APECS: A Solution to the Sequential Learning Problem

I.P.L. McLaren
Department of Psychology, University of Warwick, Coventry, CV4 7AL, UK.
Tel. 0203 523188, e-mail psrap@warwick.csv

Abstract

This paper contains some modifications to Back Propagation that aim to remove one of its failings without sacrificing power. Adaptively Parametrised Error Correcting Systems (APECS) are shown not to suffer from the sequential learning problem, and to be capable of solving EOR, higher order parity, and negation problems. This opens the way to development of connectionist models of associative learning and memory that do not suffer from "catastrophic interference", and may shed light on issues such as the episodic / semantic memory distinction.

The development of novel connectionist algorithms (Rumelhart, Hinton, and Williams, 1986; Ackley, Hinton, and Sejnowski, 1985) capable of driving learning in multi-layer networks was one of the major developments in cognitive science in the nineteen eighties. One of these algorithms, Back Propagation (Rumelhart, Hinton, and Williams, 1986) uses gradient descent to find mappings that capture input / output relationships, typically instantiated in feed-forward architectures. In so doing, it comes up against the sequential learning problem identified by McCloskey and Cohen (1989) and further analysed by Ratcliff (1990).

A general statement of this problem is that if a network employing Back Propagation is first taught one set of input / output relations, and then some other mapping is learnt whose input terms are similar to those first used in training, then a near complete loss of performance on the first mapping is observed on test. The new learning wipes out the old. This is not a necessary characteristic of the feed-forward architecture, because if training alternates between the two mappings, repeatedly teaching first one and then the other, eventually a solution is reached that captures both sets of input / output relationships. Thus, this "catastrophic interference", when new learning erases old, is only seen if the two mappings are learnt in sequence. This does not mean that this property of the learning algorithm can be ignored, however, as learning (in humans and networks) often takes place within a sequential format (eg see Ratcliff, 1990; Hinton and Plaut, 1987; Sejnowski and Rosenberg, 1987).

As a simple example of this general type of problem, consider modelling a paired-associate experiment (Barnes and Underwood, 1959) in which human subjects are required to learn a list (list 1) of eight nonsense syllable - adjective pairs to a criterion of 100%. That is, after some number of training trials, the subject is able to provide the correct adjectival response to each nonsense syllable stimulus. After learning list 1, the subjects learn list 2, which employs the same nonsense syllables as the first, but new adjectives paired with them. Training continues until subjects are near perfect on this list (>90%). They are then asked to recall the original list 1 adjectival responses for the

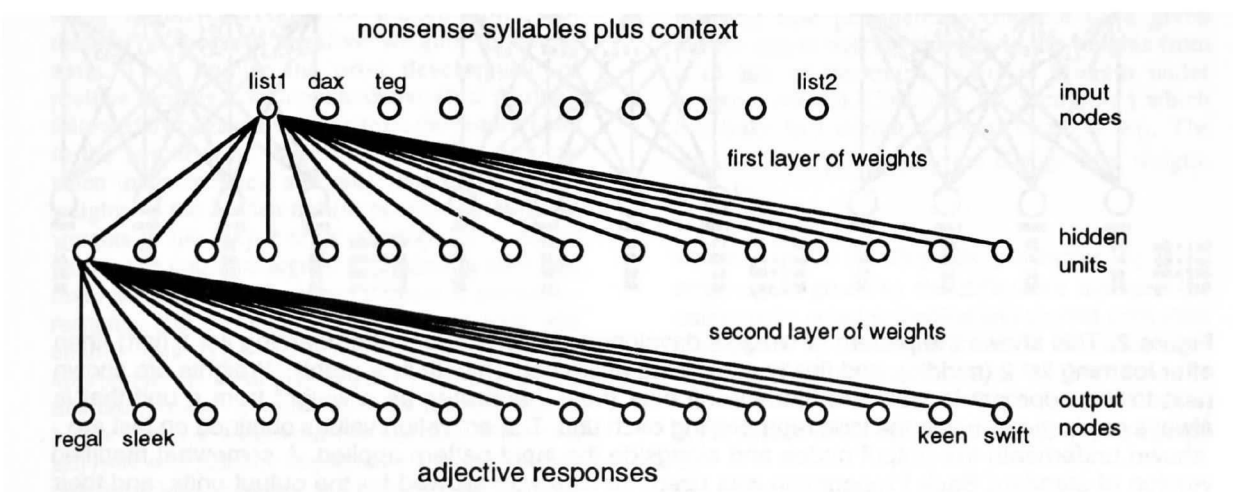

Figure 1. Each node in the input and output layers stands for some stimulus, context, or response; signalling its presence via its activity. Learning proceeds by changing the values of the connection strengths between nodes (called 'weights') so that the input nodes can transmit activation, via the hidden units, to the output nodes.

nonsense syllables. Performance drops to around 50% for this list, which is taken to be an instance of retroactive interference (control groups suggest that it is not simply the passage of time that is responsible for this decline in performance).

Following McCloskey and Cohen this task can be modelled in a feed-forward two layer network running Back Propagation. The list 'context' and the nonsense syllables (eg dax, teg) are the input and the adjectives (eg regal, sleek) are the output (see Figure 1). After cycling through the list several times, activation of the input nodes representing list context plus a nonsense syllable results in the activation of the output nodes corresponding to the correct adjective via the pattern of connection strengths or weights developed by the network. During learning of the second list, nodes standing for the list 2 context are used in conjunction with the old nonsense syllable nodes, together with extra output nodes representing the new adjectives (keen, swift). Training proceeds until activation of nodes representing list 2 + dax (for example) results in activation of the 'keen' node. Now, list 1 recall can be tested by presenting list 1 + dax as input. The result is - 'keen'. No sign of previously having learnt 'regal' to this input is evident. McCloskey and Cohen were able to show that even minimal training on list 2 resulted in (at best) nearly complete loss of list 1 on test, rather than the 50% loss shown in humans (at worst). This result does not depend on the local coding scheme employed here, as they obtained the same outcome using distributed representations of contexts, stimuli and responses as well. If, however, the network was alternated on the two lists, then it could achieve 100% performance on each list.

Figure 2 gives simulation results for this sequential learning task on a two item list using a modified version of Back Propagation (used throughout this paper). Despite these differences, the results parallel McCloskey and Cohen's.

After training on list 1 until performance meets their "within 0.1" criterion on test, ie activation of an input pattern produces the correct response to within 0.1 of each node's target activation level, learning list 2 to the same criterion destroys list 1 performance. In fact, testing on list 1 now fails to meet a "best match" criterion which requires that the output be more similar to the target response than to any of the other possible responses in the lists. Analysis of these simulation results indicates that the difficulty facing the network is that the initial list 1 solution (ie the pattern of weights) is not one that can survive learning list 2, because the list 1 responses to the nonsense syllables have to be suppressed in some fashion, and once this is done they cannot be recovered. Only when the lists are alternated can a list 1 solution that is protected from the effects of list 2 learning be found.

