Distributed ARTMAP

Gail A. Carpenter and Boriana L. Milenova
Department of Cognitive and Neural Systems, Boston University, 677 Beacon Street
Boston, Massachusetts 02215 USA
gail@cns.bu.edu, boriana@cns.bu.edu

Abstract

Distributed coding at the hidden layer of a multi–layer perceptron (MLP) endows the network with memory compression and noise tolerance capabilities. However, an MLP typically requires slow off–line learning to avoid catastrophic forgetting in an open input environment. An adaptive resonance theory (ART) model is designed to guarantee stable memories even with fast on–line learning. However, ART stability typically requires winner–take–all coding, which may cause category proliferation in a noisy input environment. Distributed ARTMAP (dARTMAP) seeks to combine the computational advantages of MLP and ART systems in a real–time neural network for supervised learning. This system incorporates elements of the unsupervised dART model as well as new features, including a content–addressable memory (CAM) rule. Simulations show that dARTMAP retains fuzzy ARTMAP accuracy while significantly improving memory compression. The model's computational learning rules correspond to paradoxical cortical data.

Distributed Coding By Adaptive Resonance Systems

Adaptive resonance theory (ART) began with an analysis of human cognitive information processing [19]. Fundamental computational design goals have always included memory stability with fast or slow learning in an open and evolving input environment. As a real–time model of dynamic processes, an ART network is characterized by a system of ordinary differential equations, which are approximated by an algorithm for implementation purposes. In a general ART system, an input is presumed to generate a characteristic pattern of activation, or *spatial code*, that may be distributed across many nodes in a field representing a brain region such as the inferior temporal cortex (e.g., Miller, Li, and Desimone [23]).

While ART code representations may be distributed in theory, in practice nearly all ART networks feature winner–take–all (WTA) coding. These systems include ART 1 [5] and fuzzy ART [8], for unsupervised learning, and ARTMAP [7] and fuzzy ARTMAP [6], for supervised learning. The coding field of a supervised system is analogous to the hidden layer of a multi–layer perceptron (MLP) [25, 26, 27, 28], where distributed activation helps the network achieve memory compression and generalization. However, an MLP employs slow learning, which limits adaptation for each input and so requires multiple presentations of the training set. With fast learning, where dynamic variables are allowed to converge to asymptote on each input presentation, MLP memories suffer catastrophic forgetting. However, features of a fast–learn system, such as its ability to encode significant rare cases and to learn quickly in the field, may be essential for a given application domain. Additional ART capabilities, including stable coding and scaling to accommodate large databases, are also essential for many applications, such as the Boeing parts design retrieval system [12].

An overall aim of the distributed ART (dART) research program is to combine the computational advantages of ART and MLP systems. Desirable properties include code stability when learning is fast and on–line, memory compression when inputs are noisy and unconstrained, and real–time system dynamics. Global system design goals, such as stable fast learning, led to the introduction of novel rules for learning and synaptic transmission. These rules, in turn, exhibit dynamics which appear paradoxical at the synaptic level but which are seen to support stable coding at the network level. Markram and Tsodyks [21] have recently discovered similar paradoxical dynamics in cortical neurons.

Distributed Learning

A key step in the derivation of the first family of dART models [3, 4] was the specification of dynamic learning laws for stable distributed coding. These laws generalize the instar [17] and outstar [15, 16] laws used, for

0-7803-5529-6/99/$10.00

example, in fuzzy ART. Instar and outstar learning features a gating operation that permits weight change only when a coding node is active. This property is critical to ART stability. With a distributed code and fast learning, however, instar and outstar dynamics cause catastrophic forgetting. A system such as Gaussian ARTMAP [29] includes many features of a distributed coding network, but retains the instar and outstar learning laws of earlier ART and ARTMAP models. The weight update rules in a Gaussian ARTMAP algorithm therefore approximate a real–time system only in the slow–learn limit. Other ARTMAP variations, such as ART–EMAP [11] and ARTMAP–IC [9] acquire some of the advantages of distributed coding but sidestep the learning problem by permitting distributed activation during testing only.

