A Kernel Autoassociator Approach to Pattern Classification

Haihong Zhang, Weimin Huang, *Member, IEEE*, Zhiyong Huang, *Member, IEEE*, and Bailing Zhang

***Abstract*—Autoassociators are a special type of neural networks which, by learning to reproduce a given set of patterns, grasp the underlying concept that is useful for pattern classification. In this paper, we present a novel nonlinear model referred to as kernel autoassociators based on kernel methods. While conventional nonlinear autoassociation models emphasize searching for the nonlinear representations of input patterns, a kernel autoassociator takes a kernel feature space as the nonlinear manifold, and places emphasis on the reconstruction of input patterns from the kernel feature space. Two methods are proposed to address the reconstruction problem, using linear and multivariate polynomial functions, respectively. We apply the proposed model to novelty detection with or without novelty examples and study it on the promoter detection and sonar target recognition problems. We also apply the model to $m$class classification problems including wine recognition, glass recognition, handwritten digit recognition, and face recognition. The experimental results show that, compared with conventional autoassociators and other recognition systems, kernel autoassociators can provide better or comparable performance for concept learning and recognition in various domains.**

***Index Terms*—Kernel machine, nonlinear associative memory, pattern recognition.**

I. Introduction

PATTERN classification is an important issue in a variety of scientific and engineering disciplines. As Medin suggests, two elements play key roles in a classification process: concept and category [1]. *Concept* refers to an abstract representation of a category, while *category* refers to the set of entities picked out by the concept. The problem of concept learning is often made difficult by the shear complexity of patterns present in practical tasks. A well-known example is face recognition, in which facial images can be very complicated and highly nonlinear, especially when faces are subject to changes in view angles or lighting conditions [2]. It is also known that linear approaches such as Fisher linear discriminant and principal component analysis (PCA) [3] are not well suited for learning the nonlinear concept.

This paper emphasizes an alternative approach to nonlinear concept learning and pattern classification by using a special type of artificial neural networks called autoassociators. An autoassociator is a brain-like distributed network that learns from the samples $\{\mathbf{x}_i\}$ in a category to reproduce each sample at the output with a mapping [4, p. 66]

$$\hat{\mathbf{x}}_i = F(\mathbf{x}_i). \tag{1}$$

The reproduction may seem pointless, whereas through learning the autoassociation the network may find the commonalities in the samples and, thus, grasp the underlying concept [5, p. 72]. For instance, Kohonen has demonstrated in an early work that an autoassociator can be used to store and retrieve face images [6]. Daunicht has also demonstrated that autoassociators are useful for modeling neuromechanics [7].

Autoassociators are generally used as one-class learning machines. In other words, each network corresponds to a particular category, and during training, it receives only the samples within the category. An important consequence is that the network will learn to accurately reproduce *positive samples* (samples in the corresponding category), producing a reproduction error surface $E_r(\vec{x}) = \|F(\mathbf{x}) - \mathbf{x}\|$ that reflects the distribution of the samples. Thus, autoassociators provide an alternative approach to concept learning. In particular, the higher the reproduction quality for an input pattern, the more likely it belongs to the category for which the autoassociator is constructed. That provides a basis for various autoassociator-based classifiers as below, which depend on reproduction error surfaces to discriminate between classes.

Autoassociators are well suited for addressing a special binary classification issue called novelty detection in which novel or abnormal patterns are expensive or difficult to obtain; thus, only a few or even no novelty examples are available for learning. An extensive survey in the area of novelty detection can be found in [8], and Markou and Singh have especially presented in [9] an excellent review on autoassociator-based approaches. Autoassociator-based approaches rely on the fact that, since an autoassociator only learns to give high-quality reproductions for normal patterns, the reproduction of a novel pattern will yield a large error which can be thresholded to signal novelty. This classification methodology has been applied to various detection problems such as face detection [10], motor failure detection [11], network security [12], and natural language grammar learning [13].

Autoassociators are also useful for $m$class classification with a competitive scheme. The system creates a set of networks for each class, and then a probe pattern is reproduced by each network in testing phase. The respective reproduction error provides the basis for competition among the networks. The particular network with the smallest value of reproduction error is

Manuscript received February 20, 2004; revised August 5, 2004. This paper was recommended by Associate Editor S. Singh.

H. Zhang is with the Institute for Infocomm Research, Singapore 119613, and also with the School of Computing, National University of Singapore, Singapore 117543 (e-mail: hhzhang@i2r.a-star.edu.sg).

W. Huang is with the Institute for Infocomm Research, Singapore 119613 (e-mail: wmhuang@i2r.a-star.edu.sg).

Z. Huang is with the School of Computing, National University of Singapore, Singapore 117543 (e-mail: huangzy@comp.nus.edu.sg).

B. Zhang is with the School of Computer Science and Mathematics, Victoria University, VIC3011 Victoria, Australia (e-mail: bzhang@csm.vu.edu.au).

Digital Object Identifier 10.1109/TSMCB.2005.843980

declared winner of the competition. This classification methodology has been successfully demonstrated in various applications, such as handwritten character recognition ([14], [15]) and face recognition [16].

In the field of autoassociative networks, linear autoassociators, such as correlation-associative memories, have been extensively studied. Kohonen has pointed out that using a linear autoassociator to store and recall patterns is equivalent to computing a PCA of the cross-product matrix of the patterns and reconstructing them as a weighted sum of eigenvectors [6]. It is the same case with multilayer linear autoassociative networks, according to Baldi and Hornik's theoretical study in [17]. As the consequence, linear autoassociators have serious limitations in exploring high-order dependency among data. Naturally, nonlinear autoassociators are favorable.

Existing nonlinear autoassociator models are generally based on a special type of backpropagation networks [18] called autoassociative multilayer perceptrons. Such a network includes nonlinear hidden units between the input and the output units. The input-to-hidden layer connections perform the encoding with the hidden units building for input patterns internal representations, while the hidden-to-output layer connections do the decoding. That is why this type of networks are often referred to in the literature as *autoencoders*.

Autoencoders with one hidden layer have demonstrated their capability for learning low-dimensional nonlinear features [19]. However, it has been claimed that in some image processing cases [20]–[22] they are comparable to linear PCA. An existing method to overcome this problem is by having multiple hidden layers [23]. The consequent architecture called nonlinear principal component analysis (NLPCA) often consists of three hidden layers, and its capability has been demonstrated in various domains [24], [25].

According to Moghaddam's study on face recognition [26], however, NLPCA could be outperformed by other methods including independent component analysis (ICA) [27] and even the linear method of PCA. The author suggests that the NLPCA's poor performance can be attributed to the general difficulty of computing nonlinear manifolds and the complexity of cost functions riddled with local minima. Furthermore, Malthouse has also pointed out certain limitations of autoencoders [28]. For example, when the network's solution is used to extrapolate, or when there are training values close to ambiguity points on principal curves, the encoding results by the network will be incorrect.

The goal of this paper is to propose an alternative, more efficient approach to modeling nonlinear autoassociations. To this end, we emphasize a special type of nonlinear methods called kernel methods which have been established as a context for solving a variety of nonlinear problems [29]. In brief, kernel methods operate a special class of functions called *reproducing kernels* [30] $k(\mathbf{x}_1, \mathbf{x}_2)$ in the input pattern space (R^N), which amounts to casting the data into a high-dimensional kernel feature space $(\mathcal{H})$ by a possibly implicit map Φ, and taking the dot product there

$$k(\mathbf{x}_1, \mathbf{x}_2) = \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle . \tag{2}$$

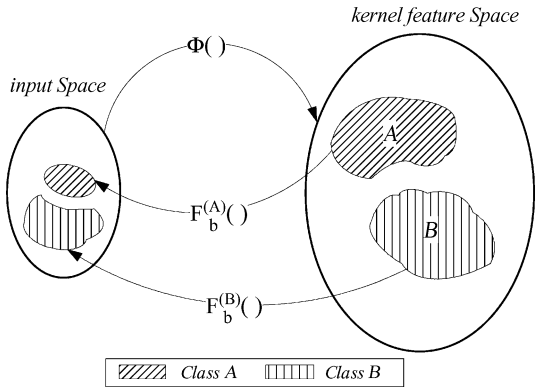


Fig. 1. Illustration of kernel autoassociation. Input patterns are projected through $\Phi(\cdot)$ to a kernel feature space $\mathcal{H}$. Then, each kernel autoassociator learns to reconstruct [e.g., via $F_b^{(m)}()$] a particular class of patterns from their kernel features.

By virtue of this property, many linear algorithms have been extended to the kernel feature space, with the outcomes being nonlinear in the input space. Well-known examples include support vector machines (SVMs) [31], kernel Fisher discriminant (KFD) [32], and kernel principal component analysis (KPCA) [33].

The paper provides a new perspective on kernel methods by using them to address the nonlinear autoassociation issue. In particular, we propose a kernel autoassociator model which associates the input and the output by mapping through the kernel feature space. Fig. 1 gives an illustration, where the autoassociation is accomplished through two phases. First, an input pattern $\mathbf{x}$ is cast into a kernel feature space by $\Phi(\mathbf{x}) : \mathbf{x} \rightarrow \Phi(\mathbf{x})$, and it is then, subsequently, mapped backward to the input space via $F_b : \Phi(\mathbf{x}) \rightarrow \hat{\mathbf{x}}$, where F_b is class dependent ($F_b^{(A)}$ for class A and $F_b^{(B)}$ for class B). The subscript b denotes that the function is for reverse mapping. Unless otherwise specified, we will omit the class label m in describing reverse mapping functions in general.

