h \end{bmatrix} = \begin{bmatrix} y_h & \Omega_{hs} \\ y_r & \Omega_{rs} \end{bmatrix} \begin{bmatrix} \Delta b \\ \Delta \alpha_s \end{bmatrix} + \begin{bmatrix} \Omega_{hr}^T \\ \Omega_{hh}^T \end{bmatrix} \Delta \alpha_h$$

Tackling those equations we may infer the change due to the presence of a new data as a function of the $\Delta\alpha_h$. Unfortunately, with a generic value $\Delta\alpha_h$ we cannot directly use the systems above to obtain the new state, due to the possible change of composition in the sets of support vectors and non support vectors while α_h increases. Handling this situation demands a book keeping of possible adiabatic modifications. So, the procedure consists in incrementing α_h to the *maximum until one of the following cases arise*: the problem is solved, or the keeping of the *KKT* conditions of the other points forces to

change the set of support vectors¹. In particular dealing with the increment of $\Delta\alpha_h > 0$ may bring to changes in the support vectors composition depending on the amount of $\Delta\alpha_h$. Here we detail the possible situations:

- $\Delta\alpha_h = 0$ and $g_h \geq 0$, the new point with the chosen label y_h won't change the set of support vector and we keep the same solution $\mathcal{W}(\alpha_s)$
- $\Delta\alpha_h > 0$ and $g'_h = 0$ (g value *after* the increment of α_h), the new point with the chosen label y_h is a new support vector while the *support vector set does not change composition* (no migrations) hence the new KKT can be written as

$$\begin{bmatrix} 0 & y_s^T & y_h \\ y_s & \Omega_{ss} & \Omega_{hs}^T \\ y_h & \Omega_{sh} & \Omega_{hh} \end{bmatrix} \begin{bmatrix} b' \\ \alpha'_s \\ \alpha'_h \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

ensuing a connection with the old solution across the Sherman-Morison-Woodbury formula for block matrix inversion

$$\begin{bmatrix} 0 & y_s^T & y_h \\ y_s & \Omega_{ss} & \Omega_{hs}^T \\ y_h & \Omega_{sh} & \Omega_{hh} \end{bmatrix}^{-1} = \begin{bmatrix} Q_s^{-1} & 0 \\ 0 & 0 \end{bmatrix} + q_h^{-1} \begin{bmatrix} -Q_s^{-1} z_h \\ 1 \end{bmatrix} \begin{bmatrix} -z_h^T Q_s^{-1} & 1 \end{bmatrix}$$

with

$$q_h = [-Q_s \ z_h \ 1]^T \begin{bmatrix} 0 & y_s^T & y_h \\ y_s & \Omega_{ss} & \Omega_{hs}^T \\ y_h & \Omega_{sh} & \Omega_{hh} \end{bmatrix} \begin{bmatrix} -Q_s \ z_h \\ 1 \end{bmatrix} = \Omega_{hh} - z_h^T Q_s^{-1} z_h \quad (4)$$

where $q_h \geq 0$ due to semidefinite positive kernel wherefrom we can express the updated values

$$\Delta\alpha_h = \alpha'_h = \frac{\max(0, 1 - y_h b - \Omega_{sh} \alpha_s)}{\Omega_{hh} - z_h^T Q_s^{-1} z_h} = \frac{1 - y_h f(\mathbf{x}_h)}{q_h} = \frac{\eta_h}{q_h}$$

and

$$\begin{bmatrix} b' \\ \alpha'_s \end{bmatrix} = \begin{bmatrix} b \\ \alpha_s \end{bmatrix} - Q_s^{-1} z_h \Delta\alpha_h$$

using $\eta_h = \max(0, 1 - y_h f(\mathbf{x}_h))$ with optimum update

$$\mathcal{W}(\alpha'_s, \alpha'_h) = \mathcal{W}(\alpha_s) + \frac{1}{2} \Delta\alpha_h^2 q_h = \mathcal{W}(\alpha_s) + \frac{1}{2} \frac{\eta_h^2}{q_h}$$

which entails $\mathcal{W}(\alpha_s) \leq \mathcal{W}(\alpha'_s, \alpha'_h)$

- $\Delta\alpha_h > 0$ and $g'_h < 0$, the new point with the chosen label y_h is a new support vector while *the support vectors set changes composition*. For quadratic losses this

