Int. J. Mach. Learn. & Cyber. (2018) 9:1707–1717
DOI 10.1007/s13042-017-0675-7

ORIGINAL ARTICLE

Kernel learning and optimization with Hilbert–Schmidt independence criterion

Tinghua Wang[1,2] · Wei Li[1]

Received: 6 November 2016 / Accepted: 30 March 2017 / Published online: 11 April 2017

Abstract Measures of statistical dependence between random variables have been successfully applied in many machine learning tasks, such as independent component analysis, feature selection, clustering and dimensionality reduction. The success is based on the fact that many existing learning tasks can be cast into problems of dependence maximization (or minimization). Motivated by this, we present a unifying view of kernel learning via statistical dependence estimation. The key idea is that good kernels should maximize the statistical dependence between the kernels and the class labels. The dependence is measured by the Hilbert–Schmidt independence criterion (HSIC), which is based on computing the Hilbert–Schmidt norm of the cross-covariance operator of mapped samples in the corresponding Hilbert spaces and is traditionally used to measure the statistical dependence between random variables. As a special case of kernel learning, we propose a Gaussian kernel optimization method for classification by maximizing the HSIC, where two forms of Gaussian kernels (spherical kernel and ellipsoidal kernel) are considered. Extensive experiments on real-world data sets from UCI benchmark repository validate the superiority of the proposed approach in terms of both prediction accuracy and computational efficiency.

✉ Tinghua Wang
wthpku@163.com

1 School of Mathematics and Computer Science, Gannan Normal University, Ganzhou 341000, People's Republic of China

2 Decision Systems and e-Service Intelligence Laboratory, Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Broadway, NSW 2007, Australia

Keywords Kernel method · Kernel learning · Hilbert–Schmidt independence criterion (HSIC) · Statistical dependence · Gaussian kernel optimization · Classification

1 Introduction

Kernel methods such as support vector machines (SVM) and kernel Fisher discriminant analysis (KFDA) have delivered extremely high performance in a wide variety of learning tasks [1, 2]. Basically, kernel methods work by mapping the data from the input space to a high-dimensional (possibly infinite) feature space, which is usually chosen to be a reproducing kernel Hilbert space (RKHS), and then building linear algorithms in the feature space to implement nonlinear counterparts in the input space. The mapping, rather than being given in an explicit form, is determined implicitly by specifying a kernel function (or simply a kernel), which computes the inner product between each pair of data points in the feature space. Since the geometrical structure of the mapped data in the feature space is totally determined by the kernel function, choosing an appropriate kernel function and thus an appropriate feature space has a crucial effect on the performance of any kernel method. Kernel design and optimization is often considered as the problem of kernel learning, which is one of the central interests in kernel methods [3–5].

Measuring dependence of random variables is one of the main concerns of statistical inference. A typical example is the inference of a graphical model, which expresses the relations among variables in terms of independence and conditional independence [6]. A generalization of this idea in kernel methods is to embed probability distributions into RKHSs, giving us a linear method to infer properties of the distributions, such as independence and homogeneity [7–9]. In the last decade, various kernel statistical dependence measures have been proposed for statistical analysis, such as kernel generalized variance [10], kernel constrained covariance [11] and Hilbert–Schmidt independence criterion (HSIC) [12]. These measures differ in the way they summarize the covariance operator spectrum and the normalization they use. Among these measures, HSIC is the most well known, which is defined as the Hilbert–Schmidt norm of the cross covariance operator between RKHSs. With several key advantages over other classical metrics on distributions, namely easy computability, fast convergence and low bias of finite sample estimates, HSIC has been successfully applied in statistical test of independence [7, 9]. More interestingly, HSIC is a very useful tool for many machine learning problems. For instance, clustering can be viewed as a problem where one strives to maximize the dependence between the observations and a discrete set of labels [13, 14]. If labels are given, feature selection can be achieved by finding a subset of features in the observations which maximize the dependence between features and labels [15, 16]. Similarly in subspace learning, one looks for a low dimensional embedding which retains additional side information such as class labels and distances between neighboring observations [17, 18].

Although HSIC has been widely applied in machine learning, there is as yet no application in kernel learning. The core objective of kernel learning is to guarantee good generalization performance of the learning machine (predictor). Kernel learning is usually implemented by minimizing the generalization error, which can be estimated either via testing on some unused data (hold-out testing or cross validation) or with theoretical bounds [19, 20]. The notion bounding the generalization error provides approaches not only for selecting single optimal kernel but also for combining multiple base kernels [4, 21]. There are mainly two limitations for these methods. Firstly, they are dependent of the predictor. For instance, the radius-margin bound [19, 20] is only applicable to SVM. Second, they require the whole learning process for evaluation. For example, usually the most sophisticated techniques for kernel learning based on the radius-margin bound are gradient descent algorithms. However, these algorithms, at each iteration, require training the learning machine and solving an additional quadratic program to compute the radius of the smallest ball enclosing the training data in the RKHS [3]. Many universal kernel evaluation measures have been proposed to address these limitations, such as kernel alignment [22–24], kernel polarization [3, 25, 26], feature space-based kernel matrix evaluation measure [27] and class separability [28]. From geometric points of view, kernel learning with these measures actually searches for an optimal RKHS, in which data points associated with the same class come close while those belonging to different classes go apart.

In this paper, we investigate the kernel learning problem from a statistical viewpoint: given the input data (such as features) and outputs (such as labels), we aim to find a kernel such that the statistical dependence between the input data and outputs is maximized. Specifically, we first present a general kernel learning framework with the HSIC based on the observation that rich constraints on the outputs can be defined in RKHSs. We will see that this framework is directly applicable to classification, clustering and other learning models. As a special case of kernel learning, we propose a Gaussian kernel optimization method by maximizing the HSIC, where two forms of Gaussian kernels (spherical kernel and ellipsoidal kernel) are considered. Using the derivatives of HSIC with respect to kernel parameters, gradient-based optimization techniques can be employed to address these maximization formulations. The relationship between the proposed approach and the centered kernel alignment method [23], which is an improved kernel evaluation technique based on the popular kernel target alignment method [22], is also discussed.

The rest of the paper is structured as follows. We review the basics of HSIC in Sect. 2. In Sect. 3, we first present a general kernel learning framework that can be applied to different learning models, and then propose a Gaussian kernel optimization method for classification in this framework. We also discuss the relationship between the HSIC and centered kernel alignment measure. In Sect. 4, the empirical results are provided, followed by the conclusion and future work in Sect. 5.

