Normalization in Support Vector Machines

Arnulf B. A. Graf[1,2] and Silvio Borer[1]
arnulf.graf@tuebingen.mpg.de

1. Swiss Federal Institute of Technology Lausanne
Laboratory of Computational Neuroscience, DI-LCN
1015 Lausanne EPFL, Switzerland
2. Max Planck Institute for Biological Cybernetics
Spemannstrasse 38, 72076 Tübingen, Germany

Abstract. This article deals with various aspects of normalization in the context of Support Vector Machines. We consider fist normalization of the vectors in the input space and point out the inherent limitations. A natural extension to the feature space is then represented by the kernel function normalization. A correction of the position of the Optimal Separating Hyperplane is subsequently introduced so as to suit better these normalized kernels. Numerical experiments finally evaluate the different approaches.
Keywords. Support Vector Machines, input space, feature space, normalization, optimal separating hyperplane

1 Introduction

Support Vector Machines (SVMs) have drawn much attention because of their high classification performance [1]. In this article, they are applied in a computer vision problem, namely the classification of images. SVMs are often combined with a preprocessing stage to form pattern recognition systems. Moreover, it turns out to be intrinsically necessary for the SV algorithm (see [1]-[4]) to have data which is preprocessed. It has been shown [5] that *normalization* is a preprocessing type which plays an important role in this context. Theoretical considerations on the kernel interpretation of normalization and an adaptation of the SV algorithm to normalized kernel functions will be developed in this paper in order to shed new light on such pattern recognition systems.

In this study, we deal with normalization aspects in SVMs. First, normalization in the input space is considered in Sec. 2 and a resulting problem related to SV classification is outlined. A possible solution, namely the normalization of the kernel functions in the feature space, is then subsequently presented. A modification of the SV algorithm is then presented in Sec. 3. This modification takes into account the properties of the normalized kernels and amounts to a correction of the position of the Optimal Separating Hyperplane (OSH). The corresponding numerical experiments are reported in Sec. 4 and Sec. 5 concludes the paper.

2 Normalization in the Input and Feature Space

The normalization of the vectors of the input space can be considered as the most basic type of preprocessing. Assume $\boldsymbol{x} \in \mathbb{R}^N$ is an input vector, the corresponding normalized vector $\tilde{\boldsymbol{x}}$ may be expressed as:

$$\tilde{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|} \in \mathbb{R}^N \tag{1}$$

where $\|\boldsymbol{x}\|^2 = \sum_{i=1}^N x_i^2$. This vector lies on a unit hypersphere of $\mathbb{R}^N$. Thus, if the input vectors are images, normalization in the input space amounts to rescaling the intensity of the pixels of the images since such a preprocessing only changes the norm of the image vector. The SV algorithm is constructed to find the OSH in the feature space, the latter being obtained by a non-linear mapping from the normalized input space. When considering the effect of such a mapping on the normalized input vectors, it appears, in most cases, to cause a loss of normalization or a scale problem as shown in figure 1. This may create a

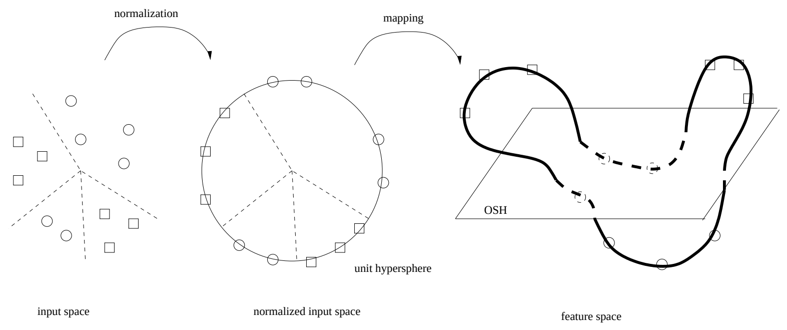

Fig. 1. Normalization in the input space and its representation in the feature space

problem for the SV algorithm since the latter mainly needs "input" vectors in the feature space which are in some way "scaled". As suggested in [5], normalization in the feature space presents a solution to this problem.

