VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Department of Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 471 5328
Fax: +64 4 495 5232
Internet: Tech.Reports@comp.vuw.ac.nz

ART-based Modular Networks for Incremental Learning of Hierarchical Clusterings

Guszti Bartfai and Roger White

Technical Report CS-TR-96/11
August 1996

Abstract

This paper introduces HART-S, a new modular neural network that can incrementally learn stable hierarchical clusterings of arbitrary sequences of input patterns by self-organisation. The network is a cascade of Adaptive Resonance Theory (ART) modules, in which each module learns to cluster the *differences* between the input pattern and the selected category prototype at the previous module. Input patterns are first classified into a few broad categories, and successive ART modules find increasingly specific categories until a threshold is reached, the level of which can be controlled by a global parameter called "resolution". The network thus essentially implements a divisive (or splitting) hierarchical clustering algorithm: hence the name HART-S (for "Hierarchical ART with Splitting"). HART-S is also compared and contrasted to HART-J (for "Hierarchical ART with Joining"), another variant that was proposed earlier by the first author. The network dynamics are specified and some useful properties of both networks are given and then proven. Experiments were carried out on benchmark datasets to demonstrate the representational and learning capabilities of both networks and to compare the developed clusterings with those of two classical methods and a conceptual clustering algorithm. A brief survey of related neural network models is also provided.

Keywords: Modular networks, Adaptive Resonance Theory, Hierarchical ART, Self-organization, Hierarchical clustering, Mushroom dataset.

Publishing Information

This paper will appear in a Special Issue of *Connection Science* on "Combining Artificial Neural Networks".

Author Information

Guszti Bartfai is a lecturer at the Department of Computer Science. Roger White is working on his MSc thesis under the supervision of Guszti Bartfai. The authors can be contacted through the department or directly by e-mail at the `guszti@comp.vuw.ac.nz` address.

1 Introduction

The combination of neural networks has gained increasing attention in recent years. Although some monolithic "black-box like" networks, like the multi-layer perceptron, have been shown to be universal function approximators and used in many practical applications, they have some well-known disadvantages, the most severe of those is intractable training time. One way to alleviate this problem is to divide the task into smaller ones and train the individual — and proportionally smaller — networks on each sub-task separately. This modular approach appears to be supported by findings in brain study which suggest that the human brain is not just one big mass of neurons. For example, neurons of the neocortex, where most high-level cognitive processes occur, are of few types, and their connections are organised in relatively few schemes or basic circuits (Shepherd, 1974). The primate visual system has also been found to be highly modularised with extensively interconnected subdivisions performing linear and non-linear filtering, object recognition and so on (Van Essen et al., 1992). The arguments for decomposition have engineering, complexity, and psychological aspects as well (Hrycej, 1992). There are also a number of theoretical advantages: coding can be sparse; scalability of solutions; modularity of solution leading to self-similar fractal-like architectures (Happel and Murre, 1994). Consequently, modular neural network models have been proposed, (see for example (Hecht-Nielsen, 1987; Moody and Darken, 1989)) which possess distinctive advantages over monolithic ones like faster training (LeCun et al., 1990) and better overall performance (Jordan and Jacobs, 1993).

Besides smaller computational costs, modular networks can also be used to solve more complex tasks that simply cannot be done (i.e. represented and learned) by a single network module. One such area is unsupervised clustering, which is the main focus of our paper. The ability to learn about the environment without a teacher has long been considered an important characteristic of intelligent systems. Unsupervised learning can be found both at sensory-level of mammals, and at higher, cognitive levels of humans. Therefore, it has been an important topic in neural network research (e.g. (Kohonen, 1982), (Oja, 1989), (Rumelhart et al., 1986)) in recent years. Unsupervised learning networks typically perform dimensionality reduction or pattern clustering. In the latter case, upon presentation of an input pattern, the node whose connection weight vector is the closest to the current input vector (in some distance metrics) will become the only active node (winner-take-all competition), and will be allowed to learn, i.e. modify its connection weights. After repeated exposure to the input environment, the network will store a prototypical element of each category, or class, in its connections weights. The number and size of classes the network finds on a given training set depends on the number of output nodes. The more neurons the network has, the larger number of more specific categories it will find. Because of its "winner-take-all" characteristics, the network will only be able to find *one* category for a given input pattern, regardless of how large the network is. However, environments are often more complex and may exhibit a hierarchical structure (e.g. taxonomy of animals), which humans can learn without apparent difficulty. It is therefore reasonable to assume that network circuitry which performs hierarchical clustering is present in the brain. An example for such a network is discussed in (Ambros-Ingerson et al., 1990), where it was suggested that the paleocortex performs hierarchical clustering. The authors also hypothesise that "networks and circuitry of similar design in other brain regions contain computational elements sufficient to construct perceptual hierarchies for use in recognizing environmental cues", and suggest the "possibility that this biologically generated mechanism for hierarchical clustering may be a routine part of perceptual recognition memory behavior in animals and humans".

It is, therefore, worthwhile investigating neural network architectures that can learn class hierarchies. Networks that contain multiple competitive layers appear to be suitable candidates. For example, Rumelhart et al. (Rumelhart et al., 1986) suggest that a hierarchical structure can be formed if a network has several "inhibitory clusters" in each layer that work in parallel, and each of them receives input from the output of the previous layer only (which is a vector with the winners in each active cluster). This method was shown to have some advantages on an artificial example, but it is not clear — due to its inherently probabilistic nature — whether it can be used successfully in other, possibly more complex tasks. Standard competitive learning networks also have several drawbacks like long training time, finding and getting stuck in suboptimal solutions, and instability against changes in the environment.

An alternative model that is based on solid neurophysiological and cognitive principles, and also overcomes the above problems, is the Adaptive Resonance Theory (ART) neural network (Carpenter and Grossberg, 1987a). They are more suitable for the systematic construction of modular networks because of their well-defined interfaces as well as features that most other networks lack. In specific, ART networks have the ability to create new output nodes (i.e. categories) dynamically, and do not suffer from the problem of forgetting previously learned categories if the environment changes. In other words, it provides a solution to the stability-plasticity dilemma (Carpenter and Grossberg, 1988). A neural network with similar functionalities (CALM, for "CAtegorizing and Learning Module") was proposed as a computational primitive in more complex, modular networks (Happel and Murre, 1994). Despite their useful properties, however, ART networks by themselves can only learn input categories at a given level of specificity (i.e. partition the input space), which depends on a global parameter called *vigilance*.

The central idea of this paper is to combine ART networks in a systematic way into a modular architecture which is capable of learning *class hierarchies* from arbitrary sequences of input patters, while retaining the main properties of single ART networks, namely autonomous, fast and stable learning by self-organisation. In specific, we introduce a new ART-based modular network that develops cluster hierarchies by first creating large, general classes, which are subsequently refined in lower layers by splitting up each super-class into sub-classes (by clustering features that are different from the super-class). The biological plausibility of such mechanism was suggested in (Ambros-Ingerson et al., 1990). We call this "splitting" — a term used in statistical clustering after (Hartigan, 1975) — hierarchical ART network, or HART-S (for "Hierarchical ART with Splitting"). In particular, we contrast its functionality and properties to those of the "joining" hierarchical ART network (HART-J), which was proposed independently by Bartfai (Bartfai, 1995b) and Ishihara (Ishihara et al., 1995) under the names *HART* and *arboART*, respectively. (We discuss some related works in more detail in section 5.) The HART-J network has several layers of ART modules where each layer learns to cluster the *prototypes* the previous layer has developed. The network therefore builds successively larger classes in upper layers by joining the lower level, more specific, ones.

Section 2 describes the ART network and summarises its properties. The hierarchical ART models are introduced in section 3. The joining hierarchical ART model (HART-J) and its properties are discussed in sections 3.1 and 3.1.2, respectively. The splitting hierarchical ART model (HART-S) and its properties are discussed in sections 3.2 and 3.2.2, respectively. The experiments that were carried out on a machine learning database are shown in section 4, and the results are discussed in section 5. Finally, conclusions are drawn in section 6.

2 The ART network

This section describes the ART architecture at a level of detail that is necessary for understanding the rest of the paper. Unless otherwise stated, we refer to ART1, the binary version of ART (Carpenter and Grossberg, 1987a)[1].

Adaptive Resonance Theory (ART) architectures are neural networks that develop stable recognition codes by self-organisation in response to arbitrary sequences of input patterns. They were designed to solve the "stability-plasticity dilemma" that every intelligent machine learning system has to face: how to keep learning from new events without forgetting previously learned information.

An ART network is built up of three layers: the *input* layer (F0), the *comparison* layer (F1) and the *recognition* layer (F2) with M, M and N neurons, respectively (see figure 1). The input layer stores the input pattern, and each neuron in the input layer is connected to its corresponding node in the comparison layer via one-to-one, non-modifiable links. Nodes in the F2 layer represent input categories. The F1 and F2 layers interact with each other through weighted bottom-up and top-down connections that are modified when the network learns. The G1, G2 and R signals control the operation of the network: the non-specific *gain control* signals G1 and G2 are used to enable/disable all the nodes in the F1 and F2 layers, respectively. The main role of G1 is to implement template matching in the F1 layer (see equation (2) below, where use of the *reset signal* R is also described). The learning process of the network can be described as follows.

