

Excerpts from:

**CORTICAL LEARNING OF RECOGNITION CATEGORIES:
A RESOLUTION OF THE EXEMPLAR VS. PROTOTYPE DEBATE**

Gregory P. Amis, Gail A. Carpenter, Bilgin Ersoy, Stephen Grossberg

Department of Cognitive and Neural Systems
Boston University
677 Beacon Street
Boston, MA, 02215

[gamis, gail, steve] @cns.bu.edu

(617) 353-7858
Fax: (617) 353-7755

March, 2009

Technical Report CAS/CNS TR-2009-002
Boston, MA: Boston University

Corresponding author: Stephen Grossberg

Acknowledgements: This research was supported by the SyNAPSE program of the Defense Advanced Projects Research Agency (Hewlett-Packard Company, DARPA prime contract HR0011-09-3-0001 and HRL Laboratories LLC, subcontract #801881-BS under DARPA prime contract HR0011-09-C-0011) and by the Science of Learning Centers program of the National Science Foundation (NSF SBE-0354378).

APPENDIX: Distributed ARTMAP Algorithm

In the general case, distributed ARTMAP (dARTMAP) learns to predict an arbitrary output vector $\mathbf{b} = (b_1, \dots, b_k, \dots, b_L)$ in response to an input vector $\mathbf{a} = (a_1, \dots, a_i, \dots, a_M)$. The special case of a categorization problem sets one output component $b_K = 1$ and all $b_j = 0$, $j \neq K$, to learn that input \mathbf{a} belongs in the output category K . A complete dARTMAP system may be implemented as a real-time network with local computations (Figure 2). The current algorithm uses only variables that are needed for category learning and performance (Table A1). Table A2 lists system parameters, along with their domains and values used in simulations. See also Section 4 for a heuristic description of important dARTMAP computations.

Table A1

Table A2

A.1. Network activation

dARTMAP network activation begins with the presentation of an M -dimensional input vector \mathbf{a} , with input feature values a_i such that $0 \leq a_i \leq 1$, to the dART_a field F_0 and, during training only, an L -dimensional output training vector \mathbf{b} to field F_0^b (Figure 2). In the 5-4 category structure, $M = 4$ and $L = 2$.

Field F_0 contains $2M$ nodes, indexed by i , whose activation *complement codes* the input pattern:

$$A_i = \begin{cases} a_i & \text{if } 1 \leq i \leq M \\ 1 - a_{i-M} & \text{if } M+1 \leq i \leq 2M \end{cases} \quad (\text{A1})$$

Complement coding captures both the presence of a feature as an ON cell response ($a_i = 1, A_i = 1$) and the absence of a feature as an OFF cell response ($a_i = 0, A_{i+M} = 1$) as possible critical pattern components. (See also Section 4.1.)

Each category coding node j in field F_2 stores a learned categorization pattern in its bottom-up long-term memory (LTM) traces τ_{ij} . Activation of category coding nodes by F_0 is determined jointly by a *tonic* component (independent of F_0 activation \mathbf{A}) and a *phasic* component (dependent on the match of node j to \mathbf{A}). F_2 nodes that best match \mathbf{A} receive the greatest input. In geometric terms (see Section 6.2), the input from F_0 to node j is

$$T_j = M - d(R_j, \mathbf{a}) - \alpha |R_j|, \quad (\text{A2})$$

where $d(R_j, \mathbf{a})$ is the city-block (L_1 norm) distance between category coding box j and input \mathbf{a} , $|R_j|$ is the size of coding box j , and the signal rule parameter α is a fixed positive value. This *choice-by-difference* form of T_j favors nodes with small coding boxes ($|R_j| \cong 0$) that are close to the input \mathbf{a} ($d(R_j, \mathbf{a}) \cong 0$). For small values of the signal rule parameter ($\alpha \cong 0$), T_j is largest when \mathbf{a} is inside R_j , ($d(R_j, \mathbf{a}) = 0$). When \mathbf{a} is inside multiple category boxes, T_j favors the smallest box containing \mathbf{a} .

