Representation of Temporal Patterns in Recurrent Networks*

Fred Cummins†
Cognitive Science Program
Indiana University
Bloomington, IN 47405
fcummins@ucs.indiana.edu

Abstract

In order to determine the manner in which temporal patterns are represented in recurrent neural networks, networks trained on a variety of sequence recognition tasks are examined. Analysis of the state space of unit activations allows a direct view of the means employed by the network to solve a given problem, and yields insight both into the class of solutions these networks can produce and how these will generalize to sequences outside the training set. This intuitive approach helps in assessing the potential of recurrent networks for a variety of modelling problems.

Some Problems of Representation

This paper investigates the way temporal patterns are represented by recurrent networks. Small sequence discrimination tasks were devised with the specific intention of requiring the network to base its discrimination on the temporal structure of the input sequences. The resulting solutions were then analyzed using elementary tools from dynamic systems theory. Our primary interest was to establish the relationship between the temporal characteristics of the training set and the representations developed by the network.

The relationship between a representation and its counterpart in the physical world may be completely arbitrary, as between an instance of the written or spoken word 'penguin' and any particular bird of that sort, or there may be a more direct relationship, e.g as between a tonotopic map and the frequencies it represents. Many of the issues involved in distinctions among representational systems have been discussed by van Gelder and Port (1993, van Gelder, 1992). If the relationship between a symbolic representation and its semantic properties is completely arbitrary, questions about the difficulty of symbol grounding arise.

Questions of symbol grounding and semantic arbitrariness are not often addressed in developing computational systems with representational properties. It has been more usual to determine by *fiat* the nature of the representational primitives and to attempt to create meaningful relations between these units. An alternative is to induce the representational system from the properties of the input to the system. This is now quite common in the connectionist world, where one determines an architecture, a training algorithm, an input set, and possibly an output set and the network is then left to devise a suitable representation. In this manner, one is assured that whatever the final representation, its relationship to the properties of the input which are relevant to the task at hand will be non-arbitrary.

Distributed representations, as found in connectionist networks, possess many well known desirable properties (Hinton et al., 1986). One significant drawback, however, is that these representations are hard to interpret. A network may have solved a task, and we can test the solution and its generality, but we may still have only a vague notion of how this has been achieved. In some cases, analytical mathematical techniques are available. For example, we can determine how a perceptron has partitioned the input space, and we can predict that where no such linear decomposition of the task exists, the network will never be able to solve it (Minsky & Papert, 1969). Another way to peek into the dark world of distributed representations is the use of cluster analysis to examine the similarities between the representations developed for vectors in the training set (Elman, 1990).

This paper will present an alternative technique for studying representation — a technique that is particularly useful for examining the nature of solutions arrived at by a recurrent network in a series of simple sequence discrimination problems. Recurrent networks preserve some history of previous states through their recurrent links, and, accordingly, they have been widely used in the processing of temporal patterns. Since we are then dealing with the temporal evolution of a system, it is natural to consider the network as an instance of a dynamic system, so that the tools of Dynamic Systems Theory may be employed.

*Submitted to the 15th Annual Conference of the Cognitive Science Society

†Also of Department of Linguistics. Thanks to Sven Anderson, Devin McAuley, Robert Port, Timothy van Gelder and Michael Gasser for their comments and assistance. This research was supported by ONR grant N00014-91-J1261 to Prof. R. Port, Indiana University.

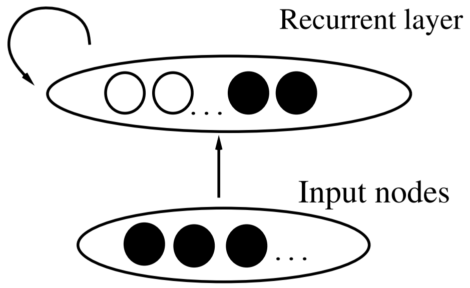

Figure 1: Network architecture. The input layer consists of a small number of linear nodes. These are fully connected to the sigmoid nodes at the recurrent layer, where all nodes are interconnected, including self-recurrent connections. Training values are provided for only one of the recurrent nodes — on (activation = 1) signifies recognition of a target pattern, otherwise the node remains off (activation = 0).

