Chapter 18

ART 2

§ 1. Basic Anatomy of ART 2

The example ART network of chapter 17 has a basic functional limitation, namely that this network is only capable of handling binary-valued input signals or, at most, input signals that are basically binary-valued with some limited amount of pattern variance and noise (corrupted binary valued input signals). ART 2 networks do not have this limitation.

Like other published ART networks, ART 2 is primarily an algorithm in which many of the fine details of adaptive resonance dynamics are omitted for the purpose of achieving better computational efficiency. The designation "ART 2 network" refers to a family of different ART networks rather than one single network topology. In this chapter we look at one such example, namely the ART 2 network discussed in detail in [CARP3]. Two other instances of ART 2 are also qualitatively presented in [CARP3]. Figure 18.1 illustrates the basic anatomy of ART 2. Like other ART networks, it consists of two fields, F_1 and F_2, an attentional/orienting subsystem, and an onset/offset reset inhibition function.

§ 1.1 The F_2 Field

In principle, the F_2 field is a contrast-enhancing dipole network similar to the one used by the

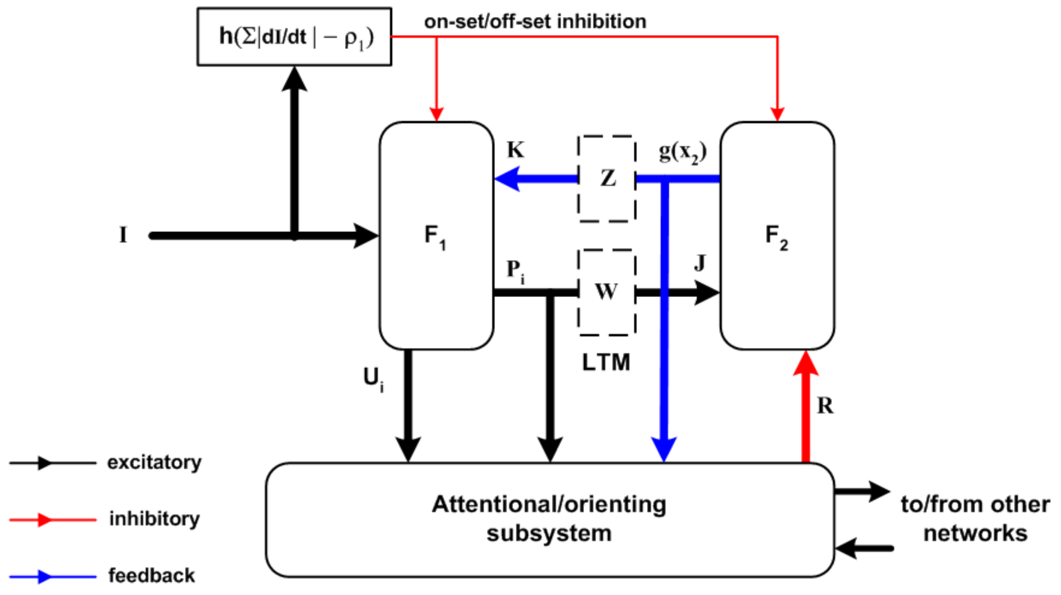


Figure 18.1: Basic ART 2 anatomy.

network of chapter 17. In practice, the simulation of the F_2 field is greatly simplified. You will recall from the Grossberg theorems of chapter 15 that one key property of shunting node Instar (SNI) on-center/off-surround networks is the preservation of excitation order from the initial condition of the network to the final condition. Specifically, if a node $x_{2j}(0)$ has the largest input at time index $t = 0$, it will also have the largest final node value in steady state. The F_2 layer is, in principle and in design, intended to operate in the 0-1 distribution. Thus, its mathematical description can be reduced to a simple switching logic representation subject to certain qualifications.

Let t_0 denote a time index at which a change of input vector **I** is first presented to the network and the accompanying onset/offset reset function has been effected. Let $T_j = W_j^T \mathbf{P_i}$ where $\mathbf{P_i}$ is the F_1 output vector depicted in figure 18.1 at time index t_0 and W_j is the Instar weight vector for the jth node of F_2. Let us further assume that associated with each node of F_2 there is a dipole vector $\mathbf{\Psi}$ with elements ψ_j such that $\psi_j = 1$ if the jth node is in a state of dipole reset and $\psi_j = 0$ otherwise. Then for $t \geq t_0$ and until F_2 undergoes either in input/offset reset or a mismatch reset from the orienting subsystem,

$$x_{2j}(t) = T_j(t) \cdot (1 - \psi_j) \tag{18.1}$$

where x_{2j} is the excitation variable of the jth F_2 node and j lies in the range from 1 to N for an N-node F_2 layer.

The output activation vector $\mathbf{Y_j} = g(\mathbf{x_2})$ is governed by a special form of step-function activation function. What makes this function special is that it has three modes of operation: (1) If $\mathbf{x_2} = \mathbf{0}$, $\mathbf{Y_j} = \mathbf{0}$; (2) If there is a unique nonzero $x_{2j}(t_0) = \max(\mathbf{x_2})$, then $y_j = d$ and $y_{k \neq j} = 0$. Here d is a system parameter in the range $0 < d < 1$; (3) If the maximum nonzero $x_{2j}(t_0)$ is not unique – that is, if two or more $x_{2j}(t_0)$ excitations are tied for the maximum value – then g implements a ***tie-breaking rule*** which selects $y_j = d$ at one node and sets the other competitors to $y_k = 0$.