The category box size $|R_j|$ has the following definition in terms of network dynamics:

$$|R_j| = M - |\mathbf{w}_j|, \quad (\text{A3})$$

where $|\mathbf{w}_j| = \sum_{i=1}^{2M} w_{ij}$, the category coding weight $w_{ij} = 1 - \tau_{ij}$, and τ_{ij} is the bottom-up LTM trace from F_0 node i to F_2 node j (see equations A19 and A25). Similarly, the distance $d(R_j, \mathbf{a})$ equals:

$$\begin{aligned}
d(R_j, \mathbf{a}) &= |R_j \oplus \mathbf{a}| - |R_j| \\
&= (M - |\mathbf{A} \wedge \mathbf{w}_j|) - (M - |\mathbf{w}_j|) \\
&= |\mathbf{w}_j| - |\mathbf{A} \wedge \mathbf{w}_j|,
\end{aligned} \tag{A4}$$

where $|R_j \oplus \mathbf{a}|$ is the size of R_j when expanded to include \mathbf{a} , and the fuzzy minimum vector components $(\mathbf{A} \wedge \mathbf{w}_j)_i = \min\{A_i, w_{ij}\}$. Intuitively, $|R_j \oplus \mathbf{a}| - |R_j|$ measures the amount R_j must expand to include \mathbf{a} , and $|\mathbf{w}_j| - |\mathbf{A} \wedge \mathbf{w}_j|$ is the amount all learned weights w_{ij} must decrease in total in order for $|\mathbf{w}_j^{new}| = |\mathbf{A} \wedge \mathbf{w}_j^{new}|$. Input function T_j can then be separated into phasic and tonic components:

$$\begin{aligned}
T_j &= M - d(R_j, \mathbf{a}) - \alpha |R_j| \\
&= M - (|\mathbf{w}_j| - |\mathbf{A} \wedge \mathbf{w}_j|) - \alpha (M - |\mathbf{w}_j|) \\
&= |\mathbf{A} \wedge \mathbf{w}_j| + (1 - \alpha)(M - |\mathbf{w}_j|) \\
&= S_j + (1 - \alpha)\Theta_j,
\end{aligned} \tag{A5}$$

where the phasic component

$$S_j = |\mathbf{A} \wedge \mathbf{w}_j| = \sum_{i=1}^{2M} (A_i \wedge (1 - \tau_{ij})), \tag{A6}$$

and the tonic component

$$\Theta_j = M - |\mathbf{w}_j| = M - \sum_{i=1}^{2M} (1 - \tau_{ij}) = \sum_{i=1}^{2M} \tau_{ij} - M. \tag{A7}$$

The final set of F_2 input equations is then:

$$\begin{aligned}
\text{Phasic: } S_j &= \sum_{i=1}^{2M} (A_i \wedge (1 - \tau_{ij})) \\
\text{Tonic: } \Theta_j &= \sum_{i=1}^{2M} \tau_{ij} - M \\
\text{Total: } T_j &= S_j + (1 - \alpha)\Theta_j.
\end{aligned} \tag{A8}$$

In order for a category coding node to remain active, it must receive at least as much input as an uncommitted node. An uncommitted node has all bottom-up LTM thresholds $\tau_{ij} \equiv 0$ and thus, by (A8), its input $T^u = \alpha M$. Suppressing the activation of committed nodes with input $T_j < T^u$ prevents learning by their LTM traces when the training pattern would be better represented by recruiting a new category coding node. The set of active category coding nodes Λ is, thus, initially

$$\Lambda \equiv \{j = 1, \dots, C : T_j \geq T^u\}, \tag{A9}$$

where C is the number of committed category coding nodes.

F_2 activation is normally *distributed*: all category coding nodes $j \in \Lambda$ are activated as a function of their input T_j . In a real-time network implementation, F_2 nodes compete with each other in a pattern of on-center excitation and off-surround inhibition. A steady-state implementation approximates this competition using the *increased-gradient content addressable memory (IG-CAM) rule*:

$$y_j = \begin{cases} \frac{1}{\sum_{\lambda \in \Lambda} \left[\frac{M - T_j}{M - T_\lambda} \right]^p} & \text{if } j \in \Lambda \\ 0 & \text{if } j \notin \Lambda \end{cases}, \tag{A10}$$