Connectionist Networks as Dynamic Systems

The simplest way to regard a network as a dynamic system is to regard the training variables (weights, biases) as fixed constants, and to consider only the variation in unit activations over time for the trained network. In fact, as the training sets used herein contained a small set of possible input vectors, we may look at the system behavior under constant vector input, determine all attractors and basins of attraction for this input, and repeat for all possible inputs. For most (but not all) cases, the phase portrait for each input vector will then simply be a single point attractor and its associated basin of attraction[1] In this simplest case, the network is functioning as a content addressable memory (Kohonen, 1987).

With more than three nodes in the recurrent layer, the state space of the network is of too high a dimension to be directly represented. Trajectories and attractors may however be plotted in a lower dimensional space using the technique of Principal Component Analysis, which identifies a smaller number of orthogonal axes in state space, along which variance is maximized. All phase portraits presented in this paper are schematic views of the first two principal components. We found analysis of these two alone to be sufficient to understand the nature of the solutions obtained by the networks.

General procedure

For the following simulations, the network architecture of Figure 1 was used. The simple linear input nodes pass each component of an input vector to all the nodes at the recurrent layer. The state space of the network can be regarded as an n-dimensional space with one axis for each recurrent node. In the simulations reported here we used 7 recurrent nodes. There were typically 2 or 3 input nodes. Input vectors were simple 2- or 3- component mutually orthogonal vectors. All tasks involved a training set divided into target and distractor sequences. A sequence consisted of a number of elements (**A, B, C** or **0**), each of which could be presented for one or more time steps. A typical sequence is shown in Table 1.

Input vectors	Symbolic representation	Teacher values
000	**0**	0
000	**0**	0
100	**A**	0
100	**A**	0
010	**B**	0
010	**B**	0
001	**C**	1
001	**C**	1

Table 1: Coding of the exemplary sequence **00AABBCC**. The single output node is trained to remain off until the final sequence element appears.

The Real-Time Recurrent Learning algorithm, using the method of teacher forcing, from Williams and Zipser (1989) was used to train the network. Prescribed values were only given for one of the recurrent nodes. This node was trained to stay off (activation = 0) until the final element of a sequence was presented and then to come on (activation = 1) if the sequence was a target sequence, and otherwise to remain off.

Simple Sequence Identification

From previous work (Anderson et al., 1991; Cummins et al., 1993), we knew both that a network of the above type could solve a simple sequence identification task, and what the qualitative dynamics of the resulting system are like. In those simulations, networks learned to discriminate sequences such as **0ABC** from ones such as **0BAC**, **0BCA** etc. Sequences were preceded by one or more zero vectors in order to restart the network from the same region of state space each time. If we trained a network by presenting each element in the sequence for more than one time step (i.e. presenting **00AABBCC** rather than simply **0ABC**), we found that the solution obtained generalized to changes in presentation rate almost without limit, and the network could still successfully discriminate between target and distractor sequences. Figure 2 illustrates how this was achieved. The training protocol adopted means that recognition of a target sequence (output node $N_0 = 1$) corresponds to passage of the system trajectory through the hyperplane $N_0 = 1$. This region appears in the principal components projection as a bounded *recognition region*. As the sequence is presented, the trajectory approaches the associated attractor for each sequence element in turn. The attractors have been fixed in a manner such that only the target sequence will induce a trajectory through this region. Furthermore, we found

[1] An excellent and intuitive introduction to those concepts from dynamic systems theory used here is Abraham & Shaw (1983).

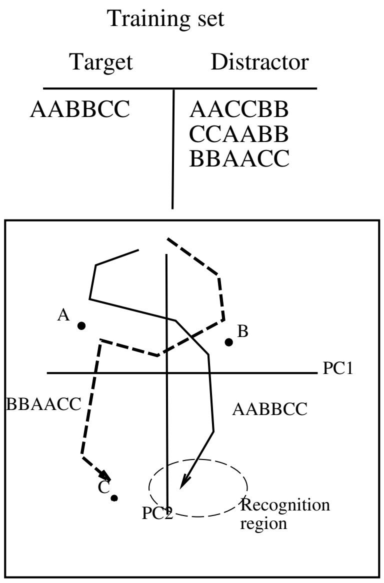

Figure 2: Training set and resultant dynamic system for a simple sequence identification task. Only the target trajectory passes through the recognition region (the hyperplane $N_0 = 1$).