Hence, the kernel autoassociation involves two important issues on the kernel mapping (Φ) and the reverse mapping (F_b), respectively. As mentioned earlier, autoencoders involve two comparable issues regarding the setup of nonlinear representations and the pattern reconstruction from the representations. Unlike autoencoders, kernel autoassociators put emphasis not on building nonlinear representations because choosing a reproducing kernel will automatically establish an associated kernel feature space. Instead, since the setup of reproducing kernel and kernel feature space has already been elegantly studied in the literature, we would like to pay particular attention to the reverse mapping.

Recall the two phases of kernel autoassociation as depicted in Fig. 1. It can be seen that the first phase of kernel autoassociation through the kernel mapping $\Phi(\mathbf{x})$ is just a straightforward process without learning the particular concept from samples. Thus, the learning task rests on the modeling of the reverse mapping F_b, which should reflect the characteristics of the category.

It is worthwhile to mention that in a relevant study [34], Schökopf has proposed an algorithm for the reverse mapping F_b. However, it is not suited for modeling autoassociators

for to two reasons. First, it does not involve the learning of class-specific reverse mapping; second, it is only applicable to Gaussian kernels.

This paper addresses the above problem by proposing two reverse mapping methods. The first method uses linear functions in the kernel feature space, while the second one uses multivariate polynomial functions. The two methods are both applicable to arbitrary kernel types, and, more importantly, they allow learning particular concepts of classes.

We apply kernel autoassociators to novelty detection and $m$class classification problems. Two detection schemes are developed for novelty detection with or without novelty examples, and they are tested on promoter detection and sonar target recognition. This paper proceeds by applying kernel autoassociators to $m$class classification problems in the domains of wine recognition, glass recognition, handwritten digit recognition, and face recognition. The experimental results show that kernel autoassociators provide better or comparable performance for concept learning and recognition in various domains than conventional autoassociators and other existing recognition systems.

In summary, the paper presents an alternative approach to nonlinear autoassociation. By making use of a kernel feature space, the approach resorts to relatively simpler functions (linear or polynomial) for learning, in contrast to conventional autoassociation machines that use a complex class of functions—such as a collection of sigmoid functions in multilayer perceptrons. The approach appears to be more efficient and easier to implement, and it is promising for a variety of novelty detection and $m$class classification applications.

The rest of the paper is organized as follows. Section II introduces the kernel autoassociator model and elaborates two methods for reverse mapping. The proposed model is evaluated with simulations in Section III. Section IV applies kernel autoassociators to novelty detection, and Section V applies the model to $m$class recognition. The discussions and the conclusion are given in Section VI.

II. Kernel Autoassociators

Kernel autoassociators produce pattern reproduction through *reproducing kernel Hilbert spaces* (RKHS) [35], which provide a unified context for solving a variety of statistical modeling and function estimation problems. Here, we would review some basic concepts such as positive-definite functions as below.

Let $\mathcal{T}$ be an index set, e.g., Euclidean space E^N. A function $k(\mathbf{x},\mathbf{t}), (\mathbf{x},\mathbf{t}) \in \mathcal{T}\otimes\mathcal{T}$ is said to be a positive-definite function on $\mathcal{T}\otimes\mathcal{T}$ if, for every number n, every set $\mathbf{t}_1,\ldots,\mathbf{t}_n$, and every $\mathbf{a}=[a_1,\ldots,a_n]^T \in \mathrm{R}^N$, we have

$$\sum_{i,j=1}^{n} a_i a_j k(\mathbf{t}_i,\mathbf{t}_j) \geq 0. \tag{3}$$

Let $k(\cdot,\cdot)$ be a *positive-definite function* on $\mathcal{T}\otimes\mathcal{T}$, and $k_{\mathbf{t}}(\cdot)=k(\mathbf{t},\cdot)$ a functional with respect to $\mathbf{t}$. When $\mathbf{t}$ is fixed, $k_{\mathbf{t}}$ will be a determined function. Corresponding to $k(\cdot,\cdot)$, there exists a unique collection of real valued functions on T, called a RKHS $\mathcal{H}_k$ [36]

$$k_{\mathbf{t}} \in \mathcal{H}_k, \quad \text{for each } \mathbf{t}\in\mathcal{T} \tag{4}$$

$$\sum_{l=1}^{L} a_l k_{\mathbf{t}_l} \in \mathcal{H}_k, \quad \text{for any finite } L \text{ and } \{a_l\}. \tag{5}$$

The inner product in $\mathcal{H}$ is defined by

$$\langle k_{\mathbf{x}}, k_{\mathbf{t}}\rangle = k(\mathbf{x},\mathbf{t}). \tag{6}$$

Let an arbitrary function in $\mathcal{H}_k$ be expressed in form of (5). Then we have

$$\begin{aligned}\langle k_{\mathbf{t}}, f\rangle &= \left\langle k_{\mathbf{t}}, \sum_{l=1}^{L} a_l k_{\mathbf{t}_l}\right\rangle = \sum_{l=1}^{L} a_l \langle k_{\mathbf{t}}, k_{\mathbf{t}_l}\rangle \\ &= \sum_{l=1}^{L} a_l k(\mathbf{t}_l,\mathbf{t}) = f(\mathbf{t})\end{aligned} \tag{7}$$

and $k(\cdot,\cdot)$ is called the *reproducing kernel* for $\mathcal{H}_k$.

Hence, by choosing a reproducing kernel function k, one can cast a pattern $\mathbf{x}$ into a RKHS $\mathcal{H}_k : k(\mathbf{x},\cdot)$, and $\mathcal{H}_k$ is called a *kernel feature space*, with respect to k.

The principle of kernel autoassociators, as mentioned earlier, is to perform autoassociation mapping via the kernel feature space, i.e., reconstructing patterns from their counterparts in $\mathcal{H}_k$. Hereafter, a *reconstruction* (the reverse mapping) function is denoted by

$$\hat{\mathbf{x}} = F_b^{(m)}\left(\Phi(\mathbf{x})\right), \quad \text{for } \mathbf{x} \in \text{ class } m \tag{8}$$

where $\Phi(\mathbf{x})=k(\mathbf{x},\cdot)$ represents the feature in functional form in $\mathcal{H}_k$. As we mentioned in the Introduction, the subscript b denotes that the function is for reverse mapping. Note that unless otherwise specified, we omit the class label m in describing a reverse mapping function hereafter.

A positive-definite function $k(.,.)$ is associated with a unique RKHS and, thus, can be used as a kernel function for kernel methods [29]. In fact, the kernel autoassociator model does not confine the selection of kernel function k, while in the present paper we tentatively examine two of the most popular kernel functions, namely, Gaussian function and the polynomial function

$$\text{Gaussian kernel:} \quad k(\mathbf{x}_1,\mathbf{x}_2)=\exp\left(-\frac{||\mathbf{x}_1-\mathbf{x}_2||^2}{2\sigma^2}\right) \tag{9}$$

$$\text{Polynomial kernel:} \quad k(\mathbf{x}_1,\mathbf{x}_2)=\left(\mathbf{x}_1^T\mathbf{x}_2+1\right)^p \tag{10}$$

where p denotes the power of the polynomial, σ the bandwidth of the Gaussian kernel.

A kernel autoassociator learns the concept of a category in a very high-dimensional feature space $\mathcal{H}_k$, in contrast to conventional methods that learn in the input space or a low-dimensional feature space. This methodology may raise doubts about whether it is good to resort to a higher dimensional feature space for learning, since the *curse of dimensionality* principle asserts that the difficulty of learning may increase drastically with the dimensionality.

The doubts can be resolved by the statistical learning theory which states: not the dimensionality, but the complexity of the function class matters [37], and learning can be simpler if one uses a class of functions of low complexity. Another underlying justification is in Cover's theorem [38] on the separability of patterns. It states that a complex classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space, provided two conditions are satisfied. First, the transformation is nonlinear. Second, the dimensionality of the feature space is high enough. A good example is a support vector machine that solves classification problems with linear decision rules in a high-dimensional feature space instead of using nonlinear, complex decision rules in the input space.

It can be seen that our kernel feature mapping $\mathbf{x} \rightarrow \Phi(\mathbf{x})$ satisfies the above two conditions. As already shown in Fig. 1, the kernel autoassociator model casts input patterns into a kernel feature space, and learns the class-specific dependencies between the feature space and the input space. The aforementioned theory suggests that we may use a simple class of functions such as linear functions or polynomials in the kernel feature space for concept learning.

A. Linear Functions for F_b

Let F_b be a linear mapping function from $\mathcal{H}_k$ to the input space E^N. The complete autoassociator is still nonlinear even though F_b is linear because of the intrinsic nonlinearity of the feature mapping $\Phi(\cdot)$. When the patterns to be reproduced are multidimensional, F_b will be composed of a set of functions $\{f_{b_n}\}$, each corresponding to an element of the output space: $F_b = [f_{b_1}, \ldots f_{b_N}]^T$. Consider an element function f_{b_n}. We will omit the element-label n, hereafter, unless otherwise specified. The function in linear form is given by

$$\hat{x} = f_b\left(\Phi(\mathbf{x})\right) = \langle \beta_\phi, \Phi(\mathbf{x}) \rangle . \tag{11}$$

Here, $\hat{x}$ denotes an element of the output vector $\hat{\mathbf{x}}$, and β_ϕ is a vector in the feature space. Suppose the vector β_ϕ can be spanned by the images of M training samples [34]

$$\beta_\phi = \sum_{i=1}^{M} b_i \Phi(\mathbf{x}_i) \tag{12}$$

then we can rewrite the linear function f_b as

$$\hat{x} = \left\langle \sum_{i=1}^{M} b_i \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \right\rangle = \sum_{i=1}^{M} b_i k(\mathbf{x}_i, \mathbf{x}) = \mathbf{b}^T \mathbf{k} \tag{13}$$

where $\mathbf{b} = [b_1, \ldots, b_M]^T$ is the vector of expansion coefficients, and $\mathbf{k} = [k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_M, \mathbf{x})]^T$ represents the vector of kernel products. Then, the complete output vector $\hat{\mathbf{x}}$ is given by

$$\hat{\mathbf{x}} = B\mathbf{k} \tag{14}$$

where $B = [\mathbf{b}_1, \ldots, \mathbf{b}_N]$ denotes the collection of linear projections for each output element. Interestingly, it is the same as the expression of a kernel associative memory (KAM) [16], which, however, is derived in a different way as an extension of correlation associative memories. This finding suggests that KAMs can be considered as a special form of kernel autoassociators.

