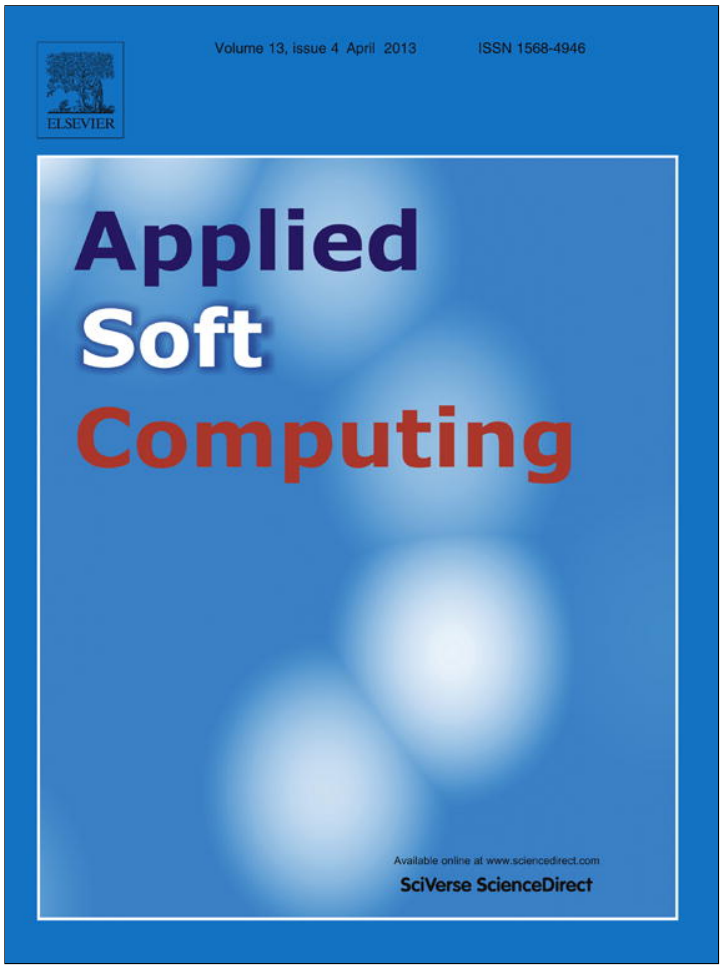

Applied Soft Computing 13 (2013) 1759–1765

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

New empirical nonparametric kernels for support vector machine classification

Essam Al Daoud*, Hamza Turabieh[1]

Computer Science Department, Faculty of Science and Information Technology, Zarqa University, Zarqa, Jordan

ARTICLE INFO

Article history:
Received 14 April 2012
Received in revised form
30 November 2012
Accepted 10 January 2013
Available online 31 January 2013

Keywords:
Kernel
SVM
Positive semidefinite matrix
Classification
Inner product

ABSTRACT

Despite the excellent applicability of kernel methods, there seems to be no systematic way of choosing appropriate kernel functions or the optimum parameters. Therefore, the performance of support vector machines (SVMs) cannot be easily optimized. To address this problem, a general procedure is suggested to produce nonparametric and efficient kernels. This is achieved by finding an empirical and theoretical connection between positive semidefinite matrices and certain metric space properties. The Gaussian kernel turns out to be a special case of the new framework. Comprehensive experiments on eleven real-world datasets and seven synthetic datasets demonstrate a clear advantage in favor of the proposed kernels. However, several important problems remain unresolved.

1. Introduction

Kernel methods are very important statistical learning tools and have recently become very popular. Kernels can be used in several learning scenarios due to their performance, accuracy, sound theoretical foundations, and ability to deal with high-dimensional datasets. Kernels can be used for clustering, density estimation, regression, and classification. Various kernel algorithms have been introduced in the literature to accomplish different tasks, such as kernel principal component analysis, support vector machines, Gaussian processes, Bayes point machines, and kernel Gram–Schmidt processes. A kernel function maps the input space $X \subset R^d$ into a feature space F, such that $\phi(x) = f$, where $x \in X$ and $f \in F \subset R$. Thus, nonlinear input in the original space can be manipulated linearly in the new space, and the kernel can be calculated as follows [1]:

$$k(x, y) = \langle \phi(x), \phi(y) \rangle \tag{1}$$

where x and y are two vectors in X, and $\langle .,. \rangle$ is an inner product. The generated matrix $k(x_i, x_j)$ is called a Gram matrix or a kernel matrix. Fortunately, we do not need to compute the inner product explicitly; the input space can be implicitly mapped to the inner product space (feature space) by simply using symmetric and positive semidefinite matrices. The matrix K is a valid Gram matrix if and only if it is positive semidefinite.

To investigate the efficiency of the new kernels, we focus on applying the kernel matrices to binary classification using support vector machines (SVMs). Let $x_i \in R^m$, $i = 1, 2, \ldots, n$ be a set of training vectors where the vector i is labeled by $y_i \in \{1, -1\}$. The nonlinear SVM optimization problem can be described as follows:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ & 0 \le \alpha_i \le C \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{2}$$

where k is a suitable kernel and the Lagrange multipliers α_i represent the relative importance of vector x_i. After solving the above problem, we can classify any test vector x by using

$$sign\left(\sum_{i=1}^{m} \alpha_i y_i K(x_i, x) + b \right) \tag{3}$$

where m is the number of support vectors, and b can be calculated by solving the equation $y_j(\sum_{i=1}^{m} (\alpha_i y_i K(x_i, x) + b) = 1$ for any x_j with $0 < \alpha_j < C$ [2,3].

A potential disadvantage of using an SVM is that there is no efficient way to find a perfect kernel and its parameters for a given application. This limitation reduces the effectiveness of kernel methods, especially since different functions and parameters

* Corresponding author. Tel.: +962 796680005.
E-mail addresses: essamdz@zu.edu.jo, essamauto@yahoo.com (E.A. Daoud), turabieh@zu.edu.jo (H. Turabieh).
[1] Tel.: +962 779407043.

