A Novel Lattice Associative Memory Based on Dendritic Computing

Gerhard X. Ritter[1], Darya Chyzhyk[2],* Gonzalo Urcid[3],** Manuel Graña[2]

[1] CISE Department, University of Florida, Gainesville, FL 32611–6120, USA
ritter@cise.ufl.edu
[2] Computational Intelligence Group, University of the Basque Country, Spain
darya.chyzhyk@ehu.es, manuel.grana@ehu.es
[3] Optics Department, INAOE, Tonantzintla, Pue. 72000, Mexico
gurcid@inaoep.mx

Abstract. We present a novel hetero-associative memory based on dendritic neural computation. The computations in this model are based on lattice group operations. The proposed model does not suffer from the usual storage capacity problem and is extremely robust in the presence of various types of noise and data corruption.

Keywords: Dendritic Computing, Hetero-associative memory, Lattice algebra, Data Corruption.

1 Introduction

The concept of an associative memory is a fairly intuitive one as it is based on the observation that an associative memory seems to be one of the primary functions of the brain. We easily *associate* the face of a friend with that of the friend's name, or a name with a telephone number. For this reason artificial neural networks (ANNs) that are capabable of storing several types of patterns and corresponding associations are referred to as *associative memories*. Such memories retrieve stored associations when presented with corresponding input patterns. An associative memory is said to be *robust in the presence of noise* if presented with a corrupted version of a prototype input pattern it is still capable of retrieving the correct association.

In classical pattern recognition, patterns are viewed as column vectors in Euclidean space. Each component of a pattern vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)' \in \mathbb{R}^n$ corresponds to one of the pattern's features. The numerical value x_i of a pattern feature can represent a variety of objects or physical features such as signal strength, curvature, a probability value, mean mass, and so on. One goal in the theory of associative memories is for the memory to recall a stored pattern $\mathbf{y} \in \mathbb{R}^m$ when presented a pattern $\mathbf{x} \in \mathbb{R}^n$, where the pattern association expresses

[*] Corresponding author.

[**] G. Urcid is grateful with CONACYT for partial financial support grant # 22036.

some desired pattern correlation. More precisely, suppose $X = \{\mathbf{x}^1, \ldots, \mathbf{x}^K\} \subset \mathbb{R}^n$ and $Y = \{\mathbf{y}^1, \ldots, \mathbf{y}^K\} \subset \mathbb{R}^m$ are two sets of pattern vectors with desired association given by the diagonal $\{(\mathbf{x}^\xi, \mathbf{y}^\xi) : \xi = 1, \ldots, K\}$ of $X \times Y$. The goal is to store these pattern pairs in some memory M such that for $\xi = 1, \ldots, K$, M recalls $\mathbf{y}^\xi$ when presented with the pattern $\mathbf{x}^\xi$. If such a memory M exists, then we shall express this association symbolically by $\mathbf{x}^\xi \to M \to \mathbf{y}^\xi$. Whenever $X = Y$, then the memory M is called an *auto-associative memory*, otherwise it is called a *hetero-associative* memory or simply an *associative* memory.

The matrix correlation memories resulting from the work of Steinbuch, Kohonen, Anderson, and Hopfield were the earliest artificial neural network (ANN) examples of associative memories [1–10]. Matrix correlation memories based on lattice computations were first introduced in the late 1990s [11–13]. These memories had the advantage of unlimited storage capacity and one step convergence. However, they were susceptible to certain types of random noise. The concept of dendritic computing was partially due to trying to eliminate the noise problem encountered in the construction of artificial memories. The other reason was to provide an artificial neural paradigm that is closer related to actual biological neural computation [14].

Lattice based Neural Networks (LNNs) - although not yet recognized as mainstream in machine learning - have become an integral part of artificial neural network theory [15, 16]. One reason for this is their simplicity and fast learning methods and another is due to their successful applicability in several disciplines [17–22]. In this paper the focus is on a novel Dendritic Lattice based (hetero) Associative Memory or, simply, DLAM. Recently two new DLAMs have appeared in the literature [23] and [24]. The former being a generalization of the DLAMs given in [25], while the latter had no predecessor within lattice theory. However, the latter model was presented as an auto-associative memory. Here we show that the model easily generalizes to a hetero-associative memory. Similar to earlier lattice based associative memories, this new DLAM has unlimited storage capacity in that it can memorize any finite number of association and provides perfect recall for non-noisy input. However, as we shall demonstrate, its greatest advantage over prior associative memories is that it can recall association even when the input is an exemplar pattern that has been corrupted by more than 90% of random noise.