At each presentation of a non-zero binary input pattern $\mathbf{x}$ ($x_i \in \{0,1\},\ \ i = 1, 2, ..., M$), the network attempts to classify it into one of its existing categories based on its similarity to the stored prototype of each category node. More precisely, for each node i in the F2 layer, the bottom-up activation T_j is calculated, which can be expressed as

$$T_j = \frac{|\mathbf{w}_j \cap \mathbf{x}|}{\beta + |\mathbf{w}_j|} \quad j = 1, \ldots, N \tag{1}$$

where $|\cdot|$ is the norm operator ($|\mathbf{x}| \equiv \sum_{i=1}^{M} x_i$), $\mathbf{w}_j$ is the (binary) weight vector [2] (or prototype) of category j, and $\beta > 0$ is the *choice parameter* (Carpenter et al., 1991a). Then the F2 node J that has the highest bottom-up activation, i.e. $T_J = \max\{T_j | j = 1, ..., N\}$, is selected (winner-take-all competition). The weight vector of the winning node ($\mathbf{w}_J$) will then be compared to the current input at the comparison layer. If they are similar enough, i.e. they satisfy the

$$\frac{|\mathbf{w}_J \cap \mathbf{x}|}{|\mathbf{x}|} \geq \rho \tag{2}$$

matching condition, where ρ is a system parameter called *vigilance* ($\rho \in [0,1]$), F2 node J will capture the current input and the network learns by modifying $\mathbf{w}_J$:

$$\mathbf{w}_J^{new} = \eta(\mathbf{w}_J^{old} \cap \mathbf{x}) + (1-\eta)\mathbf{w}_J^{old} \tag{3}$$

where η is the learning rate ($\eta \in [0,1]$). All other weights in the network remain unchanged.

If, however, the stored prototype $\mathbf{w}_J$ does not match the input sufficiently, i.e. condition (2) is not met, the winning F2 node will be reset (by activating the reset signal R in figure 1) for the period of presentation of the current input. Then another F2 node (or category) is selected with the highest T_j whose prototype will be matched against the input, and so on. This "hypothesis-testing" cycle is repeated until the network either finds a stored

[1] There are ART networks that accept both continuous and binary inputs (Carpenter and Grossberg, 1987b; Carpenter et al., 1991b).

[2] Since the bottom-up weights are essentially normalised versions of the corresponding (binary) top-down ones, subsuming them into a single weight vector — and modifying the network equations accordingly — is a commonly used simplification, which we shall adopt throughout the paper. This also assumes the network is in fast learning mode ($\eta = 1$, see below).

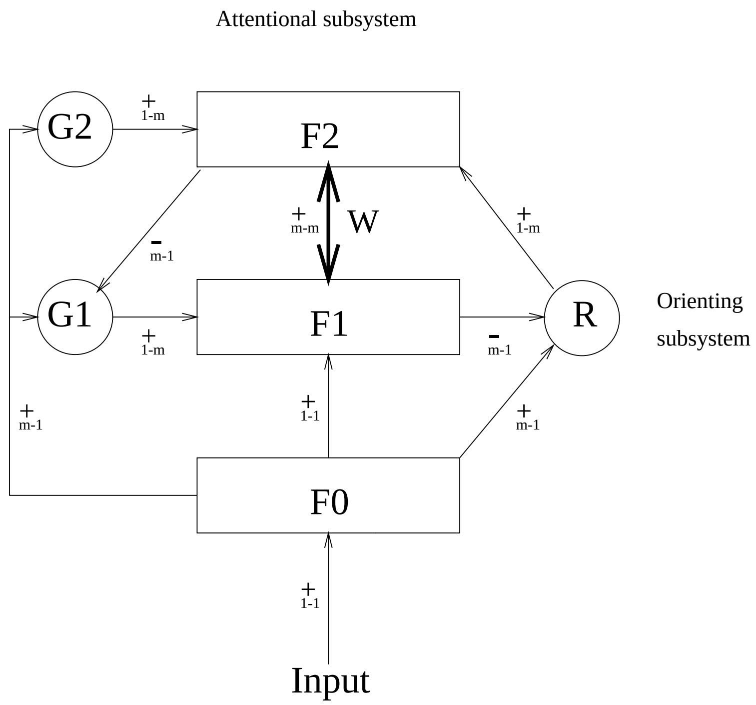

Figure 1: Architecture of the ART network. Thick arrows denote modifiable connections. The "+" and "−" signs represent excitatory and inhibitory connections, respectively. The symbols $\{1, m\} - \{1, m\}$ represent {one,many}-to-{one,many} connections. Winning F2 category nodes are selected by the *attentional* subsystem. Category search is controlled by the *orienting* subsystem. If the degree of category match at the F1 layer is lower than the vigilance level ρ, the non-specific reset signal R will be triggered, which will deactivate the winning F2 node for the period of presentation of the current input.

category whose prototype matches the input well enough, or allocates a new F2 node. Then learning takes place according to (3).

It is important to note that once a category is found, the comparison layer (F1) holds $\mathbf{w}_J \cap \mathbf{x}$ until the current input is removed. It can be shown that after an initial period of self-stabilisation, the network will directly (i.e. without search) access the prototype of one of the categories it has found in a given training set.

Note also, however, that, as a consequence of its stability-plasticity property, the network is capable of learning "on-line", i.e. refining its learned categories in response to a stream of input patterns (as opposed to being trained "off-line" on a finite training set).

The set of attribute values described by ART cluster prototypes correspond to the intersection of all those attribute values found in instances of that cluster. In terms of the clustering algorithm this is not as restrictive as it first seems. A common technique used with ART is the complement coding of input patterns (Carpenter et al., 1991a). The result of using this together with 1-of-n encoded attribute values is that one half of a cluster prototype corresponds to the intersection of the attribute values that all instances in that cluster do *not* have, or from another angle, the union of the attribute values present in that cluster. In total then a cluster prototype specifies a range of values for each attribute and has an effect which, combined with Fuzzy ART (Carpenter et al., 1991b), adds significantly to the clustering capability of the mechanism.

The number of developed categories can be controlled by setting ρ: the higher the vigilance level, the larger number of more specific categories will be created. (If $\rho = 1$, the network will create a new category for every unique input.)

In order to get two or more "views" of the training set with the same network, one can consider choosing low vigilance first to get an overall view, and then raise it at a later phase during training to gain a more detailed view as well. This, however, will result in the network not being able to access the overall view (i.e. coarser categorisation) anymore, but retaining only the more specific categories, even if vigilance is lowered again later. This is because only one category is selected for a given input, and if two categories ($\mathbf{w}_l$, $\mathbf{w}_h$) were found to match the current input perfectly, i.e.

$$|\mathbf{w}_l \cap \mathbf{x}| = |\mathbf{w}_l|, \quad \text{and} \quad |\mathbf{w}_h \cap \mathbf{x}| = |\mathbf{w}_h| \ ,$$

the more specific one (i.e. of the larger norm) will be preferred due to (1). There is no relationship between any pairs of category prototypes except that they are in the same network and compete with each other. The network, therefore, with its single layer of category nodes, is not capable of representing (and thus learning) a hierarchy of classes. This motivated us to design networks with multiple ART modules.

3 Hierarchical ART networks

This section discusses two ART-based architectures that can perform hierarchical clustering by incremental learning. Both of them are essentially a cascade of ART modules with the input pattern entering at one end. Information processed by an ART module is passed on to the next one (see figure 2). At this level, it is basically the kind of information that is passed on that distinguishes the two models.

The "joining" HART network (HART-J) first clusters input patterns into (potentially many) specific categories (with high vigilance level in the first module), and then successive layers cluster the *prototypes* that are developed at the previous layer. This architecture was first presented in (Bartfai, 1995b). Here we describe its main features as well as some further properties, some of which are particularly useful when comparing them to those of the

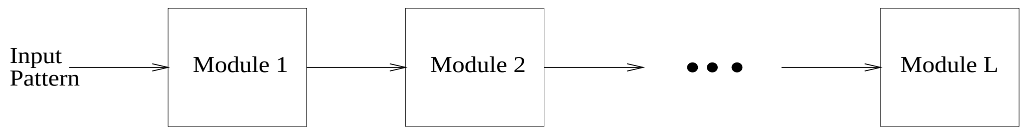

Figure 2: Overall view of the architecture of a Hierarchical ART network. Each module is an ART network.

"splitting" HART network. The HART-S network first creates a few, typically broad, clusters. Then successive layers learn to cluster the *differences* between a prototype and the input vector at the previous layer.

3.1 The joining hierarchical ART network (HART-J)

This model is based on the idea of starting with the most specific clustering of the input domain and progressively merging such clusters into more general ones. Only the similarity between an input to an ART module and the prototype of the category of that input is propagated to the next ART module. The input signal to the first module is the similarity between the input pattern and its most specific categorisation (itself), so is equivalent to the input pattern itself. This similarity is exactly the content of the F1 layer during resonance, so the structure of HART-J reflects this with two connections between successive ART modules (figure 3). One connection propagates inhibitory signals from global reset to the gain controls of a second module so that it cannot become active until a resonant state is obtained in the first [3]. The other connection propagates the contents of the F1 layer of the first module to the F0 layer of the second.

More formally, the HART-J network is described by the following parameter tuple:

$$\mathcal{HJ} = \{L, \mathbf{M}, \mathbf{N}, \boldsymbol{\beta}, \boldsymbol{\rho}, \boldsymbol{\eta}\}$$

where the meaning of each parameter is as follows:

$$\begin{aligned}
L &: \text{ the number of ART modules used} \\
\mathbf{M} = (M_1, \ldots, M_L) &: \text{ the size of each F0 and F1 layer} \\
\mathbf{N} = (N_1, \ldots, N_L) &: \text{ the size of each F2 layer} \\
\boldsymbol{\beta} = (\beta_1, \ldots, \beta_L) &: \text{ the choice parameter of each ART module } (0 < \beta_k, \text{ typically } \beta_k \ll 1) \\
\boldsymbol{\rho} = (\rho_1, \ldots, \rho_L) &: \text{ the vigilance level of each ART module, } \rho_k \in [0, 1] \\
\boldsymbol{\eta} = (\eta_1, \ldots, \eta_L) &: \text{ the learning rate of each ART module, } \eta_k \in [0, 1]
\end{aligned}$$

with the architectural constraint of $M_k = M$ for $k = 1, \ldots, L$. For further simplicity, however, we will also assume $N_k = N$, $\beta_k = \beta$ and $\eta_k = \eta = 1$ (i.e., fast learning) for $k = 1, \ldots, L$. So it is only the ρ_k vigilance levels that are allowed to be different in each ART module.

From figure 3 and noting the contents of the F1 layer in resonance (see section 2), the local input to ART module k can be expressed as

[3] In real-time implementations, this will involve specifying the dynamics of these connections so as to filter out undesirable "spikes" of the reset signal during category search.