2 Hilbert–Schmidt independence criterion

Let X and Y be two domains from which we draw a set of samples $D = \{(x_i, y_i)\}_{i=1}^{n}$ jointly from some probability distribution P_{xy}. The HSIC [12] measures the dependence (or independence) between x and y by computing the norm of the cross-covariance operator over the domain $\mathrm{X} \times \mathrm{Y}$ in RKHS. Formally, let F and G be the RKHSs on X and Y with feature maps $\phi:\mathrm{X} \to \mathrm{F}$ and $\varphi:Y \to \mathrm{G}$, respectively. The associated reproducing kernels are defined as $k(x, x') = \langle \phi(x), \phi(x') \rangle$ for any $x, x' \in \mathrm{X}$ and $l(y, y') = \langle \varphi(y), \varphi(y') \rangle$ for any $y, y' \in \mathrm{Y}$, respectively. The cross-covariance operator between feature maps ϕ and φ is defined as a linear operator $C_{xy}:\mathrm{G} \to \mathrm{F}$, such that:

$$C_{xy} = \mathrm{E}_{xy}\{(\phi(x) - \mathrm{E}_x[\phi(x)]) \otimes (\phi(y) - \mathrm{E}_y[\phi(y)]\} \tag{1}$$

where $\otimes$ is the tensor product, and the expectations E_{xy}, E_x, and E_y are taken according to some probability distribution P_{xy} and the marginal probability distributions P_x and P_y, respectively. The HSIC is then defined as the square of the Hilbert–Schmidt norm of C_{xy}:

$$\begin{aligned} HSIC(F,G,P_{xy}) &= ||C_{xy}||^2_{HS} \\ &= \mathrm{E}_{xx'yy'}[k(x,x')l(y,y')] \\ &\quad -2\mathrm{E}_{xy}\{\mathrm{E}_{x'}[k(x,x')]\mathrm{E}_{y'}[l(y,y')]\} \\ &\quad +\mathrm{E}_{xx'}[k(x,x')]\mathrm{E}_{yy'}[l(y,y')] \end{aligned} \tag{2}$$

where $\mathrm{E}_{xx'yy'}$ is the expectation over both $(x,y) \sim \mathrm{P}_{xy}$ and an additional pair of variables $(x',y') \sim \mathrm{P}_{xy}$ drawn independently according to the same law. It is easy to see that if both feature maps are linear (i.e., $\phi(x) = x$ and $\varphi(y) = y$), HSIC is equivalent to the square of the Frobenius norm of the cross-covariance matrix. Given D, an empirical estimator of HSIC is given by

$$\begin{aligned} HSIC(\mathrm{F},\mathrm{G},D) &= \frac{1}{n^2}\mathrm{tr}(\mathbf{KL}) - \frac{2}{n^3}\mathbf{e}^{\mathrm{T}}\mathbf{KLe} + \frac{1}{n^4}\mathbf{e}^{\mathrm{T}}\mathbf{Kee}^{\mathrm{T}}\mathbf{Le} \\ &= \frac{1}{n^2}\left[\mathrm{tr}(\mathbf{KL}) - \frac{1}{n}\mathrm{tr}(\mathbf{KLee}^{\mathrm{T}}) - \frac{1}{n}\mathrm{tr}(\mathbf{LKee}^{\mathrm{T}}) + \frac{1}{n^2}\mathrm{tr}(\mathbf{Lee}^{\mathrm{T}}\mathbf{Kee}^{\mathrm{T}})\right] \\ &= \frac{1}{n^2}\left\{\mathrm{tr}\left[\mathbf{KL}\left(\mathbf{I} - \frac{1}{n}\mathbf{ee}^{\mathrm{T}}\right)\right] - \frac{1}{n}\mathrm{tr}\left[K\mathbf{ee}^{\mathrm{T}}\mathbf{L}\left(\mathbf{I} - \frac{1}{n}\mathbf{ee}^{\mathrm{T}}\right)\right]\right\} \\ &= \frac{1}{n^2}\mathrm{tr}\left[\mathbf{K}\left(\mathbf{I} - \frac{1}{n}\mathbf{ee}^{\mathrm{T}}\right)\mathbf{L}\left(\mathbf{I} - \frac{1}{n}\mathbf{ee}^{\mathrm{T}}\right)\right] \\ &= \frac{1}{n^2}\mathrm{tr}(\mathbf{KHLH}) \end{aligned} \tag{3}$$

where $\mathrm{tr}(\cdot)$ is the trace operator, $\mathbf{e} = (1,\cdots,1)^{\mathrm{T}} \in \mathrm{R}^n$, and $\mathbf{K},\mathbf{L} \in \mathrm{R}^{n\times n}$ (R denotes the set of real numbers) are respectively the kernel matrices defined as $\mathbf{K}_{i,j} = k(x_i,x_j)$ and $\mathbf{L}_{i,j} = l(y_i,y_j)$. Moreover, $\mathbf{H} = \mathbf{I} - \mathbf{ee}^{\mathrm{T}}/n \in \mathrm{R}^{n\times n}$ is a centering matrix, where $\mathbf{I} \in \mathrm{R}^{n\times n}$ is the identity matrix.

The attractiveness of HSIC stems from the fact that the empirical estimator can be expressed completely in terms of kernels. For a particular class of kernels, i.e. the so-called universal or characteristic kernels [6, 29] such as Gaussian and Laplace kernels, HSIC is equal to zero if and only if two random variables are statistically independent. Note that non-universal and non-characteristic kernels can also be used for HSIC, although they may not guarantee that all dependence is detected [16]. In general, the larger HSIC is, the larger the dependence between two random variables.

