Improving the Accuracy of Support Vector Machines

E. A. Zanaty
College of Computer Sciences
Taif University, Taif, Saudi Arabia
Zanaty22@yahoo.com

Sultan Aljahdali
College of Computer Sciences
Taif University, Taif, Saudi Arabia
aljahdali@tu.edu.sa

Abstract: *In this paper we introduce a new kernel function that could improve the SVMs classification accuracy. The proposed kernel function, called polynomial radial basis function (PRBF) combines both Gauss (RBF) and Polynomial (POLY) kernels. We prove that the proposed kernel converges faster than the Gauss and Polynomial kernels and also gives a good classification accuracy in nearly all the data sets specially with high dimension ones. Thereafter SVMs algorithm based on the PRBF is implemented and experimented with non-separable data set with several attributes to prove its efficiency. Then, the obtained results are compared with SVMs algorithms that are based on Gausian and Polynomial kernels.*

Keywords: *Classification problem, SVMs, kernel functions.*

1. Introduction

Vladimir Vapnik and his co-workers have introduced SVMs with a paper at the COLT 1992 conference [1]. They were developed to solve the classification problems, and they have been extended to the domain of regression problems. Classification is one of the most important machine learning operations. Although, it is relatively a new field of research, there exist several classification learning algorithms.

One issue for improving the accuracy of SVMs is finding an appropriate kernel for the given data to improve the accuracy of SVMs. Most research relies on a priori knowledge to select the correct kernel, and then tweaks the kernel parameters via machine learning or trial-and-error. While there exist rules-of-thumb for choosing appropriate kernel functions and parameters, this limits the usefulness of SVMs to expert users, especially since different functions and parameters can have widely varying performance. Williamson et al.[2] published a method for the use of entropy numbers in choosing an appropriate kernel function. It was an attempt to explain kernel function choice by more analytical means rather than previous ad-hoc or empirical methods. The entropy numbers associated with mapping operators for Mercer kernels is discussed. In [3], it was discussed that previous work on invariance transformations was mostly appropriate only for linear SVM classifiers. For non-linear SVM classifiers, it discusses an analytical method of utilizing a kernel principal component analysis PCA map for incorporating invariance transformations [4].

In a recent paper, Tsang et al.[5] discussed a way to take advantage of the approximations inherent in kernel classifiers, by using the Minimum Enclosing Ball algorithm as an alternative means of speeding up training. Training time had previously been reduced mostly by modifying the training set in some way. Their final classifiers, which they called the Core Vector Machine, converged in linear time with space requirements independent from number of data points.

Completely achieving a Support Vector Machine with high accuracy classification, therefore requires specifying the high quality kernel function. This paper addresses the problem of data classification using SVMs. We improve the accuracy of SVMs using a new kernel function. We concentrate on non-linearly separable data sets with this kernel to improve the classification accuracy. The proposed kernel function called PRBF that combines both Gauss and Polynomial kernels, is analyzed to prove its advantages over Gaussian and Polynomial kernels. The SVMs modified by the proposed PRBF kernel function is experimented using different data sets.

The rest of this paper is organized as follows: In section 2, the formulation problem is stated. In section 3, the kernel functions are discussed. We obtain the new kernel function and discuss its analysis in section 4. A comparison between the proposed kernel and traditional ones is given in section 5. Finally, section 6 gives our conclusions.

2.The Problem Formulation

The accuracy problem is usually represented by the proportion of correct classifications. For many data sets, the SVMs may not be able to find any separating hyperplane at all (accuracy equal 0), either because the kernel function is inappropriate for the training data or because the data contains mislabeled examples. The latter problem can be addressed by using a soft margin that accepts some misclassifications of the training examples. A soft margin can be obtained in two different ways. The first is to add a constant factor to the kernel function output whenever the given input vectors are identical. The second is to define a priori an upper bound on the size of the training set weights. In either case, the magnitude of the constant factor to be added to the kernel or to bound the size of the weights controls the number of training points that the system misclassifies. The setting of this parameter depends on the specific data at hand. Completely specifying a Support Vector Machine therefore requires specifying two parameters, the kernel function and the magnitude of the penalty for violating the soft margin. Hence, in order to improve the accuracy of SVMs, we select a suitable kernel function, this is criterion for achieving better results.