¹Note that in the case of quadratic losses we only need to divide set of points in two sets: supports (S) and others (O), while in the case of linear losses, points can belong to three different sets: supports (S), errors (E) and others (O).

means two possibilities: weather one support vector, say it k with $\alpha_k > 0$ and $g_k = 0$, becomes non support vector hence $\alpha'_k = 0$ and $g'_k \geq 0$ or viceversa (from $\alpha_k = 0$ and $g_k \geq 0$ to $\alpha'_k > 0$ and $g'_k = 0$). In each one on this cases the $g'_h < 0$ condition holds. As the set of support vector does not change *during* increment $\Delta\alpha_h$ (it changes *after* the increment), the global optimum after and before each increment according may be evaluated as:

$$\begin{aligned}\mathcal{W}(\alpha'_s, \alpha'_h) &= 1^T \alpha'_s + \alpha'_h - \frac{1}{2} \begin{bmatrix} \alpha'_s \\ \alpha'_h \end{bmatrix}^T \begin{bmatrix} \Omega_{ss} & \Omega_{sh} \\ \Omega_{hs} & \Omega_{hh} \end{bmatrix} \begin{bmatrix} \alpha'_s \\ \alpha'_h \end{bmatrix} \\ \mathcal{W}(\alpha_s, \alpha_h) &= 1^T \alpha_s + \alpha_h - \frac{1}{2} \begin{bmatrix} \alpha_s \\ \alpha_h \end{bmatrix}^T \begin{bmatrix} \Omega_{ss} & \Omega_{sh} \\ \Omega_{hs} & \Omega_{hh} \end{bmatrix} \begin{bmatrix} \alpha_s \\ \alpha_h \end{bmatrix}\end{aligned}$$

where $\alpha'_s = \alpha_s + \Delta\alpha_s$ and $\alpha'_h = \alpha_h + \Delta\alpha_h$ allowing to evaluate the global optimum increment as

$$\Delta\mathcal{W} = \mathcal{W}(\alpha'_s, \alpha'_h) - \mathcal{W}(\alpha_s, \alpha_h)$$

. Now using equations (3) to express the change in the global optimum as a function of $\Delta\alpha_h$ can be written as:

$$g'_h = -1 + b'y_h + \sum_{j=1}^s \Omega_{hj}\alpha'_j + \Omega_{hh}\alpha'_h < 0$$

and

$$g_h = -1 + by_h + \sum_{j=1}^s \Omega_{hj}\alpha_j + \Omega_{hh}\alpha_h < 0$$

one can easily show that

$$\Delta\mathcal{W} = \frac{1}{2}q_h\Delta\alpha_h^2 - g'_h\Delta\alpha_h \geq 0$$

because g'_h is negative (see assumptions of this case) and q_h is always positive.

We have assumed that the set of equations describing the evolution of the α_i for all points does not change during each step. However, while increasing the $\Delta\alpha_h$, we have to check if the *KKT* conditions are still verified. Consequently, if such conditions are not longer fulfilled for a given point, his status has to change (from support to non support vector or vice versa depending on the case) and therefore the equations must be modified according to the bookkeeping. Finally it remains to see if the objective function is still monotonous when we change the equations and the support vector set. It is easy to see that for the L_2 losses the bookkeeping does not change the value of the objective function. In fact bookkeeping is necessary when one support vector becomes a non-support vector or the other way round. In both cases, *the α_k of the point changing its status is 0* (if the point is a new support vector its α_k will increase up from 0, or if it leaves the status of support vector, its alpha value has been decreased to 0). When $\alpha_k = 0$ adding or removing it does not modify the objective function. Consequently the objective function does not change by doing a bookkeeping.

