1

SURVEY[1]

1.1 INTRODUCTION OF KERNEL ANALYSIS

Kernel methods have been widely studied for pattern classification and multidomain association tasks [1–3]. Kernel analysis (KA) enables kernel functions to operate in the feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space [4, 5]. This operation is often less computational than the explicit computation of the coordinates [3, 6, 7]. This approach is called the kernel trick. Kernel functions have been introduced for sequence data, graphs, text, images, as well as vectors [8–14].

Kernel feature analysis attracts significant attention in the fields of both machine learning and signal processing [10, 15]; thus there are demands to cover this state-of-the-art topic [16]. In this survey, we identify the following popular trends and developments in KA, so that we can visualize the merits and potentials in an organized manner:

- Yield nonlinear filters in the input space to open up many possibilities for optimum nonlinear system design.
- Adapt KA into the traditionally developed machine learning techniques for nonlinear optimal filter implementations.
- Explore kernel selection for distributed databases including solutions of heterogeneous issues.

[1]This chapter is a revised version of the author's paper in IEEE Transactions on Neural Networks and Learning Systems. DOI: 10.1109/TNNLS.2014.2333664, approved by IEEE Intellectual Property Rights.

Data-Variant Kernel Analysis, First Edition. Yuichi Motai.

Constructing composite kernels is an anticipated solution for heterogeneous data problems. A composite kernel is more relevant for the dataset and adapts itself by adjusting its composed coefficient parameters, thus allowing more flexibility in the kernel choice [3, 8, 17–20].

The key idea behind the KA method is to allow the feature space to be updated as the training proceeds with more data being fed into the algorithm [15, 21–26]. This feature space update can be incremental or nonincremental. In an incremental update, the feature space is augmented with new features extracted from the new data, with a possible expansion of the feature space if necessary [21–27]. In a nonincremental update, the dimension of the feature space remains constant, as the newly computed features may replace some of the existing ones [8, 19, 20]. In this survey, we also identify the following possibilities:

- A link between offline learning and online learning using KA framework, which suggests other connections and a potential impact on both machine learning and signal processing.
- A relationship between online learning and prediction techniques to merge them together for an adaptive prediction from online learning.
- An online novelty detection with KA as an extended application of prediction algorithms from online learning. These algorithms listed in this survey are capable of operating with kernels, including support vector machines (SVMs) [12, 28–34] Gaussian processes [35–38], Fisher's linear discriminant analysis (LDA) [19, 39], principal component analysis (PCA) [3, 9–11, 20, 22, 24–27, 40], spectral clustering [41–47], linear adaptive filters [48–53], and many others.

The objective of this survey is to deliver a comprehensive review of current KA methods from offline and online learning aspects [1, 19, 21]. Research studies on KA are carried out in the areas of computer science or statistics to give precise treatments to nonlinear pattern recognition without unnecessary computational complexity [4, 5, 28]. This latest survey of KA may stimulate the emergence of neural network studies in computer science applications and encourage collaborative research activities in relevant engineering areas [8–12].

We begin to describe the offline kernel methods by basic understanding of KA in Section 1.2. Then, this survey shows advanced distributed database technology for KA in Section 1.3. In the following sections, we point out online learning frameworks (vs offline learning in Section 1.2) using kernel methods in Section 1.4 and then the prediction of data anomaly in Section 1.5. Finally, we conclude with the future directions of KA in Section 1.6.

1.2 KERNEL OFFLINE LEARNING

Offline learning is a machine learning algorithm in which the system does not change its approximation of the target function, once the initial training phase has been absolved.

KA mainly deals with two major issues: (i) how to choose the appropriate kernels for offline learning during the learning phase, and (ii) how to adopt KA into the traditionally developed machine learning techniques such as neural network (NN), SVM, and PCA, where the nonlinear learning data space is placed under the linear space via kernel tricks.

1.2.1 Choose the Appropriate Kernels

Selecting the appropriate kernels is regarded as the key problem in KA, as it has a direct effect on the performance of machine learning techniques [35, 54–65]. The proper kernels are identified by *priori* without the analysis of databases, and the kernel selection is often implemented in the initial stage of offline learning. Commonly used kernel functions are as follows [20, 56, 66, 67]:

The linear kernel: $K(x, x_i) = x^T x_i$
The polynomial kernel: $K(x, x_i) = (x^T x_i + \text{Offset})^d$
The Gaussian radial basis function (RBF) kernel: $K(x, x_i) = \exp(-\|x - x_i\|^2/2\sigma^2)$
The exponential Gaussian RBF kernel: $K(x, x_i) = \exp(-\|x - x_i\|/2\sigma^2)$
The Laplace kernel: $K(x, x_i) = \exp(-\sqrt{\|x - x_i\|^2/\sigma^2})$
The Laplace RBF kernel: $K(x, x_i) = \exp(-\sigma\|x - x_i\|)$
The sigmoid kernel: $K(x, x_i) = \tanh(\beta_0 x^T x_i + \beta_1)$
The analysis of variance (ANOVA) RB kernel: $K(x, x_i) = \sum_{k=1}^{n} \exp(-\sigma(x^k - x_i^k)^2)^d$
The Cauchy kernel: $K(x, x_i) = 1/(1 + \|x - x_i\|^2/\sigma)$
The multiquadric kernel: $K(x, x_i) = \sqrt{\|x - x_i\|^2 + \sigma^2}$
The power exponential kernel: $K(x, x_i) = \exp((-\|x - x_i\|^2/2\sigma^2)^d)$
The linear spline kernel: $K(x, x_i) = 1 + xx_i \min(x, x_i) - ((x + x_i)/2)(\min(x, x_i)^2 + (\min(x, x_i)^3/3))$.

Many KA studies do not address appropriate kernel functions; instead, they simply choose well-known traditional kernels empirically by following other studies [56, 57, 68–71].

As shown in Table 1.1, there are four proposed kernel methods for choosing proper kernels: (i) linear combinations of base kernel functions, (ii) hyperkernels, (iii) difference of convex (DC) functions programming, and (iv) convex optimization. The representative accuracy is calculated as the average of the maximum accuracy with its standard deviation for each kernel selection approach [(i)–(iv)], according to the papers listed. On the basis of Table 1.1, the DC approach consistently achieved the best accuracy, using a 95% level of confidence.

Linear Combinations of Base Kernel Functions The linear combination of base kernel functions approach is based on the fact that the combination of kernels satisfying the Mercer's condition can generate another kernel [73]. In other

Table 1.1 Kernel selection approaches and related research

Kernel selection approach	Relevant studies	Representative accuracy
Linear combinations	[58, 60, 72, 73]	93.6 ± 8.4
Hyperkernels	[35, 57, 61–63, 74]	95.9 ± 4.3
Difference of convex (DC) functions	[36, 64, 75, 76]	99.1 ± 0.76
Convex optimization	[69, 71, 77, 81]	96.5 ± 4.1

words, the appropriate kernel is chosen by estimating the optimal weighted combination of base kernels [56, 67], to satisfy the Mercer's condition $\int\int K(x, x_i)g(x)g(x_i)\,dxdx_i \geq 0$.

The proper kernels [58, 59, 68, 72, 73] are decided when base kernels $k_1, \ldots, k_D$ are given from Equation 1.1 [60, 73]:

$$K(x, \overline{x}) = \sum_{d=1}^{D} w_d k_d(x, \overline{x})K \tag{1.1}$$

where D is the number of base kernels, and each kernel function $k_d : \chi \times \chi \rightarrow \mathbf{R}$ has two inputs and produces a scalar as the output. In addition, $w_1, \ldots, w_D$ are non-negative weights [60, 73]. As shown in Equation 1.1, this method is effective because weights w_d of useless kernels k_d are ideally set to zero.

Hyperkernels Hyperkernels are proposed by Ong et al. [61]; it can be used for kernel learning for the inductive setting [57]. The definition of hyperkernels is as follows [35, 57, 61–63, 74]:

Let x be a nonempty set, $\widehat{x} = x \times x$ and $K : x \times x \rightarrow R$ with $K_x(\cdot) = K(\widehat{\chi}, \cdot) = K(\cdot, \widehat{\chi})$. Then, K is called a hyperkernel on x if and only if

1. K is a positive definite on $\widehat{x}$ and
2. For any $\widehat{\chi} \in \widehat{x}$, K_x is a positive definite on x.

These hyperkernels are flexible in kernel learning for a vast range of data sets, as they have the invariant property of translation and rotation simultaneously [63]. In other studies [35, 57, 62, 63, 74], these hyperkernels are extended into a variety of aspects as in Table 1.2.

These hyperkernel approaches listed in Table 1.2 have somehow overlapped the principles and advantages. These extensions include a method using customized optimization for measuring the fitness between a kernel and the learning task.

DC Functions-Programming Algorithm The kernel is selected by specifying its objective function [58, 61, 76–80] by optimizing a predefined family of kernels. DC-programming is one of the algorithms for those optimization problems, expressed

Table 1.2 Hyperkernel method comparison

Method	Principle	Advantage
Iterative optimization [57]	Second-order cone programming	Efficient calculation
Invariance [63]	Invariant translation and rotation simultaneously	Flexible wide range of data sets
Learning [62]	Learning hyperplane	Optimal parameters
Gaussian [35]	Wishart hyperkernels	Less computational complexity
Structural risk minimization [74]	Regularization penalty into the optimization	Low bias, high variance, prevent overfitting

as Equation 1.2 [64, 75]:

$$\begin{aligned} &\min D(\rho) = \min\left(g(\rho) - h(\rho)\right), \\ &g(\rho) = \frac{1}{n_1^2 \mathbf{1}^T \mathbf{K}_{1,1} \mathbf{1}} + \frac{1}{n_2^2 \mathbf{1}^T \mathbf{K}_{2,2} \mathbf{1}} \text{ and } h(\rho) = \frac{2}{n_1 n_2 \mathbf{1}^T \mathbf{K}_{1,2} \mathbf{1}} \end{aligned} \tag{1.2}$$

where $g(\rho)$ and $h(\rho)$ are convex functions, $\mathbf{1} = \{1, ..1\}^T$, n is the number of data, and $[\mathbf{K}]_{i,j} = k(x_i, x_j)$ for either class 1 or 2. This means the difference of convex functions $D(\rho)$ can be a nonconvex function. Thus, the DC-programming algorithm can be applied to both nonconvex and convex problems [64, 68, 75, 36].

To solve the nonconvex problem, Neumann et al. [75] adopt DC functions, which minimize the difference between convex functions as the kernel choice method. Argyriou et al. [68] propose the DC-programming algorithm on the basis of mini-max optimization problems involved in the greedy algorithm. This study [68] has a significant advantage, as choosing the kernel among basic kernels in advance is not required.

Convex Optimization Algorithm The problem for choosing the kernel can be reformulated as a manageable, convex optimization problem [69, 71, 77, 81]. Kim et al. [69] and Khemchandani et al. [71] address the kernel selection in the kernel Fisher discriminant analysis (KFDA) to maximize a Fisher discriminant ratio (FDR). The basic concept of adopting convex optimization is Equation 1.3 as follows [69, 71]:

$$\max F_{\lambda}^{*}(K) \tag{1.3}$$

subject to $K = \theta K_1 + (1 - \theta) K_2$, where kernel functions ${}^{\exists}K_1, {}^{\exists}K_2 \in \kappa$, *and* $\forall \theta \in [0,\ 1]$. F_{λ}^{*} is FDR with a positive regularization parameter λ, and κ is a set of kernel functions K.

Other kernel-based methods select the kernel optimizing over the Gram matrix with a fixed weight vector, which has similar computational complexity but cannot ensure that the globally optimal kernel is always found. In the case of the convex, the global optimization is achieved [69].

1.2.2 Adopt KA into the traditionally developed machine learning techniques

KA methods are used for the nonlinear extension to make it more applicable in addition to the existing offline methods, such as (i) neural network, (ii) SVM, and (iii) PCA.

Neural Network NN is a very successful technique, which uses input/output components that are trained to perform nonlinear classification [82]. This structured approach allows KA to be incorporated within NN in different ways [37, 83–96].

Some studies [37, 83–87] embed KA inside of the normal NN structure. A common example of combining the NN with KA is the radial basis function neural network (RBF-NN) [97]. This technique successfully uses a RBF kernel algorithm for the activation functions of the hidden layer in the form (1.4).

$$\eta(x, w) = \sum_{i=1}^{S1} \kappa_i(\|x - x_i\|) \tag{1.4}$$

where η is output, x is input with the hidden layer κ (Fig. 1.1).

Other nontraditional KA used in the general structure of the NN can be found in [88–92]. Although commonly standalone, NN techniques such as back-propagation and recurrence functions combine to train and expand the use of NN with KA in the literature [85, 86, 93–96]. Table 1.3 gives key references of back-propagation, recurrence, RBF, and some nontraditional techniques in conjunction with KA. On the basis of Table 1.3, many nontraditional approaches are proposed for reaching the best classification accuracy of each study listed in the relevant studies. The representative accuracy consists of the mean with 1 standard deviation among these relevant studies.

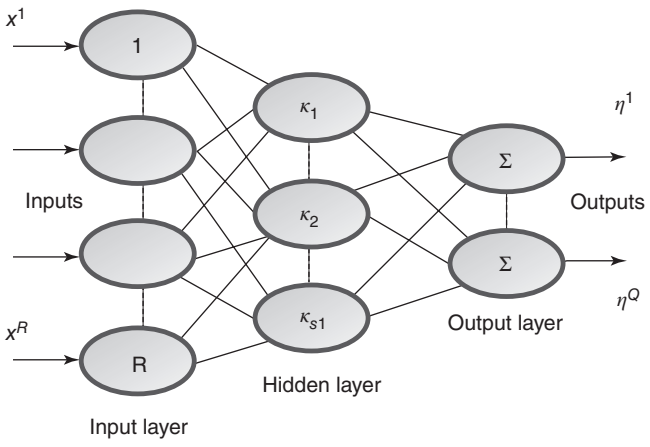

Figure 1.1 General RBF-NN [97]. The input and output layers perform functions similarly to a normal neural network. The hidden layer uses the RBF kernel, denoted as its activation function as shown in Equation 1.4.