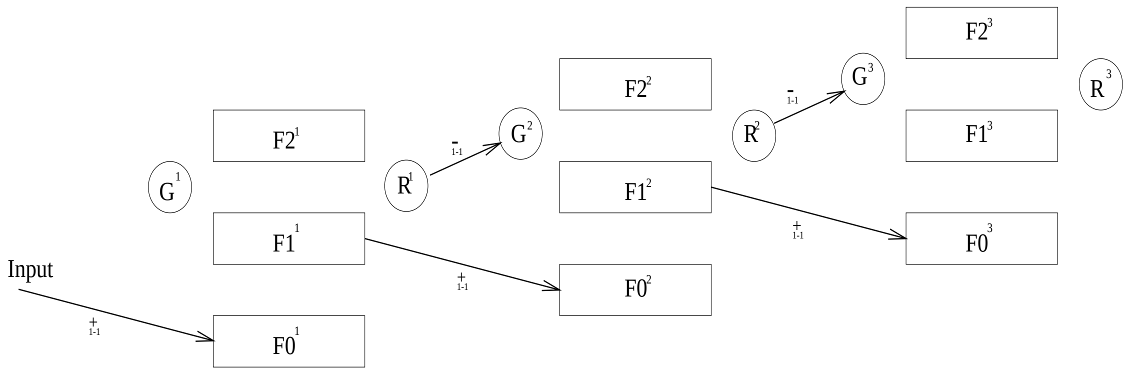

Figure 3: Architecture of a three-layer HART-J network. Connections within the ART modules are not shown. The G1 and G2 ART signals are represented with one G signal for each module ($G^i, i = 1, 2, 3$). The meanings of the "+", "−" and "$1-1$" symbols are explained at figure 1.

$$\mathbf{x}_k = \mathbf{x}_{k-1} \cap \mathbf{w}_{k-1:J}, \quad k = 2, \ldots, L \tag{4}$$

where $_{k-1:J}$ is the index of the winning F2 node in ART module $k-1$. The prototype (or weight vector) of the winning category $_{k:J}$ in layer k after training (see equation (3) if $\eta = 1$) becomes

$$\mathbf{w}_{k:J} = \mathbf{x}_k \cap \mathbf{w}_{k:J}, \quad k = 1, \ldots, L\ . \tag{5}$$

Applying (4) and (5) to layers $k-1, k-2, \ldots, 2$ recursively (and ignoring the indices of winning F2 nodes for simplicity), we get

$$\mathbf{w}_k = \mathbf{x}_1 \cap \mathbf{w}_1 \cap \ldots \cap \mathbf{w}_k, \quad k = 1, \ldots, L \tag{6}$$

where $\mathbf{x}_1$ is the *local* input to the first ART module, which is also the *global* input $\mathbf{X}$ to the entire HART-J network. Equation (6) can then be written as

$$\mathbf{w}_k = \mathbf{X} \cap \mathbf{w}_1 \cap \ldots \cap \mathbf{w}_k = \mathbf{X} \cap \mathbf{W}_k, \quad k = 1, \ldots, L \tag{7}$$

where $\mathbf{W}_k$ (or more precisely $\mathbf{W}_{k:J}$) is the *global* (or *hierarchical*) prototype of level k, i.e., the prototype of category J in level k *with respect to the global* input $\mathbf{X}$. From (4) and (5), it can be seen that

$$\mathbf{w}_k = \mathbf{w}_{k-1} \cap \mathbf{w}_k\ , \tag{8}$$

which shows that the network develops more general categories at successively higher levels. It also follows from (7) and (8) that

$$\mathbf{W}_k = \mathbf{w}_k, \quad k = 1, \ldots, L\ . \tag{9}$$

Since the HART-J network develops more general hierarchical categories, it is instructive to ask what the *maximum* degree of specificity any level k category can have given a sequence of training patterns of size S and (local) vigilance levels $\rho_1, \ldots, \rho_k$. In a single ART network, this is equivalent to asking what the minimum level of generality of any category will be, which is simply given by the (only) vigilance level ρ since $|\mathbf{w}_J| \geq \rho|\mathbf{x}|(=\rho S)$ [4] for any input $\mathbf{x}$ and winning node J. For a given level in a HART-J network, however, the two limits (i.e. minimum level of generality and maximum level of specificity) are separated since the local inputs to each level will vary due to the generalisation done by previous modules. So in a HART-J network, there will be a region of *effective vigilance* at level k, which provides bounds of the specificity that level can have with respect to the global input $\mathbf{X}$. A discussion on the lower bound of this region can be found in (Bartfai, 1995b). Here we provide the upper bound ($\Omega_{k:max}$), which is of more use for practical considerations. (For example, it allows us to put a bound on the number of ART modules needed in an application by guaranteeing a given level of specificity that level k cannot exceed.) More specifically, in each level we look for the smallest value z such that

$$\frac{|\mathbf{w}_k|}{|\mathbf{x}_k|} < \rho_k \rightarrow \frac{|\mathbf{W}_k|}{|\mathbf{X}|} < z \quad . \tag{10}$$

The value of $\Omega_{k:max}$ is given with proof in section 3.1.2.

3.1.1 Dynamics of HART-J

The dynamics of the HART-J network are closely linked to those of component ART modules, and are described as follows.

For $1 \leq i \leq M$, $1 \leq j \leq N$, and $1 \leq k \leq L$

- *Initialisation*

  The initial values of all the connection weights are

$$w_{k:ji} = 1$$

- *Layer activity vectors*

  The $\mathrm{F0}_k$ layer:

$$\mathbf{x}_k = \mathbf{y1}_{k-1}$$

  The $\mathrm{F1}_k$ layer:

$$\mathbf{y1}_k = \begin{cases} \mathbf{x}_k \cap \mathbf{w}_{k:j} & \text{when } \mathrm{F2}_k \text{ node } j \text{ is active and module } k-1 \text{ is in resonance} \\ \mathbf{x}_k & \text{otherwise} \end{cases}$$

  The $\mathrm{F2}_k$ layer:

$$y2_{k:j} = \begin{cases} 1 & \text{when } \mathrm{F2}_k \text{ node } j \text{ is active and module } k-1 \text{ is in resonance} \\ 0 & \text{otherwise} \end{cases}$$

  Whenever an input is presented, and module $k-1$ is in resonance, all $\mathrm{F2}_k$ nodes are enabled and an initial category choice is made.

[4] Inputs here are assumed to be of constant norm only for the sake of simplicity. The discussion will still hold, with minor modifications, if the inputs are not normalised.

- *Category Choice*

  For any module k, if module $k-1$ is in resonance, choose some j such that $F2_k$ node j is enabled and

  $$\frac{|\mathbf{w}_{k:j} \cap \mathbf{x}_k|}{\beta + |\mathbf{w}_{k:j}|}$$

  is maximal, with the result that $F2_k$ node j becomes active.

- *Category Match*

  In any module k a local category matches a local input when $F2_k$ node j is active, module $k-1$ is in resonance and

  $$\frac{|\mathbf{w}_{k:j} \cap \mathbf{x}_k|}{|\mathbf{x}_k|} \geq \rho_k \qquad (11)$$

  If the match is unsuccessful then $F2_k$ node j is disabled and another category choice is made.

  If the match is successful then learning takes place and the system is stable until the input is removed.

- *Learning*

  When $F2_k$ node j is active and (11) holds

  $$\mathbf{w}_{k:j}^{new} = \eta(\mathbf{w}_{k:j}^{old} \cap \mathbf{x}_k) + (1-\eta)\mathbf{w}_{k:j}^{old}$$

3.1.2 Properties of HART-J

This section describes the main properties of the HART-J network.

1. *Stability of Learning.*

   After a finite number of input presentations of an input set, all the individual ART modules will stabilise if the input is normalised. The following argument explains why this is so.

   Let the norm of the input set be S. Then the smallest size of a local input to module $k > 1$ is $S\prod_{i=1}^{k-1}\rho_i$. The learning of any ART module with normalised inputs will stabilise because such normalisation is a limit on the smallest size of an input (Carpenter et al., 1991b). This lower bound on the size of inputs has the same effect.

2. *Order of Presentation.*

   The order of presentation affects the clustering. This follows naturally from the properties of the individual ART modules (Moore, 1989).

3. *Direct Access.*

   Once learning has stabilised direct access to category prototypes is assured for each ART module. Given an input presentation, each module must wait for resonance in previous modules but no search is required.

4. *An Upper Bound on Effective Vigilance.*

   $$\Omega_{k:max} = \rho_k$$

Proof

In this model $\mathbf{w}_k = \mathbf{W}_k$.

The worst case is when $\mathbf{x}_k$ is as big as possible, in fact in this model it could be up to the same size as $\mathbf{X}$.

So the least z such that

$$\frac{|\mathbf{W}_k|}{|\mathbf{X}|} < \rho_k \rightarrow \frac{|\mathbf{W}_k|}{|\mathbf{X}|} < z$$

for any $\mathbf{X}$ and any clustering is ρ_k. □

3.2 The splitting hierarchical ART network (HART-S)

This new network model we are proposing is based on the idea of forming the most general level of clusters first and then successively splitting those clusters into more specific ones. Only the difference between an input to a particular ART module and the prototype of the category of that input is propagated to the next ART module. The input signal to the first module is the difference between the input pattern and the most general category, so is equivalent to the input pattern itself. Only when such differences are greater than a size determined by a network parameter called resolution (ϵ) is the next ART module activated.

