2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

On the discardability of data in Support Vector Classification problems

Simone Del Favero, Damiano Varagnolo, Francesco Dinuzzo, Luca Schenato, Gianluigi Pillonetto

Abstract— **We analyze the problem of data sets reduction for support vector classification. The work is also motivated by distributed problems, where sensors collect binary measurements at different locations moving inside an environment that needs to be divided into a collection of regions labeled in two different ways. The scope is to let each agent retain and exchange only those measurements that are mostly informative for the collective reconstruction of the decision boundary. For the case of separable classes, we provide the exact conditions and an efficient algorithm to determine if an element in the training set can become a support vector when new data arrive. The analysis is then extended to the non-separable case deriving a sufficient discardability condition and a general data selection scheme for classification. Numerical experiments relative to the distributed problem show that the proposed procedure allows the agents to exchange a small amount of the collected data to obtain a highly predictive decision boundary.**

Index Terms— **distributed classification, support vector machines, model reduction, distributed machine learning, simplex, convex analysis**

I. Introduction

Wireless communication and new low-cost technologies are promoting the deployment of networks containing a large number of sensors, often called also *nodes*, or *agents*, able to communicate and collaborate to achieve a common objective. Examples of applications of these networks abound and we cite e.g. environmental monitoring and remote surveillance of hazardous areas [1], [2]. Although these new technologies promise great advantages, they also lead to challenging novel theoretical and a practical questions, e.g. related to distributed learning [3] and information compression [4].

Motivated by collaborative ad-hoc wireless sensor networks where the nodes are randomly distributed over a region of interest and collaborate to achieve a common goal, we study the problem of discarding data that are non informative for support vector machines (SVM) based classification algorithms, see [5], [6], [7], [8]. A general application motivating this paper is a remote surveillance system where agents move inside an environment collecting binary measurements at different locations with the aim to divide the area into a collection of regions labeled in two different ways. A practical example is coastal monitoring by a swarm of underwater agents checking for zones that can be navigated by submarines carrying illegal narcotics. From the distributed framework setting, we assume that the agents have limited communication capabilities. Hence it is crucial to retain and exchange only those measurements that are mostly informative for the collective reconstruction of the decision boundary, a problem similar to the one of obtaining bounded-sized solutions, see e.g. [9] and references therein. We remark that, in this framework, it may be worth increasing the computational burden if this reduces the communication one.

Relevant to the problem treated in this paper is the literature on classification via SVM describing algorithms to update the training set once new data arrive. The developed strategies aim to keep not only the data that are support vectors (SVs) but also those that may become SVs with high probability. One of the first methods applied was the so-called chunking [10], where at each step a quadratic programming problem is constructed starting from all the previously computed SVs and some of the currently misclassified data. There is also an entire field where authors propose on-line recursive algorithm for training support vector machines one vector at a time, usually called *Incremental Machine Learning*, see [11]. The update of the decision boundary is performed selecting that part of the training set that is considered mostly informative. Such selection is typically done by hyperplane-distance based heuristic approaches, see e.g. [12], [13]. In [14] authors propose the reduction of the number of support vectors exploiting linear dependency of the already computed support vectors in the feature space. This work is extended in [15] and in [16] by considering *approximate* linear dependency instead of exact linear dependency. In [17] authors propose to heuristically extract candidates of SV before the SVM training step, exploiting the intuition that the couple of two data-points of opposite classes having minimal distance will likely become a couple of SV. In [18] the computational requirements for the classification step of an already trained SVM are reduced using ex-novo vectors, called *synthetic vectors*. In [19] the problem of discarding the SVs that least contribute to the decision surface is also discussed, pointing out how the selection process is time-consuming being generally a noticeable-size problem. Clustering techniques are also discussed in [20], [21], [22].

In [23] data reduction is obtained by means of heuristically constructed hyperspheres and hypercones. If the new data belong to these regions, they are discarded, otherwise they are kept and the SVM retrained. In [24] the same authors extend their previous work considering truncated hypercones, in place of hyperspheres, to select the support vector candidates. In [25] authors consider the separable case, and propose an heuristic based on differences and ratios of

The research leading to these results has received funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement n°257462 HYCON2 Network of excellence and n°223866 FeedNetBack, by Progetto di Ateneo CPDA090135/09 funded by the University of Padova, and by the Italian PRIN Project "New Methods and Algorithms for Identification and Adaptive Control of Technological Systems".

Mahalanobis distances between the data to establish if an element in the training set is potentially near the decision boundary or not.

