Künstl Intell (2013) 27:281–285
DOI 10.1007/s13218-013-0248-1

DOCTORAL AND POSTDOCTORAL DISSERTATIONS

From Supervised to Unsupervised Support Vector Machines and Applications in Astronomy

Fabian Gieseke

Published online: 29 March 2013

Abstract Support vector machines are among the most popular techniques in machine learning. Given sufficient labeled data, they often yield excellent results. However, for a variety of real-world tasks, the acquisition of sufficient labeled data can be very time-consuming; unlabeled data, on the other hand, can often be obtained easily in huge quantities. Semi-supervised support vector machines try to take advantage of these additional unlabeled patterns and have been successfully applied in this context. However, they induce a hard combinatorial optimization problem. In this work, we present two optimization strategies that address this task and evaluate the potential of the resulting implementations on real-world data sets, including an example from the field of astronomy.

Keywords Machine learning · Support vector machines · Semi-supervised learning · Astronomy

1 Introduction

The amount of data in astronomy has increased dramatically in recent years, with data volumes in the tera- and petabyte range [4]. For such projects, the sheer data volume renders a manual analysis impossible. Machine learning techniques aim at retrieving useful information in an automatic manner and the corresponding tools have been recognized as *"increasingly essential in the era of data-intensive astronomy"* [1]. In general, expert knowledge is required to teach machines how to automatically perform a task. Such a teaching process is usually based on labeled data, and the corresponding settings are called *supervised learning* [5] scenarios. Ideally, a large amount of labeled data is given. Depending on the task at hand, however, acquiring such labeled data can be extremely time-consuming or expensive. In contrast, unlabeled data can often be obtained without much additional effort. Both so-called *semi-* and *unsupervised learning* [5] schemes aim at making use of the additional information provided by the unlabeled patterns to generate appropriate models.

In this work, we describe the extension of *support vector machines* [3, 7] to such settings, and briefly sketch two different optimization strategies to address the induced (combinatorial) optimization task. Further, possible application domains of such models are sketched including an example in astronomy.

2 Support Vector Machines

Support vector machines aim at *binary classification* settings with labeled training sets of the form $T_L = \{(\mathbf{x}'_1, y'_1), \ldots, (\mathbf{x}'_l, y'_l)\} \subset \mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \{-1, +1\}$. Briefly speaking, the goal of a support vector machine is to find a hyperplane that separates the patterns of both classes well such that the induced distance (or *margin*) between the hyperplane and the patterns is maximized. At the same time, patterns lying within the corridor are penalized, see Fig. 1(a). This goal can be formalized mathematically in terms of:

$$\begin{aligned} &\underset{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}, \boldsymbol{\xi}'\in\mathbb{R}^l}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C' \sum_{i=1}^{l} \xi'_i \\ &\text{s.t.} \quad y'_i\left(\langle \mathbf{w}, \mathbf{x}'_i \rangle + b\right) \geq 1 - \xi'_i, \quad \xi'_i \geq 0 \end{aligned} \tag{1}$$

This abstract depicts parts of the author's PhD thesis [4].

F. Gieseke (✉)
Department Informatik, Carl von Ossietzky Universität Oldenburg, 26111 Oldenburg, Germany
e-mail: f.gieseke@uni-oldenburg.de

