Neural Networks, Vol. 4, pp. 565–588, 1991
Printed in the USA. All rights reserved.

0893-6080/91 $3.00 + .00
Copyright © 1991 Pergamon Press plc

ORIGINAL CONTRIBUTION

ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network

GAIL A. CARPENTER[†], STEPHEN GROSSBERG[‡], AND JOHN H. REYNOLDS[§]

Center for Adaptive Systems and Department of Cognitive and Neural Systems

(*Received* 28 *November* 1990; *revised and accepted* 13 *February* 1991)

Abstract—*This article introduces a new neural network architecture, called ARTMAP, that autonomously learns to classify arbitrarily many, arbitrarily ordered vectors into recognition categories based on predictive success. This supervised learning system is built up from a pair of Adaptive Resonance Theory modules (ART_a and ART_b) that are capable of self-organizing stable recognition categories in response to arbitrary sequences of input patterns. During training trials, the ART_a module receives a stream $[\mathbf{a}^{(p)}]$ of input patterns, and ART_b receives a stream $[\mathbf{b}^{(p)}]$ of input patterns, where $\mathbf{b}^{(p)}$ is the correct prediction given $\mathbf{a}^{(p)}$. These ART modules are linked by an associative learning network and an internal controller that ensures autonomous system operation in real time. During test trials, the remaining patterns $\mathbf{a}^{(p)}$ are presented without $\mathbf{b}^{(p)}$, and their predictions at ART_b are compared with $\mathbf{b}^{(p)}$. Tested on a benchmark machine learning database in both on-line and off-line simulations, the ARTMAP system learns orders of magnitude more quickly, efficiently, and accurately than alternative algorithms, and achieves 100% accuracy after training on less than half the input patterns in the database. It achieves these properties by using an internal controller that conjointly maximizes predictive generalization and minimizes predictive error by linking predictive success to category size on a trial-by-trial basis, using only local operations. This computation increases the vigilance parameter ρ_a of ART_a by the minimal amount needed to correct a predictive error at ART_b. Parameter ρ_a calibrates the minimum confidence that ART_a must have in a category, or hypothesis, activated by an input $\mathbf{a}^{(p)}$ in order for ART_a to accept that category, rather than search for a better one through an automatically controlled process of hypothesis testing. Parameter ρ_a is compared with the degree of match between $\mathbf{a}^{(p)}$ and the top-down learned expectation, or prototype, that is read-out subsequent to activation of an ART_a category. Search occurs if the degree of match is less than ρ_a. ARTMAP is hereby a type of self-organizing expert system that calibrates the selectivity of its hypotheses based upon predictive success. As a result, rare but important events can be quickly and sharply distinguished even if they are similar to frequent events with different consequences. Between input trials ρ_a relaxes to a baseline vigilance $\overline{\rho_a}$. When $\overline{\rho_a}$ is large, the system runs in a conservative mode, wherein predictions are made only if the system is confident of the outcome. Very few false-alarm errors then occur at any stage of learning, yet the system reaches asymptote with no loss of speed. Because ARTMAP learning is self-stabilizing, it can continue learning one or more databases, without degrading its corpus of memories, until its full memory capacity is utilized.*

Keywords—ARTMAP, Adaptive resonance theory, Supervised learning, Self-organization, Prediction, Expert system, Mushroom database, Machine learning.

1. INTRODUCTION: PREDICTIVE ART

As we move freely through the world, we can attend to both familiar and novel objects, and can rapidly learn to recognize, test hypotheses about, and learn to name novel objects without unselectively disrupting our memories of familiar objects. This article describes a new self-organizing neural network architecture—called a Predictive ART or ARTMAP architecture—that is capable of fast, yet stable, online recognition learning, hypothesis testing, and adaptive naming in response to an arbitrary stream of input patterns.

† Supported in part by BP (98-A-1204), DARPA (AFOSR 90-0083), and the National Science Foundation (NSF IRI-90-00539).

‡ Supported in part by the Air Force Office of Scientific Research (AFOSR 90-0175 and AFOSR 90-0128), and DARPA (AFOSR 90-0083).

§ Supported in part by DARPA (AFOSR 90-0083).

Acknowledgements: The authors wish to thank Cynthia E. Bradford for her valuable assistance in the preparation of the manuscript.

Requests for reprints should be sent to Professor Gail Carpenter, Center for Adaptive Systems, Boston University, 111 Cummington Street, Boston, MA 02215.

The possibility of stable learning in response to an arbitrary stream of inputs is required by an autonomous learning agent that needs to cope with unexpected events in an uncontrolled environment. One cannot *restrict* the agent's ability to process input sequences if one cannot *predict* the environment in which the agent must successfully function. The ability of humans to vividly remember exciting adventure movies is a familiar example of fast learning in an unfamiliar environment.

1.1. Fast Learning About Rare Events

A successful autonomous agent must be able to learn about rare events that have important consequences, even if these rare events are similar to frequent events with very different consequences. Survival may hereby depend on fast learning in a *nonstationary* environment. Many learning schemes are, in contrast, slow learning models that average over individual event occurrences and are degraded by learning instabilities in a nonstationary environment (Carpenter & Grossberg, 1988; Grossberg, 1988a).

1.2. Many-to-One and One-to-Many Learning

An efficient recognition system needs to be capable of many-to-one learning. For example, each of the different exemplars of the font for a prescribed letter may generate a single compressed representation that serves as a visual recognition category. This exemplar-to-category transformation is a case of many-to-one learning. In addition, many different fonts, including lower case and upper case printed fonts and scripts of various kinds, can all lead to the same verbal name for the letter. This is a second sense in which learning may be many-to-one.

Learning may also be one-to-many, so that a single object can generate many different predictions or names. For example, upon looking at a banana, one may classify it as an oblong object, a fruit, a banana, a yellow banana, and so on. A flexible knowledge system may thus need to represent in its memory many predictions for each object, and to make the best prediction for each different context in which the object is embedded.

1.3. Control of Hypothesis Testing, Attention, and Learning by Predictive Success

Why does not an autonomous recognition system get trapped into learning only that interpretation of an object which is most salient given the system's initial biases? One factor is the ability of that system to reorganize its recognition, hypothesis testing, and naming operations based upon its predictive success or failure. For example, a person may learn a visual recognition category based upon seeing bananas of various colors and associate that category with a certain taste. Due to the variability of color features compared with those of visual form, this learned recognition category may incorporate form features more strongly than color features. However, the color green may suddenly, and unexpectedly, become an important differential predictor of a banana's taste.

The different taste of a green banana triggers hypothesis testing that shifts the focus of visual attention to give greater weight, or salience, to the banana's color features without negating the importance of the other features that define a banana's form. A new visual recognition category can hereby form for green bananas, and this category can be used to accurately predict the different taste of green bananas. The new, finer category can form, moreover, without recoding either the previously learned generic representation of bananas or their taste association.

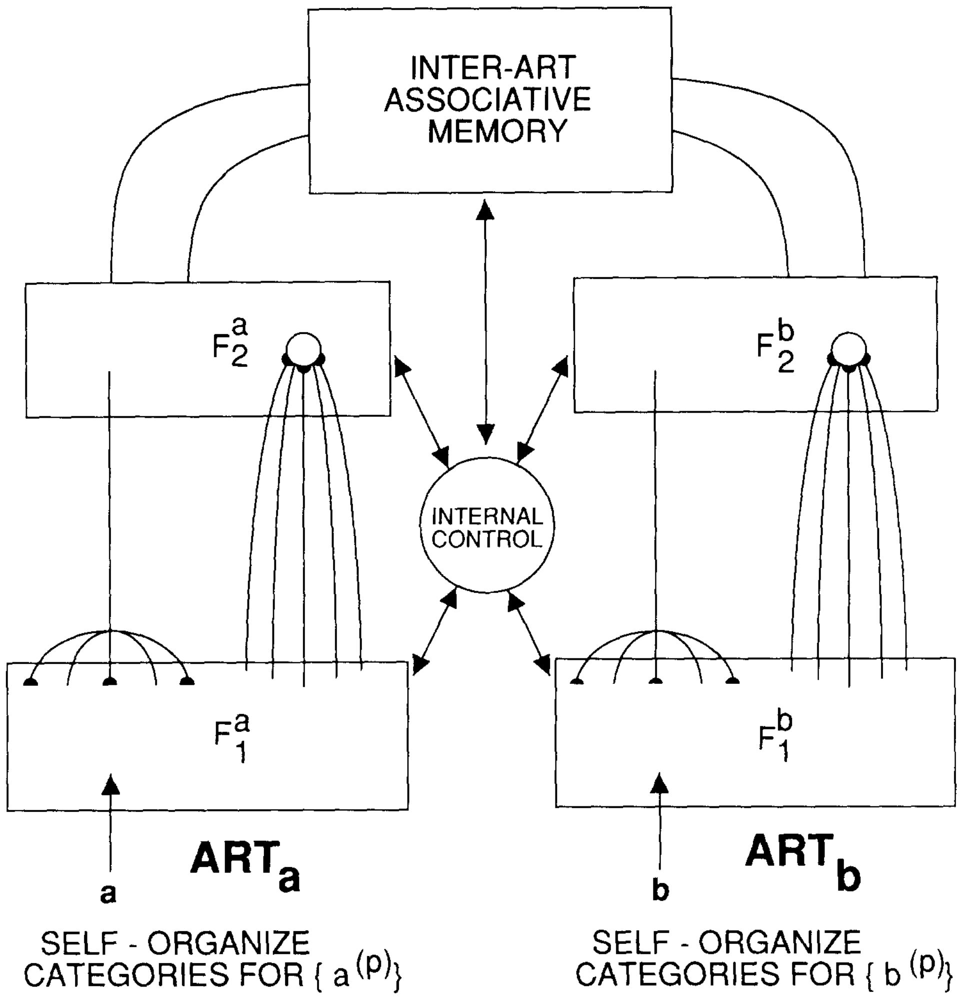
Future representations may also form that incorporate new knowledge about bananas, without disrupting the representations that are used to predict their different tastes. In this way, predictive feedback provides one means whereby one-to-many recognition and prediction codes can form through time, by using hypothesis testing and attention shifts that support new recognition learning without forcing unselective forgetting of previous knowledge.

1.4. Adaptive Resonance Theory

The architecture described herein forms part of Adaptive Resonance Theory, or ART, which was introduced in 1976 (Grossberg, 1976a, 1976b) in order to analyze how brain networks can autonomously learn in real time about a changing world in a rapid but stable fashion. Since that time, ART has steadily developed as a physical theory to explain and predict ever larger data bases about cognitive information processing and its neural substrates (Grossberg, 1982a, 1987a, 1987b, 1988b). A parallel development has described a series of rigorously characterized neural architectures—called ART 1, ART 2, and ART 3—with increasingly powerful learning, pattern recognition, and hypothesis testing capabilities (Carpenter & Grossberg, 1987a, 1987b, 1988, 1990).

1.5. Self-Organizing Predictive Maps

The present class of architectures are called Predictive ART architectures because they incorporate ART modules into systems that can learn to predict a prescribed m-dimensional output vector $\mathbf{b}$ given a prescribed n-dimensional input vector $\mathbf{a}$ (Figure 1). The present example of Predictive ART is called ARTMAP because its transformation from vectors


INTER-ART ASSOCIATIVE MEMORY
F_2^a
F_2^b
INTERNAL CONTROL
F_1^a
F_1^b
ART_a
ART_b
a
b
SELF - ORGANIZE CATEGORIES FOR { $a^{(p)}$}
SELF - ORGANIZE CATEGORIES FOR { $b^{(p)}$}

	Predictive ART	Back Propagation
supervised	yes	yes
self-organizing	yes	no
real-time	yes	no
self-stabilizing	yes	no
learning:	fast or slow match	slow mismatch

FIGURE 1. A Predictive ART, or ARTMAP, system includes two ART modules linked by an inter-ART associative memory. Internal control structures actively regulate learning and information flow. Back Propagation and Predictive ART both carry out supervised learning, but the two systems differ in many respects, as indicated.

in $\Re^n$ to vectors in $\Re^m$ defines a *map* that is learned by example from the correlated pairs $\{\mathbf{a}^{(p)}, \mathbf{b}^{(p)}\}$ of sequentially presented vectors, $p = 1, 2, \ldots$ (Carpenter, 1989). For example, the vectors $\mathbf{a}^{(p)}$ may encode visual representations of objects, and the vectors $\mathbf{b}^{(p)}$ may encode their predictive consequences, such as different tastes in the banana example above. The degree of code compression in memory is an index of the system's ability to *generalize* from examples.

Figure 1 compares properties of the ARTMAP network with those of the Back Propagation network (Parker, 1982; Rumelhart & McClelland, 1986; Werbos, 1974, 1982). Both ARTMAP and Back Propagation are supervised learning systems. With supervised learning, an input vector $\mathbf{a}^{(p)}$ is associated with another input vector $\mathbf{b}^{(p)}$ on each training trial. On a test trial, a new input **a** is presented that has never been experienced before. This input predicts an output vector **b.** System performance is evaluated by comparing **b** with the correct answer. This property of *generalization* is the system's ability to correctly predict correct answers to a test set of novel inputs **a.**

1.6. Conjointly Maximizing Generalization and Minimizing Predictive Error

The ARTMAP system is designed to conjointly *maximize* generalization and *minimize* predictive error under *fast learning* conditions in *real time* in response to an *arbitrary ordering* of input patterns. Remarkably, the network can achieve 100% test set accuracy on the machine learning benchmark database described below. Each ARTMAP system learns to make accurate predictions quickly, in the sense of using relatively little computer time; efficiently, in the sense of using relatively few training trials; and flexibly, in the sense that its stable learning permits continuous new learning, on one or more databases, without eroding prior knowledge, until the full memory capacity of the network is exhausted. In an ARTMAP network, the memory capacity can be chosen arbitrarily large without sacrificing the stability of fast learning or accurate generalization.

1.7. Match Tracking of Predictive Confidence by Attentive Vigilance

An essential feature of the ARTMAP design is its ability to conjointly maximize generalization and minimize predictive error on a *trial-by-trial* basis using *only local operations.* It is this property which enables the system to learn rapidly about rare events that have important consequences even if they are very similar to frequent events with different consequences. This property builds upon a key design feature of all ART systems; namely, the existence of an *orienting subsystem* that responds to the unexpectedness, or novelty, of an input exemplar **a** by driving a hypothesis testing cycle, or parallel memory search, for a better, or totally new, recognition category for **a.** Hypothesis testing is triggered by the orienting subsystem if **a** activates a recognition category that reads out a learned expectation, or prototype, which does not match **a** well enough. The degree of match provides an analog measure of the predictive *confidence* that the chosen recognition category represents **a,** or of the *novelty* of **a** with respect to the hypothesis that is symbolically represented by the recognition category. This analog match value is computed at the orienting subsystem where it is compared with a dimensionless parameter that is called *vigilance* (Carpenter & Grossberg, 1987a, 1987b). A cycle of hypothesis testing is triggered if the degree of match is less than vigilance. Conjoint maximization of generalization and minimization of predictive error is achieved on a trial-by-trial basis by increasing the vigilance parameter in response to a predictive error on a training trial (Carpenter & Grossberg, 1987a). The minimum change is made that is consistent with correction of the error. In fact, the predictive error causes the vigilance to increase rapidly until it just exceeds the analog match value, in a process called *match tracking.*

Before each new input arrives, vigilance relaxes to a baseline vigilance value. Setting baseline vigilance to 0 maximizes code compression. The system accomplishes this by allowing an "educated guess" on every trial, even if the match between input and learned code is poor. Search ensues, and a new category is established, only if the prediction made in this forced-choice situation proves wrong. When predictive error carries a cost, however, baseline vigilance can be set at some higher value, thereby decreasing the "false alarm" rate. With positive baseline vigilance, the system responds "I don't know" to an input that fails to meet the minimum matching criterion. Predictive error rate can hereby be made very small, but with a reduction in code compression. Search ends when the internal control system (Figure 1) determines that a global consensus has been reached.