The distributed instar [4] and distributed outstar [2] laws used in dART dynamically apportion learned changes according to the degree of activation of each coding node, with fast as well as slow learning. The update rules in a dARTMAP implementation algorithm represent exact, closed form solutions of the model differential equations. These solutions are valid across all time scales, with fast or slow learning. When coding is WTA, the distributed learning laws reduce to instar and outstar equations, and dART reduces to fuzzy ART. Similarly, with coding that is WTA during training but distributed during testing, the dARTMAP algorithm reduces to ARTMAP–IC, and further reduces to fuzzy ARTMAP with coding that is WTA during both testing and training.

dARTMAP Design Choices

An ART module is embedded as the primary component of ARTMAP, and similarly an unsupervised dART module is embedded in a supervised dARTMAP network. In applications, ARTMAP requires few design choices: the number of coding nodes is determined by on–line performance, and the default network parameters work well in most settings. In contrast, a general dARTMAP system presents the user with a far greater array of choices, due to the new degrees of freedom afforded by distributed code possibilities. In practice, a number of the "obvious" design choices have failed to produce good performance in simulation studies.

A family of dARTMAP networks that have performed well in pilot studies has been developed as a set of algorithms for implementation [10]. In particular, dARTMAP retains fuzzy ARTMAP test set accuracy while significantly reducing network size. The dARTMAP algorithm is designed both to expedite ready implementation and to foster the development of alternative designs adapted to the demands of new applications.

dARTMAP Algorithm

A number of computational devices that were not part of the more general distributed ART theory were found to be useful in dARTMAP simulations. These include a new rule characterizing the content–addressable memory stored at the coding field in response to a given input, an internal control device that causes the system to alternate between distributed and winner–take–all coding modes, and credit assignment and instance counting.

A geometric representation aids the visualization of distributed ARTMAP computational dynamics. Since the algorithm reduces to fuzzy ARTMAP when coding is winner–take–all, the geometric characterization of dARTMAP builds upon the geometry of fuzzy ARTMAP, which represents weight vectors as category boxes in input space. The relationship between these boxes and a system input determines the order in which categories are searched, and box expansion represents weight changes during winner–take–all learning.

Distributed ARTMAP replaces the long–term memory weights of fuzzy ARTMAP with dynamic weights, which depend on short–term memory coding node activations as well as long–term memory. The corresponding geometric representation replaces each fuzzy ARTMAP category box with a nested family of boxes, one for each coding node activation value. Some or all of these coding boxes may expand during dARTMAP learning, but the geometry shows how the system preserves dynamic range with fast as well as slow learning. The rule in the dARTMAP algorithm that characterizes the signal transmitted to the coding field in response to a given input admits a geometric interpretation, as does the rule characterizing the response of the content–addressable memory to the incoming signal.

A series of simulations indicate how the dARTMAP algorithm works [10]. Distributed prediction in the basic algorithm reduces network size, but this system uses only binary connections from the coding field to the output field. Performance can be improved by augmenting the trained dARTMAP system with a linear output map such as Adaline. Other simulations analyze the role of dARTMAP learning that takes place in the distributed mode, as opposed to the winner–take–all mode. By varying the degree of pattern contrast in the content–addressable memory system, dARTMAP performance can be improved, without increasing network size. Possible dARTMAP variations point to directions for future research.

CAM Rules, Coding Modes, and Credit Assignment

The unsupervised distributed ART network [3,4] features a number of innovations that differentiate it from previous ART networks, including a new architecture configuration and distributed instar and outstar learning laws. In order to stabilize fast learning with distributed codes, dART represents the unit of long–term memory (LTM) as a subtractive threshold rather than a traditional multiplicative weight. Despite their different architectures, a dART algorithm reduces to fuzzy ART when coding is winner–take–all. While a dART module is the basic component of a supervised dARTMAP system, the algorithm also employs additional devices not included in the previous distributed ART description. These features, including a new rule defining coding field activation, alternation between WTA and distributed coding modes, and credit assignment, will now be described.

Increased Gradient CAM Rule

A neural network field of strongly competitive nodes can, once activated by an initial input, maintain a short–term memory (STM) activation pattern even after the input is removed. A new input then requires some active reset process before it can instate a different code, or content–addressable memory (CAM). A *CAM rule* specifies a function that characterizes the steady–state STM response to a given vector of inputs converging upon a field of neurons.