The remainder of this paper is partitioned into three sections. In Section 2 we provide a brief overview of Dendritic Lattice based Neural Networks (DLNNs). Rationale for the DLNN approach is not discussed as it can be found in [14]. Section 3 introduces the new DLAM model and explains the computations occuring in the different layers as well as the funtion of each layer. The robustness of the DLAM in the presence of various types of noise is demonstrated in Section 4. Conclusions and some pertinent observations are presented in the final section.

2 The Dendritic Lattice Based Model of ANNs

Roughly speaking, a lattice based neural network is an ANN in which the basic neural computations are based on the operations of a lattice ordered group. By a lattice ordered group we mean a set L with an associated algebraic structure $(L, \vee, \wedge, +)$, where $(L, \vee, \wedge)$ is a lattice and $(L, +)$ is a group with the property that every group translation is isotone; that is, if $x \leq y$, then $a+x+b \leq a+y+b$ $\forall\, a,\, b \in L$. Given the set $\mathfrak{O} = \{\vee, \wedge, +\}$ of lattice goup operations, then the symbols $\oplus$ and $\otimes$ will mean that $\oplus, \otimes \in \mathfrak{O}$ but are not explicitly specified lattice operations. Similarly, symbols of form $\bigoplus$ and $\bigotimes$ will denote lattice operations derived from the operations $\oplus$ and $\otimes$, respectively. For example, $\bigoplus_{i=1}^{n} a_i = a_1 \oplus a_2 \oplus \cdots \oplus a_n$. Hence, specifying $\oplus = \vee$, then $\bigoplus_{i=1}^{n} a_i = \bigvee_{i=1}^{n} a_i = a_1 \vee a_2 \vee \cdots \vee a_n$.

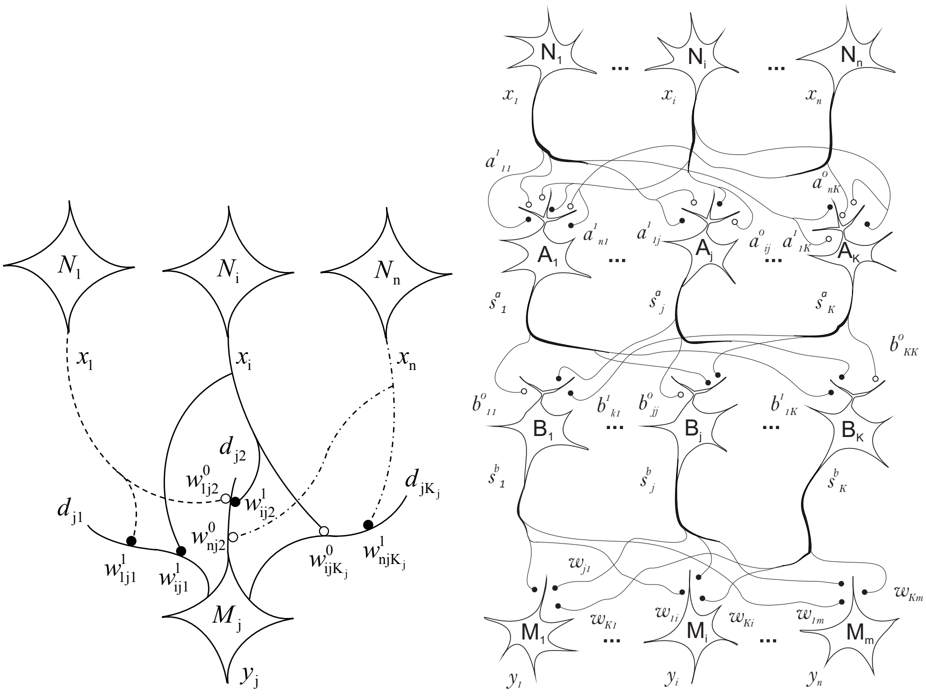
In the dendritic model of ANNs, a finite set of presynaptic neurons $N_1, \ldots, N_n$ provides information through its axonal arborization to the dendritic trees of some other finite set of postsynaptic neurons $M_1, \ldots, M_m$. The dendritic tree of a postsynaptic neuron M_j is assumed to consist of a finite number of branches $d_{j1}, \ldots, d_{jK_j}$ which contain the synaptic sites upon which the axonal fibers of the presynaptic neurons terminate. The *strength* of the synapse on the kth dendritic branch d_{jk} $(k \in \{1, \ldots, K(j)\})$ which serves as a synaptic site for a terminal axonal branch fiber of N_i is denoted by w_{ijk}^{ℓ} and is also called its *synaptic weight*. The superscript ℓ is associated with the postsynaptic response that is generated within and in close proximity of the synapse. Specifically, $\ell = 0$ and $\ell = 1$ denote an inhibitory or excitatory postsynaptic response, respectively. It is possible for several axonal fibers to synapse on the same or different synaptic sites on a given branch d_{jk}, with the former case implying that $w_{ijk}^{\ell} = w_{hjk}^{\ell}$. The total response (or output) of d_{jk} to the received input at its synaptic sites is given by