The solution to the sequential learning problem would hence seem relatively straightforward. Modify Back Propagation so that list 1 learning takes a form which survives list 2 learning by ensuring that list 1 responses do not pose problems during the learning of the second

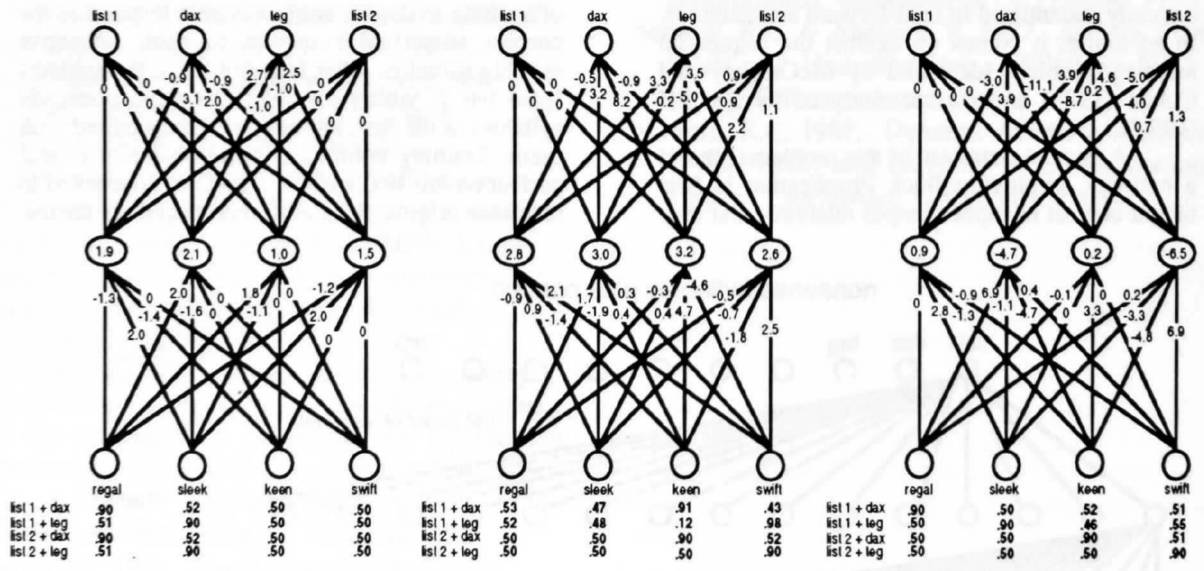

Figure 2. This shows the pattern of weights developed by the network after learning list 1 (left), then after learning list 2 (middle), and finally after being alternated on both lists (right). Weights are shown next to the appropriate link, and hidden unit bias (best interpreted as a weight from a unit that is always on) is given inside the icon representing each unit. The activation values obtained on test are shown underneath the output nodes and alongside the input pattern applied. A somewhat modified version of standard Back Propagation was used; no bias was allowed for the output units, and their target activation level when 'off' was taken as 0.5, corresponding to zero net input. Each list was cycled through 250 times (left, middle) or 500 times (right), with 100 weight changes per learning episode involving a given list pair .

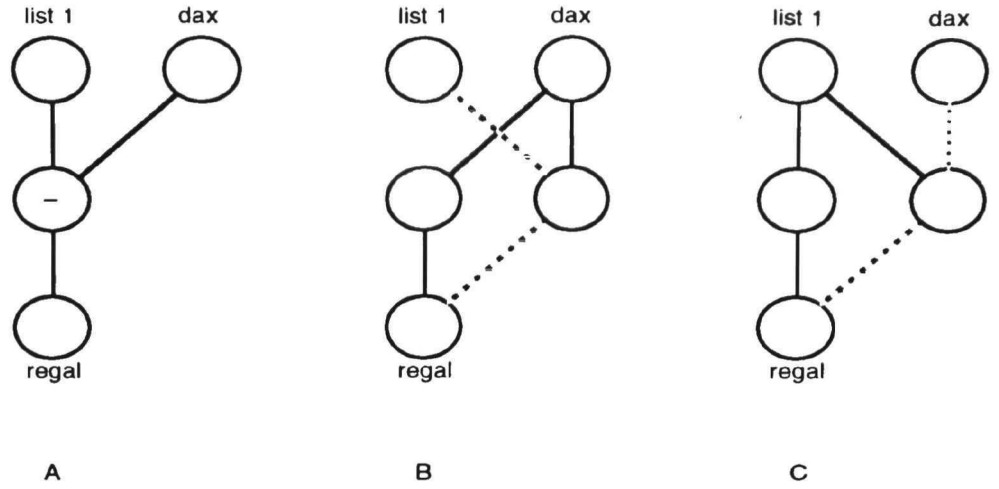

Figure 3. Three possible solutions to learning a list 1 item that would survive the learning of list 2. Solid lines denote positive weights, dashed lines negative weights. Solution A requires a negative bias on the hidden unit. In all cases, only if both input nodes are activated is there activation of the output node.

list. Inspection of the rightmost set of weights in Figure 2 indicates three simple list 1 solutions that might work, shown in Figure 3.

All of A, B, and C in Figure 3 seem plausible candidate solutions, in that none of them will produce much in the way of a response to a nonsense syllable input on its own. They each require the conjunction of nonsense syllable and list context to produce the appropriate adjective, and thus should not be susceptible to unlearning of list 1 during acquisition of list 2.

It is difficult to see how solutions B and C would develop, however, and a stricter version of the sequential learning problem poses difficulties for them. If learning on each list is in strict sequence, ie a pair is learnt and then the next pair and so on until the list is learnt without ever returning to a pair, then it would seem that those solutions which depend on cycling through a given list could not develop. This is the case for solutions using negative weights to hidden units that themselves possess negative weights to output units. They require the prior development of positive weights from input to output (first pairing), followed by negative weights from the hidden units to the output units (suppression of the response when inappropriate), and only then can negative weights to the hidden unit(s) possessing negative weights to the output units develop (on a second pairing). It could be argued that learning each pair involves alternating between context + stimulus - response and context - no response, which will enable this type of solution. This may be appropriate in modelling human performance on this task, though it seems unlikely that this has to be the case for successful list learning. Note that acquisition of pairs by sequentially learning context plus stimulus - response and context - no response (rather than alternating between them) will not suffice to arrive at solution B or C, and this is also a plausible model for human performance when learning a list in strict sequence (and the one used in the simulation reported later). In any case it would seem unwise to adopt solutions that suffered from this restriction.

