Properties of CLARION-H

Sébastien Hélie & Ron Sun

Cognitive science department, Rensselaer Polytechnic Institute, Troy, NY, USA.

Abstract

CLARION's latest implementation, CLARION-H, uses Hebbian learning to build all its connection matrices in the NACS. In this technical report, we present a brief overview of the implementation, along with two theorems and two lemmas that present useful properties of Hebbian learning.

This technical report is a description of CLARION-H's Non-Action-Centered-Subsystem in a matrix algebra form. The first subsection describes the connection matrices, the information flow, and the decision. The last subsection lists some mathematical properties of the learning algorithm.

CLARION-H's Non-Action-Centered Subsystem

The general architecture of the model is shown in Figure 1. As can be seen, the top level of the Non-Action-Centered Subsystem (NACS) encodes the explicit knowledge using a linear two-layer connectionist network (similar to the previous implementations of CLARION; see Sun, 2003). However, implicit knowledge in the bottom level of the NACS is modeled by an attractor neural network (Anderson et al., 1977; Hopfield, 1982; this implementation is different from the previous). Hence, all the associations are symmetric and learned using Hebbian learning. The activation resulting from the integration of the outputs of explicit and implicit processing follows a Boltzmann distribution.

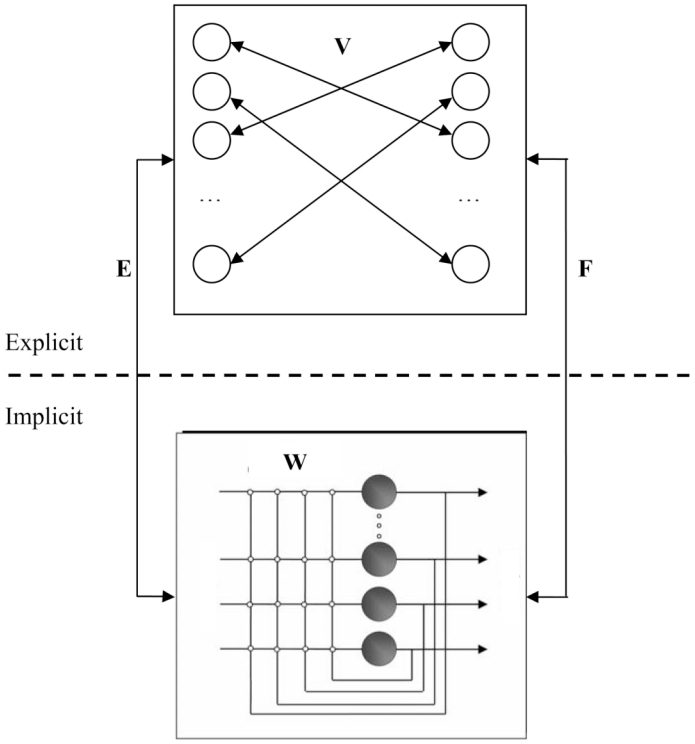

Figure 1. General architecture of CLARION-H's NACS. The letters refer to the connection matrices.

With regard to the NACS, the environment activates the model in one of three ways: 1) top level only, 2) bottom level only or, 3) both top and bottom levels. The general case happens when a perceptual stimulus is presented and there is enough time to retrieve the knowledge associated with the stimulus in both levels: both levels are activated. However, when a perceptual stimulus is presented to the model and the explicit knowledge associated with the stimulus cannot be retrieved (e.g., it is non-existent, there is insufficient time, etc.), only the bottom level is activated. Lastly, it is theoretically possible that mostly the top level is activated when the stimulus is highly abstract and explicit (although somewhat unlikely in everyday situations).

Regardless of input, many top-level representations are redundantly encoded in the bottom level. Hence, if only the top-level representation is activated, a top-down signal is sent to activate the corresponding representation in the bottom level (implicitation). Likewise, stimuli that uniquely activate the bottom level send a bottom-up signal to activate the top level (explicitation). Hence, a stimulus is often processed

both in the top and bottom levels. The following equations formalize the above discussion.