We end up to the conclusion that for an incremental *SVM* the optimum grows monotonically. Henceforth, using an exact method to solve the *TSVM* over $G_{\mathcal{F}}$ means that we look for the global optimal solution among the 2^r *SVMs*, corresponding to the possible labellings $\mathcal{F}(G_{\mathcal{F}}) = \min\{y_r \mid 1 : 2^r \mid \mathcal{W}(\alpha_s, b)\}$ while over $G_{\mathcal{F}} \cup h_{\mathcal{F}}$ we look for the global optimal solution among the 2^{r+1} *SVMs*, corresponding to the possible labellings $\mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}}) = \min\{y_{r+1} \mid 1 : 2^{r+1} \mid \mathcal{W}(\alpha'_s, \alpha'_h, b')\}$. Then we may realize that the labelling for the global optimum over $G_{\mathcal{F}}$, except for the new constraint $h_{\mathcal{F}}$, may be different from the labelling corresponding to the global optimum over $G_{\mathcal{F}} \cup h_{\mathcal{F}}$. Therefore, thanks to the fact that the *SVM* global optimal solution monotonically increases moving from $G_{\mathcal{F}}$ to $G_{\mathcal{F}} \cup h_{\mathcal{F}}$ and that the global optimal solution for an exact method seeks for the minimum among the all possible *SVMs* follows $\mathcal{F}(G_{\mathcal{F}}) \leq \mathcal{F}(G_{\mathcal{F}} \cup h_{\mathcal{F}})$. $\square$

Last lemma providing monotonicity of the incremental optimal solution for a *TSVM* allow to infer that $(H_{\mathcal{F}}, w_{\mathcal{F}}, W_{\mathcal{F}}, \leq)$ represents a violator space (from LP-type) problem and we report here the proof for the proposition 4.1.

Proof. We only need to show that monotonicity and locality hold:

- **Monotonicity:** Given the subsets of constraints $F_{\mathcal{F}}$ and $G_{\mathcal{F}}$ such that $F_{\mathcal{F}} \subseteq G_{\mathcal{F}} \subseteq H_{\mathcal{F}}$ we can always build $G_{\mathcal{F}}$ from $F_{\mathcal{F}}$ through an incremental process adding new constraint $h \in H_{\mathcal{F}} \setminus F_{\mathcal{F}}$. From lemma (4.2) follows $\mathcal{F}(F_{\mathcal{F}}) \leq \mathcal{F}(G_{\mathcal{F}})$
- **Locality:** Given the subsets of constraints $F_{\mathcal{F}}$ and $G_{\mathcal{F}}$ such that $F_{\mathcal{F}} \subseteq G_{\mathcal{F}} \subseteq H_{\mathcal{F}}$ and a constraint $h \in H_{\mathcal{F}}$. If $\mathcal{F}(F_{\mathcal{F}}) = \mathcal{F}(G_{\mathcal{F}})$ the two sets entail the same global solution. Adding incrementally the constraint $h \in H_{\mathcal{F}}$ to the subset $G_{\mathcal{F}}$ with $\mathcal{F}(G_{\mathcal{F}}) \leq \mathcal{F}(G_{\mathcal{F}} \cup \{h\})$ conveys that the new constraint has changed the global solution. Consequently adding $h \in H_{\mathcal{F}}$ to $F_{\mathcal{F}}$ will also change the global solution then $\mathcal{F}(G_{\mathcal{F}}) = \mathcal{F}(F_{\mathcal{F}}) \leq \mathcal{F}(F_{\mathcal{F}} \cup \{h\})$

$\square$

Facing our definition of the associated violator space problem for a *TSVM*, a basis is represented by a set of the support vectors in the global solution, the cardinality of the largest one being the combinatorial dimension. At this stage we have shown that is indeed feasible to formulate a *TSVM* as violator space problem. In addition, adopting the Clarkson's algorithm we may also use the violator mapping through the violation test.