More formally, the HART-S network is described by the following parameter tuple:

$$\mathcal{HS} = \{L, \mathbf{M}, \mathbf{N}, \boldsymbol{\beta}, \boldsymbol{\rho}, \epsilon, \boldsymbol{\eta}\}$$

where the meaning of each parameter is as follows:

$$\begin{aligned} L &: \text{the number of ART modules used} \\ \mathbf{M} = (M_1, \ldots, M_L) &: \text{the size of each F0 and F1 layer} \\ \mathbf{N} = (N_1, \ldots, N_L) &: \text{the size of each F2 layer} \\ \boldsymbol{\beta} = (\beta_1, \ldots, \beta_L) &: \text{the choice parameter of each ART module } (0 < \beta_k, \text{ typically } \beta_k \ll 1) \\ \boldsymbol{\rho} = (\rho_1, \ldots, \rho_L) &: \text{the vigilance level of each ART module, } \rho_k \in [0,1] \\ \epsilon &: \text{resolution parameter, } \epsilon \in [0,1] \\ \boldsymbol{\eta} = (\eta_1, \ldots, \eta_L) &: \text{the learning rate of each ART module, } \eta_k \in [0,1] \end{aligned}$$

with the architectural constraint of $M_k = M$ for $k = 1, \ldots, L$. For further simplicity, however, we will also assume $N_k = N$, $\beta_k = \beta$ and $\eta_k = \eta = 1$ (i.e., fast learning) for $k = 1, \ldots, L$. So it is only the ρ_k vigilance levels that are allowed to be different in each ART module.

The architecture of a three-layer HART-S network is shown in figure 4. There are four sets of connections between each successive module in a HART-S network (figure 4). An adaptive associative connection (V) called hierarchical memory between the F2 layers of successive ART modules provides the links between any cluster and its immediate sub-clusters. When a category is active in the first of a pair of modules, only those categories in the second module that can be sub-categories of the first are allowed to become active. These links are important in this network (which are not needed in HART-J) so the correct hierarchies can be maintained, since the individual ART modules cluster *differences* only. The other inter-ART connections are explained in figure 4.

The local input to level k is

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \mathbf{x}_{k-1} \cap \mathbf{w}_{k-1:J}, \quad k = 2, \ldots, L \tag{12}$$

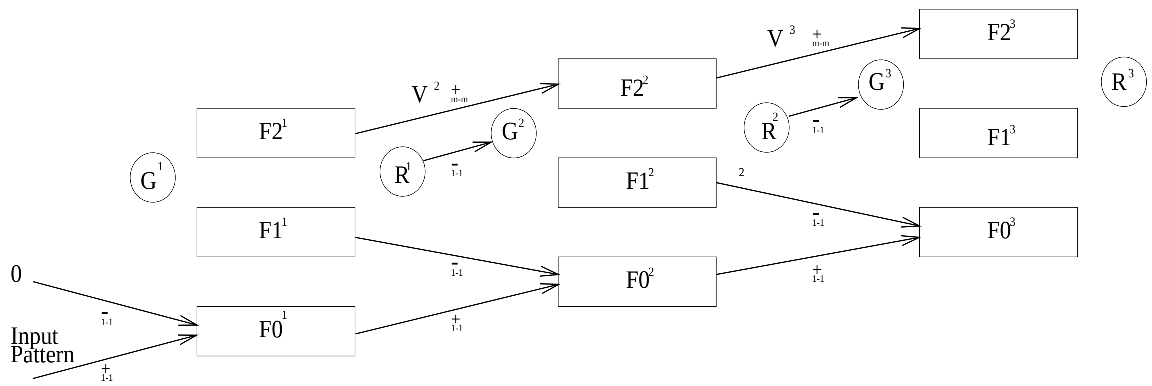

Figure 4: Architecture of a three-layer HART-S network. Only the connections between ART modules are shown. V^2 and V^3 are (many-to-many) associative links between F2 nodes of neighbouring modules. The $R^i \to G^{i+1}$ links have the same purpose as in HART-J. The $F0^i \to F0^{i+1}$ and $F1^i \to F0^{i+1}$ one-to-one links are used to propagate the difference between the current input and the selected category prototype to the next layer.

where ${}_{k-1:J}$ is the index of the winning F2 node in module $k-1$. Since both the inputs and prototypes are binary, (12) can be written as

$$\mathbf{x}_k = \mathbf{x}_{k-1} \cap \overline{\mathbf{w}}_{k-1:J}, \quad k = 2, \ldots, L \tag{13}$$

where $\bar{\cdot}$ denotes the Boolean negation operator. Applying (5) and (13) to layers $k, k-1, \ldots, 2$ recursively (and ignoring the indices of winning F2 nodes for simplicity), for the local input to layer $k+1$ we get

$$\mathbf{x}_{k+1} = \mathbf{x}_1 \cap \overline{\mathbf{w}}_1 \cap \ldots \cap \overline{\mathbf{w}}_k = \mathbf{X} \cap (\overline{\mathbf{w}_1 \cup \ldots \cup \mathbf{w}_k}) = \mathbf{X} \cap \overline{\mathbf{W}}_k, \quad k = 1, \ldots, L \tag{14}$$

where $\mathbf{W}_k$ (or more precisely $\mathbf{W}_{k:J}$) is the *global prototype* of category J at level k:

$$\mathbf{W}_k = \mathbf{w}_1 \cup \ldots \cup \mathbf{w}_k, \quad k = 1, \ldots, L \tag{15}$$

Note that

$$\mathbf{w}_i \cap \mathbf{w}_j = \emptyset, \quad i \neq j \in 1, \ldots, k \; ,$$

which follows from the property that $\mathbf{w}_i = \mathbf{x}_i \cap \mathbf{w}_i$ and $\mathbf{w}_j = \mathbf{x}_j \cap \mathbf{w}_j$ and that $\mathbf{x}_j \cap \mathbf{w}_j = \mathbf{x}_i \cap \overline{\mathbf{w}}_i \cap \overline{\mathbf{w}}_{j-1} \cap \ldots \cap \mathbf{w}_j$ assuming $i < j$.

A distinctive feature of the HART-S network is that the local input $\mathbf{x}_k$ will *not* activate module k unless $|\mathbf{x}_k| > \epsilon S$. Thus ϵ (or resolution) controls the amount of sub-clustering that can occur.

Since the HART-S network develops more specific hierarchical (or global) categories in successive levels, it is useful to ask what the *minimum* degree of specificity any level k category can have given a sequence of training patterns of size S and (local) vigilance levels $\rho_1, \ldots, \rho_k$. For similar reasons to those in HART-J, a single *effective vigilance* level cannot be determined from the individual vigilance levels. Instead, there will be a region, whose lower bound ($\Omega_{k:min}$) will be of interest to us. More specifically, in each level we look for the smallest value z such that

$$\frac{|\mathbf{w}_k|}{|\mathbf{x}_k|} \geq \rho_k \rightarrow \frac{|\mathbf{W}_k|}{|\mathbf{X}|} \geq z \ . \tag{16}$$

The value of $\Omega_{k:min}$ is given with proof in section 3.2.2.

Note that the effective vigilance for different clusters in a given module of a HART-S network may be quite different (and this precludes using effective vigilance as a global parameter in some way). It is an important part of the mechanism since it allows the information found in the input environment to determine the shape of the resulting hierarchy.

3.2.1 Dynamics of HART-S

The dynamics of the HART-S network can be described by those of the component ART modules and the data/memory connections between them as follows.

For $1 \leq i \leq M$, $1 \leq j, h \leq N$, and $1 \leq k \leq L$

- *Initialisation*

  The initial values of all connection weights are:

$$v_{k:jh} = 1$$
$$w_{k:ji} = 1$$

- *Layer activity vectors*

  The F0$_k$ layer:

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \mathbf{y1}_{k-1}$$

  with $\mathbf{x}_1 \equiv \mathbf{x}_0 \equiv \mathbf{X}$ and $y1_{0:i} = 0$.

  The F1$_k$ layer:

$$\mathbf{y1}_k = \begin{cases} \mathbf{x}_k \cap \mathbf{w}_{k:j} & \text{when F2}_k \text{ node } j \text{ is active and module } k-1 \text{ is in resonance} \\ \mathbf{x}_k & \text{otherwise} \end{cases}$$

  The F2$_k$ layer:

$$y2_{k:j} = \begin{cases} 1 & \text{when F2}_k \text{ node } j \text{ is active and module } k-1 \text{ is in resonance} \\ 0 & \text{otherwise} \end{cases}$$

  Whenever an input is presented, module $k-1$ is in resonance, and $|\mathbf{y1}_{k-1}| > \epsilon S$, then any F2$_k$ node j, such that F2$_{k-1}$ node h is active and $v_{k:jh} = 1$, is enabled and an initial category choice is made.

- *Category Choice*

  When module $k-1$ is in resonance, choose j such that F2$_k$ node j is enabled and

$$\frac{|\mathbf{w}_{k:j} \cap \mathbf{x}_k|}{\beta + |\mathbf{w}_{k:j}|} \tag{17}$$

  is maximal, with the result that F2$_k$ node j becomes active.

- *Category Match*

  A category matches an input when $F2_k$ node j is active, module $k-1$ is in resonance and

$$\frac{|\mathbf{w}_{k:j} \cap \mathbf{x}_k|}{|\mathbf{x}_k|} \geq \rho_k \tag{18}$$

  If the match is unsuccessful then $F2_k$ node j is disabled and another category choice is made.

  If the match is successful then learning takes place and the module is stable until the input is removed.

- *Learning*

  When $F2_k$ node j is active, and (18) holds then

$$\mathbf{w}_{k:j}^{new} = \eta(\mathbf{w}_{k:j}^{old} \cap \mathbf{x}_k) + (1-\eta)\mathbf{w}_{k:j}^{old}$$

  (adjustment of category prototype) and

$$\mathbf{v}_{k:j}^{new} = \mathbf{y2}_{k-1}$$

  whereby the association between a category and its chosen sub-category is created.

3.2.2 Properties of HART-S

This section describes the main properties of the HART-S network. Here, we are making the further assumption that input patterns are normalised.

1. *Stability of Learning.*

   As with the HART-J network, all we need to do is show there is a lower bound on the size of local inputs for every ART module. There is such a bound since $\epsilon > 0$ and a module will only be activated when an input has a size larger than ϵS.

2. *Direct Access.*

   Once learning has stabilised direct access to category prototypes is assured for each ART module. As in HART-J, each module must wait for resonance in the previous module.

