CHAPTER 6

LATERAL PRIMING ADAPTIVE RESONANCE THEORY (LAPART)-2: INNOVATION IN ART

T.P. Caudell
Department of Electrical and Computer Engineering
and
The Albuquerque High Performance Computing Center
University of New Mexico
Albuquerque, N.M. 87131
U.S.A.
tpc@eece.unm.edu

M.J. Healy
Phantom Works
The Boeing Company
PO Box 3707 Mail Stop 7L-66
Seattle, Washington 98124-2207
U.S.A.
Michael.J.Healy@boeing.com

In this chapter, we present the results of a study of a new version of the LAPART adaptive inferencing neural network [1], [2]. We will review the theoretical properties of this architecture, called LAPART-2, showing it to converge in at most two passes through a fixed training set of inputs during learning, and showing that it does not suffer from template proliferation. Next, we will show how real-valued inputs to ART and LAPART class architectures are coded into special binary structures using a preprocessing architecture called Stacknet. Finally, we will present the results of a numerical study that gives insight into the generalization properties of the combined Stacknet/LAPART-2 system. This study shows that this architecture not only learns quickly, but maintains excellent generalization even for difficult problems.

1 Introduction

A Holy Grail of neural networks is *fast learning with good generalization*. In many neural architectures, these two trade off against each other, making it difficult to achieve them simultaneously. In this chapter, we present a version of the LAPART adaptive inferencing neural network architecture [1]-[3] that has excellent learning and generalization properties. LAPART architectures are constituted from two or more ART architectures bilaterally connected with adaptive connections. The centerpiece of the chapter is the theorem that under certain broad conditions, LAPART-2 converges in at most two passes or epochs through a fixed set of binary training inputs, where an epoch is the single-time application of a complete list of input patterns to a neural network for learning. In [4], Georgiopoulos, Heileman and Huang proved the upper bound *n-1* on the number of epochs required for convergence for the similar ARTMAP architecture, where *n* is the size of the binary pattern input space; they also proved that the bound can decrease with increasing vigilance parameter values, ρ. ARTMAP performs a function similar to LAPART; both require binary-valued input patterns although, as we will show, they can process real-valued input patterns in a manner equivalent to that of Fuzzy ARTMAP through the use of stack interval pre-processing networks [6]. The ARTMAP result can be thought of as an *n* pass, or finite-pass, convergence result. In these terms, LAPART-2 is then a 2-pass, or fixed-pass, convergence result. To our knowledge this is the first fixed-pass convergence result of its kind.

LAPART-2 is a byproduct of theoretical and empirical investigations into the learning properties of the previous version, LAPART-1. Based upon formal modeling of the semantics of neural networks [7], LAPART-1 was developed specifically to learn logical inference relationships, or rules, between classes of objects from an application. Both the classes and the rules are formed in the synaptic memory of the network according to neural design principles embodied in ART-1 [8], but with a logical design principle from the formal semantic analysis: adaptive neural network connections implement logical implication [7], [9]. The logic, however, changes to adapt to the data. The underlying reason for this is that many inferences made by a neural network prove to be unsound when tested on new data, so the logic must be corrected.

We regard this principle as a point of departure not only for theories of learning with neural networks, but for learning in general.

The overarching question facing us is the following: Can a neural network – ART-1 or LAPART, for example – adapt its logic so that, from some point in time forward, its inferences are valid provided that it is presented with data sufficiently similar to that which it has previously experienced? A positive answer has been provided for ART-1 with a learning parameter set within a range of values commonly used [10]: ART-1 converges on a fixed training set of binary input patterns in a number of presentation epochs that can be calculated from information about the data. The required information is the number N of *different sizes* of input patterns, where the size of a binary-valued pattern is the number of 1-valued components it possesses. If all input patterns have the same size, then only a single epoch is required. This is an N -pass convergence result for unsupervised learning in terms of the stratification of the input space into size classes. This result is especially interesting in that it specializes to one-pass convergence for fixed input pattern size.

The inferences made by an ART-1 network are simply its self-organized classification decisions: The nodes in the F2 (classification) layer of the network compete in a winner-take-all fashion, and a binary *template* pattern comprising adaptive connections from the winner to the F1 (comparison) layer is compared with the input pattern to determine if the input belongs in the corresponding class of patterns. Thus, the classes have two representations: F2 nodes and templates. An ART-1 network has converged on a fixed training set of patterns when all inputs have a *direct access* template in the system – one that causes immediate classification. The point here is that the logic of a trained ART-1 system is valid not only for the training patterns, but for any future input patterns that have direct access templates in the ART-1 memory. However, conditions for other ART-type architectures, in particular LAPART-1, are difficult to derive because of the phenomena that can occur in the more complex situations of inference learning. Assumptions must be made, either upon the architecture in the form of added design constraints or upon the data in the form of "domain constraints." The former is the basis for the design change to create LAPART-2, and the latter is the basis for the hypothesis that makes our two-pass convergence theorem possible.

The LAPART-1 architecture has been described in [1], [2], [3]. It couples two ART-1 networks, designated subnetworks *A* and *B*, in such a way that if subnetwork *A* attempts to assign a class A_i to a binary input pattern I_A the result is an inference that subnetwork *B* will assign its simultaneously-occurring input I_B to a class B_j. The inference is the result of a strong connection between the F2 nodes for classes A_i and B_j in the two subnetworks, and this is denoted $A_i \rightarrow B_j$. In LAPART, each inference is tested in subnetwork *B* through its own vigilance pattern matching operation; if the subnetwork *B* vigilance system is not aroused (hence, the match of I_B to the B_j template pattern is accepted), the inference and, therefore, the subnetwork *A* classification decision were valid. Otherwise, the subnetwork *A* decision is assumed invalid. This is where the LAPART logic must adapt: The subnetwork A class templates must be modified appropriately if a mistaken inference is to result in a lasting correction. On the other hand, the inferencing connections between *A* and *B* classes, once formed, are assumed always correct. Further, they are assumed exclusive: Each subnetwork A class can infer only a single subnetwork *B* class. The phenomena that characterize the complexity of learning with LAPART-1 stem from these assumptions.

