In Proceedings of ICCM - 2007- Eighth International Conference on Cognitive Modeling. 97 - 102. Oxford, UK: Taylor & Francis/Psychology Press.

Category Development and Reorganization Using a Bidirectional Associative Memory-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

This paper shows that the recently proposed Feature-Extracting Bidirectional Associative Memory (FEBAM) can create and reorganize item clusters (or categories). This model, contrary to most other cluster-creating architectures, is based on projection methods and follows associative model principles (e.g. distribution of information, pattern completion and noise tolerance). Using a bidirectional associative memory (BAM)-inspired architecture, the resulting model is tested by simulating iterative cluster development and reorganization of artificial stimuli and alphanumeric patterns. In contrast to classic clustering techniques, the model is able to reproduce predetermined categories by iteratively reevaluating cluster membership, and allows given category members to move from a category to another. Because FEBAM has been shown to possess many more interesting properties, it is argued that the model possesses more cognitive explanative power than other comparable models and algorithms.

Introduction

Categories

In everyday life, humans are constantly exposed to situations in which they must group perceptual patterns (such as visual objects) into categories, in order to act upon the identity and properties of the encountered stimuli. To achieve this task properly, a cognitive system must adapt to many different environments which necessitate a broad range of behaviors according to context. Cognitive scientists have historically argued over the fact that the human cognitive system either uses generic abstractions (such as prototypes) or very specific perceptual stimulations (such as complete exemplars) to achieve category learning and further classification. Seemingly, the use of one of these representations is likely to be closely linked to specific environmental demands and system goals (Murphy, 2002).

Clustering in Neural Nets

More generally speaking, category formation, in a perceptual framework, is often seen as a "clustering" of similar patterns in common categories, a process akin to classic clustering techniques, which involve partitioning stimulus spaces in a number of finite categories ("clusters").

In artificial neural networks, clustering is a well-developed technique used mainly in competitive models (Kohonen, 1989; Grossberg, 1988). In these models, each output unit represents a specific cluster of items. When taking a decision, the association between an exemplar and its determined cluster unit in the output layer is strengthened. In "hard competitive" networks (Grossberg, 1988), exemplars may only be associated with one cluster (i.e. only one output unit at a time can be activated).

An example of hard competitive network is the Adaptive Resonance Theory (ART: Carpenter & Grossberg, 1987; Grossberg, 1988). ART networks are able to deal effectively with the exemplars-prototype scheme, while being able to answer the stability-plasticity dilemma. These unsupervised competitive models achieve the desired behavior by using a novelty detector (through "vigilance"); various degrees of generalization can be achieved with this procedure. If the value of the vigilance parameter is low, broad categories are developed; if it is high, narrow categories are developed, with the network ultimately performing exemplar learning.

In "soft computing" (Kohonen, 1989), exemplars may be associated with many clusters at differing degrees. This provides a more distributed classification; for instance, an exemplar may be geometrically positioned between two clusters, and possess various degrees of membership.

PCA networks (Diamantaras & Kung, 1993) can also be used to achieve clustering; in this case, each category is defined by a linear sum of extracted orthogonal components. Nonlinear PCA networks (Karhunen, Pajunen & Oja, 1998) are not restrained by this orthogonal requirement; hence, correlated components can be found (Hyvarinen & Oja, 2000). In all cases, once an item has been linked to a specific cluster, there is no mechanism allowing this item's category membership to be modified. Each model's internal structure is based on a specific metric that is constant over the training period.

Model Overview

In this paper, we show that the FEBAM model can be used to form clusters using various exemplar sets. First, it is shown that prototypes can be stored in the network's memory, regardless of the number of units. In this situation, exemplars form transient memories that can be used for identification, while prototypes constitute attractors that can be used for categorization. Second, when using a unit recruiting procedure, the model can ultimately develop exemplar-based attractors. In FEBAM, it is the number of units that specifies category broadness (in comparison to novelty detection in ART models). If, for instance, additional units are recruited and trained during learning, then the model is able to develop a greater amount of narrower categories. Ultimately, the network can perform "exemplar clustering".

Model Description

Architecture

FEBAM's architecture is based on a BAM architecture (Kosko, 1988) proposed by Hassoun (1989) and Chartier & Boukadoum (2006a). It consists in two Hopfield-like neural networks interconnected in head-to-toe fashion.

