VICTORIA UNIVERSITY OF WELLINGTON

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

Implementation of a visualisation tool for ART1 neural networks

G. Bartfai, M. Elliffe and P. Wood

Technical Report CS-TR-93/3
November 1993

Abstract

This paper introduces the prototype of a visualisation tool (SOTA) for Adaptive Resonance Theory (ART) neural networks. The tool is designed to provide novel views that help researchers understand the dynamic processes inherently present in ART networks, as a result of the interaction between the two layers and subsystems of ART. Two such views, *circle diagrams* and *input-relative diagrams*, are discussed in detail. Examples are also given showing some features of these views and SOTA itself.

Publishing Information

A shortened version of this report is to appear in *Proc. ANNES'93, The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Dunedin, New Zealand*, IEEE Computer Society Press.

The authors can be contacted through Guszti Bartfai (e-mail: `guszti@comp.vuw.ac.nz`).

1 Introduction

Scientific visualisation have always played an important role in understanding the behaviour of complex systems. Artificial neural networks are such systems where the emergent behaviour of the network as a whole is often more complex than the "rules", or mathematical formulae, that govern the interactions between a large number of neurons through adaptive links. It is, therefore, vitally important to have tools that can map the space of numbers into visuals giving different "views" of that space, thereby providing the user with valuable insights into the behaviour of these networks. Several visualisation techniques for neural networks have been developed in the past decade [3, 9], some of which have become de facto standards (for example Hinton diagrams). Most of these are primarily for representing the learning processes and network configurations of feedforward mapping networks, like the popular back-propagation network [8].

Self-organising networks with unsupervised, competitive learning (see e.g. [5]), however, do not belong to this category. While we can still apply some of the standard methods, here the nature of the learning process and the interaction between neurons dictates the need for other kinds of views as well (for example, the representation of developed categories and their attractive regions). Furthermore, Adaptive Resonance Theory (ART) networks have components and subsystems that make them a unique class of networks within the broader class of self-organising networks. ART networks are essentially classifiers[1], that develop categories by self-organisation. The complex processes of *hypothesis testing* (implementing an effective *adaptive search*) are a result of node interactions through intralayer and interlayer feedback connections, and between two subsystems. Traditional techniques like Hinton diagrams are of less value here; for example, the binary pattern of connections in ART1 conveys less information (see discussion in section 2). Also, the ones that are used in [1] show the general architecture of the network[2], as well as snapshots from the search cycle showing patterns of activity across the two layers (see details in section 2). While these diagrams are useful for understanding the basic mechanisms, they do not give us an insight of what triggers search cycles and, more importantly, how input patterns get classified (not in a simple nearest neighbour sense!), or how the past experience of the network affects the classification of new inputs.

In section 2, we describe the operation of the ART1 network in a detail that is necessary for understanding *circle diagrams* and *input-relative diagrams* introduced in section 3, together with SOTA itself[3]. Conclusions are drawn in section 4.

2 The ART1 network

The ART1 neural network is the binary version of the family of ART networks [1]. The ART architectures are neural networks that develop stable recognition codes in real time 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. Namely, how to keep learning new things without forgetting previously learned information. The architecture of ART networks can be seen on Figure 1. It has two layers: F1, the *comparison* layer (with N neurons), and F2, the *recognition* layer (with M neurons). The neurons in these layers store binary information during processing (short-term memory, or STM). The two layers interact with each other through bottom-up and top-down long-

[1] There are mapping versions of ART networks as well (see ARTMAP [2]). However, they are still built up of ART modules that retain the basic self-organising feature.

[2] Figure 1 is taken from [1].

[3] The full discussion of the design of SOTA can be found in [4].