where the CAM rule power p is a free parameter that determines the amount of contrast enhancement. In particular, by equation (A2), $M - T_j \cong d(R_j, \mathbf{a})$ for small values of the signal rule parameter α . Thus, as the distance between input \mathbf{a} and a single category coding box R_j decreases toward zero, activation y_j increases toward one, all other y_j decrease toward zero, and F_2 approaches a winner-take-all (WTA) activation pattern. If \mathbf{a} is inside multiple boxes, activation is shared amongst those nodes, biased toward the node with the smallest box, and all other node activations are significantly reduced. If \mathbf{a} is outside all category coding boxes, the activation pattern remains at a low contrast. Increasing p increases the degree to which F_2 activation contrast is enhanced in favor of the node with the most input T_j . If one or more nodes exactly match \mathbf{A} (i.e., their category coding boxes are point boxes at \mathbf{a}), then $T_j = M$, and (A10) is computationally undefined. Let

$$\Lambda'' = \{j \in \Lambda: T_j = M\} \quad (\text{A11})$$

be the set of nodes with point-boxes at \mathbf{a} . If Λ'' is not empty, then y_j is defined to distribute activation evenly among those nodes in Λ'' , intuitively by a mechanism of self-normalizing competition:

$$y_j = \begin{cases} \frac{1}{|\Lambda''|} & \text{if } j \in \Lambda'' \\ 0 & \text{if } j \notin \Lambda'' \end{cases}. \quad (\text{A12})$$

Field F_3 contains C nodes, each node corresponding to a category coding node in F_2 (see Figure 2). F_3 activation biases the distributed input from F_2 (y_j) with *instance counting weights* c_j :

$$Y_j = \frac{c_j \mathcal{Y}_j}{\sum_{\lambda=1}^C c_\lambda \mathcal{Y}_\lambda}. \quad (\text{A13})$$

Counting weights reflect the previous category coding node activations summed over training set inputs. Thus, a node j with greater and more frequent training activations contributes more to F_3 activation Y_j . The denominator of (A13) ensures that the total F_3 activation is normalized:

$$\sum_{j=1}^C Y_j = 1, \text{ again reflecting a process of self-normalizing competition.}$$

This instance-biased activation drives the $F_3 \rightarrow F_1$ top-down expectation signal σ_i , filtered by adaptive thresholds τ_{ji} :

$$\sigma_i = \sum_{j=1}^C [Y_j - \tau_{ji}]^+, \quad (\text{A14})$$

$i = 1, \dots, 2M$, where the threshold rectification $[w]^+ \equiv \max\{w, 0\}$. In (A14), σ_i represents the *critical feature pattern* for the distributed F_2 category code. Including Y_j , which is always less than or equal to one, modulates each node's contribution to σ_i according to its F_3 activation: nodes with strong Y_j contribute more, and nodes with weak Y_j contribute less, or not at all when $Y_j \leq \tau_{ji}$. Geometrically, the σ_i represent category coding boxes $R_j(Y_j)$ that expand as Y_j increases. (For more detail on the geometry of distributed coding, see Carpenter, 1997, and Carpenter et al., 1998.)

Activation x_i of each F_1 node i (see Figure 2) reflects the match between the top-down expectation signal σ_i and the bottom-up input A_i :

$$x_i = A_i \wedge \sigma_i, \quad (\text{A15})$$

$i = 1, \dots, 2M$. In order for system activation to persist and for learning to occur, F_1 activation must pass the *vigilance test*: its total activation must be greater than the threshold ρM ; namely,

$$\sum_{i=1}^{2M} x_i \geq \rho M, \quad (\text{A16})$$

where vigilance ρ varies between an initial baseline value $\bar{\rho}$ and 1. A large value of ρ requires a very good match: all σ_i must be very similar to A_i . The baseline vigilance value of $\bar{\rho} = 0$ allows category coding boxes $R_j(Y_j)$ to expand as much as necessary, as long as the system predicts the correct category (see Section 4.2.).