In this scenario, the contributions of the present paper are the following ones. In the first part of the paper the case of linearly separable data set is discussed. Differently from [25] and [26], the works that are closest to this work, we provide the exact conditions that establish if an element in the training set is a *potential support vector* (PSV), i.e. if it can become a SV in the future. We also develop a simple and efficient algorithm that allows each sensor to retain only those data that are useful for the collective reconstruction of the decision boundary. The analysis also clarifies why data selection based only on the concept of margin, i.e. on the distance of the data from the optimal separatory hyperplane, may be misleading. In the second part of the paper, the non-separable case is treated. A sufficient condition that establishes when one datum cannot become a SV if only one element is added to the training set is provided. Then, a data selection algorithm, relying upon the analysis developed in the separable case, is worked out. Numerical experiments show that the proposed procedure allows the agents to exchange a small amount of the collected data to obtain a highly predictive decision boundary.

The paper is organized as follows. Section II reports some of the notation used in the paper together with some convex analysis notions. Section III provides a review of SVM for classification. The concept of PSV is also mathematically formalized. In Section IV such concept is analyzed in the case of separable data sets and an exact data selection algorithm useful for distributed classification is worked out. In Section V the non-separable case is addressed, developing an heuristic algorithm inspired by the finding in the separable data framework. In Section VI some numerical experiments are reported. Due to space limitation we defer to future works an extensive comparison with other heuristic techniques available in literature. Conclusions are finally reported. The proofs of the propositions contained in the paper can be found in [27].

II. Notation and review of some convex analysis concepts

The following notation will be adopted in what follows:

- i indexes the elements in the training set;
- d is the dimension of the inputs domain;
- $x_i \in \mathbb{R}^d$ is a generic input location;
- $y_i \in \{+1, -1\}$ is the generic feature;
- n is the total number of input locations;
- (w, b), with $w \in \mathbb{R}^d, b \in \mathbb{R}$ is a generic hyperplane;
- $\mathcal{D} := \{(x_i, y_i)\}_{i=1,\dots,n}$ is the training set. We will also use the symbols "$+$" and "$-$" to indicate set-theoretic additions and subtractions;
- $\mathcal{X} := \{x_i\}_{i=1,\dots,n}$ is the set of input locations;
- the sets of input locations corresponding to positive and negative features are respectively denoted by

$$\mathcal{X}^+ := \{x_i \in \mathcal{X} \text{ such that } y_i = +1\} \qquad (1)$$

$$\text{and} \quad \mathcal{X}^- := \{x_i \in \mathcal{X} \text{ such that } y_i = -1\}\,; \qquad (2)$$

- $r := |\mathcal{X}^+|$ (implies that $|\mathcal{X}^-| = n - r$). Without loss of generality, we also assume that

$$\begin{cases} y_i = +1 & \text{if } i = 1, \dots, r \\ y_i = -1 & \text{if } i = r+1, \dots, n\,; \end{cases} \qquad (3)$$

- m is a generic hyperplane margin;
- operator $\neg$, defined by $\neg(+1) = -1$, $\neg(-1) = +1$;
- $\mathcal{B}(A)$ is the boundary of the (topological) set A;
- $\mathrm{int}(A)$ is the interior of the (topological) set A.

The following definition of separability is adopted in the paper.

Definition 1. The data set $\mathcal{D}$ is said to be *separable* if there exists $w \neq 0$ such that $w^T x_i \geq 0$ for $i = 1, \dots, r$ and $w^T x_i \leq 0$ for $i = r+1, \dots, n$.

We recall also the following basic definitions and facts on geometry and convex analysis:

- a *cone* $K \subseteq \mathbb{R}^d$ is a set such that if $x \in K$ and $\lambda \geq 0$ then $\lambda x \in K$;
- the *convex hull* of the set $\{x_1, \dots, x_n\}$ is defined as

$$\mathcal{H}_{\text{cvx}}(x_1, \dots, x_n) := \sum_{i=1}^{n} \lambda_i x_i \qquad (4)$$

  with the additional constraints $\lambda_i \geq 0$ for $i = 1, \dots, n$ and $\sum_{i=1}^{n} \lambda_i = 1$;
- a *convex cone* $K \subseteq \mathbb{R}^d$ is a set such that if $x_1, x_2 \in K$ and $\lambda_1, \lambda_2 \geq 0$ then $\lambda_1 x_1 + \lambda_2 x_2 \in K$;
- the *conical hull* of the set $\{x_1, \dots, x_n\}$ is defined as

$$\mathcal{H}_{\text{cnc}}(x_1, \dots, x_n) := \sum_{i=1}^{n} \lambda_i x_i \qquad (5)$$

  with the additional constraint $\lambda_i \geq 0$ for $i = 1, \dots, n$. Notice that every conical hull is a convex cone;

