Pseudo-recurrent connectionist networks: An approach to the "sensitivity–stability" dilemma

(French, R. M. (1997) *Connection Science, 9*(4). 353-379)

Robert M. French
Department of Psychology
University of Liège, Belgium
rfrench@ulg.ac.be

Abstract

In order to solve the "sensitivity-stability" problem — and its immediate correlate, the problem of sequential learning — it is crucial to develop connectionist architectures that are simultaneously sensitive to, but not excessively disrupted by, new input. French (1992) suggested that to alleviate a particularly severe form of this disruption, catastrophic forgetting, it was necessary for networks to dynamically separate their internal representations during learning. McClelland, McNaughton, & O'Reilly (1995) went even further. They suggested that nature's way of implementing this obligatory separation was the evolution of two separate areas of the brain, the hippocampus and the neocortex. In keeping with this idea of radical separation, a "pseudo-recurrent" memory model is presented here that partitions a connectionist network into two functionally distinct, but continually interacting areas. One area serves as a final-storage area for representations; the other is an early-processing area where new representations are first learned by the system. The final-storage area continually supplies internally generated patterns (pseudopatterns, Robins (1995)), which are approximations of its content, to the early-processing area, where they are interleaved with the new patterns to be learned. Transfer of the new learning is done either by weight-copying from the early-processing area to the final-storage area or by pseudopattern transfer. A number of experiments are presented that demonstrate the effectiveness of this approach, allowing, in particular, effective sequential learning with gradual forgetting in the presence of new input. Finally, it is shown that the two interacting areas automatically produce representational compaction and it is suggested that similar representational streamlining may exist in the brain.

Introduction

One of the most important problems facing connectionist models of memory — in fact, facing *any* model of memory — is how to make them simultaneously sensitive to, but not disrupted by, new input. This problem is often referred to as the "sensitivity-stability" dilemma (D. O. Hebb, 1949) or the "stability-plasticity" problem (Carpenter & Grossberg, 1987). It is particularly relevant for connectionist networks, especially since they can be afflicted by a particularly severe manifestation of the sensitivity-stability problem known as catastrophic interference. Catastrophic interference is the tendency of neural networks to abruptly and completely forget previously learned information in the presence of new input (McCloskey & Cohen, 1989; Ratcliff, 1990).

Connectionist networks and lookup tables lie at opposite ends of the stability-sensitivity spectrum. While the latter are completely stable in the presence of new information, they lack the crucial ability to generalize on new input or to function effectively with degraded input. By contrast, standard backpropagation networks are highly sensitive to new input because of their highly distributed, overlapping internal representations. Internal representations of this kind are responsible for these networks' much touted abilities to generalize on previously unseen input, but they are also responsible for the radical loss of old information when learning new information.

Because of the sensitivity-stability problem, *all* of the patterns given to a connectionist network must be learned "concurrently". In other words, the entire set of patterns to be learned must be presented over and over, each time adjusting the weights of the network by small increments until the network gradually finds a weight-space solution for the entire set of patterns. This a far cry from how humans learn a series of patterns, however. Much of human learning tends to be *sequential*. A particular pattern is learned, then another, and another, and so on. While some of the earlier patterns may be seen again, this is not necessary for them to be retained in memory. As new patterns are learned, forgetting of old, unrepeated patterns occurs gradually as a function of time.

In this paper, a connectionist memory model will be presented that is capable of effective sequential learning, exhibiting gradual forgetting where a standard backpropagation architecture forgets catastrophically. The proposed architecture relies on separating the previously learned representations from those that are currently being learned. Further — and crucially — a method is described in which an *approximation* of the previously learned data (not the original patterns themselves, which the network will not see again) will be extracted from the network and will be mixed in with the new patterns to be learned.

Distributed models that are sensitive and stable in the presence of new information

It is not the case that all models that rely on distributed, overlapping representations forget catastrophically in the presence of new information. In particular, the class of convolution-correlation models (e.g., CHARM (Metcalfe, 1982) and TODAM (Murdock, 1983)), and Sparse Distributed Memory (SDM) (Kanerva, 1989) can learn new information in a sequential manner and can, in addition, generalize on new input. The performance of these models on previously learned information declines gradually, rather than falling off abruptly, when learning new patterns. One might argue that CHARM, TODAM, and SDM are not "connectionist" models. While, strictly speaking, this may be true, convolution-correlation models are readily shown to be isomorphic to sigma-pi connectionist models and SDM has been shown to be isomorphic to a Hopfield network (Keeler, 1988).

How do these models achieve stable, sensitive performance in the presence of new input? All of them, in one way or another, rely on two principles: i) representational separation and ii) explicit use of previously stored representations to influence the course of new learning. In the case of convolution-correlation models, representational separation is achieved by orthogonal recoding of the input and SDM does so by the use of sparse coding. In this way, representational overlap between newly arriving and previously stored patterns is reduced. But in order to produce the desired abilities to generalize, previously stored information is used to affect the memory trace of incoming information. For example, in both CHARM and SDM the new input vector is "folded into" an internal representation of the previously learned input. These two principles form the basis of the pseudo-recurrent architecture proposed in this paper.

The need for and consequences of separating old and new representations

Most approaches to reducing catastrophic interference in traditional connectionist architectures have relied on reducing the overlap of representations either by orthogonal recoding of the input patterns (Kortge, 1990; Lewandowsky and Goebel, 1991; Lewandowsky and Shu-Chen, 1993), or, alternately, by orthogonalizing the network's hidden layer representations (French, 1992, 1994; Murre, 1992; Krushke, 1993; McRae & Hetherington (1993)). A thorough discussion of these techniques and an analysis of the underlying causes of catastrophic interference can be found in Sharkey & Sharkey (1995).

McClelland, McNaughton, and O'Reilly (1995) have argued that radical separation of

representations might have been the approach arrived at by evolution in the development of the hippocampus and the neocortex. They justify the brain's bi-modal architecture as follows:

> "The sequential acquisition of new data is incompatible with the gradual discovery of structure and can lead to *catastrophic interference* with what has previously been learned. In light of these observations, we suggest that the neocortex may be optimized for the gradual discovery of the shared structure of events and experiences, and that the hippocampal system is there to provide a mechanism for rapid acquisition of new information without interference with previously discovered regularities. After this initial acquisition, the hippocampal system serves as a teacher to the neocortex..."

But this stills leaves unanswered a crucial question— namely: *How* does the neocortex acquire new information, whether it comes from the hippocampus or elsewhere, without disrupting information already stored there?

McClelland, McNaughton, and O'Reilly distinguish between *focused* and *interleaved* learning. In focused learning, new information is simply presented to the system, perhaps a number of times, but without interleaving it with old, previously acquired knowledge. In interleaved learning, the new knowledge is interleaved with the rest of the database of already acquired knowledge. They show that significant disruption of previously acquired information (very much like catastrophic interference) occurs in focused learning.

Their solution involves the very gradual incorporation of the new information from the hippocampus into the neocortical structure (i.e., long-term memory). Hippocampal representations very gradually train the neocortex. This is very likely correct. The problem, as we will show below, is that no matter how slowly the hippocampal information is passed to the neocortex, radical forgetting of the old information can result, *unless a way is found to interleave the already stored neocortical patterns* with the new patterns being learned. This interleaving cannot always use "the rest of the [original] database" of previously learned patterns because many of these patterns will no longer be explicitly available for re-presentation to the network. Once we have learned about penguins and wish to learn about, say, mice or tractors, we do not have to continue to be shown penguins so as not to forget them. Unfortunately, this *is* necessary for standard backpropagation networks to prevent forgetting of previously learned patterns. When learning new information, the previously learned information must once again be explicitly presented to the network, otherwise it may be completely forgotten.

The "pseudo-recurrent" memory model proposed in this paper will provide a way to automatically refresh the network without recourse to the original patterns. Instead of the original patterns, internally-produced *approximation*s of these patterns, called pseudopatterns (Robins, 1995), will be used and interleaved with the new patterns to be learned. The architecture proposed will argue for two functionally distinct areas of long-term memory: one, an "early-processing area" in which new information will be initially processed and a second, a "final-storage area," in which information will be consolidated. This model of long-term memory will suggest a means of consolidating information in long-term memory that has a number of compelling neurobiologically parallels. Some of these are detailed in Robins (1996). He discusses, in particular, parallels between consolidation of learning during sleep and pseudorehearsal (i.e., the use of pseudopatterns). Two of these parallels are of particular interest here:

Preserving the structure of old information

> While operating in different domains, both pseudorehearsal and sleep consolidation have the same underlying function – the integration of new

information into old information while preserving the structure of the old information.

Rehearsing approximations of old information

While it is an effective method in ANNs, rehearsal is unlikely to be a realistic model of biological learning mechanisms, as in this context the actual old information (accurate and complete representations of all items ever learned by the organism) is not available. Pseudorehearsal is significantly more likely to be a mechanism which could actually be employed by organisms as it does not require access to this old information, it just requires a way of approximating it. (Robins, 1996).

In addition, the pseudo-recurrent network architecture, relying on continual interaction of an early-processing area and the final-storage area via pseudopatterns, will produce precisely the type of semi-distributed representations that French (1992, 1994) felt were essential to reduce catastrophic interference. Unlike French (1992, 1994), however, in which specific learning algorithms were incorporated to produce the semi-distributed representations that alleviated catastrophic interference, in the case of the pseudo-recurrent network, these internal semi-distributed representations emerge naturally.

In an attempt to relate this model to other well-known high-level phenomena, we will also show that this model, unlike standard backpropagation, exhibits a plausible list-length effect (Strong, 1912; Atkinson & Joula, 1973; Ratcliff & Murdock, 1976, Gillund & Shiffrin, 1984). In addition, it also shows a list-strength effect (Murnane & Shiffrin, 1991; Ratcliff, Clark, & Shiffrin, 1990; Shiffrin, Ratcliff, & Clark, 1990).