If a mismatch occurs; namely,

$$\sum_{i=1}^{2M} x_i < \rho M, \quad (\text{A17})$$

F_1 activation fails the vigilance test. Then the distributed top-down expectation was not a good enough match to the input pattern, so the system will begin to search for a category coding node with a better match. Search resets the activity at the category coding field F_2 and forces a new *winner-take-all (WTA)* activation, wherein

$$y_j = \begin{cases} 1 & j = J = \arg \max_{j \in \Lambda} \{T_j\} \\ 0 & \text{otherwise} \end{cases}. \quad (\text{A18})$$

In (A18), the winning node index J represents the category coding node with the greatest input. Activation at the instance counting and match fields, F_3 and F_1 , respectively, updates according to (A13-15) to reflect the new WTA activation at the category coding field F_2 . If the new F_1 activation still does not pass vigilance, F_2 activation is reset again. Node J enters a refractory state, wherein it can no longer be reactivated for the duration of input \mathbf{a} 's presentation, and so it

is removed from the active search node set Λ in (A9). Search via WTA activation at F_2 and subsequent reset continues with a new winning node J that has the next largest input T_j in (A18) until a node that passes the vigilance test (A16) is found. If no committed node is found that can pass vigilance, a new category coding node is recruited, activated, and associated to the target output category label K . Recruitment triggers fast learning at the newly selected category; namely, a *fast commitment* of LTM thresholds, such that the new category coding box R_J is a point box at \mathbf{a} :

$$\tau_{iJ} = \tau_{Ji} = 1 - A_i, \quad (\text{A19})$$

$$i = 1, \dots, 2M.$$

In all, when the match at F_1 passes the vigilance test, the F_3 activity pattern Y_j forms the category prediction signal σ_k from F_3 to F_0^{ab} (see Figure 2). Each F_3 node j is associated with a single output category k according to the mapping $\kappa(j) = k$, supported by $F_3 \rightarrow F_0^{ab}$ connection weights tuned when node j was first recruited. The prediction field F_0^{ab} has L nodes, one for each output category, and each node's input σ_k sums the associated F_3 node activations:

$$\sigma_k = \sum_{\substack{j=1 \\ \kappa(j)=k}}^C Y_j. \quad (\text{A20})$$

This distributed prediction signal undergoes winner-take-all competition at F_0^{ab} . Node index K' is the smallest index k with the maximum input:

$$K' = \arg \max \{ \sigma_k \}, \quad (\text{A21})$$

and winning F_0^{ab} activation:

$$Z_k = \begin{cases} 1 & k = K' \\ 0 & \text{otherwise} \end{cases}. \quad (\text{A22})$$

During training, F_0^{ab} node activations Z_k are matched at F_1^{ab} with the target training outputs b_k at F_0^b (see Figure 2). If the two patterns do not match, the system made an incorrect category prediction ($K' \neq K$) and must be corrected. *Match tracking (MT)* helps correct the error by increasing vigilance just enough to reset activation (see (A17)) at the category coding field F_2 :

$$\rho = \frac{1}{M} \sum_{i=1}^{2M} x_i + \varepsilon, \quad (\text{A23})$$

where the match-tracking parameter ε is a small, fixed value. As noted in Section 4.2, match tracking implements a minimax learning property that conjointly maximizes generalization while minimizing predictive error. The MT– match tracking search rule, defined as (A23) with a negative value for the match tracking parameter ε , permits the next winning category coding node to match \mathbf{A} just as well as the currently active code; MT– supports the encoding of inconsistent cases. In contrast, the MT+ match tracking search rule, defined as (A23) with a positive value for ε , requires the next winning node to be a better match than the currently active code. Match tracking resets F_2 activation and switches it to winner-take-all mode, and a search for a better matching code continues.

If the top-down expectation and bottom-up input patterns do match (that is, (A16) is satisfied), signifying that the network made a correct prediction, then F_2 category coding nodes learn via a process of *adaptive resonance*, described as follows.

$F_0^b \rightarrow F_2$ *credit assignment* connections deactivate all category coding nodes except those mapped to the target category K , and F_2 activities are renormalized such that the total activation $\sum_{j=1}^C y_j = 1$:

$$y_j = \begin{cases} \frac{y_j^{old}}{\sum_{\substack{\lambda=1 \\ \kappa(\lambda)=K}}^C y_\lambda^{old}} & \text{if } \kappa(j) = K \text{ (correct category)} \\ 0 & \text{if } \kappa(j) \neq K \text{ (incorrect category)} \end{cases}, \quad (\text{A24})$$

where y_j^{old} is the activation of node j prior to credit assignment. F_3 and F_1 activations update to reflect the new y_j according to (A13-A15). Credit assignment, facilitated by $F_0^b \rightarrow F_2$ connection weights learned during F_2 node recruitment, ensures that only nodes mapped to K will learn.