Given a set of samples, say, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M)$ for training, one can first compute the kernel product vectors $\{\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_M\}$. The desired output of the network can then be expressed by

$$(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M) = B(\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_M) \text{ or } X = BK. \tag{15}$$

Here, X is the matrix with each column an example pattern, and K represents the matrix with each column a corresponding kernel product vector.

One way to learn the projection matrix B is by finding a matrix that minimizes the empirical square error $\sum_i \|\mathbf{x}_i - B\mathbf{k}_i\|^2$. A method to the minimization is given by

$$B = XK^+ \tag{16}$$

where K^+ is the pseudo-inverse of the kernel product matrix $K : K^+ = (K^T K)^{-1} K^T$.

B. Polynomials for F_b

In this subsection, we will develop a polynomial model for the reverse mapping functions. It starts with a study on polynomials in the kernel feature space. Next, we propose an approximation method based on kernel PCA and study a regularization method for the polynomial functions.

Let F_b consist of more complex functions, namely, second-order multivariate polynomials

$$\hat{x} = f_b\left(\Phi(\mathbf{x})\right) = \Phi^T(\mathbf{x}) W_\phi \Phi(\mathbf{x}) + \beta_\phi^T \Phi(\mathbf{x}) + c_\phi \tag{17}$$

where W_ϕ, β_π and c_ϕ are the polynomial coefficients. This formulation takes the feature $\Phi(\mathbf{x})$ as a column vector (i.e., $\beta_\phi^T \Phi(\mathbf{x}) = \langle \beta_\phi, \Phi(\mathbf{x}) \rangle$), and allows exploring up to second-order nonlinearity in the reverse mapping f_b.

The direct calculation of (17) is not feasible because generally the kernel feature vector Φ is given in an implicit or extremely high-dimensional form. Thus, we need to resort to approximation techniques to solve the problem.

Suppose that $\Phi(\mathbf{x})$ can be approximated by a low-dimensional representation

$$\Phi(\mathbf{x}) = \sum_{i=1}^{N_a} \alpha_i \mathbf{v}_i = (\mathbf{v}_1, \cdots, \mathbf{v}_{N_a})(\alpha_1, \cdots, \alpha_{N_a})^T = V\alpha. \tag{18}$$

Here, V is a matrix with each column a basis vector $\{\mathbf{v}_i\}$ of the N_a-dimensional subspace, and α denotes the projections of Φ onto V. We then have a new expression of the second-order polynomial (17), given by

$$\hat{x} = \alpha^T W \alpha + \beta^T \alpha + c \tag{19}$$

where $c = c_\phi$, $W = V^T W_\phi V$, and $\beta = V\beta_\phi$. Clearly, it turns out to be a polynomial function with respect to the coefficient vector α.

1) Polynomials on Kernel Principal Components: A kernel subspace for the above representation can be set up by the KPCA

technique [33], which essentially performs a linear PCA in the kernel feature space

$$\Phi(\mathbf{x}) = \bar{\Phi} + \left(\Phi(\mathbf{x}) - \bar{\Phi}\right) = \bar{\Phi} + \sum_{i}^{N_a} \alpha_i \mathbf{v}_i \tag{20}$$

where $\bar{\Phi}$ is the average of all $\Phi(\mathbf{x}_i)$: $\bar{\Phi} = (1/M)\sum_{i=1}^{M}\Phi(\mathbf{x}_i)$ over M training samples. The set $\{\mathbf{v}_i\}$ consists of the orthogonal bases of the principal subspace and $\mathbf{v}_i \perp \mathbf{v}_j$. That allows one to calculate the expansion coefficients α_i using direct projection

$$\alpha_i = \left\langle \left(\Phi(\mathbf{x}) - \bar{\Phi}\right), \mathbf{v}_i \right\rangle. \tag{21}$$

Now, consider the calculation of $\mathbf{v}_i$ and α_i. Given a collection of training feature vectors $\Phi(\mathbf{x}_j)$, the corresponding principal components must lie in the subspace spanned by $\Phi(\mathbf{x}_j)$

$$\mathbf{v}_i = \sum_{j=1}^{M} \gamma_{ij}\left(\Phi(\mathbf{x}_j) - \bar{\Phi}\right) \tag{22}$$

where the coefficient vector $\boldsymbol{\gamma}_i := (\gamma_{i1}, \ldots, \gamma_{iM})^T$ is a vector of expansion coefficients. The coefficients are determined with an eigenvector problem [33]

$$M\lambda\boldsymbol{\gamma} = \tilde{K}\boldsymbol{\gamma} \tag{23}$$

for nonzero eigenvalues λ. Here, $\tilde{K}_{ij} := \langle(\Phi(\mathbf{x}_i) - \bar{\Phi}), (\Phi(\mathbf{x}_j) - \bar{\Phi})\rangle$. It follows that

$$\begin{aligned}\alpha_i &= \sum_{j=1}^{M} \gamma_{ij}\left\langle \left(\Phi(\mathbf{x}) - \bar{\Phi}\right), \left(\Phi(\mathbf{x}_j) - \bar{\Phi}\right)\right\rangle \\ &= \sum_{j=1}^{M}\gamma_{ij}\left[k(\mathbf{x},\mathbf{x}_j) - \frac{1}{M}\sum_{n=1}^{M}\left(k(\mathbf{x},\mathbf{x}_n) - k(\mathbf{x}_j,\mathbf{x}_n)\right)\right. \\ &\qquad \left. + \frac{1}{M^2}\sum_{n1}^{M}\sum_{n_2}^{M}k(\mathbf{x}_{n1},\mathbf{x}_{n2})\right].\end{aligned} \tag{24}$$

It can be seen that, due to the presence of $\bar{\Phi}$, the expression of α in (20) is not compatible with the original polynomial formulations [(18) and (19)]. Thus, we need to rewrite the polynomial function by

$$\begin{aligned}\hat{x} &= f_b\left(\Phi(\mathbf{x})\right) \\ &= \left(\Phi^T(\mathbf{x}) - \bar{\Phi}\right) W_\phi \left(\Phi^T(\mathbf{x}) - \bar{\Phi}\right) \\ &\quad + \beta_\phi^T\left(\Phi^T(\mathbf{x}) - \bar{\Phi}\right) + c_\phi.\end{aligned} \tag{25}$$

This formulation will lead to a polynomial on the subspace feature vector α, similar to (19), but the vector α here is given by (21) instead of (18).

Obviously, training an autoassociator amounts to estimating the parameters W, β, and c from a given set of samples. Although the function (19) is nonlinear in the variable α, it can be favorably expressed as a linear function with respect to $\{W, \beta, c\}$

$$\hat{x} = \sum_{i=1}^{N_a}\sum_{j=1}^{N_a}\alpha_i\alpha_j w_{ij} + \sum_{i=1}^{N_a}\alpha_i\beta_i + c. \tag{26}$$

TABLE I
POLYNOMIAL IN KPCA SUBSPACE VERSUS THAT ON KERNEL PRODUCTS. I DENOTES THE IDENTITY MATRIX

$\alpha^T W\alpha + \beta^T\alpha + c$	$\mathbf{k}W_k\mathbf{k} + \beta_k\mathbf{k} + c_k$
W	$W_k = (\Gamma^T(I-E))^T W \Gamma^T(I-E)$
β	$\beta_k = 2(\Gamma^T(I-E))^T W\Gamma^T\mathbf{u} + (\Gamma^T(I-E)^T\beta$
c	$c_k = (\Gamma^T\mathbf{u})^T W\Gamma^T\mathbf{u} + \Gamma^T\beta + c$

Hence, the learning problem can be conveniently solved with linear algebra.

In the following, we show that there exists a polynomial function on the kernel product $\mathbf{k}$ equivalent to that on α. In other words, one can avoid computing KPCA in running autoassociators, allowing fast implementation.

Let us first study the calculation of α in (24). A few terms there depend only on the training examples $\{\mathbf{x}_i\}$ while being irrelevant to the input pattern $\mathbf{x}$. They can be rewritten as

$$u_i = \sum_{n=1}^{M} k(\mathbf{x}_i,\mathbf{x}_n) + \frac{1}{M^2}\sum_{n1=1}^{M}\sum_{n2=1}^{M}k(\mathbf{x}_{n1},\mathbf{x}_{n2}). \tag{27}$$

It follows that

$$\alpha_i = \boldsymbol{\gamma}_i^T[\mathbf{k}+\mathbf{u}] - \frac{1}{M}\left(\gamma_i^T 1_M\right)\left(\mathbf{k}^T 1_M\right) \tag{28}$$

where 1_M is a M-unit long vector of all 1s. Denoting $\Gamma = (\gamma_1, \gamma_2, \ldots)$, the whole vector α reads

$$\alpha = \Gamma^T[\mathbf{k}+\mathbf{u}] - \frac{1}{M}\left(\Gamma^T 1_M 1_M^T\right)\mathbf{k} = \Gamma^T[\mathbf{k}+\mathbf{u}] - (\Gamma^T E)\mathbf{k} \tag{29}$$

where E is an $M \times M$ matrix consisting of all 1s.