1.8. Self-Organizing Expert System

ARTMAP achieves its combination of desirable properties by acting as a type of self-organizing expert system. It incorporates the basic properties of all ART systems (Carpenter & Grossberg, 1988) to carry out autonomous hypothesis testing and parallel memory search for appropriate recognition codes. Hypothesis testing terminates in a sustained state of resonance that persists as long as an input remains approximately constant. The resonance generates a focus of attention that selects the bundle of critical features common to the bottom-up input and the top-down expectation, or prototype, that is read-out by the resonating recognition category. Learning of the critical feature pattern occurs in this resonant and attentive state, hence the term *adaptive resonance.*

1.9. 2/3 Rule Matching, Priming, Intentionality, and Logic

The resonant focus of attention is a consequence of a matching rule called the 2/3 Rule (Carpenter & Grossberg, 1987a). This rule clarifies how a bottom-up input pattern can *supraliminally* activate its feature detectors at the level F_1 of an ART network, yet a top-down expectation can only *subliminally* sensitize, or *prime*, the level F_1. Supraliminal activation means that F_1 can automatically generate output signals that initiate further processing of the input. Subliminal activation means that F_1 cannot generate output signals, but its primed cells can more easily be activated by bottom-up inputs. For example, the verbal command "Look for the yellow banana" can prime visual feature detectors to respond more sensitively to visual inputs that represent a yellow banana, without forcing these cells to be fully activated, which would have caused a visual hallucination.

Carpenter and Grossberg (Grossberg, 1987a) have shown that the 2/3 Rule is realized by a kind of analog spatial logic. This logical operation computes the spatial intersection of bottom-up and top-down information. The spatial intersection is the focus of attention. It is of interest that subliminal top-down priming, which instantiates a type of "intentionality" in an ART system, implies a type of matching law, which instantiates a type of "logic." Searle (1983) and others have criticized some Artificial Intelligence models because they sacrifice intentionality for logic. In ART, intentionality implies logic.

2. THE ARTMAP SYSTEM

The main elements of an ARTMAP system are shown in Figure 2. Two modules, ART_a and ART_b, read vector inputs **a** and **b**. If ART_a and ART_b were disconnected, each module would self-organize category groupings for the separate input sets. In the application described below, ART_a and ART_b are fast-learn ART 1 modules coding binary input vec-

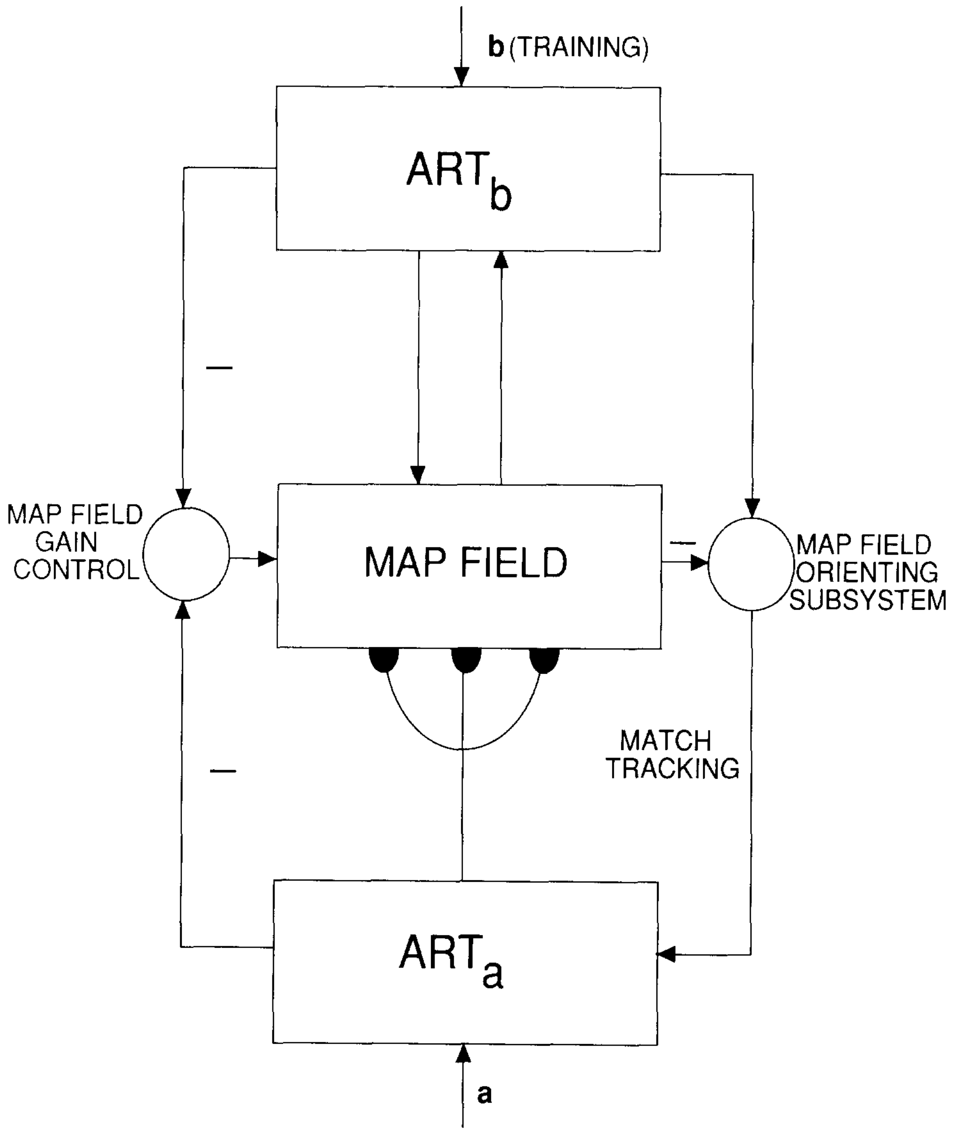


FIGURE 2. Block diagram of an ARTMAP system. Modules ART_a and ART_b self-organize categories for vector sets a and b. ART_a and ART_b are connected by an inter-ART module that consists of the Map Field and the control nodes called Map Field gain control and Map Field orienting subsystem. Inhibitory paths are denoted by a minus sign; other paths are excitatory.

tors. ART_a and ART_b are here connected by an inter-ART module that in many ways resembles ART 1. This inter-ART module includes a *Map Field* that controls the learning of an associative map from ART_a recognition categories to ART_b recognition categories. This map does not directly associate exemplars **a** and **b**, but rather associates the compressed and symbolic representations of families of exemplars **a** and **b**. The Map Field also controls match tracking of the ART_a vigilance parameter. A mismatch at the Map Field between the ART_a category activated by an input **a** and the ART_b category activated by the input **b** increases ART_a vigilance by the minimum amount needed for the system to search for and, if necessary, learn a new ART_a category whose prediction matches the ART_b category.

This inter-ART vigilance resetting signal is a form of "back propagation" of information, but one that differs from the back propagation that occurs in the Back Propagation network. For example, the search initiated by inter-ART reset can shift attention to a novel cluster of visual features that can be incorporated through learning into a new ART_a recognition category. This process is analogous to learning a category for "green bananas" based on "taste" feedback. However, these events do not "back propagate" taste features into the visual representation of the bananas, as can occur using the Back Propagation network. Rather, match tracking reorganizes the way in which visual features are grouped, attended, learned, and recognized for purposes of predicting an expected taste.

The following sections describe ARTMAP simulations using a machine learning benchmark database. The ARTMAP system is then described mathematically. The Appendix summarizes ART 1 and ARTMAP system equations for purposes of simulation, and outlines system responses to various input protocols.

3. ARTMAP SIMULATIONS: DISTINGUISHING EDIBLE AND POISONOUS MUSHROOMS

The ARTMAP system was tested on a benchmark machine learning database that partitions a set of vectors **a** into two classes. Each vector **a** characterizes observable features of a mushroom as a binary vector, and each mushroom is classified as edible or poisonous (Schlimmer, 1987a). The database represents the 11 species of genus *Agaricus* and the 12 species of the genus *Lepiota* described in "The Audubon Society Field Guide to North American Mushrooms" (Lincoff, 1981). These two genera constitute most of the mushrooms described in the "Field Guide" from the family *Agaricaceae* (order *Agaricales*, class *Hymenomycetes*, subdivision *Basidiomycetes*, division *Eumycota*). All the mushrooms represented in the database are similar to one another: "These mushrooms are placed in a single family on the basis of a correlation of characteristics that include microscopic and chemical features . . ." (Lincoff, 1981, p. 500). The "Field Guide" warns that poisonous and edible species can be difficult to distinguish on the basis of their observable features. For example, the poisonous species *Agaricus californicus* is described as a "dead ringer" (Lincoff, 1981, p. 504) for the Meadow Mushroom, *Agaricus campestris*, that "may be known better and gathered more than any other wild mushroom in North America" (Lincoff, 1981, p. 505). This database thus provides a test of how ARTMAP and other machine learning systems distinguish rare but important events from frequently occurring collections of similar events that lead to different consequences.

The database of 8,124 exemplars describes each of 22 observable features of a mushroom, along with its classification as poisonous (48.2%) or edible (51.8%). The 8,124 "hypothetical examples" represent ranges of characteristics within each species; for example, both *Agaricus californicus* and *Agaricus campestris* are described as having a "white to brownish cap," so in the database each species has corresponding sets of exemplar vectors representing their range of cap colors. There are 126 different values of the 22 different observable features. A list of the observable features and their possible values is given in Table 1. For example, the observable feature of "cap-shape" has six possible values. Consequently, the vector inputs to ART_a are 126-element binary vectors, each vector having 22 1's and 104 0's, to denote the values of an exemplar's 22 observable features. The ART_b input vectors are (1, 0) for poisonous exemplars and (0, 1) for edible exemplars.

3.1. Performance

The ARTMAP system learned to classify test vectors rapidly and accurately, and system performance compares favorably with results of other machine learning algorithms applied to the same database. The STAGGER algorithm reached its maximum performance level of 95% accuracy after exposure to 1,000 training inputs (Schlimmer, 1987b). The HILLARY algorithm achieved similar results (Iba, Wogulis, & Langley, 1988). The ARTMAP system consistently achieved over 99% accuracy with 1,000 exemplars, even counting "I don't know" responses as errors. Accuracy of 95% was usually achieved with on-line training on 300–400 exemplars and with off-line training on 100–200 exemplars. In this sense, ARTMAP was an order of magnitude more efficient than the alternative systems. In addition, with continued training, ARTMAP predictive accuracy always improved to 100%. These results are elaborated below.

TABLE 1
126 Values of 22 Observable Features Represented in ART_a Input Vectors

Number	Feature	Possible Values
1	Cap-Shape	Bell, Conical, Convex, Flat, Knobbed, Sunken
2	Cap-Surface	Fibrous, Grooves, Scaly, Smooth
3	Cap-Color	Brown, Buff, Gray, Green, Pink, Purple, Red, White, Yellow, Cinnamon
4	Bruises	Bruises, No Bruises
5	Odor	None, Almond, Anise, Creosote, Fishy, Foul, Musty, Pungent, Spicy
6	Gill-Attachment	Attached, Descending, Free, Notched
7	Gill-Spacing	Close, Crowded, Distant
8	Gill-Size	Broad, Narrow
9	Gill-Color	Brown, Buff, Orange, Gray, Green, Pink, Purple, Red, White, Yellow, Chocolate, Black
10	Stalk-Shape	Enlarging, Tapering
11	Stalk-Root	Bulbous, Club, Cup, Equal, Rhizomorphs, Rooted, Missing
12	Stalk-Surface-Above-Ring	Fibrous, Silky, Scaly, Smooth
13	Stalk-Surface-Below-Ring	Fibrous, Silky, Scaly, Smooth
14	Stalk-Color-Above-Ring	Brown, Buff, Orange, Gray, Pink, Red, White, Yellow, Cinnamon
15	Stalk-Color-Below-Ring	Brown, Buff, Orange, Gray, Pink, Red, White, Yellow, Cinnamon
16	Veil-Type	Partial, Universal
17	Veil-Color	Brown, Orange, White, Yellow
18	Ring-Number	None, One, Two
19	Ring-Type	None, Cobwebby, Evanescent, Flaring, Large, Pendant, Sheathing, Zone
20	Spore-Print-Color	Brown, Buff, Orange, Green, Purple, White, Yellow, Chocolate, Black
21	Population	Abundant, Clustered, Numerous, Scattered, Several, Solitary
22	Habitat	Grasses, Leaves, Meadows, Paths, Urban, Waste, Woods

Almost every ARTMAP simulation was completed in under 2 minutes on an IRIS 4D computer, with total time ranging from about 1 minute for small training sets to 2 minutes for large training sets. This is comparable to 2–5 minutes on a SUN 4 computer. Each timed simulation included a total of 8,124 training and test samples, run on a time-sharing system with nonoptimized code. Each 1–2 minute computation included data read-in and read-out, training, testing, and calculation of multiple simulation indices.

3.2. On-Line Learning

On-line learning imitates the conditions of a human or machine operating in a natural environment. An input **a** arrives, possibly leading to a prediction. If made, the prediction may or may not be confirmed. Learning ensues, depending on the accuracy of the prediction. Information about past inputs is available only through the present state of the system. Simulations of on-line learning by the ARTMAP system use each sample pair (**a, b**) as both a test item and a training item. Input **a** first makes a prediction that is compared with **b.** Learning follows as dictated by the internal rules of the ARTMAP architecture.

Four types of on-line simulations were carried out, using two different baseline settings of the ART_a vigilance parameter ρ_a: $\overline{\rho_a} = 0$ (forced choice condition) and $\overline{\rho_a} = 0.7$ (conservative condition); and using sample replacement or no sample replacement. With sample replacement, any one of the 8,124 input samples was selected at random for each input presentation. A given sample might thus be repeatedly encountered while others were still unused. With no sample replacement, a sample was removed from the input pool after it was first encountered. The replacement condition had the advantage that repeated encounters tended to boost predictive accuracy. The no-replacement condition had the advantage of having learned from a somewhat larger set of inputs at each point in the simulation. The replacement and no-replacement conditions had similar performance indices, all other things being equal. Each of the 4 conditions was run on 10 independent simulations. With $\overline{\rho_a} = 0$, the system made a prediction in response to every input. Setting $\overline{\rho_a} = 0.7$ increased the number of "I don't know" responses, increased the number of ART_a categories, and decreased the rate of incorrect predictions to nearly 0%, even early in training. The $\overline{\rho_a} = 0.7$ condition generally outperformed the $\overline{\rho_a} = 0$ condition, even when incorrect predictions and "I don't know" responses were both counted as errors. The primary exception occurred very early in training, when a conservative system gives the large majority of its no-prediction responses.