III. Support vector machines for classification and the k-step PSV concept

Given a separable training set $\mathcal{D}$, the support vector classifier determines the optimal hyperplane by solving the following convex optimization problem

$$\begin{aligned} &\min_{w,b} \|w\| \qquad (6) \\ &\text{s.t.} \quad y_i(w^T x_i + b) \geq 1, \quad i = 1, \dots, n. \end{aligned}$$

The margin m is $1/\|w\|$ so that the optimal hyperplane provides the largest margin. The elements of the training set whose distance is exactly m are the SVs.

Given a non-separable training set $\mathcal{D}$, the usual formulation of the support vector classifier instead involves a convex optimization problem that allows some elements to be misclassified. It is given by

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \qquad (7) \\ &\text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i \end{aligned}$$

where the positive scalar C is the so called regularization parameter. An alternative formulation where also the bias term b is penalized is given by

$$\min_{w} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (8)$$
$$\text{s.t.} \quad y_i(w^T\bar{x}_i) \geq 1-\xi_i, \quad \xi_i \geq 0, \quad \forall i$$

where $\bar{x}_i = [x_i^T \quad 1]^T$. Notice that, after the solution is computed, the elements in the training set belong to three different groups. The first one contains the data that are correctly classified and outside their margins: these data are not SVs. The second group contains SVs given by the elements sitting on their margins: these data are also called *marginal* SVs. The third group contains the SVs associated to the possibly misclassified observations inside the margins.

The following new definitions are fundamental for our purposes. It is important to stress that they depend on which situation is studied. To be more specific, in the separable case x_i is a SV according to the properties of the problem (6), while in the non separable case x_i is a SV according to formulations (7) or (8). In addition, in the separable context it is assumed that all the future data added to $\mathcal{D}$ lead to a new training set which is still separable.

Definition 2. We say that x_i is a k-step PSV for the dataset $\mathcal{D}$, if there exists a data set $\mathcal{D}^*$ containing k elements s.t. x_i is a SV for the augmented dataset $\mathcal{D}+\mathcal{D}^*$. In addition, $\text{PSV}_k(\mathcal{D})$ indicates the set of all the k-step PSV contained in $\mathcal{D}$.

Definition 3. We say that x_i is a k-step discardable input location for the dataset $\mathcal{D}$ if it is not a k-step PSV for $\mathcal{D}$. In addition, $\text{Disc}_k(\mathcal{D})$ indicates the set of all the k-step discardable elements contained in $\mathcal{D}$.

Hence, notice that $x_i \in \text{PSV}_k(\mathcal{D}) \Leftrightarrow x_i \notin \text{Disc}_k(\mathcal{D})$. In what follows, when $k=1$, we write $\text{PSV}(\mathcal{D})$ and $\text{Disc}(\mathcal{D})$ in place of $\text{PSV}_1(\mathcal{D})$ and $\text{Disc}_1(\mathcal{D})$, respectively.

IV. Potential Support Vectors: The Separable Case

A. Necessary and sufficient conditions for being a PSV

We start assuming that $\mathcal{D}$ is linearly separable according to Definition 1. Given a dataset $\mathcal{D}$ and one of its input locations x_i, we define Δ_j to be

$$\begin{cases} \Delta_j := x_i - x_j & \text{if } j=1,\ldots,r, \quad i=1,\ldots,r \\ \Delta_j := x_j - x_i & \text{if } j=r+1,\ldots,n, \quad i=1,\ldots,r \\ \Delta_j := x_j - x_i & \text{if } j=1,\ldots,r, \quad i=r+1,\ldots,n \\ \Delta_j := x_i - x_j & \text{if } j=r+1,\ldots,n, \quad i=r+1,\ldots,n \end{cases} \qquad (9)$$

Notice that the various Δ_j depend both on x_i and the whole dataset $\mathcal{D}$, but we omit this dependence to simplify the notation. Without loss of generality, we often assume $i=1$ (thus $x_i = x_1$ and $y_i = y_1 = +1$).

The following proposition provides a full characterization of the concept of k-step PSV under the separability assumption.