1568-4946/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2013.01.010

exhibit vastly different performances. To overcome this restriction, we introduce a new framework for designing efficient kernels. The key contribution of this paper is two-fold. First, we propose a new procedure for generating an unlimited number of practical kernels. Second, the suggested kernels can be applied without parameters.

2. Related work

In the literature, the most well-known kernels are Gaussian, polynomial, and linear. However, many new kernels have been proposed, which can be divided into three approaches: combinations of existing kernels, application-specific kernels and general kernels [1].

Combinations of existing kernels or multiple kernel learning (MKL) can be used to create new kernels by using a set of old kernels. The general formula of MKL is [4]

$$k_\lambda(x_i, x_j) = f_\lambda(\{k_m(x_i^m, x_j^m)\}_{m=1}^p) \quad (4)$$

where p is the number of kernels, and where each vector x^m may have special feature representations and can be extracted from different sources. The kernels k_m may correspond to different similarity measures, and λ is the kernels' weights. The function f_λ can be linear, nonlinear, or a data-dependent combination [5,6]. Gonen and Alpaydın showed that using a data-dependent combination or nonlinear combination is more promising than using a linear combination [4]. To optimize the kernels' weights, different learning methods have been used, such as fixed-rules approaches [7], heuristic approaches [8], Bayesian approaches [9], boosting approaches [10], approaches integrated to a kernel-based learner, and genetic algorithm-incorporating approaches [2,11]. The main benefit of MKL is that it can be used for automatically determining the best kernels or combining the most useful ones. However, MKL needs more time for training and testing.

Many successful kernels have been implemented and designed for specific applications; for example, Leslie and Kuang used fast kernels such as mismatch kernels, substitution kernels, wildcard kernels, and gappy kernels to compare protein sequences [12]. Another popular application is text categorization, where a sparse vector kernel is proposed to use a bag-of-words representation [13,14]. Decoste and Schölkopf introduced the pyramidal kernel to consider the local interactions between image patches [15], while Collins and Duffy suggested an efficient tree kernel to be used with structured objects [16]. Although application-specific kernels are very suitable for specific domains, such kernels cannot be used for general purposes.

The third approach is the use of general kernels. Gentom listed several isotropic stationary kernels for general purposes, such as exponential kernels, rational quadratic kernels, wave kernels, circular kernels, and spherical kernels. All are positive semidefinite in R^d except the last two kernels, which are valid in R^2 and R^3, respectively. Zhang et al. suggested a multidimensional wavelet kernel and showed that their wavelet kernel outperforms the Gaussian kernel for some artificial datasets generated by using single-variable and two-variable functions [17]. Basak used a long-tailed kernel function based on the Cauchy distribution and observed that the Cauchy kernel is useful with constrained least square kernel machines for some datasets [18]. Lin and Lin illustrated that the sigmoid (hyperbolic tangent) kernel is a valid kernel when used with certain parameters. However, the researchers explained that the radial basis function (RBF) kernel is better than the sigmoid kernel [19].

The suggested kernels fall in the general kernels class. However, the new kernels enrich the MKL and application-specific kernels by giving researchers more choices. Moreover, the new kernels perform very well with most of the tested datasets without any need to tune the parameters.

3. Proposed method

Some noteworthy properties of Gaussian kernels are abstracted and introduced in Definition 1. Using these, we can generalize the Gaussian matrix. Moreover, we argue that these properties are a sufficient condition for a matrix to be positive semidefinite. At the end of this section, we introduce a general procedure for generating nonparametric and empirical kernels.

Definition 1. Let A be a matrix of size $n \times n$ that has the following properties:

1. All the diagonal elements are 1.
2. The non-diagonal elements are less than 1 and greater than or equal to zero.
3. If a_{ij}, a_{ik} and a_{jk} are any three non-diagonal elements, then

$$a_{ij} + a_{ik} - 1 \le a_{jk}$$
$$a_{ij} + a_{jk} - 1 \le a_{ik}$$
$$a_{jk} + a_{ik} - 1 \le a_{ij}$$

Proposition 1. *Let **A** be a matrix of size* 3×3 *satisfying the properties given in Definition 1; then, **A** is positive semidefinite.*

$$\mathbf{A} = \begin{bmatrix} 1 & a & b \\ a & 1 & c \\ b & c & 1 \end{bmatrix}$$

Proof. The determinant of the upper left 2×2 corner of A is $1 - a^2$; since $a \le 1$, this implies $1 - a^2 \ge 0$. The determinant of the upper left 3×3 corner of **A** *is* $z(a, b, c) = 1 - a^2 - b^2 - c^2 + 2abc$. Considering the constraints in Definition 1, we see that the minimum value of z is 0. Therefore, according to Sylvester's criterion, **A** is positive semidefinite.

In general, $n \times n$ matrices that have the properties in Definition 1 seem to be positive semidefinite. Although we could not prove this claim analytically, thousands of matrices generated randomly and with different sizes were tested, and all the experiments confirmed this claim.

Definition 2. Let $a + b \ge c$, $\forall a$, b and c in [0,1]. Then, the function f: $[0,1] \to [0,1]$ preserves triangle inequality if $f(a) + f(b) \ge f(c)$ and $f(0) = 0$.

Example. The function $f(x) = x - (x^2/2)$ preserves triangle inequality.

We have to show that $Z(a,b,c) = f(a) + f(b) - f(c) \ge 0$, which is equivalent to the quadratic programming problem

$$\begin{array}{ll} \text{Minimize} & Z(a, b, c) = a - \dfrac{a^2}{2} + b - \dfrac{b^2}{2} - c + \dfrac{c^2}{2} \\ \text{where} & a + b \ge c \\ \text{and} & a, b \text{ and } c \in [0, 1) \end{array}.$$

Because the minimum value of the above problem is 0, $f(a) + f(b) \ge f(c)$ and $f(x)$ is preserve triangle inequality.