Results are summarized in Table 2. Each entry gives the number of correct predictions over the previous 100 trials (input presentations), averaged over 10 simulations. For example, with $\overline{\rho_a} = 0$ in the no-replacement condition, the system made, on the average, 94.9 correct predictions and 5.1 incorrect predictions on trials 201–300. In all cases a 95% correct-prediction rate was achieved before trial 400. With $\overline{\rho_a} = 0$, a consistent correct-prediction rate of over 99% was achieved by trial 1,400, while with $\overline{\rho_a} = 0.7$ the 99% consistent correct-prediction rate was achieved earlier, by trial 800. Each simulation was continued for 8,100 trials. In all four cases, the minimum correct-prediction rate always exceeded 99.5% by trial 1,800 and always exceeded 99.8% by trial 2,800. In all cases, across the total of 40 simulations summarized in Table 2, 100% correct prediction was achieved on the last 1,300 trials of each run.

Note the relatively low correct-prediction rate for $\overline{\rho_a} = 0.7$ on the first 100 trials. In the conservative mode, a large number of inputs initially make no prediction. With $\overline{\rho_a} = 0.7$ an average total of only 2 *incorrect* predictions were made on each run of 8,100 trials. Note too that Table 2 underestimates prediction accuracy at any given time, since performance almost always improves during the 100 trials over which errors are tabulated.

3.3. Off-Line Learning

In off-line learning, a fixed training set is repeatedly presented to the system until 100% accuracy is achieved on that set. For training sets ranging in size from 1 to 4,000 samples, 100% accuracy was almost always achieved after one or two presentations of each training set. System performance was then measured on the test set, which consisted of all 8,124 samples not included in the training set. During testing no further learning occurred.

The role of repeated training set presentations was examined by comparing simulations that used the 100% training set accuracy criterion with simulations that used only a single presentation of each input during training. With only a few exceptions, performance was similar. In fact, for $\overline{\rho_a} = 0.7$, and for small training sets with $\overline{\rho_a} = 0$, 100% training-set accuracy was achieved with single input presentations, so results were identical. Performance differences were greatest for $\overline{\rho_a} = 0$ simulations with mid-sized training sets (60–500 samples), when 2–3 training set presentations tended to add a few more ART$_a$ learned category nodes. Thus, even a single presentation of training-then-testing inputs, carried out on-

TABLE 2
On-line Learning and Performance in Forced Choice ($\overline{\rho_a} = 0$) or Conservative ($\overline{\rho_a} = 0.7$) Cases, With Replacement or No Replacement of Samples After Training

	Average number of correct predictions on previous 100 trials			
Trial	$\overline{\rho_a} = 0$ No Replace	$\overline{\rho_a} = 0$ Replace	$\overline{\rho_a} = 0.7$ No Replace	$\overline{\rho_a} = 0.7$ Replace
100	82.9	81.9	66.4	67.3
200	89.8	89.6	87.8	87.4
300	94.9	92.6	94.1	93.2
400	95.7	95.9	96.8	95.8
500	97.8	97.1	97.5	97.8
600	98.4	98.2	98.1	98.2
700	97.7	97.9	98.1	99.0
800	98.1	97.7	99.0	99.0
900	98.3	98.6	99.2	99.0
1000	98.9	98.5	99.4	99.0
1100	98.7	98.9	99.2	99.7
1200	99.6	99.1	99.5	99.5
1300	99.3	98.8	99.8	99.8
1400	99.7	99.4	99.5	99.8
1500	99.5	99.0	99.7	99.6
1600	99.4	99.6	99.7	99.8
1700	98.9	99.3	99.8	99.8
1800	99.5	99.2	99.8	99.9
1900	99.8	99.9	99.9	99.9
2000	99.8	99.8	99.8	99.8

line, can be made to work almost as well as off-line training that uses repeated presentations of the training set. This is an important benefit of fast learning controlled by a match tracked search.

Under all training conditions, each of the 8,124 ART_a input vectors is a 126-dimensional binary vector with 22 positive entries. Simulation dynamics are illustrated by projecting these vectors onto the first two principal components of the data set (Kendall & Stuart, 1966). These two components represent 31% of the total variance of the data set.

Figure 3a shows the projections of all 3,916 exemplars representing poisonous mushrooms, and Figure 3b shows the 4,208 exemplars representing edible mushrooms. These figures show that, in these two dimensions, certain clusters are readily distinguishable, such as the clusters of poisonous samples on the top and left portions of Figure 3a. However,

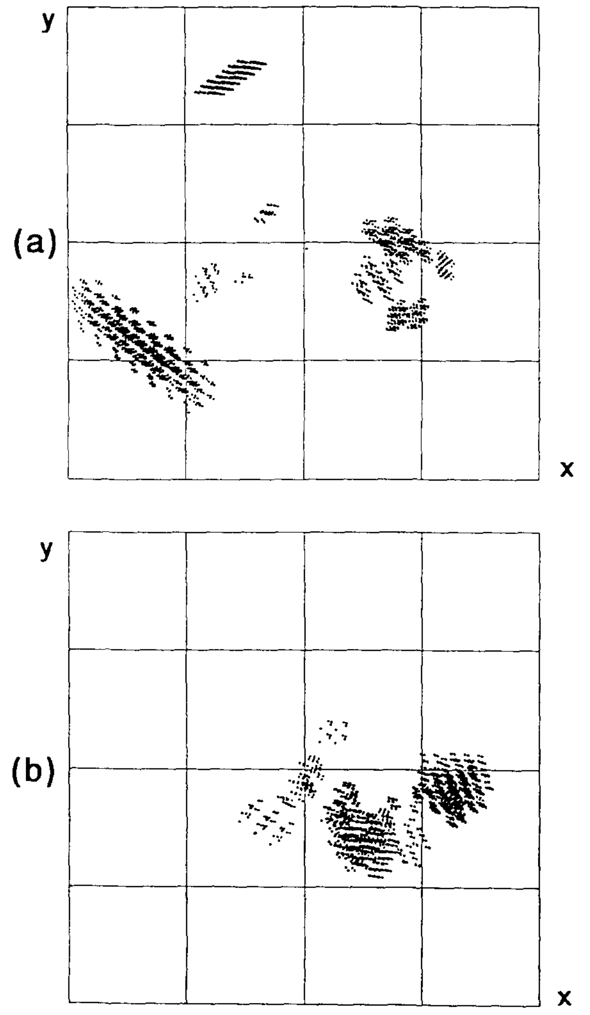


FIGURE 3. Mushroom observable feature data projected onto first 2 principal components. Each point represents a 126-dimensional ART_a input vector. Axes are scaled to run from -1 to $+1$. (a) 3,916 exemplars representing poisonous mushrooms (48.2%); (b) 4,208 exemplars representing edible mushrooms (51.8%).

poisonous and edible samples are densely mixed near the positive x-axis.

3.4. Off-Line Forced-Choice Learning

The simulations summarized in Figure 4 and Table 3 illustrate off-line learning with $\overline{\rho_a} = 0$. In this forced choice case, each ART_a input led to a prediction of poisonous or edible. The number of test set errors with small training sets was relatively large, due to the forced choice.

Figure 4 shows the evolution of test set errors as the training set is increased in size from 5 to 500. In Figure 4a, a set of 5 randomly chosen exemplars (3 poisonous, 2 edible) established 2 ART_a categories (1 poisonous, 1 edible) during training. For each of the 8,119 test set exemplars, the system was forced to choose between poisonous and edible, even if no category representation was a close match. The system made 73% correct predictions. Many of the errors were in the dense cluster of poisonous exemplars in the upper quarter of the graph (Figure 3a). By chance, this cluster was not represented in the 5-sample training set.

Table 3 summarizes the average results over 10 simulations at each size training set. For example, with very small, 5-sample training sets, the system established between 1 and 5 ART_a categories, and averaged 73.1% correct responses on the remaining 8,119 test patterns. Success rates ranged from chance (51.8%, 1 category) in one instance where all 5 training set exemplars happened to be edible, to surprisingly good (94.2%, 2 categories). The range of success rates for fast-learn training on very small training sets illustrates the statistical nature of the learning process. Intelligent sampling of the training set or, as here, good luck in the selection of representative samples, can dramatically alter early success rates. In addition, the evolution of internal category memory structure, represented by a set of ART_a category nodes and their top-down learned expectations, is influenced by the selection of early exemplars. Nevertheless, despite the individual nature of learning rates and internal representations, all the systems eventually converge to 100% accuracy on test set exemplars using only (approximately) 1/600 as many ART_a categories as there are inputs to classify.

Figure 4 and Table 3 summarize the rate at which learning converges to 100% accuracy. In Figure 4b, 25 exemplars were added to the 5 used for Figure 4a, and the resulting 30-sample training set was presented to a new ARTMAP system. The 25 additional training exemplars increased the number of ART_a categories to 3 and improved the test set correct-prediction rate to 92.3%. The addition of poisonous training exemplars in the upper quarter of the graph

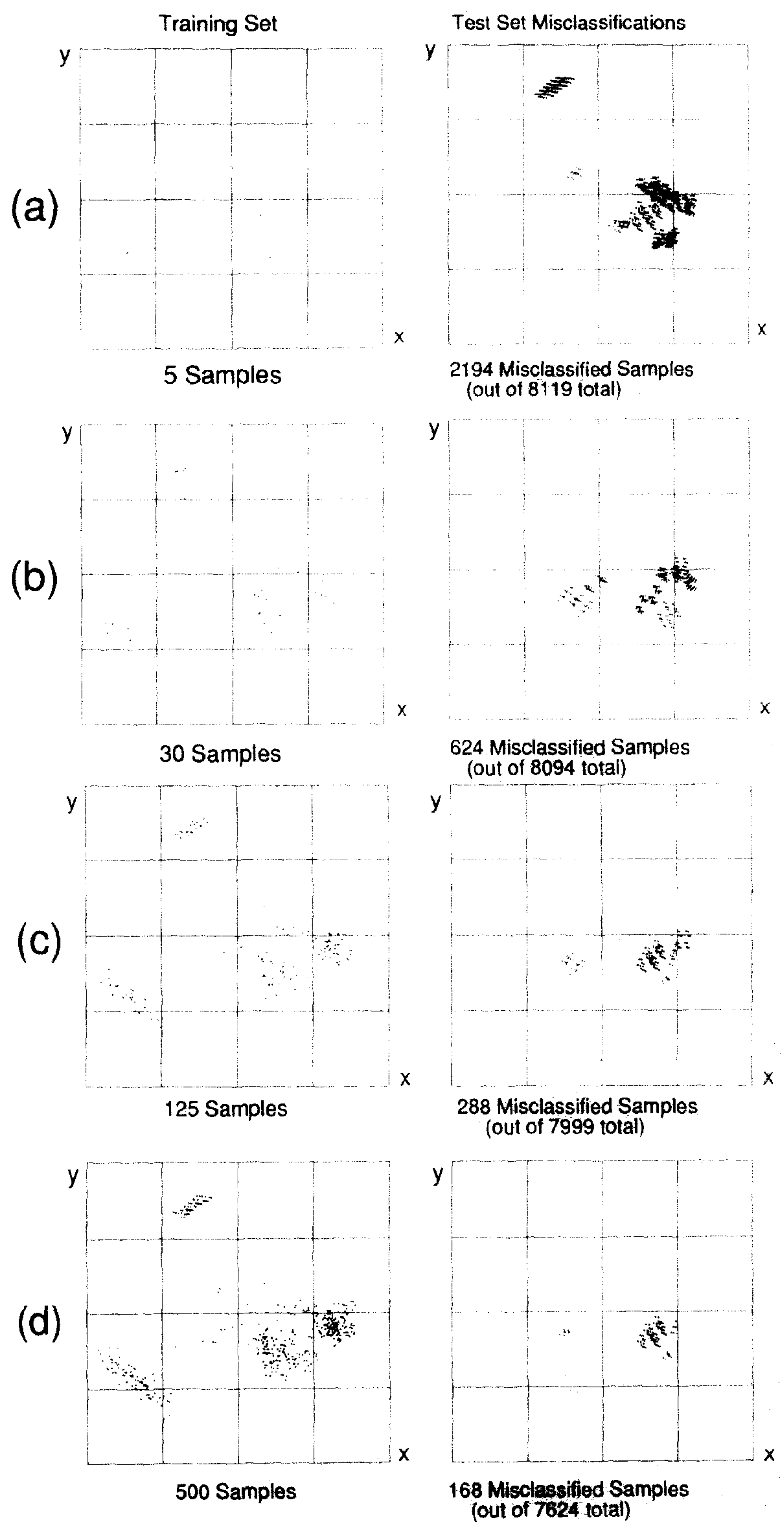


FIGURE 4. Training sets of increasing size (left column) and test set exemplars that were incorrectly classified (right column), projected onto first two principal components. Baseline vigilance $\overline{\rho_a}$ equals 0. (a) With a 5-sample training set that established 2 ART$_a$ categories, the test set of 8,119 inputs made 2,194 errors (27.0%). On 10 other 5-sample runs, the number of ART$_a$ categories ranged from 1 to 5 and the error rate ranged from 5.8% to 48.2%, averaging 26.9%; (b) With a 30-sample training set that established 3 ART$_a$ categories, the test set of 8,094 inputs made 624 errors (7.7%). On 10 other 30-training sample runs, the number of ART$_a$ categories ranged from 4 to 6; and the error rate ranged from 6.7% to 25.1%, averaging 12.4%; (c) With a 125-sample training set that established 9 ART$_a$ categories, the test set of 7,999 inputs made 288 errors (3.6%). On 10 other 125-training sample runs, the number of ART$_a$ categories ranged from 5 to 14, and the error rate ranged from 1.2% to 8.5%, averaging 4.4%; (d) With a 500-sample training set that established 15 ART$_a$ categories, the test set of 7,624 inputs made 168 errors (2.2%). On 10 other 500-training sample runs, the number of ART$_a$ categories ranged from 9 to 22; and the error rate ranged from 0.7% to 3.1%, averaging 1.6%.

TABLE 3
Off-line Forced Choice ($\overline{\rho_a} = 0$) ARTMAP System Performance After Training on Input Sets Ranging in Size From 3 to 4,000 Exemplars. Each Line Shows Average Correct and Incorrect Test Set Predictions Over 10 Independent Simulations, Plus the Range of Learned ART$_a$ Category Numbers

Training Set Size	Average % Correct (Test Set)	Average % Incorrect (Test Set)	Number of ART$_a$ Categories
3	65.8	34.2	1–3
5	73.1	26.9	1–5
15	81.6	18.4	2–4
30	87.6	12.4	4–6
60	89.4	10.6	4–10
125	95.6	4.4	5–14
250	97.8	2.2	8–14
500	98.4	1.6	9–22
1000	99.8	0.2	7–18
2000	99.96	0.04	10–16
4000	100	0	11–22

eliminated all errors there. However, errors persisted for exemplars near the positive x-axis. On 10 other simulations with 30-sample training sets, the correct prediction rate averaged 87.6% and ranged from 74.9% (4 categories) to 93.3% (6 categories).

