Kernel Association for Classification and Prediction: A Survey

Yuichi Motai, *Senior Member, IEEE*

***Abstract*— Kernel association (KA) in statistical pattern recognition used for classification and prediction have recently emerged in a machine learning and signal processing context. This survey outlines the latest trends and innovations of a kernel framework for big data analysis. KA topics include offline learning, distributed database, online learning, and its prediction. The structural presentation and the comprehensive list of references are geared to provide a useful overview of this evolving field for both specialists and relevant scholars.**

***Index Terms*— Kernel methods, Mercer kernels, neural network (NN), principal component analysis (PCA), support vector machine (SVM).**

I. Introduction

KERNEL methods have been widely studied for pattern classification and multidomain association tasks [1]–[3]. Kernel association (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 [6]–[8]. This approach is called the kernel trick. Kernel functions have been introduced for sequence data, graphs, text, images, as well as vectors [12], [22]–[24], [32]–[34].

Kernel feature analysis attracts significant attention in both fields of machine learning and signal processing [14], [23], thus there are demands to cover the state-of-the-art of this topic [142]. In this survey paper, we identify the following popular trends and developments on KA so that we can visualize the merits and potentials in an organized manner.

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

Manuscript received June 12, 2013; revised June 3, 2014; accepted June 23, 2014. Date of publication July 10, 2014; date of current version January 15, 2015. This work was supported in part by the Institutional Research under Grant IRG-73-001-31 through the American Cancer Society, in part by the Center for Clinical and Translational Research through the National Center for Advanced Translational Sciences under Grant UL1TR000058, and in part by the National Science Foundation CAREER under Award 1054333.

The author is with the Department of Electrical and Computer Engineering, Virginia Commonwealth University, Richmond, VA 23284 USA (e-mail: ymotai@vcu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2014.2333664

Constructing composite kernels is an anticipated solution for heterogeneous data problems. A composite kernel is more relevant for the data set and adapts itself by adjusting its composed coefficient parameters, thus allowing more flexibility in the kernel choice [8]–[13].

The further 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 [14]–[20]. The 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 to the feature space if necessary [15]–[21]. In a nonincremental update, the dimension of the feature space remains constant as the newly computed features may replace some of the existing ones [11]–[13]. In this survey paper, we also identify the following possibilities.

1) A link between the offline learning and online learning using KA framework, which suggests other connections and potential impact both on machine learning and signal processing.
2) A relationship between online learning and prediction techniques to merge them together for adaptive prediction from online learning.
3) 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 machine (SVM) [25]–[32] Gaussian processes [47], [62], [73], [103], [118], Fisher's linear discriminant analysis (LDA) [11], [36], principal component analysis (PCA) [3], [8], [13], [16], [18]–[24], [35] spectral clustering [108], [111], [132], [133], [153], [155], [157], linear adaptive filters [154], [167], [170], [189], [200], [210], and many others.

The objective of this survey paper is to deliver a comprehensive review of current KA methods from offline and online learning aspects [1], [11], [15]. 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], [25]. This latest survey of KA may stimulate the emergence of neural network (NN) studies in computer science applications and encourage collaborative research activities in relevant engineering areas [12], [22]–[24], [32].

We start to describe the kernel offline methods by basic understanding of KA in Section II. Then, this survey paper shows advanced distributed database technology for KA in Section III. In the following sections, we point out online

learning frameworks (versus offline learning in Section II) using kernel methods in Section IV, and then prediction of data anomaly in Section V. Finally, we conclude this paper for the future directions of KA in Section VI.

II. 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: 1) how to choose the appropriate kernels for offline learning during the learning phase and 2) how to adopt KA into the traditionally developed machine learning techniques such as NN, SVM, and PCA, where the (nonlinear) learning data-space is placed under the linear space via kernel tricks.

A. Choose the Appropriate Kernels

Selecting appropriate kernels is regarded as the key problem in KA since it has a direct effect on the performance of machine learning techniques [37]–[49]. The proper kernels are identified by priori without analysis of database, and the kernel selection is often implemented in the initial stage of the offline learning. Commonly used kernel functions are as follows [39], [50]–[52].

1) The linear kernel: $K(x,\bar{x}) = x^T\bar{x}$.
2) The polynomial kernel: $K(x,\bar{x}) = \left(x^T\bar{x} + \text{offset}\right)^d$.
3) The Gaussian radial basis function (RBF) kernel: $K(x,\bar{x}) = \exp(-\|x-\bar{x}\|^2/2\sigma^2)$.
4) The exponential Gaussian RBF kernel
$$K(x,\bar{x}) = \exp\left(-\|x-\bar{x}\|/2\sigma^2\right).$$
5) The Laplace kernel: $K(x,\bar{x}) = \exp(-\|x-\bar{x}\|^2/\sigma^{2\,1/2})$.
6) The Laplace RBF kernel: $K(x,\bar{x}) = \exp\left(-\sigma\|x-\bar{x}\|\right)$.
7) The sigmoid kernel: $K(x,\bar{x}) = \tanh\left(\beta_0 x^T\bar{x} + \beta_1\right)$.
8) The ANOVA RB kernel: $K(x,\bar{x}) = \sum_{k=1}^{n}\exp\left(-\sigma(x^k-\bar{x}^k)^2\right)^d$.
9) The Cauchy kernel: $K(x,\bar{x}) = 1/\left(1+\|x-\bar{x}\|^2/\sigma\right)$.
10) The multiquadric kernel: $K(x,\bar{x}) = \left(\|x-\bar{x}\|^2+\sigma^2\right)^{1/2}$.
11) The power exponential kernel: $K(x,\bar{x}) = \exp((-\|x-\bar{x}\|^2/2\sigma^2)^d)$.
12) The linear spline kernel in one dimension
$$K(x,\bar{x}) = 1 + xx_i\min(x,\bar{x}) - \frac{x+\bar{x}}{2}\times\left(\min(x,\bar{x})^2 + \frac{\min(x,\bar{x})^3}{3}\right)$$
where x is the data and $\bar{x}$ is the kernel center. Many KA studies do not address appropriate kernel functions; instead, they simply choose well-known traditional kernels empirically by following other studies [39], [40], [53]–[56].

As shown in Table I, there are four proposed kernel methods for choosing proper kernels: 1) linear combinations of base kernel functions; 2) hyperkernels; 3) difference of convex (DC) functions programming; and 4) convex optimization. The representative accuracy is calculated as the average of maximum accuracy with its standard deviation for each kernel selection approach 1)–4), according to the papers listed. Based on Table I, DC approach consistently achieved the best accuracy, assuming with a 95% level of confidence.

TABLE I
Kernel Selection Approaches and Related Research

Kernel selection approach	Relevant studies	Representative accuracy
Linear combinations	[41], [43], [57], [58]	93.6 ± 8.4
Hyperkernels	[40], [44], [45], [46], [47], [59]	95.9 ± 4.3
Difference of convex (DC) functions	[48], [60], [61], [62]	99.1 ± 0.76
Convex optimization	[54], [56], [63], [64]	96.5 ± 4.1

1) 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 Mercer's condition can generate another kernel [58]. In other words, the appropriate kernel is chosen by estimating the optimal weighted combination of base kernels [39], [52], to satisfy Mercer's condition $\iint K(x,x_i)g(x)g(x_i)dxdx_i \geq 0$.

The proper kernels [41], [42], [53], [57], [58] are decided when base kernels $k_1,\ldots,k_D$ are given from (1) [43], [58]

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

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 the scalar as the output. Also, $w_1,\ldots,w_D$ are nonnegative weights [43], [58]. As shown in (1), this method is effective because weights w_d of useless kernels k_d are ideally set to zero.

2) Hyperkernels: Hyperkernels is proposed in [44], which can be used to kernel learning for the inductive setting [40]. The definition of hyperkernels is as follows [40], [44]–[47], [59].

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

1) K is positive definite on $\hat{x}$;
2) for any $\hat{\chi} \in \hat{x}$, K_x is positive definite on x.

These hyperkernels are flexible in kernel learning for a vast range of datasets since they have the property of translation and rotation invariant simultaneously [46]. In [40], [45]–[47], and [59], these hyperkernels are extended into a variety of aspects in Table II.

Those hyperkernel approaches listed in Table II 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.

TABLE II
HYPERKERNEL METHOD COMPARISON

Method	Principle	Advantage
Iterative optimization [40]	second-order cone programming	efficient calculation
Invariance [46]	translation and rotation invariant simultaneously	flexible wide-range of data sets
Learning [45]	learning hyperplane	optimal parameters
Gaussian [47]	Wishart hyperkernels	less computational complexity
Structural risk minimization [59]	regularization penalty into the optimization	low bias, high variance, prevent overfitting

3) DC Functions-Programming Algorithm: The kernel is selected by specifying its objective function [41], [44], [61], [63], [65]–[67] by optimizing a predefined family of kernels. DC-programming is one of algorithms for those optimization problems, expressed as (2) [48], [60]

$$\begin{aligned} \min D(\rho) &= \min(g(\rho) - h(\rho) \\ g(\rho) &= \frac{1}{n_1^2 1^T K_{1,1} 1} + \frac{1}{n_2^2 1^T K_{2,2} 1} \\ h(\rho) &= \frac{2}{n_1 n_2 1^T K_{1,2} 1} \end{aligned} \tag{2}$$

where $g(\rho)$ and $h(\rho)$ are convex functions, $\mathbf{1} = \{1, \ldots 1\}^T$, n the number of data, and $[\mathbf{K}]_{\mathbf{i},\mathbf{j}} = k(x_i(x_i x_j)$ for either class 1 or 2. This means the DC functions $D(\rho)$ can be a nonconvex function. Thus, DC-programming algorithm can be applied to both nonconvex and convex problems [48], [53], [60], [62].