2.1 Binary Classification Problem.

Binary classification is the simplest one over all the classification tasks. Given a data set D of N samples: (x_1,y_1), (x_2, y_2), $\ldots(x_n, y_n)$ each sample is composed of a training example x_i of length M, with elements $x_i = (x_1, x_2, \ldots, x_m)$, and a target value $y_i \in \{-1,1\}$. The goal is to find a classifier with decision function, $f(x)$, such that $f(x_i) = y_i, \forall (x_i, y_i) \in D$. The performance of such a classifier is measured in terms of the classification error defined in equation (1):

$$error(f(x),y)=\begin{cases}0 & if \;\; f(x)=y \\ 1 & otherwise\end{cases} \qquad (1)$$

In order to compute the classification error SVM, we use the Structural Risk Minimization (SRM). The Structural Risk Minimization (SRM) considers the complexity of the learning machine when it searches for α to learn the mapping $x \rightarrow y$. This is done by minimizing the expected risk,

$$R_{\exp}(\alpha) = \int error(f(x_i,\alpha), y_i)dp(x,y),$$

where $p(x,y)$ is a prior probability [1].

2.1.2 Linear Classifiers.

There are two cases of linear classifiers. The first where a perfect mapping $x \rightarrow f(x,a)$ can be learned is called the separable case, and the other case where a perfect mapping is unattainable is called the non-separable case [5].

The Separable Case.

Consider the binary classification problem of an arrangement of data points as shown in Fig. (1a) the "square" denotes positive examples with target y_i =+1, belonging to the set S_+ , and the "round" denotes negative examples with target y_i=-1, belonging to S_-. One mapping that can separate S_+ and S_- is,

$$f(x,y) = sign(w.x + b) \qquad (2)$$

where w is a weight vector and b the offset from origin. Given such a mapping, the hyperplane,

$$w.x + b = 0 \qquad (3)$$

defines the decision boundary between S_+ and S_-.

The two data sets are said to be linearly separable by the hyperplane if a pair $\{w, b\}$ can be chosen such that the mapping in equation (1) is perfect, this is the case in Fig. (1a). There are numerous values of $\{w, b\}$ that creates separating hyperplanes. The SVMs classifier finds the only hyperplane that maximizes the margin between the two sets (optimal separating hyperplane) this is shown in Fig. (1b).

The Optimal Separating Hyperplane.

Consider the problem of separating the set of training vectors belonging to two separate classes,

$$D = \{(x_1, y_1), \ldots\ldots, (x_m, y_m)\}, x \in R^n, y \in \{-1,1\} \qquad (4)$$

with the following decision function, $f(x) = sign(w.x + b)$

If the data is linearly separable then,

$$y_i(w.x + b) > 0 \qquad (5)$$

where w is the normal to the hyperplane, $|b|/\|w\|$ is the perpendicular distance from the hyperplane to the origin, and $\|w\|$ is the Euclidean norm of w. A canonical hyperplane is defined for the support vectors on one side of the separating hyperplane,

$$w.x + b = 1 \qquad (6)$$

for the support vectors on the other side,

$$w.x + b = -1 \qquad (7)$$

and for the separating hyperplane,

$$w.x + b = 0 \qquad (8)$$

if we consider a data point x_1, a support vector on one side of the separating hyperplane and x_2 another support vector on the other side then by substituting into (6) and (7) and subtracting we get,

$$w.x_1 + b - (w.x_2 + b) = w(x_1 - x_2) = 2 \qquad (9)$$

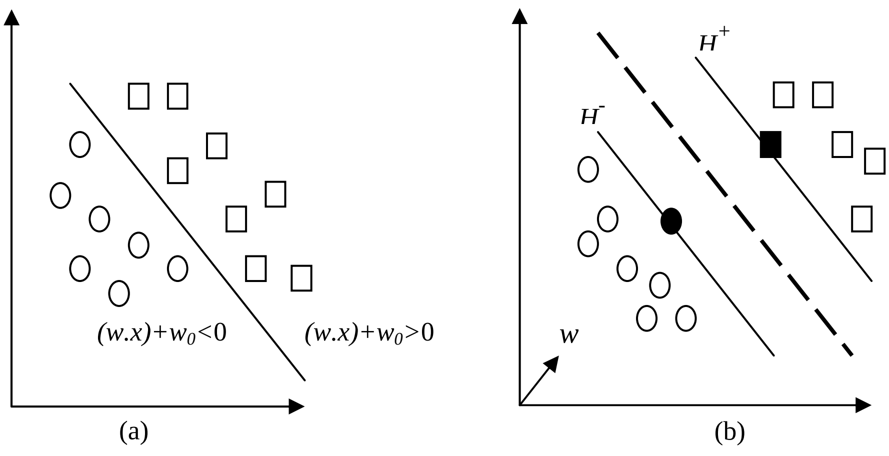
for the separating hyperplane the normal vector is,

