Pattern Recognition 44 (2011) 1435–1447

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A set of new Chebyshev kernel functions for support vector machine pattern classification

Sedat Ozer [a,*], Chi H. Chen [b], Hakan A. Cirpan [c]

[a] Electrical & Computer Engineering Department, Rutgers University, 96 Frelinghuysen Rd, CAIP, CORE Building, Piscataway, NJ, 08854-8018, USA
[b] Electrical & Computer Engineering Department, University of Massachusetts, Dartmouth, N. Dartmouth, MA, 02747-2300, USA
[c] Electronics & Communications Engineering Department, Istanbul Technical University, Istanbul, 34469, Turkey

ARTICLE INFO

Article history:
Received 16 December 2008
Received in revised form
19 October 2010
Accepted 27 December 2010
Available online 11 January 2011

Keywords:
Generalized Chebyshev kernel
Modified Chebyshev kernel
Semi-parametric kernel
Kernel construction

ABSTRACT

In this study, we introduce a set of new kernel functions derived from the generalized Chebyshev polynomials. The proposed generalized Chebyshev polynomials allow us to derive different kernel functions. By using these polynomial functions, we generalize recently introduced Chebyshev kernel function for vector inputs and, as a result, we obtain a robust set of kernel functions for Support Vector Machine (SVM) classification. Thus in this study, besides clarifying how to apply the Chebyshev kernel functions on vector inputs, we also increase the generalization capability of the previously proposed Chebyshev kernels and show how to derive new kernel functions by using the generalized Chebyshev polynomials. The proposed set of kernel functions provides competitive performance when compared to all other common kernel functions on average for the simulation datasets. The results indicate that they can be used as a good alternative to other common kernel functions for SVM classification in order to obtain better accuracy. Moreover, test results show that the generalized Chebyshev kernel approaches to the minimum support vector number for classification in general.

1. Introduction

Support Vector Machine (SVM) is a state of the art supervised learning algorithm commonly used in many applications for both regression and classification purposes as in [1–5]. Its theory is based on the structural risk minimization by using the maximum margin idea [1].

SVM is considered "state of the art" due to its strong generalization capability, but its generalization performance depends on using two main steps. The first step is constructing the cost function which needs to be optimized and the constraints for the cost function [6]. The second step is using a kernel function that maps the input data onto a higher dimensional feature space where the data can be linearly separable [7]. An ideal kernel function should not require a parameter, while providing useful performance for all applications. The Gaussian kernel function, which is the most widely employed kernel by the SVMs, requires only one parameter [1]. Choosing the optimal kernel parameter is another problem, and there are various approaches proposed to find this optimal parameter as in [8–10]. However, finding this optimal parameter may require additional computation including an additional optimization which is time and power consuming.

Recently the Chebyshev kernel has been proposed for SVM and it has been proven that it is a valid kernel for scalar valued inputs in [11]. However in pattern recognition, many applications require multidimensional vector inputs. Therefore there is a need to extend the previous work onto vector inputs. In [11], although it is not stated explicitly, the authors recommend evaluating the kernel function on each element pair and then multiplying the outputs (see Chapter V). However, since the kernel functions are defined as the inner product of two given vectors in the higher dimensional space for SVM and since a kernel function provides a measure for the similarity between two vectors, it would be expected that instead of applying kernel functions on each input element (feature), applying them onto vector inputs directly would yield better generalization ability as we will discuss in the following chapters. Therefore in this study we propose generalized Chebyshev kernels by introducing vector Chebyshev polynomials. Using the generalized Chebyshev polynomials, we construct a new family of kernel functions and we show that they are more robust than the ones presented in [11]. On experiments, the proposed generalized Chebyshev kernel function gives its best performance within a small range of integer numbers of the kernel parameter. This property of the kernel function can be used to construct SVMs, where one needs to use semi-parametric kernel functions by choosing the kernel parameter from a small set of integers. In this study, we also show how to construct different kernel functions by using the *Generalized Chebyshev Polynomials*. In order to capture the highly nonlinear boundaries in the Euclidian space, the weighting

* Corresponding author.
E-mail address: sozer1@iit.edu (S. Ozer).

0031-3203/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.12.017

function can be modified. Therefore in this study, we modify the *generalized Chebyshev kernel* for better accuracy by changing the weighting function with an exponential function and propose the *modified Chebyshev kernel* function. In simulations, we also compare the proposed kernel functions to the other commonly used kernel functions. Experimental results show that the proposed set of new kernel functions, on average, show better performance than all other kernel functions used in the test. Moreover, during the tests, we observed that the generalized Chebyshev kernel function approaches the minimum support vector (SV) number in general. This property can be useful for the researchers who need to reduce support vector number in their datasets.

2. Support vector machine

The fundamentals of SVM can be traced back to the statistical learning theory [1]. However, in its current form, SVM is a deterministic supervised learning algorithm, rather than being a statistical learning method. SVM has several different variations based on its cost function as in [1,6], but regardless of these variations, the fundamental form of SVM is based on the idea of inserting an hyperplane between the two (binary) classes which can be done by either inserting such hyperplane in the current data space, (linear SVM), or in the higher dimensional space by using the kernel functions (nonlinear SVM). SVM uses the following formula to find the label of a given test data [1,12]:

$$f(\mathbf{x}) = \mathrm{sgn}\left(\sum_{i=1}^{k} \alpha_i y_i K(\mathbf{x},\mathbf{x_i}) + b\right) \tag{1}$$

where α_i is nonzero Lagrange multiplier of the associated support vector $\mathbf{x_i}$, k the support vector number, $K(.)$ the kernel function, $f(\mathbf{x})$ the class label of the given test data $\mathbf{x}$. The class labels y_i corresponding to the SV $\mathbf{x}_i$, can only have binary values, i.e., $y_i \in \{-1, +1\}$, and b is the bias value. α values are found by maximizing the following function:

$$w(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x_i},\mathbf{x_j}) \tag{2}$$

subject to $\sum_{i=1}^{l} \alpha_i y_i = 0$ and $\alpha_i \geq 0$, where l is the number of training samples. Thus $\mathbf{x_i}$ input vectors, for which the corresponding Lagrange multiplier α_i is nonzero, are called support vectors in the training data.

3. SVM kernel functions

An ideal SVM kernel function yields an inner product of given two vectors in a high dimensional vector space where all the input data can be linearly separated, [1,12]. Therefore, the inner product of any given pair of transformed vectors in the higher dimensional space can be found by applying the kernel function onto the input vectors directly without the need of an appropriate transformation function $\phi(.)$ as

$$K(\mathbf{x},\mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \tag{3}$$

where $K(.)$ is the kernel function. Some of the most common kernel functions are listed below:

Gaussian kernel [1,16]:

$$K(\mathbf{x},\mathbf{z}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}\right) \tag{4}$$

Polynomial kernel [1]:

$$K(\mathbf{x},\mathbf{z}) = \left(\frac{\langle \mathbf{x},\mathbf{z} \rangle + 1}{\beta}\right)^n \tag{5}$$

Wavelet kernel [7]:

$$K(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m} \left(\mathrm{Cos}\left(1.75\frac{x_j - z_j}{a}\right)\exp\left(-\frac{\|x_j - z_j\|^2}{2a^2}\right)\right) \tag{6}$$

where the σ, n and a are the kernel parameters for the Gaussian, polynomial and wavelet kernels, respectively. β is the scaling parameter for the polynomial kernel.

Kernel functions should be applied onto input vectors directly instead of applying them onto each element and combining the results by a product, since the kernel functions are supposed to provide a measure of the correlation of two input vectors in a higher dimensional space.

If we consider the family of the kernel functions where each kernel function is applied onto the pairs of elements individually, for a given pair of two input vectors $\mathbf{x}$ and $\mathbf{z}$, the resulting kernel can be formulated as

$$K_j(x,z) = \langle \phi(x_j), \phi(z_j) \rangle \tag{7}$$

where $K_j(.)$ is the kernel function evaluated on the jth elements of the vector pair $\mathbf{x}$ and $\mathbf{z}$. Thus the final kernel value can be found as

$$K(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m} K_j(x_j, z_j) \tag{8}$$

Both previously proposed Chebyshev kernel and the wavelet kernel functions are constructed in the form of Eq. (8). This approach, however, may yield poor generalization ability. Consider that both input vectors $\mathbf{x}$ and $\mathbf{z}$ may be relatively closer to each other and belong to the same class. In that case it would be expected that the kernel function would give a relatively higher value but if their one pair of elements yields a kernel value $K_j(x_j,z_j)$ close to zero then the whole product will yield a very small value indicating that both vectors have a very small correlation. Thus, SVM will be forced to learn along each element instead of along each vector. (This situation is illustrated on spiral dataset experiments in Figs. 3, 6 and 10.) Therefore, we believe that the kernel functions should provide better results if they are applied onto input vectors directly instead of applying the kernel functions on each element pair first and then combining these results with a multiplication operation.

4. Chebyshev polynomials

Chebyshev polynomials are a set of orthogonal polynomials commonly being used in many applications including filtering. The orthogonal set of polynomials is denoted by $T_n(x)$ $n=0$, 1,2,3,… for the x values between $[-1,1]$. The first kind of Chebyshev polynomials $T_n(x)$, of order n, is defined as [13]

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_n(x) &= 2xT_{n-1}(x) - T_{n-2}(x) \end{aligned} \tag{9}$$

The Chebyshev polynomials of the first kind are orthogonal with respect to the weighting function $1/\sqrt{1-x^2}$, therefore, for given two Chebyshev polynomials, if integrated between the interval $[-1,1]$ we have [13]:

$$\int_{-1}^{1} T_i(x)T_j(x)\frac{1}{\sqrt{1-x^2}}\,dx = \begin{cases} 0 & i \neq j \\ \pi/2 & i = j \neq 0 \\ \pi & i = j = 0 \end{cases} \tag{10}$$

Although we do not have an analytical proof yet, during the simulations, while exploiting the properties of the Chebyshev polynomials for large n values, we noticed that the Chebyshev

polynomials hold the following property:

$$\frac{\sum_{i=0}^{n} T_i(x)T_i(z)}{\sqrt{1-xz}} < \frac{\sum_{i=0}^{n} T_i(x)T_i(x)}{\sqrt{1-x^2}} \tag{11}$$

where x and z are scalars.

