Kernel Learning in Support Vector Machines using Dual-Objective Optimization

Auke-Dirk Pietersma [a] Lambert Schomaker [a] Marco Wiering [a]

[a] *Dept. of Artificial Intelligence, University of Groningen, Groningen*

Abstract

Support vector machines (SVMs) are very popular methods for solving classification problems that require mapping input features to target labels. When dealing with real-world data sets, the different classes are usually not linearly separable, and therefore support vector machines employ a particular kernel function. Such a kernel function computes the similarity between two input patterns, but has as drawback that all input dimensions are considered equally important for computing this similarity. In this paper we propose a novel method that uses the dual objective of the SVM in order to update scaling weight vectors to scale different input features. We developed a gradient descent method that updates the scaling weight vectors to minimize the dual objective, after which the SVM is retrained, and this procedure is repeated a number of times. Experiments on noisy data sets show that our proposed algorithm leads to significantly higher accuracies than obtained with the standard SVM.

1 Introduction

Machine learning and data mining methods try to compute or learn models that are useful for categorizing new unseen patterns or inputs. Although nowadays many machine learning algorithms exist, support vector machines (SVMs) [17, 2, 3] are very popular in many research disciplines [10, 6], because when compared to other methods [9, 4] they often achieve equal or higher accuracies on test data sets. One important choice when using SVMs is the selection of an appropriate kernel function that is needed for efficiently handling non-linearly separable data sets. The radial basis function (RBF) kernel is often chosen for this purpose, but has as drawback that all input features are considered equally important when computing similarities between two feature vectors. Therefore, to make optimal use of SVMs with RBF kernels, preprocessing the input features is important when one wants to achieve the highest possible accuracies.

In order to automatically learn to weigh or scale features, or for selecting a subset of the most informative features, which corresponds to setting particular weights to zero, several methods have been proposed. In [15] SVMs with weighted kernel functions are used, where scaling weights of different features are computed using the information gain ratio. Although the results show that this method outperforms the unweighted counterpart in terms of accuracy and model complexity, the use of information gain ratio handles each feature independently of others, so that context information is ignored. In [1] different techniques for scaling weight vectors are proposed. Although the tested methods obtain similar accuracies as using cross-validation, it is shown that the proposed methods require much less training time to obtain good scaling weight vectors. In [7] a gradient descent method is described that minimizes the slack variables of the primal objective used in SVMs. Then the resulting weights are used to select the most relevant features. Although the results compare favorably to other feature selection methods, the use of gradient descent on the slack variables may have some problems. First of all, note that the slack variables are related to the errors on individual training examples. By minimizing this error, there is an increased risk of overfitting. Second, in case there are few errors on the training data, the method has little information to change the scaling weight vector. Finally, in [12] weighted kernel functions are introduced for kernel perceptrons. In this approach an evolutionary algorithm was used to evolve scaling weights based on maximizing test accuracies. A problem with that approach is that a separate data set is needed for computing the fitness scores, and furthermore, if the accuracy is already high, little improvement can be expected.

Contributions of our research. We propose a novel way to learn scaling weight vectors in which the dual objective is minimized. The value of the dual objective of SVMs is related to the cost of a learned model, and we show that there is a strong correlation between reducing the value of the dual objective by learning scaling weight vectors and the final obtained accuracies on test data. We have developed a gradient descent method that changes the scaling weight vectors in such a way that for each trained SVM, the dual objective is decreased. Since the kernel is updated when changing the scaling weight vectors, we retrain the SVM with the new kernels and repeat this a number of times. For selecting the best model, we finally use the model with the lowest dual objective. Experimental results on 42 noisy data sets show the effectiveness of our proposed algorithm.

Outline of this paper. The next section briefly describes support vector machines, we assume most of this is known to the reader. In Section 3, we describe the weighted radial basis function kernel that we have used. In Section 4 we describe the algorithm for updating the scaling weight vectors. In Section 5, experimental results on very noisy data sets are described. Section 6 concludes this paper.