In Section 2, we briefly review the architecture and operation of ART-1, Stacknet, which converts real-valued input patterns to a binary structure, and LAPART-1. In Section 3 we present the LAPART-2 [11] algorithm as a means of addressing a phenomenon that can impede learning significantly in LAPART-1. Section 4 presents the learning theorems stating that a LAPART-2 network converges in two passes through a fixed training set. Along with this, we state a theorem showing that the architecture does not generate more templates than there are input examples. Section 5 describes the generalization study and its numerical results. Section 6 is the Discussion.

2 ART-1, Stacknet, and LAPART-1

In this section we briefly review the architecture and operation of ART-1, Stacknet, which converts real-valued input patterns to a binary structure, and LAPART-1.

2.1 Binary Patterns

First, we briefly review some notation and terminology. We shall regard a binary pattern X as a string of numerical 1s and 0s. Certain operations are defined upon binary patterns. First, if n is the number of 0-1 components, each denoted X_k, we write $length(X) = n$. For any two binary patterns X and Y having the same length, we refer to their component-wise minimum $X \wedge Y$, where the minimum operation on components has the properties, $0 \wedge 0 = 0, 1 \wedge 1 = 1, 0 \wedge 1 = 0, 1 \wedge 0 = 0$. For a set, S, of binary patterns all having the same length, with $S = \{X1, X2,\ldots,XN\}$, let $\bigwedge S$ denote the pattern minimum over the set, $\bigwedge S = X1 \wedge X2 \wedge\ldots \wedge XN$. We define the size, |X|, of a binary pattern to be the number of 1s it contains. Finally we denote a "subset" relationship as $X \subseteq Y$, indicating that for every component in binary pattern X that has a 1 value, the same component in Y also has a 1 value.

2.2 ART-1 Architecture

To support our discussion of the LAPART architecture, we briefly summarize the function of an ART-1 network [8]. ART-1 is called an unsupervised learning architecture because it autonomously classifies its input patterns and "remembers" the classes in the form of binary connection-weight template patterns. An ART-1 network has three main layers of nodes. These layers consist of m_1 input (I) nodes, m_1 matching ($F1$) nodes, and m_2 classification (F2) nodes. The I layer serves as the network input interface, with each input node, I_k, supplying excitatory input to its corresponding $F1$ node, $F1_k$. Each binary input pattern I, where $length(I) = m_1$, specifies the activation values of the input nodes for the duration of the presentation of I as the current input. Thus, if input pattern component I_k has the binary value 1, then input node I_k has an activation value of 1 for that pattern, and 0 otherwise. Since the activation value I_k of each input node is directly transmitted to the corresponding node $F1_k$ through the $I_k \rightarrow F1_k$ connection, which has a fixed weight of unity, the initial pattern of activation values over $F1$ is identical with the input pattern. The $F1$ and $F2$ layers interact through adaptive connections, under the control of the gain control (GC) and vigilance (VIG) nodes. The template pattern for each class comprises the connection weights in the unique set of adaptive connections associated with an $F2$ node. At any time, each template, T_i for class *i*, corresponding to node $F2_i$ for $(1 \le i \le m_2)$, has the form

$$T_i = \bigwedge S, \tag{1}$$

where *S* is the set of binary input patterns that has previously been assigned to the class corresponding to T_i and, consequently, may have contributed to the adaptive recoding of the template pattern. An input pattern may contribute to a template at one time and yet may become associated with a different template at a later time, as templates continue to undergo recoding. This effect will occur until the ART-1 network has *perfectly learned* its input space. The authors of the ART-1 architecture characterize the behavior of its unsupervised classification algorithm through stability results in [8]. Further results in [10] include a key learning theorem that states that if a fixed set of patterns is repeatedly presented to an ART-1 network, the algorithm will converge (i.e., perfect learning of the input set will occur) in a finite number of epochs, with the input patterns arbitrarily re-ordered on each epoch. Perfect learning means that each training pattern *I* in the set will have a maximal subset template T_i with $T_i \subseteq I$, where T_i is the largest such template $|T_i| \ge |T_{i'}|$ for all $T_i \subseteq I$. As a consequence, *I* will resonate directly with T_i; that is, *I* will be classified as a member of class *i* (this is called the *direct access property* [8]). Finally, no recoding of T_i will occur, since $T_i \subseteq I$.

Since the vigilance nodes of its ART-1 subnetworks play a fundamental role in the operation of a LAPART network, we review the role of a vigilance node. During the *F2* competition following input of a binary pattern *I*, some *F2* node, $F2_j$ say, wins the competition and tentatively becomes the exclusive class representative for *I*. However, if its associated template pattern T_j is such that

$$| I \wedge T_j | \,/\, | I | < \rho, \tag{2}$$

where ρ is the ART-1 vigilance parameter, then the vigilance node, *VIG*, becomes activated. When this happens, a reset occurs over the *F2* layer, and $F2_j$ becomes suppressed for the duration of the presentation of *I*. This eliminates $F2_j$ from the competition for representing *I* during the current input presentation. When no more resets occur, *resonance* is said to have occurred, and the input has finally been assigned a class. The ART-1 classification algorithm can be summarized as one that solves the combinatorial optimization problem stated as follows:

$$\underset{w.r.t.\ \mu}{maximize}\ |\, I \wedge T_\mu \,| \,/\, (\beta + |\, T_\mu \,|\,)$$

$$subject\ to\ |\, I \wedge T_\mu \,| \,/\, |\, I \,| \geq \rho \qquad (3)$$

A solution value *i* for μ is the index of the *F2* node $F2_i$ that represents the class assigned to *I* with associated template T_i.

For each ART-1 input pattern, unsupervised learning occurs in two phases: (1) recognition of the input pattern as a member of some class, and (2) updating of the class template through synaptic learning. During a resonance, the commonality of the input pattern and template is synaptically learned by the network by adapting the template weight values. This is expressed in the following binary pattern equation:

$$T_{i\text{-}new} = I \wedge T_{i\text{-}old} \qquad (4)$$

which leads to the template property expressed in equation (1). When a class is first established, all connection-weight values in its template are *1s*. Many of these are changed to *0s* via the learning process as the network assigns input patterns to the class.

The next subsection presents a preprocessing network that converts a single real-valued input into a multicomponent pattern containing binary-valued components. The resulting coded pattern is well suited for the processing of an ART-1 network.

