Recurrent Correlation Associative Memories: A Feature Space Perspective

Renzo Perfetti and Elisa Ricci

***Abstract*—In this paper, we analyze a model of recurrent kernel associative memory (RKAM) recently proposed by Garcia and Moreno. We show that this model consists in a kernelization of the recurrent correlation associative memory (RCAM) of Chiueh and Goodman. In particular, using an exponential kernel, we obtain a generalization of the well-known exponential correlation associative memory (ECAM), while using a polynomial kernel, we obtain a generalization of higher order Hopfield networks with Hebbian weights. We show that the RKAM can outperform the aforementioned associative memory models, becoming equivalent to them when a dominance condition is fulfilled by the kernel matrix. To ascertain the dominance condition, we propose a statistical measure which can be easily computed from the probability distribution of the interpattern Hamming distance or directly estimated from the memory vectors. The RKAM can be used below saturation to realize associative memories with reduced dynamic range with respect to the ECAM and with reduced number of synaptic coefficients with respect to higher order Hopfield networks.**

***Index Terms*—Exponential correlation associative memory (ECAM), feature space, higher order Hopfield network, kernel associative memory, recurrent correlation associative memory (RCAM), support vector machine (SVM).**

I. Introduction

ASSOCIATIVE MEMORIES (AMs) are neural networks (NNs) characterized by information storage and recall capabilities. Such networks, by virtue of their ability to recall information from a partial or imprecise version of the stored pattern, can be useful in a number of applications, including error correction, image restoration, and pattern recognition. The idea of AM can be traced back to the linear AM proposed by Kohonen [1]. However, it was the seminal paper of Hopfield [2] in 1982 which attracted the interest of many researchers in the field of dynamical (recurrent) associative memories. A dynamic AM evolves in time until it reaches a fixed point which is interpreted as one of the stored patterns. The points starting from which the trajectory will converge to the equilibrium point represent its basin of attraction. An attractive feature of the Hopfield model is the delocalization of information storage: the $N \times N$ connection matrix encodes the N-bit stored vectors, so that the memory size is independent of the number of stored patterns.

The dynamic AM should have as many equilibrium points as the patterns to be stored; moreover, the basins of attraction should be as large as possible and the number of undesired equilibrium points should be as small as possible. Many design methods have been proposed to design AMs fulfilling the above mentioned constraints. In the Hopfield network, following the Hebbian learning, perfect recall is possible only for orthogonal patterns; if the patterns are correlated with each other or noisy, the network cannot perform correctly the task. The capacity of the Hopfield network can be estimated as $\sim 0.14N$, where N is the dimensionality of the stored vectors [3]. Using more sophisticated techniques to design the connection matrix, $o(N)$ correlated patterns can be stored but the error correction is often unsatisfactory [4]–[7].

The Hopfield network is an additive network, i.e., the input to a neuron is a weighted sum of the outputs of the remaining neurons. The limitations of these networks in terms of capacity and error correction stimulated the research in the area of AMs with nonlinear interactions [8]–[11]. In higher order Hopfield networks, the input to a neuron depends on the products of two or more neuron outputs. The main inconvenience of these systems is the huge number of connection weights, which rapidly becomes prohibitive for a practical realization.

An ideal (nearest neighbor) AM should produce as output the stored vector most similar to the input in some sense. It could be realized by computing the distance (correlation) between the input vector and each of the stored items, then selecting the closest one. In a serial implementation of this system, both space complexity and recall time increase as $o(M)$ where M is the number of prototypes. A parallel solution, avoiding the selection of the best matching pattern, was proposed by Marks *et al.* [12]. Here, the outputs of a matched filter bank are transformed by a nonlinear function and then used to weight the stored vectors. The weighted sum gives an estimate of the memory pattern closest to the input. This idea was embedded in a recurrent network by Chiueh and Goodman [13] who invented the recurrent correlation associative memory (RCAM) and generalized to multivalued patterns in [14]. In an RCAM the nonlinear transformation of the inner product is used to obtain an approximation of one of the stored patterns; this distorted version is fed back to the input and the process is iterated until convergence to the correct pattern. In this network, the convergence time is almost unaffected by the number of stored vectors, hence, while space complexity is still $o(M)$, the recall time is independent of M. When the nonlinear function is an exponential, we obtain the exponential correlation associative memory (ECAM), which has been deeply investigated both analytically and through simulations by several authors [13]–[18]. It has been shown that the ECAM can reach the capacity of an ideal AM, which is exponential with respect to the length of memory vectors [16]. However, also the dynamic range required by the exponentiation circuits grows exponentially, making the very large scale integration (VLSI) realization very difficult [13].

Manuscript received May 18, 2006; revised May 17, 2007; accepted July 2, 2007.

The authors are with the Department of Electronic and Information Engineering, University of Perugia, I-06125 Perugia, Italy (e-mail: perfetti@diei.unipg.it; elisa.ricci@diei.unipg.it).

Digital Object Identifier 10.1109/TNN.2007.909528

The idea of a nonlinear transformation of the inner product leads naturally to the kernelization of the associative memory [19]–[21]. A kernel function satisfying Mercer's theorem can be interpreted as a scalar product in a higher dimensional (possibly infinite-dimensional) feature space where the memory patterns are orthogonalized [22]. Having the mapped patterns uncorrelated, the Hopfield's philosophy can be used to design the AM: construct an energy function whose minima correspond to the memory vectors. The approaches in [9], [12], [13], [19], and [20] can be considered as methods to orthogonalize the memory vectors in order to reduce cross talk during the recall phase. Recently, a novel recurrent AM has been proposed by Garcia and Moreno [23] exploiting the kernel trick used in the framework of support vector machines (SVMs).

Recent advances in machine learning research have pointed out the advantages of SVM over other classification techniques [24]. Sound theoretical basis, good generalization capabilities, and uniqueness of solution are among the most appealing aspects. SVM learning consists in the solution of a constrained quadratic optimization problem whose Hessian is the kernel matrix. In [23], the update equation of the dynamic AM is interpreted as a classification step and the classifier is built in a learning phase using the kernel–adatron algorithm [25].

In this paper, motivated by [23], we investigate the properties of recurrent kernel associative memories (RKAMs) based on SVM training and compare them to existing models of nonlinear dynamic AMs, like the ECAM and the higher order Hopfield network. We show that the RKAM is equivalent to the well-known RCAM when a dominance condition is fulfilled by the kernel matrix. To evaluate this condition in a principled way, we propose a statistical measure which can be easily computed and results to be meaningful for the kernel functions considered herein. As shown through several simulation results, when the dominance condition is not met the RKAM outperforms the RCAM. This behavior makes possible the realization of correlation-based AMs with good performance and reduced dynamic range.

The rest of this paper is organized as follows. In Section II, nonlinear SVMs are briefly reviewed. In Section III, we introduce the kernel AM model and show its relation with the RCAM. Section IV addresses the properties of the exponential RKAM and compares it with the ECAM. Section V is dedicated to the polynomial RKAM which is compared to the higher order Hopfield network. In Section VI, several simulation results are presented. Finally, some comments conclude this paper.

II. Support Vector Machines

Let $(\mathbf{x}^{(m)}, y^{(m)}), m = 1, \ldots, M$ represent the training examples of a classification problem; each example $\mathbf{x}^{(m)} \in R^N$ belongs to the class $y^{(m)} \in \{-1, +1\}$. In the case of linearly separable classes, there exist infinite separating hyperplanes. The distance between a separating hyperplane and the closest data points is called *margin*. An SVM identifies the optimal separating hyperplane (OSH) that maximizes this margin. The data points $\mathbf{x}^{(m)}$ which lie closest to the OSH are called support vectors (SVs). The solution with maximum margin corresponds to the best generalization ability [24].

Linearly nonseparable data points in *input space* can be mapped into a higher dimensional (possibly infinite-dimensional) *feature space* H through a nonlinear mapping function $\Phi(\cdot)$, so that the images of data points become linearly separable.

The maximization of the margin in feature space can be obtained by solving the following optimization problem in dual form [24].

Minimize

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{m=1}^{M}\sum_{n=1}^{M}\alpha^{(m)}\alpha^{(n)}y^{(m)}y^{(n)}K\left(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right) - \sum_{m=1}^{M}\alpha^{(m)} \quad (1)$$

subject to the linear constraints

$$\sum_{m=1}^{M}\alpha^{(m)}y^{(m)} = 0 \quad (2a)$$

$$\alpha^{(m)} \geq 0, \qquad m = 1, \ldots, M \quad (2b)$$

In (1), $\alpha^{(m)}$ is a Lagrange multiplier and $K(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}) = \langle \Phi(\mathbf{x}^{(m)}), \Phi(\mathbf{x}^{(n)}) \rangle$ represents the inner product between $\mathbf{x}^{(m)}$ and $\mathbf{x}^{(n)}$ in H; it is called *kernel* function. K is a valid kernel for the SVM if the symmetrical *Gram matrix* $\mathbf{K}$ with entries $K_{mn} = K(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})$ is positive semidefinite for every training set. Widely used kernels, satisfying this condition, are the Gaussian radial basis function (RBF) $K(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}) = \exp(-\|\mathbf{x}^{(m)} - \mathbf{x}^{(n)}\|^2/\sigma^2)$ and the polynomial function $K(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}) = (\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \rangle + R)^d$ with d positive integer and $R > 0$.

