This page is left blank

This page is left blank

DEFENCE R&D DÉFENSE

Associative Memory in a Recurrent Neural Network

Simon A. Barton
Defence Research Establishment Suffield

Defence R&D Canada
Technical Memorandum
DRES TM 2001-053
May 2001

Associative Memory in a Recurrent Neural Network

Simon A. Barton
Defence Research Establishment Suffield

Defence Research Establishment Suffield

Technical Memorandum

DRES TM 2001-053

May 2001

Author

Simon A. Barton

Approved by

D.M. Hanna
Head/TVSS

Approved for release by

R.W. Herring
Head/Document Review Panel

Abstract

It is shown that unsupervised learning in a recurrent neural network (RNN) can lead to explicit associative memory. Using two separate sensor arrays during training, the RNN stores the information necessary to reproduce both images when presented with a single sensor image after training. This is a system that is capable of learning to make connections between different forms of sensory data. Memory is encoded in the strengths of the connections between the nodes of the RNN. Only patterns that have been seen during training generate strong recognition signals. Other patterns generate signals indicating that they can not be identified.

The storage capacity of the RNN depends on the number of nodes and the number of their input connections. It is shown that the RNN memory capacity does not decrease abruptly with increasing number of training patterns. In fact, the reproduction precision decays gradually. For a given RNN size, the memory simply becomes less clear as more storage is attempted. If unacceptable indications of recognition and identification are obtained, it is only necessary to increase the number of nodes.

Since the RNN uses a structure and learning rules that have previously been shown to result in autonomous goal-driven motion in a simulated mobile machine, it may now be possible to generate adaptive control for an autonomous vehicle by relating the input from different sensor arrays with the actions taken by the machine in response to its sensors. This may lead to a mobile machine that can learn and improve its actions with experience.

This page intentionally left blank.

Executive summary

Development of autonomous intelligent systems has been identified as an area of high priority within DRDC, and autonomous land vehicle R&D will continue to be performed at DRES.

Part of the DRES R&D has focussed on recurrent neural networks (RNNs) because they appear to have the potential to generate some of the characteristics believed to be required for the creation of intelligent behaviour in a machine. Memory of sensor information, and the formation of relationships between patterns from different types of sensors is one such characteristic. The ability to use sensor input to drive action is another. Useful intelligence is expected to emerge when memory and association of previously seen patterns can influence the machine action in such a way as to demonstrably improve the achievement of goals. In an autonomous system, goals must be internally generated by the structure of the machine itself. The structure and learning mechanisms of an RNN are based on biological neural systems. An RNN is a collection of artificial neurons (nodes) with multiple feedback connections. Previous work has shown that RNNs can be used to store and classify images using a form of explicit memory, and can control the motion of a simulated mobile machine equipped with sensors that respond to a radiating target.

In explicit memory, close reproductions of the patterns used in training are generated. To achieve this, feedback learning in a memory region of the RNN is employed. When an image that has been previously used in training is presented to the RNN through sensor nodes, it stimulates reproduction of the image in an output array coupled to the RNN memory region, and at the same time generates a signal indicating recognition. If an image that has not been seen before is presented, no reproduction is generated and the recognition signal is weak.

In previous work, only one sensor array was coupled to a memory storage region, and there was no internal method for associating patterns from different sensor arrays. Identification of a pattern was performed externally, by a comparison of recognized images with stored patterns. The present RNN structure, however, now contains separate memory regions for different sensor inputs, and this leads to self-organized associative memory and direct, internally generated identification of known patterns.

The storage capacity of the RNN depends on the number of nodes and the number of their input connections. The present work shows that the RNN memory capacity does not decrease abruptly with increasing number of training patterns. In fact, the reproduction precision decays gradually. For a given RNN size, the memory simply becomes less clear as more storage is attempted. If unacceptable indications of recognition and identification are obtained, it is only necessary to increase the number of nodes.

By the repetition of image pairs during training, this system stores both images and regenerates both even when only one of the pairs is presented after training. It is thus able to name patterns that it has repeatedly seen previously. Future work will now attempt to link automatic associative memory with goal-driven motion control to create a simulation of an adaptive, autonomous machine that can learn to improve its performance in obstacle avoidance and navigation. This is a requirement for an autonomous vehicle operating in an unknown and dynamically changing environment.

This work was performed under a Technology Investment Fund project, entitled, "Self-Organized, Goal-Driven Adaptive Learning".

Simon A. Barton. 2001. Associative Memory in a Recurrent Neural Network. DRES TM 2001-053. Defence Research Establishment Suffield.