Primitive 2: (*TSVM violation test*) Given a *TSVM* with global solution $\mathcal{F}(G_{\mathcal{F}})$ over subset of constraints $G_{\mathcal{F}}$ and decision function $f_{G_{\mathcal{F}}}(\mathbf{x}_h)$, any other constraint $h \in H_{\mathcal{F}} \setminus G_{\mathcal{F}}$ will violate the solution ($h \in V_{\mathcal{F}}(G_{\mathcal{F}})$), if $\eta_h = \max(0, 1 - y_h f_{G_{\mathcal{F}}}(\mathbf{x}_h)) \geq 0$ with the label obtained through $y_h = \text{sign}(f_{G_{\mathcal{F}}}(\mathbf{x}_h))$.

Beforehand we have shown that we may associate a violator space problem to a *TSVM* once we deal with an exact method to get the optimal global solution. Remarkably, Clarkson's randomized method is viable to solve a *TSVM* acquiring a basis using the violation test primitive 2. Efficiency relies on the sparsity property of *TSVM*.

5. Algorithm implementation

Exact methods like Integer Programming (*IP*) (Bennett and Demiriz, 1998) and Branch and Bound (*BB*) (Chapelle et al., 2006) have been investigated for *TSVM*. The basic

Algorithm 1 STSVM ($X_l, Y_l, X_u, maxit, r$)**Require:** $r = 2g_0\delta \ll |X_l \cup X_u|$ **repeat**Choose X_r at random according to μ distribution: $X_r \subset \binom{u}{r}$ $Base(obj_R, Y_r, \alpha, \mathbf{w}, b) := BB(X_l, X_r, Y_l)$ // call branch and bound $Y_{u \setminus r} := \text{sign}(f_{(\mathbf{w}, b)}(\mathbf{x}_{u \setminus r}))$ // compute labels... $\xi_{u \setminus r}^2 := \max(0, 1 - Y_{u \setminus r} f_{(\mathbf{w}, b)}(\mathbf{x}_{u \setminus r}))^2$ // ... and slacks $V_{Base} := \{h \in \mathcal{U} \setminus \mathcal{R} : \xi_{u \setminus r} \neq 0\}$ // check violators**if** $\mu(V_{Base}) \leq \mu(\mathcal{U})/g_0$ **then** $\mu(h) := 2\mu(h), \quad h \in V_{Base}$ // update weights of violators**end if** $OBJ_R := obj_R + 1/2 \sum_{u \setminus r} \eta_{u \setminus r}^2 / q_{u \setminus r}$ // update best... $Base_{best} := \min_{OBJ} (Base(OBJ_R, Y_r, \alpha, \mathbf{w}, b), Base_{best})$ // ...known solution**until** $maxit$ or $V_{Base} = \emptyset$ **if** $V_{Base} = \emptyset$ **then****return** $Base(obj_R, Y_r, \alpha, \mathbf{w}, b)$ // global optimum**else****return** $Base_{best}$ // best solution found**end if**

idea of *IP* for *TSVM* is to add an integer decision variable indicating the class for each point in the working set. Hence the solution can be found using any mixed integer programming code provided that the computer resources are suitable for the size of the problem. Branch and Bound is another method able to solve combinatorial optimization problems using an enumerative technique to implicitly and efficiently search into the entire space of solutions.

Clearly both methods, or even any other exact *TSVM* could be used to implement the *Trivial* algorithm. For practical reasons in the following we only focus on *BB* implementation indicating with *BBTSVM* the one devised by Chapelle for *TSVM*. It solves the problem over a set of binary variables where each evaluation of $\mathcal{J}(y_{\mathbf{u}})$ embodies an inductive *SVM*. *BBTSVM* minimizes over 2^u possible choices of $y_{\mathbf{u}}$. At each node in the binary search tree, a partial labelling of the data sets is generated and for the corresponding sons casts the labelling of some unlabelled data. Core of the *BBTSVM* is a subroutine akin the primitive evaluation of a *SVM* on the data already labelled as we proceed in the search tree. *SVM* are usually solved as constrained quadratic programming problem in the dual, but they can also be worked out in the primal as unconstrained. *BBTSVM* uses this method requiring few iterations to get an approximate solution with a (worst case) complexity $O(n^3)$. *SVM* being convex quadratic program problems are polynomial-time solvable, the currently best method raising (worst case) a complexity of $O(n^{3/2}d^2)$. By the enumeration process *BBTSVM* may reach a complexity from $O(2^n n^3)$ to $O(2^n n^{3/2}d^2)$ depending on the *SVM* solver used.