Substituting the expression for α in (19), the equation becomes

$$\hat{x} = f_b\left(\Phi(\mathbf{x})\right) = \mathbf{k}^T W_k\mathbf{k} + \beta_k^T\mathbf{k} + c_k \tag{30}$$

which is a multivariate polynomial on the kernel product vector $\mathbf{k}$. Details of W_k, β_k, and c_k are given in Table I, which reveals the relationship between a polynomial function on α with its equivalence on the kernel product vector $\mathbf{k}$. Thus, runing autoassociators will use precomputed W_k, β_k, and c_k without computing kernel principal analysis.

2) Regularization of Kernel Polynomials: Learning machines may have the so-called "over-fitting" problem in which the machines specialize well to training patterns but generalize poorly to new patterns. To enhance the generalization performance, a common method is regularization which aims to stabilize the solution by means of some auxiliary nonnegative functional that embeds prior information about the solution [39]. The widely used prior information involves an assumption that the input–output mapping function is smooth, in the sense that similar inputs correspond to similar outputs. Hence, an objective function for training should take into account both empirical reconstruction error and smoothness of the networks

$$G(f_b) = \sum_{i=1}^{M}\left[x_i - f_b(\mathbf{x}_i)\right]^2 + \lambda R_r(f_b). \tag{31}$$

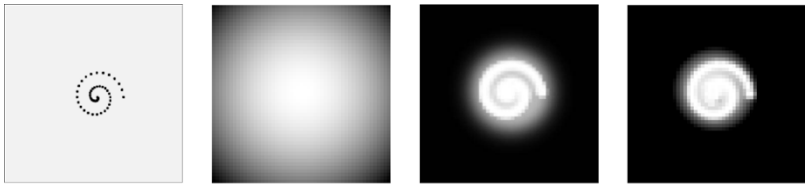


Fig. 2. Concept learning on spiral pattern. Results are shown as reconstruction error surfaces. KAA-1 or KAA-2 represents kernel autoassociator with linear or polynomial reconstruction functions.

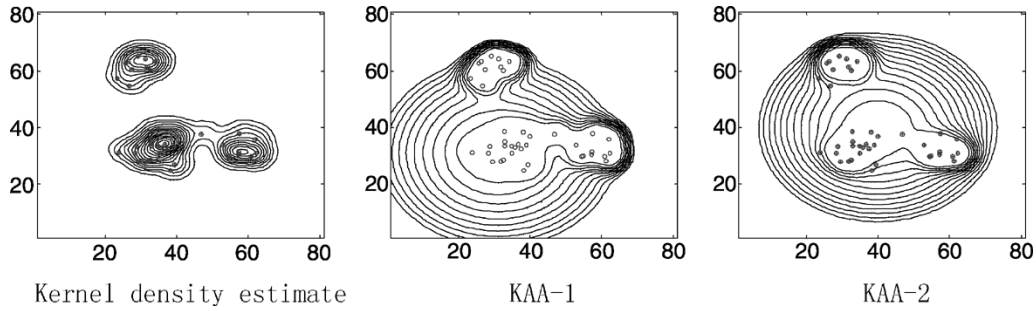


Fig. 3. Results of concept learning on multimodal pattern. The results are shown as probability density surface (by kernel density estimate) or reconstruction error surface (by kernel autoassociators).

Here, the coefficient λ determines the tradeoff between the empirical error and the smoothness measure R_r. We define the smoothness of a polynomial function f_b by

$$R_r(f_b) = \int \|\mathrm{D}f_b\|^2 d\alpha \tag{32}$$

where α is the kernel features by KPCA [see (24)], and D is a first-order linear differential operator. There is

$$\begin{aligned}\|\mathrm{D}f_b\|^2 &= \left\| \frac{\partial f_b(\alpha)}{\partial \alpha} \right\|^2 \\ &= (\beta + 2W\alpha)^T(\beta + 2W\alpha) \\ &= \beta^T\beta + 4\beta^T W\alpha + 4\alpha^T W^T W\alpha. \end{aligned} \tag{33}$$

For simplicity, all the feature vectors α in the regularization are supposed to be normalized beforehand to fulfill $0 \le \alpha \le 1$. It leads to an efficient way to compute the integral $R_r(f_b)$ by

$$\begin{aligned} R_r(f_b) = \beta^T\beta + 2\sum_{i=1}^{N_a} \beta^T \mathbf{w}^{(i)} \\ + \sum_{i=1}^{N_a}\sum_{j=1}^{N_a} \mathbf{w}^{(i)T}\mathbf{w}^{(j)} + \frac{1}{3}\sum_{i=1}^{N_a} \mathbf{w}^{(i)T}\mathbf{w}^{(i)} \end{aligned} \tag{34}$$

where $\mathbf{w}^{(i)}$ is the ith column vector of the matrix W.

Training a kernel autoassociator means minimizing the objective function (31). Because the function is continuous and differentiable, the minimization can be obtained at $\partial G(f_b)/\partial c = 0$, $\partial G(f_b)/\partial \beta = 0$, and $\partial G(f_b)/\partial w_{jk} = 0$. To solve the minimization problem, we tentatively use Matlab nonliear optimization toolbox in the implementation.

III. Simulations

We generated an artificial dataset to test the capability of the proposed models for nonlinear concept learning. The dataset consists of a few samples on a two-dimensional (2-D) spiral, as plotted in the leftmost of Fig. 2. We constructed a kernel autoassociator with linear F_b and another one with polynomial F_b (hereafter, referred to as KAA-1 and KAA-2, respectively), both with Gaussian kernels. Besides, an autoencoder with sigmoid transfer function was also tested for comparison. The reproduction error surfaces are displayed as images in Fig. 2, where the error value is denoted by gray level (i.e., bright means small). Note the results have been thresholded to be better shown in forms of images. For the autoencoder, we tested various numbers (4 to 40) of hidden neurons and observed similar results. The kernel bandwidth for kernel autoassociators was set to be the standard deviation of samples.

The experimental results show that, in spite of the complex structure of spiral patterns, the kernel autoassociators were able to produce reconstruction error surfaces that correctly reflected the data structure. But the autoencoder produced a unimodal-Gaussian-alike error surface (often seen with linear models).

We also conducted a comparison between kernel density estimate [40] (a nonparametric technique for density estimation) and kernel autoassociators, by using a set of examples generated from a three-mode 2-D random distribution. The samples and the experimental results are plotted in Fig. 3.

The results show that every method successfully captured the three modes by the estimated density surface or reconstruction error surface. In particular, with the same kernel bandwidth (σ), kernel autoassociators appear to produce smoother surfaces than kernel density estimate. Thus, they tend to have better generalization capability while maintaining good specialization capability.

IV. Applications to Novelty Detection

As mentioned earlier, novelty detection is the identification of novel patterns of which the learning system is given few samples in the training stage. This problem happens when novel or

abnormal examples are expensive or difficult to obtain. Here, let us consider two novelty detection problems, i.e., promoter detection and sonar target recognition.

The ***promoter*** problem takes as input segments of DNA, some subset of which represent promoters. A promoter is a sequence that signals to the chemical processes acting on the DNA where a gene begins. The goal of the task is to train a classifier that is able to detect promoters—the novel patterns. The ***sonar target*** recognition problem takes as input the signals returned by a sonar system in the cases where mines and rocks were used as targets. We choose mine patterns as novelty.

We acquired both the promoter database and the sonar target database from *UCI Machine Learning Repository*. The promoter database consists of 106 samples, 53 for promoters, while the others for nonpromoters. Each sample is a 57-unit-long string composed of four chars $\{a, c, g, t\}$, which we convert to $\{1, 2, 3, 4\}$ in the experiment. The sonar target database comprises 111 positive and 97 negative patterns (60-unit long).

The autoassociator detection system relies on the fact that an autoassociator is designed to learn normal patterns, thus, the network would tend to produce relatively larger reproduction errors for novel patterns. The errors can be thresholded to signal novelty. Fig. 4 plots the detection scheme.

The specific threshold (ξ) is crucial for the system performance, and should be chosen carefully. In respect to the threshold setting, there are two different cases in novelty detection. In one case, a small number of novel patterns are available for training the detection system; in the other case, one can obtain merely normal patterns for the training. For the two cases, we propose and study two different approaches.

A. Novelty Detection With Novel Examples

With samples from both novel and normal patterns, novelty detection can be viewed as a usual binary classification problem. The goal is to find a rule that best separates the positive and negative classes.