The optimal decision function in input space is

$$f(\mathbf{x}) = \sum_{m=1}^{M}\alpha^{(m)}y^{(m)}K\left(\mathbf{x}^{(m)}, \mathbf{x}\right) + b. \quad (3)$$

Only the Lagrange multipliers associated to SVs are greater than zero, so the decision function depends uniquely on them.

Usually, the bias b has a negligible effect on class separation due to the high dimensionality of the feature space [26]. Thus, it can be set to zero in (3); with respect to the general formulation, the equality constraint (2a) disappears. It is worth noting that functions (1) and (3) involve only K, hence the nonlinear mapping Φ is never explicitly used. This is known as the *kernel trick*.

Previous formulation can be extended by allowing a small number of classification errors on the training set (the *soft margin* SVM). A user-defined constant $C > 0$ is introduced which controls the tradeoff between the maximization of the margin and the minimization of classification errors on the training set [24]. With respect to the previous case, the dual formulation of the QP is the same with a difference in the bound constraints

$$0 \leq \alpha^m \leq C, \qquad m = 1, \ldots, M. \quad (4)$$

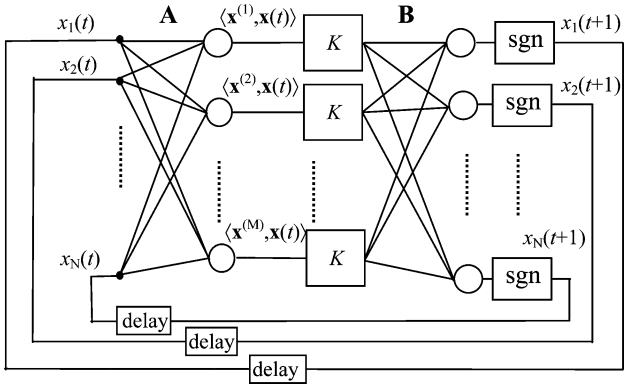

Fig. 1. Scheme of an RKAM storing M N-bit vectors.

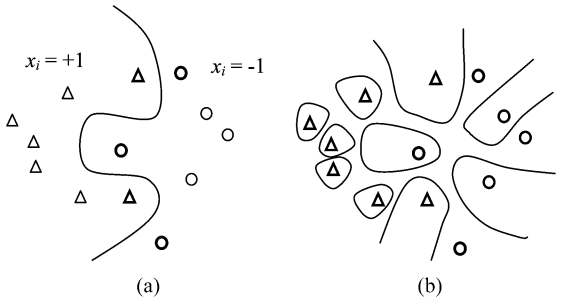

Fig. 2. (a) Separation of memory patterns in input space in presence of generalization. The class labels correspond to the ith component. SVs are highlighted. (b) Separation in presence of saturation. All patterns are SVs.

III. Recurrent Kernel Associative Memory

Let $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(M)}$ be M N-bit binary vectors whose components are -1 or $+1$. They are the memory patterns (prototypes) to be stored in the associative memory. The dynamic evolution of an RKAM is governed by the following [23]:

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}(t)\right\rangle\right)\right), \quad i = 1, \ldots, N \quad (5)$$

where $\mathbf{x}(t) = [x_1(t) \ldots x_N(t)] \in \{-1, +1\}^N$ is the state vector at time t. $K(\langle \mathbf{x}^{(m)}, \mathbf{x}\rangle)$ is a valid kernel function of the dot product between $\mathbf{x}^{(m)}$ and $\mathbf{x}$. $\alpha_i^{(m)}, i = 1, \ldots, N, m = 1, \ldots, M$ are nonnegative coefficients, which can be computed through SVM learning, as explained in the following.

Fig. 1 illustrates the architecture of the RKAM. Connection weights are represented by matrices $\mathbf{A}$ and $\mathbf{B}$. $\mathbf{A}$ is an $M \times N$ matrix whose entries are $A_{mi} = x_i^{(m)}$. $\mathbf{B}$ is an $N \times M$ matrix with entries $B_{im} = \alpha_i^{(m)} x_i^{(m)}$. Equation (5) can be used both in asynchronous and parallel modes. Comparing expressions (3) and (5), the update step can be interpreted as a classification operation assigning to the previous state vector $\mathbf{x}(t)$ the label represented by the new state component $x_i(t+1)$. Thus, assuming asynchronous operation, the network dynamics corresponds to an iterative assignment of new labels, starting from a corrupted memory vector $\mathbf{x}(0)$.

The design of the associative memory based on model (5) can be obtained by using a popular approach in the AM literature: Find the design parameters (in this case, the $\alpha_i^{(m)}$) so as to impose equilibrium points corresponding to the memory patterns. An equilibrium point $\mathbf{x}$ of system (5) satisfies the following equilibrium conditions:

$$x_i = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}\right\rangle\right)\right), \quad i = 1, \ldots, N. \quad (6)$$

Hence, we can formulate the following synthesis problem.

Find $\alpha_i^{(m)}, i = 1, \ldots, N; m = 1, \ldots, M$, such that

$$x_i^{(\ell)}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(\ell)}\right\rangle\right)\right) > 0, \quad \ell = 1, \ldots, M. \quad (7)$$

The RKAM design can be formulated as the solution of N *classification* problems, with training sets given, respectively, by $(\mathbf{x}^{(\ell)}, x_i^{(\ell)}), \ell = 1, \ldots, M$; each vector $\mathbf{x}^{(\ell)}$ belongs to the class represented by $x_i^{(\ell)} \in \{-1, +1\}$. These problems can be solved by a set of N independent SVMs, where $\mathrm{SVM}(i)$ is characterized by the Lagrange multipliers $\alpha_i^{(m)}, m = 1, \ldots, M$.

A pictorial interpretation of the SVM design is shown in Fig. 2(a). The ith SVM separates the memory vectors $\mathbf{x}^{(1)} \ldots \mathbf{x}^{(M)}$ on the basis of the ith component. The nonlinear separating surface in input space corresponds to a linear separation with maximum margin in feature space. Assume the state vector $\mathbf{x}(t)$ differs from a memory vector $\mathbf{x}^{(m)}$ in the ith component, i.e., $x_i(t) = -x_i^{(m)}$. If $\mathbf{x}(t)$ belongs to the same decision region of $\mathbf{x}^{(m)}$ with respect to the ith SVM, the update (5) assigns the new state vector to the correct class, making $x_i(t+1) = x_i^{(m)}$. Hence, the system evolution pushes the state vector nearest to the stored vector $\mathbf{x}^{(m)}$.

Note that for each machine it is $b_i = 0$. Moreover, we require that all the prototypes are stored in the AM as equilibrium points, so that no violation is allowed in conditions (7). Hence, in the following, we consider only the *hard margin* case, corresponding to $C = \infty$ in (4).

The computational complexity of the RKAM design is not negligible, since we need to train as many SVMs as the dimensionality of the stored patterns. However, efficient packages are available to solve the SVM optimization problem, e.g., SVM$^{\mathrm{light}}$ [27] or Libsvm [28]. In particular, Libsvm, based on Platt's sequential minimal optimization (SMO) [29], can successfully handle classification problems with large amounts of vectors (10^3–10^5). An evaluation of the run time for the SMO algorithm can be found in [29]. The computation time scales between $o(M)$ and $o(M^{2.2})$ with the number M of data points; it depends on the training set and the kernel function. With respect to the general case, the present design problem is simplified since, setting b to zero, the equality constraint (2a) vanishes. In particular, when the bias is zero, both the SMO and the kernel–adatron [25] algorithms reduce to a simple gradient descent technique, equivalent to the Gauss–Siedel method [30]. The runtime computational load of an RKAM is dominated by evaluation of the kernel between the incoming vector and each of the memory vectors. This phase can be efficiently implemented using special purpose digital hardware [31] or a suitably modified multiplierless kernel [32].

The dynamic rule (5) and the scheme in Fig. 1 are similar to those of an RCAM, whose state equation is [13]

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} x_i^{(m)} f\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}(t)\right\rangle\right)\right), \quad i = 1, \ldots, N. \tag{8}$$

We observe the following two main differences between the two models.

— In an RCAM ,we have $\alpha_i^{(m)} = 1$ for every m and i; in the RKAM, the coefficients $\alpha_i^{(m)}$ are computed through SVM learning.

— The excitation function f of the RCAM must be a valid kernel ($\mathbf{K}$ positive semidefinite) to be used in the RKAM.