$$\tau_k^j(\mathbf{x}) = p_{jk} \bigoplus_{i \in I(k)} \bigotimes_{\ell \in \mathcal{L}(i)} [(-1)^{1-\ell}\,(x_i + w_{ijk}^{\ell})], \qquad (1)$$

where $\mathbf{x} = (x_1, \ldots, x_n) \in L^n$ with L^n denoting the n-fold cartesian product of L, $x_i \in L$ denotes the information propagated by N_i via its axon and axonal branches, $\mathcal{L}(i) \subseteq \{0, 1\}$ corresponds to the postsynaptic response generated at the synaptic region to the input received from N_i, and $I(k) \subseteq \{1, \ldots, n\}$ corresponds to the set of all presynaptic neurons with terminal axonal fibers that synapse on the kth dendritic branch of M_j. The value $p_{jk} \in \{-1, 1\}$ marks the final signal outflow from the kth branch as inhibitory if $p_{jk} = -1$ and excitatory if $p_{jk} = 1$. The value $\tau_k^j(\mathbf{x})$ is passed to the cell body of M_j and the state of M_j is a function of the combined values received from its dendritic structure and is given by

$$\tau^j(\mathbf{x}) = p_j \bigotimes_{k=1}^{K_j} \tau_k^j(\mathbf{x}), \qquad (2)$$

where K_j denotes the total number of dendritic branches of M_j and $p_j = \pm 1$ denotes the response of the cell to the received input. Here again $p_j = -1$ means



(a) (b)

Fig. 1. (a) Terminal branches of axonal fibers originating from the presynaptic neurons make contact with synaptic sites on dendritic branches of M_j.
(b) Structure of a dendritic network

rejection (inhibition) and $p_j = 1$ means acceptance (exitation) of the received input. Figure 1(a) illustrates the neural pathways from the presynaptic neurons to the postsynaptic neuron M_j. Figure 1(b) illustrates a dendritic network. The prime example of a lattice ordered group is the set $\mathbb{R}$ of real numbers together with the binary operations of the maximum ($\vee$) and minimum ($\wedge$) of two numbers and the group operation of addition, denoted by ($\mathbb{R}$, $\vee$, $\wedge$, $+$). It is also the lattice employed in this paper. Thus, for example, eqn.1 could assume the form

$$\tau_k^j(\mathbf{x}) = p_{jk} \bigvee_{i \in I(k)} \bigwedge_{\ell \in \mathcal{L}(i)} (-1)^{1-\ell} \left(x_i + w_{ijk}^{\ell} \right), \tag{3}$$

where $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, and $x_i \in \mathbb{R}$, while eqn.2 could be of form

$$\tau^j(\mathbf{x}) = p_j \sum_{k=1}^{K_j} \tau_k^j(\mathbf{x}). \tag{4}$$

3 Dendritic Lattice Associative Memories

The Dendritic Lattice based Associative Memory or DLAM described in this section can store any desirable number of pattern associations and has perfect recall when presented with an exemplary pattern. Furthermore, it is extremely robust in the presence of noise and can be applied to both Boolean and real number value patterns.

The proposed DLAM consists of four layers of neurons: an input layer, two hidden layers, and an output layer. The number of neurons in each layer is predetermined by the dimensionality of the pattern domains. Explicitely, if $X = \{\mathbf{x}^1, \ldots, \mathbf{x}^K\} \subset \mathbb{R}^n$ and $Y = \{\mathbf{y}^1, \ldots, \mathbf{y}^K\} \subset \mathbb{R}^m$, then the number of neurons in the input layer is n, in the two hidden layers it is K, and the number in the output layer is m. We denote the neurons in the input layer by $N_1, \ldots, N_n$, in the first hidden layer by $A_1, \ldots, A_K$, in the second hidden layer by $B_1, \ldots, B_K$ and in the output layer by $M_1, \ldots, M_m$. We refer to the first and second hidden layer as the *A-layer* and the *B-layer*, respectively. For a given input pattern $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, the ith neuron N_i will assume as its value the ith coordinate x_i of $\mathbf{x}$ and will propagate this value through its axonal arborization to the dentrites of the hidden layer neurons. The dendritic tree of each hidden neuron A_j has n single branches $d_{j1}, \ldots, d_{jn}$, and each neuron N_i has two axonal fibers terminating on the synaptic sites located on the corresponding branch d_{ji} of the hidden layer neuron A_j as depicted in Figure 1(b). Observe that in this formulation the dendritic branch counter $k = i$, making the extra counter k unecessary. The two synaptic weights associated with the two synaptic sites of d_{ji} will be denoted by a_{ij}^{ℓ} and defined by $a_{ij}^{\ell} = -x_i^j$ for $\ell = 0, 1$. The output of each dendritic branch is denoted by $\tau_i^j(\mathbf{x})$. Here we use the formula given by eqn. 3 in order to compute this value. Setting $p_{jk} = -1$ and using the fact that $I(k) = I(i) = \{i\}$, eqn. 3 reduces to