Case (3) deserves some additional commentary. The reason for having a case (3) is to ensure that F_2 operates in a 0-1 distribution. The activation function g when operating in this mode is called a ***select-first-responder function*** in the language of computer engineers. It is a standard function commonly employed in content-addressable memories and in computers that use what is known as a content-addressable parallel processor. There are a number of ways to implement this function in logic, e.g. [LAVE], and from the viewpoint that digital logic circuits are merely neural networks using McCulloch-Pitts neuron models, the select-first-responder function is not biologically unreasonable. However, the select-first-responder function is *not* compatible with the nature of a Grossberg contrast-enhancing on-center/off-surround network, which would produce a locally uniform output distribution under the condition in which case (3) arises. Therefore, we must view the select-first-responder function within $g(\mathbf{x_2})$ as implying the CE function in F_2 feeds into an additional network layer prior to $\mathbf{Y_j}$ being presented at the F_2 output vector in figure 18.1. *All* this detail is, of course, subsumed under the simple input-output description of F_2 used by the ART 2 algorithm.

It was stipulated above that once the F_2 layer has made a *choice* (0-1 distribution output), this choice is maintained until F_2 undergoes a reset from either the orienting subsystem (whereby the currently selected node has its ψ_j state variable set to 1) or from the onset/offset reset function (whereby all F_2 nodes are reset and $\mathbf{\Psi}$ is reset to $\mathbf{\Psi} = \mathbf{0}$). This characteristic of the F_2 function implicates an F_2 on-center/off-surround network in which the short term memory STM_2 is ***persistent***, i.e. $\mathbf{Y_j}$ does not respond to changes in input once the final distribution has been established unless a reset inhibition intervenes. Again, the details of an on-center/off-surround SNI network having this characteristic are suppressed in the ART 2 algorithm. This operational condition of **stable choice until reset** [CARP3] is a vital function in ART 2. If F_2 is not given this functional characteristic, the effects on the system dynamics are quite interesting. During the learning mode – when LTM categories are being established – the absence of this operational characteristic of F_2 leads to what can justly be called ***hallucinations***. In a manner of speaking, the network "sees things that aren't there" or "hears voices" or "suffers from an over-active imagination." It thereafter often resets when presented input vectors it had previously "learned" – a sort of neural network version of autism.

§ 1.2 The F_1 Field

Like the ART network presented in chapter 17, the F_1 field of ART 2 is a contrast-enhancing field. However, the manner in which this field is implemented is quite different from our earlier example. Figure 18.2 illustrates the details of a single node in the F_1 field. The node is comprised of a three-layer network structure consisting of network maps w_i, v_i, p_i, x_i, u_i and q_i. It also contains three nonspecific gain networks projecting into the F_1 node from the other nodes in the F_1 layer, and three nonspecific gain networks associated with the mismatch-reset function of the orienting subsystem. The nonspecific networks are shown in green in figure 18.2. For the F_1 field as a whole, we refer to the vectors representing the various maps as $\mathbf{W_i}$, $\mathbf{V_i}$, $\mathbf{P_i}$, $\mathbf{X_i}$, $\mathbf{U_i}$, and $\mathbf{Q_i}$. The "+" superscripts in figure 18.2 denote projections from the other nodes in F_1. Maps w_i, v_i, and p_i are shunting node Instars of type $SNI^{(4)}$. The other three are maps implement normalization. The figure also depicts the mismatch-reset operation of the orienting subsystem.

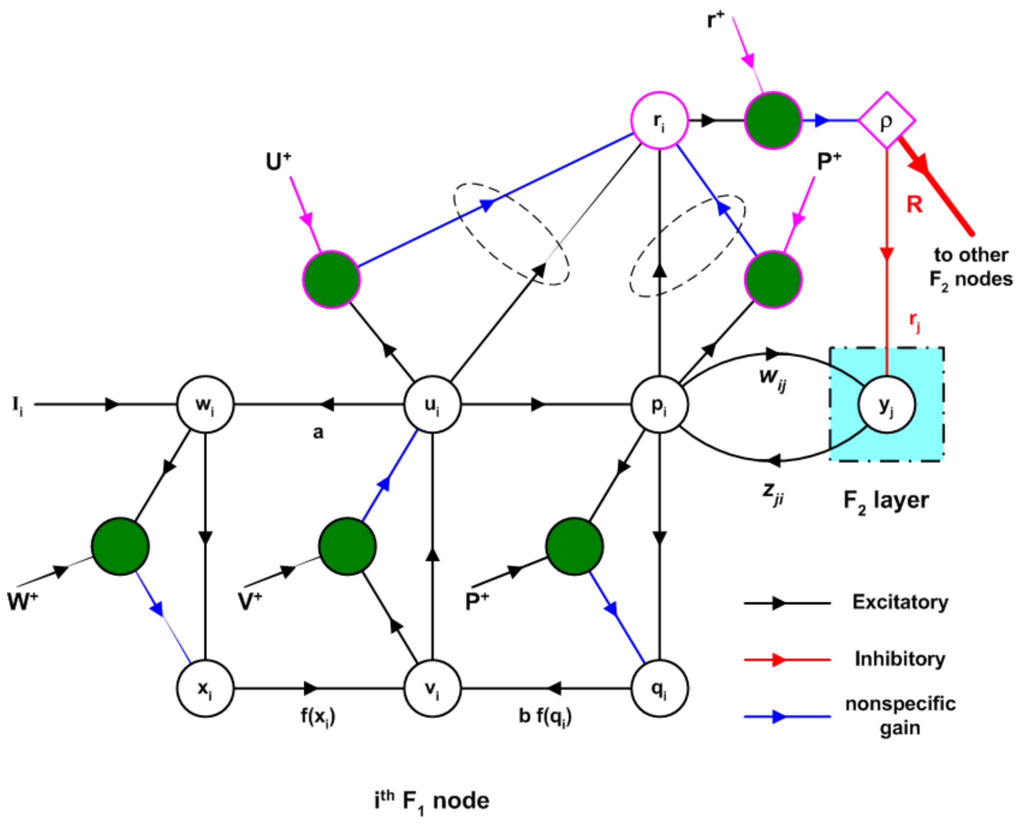


Figure 18.2: Details of the ART 2 F$_1$ field. The F$_1$ nodes do not have any immediate lateral connectivity to each other, although there is an indirect connection by means of the nonspecific gain nodes (shown in green in this figure). Each F$_1$ node is comprised of a three-layer recurrent neural network of SNI$^{(4)}$-type Instars. The figure also depicts the structure of the mismatch-reset function of the orienting subsystem (pink elements).

