UC Merced
Proceedings of the Annual Meeting of the Cognitive Science Society

Title
In Search Of Articulated Attractors

Permalink
https://escholarship.org/uc/item/5j69w416

Journal
Proceedings of the Annual Meeting of the Cognitive Science Society, 18(0)

Authors
Noelle, David C.
Cottrell, Garrison W.

Publication Date
1996

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

In Search Of Articulated Attractors

David C. Noelle and **Garrison W. Cottrell**
Computer Science & Engineering 0114
University of California, San Diego
La Jolla, CA 92093-0114
{dnoelle,gary}@cs.ucsd.edu

Abstract

Recurrent attractor networks offer many advantages over feed-forward networks for the modeling of psychological phenomena. Their dynamic nature allows them to capture the time course of cognitive processing, and their learned weights may often be easily interpreted as soft constraints between representational components. Perhaps the most significant feature of such networks, however, is their ability to facilitate generalization by enforcing "well formedness" constraints on intermediate and output representations. Attractor networks which learn the systematic regularities of well formed representations by exposure to a small number of examples are said to possess *articulated attractors*. This paper investigates the conditions under which articulated attractors arise in recurrent networks trained using variants of backpropagation. The results of computational experiments demonstrate that such structured attractors can spontaneously appear in an emergence of systematicity, if an appropriate error signal is presented directly to the recurrent processing elements. We show, however, that distal error signals, backpropagated through intervening weights, pose serious problems for networks of this kind. We present simulation results, discuss the reasons for this difficulty, and suggest some directions for future attempts to surmount it.

Introduction

Recurrent attractor networks have been studied by cognitive modelers since the onset of the recent connectionist renaissance. The hallmark of these networks is the complex evolution of their processing element activity over time. Though dynamic by nature, these networks have been applied to many associational mapping tasks which possess no inherent temporal component. In domains as diverse as content-addressable memory (Hopfield, 1982), schema formation (Rumelhart et al., 1986b), and word naming (Plaut & McClelland, 1993), recurrent networks which settle over time to some stable activation state have displayed some noteworthy advantages over feed-forward models. The attractor networks directly exhibit time-varying processing, allowing them to capture the dynamics of cognition in a manner which may be validated against common psychological measures, such as reaction times. Also, learned connection weights in such recurrent networks often lend themselves to interpretation as soft constraints between representational units, facilitating analysis. Perhaps the most interesting potential advantage of such networks, however, is the manner in which attractor basin formation may aid in generalization to novel inputs.

Attractor networks can encourage generalization by enforcing "well formedness" constraints on the intermediate and output representations produced by an otherwise feed-forward process (Mathis & Mozer, 1995). Such constraints are embodied in these networks as distinct fixed-point attractors for every possible well formed representation. Patterns may be "cleaned up" by such a network via a process of settling over time to one of these meaningful, well formed, and stable activation states. The potentially combinatoric space of valid attractor basins need not be explicitly trained, however, but may arise in the compositional interaction of trained attractors (Plaut & McClelland, 1993). It has long been known that the training of recurrent networks may result in *spurious* attractor basins: fixed-point attractors which are not explicitly trained (Hopfield, 1982). Under appropriate conditions, however, these spurious attractors may actually arise in a systematic manner, producing *serendipitous basins* which encode novel but meaningful patterns of activation. We refer to the dynamics of such networks as containing *articulated attractors* – meaningful attractor basins arising from the compositional interaction of explicitly trained attractors.

This paper provides an empirical analysis of the conditions under which articulated attractors form in recurrent neural networks trained using various versions of backpropagation (Rumelhart et al., 1986a). This work stemmed from our initial attempts to incorporate an attractor network of this kind into our connectionist model of instruction following (Noelle & Cottrell, 1995), a model which develops an internal representation of verbal instructions in the service of a task (St. John, 1992). We discovered that articulated attractors did *not* appear in this model, and this paper sprang from our attempt to explain why. In hopes of acquiring a deep understanding of the learning difficulties experienced by our model, we began with the most simple attractor network architecture possible – a single recurrent layer of processing elements. We incrementally augmented this network with further layers of units, expanding the complexity of the architecture towards the configuration of our instruction following model. This investigation revealed that articulated attractors form readily when the network's recurrent layer is directly provided with a teaching signal, but such systematic dynamics do *not* appear when recurrent weights are shaped by backpropagated error. In addition to demonstrating this finding, this paper also presents some possible explanations for why this is so.