5.1. Implementation of randomized TSVM

Having set out the matter in this way, we are ready to encompass the details of our implementation of a randomized version for an exact method to solve *TSVM* which we will refer as *STSVM*. The algorithm (1) implementing the *STSVM* slightly revises the Clarkson's scheme. In general we may ignore the combinatorial dimension of the corresponding violator space. In fact, apart for the linear case, theoretical bound on the number of support vectors are not of practical use. Henceforth, we start with a fixed given

value for r_o . According to *Basis2*, to stress the probability of selecting a given violator in further rounds we double the corresponding weight whenever the slack $\xi_{u \setminus r} \neq 0$. In the quadratic loss implementation for example, where $\eta_{u \setminus r} = \alpha_{u \setminus r}$ it means that the point would change the margin if added to the basis. The basic implementation of *Trivial* makes use of an implementation of an exact method working over random samples $\mathcal{L} \cup \mathcal{R} \subseteq \mathcal{L} \cup \mathcal{U}$. As envisaged, in the practical implementation of *Trivial* we will make use of *BB*. However, using a different exact algorithm to provide the implementation of *Trivial* does not change the convergence of *STSVM* but can only affects the performances. Two stop conditions are foreseen: (1) $V_{Base} = \emptyset$ (*no violators condition*: returns global optimum) or (2) max number of iterations (returns best known solution). In the last case the best known solution is represented by the minimum of $\mathcal{F}_{best} = \min_{OBJ_R} BB(\mathcal{L} \cup \mathcal{R})$ where $OBJ_R = obj_R + \Delta(V_R) = obj_R + 1/2 \sum_{u \setminus r} \eta_{u \setminus r}^2 / q_{u \setminus r}$, with $q_{u \setminus r}$ the correction factor in the incremental update of the equation 4. Such criterium takes into account obj_R the best result from *BB*, also minimizing the violators contribution $\Delta(V_R)$. Our implementation of *BB* is able to optimize the undergoing *SVM* either in the primal or dual formulation with quadratic or linear losses.

5.2. Convergence and complexity of STSVM

Applying the sampling lemma, the expected value $E[V_{Base}]$ for elements in $\mathcal{U}$ (with $u = |\mathcal{U}|$) violating the random set R , (casting $r = 2g_0\delta < r_{max}$ with $1 \leq g_0 \leq \delta$) can be bound as

$$E[|V_{Base}|] \leq \delta \frac{\mu(\mathcal{U}) - r}{r + 1} < \delta \frac{\mu(\mathcal{U})}{2g_0\delta} = \frac{\mu(\mathcal{U})}{2g_0}$$

Markov inequality predicts

$$Pr(|V_{Base}| \leq \frac{\mu(\mathcal{U})}{g_0}) \geq Pr(|V_{Base}| \geq 2E[|V_{Base}|]) \leq 1/2$$

implying that the expected number of rounds is at most twice as large as the number of successful ones. Let $\mathcal{SV}$ be the set of support vectors for the *TSVM*, at each successful iteration some $x_i \in \mathcal{SV}$ won't be in $\mathcal{L} \cup \mathcal{R}$ hence $\mu(x_i)$ gets doubled. Since $|\mathcal{SV}| \leq \delta$ it means some x_i gets doubled at least once every δ successfully round and after k rounds $\mu(x_i) \geq 2^{k/\delta}$. On the other hand, every successful round raise the total weight by at most

$$(1 + \frac{\delta}{r}) = (1 + \frac{1}{2g_o})$$

giving the bound

$$2^{k/\delta} \leq E[\mu(\mathcal{U})] \leq u(1 + \frac{1}{2g_0})^{k\delta} \leq n e^{\frac{k\delta}{2g_0}} \leq n e^{\frac{k}{2}}$$