$$\hat{w} = \frac{w}{\|w\|}, \qquad (10)$$

the margin can be defined as half the projection of $(x_1$-$x_2)$ onto the normal vector giving,

$$2\gamma = \frac{w(x_1 - x_2)}{\|w\|}, \qquad (11)$$

from equation (9), we can obtain,



(a) a separating hyperplane
(b) the optimal separating hyperplane

Fig. (1): The hyperplane is separated into two separable sets.

$$2\gamma = \frac{2}{\|w\|},$$

which implies that,

$$\gamma = \frac{1}{\|w\|} \qquad (12)$$

For the linearly separable case, the support vector algorithm looks for the separating hyperplane with largest margin. In order to maximize the margin(γ) the following term is minimized,

$$Min(\frac{1}{2}\|w\|^2), \qquad (13)$$

subject to:

$$y_i(w.x_i + b) \geq 1. \qquad \forall i \qquad (14)$$

Now, Lagrange multipliers are applied for two reasons.

First, the constraints will be replaced by constraints on the Lagrange multipliers, which will be much easier to handle. The second is that in this first formulation of the problem, the training data will only appear in the form of data products between vectors. This is a crucial property which will allow us to generalize the procedure to the non-linear case. The following Lagrangian is:

$$l = \frac{1}{2}(w.w) - \sum_{i=1}^{m} \alpha_i (y_i(w.x_i + b) - 1),) \qquad (15)$$

$$\alpha_i \geq 0,$$

Thus, a positive Lagrange multipliers is introduced as, $\alpha_i, \quad i = 1,\ldots,m.$ One for each of the inequality constraints.

If,

$$\frac{\partial l}{\partial w} = 0, \qquad \frac{\partial l}{\partial b} = 0, \qquad (16)$$

then,

$$w = \sum_{i=1}^{m} \alpha_i x_i y_i = 0, \qquad (17)$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0, \qquad (18)$$

re-substituting (17) and (18) back into (15) we obtain the following Wolf dual[1] which is maximized with respect to a_i,

$$w(\alpha) = \sum_{i=1}^{m} \alpha_i - \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j),$$

subject to,

$$\sum_{i=1}^{m} \alpha_i y_i = 0, \qquad \alpha_i \geq 0.$$

An important detail is that a_i=0 for every x_i except the ones that lie on the hyperplanes H^+ and H^-. These points where $a_i \geq 0$ are called support vectors. The number of support vectors in the solution is much less than the number of training examples. This is referred to the sparsity of the solution. When we solved the optimization problem and found the optimal hyperplane, the SVMs can attempt to predict unseen instances [6].

Non-Separable Case.

The SVMs have been restricted to the case where a perfect mapping, $x \rightarrow f(x, a)$, can be learned. Most real-world data sets don't satisfy this condition, so an extension to the above formulation to handle non-separable data is done by creating an objective function that trades of misclassifications against minimizing $||w||^2$. Misclassifications are considered by adding a "slack" variable $\zeta \geq 0$ for each training example, and require that,

$$w.x_i - b \geq +1 - \zeta \qquad for\ y_i = +1,$$
$$w.x_i - b \leq -1 + \zeta \qquad for\ y_i = -1.$$

The new problem we are faced with is to minimize the sum of misclassification errors as well as minimizing $||w||^2$, $\| w \|^2 + c(\sum_I \zeta_i)\kappa$, where c is a regularization parameter used to control the relation between the slack variables and $||w||^2$, k is an integer with typical values of 1 or 2. This minimization problem is also convex as in the linearly separable case. If we choose k to be 1, it has the advantage that ζ_is and their Lagrange multipliers disappear from the dual Lagrangian problem.