To solve the nonconvex problem, Neumann *et al.* [60] adopt DC functions, which minimize the difference between convex functions as the kernel choice method. Argyriou *et al.* [53] propose that the DC-programming algorithm, based on minimax optimization problems involved in the greedy algorithm. Reference [53] has a significant advantage since choosing the kernel among basic kernels in advance is not required.

4) Convex Optimization Algorithm: The problem for choosing the kernel can be reformulated as a manageable convex optimization problem [54], [56], [63], [64]. References [54] and [56] addressed the kernel selection in the kernel Fisher discriminant analysis to maximize a Fisher discriminant ratio (FDR). The basic concept of adopting convex optimization is (3) as follows [54], [56]:

$$\max F_\lambda^*(K) \quad \text{s. t. } K = \theta K_1 + (1-\theta) K_2 \tag{3}$$

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 [54].

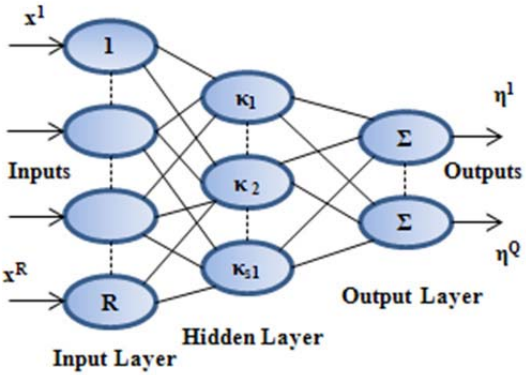

Fig. 1. General RBF-NN [84]. The input and output layers perform functions similar to a normal NN. The hidden layer uses the RBF kernel, denoted as its activation function, as shown in (4).

TABLE III
NN APPROACHES RELATED TO KERNEL ALGORITHM

Algorithm Type	Relevant studies	Representative accuracy
Back-propagation	[71], [72], [82], [83]	83.6 ± 10.8
Recurrent	[80], [81]	91.6 ± 10.5
Radial Basis Function (RBF)	[69], [70], [72], [73], [74]	90.0 ± 2.5
Non-traditional	[75], [76], [77], [78], [79]	96.7 ± 4.6

B. 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: 1) NN; 2) SVM; and 3) PCA.

1) Neural Network: NN is a very successful technique, which uses input/output components that are trained to perform nonlinear classification [68]. This structured approach allows KA to be incorporated within NN in different ways [69]–[83].

Some studies [69]–[74] embed KA inside of the normal NN structure. A common example of combining the NN with KA is the NN RBF-NN [84]. This technique successfully uses a RBF kernel algorithm for the activation functions of the hidden layer in the form

$$\eta(x, w) = \sum_{i=1}^{S1} \kappa_i(\|x - x_i\|) \tag{4}$$

where η is output and x is input with the hidden layer κ.

Other nontraditional KA used in the general structure of the NN can be found in [75]–[79]. While commonly standalone, NN techniques such as back-propagation and recurrence functions combine to train and expand the use of NN with KA in [71], [72], and [80]–[83]. Table III shows the key references of back-propagation, recurrence, RBF, and some nontraditional techniques in conjunction with KA. Based on Table III, many nontraditional approaches are proposed for reaching the best classification accuracy of each study listed

TABLE IV
SVM APPROACHES RELEVANT TO SOLVE OPTIMIZATION PROBLEM

Algorithm	Relevant studies	Representative accuracy
Sequential Minimal Optimization (SMO)	[96], [97]	N.A.
Decomposition methods	[98], [99]	99.9 ± 0.0
Other methods	[100], [101]	92.3 ± 10.1

in the relevant studies. The representative accuracy consists of the mean with one standard deviation among these relevant studies.

A comparison among the KA and NN studies are conducted. Some NN [72], [81], [82] outperform non-KA-based NN. In the comparison of accuracy, more common KA techniques outperform standalone ANN [85]–[88]. None of these references provide definitive evidence that KA or NN is generally better for all situations.

2) Support Vector Machine: SVM originates from statistical learning theory developed in [89], as a binary classifier. SVM is developed as a margin classifier [90]. The task in SVM is to find the largest margin separating the classifier hyper plane [91]. Equation (5) is an optimization problem that involves cross product of training data

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \mathbf{x_i} \cdot \mathbf{x_j} \tag{5}$$

where $\mathbf{x}$ is input, α is a Lagrange multiplier, and y is the label for the distance criterion L. Equation (5) is a formulation of SVM for linearly separable data. The cross product is also called linear kernel [92]. In order for SVM to work in nonseparable (i.e., nonlinear) data, the data are mapped into a higher dimensional feature space, defined by the kernel function [93]. Kernel function is the inner product in the feature space and in the form of

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j). \tag{6}$$

Tolerance to error needs to be introduced. The SVM learning task then becomes to minimize

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \tag{7}$$

subjected 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 (7) is a quadratic programming problem and can be solved using standard quadratic programming solver, called the C-SVM formulation [94]. A review on different methods to solve the optimization problem is discussed in [95]. Other methods, sequential minimal optimization, and decomposition methods and others to solve the optimization problems are shown in Table IV. This table shows SVM approaches consistently reach high classification accuracy.

Among the kernel functions used for SVM [102], [103], the three most common kernel functions used are polynomial, Gaussian, and sigmoid function. SVM shows competitive performance for classification, but not outperforms other techniques in regression problems [104]. Recent SVM developments introduce a new variant called twin SVM [105], [106], 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 other.

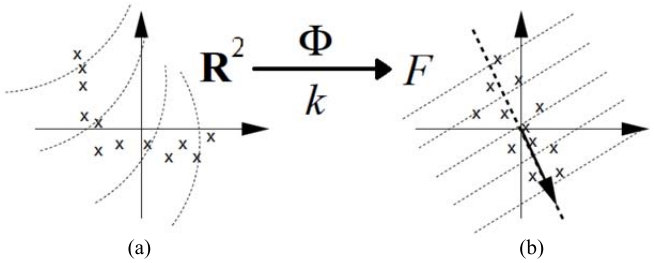

Fig. 2. Principal of kernel PCA. (a) Nonlinear in input space. (b) High-dimensional feature space corresponding to nonlinear input space. The contour lines of constant projections enable to generate the principal eigenvectors. Kernel PCA does not actually require Φ to be mapped into F. All necessary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

3) Principal Component Analysis: PCA, also known as Karhunen–Loève transform or proper orthogonal decomposition, is a mathematical method for reducing dimension of feature space by representing in the orthogonal eigenvector space, as shown in Fig. 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 data set 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 KA [107] in the form of higher dimension—making KPCA a nonlinear component analysis tool. Bouveyron *et al.* [108] conduct an evaluation of PCA and KPCA in comparison with other similar methods for high-dimensional data clustering. In [109], 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 principal components with maximum dependence on the response variables, which can be achieved by

$$\arg\max_{U} \quad tr(U^T X H L H X^T U) \quad \text{s.t. } U^T U = \mathbf{I} \tag{8}$$

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

Schölkopf *et al.* [107] have introduced KA into PCA. Yang *et al.* [110] extend Fisher discriminant analysis to associate kernel methods by incorporating KPCA and Fisher LDA. In LDA algorithm, first the feature space is transformed by KPCA, then, between- 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.