. After $k\delta$ successful iterations the lower bound exceeds the upper when $k \geq 2 \log n$. Finally the expected number of violation tests is at most $O(\delta n \log n)$ while the expected number of calls to *BBTSVM* is at most $g_0 \log n = O(\delta \log n)$ and sets R have average size $r = 2g_0\delta = O(\delta^2)$ (at most r_{max}).

Time complexity of *STSVM* shows that the sparsity property allows for a relative gain with respect to the use of a *BB* over the whole set of data from $O(2^n n^{3/2} d^2)$ to

Figure 2. (a) Distribution and exact solution for the **two moons** data set (4.000 unlabelled, 2 labelled as triangle and cross). (b) Weights distribution for the whole set of points at the final round. Support vectors of the optimal solution are encircled.

$O(\delta \log n 2^r (r+l)^{3/2} d^2)$. Our method being more effective when the number of support vectors is much lower than the number of instances.

6. Experiments

In this section we briefly describe results obtained using the proposed randomized method in two well known benchmark problems that have not been previously solved using exact methods.

STSVM stands out on the well known benchmark problem of **two moons** now composed of 4.000 unlabeled examples (Figure 2). Two moons problem has been reported burdensome to solve for state-of-the-art transduction methods (Chapelle, 2008). Table 1 shows error rates produced by some of these methods in a small data set containing 200 unlabeled points. All these methods fall in local minima very far from the global one. However, this problem has been solved previously applying *BB* over the space of all possible labelling of instances (Chapelle et al., 2006). Unfortunately, albeit the method is able to find out the optimal solution, it has an exponential time complexity on the number of unlabeled instances, consequently, it is only able to solve the **two moons** for data sets up to few hundreds of unlabeled instances. The method described in this paper finds out the optimal solution but instead of scaling with the number of instances, it scales with the number of support vectors. In the case of the **two moons**, the number of support vectors does not scale with the number of points. Henceforth, facing with 4.000 unlabeled examples through *STSVM* allows us to find out the *exact* solution within few minutes.

STSVM is also able to find the *exact* solution for another well known benchmark problem that has not been previously solved using exact methods: the **Coil20** data set (described in (Nene, Nayar, and Murase, 1996)). It contains 1440 images of 20 objects photographed with different perspectives (72 images per object) with 2 labeled images for each object. **Coil20** belongs to multi-class problems commonly solved through a one-vs-all scheme. This scheme, builds a *SVM* for each class assigning labels according to the class returning a maximum output for a given point, which is a measure of confidence of classification for that point. Table 1 shows the success of state-of-the-art transduction

data sets	SVM	∇S^3VM	cS^3VM	CCCP	S^3VM^{light}	∇DA	Newton	STSV
TwoMoons	50.2	61	37.7	63.1	68.8	22.5	11	0
Coil3	66.6	61.6	61	56.6	56.7	61.6	61.5	0
Coil20	24.1	25.6	30.7	26.6	25.3	12.3	24.1	1.4

Table 1. Error rate on datasets for supervised SVM, state of the art transduction algorithms (as reported in (Chapelle, 2008)), and *STSV*.