This objective function of the dual formulation becomes,

$$\text{maximize}\left\{ L_D \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j x_i x_j \right\} \quad (19)$$

subject to the constraints,

$$\sum_{i=1}^{N} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq c \qquad (20)$$

When this minimization problem is optimized we obtain,

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i,$$
$$b = -\frac{1}{2} w(x_+ + x_-),$$

where x_+ is the positive example with shortest perpendicular distance from the decision boundary, and x_- is the closest negative example.

The difference between the separable and non-separable is the added constraint in equation (20), now a_i's have an upper bound of c.

The support vectors in this solution are not only the training examples that lie on the hyperplane boundary but also the training examples that either falls between the two hyperplanes H_+ and H_- or falls on the wrong side of the decision surface.

2.2. Multi-Class Support Vector Machines.

The multi-class problem is defined as the classification problem that has many classes. While SVMs are binary classifiers, i.e. they can classify two classes, we need some techniques to extend these classifiers to handle multiple classes. The goal of such a technique is to map the generalization abilities of the binary classifiers to the multi-class domain. Multi-class SVMs are usually implemented by combining several two class SVMs. In literature, numerous schemes have been proposed to solve this problem, popular methods for doing this are: one-versus-all method using Winner-Takes-All strategy (WTA SVMs), one-versus-one method implemented by Max-Wins Voting (MWV SVMs), DAG SVMs and error-correcting codes [7]. Hsu and Lin [8] compared these methods on a number of data sets and found that MWV SVMs and WTA SVMs give similar generalization performance [8]. Hastie and Tibshirani [9] proposed a good general strategy called pairwise coupling for combining posterior probabilities provided by individual binary classifiers in order to do multi-class classification and extended it in [10] for speeding up multi-class. Since SVMs do not naturally give out posterior probabilities, they suggested a particular way of generating these probabilities from the binary SVMs outputs and then used these probabilities together with pairwise coupling to do multi-class classification.

Here, we use a multi-class SVM classifier based on one versus one algorithm (the voting strategy) [15,16], and use the Support Vector Machine toolbox for MATLAB. The classifier is designed to read two input data files, the training data and the test data.

3. Kernel Functions.

Kernel functions are used to non-linearly map the input data to a high-dimensional space (feature space). The new mapping is then linearly separable [6,11,12]. The idea of the kernel function is to enable operations to be performed in the input space rather than the potentially high dimension feature space. Hence the inner product does not need to be evaluated in the feature space. The mapping is achieved by replacement of the inner product (x. y)→Φ(x).Φ(y) this mapping is defined by the kernel, K(x, y) = Φ(x).Φ(y).

4. The Proposed Kernel Function.

As we mentioned, kernel functions were proposed to handle non-separable data. They are used to map the input data to a high-dimensional space (feature space). So, for a given non-separable data in order to be linearly separable a suitable kernel is chosen. Classical kernels such as Gauss and Polynomial functions, each one performs better with some data sets. Here, we try to formulate a new kernel that could obtain good performance with all data sets and specially high dimension ones (data sets with many attributes).

The following Polynomial function performs good with nearly all data sets, except high dimension ones, $POLY = (1+ < x_1, x_2 >)^d$, where d is the polynomial degree. The same performance is obtained with Gauss Radial Basis function of the following form,

$$RBF = \exp(-sum(x_{1i}-x_{2i})^2) / (PD),$$

where p is the kernel parameter, D the dimension of the input vector (number of attributes).

We propose a new form of kernel functions which is more complex that could handle high dimension data sets, we denote it as PRBF. This kernel combines both Gauss and Polynomial functions, so it performs better with nearly all data sets,

$$PRBF = (1+ \exp(-sum(x_{1i}-x_{2i})^2) / (PD))^d.$$

The performance of this kernel is shown later by the experimental results.