Category learning occurs using a *distributed instar learning law* in $F_0 \rightarrow F_2$ bottom-up pathways. Learning increases the thresholds τ_{ij} :

$$\tau_{ij} = \tau_{ij}^{old} + \beta [y_j - \tau_{ij}^{old} - A_i]^+, \quad (\text{A25})$$

where τ_{ij}^{old} is the value of τ_{ij} prior to learning, and the learning rate β is a positive parameter. The second term in (A25) represents the amount of learning required for the dynamic bottom-up weight $w_{ij} = [y_j - \tau_{ij}]^+$ to decrease toward A_i as the threshold τ_{ij} increases. Thus, fast learning ($\beta = 1$) quickly expands the dynamic category coding box $R_j(y_j)$ to include \mathbf{a} .

Learning of top-down expectations occurs using a *distributed outstar learning law* in $F_3 \rightarrow F_1$ top-down pathways. Learning increases the thresholds τ_{ji} :

$$\tau_{ji} = \tau_{ji}^{old} + \beta \frac{[\sigma_i - A_i]^+}{\sigma_i} [Y_j - \tau_{ji}^{old}]^+ . \quad (\text{A26})$$

During fast learning, dynamic top-down weights $w_{ji} = [Y_j - \tau_{ji}]^+$ decrease until $\sigma_i \leq A_i$, at which time the category boxes $R_j(Y_j)$ enclose \mathbf{a} . The first multiplicative term, $\frac{[\sigma_i - A_i]^+}{\sigma_i}$, represents the proportion by which σ_i must decrease to be less than or equal to A_i . The second multiplicative term, $[Y_j - \tau_{ji}]^+$, represents the total amount of learning possible on τ_{ji} while also restricting the dynamic weight w_{ji} to be a non-negative value. The second additive term of (A26) in total, $\beta \frac{[\sigma_i - A_i]^+}{\sigma_i} [Y_j - \tau_{ji}^{old}]^+$, represents τ_{ji} 's share of learning toward the goal of decreasing σ_i toward A_i .

Lastly, instance counting weights from F_2 to F_3 are updated as follows:

$$c_j = c_j^{old} + y_j, \quad (\text{A27})$$

where c_j^{old} is the value of c_j prior to learning.

During testing, no target category K exists. The active F_0^{ab} node index K' represents the predicted category, and LTM traces adapt according to one of four models. In Model 0, no adaptation occurs. In Model 1, LTM traces adapt according to equations (A25) and (A26) above, without any credit assignment (A24). Model 2 adapts LTM traces exactly as in training, treating the category prediction K' as if it were a target output category K . Models 3 and 4 adapt LTM traces by adding random, normally distributed noise

$$\xi \sim N(0, \nu^2) \quad (\text{A28})$$

to every active category coding node's thresholds:

$$F_0 \rightarrow F_2 \text{ bottom-up threshold: } \tau_{ij} = \tau_{ij}^{old} + \xi \quad (\text{A29})$$

$$F_3 \rightarrow F_1 \text{ top-down threshold: } \tau_{ji} = \tau_{ji}^{old} + \xi \quad (\text{A30})$$

Both τ_{ij} and τ_{ji} are constrained to the unit interval; that is, $\tau_{ij} = \min\{\max(0, \tau_{ij}^{old} + \xi), 1\}$. Model 3 updates thresholds for all active nodes without any prior credit assignment. Model 4 first imposes credit assignment (A24), and as such, only thresholds for nodes mapped to the predicted category K' are updated.

The following iterative steps condense the above description into an implementable algorithm. For clarity and conciseness, network activation steps are omitted when they do not impact category learning or prediction.

A.2. dARTMAP Training

Set the initial category coding node count $C = 0$.

For each training input and target category label (\mathbf{a}, K):

1. Initialize the network:
 - 1.1. Set the vigilance variable to its baseline value: $\rho = \bar{\rho}$.
 - 1.2. Set the coding field F_2 activation mode to distributed.
2. Activate the input field F_0 (A_i) via the complement-coding equation (A1).
3. Calculate F_2 inputs T_j via the choice-by-difference equation (A8).
4. Define the active F_2 node sets Λ and Λ'' via equations (A9) and (A11).