3 Kernel learning with HSIC

3.1 A general kernel learning framework

As a typical kernel method, HSIC used by previous work requires us to choose kernels manually because no objective model selection approaches are available. In practice, using Gaussian kernel with width parameter set to the median distance between samples is a popular heuristic [7, 12], although such a heuristic does not always work well [30]. Different from previous work, we here use the HSIC as an evaluation criterion to assess the quality of a kernel for kernel learning. For machine learning, X and Y can be seen as the input space and output space, respectively. Correspondingly, **K** and **L** are the kernel matrices for input data and output labels, respectively. Intuitively, good kernels should maximize the statistical dependence between the kernel for input data and the kernel for output labels. Hence kernel learning can be cast into the dependence maximization framework in the following way:

$$\begin{aligned} &\mathbf{K}^* = \max_{\mathbf{K}} \ \mathrm{tr}\mathbf{KHLH} \\ &\text{s.t. constraints on } \mathbf{K} \text{ and } \mathbf{L} \end{aligned} \tag{4}$$

That is to say, kernel learning can be viewed as maximizing the empirical HSIC subject to constraints on **K** and **L**, for particular Hilbert spaces on the inputs and labels.

There are several advantages of this framework. First, the empirical HSIC is stable with respect to different splits of the data since it is sharply concentrated around its expected value (the empirical HSIC asymptotically converges to the true HSIC with $O(1/\sqrt{n})$ [12]). This means that the same kernels should be consistently selected to achieve high dependence when the data are repeatedly drawn from the same distribution. Second, the empirical HSIC is easy to compute (it can be computed in $O(n^2)$ time [12, 16]) since only the kernel matrices **K** and **L** are needed and no density estimation is involved. Furthermore, evaluating the empirical HSIC is independent of the specific learning machines. These mean that we can use only the training data to select the good kernels in an efficient way prior to any computationally intensive training of the kernel machines. Finally and most importantly, rich choices of kernels can be directly applicable to the inputs and labels. This freedom of choosing kernels allows us to generate a family of kernel learning models via simply defining appropriate kernels (in some sense, these kernels incorporate prior knowledge of the leaning tasks at hand) on the inputs and outputs, respectively. Some examples are given as follows:

1. *Kernels on inputs*. Kernels on the input data can be either the popular used kernels, suchas polynomial kernel and Gaussian kernel, or kernels defined on non-vectorial data, such as string kernel, tree kernel and graph kernel. Moreover, instead of using a single kernel, we can use kernel combinations. A popular technique addressing this issue is multiple kernel learning (MKL) [4, 26], which aims to learn an optimal combination of a set of predefined base kernels. If we consider the convex combination of the base kernels, the

problem of kernel learning can be transformed to the problem of determining the combination coefficients:

$$\boldsymbol{\mu}^* = \max_{\boldsymbol{\mu}} \ \mathrm{tr}\mathbf{KHLH}$$
$$\text{s.t. } \mathbf{K} = \sum_{i=1}^{m} \mu_i \mathbf{K}_i, \ \mu_i \geqslant 0, \ \sum_{i=1}^{m} \mu_i = 1, \tag{5}$$

and constraints on $\mathbf{L}$

where $\mathbf{K}_i$ $(i = 1, \dots, m)$ are the base kernels and $\boldsymbol{\mu} = (\mu_1, \dots, \mu_m)^{\mathrm{T}}$ is the weight coefficient vector of the given base kernels.

2. *Kernels on outputs*. Generally speaking, kernels on the output labels can be as general as those defined on the input data. However, in kernel learning, prior knowledge of the learning models should be more considered to define such kernels. We can define different kernels according to different leaning models, such as classification and clustering. For classification, the kernel can be defined as

$$l(y_i, y_j) = \begin{cases} -1 & y_i = y_j \\ +1 & y_i \neq y_j \end{cases} \tag{6}$$

This definition reveals the ideal pairwise similarities between samples, i.e. the similarities from the same class are set to +1 while those from different classes are −1. For clustering, since the labels are not provided, the definition of kernels is somewhat more complex. Song et al. [13] proposed a method to define such kernels. They first chose a symmetric positive semidefinite matrix A of size $b \times b$ (b denotes the number of clusters and $b \ll n$) defining the similarities between samples in Y, and then used a partition matrix $\mathbf{\Pi}$ of size $n \times b$ to parameterize the kernel matrix of the outputs as

$$\mathbf{L} = \mathbf{\Pi} \mathrm{A} \mathbf{\Pi}^{\mathrm{T}} \tag{7}$$

Each row of $\mathbf{\Pi}$ contains all zeros but a single entry of 1. Actually, the partition matrix $\mathbf{\Pi}$ constrains us to assign each sample to a particular cluster by putting 1 in an appropriate column, which means each sample should be assigned to one and only one cluster.

To summarize, our formula (4) is very general: we can obtain a family of kernel learning algorithms by combining a kernel for the input space and another for the output space. For instance, we can have a model of multiple kernel classification [23] by combining the formulas (5) and (6), and obtain an algorithm of multiple kernel clustering [31] by combining the formulas (5) and (7).

3.2 Gaussian kernel optimization for classification

In this section, we will illustrate a special case of kernel learning in the proposed framework, i.e. the kernel on the inputs is the Gaussian kernel and that on the outputs is defined as (6). Our objective is to learn the parameters of the Gaussian kernel for classification. Two forms of Gaussian kernel are considered. One is the spherical kernel and the other is the ellipsoidal kernel, which are given by (8) and (9), respectively:

$$k(x_i, x_j) = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \tag{8}$$

$$k(x_i, x_j) = \exp\left[-\sum_{z=1}^{d}\left(\frac{\left\|x_{iz} - x_{jz}\right\|^2}{2\sigma_z^2}\right)\right] \tag{9}$$

For the spherical kernel, the σ $(\sigma > 0)$ is the kernel width parameter. For the ellipsoidal kernel, d denotes the number of dimensions, and x_{iz} and σ_z $(\sigma_z > 0)$ are respectively the zth component of the sample x_i and the width for the zth dimension.