Table of contents

Abstract i

Executive summary iii

Table of contents v

List of figures vi

List of tables vi

1. Introduction 1

2. Structure and Learning Rules 3

   2.1 Recurrent Node Responses 4

   2.2 Learning Rules 5

      2.2.1 Hebbian Learning 5

      2.2.2 Feedback Learning 6

3. Results 8

   3.1 Feedback Learning Validity 9

   3.2 Network Storage Capacity 10

4. Conclusions 13

References 14

List of figures

1 RNN with associative memory 3

2 Ten image sensor patterns 8

3 Correlations for images (lower) and codes (upper), using a training set of size two with 250 R nodes 10

4 Correlations for images (lower) and codes (upper), using a training set of size two with 500 R nodes 11

5 Code correlations using 2, 6 and 10 training pairs (top to bottom), with 250 R nodes 11

6 Code correlations using 10 training pairs, with 250 (bottom) and 500 (top) R nodes 12

List of tables

1 Optimal learning parameters 9

1. Introduction

A recurrent neural network (RNN) is a set of connected processing units, called nodes. Unlike the nodes in a feed-forward network, which are arranged in layers and are only connected to nodes in the next layer, a node in an RNN can be connected to any number of other nodes anywhere in the network. External signals may be supplied by one or more sensor arrays connected directly to nodes in specified input regions of the RNN. This is a dynamic system with ubiquitous internal feedback, so that even with a fixed input pattern, the node outputs may take many cycles before stabilizing, or they may become periodic or chaotic depending on the network parameters. As an RNN operates, the connection strengths can be made to vary with the signal strengths, and this can lead to properties that may be useful in the development of an intelligent machine. The ways in which the connections vary are called the *learning rules.* Previously [1], we have established conditions under which an RNN will operate in the stable regime; i.e. with a fixed input pattern the RNN rapidly moves to a state where all the nodes give constant outputs.

It is well known that fully connected networks are capable of self-organized image storage and classification [2]-[6], and we have shown that our form of RNN is capable of memory,re cognition and identification of patterns in images [7],[8]. Our RNN structure and learning rules also lead to goal-driven behaviour in a simulated mobile machine [9],[10]. In the latter work, a mobile machine with light sensors was able to move automatically to a stationary radiating energy source. In recent, unpublished work we have found that an RNN-controlled machine is strongly attracted to a moving source, and will continue to closely follow the source for thousands of network cycles. The attractive behaviour was found to be most effective if the RNN is composed of neurons that are either excitatory or inhibitory; i.e. a neuron either emits a positive signal to all neurons connected to it, or it emits a negative signal. In this case the connection weights are all positive but each neuron has an associated sign. The initial work had used neurons that could be both excitatory and inhibitory; i.e. the neuron outputs were all positive, but the connection weights were positive or negative.

In our last reports on pattern recognition [7],[8], identification was achieved by comparing stored images with the image in an output array generated by the RNN when a test image was presented to the RNN in the sensor array. The "closest" image in the stored set was taken to be an identification. This required a pixel by pixel comparison with all the images in the stored set, so for large images and sets this would be very time consuming. The only perceived advantage of using an RNN over a standard statistical classifier in that case was that the correlation between input pattern and RNN output could be used as a measure of recognition, before attempting a classification. If the image had already been seen and remembered by the RNN, the correlation was high. If it was previously unseen, the correlation was low. The correlation factor could also be used as a measure of identification confidence. However, in that work we also suggested that when a known image is presented to an RNN, it should be possible for the RNN to generate both the remembered image *and* a code vector identifying or classifying it. In this case, the classification would be read directly, with a confidence level given by the image correlation.

The process of automatically remembering patterns and their identifications is a form of *associative memory*. The aim of the work presented in this memorandum is to confirm that associative memory of two separate sensor image streams is indeed possible with our RNNs.

This associative memory process is explicit, self-organizing and adaptive. By explicit, we mean that the images are stored in such a way that the original images may be regenerated pixel by pixel. They are not regenerated in an abstract representation or as extracted features.

This RNN system can learn to name objects or patterns by repeatedly being told what they are; i.e. the name is the associated pattern. In a mobile machine, it could retain sensor patterns and the future sensor state resulting from actions taken. It could also associate sensor patterns with actions that lead to "improved" future sensory states, if an internal mechanism exists for qualifying the concept of improvement.

