Creating Perceptual Features Using a BAM-inspired Architecture

Gyslain Giguère (giguere.gyslain@courrier.uqam.ca)
UQÀM, Département de psychologie, A/S LEINA,
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

Sylvain Chartier (chartier.sylvain@gmail.com)
Université d'Ottawa, Département de psychologie,
550 Cumberland, Ottawa, Ont, K1N 6N5

Robert Proulx (proulx.robert@uqam.ca)
UQÀM, Département de psychologie, A/S LEINA,
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

Jean-Marc Lina (jean-marc.lina@etsmtl.ca)
École de Technologie Supérieure, Département de génie électrique
C.P. 8888, Succ. CV, Montréal, Qc, H3C 3P8

Abstract

In this paper, it shown that the Feature-Extracting Bidirectional Associative Memory (FEBAM) can create its own set of perceptual features. Using a bidirectional associative memory (BAM)-inspired architecture, FEBAM inherits properties such as attractor-like behavior and successful processing of noisy inputs, while being able to achieve principal component analysis (PCA) tasks such as feature extraction. The model is tested by simulating prototype development in a noisy environment. Simulations show that the model fares particularly well compared to current neural PCA and independent component analysis (ICA) algorithms. Therefore, it is argued that the model possesses more cognitive explanative power than any other PCA/ICA algorithm or BAMs taken separately.

Keywords: Perceptual feature creation; Neural networks; Bidirectional associative memory; PCA; ICA.

Introduction

Cognitive psychologists have historically taken for granted that the human cognitive system uses sets of properties (or dimensions) to achieve perceptual and cognitive tasks (Garner, 1974). While most researchers view these properties and dimensions as symbolic (see Murphy, 2002), some have argued for *perceptual* symbol or feature systems, in which properties would be defined in an abstract iconic fashion and self-created in major part by associative bottom-up processes (Harnad, 1990; Schyns, Goldstone & Thibaut, 1998). To achieve this "perceptual feature creation" scheme, humans are sometimes hypothesized to implicitly use an information reduction strategy, in order to form cognitive representations that can be processed more efficiently. While the idea of a system creating its own "atomic" perceptual representations has been argued to be crucial to understanding many cognitive processes (Schyns, Goldstone & Thibaut, 1998), few efforts have been made to model perceptual feature creation processes in humans.

When faced with the task of modeling this strategy, one statistical solution is to use a principal component analysis (PCA) approach (Abdi, Valentin & Edelman, 1998). PCA neural networks (Diamantaras & Kung, 1996) can deal quite effectively with input variability by creating (or "extracting") statistical low-level features representing the data's intrinsic information to a certain degree. Unfortunately, these networks, being feedforward by definition, cannot replicate attractor-like (or "categorical" behavior). Also, the absence of explicitly depicted connections in the model forces supervised learning to be done using an external teacher. Moreover, some PCA models cannot even perform online or local learning (for an extensive review, see Diamantaras & Kung, 1996).

A class of neural networks known as recurrent autoassociative memory (RAM) models does respond to these requirements. One characteristic of RAMs is the use of a feedback loop, which allows for generalization to new patterns, noise filtering and pattern completion, among other uses (Hopfield, 1982). Feedback enables a given network to shift progressively from an initial pattern towards an invariable state, namely, an attractor. Since the information is distributed among the units, the network's attractors are global. If the model is properly trained, the attractors should then correspond to the learned patterns (Hopfield, 1982).

Direct generalization of RAM models is the development of bidirectional associative memory (BAM) models (Kosko, 1988). BAMs can associate any two data vectors of equal or different lengths (representing for example, a visual input and a prototype or category). These networks possess the advantage of being both autoassociative and heteroassociative memories (Karhunen, Pajunen & Oja, 1998), and therefore encompass both unsupervised and supervised learning.

A problem with RAM/BAM models is that contrarily to what is found in real-world situations, they store information using noise-free versions (prototypes) of the input patterns. In comparison, to overcome the possibly

infinite number of stimuli (exemplars) stemming, for instance, from multiple perceptual events, humans must regroup these unique noisy inputs into a finite number of stimuli or prototypes. This type of learning in the presence of noise should therefore be taken into account by RAM/BAM models.