2.3 Stacknet

The neural network described in this sub-section, called Stacknet [6], transforms (codes) real-valued components into binary patterns that possess an important property vis-à-vis the processing that occurs within ART-1 networks: binary patterns that are “similar” in an ART-1 sense correspond to real values that are similar in magnitude. This is not true of the usual binary-coded-decimal format used in digital computers, in which *0* and *1* are coefficients of powers of *2*. The codings used here are referred to as *stack numerals* and are similar to “thermometer codes” where a real number is mapped into an interval defined by real-valued *minimum* and *maximum* values. This interval is quantized into *m* subintervals, one of which contains the real input value. Associated with each subinterval is a logical variable. The stack numeral is constructed by setting all of the logical variables for subintervals less than or equal to the one containing the real-valued input to TRUE (or *1*) and those above to FALSE or UNCERTAIN (*0*). If the interval is thought of as being a vertical structure, the set of logical variables forms a stack of *1s* topped by *0s*, totaling *m* components high. The precision of representation is set by the choice of the *max*, *min*, and *m* stack parameters and can be easily matched to the accuracy of a measured input value.

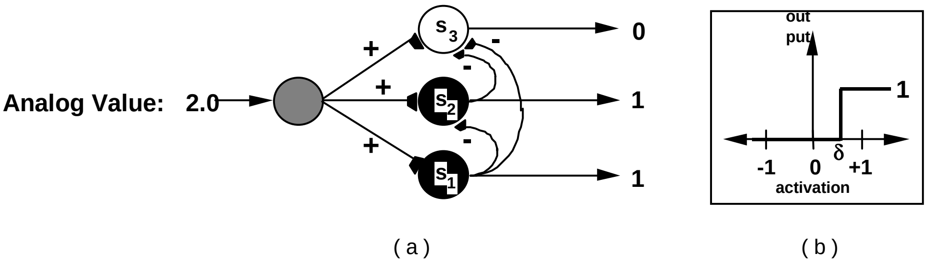


Figure 1. (a) A simple example of a neural implementation of a stack numeral. The complement stack has a different connectivity. (b) The activation function for the stack units with threshold δ.

A simple Stacknet is depicted in Figure 1. The connection strengths of stack inputs are all unity. Each stack node s_i $(1 \leq i \leq m)$ has an activation threshold of magnitude $\delta > 0$. Thus, stack node s_1 can be activated by a signal from the analog source node of magnitude δ. When activated, it emits a signal of strength unity through the connection to its corresponding ART-1 *F1* node. Simultaneously, it emits a signal of strength unity through a system of $(m-1)$ inhibitory connections to higher stack nodes s_2., s_3.,… s_m.. These connections each have weight δ, so that the connection-weighted inhibitory signal arriving at each target node above s_1 has strength $1\cdot \delta = \delta$. The consequence of this is that an input signal of strength $x \geq \delta$ to the stack network from the analog source node is required to activate s_1. However, if $\delta \leq x < 2 \cdot \delta$, only s_1 will be activated, for the inhibitory weighted signal δ from s_1 arriving at each higher of the $m-1$ target nodes s_2., s_3.,… s_m causes the total connection-weighted input t_i into target node s_i $(i > 1)$ to be below threshold, that is

$$t_i = x - (1 \bullet \delta) < 2\delta - \delta = \delta$$

$$or\ t_i < \delta$$

which is below threshold, implying that all higher nodes will remain in their off state. Similarly, stack node s_2 sends out a set of m-2 inhibitory connections of strength δ to stack nodes s_3., s_4.,… s_m.. In general, stack node s_i inhibits the $m - i$ higher nodes. As a consequence, stack node s_i will be activated *if and only if* the input analog signal has strength $x \geq i \cdot \delta$. Thus, an analog number $\delta \leq x \leq (m+1) \cdot \delta$ in magnitude can be represented to within an absolute precision of magnitude δ by the Stacknet network. Thus, if $n \cdot \delta \leq x < (n + 1) \cdot \delta$ stack nodes *1, 2,...,n* will be activated, producing the binary pattern

$$I = [\ 1111\ldots1000\ldots0]$$

where there are *n 1s* and *m-n 0s*. Stacknet takes a single real-valued input and produces a binary-valued output pattern of fixed length *m*.

Two stack numeral binary patterns are similar in the ART1 sense if and only if they fall into a class which is represented by the same template pattern. If the real-valued inputs were coded as powers of two, the usual representation on digital computers, this equivalence would not hold. For example, the numbers *127* and *128* represented in powers of two require *8* bits, with low-order binary digits to the right, yielding the patterns *01111111* and *10000000*, respectively, with a difference in bits equaling *8* out of *8* total, or *100%*. By contrast, if $\delta = 1$ a stack representation requires a minimum of *128* stack nodes (bits) to exactly code the numbers *1, 2, ..., 127, 128,* yielding a difference in bits equaling *1* out of *128* total, or less than *1%*. Stack numerals require more binary components but are more appropriate for coding numbers for ART-1 networks.

Now suppose that several Stacknets, each reading from a different analog source node, are arranged in an input array for an ART-1 system. Here, the total number of binary-valued components that will be generated will be $m = m_3 . m_4$, where m_3 is the number of real-valued inputs to be represented and m_4 is the length of each output stack (assuming uniform stack size). Let $X = (x_1, x_2, \ldots, x_{m1})$ be an array of real-valued variables that are input to the array of Stacknets, and then $I = (I_1, I_2,\ldots I_{m2})$ denotes the *concatenation* of binary stack outputs that represent the components of X to a precision δ, each of length m_4, so $m_1 = m_3 . m_4$. The ART-1 network receives this composite pattern at its F1 layer.

Finally, as pointed out in the Introduction, ART-1 converges on a fixed training set of binary-valued input patterns in a number of presentation epochs equal to the number N of *different sizes* of input patterns, where the size of a binary-valued pattern is the number of *1*-valued components it possesses. If all input patterns have the same size, then only a single epoch is required. Stacknet has a variant that accomplishes this through the use of complement coding. If I is the normal "positive" binary-valued ART-1 input pattern of length m, then we can define the complement "negative" of this pattern to be a pattern I^c of length m, as $I^c = \mathbf{1} - I$, where $\mathbf{1}$ is a pattern of all *1* components. By concatenating the positive and negative patterns together, we form a pattern $C = (I^c, I)$ of length $2m$. If $| I | = n$, then $| I^c | = m - n$. Therefore, $| C | = | I | + | I^c | = m$, independently of n. Using complement coding of stacks representing real-valued inputs will allow ART-1 learning to converge in a *single epoch* for any set of input data.