The simulation that generated Figure 4c added 95 training samples to the 30 used for Figure 4b. The number of ART$_a$ categories increased to 9 and the correct prediction rate increased to 96.4%. On 10 other simulations with 125 randomly chosen training exemplars, the correct-prediction rate averaged 95.6%, ranging from 91.5% (10 categories) to 98.8% (9 categories).

The simulation of Figure 4d added 375 samples to the set used in Figure 4c. This 500-sample training set increased the correct-prediction rate to 97.8% on the test set, establishing 15 categories. On 10 other runs, each with 500 randomly chosen training exemplars, the correct-prediction rate averaged 98.4%, ranging from 96.9% (14 categories) to 99.3% (9 categories). The low error rate of this latter 9-category simulation appears to reflect success of early sampling. On other runs, additional categories were added as errors in early category structures were detected.

With 1,000-sample training sets, 3 out of 10 simulations achieved 100% prediction accuracy on the 7,124-sample test set. With 2,000-sample training sets, 8 out of 10 simulations achieved 100% accuracy on the 6,124-sample test sets. With 4,000-sample training sets, all simulations achieved 100% accuracy on the 4,124-sample test sets. In all, 21 of the 30 simulations with training sets of 1,000, 2,000, and 4,000 samples achieved 100% accuracy on test sets. The number of categories established during these 21 simulations ranged from 10 to 22, again indicating the variety of paths leading to 100% correct prediction rate.

3.5. Off-Line Conservative Learning

As in the case of poisonous mushroom identification, it may be important for a system to be able to respond "I don't know" to a novel input, even if the total number of correct classifications thereby decreases early in learning. For higher values of the baseline vigilance $\overline{\rho_a}$, the ARTMAP system creates more ART$_a$ categories during learning and becomes less able to generalize from prior experience than when $\overline{\rho_a}$ equals 0. During testing, a conservative coding system with $\overline{\rho_a} = 0.7$ makes no prediction in response to inputs that are too novel, and thus initially has a lower proportion of correct responses. However, the number of incorrect responses is always low with $\overline{\rho_a} = 0.7$, even with very few training samples, and the 99% correct-response rate is achieved for both forced choice ($\overline{\rho_a} = 0$) and conservative ($\overline{\rho_a} = 0.7$) systems with training sets smaller than 1,000 exemplars.

Table 4 summarizes simulation results that repeat

TABLE 4
Off-line Conservative ($\overline{\rho_a} = 0.7$) ARTMAP System Performance After Training on Input Sets Ranging in Size From 3 to 4,000 Exemplars. Each Line Shows Average Correct, Incorrect and No-Response Test Set Predictions Over 10 Independent Simulations, Plus the Range of Learned ART$_a$ Category Numbers

Training Set Size	Average % Correct (Test Set)	Average % Incorrect (Test Set)	Average % No-Response (Test Set)	Number of ART$_a$ Categories
3	25.6	0.6	73.8	2–3
5	41.1	0.4	58.5	3–5
15	57.6	1.1	41.3	8–10
30	62.3	0.9	36.8	14–18
60	78.5	0.8	20.8	21–27
125	83.1	0.7	16.1	33–37
250	92.7	0.3	7.0	42–51
500	97.7	0.1	2.1	48–64
1000	99.4	0.04	0.5	53–66
2000	100.0	0.00	0.05	54–69
4000	100	0.00	0.02	61–73

the conditions of Table 3 except that $\overline{\rho_a} = 0.7$. Here, a test input that does not make a 70% match with any learned expectation makes an "I don't know" prediction. Compared with the $\overline{\rho_a} = 0$ case of Table 3, Table 4 shows that larger training sets are required to achieve a correct prediction rate of over 95%. However, because of the option to make no prediction, the average test set error rate is almost always less than 1%, even when the training set is very small, and is less than .1% after only 500 training trials. Moreover, 100% accuracy is achieved using only (approximately) 1/130 as many ART_a categories as there are inputs to classify.

3.6. Category Structure

Each ARTMAP category code can be described as a set of ART_a feature values on 1 to 22 observable features, chosen from 126 feature values, that are associated with the ART_b identification as poisonous or edible. During learning, the number of feature values that characterize a given category is monotone decreasing, so that generalization within a given category tends to increase. The total number of classes can, however, also increase, which tends to decrease generalization. Increasing the number of training patterns hereby tends to increase the number of categories and decrease the number of critical feature values of each established category. The balance between these opposing tendencies leads to the final net level of generalization.

Table 5 illustrates the long term memory structure underlying the 125-sample forced-choice simulation shown in Figure 4c. Of the 9 categories established at the end of the training phase, 4 are identified as poisonous (P) and 5 are identified as edible (E). Each ART_a category assigns a feature value to a subset of the 22 observable features. For example, Category 1 (poisonous) specifies values for 5 features, and leaves the remaining 17 features unspecified. The corresponding ART_a weight vector has 5 ones and 121 zeros. Note that the features that characterize Category 5 (poisonous) form a subset of the features that characterize Category 6 (edible). Recall that this category structure gave 96.4% correct responses on the 7,999 test set samples, which are partitioned as shown in the last line of Table 5. When 100% accuracy is achieved, a few categories with a small number of specified features typically code large clusters, while a few categories with many specified features code small clusters of rare samples.

Table 6 illustrates the statistical nature of the coding process, which leads to a variety of category structures when fast learning is used. Test set prediction accuracy of the simulation that generated Table 6 was similar to that of Table 5, and each simulation

TABLE 5

Critical Feature Values of the 9 Category Prototypes Learned in the 125-Sample Simulation Illustrated in Figure 4c ($\overline{\rho_a} = 0$). Categories 1, 5, 7 and 8 are Identified as Poisonous (P) and Categories 2, 3, 4, 6, and 9 are Identified as Edible (E). These Prototypes Yield 96.4% Accuracy on Test Set Inputs.

#	Feature	1 = P	2 = E	3 = E	4 = E	5 = P	6 = E	7 = P	8 = P	9 = E
1	Cap-Shape									
2	Cap-Surface									
3	Cap-Color									
4	Bruises?							Yes	No	Yes
5	Odor		None				None			
6	Gill-Attachment	Free	Free		Free	Free	Free	Free	Free	Free
7	Gill-Spacing	Close			Close	Close	Close	Close	Close	Close
8	Gill-Size		Broad						Narrow	Broad
9	Gill-Color								Buff	
10	Stalk-Shape								Tapering	Enlarged
11	Stalk-Root								Missing	Club
12	Stalk-Surface-Above-Ring			Smooth	Smooth	Smooth	Smooth	Smooth	Smooth	Smooth
13	Stalk-Surface-Below-Ring			Smooth						Smooth
14	Stalk-Color-Above-Ring					White	White	White	Pink	White
15	Stalk-Color-Below-Ring							White		White
16	Veil-Type	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial	Partial
17	Veil-Color	White	White		White	White	White	White	White	White
18	Ring-Number	One		One	One		One	One	One	One
19	Ring-Type			Pendant				Pendant	Evanescent	Pendant
20	Spore-Print-Color								White	
21	Population					Several	Several	Scattered	Several	Scattered
22	Habitat									
# Coded/Category:		2367	1257	387	1889	756	373	292	427	251

TABLE 6
Critical Feature Values of the 4 Prototypes Learned in a 125-Sample Simulation With a Training Set Different From the One in Table 5. Prediction Accuracy is Similar (96.0%), but the ART$_a$ Category Boundaries are Different

#	Feature	1 = E	2 = P	3 = P	4 = E
1	Cap-Shape				
2	Cap-Surface				
3	Cap-Color				
4	Bruises?			No	
5	Odor	None			
6	Gill-Attachment	Free	Free		
7	Gill-Spacing			Close	Close
8	Gill-Size	Broad			Broad
9	Gill-Color				
10	Stalk-Shape				Enlarging
11	Stalk-Root				
12	Stalk-Surface-Above-Ring				Smooth
13	Stalk-Surface-Below-Ring				
14	Stalk-Color-Above-Ring				
15	Stalk-Color-Below-Ring		White		
16	Veil-Type	Partial	Partial	Partial	Partial
17	Veil-Color	White	White	White	
18	Ring-Number		One		One
19	Ring-Type				Pendant
20	Spore-Print-Color				
21	Population				
22	Habitat				
	# Coded/Category:	3099	1820	2197	883

had a 125-sample training set. However, the simulation of Table 6 produced only 4 ART$_a$ categories, only one of which (Category 1) has the same long-term memory representation as Category 2 in Table 5. Note that, at this stage of coding, certain features are uninformative. For example, no values are specified for features 1, 2, 3, or 22 in Table 5 or Table 6; and feature 16 (veil-type) always has the value "partial." However, performance is still only around 96%. As rare instances form small categories later in the coding process, some of these features may become critical in identifying exemplars of small categories.

We will now turn to a description of the components of the ARTMAP system.

4. ART MODULES ART$_a$ and ART$_b$

Each ART module in Figures 1 and 2 establishes compressed recognition codes in response to sequences of input patterns **a** and **b**. Associative learning at the Map Field links pairs of pattern classes via these compressed codes. One type of generalization follows immediately from this learning strategy: If one vector **a** is associated with a vector **b**, then any other input that activates **a**'s category node will predict the category of pattern **b**. Any ART module can be used to self-organize the ART$_a$ and ART$_b$ categories. In the application above, **a** and **b** are binary vectors, so ART$_a$ and ART$_b$ can be ART 1 modules. The main computations of an ART 1 module will here be outlined. A full definition of ART 1 modules, as systems of differential equations, along with an analysis of their network dynamics, can be found in Carpenter and Grossberg (1987a).

In an ART 1 module, an input pattern **I** is represented in field F_1 and the recognition category for **I** is represented in field F_2. We consider the case where the competitive field F_2 makes a choice and where the system is operating in a fast-learn mode, as defined below. An algorithm for simulations is given in the Appendix.

4.1. F_1 Activation

Figure 5 illustrates the main components of an ART 1 module. A field, F_1 of M nodes, with output vector $\mathbf{x} \equiv (x_1, \ldots, x_M)$, registers the $F_0 \rightarrow F_1$ input vector $\mathbf{I} \equiv (I_1, \ldots, I_M)$. Each F_1 node can receive input from 3 sources: the $F_0 \rightarrow F_1$ bottom-up input; nonspecific gain control signals; and top-down signals from the N nodes of F_2, via an $F_2 \rightarrow F_1$ adaptive filter. A node is said to be *active* if it generates an output signal equal to 1. Output from inactive nodes equals 0. In ART 1 an F_1 node is active if at least 2 of the 3 input signals are large. This rule for F_1 activation is called the *2/3 Rule*. The 2/3 Rule is realized in its simplest, dimensionless form as follows:

2/3 Rule matching

The ith F_1 node is active if its net input exceeds a fixed threshold. Specifically,

$$x_i = \begin{cases} 1 & \text{if } I_i + g_1 + \Sigma_{j=1}^{N} y_j z_{ji} > 1 + \bar{z} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where term I_i is the binary $F_0 \rightarrow F_1$ input, term g_1 is the binary nonspecific F_1 gain control signal, term $\Sigma\, y_j z_{ji}$ is the sum of $F_2 \rightarrow F_1$ signals y_j via pathways with adaptive weights z_{ji}, and $\bar{z}$ is a constant such that

$$0 < \bar{z} < 1. \tag{2}$$

***F*₁ gain control**

The F_1 gain control signal g_1 is defined by

$$g_1 = \begin{cases} 1 & \text{if } F_0 \text{ is active and } F_2 \text{ is inactive} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Note that F_2 activity inhibits F_1 gain, as shown in Figure 5. These laws for F_1 activation imply that, if F_2 is inactive,

$$x_i = \begin{cases} 1 & \text{if } I_i = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

If exactly one F_2 node J is active, the sum $\Sigma\, y_j z_{ji}$ in eqn (1) reduces to the single term z_{Ji}, so

$$x_i = \begin{cases} 1 & \text{if } I_i = 1 \text{ and } z_{Ji} > \bar{z} \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

4.2. F_2 Choice

Let T_j denote the total input from F_1 to the jth F_2 node, given by

$$T_j = \sum_{i=1}^{M} x_i Z_{ij}, \tag{6}$$

where the Z_{ij} denote the $F_1 \rightarrow F_2$ adaptive weights. If some $T_j > 0$, define the F_2 choice index J by

$$T_J = \max\{T_j : j = 1 \ldots N\}. \tag{7}$$

In the typical case, J is uniquely defined. Then the F_2 output vector $\mathbf{y} = (y_1, \ldots, y_N)$ obeys

$$y_j = \begin{cases} 1 & \text{if } j = J \\ 0 & \text{if } j \neq J. \end{cases} \tag{8}$$

If two or more indices j share maximal input, then they equally share the total activity. This case is not considered here.

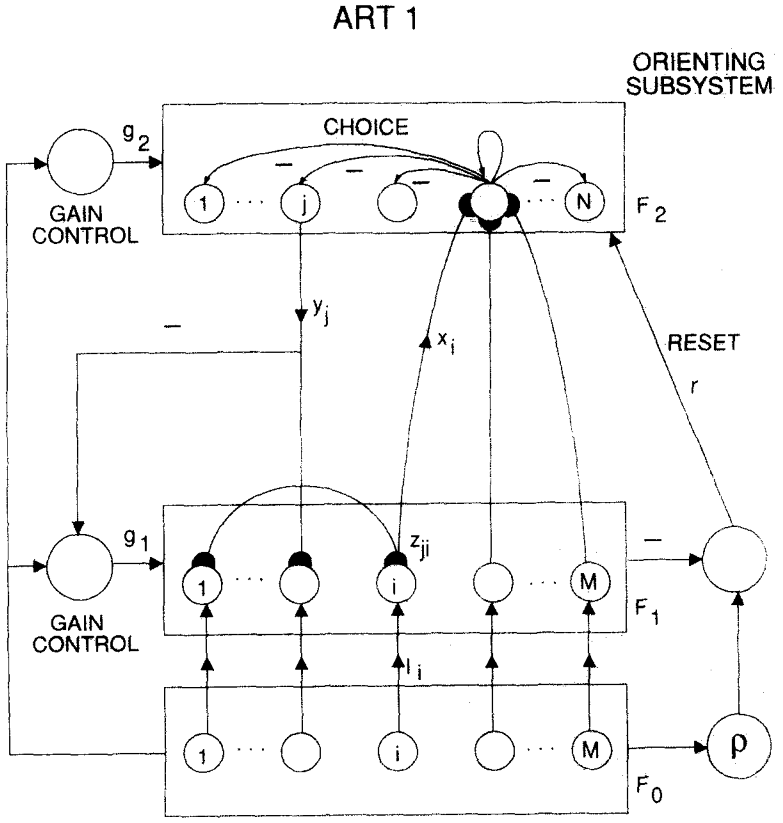


FIGURE 5. ART 1 schematic diagram (Carpenter & Grossberg, 1987a). The binary vector I forms the bottom-up input to the field F_1 whose activity vector is denoted x. The competitive field F_2 is designed to make a choice. Adaptive pathways lead from each F_1 node to all F_2 nodes, and from each F_2 node to all F_1 nodes. Reset occurs when the match between x and I fails to meet the criterion established by the vigilance parameter ρ. All paths are excitatory unless marked with a minus sign.

