Neural networks with a self-refreshing memory: knowledge transfer in sequential learning tasks without catastrophic forgetting

BERNARD ANS and STÉPHANE ROUSSET

Laboratoire de Psychologie Expérimentale,
Université Pierre Mendès France – CNRS UMR 5105,
BP 47, 38040 Grenoble cedex 09, France
email: (Bernard.Ans or Stephane.Rousset) @upmf-grenoble.fr

Abstract. We explore a dual-network architecture with self-refreshing memory (Ans and Rousset 1997) which overcomes catastrophic forgetting in sequential learning tasks. Its principle is that new knowledge is learned along with an internally generated activity reflecting the network history. What mainly distinguishes this model from others using pseudorehearsal in feedforward multilayer networks is a reverberating process used for generating pseudoitems. This process, which tends to go up to network attractors from random activation, is more suitable for capturing optimally the deep structure of previously learned knowledge than a single feedforward pass of activity. The proposed mechanism for 'transporting memory' without loss of information between two different brain structures could be viewed as a neurobiologically plausible means for consolidation in long-term memory. Knowledge transfer is explored with regard to learning speed, ability to generalize and vulnerability to network damages. We show that transfer is more efficient when two related tasks are sequentially learned than when they are learned concurrently. With a self-refreshing memory network knowledge can be saved for a long time and therefore reused in subsequent acquisitions.

Keywords: sequential learning, catastrophic forgetting, self-refreshing memory, pseudorehearsal, reverberating process, memory transport, long-term memory consolidation, knowledge transfer.

1. Introduction

Learning in distributed multilayer neural networks is most often achieved through a gradient descent adaptive algorithm, of which the most popular and widely used is the backpropagation procedure (Rumelhart *et al.* 1986). It is well known that when gradient descent learning procedures are used in sequential learning tasks, a major drawback, termed catastrophic forgetting (or catastrophic interference), generally arises: when a network having previously learned a first set of items is retrained on a second set of items, the newly learned information may completely destroy the information learned about the first set (McCloskey and Cohen 1989, Ratcliff 1990). Since this behaviour is unacceptable for models of human learning and memory, a number of authors have explored several ways of reducing the retroactive interference in sequential learning tasks (Hetherington and Seidenberg

Connection Science ISSN 0954-0091 print/ISSN 1360-0494 online © 2000 Taylor & Francis Ltd
http://www.tandf.co.uk/journals

1989, McCloskey and Cohen 1989, Kortge 1990, Ratcliff 1990, Lewandowsky 1991, 1994, Murre 1992, French 1992, 1994, 1997, McRae and Hetherington 1993, Lewandowsky and Li 1995, McClelland *et al.* 1995, Sharkey and Sharkey 1995, Robins 1995, 1996a, Ans and Rousset 1997, Robins and McCallum 1998; for a review, see French 1999).

The resolution of this problem constitutes a difficult task because the distributed nature of represented information, essentially required within networks to achieve generalization, seems to be incompatible with a weak interference level. In highly distributed systems, knowledge representations about different learned items extensively share the same connection weights. When a new set of items is learned, the same connection weights, which were already adjusted for previously learned items, will once more be modified. This may completely abolish memory of old information, resulting in the classical 'sensitivity–stability dilemma' (Hebb 1949) or 'stability–plasticity dilemma' (Grossberg 1987, Carpenter and Grossberg 1988). Other memory models that use separate or sparse distributed representations face this dilemma to a lesser extent (e.g. Hintzman 1986, Grossberg 1987, Kanerva 1988, Krushke 1992, 1993, Ans *et al.* 1998). However, when a high level of generalization is required in cognitive modelling, it is necessary to use a highly distributed system, implying that the catastrophic forgetting problem should be solved.

Catastrophic interference can be eliminated in sequential learning by using a rehearsal mechanism: the old information previously learned by a network is continually refreshed (i.e. retrained) during the learning of new information. This trivial solution, which requires permanent access to all events on which the network was trained during its history, is unacceptable when it is seen as the only solution for the human brain. Indeed, humans have in general the ability to learn new events without the complete abolition of memory for old events, events which do not occur again systematically for their consolidation. Nevertheless, this potential solution proves to be useful in the understanding of the attractive *pseudorehearsal* mechanism, recently proposed by Robins (1995, 1996a), which works without recourse to old events for refreshing memory. To describe the basic principles of this mechanism, consider a series of item sets which have to be learned sequentially (set A, next set B, next set C, . . . , etc.) by a feedforward multilayer network using a gradient descent learning algorithm. Each set contains a number (that can be reduced to only one) of input–target items which have to be associated after learning. Once the learning of the first set A of associative pairs is completed and before the learning of the second set B starts, the network is stimulated by random input patterns, each generating a corresponding output pattern. These input–output pairs are successively stored in a pseudopopulation which is then considered as having captured something reflecting the set A structure. During the learning of the second set B, the network is concurrently trained on the input–output pairs previously stored in the pseudopopulation. These last pairs are seen as pseudo-associations reflecting the old knowledge. Learning the second set is considered as being completed when a learning criterion is reached for all set B input–output pairs (the pseudo input–output are not subject to a learning criterion). The same process applies again for the learning of the third set C: before learning set C a pseudo-population has to be built up, hence capturing some representation of the A–B structure, and then the new set is trained in conjunction with the refreshed A–B pseudo-knowledge. The other sequentially learned new sets will then be processed in the same way. This pseudorehearsal mechanism was applied to several sequential tasks (Robins 1995, 1996a) in the framework of the standard backpropagation. The results showed a significant decrease of retroactive interference compared with those obtained on the same tasks processed without the pseudorehearsal mechanism.

This very promising approach inspired the authors (Ans and Rousset 1997). We proposed a learning connectionist architecture, with a self-refreshing memory, which overcomes catastrophic forgetting in an efficient way. Two basic questions were addressed: (i) how the pseudopopulation notion can be neurally implemented in the framework of a pure connectionist architecture; and (ii) how the deep structure of the knowledge represented in the connection weights of a neural system can be optimally captured. These two points were also addressed independently by Frean and Robins (1997), French (1997), Robins (1997a, b) and Robins and McCallum (1998). This paper highlights the essential properties of the learning connectionist architecture we propose for implementing self-refreshing memory, then it explores some fundamental consequences of self-refreshing on knowledge transfer in sequential learning tasks.

2. Reverberating networks with a self-refreshing memory: sequential learning without catastrophic forgetting

2.1 *A dual-network architecture*

The neural network architecture proposed to overcome catastrophic interference (Ans and Rousset 1997) consists of two coupled multilayer networks *NET 1* and *NET 2* (see figure 1). Within each network, as usual, an input layer is fully connected to a hidden layer, which is itself fully connected to an output layer. In contrast, with classical feedforward networks the hidden layer is also fully connected to the input layer. In the dual architecture, the *NET 1* network can learn external (or environmental) items, but also information issuing from *NET 2*, whereas the *NET 2* network can learn information only from *NET 1*. Backpropagation is used in the two networks. When *NET 1* is presented with a given external input–target pair to learn, the error function to be minimized in the learning algorithm is based not only on the error between the computed output and the output target, but also on the error between the computed activation from hidden units to input units and the external input pattern. The latter thus plays the role of a second desired target. In this way, the connections from hidden units to output units implement hetero-associations and those from hidden to input units implement autoassociations. The same holds for the *NET 2* network, which is also composed of hetero-associative and auto-associative parts. It is noteworthy that the auto-associative part is always required in the architecture, even when hetero-association is the focus of the task under study.