Training in standard error-driven connectionist networks

The ideal way, of course, to solve the stability-sensitivity problem would be to store all previously learned patterns "out of reach" of the disruptive influence of new patterns. Then, when new input was presented to the system, all of the previously learned patterns would be taken out of storage, so to speak, and would be mixed with the new patterns. The system would then learn the mixed set of old and new patterns. After the augmented set of patterns had been learned by the network, they would all be put in storage, awaiting the next time new information was presented to the network. There would be no forgetting, catastrophic or otherwise, in this ideal world and there would be no deleterious effect on the network's ability to generalize, categorize or discriminate. The system could learn the new patterns and the old patterns would be unaffected by the acquisition of this new information.

Unfortunately, this way of learning new data is rarely possible in the real world except in the most artificial situations. It is in essence impossible to explicitly store all, or even a small fraction of previous training exemplars for future learning. True, one could argue that they are "stored" in the environment around us. My internal representation for, say, "car" or "child" is constantly being interleaved with any new input that I learn since these exemplars are continually available in the environment. But "my grandfather's kitchen table" is not. It is an object — a perfectly ordinary handmade wooden table — that I have not seen in twenty years, but that I nonetheless still remember with vivid clarity. Since I have seen literally thousands of tables (and all manner of other objects) since that time and if interleaved pattern presentation were necessary to avoid forgetting, then I should have a great deal of trouble remembering his kitchen table. Why has this memory remained intact in the absence of any interleaved (real) input from the environment? I suggest that *an internal approximation* of the original pattern is generated in long-term memory and it is this approximation that, in the

absence of the real pattern-in-the-environment, serves to continually reinforce the long-term memory trace of the original pattern.

In summary, in order to avoid severe interference in long-term memory, new input must be mixed with some approximation of the originally learned patterns. The better the approximation, the less the interference. In the ideal case where the originally learned patterns are still available for interleaved presentation, forgetting is eliminated altogether. As we have seen, sometimes the original patterns are available (as in the case of "car" or "child") but, more often, internally generated approximations of the original patterns must suffice. This paper presents an implementation of a long-term memory model that uses internally generated pseudopatterns as the means of mixing old and new information. This model is shown to be capable of effective sequential pattern learning and produces gradual forgetting rather than catastrophic forgetting. The use of pseudopatterns to improve performance of connectionist networks on catastrophic forgetting was first proposed by Robins (1995) and their plausibility has been further explored in Frean & Robins (1996) and Robins (1996).

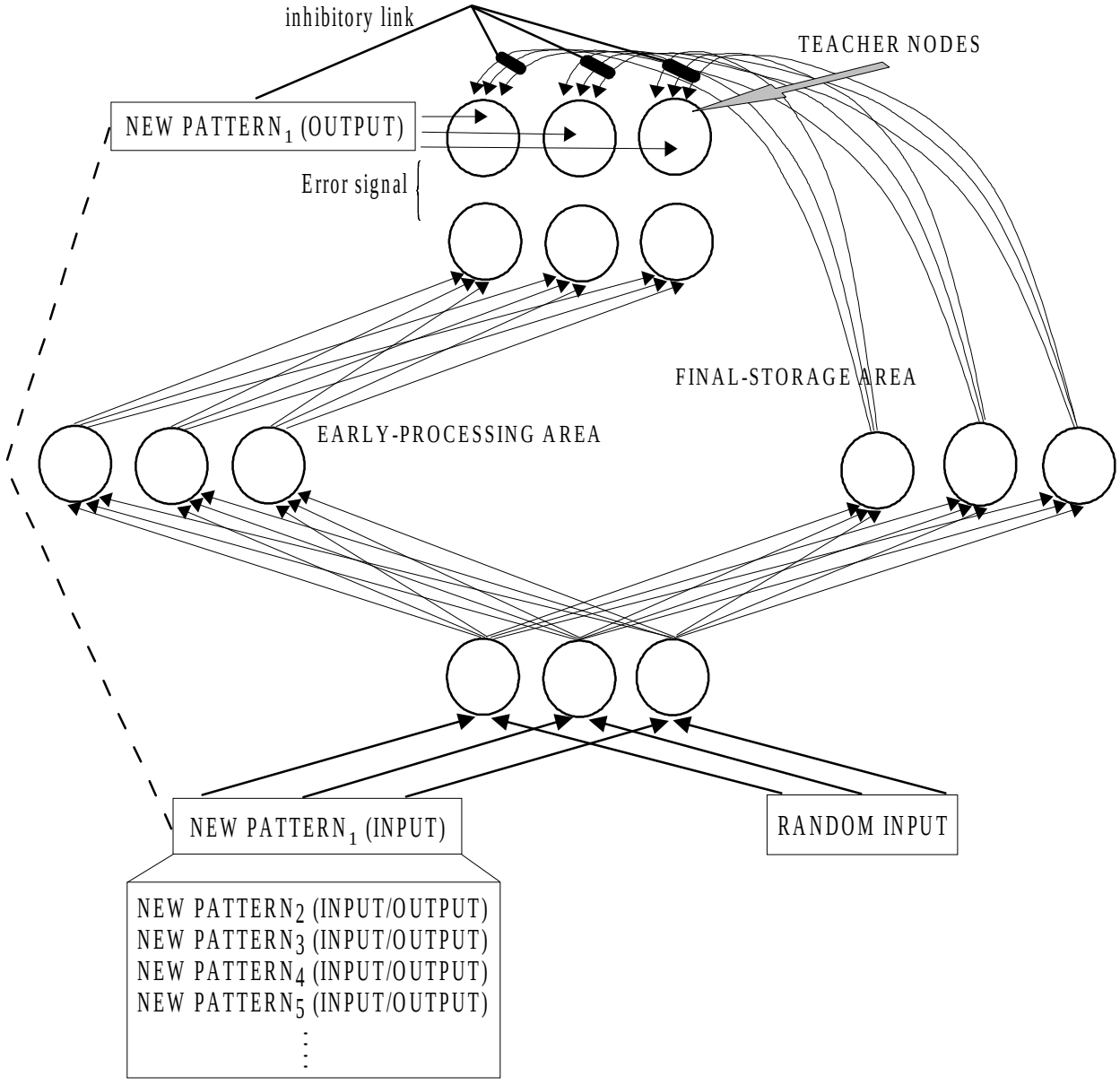

Figure 3. The architecture of the pseudo-recurrent network

The architecture of the "pseudo-recurrent" memory model

The architecture discussed in this paper consists of a feedforward backpropagation network that is divided into two parts, one used to help train the other (Figure 1). We will call the left-hand side of the network the "early-processing memory" and the right-hand side the "final-storage memory." The system works as follows. When a new pattern from the environment, consisting of an input and an associated output (the "teacher"), is presented to the system, this will cause activation to spread through both sides of the network. The fact that the new pattern is "real" (i.e., from the environment) means that its "teacher" will inhibit the output coming from final storage. Thus, the new pattern will be learned first in the "early-processing memory" by the standard backpropagation algorithm.

At the same time, a number of "pseudopatterns" will be generated in the final-storage memory. In other words, random input will be fed into the network. Activation will spread through both the early-processing and final-storage areas of the network. Crucially, *the output of this random input sent through the final-storage part of the network will be used as a teacher for the early-processing part of the network*. In other words, the early-processing memory will then learn a series of pseudopatterns produced in the final-storage memory that reflect the patterns previously stored in final-storage. So, rather than interleaving the real, originally learned patterns with the new input coming to the early-processing memory, we do the next best thing — namely, we interleave pseudopatterns that are *approximations* of the previously stored patterns. Once the new pattern and the pseudopatterns are learned in the early-processing area, the weights from the early-processing network are copied to the corresponding weights in the final-storage network.

The architecture is called "pseudo-recurrent" not only because of the recurrent nature of the training of the early-processing memory by the final-storage memory, but also as a means of acknowledging the all-important means of interleaving information from final-storage and the new information arriving at the early-processing area — namely, pseudopatterns.

A specific example may help to clarify the learning mechanism. Suppose that there are 20 patterns, P_1, P_2, ... P_{20}. Each of these patterns, P_i consists of an input and an output ("teacher") association (I_i, T_i). The system will be required to *sequentially* learn all 20 patterns. In other words, each individual pattern must be learned to criterion — along with all of the pseudopatterns generated by the final-storage memory — before the system can begin to learn the subsequent pattern. To learn pattern P_1, its input I_1 is presented to the network. Activation flows through both parts of the network but the output from the final-storage part is prevented from reaching the teacher nodes by the "real" teacher T_1. The early-processing network then adjusts its weights with the standard backpropagation algorithm using as the error signal the difference between T_1 and the output, O_1, of the early-processing network. Internally created pseudopatterns from the final-storage memory are now generated and will be learned by the early-processing memory. This is done by presenting random input to the network, which causes activation to spread through both the early-processing and the final-storage memories. For "pseudo"-input, unlike for "real" input, there is no "real" teacher to inhibit the arrival of final-storage activation to the teacher nodes (i.e., the third layer of the final-storage network). The activation on the teacher nodes is thus produced by the spreading of activation by the pseudo-input through the final-storage memory. These teacher nodes then serve to train the early-processing memory.

It is instructive to examine in detail the operation of one of the pseudopatterns. A random input pattern, i_1, is presented to the input nodes of the system. This input produces an output, o_1, at the output layer of the early-processing memory and also produces an output, t_1, on the teacher nodes of the final-storage memory. This input-output pair (i_1, t_1) defines a pseudopattern, ψ_1, that reflects the contents of the final-storage memory. The

difference between t_1 and o_1 determines the error signal for changing the weights in the early-processing memory. The other random inputs, i_2, i_3, . . . i_n, are successively presented to the system and the resulting pseudopatterns, ψ_2, ψ_3, . . . ψ_n will also be learned by the early-processing memory. Once the weight changes have been made for the first epoch consisting of $\{P_1, \psi_1, \psi_2, \ldots \psi_n\}$, the early-processing memory cycles through this set of patterns again and again until it has learned them all to criterion. By learning the pattern P_1 the early-processing memory is learning the new information presented to it; by learning the pseudopatterns $\psi_{1, \ldots,} \psi_n$, the early-processing memory is learning an approximation of the information previously stored in final storage. Obviously, the more pseudopatterns that are generated, the more accurately they will reflect the contents of final storage. (However, with too many pseudopatterns the early-processing memory may fail to converge. The type and number of pseudopatterns to use is an area of on-going investigation.) Once learning in the early-processing network has converged for P_1, ψ_1, ψ_2, . . . ψ_n, the early-processing weights then replace the final-storage weights. In other words, the early-processing memory *becomes* the final storage memory and the network is ready to learn the next pattern, P_2.