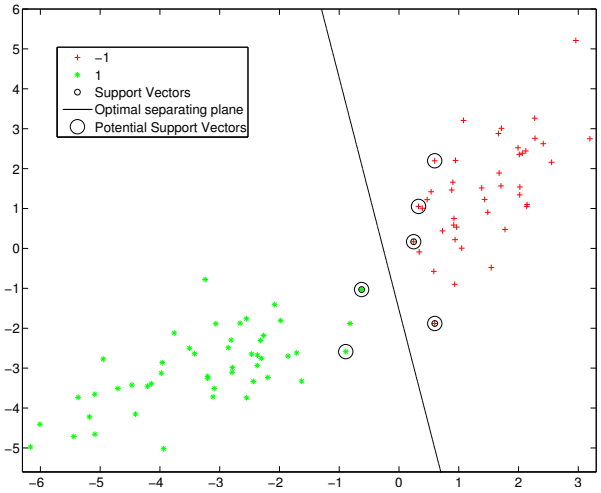

Fig. 1. Data selection in the separable case. The optimal separating plane is obtained solving problem (6), finding 3 SVs. The use of Algorithm 1 reveals that only other 3 data points must be retained to summarize all the training set also when other data arrive.

Proposition 4. Let $\mathcal{D}$ be separable. Then, the following assertions are equivalent:

1) $x_1 \in \text{PSV}(\mathcal{D})$;
2) there exists an integer k s.t. $x_1 \in \text{PSV}_k(\mathcal{D})$;
3) there exists $w \in \mathbb{R}^d$, $w \neq 0$ s.t. $\begin{cases} w^T\Delta_2 \leq 0 \\ \vdots \\ w^T\Delta_n \leq 0 \end{cases}$;
4) $\mathcal{H}_{\text{cnc}}(\Delta_2,\ldots,\Delta_n) \neq \mathbb{R}^d$.

Thus, in the separable case the concepts of one-step and k-step PSV are equivalent. In other words, if it is not possible to make x_i a SV enriching the training set with a single element, x_i will never become a SV even adding an arbitrary number of data. In addition, establishing if x_i is a PSV involves the linear feasibility problem 3).

B. Algorithm for data selection

The following sufficient discardability condition is useful to obtain an efficient data selection algorithm. It formalizes the intuitive fact that data not sitting on the boundary of the convex hulls of the two classes can never become a PSV.

Proposition 5. If $x_1 \in \text{int}(\mathcal{H}_{\text{cvx}}(x_1,\ldots,x_r))$ then $x_1 \in \text{Disc}(\mathcal{D})$.

We are now in a position to report the following numerical procedure 1, allowing every sensor to retain only those data that may influence the collective reconstruction of the boundary.

In Fig. 1 the algorithm is applied to a bi-dimensional data set of 100 elements. The set $\text{PSV}(\mathcal{D})$ obtained with Algorithm 1 contains only 6 points. All the other data can be discarded without any loss of information also when new data arrive. Notice also that there are many data points which are not PSV that are closer to the optimal hyperplane than the true PSVs, hence dataset reduction based only on the distance of the data from the optimal separatory hyperplane may be misleading.

Algorithm 1 Computation of PSV $(\mathcal{D})$ (separable case)

1: input: the original training set $\mathcal{D}$;

2: compute $\mathcal{H}_{\text{cvx}}(x_1, \ldots, x_r)$ and $\mathcal{H}_{\text{cvx}}(x_{r+1}, \ldots, x_n)$;

3: discard from $\mathcal{D}$ all the data in the interior of the two convex hulls. Hence, obtain the reduced data set $\mathcal{D}^{(1)}$ containing $n^{(1)}$ ordered elements such that $y_i = 1$ for $i = 1, \ldots, r^{(1)}$ and $y_i = -1$ for $i = r^{(1)} + 1, \ldots, n^{(1)}$;

4: set the algorithm counter k to 1. For $k = 1$ to $n^{(1)}$, until all the elements in $\mathcal{D}^{(1)}$ have been analyzed, perform the following operations:

(a) consider an element x_i of $\mathcal{D}^{(k)}$ that has not been analyzed in the previous iterations;

(b) build the various Δ_j according to (9) using the elements contained in $\mathcal{D}^{(k)}$ with $r = r^{(k)}$ and $n = n^{(k)}$. Define Δ as the matrix with rows given by $[\Delta_2^T, \Delta_3^T, \ldots, \Delta_{n^{(k)}}^T]$;