We begin by describing the simple structured memory task which we used to examine attractor formation in a number of recurrent network architectures. We then present simulation results for three successively more complex architectures, and we close with a discussion of these results.

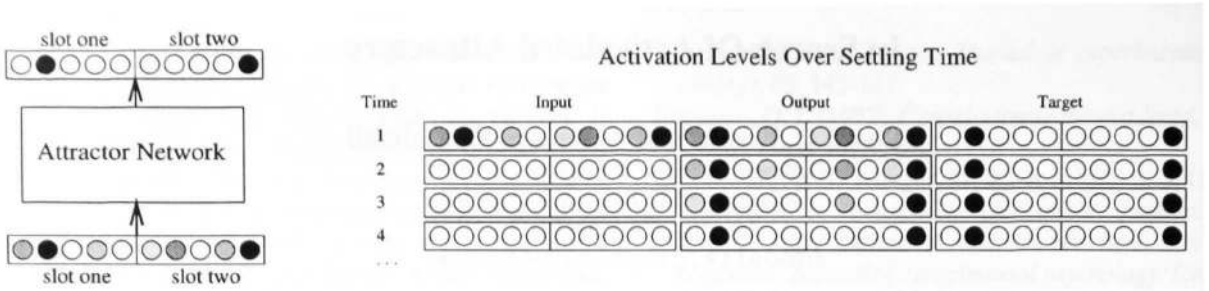


Figure 1: The Slot-Filler Structure Memory Task

A Structured Memory Task

In hopes of facilitating analysis, we selected an extremely simple task for our attractor networks. Each network was presented, for a single time step only, with an encoding of a simple slot-filler structure. The goal of the network was to "clean up" any noise in this representation and to retain the result at the network's output indefinitely, even after the input pattern was removed. Thus, through training, the network needed to acquire a distinct fixed-point attractor for each valid input slot-filler structure. These attractor basins had to be sufficiently wide to capture slightly noisy patterns, and the fixed-points needed to be sufficiently stable to remember the input for an indefinite period.

To be specific, each pattern represented a structure containing two slots, each holding exactly one of five distinct fillers. The contents of the slots were considered independent, with the specific filler in one slot in no way constraining the filler for the other. The whole was encoded as a 10 element binary vector, divided into two groups of five. Since each slot could contain only a single filler, exactly one element in each group of five was turned "on" in each valid pattern. The networks, then, were to learn an attractor for each input pattern involving exactly one of the first five elements "on" and exactly one of the last five elements "on". Thus, with five possibilities for each of two slots, there were only $5^2 = 25$ patterns considered "well formed" out of the $2^{10} = 1024$ possible binary input vectors. This task is depicted schematically in Figure 1. The diagram on the left side of that figure depicts the mapping being performed as the network settles, and the table on the right provides an example of the time course of input activity, expected output activity, and the target output. Note that the input pattern is made available to the network for the first time step *only*, requiring the network to both "clean up" and remember the pattern over time.

Systematic generalization was the focus of these experiments. The goal was to produce a fixed-point attractor for every valid slot-filler structure, given training on only a fraction of these valid patterns. To this end, each network was explicitly trained on some subset of the allowable input patterns, encouraging the formation of attractors for these patterns by the presentation of an error signal on every time step for a fixed settling period. Once trained, each network was then tested on *all* valid slot-filler representations, and the number of fixed-point attractors corresponding to these valid patterns was determined. The dynamic behavior of each trained network was also examined to locate any spurious attractors corresponding to ill formed patterns.

For small training sets, consisting of only a few valid patterns, we expected the networks to simply memorize the training instances – to build attractors only for the presented patterns. We predicted, however, that beyond some threshold in training set size the networks would generalize to *all* valid structures. In order to test this hypothesis, we trained each network architecture on multiple training sets, varying in size. Each trained network was examined to determine the attractor structure resulting from its training. At least five patterns were present in each training set, as this was the minimum number needed to turn each input element "on" at least once over a training set. The largest training set consisted of all 25 well formed patterns. The frequency of each filler in each training set was balanced as much as was possible given the small size of the training sets. Noise was added to input elements during training, but this noise never exceeded 5% of the activation range of the elements (i.e., 0.05 for binary units and 0.10 for bipolar units). Network output targets consisted of the "clean" patterns over the entire time course of network settling, as shown in Figure 1. A settling period of 10 time steps was used during training, and 100 time steps were used during testing.

