LETTER

Sequential Learning in Distributed Neural Networks without Catastrophic Forgetting: A Single and Realistic Self-Refreshing Memory Can Do It

Bernard Ans

Psychology and NeuroCognition Laboratory (UMR CNRS 5105)
Pierre Mendes-France University, BP 47, 38040 Grenoble cedex 09, France
E-mail: Bernard.Ans@upmf-grenoble.fr

(Submitted on June 17, 2004)

Abstract – In sequential learning tasks artificial distributed neural networks forget catastrophically, that is, new learned information most often erases the one previously learned. This major weakness is not only cognitively implausible, as human gradually forget, but disastrous for most practical applications. An efficient solution to catastrophic forgetting has been recently proposed for backpropagation networks, the reverberating self-refreshing mechanism: when new external events are learned they have to be interleaved with internally-generated pseudo-events (from simple random activations) reflecting the previously learned information. Since self-generated patterns cannot be learned by a same backpropagation network, because desired targets are lacking, this solution used two complementary networks. In the present paper it is proposed a new self-refreshing mechanism based on a single-network architecture that can learn its own production reflecting its history (i.e., a self-learning ability). In addition, in place of backpropagation, widely considered to be not biologically realistic, a more plausible learning rule is used: the deterministic version of the Contrastive Hebbian Learning algorithm, or CHL. Simulations of sequential learning tasks show that the proposed single self-refreshing memory has the ability to avoid catastrophic forgetting.

Keywords – Catastrophic forgetting, self-refreshing memory, retroactive interference, contrastive Hebbian learning

1. Introduction

Artificial neural networks with highly distributed memory forget catastrophically [1-3] when faced with sequential learning tasks: new learned information most often erases the one previously learned. This major weakness is not only cognitively implausible, as human gradually forget, but disastrous for most practical applications. However, distributed neural systems are extensively used in concurrent learning, mainly for their remarkable ability to generalize and their graceful degradation. In distributed memory, experienced events share the same set of connection weights, which is at the root of the fundamental property of generalization, but is also precisely the root cause of catastrophic interference: in sequential learning tasks new learned information modifies the same set of weights that represents the previously learned information. Numerous authors have developed ways to overcome this stability-plasticity dilemma (for a review see [4]). The simplest way to avoid catastrophic forgetting is to 'rehearse' the old items as new learning occurs, which amounts to transform sequential learning into concurrent learning. This trivial solution is uninteresting for practical applications and unrealistic for human memory since it requires permanent access to all previously experienced events. Another solution, not requiring permanent access to old events, is to use a *pseudorehearsal* mechanism in place of a true rehearsal process, that is, when new external patterns are learned they are interleaved with *internally-generated* activity patterns. These entities, called *pseudopatterns*, self-generated by the network from just *random* activations, reflect (but are not identical to) the previously learned information. It has now been established [5-8]

that this pseudorehearsal mechanism effectively eliminates catastrophic forgetting. In its most efficient implementation [6,8], the pseudorehearsal mechanism uses a *reverberating* process where pseudopatterns are *attractor* patterns generated from multiple reverberations within a recurrent part of the network. This reverberating *self-refreshing* mechanism was also generalized [9,10] to allow learning of multiple temporal sequences without catastrophic interference.

The above cited papers have in common the use of the backpropagation learning algorithm and a *dual*-network architecture in which two complementary networks exchange pseudopatterns. Two networks were needed because a single associative network cannot learn self-generated outputs (pseudopatterns) since the desired target outputs, required in error gradient descent algorithms, are lacking. The main objective of the present paper is to propose a new self-refreshing mechanism based on a *single*-network architecture that can learn its own production reflecting its history. In addition, since the backpropagation learning algorithm is widely considered to be not biologically realistic, it is shown that this single self-refreshing memory can sequentially learn, without catastrophic forgetting, using a more plausible learning rule: the deterministic version of the Contrastive Hebbian Learning algorithm, or CHL [11,12], formally equivalent to backpropagation [13]. Indeed, a crucial feature of CHL is its biological plausibility as it is a Hebbian-type learning algorithm, relying on only local pre- and postsynaptic activities locally available.