2 Support Vector Machines

2.1 Introduction

Support vector machines (SVMs) have their origin in statistical learning theory and were first proposed in [16]. In its original (linear) form this supervised learning method and classifier, when given a set of binary labeled training patterns, generates a model describing the underlying function that coincides with the two classes. Typically the set of training patterns used within SVMs has the following form:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in I\!R^p, y_i \in \{-1, 1\}\}_{i=1}^{N} \tag{1}$$

where N and p respectively denote the number of training samples and the dimension/feature size of the patterns. Every pattern $(\mathbf{x}, y)$ consists of a feature vector $(\mathbf{x})$ and a corresponding target class (y). In order to learn a function that describes both classes, the SVM will construct an optimal separating hyperplane between the two. This separation is achieved by constructing an affine hyperplane, and results in two half-spaces. Each of these two half-spaces corresponds to one of the binary classes and therefore can be used to classify unknown patterns. The function $g : I\!R^p \rightarrow \{-1, 1\}$, that maps a pattern $\mathbf{x}$ into one of the half-spaces is often referred to as a dichotomizer or discriminant [5], and is an approximation of the "real" function describing the classes. The problem of categorization brings us to the following problem: in the field of pattern recognition we want to generalize concepts and objects, and like to compare "things" in terms of "this thing is similar to that thing". Moreover we would like the function g in some sense to categorize an unseen pattern similar to those it encountered in the training set. In order to achieve this categorization we need a measure of similarity of the following form:

$$K : \mathcal{X} \times \mathcal{X} \rightarrow I\!R, \tag{2}$$

where $\mathcal{X}$ represents a particular concept. As will we see later, K corresponds to a chosen metric and is called a *kernel function*. The function K takes two instances, $x, x' \in \mathcal{X}$, and returns a real number characterizing their similarity. The first similarity measure used for SVMs was the *canonical dot product* and is defined as:

$$\langle x, x' \rangle = \sum_{i=1}^{p} x_i x'_i. \tag{3}$$

Note that this kernel leads to a linear classifier.

2.2 Separating Hyperplanes, Primal and Dual form

The affine hyperplane or **maximum margin separator** as seen in Equation (4) is known as the *primal-form*, and defines a convex optimization problem, that when solved leads to the maximum-margin hyperplane.

$$\begin{aligned} &\textbf{minimize} && \frac{1}{2}||\mathbf{w}||^2 \\ &\textbf{subject to} && \{y_i(\langle \mathbf{x_i}, \mathbf{w} \rangle + b) \geq 1\} \end{aligned} \tag{4}$$

However, as it turns out there is another formulation which solves the separation problem:

$$\begin{aligned} \max_{\alpha} L_{Dual} \quad &= \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \textbf{subject to} \quad & \alpha_j \geq 0 \\ & \sum_j \alpha_j y_j = 0 \end{aligned} \tag{5}$$

which is known as the *Dual form*. The value L_{dual} will be denoted as the value of the dual objective.

The resulting classification function, Equation (6):

$$h(\mathbf{x}) = sign\left(\sum_i \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle\right) \tag{6}$$

shows that only the examples with non-zero α's contribute to the decision and are called **Support Vectors**. Usually these support vectors are only a fraction of the available training patterns, and therefore this leads to better generalization.

3 Kernel Machines

The hypothesis space in real-world problems can often not be simply described by a linear combination of attributes, but more complex relationships between the attributes need to be found. The latter laid the foundation of multi-layers of linear thresholded functions, resulting in *Neural Networks* [13, 8]. For linear classifiers the usage of "Kernels" offers an alternative solution for getting a more complex hypothesis space. A kernel or combination of kernels can project data into a higher dimensional feature space, thereby increasing the computational power of the linear classifier. This projection corresponds to the following mapping function ϕ:

$$\phi : \mathcal{X} \rightarrow \mathcal{H}, \tag{7}$$

where ϕ will map a pattern's features ($\mathcal{X}$) into a new feature space $\mathcal{H}$. We can extend our previous similarity measure K, equation (2), with this notion of *feature mapping* leading to:

$$K(x, x') = \langle \phi(x), \phi(x') \rangle \tag{8}$$

In the 90's Vapnik and others suggested a way to create non-linear classifiers through the introduction of the "kernel trick". This "trick" merely suggested replacing the $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ in equations (5) and (6) with different types of kernels.

3.1 Weighted Radial Basis Function Kernels

Kernel spaces in general assume that each dimension in feature space is equally important, but there is no *a priori* guarantee that this holds true. Therefore we will introduce a feature space capable of distinguishing the importance between the different features.