$$\begin{aligned}\tau_i^j(\mathbf{x}) &= -\bigwedge_{\ell=0}^{1}(-1)^{1-\ell}\left(x_i + a_{ij}^{\ell}\right) = -[-(x_i - x_i^j) \wedge (x_i - x_i^j)] \\ &= -[-(x_i - x_i^j) \wedge -(x_i^j - x_i)] = (x_i - x_i^j) \vee (x_i^j - x_i). \qquad (5)\end{aligned}$$

It follows from eqn. 5 that $\tau_i^j(\mathbf{x}) = 0 \Leftrightarrow x_i = x_i^j$ and $\tau_i^j(\mathbf{x}) > 0 \Leftrightarrow x_i \neq x_i^j$. The value $\tau_i^j(\mathbf{x})$ is passed to the cell body of A_j and its state is a function of the combined values received from its dendritic structure. This state is computed using eqn. 3 with $p_j = 1$. Specifically, we have

$$\tau_A^j(\mathbf{x}) = \sum_{i=1}^{n} \tau_i^j(\mathbf{x}) = \sum_{i=1}^{n}(x_i - x_i^j) \vee (x_i^j - x_i) = \sum_{i=1}^{n} |x_i - x_i^j|. \qquad (6)$$

It follows that each neuron A_j in the A-layer computes the L_1-distance between the input pattern $\mathbf{x}$ and the jth exemplar pattern $\mathbf{x}^j$. That is, $\tau_A^j(\mathbf{x}) = d_1(\mathbf{x}, \mathbf{x}^j)$. The activation function for the A-layer neurons is derived from the identity function, namely

$$f_A(z) = \begin{cases} z & \text{if } z \leq T \\ \infty & \text{if } z > T \end{cases}, \tag{7}$$

where T is a user defined threshold. We denote the output of A_j by $s_A^j = f_A(\tau_A^j(\mathbf{x}))$ and the collective output of the A-level neurons by s_A.

The output s_A of the A-layer serves as input to the neurons in the B-layer. Here each neuron B_j has two dendrites d_{j1} and d_{j2}. The dendrite d_{j1} has only one synaptic site on which only an axonal fiber of A_j terminates. The synaptic weight of this synapse is given by $b_{jj}^\ell = 0$, with $\ell = 0$. The second branch, d_{j2}, receives input from all the remaining neurons of the A-layer; i.e., from $\{A_1, \ldots, A_K\} \setminus \{A_j\}$. The synaptic weight of the synaptic site on d_{j2} for the terminal axonal fiber of neuron A_r, with $r \neq j$, is given by $b_{rj}^\ell = 0$, where $\ell = 1$. To compute the values $\tau_k^j(\mathbf{x})$ for the two dendrites of B_j, we use the general formula

$$\tau_k^j(\mathbf{x}) = p_{jk} \bigwedge_{i \in I(k)} \bigwedge_{\ell \in \mathcal{L}(i)} (-1)^{1-\ell} \left(x_i + w_{ijk}^\ell\right) \tag{8}$$

which is similar to eqn. 3. For $k = 1$ and $i = j$ we have $I(1) = \{1\}$ and $\mathcal{L}(j) = \{0\}$. Setting $p_{j1} = 1$ and employing eqn. 8 one obtains

$$\tau_1^j(s_A) = \bigwedge_{i \in I(1)} \bigwedge_{\ell \in \mathcal{L}(j)} (-1)^{1-\ell} \left(s_A^j + b_{jj}^\ell\right) = -s_A^j. \tag{9}$$

Similarly, for d_{j2} we have $k = 2$, $i = r$, $I(2) = \{1, \ldots, k\} \setminus \{j\}$, and $\mathcal{L}(r) = \{1\}$. Again setting $p_{j2} = 1$, one obtains

$$\tau_2^j(s_A) = \bigwedge_{r \in I(2)} \bigwedge_{\ell \in \mathcal{L}(r)} (-1)^{1-\ell} \left(s_A^r + b_{jr}^\ell\right) = \bigwedge_{r \neq j} s_A^r. \tag{10}$$