5. If the active F_2 node set Λ is empty, recruit a new category coding node.
 - 5.1. Set the winning node index $J = C + 1$.
 - 5.2. Set its LTM thresholds via the fast-commit equation (A19).
 - 5.3. Set the instance count $c_J = 1$.
 - 5.4. Map the new node to the target category: $\kappa(J) = K$.
 - 5.5. Increment the category coding node count C by 1.
 - 5.6. Continue to the next training input (Step 1).
6. If F_2 is in distributed mode:
 - 6.1. If the point-box node set Λ'' is empty, calculate F_2 activation y_j via the IG-CAM rule (A10).
 - 6.2. Otherwise (Λ'' is not empty), calculate F_2 activation y_j via the alternate IG-CAM rule (A12).
7. Otherwise (F_2 is in WTA mode), determine the winning node J , and calculate F_2 activation y_j via the WTA equation (A18).
8. Activate the counting field F_3 (Y_j) via equation (A13).
9. Activate the matching field F_1 :
 - 9.1. Calculate the top-down expectation signal σ_i via equation (A14).
 - 9.2. Calculate the F_1 activation x_i via equation (A15).
10. If F_1 fails vigilance via equation (A17):
 - 10.1. If F_2 is in distributed mode, change it to WTA mode.
 - 10.2. Otherwise, remove the winning node J from the active F_2 node set Λ .
 - 10.3. Return to Step 5.

11. Otherwise (F_1 passes vigilance via equation (A16)), continue to Step 12 for category prediction.
12. Calculate the category prediction signal σ_k via equation (A20).
13. Calculate the category prediction K' via equation (A21).
14. If K' does not equal the target category K :
 - 14.1. Raise vigilance ρ via the match-tracking equation (A23).
 - 14.2. If F_2 is in WTA mode, remove the winning node J from active node set Λ .
 - 14.3. Otherwise (F_2 is in distributed mode), set the F_2 activation mode to WTA.
 - 14.4. Return to Step 5.
15. Otherwise ($K' = K$), continue to Step 16 for credit assignment and learning.
16. Reactivate fields F_2 , F_3 , and F_1 according to the credit-assignment equation (A24) and (A13-A15).
17. Learn:
 - 17.1. Update the bottom-up thresholds τ_{ij} via the distributed instar equation (A25).
 - 17.2. Update the top-down thresholds τ_{ji} via the distributed outstar equation (A26).
 - 17.3. Update instance-counting weights c_j via equation (A27).

A.3. dARTMAP Testing

For each testing input \mathbf{a} :

1. Activate the input field $F_0 (A_i)$ via the complement-coding equation (A1).
2. Calculate F_2 inputs T_j via the choice-by-difference equation (A8).
3. Define the active F_2 node sets Λ and Λ'' via equations (A9) and (A11).
4. If the point-box node set Λ'' is empty, calculate F_2 activation y_j via the IG-CAM rule (A10).

5. Otherwise (Λ'' is not empty), calculate F_2 activation y_j via the alternate IG-CAM rule (A12).
6. Activate the counting field F_3 (Y_j) via equation (A13).
7. Calculate the category prediction signal σ_k via equation (A20).
8. Calculate the category prediction K' via equation (A21).
9. For LTM update models 2 and 4, reactivate fields F_2 , F_3 , and F_1 according to the credit-assignment equation (A24) and (A13-A15).
10. For LTM update models 1 and 2, update bottom-up thresholds τ_{ij} via the distributed instar equation (A24) and top-down thresholds τ_{ji} via the distributed outstar equation (A26).
11. For LTM update models 3 and 4, update the bottom-up and top-down thresholds by noise-addition equations (A28-A30).

Table A1: dARTMAP notation.