The Emergence Of Systematicity

The first architecture examined was a single recurrent layer of sigmoidal processing elements. The entire network consisted of ten units which acted as both input and output for the network. This single layer was provided with complete recurrent connections to itself. Each unit also had a bias weight, resulting in a total of 110 adaptable connections. Unlike some single layer networks of this kind, this weight matrix was *not* constrained to be symmetric. The architecture is shown in Figure 2, on the left. The network received an input pattern by clamping the activation state of the processing elements to the input values for a single time step. After this initial time step, the activation state of the network was allowed to freely evolve according to the connection weights. Training was provided by a version of *backpropagation through time* (BPTT) (Rumelhart et al., 1986a) in which error is backpropagated for only a single time step, much as is done for Simple Recurrent Networks (Elman, 1990). An illustration of how this was implemented is also shown in Figure 2. Binary sigmoidal units were used, with a learning rate of 0.01, no momentum term, and a mean squared error objective function. Connection weights were initialized to small random values, normally distributed about 0 with a variance of 0.5. For each training set size, training was conducted for 4000 epochs (i.e., passes through the entire training set) with patterns randomly reordered on each epoch.

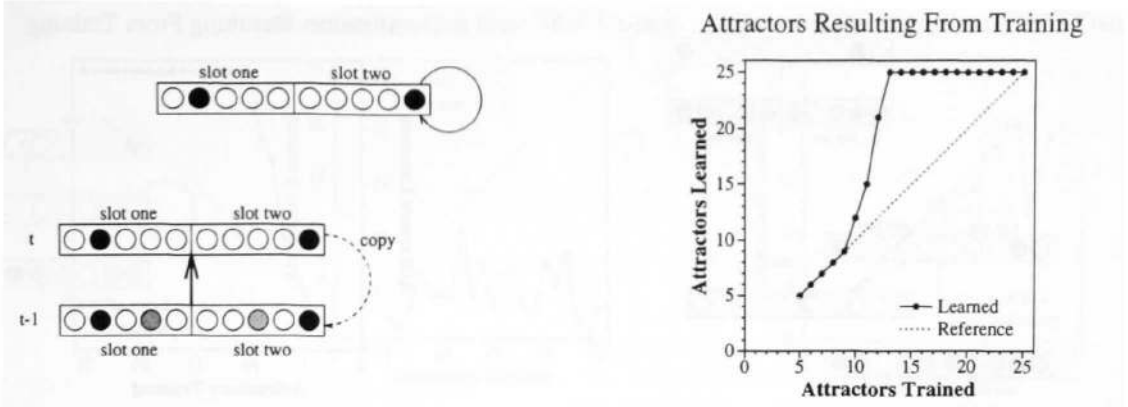


Figure 2: Single Layer Network: The Architecture, Unrolled In Time, And Generalization Performance

A summary of the results over all training set sizes is shown on the right side of Figure 2. That graph displays the number of well formed attractors found in a trained network as a function of the training set size. The plot also includes a reference line which depicts the hypothetical case of *no* generalization outside of the training set. Notice that small training sets resulted in the simple memorization of the trained attractors, but networks that saw at least half of all valid patterns consistently generalized to all 25 allowable structures. Furthermore, none of these networks constructed spurious attractor basins corresponding to ill formed patterns. The weights of these successful networks took the unsurprising form of two uncoupled winner-take-all networks. Each unit had a highly weighted self-connection and inhibited the other four units in its group of five. Weights on connections between units for different slots (i.e., between the two groups of five units) remained close to zero.

Given a sufficiently large training set, these networks consistently exhibited an emergence of systematicity. Generalization was perfect, with a fixed-point attractor formed for every valid pattern.