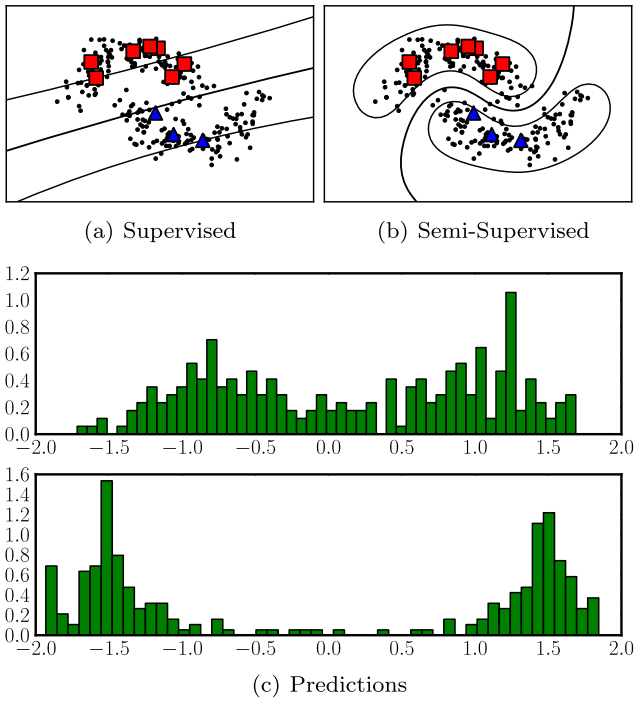

Fig. 1 The goal of a support vector machine consists in finding the hyperplane that maximizes the distance to all patterns (*red squares* and *blue triangles*), see (**a**). In real world scenarios, however, labeled data is usually scarce, while unlabeled ones can often be obtained easily. In (**b**), the output of a semi-supervised support vector machine is shown, which takes the additional information provided by the unlabeled patterns into account. The corresponding real-valued prediction values are shown in (**c**). Note that the semi-supervised model *avoids* high-density areas induced by the unlabeled patterns (i.e., the model does not go "through" the unlabeled patterns) (Color figure online)

Here, the first term of the objective corresponds to maximizing the margin whereas the second term corresponds to penalizing the patterns lying within the corridor [3, 7]. The parameter $C' > 0$ determines the trade-off between these two aims.

Support vector machines can also be seen as instance of *regularization problems* [7] having the form

$$\inf_{f\in\mathcal{H}_k, b\in\mathbb{R}} \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\big(y_i, f(\mathbf{x}_i)+b\big) + \lambda \|f\|_{\mathcal{H}_k}^2, \tag{2}$$

where $\mathcal{H}_k \subset \mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$ is a *reproducing kernel Hilbert space* [7] and where $\lambda \in \mathbb{R}^+$ is a user-defined parameter that determines the trade-off between data fit and complexity $\|f\|_{\mathcal{H}_k}^2$ of the model $f \in \mathcal{H}_k$. The hypothesis space $\mathcal{H}_k$ is induced by a *kernel function* $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which can be seen as similarity measure for the input patterns. Common choices for the loss function are the *hinge loss* $\mathcal{L}(y,t) = \max(0, 1-yt)$ and the *square loss* $\mathcal{L}(y,t) = (y-t)^2$.

3 Semi-Supervised Support Vector Machines

The semi-supervised extension takes both a labeled set $T_L = \{(\mathbf{x}'_1, y'_1), \ldots, (\mathbf{x}'_l, y'_l)\} \subset \mathbb{R}^d \times \{-1, +1\}$ and an unlabeled set $T_U = \{\mathbf{x}_1, \ldots, \mathbf{x}_u\} \subset \mathcal{X} \subset \mathbb{R}^d$ of patterns into account.[1]

In a nutshell, the idea is to consider the same objective as before, but, at the same time, to additionally enforce the decision hyperplane not to go "through" high-density areas induced by the unlabeled patterns. An illustration of this extension is given in Figs. 1(b) and (c). From an optimization point of view, semi-supervised support vector machines aim at identifying a partition of the unlabeled patterns into two classes such that a *subsequent* application of a modified support vector machine yields the best overall result. This leads to a *mixed integer programming* problem of the form

$$\begin{aligned}\underset{\mathbf{y}\in\{-1,+1\}^u, f\in\mathcal{H}_k, b\in\mathbb{R}}{\text{minimize}} \quad & \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\big(y'_i, f(\mathbf{x}'_i)+b\big) \\ & + \gamma \frac{1}{u}\sum_{i=1}^{u} \mathcal{L}\big(y_i, f(\mathbf{x}_i)+b\big) \\ & + \lambda \|f\|_{\mathcal{H}_k}^2, \end{aligned} \tag{3}$$

with $\gamma \in \mathbb{R}^+$ as additional model parameter. Hence, the main difficulty consists in finding an appropriate assignment for the vector $\mathbf{y} \in \{-1, +1\}^u$, which encodes the partition of the unlabeled patterns.[2]

[1] Due to lack of space, we focus on the semi-supervised case in this abstract.

[2] Note that an additional balancing constraint is often considered, which avoids unbalanced partitions.

Algorithm 1 Local search scheme

1: Initialize candidate solution $\mathbf{y} \subseteq \{-1, +1\}^u$.
2: **while** not converged **do**
3: $\quad \bar{\mathbf{y}} \leftarrow \mathbf{y}$ with single coordinate flipped.
4: $\quad$ **if** $F(\bar{\mathbf{y}}) < F(\mathbf{y})$ **then**
5: $\quad\quad \mathbf{y} \leftarrow \bar{\mathbf{y}}$
6: $\quad$ **end if**
7: **end while**

4 Combinatorial and Non-convex Optimization

A variety of optimization schemes has been proposed in the literature, and we will briefly sketch two of them in the following. For a detailed description as well as for literature overview, see the author's thesis [4].