In [13], it is proved that the RCAM dynamics is convergent provided that the excitation function f is monotone nondecreasing on $[-N, N]$ [13]. The proof in [13] cannot be extended to the RKAM due to the presence of the Lagrange coefficients. Indeed, no convergence result is known for model (5). However, our extensive simulations showed a convergent behavior of RCAM and RKAM with both asynchronous and synchronous dynamics, even if the nonlinear excitation function (i.e., the kernel function) is not monotone nondecreasing, as with polynomial functions of even degree.

It is known that the performance of an SVM classifier can be poor if the Gram matrix exhibits *diagonal dominance*, i.e., if the diagonal entries of the symmetric matrix $\mathbf{K}$ are large compared to the off-diagonal values [33]

$$K_{mm} \gg |K_{mn}| \quad \text{for } m \neq n, \qquad m, n = 1, \ldots, M. \tag{9}$$

In this case, the generalization ability is compromised since the classifier memorizes the training data. This phenomenon, well known as *overfitting* in the machine learning literature, corresponds exactly to the desired behavior of an associative memory: the training data (prototypes) are stored in the system parameters. Diagonal dominance corresponds to approximate orthogonality of any pair of different memory patterns in feature space. This can be easily verified observing that

$$\cos\theta_{mn} = \frac{\left\langle \Phi\left(\mathbf{x}^{(m)}\right), \Phi\left(\mathbf{x}^{(n)}\right)\right\rangle}{\left\|\Phi\left(\mathbf{x}^{(m)}\right)\right\| \left\|\Phi\left(\mathbf{x}^{(n)}\right)\right\|} = \frac{K_{mn}}{\sqrt{K_{mm}K_{nn}}} \approx 0 \tag{10}$$

where θ_{mn} represents the angle between vectors $\mathbf{x}^{(m)}$ and $\mathbf{x}^{(n)}$ mapped in feature space.

A simple relation can be found between an RKAM and the corresponding RCAM.

Property: The RKAM (5) becomes equivalent to the RCAM (8) with $f(\cdot) = K(\cdot)$ if the Gram matrix satisfies the dominance condition (9) and $K_{mm} = K_o$ for every m (identical diagonal entries).

Proof: In the hard margin case with $b = 0$, the SVM solution corresponds to the following QP.

Minimize

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{1} \tag{11a}$$

subject to the linear constraint

$$\boldsymbol{\alpha} \geq \mathbf{0} \tag{11b}$$

where $\alpha = [\alpha^{(1)} \ldots \alpha^{(M)}]^T, 1 = [1 \ldots 1]^T$. According to the previous assumptions, it is

$$J(\boldsymbol{\alpha}) \cong \frac{1}{2} K_o \boldsymbol{\alpha}^T \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{1} \tag{12}$$

where $K_o > 0$ represents the diagonal entries. It is easy to verify that the constrained minimum of (12) corresponds to $\boldsymbol{\alpha} = (1/K_o)\mathbf{1}$. Hence, all $\alpha^{(m)}$'s are identical. Since (5) is not affected by the common value $1/K_o > 0$, the RKAM is equivalent to the RCAM (8) with $f = K$. Note that when $\boldsymbol{\alpha} = (1/K_o)\mathbf{1}$ all the memory patterns are SVs [Fig. 2(b)]. ■

To summarize we can distinguish two operating modes for an RKAM.

1) *Generalization mode:*
   — the memory vectors are not orthogonal in feature space;
   — the number of SVs is less than M;
   — at each time step, the state vector is assigned to the most *similar* class of memory patterns;
   — with respect to the corresponding RCAM, the RKAM exploits the Lagrange multipliers to maximize the error correction at each time step.
2) *Saturation mode:*
   — the memory vectors are orthogonal in feature space;
   — all the memory vectors are SVs;
   — at each time step, the state vector is assigned to the nearest memory pattern;
   — the RKAM is equivalent to the corresponding RCAM.

The orthogonality of mapped memory patterns corresponds to the best recall performance in Hamming sense. In this case, the best we can do is to use an RCAM. However, as it will be explained in Section IV, the realization complexity of an RCAM increases when we approach the condition of orthogonality, due to the increased dynamic range. On the other hand, the RKAM outperforms the RCAM when the underlying SVMs do not overfit the data; so, it is possible to exploit this property to design recurrent associative memories with good performance and reduced dynamic range.

Having established that SVM learning can enhance the performance of RCAMs when the dominance condition is not met, the important question is as follows: How to check this condition? For positive kernels the tendency to overfitting can be measured computing the following *dominance ratio* [34]:

$$\rho = \frac{\frac{1}{M}\sum_m K_{mm}}{\frac{1}{M(M-1)}\sum_m \sum_{n,n\neq m} K_{mn}}. \tag{13}$$

Assuming identical diagonal entries, we can write

$$\frac{1}{\rho} = \frac{1}{M(M-1)}\sum_m \sum_{n,n\neq m} \frac{K_{mn}}{K_o} = \frac{1}{M(M-1)}\sum_m \sum_{n,n\neq m} \cos\theta_{mn}. \tag{14}$$

Hence, to evaluate the benefit of Lagrange multipliers in (5), we can compute the dominance ratio (13) or, equivalently, the expectation of $\cos\theta, \theta$ being the angle between mapped vectors. The utility of this measure is shown in the following sections.

IV. Exponential Kernel

The first kernel we consider in (5) is the exponential kernel

$$K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \right\rangle\right) = a^{\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \rangle} \tag{15}$$

with $a > 1$. The dynamic equation becomes

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} a^{\langle \mathbf{x}^{(m)}, \mathbf{x}(t) \rangle}\right), \quad i = 1, \ldots, N. \tag{16}$$

It can be shown that function (15) is a valid kernel [35]. Note that $\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \rangle = N - 2h(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})$, where $h(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})$ represents the Hamming distance between $\mathbf{x}^{(m)}$ and $\mathbf{x}^{(n)}$; hence

$$a^{\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \rangle} = a^N a^{-2h(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})}. \tag{17}$$

Since $a > 1, 0 < a^{-2} < 1$ and our exponential kernel is equivalent to the *sparse kernel* used in text classification [36]. It is also easy to show the relation of the exponential kernel with the widely used Gaussian RBF kernel: $K(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}) = \exp(-\|\mathbf{x}^{(m)} - \mathbf{x}^{(n)}\|^2/\sigma^2)$. Taking into account that

$$\left\|\mathbf{x}^{(m)} - \mathbf{x}^{(n)}\right\|^2 = 2N - 2\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \right\rangle \tag{18}$$

we have

$$\begin{aligned} &\exp\left(-\left\|\mathbf{x}^{(m)} - \mathbf{x}^{(n)}\right\|^2 \Big/ \sigma^2\right) \\ &\quad = \exp(-2N/\sigma^2) \exp\left(2\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \right\rangle \Big/ \sigma^2\right) \\ &\quad = G a^{\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \rangle} \end{aligned}$$

i.e., the Gaussian kernel corresponds to an exponential kernel with $a = \exp(2/\sigma^2)$ apart from multiplication of the Gram matrix by the scaling factor $G = \exp(-2N/\sigma^2)$. The corresponding Lagrange multipliers are simply scaled by $1/G$, so this scaling factor does not influence the dynamic behavior of system (16). Hence, the Gaussian kernel is equivalent to the exponential kernel. A recurrent associative memory based on a Gaussian kernel was proposed in [23].

For the exponential kernel, we have

$$K_{mm} = a^N \quad K_{mn} = a^N a^{-2h(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})}$$

hence, taking into account (10)

$$\cos\theta_{mn} = a^{-2h(\mathbf{x}^{(m)}, \mathbf{x}^{(n)})}. \tag{19}$$

The exponential kernel orthogonalizes the input patterns if the Hamming distance between them is large enough. The range of h values corresponding to orthogonal patterns in feature space increases with a. Since $\cos\theta_{mn} \to 0$ for every h when $a \to \infty$, the exponential kernel can orthogonalize arbitrary input patterns if a is sufficiently large. This behavior explains the good capacity of the ECAM, whose nonlinear function is just the exponential kernel [13]. The ECAM dynamics is represented by the following:

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} x_i^{(m)} a^{\langle \mathbf{x}^{(m)}, \mathbf{x}(t) \rangle}\right), \quad i = 1, \ldots, N. \tag{20}$$

Note that the ECAM is equivalent to the proposed RKAM if $\alpha_i^{(m)} = 1$ for every m and i. It has been shown that the ECAM exhibits exponential storage capacity with N and meets the upper bound for the capacity of associative memories when $a \to \infty$ [13], [16]. However, also the required dynamic range D grows exponentially with N, being $D = a^N$. As pointed out in [13], "in any real implementation this requirement is very difficult to meet, if not impossible." Exploiting the Lagrange multipliers, it is possible to improve the performance of the ECAM when the dynamic range is small, i.e., $a \cong 1$, which is the most appealing case from a realization viewpoint.