The long-term aim is to link goal-driven behaviour with the memory and association of multi-sensor input to develop adaptive, self-organized learning that improves the control (navigation and obstacle avoidance) of a mobile machine.

The work is part of a Technology Investment Fund (TIF) project (under 2fm01) with the long term goal of demonstrating self-organized, goal-driven adaptive learning in a simulated mobile vehicle with multiple sensors. This is a requirement for an adaptive, autonomous vehicle operating in an unknown and dynamically changing environment.

The details of the RNN structure and learning rules are given in Chapter 2. Chapter 3 presents results demonstrating associative memory, and Chapter 4 gives a discussion and conclusions.

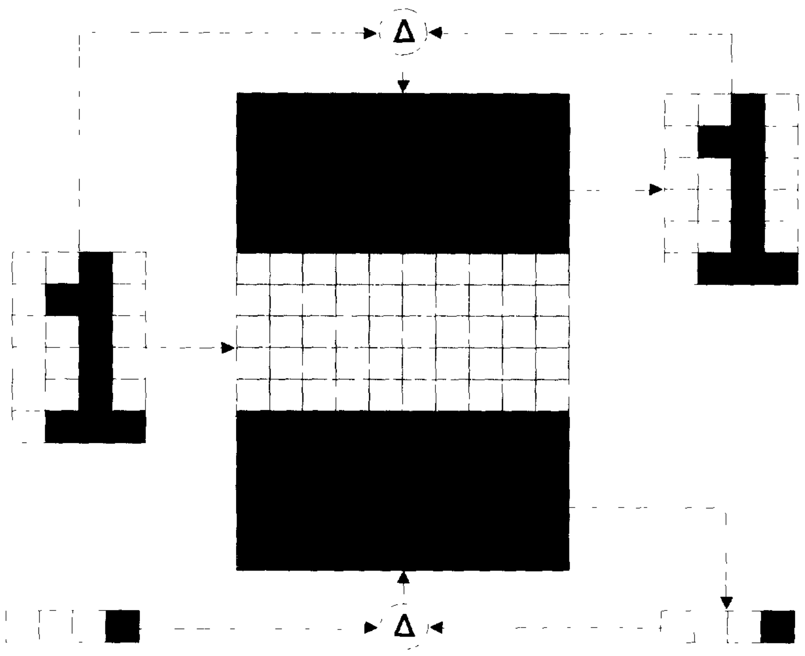

***Figure 1:** RNN with associative memory*

2. Structure and Learning Rules

RNN structures and properties have been described previously [1], [7]. The principal addition to the previous RNN structures is that there are now two memory regions that store and regenerate patterns from two independent sensor arrays. We call these the *image* sensor array (S_I) and its associated *code* (S_C).

In Figure 1, an associative RNN is shown with a 4x6 image array on the left. The image array is connected to the central region of 150 recurrently connected nodes (R nodes). This input region is shown as 50 yellow R nodes. Each pixel in the image has a fixed number of connections to R nodes in the input region, chosen randomly, but with the constraint that no two image pixels have the same set of output connections. The code sensor array S_C is shown as a 4x1 array. It provides no direct input to the RNN. Output arrays are shown on the right, connected to blue upper and lower R node regions; memory node arrays M_I and M_C provide output generation of S_I and S_C respectively.

Figure 1 indicates that it is only input from S_I that generates responses in the memory regions M_I and M_C, and that during training there is feedback from the differences between corresponding pixels in the image and code arrays. This feedback adjusts the connection weights to R nodes in the memory regions so that the differences are reduced. The details are given later in this section.

Each connection between two R nodes has a weight that modifies the signal transferred between the nodes by a multiplicative factor. Each R node has the same number of input connections. established randomly after the input connections from the sensor nodes (S nodes) have been chosen. An R node can be connected to any other R node in the RNN, but self-connection is not permitted. To ensure that there are no isolated groups of nodes. each R node supplies input to the next one in the array.

The M_I and M_C nodes (M nodes in general) have input connections that are chosen randomly from R nodes in the memory regions, but with the requirement that each M node has an equal number of inputs from excitatory and inhibitory R nodes.

The connection sets for all nodes are unique. No S node can have more than one output to any R node, and no two S nodes can have the same set of outputs. No M node can have more than one input from any R node, and no two M nodes can have the same set of inputs. No R node can have more than one input from any other node.

The weights are all positive in this RNN, and each R node has a fixed output sign. The nodes in the RNN are thus either excitatory or inhibitory. Signs are chosen randomly, but with a user supplied weighting: e.g. one may request the fraction of positive R nodes to be 0.5, or one may generate an RNN that is predominantly excitatory or inhibitory.