Hence solution A, which can be characterised as an "AND" solution capturing the need for both list 1 context and dax to be input to give regal as an output, seems worthy of investigation. Given this, the problem becomes one of how to modify Back Propagation so as to achieve solution A without losing the desirable characteristics of this algorithm: in particular, the power to solve problems such as EOR.

One answer is to proceed by allowing some of the parameters of the Back Propagation learning algorithm to vary in an adaptive fashion, an idea borrowed from animal learning theory (eg Mackintosh, 1975; Pearce and Hall, 1980; M^{c}Laren and Dickinson, 1990). Specifically, the learning rate parameter applicable to a given hidden unit in making changes in the weights from it to any of the output units is brought under adaptive control. Consider a hidden unit j which has links to i output units of strength w_{ij}. The standard rule prescribing the change in a weight, ∂w_{ij}, is...

1. $\partial w_{ij} = f\Delta_i a_j$

where f is the learning rate parameter, Δ_i is an error signal given by the difference between the output units target activation and current activation multiplied by the derivative of the activation function with respect to input evaluated at the current value of the input to that output unit, and a_j is the hidden unit's activation. The proposed change is...

2. $\partial w_{ij} = f_j\Delta_i a_j$

so that each hidden unit has its own variable learning rate parameter. To achieve solution A, we

need presentation of an input / output pair to result in one (or a few) hidden units being selected to mediate the positive weights from input to output nodes. For arbitrary mappings such as those modelled here there is little point in dedicating more than one hidden unit to bind input to output, and it simplifies the next stage, which is to protect the mapping once it is established (the case of non-arbitrary mappings possessing statistical structure will be considered later). Protection is needed when only a part of the input (eg context or stimulus alone) is applied without concomitant application of the output previously paired with it (which will be termed "extinction" here), the weights formed in the previous learning episode(s) should not change much, rather the hidden unit's bias should increase so as to inactivate it. This means that the results of previous learning episodes are not so much frozen as taken out of circulation. This can be achieved if determination of the f_j is made competitive, so that the hidden unit whose connections best meet the demands imposed by current output activation is selected by having its f set high and the other f_j set near zero. The same mechanism can ensure that the f of this hidden unit is near zero during extinction, by disqualifying units receiving an appreciable negative error component from some output unit from the selection process. To promote shifts in bias, the parameters controlling changes in bias for the hidden units should be high when the error term is negative (ie during extinction), and low otherwise. The parameters controlling weight changes in the input to hidden layer should be high when the hidden unit error is positive, but low when negative, so that the mechanisms for raising and damping hidden unit activity are equal in strength. This gives three guiding principles to follow in modifying Back Propagation. The first is selection: picking out one (or a few) hidden unit(s) to mediate a mapping. This facilitates implementation of the second, protection: preventing unlearning in the hidden to output layer by reducing the learning rate parameter for that hidden unit to near zero. The third is the asymmetric parametrisation of learning in the input to hidden layer and for the hidden unit thresholds, so that extinction is mainly accomplished by bias changes rather than weight reduction in the input to hidden layer. Figure 4 gives the algorithm used for the APECS simulations reported here.

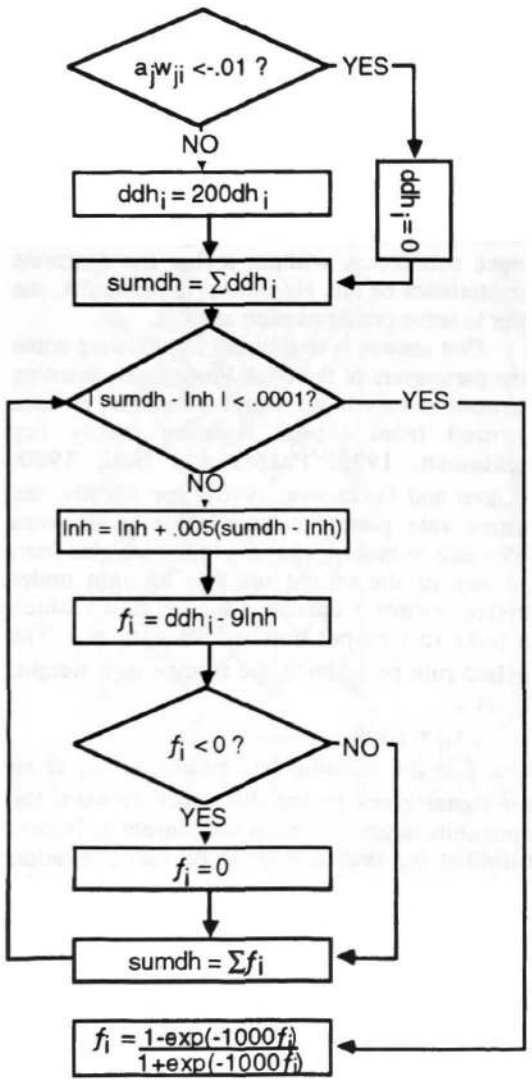

Figure 4. One possible algorithm for APECS. Start by setting $dh_i = (\Sigma \Delta_j w_{ji})(a_i)(1-a_i)$, where a_i is the activation of hidden unit i, so that dh_i is the error for hidden unit i. Now ddh_i = $200dh_i$ if there is no significant negative term (i.e. <-.01) in the sum of the raw errors propagated back from the output layer, it's zero otherwise; and sumdh = Σddh_i ,where the sum is over all hidden units (this initialises sumdh). The ddh_i compete to determine the f_i in an algorithm which is based on the idea of all the ddh_i inputting to a system which then provides negative feedback. Only the largest ddh_i will result in a non-zero f_i. The competition is done by setting a variable, Inh, to gradually converge to within .0001 of sumdh which itself is updated to equal Σf_i. This negative feedback system can then be used to select the largest of the ddh_i by updating the f_i to equal ddh_i - 9Inh. Any f_i that drops below zero is set to zero during this procedure. The system settles down to stable values for sumdh and Inh and only the largest ddh_i gives rise to a positive f_i. The final transform ensures that f_i changes rapidly over the range that matters, and is constrained between 0 and 1. This is a convenient algorithm to use, but no doubt there are many other methods of arriving at the same end result. At the end, the f_i are used to determine the parameters for learning between that hidden unit and the output layer ($0.25f_i$), for the bias ($10(1-f_i)$), and the input layer to that hidden unit ($15f_i$).