The F$_1$ nodes have no direct lateral interconnections. The only source of lateral connectivity in F$_1$ is the indirect connection via the nonspecific gain maps. Thus each six-map F$_1$ node operates independently of the others except for the normalization functions. SNI$^{(4)}$ parameters are chosen such that steady state is reached within one time step of the simulation and

$$\begin{aligned} w_i &= I_i + a \cdot u_i \\ v_i &= f(x_i) + b \cdot f(q_i) \\ p_i &= u_i + Z_i \, \mathbf{Y_j} \end{aligned} \quad . \tag{18.2}$$

Here *a* and *b* are network parameters. Typically *a* = *b*. In [CARP3] these parameters are set equal to 10; in the examples presented in this chapter, they are set equal to 5. Z_i is, of course, the top-down LTM vector for the *i*th node of F$_1$ (a row vector in **Z**). The activation function *f* is not the typical activation function used in adaptive resonance theory. Instead, it is a form of thresholding Heaviside extractor with threshold parameter θ defined by

$$f(x) = \begin{cases} 0, & x < \theta \\ x, & x \ge \theta \end{cases} \quad . \tag{18.3}$$

It is assumed that the gain networks (the three **W**, **V**, and **P** nodes in figure 18.2) project a nonspecific gain to the normalization maps x_i, u_i, and q_i in each layer of F_1. The details of this are left out of the ART 2 algorithm and merely the end result is reflected in the definition of the map functions. The algorithm uses the L_2 norm for its normalization gain function so that

$$\begin{aligned} x_i &= \frac{w_i}{\|\mathbf{W_i}\|}, \quad x_i = 0 \text{ if } \mathbf{W_i} = \mathbf{0} \\ u_i &= \frac{v_i}{\|\mathbf{V_i}\|}, \quad u_i = 0 \text{ if } \mathbf{V_i} = \mathbf{0}\,. \\ q_i &= \frac{p_i}{\|\mathbf{P_i}\|}, \quad q_i = 0 \text{ if } \mathbf{P_i} = \mathbf{0} \end{aligned} \tag{18.4}$$

All maps in F_1 are presumed to reach steady state (i.e. adaptive resonance) within one time step of the simulation. This means (18.2) gives the equations for the *next* values of the map variables and likewise for (18.4). This is the standard discrete-time state variable formulation for system models, i.e. the left-hand side of the equation is the next time step value and the right-hand side is the current time step value.

An exception occurs when a mismatch-reset is generated by the orienting subsystem. Such a reset clears the currently-activated node of F_2, and it is presumed that this clearing is reflected in the F_1 layer within the current simulation time step. The way this is implemented is to set the presently active y_j equal to zero and to re-compute (18.2) using $\mathbf{Y_j} = \mathbf{0}$. Specifically, (18.2) is computed as shown, the mismatch condition is calculated, and if mismatch occurs then the reset-clear-compute again calculation is made. The algorithm presumes an adaptive resonance is always set up within a single time step in the simulation, and this is one of the reasons why the F_2 layer is computed in the way it is, and one of the reasons why the stable-choice-until-reset constraint is necessary.

§ 1.3 Mismatch, Reset, and the Orienting Subsystem

Other than for a few differences in detail, the mismatch-reset function in ART 2 is essentially the same as the one used for the example network in chapter 17. Each F_1 node contributes to a pattern match vector $\mathbf{r} = [r_1 \cdots r_n]$. For the *i*th node,

$$r_i = \frac{u_i + c \cdot p_i}{\varepsilon + \|\mathbf{U_i}\| + c \cdot \|\mathbf{P_i}\|} \tag{18.5}$$

where ε is any small positive constant and is used to prevent divide-by-zero errors. c is a network parameter, which is set to $c = 0.1$ for the example simulations of this chapter.

A mismatch is declared when the length of the **r** vector falls below a ***vigilance parameter***, ρ. Specifically, F_2 is reset whenever $\rho > \|\mathbf{r}\|$ and input pattern **I** has one or more elements I_i that exceed a "no pattern" threshold. (Resets are not enabled in the absence of an input **I**). As noted above, when an F_2 reset is issued the F_1 field is presumed to respond quickly enough to $\mathbf{Y_j} = \mathbf{0}$ so that the next state of F_1 is determined from the original state with $\mathbf{Y_j} = \mathbf{0}$ in (18.2). Dipole resets in F_2 are persistent, i.e. once an F_2 node undergoes a mismatch reset, its dipole state remains in the reset condition until it is cleared by an onset/offset reset.

The normalizations in (18.5) and in $\rho > \|\mathbf{r}\|$ are directly computed in the ART 2 algorithm and the details of the nonspecific gain networks associated with these operations in figure 18.2 are left unspecified. The comments made in chapter 17 regarding the use of the L_2 norm in ART models apply as well to the ART 2 algorithm.

§ 1.4 Adaptation

Adaptation is carried out using the IAR for bottom-up weight matrix **W** and the OAR for top-down weight matrix **Z**. (Note that the weight matrix symbol **W** is not the same as the F_1 map vector $\mathbf{W_i}$). In difference equation form,

$$\begin{aligned} W_j(t+1) &= W_j(t) + \eta \cdot \left(\mathbf{P_i}(t) - W_j(t)\right) \cdot g\left(x_{2j}\right) \\ Z_j(t+1) &= Z_j(t) + \eta \cdot \left(\mathbf{P_i}(t) - Z_j(t)\right) \cdot g\left(x_{2j}\right) \end{aligned} . \tag{18.6}$$

Because it is assumed that F_1 and F_2 come to a state of adaptive resonance within one time step of the simulation, (18.6) is applied in each time step after (18.1)-(18.5) have been calculated and the mismatch reset operation has been determined. When a reset event is activated during a simulation step, (18.6) is not applied because $\mathbf{Y_j} = \mathbf{0}$ in this case. Likewise, adaptation is not applied if the input vector **I** is below the "no pattern" threshold. Note that (18.6) is a straight difference equation and contains no Δt term. This is because F_1 and F_2 are modeled by the steady state expressions and the ART 2 algorithm involves no explicit numerical solution of any *differential* equation. Note also that $g(x_{2j})$ equals either d or 0, depending on whether F_2 node j is the selected node or one of the not-chosen or reset nodes.