(c) if the null space of Δ contains only the origin, apply the simplex procedure to the problem

$$\max_{s} s_1 + s_2 + \ldots + s_{n^{(k)}-1} \quad \text{s.t. } \Delta w + s \leq 0, \;\; s_i \geq 0, \quad i = 1, \ldots, n^{(k)} - 1 \tag{10}$$

just checking if the algorithm can move from the origin ($s = 0, w = 0$) used as starting point;

(d) if the null space of Δ contains only the origin and the simplex does not move from the origin, define $\mathcal{D}^{(k+1)}$ as $\mathcal{D}^{(k)} - (x_i, y_i)$, updating the values for $r^{(k+1)}, n^{(k+1)}$. Otherwise, set $\mathcal{D}^{(k+1)}$ to $\mathcal{D}^{(k)}$;

(e) if $k = n^{(1)}$, set PSV$(\mathcal{D})$ to $\mathcal{D}^{(k+1)}$.

C. Computational considerations

The first and second step of Algorithm 1 require the computation of convex hulls and may permit to find many data points which are not PSVs. Using the "gift wrapping" algorithm, the number of required operations is $O\left(n^{\lfloor d/2 \rfloor} + 1\right)$, where $\lfloor \cdot \rfloor$ is the floor function, which reduces to $O(n \log(n))$ when $d = 2$ [28]. When d is large or the points are mapped into a high-dimensional feature space by the use of basis expansions, the computational cost is prohibitive so that these two steps should be skipped. However, as clear in the sequel, in the distributed setting the work made by the agents for data selection may require the use of a very low value for d, e.g. equal to 2 or 3. A more sophisticated input space, possibly also infinite-dimensional, might be needed only for the final reconstruction of the decision boundary. This point will be further developed in the numerical experiments section.

Finally, for what regards the second part of Algorithm 1, problem (10) must be built for each element contained in $\mathcal{D}^{(1)}$. However, the procedure is more efficient because at every iteration very few steps of the simplex algorithm must be performed, just to verify if it is possible to leave the origin. Notice also that the dimensions of the linear programming problems to be faced may decrease over time as the iteration counter increases.

V. Potential support vectors: the non separable case

A. Sufficient condition for one-step discardability

Now, let $\mathcal{D}$ be a possibly non-separable set. Recall that the optimal $w \in \mathbb{R}^{d+1}$ which solves problem (8) admits the representation

$$\hat{w} = \sum_{i=1}^{n} a_i \bar{x}_i \; . \tag{11}$$

Then, the following sufficient condition for one-step discardability holds.

Proposition 6. Let $\mathcal{D}$ be non separable and consider problem (8) with solution given by (11). Define

$$\eta_i := y_i - \sum_{j=1, j \neq i}^{n} a_j \bar{x}_j, \qquad i = 1, \ldots, n \; . \tag{12}$$

Then, if $\eta_i < -1 - nC\bar{x}_i^T \bar{x}_i$ it holds that $\bar{x}_i \in \text{Disc}_1(\mathcal{D})$.

B. Algorithm for data selection

Although interesting under a theoretical point of view, the result reported in the previous subsection has some drawbacks: even if restricted to the case $k = 1$, the discardability condition depends only on the margin and tends to be conservative, retaining many data points. For this reason, we also propose an algorithm for data selection in the non separable case based on the procedure obtained for the separable scenario. The rationale is the following one. After one sensor solves problem (7) using its own $\mathcal{D}$, it obtains the SVs providing information on the decision boundary. Removing from $\mathcal{D}$ the SVs that are not marginal, it obtains a separable data set denoted by $\mathcal{D}_1$. Algorithm 1 can then be used to obtain PSV$(\mathcal{D}_1)$ that represent other points informative on the boundary. The data points that summarize $\mathcal{D}$ are finally given by the union of PSV$(\mathcal{D}_1)$ with the SV not contained in $\mathcal{D}_1$. The algorithm is reported below.

Algorithm 2 Computation of PSV $(\mathcal{D})$ (non-separable case)

1: inputs: the original training set $\mathcal{D}$ and the regularization parameter C;

2: solve problem (7) and denote with $\mathcal{D}_2$ the set of SVs which do not sit on the margin;

3: define the separable set $\mathcal{D}_1 := \mathcal{D} - \mathcal{D}_2$;

4: compute PSV$(\mathcal{D}_1)$ using Algorithm 1 with input $\mathcal{D}_1$;

5: return PSV$(\mathcal{D}_1) + \mathcal{D}_2$ as the data points summarizing the training set $\mathcal{D}$.