2.4 LAPART-1

The basic LAPART-1 network architecture [1] is based upon the lateral coupling of two ART-1 subnetworks, referred to as A and B. The interconnects between these two subnetworks force an interaction of the respective classifications performed by the ART-1 subnetworks on their inputs. This modifies their unsupervised learning properties to allow the learning of inferencing relationships between their respective input domains. This can be thought of as supervised learning, or supervised classification. In actuality, however, it is much more general. The usual sense of classification is that of creating a partition of the inputs, that is, separating them into disjoint sets, with a label (the desired output specified by the "teacher") attached to each element of the partition. With the LAPART architecture, we may actually label sets with *sets* – in other words, the network extracts rules with antecedent and consequent predicates. In this discussion, the sets in question will be referred to as classes, because they are sets labeled by ART-1 F2 nodes and coded in templates. Also, the inputs, ART layers, and templates will be labeled with an *A* or *B* referring to the *A* and *B* ART-1 subnetworks.

In a typical LAPART application, two ART-1 subnetworks are presented with a sequence of pairs of simultaneously-occurring input patterns IA_k and IB_k for subnetworks *A* and *B*, respectively. As *A* and *B* form class templates for their inputs, the LAPART-1 network learns inference relations between their classes by forming strong $F2A \rightarrow F2B$ interconnections between pairs of simultaneously-activated *F2A* and *F2B* nodes. Convergence of a LAPART network in a finite number of passes through a training set requires that it reach the following operational state: Presentation of any input pair *(IA, IB)* from the set shall result in pattern *IA* being immediately assigned a class in ART-1 subnetwork *A* through direct access to the class template. Through a strong, learned inferencing connection, the class *F2A* node shall signal a unique *F2B* node to which it is connected, forcing it to become activated. This results in the inferred *B* class template being read out over the *F1B* layer just as pattern *IB* reaches the *F1B* layer. The ensuing vigilance test in subnetwork *B* shall then confirm that the inferred class is an acceptable match for *IB*. That is, the network *B* vigilance node shall remain inactive. Further, the *B* class template shall be a subset template for *IB*. In summary, a final pass through the data shall result in no resets and no synaptic strength changes (i.e., no learning).

To show how a LAPART-1 network learns class-to-class inferences, or rules, from example input pairs, we give a brief summary of its algorithm. Initially, subnetworks *A* and *B* are untrained ART-1 networks. Their *F2* nodes are fully interconnected by $F2A \rightarrow F2B$ connections which are too weak to carry a signal of significant strength; that is, there are no learned inferences. As it processes each input pattern pair *(IA, IB)*, the LAPART network does one of two things: It either forms a new rule or tests a previously learned one. It forms a new rule exactly when subnetwork *A* forms a new class for its input *IA*. That is, if *A* has no acceptable template pattern for *IA* it selects a previously uncommitted *F2A* node, $F2A_i$ according to the ART-1 algorithm. Then, it modifies the adaptive $F1A \rightarrow F2A_i$ and $F2A_i \rightarrow F1A$ connections so that the newly committed template pattern TA_i equals the input *IA*. We denote the newly initialized class by A_i. Following the selection of $F2A_i$, meanwhile, subnetwork *B* has been allowed to read its input *IB*. It selects a node $F2B_j$ and either initializes a new template TB_j or recodes (modifies) a previously committed one. A subnetwork *B* class, $\mathrm{B_j}$, has now been selected simultaneously with the newly initialized subnetwork *A* class A_i. Finally, the $F2A_i \rightarrow F2B_j$ connection strength increases to a maximum, implementing an *inference relationship*, or rule, $A_i \rightarrow B_j$. We write the rule in the form of an implication formula because the future presentation of an input pair for which A_i is the resonating class for the *A* input will result in the inference through the strong $F2A_i \rightarrow F2B_j$ connection that class B_j is appropriate for the *B* input. This strong connection will remain the sole one from A_i to a subnetwork *B* class node.

If subnetwork *A* already contains a class template TA_i that resonates with *IA* on the other hand, then it also has a previously learned class-to-class inference relationship $A_i \rightarrow B_j$. Thus, $F2A_i$ primes $F2B_j$ through the strong $F2A_i \rightarrow F2B_j$ connection, forcing $F2B_j$ to become active and read out the class B_j template over the *F1B* layer. Thus, when it is allowed to read its input, *IB*, subnetwork *B* performs its vigilance pattern-matching test using the template pattern TB_j instead of one that would have been selected through the ART-1 winner-take-all competition in layer *F2B*. This is where the LAPART network tests an existing rule: If the pattern match between the inferred class template TB_j and the input pattern *IB* is not acceptable, that is if

$$| \mathrm{IB} \wedge \mathrm{TB_j} | / | \mathrm{IB} | < \rho\mathrm{B},$$

where ρB is the vigilance threshold for subnetwork B, then a reset occurs in subnetwork *B* – the inference has been disconfirmed. Through the fixed, strong connection $VIGB \rightarrow VIGA$ between the two vigilance nodes, subnetwork *A* is subsequently forced to also undergo a reset, which we call a *lateral reset*. A lateral reset overrides subnetwork *A*'s autonomous, or unsupervised, classification decision and forces it to find an alternative class for its input. The entire process must then be repeated using the reduced set of nodes obtained by inactivating $F2A_i$ and therefore $F2B_j$. Finally, the network either forms a new rule or modifies the templates that are linked through a pre-existing one.

It is interesting to ask whether the LAPART-1 algorithm always converges to the state in which no more resets or template modifications occur – all inferences are correct and learning has ceased. If it does not, are there conditions that can be specified under which it can be guaranteed to converge? Is there, at least, a set of well-defined necessary conditions for convergence? Unfortunately, it cannot be guaranteed that a LAPART-1 network will reach an operational state of convergence on the training set. Our attempts at addressing this issue resulted in the design of the LAPART-2 network and proofs of theorems stating that a LAPART-2 network converges in two passes through a training set.