5. Chebyshev kernel: previous work

The Chebyshev kernel for the given scalar valued inputs x and z is defined as [11]

$$K(x,z) = \frac{\sum_{i=0}^{n} T_i(x)T_i(z)}{\sqrt{1-xz}} \tag{12}$$

For instance, for scalar values, 3rd order orthogonal Chebyshev kernel is defined as

$$K(x,z) = \left[\frac{1+xz+(2x^2-1)(2z^2-1)}{\sqrt{1-xz}} + \frac{(4x^3-3x)(4z^3-3z)}{\sqrt{1-xz}}\right] \tag{13}$$

As the Chebyshev polynomials are orthogonal only within the region $[-1,1]$, the input data needs to be normalized within this region according to the following formula:

$$x^{new} = \frac{2(x-\mathrm{Min})}{\mathrm{Max}-\mathrm{Min}} - 1 \tag{14}$$

where Min and Max are the minimum and maximum values of the entire data, respectively.

Although it is not clear how the kernel function has been applied onto the vector inputs on the experiments in [11], the computer code that has been sent by the authors of [11] allows us to derive the following equation for the Chebyshev kernel:

$$K(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m} \frac{\sum_{i=0}^{n} T_i(x_j)T_i(z_j)}{\sqrt{1-x_jz_j}} \tag{15}$$

where m is the dimension of the training vectors $\mathbf{x}$ and $\mathbf{z}$.

6. Generalized Chebyshev kernels

Here, we propose a generalized way of expressing the kernel function to clarify the ambiguity on how to implement Chebyshev kernels. To the best of our knowledge, there was no previous work defining the Chebyshev polynomials for vector inputs recursively. Therefore for vector inputs, we define the generalized Chebyshev polynomials as

$$\begin{aligned} &T_0(\mathbf{x}) = 1 \\ &T_1(\mathbf{x}) = \mathbf{x} \\ &T_n(\mathbf{x}) = 2\mathbf{x}T_{n-1}^T(\mathbf{x}) - T_{n-2}(\mathbf{x}) \quad \text{for } n = 2,3,4,\ldots \end{aligned} \tag{16}$$

where T_{n-1}^T is the transpose of the $T_{n-1}(\mathbf{x})$ and $\mathbf{x}$ is a row vector. Therefore, if the polynomial order n is an odd number, the generalized Chebyshev polynomial, $T_n(\mathbf{x})$, yields a row vector, otherwise, it yields a scalar value. Thus by using generalized Chebyshev polynomials, we define generalized nth order Chebyshev kernel as

$$K(\mathbf{x},\mathbf{z}) = \frac{\sum_{j=0}^{n} T_j(\mathbf{x})T_j^T(\mathbf{z})}{\sqrt{a - <\mathbf{x},\mathbf{z}>}} \quad \text{where } a = m \tag{17}$$

where $\mathbf{x}$ and $\mathbf{z}$ are m-dimensional vectors.

In Eq. (17), the denominator must be greater than zero:

$$\sqrt{a - \langle \mathbf{x},\mathbf{z} \rangle} > 0 \tag{18}$$

To satisfy (18), as each element in $\mathbf{x}$ and $\mathbf{z}$ vectors has a value between $[-1,1]$, the maximum value for the inner product $\langle \mathbf{x},\mathbf{z} \rangle$ is equal to $\sum_{i=1}^{m} 1 = m$, thus minimum a value will be equal to m which is the dimension of input vector $\mathbf{x}$.

Table 1
List of the generalized Chebyshev kernel functions up to 4th order.

Kernel parameter: n	Kernel function: $K(\mathbf{x},\mathbf{z})$
0	$\frac{1}{\sqrt{m-c}}$
1	$\frac{1+c}{\sqrt{m-c}}$
2	$\frac{1+c+(2a-1)(2b-1)}{\sqrt{m-c}}$
3	$\frac{1+c+(2a-1)(2b-1)}{\sqrt{m-c}} + \frac{c(4a-3)(4b-3)}{\sqrt{m-c}}$
4	$\frac{1+c+(2a-1)(2b-1)}{\sqrt{m-c}} + \frac{c(4a-3)(4b-3)}{\sqrt{m-c}} + \frac{(8a-8a+1)(8b^2-8b+1)}{\sqrt{m-c}}$

As a result, the 6th order Generalized Chebyshev Kernel can be written as

$$\begin{aligned} K(\mathbf{x},\mathbf{z}) = {} & \frac{1+c+(2a-1)(2b-1)}{\sqrt{m-c}} + \frac{c(4a-3)(4b-3)}{\sqrt{m-c}} \\ & + \frac{(8a^2-8a+1)(8b^2-8b+1)}{\sqrt{m-c}} \\ & + \frac{c(16a^2-20a+5)(16b^2-20b+5)}{\sqrt{m-c}} \\ & + \frac{(32a^3-48a^2+18a-1)(32b^3-48b^2+18b-1)}{\sqrt{m-c}} \end{aligned} \tag{19}$$

where $a = \langle \mathbf{x},\mathbf{x} \rangle$, $b = \langle \mathbf{z},\mathbf{z} \rangle$ and $c = \langle \mathbf{x},\mathbf{z} \rangle$. Also the first 4th order kernel functions are listed in Table 1.

Fig. 1 shows the generalized Chebyshev kernel output $K(z,x)$ for various kernel parameters, where z changes within the range of $[-0.999, 0.999]$, and where x is fixed at a constant value. Fig. 1(a) and (b) shows the kernel function $K(z, 0.77)$, while Fig. 1(c) and (d) shows the $K(z,0)$ value and Fig. 1(e) and (f) shows the $K(z,-0.77)$ value for various kernel parameters.

Fig. 1(a), (c) and (e) shows the results by using Eq. (20):

$$K(\mathbf{x},\mathbf{z}) = \frac{T_n(\mathbf{x})T_n^T(\mathbf{z})}{\sqrt{m - <\mathbf{x},\mathbf{z}>}} \tag{20}$$

Notice that Eq. (20) differs from Eq. (17) since it does not have the summation term in it. Fig. 1(b), (d) and (f) show the result by using Eq. (17). One can observe that, without using the sum, the kernel function cannot be used in SVM for the similarity purpose. The generalized Chebyshev kernel function shape is not symmetric around the x-value, and can be asymmetric as shown in Fig. 1(b) and (f). Unlike the Gaussian kernel whose shape can be altered by the kernel parameter only, the Chebyshev kernels also alter their shape based on the input values.

The following two subsections briefly discuss the validity and the robustness of the generalized Chebyshev kernel.

6.1. Validity

To be a valid SVM kernel, a kernel should satisfy the Mercer Conditions [1,12]. If the kernel does not satisfy the Mercer Conditions, SVM may not find the optimal parameters, but rather it may find suboptimal parameters. Also if the Mercer conditions are not satisfied, then the Hessian matrix for the optimization part may not be positive definite.

Therefore we examine if the Generalized Chebyshev kernel satisfies the Mercer conditions:

Mercer Theorem: To be a valid SVM kernel, for any finite function $g(x)$, the following integration should always be non-negative for the given kernel function $K(x,z)$ [1]:

$$\iint K(x,z)g(x)g(z)\,dx\,dz \geq 0 \tag{21}$$

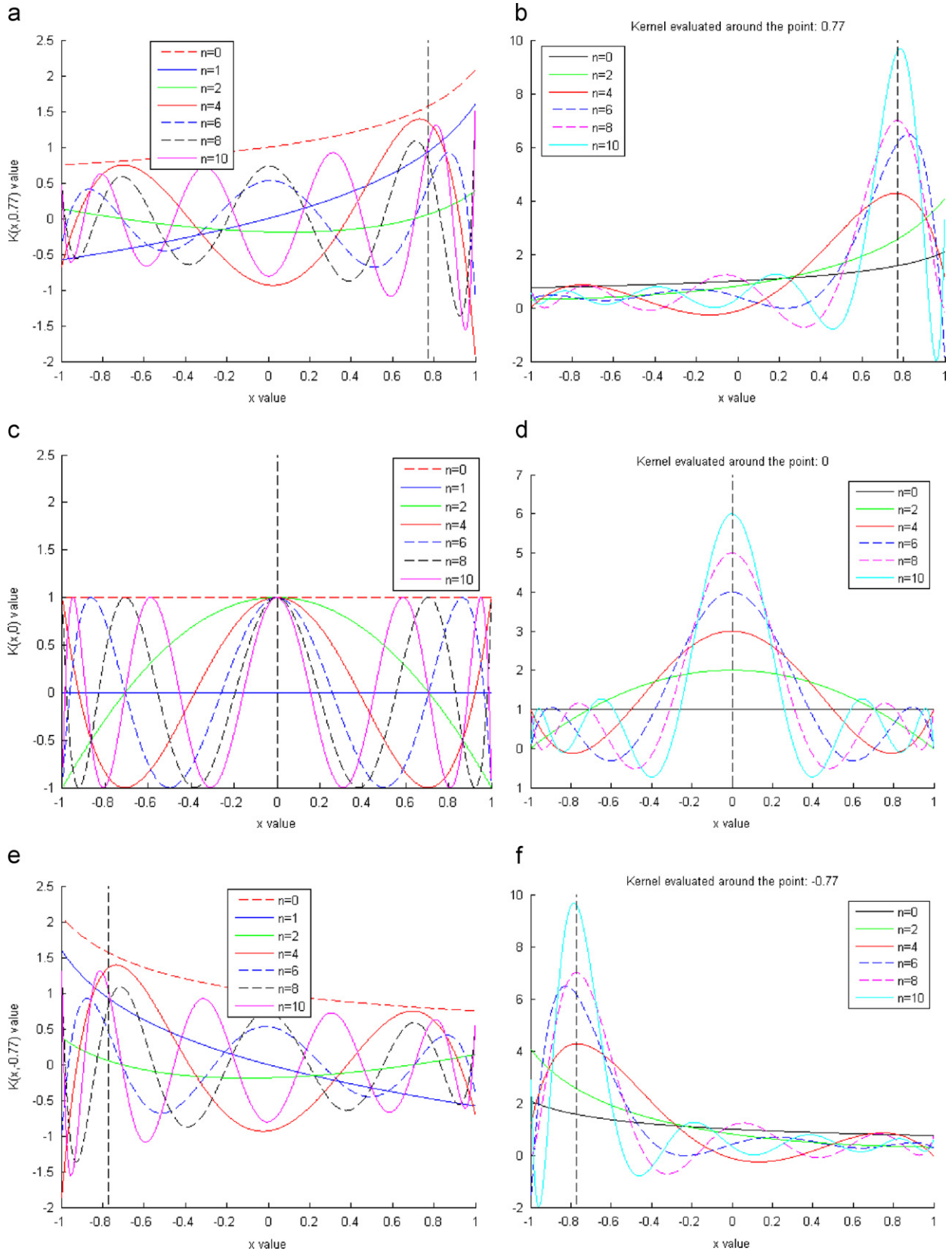

Fig. 1. x-value vs. the kernel output for 3 separate fixed values, i.e. for $K(x,0.77)$, for $K(x,0)$ and for $K(x,-0.77)$ for various kernel parameters. (a) The results by using Eq. (20). (b) The results by using Eq. (17). (c) The results by using Eq. (20). (d) The results by using Eq. (17). (e) The results by using Eq. (20). (f) The results by using Eq. (17).

Proposition. *The multiplication of two valid kernels is also a valid kernel* [12].

Therefore, we can express the nth order Chebyshev kernel as a product of two kernel functions:

$$K(\mathbf{x},\mathbf{z}) = K_{(1)}(\mathbf{x},\mathbf{z})K_{(2)}(\mathbf{x},\mathbf{z}) \tag{22}$$

where

$$K_{(1)}(\mathbf{x},\mathbf{z}) = \sum_{j=0}^{n} T_j(\mathbf{x})T_j^T(\mathbf{z}) = T_0(\mathbf{x})T_0^T(\mathbf{z}) + T_1(\mathbf{x})T_1^T(\mathbf{z}) + \cdots + T_n(\mathbf{x})T_n^T(\mathbf{z}) \tag{23}$$

and

$$K_{(2)}(\mathbf{x},\mathbf{z}) = \frac{1}{\sqrt{m - \langle \mathbf{x},\mathbf{z} \rangle}} \tag{24}$$

Consider that $g(\mathbf{x})$ is a function where g: $\Re^m \to \Re$, then we can evaluate and verify the Mercer condition for $K_{(1)}(\mathbf{x},\mathbf{z})$ as follows by assuming each element is independent from others:

$$\begin{aligned} \iint K_{(1)}(\mathbf{x},\mathbf{z})g(\mathbf{x})g(\mathbf{z})\,d\mathbf{x}\,d\mathbf{z} &= \iint \sum_{j=0}^{n} T_j(\mathbf{x})T_j^T(\mathbf{z})g(\mathbf{x})g(\mathbf{z})\,d\mathbf{x}\,d\mathbf{z} \\ &= \sum_{j=0}^{n} \iint T_j(\mathbf{x})T_j^T(\mathbf{z})g(\mathbf{x})g(\mathbf{z})\,d\mathbf{x}\,d\mathbf{z} = \sum_{j=0}^{n} \left[\int T_j(\mathbf{x})g(\mathbf{x})\,d\mathbf{x} \int T_j^T(\mathbf{z})g(\mathbf{z})\,d\mathbf{z} \right] \\ &= \sum_{j=0}^{n} \left[\left(\int T_j(\mathbf{x})g(\mathbf{x})\,d\mathbf{x} \right) \left(\int T_j^T(\mathbf{x})g(\mathbf{x})\,d\mathbf{x} \right) \right] \geq 0 \end{aligned} \tag{25}$$

Therefore, the kernel $K_{(1)}(\mathbf{x},\mathbf{z})$ is a valid kernel.

Theorem. *Power Series of Dot Product Kernels*:

If a kernel is a function of dot product:

$$K(\mathbf{x},\mathbf{z}) = K(<\mathbf{x},\mathbf{z}>) \tag{26}$$

then its power series expansion [12], $K(t) = \sum_{j=0}^{\infty} a_j t^j$ *is a positive definite kernel if* $a_j \geq 0$ *where* $t = \langle \mathbf{x},\mathbf{z} \rangle$, *for all* j. *As* $K_{(2)}(\mathbf{x},\mathbf{z})$ *is a function of inner product* t, *we can find its Maclaurin expansion, and the expansion coefficients. If all the coefficients are non-negative for the expansion, then this kernel will be a valid kernel*

$$K(t) = K(0) + \sum_{j=1}^{\infty} \frac{K^j(0)}{j!} t^j \tag{27}$$

where jth derivative is defined as

$$K^j(0) = \left. \frac{d^j K(t)}{dt^j} \right|_{t=0} = \frac{\left(\prod_{k=1}^{j} 2k-1 \right)}{2^j m^{(2j+1)/2}} \tag{28}$$

So the Maclaurin expansion takes the form of

$$K_{(2)}(\mathbf{x},\mathbf{z}) = \frac{1}{\sqrt{m}} + \sum_{j=1}^{\infty} \left(\frac{m^{-(2j+1)/2} <\mathbf{x},\mathbf{z}>^j}{2^j j!} \prod_{k=1}^{j} (2k-1) \right) \tag{29}$$

where each coefficient is positive since $m \geq 1$, hence $K_{(2)}(\mathbf{x},\mathbf{z})$ is a valid kernel function too.

As a result the kernel $K(\mathbf{x},\mathbf{z}) = K_{(1)}(\mathbf{x},\mathbf{z})K_{(2)}(\mathbf{x},\mathbf{z})$ is also a valid kernel.

6.2. Robustness

The previously proposed Chebyshev kernel suffered from ill-posed problems as mentioned in [11]. The main reason for the ill-posed problems was the square-root at the denominator of the kernel function. When the value of $\sqrt{1-xz}$ is very close to zero, the kernel value may yield an infinitely big number that can affect the Hessian matrix badly, forcing it to become singular. Although a small ε-value can be added to this expression to avoid division by zero, it still affects the Hessian matrix and it may be needed to re-set for each dataset. Alternatively, using another function such as $\sqrt{q-xz}$ helps to overcome this problem; however, such function reduces the performance of the previously proposed Chebyshev kernel function for SVM; as the q value increases the effect of the denominator will vanish by converging to a constant value. The generalized Chebyshev kernel particularly solves this problem as it uses the function $\sqrt{m - \langle \mathbf{x},\mathbf{z} \rangle}$ as the denominator. Under the assumption of each element and each vector are I.I.D., then the probability of this function being zero is less than the probability of the function $\sqrt{1-xz}$ being zero for $m > 1$. Therefore in real world applications where often $m \gg 1$, the ill-posed problem which is caused by the denominator being zero will be eliminated with a higher probability, if one employees the generalized Chebyshev kernel function.

In experiments, we also show that the generalized Chebyshev kernel is also more robust with respect to the kernel parameter when compared to the Chebyshev kernel.

7. Modified Chebyshev kernels

By introducing the use of the generalized Chebyshev polynomials, we make it easier to derive new kernel functions from the generalized Chebyshev kernels. Based on the generalized Chebyshev polynomials, one can construct new kernel functions or can modify the generalized Chebyshev kernel as needed. Generalized Chebyshev polynomials of the second kind ($U(\mathbf{x})$) as defined in [14], can also be used to create another set of kernel function by using another weighting function as follows:

$$K(\mathbf{x},\mathbf{z}) = \sum_{j=0}^{n} U_j(\mathbf{x})U_j^T(\mathbf{z}) \sqrt{a - <\mathbf{x},\mathbf{z}>} \tag{30}$$

However, the preliminary test results showed very similar results to the kernel function that uses the first kind of generalized Chebyshev kernel polynomials; therefore, in this study we will not include Eq. (30) in the tests.

Here we provide an example on how to modify the generalized Chebyshev kernels by replacing the weighting function in the generalized Chebyshev kernel with an exponential function. As exponential functions can decay faster than the square root function can, in Eq. (25) we replace the weighting function $K_{(2)}(\mathbf{x},\mathbf{z})$, with an exponential function (which is a Gaussian kernel function) to obtain a more nonlinear kernel that captures the nonlinearity better along the decision surface where the surface can change its shape more rapidly.

For this purpose, if we use the function $K_{(2)}(\mathbf{x},\mathbf{z}) = \exp(-\gamma \|\mathbf{x}-\mathbf{z}\|^2)$ which is a Gaussian kernel with $\sigma^2 = (2\gamma)^{-1}$, we can obtain another valid SVM kernel function. The resulting new kernel becomes:

$$K(\mathbf{x},\mathbf{z}) = \frac{\sum_{j=0}^{n} T_j(\mathbf{x})T_j^T(\mathbf{z})}{\exp\left(\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)} \tag{31}$$

where n is the Chebyshev polynomial order and γ is the decaying parameter.

8. Data normalization

Data normalization plays an important role for the generalized Chebyshev kernel as $K_{(2)}(\mathbf{x},\mathbf{z})$ may become complex valued if the condition $\sqrt{m - <\mathbf{x},\mathbf{z}>} \geq 0$ cannot always be satisfied. By normalizing $\mathbf{x}$ and $\mathbf{z}$ values, this condition will always be satisfied. Also the Chebyshev polynomials are defined for the values between

[−1,1]. Therefore a data normalization step prior to SVM training is essential.