Traditional CAM rules include McCulloch–Pitts activation, which makes STM proportional to input [22]; a power rule, which makes STM proportional to input raised to a power p; and a WTA rule, which concentrates all activation at the node receiving the largest net input. Other CAM rules include Gaussian activation functions, as used, for example, in radial basis function networks [24]. A power rule reduces to a McCulloch–Pitts rule when $p=1$ and converges to a WTA rule as $p \rightarrow \infty$. Moving p from 0 toward infinity produces a stored STM pattern that is a progressively contrast–enhanced transformation of the input vector. In many examples, however, a power rule is problematic because differences among input components are small. A CAM system may then require unreasonably large powers p to produce significant differences among STM activations.

The CAM rule used in the dARTMAP algorithm is designed to enhance input differences as represented in the distributed internal code without raising input components to high powers. It is therefore called the *increased gradient CAM rule.* Beyond its role in the present system, this rule is useful for defining the steady–state activation function in other neural networks. The increased gradient rule includes a power p for contrast control. The role of p is analogous to the role of variance in Gaussian activation functions [20, 24]. A geometric representation of dARTMAP provides a natural interpretation of the increased gradient CAM rule.

Distributed and Winner–take–all Coding Modes

The increased gradient CAM rule solves a pattern separation problem that often arises in neural systems, where each element has a limited dynamic range. A second common problem is how to choose the size of a neural network. In a multi–layer perceptron, for example, deciding on the number of hidden units is a critical design choice. With WTA coding, ARTMAP determines network size by adding category nodes incrementally, to meet the demands of on–line predictive accuracy. Some types of MLP networks have also been designed to add hidden units incrementally. A cascade correlation architecture, for example, creates a hierarchy of single–unit hidden layers until the error criterion is met [14], but weights in all lower layers are frozen during learning associated with the top layer.

With distributed coding, a dARTMAP network could, in principle, operate with a field of coding nodes that are fixed *a priori.* In practice, this type of network did not produce satisfactory results in simulation studies, where fast learning tended to make the learned representations too uniform. To solve this problem, the dARTMAP algorithm alternates between distributed and winner–take–all coding modes, as follows.

Each dARTMAP input first activates a distributed code. If this code produces a correct prediction, learning proceeds in the *distributed coding mode.* If the prediction is incorrect, the network resets the active code via ARTMAP *match tracking feedback* [7]. In ARTMAP networks, the reset process triggers a search for a category node that can successfully code the current input. In dARTMAP, reset also places the system in a *WTA coding mode* for the duration of the search. The switch from a distributed mode to a WTA mode could be implemented in a competitive network by means of a nonspecific signal that increases the strength of intrafield inhibition [13, 18]. Such an arousal signal might be interpreted as an increase in overall attentiveness in response to an error signal or alarm, the computational result being a sharpened focus on the most salient input features.

In WTA mode, dARTMAP can, like ARTMAP, add nodes incrementally as needed. When a coding node is added to the network, it becomes permanently associated with the output class that is active at the time. From then on, the network predicts this class whenever the same coding node is chosen in WTA mode. In distributed mode, STM activations across all nodes that project to a given output class provide evidence in favor of that outcome. Despite its computational advantages, the winner–take–all possibility implies that dARTMAP coding is not fully distributed all the time, indicating one possible direction for future system modifications.

Credit Assignment, Instance Counting, and Match Tracking

When a dARTMAP network makes a distributed prediction, some of the active coding nodes may be linked to an incorrect outcome. In a real–time network, a feedback loop for credit assignment would suppress activation in these nodes during training. Credit assignment allows learning to enhance only those portions of an active code that are associated with the correct outcome. This procedure is similar to credit assignment algorithms widely used in other neural networks (e.g., [29]) and genetic algorithms (e.g., [1]).

The current simulations were also found to benefit from design features used in the ARTMAP–IC network. These include instance counting of category exemplars and the MT– match tracking search rule. Instance counting biases output predictions according to previous coding node activations summed over training set inputs. The MT– search rule generally improves memory compression compared to the original ARTMAP match tracking algorithm (MT+). It also permits a system to encode inconsistent cases, where two identical training set inputs are associated with different outcomes. Inconsistent cases are common in medical databases, for example.

Aspects of the dARTMAP algorithm such as the increased gradient CAM rule, the combination of WTA with distributed coding during training, credit assignment, and instance counting are not necessarily fundamental principles intrinsic to the class of all dARTMAP networks. Rather, they are developed for the pragmatic purpose of defining one set of dARTMAP systems with the desired computational properties.

Acknowledgements

This research was supported in part by the Office of Naval Research (ONR N00014–95–1–0409 and ONR N00014–95–1–0657).