The values $\tau_1^j(s_A^j)$ and $\tau_2^j(s_A)$ flow into the cell body of B_j and its state is a function of the combined values received from its dendrites:

$$\tau_B^j(s_A) = \sum_{k=1}^{2} \tau_k^j(s_A) = \tau_1^j(s_A) + \tau_2^j(s_A) = \bigwedge_{r \neq j} s_A^r - s_A^j. \tag{11}$$

We consider the two possibilities of $\bigwedge_{r \neq j} s_A^r > s_A^j$ and $\bigwedge_{r \neq j} s_A^r \leq s_A^j$. The first possible case implies that $s_A^j \neq \infty$ and, hence, $s_A^j = d_1(\mathbf{x}, \mathbf{x}^j)$. That is, the pattern vector $\mathbf{x}$ is closer to the exemplar pattern $\mathbf{x}^j$ than any of the other exemplar pattern and within the allowable threshold T. The second possibility implies that either there is another exemplar $\mathbf{x}^r$ which is closer (or just as close) to $\mathbf{x}$ as $\mathbf{x}^j$, or that $\mathbf{x}^j$ surpassed the threshold T. In the first case we want the neuron B_j to send that information to the output neurons while in the second case we do not want B_j to fire. In order to achieve this we define the activation function to be the lattice-based hardlimiter

$$f_B(z) = \begin{cases} 0 & \text{if } z > 0 \\ -\infty & \text{if } z \leq 0 \end{cases}. \tag{12}$$

Thus, the output of B_j is given by $s_B^j = f_B[\tau_B^j(s_A)]$ and serves as the input to the output layer M. Each output neuron M_i, $i = 1, \ldots, m$, has only a single dendrite d_{i1} receiving excitatory input from all K neurons of the B-layer. The weight associated with the synaptic site on d_{i1} of the terminal axonal fiber of B_j is defined as $w_{ji}^1 = y_i^j$. Here $j = 1, \ldots, K$ and $i = 1, \ldots, m$. Using eqn. 3 to compute the output pattern, we note that since each M_i has only one dendrite d_{i1} we have $k = 1$ (for each i) and $I(1) = \{1, \ldots, , K\}$. Also, since we are dealing with excitatory synaptic responses only, we have that for each $j \in I(1)$, $\mathfrak{L}(j) = \{1\}$. By setting $p_{i1} = 1$, eqn. 3 now reduces to

$$\tau_1^i(s_B) = \bigvee_{j=1}^{K} (s_B^j + w_{ji}^1) = \bigvee_{j=1}^{K} (s_B^j + y_i^j). \tag{13}$$

Observe that $\tau^i(s_B) = \tau_1^i(s_B)$ The activation function for each neuron M_i is simply the identity function so that the output y_i of M_i is given by $y_i = \tau^i(s_B)$. The total output of the the set $M_1, \ldots, M_m$ is the vector $\mathbf{y} = (y_1, \ldots, y_m)$. It remains an easy excercise to show that for an uncorrupted input $\mathbf{x}^j$ the output at the M-level will be $\mathbf{y}^j$.

4 Experiments with Noisy and Corrupted Inputs

In this section we present results of some computational experiments that demonstrate the performance of the proposed DLAM in recalling stored associations when presented with corrupted versions of exemplar patterns. We use images to form pattern vectors only to provide a visual interpretation of the recall. In general. Associative memories are used for pattern recall, not image recall. The transformation of images into vectors is accomplished via the usual column-scan method. We created a database of image patterns from image obtained from various websites.

Experiment 1. In this experiment, each of the sets X and Y consists of six Boolean exemplar patterns. The set X is derived from the set of six 700×350 Boolean images shown in the top row of Figure 2, while the set of associated output patterns is derived from the six 380×500 Boolean images shown in the botton row of Figure 2. Thus, $X = \{\mathbf{x}^1, ..., \mathbf{x}^6\}$, with $x^j \in \{0,1\}^{245000}$, and $Y = \{\mathbf{y}^1, ..., \mathbf{y}^6\}$ with $\mathbf{y}^j \in \{0,1\}^{190000}$.

Every pattern image was corrupted adding “salt and pepper” noise. Each noisy pixel of corrupted image is rounded to either 0 or 1 to preserve the Boolean character of the images.

The range of the noise levels varied from 1% to 99% and was tested on all the images. Instances of corrupted input images are shown in Figure 3. The corresponding output images recalled by the DLAM are shown in the bottom row. The DLAM shows perfect recall robustness to salt and pepper noise.

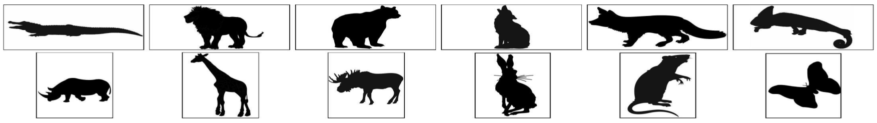


Fig. 2. Set of Boolean images of six predators in the first row and corresponding six preys in the second row.