In the top level, explicit knowledge is localistically represented using binary vectors $\mathbf{x}_i = \{0, 1\}^n$, $\mathbf{y}_j = \{0, 1\}^m$, and $\|\mathbf{x}_i\| = \|\mathbf{y}_j\| = 1$ (in Figure 1, $\mathbf{x}_i$ and $\mathbf{y}_j$ are vectors describing the activation in the left and right layers of the top level respectively), and $\|\bullet\|$ is the Euclidean norm[1]. These layers are connected using the matrix $\mathbf{V}$, which was trained to encode the explicit associations using standard Hebbian learning (Kohonen, 1972):

$$\mathbf{V} = \sum_i \mathbf{y}_i \mathbf{x}_i^{\mathrm{T}} \tag{1}$$

where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_k\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_k\}$ are the sets containing the stimuli ($k \leq n$ and $k \leq m$), $\mathbf{V} = [v_{ij}]$ is the Hebbian matrix containing the associations, and $\mathbf{x}_i$ is associated to $\mathbf{y}_i$. The use of Hebbian learning to encode the associations ensures that $v_{ij} = 1$ if $\mathbf{x}_i$ is associated to $\mathbf{y}_j$ and zero otherwise (Kohonen, 1972). Using this matrix, the top-level associations can be exactly retrieved using the following linear transmission rules (see Theorem 1 and Lemma 2 below):

$$\mathbf{y}_i = \left(\mathbf{V}\mathbf{N}_1\right)\mathbf{x}_i \tag{2}$$

$$\mathbf{x}_i = \left(\mathbf{V}^{\mathrm{T}}\mathbf{N}_2\right)\mathbf{y}_i \tag{3}$$

where $\mathbf{N_1}$ and $\mathbf{N_2}$ are defined as:

$$\mathbf{N}_1 = \begin{pmatrix} \left\|\mathbf{v}_1\right\|^{-2} & 0 & \ldots & 0 \\ 0 & \left\|\mathbf{v}_2\right\|^{-2} & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \left\|\mathbf{v}_n\right\|^{-2} \end{pmatrix} \qquad \mathbf{N}_2 = \begin{pmatrix} \left\|\mathbf{v}_1^{\mathrm{T}}\right\|^{-2} & 0 & \ldots & 0 \\ 0 & \left\|\mathbf{v}_2^{\mathrm{T}}\right\|^{-2} & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \left\|\mathbf{v}_m^{\mathrm{T}}\right\|^{-2} \end{pmatrix} \tag{4}$$

where $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ is the matrix containing the associations (same as in Eq. 1).

---

[1] Mathematically, these constraints ensure that the patterns of activation in each layer are forming orthonormal bases. This is necessary for the application of all the theorems presented in the following section.

The $\mathbf{N_1}$ and $\mathbf{N_2}$ matrices are used to normalize the activation so that a node's activation cannot be higher than one. To do that, the number of associates of each node must be determined, and used to divide the summed activation (the dot product). This number can be obtained by counting the number of non-zero elements in each row or in each column of the **V** matrix. Because the **V** matrix is binary, this can be computed using the squared norm. As a result, if proportion p of $\mathbf{y}_j$'s associates are activated in $\mathbf{x}_i$, the activation of $\mathbf{y}_j$ is p.

In CLARION-H's NACS, the bottom level has been implemented using *NDRAM* (Chartier & Boukadoum, 2006; Chartier & Proulx, 2005). Each top-level association is redundantly encoded using a random vector $\mathbf{z}_i = \mathbf{t_{1}}_i + \mathbf{t_{2}}_i$, where $\mathbf{t_{1}}_i = \{-1, 1\}^s \cup \{0\}^{r-s}$ is a vector representing the first s units of the bottom-level, which are connected to the top-level's left layer using matrix **E**, while $\mathbf{t_{2}}_i = \{0\}^s \cup \{-1, 1\}^{r-s}$ is a vector representing the remaining $r - s$ units, which are connected to the right layer in the top level using matrix **F** (see Figure 1).