Table 1.3 Neural network approaches related to kernel algorithm

Algorithms	Relevant studies	Representative accuracy
Back-propagation	[85, 86, 95, 96]	83.6 ± 10.8
Recurrence	[93, 94]	91.6 ± 10.5
Radial basis function (RBF)	[37, 83, 84, 86, 87]	90.0 ± 2.5
Nontraditional	[88–92]	96.7 ± 4.6

A comparison between KA and NN studies are conducted. Some NN [86, 94, 95] outperform non-KA-based NN. In the comparison of accuracy, more common KA techniques outperform standalone NN [98–101]. None of these references provides definitive evidence that KA or NN is generally better for all situations.

Support Vector Machine SVM originates from statistical learning theory developed by Vapnik [102], as a binary classifier. SVM is developed as a margin classifier [103]. The task in SVM is to find the largest margin separating the classifier hyperplane [104]. Equation 1.5 is an optimization problem that involves the cross-product of training data.

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{1.5}$$

where $\mathbf{x}$ is the input, α is a Lagrange multiplier, and y is the label for the distance criterion L. Equation 1.5 is a formulation of SVM for linearly separable data. The cross-product is also called linear kernel [105]. In order for SVM to work in non-separable (i.e., nonlinear) data, the data is mapped into a higher dimensional feature space, defined by the kernel function [52]. Kernel function is the inner product in the feature space and in the form of Equation 1.6:

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) \tag{1.6}$$

Tolerance to error needs to be introduced. The SVM learning task then becomes to minimize (Eq. 1.7),

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \tag{1.7}$$

subject to $0 \leq \alpha_i \leq C$ and $\sum \alpha y$ is a measure of tolerance to error. C is an additional constraint on the Lagrange multiplier. Equation 1.7 is a quadratic programming problem and can be solved using standard Quadratic Programming solver, called the C-SVM formulation [106]. A review on different methods to solve the optimization problem is discussed in [107]. Sequential minimal optimization (SMO) and other decomposition methods are used to solve the optimization problems listed in Table 1.4. This table shows SVM approaches consistently reach high classification accuracy.

Table 1.4 SVM approaches relevant to solve optimization problem

Algorithm	Relevant studies	Representative accuracy
Sequential minimal optimization (SMO)	[108, 109]	N.A.
Decomposition methods	[110, 111]	99.9 ± 0.0
Other methods	[112, 113]	92.3 ± 10.1

Among kernel functions used for SVM [37, 114], the three most common kernel functions used are polynomial, Gaussian, and sigmoid functions. SVM shows competitive performance for classification but does not outperform other techniques in regression problems [115]. Recent SVM developments introduce a new variant called twin support vector machine (TSVM) [116, 117], which uses nonparallel separating hyperplane unlike original formulation of SVM. Nonparallel hyperplanes are beneficial for preferential classification task, where one class is more significant than the others.

Principal Component Analysis PCA, also known as Karhunen–Loève transform (KLT) or proper orthogonal decomposition (POD), is a mathematical method for reducing dimension of feature space by representing in the orthogonal eigenvector space as shown in Fig. 1.2.

The transformation incorporates an eigenvector and eigenvalue calculation and is a linear transformation. The transformed space, in descending order of accounted variance, is orthogonal if the original dataset is jointly normally distributed. PCA is sensitive to relative scaling and expected value of the original feature. Being a linear model, PCA efficiently uses the nonlinearity provided by kernel association [118] in the form of higher dimension—making kernel principal component analysis (KPCA) a nonlinear component analysis tool. Bouveyron et al. [41] conduct an evaluation of PCA and KPCA in comparison to other similar methods for high-dimensional data clustering. In [119] PCA is generalized to use kernel for higher dimension input space. The developed supervised PCA as well as the supervised KPCA aims to find the

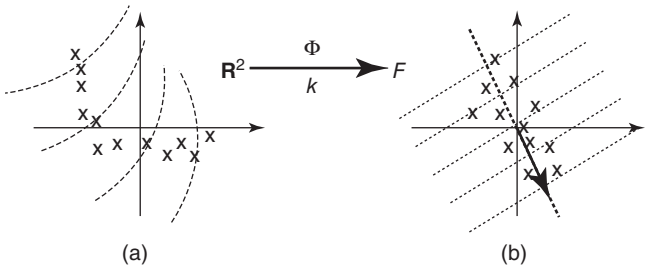

Figure 1.2 Principle of kernel PCA. (a) nonlinear in input space. (b) high-dimensional feature space corresponding to nonlinear input space. The contour lines of constant projections generate the principal eigenvectors. Kernel PCA does not actually require Φ to be mapped into F. All necessary computations are executed through the use of a kernel function k in input space $\mathbf{R}^2$.

principal components with maximum dependence on the response variables, which can be achieved by Equation 1.8.

$$\begin{aligned} &\underset{U}{\text{argmax}} \quad \text{tr}(U^T XHLHX^T U) \\ &\text{subject to} \quad U^T U = \mathbf{I} \end{aligned} \tag{1.8}$$

where U is the optimal solution space, X is the input set, H is the Hilbert–Schimdt space, and L is the space orthogonal to U.

Schölkopf et al. [118] have introduced KA into PCA. Yang et al. [120] extend Fisher discriminant analysis to associate kernel methods by incorporating KPCA and Fisher's LDA. In the LDA algorithm, firstly, the feature space is transformed by KPCA, then between-class and within-class scatter matrices S_b and S_w are computed. Finally, regular and irregular discriminant features are extracted and fused using a cumulative normalized-distance for classification.

Girolami [42] uses KPCA to develop one of the early iterative data clustering algorithms that uses KPCA. The implicit assumption of hyperspherical clusters in the sum-of-squares criterion is based on the feature space representation of the data, defined by the specific kernel chosen. Proposed generalized transform of scatter matrix is given by Equation 1.9.

$$Tr(S_W^{\theta}) = 1 - \sum_{k=1}^{\mathrm{K}} \gamma_k \mathfrak{R}(x|C_k) \tag{1.9}$$

where $\mathfrak{R}(x|C_k)$ denotes the quadratic sum of the elements, which are allocated to the kth cluster and $\gamma_k = N_k/N$.

Liu [121] applied an extension of KPCA on Gabor-wavelet representation of face images for recognition. He uses a subset of fractional power polynomial models, which satisfies Mercer condition to compute the corresponding Gram matrix. Lemm et al. [122] used KPCA in brain imaging and makes an analytic comparison with LDA. KPCA is also used for feature space dimension reduction in biodiesel analysis [123], characterization of global germplasm [124], and bioprocess monitoring for its derivatives [125]. Zhang and Ma [126] use multiscale KPCA for fault detection in nonlinear chemical processes. Yu develops a Gaussian mixture model for fault detection and diagnosis of nonlinear chemical processes [38]. These studies are listed in Table 1.5. All these studies are shown in Fig. 1.3, with the publication year and number of citations. Table 1.5 shows that PCA approaches cover many applications, with relatively high performances.

1.2.3 Structured Database with Kernel

Structured databases pose an interesting extension of offline kernel learning. Kernel methods in general have been successful in various learning tasks on numerical (at least ordered) data represented in a single table. Much of "real-world" data, however, is structured—has no natural representation in a single table [137]. Usually, to apply kernel methods to "real-world" data, extensive preprocessing is performed to

Table 1.5 PCA approaches related to kernel algorithm

Field of application	Relevant studies	Representative accuracy
RADAR detection/prediction	[123, 128, 131]	94.6 ± 7.5
Sensor/fault detection	[125, 126, 130, 132]	96.1 ± 5.4
Prediction analysis/algorithm	[38, 41, 42, 119, 133, 135]	96.7 ± 3.4
Computer vision/biometrics	[120–122, 124, 127, 134]	99.1 ± 1.2
Geography/satellite	[129]	88.0

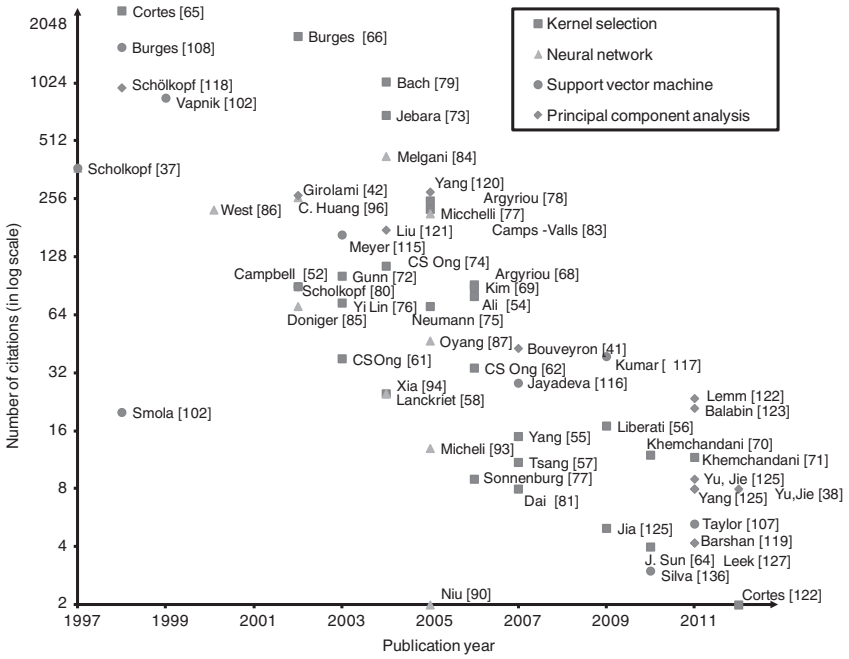

Figure 1.3 Significant research publications in various fields of KA and kernel selection for offline learning. Works related to neural network, SVM, PCA, and kernel selection are displayed according to citation number versus publishing year.

embed the data into a real vector space and thus in a single table. These preprocessing steps vary widely with the nature of the corresponding data. Several approaches of defining positive definite kernels on structured instances exist. Under this condition, structured databases are handled in different manners by analytically optimized algorithms. We briefly discuss those five kernel methods respectively: (i) kernel partial least squares (KPLS) regression [138], (ii) kernel canonical correlation analysis (KCCA) [139], (iii) kernel-based orthogonal projections (KOP), (iv) kernel multivariate analysis (KMA), and (v) diffusion kernels (DK) [140]. For the interested readers, we also include a comprehensive list of recent significant works on kernel methods for structured databases in Table 1.6.

Table 1.6 Application of kernels for structured databases

Method	Field of application	Research works
KPLS [138]	Algorithm	[141–143]
	Signal processing	[144, 145]
	Pattern recognition/Classification	[145, 146]
KCCA [139, 147, 148]	Classification	[149, 150]
	Statistical analysis	[151, 153]
KOP [166, 167]	Data compression	[152, 154]
	Classification	[155–157]
KMA [158, 159]	Image analysis/Classification	[160, 161]
	Statistical analysis	[162, 163]
DK [140]	Computer vision/Graph analysis	[164, 165]

Kernel Partial Least Squares Regression A family of regularized least squares regression models in a Reproducing Kernel Hilbert Space is extended by the KPLS regression model [138]. Similarly to principal components regression (PCR), partial least square (PLS) is a method based on the projection of input (explanatory) variables to the latent variables (components). However, in contrast to PCR, PLS creates the components by modeling the relationship between input and output variables while maintaining most of the information in the input variables. PLS is useful in situations where the number of explanatory variables exceeds the number of observations and/or a high level of multicollinearity among those variables is assumed.

Kernel Canonical Correlation Analysis (KCCA) Lai et al. propose a neural implementation of the statistical technique of canonical correlation analysis (CCA) [147] and extend it to nonlinear CCA. They derive the method of kernel-based CCA and compare these two methods on real and artificial data sets before using both on the blind separation of sources [139]. Hardoon et al. develop a general method using kernel canonical correlation analysis to learn a semantic representation to web images and their associated text. The semantic space provides a common representation and enables a comparison between the text and images [147]. Melzer et al. [148] introduce a new approach to constructing appearance models on the basis of KCCA.

Kernel-based Orthogonal Projections (KOP) Kernel-based classification and regression methods have been applied to modeling a wide variety of biological data. The kernel-based orthogonal projections to latent structures (K-OPLS) method offers unique properties facilitating separate modeling of predictive variation and structured noise in the feature space [166]. While providing prediction results similar to other kernel-based methods, K-OPLS features enhance interpretational capabilities—allowing detection of unanticipated systematic variation in the data such as instrumental drift, batch variability, or unexpected biological variation.

The orthogonal projections to latent structures (OPLS) method have been successfully applied in various chemical and biological systems as well for modeling and interpretation of linear relationships between a descriptor matrix and

response matrix. A kernel-based reformulation of the original OPLS algorithm is presented where the kernel Gram matrix is used as a replacement for the descriptor matrix. This enables usage of the "kernel trick" to efficiently transform the data into a higher dimensional feature space where predictive and response-orthogonal components are calculated. This strategy has the capacity to considerably improve predictive performance, where strong nonlinear relationships exist between descriptor and response variables with retaining the OPLS model framework. The algorithm enables separate modeling of predictive and response-orthogonal variation in the feature space. This separation can be highly beneficial for model interpretation purposes, which provides a flexible framework for supervised regression [167].

Kernel Multivariate Analysis (KMA) Feature extraction and dimensionality reduction are important tasks in many fields of science dealing with signal processing and analysis. The relevance of these techniques is increasing as current sensory devices are developed with ever higher resolution, and problems involving multimodal data sources become more common. A plethora of feature extraction methods are available in the literature collectively grouped under the field of multivariate analysis (MVA) [158]. Other methods for classification and statistical dependence deal with the extreme cases of large-scale and low-sized problems. Aitchison and Aitken [159] develop an extension of the kernel method of density estimation from continuous to multivariate binary spaces. This simple nonparametric has consistent properties in discrimination problems, with some advantages over already proposed parametric counterparts.

