ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

A Learning Machine with a Bit-Based Hypothesis Space

Davide Anguita, Alessandro Ghio, Luca Oneto and Sandro Ridella

DITEN - University of Genova
Via Opera Pia 11A, I-16145 Genova - Italy

Abstract. We propose in this paper a bit–based classifier, picked from an hypothesis space described accordingly to sparsity and locality principles: the complexity of the corresponding space of functions is controlled through the number of bits needed to represent it, so that it will include the classifiers that will be most likely chosen by the learning procedure. Through an introductory example, we show how the number of bits, the sparsity of the representation and the local definition approach affect the complexity of the space of functions, where the final classifier is selected from.

1 Introduction

The learning process to train a classifier, according to the Structural Risk Minimization (SRM) principle [1], consists first in selecting an appropriate hypothesis space and then, in this space, choosing the function characterized by the best trade-off between underfitting and overfitting tendencies [1]. The first task is known as *model selection* phase and is strictly linked with the estimation of the size (and, thus, of the complexity) of the hypothesis space [1, 2, 3]: examples are the selection of the number of hidden neurons in Artificial Neural Networks or kernel hyperparameters tuning in Support Vector Machine (SVM) models. The second step, instead, is known as *training* phase for creating a model, which is subsequently exploited on new samples in the *feed-forward* phase.

In this paper, we move the spotlights on the model selection phase and we show how general-purpose benefits on the learning process of classifiers can be obtained by introducing bit–based hypothesis spaces, i.e. classes of functions where models are described through a limited number of bits. In the last decades several works have been devoted to adapt Machine Learning (ML) approaches to specific hardware platforms [4, 5, 6, 7] and, in particular, to analyze the effects of parameter quantization on the training and feed-forward phases [8, 9, 10, 11]. Motivations for these activities are usually linked to application-specific requirements and thus include the necessity of implementing a trained system into a resource limited hardware device, the need to accelerate the process of learning with dedicated hardware, and the energy-sparing requirements of applications based on mobile stand-alone devices (e.g. smartphones).

We propose, instead, an innovative bit-based approach to properly shrink the size of the hypothesis space (and thus to reduce its complexity) by tuning the number of bits, used for representing the classifiers, so to more reliably estimate the generalization ability of models. In this framework we also show

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

how encapsulating the notions of sparsity [12] and local complexity [13] in the description of the hypothesis space can lead to further improvements in the estimation of the generalization ability of classification models.

For these purposes, the paper is organized as follows: in Section 2 we introduce the bit–based hypothesis space, while in Section 3 we recall some measure of complexity in order to present, in Section 4, the positive effects of the exploitation of this type of hypothesis space on the learning process.

2 Defining a bit–based hypothesis space

In the framework of supervised learning the goal is to approximate a relationship between examples from a set $\mathcal{X}$ and outputs from a set $\mathcal{Y}$: we assume in this work $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{Y} \in \{\pm 1\}$. The relationship between examples and outputs is encapsulated by a fixed, but unknown, probability measure $\mathcal{P}$ over $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. Let us suppose that a training set $D_n = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ is sampled according to $\mathcal{P}$ and let us denote as $\mathcal{F}$ a class of $\{\pm 1\}$–valued functions $f \in \mathcal{F}$ on $\mathcal{X}$. The learning algorithm maps D_n to $f \in \mathcal{F}$, while the accuracy in representing the hidden relationship $\mathcal{P}$ is measured with reference to a loss function $\ell(f(\boldsymbol{x}), y)$: in particular, we will exploit the *hard loss function* $\ell_H(f(\boldsymbol{x}), y) = (1 - yf(\boldsymbol{x}))/2$ that counts the number of misclassified samples.