Unfortunately, few working solutions have been proposed to overcome this situation. The most popular solution consists in using an *unlearning* procedure (e.g. Christos, 1996; Hopfield, Feinstein & Palmer, 1983; Yen & Michel, 1996). The rationale is that if a given noisy input is composed of a stimulus (x) and some noise (+n), then by "unlearning" the noise (-n) we should only retrieve the stimulus. This procedure results in good performances, but the noise proportion must be known beforehand in order to set the ratio of learning/unlearning trials accordingly. Moreover, other parameter values must also be determined *a priori*, and the procedure implies the inversion of learning parameters, which renders the model quite inconsistent.

To sum up, on one hand, PCA models can deal with noise but do not encompass the properties of attractor networks. On the other hand, BAM models exhibit attractor-like behavior but have difficulty dealing with noise without the addition of free parameters, which must be tuned *a priori* in order to accomplish the task.

Recently, Chartier, Giguère, Renaud, Lina & Proulx (in press) have shown that the Feature-Extracting Bidirectional Associative Memory (FEBAM) is an adequate candidate to unify both classes of networks under the same framework. They showed that FEBAM can perform competitively in image reconstruction and blind source separation tasks, tasks that are exclusively associated with PCA/ICA networks. However, the model's ability to conciliate feature extraction with attractor behavior has never been tested.

In the present study, it will be shown that FEBAM can extract features and develop prototypes in a noisy environment while preserving its attractor properties. As previously stated, this "noisy learning" task is a challenge for RAM/BAM networks. FEBAM has already been shown to exhibit bipolar and continuous-valued attractor-like behavior, supervised and unsupervised learning, as well as feature extraction. Because these various behaviors would usually be achieved by separate models, it is believed that FEBAM possesses a clear advantage in explanatory power for perceptual and cognitive modeling over its predecessors.

Theoretical background

Principal component analysis and neural nets

PCA neural networks (Figure 1) are based on a linear output function defined by:

$$\mathbf{y} = \mathbf{W}\mathbf{x} \tag{1}$$

where $\mathbf{x}$ is the original input vector, $\mathbf{W}$ is the stabilized weight matrix, and $\mathbf{y}$ is the output. When the weight matrix maximizes preservation of relevant dimensions, then the reverse operation:

$$\hat{\mathbf{x}} = \mathbf{W}^{\mathrm{T}}\mathbf{y} \tag{2}$$

should produce a result that is close to the original input. Thus, the norm of the difference vector $\|\mathbf{x} - \hat{\mathbf{x}}\|$ should be minimized. This translates by finding a $\mathbf{W}$ which minimizes:

$$E = \sum_{i=1}^{N} \left\| \mathbf{x}_i - \hat{\mathbf{x}}_i \right\|^2 = \sum_{i=1}^{N} \left\| \mathbf{x}_i - \mathbf{W}^T \mathbf{W} \mathbf{x}_i \right\|^2 \tag{3}$$

where E is the error function to minimize, and N is the input's dimension. Several solutions exist for Equation 3 (for a review, see Diamantaras & Kung, 1996).

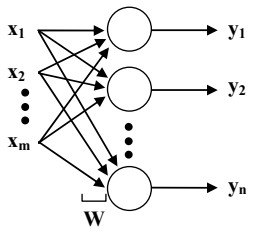


Figure 1: General neural PCA architecture.

Generalization to Nonlinear PCA (Karhunen, Pajunen & Oja, 1998) is straightforward; in this case, a nonlinear output function is to be used:

$$\mathbf{y} = f\left(\mathbf{W}\mathbf{x}\right) \tag{4}$$

The corresponding minimization function is thus given by

$$E = \sum_{i=1}^{N} \left\| \mathbf{x}_i - \mathbf{W}^T f\left(\mathbf{W}\mathbf{x}_i\right) \right\|^2 \tag{5}$$

Depending on the type of nonlinear output function, different types of optimization can be achieved. If the output function uses cubic or higher nonlinearity, then the network directly relates to independent component analysis (ICA; Hyvarinen & Oja, 2000).