NOTATION	DESCRIPTION	RANGE	EQUATIONS
M	Number of input features		A1-A8, A10, A11, A14-A17, A19, A23
L	Number of output categories		
K	Correct output category for training input	$K \in \{1, \dots, L\}$	A24
i	Input component index and node index for fields F_0 and F_1	$i = 1, \dots, M$ $i = 1, \dots, 2M$	A1, A6-A8, A14-A17, A19, A23, A25, A26, A29, A30
\mathbf{a}	Input feature vector (a_i) , $i = 1, \dots, M$	$0 \leq a_i \leq 1$	A1, A2, A4, A5
\mathbf{A}	F_0 field activity (A_i) , $i = 1, \dots, 2M$	$0 \leq A_i \leq 1$	A1, A4-A6, A8, A15, A19, A25, A26
j	Category coding node index for fields F_2 and F_3	$j = 1, \dots, C$	A2-A14, A18, A20, A24-A27, A29, A30
C	Number of committed category coding nodes		A9, A13, A14, A20, A24
τ_{ij}	$F_0 \rightarrow F_2$ bottom-up long-term memory threshold from input node i to category coding node j	$\tau_{ij} \in [0, 1]$	A6-A8, A19, A25, A29
S_j	Phasic input to category coding node j	$S_j \in [0, M]$	A5, A6, A8
Θ_j	Tonic input to category coding node j	$\Theta_j \in [0, M]$	A5, A7, A8
T_j	$F_0 \rightarrow F_2$ input signal to category coding node j	$T_j \in [0, M]$	A2, A5, A8-A11, A18
Λ, Λ''	In distributed mode, the index sets of active category coding nodes	$\subseteq \{1, \dots, C\}$	A9, A10-A12, A18
J	In winner-take-all mode, the index of the chosen category coding node	$J \in \{1, \dots, C\}$	A18, A19
\mathbf{y}	F_2 category coding field activation (y_j)	$y_j \in [0, 1]$	A10, A12, A13, A18, A24, A25, A27
c_j	Instance-counting weight to counting node j	$c_j \geq 1$	A13, A27
\mathbf{Y}	F_3 counting field activation (Y_j)	$Y_j \in [0, 1]$	A13, A14, A20, A26
τ_{ji}	$F_3 \rightarrow F_1$ top-down long-term memory threshold from category coding node j to matching node i	$\tau_{ji} \in [0, 1]$	A14, A19, A26, A30
σ_i	$F_2 \rightarrow F_1$ top-down expectation, $i = 1, \dots, 2M$	$\sigma_i \in [0, 1]$	A14, A15, A26
\mathbf{x}	F_1 match field activation (x_i) , $i = 1, \dots, 2M$	$x_i \in [0, 1]$	A15-A17, A23
ρ	Vigilance variable	$\rho \in [\bar{\rho}, 1]$	A16, A17, A23

NOTATION	DESCRIPTION	RANGE	EQUATIONS
k	Output category index for F_0^{ab} , F_1^{ab} , and F_0^b	$k = 1, \dots, L$	A20-A22
$\kappa(j) = k$	Association between the category coding node j and the output category k	$\kappa(j) \in \{1, \dots, L\}$	A20, A24
σ_k	$F_3 \rightarrow F_0^{ab}$ prediction signal	$\sigma_k \in [0, 1]$	A20, A21
K'	Predicted output category	$K' \in \{1, \dots, L\}$	A21, A22
\mathbf{Z}	F_0^{ab} category field activation (Z_k)	$Z_k \in \{0, 1\}$	A22
ξ	Noise added to long-term memory		A28-A30

Table A1: dARTMAP notation (Continued).

PARAMETER		RANGE	SIMULATION VALUE	EQUATIONS
$F_0 \rightarrow F_2$ signal rule parameter	α	(0,1)	0.01	A2, A5
$F_0 \rightarrow F_2$ signal to uncommitted nodes	T^u	$T_j _{\tau_{ij}=0}$	αM	A9
F_2 CAM rule power	p	(0, ∞]	1.0	A10
Baseline vigilance	$\bar{\rho}$	[0,1]	0.0	
Match tracking parameter	ε	$ \varepsilon $ small	-0.001 (MT-)	A23
Learning rate	β	(0,1]	1.0 (fast learning)	A25, A26
Standard deviation of noise added to long-term memory	ν		0.0332 (Model 3) 0.0626 (Model 4)	A28, A29

Table A2: dARTMAP parameters.