Diffusion Kernels (DK) DK may be viewed as a distinguished example of exponential kernels [140]. These applications of DK-based learning algorithms are applicable to real valued data and a few special data types, such as strings. Kondor and Lafferty [140] propose a general method of constructing natural families of kernels over discrete structures on the basis of the matrix exponentiation idea.

1.3 DISTRIBUTED DATABASE WITH KERNEL

Various KA using offline learning may be extended to address big data by considering distributed databases. In this section, we show how KA adaptation can benefit distributed databases by the procedure described in the following three subsections: Sections 1.3.1–1.3.3.

1.3.1 Multiple Database Representation

Providing enhanced multiple databases to users has received a lot of attention in the field of data engineering [43, 44, 168–172]. As shown in Fig. 1.4, the multiple databases consist of various datasets in the forms of distributed databases and single databases. They are sometimes separately stored on multiple storage spaces in a network [169, 170, 173].

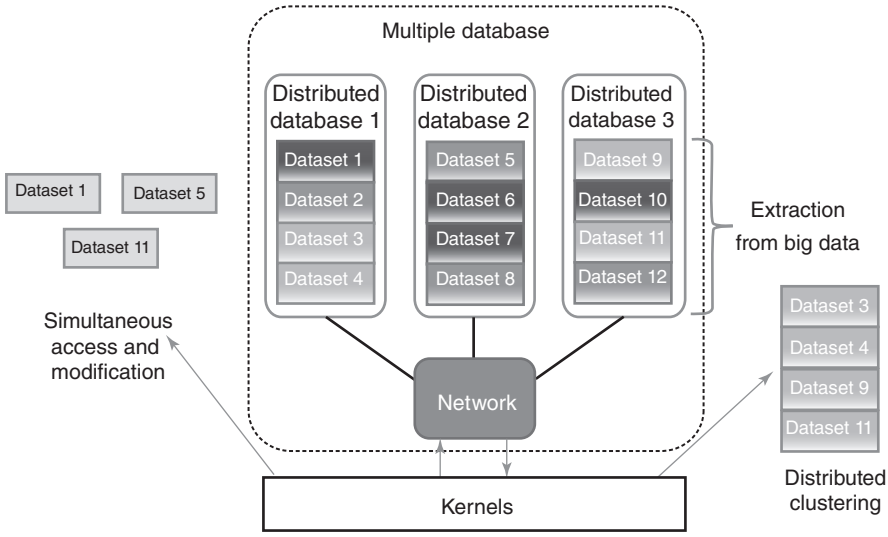

Figure 1.4 The composition of the multiple databases. The multiple databases include distributed databases and single databases, which are stored on different storage spaces connected within a network. Three key attributes of the multiple databases are (i) simultaneous access and modification, (ii) extraction from huge amounts, and (iii) distributed clustering/classification.

The multiple databases have the following three key attributes. Firstly, users of multiple databases can access and modify data in the distributed database simultaneously [169, 170]. Secondly, they can extract what they need among a huge amount of data in the distributed environments. Thirdly, all data stored in multiple databases can be clustered/classified [43, 44, 171, 172].

In order to bolster these key attributes of the multiple database, the data needs to be efficiently processed in separated storage spaces, to be classified, or to be gathered accurately per the requests of users [43, 44, 171, 172, 174]. Hence, the studies of multiple databases can be linked to multiple-learning or meta-learning, clustering, and classification problems. KA-based machine learning can be effective solutions of those issues due to KA flexibility [54, 70, 169].

1.3.2 Kernel Selections among Heterogeneous Multiple Databases

Sets of KA can be used to sort distributed databases for quick and efficient access. Lin shows that Kernel SVM can be used to decrease the total loading time of large distributed datasets at the cost of increased processing [175–178]. This is important because the data may not be accessible otherwise. The reason for the increased computational load is that the Kernel SVM must be retrained with each new dataset [175]. That is why kernels themselves are distributed and the computational power is divided. An early method of such distributed KA is described in [179]. Other more

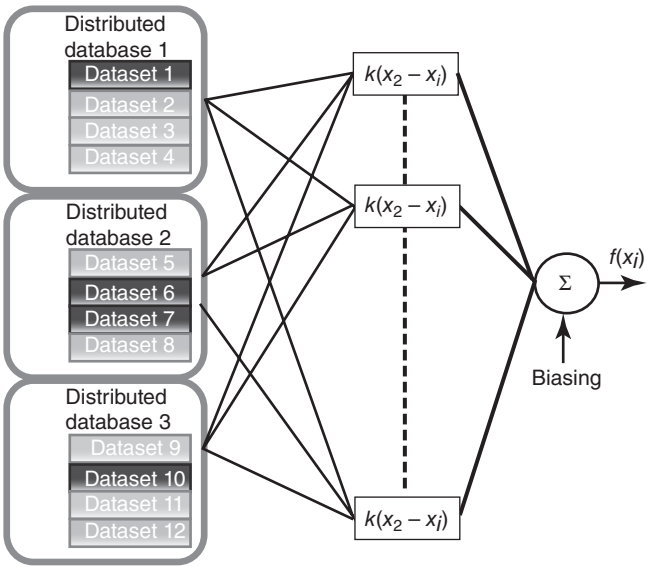

Figure 1.5 Several pregenerated kernels are trained to access different features in several distributed databases. The output of this kernel network is a sum of the kernel functions that maps to the data of interest. The summation and biasing make this method behave similarly to a neural network [178].

recent methods can be seen in [16, 178, 180–183]. Fig. 1.5 gives an example where multiple kernels (MKs) can be used in a network setting, in which each kernel holds a different configuration of the dataset [178]. The non-kernel SVM has a faster run-time but less precision than that of KA distributed SVM [175, 184].

1.3.3 Multiple Database Representation KA Applications to Distributed Databases

KA is applied for a wide range of learning problems in distributed environment. Some examples of these applications include anomaly/fault detection, medical field/biometrics/computer vision, and economic modeling/forecasting/prediction. In Table 1.7, some relevant KA applications of distributed databases are listed. Table 1.7 shows that many SVM approaches exist for a broad range of applications.

We show specific applications as follows to demonstrate how KA distributed databases using SVM-related algorithms are practically implemented in the real-world setting: (i) market analysis, (ii) wireless sensor networks, and (iii) distributed text classification.

Market Analysis One field that will benefit from the distributed computing task is the study of market behavior because it involves large and diverse datasets from

Table 1.7 Distributed database application-wise concentration

Distributed database application field	NN	SVM	PCA
Anomaly/Fault detection	[43]	[48, 171, 176, 177, 182]	[173]
	100	90.2 ± 10.7	100
Medical field/Biometrics/ Computer vision	[170]	[136, 168, 172, 180, 184, 187]	[44, 45]
	96.2	98.1 ± 1.1	91 ± 8.5
Economic modeling/ Forecasting/Prediction	[169]	[16, 174, 178, 185, 186, 188]	[239]
	95	90.0 ± 10.7	99.2

various sources. In [186], customer behavior prediction is stored in separate places, so that the proposed machine learning technique can be implemented in a distributed manner. Multiple kernel support vector machines (MKSVM) and author-developed approach called collaborative MKSVM are compared. The MKSVM is a variant of SVM that uses MKs in computation, while Collaborative MKSVM is a derivation of MKSVM to accommodate the distributed and diverse manner of data stored in a market problem. The task of learning in this study is to classify customers (returning or reluctant) and their willingness (willing to repeatedly purchase some group of products or not). Experiment results show that Collaborative MKSVM obtains higher accuracy and lower computational time and is more robust to noise data than MKSVM.

Wireless Sensor Networks This study implements SVM in classification problem over distributed WSN [16] on the basis of a sequential ascent-based algorithm. This algorithm essentially works in a parallel way with a discriminant function to appropriately handle this distributed scheme. SVM is trained locally for each network node to get the Lagrange multiplier corresponding to the local dataset. This scheme tests the data from the UCI machine learning repository [189]. Results show no significant difference in effectiveness compared to the original SVM, but the scheme minimizes the exchange of information between networks (and hence increases security) and is scalable for large-scale network.

Distributed Text Classification Another implementation of SVM is applied to classification of text in distributed computing [136]. In addition to SVM, the Relevance Vector Machine is proposed. The texts data to be classified is obtained from Reuters-21578 financial corpus and Reuters Corpus Volume 1. This text classification task is split into subtasks that can be disposed in a directed acrylic graph (DAG). The DAG is then optimized for distributed computing [190]. This distributed computing framework shows that the performance of the classification task is better using distributed scheme compared to sequential processing.

1.4 KERNEL ONLINE LEARNING

In contrast to offline learning, online learning handles one instance at a time as new data becomes available. Online learning may be used as a faster, more efficient replacement for batch processing due to its lower computational and memory requirements, or it may be implemented in a real-time system [15]. Because new data is continuously used for training, online algorithms must make use of memory resources to prevent unbounded growth of the support of the classifier or regression curve. This results in unbounded memory consumption and growth in computational complexity [15, 191–193].

In Section 1.4.1, we illustrate general principles of kernel-based online learning algorithms, and in Section 1.4.2, we demonstrate how an online kernel framework is adopted into some traditionally developed machine learning techniques such as (i) NN, (ii) SVM, and (iii) PCA. Finally, in Section 1.4.3, we establish the link between online learning and prediction.

1.4.1 Kernel-Based Online Learning Algorithms

The kernel-based online learning is introduced by shifting kernel methods to an online format in the growth of the dictionary size over time as new data is added as shown in Fig. 1.6. This growth is super-linear when batch-type offline techniques are directly applied [194] and linear when incremental methods are applied to naïve application [49]. To remedy the data size, a variety of techniques are used to discard or ignore the data on whether any new information is added to the classifier or filter [15, 49].

The so-called "growing sum problem" [192] is tackled by a variety of means. The NORMA algorithm [15] uses a "forgetting factor" such that samples are weighted in a sliding-window manner, with older samples weighted less than newer ones. This technique assumes that we are interested only in the statistical structure of the newest samples in a nonstationary series.

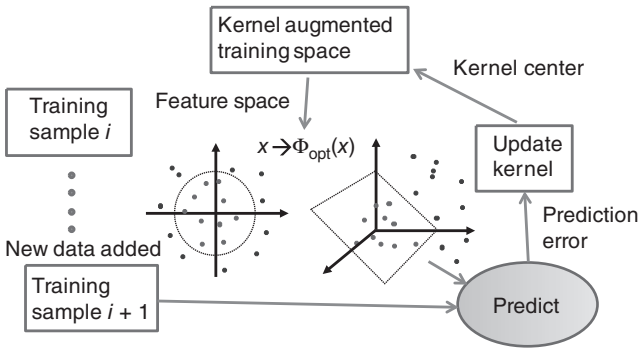

Figure 1.6 Online kernel-based learning algorithm. As a new training sample is added, the kernel is updated on the basis of the online learning algorithms described in the following section.

Alternatively, when it is desired to take older data into account, sparsification techniques may be used [49]. In these methods, only data that add new information to the classifier or filter are added to the support. Various information criteria are proposed, such as the approximate linear dependency criterion [195], the surprise criterion [50], and the variance criterion [196].

While sparsification methods seek to discard redundant data, quantization methods instead limit the number of possible input space regions that will lead to support expansion [197]. The input space is divided into discrete regions of a given size. The region expands the support only if it has no assigned support vector. Otherwise, the coefficient for that region is updated instead, which limits the growth of the system [197].

Other methods are proposed for dealing with the growth problem, including [192], which uses only finite-dimensional kernels and replaces infinite-dimensional kernels with finite-dimensional approximations. This method achieves constant computational complexity over time [192]. Another method is presented in [193], which projects new inputs into a feature space spanned by the previous hypothesis. This method is guaranteed to be bounded and outperforms other Perceptron-based methods [193].

The complication of online kernel methods is the selection of the kernel itself. Often, it is assumed that a suitable function is known as *a priori*, while in reality, this is not necessarily the case [49, 198]. This issue is less well addressed in the literature but is handled through the use of MK algorithms, which select the best kernel or combination of kernels from a dictionary of possible choices on the basis of the data [198, 199]. Rosenblatt [200] uses a combination of the Perceptron algorithm and the Hedge algorithm [191] to learn the kernel combination and the classifier and present both deterministic and stochastic [201] algorithms for doing so.

1.4.2 Adopt "Online" KA Framework into the Traditionally Developed Machine Learning Techniques

Figure 1.7 shows the representative online learning studies on Kernel-based online learning using (i) NN, (ii) SVM, and (iii) PCA.

Neural Network (NN) Online NN is designed to have the learning capabilities of online time systems [202]. Online NN also consists of input, hidden, and output layers interconnected with directed weights (w). w_{ij} denotes the input-to-hidden layer weights at the hidden neuron j, and w_{jk} denotes the hidden-to-output layer weights at the output neuron k. Learning is accomplished by comparing the actual output to the desired output, then adjusting the weights accordingly. A typical kernel online NN is depicted in Fig. 1.8.

Online updates to the weights of a NN relies on algorithms such as the commonly used Gaussian RBF, which resides in the hidden layer of the NN. The centers of the RBF function are initially chosen on the basis of prior knowledge of the structure of the uncertainty, and typically, we assume that the RBF centers are fixed. The application of kernel methods allows this assumption to be relaxed [219].