In the network as described in this paper, a new set of pseudopatterns was chosen each time a new pattern had to be learned. In other words, each time a new pattern had to be learned, the pseudo-inputs to the final-storage memory were randomly chosen. Another technique would be to always use the same set of pseudo-inputs. Or it might even be possible for the network to gradually learn an set of pseudo-inputs that would generate pseudopatterns that reflect the contents of the final-storage area better than pseudopatterns with random pseudo-inputs. For example, if we have a yes-no classifier network (i.e., all learned items produce either a **1** or **0** on output), pseudo-inputs that produced outputs close to **0** or **1** might be "better" than pseudo-inputs that were purely random. Or another important issue: must the early-processing area converge for the entire set $\{P_i, \psi_1, \psi_2, \ldots \psi_n\}$, as in done here, or only for P_i, as is done in Robins (1995)? These questions are all the focus of on-going research.

Databases used for the experiments

A number of experiments were run to test this architecture. Four different data sets were used for the experiments discussed in this paper. These were:

- A slightly modified version of data used in French (1992). These patterns are used to show that even if a system learns new data very slowly, in the absence of interleaving of previously learned patterns (or approximations of these patterns) serious interference can still occur;
- A mushroom database (Murphy & Aha, 1992) in which mushrooms were classified as either edible or poisonous on the basis of 22 technical attributes such as "cap shape," "cap color," "stalk shape," "veil color," "gill color," "habitat," etc. There were up to twelve different possibilities for each attribute;
- The 1984 U.S. Congressional Voting Records database (Murphy & Aha, 1992), also used to study catastrophic interference in French (1992, 1994). Inputs consisted of "yea-nay" votes (i.e., 1 or 0) on 16 separate issues and the output was the political party to which the voter belonged (Republican or Democrat);
- The original Cat and Dog pictures used by Eimas, Quinn, & Cowan (1994) and Mareschal & French (1997) to study infant categorization. These consisted of 18 dogs and 18 cats classified according to the following ten traits: head length, head width, eye separation, ear separation, ear length, nose length, nose width, leg length vertical extent, and horizontal extent.

Overview of the tests of the pseudo-recurrent memory model

The tests described below for the pseudo-recurrent memory model will be of three types. The first experiment shows that, without some form of interleaving of old information (or approximate old information) during learning of new information, regardless of how slowly the new information is learned, severe forgetting of the old information can occur. Next, we will show that if a network can continually train itself (pseudo-recurrence) on an approximation of previously acquired information, that the problem of catastrophic interference is drastically reduced and the system will not be overly disrupted by new information. We then show how this type of system can achieve true sequential learning, gradually, rather than suddenly, forgetting old information. We then show, somewhat surprisingly, that this memory model automatically produces the type of semi-distributed internal representations that were suggested by French (1992, 1994) as a means of overcoming catastrophic forgetting. However, in French (1992, 1994) the learning algorithms were specifically tailored to produce this kind of semi-distributed representation. In the present model, however, these semi-distributed representations emerge as a natural by-product of the architecture. We also demonstrate that the system exhibits both plausible list-length and list-strength effects. Finally, we show how a straightforward extension of the model allows the requirement of an equal number of hidden units for the early-processing and final-storage memories to be dropped. This extension involves using pseudopatterns to transfer information between both parts of the memory. To recapitulate, the experimental sequence is as follows:

Experiment 1: To show the necessity of *interleaving* old learning with new learning and why, if this is not done, severe interference can result.

Experiment 2: To show that the pseudo-recurrent architecture works with real data from a real-world data base.

Experiment 3: To demonstrate the pseudo-recurrent architecture's ability to achieve truly sequential learning with no refresh from previously learned items. This shows that the present architecture exhibits far more gradual (i.e., realistic) forgetting that standard backpropagation.

Experiment 4: To show that this memory model automatically produces the semi-distributed internal representations that French (1992, 1994) concluded were necessary to alleviate catastrophic interference.

Experiment 5: This experiment is an attempt to directly relate the model to the psychological literature involving sequential learning. The model reproduces both the list-length and list-strength effects referred to by Murnane & Shiffrin (1991) and others.

Experiment 6: To show that the model can be easily and effectively extended by allowing information to pass back and forth from the early-processing to the final-storage memory by means of pseudopatterns.

Most papers on catastrophic interference (e.g., French, 1992, 1994; Lewandowsky and Goebel, 1991; Lewandowsky & Shu-Chen, 1993; McRae and Hetherington, 1993; Robins, 1995; etc.) since Hetherington & Seidenberg (1989) have used an Ebbinghaus-like "savings" measure of forgetting. In other words, the network is trained to convergence on a particular set of data, D_1, and then given a new set of data, D_2, to learn. After learning D_2 the amount of time required by the network to relearn D_1 to criterion is measured. The faster relearning occurs, the more memory savings. But memory savings is only part of the story. Explicit recognition of previously learned patterns without any relearning is also very important. Consequently, this paper will focus not only on memory savings, but also on explicit

recognition measures. In short, if the system learns a set of patterns, D_1, to criterion and then subsequently learns a new set, D_2, to criterion, we need to know: How much of D_1 is still explicitly recognized by the network *with no relearning of D_1*?

The reason for focusing on explicit recognition is the assumption that the previously learned patterns *are often no longer available*. It is unrealistic to assume that we will always be able to refresh our memories through re-presentation of old patterns. It is not good enough for a creature that has to survive in a harsh environment to be capable of rapidly relearning old associations, as long as they are presented to it again. Old patterns must be able to be remembered *without refresh from the environment*. The pseudo-recurrent memory model presented here will attempt to indicate one way in which this might be achieved.

We will conclude with a section on a number of challenges and unsolved problems for this type of memory model.

Experiments

Experiment 1: The necessity of interleaving old patterns (or approximations of old patterns) with new information

In this simple experiment, we used a twelve item training set is used to train a standard 8-4-8 feedforward backpropagation network. The network had to autoassociate the following twelve patterns:

1 1 0 0 0 0 0 0
0 0 1 1 0 0 0 0
0 0 0 0 1 1 0 0
0 0 0 0 0 0 1 1
1 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0
0 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1

Once the network was completely trained to autoassociate these patterns, it was then given the following two patterns to learn:

1 1 1 1 0 0 0 0 –> 0 0 0 0 1 1 1 1
0 0 0 0 1 1 1 1 –> 1 1 1 1 0 0 0 0

Learning these two pairs should have an extremely detrimental effect on the previous learning. That is the intent. Perhaps, though, if the network were to learn this new information *very slowly*, it would be able to be incorporate the new patterns into the previously learned information without disrupting it.

In an attempt to incorporate the two new patterns into network without disrupting the 12 previously learned patterns, various learning rates, ranging from η=0.01 to η=0.0001 with various values for the momentum term, ranging from 0.05 to 0.9, were tested. Once the network had learned the two new (disruptive) patterns (often quite slowly because of the low learning rate), it was then tested on how well it remembered the previously learned set of twelve associations. Retraining on the original data was again done with a learning rate of 0.2 with momentum 0.9. Both the Ebbinghaus-savings and the exact-recognition of forgetting

were used. It took as long to relearn the original data after learning the two new patterns as it had to taken to learn it in the first place. Equally significantly, after learning the two new associations, the network never recognized more than one of the original patterns exactly (i.e. all outputs for a given input within 0.2, the convergence criterion, of the output). And, over a total of 50 runs, the overall percentage of exact recognition never exceeded 1%.

The point illustrated here is an important one — namely, regardless of how slowly a network learns new information, the previously learned information can nonetheless be overwritten. The key issue is not how slowly or quickly the network incorporates the new information, but rather *how* this new information is incorporated. As can be seen in the present case, learning only two new patterns, even using a very small rate of learning, can completely destroy all of the old information in the network. As Robins (1995) has shown, and as we will show in the remainder of this paper, by interleaving pseudopatterns along with the new information to be learned, this type of catastrophic interference can be significantly reduced.

Discussion of Experiment 1

McClelland, McNaughton, & O'Reilly (1995) have suggested that the brain overcomes catastrophic interference by having evolved two separate areas for information storage and processing: the hippocampus, where rapid learning of new information occurs and the neocortex, where the hippocampal learning is consolidated by being trained very slowly by the hippocampus. Their idea is that because the neocortex is learning hippocampal information very slowly, that this will suffice to prevent catastrophic interference of the information previously stored in the neocortex. The present experiment demonstrates that regardless of how slowly new information is learned by a connectionist network, this information can destroy previously learned information. We suggest that it is crucial that approximations of the old information — pseudopatterns — be interleaved with the new information.

Experiment 2: Performance of a pseudo-recurrent network on a real-world database

One of the objections of much of the work that has been done in the area of catastrophic interference is that the patterns to be learned are tailored to produce the desired phenomenon. For this experiment, we will consider data from the UCI mushroom database (Murphy & Aha, 1992). The reason that it is important to use real databases is that these presumably code for "real world" regularities. It can be shown (Hetherington & McRae, 1993) that when a neural network is pretrained on a random sample of data from the domain in which it will be operating, catastrophic interference can be eliminated. This assumes, of course, that the original random sample actually captures the regularities of the domain. When new data from the same domain is then presented to the network there will be little or no interference. This is because the new data will largely share the regularities learned by the network during pre-training on the original random sample. As a consequence, the new regularities will not interfere will the already-learned ones.