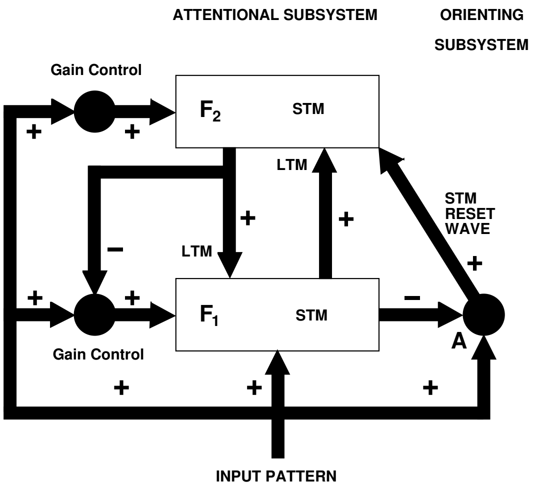

Figure 1: ART network architecture

term memory (LTM) connections, which are modified when the network learns. At each presentation of a new non-zero binary input pattern $\mathbf{a}$ ($a_i \in \{0,1\}, i = 1,2,...,N$), the network attempts to classify it into one of the existing categories based on its similarity to the stored prototype of each category, represented by individual nodes in the F2 layer. More precisely, for each node j in the F2 layer, $B_j = \sum_{i=1}^{N} b_{ij} a_i$ is calculated, where b_{ij} is the strength of the bottom-up connection between F1 node i and F2 node j. Then the top-down template $\mathbf{T_j}$ ($T_{ji} \in \{0,1\}, i = 1,2,...,N$) of the F2 node with the highest B_j is compared to the current input pattern at F1. If $|\mathbf{a} \cap \mathbf{T_j}||\mathbf{a}|^{-1} \geq p$ — where p is a system parameter called *vigilance*, and $|\cdot|$ is the norm operator, $|\mathbf{x}| \equiv \sum_{i=1}^{N} x_i$ — then input $\mathbf{a}$ is classified to F2 node j and both the bottom-up and top-down connection weights are adjusted for that node: $b_{ij}^{new} = c_i(\alpha + |\mathbf{c}|)^{-1}$ and $T_{ji}^{new} = c_i$, respectively, where c_i ($i = 1,2,...N$) is the ith component of the resulting pattern of activity (or STM) in the F1 layer, and $\alpha > 1$ is constant. Otherwise, i.e. when the stored template ($\mathbf{T_j}$) does not match the input sufficiently, the orienting subsystem sends a reset wave to the F2 layer, and inhibits the winning F2 node for the period of presentation of the current input. That inhibition reinstates the current input at the F1 layer, through the effects of non-specific gain control signals, and a new F2 node will be selected (from the remaining non-inhibited F2 nodes) with the maximum B_j whose template $\mathbf{T_j}$ will subsequently be compared to the input. This "hypothesis-testing" cycle is repeated until either the network finds a stored category (whose prototype matches the input closely enough) or allocates a new F2 node and stores the input pattern in its top-down template ($\mathbf{T_j}$) as the prototype of the newly-created category. The vigilance level affects the developed categories; the higher the vigilance, the more, finer categories will be developed.

3 The visualisation tool

The tool, named SOTA (for State Of The ART), provides dynamic display of various views of an ART simulation. There are two major software components to SOTA: the graphical user interface front-end, and the ART simulator back-end[4].

[4]Although SOTA is *not* a simulation tool, it works closely with an arbitrary ART simulator program that provides suitable information for SOTA's graphical front-end. For the prototype, we have modified a publicly available ART1 simulator to provide such information.

The graphical user interface front-end runs on X windows. It has been developed using the toolkit **Tk** [7] and tool command language **Tcl** [6] developed by John Ousterhout of Berkeley University. The Tcl/Tk package enables the easy creation of graphical user interfaces for X windows. Tk includes commands for the creation of widgets, including *buttons*, *entry* fields, *menus*, *text* widgets, and the *canvas*. The simple syntax of Tcl/Tk and its interactive nature facilitates the prototyping of X windows graphical user interfaces.