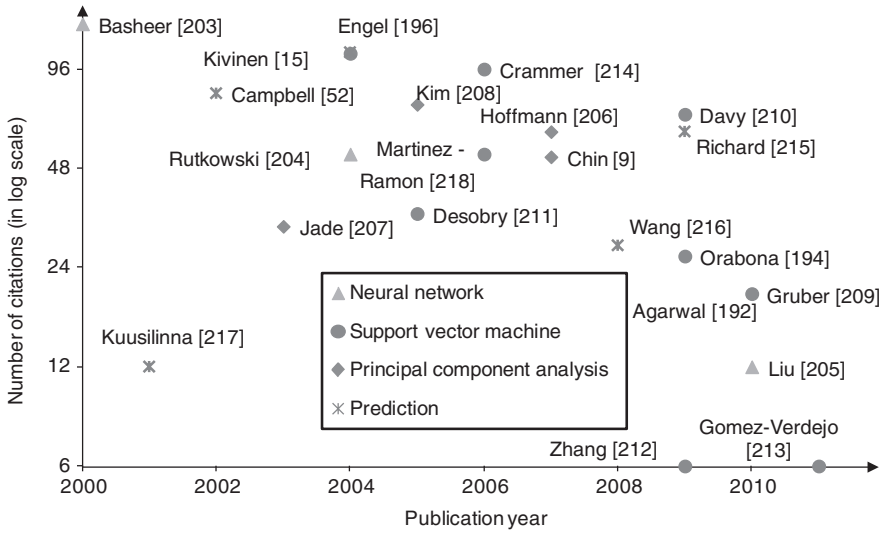

Figure 1.7 Key papers for Kernel Online learning based on the number of citations for the years 1998–2013, for Principal Component Analysis (PCA), Support Vector Machine (SVM), Neural Network (NN), Autoregressive Moving Average (ARMA), and Finite-State Model (FSM).

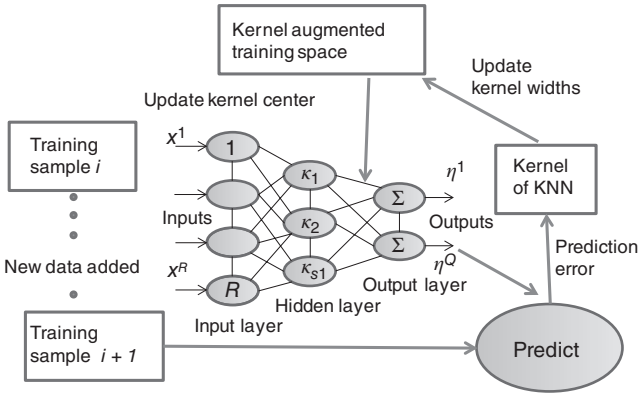

Figure 1.8 Online NN learning algorithm. An artificial neural network consists of input, hidden, and output layers interconnected with directed weights (w), where w_{ij} denotes the input-to-hidden layer weights at the hidden neuron j and w_{jk} denotes the hidden-to-output layer weights at the output neuron k [218].

Kingravi et al. propose an online algorithm that connects kernel methods to persistently exciting (PE) signals, known as budgeted kernel restructuring (BKR) for model reference adaptive control (MRAC).

Similar techniques have been applied to such problems as time-series prediction [203], and pattern classification tasks such as facial recognition [15]. NN is frequently

applied in engineering applications that require optimization of nonlinear stochastic dynamic systems [220]. Yuen et al. [218] propose a kernel-based hybrid NN for online regression of noisy data, combining fuzzy ART (FA) and general regression neural network (GRNN) models. The performance of kernel NN continues to improve due to the recent development of new algorithms; Liu et al. [204] describe kernel affine projection algorithms (KAPA), which outperforms other recently developed algorithms such as the kernel least-mean-square (KLMS) algorithm in online time-series prediction, nonlinear channel equalization, and nonlinear noise cancellation.

Support Vector Machine (SVM) Online SVM [208, 209] preserves the skeleton samples on the basis of the geometric characteristics of SVM as shown in Fig. 1.9. The SVM classifier is updated when the distances between samples in the kernel space and the newly arriving samples are within a given threshold to the current classification hyperplane. Online updating of the classifier can be achieved, as only very limited training samples are maintained in the offline step [208].

The implementation of online SVM has two possible cases. One case is to replace batch processing when more than a feasible amount of memory is required to generate the classifier. The second case involves data arriving in real time, when the algorithm classifies data as it is received. In this case, we consider nonstationary aspects so that the classifier can adapt—that is, retrain—to the changing distribution of the input data. This illuminates the need for specially designed algorithms, as the training process for traditional batch SVMs is notoriously slow [15].

Gruber et al. adopt an online SVM to signature variation, using kernel functions based on the longest common subsequences (LCSS) detection algorithm. The model compensates local variability of individual signatures, performs more reliably than other SVM kernel algorithms such as dynamic time warping, and easily defeats them [208].

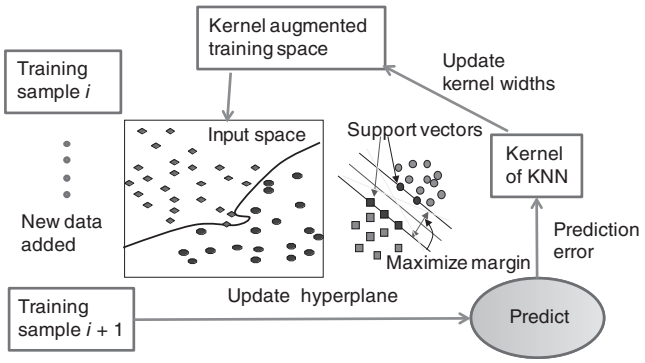

Figure 1.9 Scheme for naïve online SVM classification. The input data are mapped to the kernel Hilbert space, and the hyperplane is generated as in offline SVM. If an optimal solution cannot be found, a new kernel is selected. The process repeats for newly incoming data.

Online SVM is also widely used for novelty detection, known as anomaly detection. The novelty detection is the process when a learning system recognizes unprecedented inputs [209]. SVMs adapted to online novelty detection are referred to as one-class support vector machines (1-SVM). 1-SVM distinguishes between inputs in the "normal" class and all other inputs, which are considered outliers or anomalies. Because such a classification does not require expert labels, 1-SVMs are considered unsupervised methods. The classifying function of a 1-SVM returns $+1$ within a region capturing most of the data points, and -1 elsewhere in the input space [204]. This is accomplished by separating the data from the origin using the greatest possible margin [204, 221]. Similarly to other online SVMs, online 1-SVMs typically prune old or redundant data from the classifier to prevent indefinite complexity expansion [209].

Desobry et al. [210] propose an adaptive online 1-SVM method called kernel change detection (KCD), designed for the online case of the sequential input data. For T feature vectors $\boldsymbol{x}_t$, $t = 1, 2, ... T$, they train two 1-SVMs independently: one on the set of preceding feature vectors, and one on the set of future vectors. Using the outputs from each SVM, they compute a decision index and compare it to a predetermined threshold to ascertain whether a change has occurred in the feature statistics at the given t. They find that KCD is capable of segmenting musical tracks into their component notes.

Zhang et al. implement a sliding-time-window 1-SVM using quarter-sphere formulation of the SVM, which applies to one-sided non-negative distributions of features [211]. Rather than fitting a hyperplane to the data, this method fits a quarter-(hyper)sphere, centered at the origin [211]. This reduces the quadratic programming solution of the SVM dual problem to a linear programming solution, reducing the computational effort required for training at each time window.

Gomez-Verdejo et al. [212] create an adaptive one-class support vector achine (AOSVM) that updates the SVM at every time-step using a recursive least squares algorithm. AOSVM weights old patterns with an exponential window and allows them to decay so that the SVM can adapt to changing data statistics. Sofman et al. [222] use a modified version of the NORMA algorithm [15] for online novelty detection in the context of mobile robot terrain navigation. NORMA is a stochastic gradient descent algorithm suitable for online kernel-based learning. Sofman uses a hinge loss function, such that its gradient is nonzero only in the case of novel inputs.

Principal Component Analysis (PCA) Online KPCA extracts principal components in the feature space related to nonlinear inputs. The algorithm for extracting the components is Equation 1.10 expressed mathematically as

$$V^n \cdot \Phi(x) = \sum_{i=1}^{M} \alpha_i^n k(x_i, x) \tag{1.10}$$

where $k(x_i, x) = (\Phi(x_i) \cdot \Phi(x))$ is the kernel function, α_i the eigenvectors, and V^n the corresponding eigenvectors in the feature space [118]. The success of online KPCA depends heavily on the kernel updates from the modified Gram matrix; thus, the classification accuracy relies on kernel iteration for problem-specific datasets [240–242].

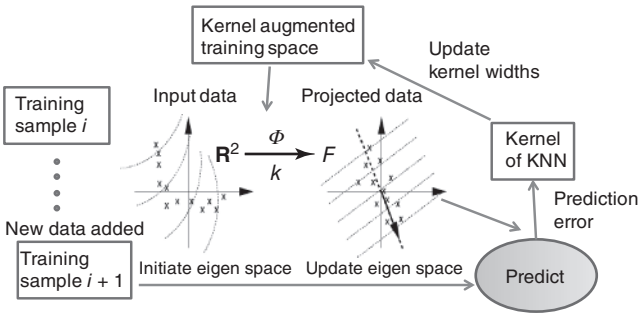

Figure 1.10 Online KPCA diagram. In the initial (offline) training phase, the initial kernel parameters and eigen-feature space are calculated. As new training sets are introduced, incremental (online) adjustments are made to the parameters and an updated eigen-space is generated.

Most online methods begin with one or more offline learning steps to determine suitable kernel parameters [223]. The incremental kernel principal component analysis (IKPCA) developed by Ozawa et al. [224] progresses in two stages: (i) in the initial learning phase, a number of training sets are presented offline and the initial eigen-feature space is found, and (ii) in the incremental learning phase, the feature space is defined by solving the eigenvalue problem with dimensions determined by the amount of independent data. Online KPCA algorithm is depicted in Fig. 1.10.

Hoffman [205] demonstrates the use of online KPCA in novelty detection, specifically applied to recognition of handwritten digits and breast cancer detection. Classification performance is measured against results from the Parzen window density estimator, standard PCA, and 1-SVM. By adjusting the parameters q (number of eigenvectors) and σ (kernel width), KPCA achieves better generalization than Parzen density and greater resistance to noise. Online KPCA methods are applied to the denoising of chaotic time series [206] and Gaussian noise in images [207].

1.4.3 Relationship Between Online Learning and Prediction Techniques

Prediction is a scheme to estimate the future behavior of a system. It is intimately connected with regression [52, 225] and system modeling [51, 214], each of which seeks to build a mathematical representation of an unknown system or process. This mathematical representation model can be subsequently used to predict future states.

Prediction and online learning have traditionally been studies of two independent communities: (i) in signal processing and (ii) in machine learning. The two disciplines possess different momentum and emphasis, which makes it attractive to periodically review trends and new developments in their overlapping spheres of influence. These two existing communities notice that this is the key area of study for the next generation of any intelligent systems [191, 214, 215].

Table 1.8 Comparisons of prediction and online learning algorithms

Methods	Purpose	Main community
Prediction	Estimate forthcoming outcomes	Statistics/signal processing
Online learning	Induce the coming new data for prediction	Statistics/machine leaning

As shown in Table 1.8, the term "prediction" implies learning how to forecast, that is, what will happen in the forthcoming data? Online learning, on the other hand, is a more general term, in which newly arriving data may be used to update a classifier without necessarily generating a prediction of the next data point. However, they are often used to mean the same thing, as knowledge of the structure of a system allows one to predict its future behavior [52, 225].

A prediction is a future function, as a forecast of the new coming dataset will be automatically classified on the basis of the knowledge of offline or online learning. As shown in the previous section, there are many prediction approaches. These predictions can be incorporated with KA representation, in a manner that may extend the learning space through the nonlinear online learning. In the following section, "prediction" will be examined, specifically how the coming datasets are used for testing a future feature space prediction [213, 214] as "online learning" has been shown.

1.5 PREDICTION WITH KERNELS

Prediction is represented by top-down model-based representation. In this section, we show how kernels may enhance the prediction through the four major representations: (i) linear prediction, (ii) Kalman filter, (iii) finite-state model, and (iv) autoregressive moving average model. Finally, we compare them in (v) comparison of four models.

1.5.1 Linear Prediction

A linear prediction is a simplified model where future output values are estimated as a linear function (Eq. 1.11) of previous values and predictor coefficients, as follows [226]:

$$\hat{x}(n) = a_0 + a_1 x(n-1) + \cdots + a_n x(n-k) = \sum_{i=0}^{k} a_i x(n-i) \tag{1.11}$$

where $\hat{x}(n)$ is the predicted value or position at n. By applying a kernel to the input data, the dataset can be mapped to a space where it is linear.

The predicted value is a linear combination of previous observations $x(n-k)$ and predictor coefficients a_n, as shown in Fig. 1.11. The key is to solve a linear equation to find out the coefficients, a_n, that can minimize the mean squared error between the predicted values and previous values [227]. The linear model is widely used in the early stage to compare the prediction performance with other models, for example, Kalman filtering [226].

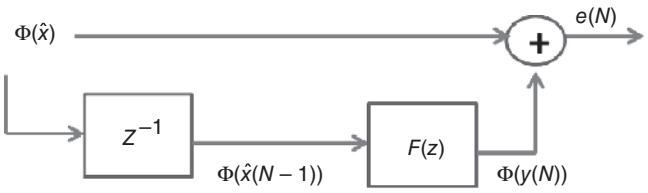

Figure 1.11 Linear predictor with an initial kernel mapping. The input data is mapped from its original space to a new Hilbert space where $x(n)$ is linear, by the function Φ to allow for the use of the kernel trick. The model is optimized by minimized the function $e(N)$, which is the difference in the current and previous predicted value.

1.5.2 Kalman Filter

The Kalman filter (KF) is one of the most commonly used prediction methods in real-time filtering technologies [53, 226, 228–233]. KF provides a recursive solution 1.12 to minimize mean square error within the class of linear estimators, where linear process and measurement equations can be expressed as follows [228]:

$$\widehat{x}(t) = Fx(t-1) + Bu(t-1) + W,\ z(t) = H\widehat{x}(t) + V \tag{1.12}$$

where we denote the state transition matrix as F, the control-input matrix as B, and the measurement matrix as H. $u(t)$ is an n-dimensional known vector, and $z(t)$ is a measurement vector. The random variables W and V represent the process and measurement noise with the property of the zero-mean white Gaussian noise with covariance, $E[W(t)W(t)^T] = R(t)$ and $E[\mathrm{V}(t)V(t)^T] = Q(t)$, respectively. The matrices F, B, W, H, and V are assumed known and possibly time varying.