3. *Order of Presentation.*

   The order of presentation affects the clustering. Again this follows naturally from the properties of the individual ART modules.

4. *A Bound on the Number of Levels of Clustering.*

   The most levels of clustering possible is B when the following equation holds.

$$B = min\{k : \prod_{i=1}^{k}(1-\rho_i) \leq \epsilon\} \tag{19}$$

   Proof

Let the norm of the inputs be denoted S.

Let the size of the largest input possible to module k be I_k.

$I_1 = S$.

The smallest prototype formed in module k has size $\rho_k\ I_k$.

For $k > 1$

$$I_{k+1} = I_k(1-\rho_k) \ = \ S\prod_{i=1}^{k}(1-\rho_i).$$

The largest input to a possible module $k+1$ that will not activate it has size $S\epsilon$. So when $I_k \le \epsilon$ module $k+1$ can never be activated.□

So $(1-\rho_1) \le \epsilon$ reduces the model to the standard ART model ($\epsilon = 1$ ensures this since $\rho_1 \ge 0$).

5. *Consistent Sub-clustering.*

   If an input pattern that is clustered at a level k of a clustering hierarchy is also clustered at level $k+1$, then all inputs clustered with it at level k will also be clustered at level $k+1$ if it exists.

   The difference in size between a local input and its local prototype in a given module is the size of the input to the next module. So it is sufficient to show that the size of all local inputs clustered together in a given module is the same since then they will all generate the same sized input to the next module. It is the size of this input which determines if the next module will become active and sub-clustering occurs. All inputs clustered together in a given module must be clustered together in any previous modules, so are all of the same size since the inputs to module 1 are normalised.

6. *A Lower Bound on Effective Vigilance.*

$$\Omega_{k:min} = 1 - \prod_{i=1}^{k}(1-\rho_i)$$

Proof

For any $\mathbf{X}$ and any clustering, if the left hand side of (16) is true so must be the right hand side. The worst case is when the size of $\mathbf{W}_k$ is as small as possible.

In this model, the size of $\mathbf{W}_k$ is the sum of the sizes of the local prototypes at level 1 to k, so clearly $\mathbf{W}_k$ is smallest when the size of the local prototypes in levels 1 to k is smallest.

The smallest prototype size at level 1 is $\rho_k|\mathbf{X}|$.

The smallest prototype size at level 2 is $\rho_2(1-\rho_1)|\mathbf{X}|$.

The smallest prototype size at level k is $\rho_k\prod_{i=1}^{k-1}(1-\rho_i)|\mathbf{X}|$.

So the smallest size for $\mathbf{W}_k$ is $(1-\prod_{i=1}^{k}(1-\rho_i))|\mathbf{X}|$.

So the greatest z such that the right hand side of (16) is true, is $(1-\prod_{i=1}^{k}(1-\rho_i))$. □

4 Experimental results

We carried out three sets of experiments with the hierarchical ART networks presented here. The primary goal of each was:

1. To demonstrate the representational capabilities of HART-J and HART-S networks.
2. To explore the joint effect of vigilance (ρ) and resolution (ϵ) of the HART-S network on the depth of the resulting trees and on the training time.
3. To compare the classifications produced by HART networks to those of two classical hierarchical clustering methods and an AI conceptual clustering algorithm.

The details of the experiments and the results are discussed in the following sections.

4.1 Representational capabilities of HART networks

In order to demonstrate that the HART networks are capable of developing class *hierarchies*, we trained and tested the networks on the *Zoo* machine learning benchmark data set (Merz and Murphy, 1996). This data set contains 101 instances of animals described with 18 attributes such as "hair", "aquatic", "domestic" and so on. It is relatively small, but is adequate for the purposes of these experiments, which also allows us to make general observations about the behaviour of these models that will guide further analysis and development.

A number of simulations were run in a Unix/X windows environment using the neural network simulation program PlaNet (Miyata, 1991) and programs written in C language. The binary attributes of the training samples were presented to the network in *complement coding* to assign significance to both the presence and absence of features, and to keep the input vectors normalised (Carpenter et al., 1991a) to avoid proliferation of ART categories (Moore, 1989). The only non-binary attribute ("number of legs") was presented in *generalised complement coding* (Bartfai, 1994). The "type" attribute was ignored to eliminate any bias from the developed clusterings. The 18th attribute ("animal name") was simply used as a label for the individual instances. With this input encoding scheme, instances were presented to the network as 36-element binary vectors with the same norm of 16. In all the experiments, the HART networks were trained "off-line" on a training set containing all 101 instances of the "zoo" database. In each training session, the network was considered to be completely trained when each input pattern directly accessed a matching cluster at each active level of the cluster hierarchy (i.e., no category search or further learning occurred). Of particular interest are the effects of changes in vigilance and order presentation on clusterings and the comparative behaviour of the two models.

Figures 5 and 6 are examples of hierarchies learnt by three-layer HART-J and HART-S networks, respectively. The choice parameters (β_k) and learning rates (η_k) were set to 1×10^{-5} and 1, respectively, in both cases. In general, we can observe that category prototypes are often duplicated in HART-J as prototypes are propagated on whether or not any clustering of them is achieved. Prototypes in HART-S by comparison always differ and furthermore information contained in one prototype is not copied in subcluster prototypes, as occurs in HART-J. A difficulty with HART-S is in predicting the number of F2 nodes needed in lower layers of a network due to the links created in hierarchical memory during learning. A higher setting of the resolution parameter tends to reduce the number of such nodes used to learn a particular set of inputs. Lastly we noted that, as with ART, the clusterings learnt by both HART models were affected significantly by a change in the order of input pattern presentation.

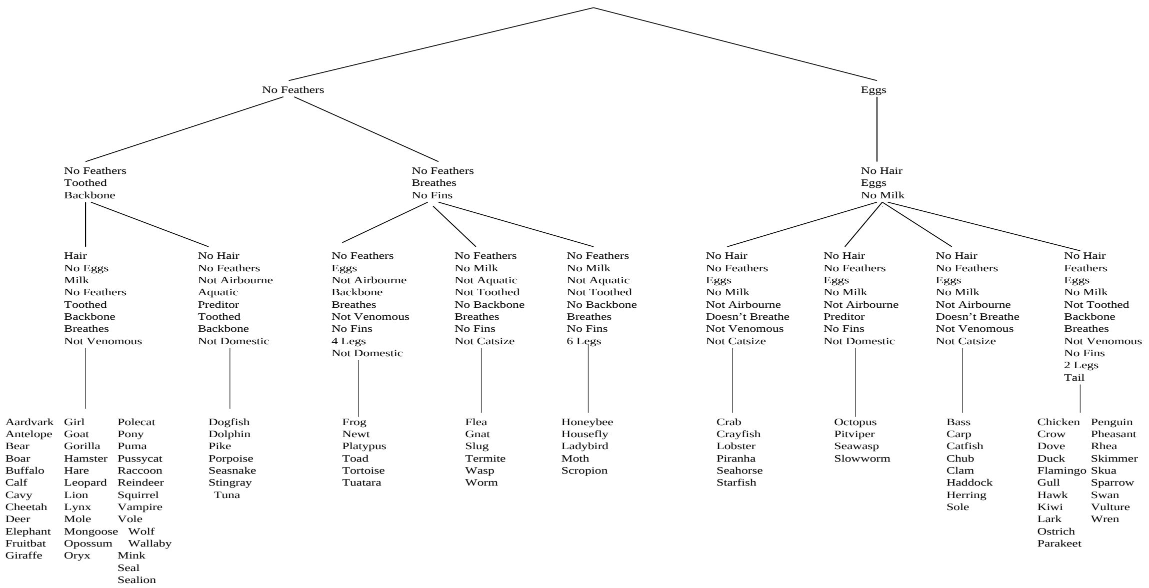

Figure 5: A cluster hierarchy learnt from the *Zoo* database by a three layer HART-J network. The vigilance levels were $\rho_1 = 0.45$, $\rho_2 = 0.3$ and $\rho_3 = 0.05$. Cluster prototypes specify all the features required for cluster membership and thus always increase in size as the hierarchy is descended. The root of the tree is only shown here for clarity, but is not actually represented in the network. In this example, ART modules 1, 2 and 3 developed 2, 3 and 9 categories, respectively. The tree structure "emerges" as input patterns are presented and the corresponding categories are activated at each level. The instances belonging to each leaf are shown here as the lowest layer (which could also have been learned by the network with an ART module inserted before the first one with a vigilance level of 1).