Because of its good performance, the radial basis function (RBF) is one of the most popular kernels. Extending the RBF with an additional *scaling weight vector* $\mathbf{v}$, which serves for providing coefficients for scaling each dimension/feature, leads to the following *weighted RBF kernel*:

$$\begin{aligned} exp\left(-\gamma \sum_{i=1}^{p} (\mathbf{x}_i - \mathbf{y}_i)^2\right) \quad &\rightarrow \quad exp\left(-\gamma \sum_{i=1}^{p} \mathbf{v}_i (\mathbf{x}_i - \mathbf{y}_i)^2\right) \\ K_{rbf}(\mathbf{x}, \mathbf{y}) \quad &\rightarrow \quad K_{wrbf}(\mathbf{x}, \mathbf{y}) \end{aligned} \tag{9}$$

Having introduced the ability to scale features is adding another optimization problem, since now we need to find the scaling weight vector $\mathbf{v} \in I\!R^p$ such that greater margins are gained.

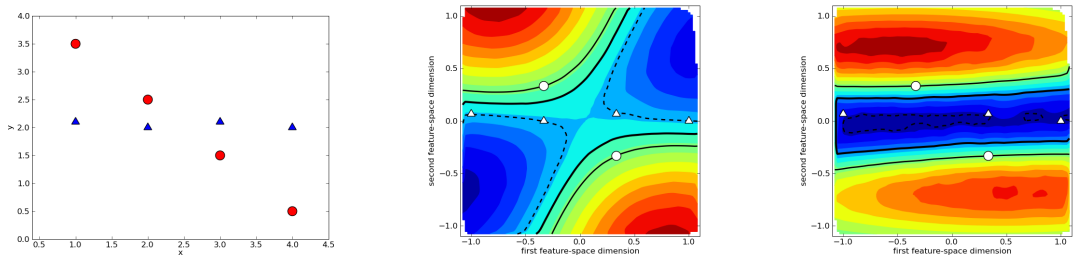
4 Kernel Learning with Dual-Objective Optimization

4.1 Dual-Objective Optimization

Vapnik showed that by optimizing the dual-objective function the SVM is able to obtain the largest possible margin. This does not imply that the *test-accuracy* will necessarily be improved, rather it is a measurement of optimization. We believe that if we can find weight configurations that decrease the dual objective, that this will correspond to a larger margin.

4.2 Toy Example

To illustrate the effect of choosing different scaling weight configurations on the hypothesis space, consider the following toy example. The left image in Figure 1 illustrates a data set consisting of 8 patterns evenly distributed over two classes. It is obvious that the x-axis is of little help in classifying one of the 8 patterns, since for every set corresponding to a particular x value there will always be elements of both classes. The y-axis however provides more information as to whether an unknown pattern belongs to the circle or triangular class.



Non-informative (x) dimension $\Rightarrow$ Standard RBF kernel $\Rightarrow$ Scale dimensions (WRBF)

Figure 1: Given the training examples of the left image, the standard RBF kernel in an SVM learns the decision surface as shown in the middle image. The figure on the right shows a much better separation of the instances that is caused by using a scaling weight vector and then training the SVM.

The middle image illustrates a hypothesis space based on a normal RBF or a WRBF with a scaling weight vector containing solely ones. In the image on the right we modified the scaling weight vector such that the y-axis became 4 times as important as the x-axis. We find, see [11] for extensive details, that not only a decrease in the dual objective was obtained, but we also see that: (1) one less support vector is used, thereby minimizing model complexity and increasing generalization, and (2) an indication of greater margins, visible in the more rapid changes in the *color landscape*.

4.3 Training the Additional Scaling Weight Vector

This section gives the derivation of the gradient descent method for computing optimal scaling weight vectors using the dual formulation, equation (5), when the weighted radial basis function (WRBF) is chosen as a similarity measure. As with most gradient descent methods we need to find the direction of change with respect to some variables, in our case the vector $\mathbf{v}$. Therefore we will use the following update rule using gradient descent leading to minimization of the dual objective:

$$\mathbf{v} = \mathbf{v} - \eta \frac{\partial L_{dual}}{\partial \mathbf{v}} \tag{10}$$

where η is the learning rate (set to 0.001 in our experiments).
To compute the derivative, we will split Equation (10) in two parts:

$$\frac{\partial L_{dual}}{\partial \mathbf{v}_n} = \sum_{i,j} \frac{\partial L_{dual}}{\partial K(\mathbf{x}_i, \mathbf{x}_j)} \times \frac{\partial K(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{v_n}} \tag{11}$$

with K being kernel-function K_{wrbf}

Solving the first part of equation (11) for instances i and j:

$$\begin{aligned}\frac{\partial L_{dual}}{\partial K(\mathbf{x}_i, \mathbf{x}_j)} &= \frac{\partial \left[\sum_{k=1}^{N} \alpha_k - \frac{1}{2} \sum_{k,l=1}^{N} \alpha_k \alpha_l y_k y_l K(\mathbf{x}_k, \mathbf{x}_l) \right]}{\partial K(\mathbf{x}_i, \mathbf{x}_j)} \\ &= -\alpha_i \alpha_j y_i y_j \end{aligned} \tag{12}$$