Following the ideas of [1], we define a function $f(\boldsymbol{x}) = \text{sign}\,[\boldsymbol{w} \cdot \boldsymbol{x} + b]$ with $\boldsymbol{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, where $\text{sign}(\cdot) = +1$ if $(\cdot) \geq 0$ or $\text{sign}(\cdot) = -1$ otherwise: in other words, $\mathcal{F}$ is the set of all the linear separators in the original input space $\mathcal{X}$. Finding the best $f \in \mathcal{F}$ can be pursued through an Empirical Risk Minimization (ERM) approach:

$$\min_{\boldsymbol{w},b} \quad \sum_{i=1}^{n} \ell_H(f(\boldsymbol{x}_i), y_i) \qquad \text{s.t.} \quad \boldsymbol{w} \in \mathbb{R}^d, b \in \mathbb{R} \tag{1}$$

which is an NP problem. The conventional approach consists in relaxing the hard loss by exploiting an upper bound of the number of errors [1] and, thus, in introducing a constraint to adjust the size of the class [14], according to the SRM principle [1].

On the contrary, in this paper we propose to introduce a bit-based regularization term in Problem (1) in order to restrict the hypothesis space. As a matter of fact, any learning algorithm will be run on a computational workstation, which will be characterized by a finite (even though large, in several cases) precision. We can thus switch from the conventional representation of $\boldsymbol{w}_{j\in\{1,\ldots,d\}}, b \in \mathbb{R}$ to a bit-based representation of the main quantities in Problem (1): $w_{j\in\{1,\ldots,d\}}, b \in \{-2^\kappa + 1, \ldots, 2^\kappa - 1\}$. κ is the number of bits needed for representing $\mathcal{F}$ and it influences the complexity of the space: if we use more bits we can represent more functions and then we have a more complex space. Consequently Problem (1) becomes:

$$\min_{\boldsymbol{w},b} \quad \sum_{i=1}^{n} \ell_H(f(\boldsymbol{x}_i), y_i) \qquad \text{s.t.} \quad w_{j\in\{1,\ldots,d\}}, b \in \{-2^\kappa + 1, \ldots, 2^\kappa - 1\}\,. \tag{2}$$

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

As a sparse representation of the solution is desirable to further improve the classifier performance on new samples generated from $\mathcal{P}$ [12], we introduce another hyperparameter, that is the number of $w_{j\in\{1,\dots,d\}}$ different from zero. In order to include this constraint, Problem (2) must be reformulated as:

$$\min_{\boldsymbol{w},b} \quad \sum_{i=1}^{n} \ell_H(f(\boldsymbol{x}_i), y_i) \qquad \text{s.t.} \begin{cases} w_{j\in\{1,\dots,d\}}, b \in \{-2^\kappa + 1, \dots, 2^\kappa - 1\} \\ \sum_{j=1}^{d} [w_j \neq 0] \leq \zeta. \end{cases} \tag{3}$$

According to the ideas of [13, 3], we can further shrink the hypothesis space: let $D'_{n'} = \{(\boldsymbol{x}'_1, y'_1), \dots, (\boldsymbol{x}'_n, y'_n)\}$ be another set of data consisting of n' samples (originated by $\mathcal{P}$). Then we can choose only those functions which are characterized by an error rate, on this set, below a predetermined threshold: in fact, these classifiers will be most likely chosen by the optimization procedure, being them the models able to combine a small misclassification rate on D_n and a good generalization performance on the set $D'_{n'}$, independent of D_n. By reformulating Problem (3), we have:

$$\min_{\boldsymbol{w},b} \quad \sum_{i=1}^{n} \ell_H(f(\boldsymbol{x}_i), y_i) \qquad \text{s.t.} \begin{cases} w_{j\in\{1,\dots,d\}}, b \in \{-2^\kappa + 1, \dots, 2^\kappa - 1\} \\ \sum_{j=1}^{d} [w_j \neq 0] \leq \zeta, \ \sum_{k=1}^{n'} \ell_H(f(\boldsymbol{x}'_i), y'_i) \leq \varepsilon. \end{cases} \tag{4}$$

Since $D'_{n'}$ is not always available, part of the samples of D_n can be reserved for this purpose, analogously to the approach proposed in [3].