To ascertain the range of the radix a for which the RKAM can outperform the ECAM we evaluate the orthogonality of random patterns in feature space. To this end, we compute the expected value of $\cos\theta$, where the random variable θ denotes the interpattern angle in feature space. This analysis requires a model of input patterns distribution. In the following, we assume input vectors whose bit values are independent identically distributed random variables. The values $+1$ and -1 occur with probabilities p and $1 - p$, respectively. Under these assumptions, the probability distribution of the interpattern Hamming distance h is binomial, with mean Np and variance $Np(1-p)$ [17]

$$P(h) = \binom{N}{h} p^h (1-p)^{N-h}. \tag{21}$$

Limiting our attention to the case of symbols with equal probability, i.e., $p = 0.5$, we have

$$P(h) = \binom{N}{h} 2^{-N}. \tag{22}$$

The expectation of $\cos\theta$ is easily obtained using the binomial formula

$$\begin{aligned} E(\cos\theta) &= \sum_{h=0}^{N} P(h) a^{-2h} = 2^{-N} \sum_{h=0}^{N} \binom{N}{h} (a^{-2})^h \\ &= 2^{-N}(1 + a^{-2})^N. \end{aligned} \tag{23}$$

Plotting expression (23) versus a for different values of N, we obtain the curves in Fig. 3. We can see that $E(\cos\theta)$ is monotone decreasing with a. It is easy to obtain a critical value a_{crit}, above which $E(\cos\theta)$ becomes less than a given threshold δ

$$a_{\text{crit}} = \frac{1}{\sqrt{2\delta^{1/N} - 1}}. \tag{24}$$

In the following, we assume $\delta = 0.01$. For $a > a_{\text{crit}}$, the memory patterns are almost orthogonal in feature space, so we expect a comparable behavior for ECAM and RKAM. On the

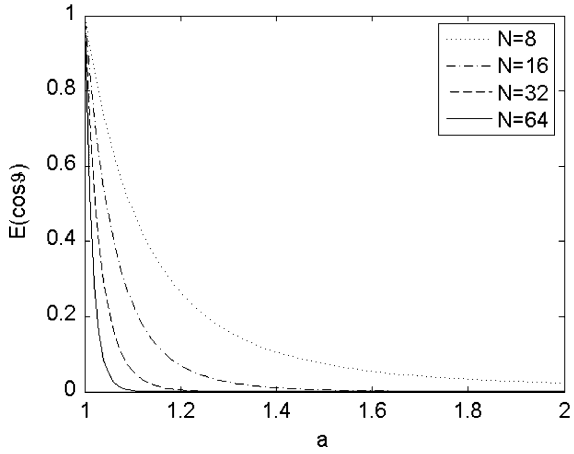

Fig. 3. Behavior of $E\ (\cos\theta)$ versus a for the exponential kernel.

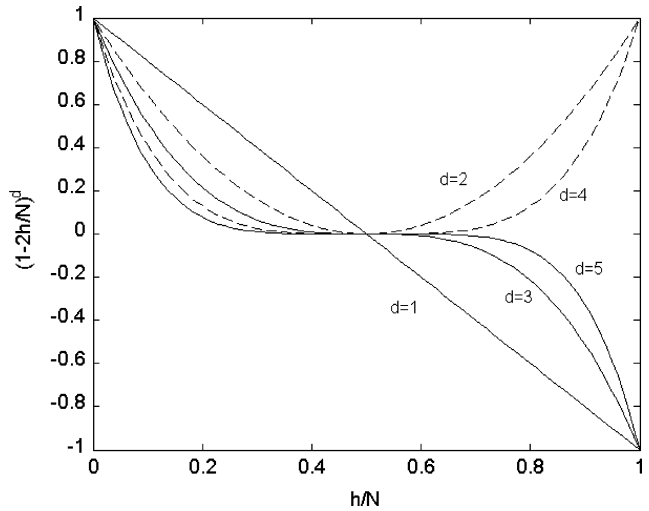

Fig. 4. Cosine of the angle in feature space for the polynomial kernel of degree$d$.

other hand, if $a < a_{\text{crit}}$, the RKAM will outperform the ECAM, as shown with some simulation results in Section VI. For the sake of comparison, let us consider [23, Fig. 4], where $N = 100$ and $\sigma^2 = 25$; the corresponding radix is $a = 1.083$ while $a_{\text{crit}} = 1.048$, hence the system is saturated and the results are the same of an ECAM.

It is worth noting that uniformly random binary patterns ($p = 0.5$) tend to be mutually orthogonal (the mode of the distribution corresponds to $h = N/2$). Hence, this assumption is especially fit to the ECAM. If the Hamming distance distribution deviates from (22), the improvement in recall performance due to SVM learning is even more evident.

V. Polynomial Kernel

In this section, we examine the RKAM behavior using the following polynomial kernel:

$$K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \right\rangle\right) = \left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)} \right\rangle + R\right)^d \tag{25}$$

with $R > 0$ and d positive integer. The dynamic equation becomes

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} \left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}(t) \right\rangle + R\right)^d\right), \quad i = 1, \ldots, N. \tag{26}$$

The orthogonalization property of polynomial kernel can be studied as in the exponential case. We have

$$\begin{aligned} K_{mm} &= (N+R)^d \\ K_{mn} &= \left(N - 2h\left(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right) + R\right)^d \\ \cos\theta_{mn} &= \left(1 - \frac{2h\left(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right)}{N+R}\right)^d. \end{aligned} \tag{27}$$

The behavior of (27) as function of the normalized Hamming distance h/N is shown in Fig. 4 for $R = 0$ and some values of d. We note that $\cos\theta_{mn}$ is symmetrical (d even) or antisymmetrical (d odd) with respect to $h = N/2$, where it is zero. The curve for $d = 1$ corresponds to the original angle between vectors in input space. We observe that polynomial kernels can orthogonalize the input patterns, except for vectors whose Hamming distance is close to 0 or N. The range of h values corresponding to orthogonal patterns in feature space increases with d.

For d even, the polynomial kernel is positive, hence we can evaluate the diagonal dominance condition computing the expected value of $\cos\theta$. Assuming uniformly random binary vectors ($p = 0.5$), the expectation of $\cos\theta$ is computed by

$$E(\cos\theta) = \sum_{h=0}^{N} P(h)\left(1 - \frac{2h}{N+R}\right)^d \tag{28}$$

where $P(h)$ is given by (22). The value $E\ (\cos\theta)$ depends on R and d. In particular, it is minimum for $R = 0$, since both $P(h)$ and $(1 - 2h/N)^d$ are symmetrical with respect to $N/2$, where $P(h)$ is maximum while $(1 - 2h/N)^d$ is zero. For $R > 0$, the null of the function (28) corresponds to $h = (N+R)/2$. For $d = 2$ and $R = 0$, the expected value (28) can be easily obtained in closed form (see the Appendix)

$$E(\cos\theta) = \frac{1}{N}. \tag{29}$$

For d odd, the polynomial kernel can assume positive as well as negative values, so the expected value of $\cos\theta$ is not meaningful to ascertain diagonal dominance of the kernel matrix. Indeed, for d odd, it is $E(\cos\theta) = 0$ for $R = 0$. In this case, we can compute the root mean square (rms) value of $\cos\theta$; as an example, for $d = 3$ and $R = 0$, it is $[E(\cos^2\theta)]^{(1/2)} < 0.01$ for $N > 50$.

From Fig. 4 and the previous discussion, we can expect a saturation of the RKAM for relatively small values of d. In this

case, the RKAM will be equivalent to the RCAM with dynamic equation

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} x_i^{(m)} \left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}(t)\right\rangle + R\right)^d\right), \quad i = 1, \ldots, N. \tag{30}$$

However, since the dynamic range of the nonlinear devices in Fig. 1 increases as N^d, we are mainly interested in the quadratic and cubic kernels; in these cases, there is still room for improvement that can be achieved by exploiting the Lagrange multipliers of the RKAM. The improvement is more evident if the Hamming distance distribution deviates from the binomial distribution.

The effect of R can be better understood by expanding (25) as follows [35]:

$$K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle\right) = \sum_{\ell=0}^{d} \binom{d}{\ell} R^{d-\ell} \left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle\right)^{\ell}. \tag{31}$$

For $R = 0$, the features are all the monomials of degree d; for $R > 0$, we have all the monomials of degree up to and including degree d, with relative weighting given by $R^{d-\ell}$. Using a value $0 < R \ll N$, the result of (28) slightly increases, i.e., the transformed patterns are less likely to be orthogonal. However, the dimensionality of the feature space increases from $\binom{N+d-1}{d}$ to $\binom{N+d}{d}$ [35].

This augmented dimensionality increases the chance of separability of patterns and could be exploited by SVM learning to improve the recall performance. In practice, a good choice for the RKAM is $R = 1$, corresponding to uniform weighting of the different features. It is worth noting that the increment in the dimensionality corresponding to $R > 0$ cannot be exploited in the RCAM, whose performance is worsened due to the reduction of orthogonality; in the case of uniformly random binary patterns, the best choice for the RCAM is $R = 0$.