Bidirectional associative memories

In Kosko's original BAM, learning is accomplished using a simple Hebbian rule, according to the following equation:

$$\mathbf{W} = \mathbf{Y}\mathbf{X}^{\mathrm{T}} \tag{6}$$

In this expression, $\mathbf{X}$ and $\mathbf{Y}$ are matrices representing the sets of bipolar pairs to be associated, and $\mathbf{W}$ is the weight matrix which, in fact, corresponds to a correlation measure between the input and output. The model uses a recurrent nonlinear output function in order to allow the network to filter out the different patterns during recall. The nonlinear output function typically used to recall noisy patterns in BAM networks is expressed by the following equations:

$$\mathbf{y}(t+1) = sgn(\mathbf{W}\mathbf{x}(t)) \quad (7)$$

and

$$\mathbf{x}(t+1) = sgn(\mathbf{W}^{\mathrm{T}}\mathbf{y}(t)) \quad (8)$$

where $\mathbf{y}(t)$ and $\mathbf{x}(t)$ represent the vectors to be associated at time t, and *sgn* is the signum function defined by

$$\mathrm{sgn}(z) = \begin{cases} 1 \text{ if } z>0 \\ 0 \text{ if } z=0 \\ -1 \text{ if } z<0 \end{cases} \quad (9)$$

Many enhancements, which increase the BAM's modeling capacities and range of applications, have been proposed over the years; these allow, for instance, for learning convergence, online learning, progressive recall, and processing of multi-valued stimuli in addition to bipolar (binary) stimuli (Chartier & Boukadoum, 2006a; Costantini, Casali & Perfetti, 2003; Wang, Hwang & Lee, 1996; Zhang & Chen, 2003).

Model description

Architecture

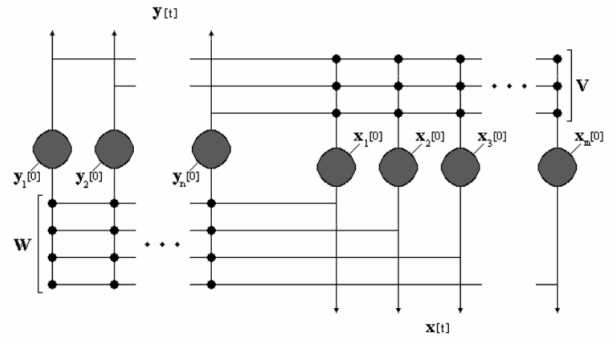


Figure 2: Architecture of the original BAM.

FEBAM's architecture is based on the BAM architecture proposed by Hassoun (1989) and Chartier & Boukadoum (2006a) (Figure 2). It consists of two Hopfield-like neural networks interconnected in head-to-toe fashion (Hassoun, 1989). When connected, these networks allow a recurrent flow of information that is processed bidirectionally. As shown in Figure 2, the **W** layer returns information to the **V** layer and vice versa. As in a standard BAM, each layer serves as a teacher for the other layer, and the connections are explicitly depicted in the model[1]. However, to enable a BAM to perform PCA, one set of those explicit connections must be removed (Figure 3). Thus, in contrast with the standard BAM architecture, $\mathbf{y}(0)$, the *initial output* is not obtained externally, but is instead acquired by iterating once through the network as illustrated in Figure 4.

In the context of feature extraction, the **W** layer is used to extract features whose dimensionality is determined by the number of **y** units; the more units in that layer, the less compression occurs. If the **y** layer's dimensionality is greater or equal to that of the **x** layer, then no compression occurs, and the network behaves as autoassociative memory[2].

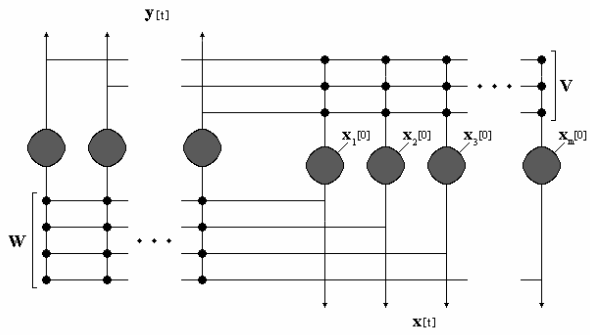


Figure 3: FEBAM network architecture.

