Expert Systems with Applications 42 (2015) 4284–4298

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Self-adaptive Support Vector Machine: A multi-agent optimization perspective

Nicolas Couellan [a,*], Sophie Jan [a], Tom Jorquera [b], Jean-Pierre Georgé [b]

[a] Institut de Mathématiques de Toulouse, Université de Toulouse, UPS IMT, F-31062 Toulouse Cedex 9, France
[b] Institut de Recherche en Informatique de Toulouse, Université de Toulouse, UPS IRIT, F-31062 Toulouse Cedex 9, France

ARTICLE INFO

Article history:
Available online 28 January 2015

Keywords:
Support Vector Machine
Classification
Model selection
Multi-agent systems
Collaborative process
Complex systems optimization

ABSTRACT

Support Vector Machines (SVM) have been in the forefront of machine learning research for many years now. They have very nice theoretical properties and have proven to be efficient in many real life applications but the design of SVM training algorithms often gives rise to challenging optimization issues. We propose here to review the basics of Support Vector Machine learning from a multi-agent optimization perspective. Multi-agents systems break down complex optimization problems into elementary "oracle" tasks and perform a collaborative solving process resulting in a self-organized solution of the complex problems. We show how the SVM training problem can also be "tackled" from this point of view and provide several perspectives for binary classification, hyperparameters selection, multiclass learning as well as unsupervised learning. This conceptual work is illustrated through simple examples in order to convey the ideas and understand the behavior of agent cooperation. The proposed models provide simple formulations of complex learning tasks that are sometimes very difficult to solve with classical optimization strategies. The ideas that are discussed open up perspectives for the design of new distributed cooperative learning systems.

1. Introduction

SVMs are known as powerful mathematical tools for classification as well as regression tasks (Cristianini & Shawe-Taylor, 2001; Scholkopf & Smola, 2001). They have proven good capabilities for the classification of complex and large datasets. Many successful implementations have been developed for various types of applications: medical diagnosis (Fung & Mangasarian, 2006; Lee, Mangasarian, & Wolberg, 1999), manufacturing (Balakrishna, Raman, Santosa, & Trafalis, 2008; Gilbert, Raman, Trafalis, Obeidat, & Aguirre-Cruz, 2009), meteorology (Trafalis, Adrianto, & Richman, 2007), hand digits recognition (Decoste & Schölkopf, 2002), fraud detection (Chan & Stolfo, 1998), and many others.

The underlying concepts are based on empirical risk theory (Vapnik, 1998) and the available algorithms make use of convex optimization techniques (Boyd & Vandenberghe, 2004). A strong focus is now put on the ever increasing size of datasets and new algorithms based on first order stochastic optimization have now emerged (Bottou, 1997; Bousquet & Bottou, 2007).

Training speed has therefore been greatly reduced by combining "cheap" first order optimization techniques with stochastic frameworks that process only samples of data at a time. The increasing dimensions of datasets and the emergence of online applications where data is only available dynamically bring new and great challenges to the machine learning and optimization communities. Additionally, most learning algorithms require the selection of hyperparameters. These parameters are usually problem dependent and control for example the trade-off between training accuracy and generalization performance (ability to generalize prediction to unseen data) or some mapping function (or usually its corresponding kernel function) that will transform the problem into a linear problem. Selecting the optimal parameters is known as *model selection*. It is often critical in real life applications and no matter how effective and fast the training procedure is, if the hyperparameters are not tuned in a proper way, the resulting model will not generalize well to new data (Hastie, Rosset, Tibshirani, & Zhu, 2003/04). Model selection is usually done through the so-called *k-fold Cross Validation* (CV) where the data is split in k subsets and k training procedures are performed with $k-1$ subsets as training data and the remaining subset as validation data (swapping the training and validation subsets for each training procedure) (Hastie, Tibshirani, & Friedman, 2009). The k-fold cross validation is combined with a

* Corresponding author.
E-mail addresses: nicolas.couellan@math.univ-toulouse.fr (N. Couellan), sophie.jan@math.univ-toulouse.fr (S. Jan), tom.jorquera@irit.fr (T. Jorquera), jean-pierre.george@irit.fr (J.-P. Georgé).

http://dx.doi.org/10.1016/j.eswa.2015.01.028
0957-4174/

grid search method to estimate the optimal hyperparameters. The CV is statistically valid but has two major drawbacks: (1) if datasets are large, it can become extremely time expensive and in practice unrealistic, (2) the grid search will only go over a very limited and discrete set of values of the hyperparameters while the optimal parameters could lie in between grid points.

To overcome these drawbacks, some techniques making use of bi-level SVM formulations have been investigated. The idea is to perform descent techniques with respect to the hyperparameters while trying to find the optimal separating hyperplane (Couellan & Wang, 2014; Du, Peng, & Terlaky, 2009; Kunapuli, Bennett, Hu, & Pang, 2008). These techniques address successfully the problem of auto-selecting the trade-off (training accuracy/generalization performance) parameter but do not extend to the problem of tuning the kernel function parameter that arises in the case of nonlinear learning systems.