3 The LAPART-2 Algorithm

In this section, we describe the LAPART-2 architecture [11], which implements neural network design constraints that we derived in order to resolve issues with LAPART-1. The LAPART-2 architecture is identical with the LAPART-1 architecture except in the procedure for a lateral reset. The modified lateral reset procedure results in a rule extraction neural network that converges in two passes through a set of training data, given that certain sufficient conditions hold for the data. Two-pass *supervised* learning is a special case of this, since, as mentioned before, rule consequents in supervised learning are simply class labels assigned by the teacher.

3.1 Forcing Learning to Occur

In LAPART-1, a lateral reset merely disqualifies the active *F2A* node, forcing ART-1 subnetwork *A* to select an alternative resonant node from the set of all *F2A* nodes that have not yet undergone a reset for the current input pair. As in the example let this pair be *(IA, IB)*. It can happen that *IA* has a direct access template TA_i whose choice results in a lateral reset, while a subsequently chosen template $TA_{i'}$ results instead in a valid inference; yet the latter template is also a subset template for *IA* (necessarily, it is smaller, having fewer binary *1* components). That no learning can occur in subnetwork *A* as a consequence of this (because only subset templates were chosen) means that the originally chosen template can remain the direct access template for *IA* afterwards. This allows the same sequence of events in subsequent passes to be repeated for the pair *(IA, IB)*, ensuring that the lateral reset (signaling an incorrect inference) will be repeated.

The LAPART-2 learning algorithm overcomes this learning deficit by allowing the choice of only an *uncommitted F2A* node to represent *IA* following a lateral reset. As a consequence, learning will occur, and in two forms. First, the uncommitted template will be re-coded as *IA*. This recoding represents the network's current state of knowledge about the new class, which consists of a single example. Since there is no $A \rightarrow B$ inference generated, the procedure for adding a new *A* class, $A_{i'}$, comes into play; subnetwork *B* produces a class template $TB_{j'}$ that resonates with *IB* in the usual ART-1 fashion. Unless this is a subset template for *IB* or else corresponds to an uncommitted *F2B* node, an existing *B*-class template will be modified. The second form of learning that occurs is the learning of a strong connection $F2A_{i'} \rightarrow F2B_{j'}$, which implements a newly learned inference relationship, $A_{i'} \rightarrow B_{j'}$.

3.2 Constraints on the Input Data

We shall state two learning theorems in Section 4, the most important result being the convergence of LAPART-2 in two passes through a fixed set of input pattern pairs. See [11] for an additional theorem. Unfortunately, the algorithmic modifications leading to the LAPART-2 architecture are insufficient, by themselves, for a proof of convergence. For this reason, the hypotheses of the learning theorems state assumptions that apply to the input data pattern pairs. Let *mA1* and *mB1* be the number of input pattern components IA_k and IB_l in the inputs to subnetworks *A* and *B*, respectively, and let *KA* and *KB* be integers such that $0 < KA < mA1$ and $0 < KB < mB1$. Hypothesis (i) is the statement that the input patterns for each ART-1 subnetwork have a fixed size, *KA* for subnetwork *A* and *KB* for subnetwork *B*. This may appear to be a strong constraint. However, it is less strong an assumption than is routinely applied with ARTMAP and Fuzzy ARTMAP [4], [5]: In applications of these architectures, it is normally assumed that each input pattern is complement coded, with the effect of making all input patterns the same size.

Hypothesis (ii) in the Two-pass Learning theorem is more complex: It is meant to ensure that LAPART-2 is a *consistent learner* (see [13]). A consistent learner is a machine which, given consistent training data, can successfully learn some specified target concept from that data. In the learning of class-to-class inferences (rule extraction), we apply the assumption that the input pattern pairs are consistent and, as a result, are able to prove that LAPART-2 converges. In the context of LAPART, consistency means that the pattern minimum ($\wedge$) of the subnetwork *B* input patterns with which each subnetwork *A* input pattern is paired can form a template with which each one of them would pass the vigilance test. Without this hypothesis, there could be a subnetwork *A* input pattern *IA* for which the LAPART network was incapable of learning correct *B* inferences; there would always be some *B* input pattern *IB* associated with *IA* that would cause a lateral reset.

4 The Learning Theorems

We can now state the learning theorems for the LAPART-2 neural network architecture. See [11] for the details of the proofs. Only the hypotheses pertaining to the data are stated explicitly. The neural network behavioral hypotheses are implicit in the statement in the theorems. In the following, let *L* be a LAPART-2 network with *mA* and *mB* input nodes for subnetworks *A* and *B* and with vigilance values ρA and ρB. Let *MA* and *MB* be sets of input patterns for subnetworks *A* and *B*, respectively, and let *M* be a set of input pattern pairs $(IA_k, IB_{k,h})$, with $IA_k \in MA$, $IB_{k,h} \in MB$ $(k = 1,\ldots,N;\ h= 1,\ldots, n_k)$. Finally, let $MB_k = \{ IB_{k,h} \mid h = 1, \ldots, n_k \}$. The first theorem follows:

> **Theorem (Two-pass Learning)** Let *L* be a LAPART-2 network whose inputs have the following two properties:
>
> (i) $| IA_k | = KA$, $| IB_{k,h} | = KB$ $(0 < KA < mA;\ 0 < KB < mB)$.

(ii) For an arbitrary subset $S \subseteq MB$ and for any $IA_k \in MA$, if an associated pattern $IB_{k,h}$ is in S (i.e., if $IB_{k,h} \in S \cap MB_k$)

and $\left| \bigwedge S \right| \geq \rho B \bullet KB$

then $\left| (\bigwedge S) \wedge (\bigwedge MB_k) \right| \geq \rho B \bullet KB.$

Then, if each of the elements of M is input to L in each of several passes, with the elements arbitrarily ordered in each pass, there will be no resets and no new templates in subnetworks A and B, and no changes in class assignments in subnetwork A, following the second pass. Recoding can occur only in subnetwork B templates following the second pass. Any such recoding will occur only on the third pass and will have no effect upon the class assignments and inference relationships that L has learned in the first two passes.