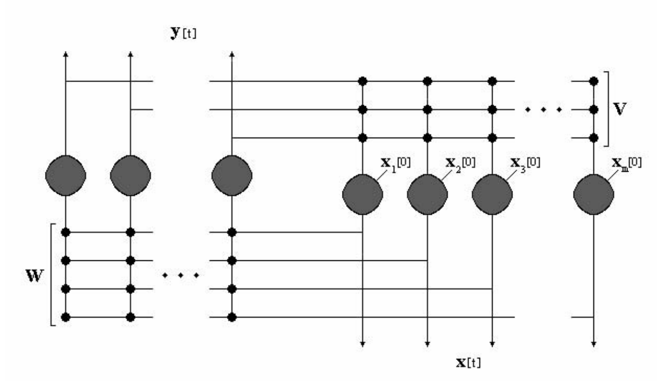


Figure 1: FEBAM network architecture.

When connected, these networks allow a recurrent flow of information to be processed bidirectionally. As shown in Figure 1, the **W** layer returns information to the **V** layer and vice versa. As in a standard BAM, both layers serve as a teacher for the other layer and the connections are explicitly depicted in the model[1]. To enable a BAM to perform clustering, one set of those explicit connections must be removed. Thus, in contrast with the standard BAM architecture, the "initial output" **y**(0) is not obtained externally, but is instead acquired by iterating once through the network, as depicted in Figure 2.

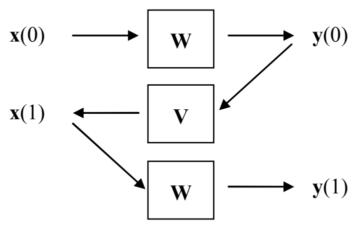


Figure 2: Output iterative process used for learning updates.

In the context of item cluster creation, the **W** layer will be used to determine the maximal number of clusters created by the network; the more units in that layer, the more possible clusters. The exact relationship between these quantities is the following: the theoretical maximal number of categories developed by the network is equal to 2^n, where n is the number of units in the network.

Output Function

FEBAM's output function is expressed by the following equations:

$$\forall\, i, \ldots, N,\ \mathbf{y}_i(t+1) = \begin{cases} 1, & \text{If } \ \mathbf{Wx}_i(t) > 1 \\ -1, & \text{If } \ \mathbf{Wx}_i(t) < -1 \\ (\delta+1)\mathbf{Wx}_i(t) - \delta(\mathbf{Wx})_i^3(t), & \textit{Else} \end{cases} \tag{1}$$

and

$$\forall\, i, \ldots, M,\ \mathbf{x}_i(t+1) = \begin{cases} 1, & \text{If } \ \mathbf{Vy}_i(t) > 1 \\ -1, & \text{If } \ \mathbf{Vy}_i(t) < -1 \\ (\delta+1)\mathbf{Vy}_i(t) - \delta(\mathbf{Vy})_i^3(t), & \textit{Else} \end{cases} \tag{2}$$

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. Like in any nonlinear dynamic system, to guarantee that a given output converges to a fixed point such as $\mathbf{x}^*(t)$ or $\mathbf{y}^*(t)$, the slope of the outputs function's derivative must be positive and smaller than one (Chartier & Proulx, 2005; Kaplan & Glass, 1995):

$$\frac{d\mathbf{y}(t+1)}{d\mathbf{Wx}(t)} = 0 < (\delta+1) - 3\delta(\mathbf{Wx}(t))^2 < 1 \tag{3}$$

$$\frac{d\mathbf{x}(t+1)}{d\mathbf{Vy}(t)} = 0 < (\delta+1) - 3\delta(\mathbf{Vy}(t))^2 < 1 \tag{4}$$

[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 neuropsychological perspective.

This condition is satisfied when $0 < \delta < 0.5$ for bipolar stimuli[2]. Figure 3 illustrates the shape of the output function when $\delta = 0.4$. This output function possesses the advantage of exhibiting continuous-valued (gray-level) attractor behavior. Such properties contrast with networks using a standard nonlinear output function, which can only exhibit bipolar attractor behavior (e.g. Kosko, 1988).

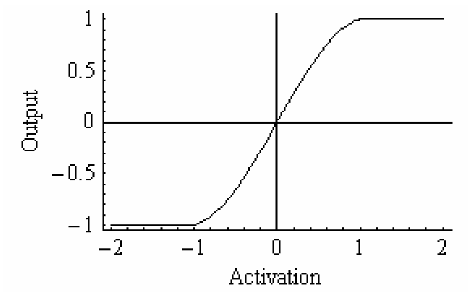


Figure 3: Output function for $\delta = 0.4$.