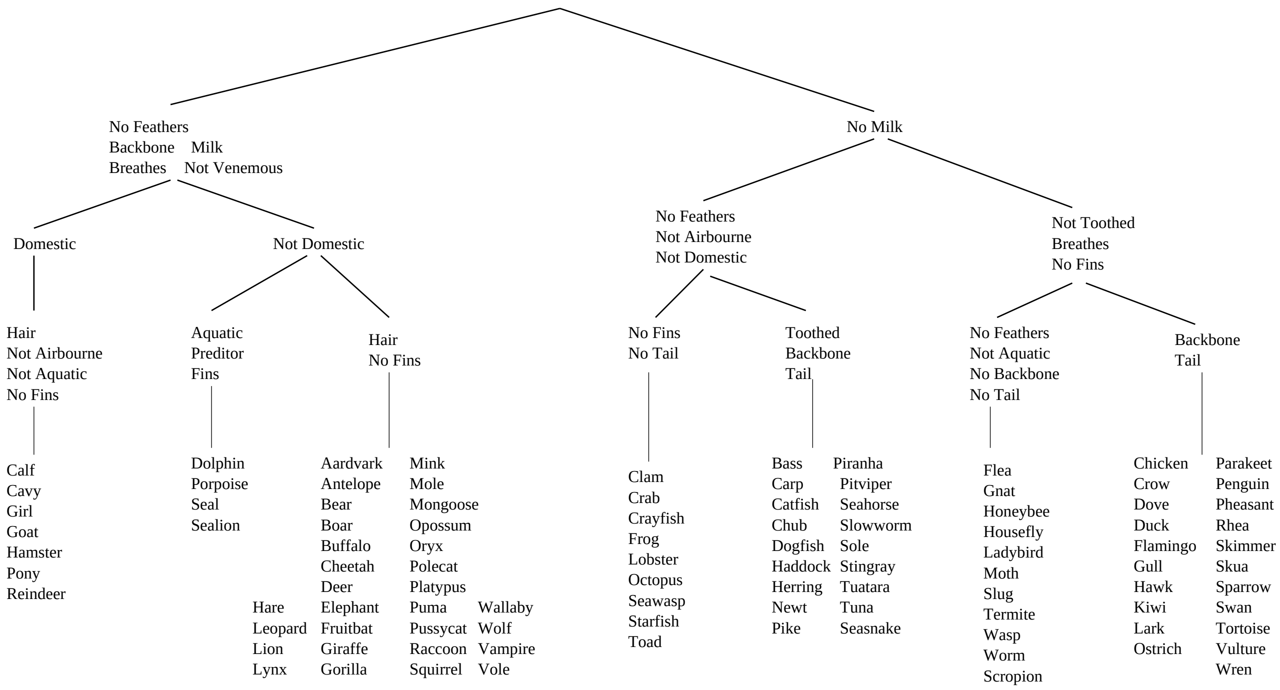

Figure 6: A cluster hierarchy learnt from the *Zoo* database by a three layer HART-S network. The resolution (ϵ) was 0.2 and all three vigilance levels were set to 0.06. Here the hierarchy is explicitly encoded in the network via the $\mathbf{v}_{k:j}$ associative links. Any particular feature is only used to specify a particular branch of the tree. The usage of the root and leaves is the same is in figure 5.

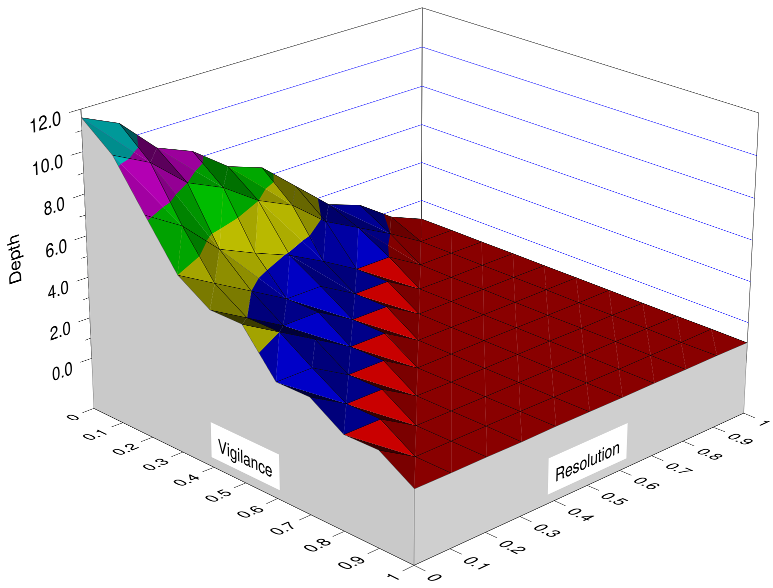

Figure 7: The average depth of HART-S hierarchies (vertical axis) with respect to vigilance and resolution (ρ and ϵ) after training with the *Zoo* database. Each point on the graph is the average of 100 clusterings with different orders of input pattern presentation. The flat area indicating only one level of clustering is formed where the network is constrained to behave as a single ART module. Lower settings of vigilance tend to result in smaller prototypes in any ART modules and thus more levels of clustering. Lower settings of resolution subcluster smaller differences with the same effect.

4.2 The joint effect of vigilance and resolution

This set of experiments explored the combined effect of vigilance (ρ) and resolution (ϵ) in HART-S on the depth of the resulting hierarchy and on required training time. The data set, training conditions and default parameter settings were identical to those used in the previous experiments.

Because of the significant effect of the order of input pattern presentation, each set of simulations consisted of 100 runs each with a different ordering of inputs and the measurements averaged out. The average deviations of these measurements indicate they should be quite reliable guides to the expected behaviour of HART-S. The depth of a hierarchy was taken as the deepest subcluster, and training time was taken as the number of presentations of the set of input patterns that caused any change of local category prototypes. Figures 7 and 8 show the average depth and training time (in epochs), respectively, while each of resolution and vigilance — which was the same for each module — was varied from 0 to 1 in steps of 0.1.

The relationship expressed in equation (19) is clearly reflected in the first figure and in the second it is clear that the training time required to create a cluster hierarchy increases relatively slowly with increasing tree depth. Perhaps the symmetry of graphs suggest a redundant parameter. However, clusterings of the same depth and learnt in the same number of epochs but with significantly different combinations of ϵ and ρ are usually very different.

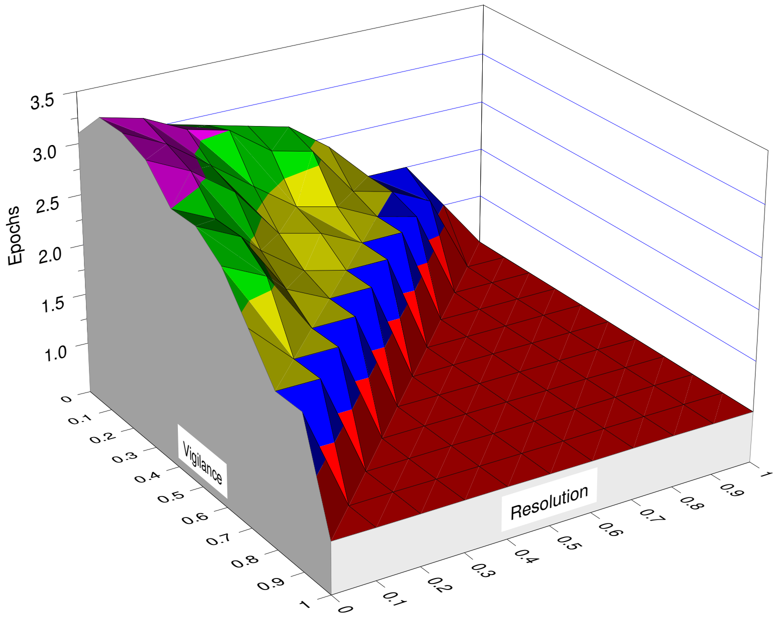

Figure 8: The average training time for HART-S hierarchies with respect to vigilance and resolution (ρ and ϵ) after training on the *Zoo* database. A unit on the vertical axis corresponds to one epoch in which the entire set of input patterns is presented. Each point on the graph is the average of 100 clusterings with different orders of input pattern presentation, these being the same ones as used in figure 7.

4.3 Comparison to other clusterings methods

The primary motivation for designing these hierarchical ART networks was to have a modular network that is capable of fast and stable incremental learning of hierarchical clusterings of arbitrary sequences of input patterns by self-organisation. Having achieved this goal, we can now take a look at these networks from a different angle and view them simply as *implementations* of incremental clustering *algorithms*. The question that arises then is how these algorithms as such compare to other existing clustering methods. We attempted to find an answer to this with this set of experiments. We compared the clusterings gained by the HART-J and HART-S networks to standard statistical clustering methods and an incremental clustering machine learning method. The following two data sets from the UCI Repository (Merz and Murphy, 1996) were used:

- *Soybean*: this data set contains 307 instances [5] described with 35 nominal attributes (and class information, which was ignored here). The attributes were encoded in generalised complement coding (see above) — with missing attribute values represented on a separate bit — yielding 134-element binary vectors of norm 35.
- *Mushroom*: this data set contains 1000 randomly selected examples from the original Mushroom set found in the repository. Each instance is described with 22 nominal features (and class information, which was ignored here). Again, attributes were encoded in generalised complement coding giving 126-element binary vectors (of norm 22) [6].

The two statistical hierarchical clustering methods that were used for comparisons are *compact* clustering (also known as *complete linkage*) and *group average* (Gordon, 1981). Both of them are agglomerative methods. In such methods, the initial configuration involves treating each instance as a cluster of size one. At any subsequent stage, two clusters are amalgamated into one using the two clusters judged closest by some suitable distance measure. Clustering ceases when all observations are combined into one cluster. In case of compact clustering, the distance between two clusters is taken as the maximum of the distances between any point of the first cluster and any point in the second cluster. Group average clustering is similar to compact clustering, differing only in the criterion by which extant clusters are to be fused at any given stage. Here the criterion is the average of the distances between points in the two clusters. The built-in library functions of the S-PLUS statistical package were used here.

While the above clustering methods are non-incremental (i.e. the whole data set has to be available for the computation), several incremental machine learning algorithms have been proposed for unsupervised concept formation. One such method is called *Froggit* [7]. Froggit is an incremental, top-down, hierarchical clustering algorithm derived from COBWEB (Fisher, 1987) and CLASSIT (Gennari et al., 1989). Like COBWEB, it processes new instances, one at a time, incorporating them into a class hierarchy, and attempting to incrementally optimise the hierarchy. Froggit differs from COBWEB and CLASSIT in three significant ways. First, Froggit uses an information theoretic measure of the goodness of a partition in its optimisation search. This measure allows Froggit to handle both numeric and discrete attributes in a well-founded way. The measure is based on a minimum message length approach (Wallace and Boulton, 1968), but is not completely rigorous. Second, the search strategy in the optimisation phase is much richer than COBWEB's. It uses a beam hill-climbing search, and

[5] There are two versions of this data set in the UCI repository: *soybean-small* and *soybean-large*. We used the latter.

[6] This is exactly the same encoding that was used in (Carpenter et al., 1991a).

[7] An earlier version of Froggit, called *DySect*, has been applied successfully to data analysis (Dawkins et al., 1994).