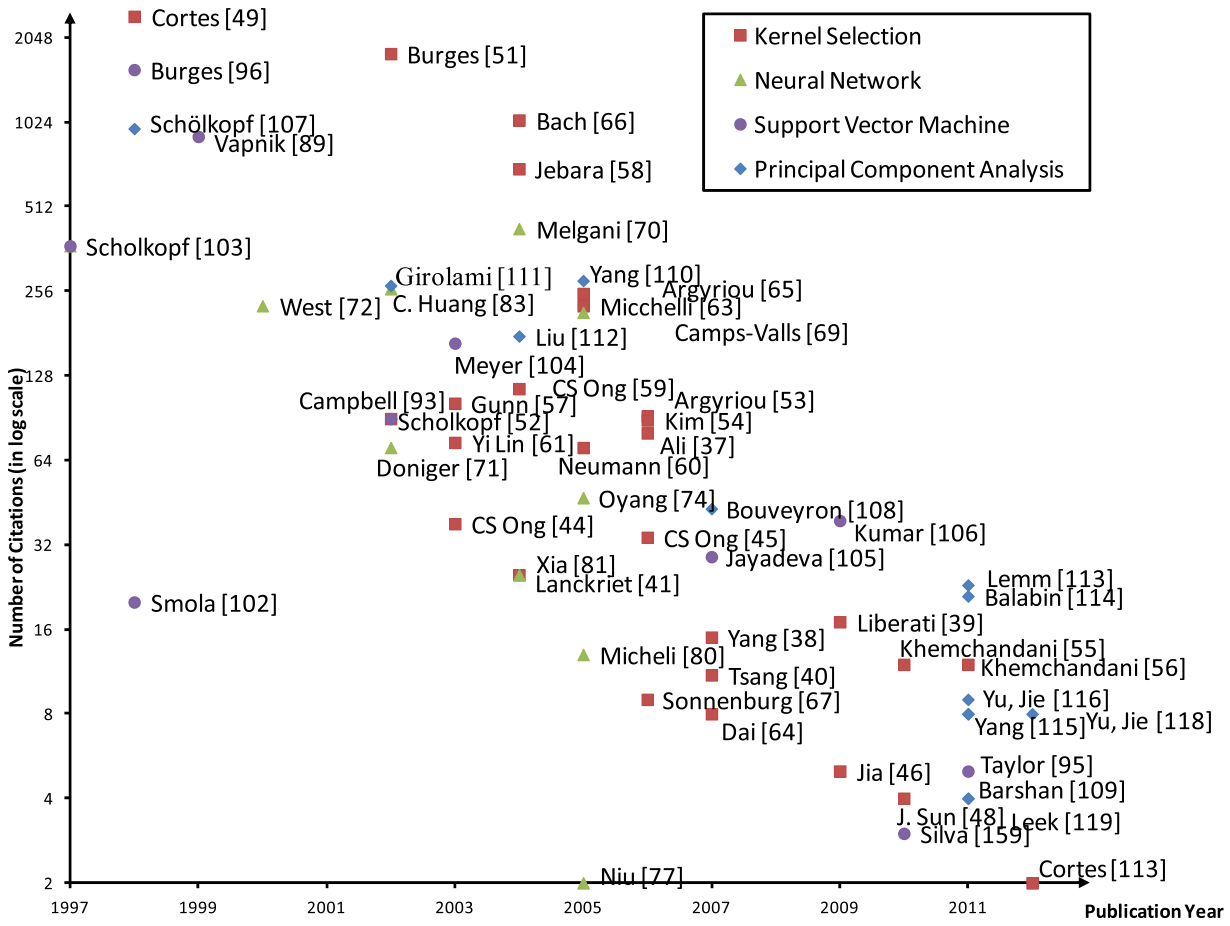

Fig. 3. Significant research publications in various fields of KA and kernel selection for offline learning. Works related to NN, SVM, PCA, and kernel selection are displayed according to citation number versus publishing year.

Girolami [111] 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

$$Tr\left(S_W^{\theta}\right) = 1 - \sum_{k=1}^{\mathrm{K}} \gamma_k \Re(x|C_k) \tag{9}$$

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

Liu [112] applies 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's condition to compute corresponding Gram Matrix. Lemm *et al.* [113] use KPCA in brain imaging and makes an analytic comparison with LDA. KPCA is also used for feature space dimension reduction in biodiesel analysis [114], characterization of global germplasm [115], and bioprocess monitoring for its derivatives [116]. Zhang and Ma [117] use multiscale KPCA for fault detection in nonlinear chemical processes. Yu [118] develops a Gaussian mixture model for fault detection and diagnosis of nonlinear chemical processes. These studies are listed in Table V. All these studies are shown in Fig. 3 with the publication year and number of citations. Table V shows that PCA approaches cover many applications, with relatively high performances.

TABLE V
PCA APPROACHES RELATED TO KERNEL ALGORITHM

Field of Application	Relevant studies	Representative accuracy
RADAR Detection/ Prediction	[114],[120],[123]	94.6 ± 7.5
Sensor/Fault detection	[116],[117],[122],[124]	96.1 ± 5.4
Prediction Analysis/ Algorithm	[108],[109],[111],[118], [125],[127]	96.7 ± 3.4
Computer Vision/Biometrics	[110],[112],[113],[115], [119],[126]	99.1 ± 1.2
Geography/Satellite	[121]	88.0

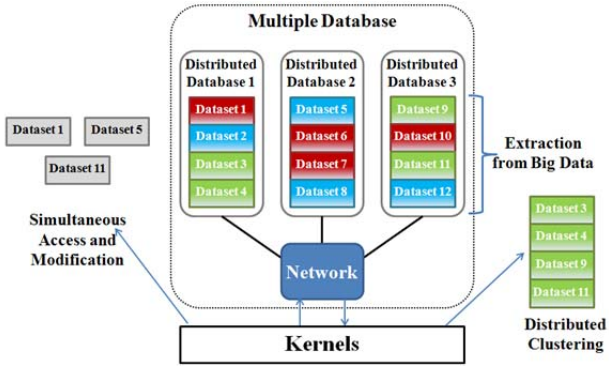

Fig. 4. Composition of the multiple databases. The multiple databases include distributed and single databases, which are stored on different storage spaces connected within a network. Three key attributes of the multiple databases are: 1) simultaneous access and modification; 2) extraction from huge amount; and 3) distributed clustering/classification.

III. Distributed Database With Kernel

Various KAs 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 Section III-A–C.

A. Multiple Database Representation

Providing enhanced multiple databases to users has received a lot of attention in the field of data engineering [128]–[134]. As shown in Fig. 4, the multiple databases consist of various data sets in the forms of distributed and single databases. They are sometimes separately stored on multiple storage spaces in a network [129], [130], [135].

The multiple databases have the following three key attributes. First, users of multiple databases can access and modify data in the distributed database simultaneously [129], [130]. Second, they can extract what they need among a huge amount of data in the distributed environments. Third, all data stored in multiple databases can be clustered/classified [131]–[134].

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 [131]–[134], [136]. Hence, the studies of multiple databases can be linked to multiple- or meta-learning, clustering/classification problems. KA-based machine learning can be effective solutions of those issues due to KA flexibility [37], [55], [129].

B. Kernel Selections Among Heterogeneous Multiple Databases

Sets of KA can be used to sort distributed databases for quick and efficient access. References [137]–[140] show that kernel SVM can be used to decrease the total loading time of large distributed data sets at the cost of increased processing. 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 data set [137]. That is why kernels themselves are distributed

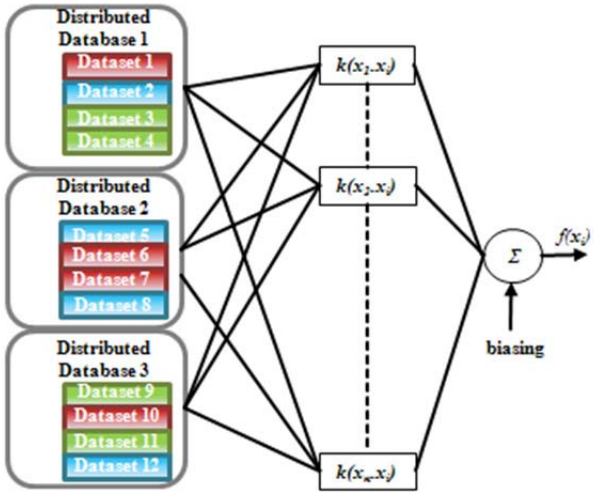

Fig. 5. Several pregenerated kernels that are trained to access different features in several distributed databases. The output of this kernel network is a sum of the kernel functions that maps to the data of interest. The summation and biasing make this method behave like a NN [140].

TABLE VI
Distributed Database Application-Wise Concentration

Distributed Database Application Field	NN	SVM	PCA
Anomaly/Fault detection	[132]	[138], [139], [145], [131], [154]	[135]
	100	90.2 ± 10.7	100
Medical field/Biometrics/Computer Vision	[130]	[143], [148], [128], [134], [159], [151]	[133],[153]
	96.2	98.1 ± 1.1	91 ± 8.5
Economic Modeling/ Forecasting/Prediction	[129]	[140], [136], [156], [150], [149], [152]	[147]
	95	90.0 ± 10.7	99.2

and the computational power is divided. An early method of such distributed KA is described in [141]. Other more recent methods can be observed in [140], and [142]–[146]. Fig. 5 shows an example where multiple kernels (MKs) can be used in a network setting, in which each kernel holds different configurations of the data set [140]. The nonkernel SVM has a faster run-time, but less precision than that of KA distributed SVM [137], [148].

C. 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 VI, some relevant KA applications of distributed databases are listed below. Table VI shows that many SVM approaches exist for a broad range of applications.

We show specific applications below to demonstrate how KA distributed databases using SVM-related algorithms are

practically implemented in the real-world setting: 1) market analysis; 2) wireless sensor networks; and 3) distributed text classification.

1) Market Analysis: One field that will benefit from distributed computing task is the study of market behavior since it involves large and diverse data set from various sources. In [150], customer behavior prediction is stored in separate places so that the proposed machine learning technique can be implemented in a distributed manner. 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 paper is to classify customer (returning or reluctant) and willingness of classified customer (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 compared with MK-SVM.