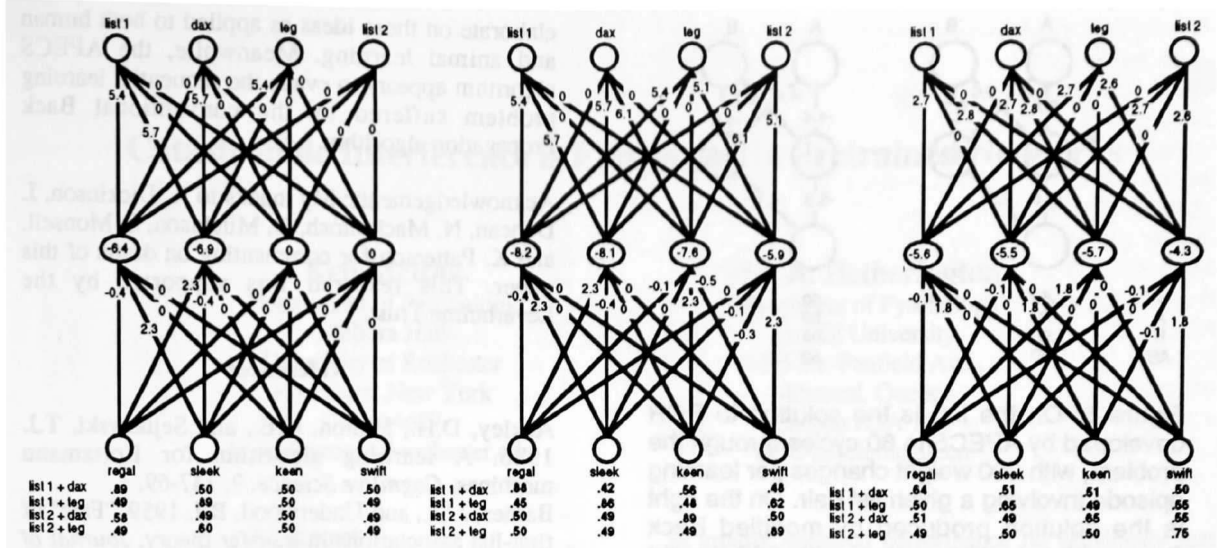

Figure 5. Left is the pattern of weights and results on test after learning list 1 (50 cycles through the list with 200 weight changes per learning episode involving a given list pair.)using APECS, in the middle are the results after learning list 2 (50 cycles through the list) as well. On the extreme right are shown the results for strictly sequential learning of both lists (8,000 iterations per list pair).

This Adaptively Parametrised Error Correcting System (APECS) can now be run on the sequential learning problem as defined by McCloskey and Cohen, using exactly the same procedures employed in the Back Propagation simulations reported earlier. The results are encouraging, and are shown in Figure 5.

After learning both lists, performance on test is perfect by the best match criterion in the two pair list simulation, and not far from meeting the .1 criterion. Though not shown in the figure, performance is even better in the eight pair list simulation, meeting the stringent .1 criterion in all cases. The reason for this seems to be that the longer list allows more complete extinction via bias becoming increasingly negative, minimising the contribution from other sources. In the two pair lists, the presence of a given stimulus whenever the other is not present introduces a strong negative contingency that the system exploits, and negative weights of some strength develop. In the eight pair lists this contingency is much weaker and no longer a significant factor. Performance in the two pair list condition can be improved by allowing presentations of the context for each list alone, which also degrades this negative contingency.

Having had some success with the sequential learning problem as defined by McCloskey and Cohen (1989), the next simulation tackles the strictly sequential version in which each input / output pair is learnt in one episode and never relearnt. The results are as shown in the rightmost panel of Figure 5. Note that the weights have developed appropriately but that the performance on test is weak, though still meeting the best match criterion. The output activations are attenuated because the thresholds have shifted so as to almost completely suppress hidden unit activity, even when optimal input patterns of activation are applied. This is a property of the activation function employed here, which requires a substantial change in bias during extinction, outweighing the influence of the positive weights developed during list pair learning. It cannot be avoided by some choice of parameters, though adopting a different activation function could help. This route will not be explored here, as the problem is one of performance rather than learning. The weights are appropriate, allowing a retrieval strategy, such as increasing input activation, to enhance performance.

The modified algorithm seems to handle sequential learning well, certainly a lot better than standard Back Propagation. But at what cost? Can APECS still solve problems that Back Propagation solved well? A wide ranging investigation is in progress, as one example the results for EOR will be given here. This is a suitable test case, as it is a standard problem requiring a multi-layer net that might be expected to pose difficulties for an algorithm that generates "AND" type solutions to problems. In fact, APECS solves EOR very well as is shown in Figure 6, where the solution produced by Back Propagation is also shown.

The APECS algorithm has also proved successful in solving Parity problems up to order 4, the Negation problem (Rumelhart, Hinton, and Williams, 1986), and several others that require the extraction of structure in some input - output mapping. Even if APECS could solve all the problems that Back Propagation can, however, one possible difficulty is that it might solve them in a way that is undesirable, ie by developing local representations at the hidden unit level. This is not a drawback in the case of arbitrary associations between input and output, APECS copes witharbitrary associations efficiently and well, but what of the case when distributed representation at the level of the hidden units might be considered appropriate?

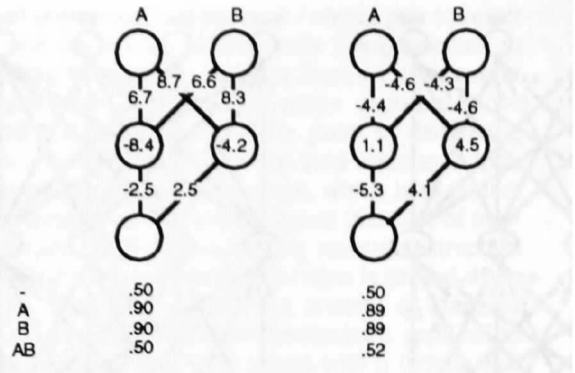

Figure 6. On the left is the solution to EOR developed by APECS in 60 cycles through the problem, with 200 weight changes per learning episode involving a given list pair. On the right is the solution produced by modified Back Propagation in 100 cycles, with 200 weight changes per learning episode involving a given list pair. Note that a unit has an activation of 0.5 given zero input, and thus will be somewhat 'on' unless turned 'off'. The bottom left shows which input nodes were active on test with the output node's response beneath the relevant solution. The "-" denotes no applied input.

In fact, APECS will develop distributed representations in some circumstances, though the results will not be given here. Imagine that if node A is active as input then node B is active as output, but this relation is embedded in noise, hence the input / output pairings are X_1A-Y_1B, X_2A-Y_2B and so on, where the Xs and Ys denote other nodes that are active. This models trying to learn a reliable relation in a variety of different contexts. Simulations show that the network develops a local representation (at the hidden unit level) for each learning episode in a different context, and as the learning episodes increase and are distributed across contexts, so the number of hidden units mapping this relation increases so that it is distributed across them. Each hidden unit contributes weakly to this relation, as each has been extinguished, and eventually the A-B relation becomes context-independent.