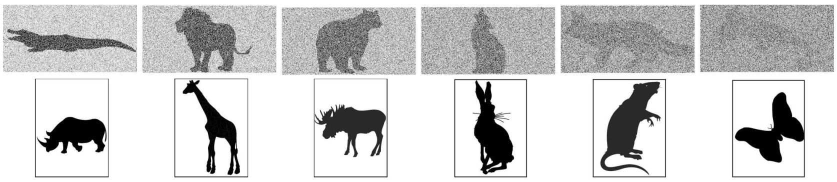


Fig. 3. First row: Boolean exemplar images corrupted with increasing levels of "salt and pepper" noise of 50%, 60%, 70%, 80%, 90%, and 94% (left to right).
Bottom row: Perfect recall associations derived from the noisy input patterns in the top row.

Experiment 2 In this example we use a database of grayscale images in which the value of each pixel has an integer intensity value in a range from 0 (black) to 255 (white). Similar to Example 1, we use predator-prey association images as shown in Figure 4. Both predator and prey images are of size 265×265. In mathematical terminology we have $X = \{\mathbf{x}^1, ..., \mathbf{x}^K\} \subset \mathbb{R}^{70225}$ and $Y = \{\mathbf{y}^1, ..., \mathbf{y}^K\} \subset \mathbb{R}^{70225}$, $K = 5$. In this experiment we use different types of pattern corruption and noise. Specifically, we simulate noise pattern acquisition by increasing and decreasing image contrast, approximating linear camera motion, applying circular averaging filter, employing the morphological transforms of dilation and erosion with different structuring elements, and by using Gaussian and uniform noise. Figure 5 shows some of the tested image corruption changes. Different types of noise corruption have been applied to different images. The first column represents a motion blur, the 2nd Gaussian noise, the 3rd the application of a circular averaging filter, the 4th a morphological erosion with a line as structuring elements and the 5th a morphological dilation with elipsoid as structuring elements.

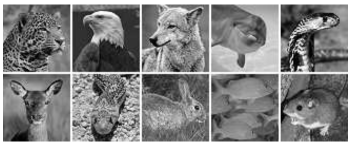


Fig. 4. Set of grayscale images: 5 Predators in the first row and corresponding 5 Preys in the second row.

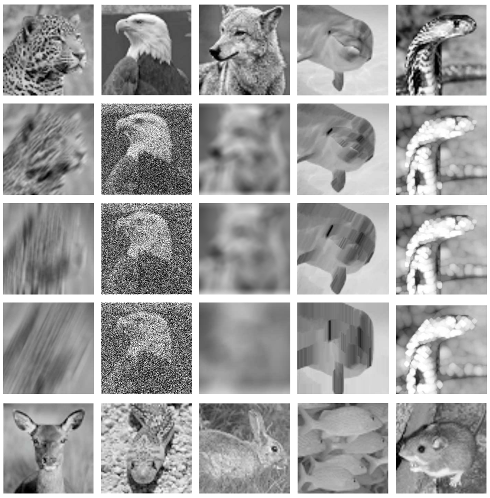


Fig. 5. The exemplar input image patterns are shown in the 1st row. The 2nd through the 4th column below a given predator show the increase in the noise level or image corruption of the predator as discussed in the text. The bottom row illustrates the DLAMs recall performance when presented with a noisy predator image above the prey.

In the above two experiments, the threshold T for the activation function given in eqn. 7 was set to $T = \infty$; i.e, f_A was simply the identity function. With this threshold, the DLAM performance is very impressive in that associations can be recalled even at 99% random noise levels of the input data. However, images with such high and even lower noise levels of corruption cannot be identified by a human observer when not first shown the original pattern images. This poses the problem of misidentifying intruders. For example, suppose we let $\mathbf{x} \in \mathbb{R}^{70225}$ be obtained from a 265×265 image of a horse and present the DLAM with $\mathbf{x}$ as input. If $T = \infty$, then the DLAM will find the closest L^1-distance to one of the stored images and will associate the horse with one of the predators and correlate it with the predator's prey. To avoid intruders, a threshold $T < \infty$ can usually be determined that avoids misclassification of intruders. In image data (such as shown here) with random noise levels in excess of 60%, most images cannot be recognized by a human observer – the best visual pattern recognizer – when not first shown the corresponding non-noisy exemplar. Thus, if $\bar{x}^j$ represent exemplar x^j corrupted by about 60% of random noise, then setting $T_j = d_1\left(x^j, \bar{x}^j\right)$ and $T = \frac{1}{k}\sum_{j=1}^{k} d_1\left(x^j,\, \bar{x}^j\right)$ will, generally, present intruders be recognized as a legitimate exemplars. The next example supports this assumption.