2) Wireless Sensor Networks: This paper implements SVM in classification problem over distributed WSN [156], based on 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 Lagrange multiplier corresponding to local data set. The scheme tests the data from UCI machine learning repository [158]. Results show no significant difference in effectiveness compared with original SVM, but the scheme minimizes exchange of information between the networks (and hence increase security), and is scalable for large-scale network.

3) Distributed Text Classification: Another implementation of SVM is applied to classification of text in distributed computing [159]. In addition to SVM, the relevance vector machines 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 acyclic graph (DAG). The DAG is then optimized for distributed computing [160]. This distributed computing framework shows that performance of classification task using distributed scheme is better compared with using sequential processing.

IV. Kernel Online Learning

In contrast to offline learning, online learning learns 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 [162]. Because new data are being 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 would result in unbounded memory consumption and growth in computational complexity [162]–[165].

In Section IV-A, we illustrate the general principles of kernel-based online learning algorithms, and in Section IV-B, we demonstrate how an online kernel framework is adopted into some traditionally developed machine learning techniques such as: 1) NN; 2) SVM; and 3) PCA. Finally, in Section IV-C, we establish the link between the online learning and prediction.

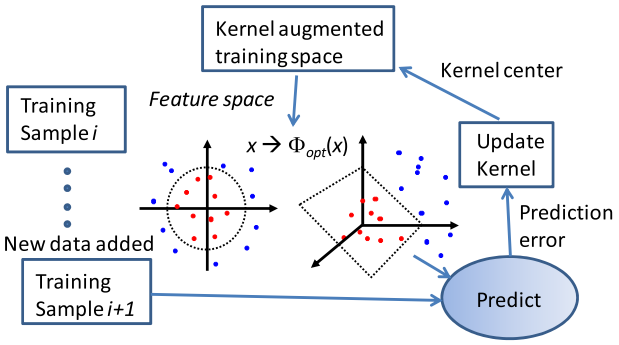

Fig. 6. Online kernel-based learning algorithm. As a new training sample is added, the kernel is updated based on the online learning algorithms described in the following section.

A. 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. 6. This growth is superlinear when batch-type offline techniques are directly applied [166], and linear in the case of naïve application of incremental methods [167]. To remedy the data size, a variety of techniques is used to discard or ignore the data whether any new information is added to the classifier or filter [162], [167].

The so-called growing sum problem [164] is tackled by a variety of means. Since the computational complexity grows linearly over time, new algorithms are required to continuously process data over prolonged periods of time. The NORMA algorithm [162] uses a forgetting factor such that samples are weighted in a sliding-window fashion, with older samples weighted less than newer ones. This technique assumes that we are only interested in the statistical structure of the newest samples in a nonstationary series.

Alternatively, when it is desired to consider older data, sparsification techniques may be used [167]. In these methods, only data that adds new information to the classifier or filter is added to the support. Various information criteria are proposed, such as the approximate linear dependency criterion [169], the surprise criterion [170], and the variance criterion [171].

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 [172]. 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 [172].

Other methods are proposed for dealing with the growth problem, including [164], which uses only finite-dimensional

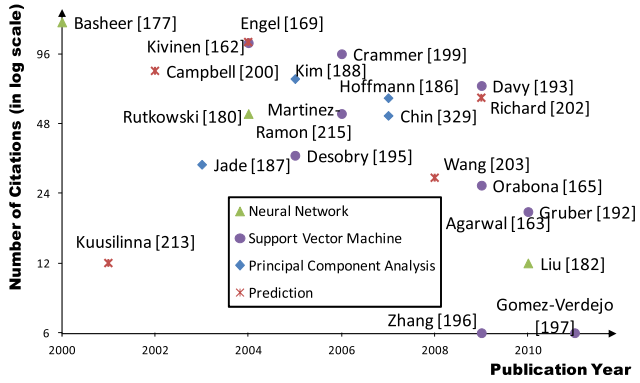

Fig. 7. Key papers for kernel online learning based on the number of citations for the years 1998–2013, for PCA, SVM, NN, ARMA, and FSM.

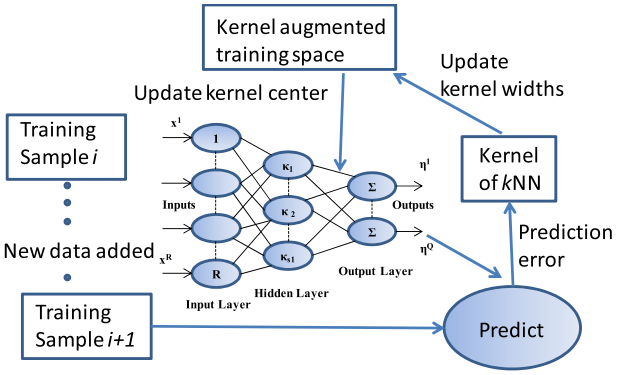

Fig. 8. Online NN learning algorithm. An artificial NN consists of input, hidden, and output layers interconnected with directed weights (w), where w_{ij} denotes the input-to-hidden layer weights at the hidden neuron j and w_{jk} denotes the hidden-to-output layer weights at the output neuron k [178].

kernels and replaces infinite-dimensional kernels with finite-dimensional approximations. This method achieves constant computational complexity over time [164]. Another method is presented in [165], which projects new inputs onto a space spanned by the previous hypothesis. This method is guaranteed to be bounded and outperforms other perceptron-based methods [165].

The complication inherent to online kernel methods is the selection of the kernel itself. Often it is assumed that a suitable function is known *a priori*, while in reality this is not necessarily the case [167], [173]. 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, based on the data [173], [174]. Jin *et al*. [174] use a combination of the perceptron algorithm [175] and the Hedge algorithm [163] to learn both the kernel combination and the classifier, and present both deterministic and stochastic [176] algorithms for doing so.

B. Adopt Online KA Framework Into the Traditionally Developed Machine Learning Techniques

Fig. 7 shows the representative online learning studies on kernel-based online learning using NN, SVM, and PCA.

1) Neural Network: Online NN is designed to have the learning capabilities of online time systems [177]. 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 with the desired output, then adjusting the weights accordingly. A typical kernel online NN is shown in Fig. 8.

Online updates to the weights of a NN relies on algorithms such as the commonly used Gaussian radial bias function (RBF), which resides in the hidden layer of the NN. The centers of the RBF function are initially chosen based on 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 [179]. Kingravi *et al.* [179] propose an online algorithm that connects kernel methods to persistently exciting signals, known as budgeted kernel restructuring for model reference adaptive control.

Similar techniques have been applied to such problems as time-series prediction [180], and pattern classification tasks like facial recognition [162]. NN is frequently applied in engineering applications that require optimization of nonlinear stochastic dynamic systems [181]. Yuen *et al.* [178] propose a kernel-based hybrid NN for online regression of noisy data, combining fuzzy ART and general regression NN models. Performance of kernel NN continues to improve due to the recent development of new algorithms. Liu *et al.* [182] describe kernel affine projection algorithms, which out-performs other recently developed algorithms such as the kernel least-mean-square algorithm in online time-series prediction, nonlinear channel equalization, and nonlinear noise cancellation.

2) Support Vector Machine: Online SVM [192], [193] preserves the skeleton samples based on the geometric characteristics of SVM, as shown in Fig. 9. The SVM classifier is updated when the distances between the 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 since only very limited training samples are maintained in the offline step [192].

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—i.e., 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 [162].

Gruber *et al.* [192] adopt an online SVM to signature variation, using kernel functions based on the longest common subsequences 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.

Online SVM is also widely used for novelty detection, known as anomaly detection. The novelty detection is the

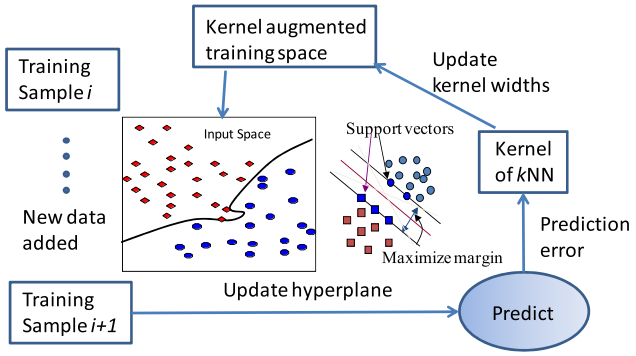

Fig. 9. Scheme for naïve online SVM classification. The input data are mapped to the kernel Hilbert space, and the hyperplane is generated as in offline SVM. If an optimal solution cannot be found, a new kernel is selected. The process repeats for newly incoming data.

process when a learning system recognizes unprecedented inputs [193]. SVMs adapted to online novelty detection are referred to as one-class SVMs (1-SVMs). 1-SVM distinguishes between inputs in the normal class and all other inputs, which are considered outliers or anomalies. Since 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 [182]. This is accomplished by separating the data from the origin using the greatest possible margin [182], [194]. Like other online SVMs, online 1-SVMs typically prune old or redundant data from the classifier to prevent indefinite complexity expansion [193].