Let $\bar{\mathbf{L}} = \mathbf{HLH}$ and θ denotes the kernel parameter set, we have $\theta = \{\sigma\}$ and $\theta = \{\sigma_1, \dots, \sigma_d\}$ for the spherical kernel and ellipsoidal kernel, respectively. The optimal θ^* can be obtained by

$$\begin{aligned}\theta^* &= \max_{\theta} \ \mathrm{tr}\mathbf{KHLH} = \max_{\theta} \ \mathrm{tr}\mathbf{K}\bar{\mathbf{L}} = \max_{\theta} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{K}_{i,j}\bar{\mathbf{L}}_{i,j} \\ &= \max_{\theta} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{K}_{i,j}\left(\mathbf{L}_{i,j} - \frac{1}{n}\sum_{i=1}^{n}\mathbf{L}_{i,j} - \frac{1}{n}\sum_{j=1}^{n}\mathbf{L}_{i,j} + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{L}_{i,j}\right) \\ &= \max_{\theta} \sum_{i=1}^{n}\sum_{j=1}^{n} k(x_i, x_j)\left[l(y_i, y_j) - \frac{1}{n}\sum_{i=1}^{n} l(y_i, y_j) - \frac{1}{n}\sum_{j=1}^{n} l(y_i, y_j) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} l(y_i, y_j)\right]\end{aligned} \tag{10}$$

The first and second derivatives of tr**KHLH** with respect to θ can be formulated as (11) and (12), respectively.

$$\frac{\partial \mathrm{tr}\mathbf{KHLH}}{\partial\theta} = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial k(x_i,x_j)}{\partial\theta}\left[l(y_i,y_j) - \frac{1}{n}\sum_{i=1}^{n} l(y_i,y_j) - \frac{1}{n}\sum_{j=1}^{n} l(y_i,y_j) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} l(y_i,y_j)\right] \tag{11}$$

$$\frac{\partial^2 \mathrm{tr}\mathbf{KHLH}}{\partial\theta^2} = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial^2 k(x_i,x_j)}{\partial\theta^2}\left[l(y_i,y_j) - \frac{1}{n}\sum_{i=1}^{n} l(y_i,y_j) - \frac{1}{n}\sum_{j=1}^{n} l(y_i,y_j) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} l(y_i,y_j)\right] \tag{12}$$

For the spherical Gaussian kernel, the first and second derivatives of $k(x_i, x_j)$ with respect to σ can be computed by (13) and (14), respectively.

$$\frac{\partial k(x_i,x_j)}{\partial\sigma} = \frac{\left\|x_i - x_j\right\|^2}{\sigma^3}\exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \tag{13}$$

$$\frac{\partial^2 k(x_i,x_j)}{\partial\sigma^2} = \left(\frac{\left\|x_i - x_j\right\|^4}{\sigma^6} - \frac{3\left\|x_i - x_j\right\|^2}{\sigma^4}\right)\exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \tag{14}$$

For the ellipsoidal Gaussian kernel, the first and second derivatives of $k(x_i, x_j)$ with respect to σ are given by (15) and (16), respectively.

$$\frac{\partial k(x_i,x_j)}{\partial\sigma_h} = \frac{\left\|x_{ih} - x_{jh}\right\|^2}{\sigma_h^3}\exp\left[-\sum_{z=1}^{d}\left(\frac{\left\|x_{iz} - x_{jz}\right\|^2}{2\sigma_z^2}\right)\right] \tag{15}$$

$$\frac{\partial k^2(x_i,x_j)}{\partial\sigma_h^2} = \left(\frac{\left\|x_{ih} - x_{jh}\right\|^4}{\sigma_h^6} - \frac{3\left\|x_{ih} - x_{jh}\right\|^2}{\sigma_h^4}\right)\exp\left[-\sum_{z=1}^{d}\left(\frac{\left\|x_{iz} - x_{jz}\right\|^2}{2\sigma_z^2}\right)\right] \tag{16}$$

According to (11)–(16), the optimal kernel parameter set θ^* can be found by using gradient-based optimization techniques.

3.3 Relation to centered kernel alignment method.

The notion of kernel alignment, which measures the degree of agreement between a kernel and a learning task, is widely used for kernel learning due to its simplicity, efficiency and theoretical guarantee [22, 24]. Mathematically, with two kernel matrices **K** and **L**, kernel alignment is given by:

$$KA(\mathbf{K},\mathbf{L}) = \frac{<\mathbf{K},\mathbf{L}>_{\mathrm{F}}}{\sqrt{<\mathbf{K},\mathbf{K}>_{\mathrm{F}}<\mathbf{L},\mathbf{L}>_{\mathrm{F}}}} \tag{17}$$

where $<\cdot,\cdot>_{\mathrm{F}}$ denotes the Frobenius inner product between two matrices.

A limitation of kernel alignment is that it does not consider the unbalanced class distribution which may cause the sensitivity of the measure to drop drastically. Cortes et al. [23] proposed to center kernels (or kernel matrices) before computing the alignment measure to cancel the effect of unbalanced class distribution. Let $\bar{\mathbf{K}} = \mathbf{HKH}$, built upon the kernel alignment, centered kernel alignment (CKA) is defined as

$$CKA(\mathbf{K},\mathbf{L}) = \frac{<\bar{\mathbf{K}},\bar{\mathbf{L}}>_{\mathrm{F}}}{\sqrt{<\bar{\mathbf{K}},\bar{\mathbf{K}}>_{\mathrm{F}}<\bar{\mathbf{L}},\bar{\mathbf{L}}>_{\mathrm{F}}}} \tag{18}$$

Although this improved definition of alignment may appear to be a technicality, it is actually a critical difference. Without that centering, the definition of alignment does not correlate well with the performance of learning machines [23].

Comparing (3) with (18), since $<\bar{\mathbf{K}},\bar{\mathbf{L}}>_{\mathrm{F}} = <\bar{\mathbf{K}},\mathbf{L}>_{\mathrm{F}} = <\mathbf{K},\bar{\mathbf{L}}>_{\mathrm{F}} = \mathrm{tr}\mathbf{K}\bar{\mathbf{L}} = \mathrm{tr}\mathbf{KHLH}$, the CKA is simply a normalized version of HSIC. However, for computational convenience the normalization is often omitted in practice [32]. Despite this similarity between HSIC and CKA, CKA has mainly been used for kernel selection and kernel learning, areas of application rather dissimilar to the applications of HSIC mentioned in Sect. 1.

4 Experiments

This section evaluates the effectiveness and efficiency of the proposed Gaussian kernel optimization method for classification. Since this Gaussian kernel optimization is a special case of the proposed general kernel learning framework, such evaluation can also demonstrate the benefits of the proposed general kernel learning framework to a certain degree. Two forms of Gaussian kernel, i.e. the spherical kernel and ellipsoidal kernel, are respectively used to evaluate the performance of the proposed method in dealing with small- and large-sized kernel parameter sets.