Technical Report CAS/CNS TR-99-013, Boston, MA: Boston University.

References

[1] Booker, L.B., Goldberg, D.E., & Holland, J.H. (1989) Classifier systems and genetic algorithms. *Artificial Intelligence*, 40, 235–282.

[2] Carpenter, G.A. (1994). A distributed outstar network for spatial pattern learning. *Neural Networks*, 7, 159–168.

[3] Carpenter, G.A. (1996). Distributed ART networks for learning, recognition, and prediction. *Proceedings of the World Congress on Neural Networks (WCNN'96)*, pp. 333–344.

[4] Carpenter, G.A. (1997). Distributed learning, recognition, and prediction by ART and ARTMAP neural networks. *Neural Networks*, 10, 1473–1494.

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

[6] Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., & Rosen, D.B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 3, 698–713.

[7] Carpenter, G.A., Grossberg, S., & Reynolds, J.H. (1991). ARTMAP: Supervised real–time learning and classification of nonstationary data by a self–organizing neural network. *Neural Networks*, 4, 565–588.

[8] 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*, 4, 759–771.

[9] Carpenter, G.A. & Markuzon, N. (1998). ARTMAP–IC and medical diagnosis: Instance counting and inconsistent cases. *Neural Networks*, 11.

[10] Carpenter, G.A., Milenova, B.L., & Noeske, B.W. (1998). Distributed ARTMAP: A neural network for fast distributed supervised learning. *Neural Networks*, 11, 793-813.

[11] Carpenter, G.A. & Ross, W.D. (1995). ART–EMAP: An neural network architecture for object recognition by evidence accumulation. *IEEE Transactions on Neural Networks*, 6, 805–818.

[12] Caudell, T.P., Smith, S.D.G., Escobedo, R., & Anderson, M. (1994). NIRS: Large scale ART–1 neural architectures for engineering design retrieval. *Neural Networks*, 7, 1339–1350.

[13] Ellias, S.A. & Grossberg, S. (1975). Pattern formation, contrast control, and oscillations in the short term memory of shunting on–center off–surround networks. *Biological Cybernetics*, 20, 69–98.

[14] Fahlman, S.E. & Lebiere, C. (1990). The cascade–correlation learning architecture. In D.S. Touretzky (Ed.), *Neural information processing systems* 2 (pp. 524–532). San Mateo, CA: Morgan Kaufmann Publishers.

[15] Grossberg, S. (1968). A prediction theory for some nonlinear functional–differential equations. I: Learning of lists. *Journal of Mathematical Analysis and Applications*, 21, 643–694.

[16] Grossberg, S. (1970). Some networks that can learn, remember, and reproduce any number of complicated space–time patterns. *Studies in Applied Mathematics*, 49, 135–166.

[17] Grossberg, S. (1972). Neural expectation: Cerebellar and retinal analogs of cells fired by learnable or unlearned pattern classes. *Kybernetik*, 10, 49–57.

[18] Grossberg, S. (1973). Contour enhancement, short term memory, and constancies in reverberating neural networks. *Studies in Applied Mathematics*, LII, 213–257.

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

[20] Hertz, J., Krogh, A., & Palmer, R.G. (1991). *Introduction to the theory of neural computation*. Reading, MA: Addison–Wesley Publishers

[21] Markram, H. & Tsodyks, M. (1996). Redistribution of synaptic efficacy between neocortical pyramidal neurons. *Nature*, 382, 807–810.

[22] McCulloch, W.S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 9, 127–147.

[23] Miller, E.K., Li, L., & Desimone, R. (1991). A neural mechanism for working and recognition memory in inferior temporal cortex. *Science*, 254, 1377–1379.

[24] Moody, J. & Darken, C.J. (1989). Fast learning in networks of locally–tuned processing units. *Neural Computation*, 1, 281–294.

[25] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65, 386–408.

[26] Rosenblatt, F. (1962). *Principles of Neurodynamics*. Washington, DC: Spartan Books.

[27] Rumelhart, D.E., Hinton, G.E., & Williams, R.J. (1986). Learning internal representations by error propagation. In D.E. Rumelhart & J.L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructures of cognitions*, I (pp. 318–362). Cambridge, MA: MIT Press.

[28] Werbos, P.J. (1974). *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. Unpublished PhD. thesis, Harvard University, Cambridge, MA.

[29] Williamson, J.R. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, 9, 881–897.