Although hypothesis (i) is essential, it is also one that is commonly applied in studies of ARTMAP and LAPART type architectures, and is even considered essential for the correct performance of ARTMAP [4], [5]. It is hypothesis (ii), together with the LAPART-2 modification itself, that is uniquely responsible for the two-step convergence result. This hypothesis, however, is difficult to verify for a given application. It specifies that *any* template that could conceivably be associated with the subnetwork B input patterns that are paired with a single subnetwork A input pattern must admit all of them. For the intended rule extraction applications of LAPART, in which sets of A inputs (rule antecedents) are to be associated with sets of B inputs (rule consequents), it would be impractical to check this. Also, the condition is rather strong – probably stronger than necessary – and is not likely to hold in some cases. See [11] for a further modification of the LAPART architecture that addresses this issue. In the csae of pure clasification problems, like those presented in Section 5, hypothesis (ii) may easily be shown to hold true.

Theorem (LAPART Data Compression) Let L be a LAPART network which is processing input pattern pairs $(IA_{k,}\ IB_{k,h})$ from a set M. Suppose that the B inputs all have the same size $|\ IB_{k,h}\ | = KB$ for all applicable values k, h. Then, the number of laterally connected template pairs

(TA_i, TB_j) generated by *L* does not exceed the number of input pairs ($IA_{k,}$ $IB_{k,h}$) in *M*.

Notice that the LAPART Data Compression theorem requires no constraint on the architecture – it can be any of LAPART-1 or 2 variants. The only restriction on the input data is that the *B* component of all input pattern pairs be the same size. This is much weaker than the hypotheses in the Two-pass Learning theorem. A consequence of the LAPART Data Compression theorem is that the number of $A_i \rightarrow B_j$ rules extracted can be no greater than the number of input data pairs. Neither template proliferation nor rule proliferation is a problem with a LAPART network. The following section further explores the properties of LAPART-2 architectures through numerical simulations.

5 Numerical Experiments

With most learning systems, it is frequently possible to achieve near perfect learning on a fixed training set of data at the expense of either using a large enough set of synaptic weights in the network or reduced performance on an independent testing set of data. The former effect is addressed by the template proliferation result mentioned above. The latter effect is referred to as poor generalization or over-training. Since a theoretical understanding of generalization in LAPART class architectures is still under development, this section addresses the topic through a series of numerical experiments on challenging problems in classification. Note that this class of problem has been used in these studies because of the simplicity of their correctness analysis and the availability of independent theoretical bounds on performance. Note also that issues in generalization exist equally in non-classification type problems, such as inference and rule learning [2], [3].

5.1 Method

Three classification problems were selected to study generalization in LAPART-2 learning [12]. Each problem has the properties of being a two-class problem, with two real-valued feature-space dimensions (x0, x1) for input into the A subnetwork, with statistical overlap between the two class boundaries, and the ability to generate the data ordered pairs algorithmically. The input variables are confined to the [0,1] interval. The three study problems are:

> 1) two equal sized rectangular uniformly distributed classes with 50% overlap in the x0 dimension,
>
> 2) two overlapping normally distributed classes each with different means of (0.333, 0.5) and (0.666, 0.5) respectively, and the same sigmas (0.166, 0.166),
>
> 3) two overlapping normally distributed classes each with the same means of (0.5, 0.5) and differing sigmas (0.166, 0.166) and (0.333, 0.333) respectively.

A computer simulation of LAPART-2 was used to experiment with the three study problems. For each problem, training and testing data sets were independently created using a numerical random number generator that modeled the appropriate statistical distribution. A total of 1000 ordered pairs were produced for a data set (training and testing), 500 for Class 1 and 500 for Class2.

LAPART-2 was configured using complement coded stack (CCS) representations for inputs to both the A and B subnetworks [6]. The input to subnetwork A consisted of two concatenated CCS representations, one for each input dimension, using 1024 bits in the positive stack. The input to unit B was a single CCS representation using 2 bits in the positive stack. The two classes were labeled 10 and 01 respectively.

An experiment consisted of training a LAPART-2 network on the training set until convergence, then computing a performance measure using the testing data set with learning disabled. Since the details of learning in this class of network depends on presentation order of the training data, the performance measures from training with twenty different random orderings were averaged and standard deviations were computed. In addition, statistics for the number of learned templates in the A subnetwork was collected. This gives an indication of the degree of data compression realized by the network. Convergence was declared for a training session when at the end of a presentation epoch, each training pattern had a direct access template [8] in both the A and

B subnetworks. Notice that this requirement is more demanding than is required for the conclusion of the Two-pass Learning theorem.