4.1 Data sets

We selected 10 popular data sets, i.e., *Zoo, Sonar, Ecoli, Ionosphere, Dermatology, Australian Credit Approval, Vehicle Silhouettes, Yeast, Image Segmentation* and *Waveform*, from the UCI machine learning repository [33]. Among them, *Sonar, Ionosphere* and *Australian Credit Approval* are binary-class data sets and the others are multiclass data sets. Table 1 provides the statistics of these data sets. It presents, for each data set, the ID number, the number of samples, the number of features, the number of classes, and the minimum, maximum and average numbers of samples per class. All the benchmark examples considered are small data sets ranging in sample number from 101 to 5000, and in dimension from 7 to 60. Some of them are characterized by unbalanced class distribution, such as *Zoo, Ecoli, Dermatology* and *Yeast*. These data sets form a good test bed for evaluating different algorithms.

For each data set, we partitioned it into a training set and a test set by stratified sampling (by which the object generation follows the class prior probabilities): 50% of the data set serves as the training set and the remaining 50% as the test set. For the training set of each data set, feature values along each dimension were linearly scaled to [0, 1]. Suppose mz and Mz are respectively the maximal and minimal values of the zth dimension. Scaling to [0, 1] means $x'_{iz} = (x_{iz} - mz)/(Mz - mz)$. The test set was also scaled accordingly.

4.2 Implementation details

4.2.1 4.2.1 Optimization

From (10)–(12), the HSIC has the first and second derivates with respect to the model parameters as long as the employed kernel has. There are many choices of gradient-based optimization techniques to efficiently solve the maximization the HSIC. In this experiment, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) quasi-Newton algorithm was used because it generally takes a smaller number of iterations before convergence [20]. In numerical optimization, the BFGS algorithm is an iterative method for solving unconstrained nonlinear optimization problems. To avoid the constraint of $\sigma > 0$ ($\sigma_z > 0$ for the ellipsoidal Gaussian kernel), the criterion was optimized according to $\ln(\sigma)$, where $\ln(\cdot)$ denotes the natural logarithm. Once the optimum is reached, the value of the parameters can be obtained by taking the exponential operation, which is always positive. Thus, the maximization of HSIC becomes an unconstrained optimization problem. The initial value of σ ($\sigma_z > 0$ for the ellipsoidal Gaussian kernel) was set as $\sigma = 1/\text{num\_features}$, where num_features is the number of the features. Suppose BFGS starts an optimization iteration at θ, then successfully completes a line search and reaches the next point $\bar{\theta}$. Optimization is terminated when the inequality $f(\bar{\theta}) - f(\theta) \leqslant 10^{-5} f(\theta)$ is met, where $f(\theta)$ and $f(\bar{\theta})$ are the corresponding values of the objective function, respectively. In other words, the optimization will be terminated if the difference of the values of the objective function in two consecutive iterations is less than a predetermined tolerance. Lastly, the BFGS quasi-Newton algorithm was implemented by using the function *fminunc*() in Matlab.

4.2.2 Multiclass classification

For kernel methods, the kernel is the only domain specific module, while the learning algorithm is a general purpose component. Potentially any kernel can work with any kernel-based algorithm. However, the SVM is employed as the classifier in this experiment, due to its great popularity in kernel methods and delivering very impressive performance in real applications [1]. The LIBSVM software [34] was used to train and test the SVM classifier. SVM was originally proposed for binary classification tasks. There are several methods available to extend binary SVM to

Table 1 Statistics of the selected ten data sets from UCI

ID	Data set	#Samples	#Features	#Classes	#Min/max/average
1	Zoo	101	16	7	4/41/14.4
2	Sonar	208	60	2	97/111
3	Ecoli	336	7	8	2/143/42
4	Ionosphere	351	34	2	126/225
5	Dermatology	366	34	6	20/112/61
6	Australian Credit Approval	690	14	2	307/383
7	Vehicle Silhouettes	846	18	4	199/218/211.5
8	Yeast	1484	8	10	5/463/148.4
9	Image Segmentation	2310	19	7	330/330/330
10	Waveform	5000	21	3	1647/1696/1666.7

multiclass SVM [35]. LIBSVM implements "one-against-one" multiclass method, hence there are $c(c-1)/2$ binary SVMs, where c is the number of classes. There are two ways to conduct kernel parameter optimization. (1) For any two classes of data, a parameter optimization procedure is first conducted and each decision function then has its own optimal kernel parameters. In other words, this method locally applies different parameter values for different binary SVMs independently. (2) The same parameters are globally used for all $c(c-1)/2$ binary SVMs and the parameters that achieve the high$c(c-1)/2$est overall performance are selected. Intuitively, both methods have drawbacks. On the one hand, the same parameter set for all binary classifiers may not be uniformly good for all $c(c-1)/2$ decision functions. On the other hand, optimizing very well each classifier individually does not ensure that they will perform well together. However, it is a wiser choice to share the same parameter set in practice because of the following considerations: (1) much less parameters need to be optimized. (2) Chen et al. [36] have experimentally shown that these two methods give similar performance. (3) From the definition of the output kernel shown as (6), it is clear that the HSIC simultaneously encodes the multiclass information, and is unified for both binary and multiclass classification problems. Lastly, it should be noted that independent parameter set implies a higher degree of tuning and so a higher chance of over-fitting. Therefore, in terms of both classification performance and computational efficiency, the second method that considers the same parameter set for all binary SVMs was employed in this experiment.

4.2.3 Comparison approaches

The following four kernel learning approaches are compared in this experiment:

- The cross-validation method [19, 20] denoted as 'CV', which is regarded as a benchmark here since it is probably the most simplest and prominent approach for general kernel learning and optimization. Because of the high computational complexity, this method is only suitable for the adjustment of very few parameters. For the SVM classifier with the spherical Gaussian kernel, there are two parameters, i.e. the kernel width parameter σ and the regularization parameter C, to be optimized. We performed the five-fold cross-validation to find the best σ and C on the training set. A grid search over two dimensions, i.e., $\gamma = \{2^{-15}, 2^{-14}, \ldots, 2^{3}\}$ and $C = \{2^{-5}, 2^{-3}, \cdots, 2^{15}\}C$, where $\gamma = 1/\sigma^2$, is applied. For the ellipsoidal Gaussian kernel, since different width values are assigned to each dimension, the number of kernel parameters amounts to the feature dimensionality. In this case, it becomes intractable using the five-fold cross-validation method for kernel optimization, even for a data set with low-dimensional features.
- The kernel alignment (KA) [22] and BFGS quasi-Newton method. We first optimized the kernel parameters using the BFGS quasi-Newton method with KA on the training set. After the kernel optimization, the regularization parameter C was selected using the five-fold cross-validation (a line search over $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$).
- The centered kernel alignment (CKA) [23] and BFGS quasi-Newton method. We first optimized the kernel parameters using the BFGS quasi-Newton method with CKA on the training set. After the kernel optimization, the regularization parameter C was selected using the five-fold cross-validation (a line search over $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$).
- The proposed HSIC and BFGS quasi-Newton method. We first optimized the kernel parameters using the BFGS quasi-Newton method with HSIC on the training set. After the kernel optimization, the regularization parameter C was selected using the five-fold cross-validation (a line search over $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$).