The RKAM contains as many nodes as memory patterns (see the scheme in Fig. 1). Varying the number of stored items requires a reconfiguration of the hardware. Moreover, a device failure entails the complete loss of one of the stored vectors. Using the polynomial kernel, an alternate distributed implementation of the RKAM is possible. To this end, assume $d = 2$ and $R = 0$; expression (26) can be rewritten as

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} \sum_{j=1}^{N}\sum_{k=1}^{N} x_j(t) x_k(t) x_j^{(m)} x_k^{(m)}\right)$$

that is

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{j=1}^{N}\sum_{k=1}^{N} T_{ijk} x_j(t) x_k(t)\right), \quad i = 1, \ldots, N \tag{32}$$

where

$$T_{ijk} = \sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} x_j^{(m)} x_k^{(m)}. \tag{33}$$

Equation (32) represents a second-order Hopfield network with N neurons, whose weights are given by (33). It results in a distributed fault tolerant realization, where memory patterns are delocalized in the interneuron connection weights. A similar result can be easily extended to any degree d. The price we must pay is an increased implementation complexity with respect to the scheme in Fig. 1, since the number of connection weights is $o(N^{d+1})$ in place of $o(MN)$. A slightly different network corresponds to $d = 2$ and $R = 1$; according to (31), we have

$$\begin{aligned} K\left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle\right) &= \left(\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle + 1\right)^2 \\ &= \left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle^2 + 2\left\langle \mathbf{x}^{(m)}, \mathbf{x}^{(n)}\right\rangle + 1. \end{aligned} \tag{34}$$

Substituting (34) in (26) and grouping repeated and constant terms, we obtain the following equivalent expression for the dynamic rule of the RKAM:

$$x_i(t+1) = \operatorname{sgn}\left(\sum_{j=1}^{N} W_{ij} x_j(t) + \sum_{j=1}^{N-1}\sum_{k=j+1}^{N} T_{ijk} x_j(t) x_k(t) + I_i\right), \quad i = 1, \ldots, N \tag{35}$$

where

$$W_{ij} = \sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} x_j^{(m)} \tag{36a}$$

$$T_{ijk} = \sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)} x_j^{(m)} x_k^{(m)} \tag{36b}$$

$$I_i = \sum_{m=1}^{M} \alpha_i^{(m)} x_i^{(m)}. \tag{36c}$$

Equation (35) represents a Hopfield network with both first- and second-order interactions, a model widely investigated in the literature [9], [37]. Several algorithms exist to design such memories. The classical *sum of outer product* rule, also known as *Hebb's law*, corresponds to the following weights [9]:

$$W_{ij} = \sum_{m=1}^{M} x_i^{(m)} x_j^{(m)} \tag{37a}$$

$$T_{ijk} = \sum_{m=1}^{M} x_i^{(m)} x_j^{(m)} x_k^{(m)}. \tag{37b}$$

Note that weights (37) are simply obtained from (36) setting $\alpha_i^{(m)} = 1$, for every i and m. To summarize, the following are true:

TABLE I
ATTRACTION RADIUS WITH THE EXPONENTIAL KERNEL OF RADIX $a = 2^{B/N}$ $(N = 32)$

M	B=4		B=5		B=6		B=7	
	ECAM	*RKAM*	*ECAM*	*RKAM*	*ECAM*	*RKAM*	*ECAM*	*RKAM*
8	4	7	6	7	7	7	7	7
16	1	6	3	6	5	6	6	6
24	1	6	3	6	4	6	6	6
32	1	5	1	5	4	5	5	5
40	1	4	1	4	4	5	5	5
48	1	3	1	4	3	5	5	5
56	1	3	1	4	1	4	5	5
64	1	2	1	3	1	4	4	4

— the polynomial RCAM with $d = 2$ and $R = 1$ is equivalent to a Hopfield network with first- and second-order interactions and Hebbian connection weights;
— the polynomial RKAM with $d = 2$ and $R = 1$ is equivalent to a Hopfield network with first- and second-order interactions following the *generalized Hebb's law* given by (36).

Higher capacity and attractivity of stored patterns can be obtained designing model (35) by pseudoinverse techniques [37]. In this case, the equilibrium condition of pattern $\mathbf{x}^{(m)}$ is written as follows:

$$\left(\sum_{j=1}^{N} W_{ij} x_j^{(m)} + \sum_{j=1}^{N-1} \sum_{k=j+1}^{N} T_{ijk} x_j^{(m)} x_k^{(m)} + I_i\right) = x_i^{(m)}, \quad i = 1, \ldots, N. \quad (38)$$

The stability of the M memory vectors can be recast as a set of linear equations, with unknowns given by the weights and thresholds, that can be solved by the pseudoinverse method. Even if this formulation of the design problem resembles the SVM approach described in Section III [compare (7) and (38)], there is a major difference. The pseudoinverse method operates in input space and, therefore, computes $N[N+N(N-1)/2] = N^2(N+1)/2$ coefficients. The proposed SVM method operates in feature space; as a consequence, it requires $<MN$ Lagrange multipliers (some of them can be zero). The advantage in terms of computation time and storage requirements is evident, provided that $M < N^2/2$. Notwithstanding the consistent reduction of design parameters, the performance of the RKAM with $d = 2$ and $R = 1$ is the same as of the NN designed by the pseudoinverse method, as it will be shown in Section VI.

VI. EXPERIMENTAL RESULTS

In this section, we present several simulation results in order to illustrate the performance of the RKAM, and compare it to the ECAM, to higher order Hopfield networks, and to the basic Hopfield network. In all the experiments presented in the following, SVM training for RKAM design was carried out using SVM$^{\text{light}}$ [27] with a Matlab interface. The bias b is zero and $C = \infty$. We carried out some of the experiments also with $b \neq 0$ but the results are almost identical, so they are not reported here. In our tests, we compared synchronous and asynchronous dynamics, obtaining substantially equivalent performance with a slight advantage for the second one. Hence, all the simulation results presented in this section correspond to the asynchronous mode (one component at a time is updated), except for the test results on one-shot behavior (Example 3). The experiments are presented separately for the exponential kernel and the polynomial kernel.

A. Exponential Kernel

Example 1: We first examine the relation between RKAM and ECAM. For sake of comparison with [13, Fig. 4], we fix N at 32 and vary the dynamic range $D = a^N$ and the number M of memory vectors. We randomly choose ten sets of M 32-bit memory patterns with equal probability of $+1$ and -1. For each set of patterns, we design an RKAM, using SVM learning, and the corresponding ECAM. For each design, we consider 100 initial states randomly generated by flipping h bits. For each initial state, we simulate the system until convergence. The resulting steady-state value is compared with the corresponding initial state and the experiment is considered a success if they match. The number of successes out of 1000 runs is computed. If this number is greater than 998, we say that the memory can correct h errors. The maximum h for a fixed M is the attraction radius ρ. In Table I, we show the attraction radii for the RKAM and the ECAM with M ranging from 8 to 64 and dynamic range $D = 2^B$ for $B = 4, 5, 6, 7$. The corresponding values of a are 1.09, 1.114, 1.138, and 1.164 respectively. We observe that the RKAM clearly outperforms the ECAM for $B < 7$. For example, with $M = 32$ to obtain an attraction radius $\rho = 5$ with the RKAM $B = 4$ is enough, while with the ECAM, we need $B = 7$. The corresponding dynamic range is eight times larger. For $B = 7$, the attraction radius is equivalent; this result confirms formula (24) since $a_{\text{crit}} \cong 1.168$ corresponds to the saturation of the RKAM for $N = 32$. Simulations with $B > 7$ give the same behavior for both memories.

From Table I, we can ascertain the different storage capacity of the two models. To this end, we can define the capacity as the maximum number of random patterns that can be stored and recalled with a given attraction radius ρ. For example, assuming $\rho = 4$ and $B = 4$, we can see that the capacities of ECAM and RKAM are, respectively, $M = 8$ and $M = 40$. For $B = 6$, the capacities are, respectively, $M = 40$ and $M = 64$.

Then, we repeated the same tests using larger values of N. The results show that the improvement given by the RKAM with respect to the ECAM diminishes with N. This behavior is easily explained taking into account that random bipolar vectors tend to be mutually orthogonal as N increases; when this happens, the SVM overfits the data rendering the exponential RKAM equivalent to the ECAM. To observe a different recall performance between the two networks, we must increase also the value of M. For example, using $B = 4$ and $N = 64, 128 <$

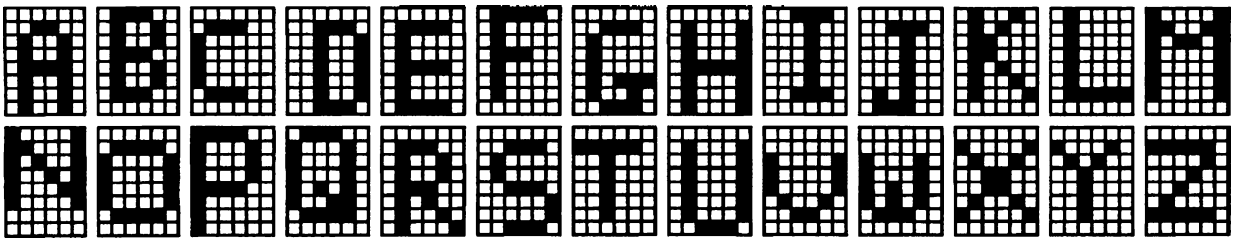

Fig. 5. The 26 patterns with $N = 48$ to be stored in the associative memory.