2.1 From theory to practice

Problem (4) is, once again, an NP problem: however, in practice, the solution can be achieved through a branch and bound approach [15]. In particular, in our work, we exploit for this purpose a state–of–the–art solver, CPLEX [16], which anyhow requires that Problem (4) is reformulated as:

$$\min_{\substack{\boldsymbol{w},\, b,\\ \boldsymbol{\xi},\, \boldsymbol{\xi}',\\ \hat{\boldsymbol{\eta}},\, \check{\boldsymbol{\eta}}}} \quad \sum_{i=1}^{n} \xi_i \qquad \text{s.t.} \begin{cases} y_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b\right) \geq \theta - \Theta\xi_i, \quad \forall i \in \{1, \dots, n\} \\ y'_k \left(\boldsymbol{w}^T \boldsymbol{x}'_k + b\right) \geq \theta - \Theta\xi'_k, \quad \forall k \in \{1, \dots, n'\} \\ w_{j\in\{1,\dots,d\}}, b \in \{-2^\kappa + 1, \dots, 2^\kappa - 1\} \\ w_{j\in\{1,\dots,d\}} \leq 2^N \hat{\eta}_j, \quad w_{j\in\{1,\dots,d\}} \geq -2^N \check{\eta}_j \\ \sum_{j=1}^{d} (\hat{\eta}_j + \check{\eta}_j) \leq \zeta, \quad \sum_{k=1}^{n'} \xi'_k \leq \varepsilon \\ \xi_{i\in\{1,\dots,n\}}, \xi'_{k\in\{1,\dots,n'\}}, \hat{\eta}_{j\in\{1,\dots,d\}}, \check{\eta}_{j\in\{1,\dots,d\}} \in \{0, 1\} \end{cases} \tag{5}$$

Problem (5) is the conventional Integer Linear Programming (ILP) problem formulation, where $\theta,\ \Theta \in \mathbb{R}$ are greater than zero. Theoretically speaking, θ should be as small as possible ($\theta \to 0^+$) and Θ as big as possible ($\Theta \to +\infty$) [17]. In practice θ and Θ should be distant (in terms of orders of magnitude) enough so to avoid modifying the nature of the problem.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

Summarizing, in this last formulation the hyperparameters that allow to control the size of the class are three[1]: (I) the number of bits $\kappa \in \{1, \ldots, \infty\}$; (II) the sparsity of the representation $\zeta \in \{1, \ldots, d\}$; (III) the minimum accuracy $\varepsilon \in \left\{\min_{f\in\mathcal{F}} \sum_{k=1}^{n'} \ell_H(f(\boldsymbol{x}'_i), y'_i), \ldots, n'\right\}$, computed on a dataset independent of the training set D_n, that a function must reach to be included in the hypothesis space.

Finally note that the kernel-based extension of Problem (5) is straightforward but out of the scope of this paper.

3 Assessing the complexity of $\mathcal{F}$

In order to assess the complexity of the hypothesis spaces described in Section 2 we can ideally exploit the Vapnik's data independent complexities [1]: though these measures are very powerful, data dependent ones of Bartlett et. al. [18, 2] give better insights on the learning process and can be more easily computed in practice (e.g. [19]).

One of the most exploited data dependent measure is the Maximal Discrepancy. In order to compute the empirical Maximal Discrepancy $\hat{\mathcal{M}}_n(\mathcal{F})$ of an hypothesis space $\mathcal{F}$, we simply have to randomly split the dataset in two halves, switch the labels to the samples in one of the two halves and calculate:

$$\hat{\mathcal{M}}_n(\mathcal{F}) = 1 - 2 \inf_{f\in\mathcal{F}} \frac{1}{n} \left[\sum_{i=1}^{\frac{n}{2}} \ell_H(f(\boldsymbol{x}_i), y_i) + \sum_{i=\frac{n}{2}+1}^{n} \ell_H(f(\boldsymbol{x}_i), -y_i) \right]. \quad (6)$$

Usually, in order to avoid unlucky splittings, $\hat{\mathcal{M}}_n(\mathcal{F})$ is computed for ≈ 30 different splittings though, rigorously speaking, one split is sufficient [2, 19].