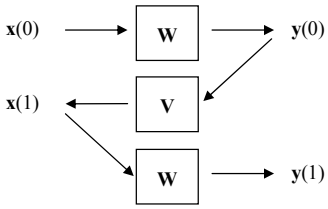


Figure 4: Output iterative process used for learning updates.

Output function

The output function is expressed by the following equations:

$$\forall\ i, ..., N, \mathbf{y}_i(t+1) = \begin{cases} 1, & \text{If } \mathbf{W}\mathbf{x}_i(t) > 1 \\ -1, & \text{If } \mathbf{W}\mathbf{x}_i(t) < -1 \\ (\delta+1)\mathbf{W}\mathbf{x}_i(t) - \delta(\mathbf{W}\mathbf{x})_i^3(t), & \textit{Else} \end{cases} \quad (10)$$

and

$$\forall\ i, ..., M, \mathbf{x}_i(t+1) = \begin{cases} 1, & \text{If } \mathbf{V}\mathbf{y}_i(t) > 1 \\ -1, & \text{If } \mathbf{V}\mathbf{y}_i(t) < -1 \\ (\delta+1)\mathbf{V}\mathbf{y}_i(t) - \delta(\mathbf{V}\mathbf{y})_i^3(t), & \textit{Else} \end{cases} \quad (11)$$

where N and M are the number of units in each layer, i is the index of the respective vector element, $\mathbf{y}(t+1)$ and $\mathbf{x}(t+1)$ represent the network outputs at time $t+1$, and δ is a general output parameter that should be fixed at a value lower than 0.5 to assure fixed-point behavior (Chartier & Proulx, 2005). Figure 5 illustrates the shape of the output function when $\delta = 0.4$.

[1] In opposition, the architecture of multi-layer Perceptrons strictly illustrates a series of input-output relationships, without ever specifying the origin of the "teacher"'s information. This is less desirable in a biopsychological perspective.

[2] Contrarily to deep belief networks (Hinton & Salakhutdinov, 2006), the model possesses no bias units, no backpropagation training algorithm, no special parameters for learning and uses no stochastic processes.

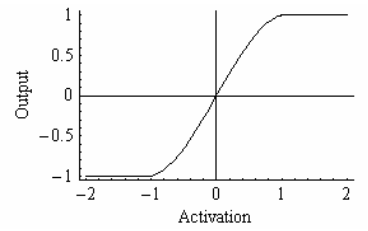


Figure 5: Output function for $\delta = 0.4$.

In contrast to a sigmoid output function, FEBAM's output function shows no asymptote at the ±1 values. This output function possesses the advantage of exhibiting continuous-valued (or gray-level) attractor behavior (for a detailed example see Chartier & Boukadoum, 2006a). Such properties contrast with networks using a standard nonlinear output function, which can only exhibit bipolar attractor behavior (e.g. Kosko, 1988).

Learning function

Learning is based on time-difference Hebbian association (e.g. Chartier & Proulx, 2005; Kosko, 1990; Sutton, 1988), and is formally expressed by the following equations:

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta(\mathbf{y}(0) - \mathbf{y}(t))(\mathbf{x}(0) + \mathbf{x}(t))^{\mathrm{T}} \quad (12)$$

and

$$\mathbf{V}(k+1) = \mathbf{V}(k) + \eta(\mathbf{x}(0) - \mathbf{x}(t))(\mathbf{y}(0) + \mathbf{y}(t))^{\mathrm{T}} \quad (13)$$

where η represents a learning parameter, $\mathbf{y}(0)$ and $\mathbf{x}(0)$, the initial patterns at $t = 0$, $\mathbf{y}(t)$ and $\mathbf{x}(t)$, the state vectors after t iterations through the network, and k the learning trial. The learning rule is thus very simple, and can be shown to constitute a generalization of hebbian/anti-hebbian correlation in its autoassociative memory version (Chartier & Proulx, 2005).