Learning Function

Learning is based on time-difference Hebbian association (Chartier & Proulx, 2005; Kosko, 1990; Oja, 1989; Sutton, 1988), and is formally expressed by the following equations:

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta(\mathbf{y}(0) - \mathbf{y}(t))(\mathbf{x}(0) + \mathbf{x}(t))^{\mathrm{T}} \qquad (5)$$

and

$$\mathbf{V}(k+1) = \mathbf{V}(k) + \eta(\mathbf{x}(0) - \mathbf{x}(t))(\mathbf{y}(0) + \mathbf{y}(t))^{\mathrm{T}} \qquad (6)$$

where η represents a learning parameter; T is the usual transpose operator, $\mathbf{y}(0)$ 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 covariation in its autoassociative memory version (Chartier & Boukadoum, 2006a). For weight convergence to occur, η must be set according to the following condition (Chartier & Proulx, 2005; Chartier, & Boukadoum, M., 2006b):

$$\eta < \frac{1}{2(1-2\delta)Max[N,M]}, \quad \delta \neq \tfrac{1}{2} \qquad (4)$$

General Procedure

To obtain the various vectors needed for weight updates, stimuli iteration is performed as depicted in Figure 2. First, an initial stimulus ($\mathbf{x}(0)$) is introduced to the **W** layer, yielding an initial output ($\mathbf{y}(0)$). This output represents the input's classification into a distributed cluster. Second, using this initial output, the information is sent back to the input layer using the **V** layer's connections; this results in another output, $\mathbf{x}(1)$. Third, this novel output is then used to obtain the final classification output ($\mathbf{y}(1)$) by using the **W** connections once again.

As stated by Equations 5 and 6, weights can only converge when "internal feedback" is identical to the initial inputs (that is, $\mathbf{y}(1) = \mathbf{y}(0)$ and $\mathbf{x}(1) = \mathbf{x}(0)$) (in other words, when the network *resonates*). 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).

Simulations

Simulation 1

A first simulation was conducted in order to demonstrate the network's ability to cluster exemplars into categories. For this simulation, artificial pixel-based stimuli were created. Stimuli examples are shown in Figure 4.

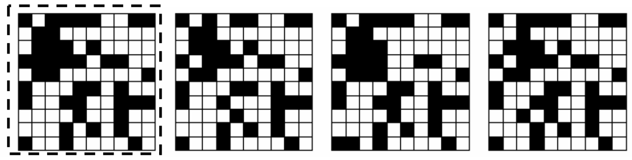


Figure 4: Examples of a prototype (left image) with three associated exemplars.

Methodology Eight category prototypes were produced by generating bipolar-valued vectors, for which the value of each vector position (or "feature") followed a random discrete uniform distribution. The presence of a feature (black pixel) was represented by a value of +1, and the absence of a feature (white pixel) by a value of -1. Each prototype vector comprised 100 features. Correlations between category prototypes are shown in Table 1.

Table 1: Correlations between category prototypes for Simulation 1[3].

	P2	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**
P1	-0.04	0.04	-0.02	0.12	0.14	-0.04	-0.16
P2		-0.04	0.06	0.08	0.30	-0.08	0.12
P3			0.02	-0.12	0.22	0.00	-0.16
P4				0.02	0.20	0.02	-0.10
P5					-0.02	0.04	0.08
P6						0.02	0.02
P7							-0.04

Ten exemplars were generated using each prototype, for a total of 80 items. Each exemplar was created by randomly "flipping" the value of between one to six features. Average within-category correlations are presented in Table 2.

[2] Generalization to real-valued stimuli can be found in Chartier & Boukadoum (2006a).

[3] **Px** represents the Category x prototype. In this Table, as well as all following Tables, *Pearson's r* scores are reported.

Table 2: Average within-category correlations for Simulation 1[4]

C1	C2	C3	C4	C5	C6	C7	C8
0.86	0.88	0.89	0.89	0.88	0.86	0.84	0.85

Learning followed the following procedure:

0. Random weight initializations;
1. Random selection of a given exemplar;
2. Weight updates according to Equations 5 & 6;
3. Repetition of 1 and 2 for 600 learning trials.

To assess the network's behavior, several simulations were performed using different numbers of units for the **W** layer.