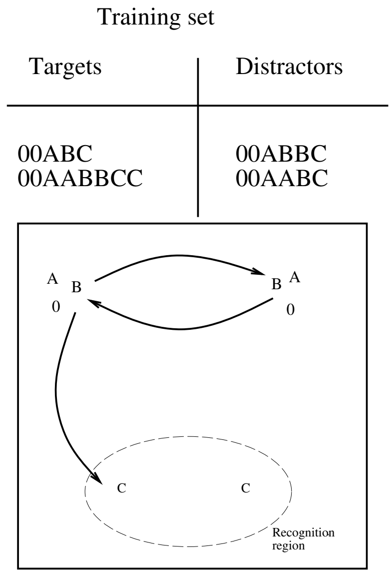

Figure 3: Training set and induced dynamics for a system which develops dual attractors for a single input vector to implement a parity checker. The first zero vector input brings the system to the group of attractors on the right hand side.

that varying the rate of presentation of the sequence (presenting each sequence element for a greater or lesser number of time frames) did not qualitatively alter the trajectories, with the result that the network still correctly discriminated between targets and distractors.

Although the rate-oblivious character of the network might be useful for some applications (Port, 1990), it is not yet a particularly good model of human temporal pattern processing. Humans exhibit some, but not unbounded, tolerance to rate variation and are very sensitive to rhythmic qualities, such as durational ratios, in temporal patterns (Sorkin et al., 1982; Sorkin, 1987). In order to assess the suitability of recurrent networks as models of human temporal pattern processing, the present study investigates what sort of temporal and rhythmic differentiation the network is capable of. In all the following simulations, both target and distractor sequences are variants of **0AB** or **0ABC** that differ only in the temporal structure of their individual components.

Dual Attractors

The first example presents us with a cautionary note. The network does not always arrive at a solution which accords with the investigator's understanding of the problem. One might hope to learn from the training set shown in Figure 3 whether the network can learn to tell sequences in which each element is presented for the same length of time from those with varying element duration. However, an alternative solution is to do a simple parity check on the sequence preceding the final element (**C**), without differentiating between **A**, **B** or **0** (zero). Target patterns have an even number of timesteps prior to the final element, distractors have an odd number. The network arrived at a rather bizarre solution which in fact implements a parity checker. Each input vector has an attractor consisting of a pair of points between which the system oscillates with constant input. The attractors for **A**, **B** and **0** are grouped closely together. On the first timestep, the system moves to the group (**A**,**B**,**0**) on the right hand side. It then jumps back and forth, irrespective of whether **A**, **B**, or **0** is presented. The route of approach followed on presentation of **C** differs according to whether the system is in a state near the right or left group of attractors. If a sequence of even length was presented, the sequence is a target and the system is in a state near the left hand group. The approach to **C** is then directly towards the left hand **C** attractor, which lies in the recognition region for this system. If the sequence was a distractor and the trajectory is therefore coming from the right hand bunch of attractors, the approach to the **C** attractor is indirect, leading through the left attractor group. In this case, a single **C** input vector is not sufficient to bring the network into a state close to the recognition region. In this rather odd way, the network has solved the problem set. Note that it used parity checking rather than counting to do so.

Inducing a Periodic Attractor

Can a recurrent network of this kind count? The training set of Figure 4 was devised so as to preclude a rate-oblivious type of solution and to require the network to count the number of **A**s pre-

Training set:

Targets	Distractors
00AAB	00AB
00AAAB	00AAAAB

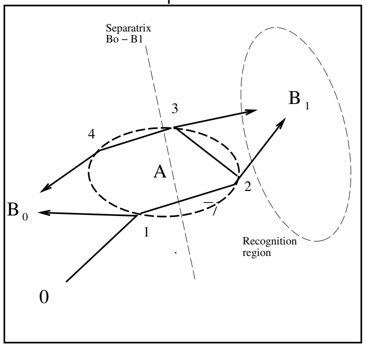

Figure 4: Training set and resultant dynamic system used to induce a periodic attractor.