All experiments are conducted on a PC with 2.6 GHz CPU and 4 GB RAM.

4.3 Results and discussion

The average classification error rates (test error rates) and running time over 10 trials are summarized in Tables 2, 3, 4 and 5. The bold font denotes the best result across the approaches compared. For more reliable comparison, we

Table 2 Classification error rates of the compared four approaches with the spherical Gaussian kernel

ID	Data set	Classification error (%)			
		CV	KA	CKA	HSIC
1	Zoo	**7.7295**	8.2106	7.8924	7.9218
2	Sonar	**10.0623**	11.3634	11.6357	11.2246
3	Ecoli	13.4467	14.6871	13.6736	**13.2213**
4	Ionosphere	**8.1843**	10.0398	10.1321	9.2145
5	Dermatology	**2.7716**	3.8246	2.9863	3.0526
6	Australian credit approval	25.2587	24.9827	**24.1253**	24.5760
7	Vehicle silhouettes	**16.7913**	18.8105	17.9753	17.8924
8	Yeast	**39.9421**	43.1327	41.7740	41.9306
9	Image segmentation	**3.4268**	4.3531	4.5249	4.4375
10	Waveform	11.8733	12.0217	**11.6402**	11.7089
W–T–L		0–7–3	3–7–0	0–10–0	–

Table 3 Running time of the compared four approaches with the spherical Gaussian kernel

ID	Data set	Running time (second)			
		CV	KA	CKA	HSIC
1	Zoo	26.5720	0.5943	0.6186	0.3295
2	Sonar	72.3589	0.8391	0.8653	0.5922
3	Ecoli	63.1945	0.6327	0.5969	0.3873
4	Ionosphere	90.1433	2.3015	2.8438	1.2564
5	Dermatology	274.7760	7.9376	10.1605	6.3941
6	Australian credit approval	362.3681	16.7325	17.9854	15.3386
7	Vehicle silhouettes	551.0466	30.3647	36.0775	18.6812
8	Yeast	853.6207	50.9156	48.1163	23.1945
9	Image segmentation	2362.3390	123.8351	226.5194	89.3783
10	Waveform	5498.4162	372.1274	432.4635	241.3056
W–T–L		10–0–0	9–1–0	10–0–0	–

Table 4 Classification error rates of the compared four approaches with the ellipsoidal Gaussian kernel

ID	Data set	Classification error (%)		
		KA	CKA	HSIC
1	Zoo	11.5497	**9.6023**	10.2118
2	Sonar	12.9674	12.8652	**12.5139**
3	Ecoli	14.1092	13.1356	**13.0924**
4	Ionosphere	**8.6971**	9.0012	8.8970
5	Dermatology	3.5194	**2.7629**	2.8347
6	Australian credit approval	23.2349	**22.6591**	22.7824
7	Vehicle silhouettes	16.8732	16.506	**16.2618**
8	Yeast	42.5901	40.0679	**39.9251**
9	Image segmentation	3.4746	**3.3498**	3.3900
10	Waveform	**11.1391**	11.6354	11.2846
W–T–L		3–7–0	0–10–0	–

Table 5 Running time of the compared four approaches with the ellipsoidal Gaussian kernel

ID	Data set	Running time (second)		
		KA	CKA	HSIC
1	Zoo	1.9836	2.1573	1.2496
2	Sonar	4.2201	4.8145	3.3694
3	Ecoli	3.8429	4.0276	2.7987
4	Ionosphere	24.5347	30.8413	19.3185
5	Dermatology	73.6030	86.2163	49.2320
6	Australian credit approval	75.9523	73.4970	58.2424
7	Vehicle silhouettes	94.7252	134.8469	76.5478
8	Yeast	70.0268	65.4332	40.9745
9	Image segmentation	412.3152	397.6647	255.4603
10	Waveform	1019.2180	1080.2591	587.3326
W–T–L		10–0–0	10–0–0	–

performed two-tailed t-test [37] with a significant level of 0.05 to determine whether there is a significant difference between the proposed method and other approaches. Based on the t-test, the win–tie–loss (W–T–L) summarizations are also attached at the bottoms of tables. A win or a loss means that the proposed method is better or worse than other method on a data set. A tie means that both methods have the same performance.

For the spherical Gaussian kernel, from Table 2, it is seen that the proposed method 'HSIC' gives rise to the classification performance comparable to that obtained by using the five-fold cross-validation method 'CV'. Although the 'CV' method always gives the lowest accuracy error rate compared with other approaches, the difference between the 'CV' method and the proposed method 'HSIC' is not significant in most cases. Compared with the 'KA' method, it is found that 'HSIC' gives a better or comparable performance on all data sets. There is no significant performance difference between the 'CKA' method and the 'HSIC' method on all data sets. In terms of the running time, from Table 3, it is found that 'HSIC' gives the shortest time on all the data sets. Take the *Zoo* data set for example: 'HSIC' achieves the running time of 0.3295s, which is significant shorter than 26.5720, 0.5943 and 0.6186s obtained by 'CV', 'KA' and 'CKA', respectively. The reason why 'HSIC' is more computationally efficient than 'CKA' may be that 'HSIC' simplifies the 'CKA' criterion by ridding the latter of its denominator, making the evaluation considerably easier.