4.1 Speedy Local Search

The first scheme can be seen as a variant of one of the first approaches for semi-supervised support vector machines [6]. It is based on iteratively improving an "initial guess" by flipping single coordinates of the intermediate candidate solutions, see Algorithm 1 (here, $F(\mathbf{y})$ denotes the objective induced by (3) for a fixed $\mathbf{y}$). A direct implementation of this heuristic, however, is rather slow since the recurrent computation of $F(\bar{\mathbf{y}})$ in Step 4 can be very time-consuming (a naive approach would need cubic time). However, one can speed up this process significantly by considering the square loss instead of the original hinge loss: By using standard matrix calculus, one can show that the induced objective can be written as

$$F(\bar{\mathbf{y}}) = \bar{\mathbf{y}}^{\mathrm{T}} \mathbf{\Lambda} \underbrace{\left(\mathbf{I} - \overline{\mathbf{K}}\mathbf{G} - \mathbf{G}\overline{\mathbf{K}} + \mathbf{G}\overline{\mathbf{K}}\overline{\mathbf{K}}\mathbf{G} + \lambda \mathbf{G}\overline{\mathbf{K}}\mathbf{G}\right)}_{=:\mathbf{H}} \mathbf{\Lambda}\bar{\mathbf{y}}$$

with $n = l + u$, kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$, diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{n \times n}$ with entries $[\mathbf{\Lambda}]_{i,i} = \sqrt{\frac{1}{l}}$ for $i = 1, \ldots, l$ and $[\mathbf{\Lambda}]_{i,i} = \sqrt{\frac{\gamma}{u}}$ for $i = l+1, \ldots, n$, and matrices $\overline{\mathbf{K}} := \mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda}$ and $\mathbf{G} := (\mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda} + \lambda \mathbf{I})^{-1}$. This decomposition paves the way for efficiently "updating" the function values per iteration:

Theorem 1 ([4]) *Step* 4 *of Algorithm* 1 *can be performed in* $\mathcal{O}(n)$ *time per iteration.*

Proof Sketch The main idea of the update scheme is to precompute the matrix $\mathbf{H}\mathbf{\Lambda}$ beforehand (spending cubic time once only) and to update the vector $\mathbf{H}\mathbf{\Lambda}\bar{\mathbf{y}} \in \mathbb{R}^n$ in each iteration via

$$\mathbf{H}\mathbf{\Lambda}\bar{\mathbf{y}} = \mathbf{H}\mathbf{\Lambda}\mathbf{y} - 2y_j(\mathbf{H}\mathbf{\Lambda})_{[n],\{j\}}, \tag{4}$$

where j is the flipped coordinate and where $(\mathbf{H}\mathbf{\Lambda})_{[n],\{j\}}$ denotes the j-th column of $\mathbf{H}\mathbf{\Lambda}$. This update can be performed in linear time, and $F(\bar{\mathbf{y}})$ can subsequently be obtained in linear time as well via simple matrix multiplications, see [4] for details. □

The update scheme permits an extremely efficient implementation of the local search, whose total runtime is roughly the same as the one needed for training a *single* classification model. Further, kernel matrix schemes can nicely be integrated into the overall framework [4].

4.2 Fast and Simple Gradient-Based Optimization

A specialty of the task (3) is the fact that one can reformulate it as a *continuous* optimization problem: For fixed $(f, b) \in \mathcal{H}_k \times \mathbb{R}$ in (3), the optimal partition is given by $y_i = \operatorname{sgn}(f(\mathbf{x}_i) + b)$ [2]. Hence, one obtains

$$\begin{aligned} \underset{f \in \mathcal{H}_k, b \in \mathbb{R}}{\text{minimize}} \quad & \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\big(y_i', f\big(\mathbf{x}_i'\big) + b\big) \\ & + \gamma \frac{1}{u}\sum_{i=1}^{u} \mathcal{L}_e\big(f(\mathbf{x}_i) + b\big) + \lambda \|f\|_{\mathcal{H}_k}^2 \end{aligned} \tag{5}$$

with *effective loss function* $\mathcal{L}_e(t) = \mathcal{L}(\operatorname{sgn}(t), t)$, see Fig. 2(b). Note that, since the function $\mathcal{L}_e$ is non-convex, the overall problem is non-convex too.