Input Preprocessing

The next architectural variant we considered involved the inclusion of a matrix of weights between the network input and the recurrent output layer. Instead of providing input patterns by clamping unit activations for the initial time step, noisy inputs were provided at an input layer for the first time step, and activity at this layer was set to zero for all remaining time steps. Activation levels at the recurrent output layer were initialized to zero. This network contained the same recurrent connection architecture as in the single layer case but also included a complete set of connections from the input layer, for a total of 210 adaptable weights. The same learning parameters were used, and training continued, once again, for 4000 epochs. As in the single layer model, weight modification was performed by BPTT, with the network unrolled in time for a single time step. The basic architecture of this network, as well as how it appears when "unrolled", is show in Figure 3. Summarized results are also graphed in that figure. Notice that systematic performance arose from even smaller training sets than in the single layer case. As before, no spurious attractors were found. The recurrent weights, once again, embodied two separate five-unit winner-take-all networks. Systematic generalization appears to arise spontaneously under this architecture, as well.

It is fairly clear why the inclusion of input weights introduced no additional difficulty for the learning of articulated attractors. The training of the input weights was essentially *decoupled in time* from the training of the recurrent weights. During the initial time step, output activity was at the initialized level of zero, which implied no change to the recurrent weights since this activity plays a multiplicative role in the backpropagation weight update equation. In other words, only the input weights could be updated on the first time step. On the other hand, on every subsequent time step the input layer activity was clamped to zero, directing all weight updates to the recurrent connections. In short, each of the weight matrices was provided with its own direct error signal at regular times during training.

An Indirect Error Signal

The final architecture examined here differed from the previous two in that an error signal was not provided directly to the recurrent layer but was backpropagated through an intervening matrix of weights. The basic architecture is shown in Figure 4. Unlike the last model, activity was propagated forward beyond the recurrent layer to a separate output layer of sigmoidal units. Error was computed at this final output layer and was backpropagated to influence the modification of the recurrent and input weights. This additional layer raised the number of adaptable parameters to 320. The network was trained as a Simple Recurrent Network (SRN) (Elman, 1990), unrolling the recurrent hidden layer for a single time step. As in the previous architecture, noisy input patterns were presented at the input layer for the first time step only, and input activity was set to zero during the rest of the settling period. The same learning parameters were used, but training time was extended to 6000 epochs.

The performance of this configuration, shown in the center of Figure 4, is grim. These networks not only failed to generalize, but they often failed to form attractors for training set patterns. Also, several spurious attractors (as many as 8 for some training set sizes) arose for ill formed patterns. The introduction of an indirect error signal presented a serious obstacle to the formation of articulated attractors.

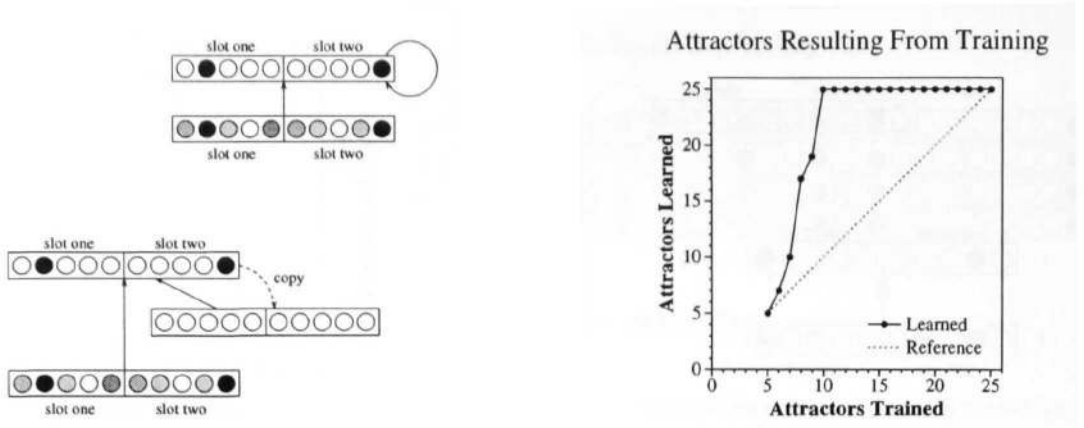


Figure 3: Network With Input Layer: The Architecture, Unrolled In Time, And Generalization Performance

In hopes of remedying this situation, the training procedure for this architecture was modified in a number of substantial ways. The first modification involved the number of time steps experienced by the network during training. An examination of the dynamics associated with well formed patterns revealed that, when presented with a valid pattern, the activation state of the network often drifted away from that well formed configuration, but it did so only very slowly. After 10 time steps of settling (which was the settling period during network training) almost all training set patterns appeared intact at the output layer. This observation suggested that the number of settling time steps experienced by the network during training was sufficient to keep the network from drifting away from the training patterns too quickly but was insufficient to construct the needed stable fixed-point attractors. To correct for this problem, we retrained these networks using incrementally larger settling times during training. In other words, whenever a network successfully retained the training patterns for t time steps during training, the settling time was advanced to $(t + 1)$ for the next training epoch. Unfortunately, this strategy did not work. Invariably, some settling time threshold would be reached, past which the networks would not learn.