C. Further data reduction using a filling measure

In some cases, one would like each sensor to send a number of input locations belonging to a certain class not exceeding a certain threshold K. On the other hand, the number of examples labeled with 1 or -1 in the set PSV$(\mathcal{D}_1) + \mathcal{D}_2$ returned by Algorithm 2 may exceed K. In such situations, Algorithm 3 below is especially useful since

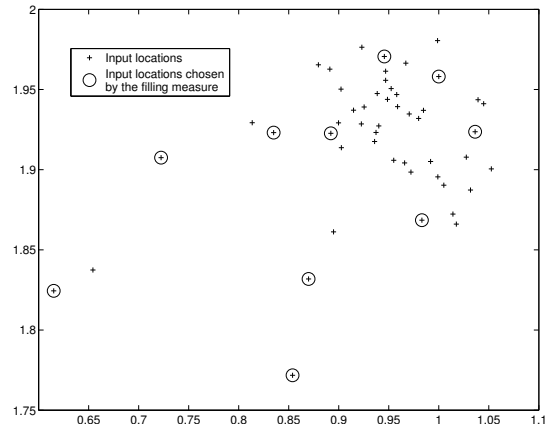

Fig. 2. Application of Algorithm 3 to select 10 elements from a set containing 50 input locations.

it returns for each class the K input locations that maximize a filling measure, see also [29, Sec. 7.3]. In particular, it selects those K data that cover as much as possible the region visited by the sensor according to a distance function $v : \mathbb{R}^d \mapsto \mathbb{R}$.

Algorithm 3 Computation of PSV ($\mathcal{D}$) (filling measure case)

1: inputs: a set I of input locations and a distance function $v : \mathbb{R}^d \mapsto \mathbb{R}$;

2: Set the algorithm counter k to 1 and $I^{(k)} = I$. Until the cardinality of $I^{(k)}$ exceeds K, perform the following operations

(a) discard from $I^{(k)}$ the element $x_{\tilde{p}}$ where

$$\tilde{p} = \arg\max_{p} \min_{i \neq p} \min_{j \neq i, j \neq p} v(x_i, x_j)$$

(b) define $I^{(k+1)} = I^{(k)} - x_{\tilde{p}}$

3: return $I^{(k+1)}$ as the set of input locations that best cover the visited space.

In the following the distance d will be the Euclidean distance. An example of application of Algorithm 3 with the cardinality of I equal to 50 and $K = 10$ is given in Fig. 2.

Remark 7. Notice that, if the data points are mapped into an enlarged feature space by a kernel $K : \mathbb{R}^d \mapsto \mathbb{R}$, associated with a reproducing kernel Hilbert space H_K with norm $\|\cdot\|_K$, it is also possible to build a distance v_K defined by

$$\begin{aligned} v_K(x_i, x_j) &= \|K(x_i, \cdot) - K(x_j, \cdot)\|_K \\ &= K(x_i, x_i) - 2K(x_i, x_j) + K(x_j, x_j)\ . \end{aligned}$$

For instance, using the Gaussian kernel

$$K(x_i, x_j) = e^{-(x_i - x_j)^T \Sigma (x_i - x_j)},$$

one obtains

$$\begin{aligned} \tilde{p} &= \arg\max_{p} \min_{i \neq p} \min_{j \neq i, j \neq p} (2 - 2e^{-(x_i - x_j)^T \Sigma (x_i - x_j)}) \\ &= \arg\max_{p} \min_{i \neq p} \min_{j \neq i, j \neq p} (x_i - x_j)^T \Sigma (x_i - x_j), \end{aligned}$$

i.e. the weighted Euclidean distance.

VI. Numerical experiments

We consider a scenario where 100 sensors are distributed on a plane. Their initial positions are independent realizations from a uniform distribution on $[-0.5, 2] \times [0, 2]$. Each sensor then moves according to a random walk with standard deviation of the increments equal to 0.03 along both the axes. After every step, the sensor acquires a measurement. More precisely, the total number of data collected by an agent is 50 and data generation is as follows. Consider the curve $\left(\frac{(t-5)^2}{25}, e^{-0.5t} + e^{-0.1t}\right)$ with $t \in \mathbb{R}$, displayed in Fig. 3 (dashed line). Let x and r denote the location of the sensor and its distance from the curve, respectively. Then, when the point x is on the left (right) of the curve, the measurement taken by the agent is $+1$ (-1) with probability $1 - 0.5e^{-8r}$.