As the partial derivatives do not exist, the objective is not differentiable as well. Unfortunately, this excludes the direct use of efficient gradient-based optimization strategies. One possible way to cope with this situation is to resort to differentiable surrogates for both $\mathcal{L}$ and $\mathcal{L}_e$, see Fig. 2. Plugging in these surrogates and using the notation $\bar{\mathbf{x}}_i = \mathbf{x}_i'$ for $i = 1, \ldots, l$ and $\bar{\mathbf{x}}_i = \mathbf{x}_{i-l}$ for $i = l+1, \ldots, n$, one can rewrite (3) as

$$\begin{aligned} F_\gamma(\mathbf{c}) = {} & \frac{1}{l}\sum_{i=1}^{l} \frac{1}{\eta} \log\big(1 + \exp\big(\eta\big(1 - y_i' f(\bar{\mathbf{x}}_i)\big)\big)\big) \\ & + \frac{\gamma}{u}\sum_{i=1}^{u} \exp\big(-3\big(f(\bar{\mathbf{x}}_{l+i})\big)^2\big) \\ & + \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j), \end{aligned} \tag{6}$$

with hypothesis $f(\cdot) = \sum_{j=1}^{n} c_j k(\cdot, \bar{\mathbf{x}}_j)$ and $\|f\|_{\mathcal{H}_k}^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j)$. The following theorem states that one can efficiently compute both the objective and its gradient for, e.g., *sparse data*:

Theorem 2 [4] *For a linear kernel with patterns in* $\mathbb{R}^d$ *and data matrix* $\mathbf{X} \in \mathbb{R}^{n \times d}$ *with* $s \ll nd$ *nonzero entries, one can compute* $F_\gamma(\mathbf{c})$ *and* $\nabla F_\gamma(\mathbf{c})$ *in* $\mathcal{O}(s)$ *time and space for an arbitrary* $\mathbf{c} \in \mathbb{R}^n$.

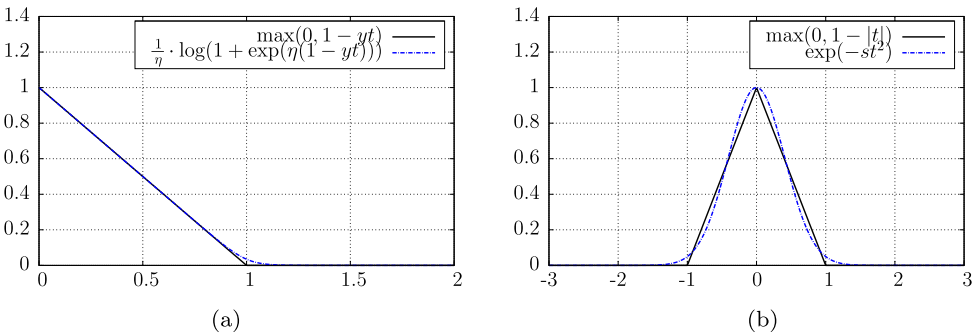

Fig. 2 The hinge loss $\mathcal{L}(y,t) = \max(0, 1 - yt)$ and its differentiable surrogate $\mathcal{L}(y,t) = \frac{1}{\eta}\log(1+\exp(\eta(1-yt)))$ with $y = +1$ and $\eta = 20$ are shown in (**a**). The effective loss $\mathcal{L}(t) = \max(0, 1 - |t|)$ along with its surrogate $\mathcal{L}_e(t) = \exp(-st^2)$ with $s = 3$ are given in (**b**)

Table 1 Classification performances of supervised and semi-supervised support vector machines on dense data (`USPS(i,j)` with $d = 256$) and sparse data (`TEXT` with $d = 7511$).

Data Set	l	u	LIBSVM	UniverSVM	S^2RLSC	QN-S^3VM
USPS(2,5)	16	806	10.5 ± 4.7	5.6 ± 3.6	6.3 ± 2.8	5.4 ± 1.7
USPS(2,7)	17	843	4.9 ± 2.9	2.3 ± 1.0	1.4 ± 0.2	3.6 ± 3.3
USPS(3,8)	15	751	12.9 ± 8.3	8.0 ± 3.5	6.2 ± 2.7	10.8 ± 8.9
USPS(8,0)	22	1108	5.0 ± 2.0	2.8 ± 2.2	1.8 ± 0.6	3.4 ± 1.3
TEXT	48	924	24.8 ± 9.6	6.7 ± 0.9	6.2 ± 0.9	8.2 ± 3.2
TEXT	389	584	4.9 ± 0.7	4.7 ± 0.4	5.0 ± 1.1	4.6 ± 0.6

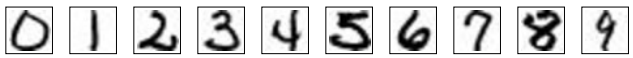

Fig. 3 The USPS data set [5]