We dropped the $\frac{1}{2}$ for simplicity. And as for the second part:

$$\frac{\partial K(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{v}} = \frac{\partial exp\left(-\gamma \sum_{r=1}^{p} \mathbf{v}_r (\mathbf{x}_{i,r} - \mathbf{x}_{j,r})^2\right)}{\partial \mathbf{v}} \tag{13}$$

For one particular dimension $\mathbf{v}_n$:

$$\frac{\partial K(\mathbf{x_i}, \mathbf{x_j})}{\partial \mathbf{v}_n} = -\gamma (\mathbf{x}_{i,n} - \mathbf{x}_{j,n})^2 K(\mathbf{x}_i, \mathbf{x}_j) \tag{14}$$

Combining equations (12) and (14) provides the rate of change for instances i and j with respect to $\mathbf{v}_n$:

$$\begin{aligned}\frac{\partial L_{dual}}{\partial \mathbf{v}_n} &= \sum_{i,j} \Big[-\alpha_i \alpha_j y_i y_j \Big] \times \Big[-\gamma (\mathbf{x}_{i,n} - \mathbf{x}_{j,n})^2 \Big] \times K(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i,j} \gamma \alpha_i \alpha_j y_i y_j \Big|_1 (\mathbf{x}_{i,n} - \mathbf{x}_{j,n})^2 \Big|_2 K(\mathbf{x}_i, \mathbf{x}_j) \Big|_3 \end{aligned} \tag{15}$$

Equation (15) consists of three different parts:

Part 1. $\gamma \alpha_i \alpha_j y_i y_j$, describes the magnitude and direction. This direction is caused by $y_i * y_j$, depending on label similarity this is either 1 or -1. The $\alpha' s$ determine the importance of the comparison.

Part 2. $(\mathbf{x}_{i,n} - \mathbf{x}_{j,n})^2$, shows the similarity of the instances i, j with respect to the n^{th} dimension. The higher the similarity the lower the second part becomes. Therefore this part emphasizes dissimilarity based on one feature.

Part 3. $K(\mathbf{x}_i, \mathbf{x}_j)$, describes the total similarity of the instances $\mathbf{x}_i$ and $\mathbf{x}_j$.

The combination of the parts results in a function that can be used to reduce the variance on dimensions between patterns from the same class sharing overall similarity. Reducing the coefficients on those dimensions that show variance automatically increases those dimensions that have little variance, because we keep the L_1 norm of $\mathbf{v}$ constant by normalizing it every time after updating it.

Using the WRBF kernel in SVMs combines the update step of Equation (10) with normal SVM training. Initially, the scaling weight vector is initialized to solely ones, then the SVM is trained, and using the gradient descent update rule the weight vector $\mathbf{v}$ is updated, after which the SVM is retrained. This combination is repeated 100 times. The final weight vector $\mathbf{v}$, for which the lowest value of the dual objective was obtained during 100 iterations, is kept and used for testing.

5 Experiments and Results

In the experiments we want to measure the performance of the WRBF kernel compared to the regular RBF kernel. Since we are only interested in this relative performance, no parameter search was performed. In all experiments the parameters of C and γ were set to 1. Since the L_1 norm of the scaling weight vector is kept constant, there is no possibility for the WRBF to learn better values for γ. Each experiment was repeated 100 times, and the data sets were randomly split into train (80%) and test (20%) sets. Both the SVM+RBF

and SVM+WRBF performed learning as well as testing on exactly the same patterns. The difference in the learning phase is additional feature-weighing, as described in equations (15) and (10), performed by the WRBF. The final weight vector $\mathbf{v}$, for which the lowest value of the dual objective was obtained, was used for testing. The initial configuration contained solely 1's, such that if no improvement was achieved the WRBF would be identical to the RBF.