4.3. Learning Laws

In fast-learn ART 1, adaptive weights reach their new asymptote on each input presentation. The learning laws, as well as the rules for choice and search, are conveniently described using the following notation. If **a** is a binary M-vector, define the norm of **a** by

$$|\mathbf{a}| \equiv \sum_{i=1}^{M} a_i. \tag{9}$$

If **a** and **b** are two binary vectors, define a third binary vector **a** ∩ **b** by

$$(\mathbf{a} \cap \mathbf{b})_i = 1 \Leftrightarrow a_i = 1 \text{ and } b_i = 1. \tag{10}$$

Finally, let **a** be a *subset* of **b** ($\mathbf{a} \subseteq \mathbf{b}$) iff $\mathbf{a} \cap \mathbf{b} = \mathbf{a}$.

All ART 1 learning is gated by F_2 activity; that is, the adaptive weights z_{Ji} and Z_{iJ} can change only when the Jth F_2 node is active. Then both $F_2 \rightarrow F_1$ and $F_1 \rightarrow F_2$ weights are functions of the F_1 vector **x**, as follows:

Top-down learning

Top-down $F_2 \rightarrow F_1$ weights in active paths learn **x**; that is, when the Jth F_2 node is active

$$z_{Ji} \rightarrow x_i. \tag{11}$$

All other z_{ji} remain unchanged. Stated as a differential equation, this learning rule is

$$\frac{d}{dt} z_{ji} = y_j(x_i - z_{ji}). \tag{12}$$

In eqn (12), learning by z_{ji} is *gated* by y_j. When the y_j gate opens—that is, when $y_j > 0$—then learning begins and z_{ji} is attracted to x_i. In vector terms, if $y_j > 0$, then $\mathbf{z}_j \equiv (z_{j1}, z_{j2}, \ldots, z_{jM})$ approaches **x**. Such a law is therefore sometimes called learning by *gated steepest descent*. It is also called the *outstar learning* rule, and was introduced into the neural modelling literature in 1969 (Grossberg, 1969).

Initially all z_{ji} are maximal:

$$z_{ji}(0) = 1. \tag{13}$$

Thus with fast learning, the top-down weight vector $\mathbf{z}_J$ is a binary vector at the start and end of each input presentation. By eqns (4), (5), (10), (11), and (13), the F_1 activity vector can be described as

$$\mathbf{x} = \begin{cases} \mathbf{I} & \text{if } F_2 \text{ is inactive} \\ \mathbf{I} \cap \mathbf{z}_J & \text{if the } J\text{th } F_2 \text{ node is active.} \end{cases} \tag{14}$$

By eqns (5) and (12), when node J is active, learning causes

$$\mathbf{z}_J \rightarrow \mathbf{I} \cap \mathbf{z}_J^{(\text{old})}, \tag{15}$$

where $\mathbf{z}_J^{(\text{old})}$ denotes $\mathbf{z}_J$ at the start of the input presentation. By eqns (11) and (14), **x** remains constant during learning, even though $|\mathbf{z}_J|$ may decrease.

The first time an F_2 node J becomes active, it is said to be *uncommitted*. Then, by eqns (13)–(15),

$$\mathbf{z}_J \rightarrow \mathbf{I} \tag{16}$$

during learning. Thereafter node J is said to be *committed*.

Bottom-up learning

In simulations it is convenient to assign initial values to the bottom-up $F_1 \rightarrow F_2$ adaptive weights Z_{ij} in such a way that F_2 nodes first become active in the order $j = 1, 2, \ldots$. This can be accomplished by letting

$$Z_{ij}(0) = \alpha_j \tag{17}$$

where

$$\alpha_1 > \alpha_2 > \ldots > \alpha_N. \tag{18}$$

Like the top-down weight vector $\mathbf{z}_J$, the bottom-up $F_1 \rightarrow F_2$ weight vector $\mathbf{Z}_J \equiv (Z_{1J} \ldots Z_{iJ} \ldots Z_{MJ})$ also becomes proportional to the F_1 output vector **x** when the F_2 node J is active. In addition, however, the bottom-up weights are scaled inversely to $|\mathbf{x}|$, so that

$$Z_{iJ} \rightarrow \frac{x_i}{\beta + |\mathbf{x}|}, \tag{19}$$

where $\beta > 0$. This $F_1 \rightarrow F_2$ learning law, called the Weber Law Rule (Carpenter & Grossberg, 1987a), realizes a type of competition among the weights $\mathbf{z}_J$ adjacent to a given F_2 node J. This competitive computation could alternatively be transferred to the F_1 field, as it is in ART 2 (Carpenter & Grossberg, 1987b). By eqns (14), (15), and (19), during learning

$$\mathbf{Z}_J \rightarrow \frac{\mathbf{I} \cap \mathbf{z}_J^{(\text{old})}}{\beta + |\mathbf{I} \cap \mathbf{z}_J^{(\text{old})}|}. \tag{20}$$

The Z_{ij} initial values are required to be small enough so that an input **I** that perfectly matches a previously learned vector $\mathbf{Z}_J$ will select the F_2 node J rather than an uncommitted node. This is accomplished by assuming that

$$0 < \alpha_j = Z_{ij}(0) < \frac{1}{\beta + |\mathbf{I}|} \tag{21}$$

for all $F_0 \rightarrow F_1$ inputs **I**. When **I** is first presented, $\mathbf{x} = \mathbf{I}$, so by eqns (6), (15), (17), and (20), the $F_1 \rightarrow F_2$ input vector $\mathbf{T} \equiv (T_1, T_2, \ldots, T_N)$ is given by

$$T_j = \sum_{i=1}^{M} I_i Z_{ij} = \begin{cases} |\mathbf{I}|\alpha_j & \text{if } j \text{ is an uncommitted node} \\ |\mathbf{I} \cap \mathbf{z}_j|/(\beta + |\mathbf{z}_j|) & \text{if } j \text{ is a committed node.} \end{cases} \tag{22}$$

In the simulations above, β is taken to be so small that, among committed nodes, T_j is determined by the size of $|\mathbf{I} \cap \mathbf{z}_j|$ relative to $|\mathbf{z}_j|$. If β were large, T_j would depend primarily on $|\mathbf{I} \cap \mathbf{z}_j|$. In addition, α_j

values are taken to be so small that an uncommitted node will generate the maximum T_j value in eqn (22) only if $|\mathbf{I} \cap \mathbf{z}_j| = 0$ for all committed nodes. Larger values of α_j and β bias the system toward earlier selection of uncommitted nodes when only poor matches are to be found among the committed nodes. A more complete discussion of this aspect of ART 1 system design is given by Carpenter and Grossberg (1987a).

4.4. Hypothesis Testing, Confidence, Novelty, and Search

By eqns (7), (21), and (22), a committed F_2 node J may be chosen even if the match between $\mathbf{I}$ and $\mathbf{z}_J$ is poor; the match need only be the best one available. If the match is too poor, then the ART 1 system can autonomously carry out hypothesis testing, or search, for a better F_2 recognition code. This search process is mediated by the orienting subsystem, which can reset F_2 nodes in response to poor matches at F_1 (Figure 5). The orienting subsystem is a type of novelty detector that measures system confidence. If the degree of match between bottom-up input $\mathbf{I}$ and top-down weight vector $\mathbf{z}_J$ is too poor, the system's confidence in the recognition code labelled by J is inadequate. Otherwise expressed, the input $\mathbf{I}$ is too unexpected relative to the top-down vector $\mathbf{z}_J$, which plays the role of a learned top-down expectation.

An unexpected input triggers a novelty burst at the orienting subsystem, which sends a nonspecific reset wave r from the orienting subsystem to F_2. The reset wave enduringly shuts off node J so long as input $\mathbf{I}$ remains on. With J off and its top-down $F_2 \rightarrow F_1$ signals silent, F_1 can again instate vector $\mathbf{x} = \mathbf{I}$, which leads to selection of another F_2 node through the bottom-up $F_1 \rightarrow F_2$ adaptive filter. This hypothesis testing process leads to activation of a sequence of F_2 nodes until one is chosen whose vector of adaptive weights forms an adequate match with $\mathbf{I}$, or until an uncommitted node is selected. The search takes place so rapidly that essentially no learning occurs on that time scale. Learned weights are hereby buffered against recoding by poorly matched inputs that activate unacceptable F_2 recognition codes. Thus, during search, previously learned weights actively control the search for a better recognition code without being changed by the signals that they process.

4.5. Vigilant Search and Resonant Learning

As noted above, the degree of match between bottom-up input $\mathbf{I}$ and top-down expectation $\mathbf{z}_J$ is evaluated at the orienting subsystem, which measures system confidence that category J adequately represents input $\mathbf{I}$. A reset wave is triggered only if this confidence measure falls below a dimensionless parameter ρ that is called the *vigilance parameter*. The vigilance parameter calibrates the system's sensitivity to disconfirmed expectations.

One of the main reasons for the successful classification of nonstationary data sequences by ARTMAP is its ability to recalibrate the vigilance parameter based on predictive success. How this works will be described below. For now, we characterize the ART 1 search process given a constant level of vigilance.

In fast-learn ART 1 with choice at F_2, the search process occurs as follows:

Step 1—Select one F_2 node J that maximizes T_j in eqn (22), and read-out its top-down weight vector $\mathbf{z}_J$.

Step 2—With J active, compare the F_1 output vector $\mathbf{x} = \mathbf{I} \cap \mathbf{z}_J$ with the $F_0 \rightarrow F_1$ input vector $\mathbf{I}$ at the orienting subsystem (Figure 5).

Step 3A—Suppose that $\mathbf{I} \cap \mathbf{z}_J$ fails to match $\mathbf{I}$ at the level required by the vigilance criterion, i.e., that

$$|\mathbf{x}| = |\mathbf{I} \cap \mathbf{z}_J| < \rho|\mathbf{I}|. \tag{23}$$

Then F_2 reset occurs: node J is shut off for the duration of the input interval during which $\mathbf{I}$ remains on. The index of the chosen F_2 node is reset to the value corresponding to the next highest $F_1 \rightarrow F_2$ input T_j. With the new node active, Steps 2 and 3A are repeated until the chosen node satisfies the resonance criterion in Step 3B. Note that reset never occurs if

$$\rho \leq 0. \tag{24}$$

When eqn (24) holds, an ART system acts as if there were no orienting subsystem.

Step 3B—Suppose that $\mathbf{I} \cap \mathbf{z}_J$ meets the criterion for resonance; i.e., that

$$|\mathbf{x}| = |\mathbf{I} \cap \mathbf{z}_J| \geq \rho|\mathbf{I}|. \tag{25}$$

Then the search ceases and the last chosen F_2 node J remains active until input $\mathbf{I}$ shuts off (or until ρ increases). In this state, called *resonance*, both the $F_1 \rightarrow F_2$ and the $F_2 \rightarrow F_1$ adaptive weights approach new values if $\mathbf{I} \cap \mathbf{z}_J^{(\text{old})} \neq \mathbf{z}_J^{(\text{old})}$. Note that resonance cannot occur if $\rho > 1$.

If $\rho \leq 1$, search ceases whenever $\mathbf{I} \subseteq \mathbf{z}_J$, as is the case if an uncommitted node J is chosen. If vigilance is close to 1, then reset occurs if $F_2 \rightarrow F_1$ input alters the F_1 activity pattern at all; resonance requires that $\mathbf{I}$ be a subset of $\mathbf{z}_J$. If vigilance is near 0, reset never occurs. The top-down expectation $\mathbf{z}_J$ of the first chosen F_2 node J is then recoded from $\mathbf{z}_J^{(\text{old})}$ to $\mathbf{I} \cap \mathbf{z}_J^{(\text{old})}$, even if $\mathbf{I}$ and $\mathbf{z}_J^{(\text{old})}$ are very different vectors.

4.6. F_2 Gain Control

For simplicity, ART 1 is exposed to discrete presentation intervals during which an input is constant and after which F_1 and F_2 activities are set to zero. Discrete presentation intervals are implemented in ART 1 by means of the F_1 and F_2 gain control signals g_1 and g_2 (Figure 5). The F_2 gain signal g_2 is assumed, like g_1 in eqn (3), to be 0 if F_0 is inactive. Then, when F_0 becomes active, g_2 and F_2 signal thresholds are assumed to lie in a range where the F_2 node that receives the largest input signal can become active. When an ART 1 system is embedded in a hierarchy, F_2 may receive signals from sources other than F_1. This occurs in the ARTMAP system described below. In such a system, F_2 still makes a choice and gain signals from F_0 are still required to generate both F_1 and F_2 output signals. In the simulations, F_2 nodes that are reset during search remain off until the input shuts off. A real-time ART search mechanism that can cope with continuously fluctuating analog or binary inputs of variable duration, fast or slow learning, and compressed or distributed F_2 codes is described by Carpenter and Grossberg (1990).

5. THE MAP FIELD

A Map Field module links the F_2 fields of the ART$_a$ and ART$_b$ modules. Figure 6 illustrates the main components of the Map Field. We will describe one such system in the fast-learn mode with choice at the fields F_2^a and F_2^b. As with the ART 1 and ART 2 architectures themselves (Carpenter & Grossberg, 1987a, 1987b), many variations of the network architecture lead to similar computations. In the ARTMAP hierarchy, ART$_a$, ART$_b$, and Map Field modules are all described in terms of ART 1 variables and parameters. Indices a and b identify terms in the ART$_a$ and ART$_b$ modules, while Map Field variables and parameters have no such index. Thus, for example, ρ_a, ρ_b, and ρ denote the ART$_a$, ART$_b$, and Map Field vigilance parameters, respectively.