One computational advantage of this in the ART 2 algorithm is that so-called ***fast learning*** is possible. In the example network of chapter 17, where numerical solutions for differential equations were required, the learning rate constant η was limited to rather small values ($\eta = 0.02$ in chapter 17). For ART 2, η can be set to considerably higher values, e.g. tenfold or so greater, without concern that this faster learning rate will lead to numerical errors. In this sense, the difference between ART 2 and the network of chapter 17 is similar to the difference between

Rulkov's map model and Izhikevich's differential equation model of the abstract neuron, and for the same reason. ART 2 begins in difference equation form, whereas the network of chapter 17 begins in differential equation form and must be converted to difference equation form. The net practical result is that ART 2 is a much faster-to-compute model with more than an order of magnitude advantage in cost-of-computation over the lower-level model of chapter 17. Indeed, in overall speed and cost-of-computation terms, ART 2 outperforms most other neural network models, e.g. feedforward network models using the famous back-propagation algorithm, in learning problems of equal complexity. Other comparisons between ART and alternative models are discussed in [GROS10].

§ 2. ART 2 Simulation Examples

As illustrations of the behavior of ART 2, we will look at several simulation examples for an ART 2 network with $n = 25$ nodes in F_1 and $N = 3$ nodes in F_2 (the same basic widths as used in the example network of chapter 17). In all cases, the network parameters are $a = b = 5$, $c = 0.10$, and $d = 0.90$. We will also use $\eta = 0.2$, ten times faster than the value used in the network of chapter 17. ART 2 is capable of faster learning performance, as we will show here. For our test cases we will use the same input patterns from chapter 17 shown in figure 16.5.

§ 2.1 Initial Pattern Classification

LTM matrices **Z** and **W** must always be initialized to some starting values. For ART 2, as for the network of chapter 17, the initial setting for the top-down weights is $\mathbf{Z} = \mathbf{0}$. For the bottom-up weights, we will initialize **W** to random settings with a uniform distribution. One requirement of ART 2 is that initial weight values satisfy the constraint $w_{ij} \le 1 \big/ \left((1-d) \cdot \sqrt{n} \right)$. Strictly speaking, this constraint applies when the initial weight distribution is uniform (all $w_{ij}(0)$ values equal), but it is a sufficient condition in any case for ensuring reliable initial learning behavior in the network. However, it remains necessary to enforce the stable-choice-until-reset constraint on the network even if $w_{ij}(0)$ values all satisfy this constraint. This is because the magnitude of $\mathbf{P_i}$ increases as learning progresses and it can be empirically observed that "node switching" in F_2 (i.e., changes in which node has the maximum x_{2j} value) can occur during the network's initial pattern presentations if the stable-choice-until-reset constraint is not applied. This mechanism can and often does lead to the network suffering from "hallucinations" in its initial learning phase.

For our initial-learning example we present the network with the pattern sequence aX ("analog X"), d, and J (see figure 16.5). This sequence is chosen because this pattern set has overlapping "pixels" between each pair of input patterns. It is therefore a more challenging learning problem for the network than is the case when the initial categories are established for input patterns that share no common features. The reader is also reminded that the network of chapter 17 was incapable of learning the aX pattern because of that network's contrast-enhancing properties in its F_1 field. As we will see shortly, ART 2 has no difficulty categorizing this pattern.

Figure 18.3 illustrates the initial learning sequence. Each pattern is learned upon first presentation within the dwell time of the pattern application. The F_2 node categories assigned are shown in the figure. LTM is completed within the first presentations, so upon the second presentation of the aX-d-J sequence, the network directly accesses the correct categories for each pattern. No mismatch resets occur at any time during this run.

The robustness of the learned categories are tested by applying an A-O-T-fJ-sJ-X pattern sequence. (fJ = "fat J"; sJ = "shifted J"). All but the sJ pattern are defined in figure 16.5. The sJ pattern is identical to the J pattern except for being shifted one pixel column to the left in the retina (see figure 18.5). In this position, the sJ pattern is actually a subset of the d pattern, being identical to it except for four zero-valued pixels.