4.1. The Proposed Kernel Function Analysis.

The proposed kernel function (PRBF) has large convex than the classical Polynomial and Gaussian functions. In addition this function is decreasing when x decreases and it is continuous. If we calculate the limit of (PRBF/RBF) we will get,

let $T = \sum (x_{1i} - x_{2i})^2$ and $V = PD,$

then,

$$\text{PRBF} = \left(1+\frac{e^{-T}}{V}\right)^d \quad \text{and} \quad \text{RBF} = \frac{e^{-T}}{V},$$

hence,

$$\lim_{x_{1i},x_{2i}\to\infty}\left(\frac{\left(1+\frac{e^{-T}}{V}\right)^d}{\frac{e^{-T}}{V}}\right) = \lim_{x_{1i},x_{2i}\to\infty}\left(\frac{\left(1+\frac{1}{e^{T}}\right)^d}{\frac{1}{e^{T}}}\right) = \infty,$$

.

therefore, the proposed function converges faster than the Gauss function. We normalized the data sets to be within the interval [-1,1], because the Polynomial function will diverge in large intervals, and the proposed function has faster convergence within this interval. The following Fig. (2), Fig. (3) and Fig. (4) show the shape of Polynomial (POLY), Gauss (RBF) and the proposed (PRBF) kernels, respectively, within the interval [-1,1] and with two-dimension vectors x1,x2.

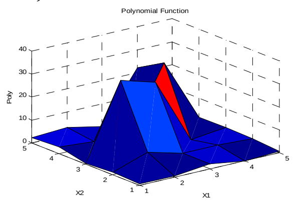


Fig. (2) the Polynomial kernel

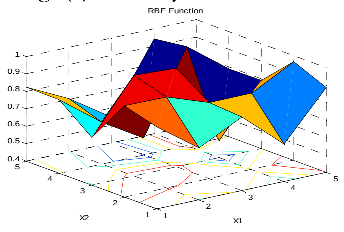


Fig. (3) The Gauss kernel

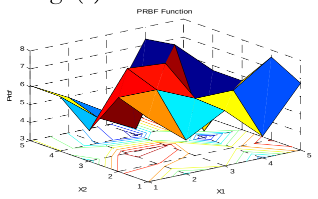


Fig. (4) The proposed kernel

5. A Comparison Between the Proposed Kernel and Traditional Ones.

5.1. Data Sets.

In order to evaluate the performance of the proposed kernel with SVMs, we carried out some experiments with different data sets. Table (1) shows the description of these data sets ([13],[14]) and for more details can be seen in [13,14]. We can divide these data sets according to the training set size into two types, large data sets (1→4) and small ones (5→8).

N0.	Data set	Classes	Attributes	Training	Test
1	Letter	26	16	15000	5000
2	pendigits	9	16	7435	3448
3	waveform	3	21	4700	300
4	Satimage	6	36	4435	2000
5	DNA	3	180	2686	500
6	Segment	7	18	1810	500
7	ABE	3	16	1763	560
8	Zoo	7	17	70	31

Table(1): Data Sets

5.2. Experiments.

We design a multi-class SVM classifier based on one versus one algorithm (the voting strategy)[15] and we use the Support Vector Machine toolbox for MATLAB. We design the classifier to read two input data files, the training data and the test data. Each file is organized as records, each of which consists of a vector of attributes $X=[x_1, x_2, ..., x_M]$ followed by the target $Y \in \{y_1, y_2, ..., y_C\}$ where M is the number of attributes and C is the number of classes. It constructs C(C-1)/2 binary classifiers, and uses the training data to find the optimum separating hyperplane. Finally, we use the test data to compute the accuracy of our classifier from the equation, $Acc = (n / N) *100$, where n is the number of correct classified examples and N is the total number of the test examples. We compare the results of Polynomial, Gauss and the proposed kernels with the classifier as in Table(2).

Data Set	RBF(γ=3)	POLY(d=4)	PRBF(P,d=3)
1	93.9	93.9	93.9
2	82	82.52	83.67
3	95	98.67	99
4	92.95	96.7	96.1
5	94.86	89.46	98.8
6	97	92.12	99
7	100	99.82	99.82
8	87.09	87.09	90.32

Table(2): SVM classification accuracy.

5.3. Result Analysis.

From Table (2) it is obvious that Gauss Radial Basis function with γ=3 accomplishes better accuracy with the small data sets (5→8) than the Polynomial function, and the Polynomial kernel with d=4 gives better results in the large sets (1→4). Whereas our proposed kernel, PRBF, accomplishes the best accuracy in nearly all the data sets, and specially in the largest number of attributes data set number (5), because the proposed function is more complex and combines the performance of both its parents, Gauss and Polynomial functions.