5.1 The Creation of Noisy Datasets

For this experiment we decided to use 7 well known UCI datasets, namely Breast Cancer Wisconsin, Ecoli, Glass, Ionosphere, Iris, Pima Indians, and Voting. To show how well the SVM algorithms can cope with noise, we use these datasets to create 42 (7×6) new mixed (noisy) datasets. Since we perform 100 experiments with each dataset, this results in 4200 experiments. For each of the 4200 experiments we have an original and noisy dataset. The number of features of the original dataset is duplicated and the new features are filled with features of the noisy dataset. This is done by adding as many new features from the new dataset (possibly some features are used more than once) and picking random real values from each specific feature from the noisy dataset to add to each example. All features were normalized between -1 and 1. The result is that for every experiment the original dataset was doubled in size and 50% noisy features from *real* data is added. The latter is important since we do not use less realistic noisy artificial data.

5.2 Comparison: Correctly Classified Instances

An easy and intuitive approach for comparing two classifiers operating on the same problems is plotting their paired performance. Figure 2 illustrates this paired performance between the RBF (x-axis) and WRBF (y-axis) SVMs. The data points presented are the test results, the number of correctly classified instances, on all combinations of "original" and "noise" data sets. The dashed diagonal line represents equal scores between the two kernels. Looking at the data, we observe a large amount of observations on the left side of the equality line, suggesting that the WRBF kernel classified more instances correctly than the RBF kernel.

Table 1 gives a more detailed view on the scores obtained. From the fifth column we can see that in all cases the WRBF feature-space allowed the SVM to improve its hypothesis space. The scores show improvements of the average accuracies ranging from 0% to $\approx 6\%$.

In [11], we give a thorough statistical analysis of the results obtained with our experiments. By applying the *Wilcoxon Ranked-Sums Test* [18], we obtained sufficient z-scores of -5.05 for rejecting the hypotheses that both classification scores could be drawn from equally performing classifiers. Thus, our WRBF SVM significantly outperforms the standard SVM with an RBF kernel.

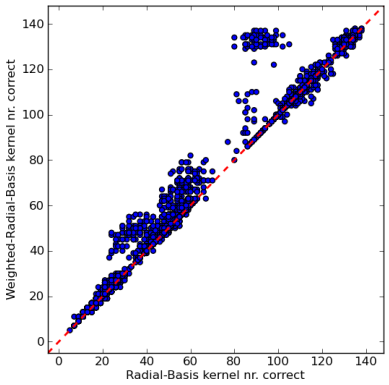


Figure 2: Comparison (4200 experiments) between the number of correctly classified instances by the RBF (x-axis) and the WRBF (y-axis). The diagonal dashed line represents "equal" performance. The WRBF has classified much more instances correctly, since most data points are located on the left side of the equality line. Some data sets profit considerably from using WRBF, see the cloud around (100, 130).

Data Set + noise	Accuracy WRBF	RBF	Total	WRBF - RBF	Dual Objective WRBF	RBF	reduction
Breast Cancer Wisconsin	94.4%	89.6%	83400	4.8%	84.31	106.49	22.17
Ecoli	77.0%	71.2%	40200	5.8%	216.95	242.79	25.83
Glass Identification	45.7%	45.1%	25200	0.6%	224.49	225.93	1.43
Ionosphere	63.9%	63.5%	42000	0.4%	119.15	122.23	3.08
Iris	92.1%	89.2%	18000	2.9%	58.59	68.98	10.39
Pima Indians Diabetes	70.0%	70.0%	91800	0.0%	253.21	253.95	0.74
Congr. Voting Records	67.0%	62.4%	52200	4.8%	141.37	154.13	12.76

Table 1: The compared performance between the average accuracies obtained with WRBF and RBF kernels in a support vector machine setting. In all situations the WRBF and RBF were trained and tested under equal circumstances, meaning equal testing data, training data, C and γ. Under the label "Dual Objective" the average values of the dual objective, L_{dual}, are given for all data sets. In the final column we show how much the dual objective was *minimized* using update rule (10).

5.3 Cost Reduction and Accuracy

In our experiments, the dual-objective optimization algorithm used for learning the scaling weight vectors, is defined as a minimization problem. For that reason the dual objective can be viewed as a cost function. The weight vector $\mathbf{v}$ we are modifying during our optimization process manipulates the feature-space in such a manner that it should make the process of *maximum-margin hyperplane* optimization better. How this "improved" feature-space looks like is not of any concern in this section, we only need to know that through this cost-controlled learning we only have obtained equal or better feature-spaces. The latter is true since we start with a $\mathbf{v}$ configuration such that WRBF = RBF. We will demonstrate that there is a strong relationship between cost reduction and performance improvement. We combined all our experimental results into one big result set, resulting in 4200 observations. From each observation we extracted two variables: $\Delta_{acc} = Acc_{wrbf} - Acc_{rbf}$ and $\Delta_{cost} = L_{dual}(rbf) - L_{dual}(wrbf)$, allowing to compute the correlation coefficient between accuracy improvement and cost reduction. We obtained a correlation coefficient $r = 0.646$ over all 42 data sets ($N = 4200$). This strong relation between accuracy improvement and cost reduction is significant, two-sided with $p < 0.0001$.