Given an unknown object $\mathbf{x}$, the system will produce a reconstruction error $e(\mathbf{x})$ and the probability of $\mathbf{x}$ to be identified as novelty is given by $P(e(\mathbf{x}) > \xi)$. Let the novel class be ω_1 and the normal class be ω_0. The probability of misclassification is

$$\begin{aligned} P_e &= P\left(e(\mathbf{x}) < \xi, \omega_1\right) + P\left(e(\mathbf{x}) > \xi, \omega_0\right) \\ &= \int \left[P\left(e(\mathbf{x}) < \xi\right) p(\mathbf{x}, \omega_1) \right. \\ &\qquad \left. + P\left(e(\mathbf{x}) > \xi\right) p(\mathbf{x}, \omega_0)\right] d\mathbf{x}. \end{aligned} \tag{35}$$

Here, $P(e(\mathbf{x}) < \xi, \omega_1)(P(e(\mathbf{x}) > \xi, \omega_0))$ is the probability of a novel (normal) pattern classified as normal (novel). The empirical value of the above error is given by

$$P_e = \frac{1}{N_i} \sum_{\mathbf{x}_i \in \omega_1} P\left(e(\mathbf{x}_i) < \xi\right) + \frac{1}{N_j} \sum_{\mathbf{x}_j \in \omega_0} P\left(e(\mathbf{x}_j) > \xi\right) \tag{36}$$

where $\mathbf{x}_i(\mathbf{x}_j)$ is a random sample generated from the distribution $p(\mathbf{x}, \omega_1)(p(\mathbf{x}, \omega_0))$, and N_i or N_j denotes the number of samples of novel or normal samples.

For the given samples and an autoassociator, $e(\mathbf{x}_i)$ is fixed and $P(e(\mathbf{x}_i) < \xi)$ takes binary value (0 or 1) that depends on the threshold ξ. Thus, setting the threshold becomes equivalent to seeking a ξ that minimizes P_e in (36). Since ξ is a one-dimensional (1-D) variable, it can be easily determined with a simple searching program. It needs to be mentioned that the novel patterns only serve for setting the threshold, not for training the autoassociator.

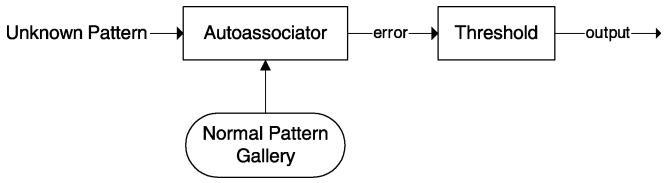


Fig. 4. Novelty detection scheme. An autoassociator learns normal examples; when an unknown pattern is presented, the reconstruction error by the autoassociator will be compared with a threshold to signal whether it is a novel pattern (with larger error) or a normal pattern (with smaller error).

We set up an autoassociator system with the above threshold setting method, and tested it over the aforementioned two detection problems. Parzen-window novelty detectors (ParzenND) [12]—a kernel density estimation technique for novelty detection, and autoencoders, are compared using the same threshold setting method.

The experiment was set up as follows. Each method was evaluated using five-fold cross validation (see [4, p. 213]): We randomly partitioned the promoter/Sonar data set into five subsets of equal size, and in each evaluation trial we selected one subset for testing while the rest was used for training. In each trial of the training test, the negative training set was used only to determine the empirical threshold [see (36)], while the positive training set was used to learn the empirical threshold and the autoassociators. Besides, we averaged the classification error rates over ten independent cross validation to enhance the accuracy of evaluation. All the compared methods were fine tuned.

Table II compares the false detection rates (FDR) (see the table for the description of the symbols). The superscript P denotes polynomial kernel function, while G denotes Gaussian kernel function. Numbers after each "$\pm$" are standard deviations of the detection accuracy. It indicates considerable variance among error rates in different trial. This may be due to the cross-validation evaluation method that could produce largely different complexity in the resulting training/test sets in different folds and trials, especially for the relatively small while complex data in promoter and sonar. Nevertheless, it can be seen that in both domain studies, KAA-2^P, KAA-2^G, and KAA-1^G tend to produce relatively lower error rates.

In addition, we examined the systems' sensitivities in the two domains to small numbers of novelty examples. In this setting, each evaluation trial used the same set of normal patterns as above, but selected randomly only a given number of novelty samples for training. By this, the setting can serve a simulation for studing class imbalance problems, since in each trial we have a positive and a negative training set with largely different size. For example, in the promoter problem, each positive training set has 42 samples or so, while the negative training set could have as few as five samples.

The imbalanced setting may pose a serious problem to typical discriminant machines such as MLP classifiers and SVMs. Our experimental results in Fig. 5 indicate that neither KAA-1

TABLE II
NOVELTY DETECTION ERROR RATE WITH NOVEL EXAMPLES

	Novelty Detection Approach					
	KAA-1^G	KAA-2^G	KAA-1^P	KAA-2^P	Parzen-window	Autoencoder
Promoter	20.4 ± 7.4	18.1 ± 9.9	26.1 ± 9.1	19.4 ± 8.9	22.7 ± 8.4	30.3 ± 13.2
Sonar	27.0 ± 5.5	26.2 ± 7.4	33.9 ± 6.1	27.5 ± 6.7	28.5 ± 6.5	32.6 ± 5.3

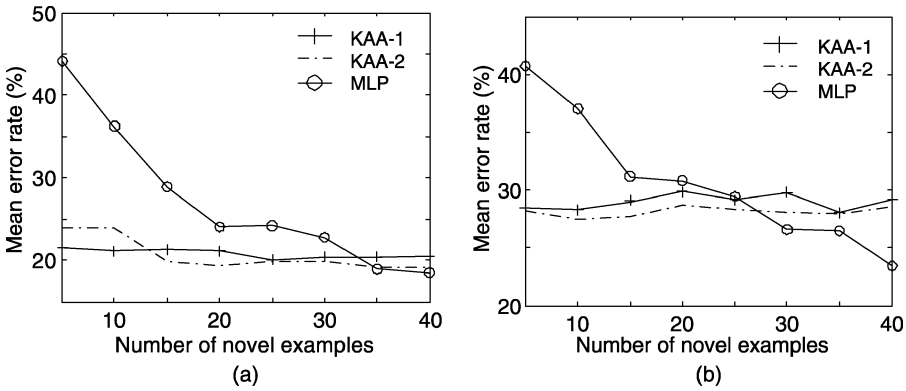


Fig. 5. Recognition error rates over the number of novel examples in the two novelty detection problems. The number of novel examples used for training ranges from 5 to 40. (a) Promoter detection. (b) Sonar target detection.

nor KAA-2 is sensitive to the imbalanced setting, but the performance of MLP and SVM classifiers deteriorated along with the decreasing number of novelty examples, i.e., more imbalanced training data.

B. Novelty Detection Without Novel Examples

In the case without novel samples for training, the above method for determining the novelty detection threshold ξ is not applicable. Instead, we adopt a method from [12] that is designed to achieve a given FDR

$$\upsilon = P\left(e(\mathbf{x}) > \xi | \mathbf{x} \in \omega_0\right). \tag{37}$$

Similar to (36), the equation can also be expressed in terms of samples, and υ will become a function with respect only to ξ. Therefore, for a given FDR, ξ can be easily determined with a 1-D search procedure.

The same threshold setting method is used for comparing kernel autoassociators with autoencoders, Parzen-window detectors ([12]), SOM-ND [41], and one-class SVM [42]. The Parzen-window detector is a nonparametric density estimation technique for novelty detection, while SOM-ND is a novelty detection technique based on self-organizing maps.

We conducted experiments with five-fold cross-validation, similar to the previous experiment of novelty detection with novel examples. The difference is that the training program in this experiment did not use novel examples, and only the normal sample set was used for learning the threshold according to a given FDR (7). We calculated the resulting error rates and averaged them over ten tests (each with a five-fold cross validation). Table III shows the results.

The standard deviations of the error rates are still prominent as in the last experiment (Table II), probably reflecting the varying complexity of training/test sets in different trials. Nevertheless, the comparison suggests that KAAs tend to produce smaller false detection rates with variance comparable to the others.

TABLE III
FALSE NOVELTY DETECTION RATES WITHOUT NOVEL EXAMPLES

	KAA-1^G	KAA-2^G	KAA-1^P	KAA-2^P
Promoter	24.2 ± 7.5	20.7 ± 8.6	29.0 ± 11.0	27.9 ± 5.5
Sonar	31.6 ± 10.7	28.2 ± 6.6	34.9 ± 7.6	29.3 ± 7.9

	Parzen-window	Autoencoder	SOM-ND	SVM (One-Class)
Promoter	23.9 ± 9.1	27.7 ± 10.2	26.5 ± 9.4	25.4 ± 8.2
Sonar	29.5 ± 5.8	37.0 ± 8.2	32.5 ± 7.03	38.3 ± 5.5

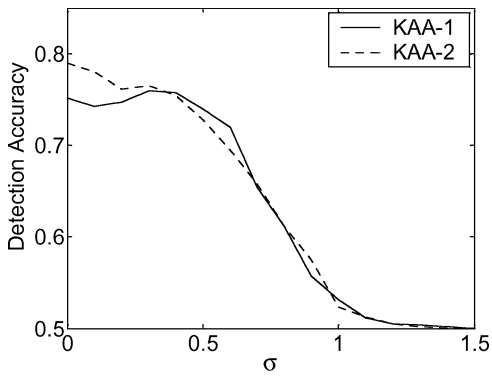


Fig. 6. Kernel autoassociators against noise for the promoter detection. Shown here is the resulting detection accuracies (without novel examples) as functions over the noise level (standard deviation).