algorithms on this data set as is described in Chapelle (2008). *BBTSVM* is not able to solve this problem, due to the large number of instances, albeit it can manage the *Coil3* with pictures of solely 3 classes but hard to discriminate. *STSV* finds a solution with 0 error for *Coil3* data set while Table 1 illustrate the errors produced by state-of-the-art transduction methods (from Chapelle (2008)). In the last two columns the error rate of a L_2 losses (used in the search tree of *STSV*) applied only to the labeled examples are reported, as well as the results obtained by *STSV* with the one-vs-all scheme acting over labeled and unlabeled. Remarkably, the difference between the baseline results shown in Chapelle (2008) and ours, come from the different losses used for the *SVM* (L_2 in our case, L_1 for the others). Even though *STSV* might be used with different losses, here we stress the error reduction (from to 10.9% in *Coil20*, and 0% in *Coil3*), more that its absolute value. However, examining the 20 *TSVMs* produced in the one-vs-all scheme (one for each class), it may be noticed that some *TSVMs* have not found 0 violators after 100 iterations or that when a solution with 0 violators was found, it did not show the right balance between positive and negative examples. On the other side, some *TSVMs* returned solutions with 0 violators and the right rate between positive and negative examples. We considered these *TSVMs* as good predictors for the classes they represent and keep them. The points labeled as positive for these classes were removed from the whole data set, delivering a smaller one. Afterwards, we tried to solve this smaller data set applying the one-vs-all scheme again on the remaining classes. As the problem has been simplified, now appear new good predictors for some classes and the procedure is repeated until the whole data set is completely solved. The final error following this procedure was 20 errors from the 1400 unlabeled initial examples reducing the error for *Coil20* to 1.4%. Quite remarkably the irreducible errors were produced between classes 5 and 19, the last classes for which the method found a good predictor. It seems that for these 2 classes does not hold the maximum margin hypothesis and that some additional constraints must be used to correctly classify them; in fact only modifying the ratio amongst labeled and unlabeled largely increasing the formers, this problem can be correctly solved.

7. Conclusions and Future Work

In this paper, we have shown an original interpretation of the *TSVM* problem in terms of violator space, which is able to extend the use of *any exact method* to find the optimal solution but now scaling in time complexity with the number of support vectors instead of the number of points. The most suitable situation for our method apply with datasets entailing a small amount of support vectors, independently of the size of the data set. Limitations of our approach appear when the size of the support vectors set is higher than few hundreds, a common situation in real data sets. In the future, we plan to investigate an implementation using very sparse SVMs formulations, which might allow to extend the application of our method to larger datasets.

Also, preliminary experiments with larger benchmark datasets (where the number of support vector were in the range of hundreds) show that the method is, as we expected,

not able to find the optimal solution in a reasonable time. However, in these cases we kept the best solution as described in section 5.1. In all cases, the returned solution was better than the returned one by a SVM on only the labeled examples. These experiments encourage us to explore error bounds for the proposed methods and try to apply it to find good approximations to the optimal solution.

Finally, we consider interesting a possible interpretation of the weight obtained for each example in the randomized method. Samples with high weights usually appear as violators which could help us to identify points interesting for the final solution even if the method is not able to find the exact one.

Acknowledgements

This work was partially supported by the FI-DGR programme of AGAUR ECO/1551/2012.

References

- K. Bennett and A. Demiriz. Semi supervised support vector machines. *In Adv. in Neural Information Processing Systems*, 12:368–374, 1998.
- G. Cauvemberghs and T. Poggio. Incremental and decremental support vector machine learning. *Adv. in Neural Inform. procesing*, MIT Press(13):409–415, 2001.
- O. Chapelle. Optimization techniques for semi supervised support vector machines. *Journal of Machine Learning Research*, 9:203–233, 2008.
- O. Chapelle, V. Sindhwani, and S. Keerthi. Branch and bound for semi-supervised support vector machines. *In Adv. in Neural Information Proc. Systems*, 2006.
- K. L. Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM*, 42, 1996.
- B. Gartner, L. Rust J. Matousek, and P. Skovron. Violator spaces: Structure and algorithms. *Discrete Applied Mathematics*, 156(11), 2008.
- Balcazar J., Dai Y., Tanaka J., and Watanabe O. Provably fast training algorithms for support vector machines. *Theory of Computing Systems*, 2008.
- T. Joachims. Transductive inference for text classification using support vector machines. *In International Conference on Machine Learning*, 1999.
- S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-20). Technical report, Columbia University, USA, 1996.
- V. Vapnik. *The nature of statistical learning theory*. Springer, New York, 1995.