Ralaivola and d' Alche-Buc [233] develop a kernel Kalman filter (KKF) method for implementing a KF on a nonlinear time-series dataset. By mapping the input data from a nonlinear space to a Hilbert space (shown in Fig. 1.12), the mercer kernel function can be satisfied. In KKF, the predicted position $\widehat{x}(t)$ can be derived from the previous state $x(t-1)$ and the current measurement $z(t)$ [226, 232]. Because of the state update kernel process with new data, KF is effective for predicting changes in both linear and nonlinear dynamic systems of multivariate Gaussian distribution [53].

KF can be enhanced to an interactive multiple model (IMM) filter with constant velocity (CV) and constant acceleration (CA) [232]. Hong et al. [53] suggest the first-order extended Kalman filter (EKF) with a kernel type method can be used to process and update the state estimate.

1.5.3 Finite-State Model

Finite-state model (FSM), or finite-state machine, consists of a finite number of discrete states, only one of which the model can occupy at any given time. FSM undergoes a change of state upon a triggering event. Kernel methods have been applied to FSM [234, 235]. Separated-kernel image processing using finite-state machines (SKIPSM) is an online kernel method used to speed processing time when dealing

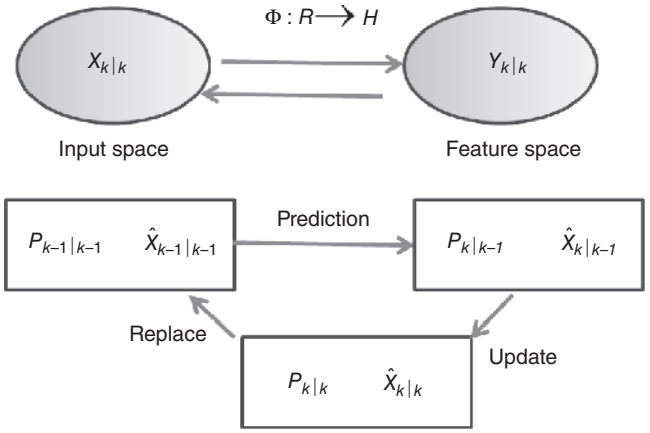

Figure 1.12 For processing a nonlinear time series, a kernelized Kalman filter is an optimal way to execute the Kalman filter. With a nonlinear series x_k and an associated series $x^{\Phi}{}_k$. x_k is mapped to $x^{\Phi}{}_k$ by the nonlinear map $\Phi: \ R^d \rightarrow H$, where R^d is the input space, H is the feature space, and $[\Phi(x), \ \Phi(y)] = k(x, y)$.

with large numbers of states, as in processing of grayscale images [234]. FSM is expressed mathematically as a sextuple $M = (I, \ O, \ S, \ \delta, \ \lambda, \ s0)$, where I is the input alphabet, O is the output alphabet, S is the finite set of states, and $s0$ is the initial state. δ is the next state function ($\delta : S \times I \rightarrow S$), and λ is the output function ($\lambda : S \times I \rightarrow O$). Most FSM are classified as either a Mealy machine, in which the inputs always affect the outputs, or a Moore machine, in which inputs affect outputs only through the states [216, 236].

1.5.4 Autoregressive Moving Average Model

Autoregressive moving average (ARMA) model is a mathematical generalization of the linear model with time-series data and signal noise and is widely used to predict motion patterns of a time series from past values. Martinez-Ramòn et al. [217] use the kernel trick to formulate nonlinear SVM-ARMA in reproducing kernel Hilbert space (RKHS). Discrete-time processes (DTP) that are nonlinear cannot be directly modeled using ARMA, but the use of composite kernels allows for a nonlinear mapping into an RKHS. The input and output DTP are represented by the composite kernel formulation 1.13:

$$\begin{aligned} K(z_i, z_j) &= F\langle [\phi_y(y_{i-1})^T, \phi_u(u_i)^T]^T, [\phi_y(y_{j-1})^T, \phi_u(u_j)^T]^T \rangle \\ &= K_y(y_{i-1}, y_{j-1}) + K_u(u_i, u_j) \end{aligned} \tag{1.13}$$

where $\boldsymbol{y}$ and $\boldsymbol{u}$ are the input and output DTP, respectively. Shpigelman et al. [237] describe a kernel autoregressive moving average (KARMA) for the tracking of hand movement in 3D space.

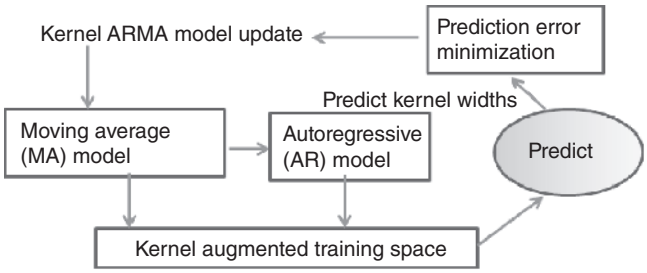

Figure 1.13 A kernel autoregressive moving average (KARMA) model with online learning for tracking of hand movement.

As shown in Fig. 1.13, ARMA consists of two models: (i) an autoregressive (AR) model represented by a weighted sum of the present and past positions with a polynomial order p, and (ii) a moving average (MA) model represented by a weighted sum of the present and past signal noise with a polynomial order q. The notation ARMA(p, q) is represented by the polynomial orders of p AR and q MA [237, 238].

1.5.5 Comparison of Four Models

The four prediction models, linear prediction, KF, FSM, and ARMA, are quantitively compared to illustrate the overall advantages of each model, as shown in Table 1.9. The prediction error is defined as root mean square error (RMSE $= \sqrt{(\Sigma(y_i - \hat{y}_i)^2 / \Sigma(y_i - m_y)^2)}$, where y_i is the ith measurement, $\hat{y}_i$ is the estimation of the ith measurement, and m_y is the mean of all the measurements). RMSE is used to convert each quantitive error to a universal metric for proper comparison. On the basis of these various conditions, we cannot definitively state which method performs better than the others. The performance of the computational speed is separately examined for the comparison by listing the primary factor contributing to model complexity.

These methods also list in Table 1.10 the computational speed to describe the overall computational complexity. In general, the number of datasets dictates the speed of each method, although some methods depend on additional experimental factors. For example, the main factors to determine the computational speed using Kernel Predictive Linear Gaussian models [227] are the number of base functions and total amount of feature dimensions.

Table 1.9 Quantitative comparisons of prediction error

Methods	Error range	Mean ± SDV	Data type
Linear [227]	[0.0,0.05]	0.026 ± 0.018	k-fold cross-validation
KF [233]	[0.03, 0.73]	10.37 ± 10.62	Three noisy datasets
FSM [234, 235]	[0.0, 0.135]	0.042 ± 0.042	10 uniform
ARMA [217]	[−2.9, −1.7]	−2.3 ± 0.60	Normalized validation

Table 1.10 Comparisons of computational speed

Methods	Speed	Main variables
Linear [227]	$O(Jn^3)$	J: base functions n: feature dimension
KF [233]	$O(l^3)$	l: time series length
FSM [234, 235]	8.34 ± 7.2 frame/s	Morphological structure
ARMA [217]	$O(n)$	n: data

1.6 FUTURE DIRECTION AND CONCLUSION

In this survey, we have shown the state-of-the-art techniques related to KA, mainly for classification and prediction including online and distributed extension and selected applications. We have also explained KA prediction approaches including model-based and hybrid prediction algorithms. The cross-section between classification and prediction is a new research area, considering the uncertainty and irregularity of data sequential configuration. Originating from physical phenomena and technical limitations, these open areas may lead to further investigations of sophisticated KA techniques. In successful implementations of KA for big-data analysis, the prediction and online classification are sometimes merged together. Unlike classical partitioning clustering algorithms for offline machine learning, modern KA for online learning and prediction are able to solve real-world problems despite the big data size. To maximize the potential innovation furthermore, an extensive validation on real-world applications remains a big challenge for these KA techniques. To realize this future direction, collaborative research activities with various disciplines including engineering (signal processing), computer science (machine learning), and statistics (data regression) are preferable.

REFERENCES

1. T. Hoya, and Y. Washizawa, "Simultaneous pattern classification and multidomain association using self-structuring kernel memory networks," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 732–744, 2007.
2. B. J. Kim, I. K. Kim, and K. B. Kim, "Feature extraction and classification system for nonlinear and online data," in *Proceedings of Advances in Knowledge Discovery and Data Mining*, Sydney, Australia, May 26–28, vol. 3056, pp. 171–180, 2004.
3. W. Zheng, C. Zou, and L. Zhao, "An improved algorithm for kernel principal component analysis," *Neural Process. Lett.*, vol. 22, no. 1, pp. 49–56, 2005.
4. V. N. Vapnik, *The nature of statistical learning theory*, 2nd ed. New York: Springer, 2000.
5. R. Duda, P. Hart, and D. Stock, *Pattern Classification*, 2nd ed. Hoboken, NJ: John Wiley & Sons, Inc, 2001.
6. T. Briggs and T. Oates, "Discovering domain specific composite kernels," in *Proc. 20th Nat'l. Conf. Artificial Intelligence*, Pittsburgh, Pennsylvania, July 9–13, pp. 732–738, 2005.
7. N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, "On kernel target alignment," in *Proc. Neural Information Processing Systems*, British Columbia, Canada, Dec. 3–8, pp. 367–373, 2001.

8. H. Xiong, Y. Zhang, and X. W. Chen, "Data-dependent kernel machines for microarray data classification," *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 4, no. 8, pp. 583–595, 2007.
9. T. J. Chin and D. Suter, "Incremental kernel principal component analysis," *IEEE Trans. Image Process.*, vol. 16, no. 6, pp. 1662–1674, 2007.
10. L. Winter, Y. Motai, and A. Docef, "Computer-aided detection of polyps in CT colonography: On-line versus off-line accelerated kernel feature analysis," Special Issue on Processing and Analysis of High-Dimensional Masses of Image and Signal Data, *Signal Process.*, vol. 90, pp. 2456–2467, 2010.
11. Y. Motai, "Synthesized articulated behavior using space-temporal on-line principal component analysis," Special Issue on Imitative Robots, *Adv. Robotics*, vol. 21, no. 13, pp. 1503–1520, 2007.
12. M. Awad, Y. Motai, J. Nappi, and H. Yoshida, "A clinical decision support framework for incremental polyps classification in virtual colonoscopy," *Algorithms*, vol. 3, pp. 1–20, 2010.
13. S. Li and J. Lu, "Face recognition using the nearest feature line method," *IEEE Trans. Neural Netw.*, vol. 10, no. 2, pp. 439–443, 1999.
14. C. Park and S. B. Cho, "Genetic search for optimal ensemble of feature-classifier pairs in DNA gene expression profiles," in *Proc. Int. Joint Conf. Neural Networks*, Portland, Oregon, July 20–24, vol. 3, pp. 1702–1707, 2003.
15. J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, 2004.
16. D. Wang, J. Zheng, Y. Zhou, and J. Li, "A scalable support vector machine for distributed classification in ad hoc sensor networks," *Neurocomputing*, vol. 75, no. 1–3, pp. 394–400, 2010.
17. X. Jiang, R. Snapp, Y. Motai, and X. Zhu, "Accelerated kernel feature analysis," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, New York, New York, June 17–22, pp. 109–116, 2006.
18. T. Damoulas and M. A. Girolami, "Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection," *Bioinformatics*, vol. 24, no. 10, pp. 1264–1270, 2008.
19. H. Xiong, M.N.S. Swamy, and M.O. Ahmad, "Optimizing the data-dependent kernel in the empirical feature space," *IEEE Trans. Neural Netw.*, vol. 16, pp. 460–474, 2005.
20. Y. Motai and H. Yoshida, "Principal composite kernel feature analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1863–1875, 2013.
21. S. Ozawa, S. Pang, and N. Kasabov, "Incremental learning of chunk data for online pattern classification systems," *IEEE Trans. Neural Netw.*, vol. 19, no. 6, pp. 1061–1074, 2008.
22. H. T. Zhao, P. C. Yuen, and J. T. Kwok, "A novel incremental principal component analysis and its application for face recognition," *IEEE Trans. Syst. Man Cyb. B*, vol. 36, no. 4, pp. 873–886, 2006.
23. Y. M. Li, "On incremental and robust subspace learning," *Pattern Recogn.*, vol. 37, no. 7, pp. 1509–1518, 2004.
24. Y. Kim, "Incremental principal component analysis for image processing," *Opt. Lett.*, vol. 32, no. 1, pp. 32–34, 2007.