Our next modification involved using a more robust estimate of the error gradient by backpropagating error through time all the way to the first time step. Using complete BPTT instead of the SRN training method showed no significant improvement by itself, but when coupled with a switch to a bipolar activation function (units which ranged in activation between -1 and 1) and with a reduced learning rate (0.001), this architecture began to successfully memorize the training set attractors. Systematic generalization remained elusive, however. This performance is shown in Figure 4, on the right.

In the previous two network architectures, the pattern of activation at the recurrent layer was consistently both polarized and sparse. Units tended to be either all the way "on" or all the way "off", and only two of the ten units were "on" for any given valid pattern. These properties of the recurrent layer activation patterns were directly enforced by the error signal provided at the output. In the case of an indirect error signal, however, these properties are no longer directly determined by training. Since the recurrent layer is a hidden layer in these networks, other patterns of activation are free to arise there. Indeed, the activation patterns at the recurrent hidden layers of these networks were quite distributed, with approximately half of the hidden units being highly positive for any given training pattern. These recurrent layer patterns still tended to be polarized, however, presumably because it is easier to construct stable fixed-point attractors in the corners of activation space.[1] Still, these networks apparently included a sufficient number of free parameters (weights) to associate a fairly arbitrary distributed hidden layer attractor with each training pattern. Unlike the dual winner-take-all structure learned by the previous two architectures, these attractors showed few signs of compositionality.

This problem of hidden layer representation is serious. It is quite possible for a network to learn a hidden layer encoding of input patterns which is consistent with the training items but is inherently incapable of generalizing to other valid patterns. This problem may be illustrated by the simple example show in Figure 5. This diagram displays a small piece of a network, including two hidden units and two output units. Two possible configurations of weights between these processing elements are shown, with the output bias weights always being slightly negative. Both configurations can produce the given training set targets at their outputs, but only the configuration on the left is capable of producing the generalization target. The weights in the right network fragment fail because they collapse too many distinct hidden layer patterns to single output patterns. For generalization to have any hope of occurring, hidden layer activation space must retain distinct correlates to the entire range of valid outputs.

One way to avoid this "collapsing" of hidden layer space is to drive the weight vectors *coming out of* each hidden unit towards mutual orthogonality. This constraint makes the con-

[1] Near the corners of the 10 dimensional activation space of the recurrent layer, the derivative of the sigmoidal activation function of each recurrent unit is close to zero. This means that a large weight change is typically needed to change the fixed-point of a corner attractor. By comparison, a fixed-point attractor in the middle of this 10 dimensional space may drift significantly as the result of even a small weight change. In general, with sigmoidal units, fixed-point attractors in the corners of activation space are much less sensitive to small perturbations in weight values than fixed-points in the middle of activation space.