For weight convergence to occur, η must be set according to the following condition (Chartier & Proulx, 2005):

$$\eta < \frac{1}{2(1-2\delta)Max[N,M]}, \quad \delta \neq 1/2 \quad (14)$$

Equations 12 and 13 show that the weights can only converge when "feedback" is identical to the initial inputs (that is, $\mathbf{y}(t) = \mathbf{y}(0)$ and $\mathbf{x}(t) = \mathbf{x}(0)$). The function therefore correlates directly with network outputs, instead of activations. As a result, the learning rule is dynamically linked to the network's output (unlike most BAMs).

Simulation

Prototype development in a noisy environment

The simulation aimed at evaluating the network's capacity to acquire multiple prototype representations by extracting perceptual features from a set of noisy input patterns.

Methodology The learning stimuli were 26 binary-valued 7x7 images representing lowercase versions of letters of the alphabet (Figure 10a). For each letter, white pixels were given a value of -1, and black pixels a value of 1. Before a given letter was presented to the network, a random vector was added to it. This vector was normally distributed with a mean of 0 and standard deviation of 0.2. Hence, the network never saw exactly the same letter twice during learning. Figure 6 illustrates some noisy examples of the letter "a". Although the noise amount is small, it is sufficient to make the weight connections grow to infinity in standard RAM/BAM networks (Bégin & Proulx, 1996).

Figure 6: Different examples of the "a" letter input after a random vector N~(0;0.2) is added.

For learning, weights were randomly initialized with values ranging from -1 to 1. The learning and output parameters were respectively set to $\eta = 0.005$ and $\delta = 0.1$. Learning followed this general procedure:

1- Random selection of an input vector;
2- Iteration (one cycle, as illustrated in Figure 4) through the network using the output function (Equations 10 and 11);
3- Weight updates (Equations 12 and 13);
4- Repetition of 1 until the desired number of learning trials has been completed or the squared error between $\mathbf{y}(0)$ and $\mathbf{y}(t)$ is sufficiently small.

FEBAM's performance was compared with that of a neural linear PCA model (Diamantaras & Kung, 1996), a neural nonlinear PCA model (Karhunen, Pajunen, & Oja, 1998) and an ICA model (Hyvarinen & Oja, 2000). Finally, the network's generalization ability was tested by generating new noisy stimuli and verifying if the network's ability to correctly recall the associated prototypes.

Results Figure 7 illustrates the various prototypes developed in function of the number of features extracted. If the network is limited to 10 features (out of a 49-dimension stimuli), then it cannot correctly abstract the noise-free versions. If there are 26 abstracted features, then the network closely matches the noise-free inputs. Although theoretically this number should be sufficient to accomplish the task, the network is not able to attain perfection, since different weight combinations may converge to the same solution. Thus, by allowing redundancy in the results, the network needs to extract more features. For example, if the network is allowed to develop 40 features, then it is able to

correctly abstract the 26 noise-free inputs[3]. In this last situation, after 2000 trials, the squared error (Figure 8) is stabilized to an average of 0.38 (The error cannot be brought lower because of stimulus variability).

Using the weights, detection feature maps ($DV_j = \sum_{i=1}^{5}(\mathbf{W}_i \mathbf{V}_{ij})$ and $DW_{ij} = \mathbf{W}^{\mathrm{T}}{}_{ij} \mathbf{V}_{ij}$) were created to illustrate what each unit in each layer is tuned for (Figure 9).

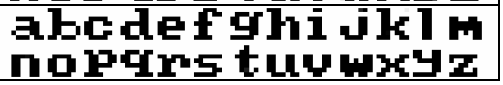
Number of features	Prototypes developed
10	
26	
40	

Figure 7: Prototypes abstracted during learning in function of the number of features.

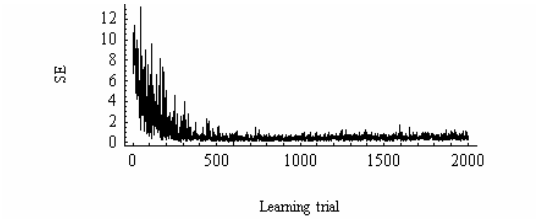


Figure 8: Squared error in function of the number of learning trials.