5.1. ART$_a$, ART$_b$, and Complement Coding

Both ART$_a$ and ART$_b$ are fast-learn ART 1 modules. With one optional addition, they duplicate the design described above. That addition, called *complement coding* (Carpenter, Grossberg, & Rosen, 1991), rep-

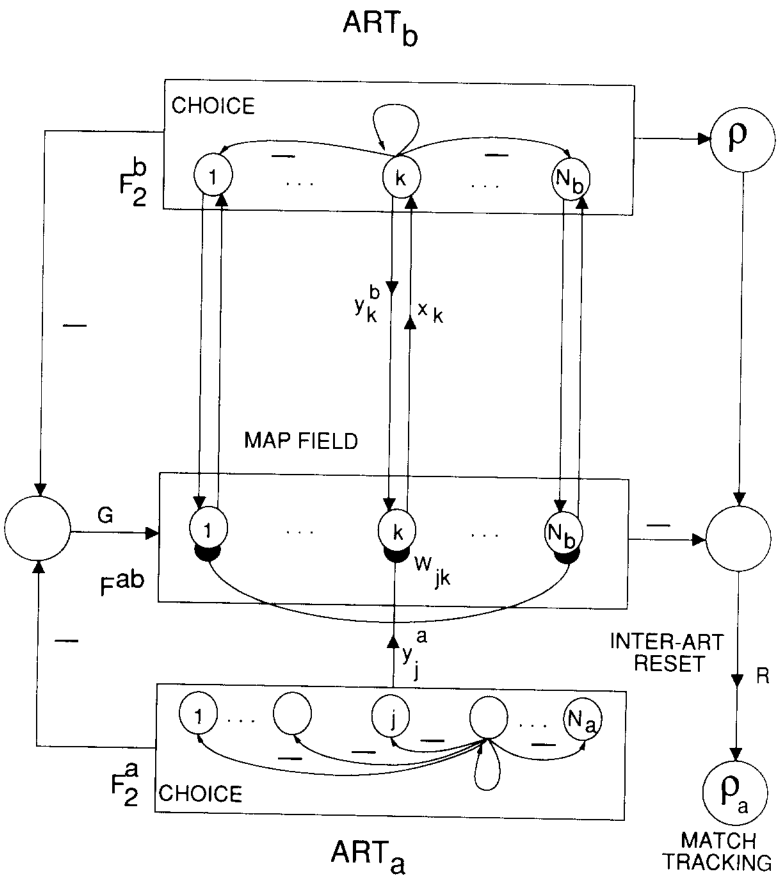


FIGURE 6. The Map Field is connected to F_2^b with one-to-one, nonadaptive pathways in both directions. Each F_2^a node is connected to all Map Field nodes via adaptive pathways. A mismatch between the category predicted by a and the actual category of b activates the Map Field orienting subsystem. This leads to F_2^a reset and increased vigilance (ρ_a) via match tracking.

resents both the on-response to an input vector and the off-response to that vector. This ART coding strategy has been shown to play a useful role in searching for appropriate recognition codes in response to predictive feedback (Grossberg, 1982b, 1984). To represent such a code in its simplest form, let the input vector **a** itself represent the on-response, and the complement of **a**, denoted by $\mathbf{a}^c$, represent the off-response, for each ART$_a$ input vector **a**. If **a** is the binary vector $(a_1, \ldots, a_{M_a})$, the input to ART$_a$ is the 2M$_a$-dimensional binary vector

$$(\mathbf{a}, \mathbf{a}^c) \equiv (a_1, \ldots, a_{M_a}, a_1^c, \ldots, a_{M_a}^c) \tag{26}$$

where

$$a_i^c = 1 - a_i. \tag{27}$$

The utility of complement coding for searching an ARTMAP system will be described below. Conditions will also be given where complement coding is not needed. In fact, complement coding was not needed for any of the simulations described above, and the ART$_a$ input was simply the vector **a**.

In the discussion of the Map Field module below, F_2^a nodes, indexed by $j = 1 \ldots N_a$, have binary output signals y_j^a; and F_2^b nodes, indexed by $k = 1 \ldots N_b$, have binary output signals y_k^b. Correspondingly, the index of the active F_2^a node is denoted by J, and the index of the active F_2^b node is denoted by K. Because the Map Field is the interface where signals from F_2^a and F_2^b interact, it is denoted by F^{ab}. The nodes of F^{ab} have the same index k, $k = 1, 2, \ldots, N_b$, as the nodes of F_2^b because there is a one-to-one correspondence between these sets of nodes. The output signals of F^{ab} nodes are denoted by x_k.

5.2. 2/3 Rule Map Field Matching

Each node of F^{ab} can receive input from three sources: F_2^a, F_2^b, and a Map Field gain control G. The F^{ab} output vector **x** obeys the 2/3 Rule of ART 1; namely,

$$x_k = \begin{cases} 1 & \text{if } y_k^b + G + \sum_{j=1}^{N_a} y_j^a w_{jk} > 1 + \overline{w} \\ 0 & \text{otherwise} \end{cases} \tag{28}$$

where term y_k^b is the F_2^b output signal, term G is a binary gain control signal, term $\Sigma\, y_j^a w_{jk}$ is the sum of $F_2^a \rightarrow F^{ab}$ signals y_j^a via pathways with adaptive weights w_{jk}, and $\overline{w}$ is a constant such that

$$0 < \overline{w} < 1. \tag{29}$$

Values of the gain control signal G and the $F_2^a \rightarrow F^{ab}$ weight vectors $\mathbf{w}_j \equiv (\mathbf{w}_{j1}, \ldots, w_{jNb})$, $j = 1 \ldots N_a$, are specified below.

5.3. F^{ab} Gain Control

Comparison of eqns (1) and (28) indicates an analogy between fields F_2^b, F^{ab}, and F_2^a in a Map Field module and fields F_0, F_1, and F_2, respectively, in an ART 1 module. Differences between these modules include the bidirectional nonadaptive connections between F_2^b and F^{ab} in the Map Field module (Figure 6) compared to the bidirectional adaptive connections between fields F_1 and F_2 in the ART 1 module (Figure 5). These different connectivity schemes require different rules for the gain control signals G and g_1.

The Map Field gain control signal G obeys the equation

$$G = \begin{cases} 0 & \text{if } F_2^a \text{ and } F_2^b \text{ are both active} \\ 1 & \text{otherwise.} \end{cases} \tag{30}$$

Note that G is a persistently active, or tonic, signal that is turned off only when both ART$_a$ and ART$_b$ are active.

5.4. $F_2^a \rightarrow F^{ab}$ Initial Values

If an active F_2^a node J has not yet learned a prediction, the ARTMAP system is designed so that J can learn to predict any ART$_b$ pattern if one is active or becomes active while J is active. This design constraint is satisfied using the assumption, analogous to eqn (13), that

$$w_{jk}(0) = 1 \tag{31}$$

for $j = 1 \ldots N_a$ and $k = 1 \ldots N_b$.

5.5. Map Field Activation

Rules governing G and $\mathbf{w}_j(0)$ enable the following Map Field properties to obtain. If both ART$_a$ and ART$_b$ are active, then learning of ART$_a \rightarrow$ ART$_b$ associations can take place at F^{ab}. If ART$_a$ is active but ART$_b$ is not, then any previously learned ART$_a \rightarrow$ ART$_b$ prediction is read out at F^{ab}. If ART$_b$ is active but ART$_a$ is not, then the selected ART$_b$ category is represented at F^{ab}. If neither ART$_a$ nor ART$_b$ is active, then F^{ab} is not active. By eqns (28)–(31), the 2/3 Rule realizes these properties in the following four cases.

F_2^a active and F_2^b active

If both the F_2^a category node J and the F_2^b category node K are active, then $G = 0$ by eqn (30). Thus by eqn (28),

$$x_k = \begin{cases} 1 & \text{if } k = K \text{ and } w_{JK} > \overline{w} \\ 0 & \text{otherwise.} \end{cases} \tag{32}$$

All $x_k = 0$ for $k \neq K$. Moreover, $x_K = 1$ only if an association has previously been learned in the pathway from node J to node K, or if J has not yet learned to predict any ART$_b$ category. If J predicts any category other than K, then all $x_k = 0$.

***F_2^a* active and *F_2^b* inactive**

If the F_2^a node J is active and F_2^b is inactive, then $G = 1$. Thus

$$x_k = \begin{cases} 1 & \text{if } w_{Jk} > \overline{w} \\ 0 & \text{otherwise.} \end{cases} \quad (33)$$

By eqns (31) and (33), if an input **a** has activated node J in F_2^a but F_2^b is not yet active, J activates all nodes k in F^{ab} if J has learned no predictions. If prior learning has occurred, all nodes k are activated whose adaptive weights w_{Jk} are still large.

***F_2^b* active and *F_2^a* inactive**

If the F_2^b node K is active and F_2^a is inactive, then $G = 1$. Thus

$$x_k = \begin{cases} 1 & \text{if } k = K \\ 0 & \text{otherwise.} \end{cases} \quad (34)$$

In this case, the F^{ab} output vector $\mathbf{x}$ is the same as the F_2^b output vector $\mathbf{y}^b$.

***F_2^a* inactive and *F_2^b* inactive**

If neither F_2^a nor F_2^b is active, the total input to each F^{ab} node is $G = 1$, so all $x_k = 0$ by eqn (28).

5.6. F_2^b Choice and Priming

If ART$_b$ receives an input **b** while ART$_a$ has no input, then F_2^b chooses the node K with the largest $F_1^b \rightarrow F_2^b$ input. Field F_2^b then activates the Kth F^{ab} node, and $F^{ab} \rightarrow F_2^b$ feedback signals support the original $F_1^b \rightarrow F_2^b$ choice. If ART$_a$ receives an input **a** while ART$_b$ has no input, F_2^a chooses a node J. If, due to prior learning, some $w_{JK} = 1$ while all other $w_{Jk} = 0$, we say that **a** *predicts* the ART$_b$ category K, as F^{ab} sends its signal vector $\mathbf{x}$ to F_2^b. Field F_2^b is hereby *attentionally primed*, or sensitized, but the field remains inactive so long as ART$_b$ has no input from F_0^b. If then an $F_0^b \rightarrow F_1^b$ input **b** arrives, the F_2^b choice depends upon network parameters and timing. It is natural to assume, however, that **b** simultaneously activates the F_1^b and F_2^b gain control signals g_1^b and g_2^b (Figure 5). Then F_2^b processes the F^{ab} prime $\mathbf{x}$ as soon as F_1^b processes the input **b**, and F_2^b chooses the primed node K. Field F_1^b then receives $F_2^b \rightarrow F_1^b$ expectation input $\mathbf{z}_K^b$ as well as $F_0^b \rightarrow F_1^b$ input **b**, leading either to match or reset.

5.7. $F_2^a \rightarrow F^{ab}$ Learning Laws

The $F_2^a \rightarrow F^{ab}$ adaptive weights w_{jk} obey an outstar learning law similar to that governing the $F_2 \rightarrow F_1$ weights z_{ji} in (12); namely,

$$\frac{d}{dt} w_{jk} = y_j^a(x_k - w_{jk}). \quad (35)$$

According to (35), the $F_2^a \rightarrow F^{ab}$ weight vector $\mathbf{w}_J$ approaches the F^{ab} activity vector $\mathbf{x}$ if the Jth F_2^a node is active. Otherwise $\mathbf{w}_J$ remains constant. If node J has not yet learned to make a prediction, all weights w_{Jk} equal 1, by eqn (31). In this case, if ART$_b$ receives no input **b**, then all $\mathbf{x}_k$ values equal 1 by eqn (33). Thus, by eqn (35), all w_{jk} values remain equal to 1. As a result, category choices in F_2^a do not alter the adaptive weights w_{jk} until these choices are associated with category choices in F_2^b.

5.8. Map Field Reset and Match Tracking

The Map Field provides the control that allows the ARTMAP system to establish different categories for very similar ART$_a$ inputs that make different predictions, while also allowing very different ART$_a$ inputs to form categories that make the same prediction. In particular, the Map Field orienting subsystem becomes active only when ART$_a$ makes a prediction that is incompatible with the actual ART$_b$ input. This mismatch event activates the control strategy, called *match tracking*, that modulates the ART$_a$ vigilance parameter ρ_a in such a way as to keep the system from making repeated errors. As illustrated in Figure 6, a mismatch at F^{ab} while F_2^b is active triggers an inter-ART reset signal R to the ART$_a$ orienting subsystem. This occurs whenever

$$|\mathbf{x}| < \rho|\mathbf{y}^b|, \quad (36)$$

where ρ denotes the Map Field vigilance parameter. The entire cycle of ρ_a adjustment proceeds as follows through time. At the start of each input presentation, ρ_a equals a fixed baseline vigilance $\overline{\rho_a}$. When an input **a** activates an F_2^a category node J and resonance is established,

$$|\mathbf{x}^a| = |\mathbf{a} \cap \mathbf{z}_J^a| \geq \rho_a|\mathbf{a}|, \quad (37)$$

as in eqn (25). An inter-ART reset signal is sent to ART$_a$ if the ART$_b$ category predicted by **a** fails to match the active ART$_b$ category, by eqn (36). The inter-ART reset signal R raises ρ_a to a value that is just high enough to cause eqn (37) to fail, so that

$$\rho_a > \frac{|\mathbf{a} \cap \mathbf{z}_J^a|}{|\mathbf{a}|}. \quad (38)$$

Node J is therefore reset and an ART$_a$ search ensues. Match tracking continues until an active ART$_a$ category satisfies both the ART$_a$ matching criterion eqn (37) and the analogous Map Field matching criterion. Match tracking increases the ART$_a$ vigilance by the minimum amount needed to abort an incorrect ART$_a \rightarrow$ ART$_b$ prediction and to drive a search for a new ART$_a$ category that can establish a correct prediction. As shown by example below, match tracking allows **a** to make a correct prediction on subsequent trials, without repeating the initial sequence of errors. Match tracking hereby conjointly maximizes predictive generalization and minimizes predictive error on a trial-by-trial basis, using only local computations.

5.9. Match Tracking Using VITE Dynamics

The operation of match tracking can be implemented in several different ways. One way is to use a variation on the Vector Integration to Endpoint, or VITE, circuit (Bullock & Grossberg, 1988) as follows. Let an ART$_a$ binary *reset signal* r_a (Figure 7) obey the equation

$$r_a = \begin{cases} 1 & \text{if } \rho_a|\mathbf{a}| - |\mathbf{x}^a| > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{39}$$

as in eqn (23). The complementary ART$_a$ *resonance signal* $r_a^c = 1 - r_a$. Signal R equals 1 during inter-ART reset; that is, when inequality (36) holds. The size of the ART$_a$ vigilance parameter ρ_a is determined by the *match tracking equation*

$$\frac{dt}{dt}\rho_a = (\overline{\rho_a} - \rho_a) + \gamma R r_a^c, \tag{40}$$

where $\gamma \gg 1$. During inter-ART reset, $R = r_a^c = 1$, causing ρ_a to increase until $r_a^c = 0$. Then $\rho_a|\mathbf{a}| > |\mathbf{x}^a|$, as required for match tracking (38). When $r_a^c = 0$, ρ_a relaxes to $\overline{\rho_a}$. This is assumed to occur at a rate slower than node activation, also called short-term memory (STM), and faster than learning, also called long-term memory (LTM). Such an intermediate rate is called medium-term memory (MTM) (Carpenter & Grossberg, 1990).

Comparing the match tracking circuit in Figure 7 to a VITE circuit, the inter-ART reset signal R is analogous to the VITE GO signal; total F_1^a output $|\mathbf{x}^a|$ is analogous to the Target Position Code (TPC); total F_0^a output, gated by ρ_a, is analogous to the Present Position Command (PPC); and the quantity $(\rho_a|\mathbf{a}| - |\mathbf{x}^a|)$ in (39) is analogous to the Difference Vector (DV). (See Bullock & Grossberg, 1988, Figure 17.)