Hence a fundamental property of the APECS algorithm is an increase in the distributedness of representations with experience, from associations formed by a single context and 'episode' to associations accumulated over many such episodes, each somewhat different. This property may hold the key to understanding how one network may learn and represent both "episodic" and "semantic" knowledge (Tulving, 1972; 1983), and why the former is more vulnerable both to interference from other knowledge, and to neurological damage. It may also be possible to explain differences in the treatment of "rules" and "exceptions" in these terms (eg in modelling word recognition, Seidenberg and McClelland, 1989). Future publications will elaborate on these ideas as applied to both human and animal learning. Meanwhile, the APECS algorithm appears to evade the sequential learning problem suffered by the conventional Back Propagation algorithm.

Acknowledgements: My thanks to A. Dickinson, J. Duncan, N. Mackintosh, G. Mitchison, S. Monsell, and K. Patterson for commenting on drafts of this paper. This research was supported by the Leverhulme Trust.

References

Ackley, D.H., Hinton, G.E., and Sejnowski, T.J. 1986. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9, 147-69.

Barnes, J.M., and Underwood, B.J. 1959. "Fate" of first-list associations in transfer theory. *Journal of Experimental Psychology*, 58, 97-105.

Hinton, G.E., and Plaut, D.C. 1987. Using fast weights to deblur old memories. In *Program of the Ninth Annual Conference of the Cognitive Science Society* (pp. 177-86). Hillsdale. NJ. Erlbaum.

Mackintosh, N.J. 1975. A theory of attention: Variations in the associability of stimuli with reinforcement. *Psychological Review*, 82, 276-98.

M^{c}Laren, I.P.L., and Dickinson, A. 1990. The conditioning connection. *Phil. Trans. R. Soc. Lond.* B, 329, 179-86.

McCloskey, M., and Cohen, N.J. 1989. Catastrophic interference in connectionist networks: the sequential learning problem. In G.H. Bower (Ed.) *The psychology of learning and motivation.* NY. AP.

Pearce, J.M., and Hall, G. 1980. A model for Pavlovian learning: variations in the effectiveness of conditioned but not of unconditioned stimuli. *Psychological Review*, 87, 532-52.

Ratcliff, R. 1990. Connectionist models of memory: constraints imposed by learning and forgetting functions. *Psychological Review*, 97, 285-308.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J. 1986. Learning internal representations by error propagation. In D.E. Rumelhart and J.L.McClelland (Eds.) *Parallel Distributed Processing.* Vol.1. Cambridge. Mass. Bradford Books.

Seidenberg, M.S., and McClelland, J.L. 1989. A distributed developmental model of word recognition and naming. *Psychological Review*, 96, 523-68.

Sejnowski, T.J., and Rosenberg, C.R. 1987. Learning and representation in connectionist models. In M.S. Gazzaniga (Ed.) *Perspectives in memory research and training.* Cambridge. MA. MIT Press.

Tulving, E. 1972. Episodic and semantic memory. In Tulving, E. and Donaldson, W. (Eds.) *Organisation of memory.* NY. AP.

Tulving, E. 1983. *Elements of episodic memory.* NY. OUP.

Catastrophic Interference is Eliminated in Pretrained Networks

Ken McRae
Department of Psychology
Meliora Hall
University of Rochester
Rochester, New York
USA 14627
kenm@progidal.rochester.edu

Phil A. Hetherington
Department of Pyschology
McGill University
1205 Dr. Penfield Ave.,
Montreal, Quebec
Canada H3A 1B1
het@psych.mcgill.ca

Abstract

When modeling strictly sequential experimental memory tasks, such as serial list learning, connectionist networks appear to experience excessive retroactive interference, known as catastrophic interference (McCloskey & Cohen, 1989; Ratcliff, 1990). The main cause of this interference is overlap among representations at the hidden unit layer (French, 1991; Hetherington, 1991; Murre, 1992). This can be alleviated by constraining the number of hidden units allocated to representing each item, thus reducing overlap and interference (French, 1991; Kruschke, 1992). When human subjects perform a laboratory memory experiment, they arrive with a wealth of prior knowledge that is relevant to performing the task. If a network is given the benefit of relevant prior knowledge, the representation of new items is constrained *naturally*, so that a sequential task involving novel items can be performed with little interference. Three laboratory memory experiments (ABA free recall, serial list, and ABA paired-associate learning) are used to show that little or no interference is found in networks that have been pretrained with a simple and relevant knowledge base. Thus, catastrophic interference is eliminated when critical aspects of simulations are made to be more analogous to the corresponding human situation.

Introduction

Learning in standard, feed forward, back propagation networks (hereafter, standard networks) is a result of altering weights on connections between units via feedback or experience. Because patterns are represented in a distributed manner, many units and weights participate in encoding an item, enabling hidden units to capture meaningful relations among items. Because items are superimposed during learning, retroactive interference may occur; events occurring later in the training regime result in poor performance on previously-learned items (McCloskey & Cohen, 1989; Ratcliff, 1990). Although this is also true for humans, McCloskey and Cohen claim that interference in these networks is "catastrophic"; learning on later trials results in grossly impaired performance on previously learned items.

The sequential learning problem was demonstrated by Ratcliff (1990) in an attempt to model serial list learning in humans. In this task, a subject is sequentially presented with a number of items and is asked to recall them after the final item has been presented. In Ratcliff's simulation, 16 items were trained individually and sequentially. The network retained only the final item; performance on items 1 to 15 was very poor. Manipulations of various parameters, such as learning rate and momentum, did not significantly improve network performance. Similar failures to resolve the sequential learning problem by manipulating network parameters was reported by McCloskey and Cohen (1989).

In this article, we outline the major cause of catastrophic interference in standard networks, describe recent approaches to the problem, and present a novel approach. In contrast to previous work on the sequential learning problem that has manipulated network parameters and architecture (e.g., Hinton & Plaut, 1987; Kortge, 1990; Kruschke, 1992; Lewandowsky, 1991; Sloman & Rumelhart, 1992) we attempt to make the simulation more similar to the human situation that is being modeled. When this is done, catastrophic interference is eliminated.

Overlap at the hidden unit layer

Standard networks tend to unlearn catastrophically because there are too few constraints on hidden unit representations when learning initial items (French, 1991; Hetherington, 1991; Kruschke, 1992; Murre, 1992). Even completely non-overlapping stimulus patterns can overlap at the hidden unit layer (Hetherington, 1991). Most or all hidden units encode initial items, so later learning necessarily involves changing weights that encode previous items. If, however, a network was constrained to allocate a limited subset of hidden units to encode initial items, then overlap between old and new items would be reduced, decreasing interference.