N_c	P_r	Compact		Group		Froggit	
		P_{hs}	P_{hm}	P_{hs}	P_{hm}	P_{hs}	P_{hm}
5	0.200	0.207	0.313	0.898	0.926	0.498*	0.346
10	0.100	0.112	0.236	0.532	0.794	0.474	0.768*
15	0.067	0.094	0.231	0.291	0.703	0.540	0.887
20	0.050	0.067	0.105	0.167	0.657	0.376*	0.501*

Table 1: This table shows estimates of "agreement" probabilities between HART-J and three other clustering methods on the Soybean and Mushroom data sets (P_{hs} and P_{hm} columns, respectively, for each method). The *'s indicate that Froggit could not find exactly N_c categories, in which case the probabilities were taken from a partition having the closest number of categories.

a wider range of hierarchy modification operators than COBWEB. These allow it to smooth the search space and avoid some of the local minima that trap COBWEB. Third, Froggit separates the optimisation from the incorporation, which not only improves the effectiveness of the search, but also allows a wider range of incorporation strategies (Andreae, 1996).

Since the class representation of all of the above methods (including HART) are markedly different, a reasonable way to compare the methods is to look at their partitionings at the instance level (instead of comparing their class descriptions). The developed clusterings of the HART networks, however, are affected by the order of input presentation. Therefore, instead of determining a single similarity value, we can estimate the probability with which categories of HART networks and other methods agree. More precisely, given the same number of categories found, we estimate

$$P_{hd}(c_{ij} = 1 | h_{ij} = 1), \quad i < j, \; i, j = 1, \ldots D$$

where D is the size of the data set d, $c_{ij} = 1$ and $h_{ij} = 1$ if instances i and j are assigned to the same class by a clustering algorithm and a HART network in a given training run, respectively. In other words, P_{hd} is the probability that if instances i and j are found to be in the same category by a HART network, they are also found to be in the same category by a clustering algorithm. The best case is when $P_{hd} = 1$, i.e. the two methods agree 100% of the time. The worst case is complete randomness, or non-correlation between the methods. However, even in this case, there is a finite probability that the random selection of categories will agree with any clustering method. In case of complete random categories, this probability (P_r) is $\frac{1}{N_c}$ where N_c is the number of classes in the given partition. So clearly, $P_r \leq P_{hd} \leq 1$.

Table 1 and 2 show the estimated P_{hd} probabilities for the HART-J and HART-S networks, respectively. The number of classes (N_c) were gained differently in each method. For the statistical clustering methods (Compact and Group), a cut of the tree was taken at the desired level. For Froggit, one of its parameters, which controls the specificity (and therefore the number) of top-level classes in the hierarchy, was varied until the desired number was produced. In HART-J, different number of classes were gained by simply observing the partitionings of different modules in the ART cascade. In HART-S, the appropriate cut of the tree was made through the use of the resolution parameter (ϵ) in a trained network (varying it between 0 and its value at training), which typically resulted in cuts spanning multiple ART modules.

First, the results show that the HART networks produce clusterings that are correlated (at least according to our measure) to the other methods at better (sometimes much better!)

N_c	P_r	Compact P_{hs}	Compact P_{hm}	Group P_{hs}	Group P_{hm}	Froggit P_{hs}	Froggit P_{hm}
5	0.200	0.206	0.313	0.897	0.925	0.532*	0.470
10	0.100	0.111	0.237	0.534	0.791	0.430	0.688*
15	0.067	0.093	0.232	0.304	0.700	0.485	0.751
20	0.050	0.069	0.105	0.173	0.659	0.341*	0.438*

Table 2: This table shows estimates of "agreement" probabilities between HART-S and three other clustering methods on the Soybean and Mushroom data sets. For more explanation, refer to table 1.

	Compact	Group	Froggit
ART	0.231	0.698	0.724
HART-J	0.231	0.703	0.887
HART-S	0.232	0.700	0.751
Froggit	0.232	0.705	1.000

Table 3: The comparison of three ART-based methods and Froggit against the three "base" ones used above. The numbers are the same P_{hd} probabilities used in tables 1 and 2. Here we show results for 15 classes (i.e., $P_r = 0.067$) learned from the Mushroom database.

than chance level. It can also be seen clearly that the numbers for HART-J and HART-S are very similar suggesting that even though the architectures and developed class representations are different, they produce similar clusterings. The similarity between HART and the other is noticeably higher for the Mushroom data set. The Group method and Froggit are apparently more similar to HART than the Compact method. For the two classical clustering methods, the level of agreement tends to increase as N_c decreases.

We also ran some experiments to gauge the similarity of clusterings of single ART networks to those of HART-J, HART-S and Froggit (against the three "base" methods used in the tables above) as well as to see how Froggit compares to the two statistical clustering methods. Table 3 shows the results on the Mushroom database when the number of classes was 15.

A very interesting result is that the HART and ART networks (and Froggit) show remarkably similar levels of agreement with the compact and group average methods. This seems to suggest that the hierarchical ART networks offer their added representational power without any apparent change in their clustering behaviour. Also, the HART networks and Froggit appear to be closer to each other than to the classical methods while they are also slightly closer to Froggit than the ART network. Recall, however, that the aim has been to provide a modular neural network model that inherits the clustering properties of some kind of ART type components and integrates them to perform the task of discovering a set of useful partitions and a hierarchical relationship between them that is both consistent and usable (rather than to find a hierarchical clustering method better than, for example, Froggit).

5 Discussion

In the following, we discuss other neural network models that we consider to be more closely related to the hierarchical ART networks introduced here. We shall focus on identifying similarities and differences.

Lampinen and Oja proposed a model, in which a hierarchy of self-organising maps (SOMs, see (Kohonen, 1982)) was used for visual feature extraction (Lampinen and Oja, 1992). In a MSOM network, there is a separate input vector for each SOM network in the first layer. The input to the second layer is a vector with the *index* of the winning node in each SOM network of the first layer. The authors admit this does not bear any resemblance to real networks, but provides a useful and effective method for feature extraction in their application. This hierarchy is rather different from that of a HART network, both in what input each layer receives and how the layers are interconnected. In particular, the hierarchy in a HART is not encoded explicitly in the network architecture, which emerges (via learned links) as the network undergoes training.

Li et al. proposed a "neural tree" architecture for hierarchical classification and vector quantisation (Li et al., 1993). It implements a splitting (or divisive) mechanism by self-organisation, like HART-S. However, in a neural tree, each node in every layer receives the same (global) input pattern, and the layers and nodes are selected during learning and recall using a global control algorithm. In HART, only the lowest layer (or module 1) receives the original input pattern, and its control is completely autonomous, and emerges as a result of interaction between layers and nodes using only local operations. After training, the neural tree is optimised by pruning and splitting taking ideas from machine learning (e.g. CART (Breiman et al., 1984)). The neural tree was applied to data compression and speech processing.

Despite their usefulness in practical applications, the above networks would not be suitable in autonomous learning tasks because of slow learning and instability against changes in the environment (see for example (Moore, 1989)). Partly motivated by these problems, researchers have looked at hierarchical ART systems in recent years.

The ART3 architecture (Carpenter and Grossberg, 1990) can be considered as the first example of hierarchical ART networks. Carpenter et al. define two modular ART-based models ("ART bidirectional hierarchy" and "ART cascade") for learning hierarchical clusterings, in which there is a close coupling between each ART module mediated by special gate control signals, and each module learns information that is already processed by previous layers. This mechanism is very similar to HART except for one main difference: in ART3, each component network performs *contrast enhancement* (Carpenter and Grossberg, 1987a) in the F2 layer instead of selecting a winner. The output pattern of the *F2 layer* is then used as input to the next ART layer. In contrast, information that is passed on to successive layers in a HART network is based on *prototypes* (either directly or as a difference vector) rather than *distances* of inputs from the category prototypes (like in ART3). Also, because in ART3 each F2 layer is set to produce partially compressed codes via nonlinear differential equations, calculating the output pattern is computationally more expensive than in HART, where categories are accessed directly in a trained network resulting in faster recall of learned prototypes.

An interesting and seemingly closely related ART-based hierarchical model is the "ART tree" (Caudell et al., 1994) that was designed to be embedded in a Neural Information Retrieval System (NIRS) to facilitate the intelligent and fast retrieval of engineering design. The ART tree is a collection of "ART1 macrocircuits" connected in a tree topology. Layers in the tree are trained successively by first breaking up the set of designs into a few general categories, and then training the modules in lower layers on only those patterns that belong to their respective parent class. The main differences between the ART tree and HART are that in an ART tree (i) each macrocircuit is allowed to "see" the entire (global) input pattern, (ii) the hierarchy is explicitly encoded in the network prior to learning in the form of a tree and (iii) learning and recall are under strict global control as part of the NIRS CAD system.

Also, the ART tree — however generic in principle — is application specific (with different layers being selective to given features only), whereas HART does not make any assumptions about the relative importance of input features.

Soon and Tan proposed the concept hierarchy network (CHN, in (Soon and Tan, 1993)) as an ART-based memory model that forms concept hierarchy by learning sample relations between concepts. All concepts are represented in a concept layer. Relations between a concept and its defining lower level concepts are chunked as cognitive codes represented in a coding layer. This network is a good example of how ART networks can be combined to learn more complex tasks, which individual components cannot do by themselves. However, even though the network is modular and learns a hierarchy, these cannot be seen as the main design principle in either the architecture or the learning mechanism. It shows up, for example, in how the CHN is limited to two-level hierarchies whereas in HART the number of levels is variable and depends on the training set and the vigilance level of component ART modules.

The first of our ART-based models (SMART, for "Self-consistent Modular ART") was introduced in (Bartfai, 1994), in which layers in a hierarchy were connected through an associative layer much the same way as the two ART modules are connected in an ARTMAP network (Carpenter et al., 1991a). In the SMART network, each module learned to cluster the original inputs at different levels of vigilance, and self-consistency was maintained via ARTMAP's "match tracking" process. This is a rather different alternative approach to ART-based hierarchical clustering resulting in markedly different developed hierarchies (Bartfai, 1995a).