Many triangle inequality-preserving functions can be found easily, e.g., $f(x) = x$, $f(x) = \sin(\pi x/2)$, $f(x) = 1 - \sqrt{1 - x^2}$, $f(x) = \cos((1 - x)\pi/2)$, $f(x) = \tan(\pi x/4)$, and so on.

Proposition 2. *Let M be a metric space and d be a metric on M. Let f be a triangle inequality-preserving function, and let matrix A be defined as follows*:

$$a_{ij} = 1 - f\left(\frac{d(v_i, v_j)}{z}\right) \quad i, j = 1, 2, \ldots, n \tag{5}$$

where n is the length of M, and $z = max\{d(v_i, v_j): \forall i,j = 1,2,\ldots,n\}$. The matrix $\boldsymbol{A}$ satisfies the properties in Definition 1.

Proof *(:).*

1. The diagonal elements $a_{ii} = 1 - f(d(v_i, v_i)/z) = 1 - f(0) = 1$
2. Since $x = d(v_i, v_j)/z \in [0,1)$, it follows that $f(x) \in [0,1)$ and the non-diagonal elements $a_{ij} = 1 - f(x) \in [0, 1)$.
3. $3 - a_{ij} + a_{ik} = 1 - f(d(v_i, v_j)/z) + 1 - f(d(v_i, v_k)/z) = 2 - (f(d(v_i, v_j)/z) + f(d(v_i, v_j)/z)) \leq 2 - (f(d(v_j, v_k)/z)) = 2 - (1 - a_{jk}) = 1 + a_{jk}$

Thus $a_{ij} + a_{ik} \leq 1 + a_{jk}$, which implies that $a_{ij} + a_{ik} - 1 \leq a_{jk}$. Similarly, we can prove the second and third inequalities.

Through use of the above propositions, many new non-linear kernels can be suggested. For example, the following kernels were tested empirically and applied to many datasets:

$$k(x, y) = 1 - \sin\left(\frac{\pi d(x, y)}{2z}\right)$$

$$k(x, y) = 1 - \frac{\log(d(x, y)/z + 1)}{\log(2)}$$

$$k(x, y) = 1 - \sum_{i=1}^{m} \frac{1 - i}{i(i-1)!}\left(\frac{-d(x, y)}{z}\right)^i$$

$$k(x, y) = 1 - \left(1 + \sum_{i=1}^{m} \frac{(-1)^i}{((d(x, y)/z)^i + 1)}\right)$$

$$k(x, y) = 1 - \sum_{i=1}^{m} \frac{(-d(x, y)/z)^i}{2^{i-1}}$$

$$k(x, y) = 1 - \sum_{i=1}^{m} \frac{(-d(x, y)/z)^i}{i}$$

$$k(x, y) = 1 - \sum_{i=1}^{\infty} \frac{1}{i!}\left(\frac{-(d(x, y))^2}{z}\right)^i \text{ (Gaussian Kernel)}$$

$$k(x, y) = 1 - \sum_{i=1}^{m} a_i\left(\frac{-d(x, y)}{z}\right)^i \text{ for some decreasing sequences } \{a_i\}_{i=1}^{m} \text{ and } 0 < a_i < 1$$

where $z = max\{d(v_i, v_j): \forall i,j = 1,2,\ldots,n\}$. It is clear that if a smaller value than the suggested z is used, the second property of Definition 1 will not be satisfied, and if a larger value is used, the diagonal of the matrix will dominate, so the accuracy will be reduced. Furthermore, several Euclidean and non-Euclidean distances can be exploited, such as the 2-norm distance, Manhattan distance, Chebyshev distance, Helly metric, tight span, British rail metric, and Hausdorff distance. The general proof that a function preserves triangle inequality is equivalent to a well-known nonlinear programming problem that cannot be solved analytically. However, if we show empirically that function f preserves triangle inequality for all of the tested examples, then f can be used efficiently with many applications. Moreover, we note the following empirical findings:

- All the tested increasing functions in the domain [0,1] preserve triangle inequality after shifting and scaling.
- If a function preserves triangle inequality within one dataset (random or real), then the function apparently preserves triangle inequality with all other datasets.
- In the literature, researchers consider a new kernel that outperforms the Gaussian kernel in a variety of applications to be very useful, so the new techniques offer a rich resource for efficient kernels.

Procedure 1 summarizes the important steps for generating new kernels.

Procedure 1 *(:).* A new kernel

Output: a new kernel

1. Design a triangle inequality-preserving function $f(x)$ as described in Definition 2.
2. Find a metric distance $d(x,y)$.
3. Return $k(x,y) = 1 - f(d(x,y)/z)$.

4. Datasets

Eleven real-world classification problems, one spiral dataset, and six artificial datasets were used. All the datasets have several features and one binary target. The basic information of the real-world classification datasets was downloaded from the UCI Repository and is summarized in Table 1 [20]. Some preprocessing operations were performed on the datasets, namely, normalizing and deleting missing data. A fruit fly (*Drosophila melanogaster*) protein–protein interaction dataset and an Arcene dataset were collected from http://nipsfsc.ecs.soton.ac.uk/ and http://bioinformatics.org.au, respectively. Fig. 1 shows the spiral dataset, which consists of two intertwined classes, exhibits complex shapes, and includes 97 points from each class [24].

To construct the artificial datasets, random numbers in the interval [0,1] were generated and substituted in the following formulas:

$$1 - T = \begin{cases} 0 & \sin(5x)^2 y < 0.1 \\ 1 & else \end{cases}$$

$$2 - T = \begin{cases} 0 & \sin(5x)\cos(6y) < 0.2 \\ 1 & else \end{cases}$$

Table 1
Basic information of the real-world datasets.