One of the data normalization methods is using the maximum value as the maximum of all the input vectors and using the minimum value as the minimum of all the input vectors. Thus each element in the dataset will be normalized according to Eq. (14). However by doing so, the upper and lower limits for each feature will be the same, and consequently, the Euclidian distance between the vectors may be altered. Consequently, the results of this kind of normalization may yield different (altered) generalization abilities as some feature values may be normalized around zero because of the bad scaling, thus essentially ignoring that feature value. Therefore, normalization should be done for each feature separately by considering each feature's own maximum and minimum values carefully.

For a vector in the form of $\mathbf{x}=[x_1,x_2,\ldots,x_m]$, each ith element of $\mathbf{x}$ should be normalized with respect to the maximum and minimum values for that ith element in the whole dataset. Thus the normalized value for each element is defined as

$$x_i^{new}=\frac{2(x_i-\mathrm{Min}_i)}{\mathrm{Max}_i-\mathrm{Min}_i}-1 \tag{32}$$

where Min_i is the minimum value for ith elements among all the possible input vectors and Max_i is the maximum value among all the ith elements (features).

9. Experiments

Here we test and compare various kernel function performances by using different types of datasets. In multi-class experiments, we trained SVM for each class separately as one vs. all. In each experiment, we used the SVM toolbox available at [15]. For both previously proposed Chebyshev kernel and its generalized version, we added a small ε value to the denominator to eliminate the probability of division by zero. In Chebyshev kernel, we use the function $\sqrt{1.002-xz}+\varepsilon$ as the denominator for each test.

Spiral dataset: First, we have chosen spiral dataset to visualize the kernel generalization capabilities. The spiral dataset has 92 data where each feature vector is 2 dimensional. The dataset has 46 data for each class. After the normalization step by using Eq. (32), we used all the data for the training. For the testing step, we created new test grid data within the interval of [−1,1]. We plotted the contours according to Eq. (1) as shown in Figs. 2–10, where the yellow circles represent the support vectors; the brown

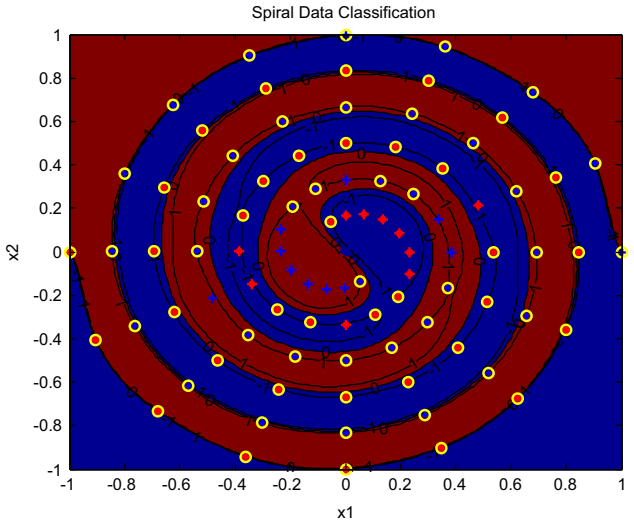

Fig. 2. Spiral dataset test visualization with various kernel functions and kernel parameters. Modified Chebyshev kernel (4th order) svno:72.

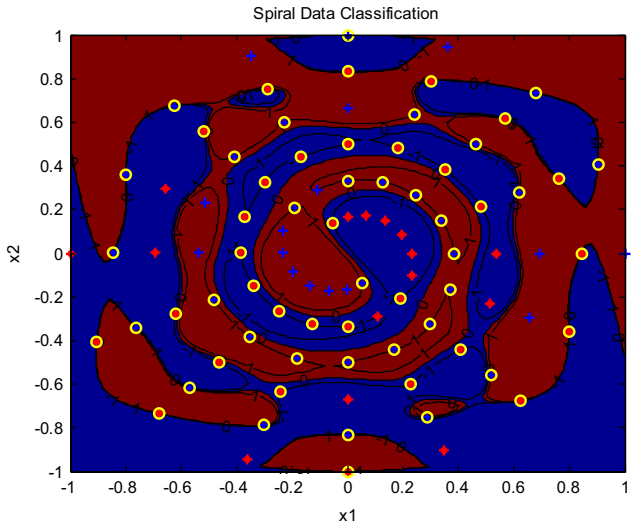

Fig. 3. Spiral dataset test visualization with various kernel functions and kernel parameters. Chebyshev kernel (4th order) svno:62.

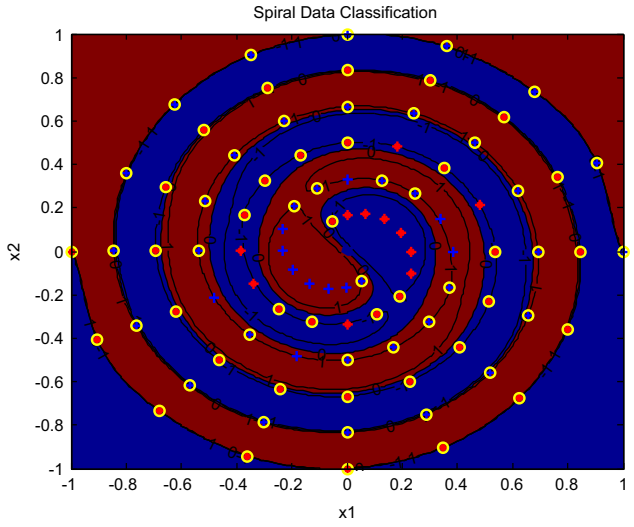

Fig. 4. Spiral dataset test visualization with various kernel functions and kernel parameters. Generalized Chebyshev kernel (4th order) svno:70.

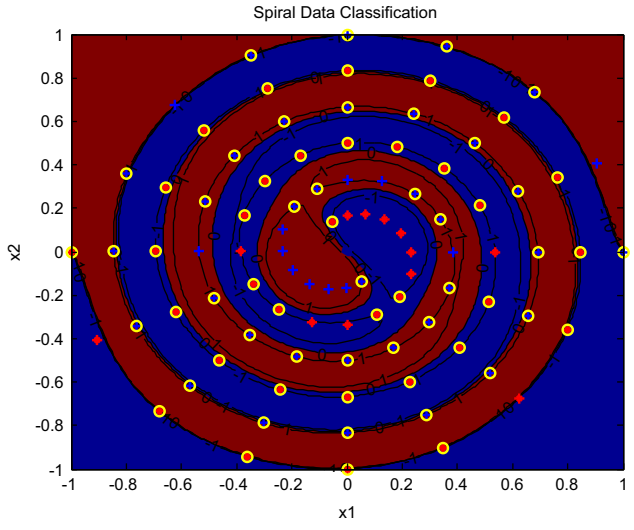

Fig. 5. Spiral dataset test visualization with various kernel functions and kernel parameters. Modified Chebyshev kernel (6th order) svno:68.

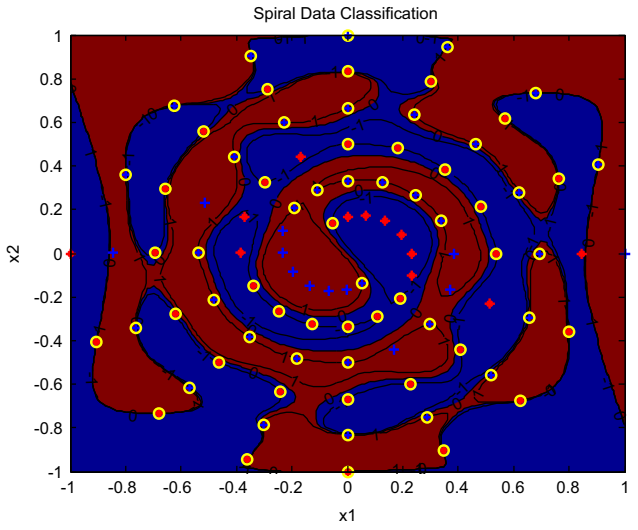

Fig. 6. Spiral dataset test visualization with various kernel functions and kernel parameters. Chebyshev kernel (6th order) svno:68.

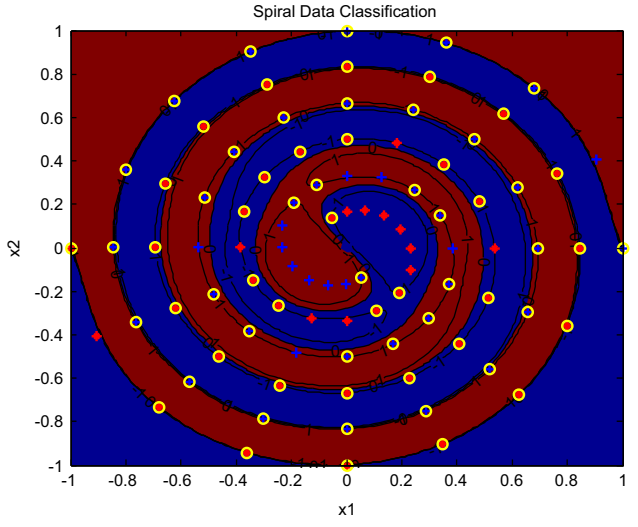

Fig. 7. Spiral dataset test visualization with various kernel functions and kernel parameters. Generalized Chebyshev kernel (6th order) svno:68.

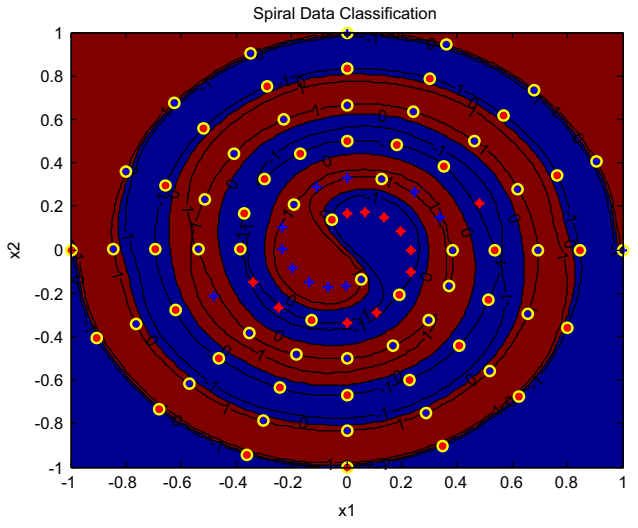

Fig. 8. Spiral dataset test visualization with various kernel functions and kernel parameters. Gaussian kernel ($\sigma=0.33$) svno:70.