Alternatively, to design learning systems that are able to adjust dynamically to online data as well as being able to self-adjust the problem hyperparameters, we investigate a novel approach. We propose to look at the SVM model and its parameter selection as a whole and perform optimization on a system involving various natures of variables. These variables are interconnected in a calculus network so that it can be considered as a complex system. To solve these types of naturally complex tasks, one way is to make use of an *Adaptive Multi-Agent System (AMAS)* where autonomous agents are each given part of the optimization problem and cooperation between them takes place to solve the overall problem.

A multi-agent system (MAS) (Weiss, 1999) is a system composed of several autonomous software entities (the agents), interacting among each others (usually by sending information and request messages) and with their environment (by observing and modifying it). The *autonomy* of an agent is the fundamental characteristic that differentiates it from, for example, the computer science concept of object. While an object is a passive entity encapsulating some data and functions, waiting to be solicited, an agent is capable of reacting to its environment and displaying pro-activity (activity originating from its own decision). From this comparison it should be clear that the concept of agent is, like the concept of object, the building brick of a paradigm which can be used to model a complex reality. And indeed, agents have been used in a great variety of fields, a fact which can contribute to explain the difficulty to produce a unified definition of the concept.

While it is not true for all MAS, some interesting properties can be achieved when taking advantage of the autonomy of the agents. This autonomy, coupled with an adequate behavior of the agents, can lead to systems able to adjust, organize, react to changes, etc. without the need for an external authority to guide them. These properties are gathered under the term self-* capabilities (Di Marzo Serugendo et al., 2011) (self-*tuning*, self-*organizing*, self-*healing*, self-*evolving*...). Not all MAS necessarily present all of these self-* capabilities but, as a result of building a system from autonomous and locally situated agents, many MAS will exhibit them to some degree. Consequently, MAS are often relevant for dynamically taking into account changes in their environment. For example, a MAS in charge of regulating the traffic of packets in a computer network could be able to react efficiently to the disappearance of some of the relay nodes.

MAS have been applied to a great variety of fields: social simulation, biological modeling, systems control, robotics, etc. and agent-oriented modeling can be seen as a programming paradigm in general, facilitating the representation of a problem.

A particular approach to MAS relying strongly on self-* properties is the *AMAS* technology and underlying theory (Georgé, Edmonds, & Glize, 2004). A designer following this approach focuses on giving the agent a local view of its environment, means to detect problematic situations and guidelines to act in a *cooperative* way, meaning that the agents will try to achieve their goals while respecting and helping the other agents around them as best as they can. The fact that the agents do not follow a global directive towards the solving of the problem but collectively build this solving, produces an *emergent problem solving process* that explores the search space of the problem in original ways.

Modeling SVMs as AMAS has several advantages over more classical mathematical strategies. It avoids running unnecessary training procedures for non-optimal regions of the hyperparameters space. The selected optimal hyperparameters values are more accurate as they are not constrained on a grid but can take freely any value of the space. Finally, the use of AMAS opens the door to parallelization, decomposition and distributed computation. The work presented here can be seen as preliminary work to illustrate the possible perspectives that further research along this area could give. Current research in SVM has generated a great deal of work on model selection for binary classification and single kernel techniques with possible but sometimes expensive (complexity wise) extensions to multi-class and multiple kernel variants. Clearly the use of AMAS gives more flexible and more natural ways to extend models to more complicated contexts.

The article is organized as follows. Section 2 recalls the basic mathematics of classification with SVMs, Section 3 describes the principles of AMAS. In Section 4 we propose models to perform training tasks with AMAS and in Section 5 we incorporate the model selection concepts into our models. Finally, in Section 6 we provide several numerical examples on simple illustrative problems. Section 7 concludes the paper.

2. SVM classification

2.1. Linear classification

Consider a set of training vectors $\{x_i \in \mathbb{R}^n,\ i = 1,\dots,L\}$ and its corresponding set of labels $\{y_i \in \{-1,1\},\ i = 1,\dots,L\}$, where L is the number of training points and n is the number of attributes of each training point.

The soft margin SVM training problem can be expressed as follows (see for example Cristianini & Shawe-Taylor (2001), Scholkopf & Smola (2001) for further details on the construction of the problem):

$$\begin{aligned} &\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}\xi_i \\ &\text{subject to} \quad y_i(w^\top x_i + b) + \xi_i \geqslant 1, \quad i = 1,\dots,L, \\ &\xi_i \geqslant 0, \quad i = 1,\dots,L, \end{aligned} \tag{1}$$

where ξ_i is a slack variable associated to a penalty term in the objective with magnitude controlled by C, a problem specific parameter. The vector w is the normal vector to the separating hyperplane ($w^\top x + b = 0$) and b is its relative position to the origin.

Problem (1) maximizes the margin $\frac{2}{\|w\|}$ between the two separating hyperplanes $w^\top x + b = 1$ and $w^\top x + b = -1$. The use of slack variables ξ_i penalizes data points that would fall on the wrong side of the hyperplanes.

In the constraints, observe that $\xi_i \geqslant \max\{0, 1 - y_i(w^\top x_i + b)\}$, therefore at optimality we have the equality:

$$\xi_i = \max\{0, 1 - y_i(w^\top x_i + b)\}.$$

Indeed, the ith point is either correctly classified ($\xi_i = 0$) or penalized ($\xi_i = 1 - y_i(w^\top x_i + b)$). Consequently, we can reformulate Problem (1) as an unconstrained optimization problem:

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}\max\{0, 1 - y_i(w^\top x_i + b)\}.$$