No.	Dataset	#Patterns	#Features	#Classes
1	Breast cancer (BC)	569	10	2
2	Heart (H)	270	13	2
3	Credit1 (C1)	690	15	2
4	German (G)	1000	24	2
5	Hepatitis (HP)	155	19	2
6	Horse (HR)	368	22	2
7	Ionosphere (I)	351	34	2
8	Oneh (O)	2536	72	2
9	Sonar (S)	208	60	2
10	Fruit-fly	20,000	7	2
11	Arcene	200	10,000	2

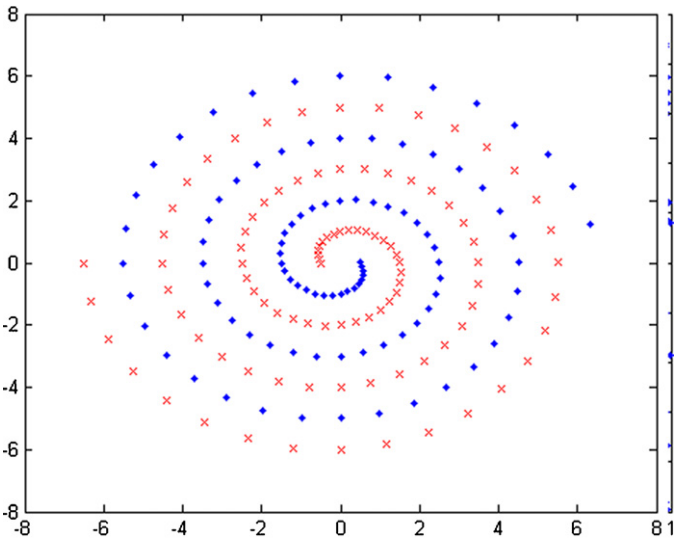

Fig. 1. Spiral dataset including 97 points from each class.

$$3-T=\begin{cases}0 & \sin(5x)^2\tan(6y)^2<0.8\\ 1 & else\end{cases}$$

$$4-T=\begin{cases}0 & \sum_{i=1}^{3}\sin(5x_i)^2<0.05\\ 1 & else\end{cases}$$

$$5-T=\begin{cases}0 & \sin(5x_1)\tan(6x_2)x_3<2\\ 1 & else\end{cases}$$

$$6-T=\begin{cases}0 & \sum_{i=1}^{5}x_i^2<0.0001\\ 1 & else\end{cases}$$

Some random noises were added to the artificial datasets, Figs. 2 and 3 represent the third and fourth artificial datasets

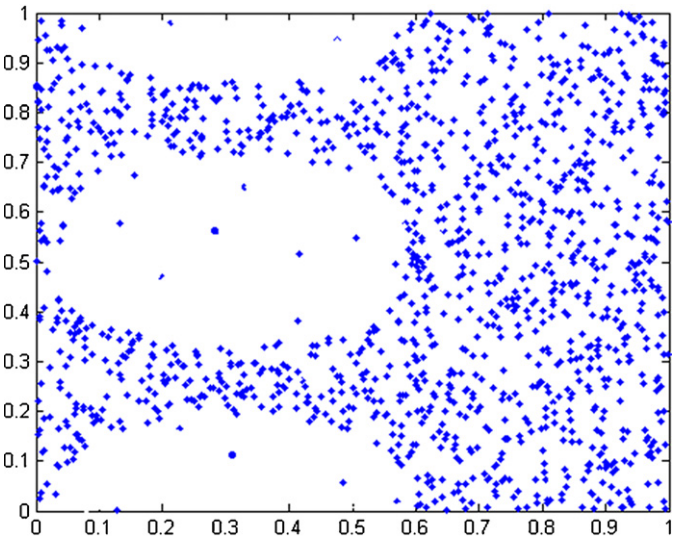

Fig. 2. Third artificial dataset with noise.

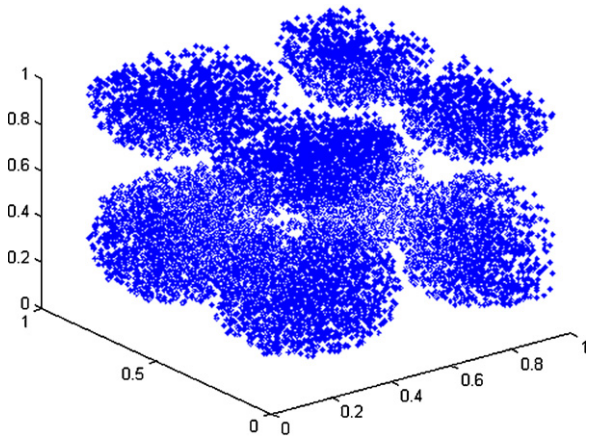

Fig. 3. Fourth artificial dataset with noise.

Table 2
Basic information of the artificial datasets.

Dataset	#Patterns	#Features	1st class	2nd class
1	2000	2	931	1069
2	2000	2	661	1339
3	2000	2	810	1190
4	3000	3	1394	1606
5	3000	3	1415	1585
6	3000	5	1490	1510
7	194	2	97	97

respectively. (see Table 2) summarizes the basic information of the artificial datasets (1-6) and the spiral dataset (7).

5. Experimental results

To test the performance of the suggested approach, four new kernels were generated by using the techniques in Section 3 and then compared to nine kernels that have been adopted in the literature over the last two decades. The old and new kernels are listed in Tables 3 and 4, respectively.

Tables 3 and 4 show that most of the efficient kernels in the literature need to have parameters such as α and d that need more time to be tuned. However, the new kernels can be applied efficiently without using parameters. To tune the parameters (α, d, C), cross validation and "grid-search" were performed. Various pairs of (α, C) or (d, C) were tested and the best accuracy was selected. We applied 5-fold cross validation on the sets $\alpha \in S_1 = \{0.01, 0.1,$

Table 3
Kernels from the literature.