The **E** and **F** matrices are trained to encode the associations between the top and bottom level representations using the same Hebbian rule as **V**:

$$\mathbf{E} = \sum_i \mathbf{t_1}_i \mathbf{x}_i^{\mathbf{T}} \tag{5}$$

$$\mathbf{F} = \sum_j \mathbf{t_2}_j \mathbf{y}_j^{\mathbf{T}} \tag{6}$$

where $\mathbf{T_1} = \{\mathbf{t}_{11}, \mathbf{t}_{12}, \ldots, \mathbf{t}_{1k}\}$ and $\mathbf{T_2} = \{\mathbf{t}_{21}, \mathbf{t}_{22}, \ldots, \mathbf{t}_{2k}\}$ are the sets containing the distributed representations. Hence, if a stimulus is presented to the left layer of the top level (by Theorem 1 below):

$$\begin{aligned} \mathbf{z}_i &= \mathbf{E}\mathbf{x}_i + \mathbf{residual} \\ &= \mathbf{t_1}_i + \mathbf{residual} \end{aligned} \tag{7}$$

where $\mathbf{z}_i$ is the bottom-level activation and **residual** is the final state of the bottom level at the end of the previous iteration of processing. If a stimulus activated the right layer in the top level (by Theorem 1 below):

$$\begin{aligned} \mathbf{z}_i &= \mathbf{F}\mathbf{y}_i + \mathbf{residual} \\ &= \mathbf{t}_{2i} + \mathbf{residual} \end{aligned} \tag{8}$$

The distributed representation of the stimulus ($\mathbf{z}_i$) is transmitted in the bottom level using the following non-linear transmission rule (Chartier & Proulx, 2005):

$$\mathbf{z}_{i[t+1]} = f(\mathbf{W}\mathbf{z}_{i[t]}) \tag{9}$$

$$f(a) = \begin{cases} +1 & , a > 1 \\ (\delta+1)a - \delta a^3, & -1 \le a \le 1 \\ -1 & , a < -1 \end{cases} \tag{10}$$

where $\mathbf{z}_{i[t]}$ is the distributed representation after t spins in the network (there is a total of p spins per trial), $\delta < 0.5$ is the slope of the transmission function, a is the activation of a bottom-level unit, and $\mathbf{W}$ is the weight matrix pre-trained to encode the implicit associations using a contrastive Hebbian learning rule (Chartier & Proulx, 2005):

$$\mathbf{W}_{[t]} = \xi\mathbf{W}_{[t-1]} + \eta\left(\mathbf{z}_{i[0]}\mathbf{z}_{i[0]}^{\mathrm{T}} - \mathbf{z}_{i[p]}\mathbf{z}_{i[p]}^{\mathrm{T}}\right) \tag{11}$$

where $\mathbf{W}_{[t]}$ is the weight matrix at trial t, $0 << \xi \le 1$ is a memory efficiency parameter, and $\eta < \frac{1}{2(1-2\delta)r}$ is a general learning parameter (for a demonstration, see Chartier & Boukadoum, 2006). Following Sun (2003), we assume that each spin in the bottom level takes roughly 350 ms of psychological time.

Once the bottom-level processing is completed, the information is sent bottom-up using the following equations. If the initial stimulus first activated the left layer in the top level, the bottom-up activation is transmitted to the right layer of the top level:

$$\mathbf{y}_{[bottom-up]} = \left(\mathbf{F}^{\mathrm{T}}\mathbf{N}_3\right)\mathbf{z}_{i[p]} \tag{12}$$

where $\mathbf{y}_{[bottom\text{-}up]}$ is the bottom-up signal sent to the right layer in the top level and $\mathbf{z}_{i[p]}$ is the bottom-level activation after p spins.

Otherwise, if the initial stimulus first activated the right layer in the top level, the bottom-up activation is transmitted to the left layer:

$$\mathbf{x}_{[bottom-up]} = \left(\mathbf{E}^{\mathbf{T}}\mathbf{N}_4\right)\mathbf{z}_{i[p]} \tag{13}$$

where $\mathbf{x}_{[bottom\text{-}up]}$ is the bottom-up signal sent to the left layer in the top level. $\mathbf{N_3}$ and $\mathbf{N_4}$ are the following square diagonal matrices:

$$\mathbf{N}_3 = \begin{pmatrix} \left\|\mathbf{f}_1^{\mathbf{T}}\right\|^{-2.2} & 0 & \ldots & 0 \\ 0 & \left\|\mathbf{f}_2^{\mathbf{T}}\right\|^{-2.2} & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \left\|\mathbf{f}_r^{\mathbf{T}}\right\|^{-2.2} \end{pmatrix}$$
$$\mathbf{N}_4 = \begin{pmatrix} \left\|\mathbf{e}_1^{\mathbf{T}}\right\|^{-2.2} & 0 & \ldots & 0 \\ 0 & \left\|\mathbf{e}_2^{\mathbf{T}}\right\|^{-2.2} & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & \left\|\mathbf{e}_r^{\mathbf{T}}\right\|^{-2.2} \end{pmatrix} \tag{14}$$

where $\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_r\}$ and $\mathbf{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_r\}$ are the matrices linking the top and bottom representations (see Eq. 5 and Eq 6). Like $\mathbf{N_2}$, the $\mathbf{N_3}$ and $\mathbf{N_4}$ matrices are counting the number of non-zero elements in each column of matrices **F** and **E** so that if a node in the top level is associated to d units in the bottom level, its total activation is divided by $d^{1.1}$. The exponent, which is not present in top-level activation, was included to capture similarity-based reasoning (Sun, 2003).

Once the bottom-up activation has reached the top level, it is integrated with the activation already present using the *Max* function. Hence, if the initial stimulus activated the left layer of the top level, the integrated activation vector is located in the right layer of the top level:

$$\forall_j,\ 1 \le j \le m:\ \ y_{[integrated]\,j} = Max\left[\left(y_{ij}, \lambda \times y_{[bottom-up]\,j}\right]\right. \tag{15}$$

where $\mathbf{y}_{[integrated]} = \{y_{[integrated]1}, y_{[integrated]2}, \ldots, y_{[integrated]m}\}$ is the integrated activation vector, $\mathbf{y}_i = \{y_{i1}, y_{i2}, \ldots, y_{im}\}$ is the top-level activation (Eq. 2), $\mathbf{y}_{[bottom\text{-}up]} = \{y_{[bottom\text{-}up]1},$

$y_{[bottom\text{-}up]2}, \ldots, y_{[bottom\text{-}up]m}\}$ is the bottom-up activation (Eq. 12), and λ is a scaling parameter that determines how implicit the task is. Likewise, if the initial stimulus activated the right layer in the top level, the integrated activation vector is located in its left layer:

$$\forall_j,\ 1 \le j \le n:\ \ x_{[integrated]\,j} = Max\left[x_{ij}, \lambda \times x_{[bottom-up]\,j}\right] \tag{16}$$

where $\mathbf{x}_{[integrated]} = \{x_{[integrated]1}, x_{[integrated]2}, \ldots, x_{[integrated]n}\}$ is the integrated activation vector, $\mathbf{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{in}\}$ is the top level activation (Eq. 3), and $\mathbf{x}_{[bottom\text{-}up]} = \{x_{[bottom\text{-}up]1}, x_{[bottom\text{-}up]2}, \ldots, x_{[bottom\text{-}up]n}\}$ is the bottom-up activation (Eq. 13).

In all cases, the integrated activation vector is further transformed into a Boltzmann probability distribution:

$$P(y_{[\text{integrated}]i}) = \frac{e^{y_{[\text{integrated}]i}/\alpha}}{\sum_j e^{y_{[\text{integrated}]j}/\alpha}}, \quad \text{or}$$
$$P(x_{[\text{integrated}]i}) = \frac{e^{x_{[\text{integrated}]i}/\alpha}}{\sum_j e^{x_{[\text{integrated}]j}/\alpha}} \tag{17}$$

where α is the temperature (randomness parameter). In CLARION-H's NACS, each $y_{[integrated]i}$ and each $x_{[integrated]i}$ represents a hypothesis.

The statistical mode of the Boltzmann distribution is used to estimate the model's internal confidence level (*ICL*) and a hypothesis is stochastically chosen. If the *ICL* is higher than a predetermined threshold (ψ), the chosen hypothesis is output to motor control; else, the reasoning process continues with the chosen hypothesis as the top-level stimulus (Eq. 2 or Eq. 3) and $\mathbf{z}_{i[p]}$ as the residual activation in the bottom level (Eq. 7 or Eq. 8). If a hypothesis is output, the reaction time of the model is computed as follow:

$$RT = a - b \times ICL \tag{18}$$

where a and b are the maximum response time and the slope respectively. The algorithm is summarized in Table 1.