The ART simulator is a terminal based program. Communication between the graphical user interface front-end and the ART simulator is via pipes. A simple command protocol is used to control the ART simulator. Only minor modifications to the terminal based ART simulator was required. These related mainly to the responding to commands given by the front-end.

The *circle* and *input-relative* diagrams (discussed below) were drawn on a Tk *canvas* widget. The *canvas* widget has commands for drawing lines, bitmaps, ovals, arcs, polygons, rectangles and text. When a new input is sent to the ART simulator, the ART simulator sends back to the Tk based front-end data about its new status. The graphical procedures for the *circle* and *input-relative* diagrams then redraw the diagrams.

Circle diagrams represent N-dimensional binary feature vectors on the 2-dimensional plane. Each feature is uniquely represented by a clockwise angle from the upward vertical of between 0 and 360 degrees. Templates are drawn as connected line segments, each segment representative of one feature of which the set is composed (i.e. pointing in the associated unique direction). So a pattern whose features are all on, will have a closed polygon shape of equal side lengths[5]. Shorter templates will result in correspondingly shorter polylines, their endpoints being uniquely defined by the exact pattern of features. The advantage of using circle diagrams is that we have a graphical representation of binary patterns, that gives us a visual interpretation of overall template lengths (which affect ART1 processing!), and also some visual indication of how similar two binary patterns are (as opposed to the visual appearance of a bit sequence). For example, "runs" of adjacent features are easier to detect, with "arcs" appearing where adjacent features are present, and sharp changes in direction indicating that several adjacent features are absent. The learning process (i.e. the development of category prototypes) is clearly visible, too.

A sample circle diagram[6] drawn by SOTA can be seen on Figure 2.

Input-relative diagrams were designed in an attempt to display the processes invoked by presentation of an input pattern. They show the order in which winning F2 nodes are chosen (the bottom-up search path); the extent of the intersection between each template and the input pattern (the similarity metric); and compares each such intersection against vigilance. The basic form of the diagram is the top half of a semi-circle, with an origin at its centre.

Within this framework, category prototypes are represented by points inside the semi-circle. The distance of each point from the origin represents the activation (B_j) at the corresponding node of F2, the semi-circle representing the position occupied by any template which exactly matched the input pattern. Thus, the order in which templates are activated (i.e. win the competition in F2) is represented explicitly by how close template points are to the semi-circle.

Further, the angle of offset (clockwise from the horizontal) of a radial to any point repre-

[5] It is clear that the diagram will never be a *circle*, but can *look* like one with larger pattern sizes. Hence the name *circle* diagram.

[6] The input samples we use in this report are defined in [1]. Also, the appearance of SOTA reflects its state at the time of writing this paper. Later versions will most probably include more user-friendly features, for example using textual labels instead of numbers.