Normalization in the feature space is not strictly-speaking a form of preprocessing since it is not applied directly on the input vectors but can be seen as a kernel interpretation of the preprocessing considered above, i.e., an extension to the feature space of the normalization of the input vectors. Normalization in the feature space essentially amounts to redefining the kernel functions of the SVM as it is applied to the unprocessed input vectors. Moreover, since the non-linear mapping is not known, this normalization only makes use of the kernel functions. Assume $K(\boldsymbol{x}, \boldsymbol{y})$ is the kernel function representing a dot product in the feature space. Normalization in the feature space then amounts to defining a new kernel function $\tilde{K}(\boldsymbol{x}, \boldsymbol{y})$ as follows:

$$\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) = \frac{K(\boldsymbol{x}, \boldsymbol{y})}{\sqrt{K(\boldsymbol{x}, \boldsymbol{x}) K(\boldsymbol{y}, \boldsymbol{y})}} \in \mathbb{R} \tag{2}$$

We clearly have $\tilde{K}(\boldsymbol{x}, \boldsymbol{x}) = 1$. Notice that this is always true for RBF kernels $K(\boldsymbol{x}, \boldsymbol{y}) = exp(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{c})$. Thus, all vectors in the feature space lie on a unit hyper-sphere. For monomial kernels $K(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} \cdot \boldsymbol{y})^p$, normalization in the input space is equivalent to normalization in the feature space. Indeed, we have: $K(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = (\tilde{\boldsymbol{x}} \cdot \tilde{\boldsymbol{y}})^p = \left(\frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\| \|\boldsymbol{y}\|}\right)^p = \tilde{K}(\boldsymbol{x}, \boldsymbol{y})$. When the kernel function is replaced by the dot product in the input space ($p = 1$), equation (2) reduces to equation (1). Moreover, note that $\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) = \tilde{\boldsymbol{\varphi}}(\boldsymbol{x}) \cdot \tilde{\boldsymbol{\varphi}}(\boldsymbol{y})$ where $\tilde{\boldsymbol{\varphi}}(\boldsymbol{x}) = \frac{\boldsymbol{\varphi}(\boldsymbol{x})}{\|\boldsymbol{\varphi}(\boldsymbol{x})\|} = \frac{\boldsymbol{\varphi}(\boldsymbol{x})}{\sqrt{K(\boldsymbol{x}, \boldsymbol{x})}}$ stands for the "normalized" mapping and thus the expression above satisfies the conditions of Mercer's theorem.

When considering single-class SVMs as introduced in [6], the normalized kernel functions play a predominant role. We consider dealing with data rescaled such that it lies in the positive orthant i.e. in $[0, \infty)^N$. The normalized kernels then place the datapoints on a portion of the unit hypersphere in the feature space allowing them to be separated from the origin by a hyperplane.

3 Adaptation of the SV Algorithm in a Normalized Feature Space

Normalization in the feature space changes the kernel functions, and thus also the SV optimization problem. We shall here study the implications of a normalized feature space, i.e. of normalized kernel functions, on the SV algorithm. The latter determines the OSH defined by its normal vector $\boldsymbol{w}$ and its position b. By construction, the margins of separation are symmetric around the OSH since both lie at a distance $\delta = \frac{1}{\|\boldsymbol{w}\|}$ from the OSH. However, all the datapoints lie on a unit hypersphere in the normalized feature space. It would thus be more accurate to do classification not according to an OSH computed such that the margins are symmetric around it, but according to an OSH determined such that the margins define equal distances *on* the hypersphere. This may be done by adjusting the value of b. The margins, and thus $\boldsymbol{w}$, are unchanged since the problem is symmetric around $\boldsymbol{w}$. In other words, the separating hyperplane is translated. In order to compute the correction to the value of b, consider figure 2. The intersection of the two margin hyperplanes with the unit hypersphere are represented by the angles α_1 and α_2 defined by $cos(\alpha_1) = b - \delta$ and $cos(\alpha_2) = b + \delta$. The bisection of the angle formed by α_1 and α_2 is represented by the angle φ and can be computed as $\varphi = \frac{\alpha_1 + \alpha_2}{2} = \frac{arccos(b-\delta) + arccos(b+\delta)}{2}$. Moreover, we set $cos(\varphi) = b'$ where b' stands for the new position of the OSH. Finally we get the following expression:

$$b'(b, \boldsymbol{w}) = cos\left(\frac{arccos(b - \frac{1}{\|\boldsymbol{w}\|}) + arccos(b + \frac{1}{\|\boldsymbol{w}\|})}{2}\right) \tag{3}$$

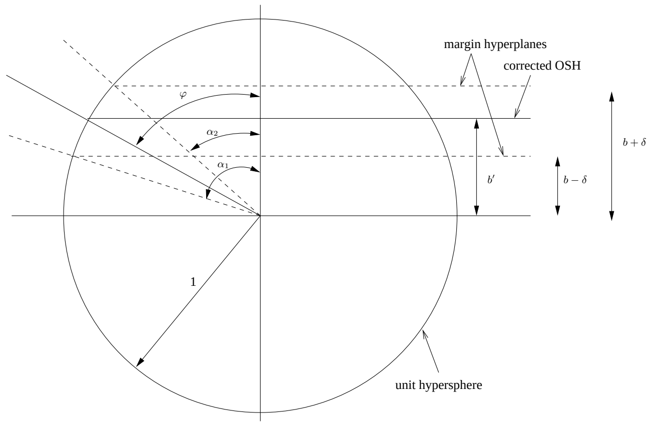

Fig. 2. Normalization in a two-dimensional feature space and computation of b'

From the above equation, we see that $b \neq b'$. The correction of b mentioned here leads to a new optimal separating hyperplane defined by $(\boldsymbol{w}, b')$. This method is valid regardless of the kernel function as long as it is normalized according to equation 2. The value of $b'(b, \boldsymbol{w})$ can be computed from equation 3 once the optimal parameters of the separating hyperplane $(\boldsymbol{w}, b)$ are found by the SV algorithm. The correction is not applied *while* the optimization process is running since this would create convergence problems.

4 Numerical Experiments

Here, we construct a pattern recognition system formed by a SVM with either input space normalization, feature space normalization or the latter combined with the correction of the position of the OSH.

Database The Columbia Object Image Library (`COIL-100`) which can be downloaded from *http://www.cs.columbia.edu/CAVE/* was chosen as in [7] but for different training and testing protocols (see underneath). The latter is composed of 100 different objects, each one being represented by 72 color images (one perspective of the object every 5 degrees) of size 128x128 pixels. These images were first converted to greyscale images and reduced to 32x32 pixels images using averages over square pixel patches of size 4x4 pixels. The database was separated into a *training* and a *testing* dataset. For each object the perspectives 0-30-60-...-330 went into the training set and the perspectives 15-45-75-...-345 into the testing set. In other words, both datasets are composed of regularly-spaced non-overlapping perspectives of each object. We are thus confronted with a multi-class classification problem and the choices mentioned below are made for the training and for the classification protocols.

Training For each object $i = 1, \ldots, 100$, a classifier C_i is generated by assigning a target +1 to the training images of the object i and a target -1 to the training images of all the remaining objects. We thus choose a "one against all" strategy. The regularisation parameter is set *a priori* to 1000.

Classification Each testing image $\boldsymbol{z}$ is presented to each of the classifiers and is assigned to class i where $C_i(\boldsymbol{z}) \geq C_k(\boldsymbol{z}) \quad \forall k \neq i$ according to a "winner take all" strategy. As shown in [8], this approach is computationally very efficient for feedforward neural networks with binary gates such as those encountered in SVMs. Since the class of the testing images is known, an error is computed for each test image. The *classification error* for the experiment is then the average of all the individual errors for each C_i and the corresponding variance over the objects is also computed.