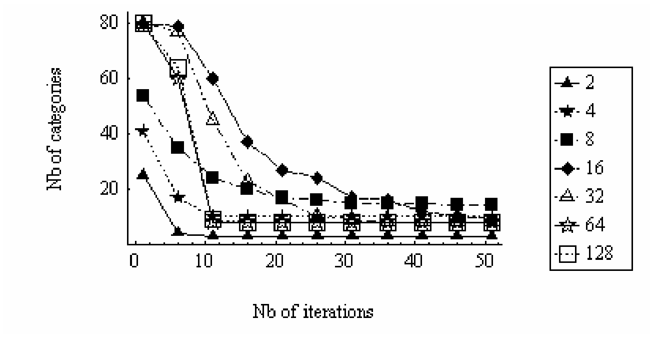


Figure 5: Number of clusters developed by the network as a function of the number of recall iterations. Each line represents a different number of units present in the **W** layer of the network.

Results Looking at Figure 5, one finds that for adequate clustering, the network needs at least $\log_2 n$ units. Here, because there were eight predetermined categories, the network thus needed a theoretical minimum of three units ($\log_2 8=3$). However, if there are much less units than exemplars, the network is likely to develop a greater number of categories than necessary. In this example, it was the case when 4, 8 or 16 units were used. If the number of units was sufficient, the number of categories developed by the network then matched the number of desired categories. In this example, when 32, 64 or 128 units were used, the network correctly developed eight categories.

In addition, when using 32 units or more, Figure 5 indicates that as the number of iterations increases, the network switches from a specific identification to a categorization process. For the first few iterations (~10 or less), there are almost as many clusters as there are exemplars. When achieving more iterations (~30 or more), the number of clusters is reduced so that it becomes similar to the number of prototypes (or categories). Consequently, if the number of units is great enough, FEBAM can be used to identify specific exemplars even though its memory has extracted the corresponding prototypes, depending on the time allowed (i.e. number of iterations performed) before an output is required.

Simulation 2

The purpose of the second simulation was to study the number of categories developed by FEBAM as a function of the number of output units. In this simulation, not all units were initially available. A unit recruiting mechanism was introduced to allow the network to slowly converge towards a number of categories equal to the number of exemplars. A new unit was recruited by the model after a certain amount of learning trials had been achieved.

Methodology Four prototypes were generated using the method detailed for Simulation 1. Correlations between category prototypes are shown in Table 3.

Table 3: Correlations between category prototypes for Simulation 2[5].

	P2	P3	P4
P1	0.02	0.08	0.08
P2		-0.02	0.02
P3			0.12

Ten exemplars were generated using each prototype, for a total of 40 items. Each exemplar was created by randomly flipping the value of between two to ten features. Average within-category correlations are presented in Table 4.

Table 4: Average within-category correlations for Simulation 2[6]

C1	C2	C3	C4
0.76	0.77	0.78	0.77

The simulation was conducted, starting with two initial units in the **W** layer. The number of units in the **V** layer remained constant at 100. After each phase of 600 learning trials, another unit was added to the **W** layer. This was repeated until there were 64 units. After each learning phase, a recall test was performed to estimate the number of categories developed by the network. At test, each given stimulus was iterated 200 times in the network before the final stabilized output was given. This was done to establish the nature of the network's attractors after each step.

Learning followed the following procedure:

0. Random weight initializations;
1a. Random selection of exemplars for learning, according to Equations 5 and 6;
1b. Repetition of 1a for 600 learning trials;
2. Addition of a new output unit;
3. Repetition of 1 and 2 until the number of output units is equal to 64.

[4] **Cx** represents Category x.

[5] **Px** represents the Category x prototype.

[6] **Cx** represents Category x.

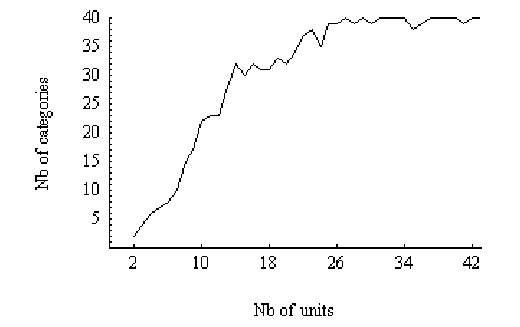


Figure 6: Simulation 2. Number of categories developed by the network as a function of the number of units iteratively added.

Results Figure 6 indicates that as more units are added to the network, more categories are developed. Hence, the network's architecture does "take advantage" of the recruiting mechanism properly, by separating the stimuli in more and more specific ways, iteratively going from groupings around a generic abstraction to single exemplars. Hence, this simulation shows that FEBAM can act as both an exemplar and a prototype memory, depending on the number of units that have been dynamically recruited.