Desobry *et al.* [195] 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 $\boldsymbol{x}_t$, $t = 1, 2, \ldots, T$, 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 with 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.* [196] implement a sliding-time-window 1-SVM using quarter-sphere formulation of the SVM, which applies to one-sided nonnegative distributions of features [195]. Rather than fitting a hyperplane to the data, this method fits a quarter-(hyper) sphere, centered at the origin [196]. 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.* [197] create an adaptive one-class SVM (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.* [198] use a modified version of the NORMA algorithm [162] 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.

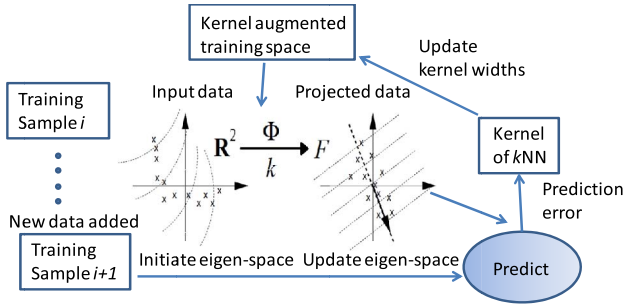

Fig. 10. Online KPCA diagram. In the initial (offline) training phase, the initial kernel parameters and eigenfeature space are calculated. As new training sets are introduced, incremental (online) adjustments are made to the parameters and updated eigenspace is generated.

3) Principal Component Analysis: Online KPCA extracts principal components in the feature space related to nonlinear inputs. The algorithm for extracting the components is (10) expressed mathematically as

$$V^n \bullet \Phi(x) = \sum_{i=1}^{M} \alpha_i^n k(x_i, \bar{x}) \tag{10}$$

where $k(\mathbf{x}_i, \mathbf{x}) = (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}))$ is the kernel function, α_i the eigenvectors, and $\boldsymbol{V}^n$ the corresponding eigenvectors in the feature space [183]. 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 data sets.

Most of the online methods begin with one or more offline learning steps to determine suitable kernel parameters [184]. The incremental KPCA developed by Ozawa *et al.* [185] progresses in two stages: 1) in the initial learning phase, a number of training sets are presented offline and the initial eigenfeature space is found and 2) in the incremental learning phase, the feature space is defined by solving the eigenvalue problem with dimensions determined by the number of independent data. Online KPCA algorithm is shown in Fig. 10.

Hoffman demonstrates the use of online KPCA in novelty detection, specifically applied to recognition of handwritten digits and breast cancer detection [186]. Classification performance is measured against the results from Parzen window density estimator, standard PCA, and one-class 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 [187] and Gaussian noise in images [188].

C. 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 [200], [201] and system modeling [189], [202], 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.

TABLE VII
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

1) Prediction and 2) online learning have traditionally been studies of two independent communities: 1) in signal processing and 2) 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 [163], [202], [203].

As shown in Table VII, the term prediction implies learning how to forecast, i.e., 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, since knowledge of the structure of a system allows one to predict its future behavior [200], [201].

A prediction is a future function as a forecast of the new coming data set will be automatically classified based on the knowledge of offline or online learning. As shown in previous section, there exist 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 data sets are used for testing a future feature space prediction [199], [202] as online learning has been shown.

V. PREDICTION WITH KERNELS

Prediction is represented by top-down model-based representation. In this section, we will show how kernels may enhance the prediction through the four major representations: 1) linear prediction; 2) Kalman filter (KF); 3) finite state model (FSM); and 4) autoregressive moving average (ARMA) model. Finally, we compare them in 5) comparison of four models.

A. Linear Prediction

A linear prediction is a simplified model where future output values are estimated as a linear function (11) of previous values and predictor coefficients, as follows [204]:

$$\hat{x}(n) = a_0 + a_1 x(n-1) + \cdots + a_n x(n-k) = \sum_{i=0}^{k} a_i x(n-i) \quad (11)$$

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

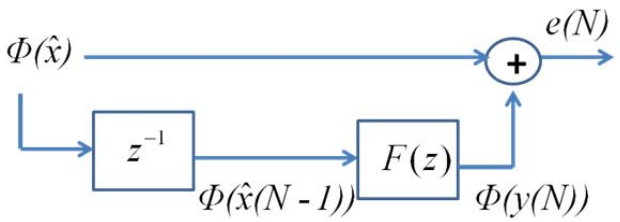

Fig. 11. Linear predictor with an initial kernel mapping. The input data are mapped from its original space to a new Hilbert space where $x(n)$ is linear, by the function Φ to allow for the use of the kernel trick. The model is optimized by minimizing the function $e(N)$, which is the difference in the current and previous predicted value.

The predicted value is a linear combination of previous observations $x(n-k)$ and predictor coefficients a_n, as shown in Fig. 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 [219]. The linear model is widely used in the early stage to compare the prediction performance with other models, e.g., Kalman filtering [204].

B. Kalman Filter

The KF is one of the most commonly used prediction methods in real-time filtering technologies [204]–[211]. KF provides a recursive solution (12) to minimize mean square error within the class of linear estimators, where linear process and measurement equations can be expressed as follows [205]:

$$\hat{x}(t) = Fx(t-1) + Bu(t-1) + W,\ z(t) = H\hat{x}(t) + V \quad (12)$$

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 $E[V(t)V(t)^T] = Q(t)$, respectively. The matrices F, B, W, H, and V are assumed known and possibly time-varying.

Ralaivola and D'Alche-Buc [211] develop a kernel KF (KKF) method for implementing a KF on a nonlinear time series data set. By mapping the input data from a nonlinear space to a Hilbert space (shown in Fig. 12), the Mercer kernel function can be satisfied. In KKF, the predicted position $\hat{x}(t)$ can be derived from the previous state $x(t-1)$ and the current measurement $z(t)$ [204], [209]. 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 [210].

KF can be enhanced to an interactive multiple model filter with constant velocity and constant acceleration [209]. Hong *et al.* [210] suggest the first-order extended KF (EKF) with a kernel-type method can be used to process and update the state estimate.

C. Finite State Model

The FSM, or finite state machine, consists of a finite number of discrete states, only one of which the model can occupy

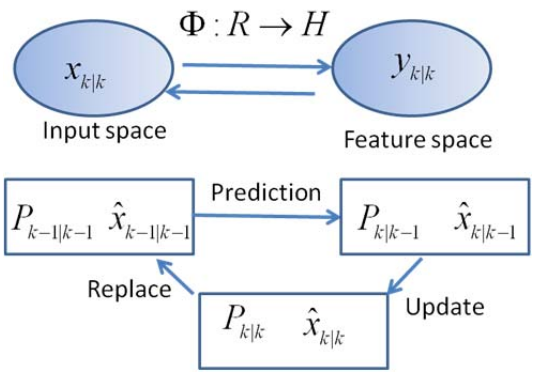

Fig. 12. For processing a nonlinear time series, a kernelized KF is an optimal way to execute the KF. With a nonlinear series x_k and an associated series x_k^{Φ}. x_k is mapped to x_k^{Φ} by the nonlinear map $\Phi : R^d \to H$, where R^d is the input space and H is the feature space, where $[\Phi(x), \Phi(y)] = k(x, y)$.

at any given time, and undergoes a change of state upon a triggering event. Kernel methods have been applied to FSM [212], [218]. Separated-kernel image processing using finite state machines, is an online kernel method used to speed processing time when dealing with large numbers of states, as in processing of grayscale images [212]. 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 ($\delta : S \times I \to S$), and λ is the output function ($\lambda : S \times I \to O$). Most of the 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 [213], [214].

D. Autoregressive Moving Average Model

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. Martìnez-Ramòn *et al.* [215] use the kernel trick to formulate nonlinear SVM-ARMA in reproducing kernel Hilbert space (RKHS). Discrete-time processes (DTPs) 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 DPT are represented by the composite kernel formulation

$$\begin{aligned} K(z_i, z_j) &= F\left\langle \left[\phi_y(y_{i-1})^T, \phi_u(u_i)^T\right]^T, \right. \\ &\qquad \left. \left[\phi_y(y_{j-1})^T, \phi_u(u_j)^T\right]^T \right\rangle \\ &= K_y(y_{i-1}, y_{j-1}) + K_u(u_i, u_j) \end{aligned} \tag{13}$$

where $\boldsymbol{y}$ and $\boldsymbol{u}$ are the input and output DTPs, respectively. Shpigelman *et al.* [216] describe a kernel ARMA (KARMA) for the tracking of hand movement in 3-D space.

As shown in Fig. 13, ARMA consists of two models: 1) an autoregressive (AR) model represented by a weighted sum of the present and past positions with a polynomial order p and 2) 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 pAR and qMA [216], [217].

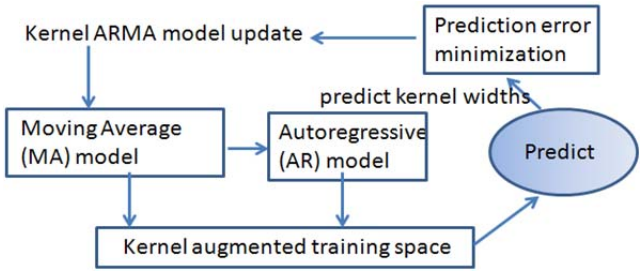

Fig. 13. KARMA model with online learning for tracking of hand movement.