A simple explanation of the basic functioning of the architecture, shown in figure 1, would be to consider an initial state in which *NET 1* has already learned completely a given set of external associative items and *NET 2* is still ‘empty’ (i.e. with random connection weights). Assume that the neural system then enters in a first processing procedure, denoted stage (I) (left part of figure 1), in which *NET 1* is no longer receptive to external events but is continually ‘bombarded’, over its input layer, by a random activation issuing from a noise generator. For a given occurring random input ‘seed’, a first resulting activity is computed, via the *NET 1* hidden layer connectivity, on the output layer and on the input layer (the random seed

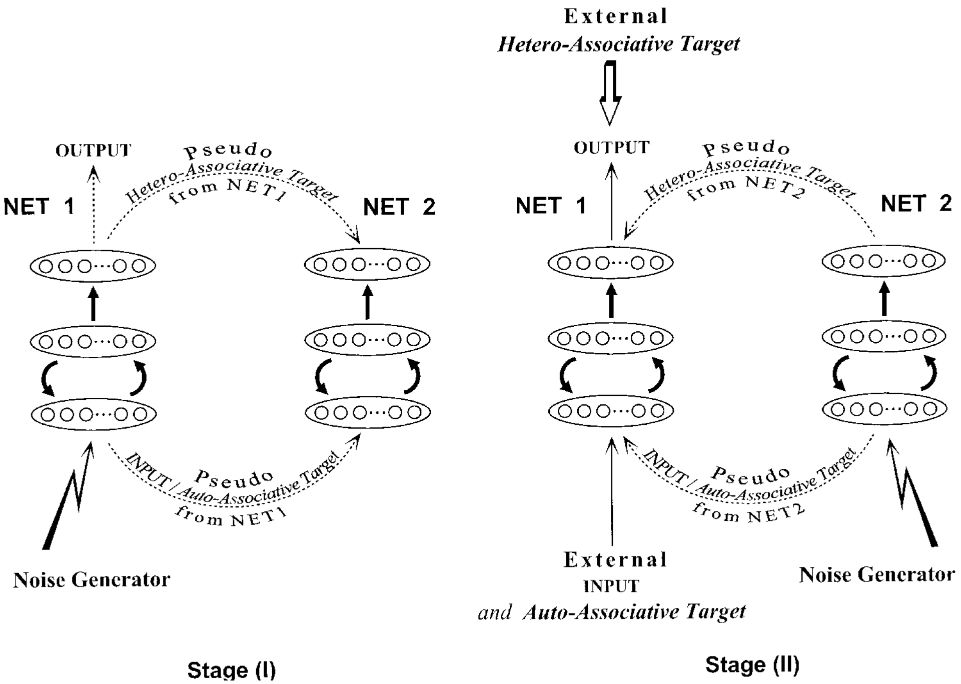
OUTPUT
Pseudo Hetero-Associative Target from NET 1
NET 1
NET 2
INPUT / Pseudo Auto-Associative Target from NET 1
Noise Generator
Stage (I)
External
Hetero-Associative Target
OUTPUT
Pseudo Hetero-Associative Target from NET 2
NET 1
NET 2
INPUT / Pseudo Auto-Associative Target from NET 2
External
INPUT
and Auto-Associative Target
Noise Generator
Stage (II)

Figure 1. The reverberating architecture with self-refreshing memory. Stage (I): the *NET 2* network is learning pseudoitems generated by the reverberating process in *NET 1* (transport of *NET 1* memory towards *NET 2*). Stage (II): the *NET 1* network is learning external items along with pseudoitems generated by the reverberating process in *NET 2* (learning with self-refreshing of old information).

no longer having an effect on the input activity computation). This first resulting input layer activity is then reinjected in the hidden layer, which creates a new output and input activity. This second input activity is reinjected in the hidden layer, hence recreating a third input–output activity, and so on. This back and forth flow of activity between the hidden and input layers is termed a 'reverberating' process. After a fixed number of reinjections (which is a simulation parameter, denoted R) within the *NET 1* auto-associative part, the current generated input and output activities are, respectively, transmitted to the *NET 2* input and output layers for training. The current *NET 1* output plays the role of a pseudo hetero-associative target for *NET 2* and the current *NET 1* input plays the role of both a pseudo input and a pseudo auto-associative target for *NET 2*. For each of the successive random input seeds, the corresponding pseudoitems generated by reverberation within *NET 1* are trained in *NET 2*. Stage (I) is in fact intended to 'transport' the previously learned information from *NET 1* to *NET 2*. A second processing procedure, denoted stage (II) (right part of figure 1), is intended to allow *NET 1* to learn new external items. During this stage, *NET 2* is 'bombarded' by the random generator and *NET 1* becomes receptive again to external events it can then learn. A new population of external items must be trained concurrently with pseudoitems originating continuously from *NET 2*, these pseudoitems being generated exactly as in *NET 1* during stage (I). Subsequently, stages (I) and (II) are supposed to work alternatively for each new occurring population of 'actual' items.

In other words, new environmental knowledge is always learned in the first network *NET 1* along with internally generated information (from *NET 2*) supposed to reflect its own history. This self-refreshing of the neural system memory is basically able to overcome catastrophic forgetting in sequential learning tasks.

The *NET 2* network is in fact a connectionist implementation of the pseudopopulation notion proposed by Robins (1995). However, Robins (1997 a) suggested, in order to avoid 'storing' pseudoitems in a pseudopopulation, using a single network with two weights per connection. A fast weight is used for new learning and a slow weight is involved for pseudoitem generation. The idea of using pseudoitems to pass information between networks has explored recently and independently (Ans and Rousset 1997, Frean and Robins 1997, French 1997, Robins, 1997b). The memory model proposed by French (1997) has several features in common with ours in so far as they both explicitly use a dual-network architecture in simulations. In French's model, one network 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 pseudopatterns, which are approximations of its content, to the early processing area. There, they are interleaved with the new patterns to be learned. By using this model, an original explanation of category-specific semantic deficits was recently proposed (French and Mareschal 1998). However, what essentially distinguishes our approach from the latter, and also from the other papers cited using pseudorehearsal in feedforward multilayer networks, is the introduction of a reverberating process for generating pseudoitems. This process, tending to go up to network attractors, is more suitable for capturing optimally the deep structure of knowledge distributed within connection weights than a single feedforward pass of activity. It was shown (Ans and Rousset 1997) that for the same sequential learning task, a high level of retroactive interference was present with pseudorehearsal without the reverberating process, whereas this interference was dramatically reduced when using the activity reinjection mechanism. It must be noticed that pseudorehearsal was explored in the framework of Hopfield-type nets (Robins and McCallum 1998), where extra (or 'spurious') attractors created in state space during learning are exploited for preserving a previously learned population. The effects of these extra attractors in the specific dynamics of Hopfield nets have something in common with the role of the reverberating process used in the framework of multilayer networks.

2.2. *Simulations*

Throughout this paper, the performances of the proposed reverberating architecture will be explored using two pairs of populations of items. The first pair defines a first condition for which the two populations are *a priori* mutually 'compatible'. The second pair defines a second condition for which the two populations are *a priori* mutually 'incompatible'. This notion of compatibility is at this point used with reference to common sense: 'doing a related sort of task'. In the compatible condition, the first population is a set of items representing the addition of two numbers; these two operands and their associated result are expressed in decimal notation. The second population is a set of items representing the same addition operation but with operands and results expressed this time in octal notation. For example, to the decimal add-item '07 + 46 = 53', belonging to the first population, will correspond (for the same numerosity) the octal add-item '07+56=65' in the second population. In the incompatible condition, the first population is a set of items representing an operation, denoted *max*, for which the two arguments are the same as those of the previous decimal addition, but with a result unrelated to an addition operation. The *max* operator is defined as follows: the first digit of the result is the greatest of the two first position digits belonging, respectively, to the two operands, and the resulting second digit is the greatest of the two second position digits of the two operands. The second population of the incompatible condition is again the octal addition population. Taking the previous example, the *max*-item '07 *max* 46=47' will correspond to the octal add-item '07+56=65'.