Name	Function
Linear	$k_1(x,y)=x'y$ [21]
Polynomial	$k_2(x,y)=(x'y+1)^d$ [21]
Exponential	$k_3(x,y)=\exp\left(-\frac{\|x-y\|}{2\alpha^2}\right)$ [21]
Gaussian	$k_4(x,y)=\exp\left(-\frac{\|x-y\|^2}{2\alpha^2}\right)$ [21]
Sigmoid	$k_5(x,y)=\tanh(\alpha x'y)$ [19]
Wavelet	$k_6(x,y)=\prod_{i=1}^{n}h\left(\frac{x_i-y_i}{\alpha}\right)$ [17] where $h(x)=\cos(1.75x)\exp(-x^2/2)$
Inverse multiquadric	$k_7(x,y)=\frac{1}{\sqrt{\|x-y\|^2+\alpha}}$ [22]
Spline	$k_8(x,y)=\prod_{i=1}^{d}1+x_iy_i+x_iy_im_i-\frac{x_i+y_i}{2}m_i^2+\frac{m_i^3}{3}$, where $m_i=\min(x_i,y_i)$ and $x,y\in R^d$ [23]
Cauchy	$k_9(x,y)=\frac{1}{1+\|x-y\|^2/\alpha}$ [18]

Table 4
New nonparametric kernels.

Name	Function
k_{10}	$k_{10}(x,y) = 1 - \sum_{i=1}^{3} \frac{(-\|\|x-y\|\|/z)^i}{2^{i-1}}$
k_{11}	$k_{11}(x,y) = 1 - \sum_{i=1}^{3} \frac{(-\|\|x-y\|\|/z)^i}{i}$
k_{12}	$k_{12}(x,y) = 1 - \left(1 + \sum_{i=1}^{3} \frac{(-1)^i}{(\|\|x-y\|\|/z)^i+1}\right)$
k_{13}	$k_{13}(x,y) = 1 - \sin(\pi\|\|x-y\|\|/2z)$

0.3, 0.5, 1, 2.5 10}, $d \in S_2 = \{2, 3, 4, 5, 6, 7, 8\}$, and $C \in S_3 = \{0.05, 0.1, 3, 10, 20\}$. Therefore, the datasets were divided randomly into five partitions, and the results were averaged for each point in the previous grid. The classification rate for each kernel and each dataset was calculated by repeating the procedure 175 times (number of partitions × $|S_3|$ × $|S_1|$ or $|S_2|$) for kernels {2, 3, 4, 5, 6, 7, 9} and 25 times (number of partitions × $|S_3|$) for kernels {1, 8, 10, 11, 12, 13}. To produce the data in Tables 5 and 6, the procedure was repeated (18(1225 + 150) = 24,750) times. The best parameters corresponding to Tables 5 and 6 are given in Tables 7 and 8, respectively.

Several issues can be noted in Tables 5 and 6. First, the suggested kernels are comparable to the classical kernels and in some cases outperform them. For example, the best classification rate

Table 5
Misclassification rate of real-world datasets using nine old kernels and four new kernels.

	Can.	Heart	Cred.	Germ.	Hepa.	Horse	Iono.	Oneh	Son.	Fly	Arc.	Avg.
Linr.	7.02	20.37	14.32	23	12.9	14.86	9.14	**4.56**	19.67	37.11	21.44	16.76
Poly.	**4.51**	20.37	14.42	23.5	16.13	17.57	**5.43**	7.94	14.9	30.12	20	15.9
Expl.	5.26	25.93	13.7	21	**9.68**	**14.62**	**5.43**	**4.56**	**12.52**	13.78	**12.01**	12.59
Gaus.	5.26	19.22	14.42	23.5	**9.68**	14.86	**5.43**	**4.56**	**12.52**	13.9	12.61	12.36
Sigd.	7.02	20.37	13.7	22	12.9	14.86	9.14	**4.56**	19.67	18.06	21.44	14.88
Wavt.	6.14	22.22	15.86	24.5	**9.68**	16.22	6.86	**4.56**	**12.52**	18.23	16.87	13.97
IMult.	5.26	25.93	**12.25**	23.5	**9.68**	16.22	**5.43**	**4.56**	22.05	23.98	18.05	15.17
Spln.	7.02	31.48	31.88	42	29.03	24.32	13.43	10.68	26.19	35.55	28.38	25.45
Cauy.	5.26	25.93	15.87	26	**9.68**	16.22	9.71	**4.56**	21.43	32.1	21.44	17.11
K_{10}	6.14	17.9	13.7	21	**9.68**	17.57	**5.43**	**4.56**	**12.52**	**13.31**	14.52	12.39
K_{11}	6.14	17.28	13.7	**20.5**	**9.68**	17.57	**5.43**	**4.56**	**12.52**	**13.31**	12.8	**12.14**
K_{12}	6.14	**16.67**	14.42	22.5	**9.68**	17.57	**5.43**	**4.56**	**12.52**	14.02	14.11	12.51
K_{13}	5.26	22.22	14.42	22.5	**9.68**	16.21	**5.43**	**4.56**	**12.52**	15.26	13.91	12.91

Bold underline values are the best classification rate.

Table 6
Misclassification rate of artificial datasets using nine old kernels and four new kernels.

	1	2	3	4	5	6	Spiral	Average
Linr.	25.75	28.00	38.75	42.83	49.50	12.00	51.28	35.44
Poly.	1.50	2.70	3.25	9.00	**4.30**	3.16	33.85	8.25
Expl.	1.50	**1.00**	1.50	6.67	6.17	3.00	25.64	6.50
Gaus.	2.00	1.50	1.50	9.00	7.33	3.17	**17.02**	5.93
Sigd.	25.75	10.00	33.50	25.83	36.33	11.83	41.03	26.32
Wavt.	2.50	3.25	3.75	13.16	7.33	2.67	46.15	11.26
IMult.	1.50	2.00	2.50	7.17	6.33	3.33	46.15	9.85
Spln.	24.75	30.75	38.75	42.50	46.83	12.50	35.90	33.14
Cauy.	**1.00**	2.25	2.00	5.67	6.00	3.67	35.90	8.07
K_{10}	1.25	1.75	1.50	6.50	6.17	**2.67**	35.90	7.96
K_{11}	1.25	1.75	1.75	5.83	6.17	3.00	35.90	7.95
K_{12}	1.50	1.75	1.75	6.67	6.00	**2.67**	35.90	8.03
K_{13}	1.50	1.25	**1.00**	**5.50**	6.00	3.00	23.08	**5.90**

Bold underline values are the best classification rate.