C. Autoassociator-Based Novelty Detection Against Noise

To examine the robustness of the proposed method, we conducted an experiment by adding Gaussian noise onto the test

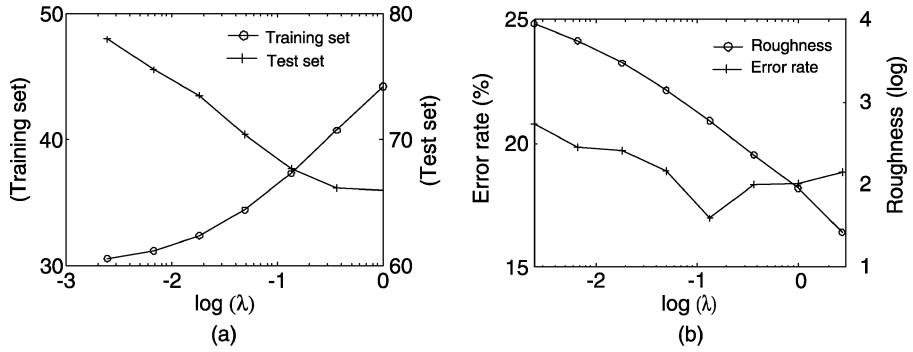


Fig. 7. Regularized networks in the promoter detection problem. Panel (a) shows the mean reconstruction errors for samples, respectively, from the training set and the test set, and the errors are drawn as the functions over the regularization parameter λ which controls the smoothness of the regularized networks. Panel (b) shows the roughness measure of the regularized networks and their performances in terms of recognition/detection error rates.

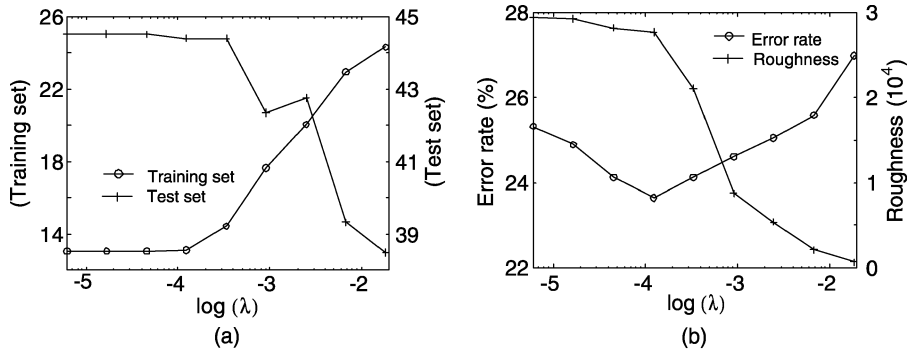


Fig. 8. Regularized networks in the sonar target recognition domain. Please refer to Fig. 7 for the explanation. Note the detection error rate is evaluated on the test set. (a) Reconstruction error. (b) Detection error rate versus polynomial roughness.

data in the promoter problem. The noise in each dimension is independent, with standard deviation varying from 0 to 1.5. Fig. 6 plots the results.

It needs to be mentioned that the standard deviation of the promoter patterns is about 1.1 on each dimension, while the detection accuracy remains larger than 70% under the noise up to $\sigma = 0.6$. The results demonstrate that the kernel machines are not very sensitive to additive noise.

D. Performance of Regularized Networks

The two novelty detection problems are used here as test beds to examine the proposed regularized kernel autoassociators with polynomial f_b. We carried out three experiments using five-fold cross-validation techniques and averaged the recognition results. A range of values for λ were tested to examine its effect on both the empirical reconstruction error and the roughness of the backward mapping function. The reconstruction error, the roughness, and the recognition performance of the regularized networks were obtained and are plotted in Fig. 7 and Fig. 8 as functions with respect to λ.

The results attest to the effectiveness of the regularization method in controlling the smoothness of the kernel networks. Although smoother networks yielded larger errors on the training set, they produced better results on the test set, showing better generalization performance. The figures also indicate that, with λ in an appropriate range, the detection performance of kernel autoassociators can be enhanced. In practice, the appropriate value for λ may be estimated empirically with cross-validation techniques on the training set. Future work is to study how to set λ using other automatic methods with lower computational complexity.

In the present study, we do not apply the regularization technique to other novelty detection and later $m$class classification tasks. The major reason is that since regularization is also an open issue in many other classification methods, for fair comparison we would like to compare the methods without considering regularization. Nonetheless, our preliminary study in the above shows that regularization could be an important issue in improving kernel autoassociators.

E. Discussions on Novelty Detection

This section has studied the autoassociator-based novelty detection system in various situations. The results indicate that either with or without novel examples, kernel autoassociators (especially the KAA-2^G) could achieve slightly better results than the others. Furthermore, the KAA systems demonstrated consistent performance against a varying number of novelty examples, in contrast to the MLP classifier that requires a large number of novelty examples for good performance.

Comparing the autoassociators in the two detection domains, those using Gaussian kernels seem to outperform others using polynomial kernels. Furthermore, the experimental results also attest to the robustness of Gaussian kernel autoassociators against additive noise.

The architectures of kernel autoassociators can be determined by only a few parameters. For the networks with linear backward mapping f_b, the user needs to choose between different types of kernels (Gaussian, polynomial, etc.). For kernel autoassociators

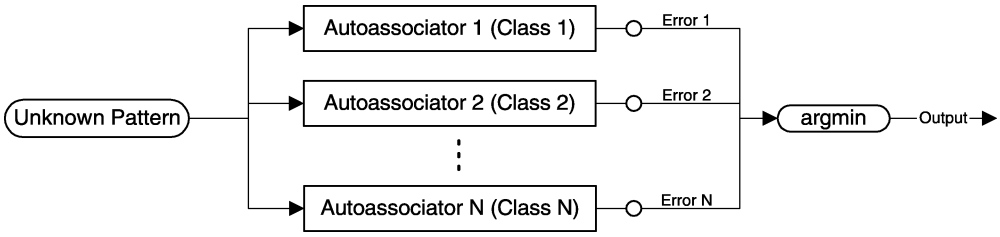


Fig. 9. m class classification scheme based on autoassociators. Each autoassociator is trained with the examples from its associated class. When an unknown pattern is presented, it will be processed by all the networks, respectively. The system will collect and compare the reconstruction errors and pick out the network with the minimal error.

with polynomial f_b, the user may need to choose an additional parameter—the number (N_a) of kernel principal components. At present, we use a simple method to choose the value for N_a. First, we can select those components that account for a large percent (e.g., 90%) of variance in the patterns. Around the value we then simply test a few numbers and choose the optimal one. This empirical configuration method also applies to some other methods under comparison, such as in choosing the number of hidden nodes for autoencoders.

However, as Markou and Singh put it [8], there is no single-best model for novelty detection and the success depends not only on the type of method used but also statistical properties of data handled. Thus, a specific parameter-setting method cannot be always suitable for different problems, and, in some cases, we may resort to empirical methods such as cross-validation for parameter setting ([4, p. 213]).

It is worthwhile to mention that Markou and Singh have summarized important principles related to novelty detection [8]. The principle of parameter minimization states that a novelty detection method should aim to minimize the number of parameters that are user set. Kernel autoassociators inherit an advantage of neural networks for novelty detection in that during network training, *a priori* information is not very critical on data distribution [43]. Furthermore, to define a kernel autoassociator, the user only needs to select one essential parameter: the kernel type, since the other parameters such as the number of principal components can be learned from the training samples. Hence, the proposed model adheres more closely to this principle than conventional autoencoders which need to define a number of parameters including the number and the dimension of layers, and the transfer functions. Besides, the principle of independence asserts that the novelty detection method should show reasonable performance in the context of imbalanced data set, small number of samples, and noise. Our experiments in Sections IV-A and IV-C indicate that the proposed method with kernel autoassociators offers satisfactory performance in such a context. An example is given in Fig. 5, when the novel examples count five, in addition, since the training/running of kernel autoassociators is of low-computational complexity, the networks enable online adaptation which is in accordance with the principle of adaptability and the principle of computational complexity.

V. Applications to Multiclass Classification

Being one-class learning machines, kernel autoassociators can be used for $m$class classification tasks if each autoassociator is associated with an individual category to learn its concept. The system would use a competitive classification scheme shown in Fig. 9: When a test pattern is presented, it will be reproduced by each autoassociator, the respective reproduction results will be collected and compared, with the best one indicating the corresponding class. In the subsections to follow, we examine this classification scheme on various recognition problems.

TABLE IV
Comparative Error Rates for Wine and Glass Classification

	KAA-1	KAA-2	Autoencoder	MLP	SVM
Wine	4.2 ± 2.7	3.1 ± 1.5	9.5 ± 3.37	7.4 ± 5.5	1.7 ± 1.3
Glass	37.9 ± 4.2	37.4 ± 5.4	40.7 ± 4.5	56.9 ± 6.1	42.2 ± 5.9

A. Wine and Glass Recognition

The *Wine Recognition* data [44] is acquired from UCL repository of machine learning databases. The data set contains three types of wines, each type has 59, 71, or 48 instances. The analysis of the wines determines the quantities of 13 constituents.

The *Glass Recognition* data is also acquired from UCL Repository of Machine Learning Databases. The data set contains instances of six types of glasses. Each type has 70, 17, 76, 13, 9, or 27 instances. The goal is to determine the glass type from nine attributes.

We used a two-fold cross-validation technique to test the autoassociator-based classification scheme in both domains. Typical classification machines such as multilayer perceptrons and support vector machine were also tested for comparison. It needs to be mentioned that all the data were normalized to the range $[-1, 1]$ to remove the scale effect, and each network was fine tuned. The comparative recognition results are summarized in Table IV.

The results show that for $m$class recognition, kernel autoassociators can be comparable to SVMs in terms of recognition accuracy, while producing lower error rates than both autoencoders and multilayer perceptrons. Besides, the small variances of error rates with most methods suggested that their performance would be consistent in the Wine/Glass domain.