For the three populations considered, the operands and results are limited to numbers with two digits. The two operands both lie between numerosity 1 and numerosity 47. The associated result must be less than numerosity 64, because the corresponding octal number has more than two digits from numerosity 64 (100 in octal notation). For processing in the learning network, items will be presented in binary code: digits belonging to the same number will be coded separately, with a maximum of three bits per digit. For example, the decimal add-item '07+46=53' will be binary coded in the following way: [(000) (111)] + [(100) (110)] = [(101) (011)]. This 3-bit coding implies that operation items containing numbers with digits '8' or '9' have to be removed from the decimal addition and *max* operation populations, as well as the corresponding items with the same numerosity in the octal addition population. Finally, with all these constraints, each population contains 916 operation items.

In order to highlight catastrophic forgetting in sequential learning tasks and show how the reverberating architecture overcomes this problem, two simulations are performed. The first one in the compatible condition where the decimal addition population (916 items), denoted *Dec-Add*, is learned first and a subset belonging to the octal addition population is subsequently learned. This subset, denoted *Oct-Add*, is composed of 229 items randomly chosen among the 916 items of the whole octal population, the remaining items being reserved for achieving the generalization tests in section 3. The second simulation is performed in the incompatible condition where, after learning the *max* operation population (916 items), denoted *Max-Op*, the same previous *Oct-Add* subset (229 items) is then learnt. Training a given operation item (binary coded) by the first network *NET 1* of the dual architecture is achieved by presenting the two operands over its input layer and the result over its output layer. It may happen that some items belonging to two populations processed in the same condition contain the same operands but give rise to a different result (e.g. '5+7 = 12' and '5+7 = 14', respectively, for decimal and octal addition). To avoid these 'ambiguities', all operand pairs are systematically accompanied by an operator pattern specifying the task: [10] and [01] coding, respectively, for the two distinct operations represented by the two populations processed in the same condition. The same coding [10] is used for both the *Dec-Add* and *Max-Op* operations since the simulations performed on compatible and incompatible conditions are independent (only two, never three, populations are processed in the same condition). The two networks have 14 input units for the two operands and the operator pattern, six output units for the result and 40 hidden units.

For each of the two simulation conditions, we start from an initial state of the system in which *NET 1* has already completely learn the first set of 916 items

(*Dec-Add* or *Max-Op* population) and *NET 2* is still empty. In stage (I), *NET 2* is trained on pseudoitems generated by *NET 1* from random input seeds. When stage (II) is at work, *NET 1* is trained on the second population of 229 items (the *Oct-Add* subset in the two conditions) concurrently with pseudoitems originating from *NET 2*, that is, with a self-refreshing of knowledge related to the previously learnt population. In this learning process, actual and pseudoitems occur alternatively, each of them inducing its corresponding weight modification in the *NET 1* network. In this continuous flow, each occurrence of an actual item (taken at random without replacement) is systematically followed by *N* (a simulation parameter) occurrences of pseudoitems generated on line from *NET 2*. It is noteworthy that this on-line flow of one actual item for *N* pseudoitems, inducing a serial weight updating, does not require any training buffer. After a number of training cycles (one cycle corresponding to a whole population presentation), learning the new population is considered as complete when the following criterion is reached: the absolute value of the difference between the computed activity of each input–output unit and the corresponding component of the auto-hetero-associative target pattern has to be less than or equal to 0.1. This criterion has to be satisfied for all the items of the actual population trained. With regard to the interleaved pseudoitems, a learning criterion is of course irrelevant.

In all subsequent simulations, the number *N* of pseudoitems, alternating with one actual item during training *NET 1*, will be fixed to N=1. In the reverberating process generating pseudoitems, the two networks use the same parameter R=5; this parameter was defined earlier as the number of activity reinjections within the auto-associative part of one network before transmitting pseudoitems to the other network. The noise generator produces binary inputs, though random real-valued inputs should be suitable as well. In the backpropagation procedure, the error function to minimize will be the cross-entropy function (Hinton 1989, Plaut *et al.* 1996) and the learning parameters will be 0.01 for the learning rate and 0.5 for the momentum term. The bias term equals one and initial connection weights are taken at random uniformly between –0.5 and 0.5.

Testing a given operation item consists of presenting to the *NET 1* input layer the item operand part (with its related operator pattern) and comparing the computed hetero-associative output with the desired operation result (the auto-associative part concerning operand memory will be not checked here). Any output pattern is considered as being correct if the following, rather rigorous, criterion is satisfied: the absolute value of the difference between each output unit activity and the corresponding target component has to be less than or equal to 0.1. The correctness of a set of items is evaluated from the percentage of correct items calculated over the whole set. The simulation results plotted in figure 2 represent recall performance (per cent of correct items) of a previously learnt population (*Dec-Add* or *Max-Op* populations) in the course of learning the new *Oct-Add* population. Performance is evaluated as a function of the number of training cycles of the new population, one cycle representing, as mentioned already, a whole population presentation. For each condition, recall performance of the previously learnt knowledge is plotted with and without the self-refreshing memory process. Catastrophic forgetting can be clearly observed when self-refreshing is not at work, especially when the two sequentially learnt populations are incompatible. In contrast, when the self-refreshing process works, the retroactive interference is dramatically reduced. In fact, recall performance reaches its initial value (100% correct items) after an early falling

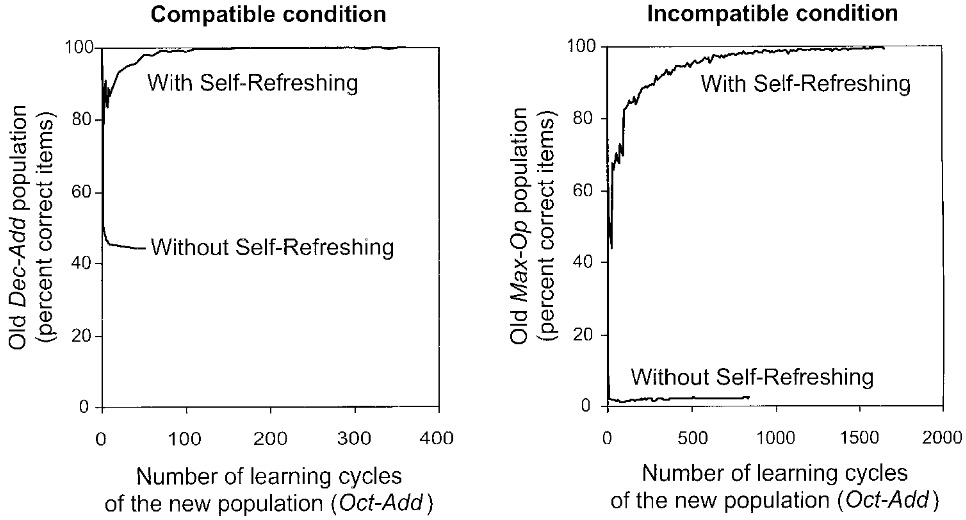

Figure 2. Recall performance (per cent of correct items) of a previously learnt population (old population) as a function of the number of learning cycles of a new population. Graphs originate from 100% performance (before the new population training starts) and stop when the new population learning is completed. Left side: condition in which old (decimal addition) and new (octal addition) knowledge are considered as compatible. Upper graph: when the self-refreshing memory process is at work, perfect recall is reached rapidly after a short restructuring phase. Lower graph: with no self-refreshing memory process, substantial forgetting can be observed. Right side: condition in which old (*max* operation) and new (octal addition) knowledge are considered as incompatible. Upper graph: with self-refreshing, perfect recall is reached after a longer restructuring phase than in the compatible case. Lower graph: with no self-refreshing, severe catastrophic forgetting immediately arises at the beginning of learning of the new population.