The weights for connections from S nodes to R nodes, and from R nodes to M nodes are kept constant, because these connections simply transfer information. Any learning occurs within the R nodes. The magnitudes of the R node connection weights are initially random, between the limits supplied by the user; e.g. 0.1 to 0.9. The S to R node weights are all initially set to a user specified constant (default 1.0), as are the R to M node weights.

The R node offsets are all initialized randomly between user defined limits; e.g. -1 to 1. The M node offsets are set to zero.

Output signals for the R nodes are initialized randomly between 0.0 and 1.0.

2.1 Recurrent Node Responses

During operation of the RNN, a set of image/code pairs are input through the sensor arrays. Each pair of patterns is held in the sensor arrays for a number of RNN cycles, set by the user. We call this the *exposure time*. During training, the weights and offsets of the R nodes are adjusted at each cycle, according to the learning rules given below

The nodes respond by passing a weighted sum of the incoming signals through a sigmoid function. The responses are calculated at each cycle of the RNN. The weighted input (x_n) to the nth node at cycle t is:

(1) $$x_n^t = \sum_i^{n_c} W_{ni}^{t-1} \sigma_i f_i^{t-1}$$

where n_c is the number of input connections, and the subscript jg ives an index for the node providing input on the ith connection, with weight W_{ni}^{t-1}. The input connection indices are stored in a matrix C,thu s $j = C_{ni}$. The output of the node on the jth connection is f_j^{t-1}, and σ_j is its sign.

The sigmoid function has an exponent s,con trolling the steepness of the function, and an offset x_0,g iving the centre of its output range. The output of the nth node is then:

(2)
$$f_n^t = [1 + \exp(-s(x_n^t - x_{0n}^{t-1}))]^{-1}$$

Fixed values for the sigmoid exponents (s_r) are used for every R node. The M nodes use a separately defined fixed value (s_m). The sensor nodes simply supply values between 0.0 and 1.0.

2.2 Learning Rules

The algorithms for changing the variable parameters in a neural network are called the learning rules.

In particular, by changing the R node weights and offsets, repeated patterns can be stored and recalled, and when the output of an RNN is used to control motion, a mobile machine with feedback from the environment through the sensors may generate goal-driven behaviour [9] [10]. In the latter case, a learning rule based on that proposed by Hebb [11] for biological systems was used. Explicit memory of images was also achieved using learning based on feedback of the correlation between S node and M node arrays [7]. In this work both Hebbian and feedback learning are used.

2.2.1 Hebbian Learning

The modified Hebbian learning rule can cause the activity of an RNN to converge to a fixed point if an input pattern is retained during the RNN cycles. Different patterns can generate unique fixed points in the RNN node output space [7].

Hebbian learning was used for all nodes in the RNN that are *not* connected to an M node; i.e. for those R nodes that are not being used to generate memory.

In Hebbian learning, neural connections are strengthened when a strong input signal results in a strong output signal from the neuron receiving the input. A decay term is added to ensure that a weight cannot grow without limit.

For an R node n at cycle t, which received an input f_j^{t-1} from the jth node and generated an output f_n^t,the change in the weight W_{ni} connecting nodes n and j is:

(3) $$\Delta W_{ni}^{t} = \alpha f_{j}^{t-1} f_{n}^{t} - \gamma W_{ni}^{t}$$

where α and γ control the growth and decay rates respectively. Recall that the connection index j is given by C_{ni}.

The node offsets are also allowed to vary. This is necessary because a fixed offset may result in an insensitive response to the input signal levels for some nodes. The R node offsets are adjusted at each cycle using:

(4) $$\Delta x_{0n}^{t} = \beta(f_{n}^{t} - 0.5)$$

This tends to move the node output towards the centre of its range. In a convergent system with a fixed input image, all the R node outputs eventually approach 0.5. With changing sensor input, or for an RNN operating in a nonconvergent regime, the offsets change slightly at each cycle. The rate of change is controlled by the parameter β, supplied by the user.

2.2.2 Feedback Learning

By adjusting the input connection weights of each R node that provides output to one or more M nodes, the correlation between the M and S array can be maximized.

For the nth R node that is connected to one or more M nodes, we first calculate a sum of differences between the signals in each connected M node and its corresponding S node, in the following way:

(5) $$D_n = \frac{\sigma_n}{n_{cm}} \sum_{i}^{n_{cm}} \Delta_i$$

where Δ_i is the difference between the outputs generated by the ith S and M nodes, n_{cm} is the number of connections to M nodes from the nth R node, and σ_n is the output sign of the R node.