B. Handwritten Digit Recognition

Here, we consider the handwritten digit recognition problem with the US-Postal Service (USPS) handwritten digit database that consists of 7291 training images and 2007 test images (16

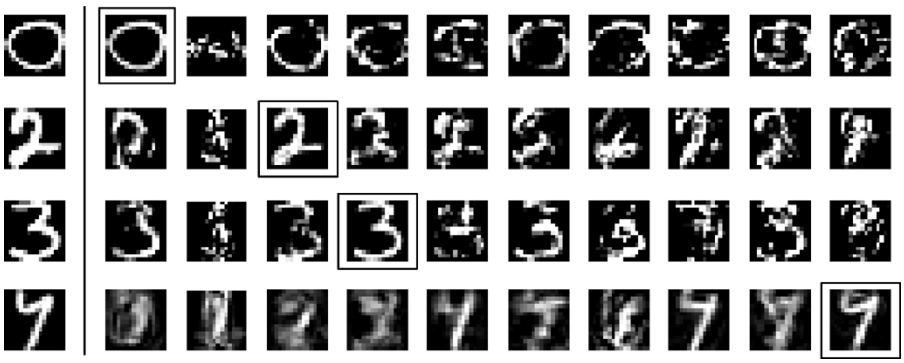


Fig. 10. Examples of handwritten digit recognition with kernel-autoassociator classifier on the USPS database. The leftmost column displays the test patterns; the columns to the right right display the reconstructed images by each of the ten kernel autoassociators. We use frames to mark the best ones in terms of reconstruction accuracy.

$\times$ 16 pixels) (the database is accessible at http://www.kernel-machines.org/data/usps.mat.gz).

To illustrate the classification process with autoassociators, some examples from the test set are shown in Fig. 10. The leftmost column displays the probe patterns, to the right their respective reproductions by the KAA-2 networks corresponding to 0 to 9. The first three rows in the figure show that three patterns of 0/2/3 get best reproduction from the correct networks (i.e., the first/third/fourth network). There exists only a few cases in which the correct network among all networks cannot reproduce the best reproduction for a new pattern. An example is given in the bottom row where a pattern of class 7 is misclassified as 9.

Table V summarizes the classification results in comparison with the published scores of other technologies. Here, we take $N_a = 9$ for the KAA-2 network. In addition, a newly proposed kernel Fisher discriminant method [45] called KDDA, as well as KPCA-NN—a 1-nearest-neighbor technique with KPCA features (see [45]), has been implemented for comparison. The results in the table demonstrate that KAA-1 and KAA-2 could match LeNet and SVM in terms of performance while significantly outperforming autoencoders or kernel density baseline (a kernel density estimate).

In the comparison, the tangent distance approach yielded the best result. This can be explained by the fact that it is the only one here that dedicatedly explores domain knowledge about the invariance of image patterns. Similarly, there is a variant of SVM referred to as virtual SVM [46] that, by incorporating prior knowledge about image invariance, gained considerable improvement and produced a good classification error rate of 3.0% on the USPS database. Similarity, incorportating domain knowledge may also yield a promising extension to kernel autoassociators.

TABLE V
RECOGNITION ERROR RATES ON USPS

Method	Error Rate
KAA-1	4.38%
KAA-2	4.68%
Autoencoder	7.42%
Kernel Density baseline [47]	5.5%
Nearest Neighbor [47]	5.7%
KDDA	10.0%
KPCA-NN	6.15%
LeNet1[48]	4.2%
SVM[49]	4.0%
Tangent Distance[50]	2.5%

C. Face Recognition on UMIST Database

The UMIST face database [51] consists of 575 gray-scale images for 20 individuals, each showing a wide range of poses from profile to frontal views. Some UMIST faces are displayed in Fig. 11, where large variations are clearly present.

In the experiment, we divide the data into a training set and a test set. The training set consists of six images per person, while the remaining images form the test set. We have tuned each system and use their best results for comparison (for instance, we empirically choose $N_a = 20$). Recognition results in terms of error rate over σ (the bandwidth of Gaussian kernel k) are compared in Fig. 12. Here, KPCA-NN denotes a 1-nearest-neighbor technique with KPCA features. In addition, two nonkernel techniques including 1-nearest-neighbor (NN) and an autoencoder are also compared.

The results indicate that the kernel autoassociator methods (KAA-1 and KAA-2) can produce better results than the others except KDDA. However, in the aforementioned OCR experiment, KAAs significantly outperformed KDDA. The difference may be due to the fact that KDDA and KAAs are based on different principles: KDDA aims to explore discriminating, nonlinear patterns in the training set, while KAAs would emphasize intra-class structural patterns (class concept). And the nature of a particular classification case would be of particular advantage to one principle. A systematical comparison between the two principles would go beyond the scope of this paper, while combining the two principles may be a promising way to improve the existing systems in the future.

D. Face Recognition on ORL Database

The Olivetti-Oracle Research Lab (ORL) database (available at http://www.uk.research.att.com/facedatabase.html) involves 40 individuals, and each person has ten different facial views

Fig. 11. Complex patterns present in multiview face recognition (examples from the UMIST database).

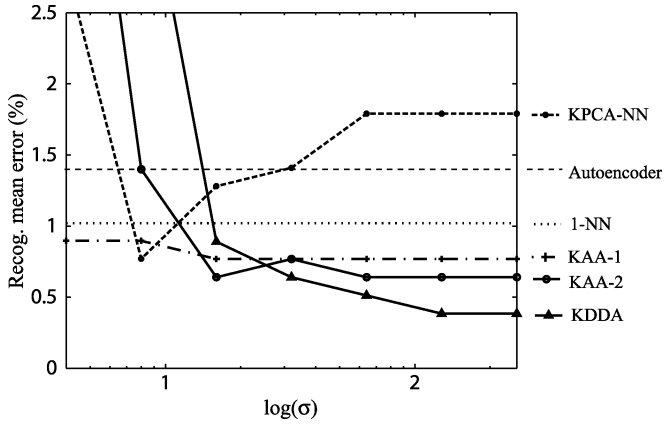


Fig. 12. Comparative face recognition results on the UMIST database. Shown here are the recognition error rates as functions of the bandwidth σ of Gaussian kernels used by the kernel machines. Note that 1-NN represents 1-nearest-neighbor technique, and it serves as a baseline together with another nonkernel machine, i.e., autoencoder.

Fig. 13. Face examples from the ORL database. Shown here are four persons, each with two face images.

with various expressions, small occlusion, different scale and orientations (Fig. 13). The resolution of all the images is 112×92. The ORL database allows us to compare the proposed method with other systems such as SOM+CN [52] (a face recognition scheme that combines the self-organizing map with convolutional neural networks) and Eigenfaces [3].

In the experiment, we randomly select a small number (three or five) of faces out of ten for each subject to set up an autoassociator, and then record the recognition accuracy on the remaining faces. Since there are only a few training samples available, the deformation variances of the faces are difficult to capture. One efficient approach for tackling the issue is to augment the training set with some synthetically-generated face images. In this experiment, we synthesize images by simple geometrical transformations including rotation and scaling. In the present work, we generate ten synthetic images from each original training image by making small, random perturbations to the original image: rotation (up to $+5$ and -5) and scaling (by a factor between 95% and 105%). Finally, the recognition results are listed in Table VI.

TABLE VI
RECOGNITION ACCURACY FOR ORL DATABASE

n	Eigenface	Autoencoder	SOM+CN	KAA-1	KAA-2
3	81.8	86.5	88.2	89.3	90.4
5	89.5	92.0	96.5	94.5	95.5

The results show that kernel autoassociators outperformed Eigenface and autoencoder techniques, while achieved comparable performance to SOM+CN.

VI. DISCUSSIONS AND CONCLUSION

In this paper, we have proposed a novel nonlinear autoassociator model. By making use of the kernel feature space, the model resorts to relatively simpler functions (linear and polynomials) for autoassociation learning, in contrast to conventional nonlinear machines using complex class of functions.

The kernel autoassociator model has been examined on novelty detection with or without novelty examples. Experimental results attest to the robustness of the model in the context of imbalanced data sets, small numbers of samples, and noise. Furthermore, the model has much less user-set parameters than conventional nonlinear autoassociators.

The $m$class recognition scheme based on autoasociators is a detection-based approach [53]. Thus, it has an advantage over discriminant methods in terms of adaptability. When a new subject is presented, the detection-based approach need not retrain existing networks. Instead, only a new network is to be created, together with the existing networks for classification.

In addition, the simulations and the extensive experiments have demonstrated that the proposed method can capture complex nonlinear features outperform conventional autoencoders. Compared with other systems, kernel autoassociators offer better or comparable performance, though they are generic one-class learning machines. In conclusion, the proposed method provides an alternative approach to nonlinear autoassociation modeling, and is promising for various nonlinear concept learning and recognition applications.

REFERENCES

[1] D. L. Medin and J. D. Coley, *Perception and Cognition at Century's End*, 2nd ed. San Diego, CA: Academic, 1998, pp. 403–440.
[2] W. Y. Zhao, R. Chellappa, A. Rosenfeld, and P. J. Phillips, "Face recognition: A literature survey," UMD CfAR Tech. Rep. CAR-TR-948, 2000.
[3] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cog. Neurosci.*, vol. 3, pp. 71–86, 1991.
[4] S. Haykin, *An Introduction to Neural Networks—A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.
[5] P. McLeod, K. Plunkett, and E. T. Rolls, *An Introduction to Connectionist Modeling of Cognitive Processes*. New York: Oxford Univ. Press, 1998.