Polynomial kernel functions $K(\boldsymbol{x}, \boldsymbol{y}) = (1 + \boldsymbol{x}\boldsymbol{y})^p$ are particularly well suited for the considered studies. The results are presented in Figure 3. When considering

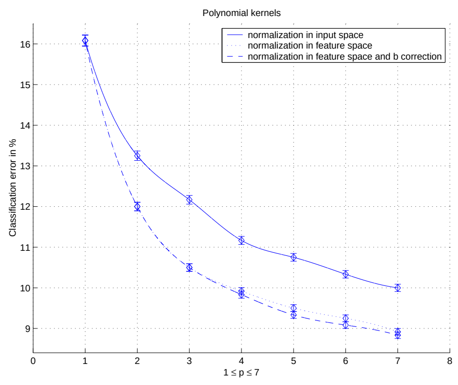

Fig. 3. Classification error and variance with normalization in the input space, in the feature space or in the feature space with correction of b

the results, we notice that the three error curves are monotonically decreasing and none are crossing each other. This seems to reflect the good generalization ability of the considered algorithms. The experiments clearly point out that the normalization in the feature space outperforms the normalization in the input space for all the considered degrees except for $p = 1$. Indeed, for linear kernels, input and feature space normalizations are of the same order of magnitude for large values of the input vectors. This is the case here since the latter represent vectors of size 1024 with values ranging between 0 and 255. Thus, for higher values of p (more sensitive the kernel functions), the difference between these two normalizations gets more pronounced, yielding a bigger difference in the classification performance. Furthermore, the correction of the position of the OSH for normalized kernels decreases the classification error further (albeit not significantly) for the four most sensitive kernels. Again, the linear case is left unchanged since both normalizations are then almost identical. The correction brought to the value of b by the previously-introduced method can be measured as being $\frac{|b-b'|}{b} \simeq 5\%$.

5 Conclusions

When considering a classification machine in computer vision, the preprocessing stage is of crucial importance. In particular when dealing with SVMs, this stage can influence dramatically the results of the classification. In this article, we mentioned that considering the preprocessing stage can be equivalent to studying the kernel functions of SVMs. In this perspective, we discussed one of the most basic types of preprocessing, namely normalization. Normalization was first considered in the input space and it was noticed that this preprocessing was not appropriate when considering SVMs. A natural extension is to move into the feature space by considering normalization of the kernel functions. Since classification is performed in this space, a correction of the position of the OSH was introduced to better suit the normalized kernels. This novel algorithm is shown to have the same optimal solutions for $\boldsymbol{w}$ as the standard SV algorithm, but the considered correction is introduced in the final computation of the position of the OSH. Numerical experiments corroborated that normalization in the feature space outperformed the one in the input space and that the correction of the SV algorithm was revealed to be most effective.

Acknowledgments The fruitful discussions with W. Gerstner, who has introduced me to this subject while staying at the EPFL, are highly appreciated. Reading of the last version was done by B. Schölkopf to whom the author expresses his thanks.

References

[1] B. Schölkopf. *Support Vector Learning.* R. Oldenburg Verlag, Munich, 1997.
[2] B. Schölkopf, C.J.C. Burges & A.J. Smola. *Advances in Kernel Methods.* The MIT Press, MA, 1999.
[3] C. Cortes & V. Vapnik. *Support-Vector Networks.* Machine Learning, Kluwer Academic Publishers, 273-297, 1995.
[4] S. Haykins. *Neural Networks: a Comprehensive Approach.* Prentice Hall, 1999.
[5] R. Herbrich & T. Graepel. *A PAC-Bayesian Margin Bound for Linear Classifiers: Why SVMs work.* Advances in Neural Information System Processing 13, 2001 (in press).
[6] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. Smola & R. C. Williamson. *Estimating the Support of a High-Dimensional Distribution.* Technical Report MSR-TR-99-87, 2000.
[7] M. Pontil & A. Verri. *Support Vector Machines for 3D Object Recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence 20, 637-646, 1998.
[8] W. Maass. *On the Computational Power of Winner-Take-All.* Neural Computation 12, 2519-2535, 2000.