But what happens if the new set to be learned *does not* share the regularities of the patterns that the network was pre-trained on? Our own neural nets, for example, have been "pre-trained" on many instances of birds. We have, in short, learned many regularities about birds. But without difficulty we can learn to recognize penguins, which share very few of the previously learned "bird regularities" but are still birds, and bats, which share a great many regularities but are not. Here are cases where "pre-training" would be of little use in preventing interference because the regularities of the pre-training set would not be shared by the regularities of the new information to be learned.

So, for our second experiment, we needed data that would cause significant interference. For this we selected two "quasi-contradictory" sets of mushrooms from the UCI mushroom data base. Quasi-contradictory pairs of mushrooms are those that differ on no more than 3 (of 22) attributes with one member of each pair being poisonous, the other edible. There were approximately 70 such pairs of closely paired edible/poisonous mushrooms in the UCI database. We picked 20 mushrooms for the original training set and 20 "quasi-contradictory" partners for the second set to be learned. The original set contained 17 edible mushrooms and 3 poisonous mushrooms. The second set of new mushrooms that closely resembled those in the first set (the "contradictory data") contained 17 poisonous and 3 edible mushrooms.

For this experiment, each mushroom attribute was coded in a binary manner and added to the input vector describing the mushroom. The system performed a yes/no (poisonous/edible) classification task. The architecture of the pseudo-recurrent network had 57 input nodes, 10 early-processing hidden units, 10 final-storage hidden units, one early-processing output unit and one teacher unit (see Figure 3 for the design of the overall architecture).

One of the important things to notice in this experiment using a larger network is the fact that the information transfer using pseudopatterns must necessarily be approximate, since the probability of actually reproducing any of the actual patterns previously stored in final storage was, for all intents and purposes, zero.

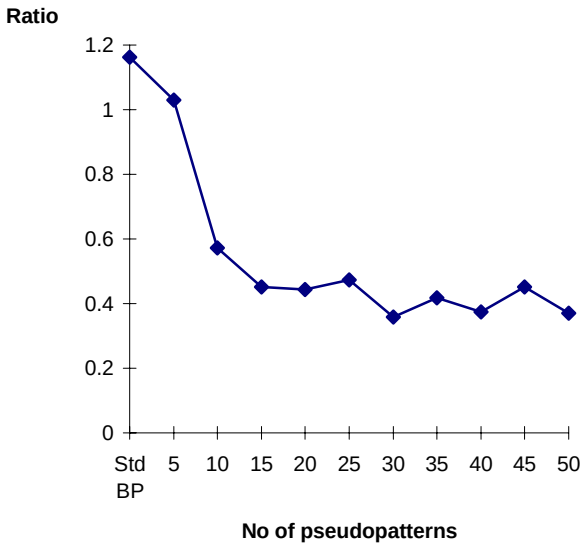

Figure 2. Ratio of the number of epochs required for relearning the original 20 patterns to the number of epochs initially required to learn these patterns (following learning of the 20 disruptive patterns), as a function of the number of pseudopatterns.

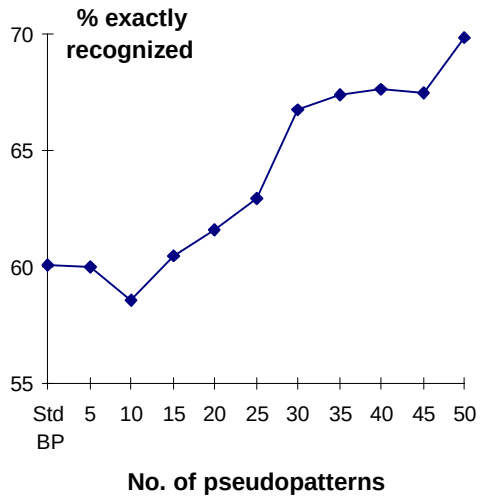

Figure 3. The increase in the percentage of items of the originally learned set exactly recognized by the network (after learning the set of disruptive items), as a function of the number of pseudopatterns.

Both Figures 2 and 3 show significant improvements on both depth-of-forgetting (as measured by the time required to relearned the original data) and exact recognition of items previously learned. In other words, the pseudo-recurrent network succeeds in maintaining the memory trace of the previously learned patterns while learning the new patterns. This should mean that true sequential learning of patterns, characterized by gradual forgetting of old information, should be possible with this type of network. Experiment 3 is designed to demonstrate the feasibility of sequential learning with a pseudo-recurrent network

Discussion of Experiment 2

When the pseudo-recurrent network is used with a database consisting of "real-world" patterns, in this case edible and poisonous mushrooms, it is able to perform consistently better than standard backpropagation. This improvement, measured both in terms of Ebbinghaus memory savings and exact recognition of old patterns, increases with the number of pseudopatterns mixed with the new data to be learned.

Experiment 3: Sequential learning

Arguably the most significant contribution of the pseudo-recurrent memory model is its ability to learn data in a serial manner without experiencing severe interference (Hetherington, 1991). In this experiment, the patterns were taken randomly from the 1984 Congressional Voting Records database at the UCI repository. Twenty members of Congress (10 Republicans, 10 Democrats, defined by their voting record on 16 issues and their party affiliation) were chosen randomly from the database and were presented sequentially to the network. The order of presentation was randomized in each run of the program and the results averaged over 50 runs.

After the twenty patterns were serially learned by the network, a test was made of the percentage of these items were exactly recognized (i.e., actual output within the criterion of 0.2 of the desired output). Figure 4 shows that with standard backpropagation this figure is around 40% whereas it climbs rapidly to 65% with the addition to each new pattern to be learned with only 5 pseudopatterns from final storage. With 50 pseudopatterns added to each new item to be learned, exact recognition of all of patterns climbs to 85%.

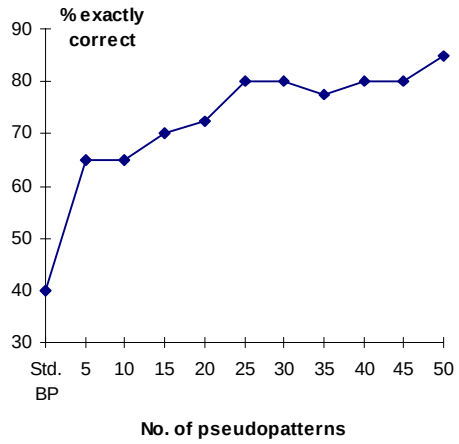

Figure 4. Percentage of all data exactly recalled by the network after serial presentation of all 20 patterns (median data), as a function of the number of pseudopatterns.

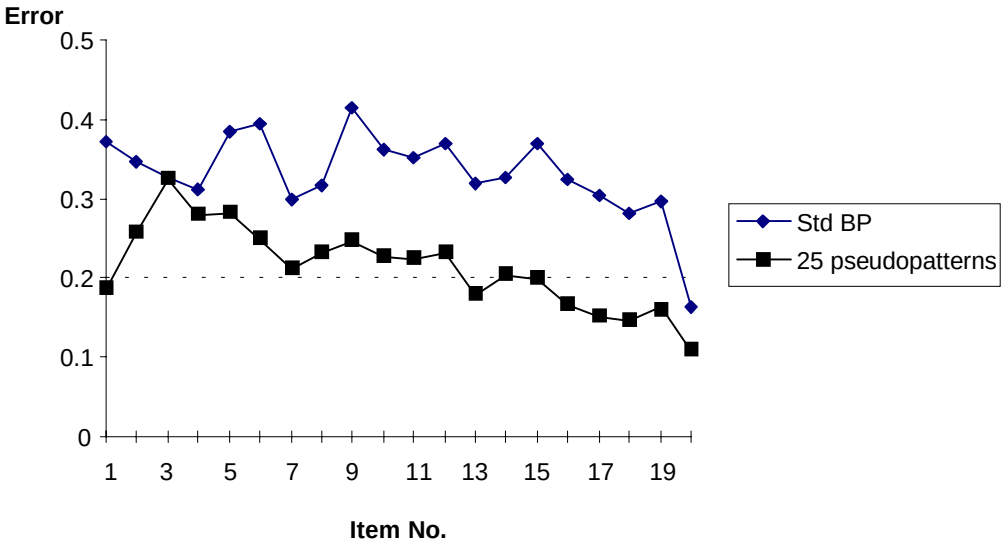

Figure 5. Amount of error for each of the 20 items learned sequentially after the final item has been learned to criterion (in this case, 0.2).

After the network had serially learned the twenty items, each item was individually tested to see how well the network remembered it. Figures 5 and 6 show that forgetting in the pseudo-recurrent network is indeed more gradual than for standard backpropagation. The standard backpropagation network is, on average, significantly above the 0.2 convergence criterion for *all* of the previously learned items, whereas the pseudo-recurrent network is at or below criterion for the last eight items learned (items 13-20) and within 0.05 of the criterion for items 7-12. By contrast, for standard backpropagation only items 18 and 19 are within 0.1 of the 0.2 exact recognition threshold. Finally, the amount of forgetting that takes place between the final item that is learned and its immediate predecessors is considerably greater in the standard BP network than for pseudo-recurrent networks (see Table 1).