Table (3) presents the mean accuracy we obtained from all kernels, it is obvious that the new proposed kernel obtains the best mean accuracy compared to the classical Gauss and Polynomial function. Fig. (5) summarizes the comparison of the performance of the SVMs with different kernels (POLY, RBF & PRBF), it is clear that the proposed kernel (PRBF) achieves the highest accuracy.

Kernel	Mean accuracy
RBF	92.85
POLY	92.53
PRBF	95.08

Table (3): SVM mean accuracy.

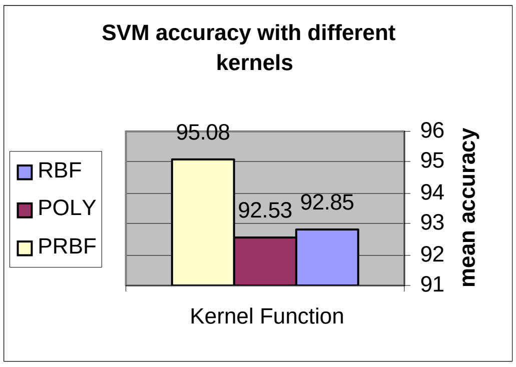


Fig.(5): SVM mean accuracy.

6. Conclusion

In this paper, SVMs have been improved to solve the classification problems by mapping the training data into a feature space by the aid of new kernel functions and then separating the data using a large margin hyperplane. The computational complexity of the classification operation does not depend on the dimensionality of the feature space, which can even be infinite.

We have used different sizes of data sets with different attributes from two sources [13,15] websites. It is obvious from experimental results that RBF gives better accuracy with the small data sets than the Polynomial function. However the Polynomial kernel gives better results in the large data sets. Whereas our proposed kernel PRBF obtains the best accuracy in nearly all the data sets and specially in the largest number of attributes data set, because the proposed function combines the performance of both its parents, Gauss and Polynomial functions.

References

[1] B. Boser , I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers", In D. Haussler (Ed.), Annual ACM Workshop on COLT, Pittsburgh, PA, 144(52), ACM Press, 1992.

[2] R. Williamson, A. Smola, and B. Schölkopf, "Entropy numbers, operators and support vector kernels", pp. 44-127, Cambridge, MA: MIT Press, 1999.

[3] O. Chapelle, and B. Schölkopf, "Incorporating invariances in non-linear support vector machines", In T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, Volume 14, pp. 594-609, Cambridge, MA: MIT Press, 2002.

[4] Hansheng Lei & Venu Govindaraju, "Speeding up multi-class SVM evaluation by PCA and feature selection", Center for Unified Biometrics and Sensors (CUBS), 2005.

[5] I. Tsang, J. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVMs training on very large data sets", Journal of Machine Learning Research 6, pp. 271-363, 2005.

[6] C. Burges, and B. Schölkopf, "Improving the accuracy and speed of support vector machines", Advances in Neural Information Processing Systems 9(375), 1997.

[7] R.Rifin, and A Klautau., "In defense of one vs. all classification", Journal of Machine Learning Research, (5), pp. 101-141, 2004.

[8] C.-W.Hsu, and C.-J. Lin, "A comparison of methods for multi-class support vector Machines", IEEE Transactions on Neural Networks, (13), pp. 415-425, 2002.

[9] T. Hastie, and R. Tibshirani, "Classification by pairwise coupling", Annals of Statistical, 26(2), pp. 451-471, 1998.

[10] Hansheng Lei & Venu Govindaraju, "Speeding up multi-class SVM evaluation by PCA and feature selection", Center for Unified Biometrics and Sensors (CUBS), 2005.

[11] Gunnar Ratsch, "A brief introduction into machine learning", Friedrich Miescher Laboratory of the Max Planck Society, Germany, 2005.

[12] B. Scholkopf, and A. J. Smola, "Learning with kernels, support vector machine, Regularization, Optimization and Beyond", Cambridge, MA.MIT Press, 2002.

[13] http://www.ics.uci.edu/~mlearn/MLRepository.html.

[14] http://www.liacc.up.pt/ML/old/statlog/datasets.html1.

[15] R. Rifin and A. Klautau, "In defense of one vs. all classification", Journal of Machine Learning Research, (5), pp. 101-141, 2004.