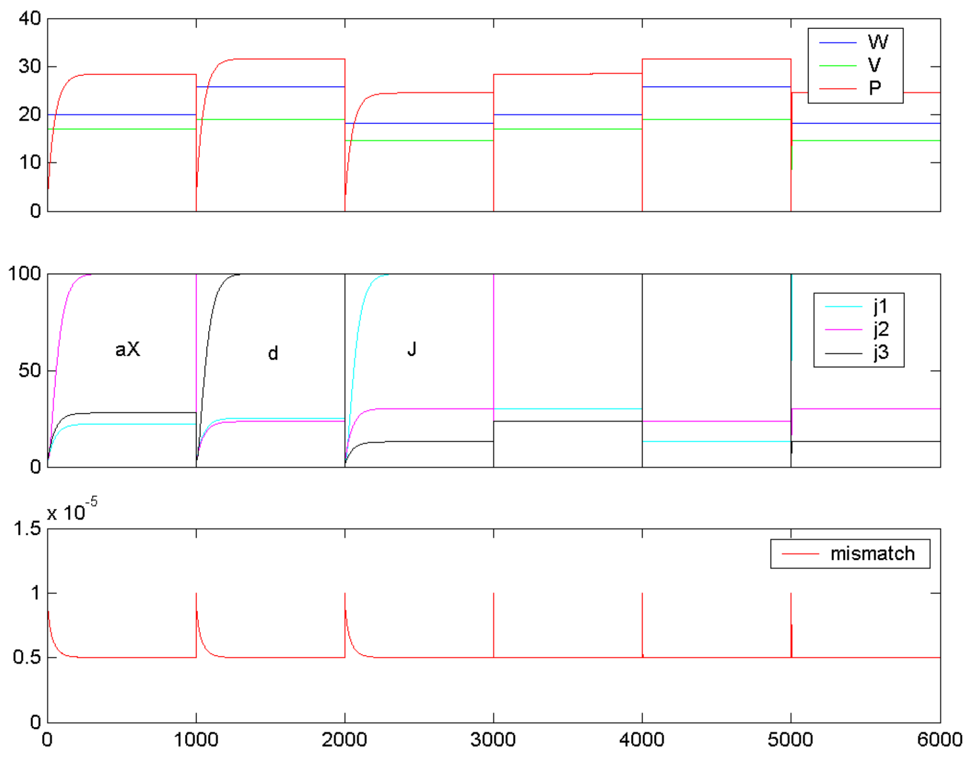


Figure 18.3: Response of ART 2 network to analog X, d, J pattern sequence. The sequence is presented to the network twice in this simulation. The input patterns are effectively stored in LTM after one presentation and properly classified when they are repeated in the latter half of the run.

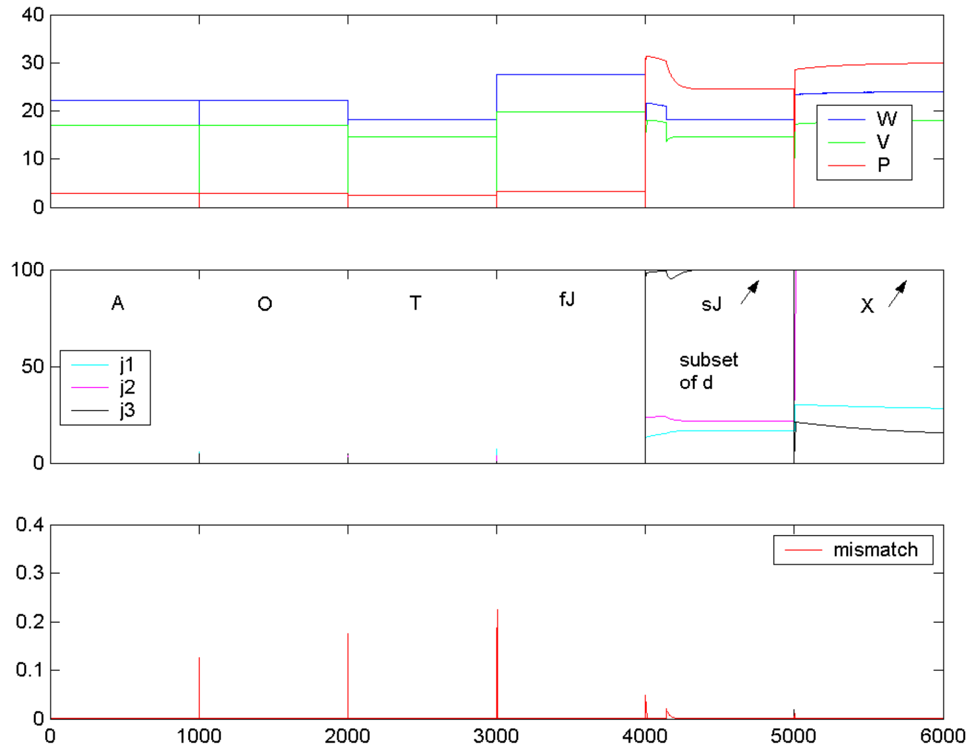


Figure 18.4: Response of the trained network to the pattern sequence A, O, T, fJ, sJ, and X. The network undergoes mismatch resets for the A, O, T, and fJ patterns, resulting in $\mathbf{x_2} = \mathbf{0}$ responses (no classification). The sJ pattern is a subset of the d pattern, and the network recodes its LTM for d to match the sJ pattern. However, upon subsequent re-presentation of d, the network correctly classifies the d pattern into F_2 node 3 (its original classification category). The network recognizes the X pattern as the same category as the analog X. Alternating presentations will recode this category between analog X and X but without misclassifying either pattern.

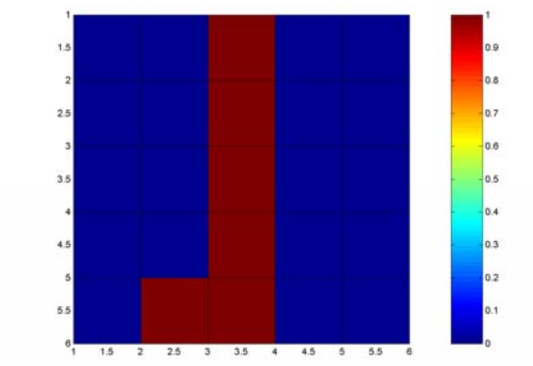
Figure 18.4 illustrates the response of the network to this sequence. The A, O, T, and fJ patterns all produce mismatch resets for each category in F_2, thus indicating a "no classification" response with $\mathbf{x_2} = \mathbf{0}$ for all four patterns. If F_2 had more than $N = 3$ nodes, it would have learned new classification codes for these patterns. The sJ pattern is recognized by the classification category established for the d pattern. However, because the d is a superset pattern of the sJ, the LTM for category 3 (originally the category for the d pattern) is re-coded to represent the sJ. A subsequent presentation of the d pattern (not shown in the figure) results in d still being properly classified as category 3; in other words, the superset pattern is subsumed under the subset category. This, however, did not happen in the case of the fJ pattern; the network's vigilance setting results in fJ being distinguishable from the normal J pattern.