There have recently been a number of proposals advanced to impose limits on the number of hidden units used to encode early items. In a standard network, the

receptive field of a hidden unit includes all input units. Thus, each hidden unit has equal probability of encoding two input patterns, maximizing the potential for interference. Kruschke (1992) used hidden units with local receptive fields to reduce overlap among items at the hidden unit layer. If a hidden unit is connected to only a subset of the input units, then the probability that two items are encoded by the same hidden unit decreases. In a variation of Kruschke's (1992) approach, French (1991) selectively "sharpened" a subset of the hidden units so that some became more active than others during initial learning, and fewer were used to encode each early item. Interference was reduced when a subset of hidden units were sharpened.

A natural solution

Catastrophic interference has been found in simulations of strictly sequential verbal learning experiments. Subjects in these experiments are typically college students who enter into an experiment possessing a large body of knowledge about words and their properties. If a network is pretrained so that it also possesses a body of knowledge prior to commencement of a sequential learning simulation, the first item (or short list of items) in a sequential learning task must be learned within the constraints resulting from prior knowledge. Many of the hidden units are already committed to representing only a limited aspect of the input space by virtue of strong weights from a limited number of input units. When a second item (or short list) is trained, interference with the first is decreased because the probability of overlap at the hidden unit layer is reduced. Thus, in accord with French's (1991) and Kruschke's (1992) ideas, our proposal also focuses on the state of the hidden unit layer at commencement of sequential training, but in contrast, we take advantage of the fact that training on a corpus of items *naturally* constrains hidden units' receptive fields and sharpens their activations.

A small, naive model

The first simulation is a simple demonstration of the effect of pretraining on the number of hidden units used to encode a pattern. An encoder network was used, with 8 input and output units, and 5 hidden units. Six distributed patterns were used. The first pattern was represented by turning on the first three units and setting the rest to zero, /11100000/, the second pattern by the next successive three units, /01110000/, and so forth. The initial weights were randomly selected from a uniform distribution, within the range ± 0.5. Training involved repeatedly presenting the fourth pattern until error fell below 0.01, at which point hidden unit activations were recorded. Figure 1 shows a typical run in which, after training pattern 4, each hidden unit contributed approximately equally to the output. Hence, as suggested above, without constraints on hidden unit activations, a standard network encodes a pattern across a large subset of its hidden units.

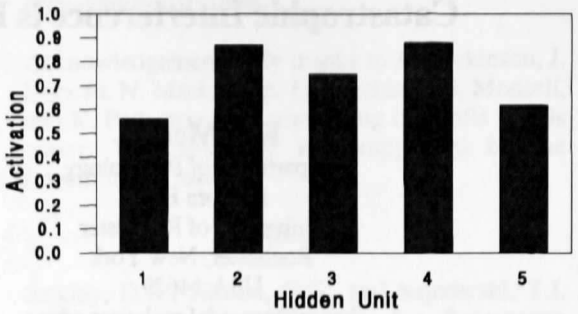

Figure 1. Representation of a single pattern across the hidden unit layer in a small, naive network. The pattern is distributed across all hidden units.

A small, pretrained model

The network was configured as in the previous simulation. Pretraining consisted of training all patterns except pattern 4 until mean error fell below 0.01. Pattern 4 was then trained to the same criterion. Figure 2 shows that only three hidden units participated in its encoding. Thus, like the constraints introduced by French (1991) and Kruschke (1992), pretraining reduced the number of hidden units used to encode an item.

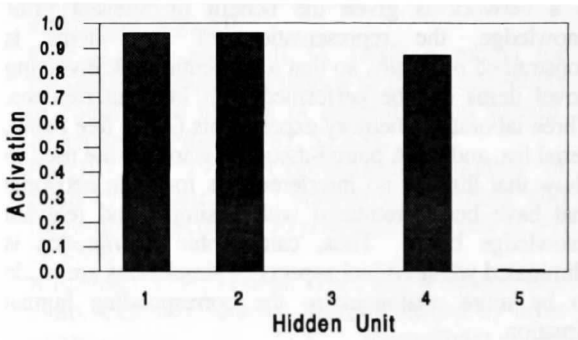

Figure 2: Representation of a single pattern across the hidden unit layer in a small, pretrained network. The pattern was encoded using only 3 of 5 hidden units.

This demonstration implies that if a network is constrained by pretraining it, then hidden units respond selectively to new input. Suppose that a network is made more analogous to adult humans by providing it with previous knowledge before subjecting it to a strictly sequential training regime associated with a recognition memory or serial list learning experiment. Given the demonstration above, previous training may enable a network to perform with little interference in strictly

sequential learning tasks. To test this prediction, larger simulations of free recall, serial-list learning, and paired-associate memory tasks were conducted, and are presented below. These simulations were not intended as detailed and accurate models of human memory experiments, but as simple models of different memory tasks, demonstrating that the effect of pretraining generalizes across tasks.

Pretraining a larger network

For three memory tasks (ABA free-recall, ABA paired-associate, serial-list), performance of naive and pretrained networks was compared. Networks were pretrained either on the orthographic representations of the set of 2897 English monosyllabic words used in the Seidenberg and McClelland (1989) model of naming, or on 1448 orthographic stimulus-response pairs constructed from words in that corpus. The input and output patterns were 400-unit wickelgraph representations of orthography (see Seidenberg & McClelland). A wickelgraph representation is a distributed representation of word spelling that encodes letter order and item similarity. For the free-recall and serial-list learning tasks, an encoder network was used; hence, input and output patterns were identical. Pretraining on the 2897 words continued for 44 epochs, until mean error fell below 10. At this point, the network correctly reproduced all words; the output for a word matched itself better than any of the other 2896. For the pattern associator, stimuli consisted of every second word (alphabetically) from the Seidenberg and McClelland corpus. A response was chosen randomly from among the 2897 items, with the constraint that no response occurred greater than three times. The network was pretrained for 40 epochs, at which point mean error was below 13 and it computed the correct response to each of the 1448 stimuli. In summary, in each simulation, a pretrained network that possessed knowledge of English orthography was compared to a naive network that was a tabula rasa.

For each simulation, the degree of hidden unit overlap is reported, followed by network performance on the list learning task. We compared the degree of hidden unit overlap between pairs of CVCs using a measure of percentage overlap ([1 - sum of absolute differences/sum of total activation] × 100). According to this measure, identical representations score 100, and non-overlapping representations score 0. Analyses of network performance will be described in each section.

Free-recall ABA

In an ABA free-recall memory task, a subject memorizes a list of items, such as consonant-vowel-consonant strings (list A), followed by a second list (list B). She is then asked to recall first list items in any order. An encoder network (400 input/output, 150 hidden units) was used to simulate this task.