performance corresponding to a network connectivity restructuring. This latter is obviously more laborious in the incompatible condition.[1]

We attempted to run the same sequential learning tasks without the reverberating process, i.e. taking $R=0$. The other parameters remained unchanged, in particular we kept the same minimal value for the N parameter ($N=1$). As mentioned already, it was shown (Ans and Rousset 1997) in another sequential learning example that when pseudoitems were generated from a single feedforward pass of activity, recall performance of old knowledge fell down compared with the quasi perfect recall obtained with the reverberating process at work. For the present tasks, a catastrophic blocking of learning was in fact observed. That is, the *NET 1* network could not learn the second set of items (*Oct-Add* population) in the two conditions. This problem originated from the fact that creating pseudoitems on the basis of a single pass of activity from a random input pattern generates too much noisy information in comparison with the reverberating process optimally extracting cleaner knowledge structures.

Another important point must be emphasized and investigated: How many learning iterations are required for a ‘good copy’ of knowledge structure from *NET 1* to *NET 2* during stage (I)? In fact, we opted for overlearning in order to have ideal conditions for reaching maximal performance. This also leads one to get closer to the ‘weight-copying’ method which is commonly used by French (1997) for

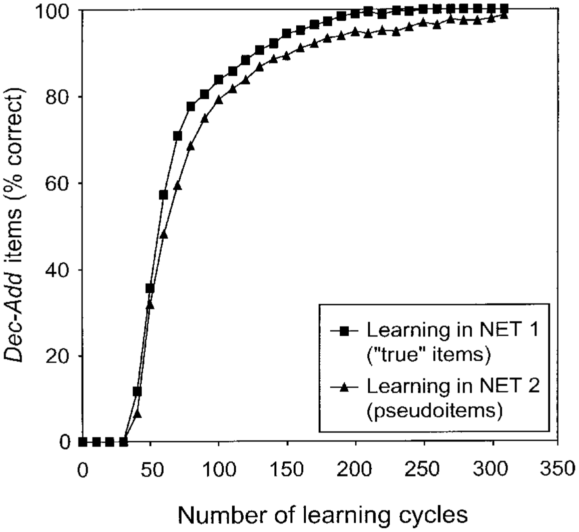

Figure 3. Comparison between learning external items by *NET 1* and learning by *NET 2* of the corresponding pseudoitems originating from *NET 1*, showing how accurately knowledge in the first network is transmitted towards the second.

transmitting new learning from an early processing area to a final storage area (note that the author also compares weight-copying with effective pseudopattern transport). However, it is worth noting that with the reverberating process at work, learning pseudoitems in *NET 2* from *NET 1* may be practically as fast as learning actual external items in *NET 1*. This can be easily evidenced by simulation. The initial training of the *Dec-Add* population by the empty *NET 1* network was compared with training, by the empty network *NET 2*, of the corresponding pseudoitems generated by *NET 1* after this latter had completely learnt the actual items (at the 0.1 learning criterion). This comparison is shown in figure 3, in which recall performance of the *Dec-Add* population (checking, as above, hetero-associative output patterns against desired responses at the 0.1 testing criterion) is plotted for two cases, the first case as a function of the number of learning cycles in *NET 1* of actual decimal add-items, one cycle corresponding to the whole population size, i.e. 916 items. The second case is plotted as a function of the number of pseudoitems trained in *NET 2*, but with this number translated in cycles, one cycle again corresponding to 916 pseudoitems, for enabling comparison with the first case. In this example, it is clear that learning of actual items and pseudoitems has similar dynamic profiles and reach practically the same final accuracy. In short, knowledge from one network is accurately transported to the second.

3. Knowledge transfer without catastrophic forgetting

In the framework of connectionist modelling, knowledge transfer between tasks has been studied through two main directions (cf. for a review, Pratt and Jennings 1996 and Robins 1996b). The former, referred to as 'functional' transfer (Silver and Mercer 1996), is based on concurrent learning. Different sets of items representing different knowledge domains are simultaneously trained from a *tabula rasa* (a network with random connection weights). The second approach, the 'representational' transfer (Baxter 1996), involves sequential learning. This highlights the advantages of learning a task on the basis of a set of initial connection weights built up during the previous learning of a related task. However, in this case, due to the catastrophic forgetting, the new task learning may be *only initially* oriented by previous learning: what a network already 'knows' cannot be generally saved for a long time and hence cannot be reused in subsequent acquisitions. Even if performance of a new task is improved when learning occurs after another related task, what could we say about the real interest of such a result from the point of view of human cognition since the related old knowledge is subsequently abolished? Fortunately, as promising solutions for catastrophic interference are now available, it will then be possible to study, in a more plausible and precise manner, the sequential transfer processes which are obviously of fundamental interest in cognitive modelling, especially in modelling cognitive development.

In this paper, we simply highlight some fundamental consequences of learning with self-refreshing with respect to knowledge transfer. Transfer will be explored with regard to learning speed, ability to generalize and vulnerability to network damages of the same target population according to whether the target population is learned *concurrently* with another population, called the 'context' population, or *after* complete learning of this context population. The same examples as used above will be used in the two conditions referring, respectively, to compatible and incompatible knowledge between the context and target population of items.

3.1. *Learning speed*

Simulation results on learning speed are shown in figure 4. Recall performance (per cent of correct items at the 0.1 testing criterion) of the *Oct-Add* target population

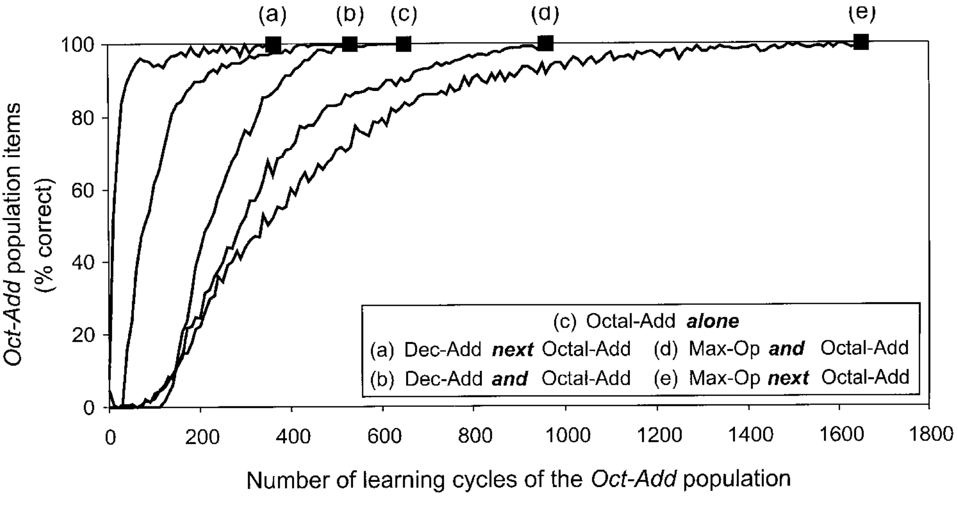

Figure 4. Recall performance of the same target population of items during its training according to whether this population is learnt alone, after or concurrently with a context population. (a, b) Sequential and concurrent learning, respectively, in the condition in which the context and target populations are compatible. (c) Isolated learning of the target population. (d, e) Concurrent and sequential learning, respectively, in the incompatible condition. The closed squares refer to the number of cycles required to reach the learning criterion for the target population.