25. B. J. Kim and I. K. Kim, "Incremental nonlinear PCA for classification," in *Proc. Knowledge Discovery in Databases* (*PKDD 2004*), Pisa, Italy, Sept. 20–24, vol. 3202, pp. 291–300, 2004.
26. B. J. Kim, J. Y. Shim, C. H. Hwang, I. K. Kim, and J. H. Song, "Incremental feature extraction based on empirical kernel map," *Found. Intell. Syst.*, Maebashi City, Japan, October 28–31, vol. 2871, pp. 440–444, 2003.
27. L. Hoegaerts, L. De Lathauwer, I. Goethals, J. A. K. Suykens, J. Vandewalle, and B. De Moor, "Efficiently updating and tracking the dominant kernel principal components," *Neural Netw.*, vol. 20, no. 2, pp. 220–229, 2007.
28. B. Schölkopf and A. J. Smola, "Learning with kernels: Support vector machines, regularization, optimization, and beyond," *Adaptive computation and machine learning*. Cambridge, MA: MIT Press, 2002.
29. H. Fröhlich, O. Chapelle, and B. Schölkopf, "Feature selection for support vector machines by means of genetic algorithm," in *Proc. 15th. IEEE Int. Conf. Tools with Artificial Intelligence*, Sacramento, California, Nov. 3–5, pp. 142–148, 2003.
30. X. W. Chen, "Gene selection for cancer classification using bootstrapped genetic algorithms and support vector machines," in *Proc. IEEE Int. Conf. Computational Systems, Bioinformatics*, Palo Alto, CA, August 11–14, pp. 504–505, 2003.
31. F. A. Sadjadi, "Polarimetric radar target classification using support vector machines," *Opt. Eng.*, vol. 47, no. 4, pp. 1314–1325, 2008.
32. M. Awad and Y. Motai, "Dynamic classification for video stream using support vector machine," Special Issue on Soft Computing for Dynamic Data Mining, *J. Appl. Soft Comput.*, vol. 8, no. 4, pp. 1314–1325, 2008.
33. S. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Netw.*, vol.6, pp. 783–789, 1999.
34. B. Souza and A. de Carvalho, "Gene selection based on multi-class support vector machines and genetic algorithms," *Mol. Res.*, vol. 4, no.3, pp. 599–607, 2005.
35. R. Kondor and T. Jebara, "Gaussian and Wishart hyperkernels," in *Advances in Neural Information Processing Systems*, Vancouver and Whistler, Cananda, Dec. 4–7, vol. 19, pp. 729–736, 2006.
36. J. B. Yin, T. Li, and H.-B. Shen, "Gaussian kernel optimization: Complex problem and a simple solution," *Neurocomputing*, vol. 74, no. 18, pp. 3816–3822, 2011.
37. B. Scholkopf, K. K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2758–2765, 1997.
38. J. Yu, "A nonlinear kernel Gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes," *Chem. Eng. Sci.*, vol. 68, no. 1, pp. 506–519, 2012.
39. K. L. Chan, P. Xue, and L. Zhou, "A kernel-induced space selection approach to model selection in KLDA," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2116–2131, 2008.
40. W. S. Zheng, J. H. Lai, and P. C. Yuen, "Penalized preimage learning in kernel principal component analysis," *IEEE Trans. Neural Netw.*, vol. 21, no. 4, pp. 551–570, 2010.
41. C. Bouveyron, S. Girard, and C. Schmid, "High-dimensional data clustering," *Comput. Stat. Data Anal.*, vol. 52, no. 1, pp. 502–519, 2007.
42. S. Girolami "Mercer kernel-based clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 13, pp. 780–784, 2002.

43. B. Ayhan, M.-Y. Chow, and M.-H. Song, "Multiple discriminant analysis and neural-network-based monolith and partition fault-detection schemes for broken rotor bar in induction motors," *IEEE Trans. Indus. Electron.*, vol. 53, no. 4, pp. 1298–1308, 2006.
44. F. Ratle, C. Gagnéb, A.-L. Terrettaz-Zuffereyc, M. Kanevskia, P. Esseivac, and O. Ribauxc, "Advanced clustering methods for mining chemical databases in forensic science," *Chemom. Intell. Lab. Syst.*, vol. 90, no. 2, pp. 123–131, 2008.
45. P. Phoungphol and Z. Yanqing, "Multi-source kernel k-means for clustering heterogeneous biomedical data," in *Proc. IEEE Intl. Conf. Bioinformatics and Biomedicine Workshops*, Atlanta, Georgia, Nov. 12–15, pp. 223–228, 2011.
46. K. Buza, P. B. Kis, and A. Buza, "Graph-based clustering based on cutting sets," in *Proc. 15th IEEE Intl. Conf. Intelligent Engineering Systems*, Poprad, Slovakia, June 23–25, pp. 143–149, 2011.
47. G Tsoumakas, L Angelis, and I Vlahavas, "Clustering classifiers for knowledge discovery from physically distributed databases," *Data Knowl. Eng.*, vol. 49, no. 3, pp. 223–242, June 2004.
48. P. Honeine, C. Richard, J. C. M. Bermudez, and H. Snoussi, "Distributed prediction of time series data with kernels and adaptive filtering techniques in sensor networks," in *Proc. 42nd Conf. Signals, Systems and Computers*, Pacific Grove, California, Nov. 6–9, pp. 246–250, 2011.
49. M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Processing*, vol. 60, no. 9, pp. 4672–4682, 2012.
50. W. Liu, I. Park, and J. C. Principe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE Trans. Neural Netw.*, vol. 20, no. 12, pp. 1950–1961, 2009.
51. G. Li, C. Wen, Z. G. Li, A. Zhang, F. Yang, and K. Mao, "Model-based online learning with kernels," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 3, pp. 356–369, 2013.
52. C Campbell, "Kernel methods: A survey of current techniques," *Neurocomputing*, vol. 48, no. 1–4, pp. 63–84, 2002.
53. S.-M. Hong, B.-H. Jung, and D. Ruan, "Real-time prediction of respiratory motion based on a local dynamic model in an augmented space," *Phys. Med. Biol.*, vol. 56, no. 6, pp. 1775–1789, 2011.
54. S. Ali and K. A. Smith-Miles, "A meta-learning approach to automatic kernel selection for support vector machines," *Neurocomputing*, vol. 70, no. 1–3, pp. 173–186, 2006.
55. S. Yan, C. Zhang, and X. Tang, "Bilinear analysis for kernel selection and nonlinear feature extraction," *IEEE Trans. Neural Netw.*, vol. 18, no. 5, pp. 1442–1452, 2007.
56. C. Liberati, et al., "Data adaptive simultaneous parameter and kernel selection in kernel discriminant analysis (KDA) using information complexity," *J. Pattern Recogn. Res.*, vol. 4, no. 4, pp. 119–132, 2009.
57. I. W. Tsang and J. T. Kwok, "Efficient hyperkernel learning using second-order cone programming," *IEEE Trans. Neural Netw.*, vol.17, no. 1, pp. 48–58, 2006.
58. G. R. G. Lanckriet et al., "Learning the kernel matrix with semi-definite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, 2004.
59. C. Ke and S. Zhong, "Kernel target alignment for feature kernel selection in universal steganographic detection based on multiple kernel SVM," in *Proc. Int. Symp. Instrumentation Measurement, Sensor Network and Automation*, Sanya, pp. 222–227, August 2012.

60. T. Jebara, "Multitask sparsity via maximum entropy discrimination," *J. Mach. Learn. Res.*, vol. 2, pp. 75–110, 2011.
61. C. S. Ong, J. Smola, and R.C. Williamson, "Hyperkernels," in *Advances in Neural Information Processing Systems*, Vancouver and Whistler, Canada, Dec. 8–13, vol. 15, 2003.
62. C. S. Ong, A. Smola, and B. Williamson, "Learning the kernel with hyperkernels," *J. Mach. Learn. Res.*, vol. 6, pp. 1045–1071, 2005.
63. L. Jia and S. Liao, "Hyperkernel construction for support vector machines," in *Proc. 4th Intl. Conf. Natural Computation*, Jinan, Shandong, China, Oct. 18–20, vol. 2, pp. 76–80, 2008.
64. J. Sun, et al., "Analysis of the distance between two classes for tuning SVM hyperparameters," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 305–318, 2010.
65. C. Cortes, et al., "Algorithms for learning kernels based on centered alignment," *J. Mach. Learn. Res.*, vol. 13, pp. 795–828, 2012.
66. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Min. Knowl. Disc.*, vol. 2, no. 2, pp. 121–167, 1998.
67. B. Scholkopf and A. J. Smola, *Learning with kernels*. Cambridge, MA: MIT Press, 2002.
68. A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil, "A DC-programming algorithm for kernel selection," in *Proc. 23rd Int. Conf. Machine Learning*, Pittsburgh, Pennsylvania, June 25–29, pp. 41–48, 2006.
69. S.-J. Kim, A. Magnani, and S. Boyd, "Optimal kernel selection in kernel Fisher discriminant analysis," in *Proc. 23rd Int. Conf. Machine Learning*, Pittsburgh, Pennsylvania, June 25–29, pp. 465–472, 2006.
70. R. Khemchandani and J. S. Chandra, "Optimal kernel selection in twin support vector machines," *Optim. Lett.*, vol. 3, no. 1, pp. 77–88, 2009.
71. R. Khemchandani and J. S. Chandra, "Learning the optimal kernel for fisher discriminant analysis via second order cone programming," *Eur. J. Oper. Res.*, vol. 203, no. 3, pp. 692–697, 2010.
72. S. R. Gunn and J. S. Kandola, "Structural modelling with sparse kernels," *Mach. Learn.*, vol. 48, pp. 137–163, 2002.
73. T. Jebara, "Multi-task feature and kernel selection for SVMs," in *Proc. 21st Int. Conf. Machine Learning*, Banff, Alberta, Canada, July 4–8, 2004.
74. K. Chaudhuri, C. Monteleoni, and A.D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol.12, pp. 1069–1109, 2011.
75. J. Neumann, C. Schnörr, G. Steidl, "SVM-based feature selection by direct objective minimisation," *Pattern Recogn. Lec. Notes Comput. Sci.*, vol. 3175, pp. 212–219, 2004.
76. Y. Lin and H. H. Zhang, "Component selection and smoothing in smoothing spline analysis of variance models," COSSO. Institute of Statistics MIMEO Series 2556, NCSU, January 2003.
77. C.A. Micchelli and M. Pontil, "Learning the kernel function via regularization," *J. Mach. Learn. Res.*, vol. 6, pp. 1099–1125, 2005.
78. A. Argyriou, C. A. Micchelli, and M. Pontil, "Learning convex combinations of continuously parameterized basic kernels," COLT'05 in *Proc. 18th Conf. Learning Theory*, Bertinoro, Italy, June 27–30, pp. 338–352, 2005.
79. F. R. Bach, and G. R. G. Lanckriet, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proc. 21th Int. Conf. Machine Learning*, Banff, Alberta, Canada, July 4–8, pp. 6–13, 2004.

80. S. Sonnenburg, G. Rätsch, and C. Schäfer, "A general and efficient multiple kernel learning algorithm," *Advances in Neural Information Processing Systems*, Vancouver and Whistler, Cananda, Dec. 4–7, vol. 18, pp. 1273–1280, 2006.
81. G. Dai and D. Yeung, "Kernel selection for semi-supervised kernel machines," in *Proc. 24th Int. Conf. Machine Learning*, Corvallis, OR, June 20–24, pp. 185–192, 2007.
82. S. O. Haykin, "Models of a neuron," *Neural networks and learning machines*. 3rd ed. New York: Macmillan College Publishing Company, pp. 8–13, 1994.
83. G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 6, pp. 1351–1362, 2005.
84. F. Melgani, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, 2004.
85. S. Donniger, T. Hofmann, and J. Yeh. "Predicting CNS permeability of drug molecules: Comparison of neural network and support vector machine algorithms," *J. Comput. Biol.*, vol. 9, no. 6, pp. 849–864, 2002.
86. D. West "Neural network credit scoring models," *Comput. Oper. Res.*, vol. 27, no. 11–12, pp. 1131–1152, 2000.
87. Y. J. Oyang, S. C. Hwang, Y. Y. Ou, C. Y. Chen, and Z. W. Chen, "Data classification with radial basis function networks based on a novel kernel density estimation algorithm," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 225–236, 2005.
88. K. Reddy and V. Ravi, "Differential evolution trained kernel principal component WNN and kernel binary quantile regression: Application to banking," *Knowl. Based Syst.*, vol. 39, pp. 45–56, 2012.
89. Y. Xiao and H. Yigang. "A novel approach for analog fault diagnosis based on neural networks and improved kernel PCA," *Neurocomputing*, vol. 74, no. 7, pp. 1102–1115, 2011.
90. D. X. Niu, Q. Wang, and J. C. Li. "Short term load forecasting model using support vector machine based on artificial neural network," in *Proc. 4th Intl. Conf. Machine Learning and Cybernetics*, Guangzhou, China, August 18–21, vol. 1–9, pp. 4260–4265, 2005.
91. D. C. Park, N. H. Tran, D. M. Woo, and Y. Lee. "Kernel-based centroid neural network with spatial constraints for image segmentation," in *Proc. IEEE 4th Intl. Conf. Natural Computation*, Jinan, Shandong, China, Oct. 18–20, vol. 3, pp. 236–240, 2008.
92. Y. G. Xiao and L. G. Feng, "A novel neural-network approach of analog fault diagnosis based on kernel discriminant analysis and particle swarm optimization," *Appl. Soft Comput.*, vol. 12, no. 2, pp. 904–920, 2012.
93. A. Micheli, F. Portera, and A. Sperduti, "A preliminary empirical comparison of recursive neural networks and tree kernel methods on regression tasks for tree structured domains," *Neurocomputing*, vol. 64, pp. 73–92, 2005.
94. Y. Xia and J. Wang, "A one-layer recurrent neural network for support vector machine learning," *IEEE Trans. Syst. Man Cyb. B*, vol. 34, no. 2, pp. 1261–1269, 2004.
95. K. H. Jang, T. K. Yoo, J. Y. Choi, K. C. Nam, J. L. Choi, M. L. Kwon, and D. W. Kim, "Comparison of survival predictions for rats with hemorrhagic shocks using an artificial neural network and support vector machine," in *33rd Intl. Conf. EMBS*, Boston, Massachusetts, USA, August–September 2011.
96. C. Huang, L. S. David, and J. R. G. Townshend, "An assessment of support vector machines for land cover classification," *J. Remote Sens.*, vol. 23, no. 4, pp. 725–749, 2002.

97. K. A. Aziz, S. S. Abdullah, R. A. Ramlee, and A. N. Jahari, "Face detection using radial basis function neural networks with variance spread value," in *Intl. Conf. Soft Computing and Pattern Recognition*, Malacca, Malaysia, Dec. 4–7, 2009.

98. R. Moraes, J. F. Valiati, and W. P. G. Neto, "Document-level sentiment classification: An empirical comparison between SVM and ANN," *Expert Syst. Appl.*, vol. 40, pp. 621–633, 2013.

99. S. Arora, D. Bhattacharjee, M. Nasipuri, L. Malik, M. Kundu, and D. K. Basu, "Performance comparison of SVM and ANN for handwritten Devnagari character recognition," *J. Comput. Sci. Iss.* vol. 7, no. 3, 2010.