Training set

Targets	Distractors
00AB	00AAB
00AAAB	00AAAAB
00AAAAAAB	00AAAAABB

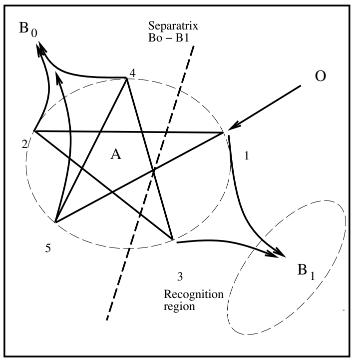

Figure 5: Training set and resultant dynamic system used to induce a complex periodic attractor.

ceding the first **B**. Patterns with 2 or 3 **A**s are targets, while those with 1 or 4 are distractors. The dynamic system arrived at by training has a periodic attractor for **A**, and two point attractors for **B**, such that the separatrix for the basins of attraction for the two **B** attractors divides the periodic attractor into approximately two halves. In this manner, after presentation of one or four **A**s, the system is in the basin of attraction of one **B** attractor, and either two or three **A**s leaves it in the other. From this solution, we can immediately see how this system will react to longer sequences of **A** followed by **B**. Patterns with 2 or 3 **A**s go to $\mathbf{B_1}$, with 4 or 5 to $\mathbf{B_0}$, 6 or 7 to $\mathbf{B_1}$, etc. The network has learned to impose a rhythmic pulse of two beats on a steady state input (**AAAA**...) and to categorize strings of **A**s according to whether they end on an even or odd number of pulses.

Because of the discrete nature of the simulations where $\Delta T = 1$, the trajectories are not continuous, but instead they step around the periodic attractor as illustrated. In fact, running the network with a smaller ΔT shows that the **A** attractor in the continuous case is in fact a point located at the center of the periodic attractor of the discrete simulation, and the solution no longer works. The solution is thus critically dependent on the discrete mode of processing.

A More Complex Periodic Solution

Given the behavior observed in the previous network, we sought to induce more complex dynamics in the same vein. Figure 5 shows a training set which cannot be solved by a simple pulse recognizer as above. Targets have 1, 3 or 6 **A**s, while distractors have 2, 4 or 5. Nonetheless, the solution obtained is qualitatively similar to the previous one. Again, the element to be 'counted' develops a periodic attractor, while the final element develops two point attractors, one associated with targets and one with distractors, with a separatrix dividing the periodic attractor depending on phase angle. This time however the cycle is traversed along the path of the five-pointed star, as illustrated. If the previous solution imposed a 2-beat pulse on a constant input and distinguished between even and odd pulses, this network imposes a 5/4 rhythm on continuous input and discriminates between beats 1 and 3 on the one hand, and beats 2, 4 and 5 on the other.

The training set used to induce this system is very small, and if the periodic nature of the solution is to successfully generalize to significantly longer sequences of **A**, the sixth step around the periodic attractor must coincide exactly with that of the first. In fact, in testing this with continuous input of **A** we obtained the phase shifting trajectory shown in Figure 6 A, which will eventually generalize 'incorrectly' for longer patterns. We then subjected the network to further training with isolated vectors having a much larger number of **A**s, giving as teacher values the categorization which would result if the sixth step coincided exactly with the first. After very little training, the trajectory shown in Figure 6 B was obtained. In effect, we had succeeded in fine tuning the phase of the 5/4 attractor so that the network recognized sequences based on their number of **A**s modulo 5.

Discussion

One objection to connectionist style modelling is the black box nature of the representational systems which result. The above analysis of several trained networks has shown one manner with which

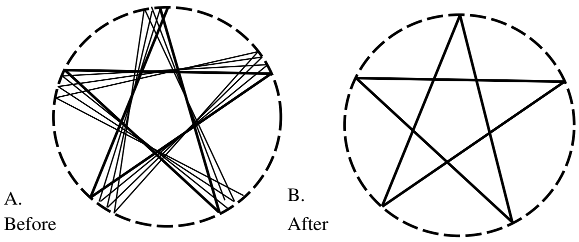

Figure 6: Fine Tuning of a Periodic Attractor. A: Trajectory on continuous presentation of **A** before tuning. B: Trajectory after tuning

to gain insight into how a network has solved a problem, and which 'tools' can be employed by the network in attempting to satisfy the constraints imposed by a training set of sequential patterns. The networks studied shared a common architecture, training algorithm and training regime, so that inferences drawn about their capacities and proclivities cannot, without further investigation, be extended to other cases. From the above results, we saw that the network solves this kind of sequence identification task by the location of point and periodic attractors and the shaping of their respective basins of attraction.