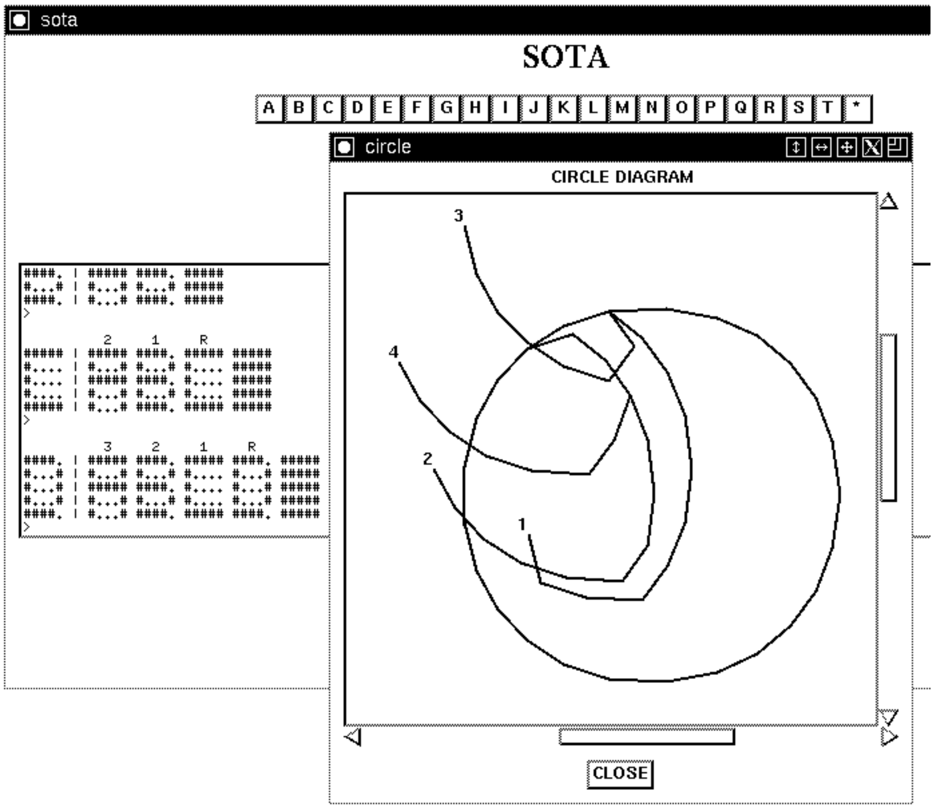

Figure 2: A SOTA circle diagram showing F2 top-down templates ($\mathbf{T_j}$, $j = 1, 2, 3, 4$) after the presentation of the capital letters A,B,C and D, represented in 5x5 pixel binary arrays (on the left; the 25-element vector is made up of the 5 vertical lines of the letters, starting from the top). Numbers identify F2 category nodes. Vigilance was set to 1 here so the network created a separate category for each input (1, 2, 3 and 4 for inputs `A`, `B`, `C` and `D`, respectively). The closed polyline represents the template for uncommitted F2 nodes, where $T_{ji} = 1$ ($i = 1, 2, ..., 25$).

sents the intersection of each template with the input pattern divided by the size of the input pattern (i.e. essentially the vigilance calculation $|\mathbf{a} \cap \mathbf{T_j}||\mathbf{a}|^{-1}$). Thus the value of the vigilance parameter itself can be shown as a radial from the origin to the semi-circle, no template left of which could achieve resonance with the input pattern.

Note that a template which corresponds exactly to the current input appears at the right hand intersection of the activation-level semi-circle and the base. Since it is an exact match, the activation will be the maximum possible for that input pattern (thus the distance from the origin will be maximal) and the similarity metric is also maximal (thus the radial angle is 180 degrees).

The uncommitted template representations always have the maximal offset angle, since they will always match any input pattern absolutely ($T_{ji} = 1$ initially). However, due to small initial bottom-up weights, their corresponding F2 nodes will only achieve small pre-competition activations (B_j), and thus the uncommitted nodes representation will be close to the origin (see F2 node 7 in Figure 3).

Two example diagrams are given in Figure 3. These diagrams are built from exactly the same ART network configuration, but represent the markedly different state achieved given