100. R. Burbidge, M. Trotter, B. Buxton, and S. Holden, "Drug design by machine learning support vector machines for pharmaceutical analysis," *Comput. Chem.*, vol. 26, no. 1, pp. 5–14, 2001.

101. A. Statnikov, C. F. Aliferis, I. Tsamardions, D. Hardin, and S. Levy, "A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis," *Bioinformatics*, vol. 21 no. 5, pp. 631–643, 2005.

102. V. N. Vapnik, "An overview of statistical learning theory," *IEEE Trans. Neural Netw.*, vol. 10, no 5, pp. 988–999, 1999.

103. B. Boser I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifier," in *Proc. 5th ACM Workshop Computational Learning Theory*, Pittsburgh, PA, July 27–29, pp. 144–152, 1992.

104. M. Pontil and A. Verri, "Properties of support vector machines," *Neural Comput.*, vol. 10, no. 4, pp. 955–974, 1998.

105. C. Campbell and Y. Ying, *Learning with support vector machines*. San Rafael, CA: Morgan & Claypool, pp. 6, 2011.

106. V.D. Sanchez A, "Advanced support vector machines and kernel methods," *Neurocomputing*, vol. 55, pp. 5–20, 2003.

107. J. Shawe-Taylor and S. Sun, "A review of optimization methodologies in support vector machines," *Neurocomputing*, vol. 74, no. 17, pp. 3609–3618, 2011.

108. J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in kernel methods—Support vector learning*, B. Scholkopf, C.J.C. Burges, and A.J Smola (Eds.). Cambridge, MA: The MIT Press, 1998.

109. G. W. Flake and S. Lawrence, "Efficient SVM regression training with SMO," *Mach. Learn.*, vol.46, pp. 271–290, 2002.

110. P. Laskov, "An improved decomposition algorithm for regression support vector machines," *Mach. Learn.*, 46, pp. 315–350, 2002.

111. E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Proc. of IEEE Neural Networks and Signal Processing*, Amelia Island, FL, Sep. 24–26, 1997.

112. O. L. Mangasarian and D. R. Musicant, "Successive overrelaxation for support vector machines," *IEEE Trans. Neural Netw.*, vol.10, no.5, pp. 1032–1037, 1999.

113. T. Friess, N. Cristianini, and C. Campbell, "The Kernel-adatron: A fast and simple learning procedure for support vector machines," in *Proc. 5th Intl. Conf. Machine Learning*, Helsinki, Finland, June 5–9, pp. 188–196, 1998.

114. A. J. Smola and B. Scholkopf, "From regularization operators to support vector kernels," in *Advances in Neural Information Processing Systems*, Denver, CO, Dec. 1–3, vol. 10, pp. 343–349, 1998.

115. D. Meyer, F. Leisch, and K. Hornik, "The support vector machine under test," *Neurocomputing*, vol. 55, no. 1–2, pp. 169–186, 2003.

116. K. Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.29, no.5, pp. 905–910, 2007.

117. M. Arun Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," *Expert Syst. Appl.*, vol.36, no. 4, pp. 7535–7543, 2009.

118. B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.

119. E. Barshan, A. Ghodsi, and Z. Azimifar, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recogn.*, vol. 44, no. 7, pp. 1357–1371, 2011.

120. J. Yang, A. F. Frangi, and J. Y. Yang, "KPCA plus LDA: A complete kernel fisher discriminant framework for feature extraction and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 230–244, 2005.

121. C. J. Liu, "Gabor-based kernel PCA with fractional power polynomial models for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 5, pp. 572–581, 2004.

122. S. Lemm, B. Blankertz, T. Dickhaus, and K.-R. Müller "Introduction to machine learning for brain imaging," *Neuroimage*, vol. 56, no. 2, pp. 387–399, May 2011.

123. R. M. Balabin and R. Z. Safieva, "Near-infrared (NIR) spectroscopy for biodiesel analysis: Fractional composition, iodine value, and cold filter plugging point from one vibrational spectrum," *Energy Fuel* vol. 25, no. 5, pp. 2373–2382, 2011.

124. X. Yang, S. Gao, S. Xu, Z. Zhang, B. M. Prasanna, L. Li, J. Li, and J. Yan, "Characterization of a global germplasm collection and its potential utilization for analysis of complex quantitative traits in maize," *Mol. Breeding*, vol. 28, no. 4, pp 511–526, 2011.

125. J. Yu, "Nonlinear bioprocess monitoring using multiway kernel localized fisher discriminant analysis," *Indus. Eng. Chem. Res.*, vol. 50, no. 6, pp. 3390–3402, 2011.

126. Y. Zhang and C. Ma, "Fault diagnosis of nonlinear processes using multiscale KPCA and multiscale KPLS," *Chem. Eng. Sci.*, vol. 66, no. 1, pp. 64–72, 2011.

127. J. T. Leek, "Asymptotic conditional singular value decomposition for high-dimensional genomic data," *Biometrics*, vol. 67, no. 2, pp. 344–352, 2011.

128. G. Liu, X. Sun, and K. Fu, "Aircraft recognition in high-resolution satellite images using coarse-to-fine shape prior," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 3, pp. 573–577, May 2013.

129. A. Erener, "Classification method, spectral diversity, band combination and accuracy assessment evaluation for urban feature detection," *J. Appl. Earth Obs. Geoinform.*, vol. 21, pp. 397–408, 2013.

130. H. Liu, M. Huang, and J. T. Kim, "Adaptive neuro-fuzzy inference system based faulty sensor monitoring of indoor air quality in a subway station," *Kor. J. Chem. Eng.*, vol. 30, no. 3, pp. 528–539, 2013.

131. W.-Y. Lin and C.-Y. Hsieh, "Kernel-based representation for 2D/3D motion trajectory retrieval and classification," *Pattern Recogn.*, vol. 46, no. 3, pp. 662–670, 2013.

132. G. Rong, S.-Y. Liu, and J.-D. Shao, "Fault diagnosis by locality preserving discriminant analysis and its kernel variation," *Chem. Eng. Sci.*, vol. 49, pp. 105–113, 2013.

133. K. N. Reddy, and V. Ravi, "Differential evolution trained kernel principal component WNN and kernel binary quantile regression: Application to banking," *Knowl. Based Syst.*, vol. 39, pp. 45–56, 2013.

134. S. Liwicki, G. Tzimiropoulos, and S. Zafeiriou, "Euler principal component analysis," *J. Comput. Vision*, vol. 101, no. 3, pp. 498–518, 2013.

135. D. K. Saxena, J. A. Duro, and A. Tiwari, "Objective reduction in many-objective optimization: Linear and nonlinear algorithms," *IEEE Trans. Evol. Comput.*, vol. 17, no. 1, pp. 77–99, 2013.

136. C. Silva, U. Lotric, B. Ribeiro, A. Dobnikar, "Distributed text classification with an ensemble kernel-based learning approach," *IEEE Trans. Syst. Man Cyb. C*, vol. 40, no.3, pp. 287-297, May 2010.

137. T. Gärtner, "A survey of kernels for structured data," *SIGKDD Explor. Newsl.*, vol. 5, no. 1, pp 49–58, 2003.

138. R. Rosipal and L. J. Trejo, "Kernel partial least squares regression in reproducing kernel Hilbert space," *J. Mach. Learn. Res.*, vol. 2, pp. 97–123, 2002

139. D. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Comput.*, vol. 16, no. 12, pp. 2639–2664, 2004.

140. R.I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *Proceeding ICML '02 Proceedings of the Nineteenth International Conference on Machine Learning*, Sydney, NSW, Australia, July 8–12, vol. 2, pp. 315–322, 2002.

141. B. Yifeng, X. Jian, and Y. Long. "Kernel partial least-squares regression," in *Neural Networks*, *IJCNN'06*. *International Joint Conference on IEEE*, Vancouver, BC, Canada, July 16–21, 2006.

142. S. Wold, et al., "The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses," *SIAM J. Sci. Statist. Comput.*, vol.5, no.3, pp. 735–743, 1984.

143. S. de Jong, "SIMPLS: An alternative approach to partial least squares regression," *Chemometr. Intell. Lab. Syst.*, vol. 18, no. 3, pp. 251–263, 1993.

144. K. Kim, J. M. Lee, and I. B. Lee, "A novel multivariate regression approach based on kernel partial least squares with orthogonal signal correction," *Chemometr. Intell. Lab. Syst.*, vol. 79, no. 1, pp. 22–30, 2005.

145. W. H. A. M.V.D. Broek, et al., "Plastic identification by remote sensing spectroscopic NIR imaging using kernel partial least squares (KPLS)," *Chemometr. Intell. Lab. Syst.*, vol. 35, no. 2, pp 187–197, 1996.

146. V. Štruc, and N. Pavešić, "Gabor-based kernel partial-least-squares discrimination features for face recognition," *Informatica*, vol. 20, no.1, pp. 115–138, 2009.

147. Lai, P. L., and C. Fyfe, "Kernel and nonlinear canonical correlation analysis," *Int. J. Neural Syst.*, vol. 10, no. 5, pp. 365, 2000.

148. T. Melzer, M. Reiter, and H. Bischof, "Appearance models based on kernel canonical correlation analysis," *Pattern Recogn.*, vol. 36, no.9, pp. 1961–1971, 2003.

149. W. Zheng, et al., "Facial expression recognition using kernel canonical correlation analysis (KCCA)," *IEEE Trans. Neural Netw.*, vol.17, no.1, pp 233–238, 2006.

150. Y. Yamanishi, et al., "Extraction of correlated gene clusters from multiple genomic data by generalized kernel canonical correlation analysis," *Bioinformatics*, vol. 19, no. 1, pp. 323–330, 2003.

151. K. Fukumizu, F. R. Bach, and A. Gretton. "Statistical consistency of kernel canonical correlation analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 361–383, 2007.

152. E. Kokiopoulou, and Y. Saad. "Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no.12, pp 2143–2156, 2007.

153. D.R. Hardoon and J. Shawe-Taylor, "Convergence analysis of kernel canonical correlation analysis: Theory and practice," *Mach. Learn.*, vol. 74, no. 1, pp. 23–38, 2009.

154. G. Boente, and R. Fraiman. "Kernel-based functional principal components," *Stat. Prob. Let.*, vol. 48, no.4, pp. 335–345, 2000.

155. D. Cai, et al., "Orthogonal laplacianfaces for face recognition," *IEEE Trans. Image Processing*, vol. 15, no.11, pp. 3608–3614, 2006.

156. A. Ruiz and P. E. López-de-Teruel, "Nonlinear kernel-based statistical pattern analysis," *IEEE Trans. Neural Netw.*, vol. 12, no.1, pp. 16–32, 2001.

157. J. Yang, et al. "Globally maximizing, locally minimizing: Unsupervised discriminant projection with applications to face and palm biometrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no.4, pp. 650–664, 2007.

158. J. Arenas-Garcia, K. Petersen, G. Camps-Valls, and L.K. Hansen, "Kernel multivariate analysis framework for supervised subspace learning: A tutorial on linear and kernel multivariate methods," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 16,29, 2013

159. J. Aitchison and C. GG Aitken, "Multivariate binary discrimination by the kernel method," *Biometrika*, vol.63, no.3, pp. 413–420, 1976.

160. E. Pereda, R. Q. Quiroga, and J. Bhattacharya, "Nonlinear multivariate analysis of neurophysiological signals," *Prog. Neurobiol.*, vol. 77, no.1, pp. 1–37, 2005.

161. B.A. Weinstock, et al. "Prediction of oil and oleic acid concentrations in individual corn (*Zea mays* L.) kernels using near-infrared reflectance hyperspectral imaging and multivariate analysis," *Appl. Spectrosc.*, vol. 60, no.1, pp. 9–16, 2006.

162. Z. D. Bai, C. R. Rao, and L. C. Zhao. "Kernel estimators of density function of directional data," *J. Multivariate Anal.*, vol. 27, no.1, pp. 24–39, 1988.

163. T. Duong, "k_s: Kernel density estimation and kernel discriminant analysis for multivariate data in R," *J. Stat. Software*, vol. 21, no.7, pp. 1–16, 2007.

164. D. K. Hammond, P. Vandergheynst, and R. Gribonval. "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmonic Anal.*, vol. 30, no. 2, pp. 129–150, 2011.

165. S. Köhler, et al. "Walking the interactome for prioritization of candidate disease genes," *Am. J. Human Genet.*, vol. 82, no.4, pp. 949–958, 2008.

166. M. Bylesjö, et al., "K-OPLS package: Kernel-based orthogonal projections to latent structures for prediction and interpretation in feature space," *BMC Bioinformatics*, vol. 9, no. 1, pp. 106, 2008.

167. M. Rantalainen, et al., "Kernel-based orthogonal projections to latent structures (K-OPLS)," *J. Chemometr.*, vol. 21, no. 7–9, pp. 376–385, 2007.

168. S. Zhao, F. Precioso, M. Cord, "Spatio-temporal tube data representation and kernel design for SVM-based video object retrieval system," *Multimed. Tools Appl.*, vol. 55, no. 1, pp. 105–125, 2011.

169. F. Dinuzzo, G. Pillonetto, and G.D. Nicolao, "Client–server multitask learning from distributed datasets," *IEEE Trans. Neural Netw.*, vol. 22 no. 2, 2011.

170. R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28 issue 1, pp. 41–75, 1997.

171. A. Kitamoto, "Typhoon analysis and data mining with kernel methods," *Pattern recognition with support vector machines*. Berlin, Heidelberg: Springer, pp. 237–249, 2002.

172. B. L. Milenova, J. S. Yarmus, M. M. Campos, "SVM in oracle database 10g: Removing the barriers to widespread adoption of support vector machines," in *Proc. 31st Int. Conf. Very Large Databases*, VLDB Endowment, Trondheim, Norway, August 30–September 2, 2005.

173. Y. Zhang, "Enhanced statistical analysis of nonlinear processes using KPCA, KICA and SVM," *Chem. Eng. Sci.*, vol. 64, no. 5, pp. 801–811, 2009.