2. The Single Self-Refreshing Memory

Figure 1 shows the architecture of an artificial neural network whose task is to learn to associate a set of pattern pairs (input X, target Y) by using the CHL learning algorithm; in the figure large gray arrows stand for modifiable connections between fully connected layers (every unit of a given source layer is connected to all units of the layer pointed out by the arrow). The input and output layers are made up of Winner Take All (WTA) clusters of units and can take two working states. When they work according to the state 'On', the units are competing within each cluster: the unit that gets the larger primary output, the 'winning' unit, has its activity set to one, while all other units are set to zero. When these layers work in the state 'Off ', there is no competition within clusters. Furthermore, once in the state 'On', a layer remains *clamped* in the contrasted activity reached by its clusters (i.e., remains insensitive to changes in its input activation) until it is switched to the *unclamped* state 'Off '. The network comprises also one hidden layer containing units without mutual links. As usual, at each computing step of the network activity, the output of every unit is obtained by applying the standard sigmoid function $\sigma(a) = 1/(1 + \exp(-a))$ to its total input activation, a, computed as the sum of all its inputs weighted by the corresponding connection strengths (including a modifiable bias weight). For the input and output layers, this unit output activity is the primary output computed only from modifiable connectivity, that is, when clusters work in the non-competing state.

One CHL learning pass, related to one training presentation of a given associative pair (X,Y), requires two successive phases. In the first 'minus' phase, the input part X is presented alone to the input layer while the output layer is free, that is, is working under the unclamped state 'Off '. The input layer is forced by the external activity

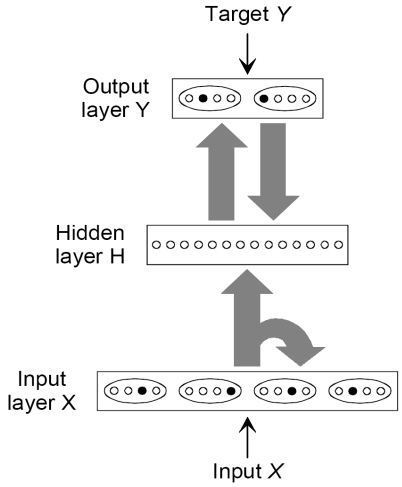

Figure 1. Network architecture. Large arrows stand for modifiable connections between fully connected layers. The input and output layers are made up of WTA clusters.

X and enters in the clamped state 'On' whose the only nonzero components are those corresponding to the winning units of its clamped WTA clusters. To simplify the following, though unnecessary, the external input patterns *X* (and also targets *Y*) will be identical to vectors of 0 and 1 complying with the WTA cluster structure. The clamped input layer activity *X* is propagated through the network and, as the output layer is not clamped, the hidden and output layer activities evolve freely as computing iterations progress between these two layers. In the subsequent 'plus' phase, the corresponding target part *Y* of the current processed pair (*X,Y*) activates and forces the output layer to take its activity pattern, and this layer enters in the clamped working state 'On'. Since the input layer remains clamped during this phase, only one additional computing step is required to get the new resulting hidden layer activity that no longer changes.

Practically, a fixed number, noted *RHY*, of computing iterations between layers *H* and *Y* is taken to delimit the 'minus' phase duration. At the end of this phase, the units of layers *H* and *Y* have reached activities denoted h_j^- and y_i^-, respectively. When the 'plus' phase occurs, these activities shift to new values denoted h_j^+ and y_i^+, respectively (y_i^+ being the elementary components of the external target *Y*). After each presentation of a pair (*X,Y*), comprising both the 'minus' and 'plus' phases, the modifiable connection weights of the network are then updated. The connection weight, denoted w_{ij}, from unit *j* of layer *H* to unit *i* of the output layer *Y* is updated according to the CHL learning rule: $\Delta w_{ij} = \alpha(y_i^+ h_j^+ - y_i^- h_i^-)$, where α is the learning rate. Notice the Hebbian part and the anti-Hebbian part of this rule and its symmetric form implying exactly the same weight change for the reverse connection w_{ij}, from layer *Y* to layer *H*. As this rule preserves any existing weight symmetry, the weight w_{ji} is simply taken the same as the weight w_{ij} (a small amount of weight decay actually would work to symmetrize initially asymmetric weights [12]). The connection weight w_{jk} from unit *k* of layer *X* to unit *j* of layer *H* is updated according to the same rule, but with a learning rate $\beta \neq \alpha$. Since the layer *X* units maintain the same clamped values, noted x_k, during the two activation phases (x_k being the elementary components of the external input *X*), the previous rule for the *X* to *H* connectivity simplifies to $\Delta w_{jk} = \beta(h_j^+ - h_i^-)x_k$. It was shown [13] that the CHL algorithm could be considered as equivalent to the backpropagation algorithm if in particular the activity vector from layer *Y* to layer *H* were reduced by a multiplicative factor, denoted γ, before activating layer *Y*, which also implied to set the learning rate $\beta = \alpha/\gamma$. In parallel to the previous CHL algorithm that implements the (*X,Y*) mapping, one current learning pass includes also the updating of the connection weights from the input layer *X* to itself according to a standard error correction rule: $\Delta w_{kl} = \lambda(x_k - p_k) x_l$, where λ is the learning rate, x_k and x_l are the clamped post and presynaptic activities (binary values, 0 or 1), and p_k denotes the computed output activity of unit *k*, that is, the real-valued activity computed only from the modifiable feedback connections (*X,X*). The modifiable bias weights of all the network units are also updated, simply making equal to one the presynaptic activities in learning algorithms.