The left panel of Fig. 3 displays the 5000 data collected by the network. Each agent observes only a small piece of the curve that can be well approximated by a low-dimensional model, e.g. a linear or second-order polynomial kernel. Hence, for data selection purposes, the feature space does not need to be much enlarged. In particular, in our case data selection is performed by Algorithm 2 using a linear kernel and the dimension d of the input space remains equal to 2. The value of C in (7) is set to 50. In addition, Algorithm 3 is used to force every sensor to retain no more than 5 data points per class. In the case the agent is not able to derive the linear decision boundary, or all the collected data belong to only one class, only 5 data belonging to one single class are sent to the other agents.

The right panel of Fig. 3 displays the data exchanged by the sensors. They correspond to almost 10% of all the data and appear a good summary of the entire training set. In the same panel the solid line is the decision boundary obtained by a SV classifier, equipped with a Gaussian kernel still adopting $C = 50$ and fed with the reduced data set. The estimate appears reasonably close to the optimal solution represented by the dashed line.

VII. Conclusions

We have discussed the problem data selection in support vector classification problems, which is of paramount importance in distributed classification frameworks. We have introduced the new concept of k-step potential support vector (PSV), completely characterizing it in the case of separable training sets. Some insights about PSV in the non separable case have been also obtained. A new general data selection algorithm has been worked out, proving its effectiveness through numerical simulations. In the near future, we aim to show its effectiveness by comparing it with other techniques proposed in literature. Moreover we aim to extend the theoretical results relative to the non separable datasets case. In this context, deriving conditions to establish if a datum is a k-step PSV appears an interesting and challenging problem.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102 – 114, August 2002.

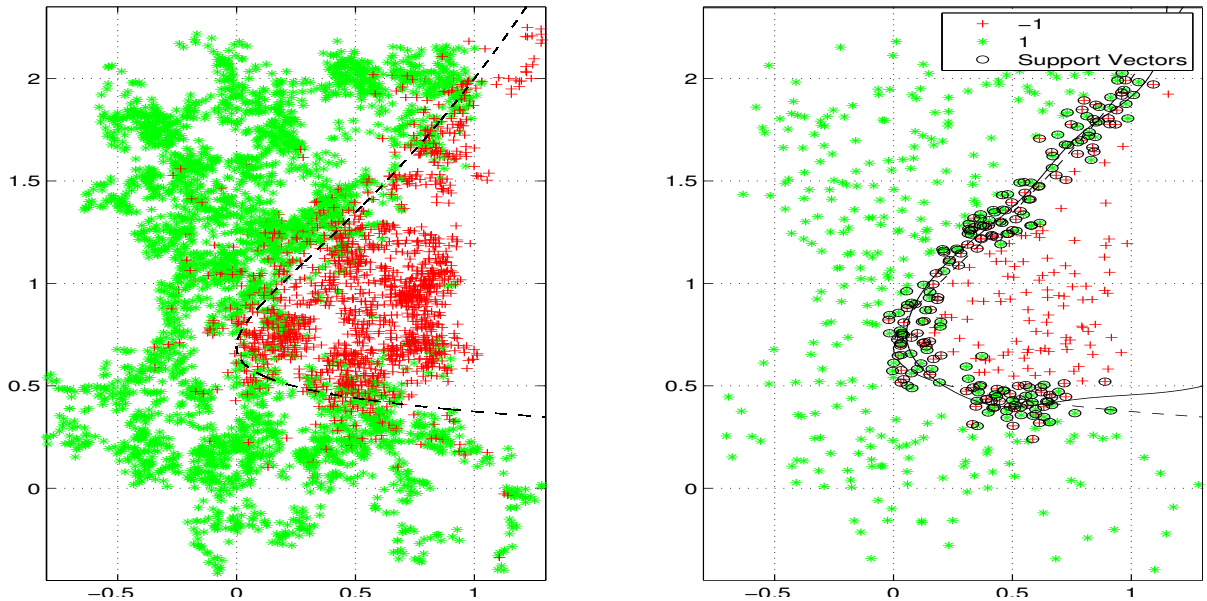

Fig. 3. Optimal nonlinear decision boundary (dashed line). *Left*: the training set given by the union of all the data collected by the sensors. *Right*: the training set obtained by the union of the data retained by the sensors using Algorithm 2. Algorithm 3 is also used to force every sensor to retain no more than 5 data points per class. The solid line is the decision boundary obtained using SV classification with a Gaussian kernel.

[2] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *IEEE Circuits and Systems Magazine*, vol. 5, no. 3, pp. 19 – 31, 2005.