From what we observed, there does not seem to be any straightforward way for these networks to count. They can however learn to perform modulo arithmetic, based on many kinds of underlying rhythmic structure in the training set. The sorts of systems we observed give indications as to where these networks might be usefully employed in cognitive modelling: sequence discrimination, where insensitivity to rate of presentation is a desired property, can be easily accomplished. An unexpected finding is that the networks have the ability to unearth periodic regularities in their input set and to interpret steady state input in terms of the learned periodicity. This suggests possible application in models of rhythm perception and, perhaps, production. The phase tuning carried out in the final example illustrates how knowledge of the desired dynamics can be applied in the exact determination of the resulting dynamic system.

Any network which solves a temporal pattern recognition problem can be usefully regarded as a dynamic system. In some cases, knowledge of the dynamics of the desired final system may be available, and recent techniques developed by Cohen (Cohen, 1992) provide analytical methods for the construction of a set of dynamic equations which allow the location of both point and periodic attractors with precision. In our approach, however, the desired dynamics is not derived analytically, but is induced by the temporal properties of the training set. This seems desirable from a cognitive modelling point of view, if we regard neural systems as complex dynamic systems which have evolved by induction from the information in and constraints of the environment. Knowledge of how the temporal properties of the environmental stimulus are related to the resulting dynamic system appears necessary in order to construct actual models of temporal processors e.g. for music or speech recognition. By inducing a representational system directly from the properties of the environmental stimulus, we are assured of a non-arbitrary relationship between that representational system and the physical world.

Analyzing cognitive models as dynamic systems requires us to adopt a new vocabulary. One problem which remains is to decide how the notion of representation may be used in talking about dynamic systems (van Gelder, 1992). In the above analyses, it is easy to believe that one is looking at the representations these systems have developed, but it is hard to put one's finger on some one thing which might qualify as a representation. Van Gelder and Port (1993) have suggested that the discrete trajectory segments might qualify as the 'symboloids' from which compositional representations might be developed. However, as argued by Freeman (Freeman & Skarda, 1990), it may be that the standard vocabulary of representational systems in unsuited to the interpretation of cognition as dynamic process rather than symbolic content.

References

Abraham, R. and Shaw, C. 1983. *Dynamics, The Geometry of Behavior, Part 1.* Santa Cruz, California: Aerial Press.

Anderson, S., Port, R., and McAuley, J. D. 1991. Dynamic memory: A model for auditory pattern recognition. Unpublished manuscript.

Cohen, M. 1992. The construction of arbitrary stable dynamics in nonlinear neural networks. *Neural Networks*, 5:83–103.

Cummins, F., Anderson, S., Port, R., and McAuley, J. D. 1993. A dynamic systems model for temporal pattern recognition. Unpublished manuscript.

Elman, J. 1990. Finding structure in time. *Cognitive Science*, 14:179–211.

Freeman, W. J. and Skarda, C. A. 1990. Representations: Who needs them? In McGaugh, J. L., Weinberger, N. M., and Lynch, G., editors, *Brain Organization and Memory : Cells, Systems, and Circuits*, pages 375–380. Oxford University Press.

Hinton, G., McClelland, J. L., and Rumelhart, D. E. 1986. Distributed representations. In Rumelhart, D. and McClelland, J., editors, *Parallel Distributed Processing, Vol. 1*, pages 77–109. Cambridge, MA: The MIT Press.

Kohonen, T. 1987. *Content-Addressable Memories.* Springer-Verlag.

Minsky, M. and Papert, S. 1969. *Perceptrons.* Cambridge, MA: MIT Press. expanded edition 1987.

Port, R. F. 1990. Representation and recognition of temporal patterns. *Connection Science*, 2:151–176.

Sorkin, R. 1987. Temporal factors in the discrimination of tonal sequences. *Journal of the Acoustical Society of America*, 82(4):1218–1226.

Sorkin, R., Boggs, G. J., and Brady, S. L. 1982. Discrimination of temporal jitter in patterned sequences of tones. *Journal of Experimental Psychology: Human Perception and Performance*, 8(1):46–57.

van Gelder, T. 1992. The proper treatment of cognition. In *Proc. of 14th Cog. Sci. Society*, pages 641–646.

van Gelder, T. and Port, R. F. 1993. Beyond symbolic: Prolegomena to a *kama-sutra* of composotionality. In Honavar, V. and Uhr, L., editors, *Symbol Processing and Connectionist Models in AI and Cognition: Steps towards Integration.* Academic Press. to appear.

Williams, R. and Zipser, D. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.