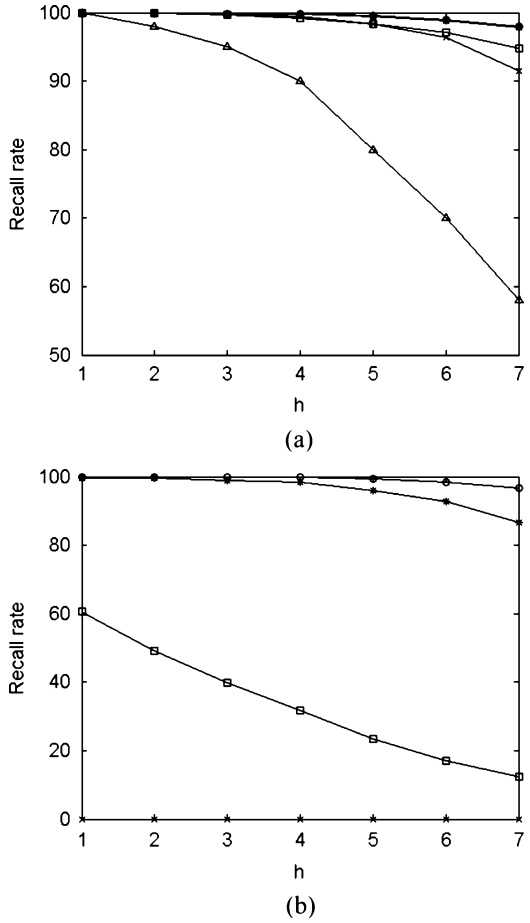

Fig. 6. Storage of the letters in Fig. 5. (a) Exponential RKAM. (b) ECAM. "o" $(B = 12)$, "$*$" $(B = 9)$, "$\square$" $(B = 6)$, "$\times$" $(B = 3)$, and "Δ" [6].

$M < 256$, the attraction radii of ECAM and RKAM are, respectively, 13 and 15 bits. The benefits of SVM training become more evident using correlated "real-life" patterns, as in Example 2.

Example 2: As a second test, we consider the storage of the 26 patterns with $N = 48$ shown in Fig. 5. Both the RKAM and the corresponding ECAM are designed to store the 26 letters and then simulated using 1000 random initial states for each memory pattern. The recall rate of the RKAM is shown in Fig. 6(a) versus the initial Hamming distance h, for $D = 2^B$ and $B = 3, 6, 9, 12$. The corresponding curves for the ECAM are shown in Fig. 6(b). The different performance of the two systems is evident [note the different scales in Fig. 6(a) and (b)]. The recall rate of the RKAM is slightly influenced by the value of B and remains good also for $B = 3$. The ECAM behavior is quite unsatisfactory for small dynamic ranges $(B = 3, 6)$ and almost comparable to the RKAM only for $B = 12$. Hence, in this example, the saturation of the RKAM begins with $B = 12$, corresponding to $a = 1.19$. This value is not in accordance with the value $a_{\text{crit}} \cong 1.1$, given by (24), due to the interpattern Hamming distance distribution which is different from the model assumed in the analysis. This is confirmed by the curves in Fig. 7: the expected value of $\cos\theta$ for the letters is greater than the expected value corresponding to a binomial distribution of the Hamming distance for $N = 48$. Finally, to compare the exponential RKAM with the basic Hopfield network, we included in Fig. 6(a) the curve in [6, Fig. 5(d)], concerning the same design example.

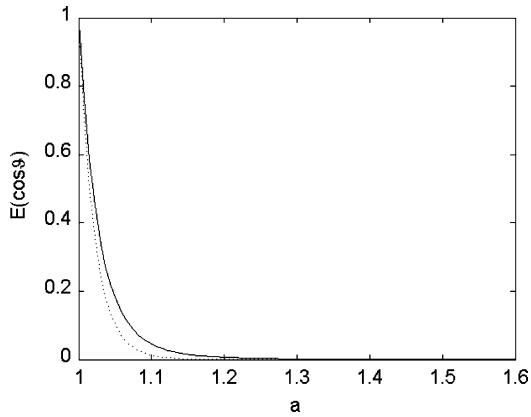

Fig. 7. Expected value of $\cos\theta$ for the 26 letters in Fig. 5 (solid line) and for 48-bit vectors following a binomial Hamming distance distribution with $p = 0.5$ (dashed line).

Example 3: Then, we investigate the effect of the recurrent dynamics on the recall performance of the exponential RKAM, evaluating its *one-shot* behavior. We randomly choose 10 sets of 32 memory patterns with $N = 32$.

Using these patterns, we design the RKAM using SVM learning. Then, we consider 100 input vectors randomly generated by flipping h bits. For each initial state, we compute the memory response by applying only once (5) to all the components in parallel. The same experiment is conducted using the ECAM. The percentage of correct recall is shown in Fig. 8 for $D = 2^B$ and $B = 4, 5, 7$. The importance of recursive updating is clear: the one-shot recall rate is always inferior to the recurrent case. Note also that the RKAM always

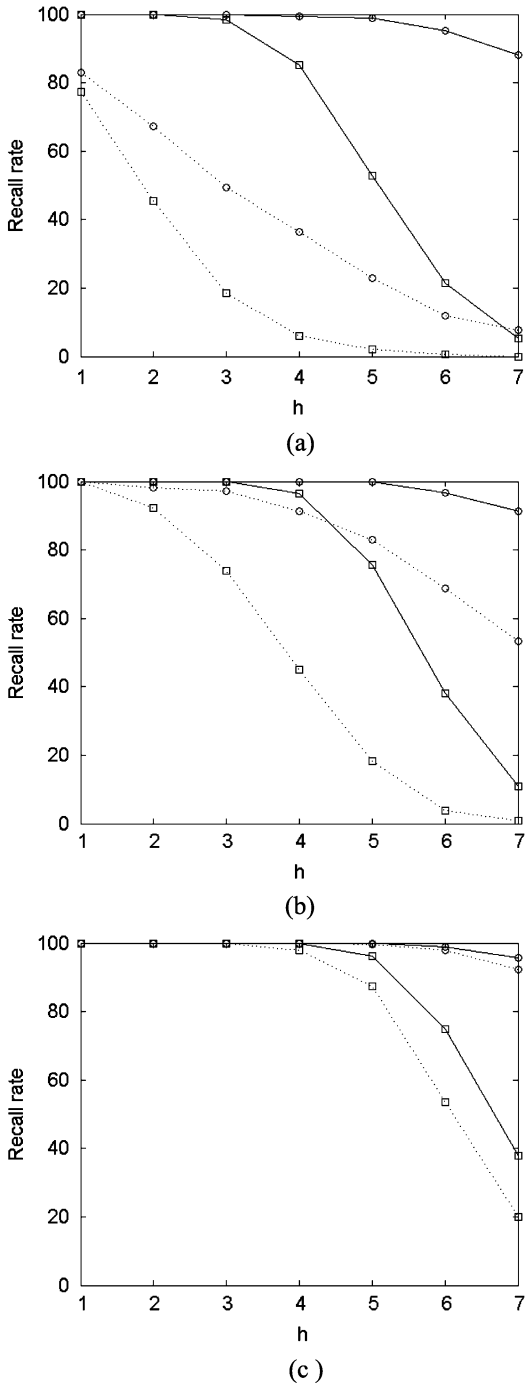

Fig. 8. Experiments on one-shot recall with the exponential kernel and $M = N = 32$. (a) $B = 4$. (b) $B = 5$. (c) $B = 7$. "o" recurrent, "□" one shot, "- - -" ECAM, and " —" RKAM.

outperforms the corresponding ECAM for the same type of updating.

B. Polynomial Kernel

Example 4: In this experiment, we compare the performance of polynomial RKAM and RCAM for $d = 2$ and $d = 3$. The experiment is conducted like Example 1, using ten sets of M 32-bit random patterns and 100 initial states for each memory. The attraction radius ρ is computed as explained previously considering 998 successes out of 1000 runs. The experiment was performed using both $R = 0$ and $R = 1$ obtaining the same results, shown in Table II. We observe that the RKAM outperforms the RCAM for $d = 2$. For $d = 3$, it is only slightly superior. This result is in accordance with the analysis in Section V.