TABLE VIII
COMPARISONS OF PREDICTION ERROR

Methods	Error range	Mean ±SDV	Data type
Linear [219]	[0.0,0.05]	0.026±0.018	k-fold cross-validation
KF [211]	[0.03, 0.73]	10.37±10.62	3 noisy datasets
FSM [212][218]	[0.0, 0.135]	0.042±0.042	10 uniform
ARMA [215]	[-2.9, -1.7]	-2.3±0.60	Normalized validation

TABLE IX
COMPARISONS OF COMPUTATIONAL SPEED

Methods	speed	main variables
Linear [219]	$O(Jn^3)$	J: base functions n: feature dimension
KF [211]	$O(l^3)$	l: time series length
FSM [212][218]	8.34± 7.2 frame/sec	morphological structure
ARMA [215]	$O(n)$	n: data

E. Comparison of Four Models

The four prediction models, linear prediction, KF, FSM, and ARMA, are quantitatively compared to illustrate the overall advantages of each model, as shown in Table VIII. 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 was used to convert each quantitative error to a universal metric for proper comparison. Based on these various conditions, we cannot definitively state, which method performs better than 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 IX, the computational speed to describe the overall computational complexity. In general, the number of data sets 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 [219] were the number of base functions and total amount of feature dimensions.

VI. Conclusion

In this survey paper, we have shown the state-of-the-art techniques related 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 the 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 sometime 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.

Acknowledgment

This work would not be possible without the help of many of the past and present members of the sensory intelligent laboratory and graduate students who took classes on pattern recognition.

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, May 2007.

[2] B. J. Kim, I. K. Kim, and K. B. Kim, "Feature extraction and classification system for nonlinear and online data," in *Advances in Knowledge Discovery and Data mining*, vol. 3056. Berlin, Germany: Springer-Verlag, 2004, pp. 171–180.

[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, NY, USA: Springer-Verlag, 2000.

[5] R. Duda, P. Hart, and D. Stock, *Pattern Classification*, 2nd ed. New York, NY, USA: Wiley, 2001.

[6] T. Briggs and T. Oates, "Discovering domain specific composite kernels," in *Proc. 20th Nat. Conf. Artif. Intell.*, 2005, pp. 732–738.

[7] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, "On kernel target alignment," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001, pp. 367–373.

[8] 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.

[9] X. Jiang, R. Snapp, Y. Motai, and X. Zhu, "Accelerated kernel feature analysis," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2006, pp. 109–116.

[10] 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.

[11] 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, no. 2, pp. 460–474, Mar. 2005.

[12] H. Xiong, Y. Zhang, and X. W. Chen, "Data-dependent kernel machines for microarray data classification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 4, no. 4, pp. 583–595, Dec. 2007.

[13] Y. Motai and H. Yoshida, "Principal composite kernel feature analysis: Data-dependent kernel approach," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1863–1875, Aug. 2013.

[14] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[15] 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, Jun. 2008.

[16] 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, Cybern. B, Cybern.*, vol. 36, no. 4, pp. 873–886, Aug. 2006.

[17] Y. M. Li, "On incremental and robust subspace learning," *Pattern Recognit.*, vol. 37, no. 7, pp. 1509–1518, 2004.

[18] Y. Kim, "Incremental principal component analysis for image processing," *Opt. Lett.*, vol. 32, no. 1, pp. 32–34, 2007.

[19] B. J. Kim and I. K. Kim, "Incremental nonlinear PCA for classification," in *Knowledge Discovery in Databases*, vol. 3202. Berlin, Germany: Springer-Verlag, 2004, pp. 291–300.

[20] B. J. Kim, J. Y. Shim, C. H. Hwang, I. K. Kim, and J. H. Song, "Incremental feature extraction based on empirical kernel map," in *Foundations of Intelligent Systems*, vol. 2871. Berlin, Germany: Springer-Verlag, 2003, pp. 440–444.

[21] 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.

[22] T. J. Chin and D. Suter, "Incremental kernel principal component analysis," *IEEE Trans. Image Process.*, vol. 16, no. 6, pp. 1662–1674, Jun. 2007.

[23] L. Winter, Y. Motai, and A. Docef, "On-line versus off-line accelerated kernel feature analysis: Application to computer-aided detection of polyps in CT colonography," *Signal Process.*, vol. 90, no. 8, pp. 2456–2467, 2010.

[24] Y. Motai, "Synthesized articulated behavior using space-temporal online principal component analysis," *Adv. Robot.*, vol. 21, no. 13, pp. 1503–1520, 2007.

[25] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2002.

[26] 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 Artif. Intell.*, Nov. 2003, pp. 142–148.

[27] X. W. Chen, "Gene selection for cancer classification using bootstrapped genetic algorithms and support vector machines," in *Proc. IEEE Bioinformat. Conf.*, Aug. 2003, pp. 504–505.

[28] F. A. Sadjadi, "Polarimetric radar target classification using support vector machines," *Opt. Eng.*, vol. 47, no. 4, p. 046201, 2008.

[29] M. Awad and Y. Motai, "Dynamic classification for video stream using support vector machine," *J. Appl. Soft Comput.*, vol. 8, no. 4, pp. 1314–1325, 2008.

[30] S. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Netw.*, vol. 12, no. 6, pp. 783–789, 1999.

[31] B. Souza and A. de Carvalho, "Gene selection based on multi- class support vector machines and genetic algorithms," *Genet. Molecular Res.*, vol. 4, no. 3, pp. 599–607, 2005.

[32] M. Awad, Y. Motai, J. Nappi, and H. Yoshida, "A clinical decision support framework for incremental polyps classification in virtual colonoscopy," *Algorithms*, vol. 3, no. 1, pp. 1–20, 2010.

[33] S. Li and J. Lu, "Face recognition using the nearest feature line method," *IEEE Trans. Neural Netw.*, vol. 10, no. 2, pp. 439–443, Mar. 1999.

[34] 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 Netw.*, vol. 3. Jul. 2003, pp. 1702–1707.

[35] 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, Apr. 2010.

[36] 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, Dec. 2008.

[37] S. Ali and K. A. Smith-Miles, "A meta-learning approach to automatic kernel selection for support vector machines," *Neurocomputing*, vol. 70, nos. 1–3, pp. 173–186, Dec. 2006.

[38] 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, Sep. 2007.

[39] C. Liberati, J. A. Howe, and H. Bozdogan, "Data adaptive simultaneous parameter and kernel selection in kernel discriminant analysis (KDA) using information complexity," *J. Pattern Recognit. Res.*, vol. 4, no. 1, pp. 119–132, Feb. 2009.

[40] 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, Jan. 2006.

[41] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, Jan. 2004.

[42] 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. Instrum. Meas., Sensor Netw. Autom.*, Aug. 2012, pp. 222–227.

[43] T. Jebara, "Multitask sparsity via maximum entropy discrimination," *J. Mach. Learn. Res.*, vol. 2, pp. 75–110, Feb. 2011.

[44] C. S. Ong, E. J. Smola, and R. C. Williamson, "Hyperkernels," in *Advances in Neural Information Processing Systems*, vol. 15. Menlo Park, CA, USA: AAAI Press, 2003.

[45] C. S. Ong, A. J. Smola, and R. C. Williamson, "Learning the kernel with hyperkernels," *J. Mach. Learn. Res.*, vol. 6, pp. 1045–1071, Jan. 2005.

[46] L. Jia and S. Liao, "Hyperkernel construction for support vector machines," in *Proc. 4th Int. Conf. Natural Comput.*, vol. 2. 2008, pp. 76–80.

[47] R. Kondor and T. Jebara, "Gaussian and Wishart hyperkernels," in *Advances in Neural Information Processing Systems*, vol. 19. Cambridge, MA, USA: MIT Press, Dec. 2006, pp. 729–736.

[48] J. Sun, C. Zheng, X. Li, and Y. Zhou, "Analysis of the distance between two classes for tuning SVM hyperparameters," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 305–318, Feb. 2010.

[49] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *J. Mach. Learn. Res.*, vol. 13, pp. 795–828, Mar. 2012.

[50] Y. Motai and H. Yoshida, "Principal composite kernel feature analysis: Data-dependent kernel approach," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1863–1875, Aug. 2013.

[51] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, Jun. 1998.

[52] B. Scholkopf and A. J. Smola, *Learning With Kernels*. Cambridge, MA, USA: MIT Press, 2002.

[53] A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil, "A DC-programming algorithm for kernel selection," in *Proc. 23rd Int. Conf. Mach. Learn.*, Jun. 2006, pp. 41–48.

[54] S.-J. Kim, A. Magnani, and S. Boyd, "Optimal kernel selection in kernel Fisher discriminant analysis," in *Proc. 23rd Int. Conf. Mach. Learn.*, Jun. 2006, pp. 465–472.

[55] R. Khemchandani, Jayadeva, and S. Chandra, "Optimal kernel selection in twin support vector machines," *Optim. Lett.*, vol. 3, no. 1, pp. 77–88, Jan. 2009.