This is clearly visible in Figure 7, which displays a tendency towards creating one-exemplar clusters as the number of possible clusters increases. Figure 7 also shows that while the unit recruiting mechanisms allows for some cluster reorganization (for example, some items associate with different cluster members when adding units), the exemplars always closely follow the predetermined categorical segmentation, that is they tend to cluster with items generated by the same prototype.

Simulation 3

Simulation 2 was replicated, but this time using stimuli with no predetermined categorical (or cluster) membership. Pixel representations of letters of the alphabet were chosen because they represent a wide range of intercorrelations.

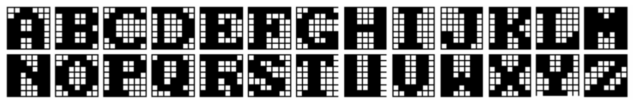


Figure 8: Set of patterns used for training.

Methodology The patterns used for the simulations are shown in Figure 8. Each pattern consisted of a 7 x 7 pixel matrix representing a letter of the alphabet. Once again, white and black pixels were respectively assigned corresponding values of -1 and +1. Correlations between the patterns varied from 0.02 to 0.84.

The simulation procedure was identical to that of Simulation 2, except for the number of stimuli involved (26 instead of 40).

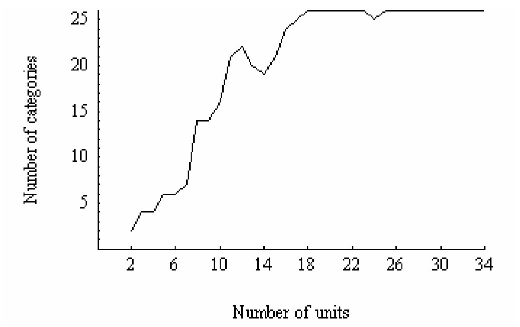
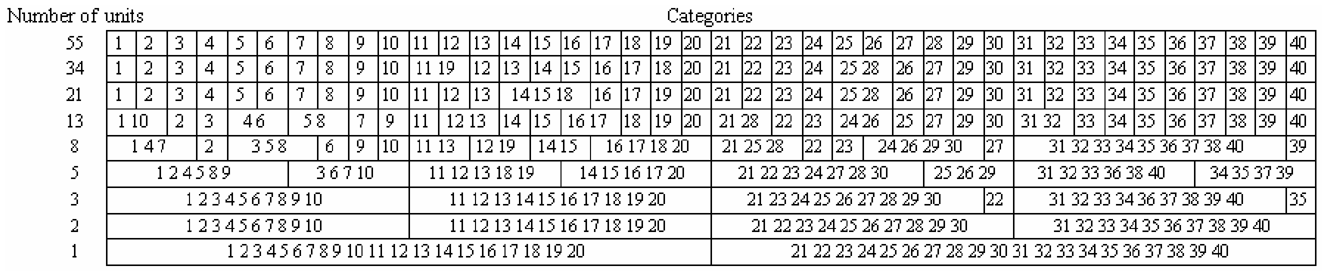


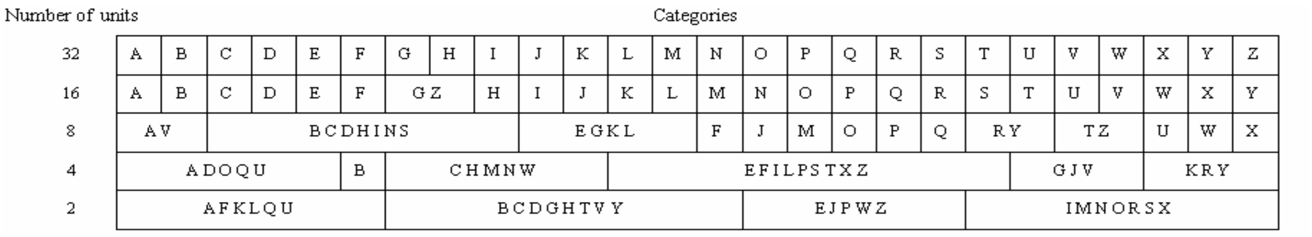
Figure 9: Simulation 3. Number of categories developed by the network as a function of the number of units iteratively added.

Results As in Simulation 2, Figure 9 indicates that the network takes advantage of the recruiting mechanism; as more units are added to the network, more categories are developed. However, as Figure 10 shows, a member from a given category is not tied to a specific type of categorical clustering. In fact, given exemplars can aggregate into a given cluster, and later on leave their present cluster to join another one, or form a new cluster with other exemplars. This contrasts markedly from classic clustering techniques.