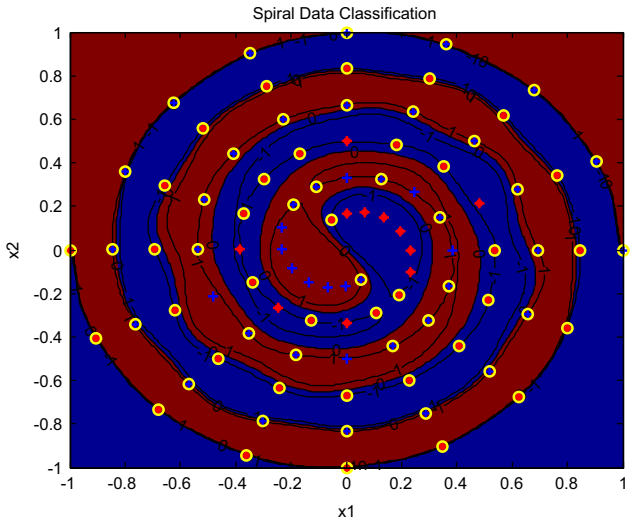

Fig. 9. Spiral dataset test visualization with various kernel functions and kernel parameters. Wavelet kernel ($a=1.5$) svno:70.

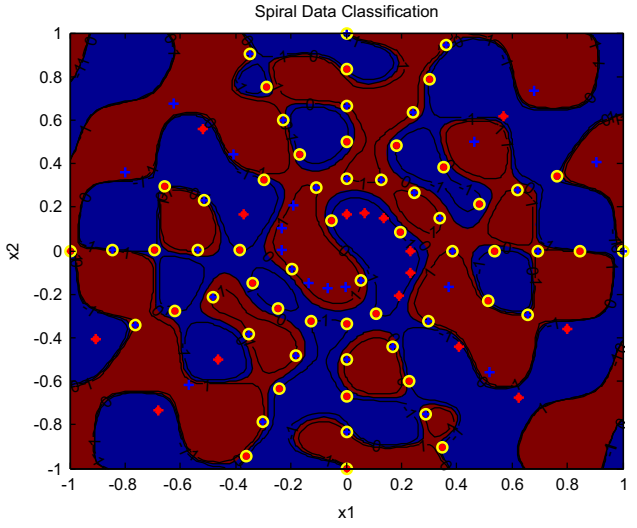

Fig. 10. Spiral dataset test visualization with various kernel functions and kernel parameters. Wavelet kernel ($a=0.2$) svno:62.

region represents $(+1)$ class, the blue region represents (-1) class, and the black lines show the margins for each class where $\boldsymbol{wx}+b=1$ *or* $\boldsymbol{wx}+b=-1$

As shown in Figs. 2–10, the generalized Chebyshev kernel and the modified Chebyshev kernel functions show the minimum SV number 68 while keeping the generalization ability correct for the dataset. For the modified Chebyshev kernel we used $\gamma=1$.

Figs. 3 and 6 show the results obtained from the Chebyshev kernel. Although the Chebyshev kernel function yielded the lowest number of support vectors, it was not able to learn the characteristics of the spiral dataset as shown in Figs. 3 and 6. Thus it yielded poor generalization ability.

The result for the best Gaussian kernel parameter with the lowest support vector number is shown in Fig. 8. The wavelet kernel result, which has good generalization ability with the lowest SV number is shown in Fig. 9. In Fig. 10, it is shown that the lowest support vector number does not always yield good generalization ability with the wavelet kernel.

Image segmentation dataset: We pick another dataset which is currently available at http://www.cs.toronto.edu/~delve/data/image-seg/, known as image segmentation data. The data has

Table 2
Image segmentation dataset test results with various kernel functions.

	Chebyshev kernel		Generalized Cheb.		Modified Cheb.		Gaussian kernel		Polynomial kernel		Wavelet kernel	
	SV no/Test%	Best n	SV no/Test%	Best n	SV no/Test%	Best n/γ	SV no/Test%	Best σ	SV no/Test%	Best n	SV no/Test%	Best a
Sky	12/99.52	0	**5/100**	2	23/100	4/0.5	**5/100**	4.3	6/100	2	86/100	1.4
Path	74/97.10	3	11/99.38	1	**104/99.81**	6/3.0	106/99.71	0.43	21/99.76	6	105/99.71	2.2
Window	105/94	4	33/93.19	1	**41/95.10**	4/1.5	122/94.57	0.4	33/94.48	7	133/94.62	2.7
Foliage	38/94.52	0	30/96.90	4	44/97.10	4/0.5	28/96.71	1.4	**34/97.29**	8	132/95.52	2.7
Cement	122/94.00	4	31/92.33	4	**54/97.14**	3/2.0	88/97.00	0.5	39/95.33	12	88/97.00	1.7
Brickface	29/97.57	0	18/99.00	14	**30/99.48**	3/1.5	93/99.48	0.4	13/99.10	2	53/99.48	1.4
Grass	20/99.48	0	11/99.86	6	28/99.86	4/0.5	**6/99.86**	11	7/99.86	2	85/99.86	1.4

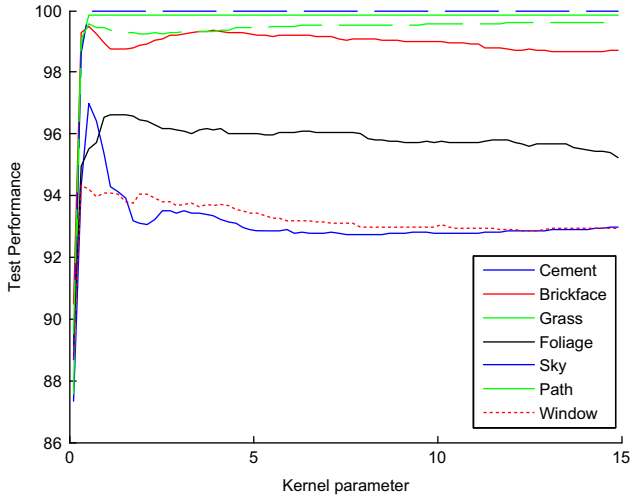

Fig. 11. Gaussian kernel parameter vs. performance.

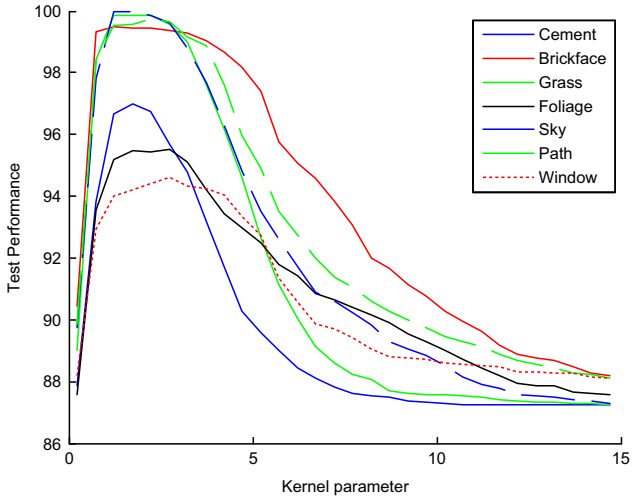

Fig. 12. Wavelet kernel parameter vs. performance.

7 different classes of images. It has 210 data for training and another 2100 data for testing. Each vector has 18 elements with different minimum and maximum values. For the training, we have 30 data for the class (+1) and 180 data for the class (−1) and similarly for testing, we have 300 vs. 1800 data, respectively for each class.

Training error was 0 on all experiments. However, on testing step, the kernel functions showed different performance values on different classes and there was no such winning kernel showing the best performance on every class as shown in Table 2. The modified Chebyshev kernel performed better than others on average. The best performance values with the lowest SV numbers are shown in bold. For the Modified Chebyshev kernel, we heuristically found the γ values.

Table 2 shows the test results for each class with different kernel functions. We provide the accuracy vs. the kernel parameter values in Figs. 11–16 for each class. Figs. 12 and 15 indicate that as the kernel parameter increases, the Chebyshev kernel and wavelet kernel decrease their performance and asymptotically reach a low performance value. Figs. 17–22 show the SV number vs. the kernel parameter value.

In all figures, $\gamma=1$ is used. In Figs. 12 and 17 we show that, as the kernel parameter increases, the Chebyshev and wavelet kernels require more SVs. Moreover, Figs. 13 and 15 show that the classification accuracy of these two kernels asymptotically decreases to a certain value as the kernel parameter increases. However, the other kernel functions are more robust with respect to the kernel parameter when compared to the Chebyshev kernel and wavelet kernel functions.

Iris dataset: Iris dataset is another well known dataset used in many pattern recognition tests. The dataset consists of 150 data for 3 classes. Each vector has 4 features and each class has 50 vectors. We formed training data by taking the first 15 data of each class. Then we used the remaining 105 data for testing. The results are shown in Table 3.

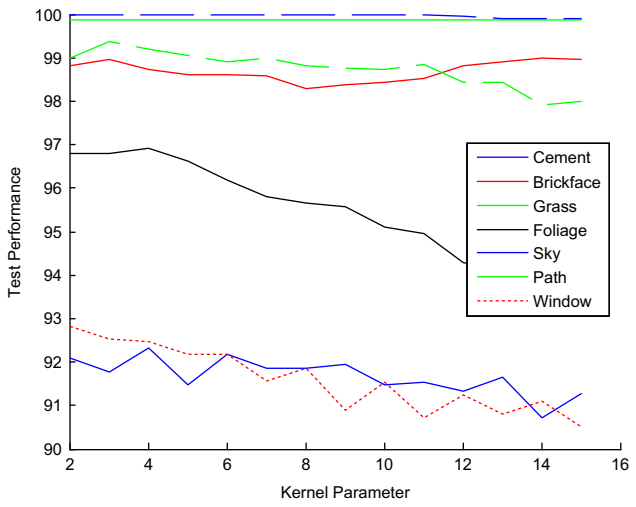

Fig. 13. Generalized Chebyshev kernel parameter vs. performance.