4 Bit–based $\mathcal{F}$ and $\hat{\mathcal{M}}_n(\mathcal{F})$: discussion and perspectives

The purpose of this section is to show that we can drastically reduce the complexity (measured as described in Section 3) of the space defined in Section 2 by varying the different hyperparemeters involved. We will also show that this reduction does not noticeably affect the capability of the hypothesis space to learn the functions that will be most likely chosen by the training procedure, while it deeply (and positively) affects the generalization ability of the learned function since the complexity of the space is drastically reduced [1, 13, 3].

The exploited D_n is characterized by $n = 30$, $d = 3$ and the distribution is a multivariate Gaussian distribution where the information (allowing to separate the two classes) is only embedded in the first and the second dimension. We also exploit a set $D'_{n'}$ with $n' = 10$, originated from the same distribution. We build 30 replicates of this dataset in order to obtain statistical relevant results.

[1]Note that, in principle, $\kappa = 0$ and $\zeta = 0$ could be admissible: in any case, we avoid considering these combinations as they lead to degenerate solutions.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

		$\kappa = 1$	$\kappa = 2$	$\kappa = 3$	$\kappa = 4$	$\kappa = 5$	$\kappa = 6$	$\kappa = 7$	$\kappa = 8, \ldots, \infty$
$\zeta = 3,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	4.6 ± 1.5	3.6 ± 1.3	3.2 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2
$\varepsilon = 10$	$\hat{\mathcal{M}}$	30.0 ± 3.5	41.3 ± 2.9	47.8 ± 2.6	49.8 ± 2.8	51.1 ± 2.9	51.6 ± 2.8	51.6 ± 2.8	51.6 ± 2.8
$\zeta = 3,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	4.6 ± 1.5	3.6 ± 1.3	3.2 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2
$\varepsilon = 7$	$\hat{\mathcal{M}}$	28.7 ± 3.4	41.3 ± 2.9	47.1 ± 2.6	49.3 ± 2.8	50.9 ± 2.7	51.6 ± 2.8	51.6 ± 2.8	51.6 ± 2.8
$\zeta = 3,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	4.6 ± 1.5	3.6 ± 1.3	3.2 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2
$\varepsilon = 5$	$\hat{\mathcal{M}}$	26.7 ± 3.8	40.7 ± 2.8	45.6 ± 2.2	48.0 ± 2.6	50.0 ± 2.4	50.7 ± 2.3	50.7 ± 2.3	50.7 ± 2.3
$\zeta = 3,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	4.8 ± 1.5	3.6 ± 1.3	3.2 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2	3.0 ± 1.2