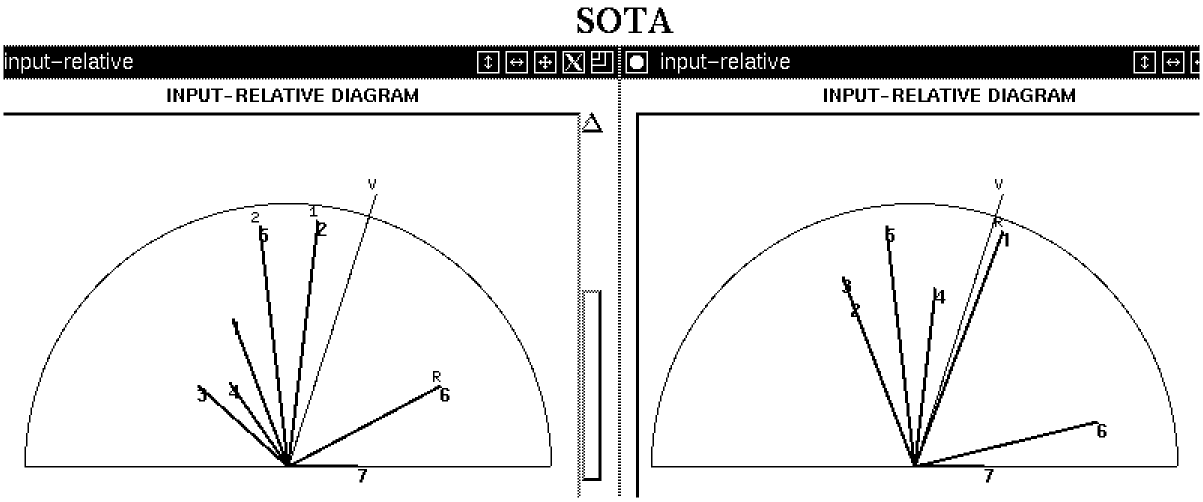

Figure 3: SOTA input-relative diagrams showing the search path of the ART1 network in response to input letters I (left; with search) and F (right; no search) after an initial training sequence of capital letters from A through T. Boldface numbers identify F2 nodes, while the others show the sequence of F2 nodes tested until resonance was achieved (R). Vigilance, at 0.6, is represented by a radial of 108 degrees (V).

the presence of different inputs.

The particularly valuable properties input-relative diagrams possess are that:

- the user can easily see the effect of a different level of vigilance, since movement of the vigilance radial could allow inclusion of some template whose similarity metric was insufficient for resonance, or exclusion of some template whose similarity metric was sufficient; and

- the diagram explicitly portrays the mechanism by which ART networks select templates. This representation is invaluable given the overall goal of assisting users to understand the processes involved in ART networks.

Where circle diagrams only convey information regarding the state of the network (the configuration of its templates, or an approximation of what this means in terms of classification of the input space), the input-relative diagram shows *why* the network behaves as it does.

4 Conclusion

In this paper, we introduced SOTA, a novel visualisation tool for ART neural networks. We described *circle diagrams* and *input-relative diagrams* that show different aspects of ART1 processes.

Throughout the experiments we carried out with SOTA, we have found it a valuable tool that helped us gain interesting insights into the behaviour of ART1 networks. Some of the experimental results have shown us in what aspects SOTA should be improved (for example, more flexible selection of inputs, ability to disable feature displays temporarily). They have also stimulated further thoughts about ART networks in general. Furthermore, we believe

SOTA could be a useful teaching aid; something which we plan to explore further in the near future.

We also would like to extend the tool in the future to include ARTMAP, a supervised version of ART networks [2].

Acknowledgements

This work was in part supported by the Victoria University of Wellington (Research Grant No.RGNT/594/371).

References

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

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

[3] M.W. Craven and J.W. Shavlik. Visualizing learning and computation in artificial neural networks. Technical report, University of Wisconsin, 1991.

[4] Martin Elliffe. Visualisation of processes in ART neural networks. Technical Report CS-TR-93/2, Department of Computer Science, Victoria University of Wellington, New Zealand, February 1993.

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

[6] J.K. Ousterhout. TCL: An embeddable command language. In *Proceedings of the 1990 Winter USENIX Conference*, pages 133–146, 1990.

[7] J.K. Ousterhout. TK: An X11 toolkit based on the TCL language. In *Proceedings of the 1991 Winter USENIX Conference*, pages 105–115, 1991.

[8] D.E. Rumelhart and J.L. McClelland, editors. *Parallel Distributed Processing; Explorations in the Microstructure of Cognition*, volume 1, chapter 8, pages 318–362. The MIT Press, 1986.

[9] J. Wejchert and G. Tesauro. Visualizing processes in neural networks. *IBM Journal of Research and Development*, 35(1/2):244–253, January/March 1991.