(229 items) is plotted as a function of the cycle number of its own training in five cases. The two graphs (a) and (b) show learning dynamics in the knowledge-compatible condition for, respectively, sequential learning with self-refreshing (denoted *Dec-Add next Oct-Add*) and concurrent learning (denoted *Dec-Add and Oct-Add*). The two graphs (d) and (e) represent learning dynamics in the incompatible condition for, respectively, concurrent (denoted *Max-Op and Oct-Add*) and sequential (denoted *Max-Op next Oct-Ad*) learning. Graph (c) refers to the isolated learning of the target population (denoted *Oct-Add alone*) from an empty network (*tabula rasa*). Performance comparison is quite straightforward. In the compatible condition, the fastest training is obtained for sequential learning followed by concurrent and isolated training. In the incompatible condition, the reverse pattern occurs: isolated training of target knowledge is faster than concurrent training, which is in turn faster than sequential learning.

3.2. *Ability to generalize*

With respect to the generalization of the target population ability, the same simulations are considered except that what is now tested is the capacity of the system to infer correct outputs in response to new inputs. That is, to produce correct octal addition results in response to octal operands never presented to the learning architecture. These operand pairs are chosen in the subset of 687 unlearnt items remaining from the initially constructed set of 916 octal additions. Among these octal items, we kept only 338 'completely new' operand pairs with respect to both the learnt target and context populations (this prevents any misleading generalizations). Simulation results are shown in figure 5 using the same learning cases and notations as above. The percentage of correct octal addition, in response to the 338 new operand pairs, is plotted as a function of the cycle number of learning the *Oct-Add* population of 229 items. In these tests, the rigorous 0.1 testing criterion is used to determine correct generalizations. It is clearly observed that, in the knowledge-compatible condition, generalization performance of the target population is best for sequential learning, followed by concurrent and isolated learning. In contrast, in the incompatible condition, performance generalization of the isolated learned knowledge is the best, followed by concurrent and sequential learning. As in figure 4, the closed squares refer to the cycle when the 0.1 learning criterion is reached for the target population. Moreover, learning is largely extended in order to emphasize that performance hierarchy does not change during overlearning.

It could be argued that, during concurrent learning, four context items for one target item are jointly trained (respectively, from 916 and 229 items), whereas during sequential learning, only one pseudo-context item (N=1) for one target item are jointly trained. To check that the observed difference of performance did not originate from this asymmetry, the two sequential learning simulations were performed again, this time taking N=4, i.e. taking four pseudo-context items for one actual target item for learning in *NET 1*, in order to stand in an analogous situation to that of concurrent learning. These simulations indicated that in the compatible condition performance of sequential learning, N=1 and N=4 were quasi identical, whereas in the incompatible condition a slight change was observed (increase of performance with N=4), without altering the observed hierarchy between isolated, concurrent and sequential learning.

As mentioned previously the *Oct-Add* target population is composed of 229 items randomly chosen among a larger set of octal addition exemplars. It happens

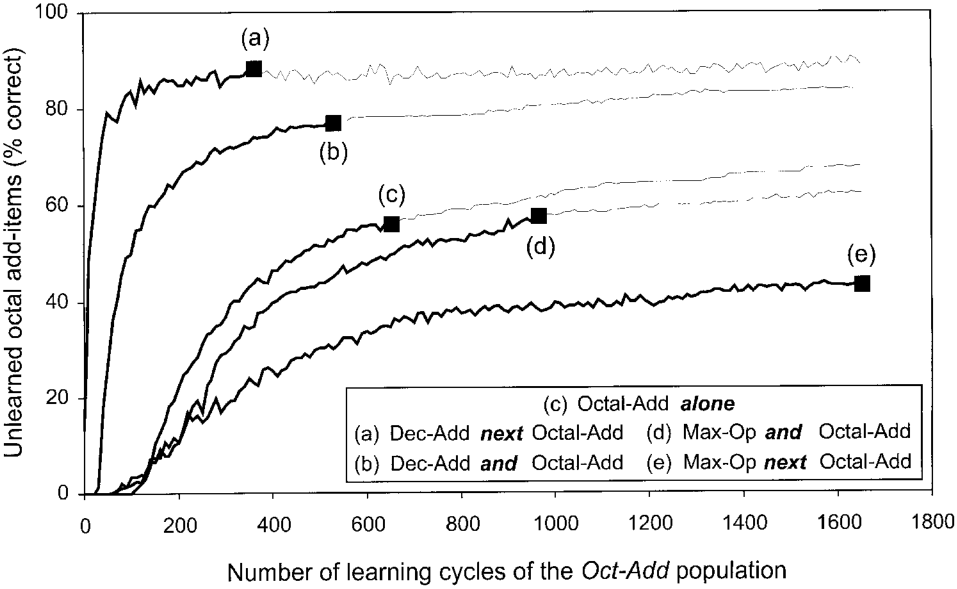

Figure 5. Generalization performance of the same target population of items in the course of its training according to whether this population is learnt alone, after or concurrently with a context population. (a, b) Sequential and concurrent learning, respectively, in the condition in which the context and target populations are compatible. (c) Isolated learning of the target population. (d, e) Concurrent and sequential learning, respectively, in the incompatible condition. The closed squares refer to the cycle when the learning criterion is reached for the target population and thin lines refer to its overlearning.

that this target population contains some items which are shared with the context populations considered: the 'shared items' are the ones where the same operands give the same result. Although in the set of the 338 new operands pairs used in generalization tests there is no operand pair contained either in the learnt target population or in the learned context population, one could argue that the presence of shared items between the two learnt populations could distort the generalization performances obtained above. In particular, in the compatible condition, one could think that we place ourselves in a particularly favourable condition for generalization, restricting suspiciously its significance with regard to transfer. To verify that it is not the case, we replicated the simulations, discarding in the context populations the items shared with the *Oct-Add* population. As can be seen in figure 6, these simulation replicated the pattern of generalization performance observed previously (figure 5). This indicates that shared items are not at the root of the positive and negative effects observed, respectively, in the compatible and incompatible conditions. Compatibility between populations hence refers to the task they represent, not to the fact they share or do not share some items.