M node arrays (M_I, M_C) for the image and its code are connected to different regions of the RNN, as shown in Figure 1, so the summation above only involves image differences or code differences for a given R node.

D_n defines the required direction for the change in output signal strength of the nth R node. Dividing by n_{cm} confines the magnitude of D_n to [0.0, 1.0]; i.e. it is a fractional value.

For the *n*th R node, the change in the *j*th input weight at cycle *t* is calculated in the following way:

$$\Delta W_{jn}^{t} = \rho D_{n}^{t} f_{n}^{t} \sigma_{k} f_{k}^{t} W_{jn}^{t} \tag{6}$$

where ρ is the feedback learning rate. The magnitude of a weight change depends on the strength of the incoming signal f on the jth input connection (from the R node whose index $k = C_{jn}$), and on the strength of the weight itself (W_{jn}).

The R node offsets are also moved in the required direction by using:

$$\Delta x_{0n}^{t} = -\beta D_{n}^{t} \tag{7}$$

As the S and M arrays become correlated, offset and weight changes become smaller.

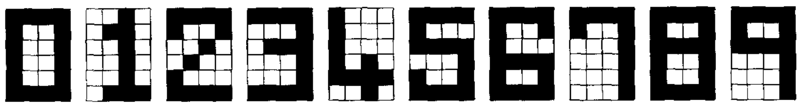

Figure 2: *Ten image sensor patterns*

3. Results

Figure 2 shows ten 4x6 sensor images (24 S nodes) that were used as a training set for the RNN. In previous work, much larger images have also been stored using Hebbian learning with feedback; a set of five 16x16 images [7], and objects in 32x32 IR images (1024 S nodes) [8] were successfully identified.

The principal questions to be answered in the present work are the following: can a pattern and its identification code be stored by the simultaneous presentation of two sensor image streams during training; and can an identification be recovered from the code memory array by the single presentation of a pattern in the image sensor stream?

The ten associated code sensor arrays are 4x1 vectors representing the binary form of the decimal images shown in Figure 2. An example of an associated pair is given in Figure 1 for the number 1.

To assess the effectiveness of an RNN in recalling a set of patterns, we have used the following correlation function:

$$C = 1 - \frac{1}{Nn_s}\sum_{j}^{N}\sum_{i}^{n_s}|\Delta_i|_j \tag{8}$$

$|\Delta_i|_j$ is the difference between the ith S and M nodes for the jth pattern, N is the number of patterns, and n_s is the number of sensor pixels. If every pattern is accurately reproduced during training, C approaches 1.0 ($|\Delta_i| \rightarrow 0$). During training, pattern j is held in the S node array for the exposure time, e_t. After e_t cycles, the i sum is calculated and the next pattern is presented for e_t cycles. C is calculated over each complete cycle of all patterns. Since there are two sensor arrays, there are actually two correlations calculated, C_i and C_c, for the image and code arrays respectively.

The memory process is affected by the following parameters: the constant value used for the S node to R node weights; the learning parameters, α, β, γ and ρ; the node sigmoid steepness factors s_r,and s_m; the number of R nodes; the number of output connections for the S nodes; the number of input connections for the R and M nodes; the exposure time, e_t; and the fraction of positive (excitatory) R nodes.

The RNN *structural* parameters (number of R nodes and connections, fraction of positive R nodes) can be separated from the *learning* parameters. We used two different RNN structures, which we will refer to as R250 and R500, containing 250 and 500 R

Parameter	Value
α	0.005
β	0.0005
γ	0.0005
ρ	0.001
s_r	0.1
s_m	8.0
e_t	12

Table 1: *Optimal learning parameters*

nodes respectively. In both cases the fraction of positive R nodes was fixed at 0.5, the number of R node input connections was 40 and the number of output connections for each S node was also 40. For R250, the image input and memory regions had 100 nodes each, and the code memory region had 50 nodes. For R500 these regions were doubled. It was found that for R250, the optimal number of input connections for the memory node arrays (M_t and M_c) was 20, while for R500 better performance was obtained using 40 input connections.

Optimal values for the learning parameters are given in Table 1. The exposure time shown (e_t) is in network cycles. In addition, the constant weight between S and R nodes was 5.0, and the fraction of positive R nodes was 0.5.