Experiment 3 The dataset is the same as in Example 2. The recall of up to 99% of "salt and pepper" noise is perfect just as in Example 1. We consider the response of the DLAM to a new image pattern $\mathbf{x}$ which is not an element of X, namely the horse image of size 265×265 pixels shown in the last column of Figure 6.

If we present the image pattern x with the predator image that is closest (in the L^1-distance) to the horse and will, therefore, recall the prey associated with this predator. In this specific case the nearest predator is the leopard as can be ascertained from Table 1. Thus, the deer will be associated with the horse when the horse is used as input to the DLAM.

Note that a human observer will have extreme difficulty in identifying any of the images shown in Figure 6 if not shown the true exemplars first. Recognition at a noise level of 70% becomes pure guess work. Computing $T_j = d_1\left(x^j, \bar{x}^j\right)$ for each j and each noise level as well as $d_1\left(x^j,\, x\right)$, we can see from Table 1 that $d_1\left(x^1,\, x\right) = 5667$, where $x^1 = \mathit{leopard}$ and $x = \mathit{horse}$, and $T = \frac{1}{5}\sum\limits_{j=1}^{5} T^j = 5637$ when $\bar{x}^j$ represents as 63% corruption of x^j. Thus, T eliminates x as an intruder. Hence, using $T = 5376$ ($\bar{x}^j$ representing 60% corruption of x^j) would be an even better choice for preventing other intruders.

Noise	0%	50%	60%	63%	65%	70%	80%	90%	100%	Horse
Leopard	0	4470	5374	5634	5813	6297	7158	8066	8932	**5667**
Eagle	0	4492	5348	5626	5844	6252	7154	8080	8947	6293
Wolf	0	4484	5396	5663	5832	6265	7177	8051	8965	6367
Dolphin	0	4452	5385	5640	5816	6281	7162	8059	8952	6713
Cobra	0	4487	5377	5621	5801	6292	7147	8052	8946	6189
Average	0	4477	5376	**5637**	5821	6277	7160	8062	8948	6246

Table 1. The distance ($\times 10^3$) between original predator image and the corrupted image with 50%, 60%, 63%, 65%, 70%, 80%, 90% and 100% of "salt and pepper" noise. The last column has the distance to the "horse" image shown in Figure 6.

5 Conclusions

We present a new hetero-associative lattice memory based on dendritic computing. We report experimental results showing that this memory exhibits extreme robustness in the presence of various types of noise. It is our opinion that this DLAM is superior to existing hetero-associative memories. Further work will be addressed to perform exhaustive comparison tests with other associative memory architectures in order to rigorously verify our opinions.

References

1. Steinbuch, K.: *Automat und Mensch*, Second Edition, Springer Verlag, Heidelberg (1963).

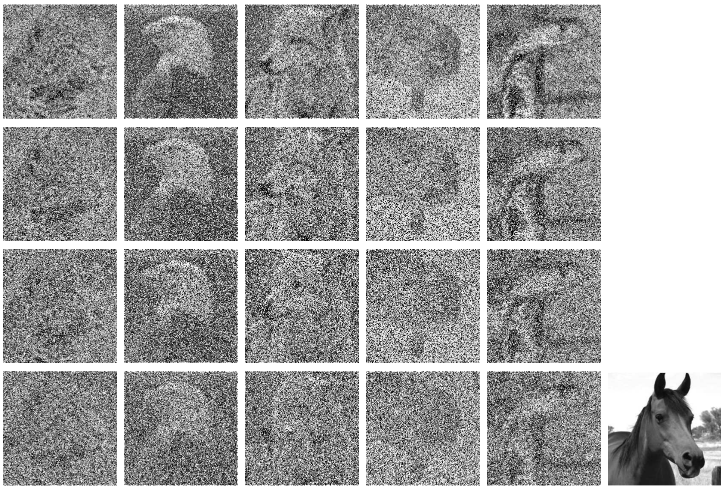


Fig. 6. Grayscale images from Experiment 3. The 1st, 2nd, and 3rd rows presents the input predator images corrupted with 50%, 60%, 63% "salt and pepper" noise. The 4th row contains corrupted images with the noise parameter set to 70%. These images are at the same distance from the original images as image "horse" in the last column.