For the ellipsoidal Gaussian kernel, although the five-fold cross-validation method becomes intractable in this case, the 'KA', 'CKA' and 'HSIC' methods can still work well. From Table 4, it can be observed that the proposed method 'HSIC' gives a better or comparable performance to the 'KA' method on all data sets. 'CKA' and 'HSIC' give nearly the same classification results on all data

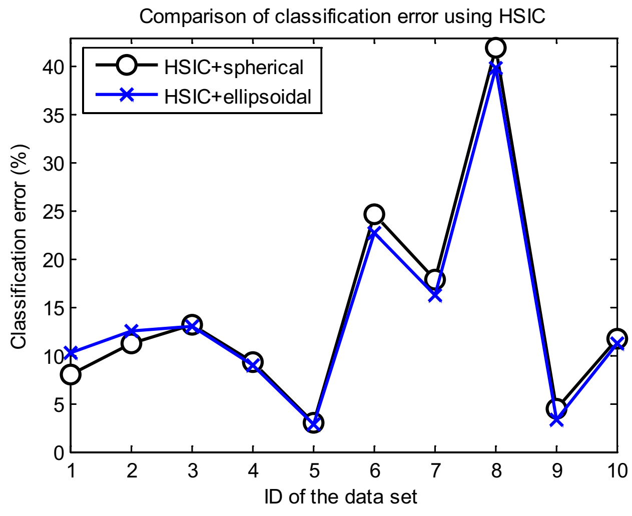

Fig. 1 Comparison of classification performance using HSIC with different Gaussian kernels

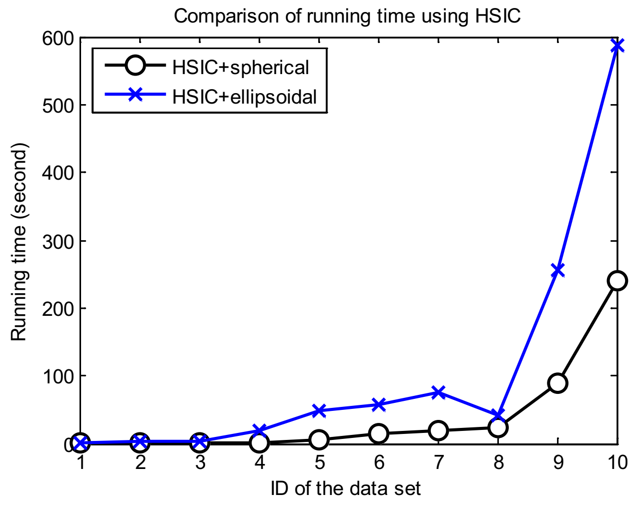

Fig. 2 Comparison of running time using HSIC with different Gaussian kernels

sets. From Table 5, 'HSIC' is seen as the most efficient method among the three compared approaches.

In a nutshell, although 'HSIC' yields almost the same classification performance as those achieved by other baseline approaches, it takes significantly less computational time. This suggests that our proposed method is both effective and efficient for Gaussian kernel optimization.

Is it necessary to use an ellipsoidal Gaussian kernel rather than a spherical Gaussian kernel for classification? Before the end of this section, we would like to discuss this issue to some extent. Figures 1 and 2 show the comparison of classification error rates and running time using 'HSIC' with these two kinds of Gaussian kernels, respectively. In terms of computational efficiency, as shown in Tables 3 and 5 and Fig. 2, the ellipsoidal Gaussian kernel clearly takes much longer running time than the spherical Gaussian kernel because for the former more kernel parameters need to be optimized. This is particularly true for the data sets with high-dimensional features. In terms of classification efficacy, as shown in Tables 2 and 4 and Fig. 1, lower classification error rates are observed on all but the *Zoo* and *Sonar* data sets for the ellipsoidal Gaussian kernel. This may be the benefit of using an ellipsoidal Gaussian kernel where different widths of the feature dimensions are adapted. For the *Zoo* and *Sonar* data sets, it is found that higher classification error rates are obtained by the ellipsoidal Gaussian kernel. Since the ellipsoidal Gaussian kernel is a generalization that includes the spherical Gaussian kernel, cases where the former performs worse than the latter suggest that over-fitting, i.e., when the size of the training set is so small the kernel parameter optimization may fit data noise and fail to capture real patterns there, has occurred. We also found that the classification error difference between these two kernels is not stably statistically significant.

In a word, compared with the ellipsoidal Gaussian kernel, the spherical Gaussian kernel can provide more efficient computation and comparable classification performances. Hence, from a practical point of view, the spherical Gaussian kernel must be preferred, especially in the cases where the training samples are not sufficient for avoiding over-fitting. Moreover, these results emphasize an important property generally related to the use of more powerful models: as the capacity of the models increases, so does the risk of over-fitting. To avoid over-fitting, it is necessary to use additional techniques such as early stopping of training process, regularization and Bayesian priors on parameters. Early stopping can be viewed as regularization in time. Intuitively, a training procedure like gradient descent will tend to learn more and more complex functions as the number of iterations increases. By regularizing in time, the complexity of the model can be controlled, improving generalization. From a Bayesian point of view, many regularization techniques correspond to imposing certain prior distributions on model parameters. How to avoid or mitigate over-fitting is an active research topic in machine learning and the regularization technique [38] seems to be a promising solution.

5 Conclusion and further study

We have explored the kernel learning problem from the statistical dependence maximization point of view. We first proposed a general kernel learning framework with the HSIC which is probably the most popular kernel statistical dependence measure for statistical analysis. This framework has been demonstrated to be directly applied in classification, clustering and other learning models. In this framework, we also presented a Gaussian kernel optimization method for classification, where two forms of Gaussian kernels (spherical kernel and ellipsoidal kernel) were investigated. Extensive experimental study on multiple benchmark data sets verifies the effectiveness and efficiency of the proposed method.

Future investigation will focus on the validation of the use of the proposed general kernel learning framework for other learning models such as MKL and clustering. Expanding the proposed model to extreme learning machine [39] and non-parametric kernel learning (NPKL) [40], as well as investigating the possibility of combining the proposed model with fuzzy set theory such as fuzzy SVM [41] are also important issues to be investigated. Moreover, we intend to use the newly proposed HSIC Lasso (least absolute shrinkage and selection operator) [42] for kernel learning.