It was previously found [7] that if the RF learning rate ρ is too high the training becomes unstable. The present work confirmed this, and we found that the value of ρ that maintains stability can be higher when fewer image/code pairs are used in the training set; e.g. when only two pairs are used, $\rho = 0.004$ maximized the correlations, but for 6 and 10 pairs $\rho = 0.001$ was required to maintain stability.

3.1 Feedback Learning Validity

To show that the learning rules described in Section 2.2 do in fact lead to associative memory, just the first two image pairs were used as a training set. Clearly, with only two training pairs, the correlations should approach 1.0 quite closely as the number of training cycles increases. This is shown to be the case in Figure 3, for the R250 network. The maximum code correlation (0.9992 at 6000 cycles) is higher than that for the images (0.9797) because the code array size is much smaller (4) than the image size (24). As expected, the correlations improve if more R nodes are used; the R500 results are shown in Figure 4, where both image and code correlations exceed 0.99. These curves show that by adjusting R node connection weights in the memory regions using our feedback learning method, the RNN can generate copies of the images and codes that are presented during training. If the connection weights in the memory regions are now fixed after training, and an image pattern that was part of the training set is presented in only the image sensor array, the RNN will then generate both a close copy

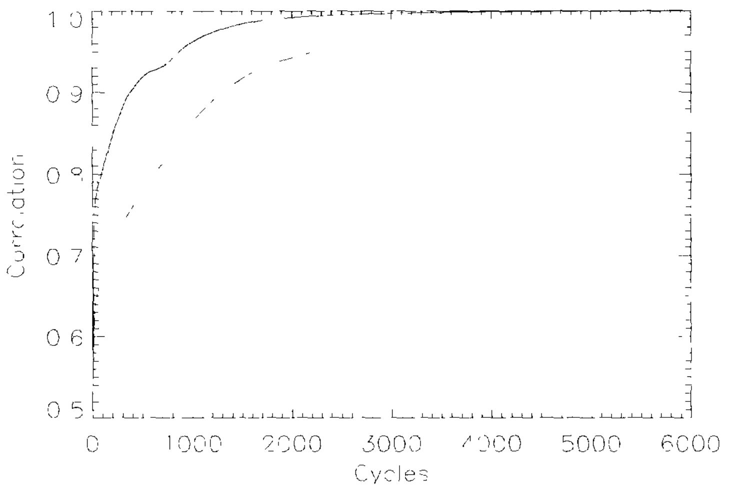

Figure 3: *Correlations for images (lower) and codes (upper), using a training set of size two with 250 R nodes*

of the image *and* the code array that was associated during training. This validates the feedback learning equations (2.5 to 2.7). A previously unseen pattern in the image array will produce a low correlation (< 0.7), indicating a lack of recognition: in this case no reliable identification can be obtained from the code array output.

3.2 Network Storage Capacity

For a network with a fixed number of R nodes, as the number of image/code training pairs increases, the total correlation decreases. For example, using R250 with 2, 6 and 10 training pairs, the code correlations are shown in Figure 5. With a value greater than 0.8, the correlation for all ten 4x6 test image/code pairs is still good enough for reliable identification using 250 R nodes. From previous work [7], we know that correlations between sensor and memory image arrays will be less than 0.7 for unrecognized (previously unseen) images.

The degradation in correlation, and hence in identification confidence, is a useful measure of the storage capacity of the network. If the network does not give correlations sufficiently high to ensure reliable identification, more R nodes may be used.

If the number of R nodes is increased, the correlations can be improved. This is shown in Figure 6, where the code correlations for ten training pairs are given for the R250 and R500 networks.

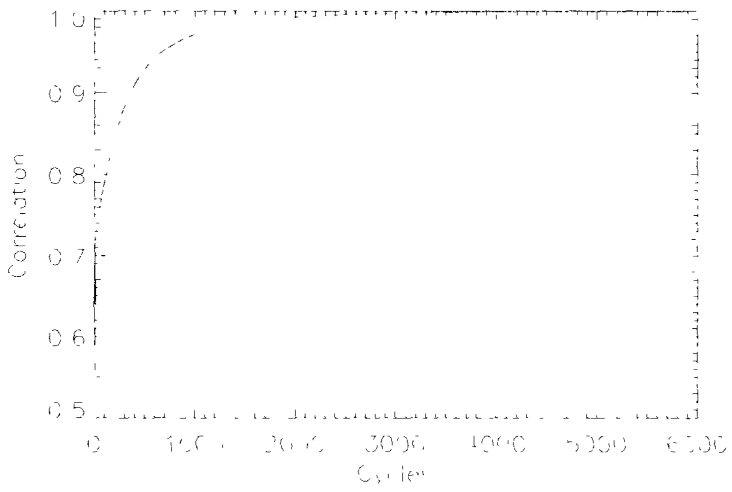

Figure 4: *Correlations for images (lower) and codes (upper), using a training set of size two with 500 R nodes*