$\varepsilon = 3$	$\hat{\mathcal{M}}$	16.7 ± 3.0	34.0 ± 3.0	40.4 ± 2.8	43.6 ± 3.1	45.1 ± 3.0	45.8 ± 3.1	47.8 ± 3.6	47.8 ± 3.6
$\zeta = 2,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	6.0 ± 1.5	4.4 ± 1.4	3.9 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2
$\varepsilon = 10$	$\hat{\mathcal{M}}$	27.8 ± 3.5	37.8 ± 3.2	43.3 ± 3.0	46.2 ± 3.1	46.7 ± 3.2	47.3 ± 3.2	47.3 ± 3.2	47.3 ± 3.2
$\zeta = 2,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	6.0 ± 1.5	4.4 ± 1.4	3.9 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2
$\varepsilon = 7$	$\hat{\mathcal{M}}$	25.8 ± 3.5	37.1 ± 3.1	42.9 ± 2.9	45.8 ± 3.0	46.4 ± 3.0	46.9 ± 3.4	46.9 ± 3.4	46.9 ± 3.4
$\zeta = 2,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	6.0 ± 1.5	4.4 ± 1.4	3.9 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2
$\varepsilon = 5$	$\hat{\mathcal{M}}$	24.2 ± 3.9	35.8 ± 2.7	41.6 ± 2.6	44.2 ± 2.7	44.9 ± 2.7	45.1 ± 2.8	45.1 ± 2.8	45.1 ± 2.8
$\zeta = 2,$	$\hat{\mathcal{E}}$	7.9 ± 2.0	6.3 ± 1.5	4.4 ± 1.4	3.9 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2	3.7 ± 1.2
$\varepsilon = 3$	$\hat{\mathcal{M}}$	15.8 ± 3.4	29.3 ± 3.0	35.6 ± 3.2	39.8 ± 3.2	40.2 ± 3.2	40.2 ± 3.2	40.4 ± 3.2	40.4 ± 3.2
$\zeta = 1,$	$\hat{\mathcal{E}}$	43.6 ± 2.0	12.7 ± 1.7	9.7 ± 1.4	8.2 ± 1.5	7.4 ± 1.4	7.3 ± 1.3	7.2 ± 1.3	7.2 ± 1.3
$\varepsilon = 10$	$\hat{\mathcal{M}}$	14.7 ± 3.2	28.9 ± 3.7	34.4 ± 3.2	35.8 ± 3.6	37.3 ± 3.4	38.0 ± 3.6	39.8 ± 4.5	38.0 ± 3.6
$\zeta = 1,$	$\hat{\mathcal{E}}$	43.6 ± 2.0	12.7 ± 1.7	9.7 ± 1.4	8.2 ± 1.5	7.4 ± 1.4	7.2 ± 1.3	7.1 ± 1.2	7.1 ± 1.2
$\varepsilon = 7$	$\hat{\mathcal{M}}$	14.7 ± 3.2	28.0 ± 3.6	34.0 ± 3.2	35.1 ± 3.5	36.2 ± 3.4	38.4 ± 5.1	37.8 ± 4.2	36.4 ± 3.4
$\zeta = 1,$	$\hat{\mathcal{E}}$	48.2 ± 3.0	12.7 ± 1.7	9.7 ± 1.4	8.2 ± 1.5	7.4 ± 1.4	7.2 ± 1.3	7.1 ± 1.2	7.2 ± 1.3
$\varepsilon = 5$	$\hat{\mathcal{M}}$	5.3 ± 6.0	24.0 ± 3.4	30.2 ± 3.0	33.1 ± 2.8	34.4 ± 2.9	34.7 ± 2.7	34.7 ± 2.7	34.7 ± 2.7
$\zeta = 1,$	$\hat{\mathcal{E}}$	–	14.4 ± 1.9	10.1 ± 1.5	8.8 ± 1.5	7.9 ± 1.5	7.7 ± 1.4	7.6 ± 1.4	7.7 ± 1.4
$\varepsilon = 3$	$\hat{\mathcal{M}}$	–	18.9 ± 4.5	23.3 ± 4.2	26.2 ± 4.1	27.3 ± 4.0	27.6 ± 3.9	27.8 ± 3.8	27.8 ± 3.8

Table 1: Results obtained on the case study. All values are in percentage. Fields including '–' symbol mean that no solutions can be identified for that combination of hyperparameters.