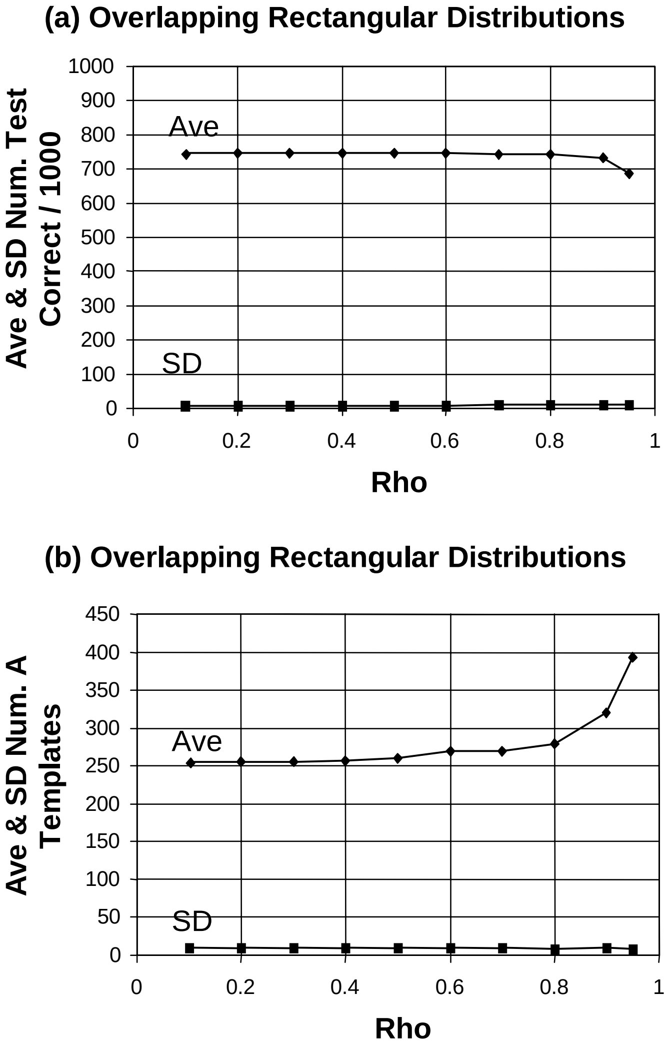
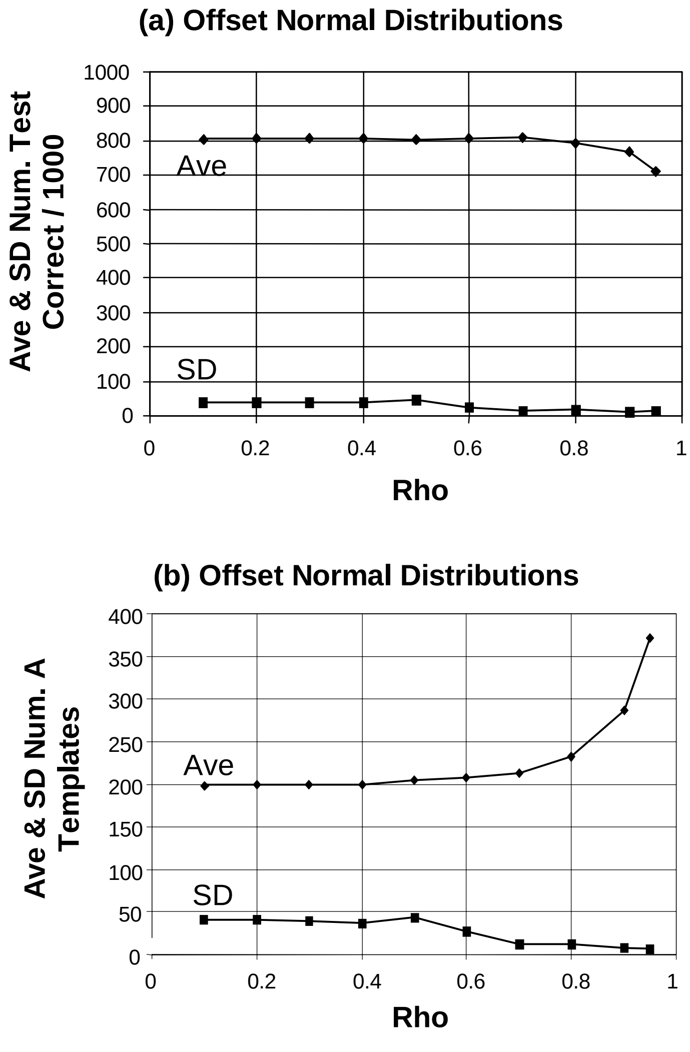


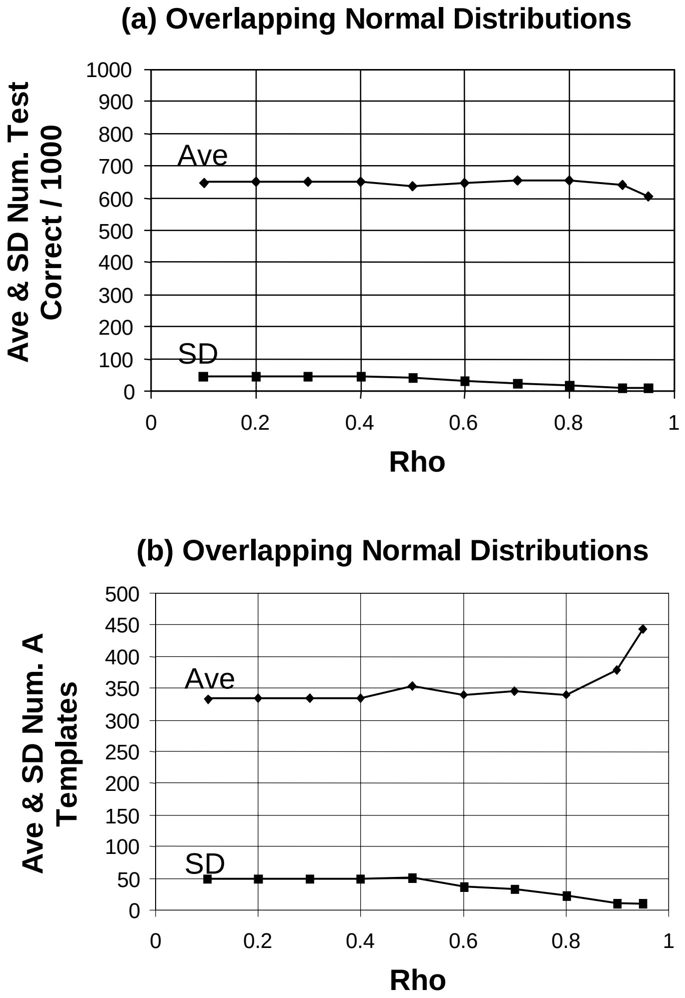
Figure 2. Overlapping Rectangular Distributions: (a) the average and standard deviation for the number of correctly classified testing samples out of 1000 as a function of rho for the A subnetwork; (b) the average and standard deviation for the number of A unit templates as a function of rho.

Since learning in ART-class architectures is also dependent upon the vigilance parameter [8], ρ, the average performance was computed on a grid of ten vigilance settings (0.1, 0.2,…,0.9, 0.95) for the A subnetwork. The vigilance setting for the B subnetwork was fixed at 1.0. This is standard for classification problems, since binary coded class labels are used as inputs to the B subnetwork. Finally, a Bayesian classifier was applied to the testing data and performance was calculated, giving a basis for comparison.


(a) Offset Normal Distributions

(b) Offset Normal Distributions

Figure 3. Offset Normal Distributions: (a) the average and standard deviation for the number of correctly classified testing samples out of 1000 as a function of rho for the A subnetwork; (b) the average and standard deviation for the number of A unit templates as a function of rho.