3.3. *Sensitivity to damages*

We start from a state in which the target *Oct-Add* population has been learnt by an intact architecture in each of the five previous training cases. The vulnerability of the architecture after normal learning is simply explored by making 160 lesions within the hidden layer of the *NET 1* network, each consisting of removing five units

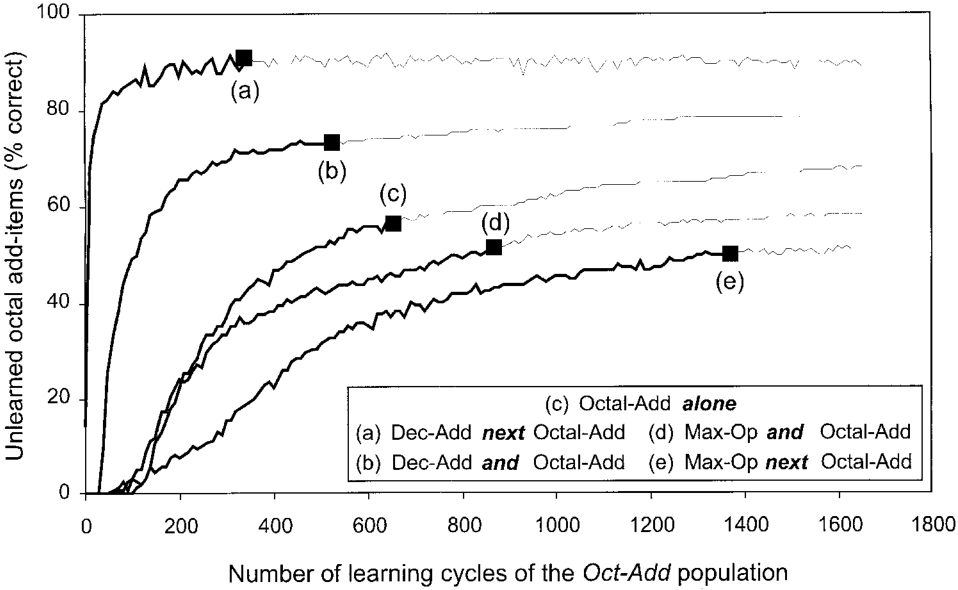

Figure 6. Replication of the generalization simulations (figure 5), only discarding in the context populations the items shared with the target *Oct-Add* population. (a, b) Sequential and concurrent learning, respectively, in the condition in which the context and target populations are compatible. (c) Isolated learning of the target population. (d, e) Concurrent and sequential learning, respectively, in the incompatible condition. The closed squares refer to the cycle when the learning criterion is reached for the target population and thin lines refer to its overlearning.

at random. For each of these five-cell lesions, recall and generalization performance of the learnt *Oct-Add* population are then checked in the five learning cases. Results are given in table 1, in which recall and generalization performance are averaged over the 160 lesioned networks. Recall performance is, as above, expressed as the percentage of correct output patterns in response to the 229 already learnt octal add-operands. Generalization performance is expressed as the percentage of correct outputs in response to the 338 new octal add-operands. If the quantitative differences are not dramatic, they can, however, reflect important properties of each type of learning. Their statistical significance was evaluated using Student's *t*-tests.

Table 1. Recall and generalization performance (per cent item correct at 0.1 testing criterion), averaged over 160 lesioned networks, checked on the same population B of items previously learnt in an intact network according to several conditions.

	Learning conditions				
	Compatible		Isolated	Incompatible	
	A *next* B	A *and* B	B *alone*	A *next* B	A *and* B
Recall	42.10	38.60	33.89	24.57	22.12
Generalization	38.25	32.62	24.18	16.37	15.99

B: *Oct-Add* target population. A: context population (*Dec-Add* or *Max-Op* population, respectively, in the compatible and incompatible learning conditions). A *next* B: sequential learning. A *and* B: concurrent learning. B *alone*: isolated learning.

- With respect to recall, in the compatible condition, sequential learning induces a higher performance than concurrent learning ($t(159)=3.45$; $p<0.001$), which in turn induces higher performance than isolated learning ($t(159)=4.06$; $p<0.001$). In the incompatible condition, isolated learning induces higher performance than the sequential one ($t(159)=7.86$; $p<0.001$), witch in turn surprisingly (with respect to normal recall) induces better performance than the concurrent one ($t(159)=2.45$; $p<0.05$).
- With respect to generalization, in the compatible condition, sequential learning induces a higher performance than concurrent learning ($t(159)=5.57$; $p<0.001$), which in turn induces higher performance than isolated learning ($t(159)=8.78$; $p<0.001$). The pattern is less contrasted in the incompatible condition: isolated learning induces better performance than sequential and concurrent ones (respectively, $t(159)=9.55$; $p<0.001$ and $t(159)=9.22$; $p<0.001$); however, the latter two do not differ ($t(159)=0.44$; $p=0.65$).

The meaning of the preceding analyses can be extended by the analysis of figures 7 and 8, which give the percentage of lesioned networks producing a given percentage of correct responses. It appears clearly that the preceding quantitative results reflect a general and uniform phenomenon that cannot be restricted to an isolated resistance of some of the network sequentially trained. The higher resistance to damages evidenced by sequential learning is hence not fortuitous. This lower vulnerability constitutes one further aspect of the general superiority of sequential learning in the compatible condition. Moreover, its general lower performance in the incompatible condition is surprisingly reduced or even reversed in lesioned network simulations.

4. Conclusions

We have described a learning connectionist architecture which overcomes catastrophic forgetting in sequential learning. Its basic principle lies in that new knowledge has to be learnt along with an internally generated activity reflecting the

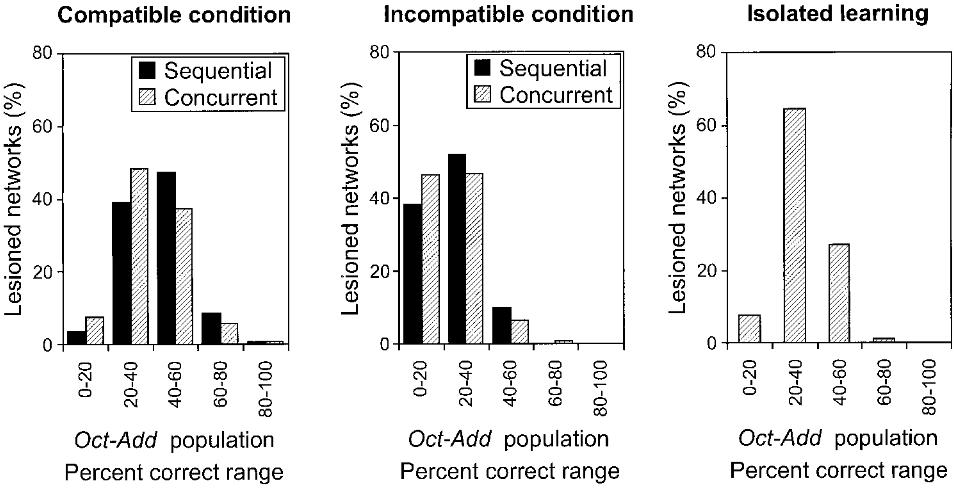

Figure 7. Percentage of lesioned networks (among 160) producing a given recall performance (percentage of correct octal add-items) according to five conditions (see text).

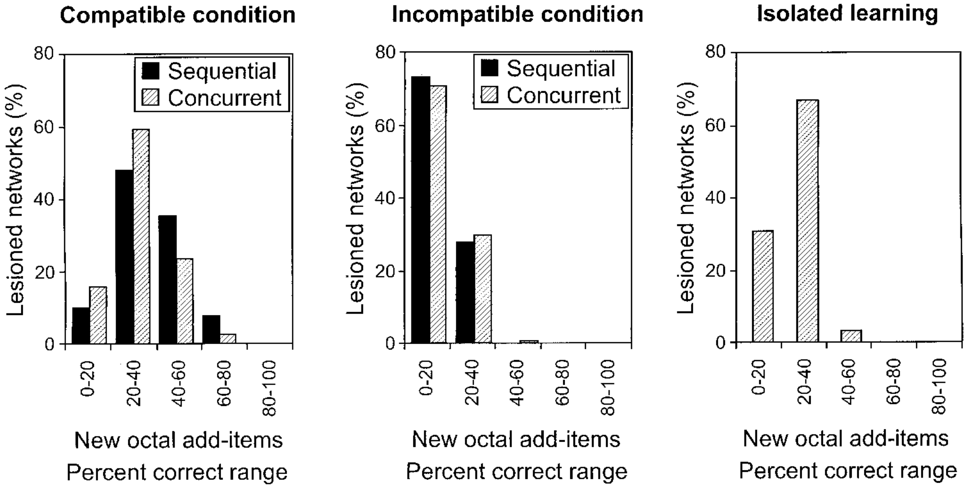

Figure 8. Percentage of lesioned networks producing, in several conditions (see text), a given generalization performance (percentage of correct output patterns in response to new octal add-operands).