In this experiment only the modified Chebyshev kernel showed the best accuracy values among all 3 classes as shown in Table 3. Figs. 23–25 show the test performance vs. kernel parameter plots for the classes Virginica, Versicolour and Setosa, respectively. Fig. 26 shows SV numbers vs. kernel parameters for the Versicolour class. Fig. 26 shows that the Wavelet and Chebyshev kernels require more

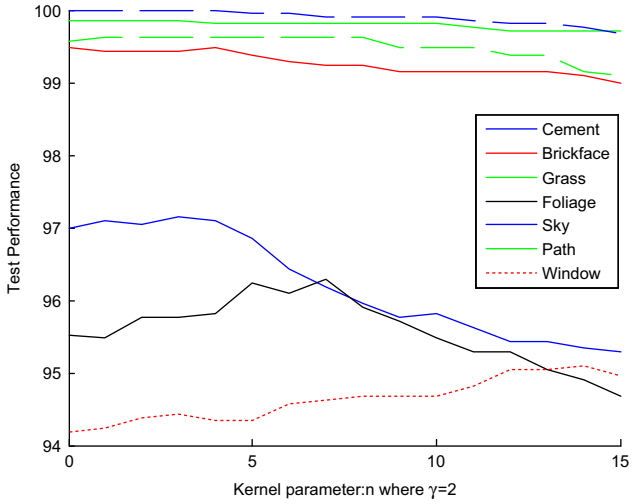

Fig. 14. Modified Chebyshev kernel parameter vs. performance.

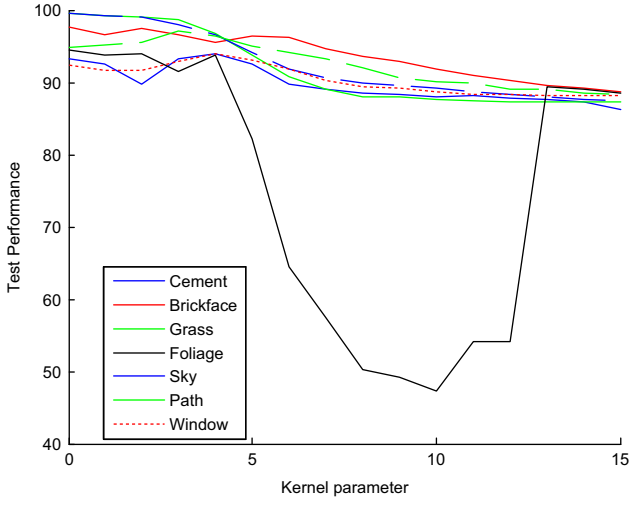

Fig. 15. Chebyshev kernel parameter vs. performance.

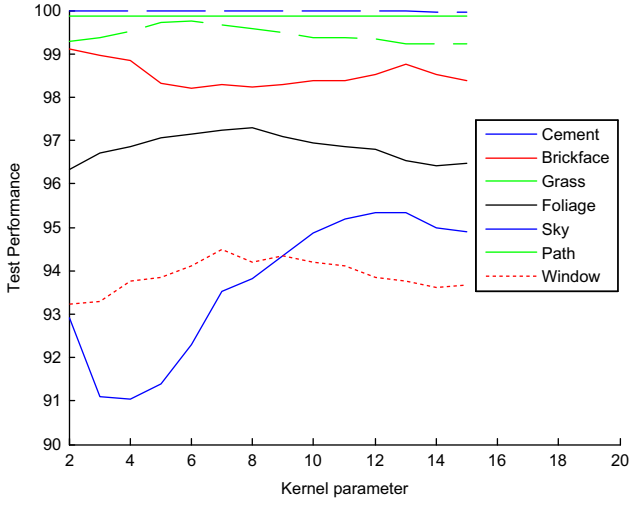

Fig. 16. Polynomial kernel parameter vs. performance.

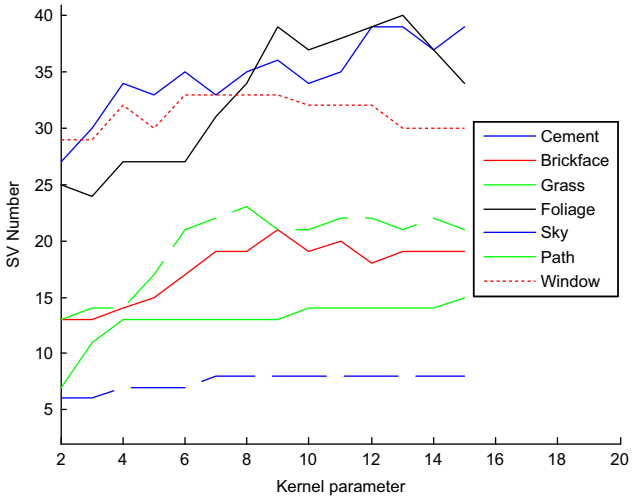

Fig. 17. Polynomial kernel parameter vs. SV number.

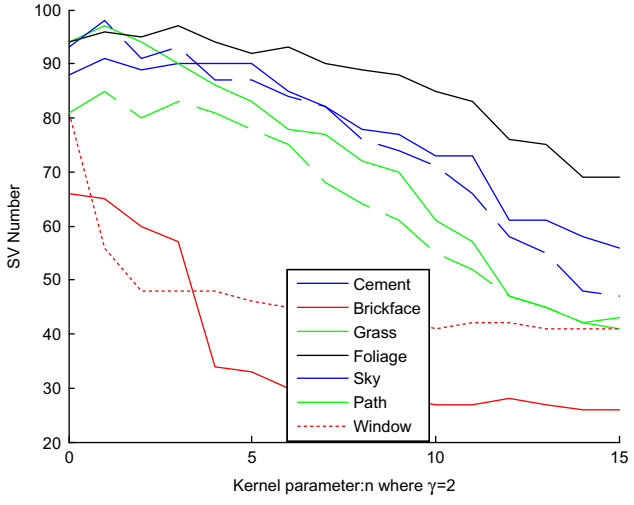

Fig. 18. Modified Chebyshev kernel parameter vs. SV number.

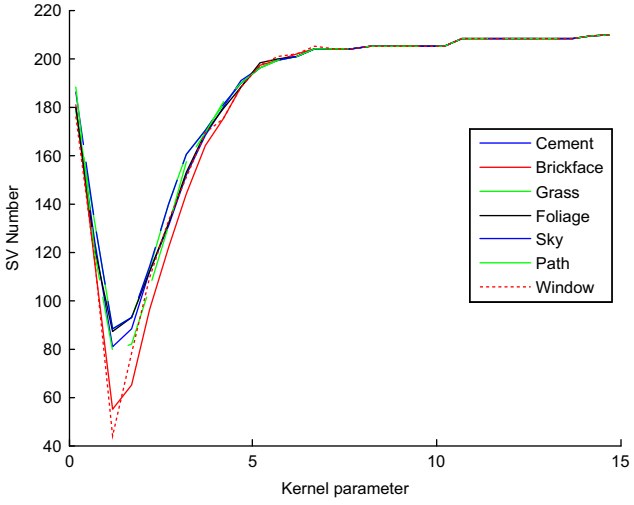

Fig. 19. Wavelet kernel parameter vs. SV number.

SVs as the kernel parameter increases, and they approach the maximum training sample numbers. In contrast, the generalized Chebyshev kernel yielded low number of SVs for all 3 classes.

Breast Cancer Wisconsin dataset: This is another dataset currently available at [17]. This dataset has 569 data where each data vector has 30 features. The data has only two classes: malignant and

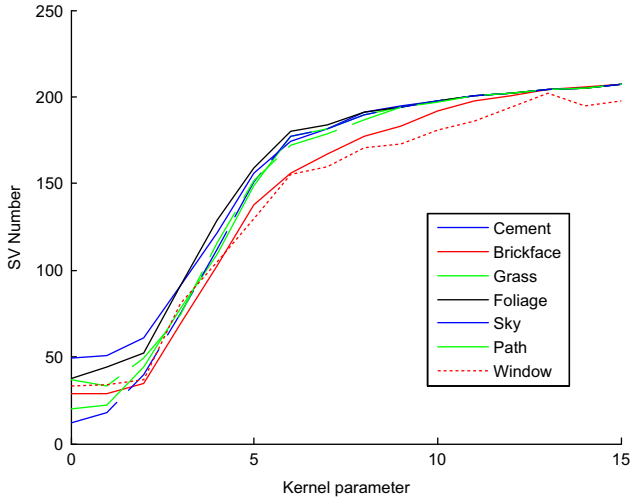

Fig. 20. Chebyshev kernel parameter vs. SV number.

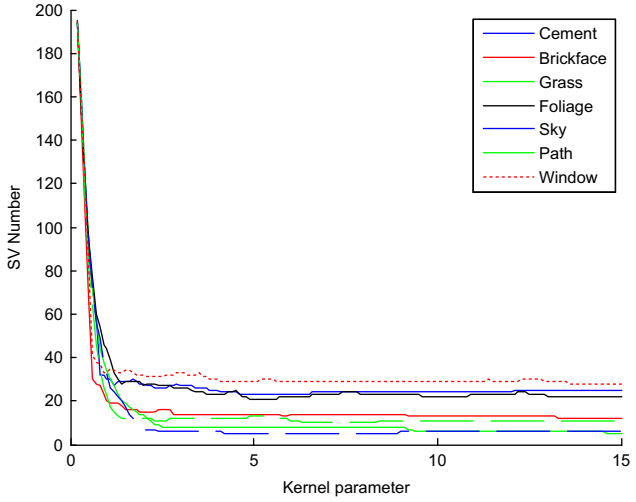

Fig. 21. Gaussian kernel parameter vs. SV number.