TABLE II
ATTRACTION RADIUS WITH THE POLYNOMIAL KERNEL OF DEGREE d AND $R = 0$ OR 1 ($N = 32$)

M	d=2		d=3	
	RCAM	*RKAM*	*RCAM*	*RKAM*
8	6	7	7	7
16	4	5	6	6
24	2	4	6	6
32	1	3	6	6
40	1	3	5	5
48	1	3	4	4
56	1	3	3	4
64	1	2	3	4

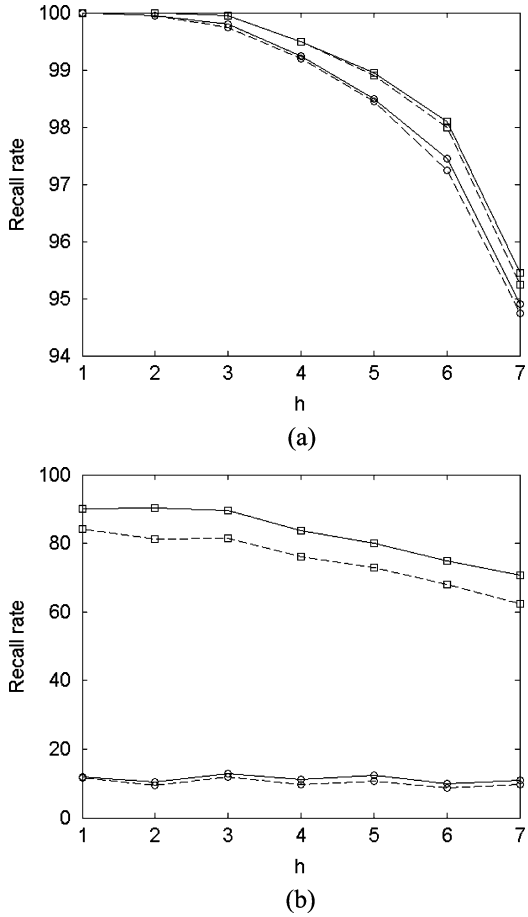

Fig. 9. Storage of the letters in Fig. 5. (a) Polynomial RKAM. (b) Polynomial RCAM. "o" ($d = 2$), "□" ($d = 3$); " —" $R = 0$, and "- - -" $R = 1$.

Example 5: The 26 letters in Fig. 5 have been stored also using the polynomial kernel. We compare the RKAM and the corresponding RCAM with $d = 2, 3$ and $R = 0, 1$. The recall rate curves are shown in Fig. 9. The SVM-based associative memory outperforms the simple RCAM both for $d = 2$ and $d = 3$.

The performance of the RKAM is almost the same for $R = 0$ and $R = 1$, while the RCAM performs better with $R = 0$. Note also the modest reduction of recall rate of the RKAM when

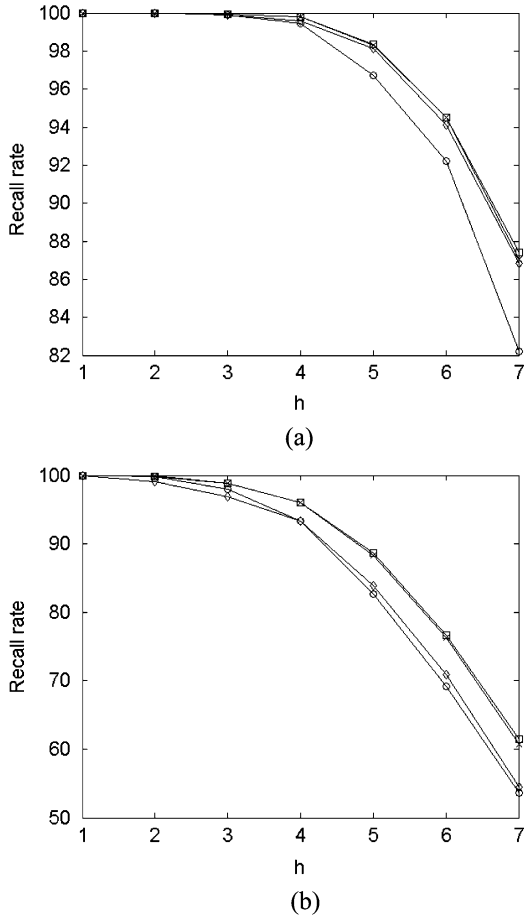

Fig. 10. Comparison between the quadratic RKAM and an Hopfield network with first- and second-order connections designed using the pseudoinverse method ($N = 32$). (a) $M = 32$. (b) $M = 64$. RKAM: "o" ($R = 0$), "□" ($R = 1$); "◇" ($R = 2$); PINV: "×."

decreasing the kernel order. This is due to the adaptive nature of the network which can compensate the reduced number of features by adjusting the Lagrange multipliers.

Example 6: In this experiment, we compare the polynomial RKAM with $d = 2$ and the Hopfield network with both first- and second-order connections, designed by the pseudoinverse method [37] (the thresholds I_i are set to zero as in [25]). The experiment is conducted using $N = 32$ and $M = 32, 64$. We randomly choose ten sets of M memory patterns with equal probability of $+1$ and -1. For each design, we consider 100 initial states randomly generated by flipping h bits. The recall rates are shown in Fig. 10. We observe that the best performance of the RKAM corresponds to $R = 1$, and it is equivalent to that of the NN, even if the RKAM uses at most 2048 coefficients (for $M = 64$) in place of 16 896.

Example 7: As a final test, in Fig. 11, we compare the quadratic RKAM with a first-order Hopfield network [the weights T_{ijk} in (35) are eliminated]. The experiment is conducted like Example 6 assuming $N = 32$ and $M = 16$. The connection weights and thresholds of the network have been computed using two different techniques: the pseudoinverse method and the linear SVM illustrated in [4]. In the second

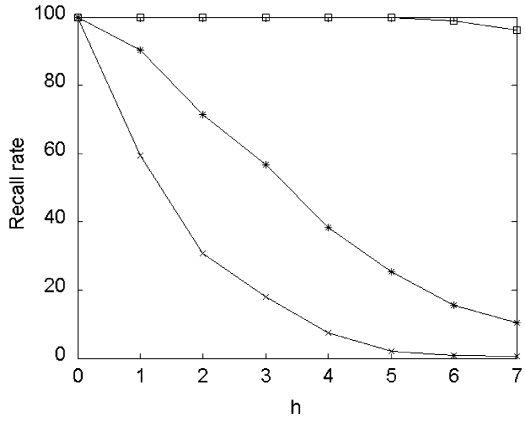

Fig. 11. Comparison between the quadratic RKAM ($R = 1$) and a first-order Hopfield network designed using the pseudoinverse method and the linear SVM in [4] ($N = 32, M = 16$). RKAM: "□," PINV: "×," linear SVM: "∗."

case, the self-connections are not allowed ($W_{ii} = 0$); as a consequence, in this example, to obtain feasible design problems, we generated random bipolar patterns with the condition that the minimum reciprocal Hamming distance is two (see [4], for details). The recall curves confirm the poor attractivity of the stored patterns in the first-order Hopfield network, as M becomes of the order of $N/2$. The corresponding curves for $M > 16$ are not shown since the first-order Hopfield network has even poorer performance.

VII. Conclusion

We studied a generalization of RCAMs, called RKAM, whose design is carried out by SVM learning. Two types of kernel have been taken into account: exponential and polynomial. We showed that they are generalizations of well-known associative memories, i.e., respectively, the ECAM and the Hopfield network with higher order Hebbian weights. The performance of the RKAM has been compared to these models through several simulation tests. The performance is equivalent when the kernel matrix of the underlying SVMs becomes diagonally dominant. On the other hand, when the SVMs generalize well, we obtain an improvement in recall performance for the same memory size and load. This can lead to a way of designing associative memories with good recall rate and capacity, but reduced dynamic range with respect to an ECAM. With respect to a high-order NN, we can use far less synaptic coefficients.

In this paper, we considered two widely used kernels because of their intriguing relation with known associative memory models. As a further development, it would be of interest to investigate if the kernel function can be tailored to the peculiarities of the memory patterns. In this case, a methodology for the selection of the best kernel for a particular data set should be developed. Further kernels with peculiar properties, like those proposed in [32] and [38], could be investigated in the context of associative memory design with the aim of reducing the recall time or to improve the noise suppression. Finally, the extension of RKAMs to multivalued or grayscale patterns [39], [40] would be of great interest. We hope that this paper

will stimulate further research leading to fruitful applications of kernel methods in associative memory design.