network history, that is, self-refreshing of its memory. What essentially distinguishes our approach from the other studies using pseudorehearsal in feedforward multilayer networks is the introduction of a reverberating process for generating pseudoitems. This process, which tends to go up to network attractors, is more suitable for capturing optimally the deep structure of old knowledge distributed within connection weights than a single feedforward pass of activity. In the latter case, catastrophic blocking of learning new knowledge can occur. The reverberating mechanism, intervening between input and hidden network layers, always requires an auto-associative part in the learning architecture even when hetero-associations are the focus of simulated tasks. The utility of the activity reinjection process in revealing the structure of learnt information has already be shown in connectionist modelling of identification (Rousset *et al.* 1988, Wang *et al.* 1989). In further research, the self-refreshing memory model will be extended to pseudorehearsal of ordered time-series which are stored by recurrent networks (e.g. Jordan 1986, Elman 1990, Reiss and Taylor 1991, Ans *et al.* 1994).

An important point has to be clarified. One could think that the reverberating process would converge close to network attractors, which might be systematically previously learnt actual items. This would mean that the network auto-associative part would in fact implement a sort of ‘clean-up’ circuitry. If this were the case, one could then argue that a space coverage problem could arise: pseudopatterns would always be actual items and some of those could be systematically neglected for more attractive ones. In this situation, the neglected items would never be transmitted from one network to the other. We studied the generated pseudopatterns and it was observed that they were mainly actual and new but ‘legal’ items, the latter being generalized items which satisfy the operation learnt by the network-generating pseudoitems. There were also pseudopatterns which were new ‘illegal’ pseudo-associations not satisfying the learnt operation. It is thought that these illegal pseudopatterns, representative of some input–output function captured by the network, are nevertheless suitable for accurate transport, towards the other network, of the actual items previously learnt. This point is not easy to evidence because, in the case of very structured information (as in the addition or *max* operation), the majority of new generated pseudopatterns are generalized items fitting the learnt operation. In order to make clearer the role of the illegal pseudoitems, we performed the following simple simulation over the same dual architecture, with the same parameters, but with other data. Twenty *arbitrary* hetero-associations of binary coded patterns (a completely *unstructured* domain) were first completely learnt by the *NET 1* network (the auto-associative and hetero-associative parts were jointly learnt as above). A given hetero-associative item was constituted by an input with 32 bits chosen at random and a desired output, also of 32 bits, taken at random. We also defined a distance between two patterns *X* and *Y* of size *Q* as:

$$d(X,\ Y) = \left[1/Q \sum_{i=1}^{Q} (x_i\ -\ y_i)^2 \right]^{1/2}$$

with *d* lying between 0 and 1. When the reverberating pseudopattern generation was at work within *NET 1*, the pseudopatterns transiting from *NET 1* to *NET 2* were filtered: only those whose distance *d* was greater than 0.5 with each of the 20 previously learnt patterns were kept for training in *NET 2* (the others being discarded). Using this filtering, 87.3% of pseudopatterns were discarded and *NET 2* was in fact trained on only 12.7% of the generated pseudopatterns which were far from the actual learnt patterns (0.5 is rather a high distance). In this situation, we observed that *NET 2* learnt perfectly, on the basis of very illegal items, the 20 arbitrary associations previously learnt in *NET 1*. This result clearly indicates that, even in the case of an essentially unstructured domain, the involvement of actual learnt items in memory transport between the two networks is not a prerequisite. Pseudoitems allow *NET 2* to acquire some input–output function which has, in particular, the essential property to fit accurately the actual items. If one wants to consider the reverberating process as a clean-up mechanism, then one must keep in mind that it is not simply a means for selecting old items but, above all, a way for selecting an optimal approximation of the input–output function within the network sending pseudopatterns.

Knowledge transfer was evaluated with regard to learning speed, ability to generalize and vulnerability to network damages. We showed that for two related tasks, knowledge transfer is more efficient when using sequential learning than concurrent learning. This result was obtained in the framework of arithmetical metaphors which were taken as representative examples of structured sets of items. It is worth noting that the efficiency of sequential learning stems from the fact that the self-refreshing memory process makes it possible to maintain previously learned knowledge, hence improving transfer during subsequent learning of related tasks. What a network (with self-refreshing memory) knows about something will be saved for a long time and therefore possibly reused in subsequent acquisitions of other things. This contrasts with sequential learning without pseudorehearsal, where old knowledge is likely to be destroyed as a network is faced with new acquisitions. In this case, since previously learnt knowledge is lost it cannot be obviously reused.

The coupled networks composing the proposed dual-network architecture do not need to have the same structure (i.e. the same number of units in the same number of hidden layers) or to work with the same learning rule. Simulations supporting these points have not been presented in this paper because they have already been reported in studies performed in the framework of pseudorehearsal in feedforward networks (Frean and Robins 1997, Robins 1997). When using a reverberating architecture with one or two networks composed of several hidden layers, the only additional feature to specify is that activity reinjections have to be carried out between the input layer and the last hidden layer that is connected to the output. The fact that the two coupled networks can work with different structures and learning rules constitutes an essential point when relating pseudorehearsal to brain processes. Indeed, knowledge transport which is carried from *NET 1* to *NET 2* underlies an original mechanism that would be able to 'copy-paste memory' between two substantially different brain structures, even in the absence of the constituent learning episodes of this memory. We shown that the use of the reverberating process results in knowledge transport between networks with virtually no loss of information, in particular with structured knowledge (see Ans and Rousset 1997 for a study using unstructured knowledge). This preservation property is in fact required for any memory model proposing a possible means for consolidation in long-term memory.

In some respects the dual-system approach (Ans and Rousset 1997, French 1997) goes in the same direction as that of McClelland *et al.* (1995), claiming, on the basis of neurophysiological and neuropsychological data, that two complementary learning systems are necessary for consolidation without catastrophic forgetting. Pseudorehersal can constitute a neurobiologically plausible and efficient way to transport information between the two systems, and then to maintain long-term memory. Robins (1996a, Robins and McCallum, 1998) suggested that consolidation of information in the neocortex may occur by means of pseudopatterns generated during rapid eye movements (REM) sleep phases. In that regard, the similar cerebral activation observed between learning of an artificial grammar and the subsequent REM sleep phase (Maquet *et al.* 1998), can be viewed as congruent with Robins's proposal. Further investigations using cerebral activation imagery are thus likely to give reliable neurophysiological support to the pseudorehearsal hypothesis.

The use of two separate networks to implement self-refreshing also opens new horizons for exploring cognitive processes related to normal and pathological forgetting. In particular, the dual nature of the memory model leads naturally to envisaging the behavioural consequences of lesioning connections from *NET 1* towards *NET 2*, while keeping intact the ones going in the reverse direction. This will lead to preservation of the stage (II) process in the presence of a deficient stage (I) process. This asymmetric architectural damage would induce an anterograde amnesia behaviour since information learnt after lesioning can no longer be transmitted to *NET 2*. Hence, post-damage learnt information will no longer be refreshed during subsequent acquisitions of other information and will consequently be lost due to the classical catastrophic interference. As a direct consequence of pseudorehearsal, no severe retrograde amnesia should occur since information learnt before lesions will continue to be refreshed, and hence will be maintained. This general pattern still needs a detailed examination in forthcoming investigations. In particular, the functioning of the two reverberating networks could give some insights for explaining precisely the limited gradients of retrograde deficit observed in anterograde amnesic patients.