Number of units	Categories																																							
55	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
34	1	2	3	4	5	6	7	8	9	10	11 19	12	13	14	15	16	17	18	20	21	22	23	24	25 28	26	27	29	30	31	32	33	34	35	36	37	38	39	40		
21	1	2	3	4	5	6	7	8	9	10	11	12	13	14 15 18	16	17	19	20	21	22	23	24	25 28	26	27	29	30	31	32	33	34	35	36	37	38	39	40			
13	1 10	2	3	4 6	5 8	7	9	11	12 13	14	15	16 17	18	19	20	21 28	22	23	24 26	25	27	29	30	31 32	33	34	35	36	37	38	39	40								
8	1 4 7	2	3 5 8	6	9	10	11 13	12 19	14 15	16 17 18 20	21 25 28	22	23	24 26 29 30	27	31 32 33 34 35 36 37 38 40	39																							
5	1 2 4 5 8 9	3 6 7 10	11 12 13 18 19	14 15 16 17 20	21 22 23 24 27 28 30	25 26 29	31 32 33 36 38 40	34 35 37 39																																
3	1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 23 24 25 26 27 28 29 30	22	31 32 33 34 36 37 38 39 40	35																																		
2	1 2 3 4 5 6 7 8 9 10	11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30	31 32 33 34 35 36 37 38 39 40																																				
1	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20	21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40																																						

Figure 7. Clusters developed by the network as a function of the number of units. Some exemplars cluster with different items as the number of units increases. As can be seen, as soon as the number of units (two) allows for the formation of four clusters, each formed cluster exhibits its predetermined belonging exemplars.



Number of units	Categories																									
32	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
16	A	B	C	D	E	F	G Z		H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
8	A V		B C D H I N S							E G K L				F	J	M	O	P	Q	R Y		T Z		U	W	X
4	A D O Q U					B	C H M N W					E F I L P S T X Z									G J V			K R Y		
2	A F K L Q U						B C D G H T V Y								E J P W Z					I M N O R S X						

Figure 10. Clusters developed by the network as a function of the number of units. Once again, cluster reorganization can be detected as the number of possible clusters increases.

Discussion

In this paper, it has been shown that FEBAM, which is based on an associative learning architecture, is able to create increasingly precise clusters by recruiting additional units, and reorganize these clusters during training. Results from the first simulation have shown that the developed memory can be used to perform identification as well as categorization. This property is made possible by the dynamic memory recall process. In this case, specific exemplars are "transient" memories, while prototypes are attractors. However, simulations 2 and 3 show that by using a unit recruiting process, exemplars can also become attractors. In previous studies, FEBAM has been shown to achieve perceptual feature extraction and learning in noisy environments (Giguère, Chartier, Proulx & Lina, in press), as well as nonlinear principal component extraction and blind source extraction (Chartier, Giguère, Renaud, Lina & Proulx, in press). Moreover, 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). FEBAM is therefore believed to constitute a serious candidate for larger-scale cognitive modeling of human perceptual and categorical processes.

Further studies should investigate the frequency effects of both exemplars and categories. Based on results expressed by Figure 5, further studies should also investigate the categorization and classification processes as a given input (or exemplar) iterates through the network. FEBAM could ultimately link, through a dynamic memory system, both exemplar and prototype approaches. In its present form, the model is able to cluster together information if between-category variability is greater than within-category variability. Various variability clustering techniques adding an external teacher or reinforcement should therefore also be explored.

References

Carpenter, G. A. and Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing. 37,* 54-115.

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.

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

Giguère, G., Chartier, S., Proulx, R., Lina, J.M. (2007, *in press*). Creating perceptual features using a BAM-inspired architecture. *Proceedings of the 29th Annual Conference of the Cognitive Science Society.*

Grossberg, S. (1988). Nonlinear Neural Networks: Principles, Mechanisms, and Architectures. *Neural Networks, 1,* 17-61.

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

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

Kaplan, D., & Glass, L. (1995). *Understanding nonlinear dynamics* (1st ed.). New-York: Springer-Verlag.

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

Kohonen, T. (1989). *Self-Organization and Associative Memory* (3rd ed.). Berlin: Springer-Verlag.

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

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

Murphy, G.L. (2002). *The Big Book of Concepts.* Cambridge, MA: MIT Press.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems, 1,* 61-68.

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