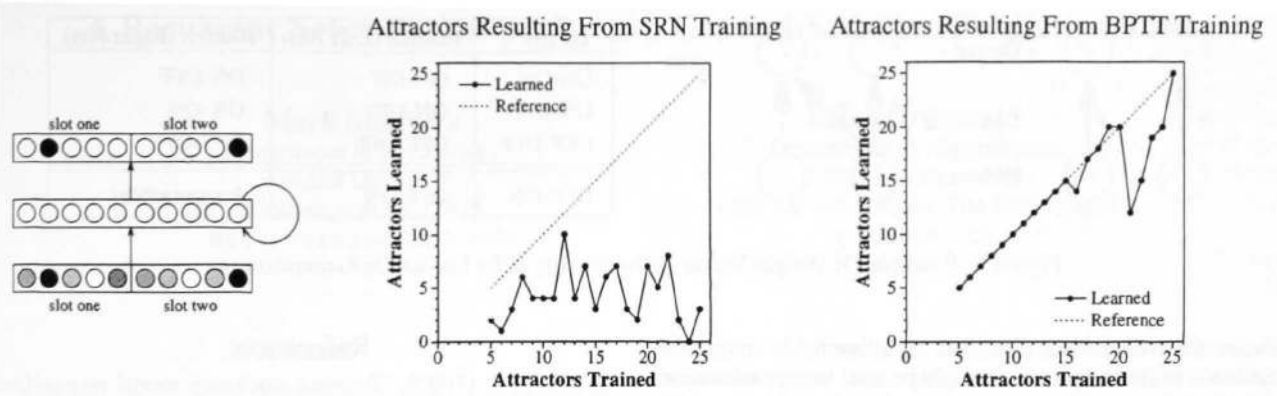


Figure 4: Network With Indirect Error: The Architecture, SRN Results, And BPTT Results

tribution of each hidden unit to the formation of an output pattern orthogonal to the contributions of the other hidden units. Note that the weight set schematically shown on the left in Figure 5, which effectively copies the hidden layer activation pattern to the output layer, is one example of a set of orthogonal outgoing weight vectors which is capable of appropriate generalization. To test this idea of an orthogonality constraint, we added a term to our squared error objective function of the form:

$$E_{\perp} = \sum_{a} \sum_{b} \cos^2 \theta_{ab}$$

... where a and b are hidden unit indices and θ_{ab} is the angle between their outgoing weight vectors. Unfortunately, depending on the proportion with which $E_{\perp}$ was mixed with squared error, orthogonalization either interfered with the learning of even training set patterns or had little effect at all. We noticed that the orthogonalization term often moved the hidden layer representations away from the corners of activation space, where attractors were typically constructed, so we also added a polarization error term which encouraged bipolar vectors at the hidden layer. This term took the form of:

$$E_p = -\sum_{a} o_a^2$$

... where o_a is the activation level of hidden unit a. Even when the objective function was augmented with both of these terms, the best networks still did little more than memorize training patterns.

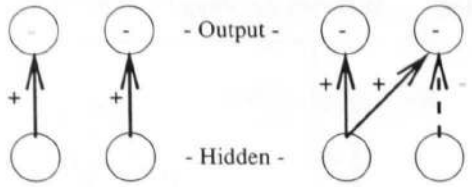
Discussion

These results suggest that systematic generalization may arise easily in recurrent attractor networks when they are presented with a direct error signal. Distal error signals, backpropagated through intervening weights, however, appear to present a profound obstacle to the formation of articulated attractors. This finding is disconcerting since many cognitive models incorporating recurrent attractor networks implicitly assume an error signal conceptually "backpropagated" through some other psychological process while, in actual simulations, they utilize an error signal applied directly to the recurrent layer (Mathis & Mozer, 1995; Plaut & McClelland, 1993). For the theories underlying these models to be valid, there must be some learning mechanism through which articulated attractors may be shaped by a distal teaching signal.

This problem may be viewed as one of finding a way to bias the learning of a multi-layer network in a way which encourages the general formation of articulated attractors without essentially "hard wiring" the structure of the input patterns. The main question that has yet to be answered is: What is the correct inductive bias for this task? We suggest that this bias should encourage recurrent hidden layer representations which use polarized activation levels and should drive hidden to output weights towards configurations which preserve, as much as possible, accessibility to the whole range of potential output patterns. Polarization is taken as a goal for the sake of the stability of attractor learning. Even with "corner attractors", however, these networks still need to avoid hidden to output mappings which restrict generalization. A technique such as activation sharpening (French, 1991) could potentially produce the kinds of representations needed, but this would require an *a priori* specification of the number of hidden elements "on" for each pattern. Still, an inductive bias of this sort may be the best that is possible under an indirect error signal.

Our future work will focus on solving this indirect error signal problem using two distinct approaches: by modifying the input pattern encoding and by modifying the network architecture. The first of these approaches involves encoding slot fillers in a non-localist fashion. Rather than assigning a single input and output unit to each filler, a more distributed representation could be used for filler values. This might involve a less sparse binary code in which different fillers share "on" elements, or it might involve a real vector encoding which retains the orthogonality of filler representations present in our localist code. Using a more distributed representation would cause weights *from* individual inputs and *to* individual output units to play a significant processing role over multiple filler values. The additional utilization of these weights may facilitate generalization to novel slot-filler patterns.