Table 1. Algorithm of CLARION-H's NACS

1. Observe the current state of the environment (in the bottom level, and possibly send to the top level);
2. Simultaneously transmit the observed information in both levels;
3. Compute the integrated activation vector and the Boltzmann distribution;
4. Stochastically choose a hypothesis and estimate the *ICL* using the mode of the Boltzmann distribution:
    a. If the *ICL* is higher than ψ, output the chosen hypothesis to motor control;
    b. Else, if there is time, go back to step 2 and use the chosen hypothesis as the input;
5. Compute the reaction time of the model.

Mathematical properties of weight matrices

This subsection introduce theorems demonstrating the properties of Hebbian learning used in the present model to train matrices **V**, **E**, and **F**. All proofs concerning the convergence of the **W** matrix can be found in Chartier & Proulx (2005) and Chartier & Boukadoum (2006); they are not repeated here.

All the theorems below use the following notation and assume these constraints:

$$\forall_{i,j},\ 1 \le i \le k,\ 1 \le j \le k,\ i \ne j: \quad \mathbf{a}_i \in \{0,1\}^n,\ \mathbf{a}_i^{\mathrm{T}}\mathbf{a}_i = 1,\ \mathbf{a}_i^{\mathrm{T}}\mathbf{a}_j = 0,\ \mathbf{b}_i \in R^m, \|\mathbf{b}_i\| = \|\mathbf{b}_j\|,\ \mathbf{M} \in R^{m \times n} \tag{19}$$

where $\|\bullet\|$ is the Euclidean norm, $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_{2,\ \ldots,}\ \mathbf{a}_k\}$ and $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_{2,\ \ldots,}\ \mathbf{b}_k\}$ are the sets containing the k different activation patterns, $n \ge k$ is the dimensionality of $\mathbf{a}_i$, $m \ge k$ is the dimensionality of $\mathbf{b}_i$, and **M** is a coefficient matrix. In words, the following theorems assume that the $\mathbf{a}_i$s are unit orthogonal binary vectors, and that the lengths of all the $\mathbf{b}_i$s are equal.

M is a regular Hebbian matrix built using the outer matrix product (Kohonen, 1972):

$$\mathbf{M} = \sum_k \mathbf{b}_k \mathbf{a}_k^{\mathrm{T}} \tag{20}$$

where $\mathbf{b}_k$ is associated to $\mathbf{a}_k$.

Theorem 1

For all vector sets and matrices that meet the constraints expressed by Eq. 19 and Eq. 20, 'forward' processing is exact, i.e:

$$\forall_i,\ 1 \le i \le k : \mathbf{M}\mathbf{a}_i = \mathbf{b}_i \tag{21}$$

where $\mathbf{b}_i$ is associated to $\mathbf{a}_i$.

Proof

$$\begin{aligned} \mathbf{M}\mathbf{a}_i &= \left(\sum_k \mathbf{b}_k \mathbf{a}_k^{\mathrm{T}}\right)\mathbf{a}_i \\ &= \sum_k \mathbf{b}_k \left(\mathbf{a}_k^{\mathrm{T}}\mathbf{a}_i\right) \\ &= \mathbf{b}_1\left(\mathbf{a}_1^{\mathrm{T}}\mathbf{a}_i\right) + \mathbf{b}_2\left(\mathbf{a}_2^{\mathrm{T}}\mathbf{a}_i\right) + \ldots + \mathbf{b}_k\left(\mathbf{a}_k^{\mathrm{T}}\mathbf{a}_i\right) \\ &= \mathbf{b}_i \end{aligned} \tag{22}$$

because $\mathbf{a}_j^{\mathrm{T}}\mathbf{a}_i = 1$ if and only if $i = j$ and 0 otherwise (by Eq. 19). ■

Explanation

This theorem applies to forward processing in the top level (through the **V** matrix; Eq. 2) and top-down activation (through the **E** and **F** matrices; Eq. 7 and Eq. 8).