The X pattern is identical to the analog X pattern except for its pixel amplitudes. The network recognizes X using the analog X category. However, it also adapts the LTM weights for this category, resulting in a weight encoding roughly midway between the weight magnitudes for the two patterns. Continued re-presentation of X will result in a re-coding of category 2 to represent the X pixel amplitudes. Nonetheless, re-presentation of analog X is properly classified into this category (and, indeed, LTM will be adapted back toward the original analog X settings).

Categories, once learned, are not absolutely robust. As was just illustrated, the subsequent application of a pattern that is a subset of a learned pattern can result in re-coding of LTM to represent the subset pattern, depending on the setting of the vigilance parameter. The sJ vs. d pattern is an example of this. There are, however, other patterns that are subsets of the sJ. One of these is the "I" pattern illustrated in figure 18.5.

Just as the network recoded the LTM for the d-pattern into an LTM for the sJ pattern, so too will it recode the LTM for the sJ into an LTM encoding for the I pattern if this pattern is presented. Figure 18.6 illustrates the outcome of a sequence I-I-I-d-sJ-d applied to the network after it had recoded the LTM for category 3 (formerly d, now sJ). The network classifies the first presentation of the I pattern under its category for sJ/d (category 3). However, it also adapts its LTM to this new input pattern, resulting in an LTM weight distribution that matches the I pattern. The network still recognizes the sJ pattern under this category, *but now fails to recognize the original d pattern*. Instead, upon presentation of the d pattern, the network generates mismatch resets which result in the d pattern becoming unclassified. Thus, ***at high vigilance the network "prefers" subset patterns***. This behavior is in keeping with the ART network property that subsets should not be re-classified by superset patterns. But, as we can easily see, there is no similar restriction that says supersets cannot be recoded by subsets.

This particular situation arises from the use of high vigilance ($\rho = 0.95$) in the simulation runs


shifted J (sJ) pattern

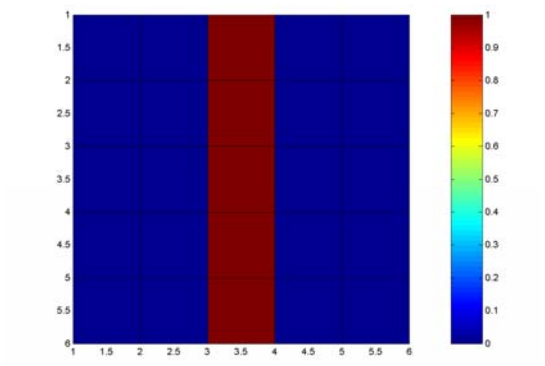

I pattern

Figure 18.5: Illustrations of the sJ and I patterns. The I pattern is a subset pattern of the sJ, which is itself a subset pattern of the d pattern.

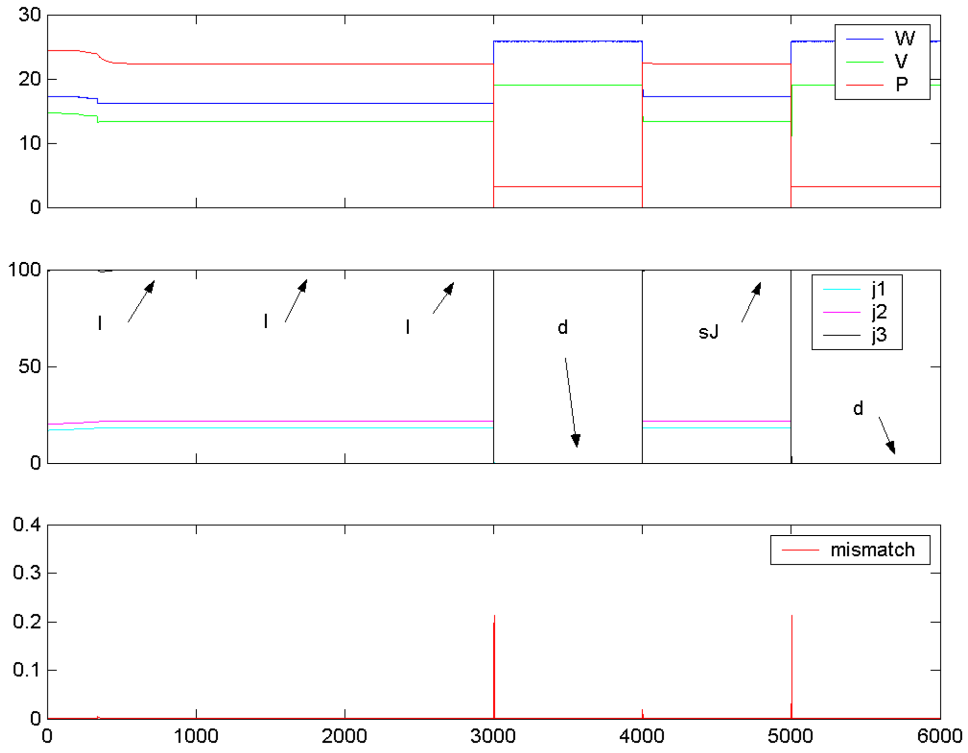


Figure 18.6: Response of the network to an I-I-I-d-sJ-d pattern. The network learns the new subset pattern (I) and recognizes the sJ in this new classification. However, it now fails to recognize the d pattern with which it was originally trained. Application of the d pattern produces mismatch resets.

for the network. Lowering the vigilance parameter will allow the supersets sJ and d to both be classified under the LTM encoding for the I pattern. This is illustrated in figure 18.7 for the case where $\rho = 0.90$. Here the I, d, and sJ pattern are all classified under the category the network establishes for the I pattern.