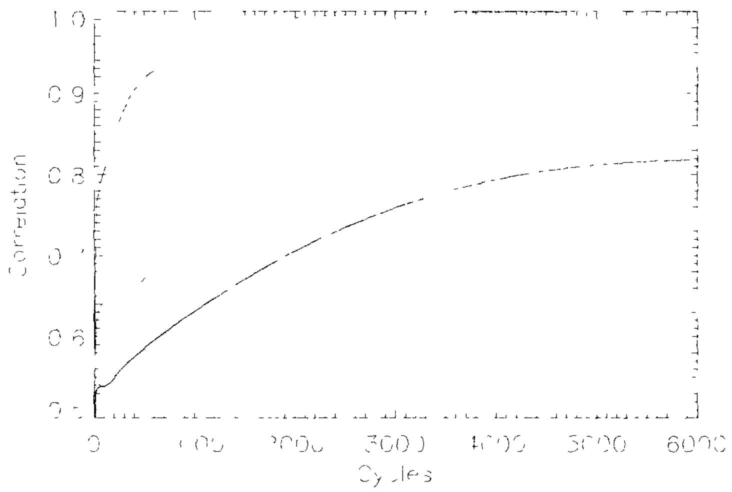

Figure 5: *Code correlations using 2, 6 and 10 training pairs (top to bottom), with 250 R nodes*

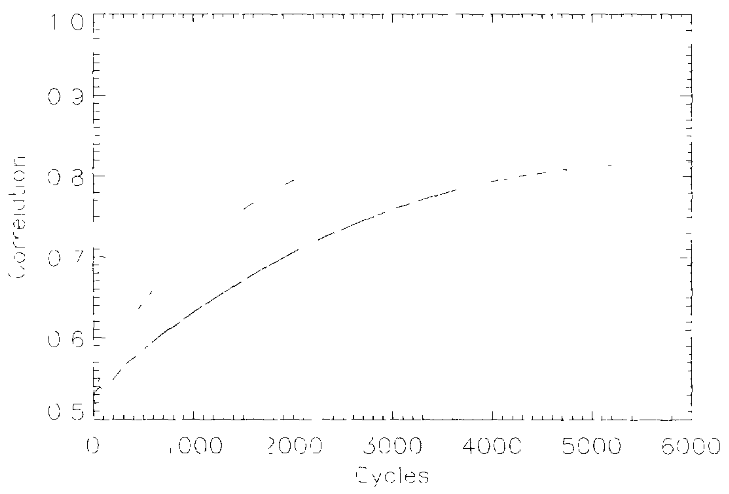

Figure 6: *Code correlations using 10 training pairs, with 250 (bottom) and 500 (top) R nodes*

4. Conclusions

The results given in Section 3 confirm that it is possible to generate explicit associative memory of two separate sensor image streams with RNNs of the same form as those that we have used in goal-driven autonomous navigation of a simulated machine. It would not be difficult to extend this technique to form associations between more than two sensor streams that contain repeated simultaneous patterns.

It has been shown that a pattern and its identification code can be stored by the simultaneous presentation of two sensor image streams during training. After training, the correlations between input and output node arrays are high for a previously seen pattern, and an image may then be directly identified by the output in the code memory array.

With our feedback learning, explicit memory is encoded in the input connection weights of the R nodes in the memory regions of an RNN, and the reproduction of a remembered image is the result of an interaction throughout the entire RNN, from the sensor array to the memory region. The memory is therefore distributed, and is only recalled by the presentation of an image that has been seen during training, or one that is very similar. An image that has not been seen before does not generate a strong correlation signal, indicating that it can not be identified. Previous work [7] has also shown that the RNN with feedback learning is quite tolerant of noise in the input image; e.g. with 40% maximum noise, 98% of the ten 4x6 images were correctly identified.

The storage capacity of an RNN depends on the number of R nodes and the number of their input connections. The reproduction precision of remembered images also depends on the number of input connections to the M nodes and the number of output connections from the sensor nodes. Our results show that the RNN memory capacity does not decrease abruptly with increasing number of training patterns. Rather, the reproduction precision, as measured by the correlation values, decays gradually. For a given RNN size, the memory simply becomes less clear as more storage is attempted. If the correlation values become too low to give acceptable indications of recognition and identification, it is only necessary to increase the number of nodes until correlation values that give satisfactory results are obtained.