Theorem 2

For all vector sets and matrices that meet the constraints expressed by Eq. 19 and Eq. 20, 'backward' processing is proportional to the correlation between the activation pattern and all the memorized patterns, i.e:

$$\forall_i,\ 1 \le i \le k :\ \mathbf{M}^{\mathrm{T}}\mathbf{b}_i \propto \mathbf{a}_1\rho_{\mathbf{b}_1\mathbf{b}_i} + \mathbf{a}_2\rho_{\mathbf{b}_2\mathbf{b}_i} + \ldots + \mathbf{a}_k\rho_{\mathbf{b}_k\mathbf{b}_i} \tag{23}$$

where ρ denotes the correlation.

Proof

$$\begin{aligned}
\mathbf{M}^{\mathbf{T}}\mathbf{b}_i &= \left(\sum_k \mathbf{b}_k \mathbf{a}_k^{\mathbf{T}}\right)^{\mathbf{T}} \mathbf{b}_i \\
&= \left(\sum_k \left(\mathbf{a}_k^{\mathbf{T}}\right)^{\mathbf{T}} \mathbf{b}_k^{\mathbf{T}}\right)\mathbf{b}_i \\
&= \sum_k \mathbf{a}_k \left(\mathbf{b}_k^{\mathbf{T}}\mathbf{b}_i\right) \\
&= \sum_k \mathbf{a}_k \|\mathbf{b}_k\| \|\mathbf{b}_i\| Cos\left(\alpha_{\mathbf{b}_k\mathbf{b}_i}\right) \\
&= \|\mathbf{b}_i\|^2 \sum_k \mathbf{a}_k Cos\left(\alpha_{\mathbf{b}_k\mathbf{b}_i}\right) \\
&\propto \mathbf{a}_1\rho_{\mathbf{b}_1\mathbf{b}_i} + \mathbf{a}_2\rho_{\mathbf{b}_2\mathbf{b}_i} + \ldots + \mathbf{a}_k\rho_{\mathbf{b}_k\mathbf{b}_i}
\end{aligned} \tag{24}$$

where $\alpha_{\mathbf{b}_k\mathbf{b}_i}$ is the angle between $\mathbf{b}_k$ and $\mathbf{b}_i$. Because $\mathbf{a}_j$ has a 1 in position l and 0 everywhere else, the activation in the lth position of the resulting vector is proportional to the correlation between $\mathbf{b}_i$ and $\mathbf{b}_j$ (where $\mathbf{b}_j$ is associated to $\mathbf{a}_j$ by Eq. 20). This simplification is possible because the lengths of all the $\mathbf{b}_i$s are equal and the $\mathbf{a}_i$s are orthonormal binary vectors (Eq. 19). ■

Explanation

This theorem applies to bottom-up transmission in the $\mathbf{E}^{\mathbf{T}}$ and $\mathbf{F}^{\mathbf{T}}$ matrices and ensures a rudimentary type of similarity-based reasoning (Eq. 12 and Eq. 13).

Lemma 1

For all matrices and vectors that follow the constraints described by Eq. 19 and Eq. 20, exact 'backward' processing can be achieved by applying a simple squashing function:

$$\delta\left(\mathbf{M}^{\mathbf{T}}\mathbf{b}_i, Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) = \mathbf{a}_i \tag{25}$$

where $\mathbf{a}_i$ is associated to $\mathbf{b}_i$ and $\delta(\mathbf{i}, j) = \{\delta(i_1, j), \delta(i_2, j), \ldots, \delta(i_n, j)\}$ is the Kronecker-Delta function applied to each element of $\mathbf{i}$.