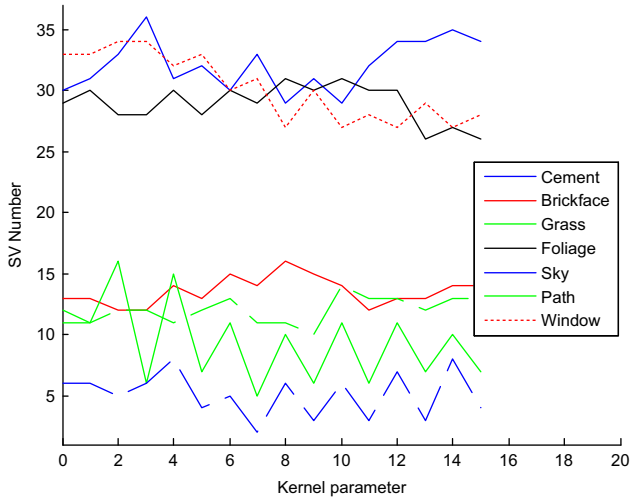

Fig. 22. Generalized Chebyshev kernel parameter vs. SV number.

benign. We used the first 50 data of each class for training and used the remaining 469 data for testing. The test results are shown in Table 4.

For the Wisconsin breast cancer dataset, the Generalized Chebyshev kernel yielded the best classification accuracy to separate the benign region from malignant region in the higher dimensional space. Fig. 27 shows the kernel parameter vs. test performance for each kernel function. Fig. 28 shows that the Wavelet and Chebyshev kernels require more training samples and use them all as SV as the kernel parameter increases and after a certain value they reach to the maximum training sample number.

10. Discussion on the experimental results

Test results show when the kernel function is in the form of Eq. (8), the generalization ability of resulting SVM may not be as desired (as shown in Figs. 3, 6 and 10) although the training step of SVM classification may yield zero error and the training data may be learnt with a lower number of support vectors. On average, both previously proposed Chebyshev and wavelet kernels decrease their performance by demanding more training data as the kernel parameter increases. We believe that this is a result of applying kernel functions onto each element and multiplying the outcomes for multidimensional data. Therefore, the constructed kernel functions should be applied onto each training vector directly instead of applying them onto each feature pair first.

A future study may investigate the relationship between the vector dimension and the kernel parameter. Intuitively, we think that there is a loose connection between the Chebyshev polynomial order and input vector dimension; however, as the polynomial order has to be an integer and the Chebyshev coefficients for the Chebyshev polynomials increase rapidly by the polynomial order, keeping the kernel parameter as low as possible would be desired. Therefore, the kernel parameter should be chosen among a few values, such as 3, 4, 5 or 6. In contrast, notice that although other kernel functions may find lower SV numbers for some certain cases, in general the *Generalized Chebyshev Kernel* yields a low number of support vectors in almost every test. Although this is directly related to the shape of the kernel function, this property may also be a result of the orthogonality feature of the Chebyshev polynomials as the generalized Chebyshev kernel function is constructed from an idea that is similar to the orthogonality property of the Chebyshev polynomials.

Generalized Chebyshev polynomials of the second type ($U(\mathbf{x})$), [14] can also be derived by using the same idea as shown for $T(\mathbf{x})$ in this study. Such functions (e.g. Eq. (30)) can also be used for kernel construction as mentioned in Section VII. However preliminary test results indicated that their results were similar to the kernel functions derived from the generalized Chebyshev polynomials of the first kind. Therefore we did not include that family of kernel functions and their results in this study. Also, since exploiting the properties of the generalized Chebyshev polynomials is out of the scope of this study, we did not study such properties in detail, yet such a study could be useful to construct new kernel functions derived from generalized Chebyshev polynomials and may be the subject of a future work.

11. Conclusions

In this study, the construction of a set of new SVM kernel, *Generalized Chebyshev Kernel* is presented. We clarify the ambiguity on how to use Chebyshev kernels on multidimensional data by proposing the use of *Generalized Chebyshev Polynomials* for vector inputs and show how to derive new kernels based on them. Then, we improve the average performance of *Generalized Chebyshev Kernel* by introducing the *Modified Chebyshev Kernel*. Modified Chebyshev kernel is another example for generating new kernel functions using

Table 3
Iris dataset test results with various kernel functions.

	Chebyshev Kernel		**Gen. Cheb.**		**Mod. Cheb.**		**Gaussian kernel**		**Polynomial kernel**		**Wavelet kernel**	
	SV no/Test%	Best n	SV no/Test%	Best n	SV no/Test%	Best n/γ	SV no/Test%	Best σ	SV no/Test%	Best n	SV no/Test%	Best a
Setosa	3/100	0	3/100	3	3/100	9/1	3/100	1.8	4/100	3	16/100	1.2
Virginica	7/98.10	0	9/96.19	3	11/99.05	3/1	24/97.14	0.35	8/98.10	3	17/98.10	0.8
Versicolour	11/99.05	2	9/96.19	0	18/99.05	2/2	19/98.10	0.43	19/98.10	3	17/98.10	0.8

Table 4
Breast Cancer Wisconsin dataset test results with various kernel functions and with the best kernel parameter results.

	Gaussian	Mod Chebyshev	Gen. Chebyshev	Wavelet	Polynomial	Chebyshev
Breast cancer test%	95.52	97.01	97.23	90.83	96.38	95.95
Best kernel parameter:	$\sigma=12$	$n=8$, $\gamma=0.25$	$n=3$	$a=1.4$	$n=9$	$n=0$
SV no:	30	30	30	72	30	30

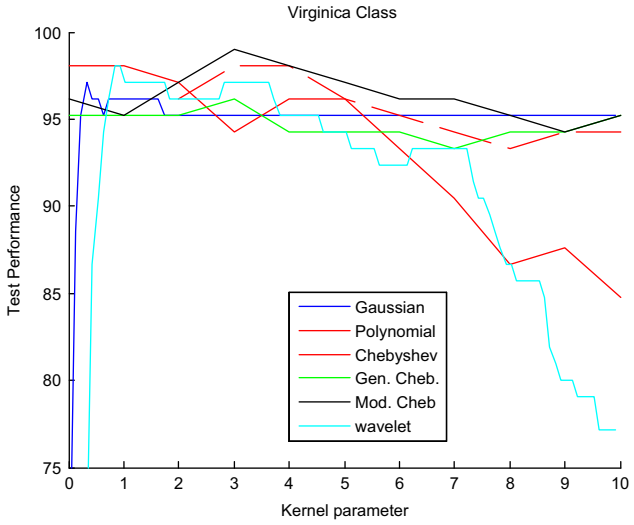

Fig. 23. Kernel parameter vs. test performance for Virginica Class.

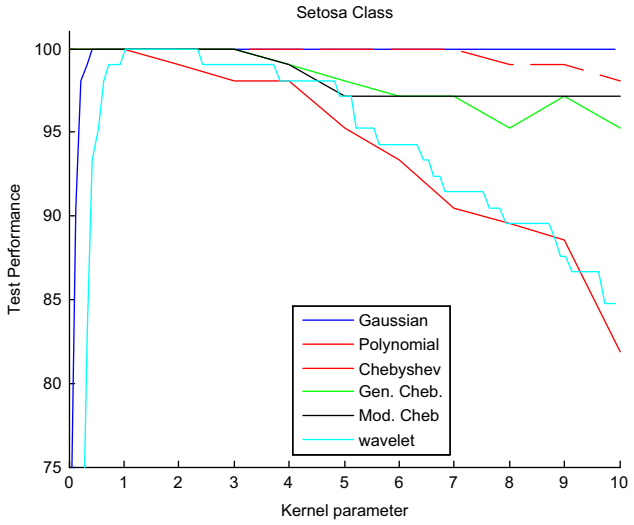

Fig. 25. Kernel parameter vs. test performance for Setosa Class.

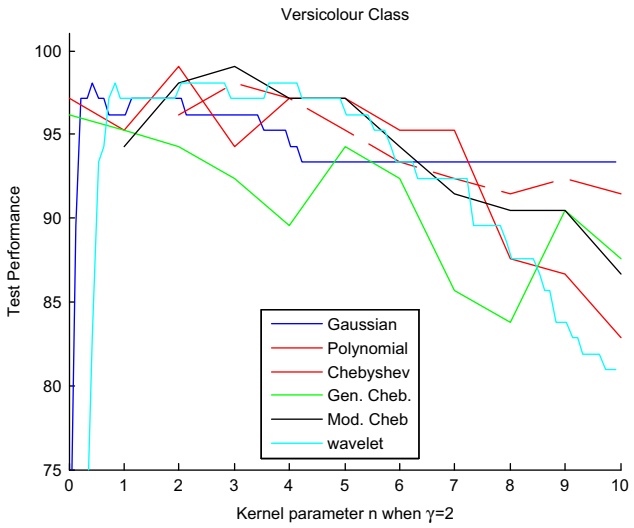

Fig. 24. Kernel parameter vs. test performance for Versicolour Class.

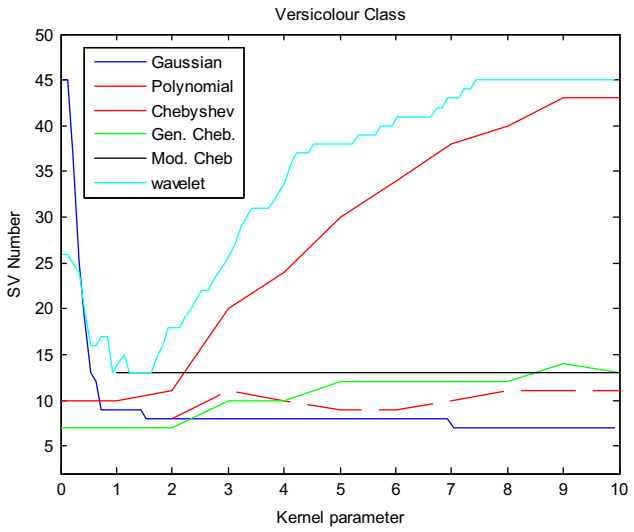

Fig. 26. Kernel parameter vs. SV number for Versicolour Class.