[3] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56 – 69, July 2006.

[4] J.-J. Xiao, A. Ribeiro, Z.-Q. Luo, and G. Giannakis, "Distributed compression-estimation using wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 27 – 41, July 2006.

[5] V. N. Vapnik, *Statistical Learning Theory*. New York: Springer-Verlag, 1998.

[6] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, pp. 1663 – 1707, 2010.

[7] R. U. Pedersen, "Using support vector machines for distributed machine learning," Ph.D. dissertation, University of Copenhagen, August 2004.

[8] T. Alpcan and C. Bauckhage, "A distributed machine learning framework," in *Proceedings of the 48th IEEE Conference on Decision and Control*, December 2009, pp. 2546 – 2551.

[9] F. Orabona, J. Keshet, and B. Caputo, "Bounded kernel-based online learning," *Journal of Machine Learning Research*, vol. 10, pp. 2643–2666, 2009.

[10] V. N. Vapnik, *Estimation of Dependence Based on Empirical Data*. Berlin: Springer-Verlag, 1982.

[11] G. Cauwenberghs and T. Poggio, *Incremental and Decremental Support Vector Machine Learning*. MIT Press, 2000, vol. 13, pp. 409 – 415.

[12] Y. Li, W. Zhang, and C. Lin, "Simplify support vector machines by iterative learning," *Neural Information Processing - Letters and Reviews*, vol. 10, no. 1, pp. 11 – 17, January 2006.

[13] C. Li, K. Liu, and H. Wang, "The incremental learning algorithm with support vector machine based on hyperplane-distance," *Applied Intelligence*, vol. 34, no. 1, pp. 1 – 9, 2009.

[14] T. Downs, K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," *Journal of Machine Learning Research*, vol. 2, pp. 293 – 297, December 2001.

[15] Y. Engel, S. Mannor, and R. Meir, "Sparse online greedy support vector regression," in *Machine Learning: ECML 2002*. Springer Berlin / Heidelberg, 2002, vol. 2430, pp. 1–3.

[16] T. Kobayashi and N. Otsu, "Efficient reduction of support vectors in kernel-based methods," in *Proceedings of the 16th IEEE International Conference on Image Processing*, November 2009, pp. 2077 – 2080.

[17] Y.-G. Liu, Q. Chen, and R.-Z. Yu, "Extract candidates of support vector from training set," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, vol. 5, November 2003, pp. 3199 – 3202.

[18] C. J. C. Burges, "Simplified support vector decision rules," in *Proceedings of the 13th International Conference on Machine Learning*, 1996, pp. 71 – 77.

[19] C. J. C. Burges and B. Schölkopf, "Improving the accuracy and speed of support vector learning machines," in *Proceedings of the 9th NIPS Conference*, 1997, pp. 375 – 381.

[20] R. Koggalage and S. Halgamuge, "Reducing the number of training samples for fast support vector machine classification," *Neural Information Processing - Letters and Reviews*, vol. 2, no. 3, pp. 57 – 65, March 2004.

[21] H. Shin and S. Cho, "Fast pattern selection for support vector classifiers," in *Advances in Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, vol. 2637.

[22] X. Liang, "An effective method of pruning support vector machine classifiers," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 26 – 38, January 2010.

[23] S. Katagiri and S. Abe, "Incremental training of support vector machines using hyperspheres," *Pattern Recognition Letters*, vol. 27, no. 13, pp. 1495 – 1507, October 2006.

[24] ——, "Incremental training of support vector machines using truncated hypercones," in *Artificial Neural Networks in Pattern Recognition*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2006, vol. 4087, pp. 153 – 164.

[25] S. Abe and T. Inoue, "Fast training of support vector machines by extracting boundary data," in *Proceedings of the International Conference on Artificial Neural Networks*, vol. 2130, 2001, pp. 308 – 313.

[26] A. Bordes and L. Bottou, "The huller: A simple and efficient online svm," in *Machine Learning: ECML 2005*. Springer Berlin / Heidelberg, 2005.

[27] S. Del Favero, D. Varagnolo, F. Dinuzzo, L. Schenato, and G. Pillonetto, "On the discardability of data in support vector classification problems," in *http://paduaresearch.cab.unipd.it*, 2011.

[28] B. C. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469 – 483, December 1996.

[29] R. Schaback and H. Wendland, "Kernel techniques: From machine learning to meshless methods," *Acta Numerica*, vol. 15, pp. 543 – 639, 2006.