Table 7
Best parameters used with the real-world datasets.

	Can.	Heart	Cred.	Germ.	Hepa.	Horse	Iono.	Oneh	Son.	Fly	Arc.
Linr.	$C=0.05$	$C=0.05$	$C=3$	$C=0.05$	$C=0.05$	$C=3$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.1$	$C=0.1$
Poly.	$C=20$	$C=0.05$	$C=0.1$	$C=3$	$C=0.05$	$C=0.1$	$C=0.1$	$C=0.05$	$C=3$	$C=0.1$	$C=3$
	$d=2$	$d=2$	$d=2$	$d=8$	$d=2$	$d=2$	$d=2$	$d=2$	$d=3$	$d=2$	$d=2$
Expl.	$C=20$	$C=3$	$C=0.01$	$C=20$	$C=0.05$	$C=3$	$C=10$	$C=0.05$	$C=10$	$C=3$	$C=10$
	$\sigma=1$	$\sigma=0.1$	$\sigma=1$	$\sigma=0.3$	$\sigma=0.01$	$\sigma=0.1$	$\sigma=0.3$	$\sigma=0.01$	$\sigma=0.3$	$\sigma=1$	$\sigma=0.3$
Gaus.	$C=10$	$C=3$	$C=0.05$	$C=10$	$C=0.05$	$C=0.1$	$C=10$	$C=0.05$	$C=10$	$C=0.1$	$C=10$
	$\sigma=1$	$\sigma=0.1$	$\sigma=0.01$	$\sigma=1$	$\sigma=0.01$	$\sigma=0.1$	$\sigma=0.3$	$\sigma=0.01$	$\sigma=0.3$	$\sigma=1$	$\sigma=1$
Sigd.	$C=3$	$C=10$	$C=20$	$C=0.05$	$C=0.05$	$C=20$	$C=3$	$C=0.05$	$C=10$	$C=3$	$C=0.1$
	$\sigma=1$	$\sigma=0.01$	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=0.01$	$\sigma=1$	$\sigma=1$	$\sigma=0.01$	$\sigma=0.3$	$\sigma=1$
Wavt.	$C=10$	$C=3$	$C=3$	$C=3$	$C=3$	$C=3$	$C=3$	$C=10$	$C=10$	$C=3$	$C=3$
	$\sigma=2.5$	$\sigma=10$	$\sigma=2.5$	$\sigma=1$	$\sigma=10$	$\sigma=2.5$	$\sigma=1$	$\sigma=10$	$\sigma=1$	$\sigma=2.5$	$\sigma=1$
IMult.	$C=20$	$C=3$	$C=0.05$	$C=3$	$C=0.05$	$C=3$	$C=3$	$C=0.05$	$C=3$	$C=3$	$C=3$
	$\sigma=0.3$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.3$	$\sigma=0.5$	$\sigma=0.3$	$\sigma=0.3$
Spln.	$C=0.1$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.05$	$C=0.1$	$C=0.1$
Cauy.	$C=20$	$C=3$	$C=3$	$C=3$	$C=0.05$	$C=3$	$C=3$	$C=0.05$	$C=3$	$C=0.1$	$C=0.1$
	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$
K_{10}	$C=10$	$C=3$	$C=10$	$C=10$	$C=0.05$	$C=3$	$C=10$	$C=0.05$	$C=10$	$C=3$	$C=3$
K_{11}	$C=10$	$C=0.1$	$C=10$	$C=10$	$C=0.05$	$C=3$	$C=10$	$C=0.05$	$C=10$	$C=3$	$C=10$
K_{12}	$C=10$	$C=3$	$C=0.1$	$C=10$	$C=0.05$	$C=0.1$	$C=10$	$C=0.05$	$C=10$	$C=3$	$C=10$
K_{13}	$C=20$	$C=0.1$	$C=0.05$	$C=3$	$C=0.05$	$C=0.1$	$C=20$	$C=0.05$	$C=3$	$C=3$	$C=3$

Table 8
Best parameters used with the artificial datasets.

	1	2	3	4	5	6	Spiral
Linr.	$C=3$	$C=20$	$C=0.1$	$C=0.05$	$C=0.05$	$C=0.1$	$C=0.05$
Poly.	$C=3$	$C=3$	$C=3$	$C=20$	$C=3$	$C=10$	$C=0.1$
	$d=8$	$d=8$	$d=8$	$d=8$	$d=7$	$d=3$	$d=8$
Expl.	$C=20$	$C=10$	$C=10$	$C=10$	$C=20$	$C=10$	$C=20$
	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=0.5$	$\sigma=1$
Gaus.	C=10	$C=10$	$C=20$	$C=3$	$C=3$	$C=3$	$C=3$
	$\sigma=10$	$\sigma=10$	$\sigma=1$	$\sigma=10$	$\sigma=10$	$\sigma=1$	$\sigma=10$
Sigd.	$C=20$	$C=20$	$C=10$	$C=20$	$C=10$	$C=3$	$C=20$
	$\sigma=0.01$	$\sigma=0.01$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$	$\sigma=0.5$
Wavt.	$C=20$	$C=20$	$C=3$	$C=20$	$C=10$	$C=3$	$C=3$
	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=1$	$\sigma=2.5$	$\sigma=1$	$\sigma=1$
IMult.	$C=20$	$C=20$	$C=20$	$C=20$	$C=20$	$C=3$	$C=3$
	$\sigma=0.01$	$\sigma=1$	$\sigma=0.01$	$\sigma=1$	$\sigma=0.01$	$\sigma=0.01$	$\sigma=0.01$
Spln.	$C=0.05$	$C=0.05$	$C=20$	$C=0.05$	$C=3$	$C=20$	$C=20$
Cauy.	$C=10$	$C=20$	$C=3$	$C=20$	$C=20$	$C=10$	$C=20$
	$\sigma=10$	$\sigma=10$	$\sigma=0.01$	$\sigma=10$	$\sigma=0.01$	$\sigma=0.3$	$\sigma=0.01$
K_{10}	$C=10$	$C=20$	$C=3$	$C=20$	$C=10$	$C=20$	$C=10$
K_{11}	$C=20$	$C=20$	$C=10$	$C=20$	$C=10$	$C=10$	$C=10$
K_{12}	$C=20$	$C=20$	$C=3$	$C=20$	$C=10$	$C=10$	$C=10$
K_{13}	$C=10$	$C=20$	$C=3$	$C=20$	$C=20$	$C=3$	$C=20$

Table 9
Training and testing time in seconds.