As a final remark, it should be emphasized that, in general, by treating various learning models under a unifying framework and elucidating their relations, we also expect our work to benefit practitioners in their specific applications.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China (No. 61562003), the Natural Science Foundation of Jiangxi Province of China (No. 20161BAB202070), and the China Scholarship Council (No. 201508360144). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

1. Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge
2. Gao X, Fan L, Xu H (2015) Multiple rank multi-linear kernel support vector machine for matrix data classification. Int J Mach Learn Cybern. doi:10.1007/s13042-015-0383-0
3. Wang T, Tian S, Huang H, Deng D (2009) Learning by local kernel polarization. Neurocomputing 72(13–15):3077–3084
4. Gönen M, Alpayın E (2011) Multiple kernel learning algorithms. J Mach Learn Res 12:2211–2268
5. Pan B, Chen WS, Xu C, Chen B (2016) A novel framework for learning geometry-aware kernels. IEEE Trans Neural Netw Learn Syst 27(5):939–951
6. Fukumizu K, Gretton A, Sun X, Schölkopf B (2007) Kernel measures of conditional dependence. Adv Neural Inf Process Syst 20:489–496
7. Gretton A, Fukumizu K, Teo CH, Song L, Schölkopf B, Smola A (2007) A kernel statistical test of independence. Adv Neural Inf Process Syst 20:585–592
8. Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A (2012) A kernel two-sample test. J Mach Learn Res 13:723–773
9. Chwialkowski K, Gretton A (2014) A kernel independence test of random process. In: Proceedings of the 31th International Conference on Machine Learning, Beijing, China, pp 1422–1430
10. Bach FR, Jordan MI (2002) Kernel independent component analysis. J Mach Learn Res 3:1–48
11. Gretton A, Smola A, Bousquet O, Herbrich R, Belitski A, Augath M, Murayama Y, Pauls J, Schölkopf B, Logothetis NK (2005) Kernel constrained covariance for dependence measurement. In: Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, Bridgetown, Barbados, pp 112–119
12. Gretton A, Bousquet O, Smola A, Schölkopf B (2005) Measuring statistical dependence with Hilbert-Schmidt norms. In: Proceedings of the 16th International Conference on Algorithmic Learning Theory, Singapore, pp 63–77
13. Song L, Smola A, Gretton A, Borgwardt K (2007) A dependence maximization view of clustering. In: Proceedings of the 24th International Conference on Machine Learning, Corvallis, USA, pp 823–830
14. Zhong W, Pan W, Kwok JT, Tsang IW (2010) Incorporating the loss function into discriminative clustering of structured outputs. IEEE Trans Neural Netw 21(10):1564–1575
15. Camps-Valls G, Mooij J, Schölkopf B (2010) Remote sensing feature selection by kernel dependence measures. IEEE Geosci Remote Sens Lett 7(3):587–591
16. Song L, Smola A, Gretton A, Bedo J, Borgwardt K (2012) Feature selection via dependence maximization. J Mach Learn Res 13:1393–1434
17. Chen J, Ji S, Ceran B, Li Q, Wu M, Ye J (2008) Learning subspace kernels for classification. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, USA, pp 106–114
18. Shu X, Lai D, Xu H, Tao L (2015) Learning shared subspace for multi-label dimensionality reduction via dependence maximization. Neurocomputing 168:356–364
19. Chapelle O, Vapnik V, Mukherjee S (2002) Choosing multiple parameters for support vector machines. Mach Learn 46(1):131–159
20. Keerthi SS (2002) Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. IEEE Trans Neural Netw 13(5):1225–1229
21. Liu Y, Liao S, Hou Y (2011) Learning kernels with upper bounds of leave-one-out error. In: Proceedings of the 20th ACM Conference on Information and Knowledge Management, Glasgow, United Kingdom, pp 2205–2208
22. Cristianini N, Shawe-Taylor J, Elisseeff A, Kandola J (2001) On kernel-target alignment. Adv Neural Inf Process Syst 14:367–373
23. Cortes C, Mohri M, Rostamizadeh A (2012) Algorithms for learning kernels based on centered alignment. J Mach Learn Res 13:795–828
24. Wang T, Zhao D, Tian S (2015) An overview of kernel alignment and its applications. Artif Intell Rev 43(2):179–192
25. Baram Y (2005) Learning by kernel polarization. Neural Comput 17(6):1264–1275
26. Wang T, Zhao D, Feng Y (2013) Two-stage multiple kernel learning with multiclass kernel polarization. Knowl-Based Syst 48:10–16
27. Nguyen CH, Ho TB (2008) An efficient kernel matrix evaluation measure. Pattern Recognit 41(11):3366–3372
28. Wang L (2008) Feature selection with kernel class separability. IEEE Trans Pattern Anal Mach Intell 30(9):1534–1546
29. Steinwart I (2001) On the influence of the kernels on the consistency of support vector machines. J Mach Learn Res 2:67–93

30. Sugiyama M (2012) On kernel parameter selection in Hilbert-Schmidt independence criterion. IEICE Trans Inf Syst E95-D(10):2564–2567
31. Lu Y, Wang L, Lu J, Yang J, Shen C (2014) Multiple kernel clustering based on centered kernel alignment. Pattern Recognit 47(11):3656–3664
32. Neumann J, Schnörr C, Steidl G (2005) Combined SVM-based feature selection and classification. Mach Learn 61(1–3):129–150
33. Lichman M (2013) UCI machine learning repository. Irvine, CA: University of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml/
34. Chang CC, Lin CJ (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2(3):27. http://www.csie.ntu.edu.tw/~cjlin/libsvm
35. Hsu CW, Lin CJ (2002) A comparison of methods for multiclass support vector machines. IEEE Trans Neural Netw 13(2):415–425
36. Chen PH, Lin CJ, Chölkopf B (2005) A tutorial on—support vector machines. Appl Stoch Models Bus Ind 21(2):111–136
37. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
38. Chen Z, Haykin S (2002) On different facets of regularization theory. Neural Comput 14(12):2791–2846
39. Liu P, Huang Y, Meng L, Gong S, Zhang G (2016) Two-stage extreme learning machine for high-dimensional data. Int J Mach Learn Cybern 7(5):765–772
40. Chen C, Zhang J, He X, Zhou ZH (2012) Non-parametric kernel learning with robust pairwise constraints. Int J Mach Learn Cybern 3(2):83–96
41. Lin CF, Wang SD (2002) Fuzzy support vector machines. IEEE Trans Neural Netw 13(2):464–471
42. Yamada M, Jitkrittum W, Sigal L, Xing EP, Sugiyama M (2014) High-dimensional feature selection by feature-wise kernelized Lasso. Neural Comput 26(1):185–207