The ART network literature often makes it a point to stress how LTM pattern classifications are robust and immune from being recoded by later patterns. As this example illustrates, this claim is a bit too broad in networks where the vigilance parameter is set to a high value for making fine discriminations among pattern differences. It is the nature of ART 2 to "prefer" subset patterns to superset patterns in its adaptation responses when vigilance is set high. Since the number of nodes in F_2 is always limited, it is simply a matter of time before all category nodes are committed and the recoding issue just demonstrated surfaces. LTM robustness to recoding is not absolute nor absolutely independent of pattern presentation order. One method of dealing with this issue is to place the vigilance parameter under the control of a second ART 2 network. When this is done, the resulting overall structure is called an ARTMAP [CARP5].

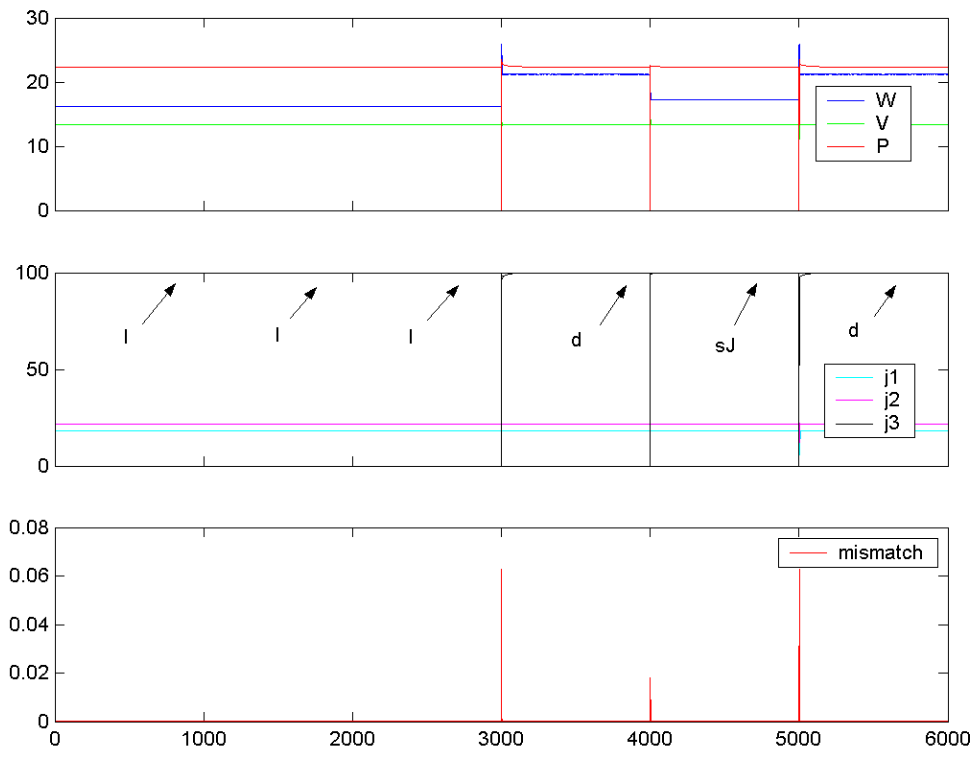


Figure 18.7: Network response to the I-I-I-d-sJ-d sequence with the vigilance parameter lowered from 0.95 to 0.90. The network now classifies all three patterns under the same category.

ARTMAP networks were first introduced as *supervised* learning networks, which at face value would seem to go against Grossberg's oft-repeated objections to the biological realism of networks that rely on a "teacher." However, as was noted earlier in this text, *supervised* does not necessarily imply *teacher* because systems with an actor-critic organization are, in a sense, *self*-supervised. Indeed, another name for ARTMAP networks is ***predictive ART***, which is a functional behavior that advanced actor-critic systems are known for.

Piaget's many years of research revealed that the child in the sensorimotor stage of development does not adapt his sensorimotor schemes unless the scheme is ***frustrated*** (fails to produce the anticipated outcome) [PIAG8]. This, as it turns out, is one of the key ideas in ARTMAP as well. The vigilance parameter is made elastic and is adjusted by a second ART network, using the minimal amount of adjustment needed to satisfy the comparison of the actual to predicted outcomes. This behavior is consistent with what the infant does in the early stages of sensorimotor development.

Generally speaking, high vigilance promotes fine discrimination of differences between input patterns, while low vigilance tends to promote grosser classifications. There are tradeoffs involved in the vigilance parameter setting. As the examples above illustrate, whether a superset pattern is classified in a subset category (sJ into I, for example) or is rejected by mismatch resets (fJ vs. J, for example) depends on both the vigilance setting and on the degree of difference between input patterns.

§ 3. Systematic ART

For many years now a great many neural network theorists – especially those who subscribe to what is known as the "parallel distributed processing" or PDP school of thought – have promoted the idea of a ***universal function approximator*** network. The UFA might be described as a "one anatomy fits all" network architecture. Critics of this idea – your author is one of them – tend to regard the UFA as a fantastic idea, i.e. as an idea that *in practice* is the idea of a fantasy.

> In principle, connectionist networks offer all the potential of universal computing devices. However, our examples of order and coefficient size suggests that various kinds of scaling problems are likely to become obstacles to attempts to exploit that potential. Fortunately, our analysis of perceptrons does not suggest that connectionist networks need always encounter these obstacles. Indeed, our book is rich in surprising examples of tasks that simple perceptrons can perform using relatively low-order units and small coefficients. However, our analysis does show that parallel networks are, in general, subject to serious scaling phenomena. Consequently, researchers who propose such models must show that, in their context, those phenomena do not occur.
>
> The authors of *PDP* seem disinclined to face such problems. They seem content to argue that, although we showed that single-layer networks cannot solve certain problems, we did not know that there could exist a powerful learning procedure for multilayer networks – to which our theorems no longer apply. However, strictly speaking, it is wrong to formulate our findings in terms of what perceptrons can and cannot do. As we pointed out above, perceptrons of sufficiently large order can represent *any* finite predicate. A better description of what we did is that, in certain cases, we established the computational costs of what perceptrons can do as a function of increasing problem size. The authors of *PDP* show little concern for such issues, and usually seem content with experiments in which small multilayer networks solve particular instances of small problems.
>
> What should one conclude from such examples? A person who thinks in terms of *can* vs. *can't* will be tempted to suppose that if toy machines can do something, then larger machines may well do it better. One must always probe into the practicality of a proposed learning algorithm. It is no use to say that "procedure *P* is capable of learning to recognize pattern *X*" unless one can show that this can be done in less time and at less cost than with exhaustive search [MINS: 264-265].