	Germ.		Oneh.		Fly		Arc		Artificial set 5	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Linr	6.53	**1.88**	72.33	20.30	2013.27	111.55	157.95	75.86	86.30	16.97
Poly.	61.14	2.27	625.73	17.30	35,815.64	180.06	1344.18	66.41	868.44	18.02
Expl	39.48	1.89	476.77	16.61	23856.21	118.16	1038.18	54.51	662.38	16.09
Gaus.	44.30	2.63	585.38	17.20	22354.78	258.23	1075.98	58.30	631.53	17.11
Sigd	81.81	2.19	760.38	19.94	42609.42	158.08	2627.80	62.88	1133.34	19.61
Wavt	37.63	1.94	422.84	38.03	26304.67	101.60	1087.44	107.98	792.64	**5.58**
IMult.	36.42	1.95	387.73	16.02	20921.32	97.06	1080.31	61.56	724.94	16.22
Spln	7.38	5.02	75.32	156.98	2180.37	321.88	220.29	159.01	93.89	17.22
Cauy	29.53	1.95	377.56	15.41	19320.70	112.22	1164.72	55.86	695.73	15.41
K_{10}	5.98	2.53	69.69	18.41	**1993.55**	134.50	**142.61**	55.65	84.48	23.14
K_{11}	6.08	2.55	68.75	17.94	2083.03	191.02	167.29	**48.43**	91.80	21.75
K_{12}	6.38	1.89	**68.72**	16.06	2105.91	**94.76**	156.93	50.51	**82.92**	16.77
K_{13}	**5.88**	2.00	69.72	**15.34**	2378.20	121.23	149.64	60.44	89.11	15.81

Bold underline values are the best training and testing time.

of the German and Heart datasets can be achieved by using the new kernels k_{11} and k_{12}, respectively. The enhancement becomes clearer when the artificial datasets are used. Second, contrary to what has been reported in the literature, the Gaussian kernel is not always the best classical kernel. We can observe that the exponential kernel shows better accuracy for the real-world and the artificial datasets. Third, polynomial kernels of degree 2 are more suitable for most of the real-world datasets, while a higher degree is more suitable for the artificial datasets. Fourth, the spline kernel has the lowest classification rate. Finally, we note that, if the value of z (in the new kernels) is multiplied by a constant smaller than 1, then the SVM algorithm will not converge in most of the tested datasets; on the other hand, if the value of z is multiplied by a constant greater than 1, then the results are less accurate. Accordingly, the question why the new kernels have higher accuracy arises. We found that two important factors affect accuracy:

Table 10
Ranking of the kernels according to the classification rate.

	Real-world datasets											Artificial datasets							Avg.
	1	2	3	4	5	6	7	8	9	10	11	1	2	3	4	5	6	7	
Linr.	4	5	3	5	2	2	3	1	3	12	10	7	10	9	11	7	8	7	6.1
Poly.	1	5	4	6	3	5	1	2	2	9	9	3	7	6	7	1	3	3	4.3
Expl.	2	7	2	2	1	1	1	1	1	2	1	3	1	2	5	3	2	2	**2.2**
Gaus.	2	4	4	6	1	2	1	1	1	3	2	4	2	2	7	5	4	1	2.9
Sigd.	4	5	2	3	2	2	3	1	3	6	10	7	9	8	9	6	7	5	5.1
Wavt.	3	6	5	7	1	4	2	1	1	7	7	5	8	7	8	5	1	6	4.7
IMult.	2	7	1	6	1	4	1	1	5	8	8	3	5	5	6	4	5	6	4.3
Spln.	4	8	7	9	4	6	5	3	6	11	11	6	11	9	11	7	9	4	7.3
Cauy.	2	7	6	8	1	4	4	1	4	10	10	1	6	4	10	2	6	4	5.0
K_{10}	3	3	2	2	1	5	1	1	1	1	6	2	4	3	4	3	1	4	**2.6**
K_{11}	3	2	2	1	1	5	1	1	1	1	3	2	4	3	3	3	2	4	**2.3**
K_{12}	3	1	4	4	1	5	1	1	1	4	5	3	4	3	5	2	1	4	2.9
K_{13}	2	6	4	4	1	3	1	1	1	5	4	3	3	1	1	2	2	1	**2.5**

Bold underline values indicate to the best rank.

Table 11
Ranking of the kernels according to the training time and the testing time.

	Germ.		Oneh.		Fly		Arc		Artificial set 5		Average	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Linr	5	**1**	5	11	2	4	4	10	3	7	3.8	6.6
Poly.	12	7	12	7	12	11	12	9	12	10	12.0	8.8
Expl	10	2	10	5	10	6	8	3	8	4	9.2	**4.0**
Gaus.	11	10	11	6	9	12	7	5	7	8	9.0	8.2
Sigd	13	6	13	10	13	9	13	8	13	11	13.0	8.8
Wavt	9	3	9	12	11	3	10	11	11	**1**	10.0	6.0
IMult.	8	4	8	3	8	2	9	7	10	5	8.6	4.2
Spln	6	11	6	13	5	13	6	12	6	9	5.8	11.6
Cauy	7	4	7	2	7	5	11	4	9	2	8.2	**3.4**
K_{10}	2	8	3	9	**1**	8	**1**	4	2	13	**1.8**	8.4
K_{11}	3	9	2	8	3	10	5	**1**	5	12	3.6	8.0
K_{12}	4	2	**1**	4	4	**1**	3	2	**1**	6	**2.6**	**3.0**
K_{13}	**1**	5	4	**1**	6	7	2	6	4	3	**3.4**	4.4

Bold underline values indicate to the best rank according to training and testing time.

the standard deviation and the range of the non-diagonal elements should be close to one. Fortunately, using z as suggested in the new kernels guarantees a higher standard deviation and a higher range than blindly selecting the parameters, as in the other kernels.