An ART$_a$ search that is triggered by increasing ρ_a according to eqn (40) ceases if some active F_2^a node J satisfies

$$|\mathbf{a} \cap \mathbf{z}_J^a| \geq \rho_a|\mathbf{a}|. \tag{41}$$

If no such node exists, F_2^a shuts down for the rest of the input presentation. In particular, if $\mathbf{a} \subseteq \mathbf{z}_J^a$, match tracking makes $\rho_a > 1$, so $\mathbf{a}$ cannot activate another category in order to learn the new prediction. The following anomalous case can thus arise. Suppose that $\mathbf{a} = \mathbf{z}_J^a$ but the ART$_b$ input $\mathbf{b}$ mismatches the ART$_b$ expectation $\mathbf{z}_K^b$ previously associated with J. Then match tracking will prevent the recoding that would have associated $\mathbf{a}$ with $\mathbf{b}$. That is, the ARTMAP system with fast learning and choice will not learn the prediction of an exemplar that *exactly* matches a learned prototype when the new predic-

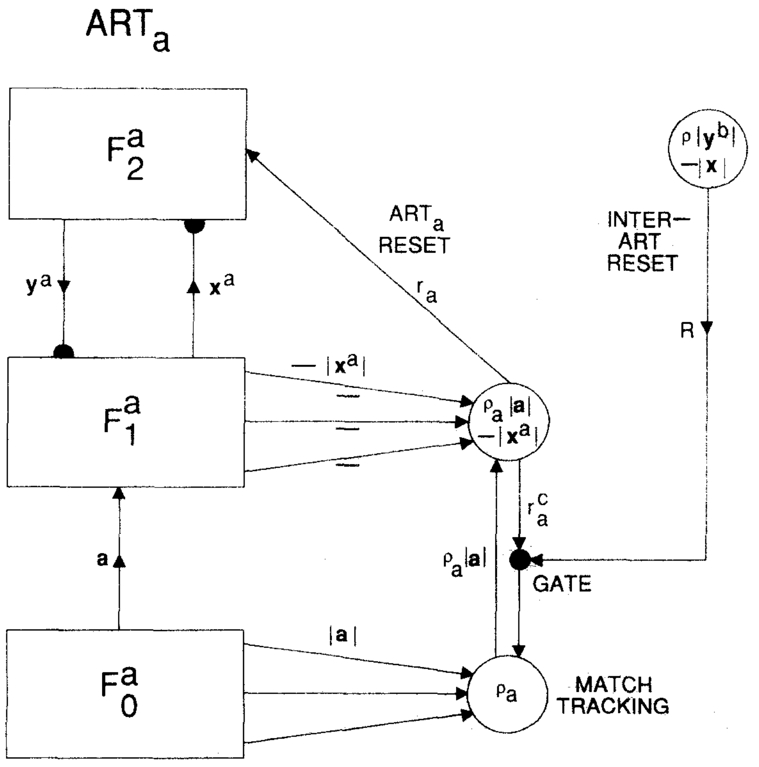


FIGURE 7. Match tracking by a scalar VITE circuit. When $r_a^c = R = 1$, ρ_a rapidly increases until $\rho_a|\mathbf{a}| > |\mathbf{x}^a|$. Once this occurs, $r_a^c = 0$ and $r_a = 1$, causing ART$_a$ reset. The inter-ART reset signal R plays a role analogous to the VITE model GO signal.

tion contradicts the previous predictions of the exemplars that created the prototype. This situation does not arise when all ART$_a$ inputs **a** have the same number of 1's, as follows.

5.10. Equal-Norm Inputs and Search

Consider the case in which all ART$_a$ inputs have the same norm:

$$|\mathbf{a}| \equiv \text{constant}. \tag{42}$$

When an ART$_a$ category node J becomes committed to input **a**, then $|\mathbf{z}_J^a| = |\mathbf{a}|$. Thereafter, by the 2/3 Rule (15), $\mathbf{z}_J^a$ can be recoded only by decreasing its number of 1 entries, and thus its norm. Once this occurs, no input **a** can ever be a subset of $\mathbf{z}_J^a$, by eqn (42). In particular, the situation described in the previous section cannot arise.

In the simulations reported in this article, all ART$_a$ inputs have norm 22. Equation (42) can also be satisfied by using complement coding, since $|(\mathbf{a}, \mathbf{a}^c)| = M_a$. Preprocessing ART$_a$ inputs by complement coding thus ensures that the system will avoid the case where some input **a** is a proper subset of the active ART$_a$ prototype $\mathbf{z}_J^a$ and the learned prediction of category J mismatches the correct ART$_b$ pattern.

Finally, note that with ARTMAP fast learning and choice, an ART$_a$ category node J is permanently committed to the first ART$_b$ category node K to which it is associated. However, the set of input exemplars that access either category may change through time, as in the banana example described in the introduction.

5.11. Match Tracking Example

The role of match tracking is illustrated by the following example. The input pairs shown in Table 7 are presented in order $(\mathbf{a}^{(1)}, \mathbf{b}^{(1)})$, $(\mathbf{a}^{(2)}, \mathbf{b}^{(2)})$, $(\mathbf{a}^{(3)}, \mathbf{b}^{(3)})$. The problem solved by match tracking is created by vector $\mathbf{a}^{(2)}$ lying "between" $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(3)}$, with $\mathbf{a}^{(1)} \subset \mathbf{a}^{(2)} \subset \mathbf{a}^{(3)}$, while $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(3)}$ are mapped to the same ART$_b$ vector. Suppose that, instead of match tracking, the Map Field orienting subsystem merely activated the ART$_a$ reset system. Coding would then proceed as follows.

TABLE 7
Nested ART$_a$ Inputs and Their Associated ART$_b$ Inputs

ART$_a$ inputs	ART$_b$ inputs
$\mathbf{a}^{(1)}$ (111000)	$\mathbf{b}^{(1)}$ (1010)
$\mathbf{a}^{(2)}$ (111100)	$\mathbf{b}^{(2)}$ (0101)
$\mathbf{a}^{(3)}$ (111110)	$\mathbf{b}^{(3)}$ (1010)

Choose $\overline{\rho_a} \le 0.6$ and $\rho_b > 0$. Vectors $\mathbf{a}^{(1)}$ then $\mathbf{b}^{(1)}$ are presented, activate ART$_a$ and ART$_b$ categories $J = 1$ and $K = 1$, and the category $J = 1$ learns to predict category $K = 1$, thus associating $\mathbf{a}^{(1)}$ with $\mathbf{b}^{(1)}$. Next $\mathbf{a}^{(2)}$ then $\mathbf{b}^{(2)}$ are presented. Vector $\mathbf{a}^{(2)}$ first activates $J = 1$ without reset, since

$$\frac{|\mathbf{a}^{(2)} \cap \mathbf{z}_1^a|}{|\mathbf{a}^{(2)}|} = \frac{3}{4} \ge \rho_a = \overline{\rho_a}. \tag{43}$$

However, node $J = 1$ predicts node $K = 1$. Since

$$\frac{|\mathbf{b}^{(2)} \cap \mathbf{z}_1^b|}{|\mathbf{b}^{(2)}|} = 0 < \rho_b, \tag{44}$$

ART$_b$ search leads to activation of a different F_2^b node, $K = 2$. Because of the conflict between the prediction ($K = 1$) made by the active F_2^a node and the currently active F_2^b node ($K = 2$), the Map Field orienting subsystem resets F_2^a, but without match tracking. Thereafter a new F_2^a node ($J = 2$) learns to predict the correct F_2^b node ($K = 2$), associating $\mathbf{a}^{(2)}$ with $\mathbf{b}^{(2)}$.

Vector $\mathbf{a}^{(3)}$ first activates $J = 2$ without ART$_a$ reset, thus predicting $K = 2$, with $\mathbf{z}_2^b = \mathbf{b}^{(2)}$. However, $\mathbf{b}^{(3)}$ mismatches $\mathbf{z}_2^b$, leading to activation of the F_2^b node $K = 1$, since $\mathbf{b}^{(3)} = \mathbf{b}^{(1)}$. Since the predicted node ($K = 2$) then differs from the active node ($K = 1$), the Map Field orienting subsystem again resets F_2^a. At this point, still without match tracking, the F_2^a node $J = 1$ would become active, without subsequent ART$_a$ reset, since $\mathbf{z}_1^a = \mathbf{a}^{(1)}$ and

$$\frac{|\mathbf{a}^{(3)} \cap \mathbf{a}^{(1)}|}{|\mathbf{a}^{(3)}|} = \frac{3}{5} \ge \rho_a = \overline{\rho_a}. \tag{45}$$

Since node $J = 1$ correctly predicts the active node $K = 1$, no further reset or new learning would occur. On subsequent prediction trials, vector $\mathbf{a}^{(3)}$ would once again activate $J = 2$ and then $K = 2$. When vector $\mathbf{b}^{(3)}$ is not presented, on a test trial, vector $\mathbf{a}^{(3)}$ would not have learned its correct prediction.

With match tracking, when $\mathbf{a}^{(3)}$ is presented, the Map Field orienting subsystem causes ρ_a to increase to a value slightly greater than $|\mathbf{a}^{(3)} \cap \mathbf{a}^{(2)}||\mathbf{a}^{(3)}|^{-1} = 0.8$ while node $J = 2$ is active. Thus after node $J = 2$ is reset, node $J = 1$ will also be reset because

$$\frac{|\mathbf{a}^{(3)} \cap \mathbf{a}^{(1)}|}{|\mathbf{a}^{(3)}|} = 0.6 < 0.8 < \rho_a. \tag{46}$$

The reset of node $J = 1$ permits $\mathbf{a}^{(3)}$ to choose an uncommitted F_2^a node ($J = 3$) that is then associated with the active F_2^b node ($K = 1$). Thereafter each ART$_a$ input predicts the correct ART$_b$ output without search or error.

5.12. Complement Coding Example

The utility of ART$_a$ complement coding is illustrated by the following example. Assume that the nested input pairs in Table 7 are presented to an ARTMAP system in order ($\mathbf{a}^{(3)}$, $\mathbf{b}^{(3)}$), ($\mathbf{a}^{(2)}$, $\mathbf{b}^{(2)}$), ($\mathbf{a}^{(1)}$, $\mathbf{b}^{(1)}$), with match tracking but without complement coding. Choose $\overline{\rho_a} < 0.5$ and $\rho_b > 0$.

Vectors $\mathbf{a}^{(3)}$ and $\mathbf{b}^{(3)}$ are presented and activate ART$_a$ and ART$_b$ categories $J = 1$ and $K = 1$. The system learns to predict $\mathbf{b}^{(3)}$ given $\mathbf{a}^{(3)}$ by associating the F_2^a node $J = 1$ with the F_2^b node $K = 1$.

Next $\mathbf{a}^{(2)}$ and $\mathbf{b}^{(2)}$ are presented. Vector $\mathbf{a}^{(2)}$ first activates $J = 1$ without reset, since $|\mathbf{a}^{(2)} \cap \mathbf{z}_1^a||\mathbf{a}^{(2)}|^{-1} = 1 \geq \rho_a = \overline{\rho_a}$. However, node $J = 1$ predicts node $K = 1$. As in the previous example, after $\mathbf{b}^{(2)}$ is presented, the F_2^b node $K = 2$ becomes active and leads to an inter-ART reset. Match tracking makes $\rho_a > 1$, so F_2^a shuts down until the pair ($\mathbf{a}^{(2)}$, $\mathbf{b}^{(2)}$) shuts off. Pattern $\mathbf{b}^{(2)}$ is coded in ART$_b$ as $\mathbf{z}_2^b$, but no learning occurs in the ART$_a$ and F^{ab} modules.

Next $\mathbf{a}^{(1)}$ activates $J = 1$ without reset, since $|\mathbf{a}^{(1)} \cap \mathbf{z}_1^a||\mathbf{a}^{(1)}|^{-1} = 1 \geq \rho_a = \overline{\rho_a}$. Since node $J = 1$ predicts the correct pattern $\mathbf{b}^{(1)} = \mathbf{z}_1^b$, no reset ensues. Learning does occur, however, since $\mathbf{z}_1^a$ shrinks to $\mathbf{a}^{(1)}$. If each input can be presented only once, $\mathbf{a}^{(2)}$ does not learn to predict $\mathbf{b}^{(2)}$. However if the input pairs are presented repeatedly, match tracking allows ART$_a$ to establish 3 category nodes and an accurate mapping.

With complement coding, the correct map can be learned on-line for any $\overline{\rho_a} > 0$. The critical difference is due to the fact that $|\mathbf{a}^{(2)} \cap \mathbf{z}_1^a||\mathbf{a}^{(2)}|^{-1}$ now equals 5/6 when $\mathbf{a}^{(2)}$ is first presented, rather than equaling 1 as before. Thus either ART$_a$ reset (if $\overline{\rho_a} > 5/6$) or match tracking (if $\overline{\rho_a} \leq 5/6$) establishes a new ART$_a$ node rather than shutting down on that trial. On the next trail, $\mathbf{a}^{(1)}$ also establishes a new ART$_a$ category that maps to $\mathbf{b}^{(1)}$.

The Appendix outlines ARTMAP system responses to various input situations, namely, combinations of: **a** without **b**, **b** without **a**, **a** then **b**, **b** then **a**, **a** making a prediction or making no prediction, and **a**'s prediction matching or mismatching **b**.

REFERENCES

Bullock, D., & Grossberg, S. (1988). Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. *Psychological Review*, **95**, 49–90.

Carpenter, G. A. (1989). Neural network models for pattern recognition and associative memory. *Neural Networks*, **2**, 243–257.

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

Carpenter, G. A., & Grossberg, S. (1987b). ART 2: Stable self-organization of pattern recognition codes for analog input patterns. *Applied Optics*, **26**, 4919–4930.

Carpenter, G. A., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, **21**, 77–88.

Carpenter, G. A., & Grossberg, S. (1990). ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, **3**, 129–152.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, in press.

Grossberg, S. (1969). On learning and energy-entropy dependence in recurrent and nonrecurrent signed networks. *Journal of Statistical Physics*, **1**, 319–350.

Grossberg, S. (1976a). Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors. *Biological Cybernetics*, **23**, 121–134.

Grossberg, S. (1976b). Adaptive pattern classification and universal recoding, II: Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, **23**, 187–202.

Grossberg, S. (1982a). *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*. Boston, MA: Reidel Press.

Grossberg, S. (1982b). Processing of expected and unexpected events during conditioning and attention: A psychophysiological theory. *Psychological Review*, **89**, 529–572.

Grossberg, S. (1984). Some psychophysiological and pharmacological correlates of a developmental, cognitive, and motivational theory. In R. Karrer, J. Cohen, and P. Tueting (Eds.), *Brain and Information: Event Related Potentials* (pp. 58–151). New York: New York Academy of Sciences.

Grossberg, S. (Ed.) (1987a). *The adaptive brain, I: Cognition, learning, reinforcement, and rhythm*. Amsterdam: Elsevier/North-Holland.

Grossberg, S. (Ed.) (1987b). *The adaptive brain, II: Vision, speech, language, and motor control*. Amsterdam: Elsevier/North-Holland.

Grossberg, S. (1988a). Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks*, **1**, 17–61.

Grossberg, S. (Ed.) (1988b). *Neural networks and natural intelligence*. Cambridge, MA: MIT Press.

Iba, W., Wogulis, J., & Langley, P. (1988). Trading off simplicity and coverage in incremental concept learning. In *Proceedings of the 5th International Conference on Machine Learning*. Ann Arbor, MI: Morgan Kaufmann, 73–79.

Kendall, M. G., & Stuart, A. (1966). *The advanced theory of statistics, Volume* 3. New York: Haffner, Chapter 43.

Lincoff, G. H. (1981). *The Audubon Society field guide to North American mushrooms*. New York: Alfred A. Knopf.

Parker, D. B. (1982). *Learning-logic*. Invention Report S81-64, File 1, Office of Technology Licensing, Stanford University, CA.

Rumelhart, D. E., & McClelland, J. L. (Eds.). (1986). *Parallel distributed processing, Volume* 1. Cambridge, MA: MIT Press.

Searle, J. R. (1983). *Intentionality, an essay in the philosophy of mind*. Cambridge: Cambridge University Press.