Acknowledgements

This research was funded by the Centre National de la Recherche Scientifique (CNRS UMR 5105). We thank Robert French and Anthony Robins for their helpful comments on an earlier version of this article. We also thank Sonia Kandel for the English revision of the manuscript.

Note

1. It was verified that these catastrophic forgetting levels were not linked to operator coding or to the presence of ambiguous items. On the one hand, pilot simulations were performed taking pattern operators with varying sizes and structures. These simulations replicated the observed results. On the other hand, it was also verified that items belonging to two sequentially learnt populations, but composed of the same operands giving rise to a different result (the so-called 'ambiguous' items discriminated by the operator pattern), do not have a major influence on the catastrophic forgetting extent. Pilot simulations on sequential learning were performed with the same first population, but with a second population in which ambiguous items, with respect to the first population, were discarded (the operator pattern, becoming unnecessary, was also removed for the two populations). The obtained results were closely similar to the ones depicted in figure 2.

References

Ans, B., Carbonnel, S. and Valdois, S., 1998, A connectionist multiple-trace memory model for polysyllabic word reading. *Psychological Review*, **105**: 678–723.

Ans, B., Coiton, Y., Gilhodes, J. C. and Velay, J. L., 1994, A neural network model for temporal sequence learning and motor programming. *Neural Networks*, **7**: 1461–1476.

Ans, B., and Rousset, S., 1997, Avoiding catastrophic forgetting by coupling two reverberating neural networks. *CR Academie Science Paris, Life Sciences*, **320**: 989–997.

Baxter, J., 1996, Learning internal representations. In D. S. Touretzky, M. C. Mozer and M. E. Hasselmo (eds) *Advances in Neural Information Processing Systems*, Vol. 8 (Cambridge MA: MIT Press).

Carpenter, G. A., and Grossberg, S., 1988, The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, **21**: 77–88.

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

Frean, M. and Robins, A. V., 1997, *Catastrophic Forgetting in Neural Networks: A Review and an Analysis of the Pseudorehearsal Solution*, Technical Report AIM-36-97-2, Department of Computer Science, University of Otago, New Zealand.

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 distributed, orthogonal representations to reduce catastrophic interference. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society* (Hillsdale NJ: Lawrence Erlbaum), pp. 335–340.

French, R. M., 1997, Pseudo-recurrent connectionist networks: an approach to the 'sensitivity-stability' dilemma. *Connection Science*, **9**: 353–379.

French, R. M., 1999, Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, **3**: 128–135.

French, R. M., and Mareschal, D., 1998, Could category-specific semantic deficits reflect differences in the distributions of features within a unified semantic memory? In *Proceedings of the 20th Annual Conference of the Cognitive Science Society* (Hillsdale NJ: Lawrence Erlbaum), pp. 374–379.

Grossberg, S., 1987, Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, **11**: 23–63.

Hebb, D. O., 1949, *The Organization of Behavior* (New York: Wiley).

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

Hinton, G. E., 1989, Connectionist learning procedures. *Artificial Intelligence*, **40**: 185–234.

Hintzman, D. L., 1986, 'Schema abstraction' in a multiple-trace memory model. *Psychological Review*, **93**: 411–428.

Jordan, M. I., 1986, *Serial Order: A Parallel Distributed Processing Approach*, Technical Report ICS-8604, University of California at San Diego, CA.

Kanerva, P., 1988, *Sparse Distributed Memory* (Cambridge MA: MIT Press).

Kortge, C. A., 1990, Episodic memory in connectionist networks. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society* (Hillsdale NJ: Lawrence Erlbaum), pp. 764–771.

Krushke, J. K., 1992, ALCOVE: an exemplar-based connectionist model of category learning. *Psychological Review*, **99**: 22–44.

Krushke, J. K., 1993, Human category learning: implications for backpropagation models. *Connection Science*, **5**: 3–36.

Lewandowsky, S., 1991, Gradual unlearning and catastrophic interference: a comparison of distributed architectures. 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), pp. 445–476.

Lewandowsky, S., 1994, On the relation between catastrophic interference and generalization in connectionist networks. *Journal of Biological Systems*, **2**: 307–333.

Lewandowsky, S., and Li, S. C., 1995, Catastrophic interference in neural networks. Causes, solutions, and data. In F. N. Dempster and C. Brainerd (eds) *New Perspectives on Interference and Inhibition in Cognition* (New York: Academic Press), pp. 329–361.

Maquet, P., Petiau, C., Peigneux, P., Phillips, C., Péters, J. M., Cleeremans, A., Smith, C., Van der Linden, M., and Luxen, A., 1998, Reactivation during rapid eye movement (REM) sleep of cerebral areas involved in the execution of a serial reaction time (SRT) task. *Society of Neuroscience Abstract*, 393.

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

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*, Vol. 24 (New York: Academic Press), pp. 109–165.

McRae, K., and Hetherington, P. A., 1993, Catastrophic interference is eliminated in pretrained networks. In *Proceedings of the 15th Annual Conference of the Cognitive Science Society* (Hillsdale NJ: Lawrence Erlbaum), pp. 723–728.

Murre, J. M. 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: Lawrence Erlbaum), pp. 54–59.

Plaut, D. C., McClelland, J. L., Seidenberg, M. S., and Patterson, K., 1996, Understanding normal and impaired word reading: computational principles in quasi-regular domains. *Psychological Review*, **103**: 56–115.

Pratt, L., and Jennings, B., 1996, A survey of transfer between connectionist networks. *Connection Science*, **8**: 163–184.

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

Reiss, M., and Taylor, J. G., 1991, Storing temporal sequences. *Neural Networks*, **4**: 773–787.

Robins, A. V., 1995, Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, **7**: 123–146.

Robins, A. V., 1996a, Consolidation in neural networks and in the sleeping brain. *Connection Science*, **8**: 259–275.

Robins, A. V., 1996b, Transfer in cognition. *Connection Science*, **8**: 185–203.

Robins, A. V., 1997a, Maintaining stability during new learning in neural networks. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (Los Alamos NM: IEEE Computer Society Press), pp. 3013–3018.

Robins, A. V., 1997b, A new method for consolidation and transfer in neural network models. In *Proceedings of the 4th Australian Cognitive Science Conference* (in press).

Robins, A. V., and McCallum, S., 1998, Catastrophic forgetting and the pseudorehearsal solution in Hopfield-type networks. *Connection Science*, **10**: 121–135.

Rousset, S., Schreiber, A. C., and Wang, S., 1988, *Modélisation et Simulation Connexionniste de l'idenfication des Visages en Contexte: Le Système FACENET*, Institut National Polytechnique de Grenoble TIM3-IMAG Technical Report RR-7420-17.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986, Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland and the PDP Research Group (eds) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (Cambridge MA: MIT Press), pp. 318–362.

Sharkey, N. E., and Sharkey, A. J. C., 1995, An analysis of catastrophic interference. *Connection Science*, **7**: 301–329.

Silver, D. L., and Mercer, R. E., 1996, The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. *Connection Science*, **8**: 277–294.

Wang, S., Schreiber, A. C., and Rousset, S., 1989, Connectionist modelling of a cognitive model of face identification. Simulation of context effects. In *Proceedings of the 1st International Joint Conference on Neural Networks*, Vol. 2 (San Diego CA: IEEE & INNS), pp. 549–556.