The stimuli were consonant-vowel-consonant trigrams (CVCs) that were medium similarity items taken from a typical verbal learning experiment (Underwood, 1952). None of them were words from the pretraining corpus. The CVCs were randomly mixed to create five replications of A and B lists, each containing eight CVCs. During CVC training in this and the following two simulations, the momentum parameter in the McClelland and Rumelhart (1988) simulation software package, was set to zero. The first list was trained until mean error fell to approximately 18. At this point in training, the output for each pattern was closer to the target CVC than any of the other 2204 (21 × 5 × 20) possible CVCs. However, they were not as well learned as the pretrained words; CVCs learned in an experiment would not be as ingrained as a college student's knowledge of common English words.

Overlap

Similar to the smal simulations, the distributions of hidden unit activations were compared for sets of CVCs in the naive and pretrained networks. Figure 3 shows a histogram of hidden unit activations for YUG after it was trained to criterion along with seven other CVCs (list A). Note that the representation of YUG in the naive network was distributed fairly evenly across all hidden units. In contrast, in the pretrained network, almost all were inactive, with 38 of 150 fully activated. Thus, the naive network spread the representation of YUG across its hidden units, but the pretrained network used only a small subset. These results were not unique to YUG: the distribution of hidden unit activations averaged across CVCs was approximately uniform in the naive network, but bimodal in the pretrained case.

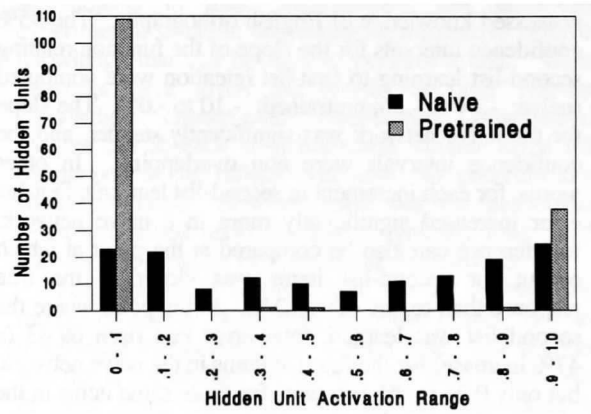

Figure 3. Distribution of hidden unit activation in representation of YUG in naive versus pretrained networks. Almost all hidden units encoded YUG in the naive network, but only a small minority in the pretrained network.

In the naive network, percentage overlap between JEP and YUG was 58%, and 82% between JEP and ZEP. In contrast, in the pretrained model, percentage overlap between JEP and YUG was 24%, and 58% between JEP and ZEP. The mean hidden unit overlap between pairs of CVCs in the naive model (65%, SD = 6%) was significantly greater than in the pretrained model (36%, SD = 8%), $t(27) = 34.34$, $p < .0001$. Note, also, that as well as reducing overlap, pretraining preserved similarity information (i.e., the hidden unit representation of JEP was more similar to ZEP than to YUG).

Interference

Given that a network with encoded knowledge of English represented CVCs with less overlap, interference in a free-recall ABA simulation should be drastically reduced. After the first list of 8 CVCs was trained, error was recorded for each pattern in both lists as a second 8-item list was trained (list B). Because the second list was learned at different rates in the naive and pretrained networks, it was inappropriate to directly contrast amount of interference at each recorded epoch. The first list could have been unlearned faster because second list learning was faster. To compare interference in the naive and pretrained networks, we required a measure of interference on first-list items given the degree to which second-list items had been learned. Therefore, linear regression analyses were conducted in which total error for second-list items was used to predict total error for first-list items. The slope of the regression line indicates the rate at which first-list error increased relative to the decrease in second-list error (i.e., first-list interference in terms of second-list learning). In both the naive and pretrained cases, second-list error predicted first-list error (naive: $r^2 = .97$; pretrained: $r^2 = .91$). Critically, significantly less interference obtained when a network possessed knowledge of English orthography. The 95% confidence intervals for the slope of the function relating second-list learning to first-list retention were computed (naive: -.34 to -.26; pretrained: -.10 to -.06). The slope for the naive network was significantly steeper, and the confidence intervals were non overlapping. In other words, for each increment in second-list learning, first-list error increased significantly more in a naive network. Interference can also be compared at the point at which output for second-list items was closer to the true response than to any other CVC. At the point where the second list was learned, total error had risen 69.43 (a 47% increase) for the first-list items in the naive network, but only 0.21 (a 0% increase) for those same items in the pretrained network. In summary, when an encoder network possessed knowledge of English orthography prior to an ABA free-recall simulation, catastrophic interference was eliminated.

Serial-list learning

The identical network and stimuli were used to simulate serial-list learning. In a free-recall serial-list learning task, subjects are presented with a number of items sequentially, and are asked to recall them in any order.

Overlap

Analogous to the previous simulation, YUG, when trained individually until its error fell below 18, was represented over the hidden units in an approximately uniform manner in the naive network, but by only 38 units in the pretrained network. Mean overlap between CVC pairs was significantly greater in the naive (65%, SD = 3%) than in the pretrained network (37%, SD = 9%), $t(119) = 35.03$, $p < .0001$. For example, when YUG was learned, followed by NOL, their representations overlapped by 60% in the naive network, compared to 29% in its pretrained counterpart. Thus, in terms of retention of YUG, there was greater potential for interference in the naive network. Interestingly, pretraining also preserved the role of item similarity in modulating interference. To illustrate, FAX and FAH, two highly similar CVCs, overlapped 68% in the naive network, barely above mean overlap (65%). In contrast, they overlapped 71% in the pretrained network, well above the mean (37%). Thus, although similarity between items plays little or no role in a network with an unconstrained hidden unit layer (see Hetherington, 1991), it plays a role when new information must be incorporated into an existing knowledge base. Thus, while the potential for interference has almost been eliminated, generalization has not (see Ratcliff, 1990).

Interference

To simulate serial list learning, five orders of the 16 CVCs were prepared. For each replication, they were sequentially trained to criterion (error less than 18). Following training on the final pattern, all CVCs were tested for retention. Error by serial position, averaged across the five replications, is displayed in Figure 4. With the naive network, a replication of Ratcliff's (1990) results obtained; retention was very poor for all but the final item. In contrast, all patterns were well retained when pretraining was provided. The performance of the naive and pretrained networks was compared using a t-test. Across serial position, interference in the pretrained network was significantly less than in the naive network, $t(15) = 13.08$, $p < .0001$. In fact, it may be that performance of the pretrained network was too good; there was less interference than is typically found in human serial-list learning experiments.