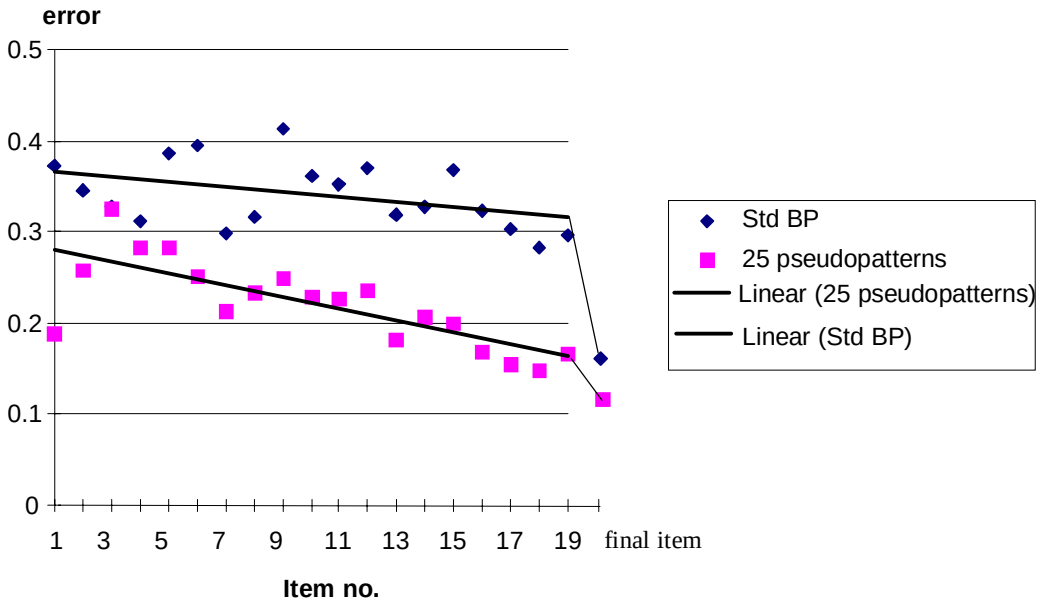

Figure 6. Linear fitting of the error values for the first 19 items learned. Notice that for standard backpropagation all 19 items are on average well above convergence threshold. Between the 19th and final item there is considerable forgetting for standard BP. This is not the case for the pseudo-recurrent network, where forgetting is considerably more gradual.

Std. BP	0.13
5 Pseudopats.	0.07
10 "	0.01
15 "	0.01
20 "	0.03
25 "	0.05
30 "	0.07
35 "	–0.02
40 "	0.03
45 "	0.08
50 "	0.05

Table 1. The differences in the amount of error between the final item learned and its immediate predecessor. Notice that in standard backpropagation, this difference is the largest (averaged over 50 runs for each network), indicating the considerable amount of forgetting of prior items when learning each new item.

Discussion of Experiment 3

The most significant contribution of the pseudo-recurrent memory model is its ability to learn patterns in a truly sequential manner. This experiment shows that the forgetting curves for this type of network are considerably more gradual (and therefore, cognitively plausible) than with standard backpropagation. This experiment also emphasizes the importance of interleaving approximations of the already-learned patterns with the new patterns to be learned.

Experiment 4: The automatic emergence of semi-distributed representations

French (1992) suggested the following basic relation between catastrophic forgetting and representations in a distributed system:

> "Catastrophic forgetting is a direct consequence of the overlap of distributed representations and can be reduced by reducing this overlap.
>
> Purely local representations will not exhibit catastrophic forgetting because there is little interaction among representations. Consider the extreme example of a look-up table, in which representations do not overlap at all. There is no catastrophic forgetting: when new information is added, the old information is completely preserved. However, because of its entirely local representations, a look-up table lacks the all-important ability to generalize.
>
> At the other extreme are fully distributed networks where there is considerable interaction among representations. This interaction is responsible for the networks' ability to generalize. However, these networks can be severely affected by catastrophic forgetting.
>
> The moral of the story would seem to be that you can't have it both ways. A system that develops highly distributed representations will be able to generalize but will suffer from catastrophic forgetting; conversely, a system that uses completely local representations may not suffer from catastrophic forgetting, but it will have little or no ability to generalize. The challenge is to develop systems capable of producing [semi-distributed] representations that are local enough to reduce, if not entirely overcome, catastrophic forgetting yet that remain sufficiently distributed to nonetheless allow generalization."

In French (1992) a "node-sharpening" algorithm was shown to decrease catastrophic interference by restricting the distribution of the network's internal representations as shown

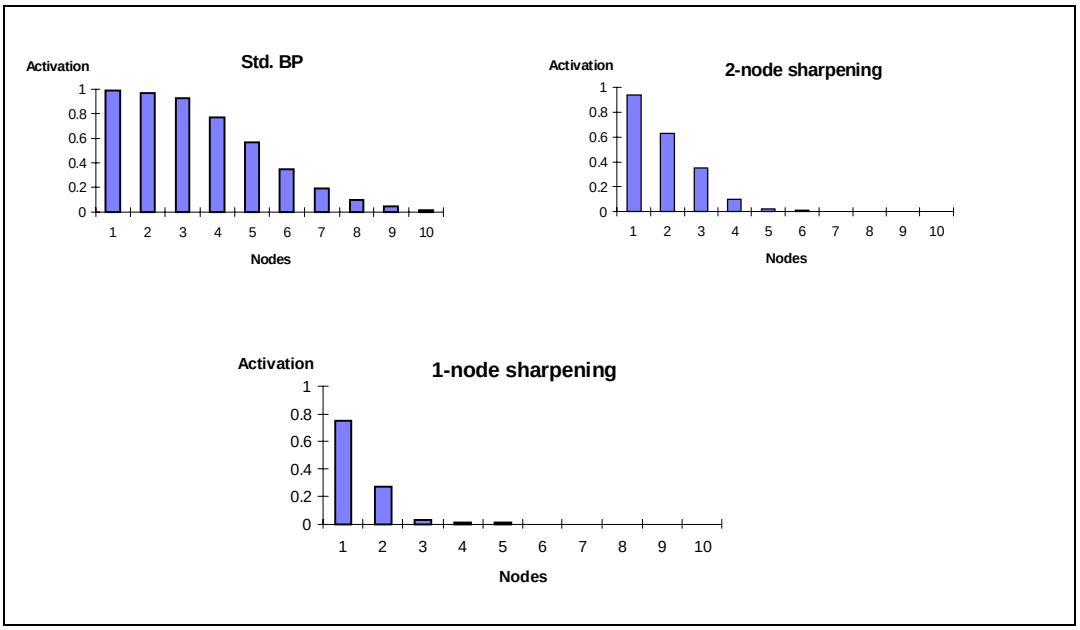

Figure 7. Semi-distributed internal representations of a category created by using a node-sharpening algorithm (French, 1992) designed to reduce internal representational overlap and to thereby reduce catastrophic interference. The one-node and two-node sharpened representations were particularly successful in reducing interference compared to the representations created by standard backpropagation

in Figure 7. A more powerful algorithm called context-biasing (French, 1994) produced similar modifications of the network's internal representations. In each case, representational overlap was diminished and catastrophic forgetting was significantly reduced. But these algorithms were, in both cases, *specifically designed* to produce semi-distributed representations. On the other hand, as can be seen in Figure 8, this type of semi-distributed representation *emerges automatically* from the continual interaction between the early-processing and the final-storage areas of the pseudo-recurrent architecture, rather than being an artifact of a specifically tailored learning algorithm.

Data set used

These data were obtained from measurements of the original Cat and Dog pictures used by Eimas, Quinn, & Cowan (1994) and by Mareschal & French (1997) to study infant categorization. They included 18 dogs and 18 cats classified according to head length, head width, eye separation, ear separation, ear length, nose length, nose width, leg length vertical extent, and horizontal extent.

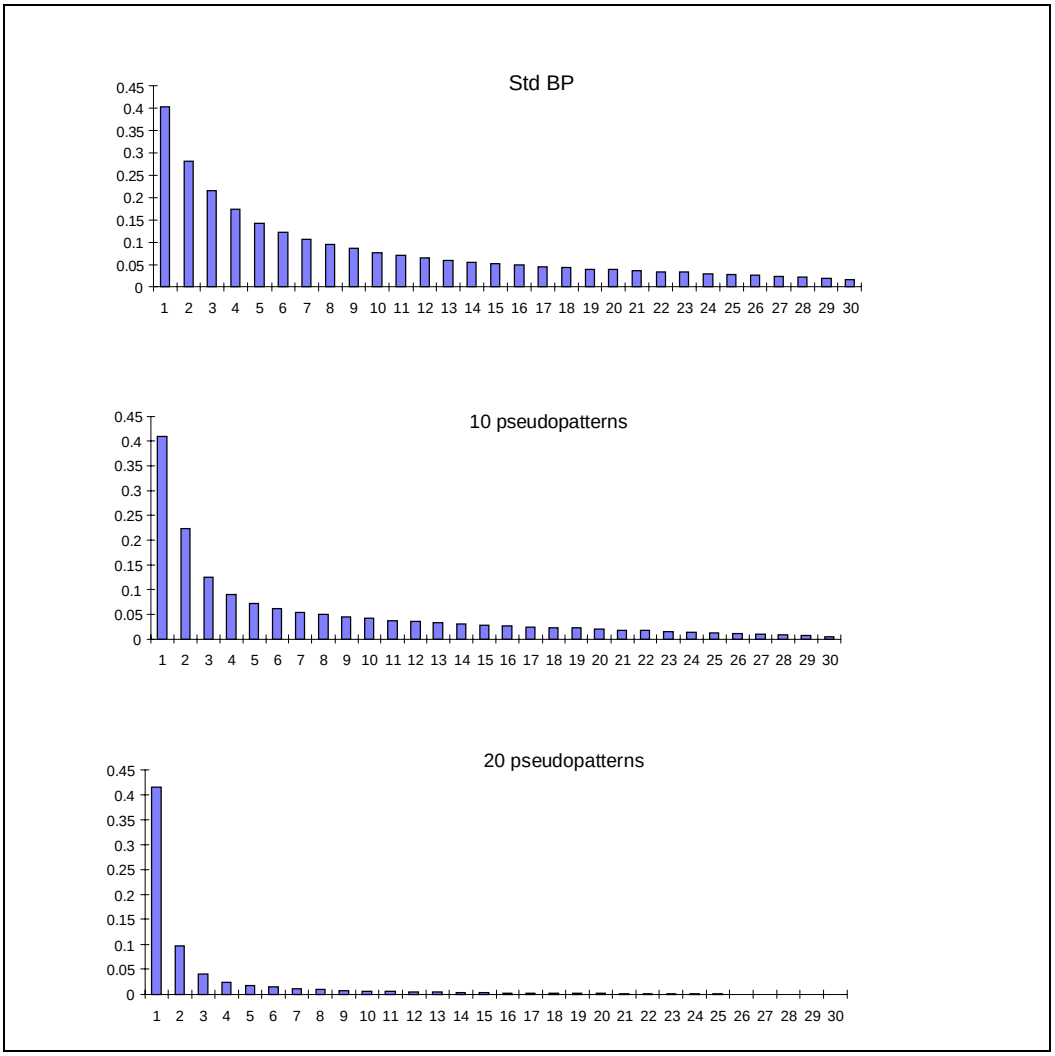

Figure 8: Here we see the same kind of decrease in the distributedness of the representation of a particular category (in this case, "cat") as we saw when a much more "artificial" algorithm (node-sharpening (French, 1992)) was used. The greater the number of pseudopatterns, the more compact the representation becomes.