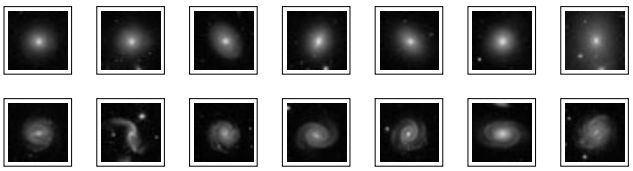

Fig. 4 Elliptical (*top*) and spiral (*bottom*) galaxies (SDSS)

Proof Sketch The gradient can be written as

$$\nabla F_\gamma(\mathbf{c}) = \mathbf{K}\mathbf{a} + 2\lambda\mathbf{K}\mathbf{c} \tag{7}$$

with an appropriately defined vector $\mathbf{a} \in \mathbb{R}^n$. Since $\mathbf{X}$ is sparse, one can compute $\mathbf{K}\mathbf{c} = \mathbf{X}(\mathbf{X}^\mathrm{T}\mathbf{c})$ in $\mathcal{O}(s)$ time for a given $\mathbf{c}$. This permits the efficient computation of $F_\gamma(\mathbf{c})$ and $\nabla F_\gamma(\mathbf{c})$ as well, see [4] for details. □

Similar shortcuts can be obtained for other data settings. For the final local search scheme search (in $\mathbb{R}^n$), one can then resort to simple black-box gradient based optimization tools. For the task at hand, *limited memory quasi-Newton* schemes depict an ideal choice [4].

5 Experiments and Applications in Astronomy

Semi-supervised learning has a tremendous potential in real-world settings in case only few labeled patterns are given. In Table 1, the classification errors of a standard support vector machine (LIBSVM) and three semi-supervised support vector machine implementations are given (S^2RLSC and QN-S^3VM correspond to Secs. 4.1 and 4.2, respectively, and Universvm is another competitor; l denotes the number of labeled, and u the number of unlabeled patterns used for training). The considered data sets stem from two application domains: digit classification (USPS, see Fig. 3) and text classification (Text). It can be clearly seen that all semi-supervised implementations yield better results than the supervised baseline. In some cases, the improvement is dramatic (see, e.g., the results for Text).

The field of astronomy is nowadays faced with enormous amounts of data. For instance, the *Sloan Digital Sky Survey* (SDSS)[3] already contains image data of more than one billion objects. Figure 4 shows a small subset of this data, containing images of *elliptical* and *spiral* galaxies. Since gathering labels for such objects requires the manual inspection by experts in this field, labeled data are usually scarce in astronomy. A direct application of the QN-S^3VM scheme in this context leads to promising results: Assuming that only about ten images are labeled, an application of LIBSVM can only achieve a classification error of 30 % (evaluated on more than 500 independent test samples). In contrast, by incorporating 117 unlabeled patterns, the QN-S^3VM yields an error of less than 20 %.[4]

[3] http://www.sdss.org.

[4] A support vector machine trained on *all* 127 patterns achieves an error of about 15 % for this toy example.

All three examples indicate the potential of such methods in real-world scenarios, like, e.g., the automatic classification of news collected from web pages or the automatic classification of stellar objects in massive astronomical catalogs.

Acknowledgements This work has been supported by funds of the *Deutsche Forschungsgemeinschaft* (grant KR 3695/2-1). The author would like to thank the anonymous reviewers for their helpful comments.

References

1. Borne K (2009) Scientific data mining in astronomy. CoRR. arXiv:0911.0505
2. Chapelle O, Zien A (2005) Semi-supervised classification by low density separation. In: Proc of the 10th int workshop on art intelligence and statistics, pp 57–64
3. Cortes C, Vapnik V (1995) Support vector networks. Mach Learn 20:273–297
4. Gieseke F (2011) From supervised to unsupervised support vector machines and applications in astronomy. PhD thesis, Carl von Ossietzky Universität Oldenburg. iBIT-Universitätsbibliothek, urn:nbn:de:gbv:715-oops-13989
5. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning, 2nd edn. Springer, Berlin
6. Joachims T (1999) Transductive inference for text classification using support vector machines. In: Proc of the 16th int conf on machine learning, pp 200–209
7. Schölkopf B, Smola AJ (2001) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge

Fabian Gieseke received his Diploma degrees in mathematics and computer science from the University of Münster, Germany, and his PhD in computer science from the Carl von Ossietzky University Oldenburg, Germany. He is currently working as a post-doctoral researcher at the University of Oldenburg. His research interests include support vector machines and their extensions to semi- and unsupervised learning settings, and applications in astronomy and energy systems.