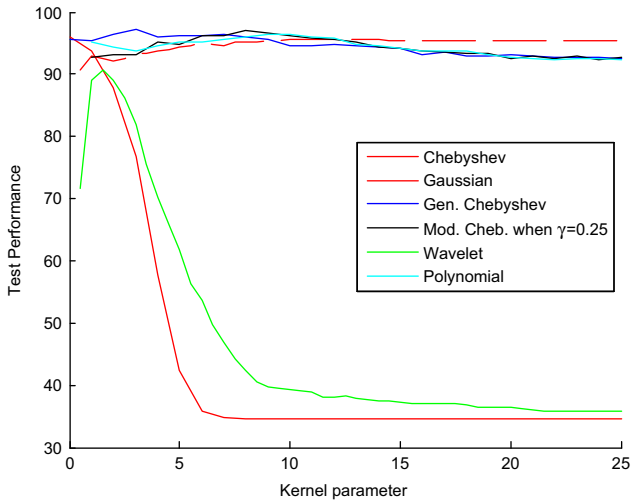

Fig. 27. Breast Cancer test data performance results.

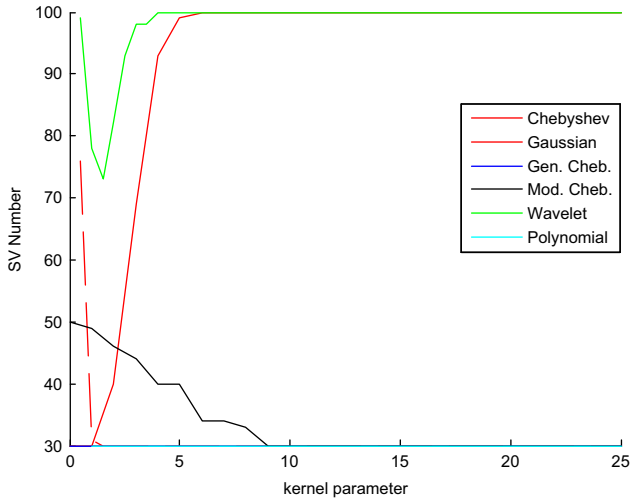

Fig. 28. Breast Cancer test data SV number results.

the generalized Chebyshev polynomials. The test results show that the modified Chebyshev kernel provides a competitive performance to those commonly being used parametric kernel functions and its performance can be better than other kernels with an appropriate γ value, as in Tables 2–4. Although, finding the optimal γ value for the modified Chebyshev kernel function seems similar to finding the optimal Gaussian kernel parameter σ, in the preliminary experiments we observed that the Modified Chebyshev kernel is less sensitive to the change in γ value. Therefore we used set of a few values (0.1, 0.25, 0.5, 1.0, 1.5, 2, 2.5, 3.0) for γ to find the best performance heuristically on the experiments.

Computationally, the generalized Chebyshev kernel function uses less multiplication when compared to the previously proposed Chebyshev kernel function. For a given pair of 2 input vectors, if we ignore the addition and subtractions, the generalized Chebyshev kernel needs about $n(2m+1)-m+r$ multiplications, whereas previously proposed Chebyshev kernel uses $n3m-1+rm$ multiplications, where r is the number of multiplications needed to calculate $1/\sqrt{(.)}$. Thus for $m>1$, the Generalized Chebyshev kernel function uses $(m-1)(n+r+1)$ multiplications less than the Chebyshev kernel for a given pair of two input vectors.

In this study, we propose a set of new kernel functions based on the generalized Chebyshev polynomials, and by experimental results we show that both proposed kernel functions can provide competitive classification results when compared to other common kernel functions. Figs. 11–16 show that the change in the performance value with respect to the kernel parameter is less than previously proposed Chebyshev kernel function on average for both the generalized and the modified Chebyshev kernels. Therefore based on the test results, we can also conclude that the proposed kernel functions are also more robust with respect to the kernel parameter change. This is another reason why we consider the proposed generalized Chebyshev kernel as a semi-parametric kernel function.

Finally, we can say that, while the generalized Chebyshev kernel approaches the minimum support vector number, the modified Chebyshev kernel approaches the maximum classification performance in tests. One of the probable reasons for high performance is that the modified Chebyshev kernel combines the well-known performance of the Gaussian kernel with the Chebyshev polynomials. Therefore, this set of new Chebyshev kernel functions can be used in classification applications as efficient alternatives to those commonly used Gaussian, Polynomial and wavelet kernel functions.

Acknowledgments

The authors would like to thank to Dr. Ye and Dr. Sun for providing the Chebyshev kernel MATLAB code that they used in [11] for comparison.

References

[1] V. Vapnik, Statistical Learning Theory, Wiley-Interscience, New York, 1998.
[2] Y. Artan, X. Huang, Combining multiple 2nu-SVM classifiers for tissue segmentation, IEEE ISBI 2008 (2008) 488–491.
[3] S. Ozer, D.L. Langer, X. Liu, M.A. Haider, T.H. van der Kwast, A.J. Evans, Y. Yang, M.N. Wernick, I.S. Yetik, Supervised and unsupervised methods for prostate cancer segmentation with multispectral MRI, J. Med. Phys. 37 (4) (2010).
[4] C.H. Chen, P.G.P. Ho, Statistical pattern recognition in remote sensing, J. Pattern Recognition 41 (9) (2008) 2731–2741.
[5] Z.L. Wu, C.H. Li, J.K.Y. Ng, K.R.H.P. Leung, Location estimation via support vector regression, IEEE Trans. Mobile Comput. 6 (3) (2007) 311–321.
[6] P.H. Chen, C.J. Lin, B. Schölkopf, A tutorial on nu-support vector machines, Appl. Stochast. Models Bus. Ind. (2005) 111–136.
[7] L. Zhang, W. Zhou, L. Jiao, Wavelet support vector machine, IEEE Trans. Systems, Man, and Cybernet.—Part B: Cybernet. 34 (1) (2004) 34–39.
[8] L.P. Bi, H. Huang, Z.Y. Zheng, H.T. Song, New heuristic for determination Gaussian kernels parameter, Mach. Learning Cybernet. 7 (2005) 4299–4304.
[9] H.J. Liu, Y.N. Wang, X.F. Lu, A method to choose kernel function and its parameters for support vector machines, Mach. Learning Cybernet. 7 (2005) 4277–4280.
[10] K.P. Wu, S.D. Wang, Choosing the kernel parameters of support vector machines according to the inter-cluster distance, in: Neural Networks, IJCNN '06, 2006, pp. 1205–1211.
[11] N. Ye, R. Sun, Y. Liu, L. Cao, Support vector machine with orthogonal Chebyshev kernel, in: Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06), 2006, 0-7695-2521-0/06.
[12] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2001.
[13] T.J. Rivlin, The Chebyshev Polynomials, John Wiley & Sons Press, 1974.
[14] S. Ozer, On the classification performance of support vector machines using Chebyshev kernel functions. M.Sc. Thesis, University of Massachusetts, Dartmouth, 2007.
[15] S. Canu, Y. Grandvalet, V. Guigue, A. Rakotomamonjy, SVM and Kernel Methods Matlab Toolbox. Perception Systèmes et Information, INSA de Rouen, 2005.
[16] C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining Knowledge Discovery 2 (1998) 1–43.
[17] A. Asuncion, D.J. Newman, UCI Machine Learning Repository", Irvine, CA: University of California, School of Information and Computer Science. 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉.

Sedat Ozer received his B.Sc. degree in Electronics Engineering from Istanbul University, Istanbul, Turkey and his M.S. degree in Electrical Engineering from University of Massachusetts Dartmouth in 2007. He is currently a Ph.D. student at the Visualization Lab at Rutgers University, NJ. He currently works on segmentation, tracking, event detection and visualization problems on scientific 3D datasets. His research interests are statistical learning algorithms including Bayesian methods, Support Vector Machines, kernel methods, signal/image processing and 3D tracking of time varying scientific data.

Chi Hau Chen received his Ph.D. in electrical engineering from Purdue University in 1965. He has been a faculty member with the University of Massachusetts Dartmouth (UMass Dartmouth) since1968 where he is now Chancellor Professor. Dr. Chen was the Associate Editor of IEEE Transactions on Acoustics, Speech and Signal Processing from 1982 to 1986, Associate Editor on information processing of IEEE Transactions on Geoscience and Remote Sensing 1985–2000. He is an IEEE Fellow (1988), Life Fellow (2003), and also a Fellow of International Association of Pattern Recognition (IAPR 1996). He has been an Associate Editor of International Journal of Pattern Recognition and Artificial Intelligence since 1985. In addition to the remote sensing, underwater acoustics and geophysical applications of statistical pattern recognition, he has been active with the signal and image processing of medical ultrasound images as well as industrial ultrasonic data for nondestructive evaluation of materials He has published 27 books in his areas of research interest.

Hakan Ali Cirpan received the B.S. degree in 1989 from Uludag University, Bursa, Turkey, the M.S. degree in 1992 from the University of Istanbul, Istanbul, Turkey, and the Ph.D. degree in 1997 from the Stevens Institute of Technology, Hoboken, NJ, USA, all in electrical engineering. From 1995 to 1997, he was a Research Assistant with the Stevens Institute of Technology, working on signal processing algorithms for wireless communication systems. In 1997, he joined the faculty of the Department of Electrical-Electronics Engineering at The University of Istanbul. In 2010 he has joined to the faculty of the department of Electronics & Communication Engineering at Istanbul Technical University. His general research interests cover wireless communications, statistical signal and array processing, system identification and estimation theory. His current research activities are focused on machine learning, signal processing and communication concepts with specific attention to channel estimation and equalization algorithms for space–time coding and multicarrier (OFDM) systems.