Training procedure

A 10-30-10 autoassociator was used in learning these two categories of animals. We compared the hidden unit representations in networks that had sequentially learned 6 pairs of cats (i.e., a total of 12 cats) using differing numbers of pseudopatterns. Subsequently, the network learned to autoassociate a number of pairs of dogs (either 1 or 2 pairs). The network was then tested on its ability to correctly autoassociate the previously learned cats.

Results

The semi-distributed representations that emerge from the pseudo-recurrent network are shown in Figure 8. These representations are similar to the semi-distributed representations in Figure 7 (French, 1992) that were generated by an algorithm (node-sharpening) specifically designed to produce representations of this type in order to reduce hidden-layer representational overlap and thereby reduce catastrophic forgetting.

In this experiment, we trained the network to autoassociate on the training set of cats. Various pseudo-recurrent networks were tested, varying the number of pseudopatterns used in each case. As the number of pseudopatterns increases, the network's internal representation of the concept "cat" undergoes the same type of compaction observed with the node-sharpening algorithm. But here the important difference is that there is no explicit algorithm designed to produce these representations. They arise naturally from the continual interaction of the two areas of the memory.

It can be seen in Figure 9 that the pseudo-recurrent model did show considerably less forgetting of the originally learned cats than standard backpropagation. Moreover, the improvement was a function of the number of pseudopatterns used.

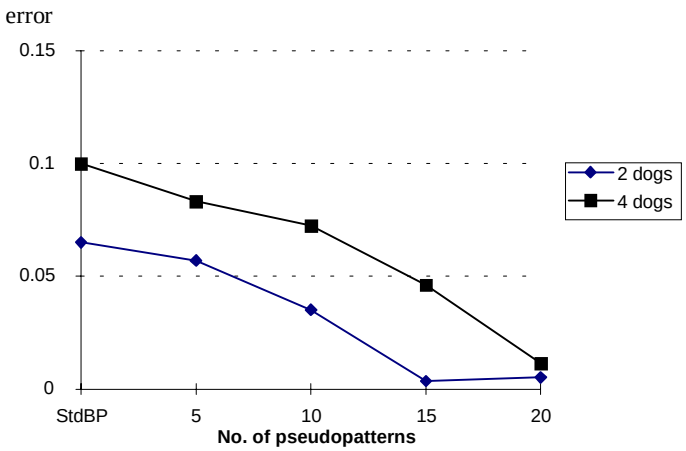

Figure 8. The improvement in the recognition of the previously learned cats after interference by either 2 or 4 dogs. The amount of forgetting is measured by the average error of the autoassociation.

Discussion of Experiment 4

. The continual interaction between the two processing areas of the pseudo-recurrent memory gives rise to more compact (i.e., less distributed) internal representations. These are the same types of representations that have been shown in other work to reduce catastrophic interference. There is, most importantly, no explicit mechanism generating these semi-distributed representations; they emerge naturally from the architecture.

This suggests an interesting possibility for the human brain. It has been shown that there is continual interaction between the hippocampus and the neocortex and that this interaction is almost certainly involved in long-term memory consolidation. If this interaction is indeed

mediated by pseudopatterns, as suggested by Robins (1996), then it would not be unreasonable to think that the representational compaction observed in the pseudo-recurrent model might also occur in the brain. Compact representations would, presumably, allow for more efficient processing of incoming stimuli. On the other hand, the more compact nature of these representations would make new category distinctions more difficult (see "Other problems with separating representations by orthogonalization at the hidden layer" below) and also lead to the category brittleness that occurs as people grow older. Finally, highly compact representations would presumably be more vulnerable to physical damage than highly distributed representations. This, too, would seem to be consistent with selective memory loss with aging.

Experiment 5: List-strength and list-length effects

This model is able to reproduce the standard list-length effect (Strong, 1912; Atkinson & Joula, 1973; Ratcliff & Murdock, 1976, Gillund & Shiffrin, 1984) and the list-strength effect reported in Murnane & Shiffrin (1991), Ratcliff, Clark, & Shiffrin (1990), Shiffrin, Ratcliff & Clark (1990).

The list-length effect means simply that adding new items to a list harms memory for other list items. For the pseudo-recurrent model, this easily falls out of the manner in which serial learning is done. The further back in a list an item occurs, the more poorly it is remembered (see Figures 5 and 6). Thus, when more items are added to a serial list, the less chance the earlier ones have of being remembered. While this is true, of course, of standard backpropagation models, as can be seen in Figures 5 and 6, beyond one or two items back, all items have basically been forgotten. The pseudo-recurrent network's ability to forget gradually allows it to show a plausible list-length effect.

The list-strength effect for recognition means that when the strength of an item is increased, there is *no effect* on the recognition of the other items. In other words, increasing the strength of a particular item in a list (usually, by repeating it) will have no effect on recognition performance of the other items. However, many current memory models predict the opposite — namely, that when certain items are strengthened, recognition performance on the other (unstrengthened) items will decrease (Murnane & Shiffrin, 1991).

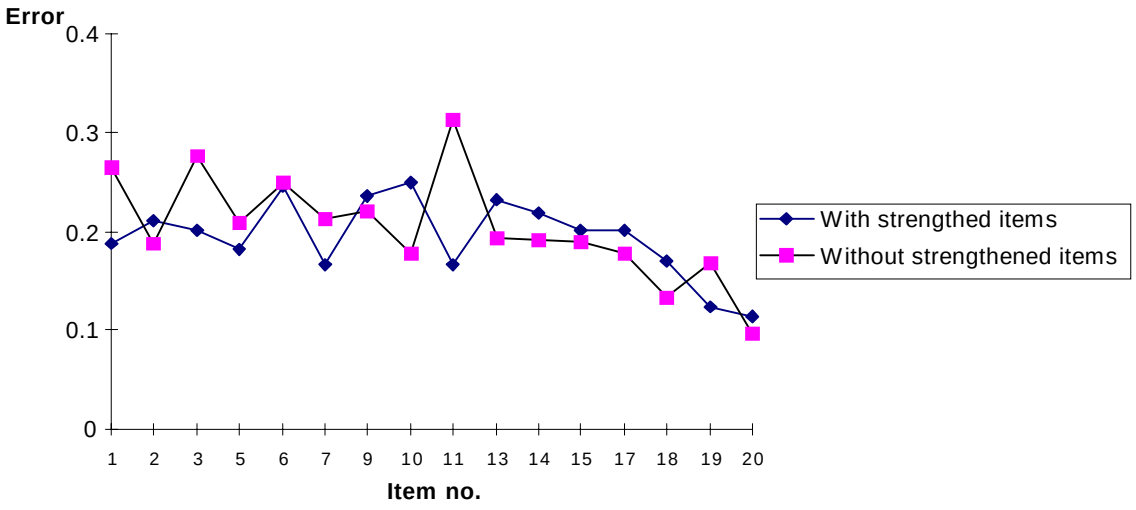

Figure 10. No significant difference between errors for the non-strengthened items in the condition where item no. 4 occurred four times and the corresponding items in the condition where no items were repeated.

To examine list-strength performance of the pseudo-recurrent network, we repeated the 4th element in the list, putting it in the 8th position, 12th and 16th position. As expected, this repetition did indeed strengthen the item in terms of its ease of recognition by the network. The decrease in average error when an item was strengthened (i.e., repeated) compared to the corresponding unstrengthened item is significant. (2-tailed, paired $T(10) = 5.71$, $p < 0.001$) .

Figure 10 compares the typical performance of two pseudo-recurrent networks each using 30 pseudopatterns. In one case item no. 4 had been repeated four times, in the other case, no item was repeated. The error curves, which are a measure of recognition competence — the smaller the error, the better the recognition — for the non-strengthened items do not differ significantly. This would appear to be in qualitative agreement with the list-strength effect for item recognition.

Discussion of Experiment 5

The pseudo-recurrent network would seem to be able to exhibit not only a traditional list-length effect in which adding additional items decreases recognition of the other items, but also a list-length effect for recognition in which strengthening several items of a list (by repetition) does not adversely affect the network's performance on the other items.

Experiment 6: Using pseudopatterns to transfer information in both directions between the early-processing memory and final-storage memory

Does this pseudo-recurrent memory model always require identical numbers of hidden nodes in the early-processing memory and in the final-storage memory? As presented in this paper, the answer is yes. But it is not difficult to modify the model so that information is transferred both from final storage to the early-processing memory as well as from the early-processing memory to final storage by means of pseudopatterns. The advantage of directly copying the contents of the early-processing memory to final storage is, of course, that there is no loss of information. The disadvantage is that the number of early-processing and final-storage hidden units must be identical and that the idea of copying weights directly from the early-processing memory to the final-storage memory lacks psychological plausibility.

Modified architecture

Instead of copying the weights from the early-processing storage to the final storage, the contents of the early-processing memory are transferred to final storage by means of pseudopatterns. In other words, once the new data (along with the pseudopatterns from final storage) have been learned to criterion in the early-processing memory, the early-processing memory generates a number of pseudopatterns that are then learned by the final-storage memory. The output nodes of the early-processing memory then become the teacher nodes for the final-storage memory.

Data set used

The same two "quasi-contradictory" sets from the UCI mushroom database that were used in Experiment 2.