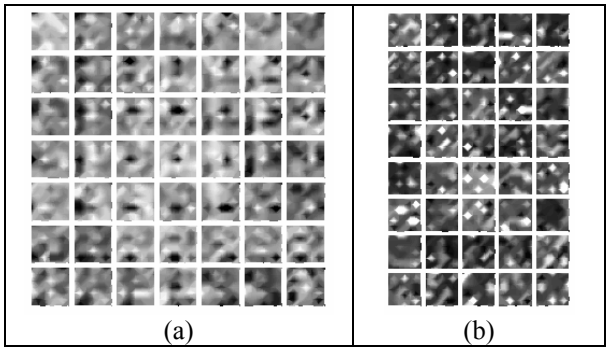


Figure 9: Feature detection performed by each of (a) the 49 units of layer **V** and (b) the 40 units of layer **W**.

Figure 9a illustrates what the **V** layer is tuned for. Seemingly, the fuzzy black dots show that this layer responds in function of the position of spatially salient traits. The **W** layer (Figure 9b) represents the 40 different features developed by the network. By combining these features, the network is able to reconstruct the prototypes correctly. Using a basis of 40 features, FEBAM was compared with different models that also extracted 40 features (Figure 10). Clearly, FEBAM is the only model that was able to reconstruct the noise-free inputs using 40 extracted features. In addition, following learning, FEBAM was also able to correctly perform a heteroassociation task (Kosko, 1988), where each lowercase letter was to be associated with its uppercase counterpart (Figure 10f), using the BAM architecture illustrated in Figure 2. Figure 11 shows an example of the noisy letters "b" and "c" that are progressively iterated through the network until convergence at the associated attractor (letter). Consequently, FEBAM still preserves its attractor properties in addition to some PCA properties.

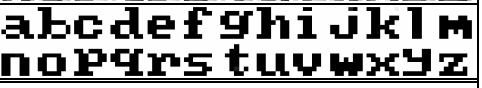
(a)	Noise-free inputs	
(b)	APEX	
(c)	NPCA	
(d)	fastICA	
(e)	FEBAM	
(f)	Hetero-association	

Figure 10: (a) Gaussian noise was added to each of these original images for learning purposes. (b-e) Simulation results using 40 extracted features. For each model, prototypes abstracted during learning are shown. Results for: (b) APEX, a linear PCA neural network; (c) a nonlinear PCA network; (d) fastICA, an ICA algorithm; (e) FEBAM, a nonlinear PCA-BAM network. (f) Results from an heteroassociation simulation using FEBAM.

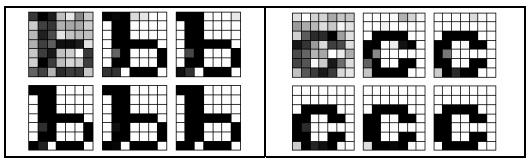


Figure 11: Examples of noisy recall. (a) For each letter, the first line represents iterations 1 to 3, and the second line represents iterations 4 to 6. As seen, from iteration to iteration, the outputs converge to the original input patterns.

Discussion

In this paper, it was shown that the BAM introduced by

[3] Of course, because this is not a data compression task, speaking of computational savings becomes irrelevant.

Chartier & Boukadoum (2006a) uses a learning function that is closely related to nonlinear PCA networks. Consequently, by simply removing a set of external connections from the original BAM architecture, FEBAM is able to perform feature extraction and reduction of the input dimensionality, just like PCA networks do. Because the **W** and **V** layers are connected, it is possible to observe what the developed attractors in the **V** layer look like. Leading to attractor-like behavior, the network can thus be used to perform prototype development from exemplar presentations.

Contrarily to the other models presented in this paper, FEBAM, being a special case of BAM, can also be used to simulate other applications such as categorization (Chartier & Proulx, 2005), classification (Chartier & Boukadoum, 2006a), many-to-one association and multi-step pattern recognition (Chartier & Boukadoum, 2006b). Other simulations (Chartier, Giguère, Renaud, Lina & Proulx, in press) have also shown that the network can achieve feature extraction and dimensionality reduction from gray-scale images, as well as blind source separation of noisy signal mixtures. Consequently, in addition to being a superior candidate for modeling human perceptual feature creation, the model possesses more cognitive explanative power than any other nonlinear/linear PCA or ICA algorithm.