[6] T. Kohonen, *Content-Addressable Memories*. Berlin, Germany: Springer, 1980.
[7] W. J. Daunicht, "Autoassociation and novelty detection by neuromechanics," *Science*, vol. 253, no. 5025, pp. 1289–1291, 1991.
[8] M. Markou and S. Singh, "Novelty detection: A review—Part 1: Statistical approaches," *Signal Process.*, pp. 2481–2497, 2003.
[9] ——, "Novelty detection: a review part 2: neural network based approaches," *Signal Process.*, pp. 2499–2521, 2003.
[10] R. Féraud, O. J. Bernier, J. E. Viallet, and M. Collobert, "A fast and accurate face detector based on neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 1, pp. 42–53, Feb. 2001.
[11] T. Petsche, A. Marcantonio, C. Darken, S. J. Hanson, G. M. Kuhn, and I. Santoso, "A neural network autoassociator for inductionmotor failure prediction," in *Adv. Neural Inf. Process. Syst.*, 1996, vol. 8, pp. 924–930.
[12] D. Y. Yeung and C. Chow, "Parzen window network intrusion detectors," in *Int. Conf. Pattern Recognit.*, 2002, pp. 385–388.
[13] S. J. Hanson and J. Kegl, "Parsnip: a connectionist network that learns natural language grammar from exposure to natural language sentences," in *Proc. 9th Annu. Conf. Cognitive Science*, 1987, pp. 106–119.
[14] H. Schwenk and M. Milgram, "Transformation invariant autoassociation with application to handwritten character recognition," in *Adv. Neural Inf. Process. Syst.*, 1995, pp. 991–998.
[15] B. Zhang and Y. Guo, "Face recognition by auto-associative radial basis function network," in *3rd Int. Conf. Audio-and Video-Based Biometric Person Authentication (AVBPA)*, 2001, pp. 52–58.
[16] B. Zhang, H. H. Zhang, and S. Ge, "Face recognition by applying wavelet subband representation and kernel associative memory," *IEEE Trans. Neural Netw.*, vol. 15, no. 2, pp. 166–177, Apr. 2004.
[17] P. Baldi and K. Hornik, "Neural networks and principal component analysis: learning from examples without local minima," *Neural Netw.*, vol. 2, pp. 53–58, 1989.
[18] D. E. Rumerlhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by back-propatation errors," *Nature*, vol. 323, pp. 533–536, 1986.
[19] N. Japkowicz, S. J. Hanson, and M. A. Gluck, "Nonlinear autoassociation is not equivalent to pca," *Neural Comput.*, vol. 12, pp. 531–545, 2000.
[20] G. W. Cottrell, P. Munro, and D. Zipser, "Learning internal representations of gray scale images: an example of extensional programming," in *Proc. 9th Annu. Cog. Sci. Soc. Conf.*, 1987, pp. 462–473.
[21] D. Valentin, H. Abdi, A. J. O'Toole, and G. W. Cotterell, "Connectionist models of face processing: a survey," *Pattern Recognit.*, vol. 27, pp. 120–1230, 1994.
[22] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Redwood City, CA: Addison-Wesley, 1991.
[23] K. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *J. AIChE*, vol. 37, pp. 233–243, 1991.
[24] D. DeMers and G. W. Cottrel, "Nonlinear dimensionality reduction," in *Adv. Neural Inf. Process. Syst.*, 1993, pp. 580–587.
[25] S. Usui, S. Nakauchi, and M. Nakano, "Internal color representation acquired by a five-layer neural network," in *Artificial Neural Networks*, T. Kohonen, O. Simula, and J. Kangas, Eds. New York: Elsevier, 1991, pp. 867–872.
[26] B. Moghaddam, "Principal manifolds and bayesian subspaces for visual recognition," in *Proc. IEEE Int. Conf. Computer Vision*, 1999, pp. 1131–1136.
[27] P. Comon, "Independent component analysis—a new concept?," *Signal Process.*, vol. 36, pp. 287–314, 1994.
[28] E. C. Malthouse, "Limitations of nonlinear pca as performed with generic neural networks," *IEEE Trans. Neural Netw.*, vol. 9, no. 1, pp. 165–173, Feb. 1998.
[29] B. Scholkopf and A. Smola, *Learning With Kernels*. Cambridge, MA: MIT Press, 2002.
[30] S. Saitoh, *Theory of Reproducing Kerels and Its Applications*. Harlow, U.K.: Longman, 1988.
[31] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[32] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural Comput.*, vol. 12, no. 10, pp. 2385–2404, 2000.
[33] B. Scholkopf, A. Smola, and K. R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, pp. 1299–1319, 1998.
[34] B. Scholkopf, S. Mika, and C. J. C. Burges, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000–1017, Oct. 1999.
[35] G. Wahba, "Soft and hard classification by reproducing kernel hilbert space method," *Proc. Nat. Acad. Sci.*, vol. 99, pp. 16 524–16 530, 2002.
[36] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 686, pp. 337–404, 1950.
[37] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[38] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 326–334, Jun. 1965.
[39] V. A. Morozov, *Method for Solving Incorrectly Posed Problems*. New York: Springer, 1984.
[40] D. W. Scott, *Multivariate Density Estimation*. New York: Wiley, 1992.
[41] A. Ypma and R. P. W. Duin, "Novelty detection using self-organizing maps," in *Progr. Connectionist Based Information Systems 2*, 1998, pp. 1322–1325.
[42] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.
[43] M. Markou and S. Singh, "An approach to novelty detection applied to the classification of image regions," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. Apr., pp. 396–407, 2004.
[44] S. Aeberhard, D. Coomans, and O. de Vel, "Comparison of classifiers in high dimensional settings," Dept. Comput. Sci. Dept. Math. Stat., James Cook Univ., North Queensland, Australia, Tech. Rep. 92-02, 1992.
[45] L. Juwei, K. N. Plataniotis, and A. N. Venetsanopoulos, "Face recognition using kernel direct discriminant analysis algorithms," *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 117–126, Feb. 2003.
[46] B. Scholkopf, *Support Vector Learning*. Munich, Germany: Oldenbourg Verlag, 1997.
[47] D. Keysers, J. Dahmen, T. Theiner, and H. Ney, "Experiments with an extended tangent distance," in *Proc. Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 38–42.
[48] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. J. Jackel, "Backproparation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, pp. 541–551, 1999.
[49] B. Scholkopf, C. J. C. Burges, and V. N. Vapnik, "Extracting support data for a given task," in *Int. Conf. Knowledge Discovery and Data Mining*, 1995, pp. 252–257.
[50] P. Simard, Y. LeCun, J. Denker, and B. Victorri, "Transformation invariance in pattern recognition—tangent distance and tangent propagation," in *Neural Networks: Tricks of the Trade, Lecture Notes Comput. Sci.*, G. Orr and K. R. Muller, Eds. Heidelberg, Germany: Springer, 1998, pp. 239–274.
[51] D. B. Graham and N. M. Allinson, "Characterizing virtual eigensignatures for general purpose face recognition," in *Face Recognition: From Theory to Applications*. New York: Springer-Verlag, 1998, pp. 446–456.
[52] S. Lawrence, C. L. Giles, A. Tsoi, and A. Back, "Face recognition: a convolutional neural network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Feb. 1997.
[53] N. Japkowicz, C. Mayers, and M. A. Gluck, "A novelty detection approach to classification," in *Proc. 14th Joint Conf Artificial Intelligence*, 1995, pp. 518–523.

Haihong Zhang received the B.Eng. degree in electronics engineering from the Hefei University of Technology, Hefei, China, and the M.Eng. degree from the University of Science and Technology of China, Hefei, in 1997 and 2000, respectively. He is currently pursuing the Ph.D. degree at the School of Computing, National University of Singapore.

He joined the Institute for Infocomm Research, Singapore, in 2004 as a Research Engineer. His research interests include neural networks and pattern recognition, neuroinformatics, and computer vision.

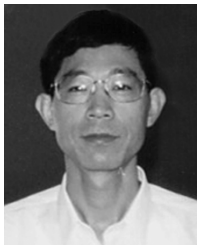
Weimin Huang (M'96) received the B.Eng. degree in automation and the M.Eng. and Ph.D. degrees in computer engineering from Tsinghua University, Beijing, China, in 1989, 1991, and 1996, respectively.

He is a Research Scientist with the Institute for Infocomm Research, Singapore. He has worked on the research of handwriting signature verification, biometrics authentication, and audio/video event detection. He is leading a group working on video surveillance and healthcare projects. His current research interests include pattern recognition, image processing, computer vision, human computer interaction, and statistical learning.

Zhiyong Huang (M'01) received the B.Eng. and M.Eng. degrees in computer science and engineering, respectively, from Tsinghua University of Beijing, Beijing, China, and the Ph.D. degree in computer science from EPFL, Switzerland, in 1986, 1988, and 1997, respectively.

He is currently an Assistant Professor with the Department of Computer Science, National University of Singapore. His research interests include visualization, computer graphics, computer vision and pattern recognition, and multimedia databases.

Dr. Huang is a member of ACM.



Bailing Zhang received the M.S. degree in communication and electronic systems from the South China University of Technology, Guangzhou, China, and the Ph.D. degree in electrical and computer engineering from the University of Newcastle, NSW, Australia, in 1987 and 1999, respectively.

He is currently a Lecturer with the School of Computer Science and Mathematics, Victoria University of Technology, Melbourne, Australia. He was a Research Staff Member in the Kent Ridge Digital Labs (KRDL), Singapore. Prior to his research activities in Singapore, he was a Postdoctoral Fellow with the School of Electrical and Information Engineering, University of Sydney, Sydney, Australia. His research interests include pattern recognition, computer vision, and artificial neural networks.