Training procedure

The final storage memory independently learned the first set of 20 mushrooms. As in Experiment 2, this was done to simulate "background knowledge" in final storage. Then the 20 new quasi-contradictory patterns were presented to the system for learning. A total of 40

pseudopatterns from the final-storage memory were added to the new set of mushrooms to be learned. Once these were learned in the early-processing memory, instead of copying the early-processing weights as before, the information was transferred to final storage by a set of 40 pseudopatterns generated by the early-processing memory and learned by final storage. The final storage memory was then tested to determine:

- how long it took to relearn the original background patterns (in epochs);
- what percentage of the original patterns were exactly recognized.

These figures were compared to the same measurements for a standard backpropagation network (with the same architecture as the final-storage area of the pseudo-recurrent network) that had first learned the background set of patterns and then learned the new quasi-contradictory set of mushrooms.

Results

Both the Ebbinghaus-savings measure and the exact-recognition measures were used to measure forgetting. Figure 11 shows the number of epochs required to relearn the original data for the pseudo-recurrent network compared to the number of epochs required by a standard backpropagation network after each had learned the new data. There was a decrease from an average of 88 epochs to 43 (averaged over 30 runs), an improvement of over 50%. Figure 12 shows the corresponding improvement in exact recognition of the originally learned items for the pseudo-recurrent network compared to the backpropagation network.

By varying the number of pseudopatterns used to transfer information in each direction it was found that the amount of information transferred from one memory to another could be varied. Can the number of pseudopatterns in both directions be set very high to always ensure maximal information transfer? One problem with doing this is convergence times. The more pseudopatterns that need to be learned, the longer it takes the network to converge. If there are too many pseudopatterns added, convergence may take an extremely long time and, in some cases, may not even occur at all.

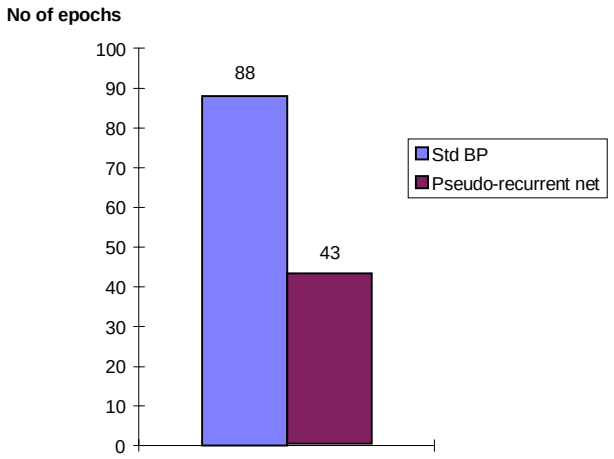

Figure 11. Decreased forgetting of the original data with respect to standard backpropagation (measured in number of epochs required for complete relearning).

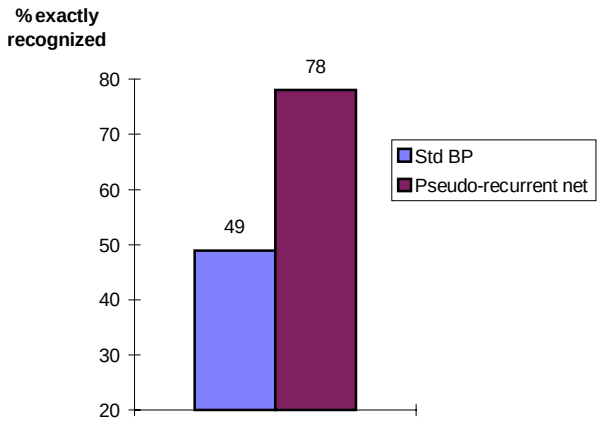

Figure 12. Improved recognition of the original data with respect to standard backpropagation (measured in percent of the originally learned patterns that were exactly recognized after the new patterns had been learned).

Discussion of Experiment 5

It is also possible to transfer information from the early-processing memory to the final-storage memory using pseudopatterns, instead of copying the early-processing weights to final storage. Although this technique is not explored in depth in this paper, it did significantly reduce interference with previously learned information, in addition to providing a more plausible transfer mechanism from the early-processing memory to final storage.

"Catastrophic remembering": a theoretical limitation of pseudo-recurrent networks

In the five experiments described above we have attempted to highlight some of the features that makes the pseudo-recurrent network a plausible memory model. But like all models, there are certain things that it can do better than others. One particular problem that this model cannot overcome is, ironically, the flip side of catastrophic forgetting, what Sharkey & Sharkey (1995) have called "catastrophic remembering.". The idea is quite simple: Assume a network (e.g., 9-9-1) has a training set of, say,

$$111000000 \rightarrow 1$$
$$000111000 \rightarrow 1$$
$$000000111 \rightarrow 1$$

where "1" represents "Have seen it." It will almost certainly learn these patterns very quickly by producing a very large weight from one of the hidden units to the output unit. The problem is that now *any* new pattern presented to the network will be ("catastrophically") remembered as having been seen as well.

One possible approach to this problem would be to use an autoassociator network. Instead of having the network learn the associations 111000000 $\rightarrow$ 1, etc., the pattern 111000000 would be autoassociated. Subsequently, if a new pattern were presented to the network, its input and corresponding output would be compared. If they were very close, the network would conclude that it had seen the pattern previously (only then producing the desired 1 on output); otherwise if the difference were too great if would conclude that it had not previously encountered the data (and only then produce a 0 on output).

However, as Sharkey & Sharkey (1995) point out, there is a problem with this technique: it only works if the number of autoassociations to be learned remains relatively low and the network is, in essence, still memorizing individual autoassociated exemplars. However, as soon as the number of autoassociated patterns becomes high enough for the network to effectively learn the identity function, the problem of catastrophic remembering returns with a vengeance. The network, having learned to produce on output what it received on input, will then "remember" any pattern, whether or not it has been seen before. The pseudo-recurrent architecture would be of little help in this situation. The problem, as Sharkey & Sharkey point out is that the ability to generalize to the identity function will necessarily mean that there will be a loss of discrimination. So, for this particular task, for a pseudo-recurrent network or for a standard backpropagation network, it would seem that one can have good generalization or good discrimination, but not both.

Another possibility might be to have input that has actually been seen to be associated with a "have seen" pattern (for example, 1), while a number of internally-generated random patterns would be associated with a "have not seen" pattern (for example, 0). So, for example, in addition to the three "real" patterns, the network would automatically also learn a number of internally generated random input patterns — say, 100101101, 001011010, etc. — that would all be associated with 0, thus providing a "background" against which the real patterns would stand out. If the input space were large enough, the probability of accidentally selecting a random pattern that was one of the actually seen patterns would be vanishingly small.

In any event, the problem raised by Sharkey & Sharkey (1995) is an important one and one for which the current pseudo-recurrent memory model has no immediate answer. Further research on this model will need to address this issue.

General Discussion

This paper addresses an important concern about the consolidation of information in long-term memory. It has been shown by a number of authors that many of the current connectionist models of memory suffer from catastrophic forgetting. In order to avoid this problem, connectionist networks in general, and backpropagation networks in particular, must be trained in a highly implausible manner. When new patterns must be incorporated into an already-trained network, all of the previously learned patterns *must be available* and must be re-presented to the network along with the new patterns to be learned. If this is not done, the previously learned patterns may be overwritten completely by the new patterns, resulting in catastrophic forgetting of the old patterns. It is unrealistic to suppose that we can continually refresh long-term memory with previously learned patterns. In many instances, these patterns simply are not available or, if they are available, they are only encountered very occasionally. (Think about how long it's been since you last saw a live giraffe, and yet, you would still recognize a giraffe without a moment's hesitation.) And yet, in spite of this lack of memory refreshing from actual instances in the world, we are able to maintain memory traces of these instances, or at least, prototypes of them. The long and the short of the problem is that we humans can do sequential learning; connectionist models cannot. And, unless the problem of catastrophic forgetting in connectionist models is overcome, true sequential learning will remain impossible.

It is also generally acknowledged that one of the main contributing factors to catastrophic interference is the overlapping nature of the network's internal representations. Decreasing representational overlap results in a decrease in catastrophic forgetting. This fact led McClelland, McNaughton, & O'Reilly (1995) to suggest that the way the brain overcomes catastrophic interference was by having separate processing areas, the hippocampus and the neocortex. They suggested that new information arrives at the hippocampus and is learned

there first, away from previously learned information, before being consolidated in the neocortex. Neocortical consolidation is done by the hippocampus slowly training the neocortex.

While I believe that this picture is basically right, a key element is missing. It is not enough, as I hope to have shown in Experiment 1, to have the hippocampus train the neocortex, however slowly. Catastrophic interference with previously learned information can still result. The unavoidable conclusion: Previously learned information — or, at the very least, *approximations* of the previously learned information — must somehow be *continually interleaved* with the new information. This fact is the basis of the pseudo-recurrent model of memory. This model consists of two continually interacting sub-systems, one continually providing approximations of the previously learned information (pseudopatterns) to be interleaved with the newly arriving patterns learned by the other.

Robins (1995) has demonstrated the effectiveness of pseudorehearsal (i.e., the use of pseudopatterns) in reducing catastrophic interference in a single network. In the present paper, however, a two-part model of memory is developed that makes use of this mechanism as a means of continually transferring information from one part of the memory to the other. The idea is not to demonstrate that pseudopatterns work, but rather to incorporate them into a memory model that has certain parallels in the neurobiological literature, in particular, with McClelland, McNaughton, and O'Reilly (1995). The key feature that sets this model apart from other models and from the earlier work on pseudopatterns is its use of two distinct areas that are in continually interaction by means of pseudopatterns. The work by McClelland, McNaughton and O'Reilly strongly suggests a two-part architecture as the means by which the brain overcomes the problem of catastrophic interference. The proposed pseudo-recurrent model, especially the version described in Experiment 6 where pseudopatterns are used to pass information back and forth from both sub-systems, is at least one two-part architecture that does effectively overcome catastrophic interference.