Table 9 shows the training and testing time for the old and new kernels. The new kernels do not need to tune their parameters; therefore, these kernels have the shortest training time, where z can be calculated once in advance. The complexity of the test phase according to equation 3 is $O(m \times k)$, where m is the number of support vectors and k is the number of operations required to evaluate the kernel. The complexity of k depends on the number of features. Unfortunately, the operations of the kernels cannot be compared directly due to the variety of operation types, e.g., geometric and arithmetic operations. Table 9 compares the test time experimentally and shows that the new kernels outperform the old kernels.

Another perspective to look at the results is by ranking the kernels for each dataset as shown in Tables 10 and 11, which indicate that the new kernels are very useful for several datasets. The average column in Table 10 shows that the three best kernels for classifying are the exponential, k_{11} and k_{13}. However, the average column in Table 11 shows that the three best kernels according to training time are k_{10}, k_{12}, and k_{13} and the three best kernels according to testing time are k_{12}, the Cauchy, and the exponential.

6. Conclusion and unresolved problems

We have suggested a new direction for generating nonparametric and efficient kernels. Implementing several kernels using a large variety of binary real-world and artificial datasets has demonstrated that the kernels generated using the new framework provide a classification rate superior to that of most of the well-known kernels defined thus far. Moreover, these kernels need less training time. This study can be extended to introduce a new frontier in applying kernel methods by solving some open problems, such as proving the generalization of proposition 1 to $n \times n$ matrices, determining the necessary and/or sufficient conditions for the decreasing sequences in the last kernel in Section 3, checking whether there is a general procedure for generating triangle inequality-preserving functions, and generalizing the suggested framework to matrices with an arbitrary diagonal.

References

[1] B. Schölkopf, A.J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2002.

[2] D. Conforti, R. Guido, Kernel based support vector machine via semidefinite programming: application to medical diagnosis, Computers and Operations Research 37 (8) (2010) 1389–1394.

[3] Y. Tan, J. Wang, A support vector machine with a hybrid kernel and minimal Vapnik–Chervonenkis dimension, IEEE Transactions on Knowledge and Data Engineering 16 (4) (2004) 385–395.

[4] M. Gönen, E. Alpaydın, Multiple kernel learning algorithms, Journal of Machine Learning Research 12 (2011) 2211–2268.

[5] J. Yang, Y. Li, Y. Tian, L. Duan, W. Gao, A new multiple kernel approach for visual concept learning, in: Proceedings of the 15th International Multimedia Modeling Conference, LNCS vol. 5371 (2009) 250–262.

[6] C. Cortes, M. Mohri, A. Rostamizadeh, Learning non-linear combinations of kernels, Advances in Neural Information Processing Systems 22 (2010) 396–404.

[7] A. Ben-Hur, W. Stafford, Kernel methods for predicting protein–protein interactions, Bioinformatics 21 (2005) 38–46.

[8] S. Qiu, T. Lane, A framework for multiple kernel support vector regression and its applications to siRNA efficacy prediction, IEEE/ACM Transactions on Computational Biology and Bioinformatics 6 (2) (2009) 190–199.

[9] M. Christoudias, R. Urtasun, T. Darrell, Bayesian localized multiple kernel learning. Technical Report UCB/EECS-2009-96, University of California at Berkeley, 2009.

[10] J. Bi, T. Zhang, K.P. Bennett, Column-generation boosting methods for mixture of kernels, Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2004) 521–526.

[11] S. Min, J. Lee, I. Han, Hybrid genetic algorithms and support vector machines for bankruptcy prediction, Expert Systems with Applications 31 (2006) 652–660.

[12] C.R. Leslie, R. Kuang, Fast string kernels using inexact matching for protein sequences, Journal of Machine Learning Research 5 (2004) 1435–1455.

[13] T. Joachims, Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms, Kluwer Academic, Boston, 2002.

[14] C. Watkins, Dynamic Alignment Kernels Advances in Large Margin Classifiers, MIT Press, Cambridge, MA, 2000, pp. 39–50.

[15] D. DeCoste, B. Schölkopf, Training invariant support vector machines, Machine Learning 46 (2002) 161–190.

[16] M. Collins, N. Duffy, Convolution kernels for natural language Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2001, pp. 625–632.

[17] L. Zhang, W. Zhou, L. Jiao, Wavelet support vector machine, IEEE Transactions on Systems Man and Cybernetics Part B: Cybernetics 34 (1) (2004) 34–39.

[18] B. Jayanta, A least square kernel machine with box constraints, International Conference on Pattern Recognition 1 (2008) 1–4.

[19] H. Lin, C. Lin, A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods, Technical Report, Department of Computer Science, National Taiwan University, 2003.

[20] A. Asuncion, D.J. Newman, 2007, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[21] T. Hofmann, B. Schölkopf, A.J. Smola, Kernel methods in machine learning, Annals of Statistics 36 (3) (2008) 1171–1220.

[22] M. Charles, Interpolation of scattered data: distance matrices and conditionally positive definite functions, Constructive Approximation 2 (1) (1986) 11–22.

[23] S.R. Gunn, Support vector machines for classification and regression, Technical Report, Faculty of Engineering, Science and Mathematics School of Electronics and Computer Science. ISIS, 1998.

[24] A. Wieland. Twin spiral dataset. http://www.cs.cmu.edu/afs/cs.cmu. edu/project/ai-repository/ai/areas/neural/bench/cmu/0.html (last accessed March 2012).