(a) Overlapping Normal Distributions

(b) Overlapping Normal Distributions

Figure 4. Aligned Normal Distributions: (a) the average and standard deviation for the number of correctly classified testing samples out of 1000 as a function of rho for the A subnetwork; (b) the average and standard deviation for the number of A unit templates as a function of rho.

5.2 Results

The averages and standard deviations for the number of correctly classified testing data set members are given for the three problems as a function of A subnetwork ρ value in Figures 2a, 3a, and 4a. The averages and standard deviations for the number of A subnetwork templates are give in Figures 2b, 3b, and 4b. Table 1 gives a summary of the LAPART-2 performance results, including Bayesian performance for comparison.

Table 1. Summary of Performance Results for A subnetwork Rho=0.1. The numbers in parentheses are the standard deviations resulting from the averaging of 20 different orderings of the training data set. "Performance" measures the average percentage correct classification on the independent testing data sets.

A Rho = 0.1	Problem 1 (Rect)	Problem 2 (Norm)	Problem 3 (Norm)
Training Epochs	1.4 (0.5)	1.3 (0.45)	1.8 (0.40)
# A Templates	260 (10)	200 (40)	335 (50)
Bayesian Perf	~75%	~84%	~73%
LAPART Perf	75% (1%)	81% (4%)	65% (5%)

6 Discussion

The testing data set performance of LAPART-2 closely matches that of a Bayesian classifier for each of the three problems. A lower average accuracy is to be expected, given that we are applying a nonstatistical method to a problem defined in terms of statistical information. Note that the performance varies very little with respect to the A subnetwork vigilance (ρ) over wide ranges of the parameter, and that performance is generally better at lower values. This is partially due to the larger maximum hyperbox size allowed by smaller ρ values, resulting in greater loss of binary 1s in the template patterns formed using complement-coded stack input patterns [2].

Note also that convergence occurs on the average in less than two epochs, as predicted by the Two-Pass Learning theorem stated in a previous section. In many cases, only a single epoch was required to perfectly learn the training data.

One important question deals with the ratio of the number of learned A subnetwork templates to the total number of training samples. If this ratio is near 1, it would indicate a high degree of pattern memorization. This is usually a predictor of poor generalization performance. However, LAPART-2 demonstrated a ratio of around 0.25. This indicates that very little memorization is occurring, consistent with the good testing performance data. Some memorization is to be expected given the propensity of LAPART-2 to create templates accessed by only one training pattern [11]. Note that because of the use of stack input representations, a “point hyperbox” is not really a point – it codes a small region of feature space within a stack interval.

7 Conclusion

LAPART-2 has a distinct advantage over LAPART-1 that stems from the modification that forces learning to occur in response to each lateral reset. We have stated that a LAPART-2 network, given the assumptions upon the input data that we described, converges in two passes through a set of training data, with the pattern pairs arbitrarily ordered on each pass. The convergence bound for ARTMAP is greater, varying with the size *mA1* of the binary input space for subnetwork *A* and with its vigilance value ρB [4]. Finally, in [11], we proved that template proliferation does not occur despite the requirement that a new subnetwork A class be initialized with each lateral reset.

Our results are especially significant in that they apply to rule extraction with a network that partitions its input and output spaces (*A* and *B*) into classes, as opposed to simple class labeling. Thus, each subnetwork *A* input can be associated with many subnetwork *B* inputs. This allows for the learning of rules as class-to-class inference relationships as well as inferencing under uncertainty.

The numerical studies presented in this chapter demonstrate that LAPART-2 has one of the tightest bounds known on learning convergence. Additionally, they provide empirical evidence that this need not compromise generalization performance. These results have many implications for the utility of this architecture in future application domains.

Acknowledgements

We would like to acknowledge the support of the University of New Mexico Albuquerque High Performance Computing Center, the University of New Mexico Department of Electrical and Computer Engineering, and The Boeing Company.

References

[1] Healy, M.J., Caudell, T.P., and Smith, S.D.G. (1993), "A neural architecture for pattern sequence verification through inferencing," *IEEE Transactions on Neural Networks*, Vol. 4, No. 1, pp. 9-20, January.

[2] Healy, M.J. and Caudell, T.P. (1997), "Acquiring rule sets as a product of learning in a logical neural architecture," *IEEE Trans. on Neural Networks*, Vol. 8, pp. 461-475.

[3] Caudell, T.P. and Healy, M.J. (1996), "Studies of inference rule creation using LAPART," presented at the IEEE Conference on Neural Networks, Washington, D.C., (ICNN`96). Published in the *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems, (FUZZ-IEEE)*, New Orleans, LA, pp. ICNN 1-6.

[4] Georgiopoulos, M., Huang, J., and Heileman, G.L. (1994), "Properties of learning in ARTMAP," *Neural Networks*, Vol. 7, No. 3, pp. 495-506.

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

[6] Healy, M.J. and Caudell, T.P. (1993), "Discrete stack internal representations and fuzzy ART1," *Proceedings of the INNS World Congress on Neural Networks*, Portland, Vol. II, pp. 82-91, July.

[7] Healy, M.J. (1993), "On the semantics of neural networks," in Caudell, T.P. (Ed.), *Adaptive Neural Systems: The 1992 IR\&D Technical Report*, Technical Report BCS-CS-ACS-93-008, available from the author c/o The Boeing Company, PO Box 3707, 7L-66, Seattle, WA, 98124-2207.

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

[9] Healy, M.J. (1999), "A topological semantics for rule extraction with neural networks," *Connection Science*, vol. 11, no. 1, pp. 91-113.

[10] Georgiopoulos, M., Heileman, G.L., and Huang, J. (1991), "Properties of learning related to pattern diversity in ART1," *Neural Networks*, Vol. 4, pp. 751-757.

[11] Healy, M.J. and Caudell, T.P. (1998), "Guaranteed two-pass convergence for supervised and inferential learning," *IEEE Trans. of Neural Networks*, Vol. 9, pp. 195-204.

[12] Caudell, T.P. and Healy, M.J. (1999), "Studies of generalizations for the LAPART-2 architecture," *Proceedings of the IJCNN*.

[13] Heilman, G.L., Georgiopoulos, M., Healy, M.J., and Verzi, S.J. (1997), "The generalization capabilities of ARTMAP," *Proceedings of the IJCNN*, Houston, TX.