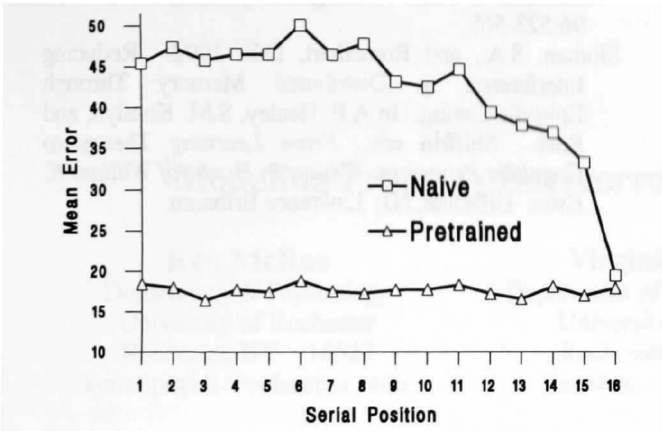

Figure 4. Mean performance of naive versus pretrained networks in the serial list-learning task. The upper (naive) and lower (pretrained) lines represent performance after all items were trained.

Paired-associate learning

In an ABA paired-associated task, a subject learns a list of stimulus-response pairs (list A), followed by a second list (list B), and then is asked to recall the first list. The importance of conducting both free-recall and paired-associate simulations was illustrated by Hetherington (1990, 1991). A factor such as item similarity that decreases interference in the simulation of one task may increase it in another. Thus, the ABA pattern association task was simulated to demonstrate that the effect of pretraining on reduction of interference is independent of task (i.e., type of network).

The paired-associate ABA simulation was conducted in the same manner as its free-recall counterpart. However, because the network learned arbitrary associations, a greater number of hidden units were required; 600 hidden units were used to encode 1448 word pairs. Sixteen random CVC-CVC stimulus-response pairs were created for the ABA simulation. They were randomly mixed to create five pairs of A and B lists, each containing eight patterns. For each replication, the first list was trained until mean error fell to approximately 18.

Overlap

The distributions of hidden unit activations for items in this simulation were approximately uniform in the naive networks, but bimodal in the pretrained network. BAB-BIX, for example, activated only nine hidden units greater than .9 in the pretrained network. Mean overlap between CVC pairs was significantly greater in the naive (62%, SD = 2%) than in the pretrained network (19%, SD = 6%), $t(27) = 35.90$, $p < .0001$.

Interference

During second-list training, error was recorded for patterns in both lists. As in the free-recall ABA simulation, in both the naive and pretrained networks, second-list error predicted first-list error (naive: $r^2 = .91$; pretrained: $r^2 = .86$). Interference was significantly reduced when a network was pretrained. The 95% confidence intervals for the slope of the function relating second-list learning to first-list retention were computed (naive: -.24 to -.15; pretrained: -.08 to -.04). The slope for the naive network was significantly steeper, and the confidence intervals were non overlapping. That is, in a pretrained network, for each increment in second-list learning, first-list error rose significantly less. Interference was also compared at the point at which the output for all second-list items was closer to the target response than to any other CVC. At the point where the second list was learned, error had risen 120.71 (an 82% increase) for the 8 items in the naive network, but only 14.15 (a 9% increase) in the pretrained case. Thus, when a standard network possessed knowledge of English orthography prior to a paired-associate ABA simulation, interference was drastically reduced. Again, interference may been reduced to below what is found with humans in paired associate ABA experiments.

Summary and conclusion

The three previous simulations demonstrated that interference was drastically reduced in simulations of verbal learning experiments using a simple and principled manipulation. College students typically serve as subjects in these experiments, and enter the experimental situation possessing an impressive body of knowledge that can be used to perform a task. When networks were given the benefit of prior relevant knowledge, catastrophic interference disappeared. Despite the absence of interference, similar items still overlapped at the hidden unit layer, so generalization should still occur. However, there was less interference in the pretrained networks than is typically found with humans in analogous verbal learning experiments.

The simulations reported here are not sophisticated models of human memory experiments; in fact, they are insufficient implementations of each task. Our purpose was not to capture the variety of behavioral effects seen in these tasks, but merely to use the simulations to demonstrate that catastrophic interference in standard networks can be alleviated using a simple and principled manipulation. The choice of specific networks and procedures was primarily motivated by a desire to deviate little from the simulations of McCloskey and Cohen (1989) and Ratcliff (1990). It is left for future

work to produce more sophisticated and realistic simulations of ABA free-recall, ABA paired-associate, and serial-list learning tasks.

References

French, R.M. 1991. Using Semi-Distributed Representations to Overcome Catastrophic Interference in Connectionist Networks. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society,* 173-178. Hillsdale, NJ: Lawrence Erlbaum.

Hetherington, P.A. 1990. Interference and Generalization in Connectionist Networks: Within-domain structure or between-domain correlation? [Review of O. Brousse & P. Smolensky (1989). *Virtual Memories and Massive Generalization in Connectionist Combinatorial Learning.*] *Neural Network Review* 4(1): 27-29.

Hetherington, P.A. 1991. The Sequential Learning Problem in Connectionist Networks. Master's Thesis, Department of Psychology, McGill University.

Hinton, G.E., and Plaut, D.C. 1987. Using Fast Weights to Deblur Old Memories. *Proceedings of the Ninth Annual Conference of the Cognitive Science Society,* 177-186. Hillsdale, NJ: Lawrence Erlbaum.

Kortge, C.A. 1990. Episodic Memory in Connectionist Networks. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society,* 764-771. Hillsdale, NJ: Lawrence Erlbaum.

Kruschke, J.K. 1992. ALCOVE: An Exemplar-Based Model of Category Learning. *Psychological Review* 99:22-44.

Lewandowsky, S. 1991. Gradual Unlearning and Catastrophic Interference: A Comparison of Distributed Architectures. To appear in W.E. Hockley and S. Lewandowsky eds., *Relating Theory and Data: Essays on Human Memory in Honor of Bennet B. Murdock.* Hillsdale, NJ: Lawrence Erlbaum.

McClelland, J. L., and Rumelhart, D. E. 1988. *Explorations in parallel distributed processing: A handbook of models, programs, and exercises.* Cambridge MA.: MIT Press.

McCloskey, M., and Cohen, N.J. 1989. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In G.H. Bower ed., *The Psychology of Learning and Motivation* 24:109-164. New York: Academic Press.

Murre, J.M.J. 1992. *Learning and Categorization in Modular Neural Networks.* Hillsdale, NJ: Lawrence Erlbaum.

Ratcliff, R. 1990. Connectionist Models of Recognition Memory: Constraints Imposed by Learning and Forgetting Functions. *Psychological Review* 97:285-308.

Seidenberg, M.S., and McClelland, J.L. 1989. A Distributed Developmental Model of Word Recognition and Naming. *Psychological Review* 96:523-568.

Sloman, S.A., and Rumelhart, D.E. 1992. Reducing Interference in Distributed Memory Through Episodic Gating. In A.F. Healey, S.M. Kosslyn, and R.M. Shiffrin eds., *From Learning Theory to Cognitive Processes: Essays in Honor of William K. Estes.* Hillsdale, NJ: Lawrence Erlbaum.