Small neural network systems are adequate for a great many engineering tasks, but our object of study in computational neuroscience is the central nervous system. It is here where issues of "problem size, scaling, and learning procedures" on the large scale are unavoidable. Furthermore, the most rudimentary examination of anatomical and physiological organization in the brain tends to implicate neural structure as being comprised of a large number of different network anatomies rather than a single one-anatomy-fits-all schema of nature.

Is ART guilty of the UFA fantasy? Certainly at times it is easy to get the idea from published papers that the UFA is what is being pursued by ART theorists. But, on the other hand, most of the interesting mainstream ART research involves the application of ART (which, the reader is reminded, is a theory and not just a network) in the context of larger scale *systems*. Within these larger scale systems one finds a number of different anatomies presented.

It is true enough that the on-center/off-surround anatomy is of central importance in ART. It is likewise true that this anatomy is employed over and over again in a great many central systems. But one should not over-generalize this and say that *only* network-systems-of-network-systems comprised of this anatomy are needed to represent brain function. It is true enough that many of the ideas contained in embedding field theory no longer seem to get much print in the literature. But, as Grossberg has pointed out, there are numerous instances where other network structures play an important role. This is especially the case when one is dealing, not merely with atomized "patterns" at a snapshot moment in time, but with ***pattern sequences*** – what Damasio has termed "type II binding codes." One of the early fruits of embedding field theory research was the ***Outstar avalanche*** [GROS11-12], depicted in simplified form in figure 18.8.

Grossberg was able to show that adaptation rules exist for networks such as the one depicted in the example figure, and that such networks are capable of producing arbitrary spatio-temporal patterns. Other ART research publications present systematic anatomies in which are found ART cascades [CARP4], various recurrently-linked ART networks [CARP5], and, more recently, very elegant large-scale models organized into networks of ART networks [LEVId]. Although such model architectures are as different as can be from the idea of a UFA, these models do appear to be more "brain-like" than any single UFA architecture could claim to be. Minsky and Papert remarked,

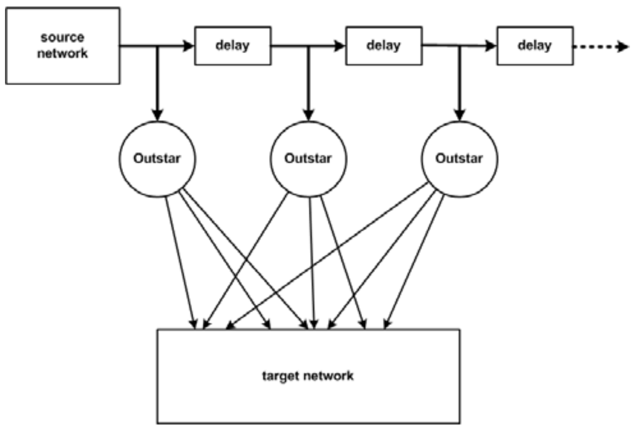


Figure 18.8: Simplified illustration of an Outstar avalanche network. An originating signal from a source network passes through a sequence of delays to activate a sequence of Outstar maps. These project to a target network to produce a time sequence of activation patterns. This particular example is a simple one and more complex cases, including those with retrograde feedback from the target network to the source network, are also possible.

> We think that the difference in abilities comes from the fact that a brain is not a single, uniformly structured network. Instead, each brain contains hundreds of different types of machines, interconnected in specific ways which predestine that brain to become a large, diverse society of partially specialized agencies. . .
>
> Why did our brains evolve so as to contain so many specialized parts? Could not a single, uniform network learn to structure itself into divisions with appropriate architectures and processes? We think this would be impractical because of the problem of representing knowledge. . . It makes no sense to seek the "best" network architecture or learning procedure because it makes no sense to say that *any* network is efficient by itself: that makes sense only in the context of some class of problems to be solved. . . This means that the study of networks in general must include attempts, like those in this book, to classify problems and learning processes; but it must also include attempts to classify network architectures [MINS: 273-274].

A computer engineer – especially an older one – might look at the systematic models in the ART literature and be reminded of a great many digital networks that have come and gone over the years in computer design. Indeed, prior to the 1970s most computer designs even included digital networks that were nothing less than Outstar avalanche networks in everything but name! That most of these older design methods passed out of favor is due principally to advances in technology that made different design approaches more economical and profitable, not to particular shortcomings in the older designs.

In neuroscience we have a specific object of interest – namely the human brain – and this object provides the central focus for all our research. ***Systematic ART*** is the name your author gives to large-scale ART-based brain modeling research aimed at discovering functional network system and large scale network-of-networks anatomies capable of explaining psychophysical phenomena while, at the same time, also possessing the quality of ***linkage*** to biological findings and models at the lower levels in the hierarchy of scientific reduction. To carry out such a research program requires one to have knowledge of *both* psychology (the objects of which are supersensible) *and* biology (the objects of which are sensible objects accessible to physical measurements).

There is a great deal more that can and should be said about this subject. This, however, would carry us past the introductory level to computational neuroscience, this fascinating young science many of us feel holds the key to one day achieving a complete understanding of the human brain and the phenomenon of mind. Many of us, including your author, think neuroscience is destined to be *the* science of the twenty-first century. But as an *introduction* to the subject, the objective of *this* book has now been met and so the time has now arrived to bring this work to a close and say

finis.