Proof

$$\begin{aligned}\delta\left(\mathbf{M}^{\mathbf{T}}\mathbf{b}_i, Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) &= \delta\left(\left\|\mathbf{b}_i\right\|^2 \sum_k \mathbf{a}_k Cos\left[\alpha_{\mathbf{b}_k\mathbf{b}_i}\right], Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) \\ &= \delta\left(\left\|\mathbf{b}_i\right\|^2 \mathbf{a}_1 Cos\left[\alpha_{\mathbf{b}_1\mathbf{b}_i}\right], Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) + \\ &\quad \delta\left(\left\|\mathbf{b}_i\right\|^2 \mathbf{a}_2 Cos\left[\alpha_{\mathbf{b}_2\mathbf{b}_i}\right], Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) + \\ &\quad \ldots + \delta\left(\left\|\mathbf{b}_i\right\|^2 \mathbf{a}_k Cos\left[\alpha_{\mathbf{b}_k\mathbf{b}_i}\right], Max\left[\mathbf{M}^{\mathbf{T}}\mathbf{b}_i\right]\right) \\ &= \mathbf{a}_i\end{aligned} \tag{26}$$

This lemma is a direct consequence of Theorem 2, because a Cosine has a maximum at 0 (i.e., when the angle is null, when $i = k$). Hence, the ith term in the summation is a vector with a 1 in position l and 0 everywhere else (because $\mathbf{a}_i$ has a 1 in position l and 0 everywhere else). All the remaining terms in the summation are null vectors (**0**), because of the application of the Kronecker-Delta function. This equality is made possible by Eq. 19, which ensures that the $\mathbf{a}_i$s are orthonormal binary vectors and that the lengths of all the $\mathbf{b}_j$s are equals. ■

Explanation

This Lemma was not used in CLARION-H's NACS but can be very useful in future simulations if the maximally activated node is to be chosen (instead of using stochastic selection in a Boltzmann distribution; see Eq. 17).

Lemma 2

For all vector sets and matrices that meet the constraints expressed by Eq. 19 and Eq. 20, the addition of an orthonormality constraint on the vectors forming set **B** renders 'backward' processing exact, i.e:

$$\forall_{i,j} : i \neq j, \mathbf{b}_i^{\mathbf{T}}\mathbf{b}_i = 1, \mathbf{b}_i^{\mathbf{T}}\mathbf{b}_j = 0 \Rightarrow \mathbf{M}^{\mathbf{T}}\mathbf{b}_i = \mathbf{a}_i \tag{27}$$

where $\mathbf{a}_i$ is associated to $\mathbf{b}_i$.

Proof

$$\forall_{i,j} : i \neq j, \mathbf{b}_i^{\mathrm{T}}\mathbf{b}_i = 1, \mathbf{b}_i^{\mathrm{T}}\mathbf{b}_j = 0 \Rightarrow \mathbf{M}^{\mathrm{T}}\mathbf{b}_i = \sum_k \mathbf{a}_k\left(\mathbf{b}_k^{\mathrm{T}}\mathbf{b}_i\right)$$
$$= \mathbf{a}_1\left(\mathbf{b}_1^{\mathrm{T}}\mathbf{b}_i\right) + \mathbf{a}_2\left(\mathbf{b}_2^{\mathrm{T}}\mathbf{b}_i\right) + \ldots + \mathbf{a}_k\left(\mathbf{b}_k^{\mathrm{T}}\mathbf{b}_i\right) \qquad (28)$$
$$= \mathbf{a}_i$$

This lemma is a direct consequence of the newly introduced orthonormality constraint, which ensure that $\mathbf{b}_i^{\mathrm{T}}\mathbf{b}_j = 1$ if $i = j$ and 0 otherwise (similar to Theorem 1). ■

Explanation

Lemma 2 applies to backward processing in the top level through the $\mathbf{V}^{\mathrm{T}}$ matrix (Eq. 3).

References

Anderson, J.A., Silverstein, J.W., Ritz, S.A., & Jones, R.S. (1977). Distinctive features, categorical perception, and probability learning: Applications of a neural model. *Psychological Review*, *84*, 413-451.

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

Chartier, S. & Proulx, R. (2005). NDRAM: A Nonlinear Dynamic Recurrent Associative Memory for learning bipolar and nonbipolar correlated patterns. *IEEE Transactions on Neural Networks*, *16*, 1393-1400.

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

Kohonen, T. (1972). Correlation matrix memories. *IEEE Transactions on Computers C, 21*, 353-359.

Sun, R. (2003). *A Tutorial on CLARION 5.0.* Technical Report, Cognitive Science Department, Rensselaer Polytechnic Institute.