References

Abdi, H., Valentin, D., Edelman, B.G. (1998). Eigenfeatures as intermediate-level representations: The case for PCA models. *Brain and Behavioral Sciences, 21,* 17-18.

Bégin, J., Proulx, R. (1996). Categorization in unsupervised neural networks: the Eidos model. *IEEE Transactions on Neural Networks, 7*(1), 147-154.

Cardoso, J.F. (1998). Blind signal separation: statistical principles. Proceedings of the IEEE, 86, 2009-2025.

Chartier, S., Boukadoum, M. (2006a). A bidirectional heteroassociative memory for binary and grey-level patterns. *IEEE Transactions on Neural Networks, 17,* 385-396.

Chartier, S., Boukadoum, M. (2006b). A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association. IEEE Transactions on Neural Networks, 17, 59-68.

Chartier, S., Giguère, G., Renaud, P., Lina, J.M., Proulx, R. (2007, *in press*). FEBAM : a feature-extracting bidirectional associative memory. *Proceedings of the 20th International Joint Conference on Neural Networks.*

Chartier, S., Proulx, R. (2005). NDRAM: nonlinear dynamic recurrent associative memory for learning bipolar and nonbipolar correlated patterns. *IEEE Transactions on Neural Networks, 16,* 1393-1400.

Christos, G. A. (1996). Investigation of the Crik-Mitchison reverse-learning dream sleep hypothesis in dynamical settings. *Neural Networks*, 9, 427-434.

Costantini, G., Casali, D., Perfetti, R. (2003). Neural associative memory storing gray-coded gray-scale images. *IEEE Transactions on Neural Networks,* 14, 703-707.

Diamantaras, K.I., Kung, S.Y. (1996). *Principal Component Neural Networks.* New York: Wiley.

Garner, W.R. (1974). *The processing of information and structure.* New York: Wiley.

Harnad, S. (1990). The symbol-grounding problem. *Physica D, 42,* 335-346.

Hassoun, M.H. (1989). Dynamic heteroassociative neural memories. *Neural Networks, 2,* 275-287.

Hinton, G. E., Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504-507.

Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences, 79,* 2554-2558.

Hopfield, J. J., Feinstein, D. I., Palmer, R. G. (1983). Unlearning has a stabilizing effect in collective memories. *Nature, 304*, 158-159.

Hyvarinen, A., Oja, E. (2000). Independent component analysis: algorithms and applications, *Neural Networks, 13,* 411-430.

Karhunen, J., Pajunen, P., Oja, E. (1998). The nonlinear PCA criterion in blind source separation: Relations with other approaches. *Neurocomputing, 22,* 5-20.

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transac. on Systems, Man and Cybernetics, 18,*49-60.

Kosko, B. (1990). Unsupervised learning in noise. *IEEE Transactions on Neural Networks, 1,* 44-57.

Murphy, G.L. (2002). *The big book of concepts.* Cambridge, MA: MIT Press.

Schyns, P.G., Goldstone, R.L., Thibaut, J.P. (1998). The development of features in object concepts. *Brain and Behavioral Sciences, 21,* 1-54.

Sutton, R.S. (1988). Learning to predict by the methods of temporal difference. *Machine learning, 3,* 9-44.

Wang, C.C., Hwang, S.M., Lee, J.P. (1996). Capacity analysis of the asymptotically stable multi-valued exponential bidirectional associative memory. *IEEE transactions on systems, Man and Cybernetics - Part B: Cybernetics, 26,* 733-743.

Yen, G. G., Michel, A. N. (1996). Unlearning algorithm in associative memory. *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Procesing, 43,* 723-729.

Zhang, D., Chen, S. (2003). A novel multi-valued BAM model with improved error-correcting capability. *Journal of electronics, 20,* 220-223.

Zurada, J.M., Cloete, I., van der Peol, E. (1996). Generalized Hopfield networks for associative memories with multi-valued stable states. *Neurocomputing, 13, 135-149.*