[56] R. Khemchandani, Jayadeva, and S. Chandra, "Learning the optimal kernel for Fisher discriminant analysis via second order cone programming," *Eur. J. Operat. Res.*, vol. 203, no. 3, pp. 692–697, 2010.

[57] S. R. Gunn and J. S. Kandola, "Structural modelling with sparse kernels," *Mach. Learn.*, vol. 48, nos. 1–3, pp. 137–163, 2002.

[58] T. Jebara, "Multi-task feature and kernel selection for SVMs," in *Proc. 21st Int. Conf. Mach. Learn.*, Jul. 2004.

[59] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, Jan. 2011.

[60] J. Neumann, C. Schnörr, and G. Steidl, "SVM-based feature selection by direct objective minimisation," in *Pattern Recognition* (Lecture Notes in Computer Science), vol. 3175. Berlin, Germany: Springer-Verlag, 2004, pp. 212–219.

[61] Y. Lin and H. H. Zhang, "Component selection and smoothing in smoothing spline analysis of variance models," Inst. Statist., North Carolina State Univ., Raleigh, NC, USA, Tech. Rep., Jan. 2003.

[62] 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.

[63] C. A. Micchelli and M. Pontil, "Learning the kernel function via regularization," *J. Mach. Learn. Res.*, vol. 6, pp. 1099–1125, Jul. 2005.

[64] G. Dai and D. Yeung, "Kernel selection for semi-supervised kernel machines," in *Proc. 24th Int. Conf. Mach. Learn.*, Jun. 2007, pp. 185–192.

[65] A. Argyriou, C. A. Micchelli, and M. Pontil, "Learning convex combinations of continuously parameterized basic kernels," in *Proc. 18th Conf. Learn. Theory*, Jun. 2005, pp. 338–352.

[66] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proc. 21th Int. Conf. Mach. Learn.*, 2004, pp. 6–13.

[67] S. Sonnenburg, G. Rätsch, and C. Schäfer, "A general and efficient multiple kernel learning algorithm," in *Advances in Neural Information Processing Systems*, vol. 18. Cambridge, MA, USA: MIT Press, 2006, pp. 1273–1280.

[68] S. O. Haykin, "Models of a neuron," in *Neural Networks and Learning Machines*, 3rd ed. New York, NY, USA: Macmillan, 1994, pp. 8–13.

[69] 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, Jun. 2005.

[70] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[71] 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.

[72] D. West, "Neural network credit scoring models," *Comput. Operat. Res.*, vol. 27, nos. 11–12, pp. 1131–1152, Sep. 2000.

[73] B. Scholkopf *et al.*, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2758–2765, Nov. 1997.

[74] 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, Jan. 2005.

[75] 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, Feb. 2012.

[76] Y. Xiao and Y. He, "A novel approach for analog fault diagnosis based on neural networks and improved kernel PCA," *Neurocomputing*, vol. 74, no. 7, pp. 1102–1115, Mar. 2011.

[77] 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 Int. Conf. Mach. Learn. Cybern.*, vol. 7. Aug. 2005, pp. 4260–4265.

[78] 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 Int. Conf. Natural Comput.*, vol. 3. Oct. 2008, pp. 236–240.

[79] Y. Xiao and L. 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, Feb. 2012.

[80] 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, Mar. 2005.

[81] Y. Xia and J. Wang, "A one-layer recurrent neural network for support vector machine learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 1261–1269, Apr. 2004.

[82] K. H. Jang *et al.*, "Comparison of survival predictions for rats with hemorrhagic shocks using an artificial neural network and support vector machine," in *Proc. 33rd Int. Conf. Eng. Med. Biol. Soc. (EMBS)*, Aug./Sep. 2011.

[83] 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, Feb. 2002.

[84] K. A. 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 *Proc. Int. Conf. Soft Comput. Pattern Recognit.*, 2009.

[85] 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, no. 2, pp. 621–633, 2013.

[86] 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. Issues*, vol. 7, no. 3, pp. 1–10, May 2010.

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

[88] 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, Mar. 2005.

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

[90] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifier," in *Proc. 5th ACM Workshop Comput. Learn. Theory*, Jul. 1992, pp. 144–152.

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

[92] C. Campbell and Y. Ying, *Learning With Support Vector Machines*. San Rafael, CA, USA: Morgan & Claypool, 2011, p. 6.

[93] C. Campbell, "Kernel methods: A survey of current techniques," *Neurocomputing*, vol. 48, nos. 1–4, pp. 63–84, 2002.

[94] A. Sanchez and V. David, "Advanced support vector machines and kernel methods," *Neurocomputing*, vol. 55, no. 1, pp. 5–20, 2003.

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

[96] 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, USA: MIT Press, 1998.

[97] G. W. Flake and S. Lawrence, "Efficient SVM regression training with SMO," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 271–290, 2002.

[98] P. Laskov, "An improved decomposition algorithm for regression support vector machines," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 315–350, 2002.

[99] E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Proc. IEEE Neural Netw. Signal Process.*, Sep. 1997, pp. 276–285.

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

[101] T. T. Friess, N. Cristianini, and C. Campbell, "The kernel-adatron algorithm: A fast and simple learning procedure for support vector machines," in *Proc. 15th Int. Conf. Mach. Learn.*, 1998, pp. 188–196.

[102] A. J. Smola and B. Schölkopf, "From regularization operators to support vector kernels," *Advances In Neural Information Processing Systems*, vol. 10. Cambridge, MA, USA: MIT Press, 1998, pp. 343–349.

[103] B. Schölkopf *et al.*, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2758–2765, Nov. 1997.

[104] D. Meyer, F. Leisch, and K. Hornik, "The support vector machine under test," *Neurocomputing*, vol. 55, nos. 1–2, pp. 169–186, Sep. 2003.

[105] 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, May 2007.

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

[107] 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.

[108] C. Bouveyron, S. Girard, and C. Schmid, "High-dimensional data clustering," *Comput. Statist. Data Anal.*, vol. 52, no. 1, pp. 502–519, Sep. 2007.

[109] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recognit.*, vol. 44, no. 7, pp. 1357–1371, Jul. 2011.

[110] J. Yang, A. F. Frangi, J.-Y. Yang, and Z. Jin, "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, Feb. 2005.

[111] S. Girolami "Mercer kernel-based clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 780–784, May 2002.

[112] C. 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, May 2004.

[113] 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.

[114] 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 Fuels*, vol. 25, no. 5, pp. 2373–2382, 2011.

[115] X. Yang *et al.*, "Characterization of a global germplasm collection and its potential utilization for analysis of complex quantitative traits in maize," *Molecular Breeding*, vol. 28, no. 4, pp. 511–526, Dec. 2011.

[116] J. Yu, "Nonlinear bioprocess monitoring using multiway kernel localized Fisher discriminant analysis," *Ind. Eng. Chem. Res.*, vol. 50, no. 6, pp. 3390–3402, Mar. 2011.

[117] 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, Jan. 2011.

[118] 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, Jan. 2012.

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

[120] G. Liu, X. Sun, K. Fu, and H. Wang, "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.

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

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

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

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

[125] 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, Feb. 2013.

[126] S. Liwicki, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "Euler principal component analysis," *J. Comput. Vis.*, vol. 101, no. 3, pp. 498–518, Feb. 2013.

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

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

[129] F. Dinuzzo, G. Pillonetto, and G. D. Nicolao, "Client–server multitask learning from distributed datasets," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 290–303, Feb. 2011.

[130] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[131] A. Kitamoto, "Typhoon analysis and data mining with kernel methods," in *Pattern Recognition with Support Vector Machines*. New York, NY, USA: Springer-Verlag, 2002, pp. 237–249.

[132] 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. Ind. Electron.*, vol. 53, no. 4, pp. 1298–1308, Jun. 2006.

[133] 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," *Chemometrics Intell. Lab. Syst.*, vol. 90, no. 2, pp. 123–131, 2008.

[134] B. L. Milenova, J. S. Yarmus, and 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)*, 2005.

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

[136] 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, Apr. 2005.

[137] C. J. Lin, "Large-scale machine learning in distributed environments," in *Proc. ACM Int. Conf. Multimedia Retr. (ICMR)*, Jun 2012.

[138] Z. A. Zhu, W. Z. Chen, G. Wang, C. G. Zhu, and Z. Chen, "P-packSVM: Parallel primal gradient descent kernel SVM," in *Proc. 9th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 677–686.

[139] 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. Sep. 2008.

[140] 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, Dec. 2012.

[141] N. A. Syed, H. Liu, and K. K. Sung, "Handling concept drifts in incremental learning with support vector machines," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 1999.

[142] 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, nos. 1–3, pp. 394–400, Dec. 2010.

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

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

[145] H. Xu, Y. Wen, and J. Wang, "A fast-convergence distributed support vector machine in small-scale strongly connected networks," in *Proc. Front. Electr. Electron. Eng.*, Jul. 2011.

[146] 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 Int. Conf. Mach. Learn.*, 1998.

[147] 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.

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

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

[150] 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, Nov. 2012

[151] 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, Sep. 2011.

[152] J. Cheng, J. Qian, and Y. Guo, "A distributed support vector machines architecture for chaotic time series prediction," in *Proc. Neural Inform. Process.*, pp. 892–899, 2006.