We will also consider encouraging articulated attractors by constraining the network architecture. In particular, we plan to investigate the possibility of initializing weights at the recurrent layer to a configuration which embodies a collection of winner-take-all networks. These will be implemented using a *softmax* constraint (Bridle, 1990), so backpropagated error can still successfully reach weights feeding into the at-



Target	Hidden (Left Net)	Hidden (Right Net)
ON-ON	ON-ON	ON-OFF
ON-OFF	ON-OFF	ON-ON
OFF-OFF	OFF-OFF	OFF-OFF
OFF-ON	OFF-ON	***Impossible!***

Figure 5: Example Of Weight Vector Orthogonality & Its Impact On Generalization

tractor network. Also, restricted receptive fields among the hidden to output connections might be used to approximate an orthogonality constraint on this mapping. Such strong architectural constraints may be necessary to consistently produce articulated attractors from distal error.

If further investigation reveals that the learning of systematic attractor structures from a distal teaching signal requires specific constraints on network architecture, cognitive models which utilize such attractor networks will need to assume some significant innate constraints on learning. This does *not* mean that an architecture specifically tuned to a particular task, such as reading aloud or proper production of verb tense, is necessary. The required innate constraints may simply involve the early presence of lateral inhibition between processing elements grouped into clusters or the existence of map-like structures arising from topologically regular connection patterns. The learning bias introduced by such general connection patterns may be all that is needed. Still, the work presented in this paper suggests that the simple presence of recurrent connections is not enough to produce systematic attractor dynamics. Learning to enforce "well formedness" constraints on internal representations may require somewhat structured network architectures.

Conclusion

Connectionist attractor networks have shown much promise as a mechanism for improving generalization performance by enforcing well formedness constraints on representations. Attractor networks which successfully learn a systematic collection of such constraints given a small training set are said to embody articulated attractors – meaningful attractor basins arising from the compositional interaction of explicitly trained stable fixed-points. We have shown that articulated attractors can readily arise in such networks when an error signal is applied directly to the recurrent processing elements. Distal error signals, however, pose surprisingly profound difficulties for such networks. A strong prior inductive bias, perhaps best seen as genetic in origin, towards compositional structure may be needed to produce articulated attractors at hidden layers in backpropagation networks.

Acknowledgements

The authors extend their thanks to the members of *Gary's & Eric's Unbelievable Research Unit (GEURU)* for their comments and suggestions on this work. Thanks are also due to three anonymous reviewers for their helpful advice concerning the clear presentation of this research.

References

Bridle, J. S. (1990). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems 2* (pp. 211–217). San Mateo, CA: Morgan Kaufmann Publishers.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2), 179–211.

French, R. M. (1991). Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 173–178). Hillsdale, NJ: Lawrence Erlbaum Associates.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79, 2554–2558.

Mathis, D. W. and Mozer, M. C. (1995). On the computational utility of consciousness. In Tesauro, G., Touretzky, D. S., and Leen, T. K. (Eds.), *Advances in Neural Information Processing Systems 7* (pp. 11–18). Cambridge, MA: MIT Press.

Noelle, D. C. and Cottrell, G. W. (1995). A connectionist model of instruction following. In Moore, J. D. and Lehman, J. F. (Eds.), *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society* (pp. 369–374). Hillsdale, NJ: Lawrence Erlbaum Associates.

Plaut, D. C. and McClelland, J. L. (1993). Generalization with componential attractors: Word and nonword reading in an attractor network. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society* (pp. 824–829). Hillsdale, NJ: Lawrence Erlbaum Associates.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986a). Learning internal representations by error propagation. In Rumelhart, D. E., McClelland, J. L., and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (ch. 8). Cambridge, MA: MIT Press.

Rumelhart, D. E., Smolensky, P., McClelland, J. L., and Hinton, G. E. (1986b). Schemata and sequential thought processes in PDP models. In Rumelhart, D. E., McClelland, J. L., and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (ch. 14). Cambridge, MA: MIT Press.

St. John, M. F. (1992). Leaning language in the service of a task. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (pp. 271–276). Hillsdale, NJ: Lawrence Erlbaum Associates.