2. Steinbuch, K., Piske, U.A.W.: Learning Matrices and Their Applications, IEEE Trans. on Electronic Computers, 846–862 (1963).
3. Steinbuch, K.: *Automat und Mensch*, Third Edition, Springer Verlag, Heidelberg (1965).
4. Steinbuch, K.: *Automat und Mensch*, Fourth Edition, Springer Verlag, Heidelberg (1972).
5. Kohonen, T.: Correlation Matrix Memory. IEEE Trans. on Computers C-21, 353–359 (1972).
6. Anderson, J.A.: A simple neural network generating an interactive memory, Mathematical Biosciences 14, 197–220 (1972).
7. Kohonen, T.: *Self-Organization and Associative Memories*, 2nd ed., Springer-Verlag, Berlin (1987).
8. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. of the National Academy of Sciences, USA, 79, 2554–2558 (1982).
9. Hopfield, J.J.: Neurons With Graded Response Have Collective Computational Properties Like Those of Two State Neurons. Proc. of the National Academy of Sciences, USA, 81, 3088–3092 (1984).
10. Hopfield J.J., Tank, D.W.: Computing with neural circuits. Science 233, 625–633 (1986).
11. G. X. Ritter and P. Sussner. Associative Memories Based on Lattice Algebra. in *IEEE Inter. Conf. Systems, Man, and Cybernetics.*, Orlando, FL, October 1997, pp. 3570–3575.
12. Ritter, G.X., Sussner, P., Diaz de Leon, D.L.: Morphological Associative Memories, IEEE Trans. on Neural Networks, vol. 9,, 281–293 (1998).

13. Ritter, G.X., Diaz de Leon, D.L., Sussner, P.: Morphological Bidirectional Associative Memories, Neural Networks, vol. 12, 851–867 (1999).
14. Ritter, G.X., Urcid, G.: Lattice Algebra Approach to Single-Neuron Computation. IEEE Trans. on Neural Networks, 14(2), 282–295 (2003).
15. Kaburlasos, V.G.: Towards a Unified Modeling and Knowledge Representation Based on Lattice Theory, Computational Inteligence 27, Springer, Heidelberg, Germany (2006).
16. Kaburlasos, V.G., Ritter, G.X. (Eds): Computational Intelligence Based on Lattice Theory, Studies in Computational Intelligence 67, Springer, Heidelberg, Germany (2007).
17. Ritter, G.X., Urcid, G.: Lattice Algebra Approach to Endmember Determination In Hyperspectral Imagery. Chapter 4 in P. Hawkes (Ed.).Advances in Imaging and Electron Physics, vol. 169, Elsevier, San Diego, CA, 113–168 (2010).
18. Kaburlasos, V.G.: Granular Enhancement of Fuzzy-ART/SOM Neural Classifyers Based on Lattice Theory. In: Computational Intelligence based on Lattice Theory. Studies in Computational Intelligence, vol. 67, Kaburlasos V.G., Ritter, G.X., (Eds.). Springer Verlag, Heidelberg, 3–23, (2007).
19. Graña, M.,Villaverde, I., Moreno, R., Albizuri, F.X.: Convex Coordinates from Lattice Independent Sets of Visual Pattern Recognition. In: Studies in Computational Intelligence, vol. 67, Kaburlasos V.G., Ritter, G.X., (Eds.). Springer Verlag, Heidelberg, 101–128, (2007).
20. Graña, M., Chyzhyk, G., Garca-Sebastin, MT., Hernandez, C., Lattice Independent Component Analysis for functional Magnetic Resonance Imaging, Information Sciences 181:1910-1928 (2011)
21. Chyzhyk, D., Graña, M.: Optimal Hyperbox Shrinking in Dendritic Computing Applied to Alzheimers Disease Detection in MRI. *Conference on Soft Computing Models in Industrial and Environmental Applications* (SOCO 2011), Salamanca, Spain. 6-8th April, 2011. Advances in Intelligent and Soft Computing, vol. 87, 543–550, (2011)
22. Chyzhyk, D., Graña, Savio,A., Maiora, J., Hybrid Dendritic Computing with Kernel-LICA applied to Alzheimers Disease detection in MRI, Neurocomputing, 2012, 75(1):72-77
23. Ritter, G.X., Urcid, G.: Perfect Recovery from Noisy Input Patterns with a Dendritic Lattice Associative Memory. Proceedings of the International Joint Conference on Neural Networks (IEEE/INNS), San Jose, CA, 503–510 (2011)
24. Urcid, G., Ritter, G.X., Valvdiviezo, J.C.N.: Grayscale Image Recall from Imperfect Inputs with a Two Layer Dendritic Lattice Associative Memory. Proceedings of IEEE, 268-273, 3rd Congress on Nature and Biologically Inspired Computing, Salamanca, Spain, 268–273 (2011).
25. Ritter, G.X., Urcid, G.: Learning in Lattice Neural Networks that Employ Dendritic Computing. In: Computational Intelligence based on Lattice Theory. Studies in Computational Intelligence, vol. 67, Kaburlasos V.G., Ritter, G.X., (Eds.). Springer Verlag, Heidelberg, 25–44 (2007).