[153] P. Phoungphol and Z. Yanqing, "Multi-source kernel k-means for clustering heterogeneous biomedical data," in *Proc. IEEE Int. Conf. Bioinf. Biomed. Workshops*, Nov. 2011, pp. 223–228.

[154] 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, Syst. Comput.*, Oct. 2008, pp. 246–250.

[155] K. Buza, P. B. Kis, and A. Buza, "Graph-based clustering based on cutting sets," in *Proc. 15th IEEE Int. Conf. Intell. Eng. Syst.*, Jun. 2011, pp. 143–149.

[156] W. Dongli, Z. Jianguo, Y. Zhou, and J. Li, "A scalable support vector machine for distributed classification in ad hoc sensor networks," *Neurocomputing*, vol. 74, nos. 1–3, pp. 394–400, Dec. 2010.

[157] 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, Jun. 2004.

[158] C. Blake, E. Keogh, and C. Merz. (1988). *UCI Repository of Machine Learning Databases* [Online]. Available: http://archive.ics.uci.edu/ml/

[159] C. Silva, U. Lotric, B. Ribeiro, and A. Dobnikar, "Distributed text classification with an ensemble kernel-based learning approach," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 3, pp. 287–297, May 2010.

[160] (1995). *Foster* [Online]. Available: http://www.mcs.anl.gov/itf/dbpp/

[161] S. R. Upadhyaya, "Parallel approaches to machine learning—A comprehensive survey," *J. Parallel Distrib. Comput.*, vol. 73, no. 3, pp. 284–292, Mar. 2013.

[162] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[163] S. Agarwal, V. V. Saradhi, and H. Karnick, "Kernel-based online machine learning and support vector reduction," *Neurocomputing*, vol. 71, nos. 7–9, pp. 1230–1237, Mar. 2008.

[164] A. Singh, N. Ahuja, and P. Moulin, "Online learning with kernels: Overcoming the growing sum problem," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, Mar. 2012, pp. 1–6.

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

[166] K. Slavakis, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in reproducing kernel Hilbert spaces: The robust beamforming case," *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4744–4764, Dec. 2009.

[167] M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4672–4682, Sep. 2012.

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

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

[170] 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, Dec. 2009.

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

[172] 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, Jan. 2012.

[173] 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, Germany: Springer-Verlag, 2010, pp. 390–404.

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

[175] 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.

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

[177] I. A. Basheer and M. Hajmeer, "Artificial neural networks: Fundamentals, computing, design, and application," *J. Microbiol. Methods*, vol. 43, no. 1, pp. 3–31, Dec. 2000.

[178] 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, Jun. 2004.

[179] 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 Control Eur. Control Conf.*, Dec. 2011, pp. 1796–1802.

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

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

[182] 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, Aug. 2010.

[183] 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, Jul. 1998.

[184] 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.

[185] S. Ozawa, Y. Takeuchi, and S. Abe, "A fast incremental kernel principal component analysis for online feature extraction," in *Trends in Artificial Intelligence*, B.-T. Zhang and M. A. Orgun, Eds. New York, NY, USA: Springer-Verlag, 2010, pp. 487–497.

[186] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognit.*, vol. 40, no. 3, pp. 863–874, Mar. 2007.

[187] 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, Oct. 2003.
[188] 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, Sep. 2005.
[189] 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, Mar. 2013.
[190] Q. Yong-Sheng, W. Pu, F. Shun-Jie, G. Xue-Jin, and J. Jun-Feng, "Enhanced batch process monitoring using Kalman filter and multiway kernel principal component analysis," in *Proc. Chin. 21st Control Decision Conf.*, vol. 5. Jun. 2009, pp. 5289–5294.
[191] T.-J. Chin and D. Suter, "Incremental kernel principal component analysis," *IEEE Trans. Image Process.*, vol. 16, no. 6, pp. 1662–1674, Jun. 2007.
[192] 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, Cybern. B, Cybern.*, vol. 40, no. 4, pp. 1088–1100, Aug. 2010.
[193] M. Davy, F. Desobry, A. Gretton, and C. Doncarli, "An online support vector machine for abnormal events detection," *Signal Process.*, vol. 86, no. 8, pp. 2009–2025, Aug. 2006.
[194] S. Marsland, "Novelty detection in learning systems," *Neural Comput. Surv.*, vol. 3, no. 2, pp. 157–195, 2003.
[195] F. Desobry, M. Davy, and C. Doncarli, "An online kernel change detection algorithm," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2961–2974, Aug. 2005.
[196] Y. Zhang, N. Meratnia, and P. J. M. Havinga, "Adaptive and online one-class support vector machine-based outlier detection techniques for wireless sensor networks," in *Proc. Int. Conf. Adv. Inform. Netw. Appl. Workshops*, May 2009, pp. 990–995.
[197] V. Gomez-Verdejo, J. Arenas-Garcia, M. Lazaro-Gredilla, and A. Navia-Vazquez, "Adaptive one-class support vector machine," *IEEE Trans. Signal Process.*, vol. 59, no. 6, pp. 2975–2981, Jun. 2011.
[198] B. Sofman, B. Neuman, A. Stentz, and J. A. Bagnell, "Anytime online novelty and change detection for mobile robots," *J. Field Robot.*, vol. 28, no. 4, pp. 589–618, 2011.
[199] 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, Dec. 2006.
[200] C. Campbell, "Kernel methods: A survey of current techniques," *Neurocomputing*, vol. 48, nos. 1–4, pp. 63–84, Oct. 2002.
[201] S. Salti and L. D. Stefano, "On-line support vector regression of the transition model for the Kalman filter," *Image Vis. Comput.*, vol. 31, nos. 6–7, pp. 487–501, Jun./Jul. 2013.
[202] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.
[203] W. Wang, C. Men, and W. Lu, "Online prediction model based on support vector machine," *Neurocomputing*, vol. 71, nos. 4–6, pp. 550–558, Jan. 2008.
[204] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, Apr. 1975.
[205] G. Welch and G. Bishop. (1995). *An Introduction to the Kalman Filter* [Online]. Available: http://clubs.ens-cachan.fr/krobot/old/data/positionnement/kalman.pdf
[206] R. S. Liptser and A. N. Shiryaev, *Statistics of Random Processes: II. Applications*. New York, NY, USA: Springer-Verlag, 2001.
[207] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. New York, NY, USA: Springer-Verlag, 1998.
[208] J. Durbin and S. Koopman, *Time Series Analysis by State Space Methods*, 2nd ed. London, U.K.: Oxford Univ. Press, 2012.
[209] C. K. Chui and G. Chen, *Kalman Filtering: With Real-Time Applications*. New York, NY, USA: Springer-Verlag, 2009.
[210] 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, Mar. 2011.
[211] L. Ralaivola and F. D'Alche-Buc, "Time series filtering, smoothing and learning using the kernel Kalman filter," in *Proc. Int. Joint Conf. Neural Netw.*, vols. 1–5. Jul./Aug. 2005, pp. 1449–1454.
[212] F. M. Waltz and J. W. V. Miller, "Gray-scale image processing algorithms using finite-state machine concepts," *J. Electron. Imag.*, vol. 10, no. 1, pp. 297–307, Jan. 2001.
[213] K. Kuusilinna, V. Lahtinen, T. Hamalainen, and J. Saarinen, "Finite state machine encoding for VHDL synthesis," *IEE Proc. Comput. Digit. Techn.*, vol. 148, no. 1, pp. 23–30, 2001.
[214] T. Natschläger and W. Maass, "Spiking neurons and the induction of finite state machines," *Theoretical Comput. Sci.*, vol. 287, no. 1, pp. 251–265, Sep. 2002.
[215] M. Martìnez-Ramòn *et al.*, "Support vector machines for nonlinear kernel ARMA system identification," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1617–1622, Nov. 2006.
[216] L. Shpigelman, H. Lalazar, and E. Vaadia, "Kernel-ARMA for hand tracking and brain-machine interfacing during 3D motor control," in *Proc. Neural Inform. Process. Syst. (NIPS)*, 2008, pp. 1489–1496.
[217] 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. Statist. Rev.*, vol. 53, no. 3, pp. 301–329, Dec. 1985.
[218] C. Saunders, A. Vinokourov, and J. S. Shawe-Taylor, "String kernels, fisher kernels and finite state automata," *Adv. Neural Inform. Process. Syst.*, vol. 15, pp. 633–640, Apr. 2003.
[219] D. Wingate and S. Singh, "Kernel predictive linear Gaussian models for nonlinear stochastic dynamical systems," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 1017–1024.

Yuichi Motai (S'00–M'03–SM'12) received the B.Eng. degree in instrumentation engineering from Keio University, Tokyo, Japan, in 1991, the M.Eng. degree in applied systems science from Kyoto University, Kyoto, Japan, in 1993, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2002.

He is currently an Associate Professor of Electrical and Computer Engineering with Virginia Commonwealth University, Richmond, VA, USA. His current research interests include sensory intelligence, in particular, medical imaging, pattern recognition, computer vision, and sensory-based robotics.