Experiment 2 uses a real-world database to establish that the pseudo-recurrent network succeeds in significantly reducing interference. Since, in the catastrophic forgetting literature, two measures of forgetting are frequently used, an Ebbinghaus-like "savings" measure and a measure of percentage of explicit recognition of previously learned items, we demonstrate in this experiment that for both of these measures, interference from newly learned patterns is significantly reduced in the pseudo-recurrent memory model.

In Experiment 3, it is shown that this model actually does succeed in doing sequential learning, producing a psychologically realistic pattern of gradual forgetting of the previously learned items.

Experiment 4 demonstrates a somewhat unexpected phenomenon — namely, that the pseudo-recurrent architecture automatically evolves compact (i.e., semi-distributed) internal representations. In previous work on catastrophic forgetting (French, 1992, 1994; Murre, 1992; etc.) these are exactly the type of representations that has been suggested were needed to reduce representational overlap, thereby reducing the interference that led to severe forgetting. That these representations arise naturally from this architecture has a number of possible implications for subsequent work. If, in fact, the brain has some architectural resemblance to the pseudo-recurrent memory model, then it, too, might over time be expected to evolve the compact internal representations that are observed in the pseudo-recurrent network. This could help shed light on the phenomenon of age-related categorical brittleness without requiring, for example, an appeal to mechanisms of neural pruning that have been suggested elsewhere (see, for example, Edelman, 1987). In addition, the more compact representations would tend to exhibit a greater sensitivity to damage, which might help to explain certain memory losses that accompany senescence.

Experiment 5 shows that this model, unlike standard backpropagation models, exhibits a plausible list-length effect. This is a direct consequence of its being able to do sequential learning in a reasonable manner, i.e., in a way that produces gradual forgetting. The list-length effect says that as the length of a list of items to be learned gets longer, earlier items are forgotten. While this is certainly true of standard backpropagation, it is true in a somewhat trivial sense. As Figure 5 shows, beyond one or two items, a backpropagation network has largely forgotten *all* of the previously learned items. On the other hand, the gradual forgetting that occurs in the pseudo-recurrent networks produces a more cognitively plausible list-length effect. In addition, this network (as well as standard backpropagation) exhibits a list-strength effect, which is to say, that strengthening certain items by repetition not only makes these repeated items easier to recognize, but, in addition, does not affect the network's ability to recognize the other non-repeated items.

One of the criticisms of the pseudo-recurrent architecture as shown in Figure 1 is that the mechanism of copying the weights from the early-processing sub-network to the final-storage sub-network lacks plausibility. In particular, this constraint also means that both sub-networks have to have identical sizes and connection topologies. In the final experiment, then, this constraint is relaxed and information is transferred from one sub-network to the other exclusively by means of pseudopatterns. Both the Ebbinghaus savings measure and the exact recognition measure of forgetting were used to demonstrate that catastrophic forgetting is significantly reduced in this type of network, just as it is in the original system that relies on weight-copying. In other words, the weight-copying mechanism, while certainly more efficient than transferring information by pseudopatterns, is not a necessary feature of the pseudo-recurrent memory model. This is important because it is clearly more difficult to justify weight-copying in a cognitive model that the transfer of information using the pseudopattern mechanism described in this paper.

There are, of course, many other unresolved issues that the pseudo-recurrent memory model would ultimately have to address. In particular, both the early-processing and the final-storage areas of the model should undoubtedly be recurrent networks of some kind — for example, simple recurrent networks (Elman, 1990). Further, no attempt was made here to model the effect of arousal on memory consolidation (Chown, 1994). In the current model, everything learned in the early-processing area is uniformly transferred to the final-storage area. Finally, the model should probably also include some sort of discriminator mechanisms in order to deal with the problem of catastrophic remembering (Sharkey & Sharkey, 1995). While these issues exceed the scope of the present paper, they will be explored in the future.

In conclusion, this paper describes a memory model that provides a possible solution to the sensitivity-stability problem, and the directly related problem of sequential learning, that makes use of separate, but continuously interacting areas to store already-learned and newly-learned patterns (McClelland, McNaughton, O'Reilly, 1995). The reasons for my belief in the necessity of this separation are tied to previous work on catastrophic forgetting. The proposed pseudo-recurrent memory model also makes the key assumption that previously-learned patterns will be unavailable for re-presentation to the network. I hope to have demonstrated that the pseudo-recurrent model not only can achieve effective sequential learning, but that the continual exchange of information between the two areas of the memory naturally gives rise to "compact" representations of the patterns learned, with the advantages (and problems) that this entails. Further research will show whether there are neural correlates to the development of this kind of semi-distributed representation.

Acknowledgments

This work was supported in part by research grant PAI P4/19 from the Belgian government. In addition, the author wishes to extend special thanks to an anonymous reviewer of this paper for numerous particularly helpful comments.

References

Atkinson, R. and Joula, J. (1973) Factors influencing speed and accuracy of word recognition. In S. Kornblum (Ed.), *Attention and performance* (Vol. 4, pp. 583-612). San Diego, CA: Academic Press..

Carpenter, G. and Grossberg, S. (1987). ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics* (26)23. 4919-4930.

Chown, E. (1991). Consolidation and learning: A connectionist model of human credit assignment. Unpublished doctoral dissertation. University of Michigan, Ann Arbor, MI.

Edelman, G. (1987) *Neural Darwinism: The Theory of Neuronal Group Selection*. NY: Basic Books.

Eimas, P. D., Quinn, P. C., & Cowan, P. (1994). Development of exclusivity in perceptually based categories of young infants, *Journal of Experimental Child Psychology, 58*, 418-431.

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

Frean, M. and Robins, A. (1996) Catastrophic Forgetting in Neural Networks: A Review and an Analysis of the Pseudorehearsal Solution. Otago University CS tech report AIM-36-97-2.

French, R. M. (1992) Semi-distributed representations and catastrophic forgetting in connectionist networks. *Connection Science*, *4*, 365-377.

French, R. M. (1994) Dynamically constraining connectionist networks to produce orthogonal, distributed representations to reduce catastrophic interference. In *Proceedings of the 16th Annual Cognitive Science Society Conference*. Hillsdale, NJ: Lawrence Erlbaum, 335-340.

Gillund,G. and Shiffrin, R. (1984) A retrieval model for both recognition and recall. *Psychological Review* (91) 1-67.

Hebb, D. O. (1949). *Organization of Behavior*. New York, N. Y.: Wiley & Sons.

Hetherington, P. A. (1991) The sequential learning problem in connectionist networks. Unpublished Master's Thesis, Dept. of Psychology, McGill University, Montreal.

Hetherington, P. A. and Seidenberg, M. S., (1989), Is there "catastrophic interference" in connectionist networks?, In *Proceedings of the 11th Annual Conference of the Cognitive Science Society*, Hillsdale, NJ: Erlbaum, 26-33.

Kanerva, P. (1989) *Sparse Distributed Memory*. Cambridge, MA: The MIT Press.

Keeler, J. D. (1988) Comparison between Kanerva's SDM and Hopfield-type neural networks. *Cognitive Science*, 12, 279–329.

Kortge, Chris A., (1990). Episodic Memory in Connectionist Networks. *Proceedings of the 12th Annual Conference of the Cognitive Science Society*, Hillsdale, NJ: Erlbaum, 764-771.

Kruschke, J. K. (1993) Human Category Learning: Implications for Backpropagation Models. *Connection Science*, Vol. 5, No. 1, 1993.

Lewandowsky, S. & Goebel, R. (1991) Gradual unlearning, catastrophic interference, and generalization in distributed memory models. University of Oklahoma Psychology Department Technical Report, presented at the 1991 Mathematical Psychology Conference, Indiana University, Bloomington, IN

Lewandowsky, S. & Shu-Chen Li (1993) Catastrophic Interference in Neural Networks: Causes, Solutions, and Data. In *New Perspectives on interference and inhibition in cognition* F.N. Dempster & C. Brainerd(eds.). New York, NY: Academic Press.

Mareschal, D. & French, R. (1997). A connectionist account of interference effects in early infant memory and categorization. (Submitted to the 1997 Cognitive Science Society Conference).

McClelland, J., McNaughton, B., & O'Reilly, R. (1995), Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*. 102, 419–457.

McCloskey, M. & Cohen, N. J. (1989). "Catastrophic interference in connectionist networks: The sequential learning problem" The Psychology of Learning and Motivation, 24, 109-165.

McRae, K. & Hetherington, P. (1993) Catastrophic interference is eliminated in pretrained networks. In *Proceedings of the 15h Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum. 723-728.

Metcalfe-Eich, J. (1982) A composite holographic associative recall model. *Psychological Review*, *89*(6) 627-661.

Murdock, B. (1983) A Distributed Memory Model for Serial-Order Information. *Psychological Review*, *100*(2) 183-203.

Murnane, K. and Shiffrin (1991) Interference and the Representation of Events in Memory. *JEP: Learning, Memory, and Cognition*. (17)5. 355-374.

Murphy, P. & Aha, D. (1992). UCI repository of machine learning databases. Maintained at the Dept. of Information and Computer Science, U. C. Irvine, CA.

Murre, J. (1992) The effects of pattern presentation on interference in backpropagation networks. In *Proceedings of the 14th Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum. 54-59.

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

Ratcliff, R. and Murdock, B. (1976) Retrieval processes in recognition memory. *Psychological Review* (83) 190-214.

Ratcliff, R., Clark, S., and Shriffin, R. (1990) The list-strength effect: Data and discussion. *JEP: Learning, Memory, and Cognition*. (16). 163-178.

Robins, A. (1995) Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, **7**(2), 123–146.

Robins. A. (1996) Consolidation in Neural Networks and in the Sleeping Brain. (under review).

Sharkey, N. & Sharkey, A., (1995) An analysis of catastrophic interference. *Connection Science*, **7**(3-4), 301–329.

Shiffrin, R. Ratcliff, R. and Clark, S. (1990) List-strength effect II. theoretical mechanisms. *JEP: Learning, Memory and Cognition* (16) 179-195.

Strong, E. (1912) The effect of length of series upon recognition memory. *Psychological Review* (19) 447-462.