From the latter we conclude that improving the objective of the support vector machine has strong positive effects on classification. The strong relation can directly be seen by comparing the fifth and final column in Table 1 and confirms the main idea behind our maximum-margin optimization.

6 Conclusion

In this paper a new method is proposed for learning to scale features in support vector machines that employ non-linear kernels. Although we focused here on weighted radial basis kernels, the method can be used with all existing kernels. The new idea we used to obtain our algorithm is that we do not directly use errors on training data to compute updates to the scaling weight vectors, but we use the dual objective for this purpose. We have shown a strong correlation between reducing the dual objective and the resulting accuracy on separate test data. In our experiments, we have used 7 data sets from the UCI repository, where each data set was combined with uninformative (and therefore noisy) features from another data set. This resulted in a challenging problem where the performance of our SVM with the trained weighted RBF kernel can be considered very good. Furthermore, the results showed that the proposed method significantly outperformed the normal RBF kernel in an SVM.

In future work, we intend to experiment with more complex real-world data sets, such as handwriting recognition of historical Dutch documents [14], and face recognition in which different features will play different roles. Finally, we want to use our method also for the problem of feature subset selection, and compare our algorithm to state-of-the-art feature selection methods.

References

[1] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, pages 131–159, 2002.

[2] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[3] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines (and Other Kernel-Based Learning Methods).* Cambridge University Press, Cambridge, United Kingdom, 2000.

[4] Scott Doniger, Thomas Hofmann, and Miao-Hui Joanne Yeh. Predicting CNS permeability of drug molecules: Comparison of neural network and support vector machine algorithms. *Journal of Computational Biology*, 9(6):849, 2002.

[5] R. O. Duda and P. E. Hart. *Pattern Classification*. John Wiley and Sons, 2000.

[6] Tony Van Gestel, Johan A. K. Suykens, Dirk E. Baestaens, Annemie Lambrechts, Gert Lanckriet, Bruno Vandaele, Bart De Moor, and Joos Vandewalle. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on Neural Networks*, 12(4):809–821, July 2001.

[7] Yves Grandvalet and Stephane Canu. Adaptive scaling for feature selection in SVMs. In *Advances in Neural Information Processing Systems*, pages 553–560. MIT Press, 2002.

[8] Simon Haykin. *Neural Networks, A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, New Jersey, 1999.

[9] Edson J.R. Justino, Flvio Bortolozzi, and Robert Sabourin. A comparison of SVM and HMM classifiers in the off-line signature verification. *Pattern Recognition Letters*, 26(9):1377 – 1385, 2005.

[10] William S. Noble. Support vector machine applications in computational biology. In Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert, editors, *Kernel methods in computational biology*, Computational molecular biology, chapter 3. MIT Press, August 2004.

[11] Auke Dirk Pietersma. Feature space learning in Support Vector Machines through Dual Objective optimization. Master's thesis, University of Groningen (RUG), Groningen, The Netherlands, 2010.

[12] S. Rojas-Galeano, E. Hsieh, D. Agranoff, S. Krishna, and D. Fernandez-Reyes. Estimation of relevant variables on high-dimensional biological patterns using iterated weighted kernel functions. *PLoS ONE*, 3(3), 2008.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing*, volume 1, pages 318–362. MIT Press, 1986.

[14] T. van der Zant, L.R.B. Schomaker, and K. Haak. Handwritten-word spotting using biologically inspired features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1945–1957, 2008.

[15] Bram Vanschoenwinkel, Feng Liu, and Bernard Manderick. Weighted kernel functions for SVM learning in string domains: A distance function viewpoint. In *Proceedings of ICMLC (International Conference on Machine Learning and Cybernetics*, pages 4226–4232, 2005.

[16] V. Vapnik. *Estimation of Dependencies Based on Empirical Data*. Springer-Verlag, New York, 1982.

[17] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[18] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.