In future work, it may now be possible to relate the input from different sensor arrays with the actions taken by a mobile machine in response to its sensors. This may lead to adaptive control in a mobile machine that can learn. Its ability to navigate through obstacles and reach a goal would then improve with experience. We are currently developing a simulation of this scenario with the intention of demonstrating self-organized, goal-driven adaptive learning.

References

1. Barton, S.A., “Structure and Convergence Properties of a Recurrent Neural Network”, DRES SM-1489, 1996.
2. Kohonen, T, “Correlation Matrix Memories”, IEEE Trans. on Computers **C-21** (1972), p. 353.
3. Anderson, J.A., “A Simple Neural Network Generating an Interactive Memory”, Math. Biosci. **14** (1972), p. 197.
4. Kohonen, T, “Self-organized Formation of Topologically Correct Feature Maps”, Biol. Cybern. **43** (1982), p. 59.
5. Kohonen, T, *Self-Organization and Associative Memory*, Springer, Berlin, 1984.
6. Hopfield, J.J., “Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons”, Proc Natl. Acad. Sci. USA, **81** (1984), p. 3092.
7. Barton, S.A., “Techniques for Pattern Classification Using a Convergent Recurrent Neural Network”, DRES SR-709, 1998.
8. Barton, S.A., “Recognition and Identification of Objects in IR Feature Images using a Recurrent Neural Network”, CAN contribution to TTCP W7 KTA 7-2, Final Report, 2000.
9. Barton, S.A., “The Effect of Sensory Input on Trajectories Generated by Recurrent Neural Networks”, DRES SR-589, 1993.
10. Barton, S.A., “Two Dimensional Movement Controlled by a Chaotic Neural Network”, Automatica, **31** (1995), p. 1149.
11. Hebb, D.O., *The Organization of Behaviour*, John Wiley and Sons, Inc., New York, 1949.

Unclassified
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document Organizations for who the document was prepared, e g. Establishment sponsoring a contractor's report, or tasking agency, are entered in Section 8.) DRES	2. SECURITY CLASSIFICATION (overall security classification of the document, including special warning terms if applicable) Unclassified

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title)

Associative Memory in a Recurrent Neural Network

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g Doe, Maj. John E.)

Barton, Simon, A

5. DATE OF PUBLICATION (month and year of publication of document) March 2001	6a NO. OF PAGES (total containing information, include Annexes, Appendices, etc) 20	6b. NO OF REFS (total cited in document) 11

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include the address.)

N/A

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) Project 2FM01	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.) N/A
10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity This number must be unique to this document.) TM 2001-053	10b OTHER DOCUMENT NOs. (Any other numbers which may be assigned this document either by the originator or by the sponsor.) N/A

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors; further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
( ) Distribution limited to government departments and agencies; further distribution only as approved
( ) Distribution limited to defence departments; further distribution only as approved
( ) Other (please specify):

12 DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally corresponded to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected).

Unlimited distribution

Unclassified
SECURITY CLASSIFICATION OF FORM

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C) or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

It is shown that unsupervised learning in a recurrent neural network (RNN) can lead to explicit associative memory. Using two separate sensor arrays during training, the RNN stores the information necessary to reproduce both images when presented with a single sensor image after training This is a system that is capable of learning to make connections between different forms of sensory data. Memory is encoded in the strengths of the connections between the nodes of the RNN. Only patterns that have been seen during training generate strong recognition signals. Other patterns generate signals indicating that they can not be identified

The storage capacity of the RNN depends on the number of nodes and the number of their input connections. It is shown that the RNN memory capacity does not decrease abruptly with increasing number of training patterns In fact, the reproduction precision decays gradually. For a given RNN size, the memory simply becomes less clear as more storage is attempted If unacceptable indications of recognition and identification are obtained, it is only necessary to increase the number of nodes.

Since the RNN uses a structure and learning rules that have previously been shown to result in autonomous goal-driven motion in a simulated mobile machine, it may now be possible to generate adaptive control for an autonomous vehicle by relating the input from different sensor arrays with the actions taken by the machine in response to its sensors This may lead to a mobile machine that can learn and improve its actions with experience.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required Identifies, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

Neural networks; machine intelligence; autonomous machines, unsupervised learning, pattern recognition, associative memory.

#516190

CA011640