In Table 1 the minimum error on the training set $\hat{\mathcal{E}}$ and the complexity of the space $\hat{\mathcal{M}}$ are reported by varying the hyperparameters in Problem (5), where we set $\theta = 10^{-3}, \Theta = 10^2$ (a more deepened sensitivity analysis for these parameters is not included in this work because of space constraints). In order to fairly compare the results we use always the same spilt when we compute $\hat{\mathcal{M}}$. It is worth noting that we can remarkably reduce the complexity of the space without losing the possibility of representing the functions that are characterized by good performance on the training set (as underlined in [13, 3]): these functions will be most likely chosen by the learning process and, then, there seem to be no reasons to search for more complex spaces. Moreover, note that few bits are required in order to represents these functions, thus contemplating the whole $\mathbb{R}$ space leads appears to be unmotivated by practical needs [8]. The influence of a sparse representation and of local hypothesis spaces over the complexity of the space itself opens rooms for further investigations, that deserve to be performed in future works.

Broadly speaking, the approach is theoretically sound. In the SRM framework we have to search for the simplest hypothesis space (before looking at the training set [1]) that guaranties the best trade off between accuracy on the training set and complexity of the space. Then the introduction of a bit–based hypothesis space is also encouraged by the basic ML idea to search for the simplest class of functions capable of solving the problem under examination.

As a final remark, an interesting perspective consists in adapting this approach to the conventional Support Vector Machine formulation in order to apply these concepts to more realistic scenarios as well as to better understand the influence of a bit-based class of functions on generalization capabilities of SVM classifiers.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

References

[1] V. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, New York, 1998.

[2] P.L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, 48(1):85–113, 2002.

[3] D. Anguita, A. Ghio, L. Oneto, and S. Ridella. Selecting the hypothesis space for improving the generalization ability of support vector machines. In *International Joint Conference on Neural Networks*, pages 1169–1176, 2011.

[4] R. Genov and G. Cauwenberghs. Kerneltron: support vector "machine" in silicon. *IEEE Transactions on Neural Networks*, 14(5):1426 – 1434, 2003.

[5] S.W. Lee, S.W. Lee, and H.C. Jung. Real-time implementation of face recognition algorithms on dsp chip. In *Audio-and Video-Based Biometric Person Authentication*, pages 1057–1057, 2003.

[6] K. Irick, M. DeBole, V. Narayanan, and A. Gayasen. A hardware efficient support vector machine architecture for fpga. In *International Symposium on Field-Programmable Custom Computing Machines*, pages 304–305, 2008.

[7] M.G. Epitropakis, V.P. Plagianakos, and M.N. Vrahatis. Hardware-friendly higher-order neural network training using distributed evolutionary algorithms. *Applied Soft Computing*, 10(2):398 – 408, 2010.

[8] D. Anguita, A. Ghio, S. Pischiutta, and S. Ridella. A support vector machine with integer parameters. *Neurocomputing*, 72(1):480–489, 2008.

[9] B. Lesser, M. Mücke, and W.N. Gansterer. Effects of reduced precision on floating-point svm classification accuracy. *Procedia Computer Science*, 4:508–517, 2011.

[10] D. Anguita and D. Sterpi. Nature inspiration for support vector machines. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 442–449, 2006.

[11] H. Neven, V.S. Denchev, G. Rose, and W.G. Macready. Training a large scale classifier with the quantum adiabatic algorithm. *Arxiv preprint arXiv:0912.0779*, 2009.

[12] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[13] V. Koltchinskii. Local rademacher complexities and oracle inequalities in risk minimization.(english. *The Annals of Statistics*, 34(6):2593–2656, 2006.

[14] A.N. Tikhonov and V.Y. Arsenin. Methods for solving ill-posed problems. *Scripta Series in Mathematics*, 1979.

[15] E.L. Lawler and D.E. Wood. Branch-and-bound methods: A survey. *Operations research*, 14(4):699–719, 1966.

[16] CPLEX 12.4. Ibm software group. *User-Manual CPLEX*, 2012.

[17] C. Orsenigo and C. Vercellis. Multicategory classification via discrete support vector machines. *Computational Management Science*, 6(1):101–114, 2009.

[18] P.L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3:463–482, 2003.

[19] D. Anguita, A. Ghio, and S. Ridella. Maximal discrepancy for support vector machines. *Neurocomputing*, 74(9):1436–1443, 2011.