174. J. X. Dong, A. Krzyzak, and C. Y. Suen, "Fast SVM training algorithm with decomposition on very large data sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 603–618, 2005.

175. C. J. Lin, "Large-scale machine learning in distributed environments." Tutorial at *ACM ICMR*. Jun 2012. [Online]. Available: http://www.csie.ntu.edu.tw/∼cjlin/talks/icmr2012.pdf. Accessed 18 November 2014.

176. Z.A. Zhu, W. Z. Chen, G. Wang, C. G. Zhu, and Z. Chen, "P-packSVM: Parallel primal gradient descent Kernel SVM," in *9th IEEE Intl Conf. Data Mining*. [Online]. http://people.csail.mit.edu/zeyuan/paper/2009-ICDM-Parallel.pdf. Accessed 18 November 2014.

177. E.Y. Chang, K. Zhu, H. Wang, and H. Bai, "PSVM: Parallelizing support vector machines on distributed computers," *Adv. Neural Inform. Process. Syst.*, vol. 20, pp. 257–264, 2008.

178. Z. Y. Chen, Z. P. Fan, and M. Sun, "A hierarchical multiple kernel support vector machine for customer churn prediction using longitudinal behavioral data," *Eur. J. Oper. Res.*, vol. 223, pp. 461–472, 2012.

179. N. A. Syed, H. Liu, and K. K. Sung, "Handling concept drifts in incremental learning with support vector machines," in *5th ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining*, San Diego, CA, USA August 15–18, 1999.

180. Z. Liang and Y. Li, "Incremental support vector machine learning in the primal and applications." *Neurocomputing*, vol. 72, pp. 2249–2258, 2009.

181. S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations." *J. Mach. Learn. Res.*, vol. 2, pp. 243–264, 2001.

182. H. Xu, Y. Wen, and J. Wang, "A fast-convergence distributed support vector machine in small-scale strongly connected networks." *Front. Electr. Electron. Eng.*, vol. 7, no. 2, pp. 216–223, 2011.

183. T. T. Frieb, N. Cristianini, and C. Campbell, "The kernel-adatron algorithm: A fast and simple learning procedure for support vector machines," in *Proc. 15th Intl. Conf. Machine Learning*, 1998.

184. K. Woodsend, G. Jacek, "Exploiting separability in large-scale linear support vector machine training," *Comput. Optim. Appl.*, vol. 49, no. 2, pp. 241–269, 2011.

185. Z. Chen, Z. Fan, "Dynamic customer lifetime value prediction using longitudinal data: An improved multiple kernel SVR approach," *Knowl. Based Syst.*, vol. 43, pp. 123–134, 2013.

186. Z.-Y. Chen and Z.-P. Fan, "Distributed customer behavior prediction using multiplex data: A collaborative MK-SVM approach," *Knowl. Based Syst.*, vol.35, pp. 111–119, 2012.

187. Z. Chen, J. Li, L. Wei, W. Xu, and Y. Shi, "Multiple-kernel SVM based multiple-task oriented data mining system for gene expression data analysis," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12151–12159, 2011.

188. J. Cheng, J. Qian, Y. Guo, "A distributed support vector machines architecture for chaotic time series prediction," *Neural Inform. Process.*, vol. 4232, pp 892–899, 2006.

189. C. Blake, E. Keogh, C. Merz, *UCI repository of machine learning databases.* [Online] http://archive.ics.uci.edu/ml/, 1998.

190. T. Rose, M. Stevenson, and M. Whitehead, "The reuters corpus volume 1-from yesterday's news to tomorrow's language resources," *LREC*, Vol. 2, pp. 827–832, 2002.

191. S. Agarwal, V. Vijaya Saradhi, and H. Karnick, "Kernel-based online machine learning and support vector reduction," *Neurocomputing*, vol. 71, no. 7–9, pp. 1230–1237, 2008.

192. A. Singh, N. Ahuja, and P. Moulin, "Online learning with kernels: Overcoming the growing sum problem," in *Proc. IEEE Intl Workshop on Machine Learning for Signal Processing*, Santander, Spain, Sep. 23–26, pp. 1–6, 2012.

193. F. Orabona, J. Keshet, and B. Caputo, "Bounded kernel-based online learning," *J. Mach. Learn. Res.*, vol. 10, pp. 2643–2666, Dec. 2009.

194. K. Slavakis, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in reproducing kernel hilbert spaces: The robust beamforming case," *IEEE Trans. Signal Processing*, vol. 57, no. 12, pp. 4744–4764, 2009.

195. Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Processing*, vol. 52, no. 8, pp. 2275–2285, 2004.

196. L. Csató and M. Opper, "Sparse on-line Gaussian processes," *Neural Comput.*, vol. 14, no. 3, pp. 641–668, Mar. 2002.

197. B. Chen, S. Zhao, P. Zhu, and J. C. Principe, "Quantized kernel least mean square algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 1, pp. 22–32, 2012.

198. R. Jin, S. C. H. Hoi, and T. Yang, "Online multiple kernel learning: algorithms and mistake bounds," in *Algorithmic learning theory*, M. Hutter, F. Stephan, V. Vovk, and T. Zeugmann, Eds. Berlin, Heidelberg: Springer, pp. 390–404, 2010.

199. S. C. H. Hoi, R. Jin, P. Zhao, and T. Yang, "Online multiple kernel classification," *Mach. Learn.*, vol. 90, no. 2, pp. 289–316, 2012.

200. F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958.

201. J. P. Rhinelander and X. X. Lu, "Stochastic subset selection for learning with kernel machines," *IEEE Trans. Syst. Man Cyb. B*, vol. 42, no. 3, pp. 616–626, 2012.

202. I. Basheer and M. Hajmeer, "Artificial neural networks: Fundamentals, computing, design, and application," *J. Microbiol. Meth.*, vol. 43, no. 1, pp. 3–31, 2000.

203. L. Rutkowski, "Adaptive probabilistic neural networks for pattern classification in time-varying environment," *IEEE Trans. Neural Netw.*, vol. 15, no. 4, pp. 811–827, 2004.

204. Y.-H. Liu, Y.-C. Liu, and Y.-J. Chen, "Fast support vector data descriptions for novelty detection," *IEEE Trans. Neural Netw.*, vol. 21, no. 8, pp. 1296–1313, 2010.

205. H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recogn.*, vol. 40, no. 3, pp. 863–874, 2007.

206. A. M. Jade, B. Srikanth, V. K. Jayaraman, B. D. Kulkarni, J. P. Jog, and L. Priya, "Feature extraction and denoising using kernel PCA," *Chem. Eng. Sci.*, vol. 58, no. 19, pp. 4441–4448, 2003.

207. K. I. Kim, M. O. Franz, and B. Scholkopf, "Iterative kernel principal component analysis for image modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 9, pp. 1351–1366, 2005.

208. C. Gruber, T. Gruber, S. Krinninger, and B. Sick, "Online signature verification with support vector machines based on LCSS kernel functions," *IEEE Trans. Syst. Man Cyb. B*, vol. 40, no. 4, pp. 1088–1100, 2010.

209. M. Davy, F. Desobry, A. Gretton, and C. Doncarli, "An online support vector machine for abnormal events detection," *Signal Processing*, vol. 86, no. 8, pp. 2009–2025, 2006.

210. F. Desobry, M. Davy, and C. Doncarli, "An online kernel change detection algorithm," *IEEE Trans. Signal Processing*, vol. 53, no. 8, pp. 2961–2974, 2005.

211. Y. Zhang, N. Meratnia, and P. Havinga, "Adaptive and online one-class support vector machine-based outlier detection techniques for wireless sensor networks," in *Proc. Intl. Conf. Advanced Information Networking and Applications Workshops*, Bradford, United Kingdom, May 26–29, pp. 990–995, 2009.

212. V. Gomez-Verdejo, J. Arenas-Garcia, M. Lazaro-Gredilla, and A. Navia-Vazquez, "Adaptive one-class support vector machine," *IEEE Trans. Signal Processing*, vol. 59, no. 6, pp. 2975–2981, 2011.

213. K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 551–585, 2006.

214. C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2009.

215. W. Wang, C. Men, and W. Lu, "Online prediction model based on support vector machine," *Neurocomputing*, vol. 71, no. 4–6, pp. 550–558, 2008.

216. K. Kuusilinna, V. Lahtinen, T. Hamalainen, and J. Saarinen, "Finite state machine encoding for VHDL synthesis," *IEE Proc. Comput. Digital Tech.*, vol. 148, no. 1, pp. 23–30, 2001.

217. M. Martinez-Ramon, J. L. Rojo-Alvarez, G. Camps-Valls, J. Munoz-Mari, A. Navia-Vazquez, E. Soria-Olivas, and A. R. Figueiras-Vidal, "Support vector machines for nonlinear kernel ARMA system identification," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1617–1622, 2006.

218. R. K. K. Yuen, E. W. M. Lee, C. P. Lim, and G. W. Y. Cheng, "Fusion of GRNN and FA for online noisy data regression," *Neural Process. Lett.*, vol. 19, no. 3, pp. 227–241, 2004.

219. H. A. Kingravi, G. Chowdhary, P. A. Vela, and E. N. Johnson, "A reproducing kernel Hilbert space approach for the online, update of radial bases in neuro-adaptive control," in *Proc. 50th IEEE Conf. Decision and Control and European Control*, Orlando, Florida, December 12–15, pp. 1796–1802, 2011.

220. J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264 –276, 2001.

221. S. Marsland, "Novelty detection in learning systems," *Neural Comput. Surv.*, vol. 3, no. 2, pp. 157–195, 2003.

222. B. Sofman, B. Neuman, A. Stentz, and J. A. Bagnell, "Anytime online novelty and change detection for mobile robots," *J. Field Robotics*, vol. 28, no. 4, pp. 589–618, 2011.

223. O. Taouali, I. Elaissi, and H. Messaoud, "Online identification of nonlinear system using reduced kernel principal component analysis," *Neural Comput. Appl.*, vol. 21, no. 1, pp. 161–169, Feb. 2012.

224. S. Ozawa, Y. Takeuchi, and S. Abe, "A fast incremental kernel principal component analysis for online feature extraction," in *Proc. PRICAI 2010: Trends in Artificial Intelligence*, B.-T. Zhang and M. A. Orgun, Eds. Berlin Heidelberg: Springer, 2010, Daegu, Korea, August 30–September 2, pp. 487–497, 2010.

225. S. Salti and L. Di Stefano, "On-line support vector regression of the transition model for the Kalman filter," *Image Vision Comput.*, vol. 31, no. 6–7, pp. 487–501, 2013.

226. J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

227. D. Wingate and S. Singh, "Kernel predictive linear Gaussian models for nonlinear stochastic dynamical systems," in *Proc. Intl. Conf. Machine Learning*, Pittsburgh, Pennsylvania, June 25–29, pp. 1017–1024, 2006.

228. G. Welch and G. Bishop, "An introduction to the Kalman filter," 1995. [Online]. http://clubs.ens-cachan.fr/krobot/old/data/positionnement/kalman.pdf. Accessed 17 Nov 2014.

229. R. S. Liptser and A. N. Shiryaev, *Statistics of random processes: II. Applications*. New York, NY: Springer, 2001.

230. E. D. Sontag, *Mathematical control theory: Deterministic finite dimensional systems*. New York, NY: Springer, 1998.

231. J. Durbin and S. Koopman, *Time series analysis by state space methods*, 2nd ed. Oxford, United Kingdom: Oxford University Press, 2012.

232. C. K. Chui and G. Chen, *Kalman filtering: With real-time applications*. New York, NY: Springer, 2009.

233. L. Ralaivola and F. d' Alche-Buc, "Time series filtering, smoothing and learning using the kernel Kalman filter," in *Proc. Intl. Joint Conf. Neural Networks*, Montreal, QC, Canada, July 31–August 4, vols. 1–5, pp. 1449–1454, 2005.

234. F. M. Waltz and J. W. V. Miller, "Gray-scale image processing algorithms using finite-state machine concepts," *J. Electron. Imaging*, vol. 10, no. 1, pp. 297–307, 2001.

235. C. Saunders, J. Shawe-Taylor, and A. Vinokourov, "String kernels, Fisher kernels and finite state automata," in *Advances in Neural Information Processing Systems*, Vancouver, Whistler, Canada, Dec. 8–13, vol. 15, pp. 633–640, 2003.

236. T. Natschläger and W. Maass, "Spiking neurons and the induction of finite state machines," *Theor. Comput. Sci.*, vol. 287, no. 1, pp. 251–265, 2002.

237. L. Shpigelman, H. Lalazar, and E. Vaadia, "Kernel-ARMA for hand tracking and brain-machine interfacing during 3D motor control," in *Proc. NIPS*, Vancouver, B.C., Canada, December 8–11, pp. 1489–1496, 2008.

238. J. G. de Gooijer, B. Abraham, A. Gould, and L. Robinson, "Methods for determining the order of an autoregressive-moving average process: A survey," *Int. Stat. Rev./Revue Internationale de Statistique*, vol. 53, no. 3, pp. 301–329, Dec. 1985.

239. L. Hoegaertsa, L. De Lathauwerb, I. Goethalsa, J.A.K. Suykensa, J. Vandewallea, and B. De Moora, "Efficiently updating and tracking the dominant kernel principal components," *Neural Netw.*, vol.20, no. 2, pp. 220-229, 2007.

240. S. R. Upadhyaya, "Parallel approaches to machine learning—A comprehensive survey," *J. Parallel Distr. Com.*, vol.73, no. 3, pp. 284–292, 2013.

241. P. P. Pokharel, W. Liu, and J. C. Principe, "Kernel least mean square algorithm with constrained growth," *Signal Processing*, vol. 89, no. 3, pp. 257–265, 2009.

242. Y. Qi, P. Wang, S. Fan, X. Gao, and J. Jiang, "Enhanced batch process monitoring using Kalman filter and multiway kernel principal component analysis," in *Proc. 21st Control and Decision Conference*, Guilin, China, Jun. 17–19, vol. 5, pp. 5289–5294, 2009.