APPENDIX

In this Appendix, we derive expression (29) corresponding to the polynomial kernel of degree $d = 2$ with $R = 0$. Assuming random bits with equal probabilities ($p = 0.5$), it is

$$E(\cos\theta) = \sum_{h=0}^{N} \binom{N}{h} 2^{-N} \left(1 - \frac{2h}{N}\right)^2 = \sum_{h=0}^{N} \binom{N}{h} 2^{-N} \left(1 + 4\frac{h^2}{N^2} - 4\frac{h}{N}\right). \quad \text{(A1)}$$

Let us consider the three terms separately. First, we observe that

$$2^{-N} \sum_{h=0}^{N} \binom{N}{h} = 2^{-N} 2^N = 1.$$

The sum

$$\sum_{h=0}^{N} \binom{N}{h} 2^{-N} h^2$$

can be interpreted as the second moment of a binomial distribution with $p = 0.5$. Since this distribution has mean value $N/2$ and variance $N/4$, the second moment is $N/4 + (N/2)^2$, hence

$$\frac{4}{N^2} \sum_{h=0}^{N} \binom{N}{h} 2^{-N} h^2 = \frac{4}{N^2} \left(\frac{N}{4} + \frac{N^2}{4}\right) = \frac{1}{N} + 1.$$

The sum

$$\sum_{h=0}^{N} \binom{N}{h} 2^{-N} h$$

is the mean value of a binomial distribution with $p = 0.5$, that is $N/2$. Then

$$-\frac{4}{N} \sum_{h=0}^{N} \binom{N}{h} 2^{-N} h = -2.$$

Summing up the three terms in (A1), we have

$$E(\cos\theta) = 1 + \frac{1}{N} + 1 - 2 = \frac{1}{N}.$$

REFERENCES

[1] T. Kohonen, "Correlation matrix memories," *IEEE Trans. Comput.*, vol. C-21, pp. 353–359, Apr. 1972.

[2] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, pp. 2554–2558, Apr. 1982.

[3] D. J. Amit, H. Gutfreund, and H. Sompolinsky, "Spin glass models of neural networks," *Phys. Rev. A*, vol. 32, p. 1007, 1985.

[4] D. Casali, G. Costantini, R. Perfetti, and E. Ricci, "Associative memory design using support vector machines," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1165–1174, Sep. 2006.

[5] J.-Y. Chang and C.-C. Wu, "Design of Hopfield-type associative memory with maximal basin of attraction," *Electron. Lett.*, vol. 29, no. 24, pp. 2128–2130, 1993.

[6] D.-L. Lee and T. C. Chuang, "Designing asymmetric Hopfield-type associative memory with higher order Hamming stability," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1464–1476, Nov. 2005.

[7] M. K. Muezzinoglu, C. Guzelis, and J. M. Zurada, "An energy function-based design method for discrete Hopfield associative memory with attractive fixed points," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 370–378, Mar. 2005.

[8] C. L. Giles and T. Maxwell, "Learning, invariance and generalization in high-order neural networks," *Appl. Opt.*, vol. 26, no. 23, pp. 4972–4978, 1987.

[9] D. Psaltis, C. H. Park, and J. Hong, "Higher order associative memories and their optical implementations," *Neural Netw.*, vol. 1, pp. 149–163, 1988.

[10] J. Si and A. N. Michel, "Analysis and synthesis of a class of discrete-time neural networks with nonlinear interconnections," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 41, no. 1, pp. 52–58, Jan. 1994.

[11] P. Simpson, "Higher-ordered and intraconnected bidirectional associative memories," *IEEE Trans. Syst. Man Cybern.*, vol. 20, no. 3, pp. 637–653, May/Jun. 1990.

[12] R. J. Marks, L. E. Atlas, J. J. Choi, S. Oh, K. F. Cheung, and D. C. Park, "Performance analysis of associative memories with nonlinearities in the correlation domain," *Appl. Opt.*, vol. 27, no. 14, pp. 2900–2904, 1988.

[13] T.-D. Chiueh and R. M. Goodman, "Recurrent correlation associative memories," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 275–284, Mar. 1991.

[14] T.-D. Chiueh and H.-K. Tsai, "Multivalued associative memories based on recurrent networks," *IEEE Trans. Neural Netw.*, vol. 4, no. 2, pp. 364–366, Mar. 1993.

[15] E. R. Hancock and M. Pelillo, "A Bayesian interpretation for the exponential correlation associative memory," *Pattern Recognit. Lett.*, vol. 19, pp. 149–159, 1998.

[16] M. Hassoun and P. Watta, "The Hamming associative memory and its relation to the exponential capacity DAM," in *Proc. IEEE Int. Conf. Neural Netw.*, Washington, DC, Jun. 1996, vol. I, pp. 583–587.

[17] R. C. Wilson and E. R. Hancock, "Storage capacity of the exponential correlation associative memory," *Neural Process. Lett.*, vol. 13, pp. 71–80, 2001.

[18] R. C. Wilson and E. R. Hancock, "A study of pattern recovery in recurrent correlation associative memories," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 506–519, May 2003.

[19] B. Caputo and H. Niemann, "Storage capacity of kernel associative memories," in *Proc. Int. Conf. Artif. Neural Newt.*, J. Dorronsoro, Ed., Madrid, Spain, 2002, pp. 51–56.

[20] B. Caputo and H. Niemann, "From Markov random fields to associative memories and back: Spin glass Markov random fields," presented at the IEEE Workshop Statist. Comput. Theories Vis., Vancouver, BC, Canada, Jul. 2001.

[21] B.-L. Zhang, H. Zhang, and S. S. Ge, "Face recognition by applying wavelet subband representation and kernel associative memory," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 166–177, Jan. 2005.

[22] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An Introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[23] C. Garcia and J. A. Moreno, "The Hopfield associative memory network: Improving performance with the kernel "trick"," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2004, vol. 3315, pp. 871–880.

[24] J. Shawe-Taylor and N. Cristianini, *An Introduction to Support Vector Machines*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

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

[26] T. Poggio, S. Mukherjee, A. Rifkin, A. Rakhlin, and A. Verri, "b," Massachusetts Inst. Technol., Cambridge, MA, MIT Tech. Rep., Jul. 2001.

[27] T. Joachims, "Making large scale SVM learning practical," in *Advances in Kernel Methods-Support Vector Learning*, B. Scholkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 169–184 [Online]. Available: http://svmlight.joachims.org/

[28] C.-C. Chang and C.-J. Lin, "LIBSVM—A library for support vector machines," [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[29] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods—Support Vector Learning*, B. Scholkopf, C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 185–208.
[30] V. Kecman, M. Vogt, and T. M. Huang, "On the equality of kernel adatron and sequential minimal optimization in classification and regression tasks and alike algorithms for kernel machines," in *Proc. Eur. Symp. Artif. Neural Netw.*, Bruges, Belgium, Apr. 2003, pp. 215–222.
[31] R. Genov and G. Cauwenberghs, "Kerneltron: Support vector machine in silicon," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 1426–1434, 2003.
[32] D. Anguita, S. Pischiutta, S. Ridella, and D. Sterpi, "Feedforward support vector machine without multipliers," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1328–1331, Sep. 2006.
[33] B. Scholkopf, J. Weston, E. Eskin, C. Leslie, and W. S. Noble, "A kernel approach for learning almost orthogonal patterns," in *Lecture Notes in Computer Science*. Berlin, Germany: Spriner-Verlag, 2002, vol. 2430/2431.
[34] D. Greene and P. Cunningham, "Practical solutions to the problem of diagonal dominance in kernel document clustering," Comput. Sci. Dept., Trinity College, Dublin, Ireland, Tech. Rep., 2006.
[35] J. Shawe-Taylor and N. Cristianini, *Kernel-Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[36] E. Eskin and E. Agichtein, "Combining text mining and sequence analysis to discover protein functional regions," in *Proc. Pacif. Symp. Biocomput.*, 2004, vol. 9, pp. 288–299.
[37] L. Personnaz, I. Guyon, and G. Dreyfus, "High-order neural networks: Information storage without errors," *Europhys. Lett.*, vol. 4, no. 8, pp. 863–867, 1987.
[38] P. Williams, S. Li, J. Feng, and S. Wu, "A geometrical method to improve performance of the support vector machine," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 942–947, May 2007.
[39] G. Costantini, D. Casali, and R. Perfetti, "Associative memory design for 256 gray-level images using a multilayer neural network," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 519–522, Mar. 2006.
[40] D.-L. Lee, "Improvements of Hopfield complex-valued associative memory by using generalized projection rules," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1341–1347, Sep. 2006.

Renzo Perfetti was born in Italy in 1958. He received the laurea degree with honors in electronics engineering from the University of Ancona, Italy, in 1982, and the Ph.D. degree in information and communication engineering from the University of Rome "La Sapienza," Rome, Italy, in 1992.

From 1983 to 1987, he was with the radar division of Selenia, Rome, where he worked on radar systems design and simulation. From 1987 to 1992, he was with the radiocommunication division of Fondazione U. Bordoni in Rome. In 1992, he joined the Department of Electronic and Information Engineering, University of Perugia, Perugia, Italy, where he is currently a Full Professor of electrical engineering. His research interests include artificial neural networks, pattern recognition, machine learning, and digital signal processing.

Elisa Ricci received the M.S. degree in electronics engineering from University of Perugia, Perugia, Italy, in 2003, where she is currently working towards the Ph.D. degree at the Department of Electronic and Information Engineering.

Her research interests include machine learning, neural networks, and image processing.