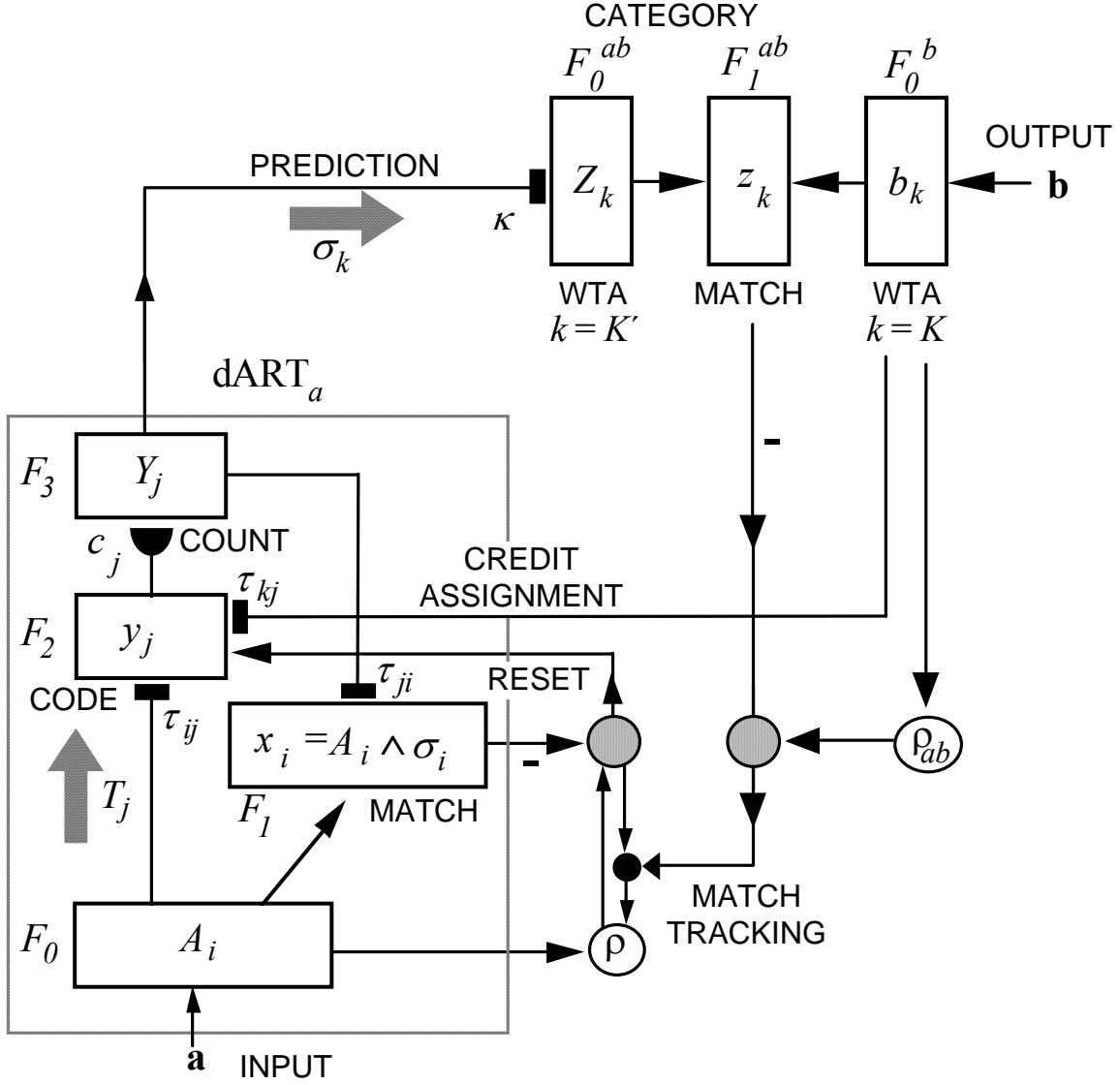


Figure 2: dARTMAP model circuit. Activity patterns \mathbf{y} at the field F_2 represent distributed category codes. Winner-take-all (WTA) activation at the field F_0^{ab} represents categorical output category predictions K' (A or B in the 5-4 category problem). During training, WTA activation at F_0^b represents the actual output category (K) provided by the teacher. A mismatch at F_1^{ab} causes a match tracking signal to raise the dART_a vigilance ρ just enough to reset the active code at F_2 . The \wedge symbol in the matching field F_1 denotes a minimum operation, that is, $x_i = A_i \wedge \sigma_i = \min\{A_i, \sigma_i\}$. See the text and Appendix for details.