More recently, Ishihara et al. introduced "arboART" (Ishihara et al., 1995), an ART-based architecture, which is identical in basic principles to our earlier model (HART-J). In other words, both arboART and HART-J develop class hierarchies by clustering the prototypes developed at the lower level. The only significant difference is that arboART is based on the "ART1.5-SSS" network (Ishihara et al., 1993), which is an adaptation of ART1 to analogue patterns for small sample size. The clustering performance of arboART was found to be comparable to that of a the centroid method in statistical clustering. The existence of arboART and HART-J demonstrates well the advantage of such architecture, where different types of ART networks can be "plugged in" to achieve certain goals according to different criteria.

We also note that the HART networks possess certain features of self-similar fractal-like architectures, which are also advantages of modular networks (Happel and Murre, 1994). Namely, ART modules in a HART-J or HART-S network interact via modifiable links as well as fixed (non-specific) control signals, similar to the way layers and individual nodes interact within a single ART network. This self-similarity can be carried even further in the HART-S network, in which training starts with a pool of uncommitted ART modules that are later allocated as needed depending on the training set and a global parameter ("resolution", in section 3), much like individual nodes become committed in a single ART module depending on a within-ART global parameter (vigilance). This opens up a number of possibilities for creating more sophisticated interactions between ART layers to direct its search in the hierarchy and enhance its performance. (Another good example for combining modularity and hierarchy in a self-similar structure is (Jordan and Jacobs, 1992).)

As a final note, the HART-S network essentially implements the same kind of hierarchical search and recall that was discussed in (Ambros-Ingerson et al., 1990). However, the recall there was achieved by temporal sampling of the same input signal (dog's sniffing) until the network found the most specific category that described the input. Because of the reliance of repetitive sampling, that network was able to represent the hierarchy in a single layer (with appropriate settings of network dynamics). So we can view the HART-S network as having explicit spatial representation of the hierarchy, which "compensates" for the lack of repetitive sampling.

6 Conclusions

This paper proposed a modular neural network architecture (HART-S) that is capable of stable and fast learning of hierarchical clusterings of input sequences by dividing initially large classes into smaller, more specific ones. The network is a cascade of Adaptive Resonance Theory modules, each clustering the *differences* between the input to the previous module and its selected category prototype. The class hierarchy is learned via associative links between F2 layers of neighbouring modules. The depth of the hierarchy can be controlled with a global parameter called resolution. We have shown that the network, like the "joining" Hierarchical ART we also discussed here, retains the useful properties of the component ART networks. The experiments demonstrated the representational and learning capabilities of both the HART-J and HART-S networks, and also suggested that the developed clusterings might be closely related those produced by some classical and/or machine learning clustering algorithms. We believe investigating such similarities (and differences too) further could provide us with more insights into the advantages and drawbacks of using ART-based modular networks in real-world problems. This may also allow us to address issues such as extracting rules from trained HART networks, and/or optimising the developed hierarchies. Along the latter line, it may be useful to view HART networks (especially in fast learning mode) as implementing the first "sorting" stage in a multi-stage clustering mechanism (Fisher, 1996).

We also note that as a consequence of the modular architecture of the HART-S and HART-J networks, it is relatively simple to modify them to achieve slightly different functionalities. For example, the network can learn analogue patterns by replacing the ART1 components with ART2 (Carpenter and Grossberg, 1987b) or Fuzzy ART (Carpenter et al., 1991b) modules. A probabilistic hierarchical network based on Bayesian decision principles can be gained by using Gaussian ART (Williamson, 1995) modules.

Finally, we believe the hierarchical networks introduced here are not only good examples of modularity, but can also suggest possible mechanisms for learning complex recognition tasks in real and artificial neural networks. We also believe this model is worth investigating further, which could lead us to a refined model capturing some of the more complex phenomena that occur in both artificial and biological systems.

Acknowledgements

This work was in part supported by Victoria University of Wellington (Research Grant No.RGNT/596/303) as well as the Department of Computer Science. We wish to acknowledge the programming support of David Gilligan. We also thank the reviewers for their comments, which helped us improve the paper.

References

Ambros-Ingerson, J., Granger, R., and Lynch, G. (1990). Simulation of paleocortex performs hierarchical clustering. *Science*, 274:1344–1348.

Andreae, P. (1996). Froggit's goodness measure and search strategy. Personal communication.

Bartfai, G. (1994). Hierarchical Clustering with ART Neural Networks. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 2, pages 940–944. IEEE Press.

Bartfai, G. (1995a). A Comparison of Two ART-based Neural Networks for Hierarchical Clustering. In Kasabov, K. and Coghill, G., editors, *ANNES'95, The Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 83–86. IEEE Computer Society Press.

Bartfai, G. (1995b). An ART-based Modular Architecture for Learning Hierarchical Clusterings. Technical Report CS-TR-95/3, Department of Computer Science, Victoria University of Wellington, New Zealand. Accepted for publication in *Neurocomputing*.

Breiman, L., Friedman, J., Plshen, R., and Sonte, C. (1984). *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

Carpenter, G. and 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. and Grossberg, S. (1987b). ART2: Self-Organizing of Stable Category Recognition Codes for Analog Input Patterns. *Applied Optics*, 26(23):4919–4930.

Carpenter, G. and Grossberg, S. (1990). ART3: Hierarchical Search Using Chemical Transmitters in Self-Organizing Pattern Recognition Architectures. *Neural Networks*, 3:129–152.

Carpenter, G., Grossberg, S., and Reynolds, J. (1991a). ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network. *Neural Networks*, 4:565–588.

Carpenter, G., Grossberg, S., and Rosen, D. (1991b). Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System. *Neural Networks*, 4:759–771.

Carpenter, G. A. and Grossberg, S. (1988). The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network. *IEEE Computer*, 21(3):77–88.

Caudell, T. P., Smith, S. D. G., Escobedo, R., and Anderson, M. (1994). NIRS: Large Scale ART-1 Neural Architectures for Engineering Design Retrieval. *Neural Networks*, 7(9).

Dawkins, B. P., Andreae, P. M., and O'Connor, P. M. (1994). Analysis of olympic heptathlon data. *Journal of the American Statistical Association*, 89(427):1100–1106.

Fisher, D. (1987). Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning*, 2(2):139–172.

Fisher, D. (1996). Iterative Optimization and Simplification of Hierarchical Clusterings. *Journal of Artificial Intelligence Research*, 4:147–179.

Gennari, J., Langley, P., and Fisher, D. (1989). Models of Incremental Concept Formation. *Artificial Intelligence*, 40:11–61.

Gordon, A. D. (1981). *Classification: Methods for the Exploratory Analysis of Multivariate Data*. London: Chapman and Hall.

Happel, B. and Murre, J. (1994). Design and evolution of modular neural network architectures. *Neural Networks*, 7(6/7):985–1004.

Hartigan, J. A. (1975). *Clustering Algorithms*. John Wiley & Sons, Inc.

Hecht-Nielsen, R. (1987). Counterpropagation networks. *Applied Optics*, 26(23):4979–4984.

Hrycej, T. (1992). *Modular Learning in Neural Networks; A Modularized Approach to Neural Network Classification*. Sixth-Generation Computer Technology Series. John Wiley & Sons, Inc.

Ishihara, S., Ishihara, K., Nagamachi, M., and Matsubara, Y. (1993). ART 1.5-SSS for Kansei engineering expert system. In *Proceedings of International Joint Conference on Neural Networks*, pages 2512–2515.

Ishihara, S., Ishihara, K., Nagamachi, M., and Matsubara, Y. (1995). arboART: ART based hierarchical clustering and its application to questionnaire data analysis. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pages 532–537. IEEE Press.

Jordan, M. I. and Jacobs, R. A. (1992). Hierarchies of Adaptive Experts. In Moody, J. E., Hanson, S. J., and Lippmann, R. P., editors, *Advances in Neural Information Processing Systems 4*, pages 985–992. Morgan Kaufmann, San Mateo, CA.

Jordan, M. I. and Jacobs, R. A. (1993). Hierarchical Mixtures of Experts and the EM algorithm. Technical Report 9203, MIT Computational Cognitive Science, MIT, Cambridge, MA.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69.

Lampinen, J. and Oja, E. (1992). Clustering properties of hierarchical self-organizing maps. *Journal of Mathematical Imaging and Vision*, 2:261–272.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1990). Back-propagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551.

Li, T., Fang, L., and Li, K. Q.-Q. (1993). Hierarchical classification and vector quantization with neural trees. *Neurocomputing*, 5:119–139.

Merz, C. J. and Murphy, P. M. (1996). UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Technical report, Department of Information and Computer Science, University of California, Irvine, CA.

Miyata, Y. (1991). *PlaNet, A Tool for Constructing, Running, and Looking into a PDP Network*.

Moody, J. and Darken, C. J. (1989). Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation*, 1:281–294.

Moore, B. (1989). ART1 and Pattern Clustering. In Touretzky, D., Hinton, G., and Sejnowski, T., editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 174–185. Morgan Kaufmann, San Mateo, CA.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems*, 1:61–68.

Rumelhart, D., McClelland, J., and the PDP Research Group (1986). *Parallel Distributed Processing; Explorations in the Microstructure of Cognition*, volume 1: Foundations, chapter 5, pages 151–193. The MIT Press.

Shepherd, G. M. (1974). *The Synaptic Organization of the Brain.* Oxford University Press, New York.

Soon, H. and Tan, A. (1993). Concept hierarchy network for inheritance systems: Concept formation, property inheritance and conflict resolution. In *Proceedings of 15th Annual Conference of Cognitive Science Society*, pages 941–946. Lawrence Erlbaum Associates.

Van Essen, D. C., Anderson, C. H., and J., F. D. (1992). Information processing in the primate visual system: An integrated systems perspective. *Science*, 255(5043):419–423.

Wallace, C. and Boulton, D. (1968). An information measure for classification. *The Computer Journal*, 11(2):185.

Williamson, J. R. (1995). Gaussian ARTMAP: A Neural Network for Fast Incremental Learning of Noisy Multidimensional Maps. Technical Report CAS/CNS-95-003, Boston University, Center of Adaptive Systems and Department of Cognitive and Neural Systems.