Schlimmer, J. S. (1987a). *Mushroom database*. UCI Repository of Machine Learning Databases. (aha@ics.uci.edu)

Schlimmer, J. S. (1987b). *Concept acquisition through representational adjustment* (Technical Report 87-19). Doctoral dissertation, Department of Information and Computer Science, University of California at Irvine.

Werbos, P. (1974). *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. Cambridge, MA: Harvard University.

Werbos, P. (1982). Applications of advances in nonlinear sensitivity analysis. In A. V. Balakrishnan, M. Thoma, R. F. Drenick, and F. Kozin (Eds.), *Lecture Notes in Control and Information Sciences, Volume* 38: *System Modeling and Optimization*. New York: Springer-Verlag.

APPENDIX

A1. Simulation Algorithms

A1.1. ART 1 algorithm

Fast-learn ART 1 with binary $F_0 \rightarrow F_1$ input vector **I** and choice at F_2 can be simulated by following the rules below. Fields F_0 and F_1 have M nodes and field F_2 has N nodes.

Initial values

Initially all F_2 nodes are said to be *uncommitted*. Weights Z_{ij} in $F_1 \rightarrow F_2$ paths initially satisfy

$$Z_{ij}(0) = \alpha_j, \tag{A1}$$

where $\mathbf{Z}_j \equiv (Z_{1j}, \ldots, Z_{Mj})$ denotes the bottom-up $F_1 \rightarrow F_2$ weight vector. Parameters α_j are ordered according to

$$\alpha_1 > \alpha_2 > \ldots > \alpha_N, \tag{A2}$$

where

$$0 < \alpha_j < \frac{1}{(\beta + |\mathbf{I}|)} \tag{A3}$$

for $\beta > 0$ and for any admissible $F_0 \rightarrow F_1$ input **I**. In the simulations in this article, α_j and β are small.

Weights z_{ji} in $F_2 \rightarrow F_1$ paths initially satisfy

$$z_{ji}(0) = 1. \tag{A4}$$

The top-down, $F_2 \rightarrow F_1$ weight vector $(z_{j1}, \ldots, z_{jM})$ is denoted $\mathbf{z}_j$.

***F_1* activation**

The binary F_1 output vector $\mathbf{x} \equiv (x_1, \ldots, x_M)$ is given by

$$\mathbf{x} = \begin{cases} \mathbf{I} & \text{if } F_2 \text{ is inactive} \\ \mathbf{I} \cap \mathbf{z}_J & \text{if the } Jth\ F_2 \text{ node is active.} \end{cases} \tag{A5}$$

***$F_1 \rightarrow F_2$* input**

The input T_j from F_1 to the jth F_2 node obeys

$$T_j = \begin{cases} |\mathbf{I}|\alpha_j & \text{if } j \text{ is an uncommitted node index} \\ |\mathbf{I} \cap \mathbf{z}_j|/(\beta + |\mathbf{z}_j|) & \text{if } j \text{ is a committed node index.} \end{cases} \tag{A6}$$

The set of committed F_2 nodes and update rules for vectors $\mathbf{z}_j$ and $\mathbf{Z}_j$ are defined iteratively below.

***F_2* choice**

If F_0 is active ($|\mathbf{I}| > 0$), the initial choice at F_2 is one node with index J satisfying

$$T_J = \max_j(T_j). \tag{A7}$$

If more than one node is maximal, one of these is chosen at random. After an input presentation on which node J is chosen, J becomes *committed*. The F_2 output vector is denoted by $\mathbf{y} \equiv (y_1, \ldots, y_N)$.

Search and resonance

ART 1 search ends upon activation of an F_2 category with index $j = J$ that has the largest T_j value and that also satisfies the inequality

$$|\mathbf{I} \cap \mathbf{z}_J| \geq \rho|\mathbf{I}| \tag{A8}$$

where ρ is the ART 1 vigilance parameter. If such a node J exists, that node remains active, or in *resonance*, for the remainder of the input presentation. If no node satisfies (A8), F_2 remains inactive after search, until **I** shuts off.

Fast learning

At the end of an input presentation the $F_2 \rightarrow F_1$ weight vector $\mathbf{Z}_J$ satisfies

$$\mathbf{z}_J = \mathbf{I} \cap \mathbf{z}_J^{(\text{old})} \tag{A9}$$

where $\mathbf{z}_J^{(\text{old})}$ denotes $\mathbf{z}_J$ at the start of the current input presentation. The $F_1 \rightarrow F_2$ weight vector $\mathbf{Z}_J$ satisfies

$$\mathbf{Z}_J = \frac{\mathbf{I} \cap \mathbf{z}_J^{(\text{old})}}{\beta + |\mathbf{I} \cap \mathbf{z}_J^{(\text{old})}|}. \tag{A10}$$

A1.2. ARTMAP Algorithm

The ARTMAP system incorporates two ART modules and an inter-ART module linked by the following rules.

ART$_a$ and ART$_b$

ART$_a$ and ART$_b$ are fast-learn ART 1 modules. Inputs to ART$_a$ may, optionally, be in the complement code form. Embedded in an ARTMAP system, these modules operate as outlined above, with the following additions. First, the ART$_a$ vigilance parameter ρ_a can increase during inter-ART reset according to the *match tracking* rule. Second, the Map Field F^{ab} can *prime* ART$_b$. That is, if F^{ab} sends nonuniform input to F_2^b in the absence of an $F_0^b \rightarrow F_1^b$ input **b**, then F_2^b remains inactive. However, as soon as an input **b** arrives, F_2^b chooses the node K receiving the largest $F^{ab} \rightarrow F_2^b$ input. Node K, in turn, sends to F_1^b the top-down input $\mathbf{z}_K^b$. Rules for match tracking and complement coding are specified below.

Let $\mathbf{x}^a \equiv (x_1^a \ldots x_{Ma}^a)$ denote the F_1^a output vector; let $\mathbf{y}^a \equiv (y_1^a \ldots y_{Na}^a)$ denote the F_2^a output vector; let $\mathbf{x}^b \equiv (x_1^b \ldots x_{Mb}^b)$ denote the F_1^b output vector; and let $\mathbf{y}^b \equiv (y_1^b \ldots y_{Nb}^b)$ denote the F_2^b output vector. The Map Field F^{ab} has N_b nodes and binary output vector **x**. Vectors $\mathbf{x}^a$, $\mathbf{y}^a$, $\mathbf{x}^b$, $\mathbf{y}^b$, and **x** are set to **0** between input presentations.

Map Field learning

Weights w_{jk}, where $j = 1 \ldots N_a$ and $k = 1 \ldots N_b$, in $F_2^a \rightarrow F^{ab}$ paths initially satisfy

$$w_{jk}(0) = 1. \tag{A11}$$

Each vector $(w_{j1}, \ldots, w_{jNb})$ is denoted $\mathbf{w}_j$. During resonance with the ART$_a$ category J active, $\mathbf{w}_J \rightarrow \mathbf{x}$. In fast learning, once J learns to predict the ART$_b$ category K, that association is permanent; i.e., $w_{JK} = 1$ for all times.

Map Field activation

The F^{ab} output vector **x** obeys

$$\mathbf{x} = \begin{cases} \mathbf{y}^b \cap \mathbf{w}_J & \text{if the } J\text{th } F_2^a \text{ node is active and } F_2^b \text{ is active} \\ \mathbf{w}_J & \text{if the } J\text{th } F_2^a \text{ node is active and } F_2^b \text{ is inactive} \\ \mathbf{y}^b & \text{if } F_2^a \text{ is inactive and } F_2^b \text{ is active} \\ \mathbf{0} & \text{if } F_2^a \text{ is inactive and } F_2^b \text{ is inactive.} \end{cases} \tag{A12}$$

Match tracking

At the start of each input presentation the ART$_a$ vigilance parameter ρ_a equals a baseline vigilance $\overline{\rho_a}$. The Map Field vigilance parameter is ρ. If

$$|\mathbf{x}| < \rho|\mathbf{y}^b|, \tag{A13}$$

then ρ_a is increased until it is slightly larger than $|\mathbf{a} \cap \mathbf{z}_J^a||\mathbf{a}|^{-1}$. Then

$$|\mathbf{x}^a| = |\mathbf{a} \cap \mathbf{z}_J^a| < \rho_a|\mathbf{a}|, \tag{A14}$$

where **a** is the current ART$_a$ input vector and J is the index of the active F_2^a node. When this occurs, ART$_a$ search leads either to activation of a new F_2^a node J with

$$|\mathbf{x}^a| = |\mathbf{a} \cap \mathbf{z}_J^a| \geq \rho_a|\mathbf{a}| \tag{A15}$$

and

$$|\mathbf{x}| = |\mathbf{y}^b \cap \mathbf{w}_J| \geq \rho|\mathbf{y}^b|; \tag{A16}$$

or, if no such node exists, to the shut-down of F_2^a for the remainder of the input presentation.

Complement coding

This optional feature arranges ART$_a$ inputs as vectors

$$(\mathbf{a}, \mathbf{a}^c) \equiv (a_1 \ldots a_{Ma}, a_1^c \ldots a_{Ma}^c), \tag{A17}$$

where

$$a_i^c \equiv 1 - a_i. \tag{A18}$$

Complement coding may be useful if the following set of circumstances could arise: an ART$_a$ input vector **a** activates an F_2^a node J previously associated with an F_2^b node K; the current ART$_b$ input **b** mismatches $\mathbf{z}_K^b$; and **a** is a subset of $\mathbf{z}_J^a$. These circumstances never arise if all $|\mathbf{a}| \equiv$ constant. For the simulations in this article, $|\mathbf{a}| = 22$. With complement coding, $|(\mathbf{a}, \mathbf{a}^c)| \equiv \mathbf{M}_a$.

A2. ARTMAP Processing

The following nine cases summarize fast-learn ARTMAP system processing with choice at F_2^a and F_2^b and with Map Field vigilance $\rho > 0$. Inputs **a** and **b** could appear alone, or one before the other.

Input **a** could make a prediction based on prior learning or make no prediction. If **a** does make a prediction, that prediction may be confirmed or disconfirmed by **b**. The system follows the rules outlined in the previous section assuming, as in the simulations, that all $|\mathbf{a}| \equiv$ constant and that complement coding is not used. For each case, changing weight vectors $\mathbf{z}_J^a$, $\mathbf{z}_K^b$, and $\mathbf{w}_K$ are listed. Weight vectors $\mathbf{Z}_J^a$ and $\mathbf{Z}_K^b$ change accordingly, by (A10). All other weights remain constant.

Case 1: ***a** only, no prediction.* Input **a** activates a matching F_2^a node J, possibly following ART$_a$ search. All $F_2^a \rightarrow F^{ab}$ weights $w_{Jk} = 1$, so all $x_k = 1$. ART$_b$ remains inactive. With learning $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$.

Case 2: ***a** only, with prediction.* Input **a** activates a matching F_2^a node J. Weight $w_{JK} = 1$ while all other $w_{Jk} = 0$, and $\mathbf{x} = \mathbf{w}_J$. F_2^b is primed, but remains inactive. With learning, $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{(\text{old})} \cap \mathbf{a}$.

Case 3: ***b** only.* Input **b** activates a matching F_2^b node K, possibly following ART$_b$ search. At the Map Field, $\mathbf{x} = \mathbf{y}^b$. ART$_a$ remains inactive. With learning, $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$.

Case 4: ***a** then **b**, no prediction.* Input **a** activates a matching F_2^a node J. All x_k become 1 and ART$_b$ is inactive, as in Case 1. Input **b** then activates a matching F_2^b node K, as in Case 3. At the Map Field $\mathbf{x} \rightarrow \mathbf{y}^b$; that is, $x_K = 1$ and other $x_k = 0$. With learning $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$, $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$, and $\mathbf{w}_J \rightarrow \mathbf{y}^b$; i.e., J learns to predict K.

Case 5: ***a** then **b**, with prediction confirmed.* Input **a** activates a matching F_2^a node J, which in turn activates a single Map Field node K and primes F_2^b, as in Case 2. When input **b** arrives, the Kth F_2^b node becomes active and the prediction is confirmed; that is,

$$|\mathbf{b} \cap \mathbf{z}_K^b| \geq \rho_b |\mathbf{b}|. \tag{A19}$$

Note that K may not be the F_2^b node **b** would have selected without the $F^{ab} \rightarrow F_2^b$ prime. With learning, $\mathbf{z}_J \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$ and $\mathbf{z}_K^b \rightarrow \mathbf{z}_J^{b(\text{old})} \cap \mathbf{b}$.

Case 6: ***a** then **b**, prediction not confirmed.* Input **a** activates a matching F_2^a node, which in turn activates a single Map Field node and primes F_2^b, as in Case 5. When input **b** arrives, (A19) fails, leading to reset of the F_2^b node via ART$_b$ reset. A new F_2^b node K that matches **b** becomes active. The mismatch between the $F_2^a \rightarrow F^{ab}$ weight vector and the new F_2^b vector $\mathbf{y}^b$ sends Map Field activity $\mathbf{x}$ to $\mathbf{0}$, by (A12), leading to Map Field reset, by (A13). By match tracking, ρ_a grows until (A14) holds. This triggers an ART$_a$ search that will continue until, for an active F_2^a node J, $w_{JK} = 1$, and (A15) holds. If such an F_2^a node does become active, learning will follow, setting $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$ and $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$. If the F_2^a node J is uncommitted, learning sets $\mathbf{w}_J \rightarrow \mathbf{y}^b$. If no F_2^a node J that becomes active satisfies (A15) and (A16), F_2^a shuts down until the inputs go off. In that case, with learning, $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$.

Case 7: ***b** then **a**, no prediction.* Input **b** activates a matching F_2^b node K, then $\mathbf{x} = \mathbf{y}^b$, as in Case 3. Input **a** then activates a matching F_2^a node J with all $w_{Jk} = 1$. At the Map Field, $\mathbf{x}$ remains equal to $\mathbf{y}^b$. With learning, $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$, $\mathbf{w}_J \rightarrow \mathbf{y}^b$, and $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$.

Case 8: ***b** then **a**, with prediction confirmed.* Input **b** activates a matching F_2^b node K, then $\mathbf{x} = \mathbf{y}^b$, as in Case 7. Input **a** then activates a matching F_2^a node J with $w_{JK} = 1$ and all other $w_{Jk} = 0$. With learning $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$ and $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$.

Case 9: ***b** then **a**, prediction not confirmed.* Input **b** activates a matching F_2^b node K, then $\mathbf{x} = \mathbf{y}^b$ and input **a** activates a matching F_2^a node, as in Case 8. However (A16) fails and $\mathbf{x} \rightarrow \mathbf{0}$, leading to a Map Field reset. Match tracking resets ρ_a as in Case 6, ART$_a$ search leads to activation of an F_2^a node (J) that either predicts K or makes no prediction, or F_2^a shuts down. With learning $\mathbf{z}_K^b \rightarrow \mathbf{z}_K^{b(\text{old})} \cap \mathbf{b}$. If J exists, $\mathbf{z}_J^a \rightarrow \mathbf{z}_J^{a(\text{old})} \cap \mathbf{a}$; and if J initially makes no prediction, $\mathbf{w}_J \rightarrow \mathbf{y}^b$, i.e., J learns to predict K.