Training a set of associations consists of presenting it to the network during a number of learning epochs, each of them comprising one learning pass for every associative pair taken at random within the set (concurrent learning). During a test phase, the previously processed inputs *X* are presented alone to the network input layer and it is checked the ability to produce over the output layer the expected associated targets *Y*. In test phases, the input layer X is permanently clamped in the state 'On' and the output layer works first in the unclamped state 'Off ', during the same number as above, R_{HY}, of computing iterations between layers *H* and *Y*, until it shifts to the state 'On' for which its competing WTA clusters give the clamped output. It is to be noticed that, since the input layer is always clamped in learning and test phases, its feedback modifiable connectivity has no influence on the network performance, and is in fact of no use in a standard concurrent learning of a set of associations (*X,Y*). However, in cases of sequential learning of distinct sets of associations, one after the other, the catastrophic forgetting problem will occur, that is, memory of previously learned sets will be erased when new sets are learned. And it is precisely to avoid this problem that the feedback connectivity within the input layer was added, and now we are going to see why.

A solution to maintain the network memory of previously learned external events when new ones are trained is to learn these latter interleaved with entities reflecting the former. These entities, called peudopatterns or pseudo-episodes in previous papers, and I call now *pseudo-events* (*PE*), consist of attractor patterns that are internally generated by the network from a random seed. How a single system can generate and learn a pseudo-event? First a random input seed activates briefly the network with input and output layers working initially in the unclamped state 'Off '. In the following simulations, the random input seed will be simply a

pattern of 0 and 1, each of theses two values being taken for every input unit according to a 0.5 probability (since WTA clusters units of the input layer are not competing, more than one unit may be active within a cluster). This random input activity is sent through the network and, in particular, is reverberated within the input layer **X** through its feedback connections during a number of computing iterations, denoted R_{XX} (between layer X and itself). After this initial *reverberating process*, the input layer enters in the clamped state 'On', and the resulting contrasted binary pattern, denoted $\hat{X}$ (now consisting only of the clamped winning units in the WTA clusters), continues to activate the hidden and output layers that evolve freely, during the same number, R_{HY}, previously defined for computing iterations between layers *H* and *Y*. This constitutes the same 'minus' phase as above (for external inputs) of the learning algorithm except that it is related now to the *internally generated* and clamped *pseudo-input* $\hat{X}$. When the 'minus' phase ends, the output layer Y enters in the clamped working state 'On' resulting in a contrasted output pattern, denoted $\hat{Y}$ (now consisting only of clamped winning units within the output WTA clusters). This constitutes the same 'plus' phase as described above for external targets, except that it is related now to an *internally generated* and clamped *pseudo-target* $\hat{Y}$. One learning pass for a pseudo-event, that is, for a pair (pseudo-input $\hat{X}$, pseudo-target $\hat{Y}$), uses exactly the same learning algorithm as for external associations (*X*,*Y*), including in particular the same error correction rule for the feedback connections from layer *X* to itself ($\hat{X}$, $\hat{X}$). In this way the *self-refreshing memory mechanism* is defined as follows: during learning of a new set of external associations, every learning pass related to an external input-target pair (actual event) has to be followed by a number, denoted N_{PE}, of learning passes related to N_{PE} pseudo-events internally generated by the network. Below we will show, on a simulation example, that this self-refreshing mechanism can avoid catastrophic forgetting during sequential learning.

3. Simulations

The network has to classify 50 input patterns over 5 categories simply coded by a network output layer made up of a single WTA cluster of 5 units in which the only one active unit represents one of the 5 target categories *Y*. The 50 category members *X* are coded in the input layer by 4 WTA clusters with also a single active unit among 5. For example, a given input pattern *X* = (00010 00001 10000 01000) is to be associated to the target category *Y* = (00100). The categories are arbitrary, which means that any category member *X* consists of clusters in which the single active unit is taken at random, with an associated target category *Y* coded by an active unit taken also at random within the output cluster (with the constraint that the 50 inputs are distinct patterns). In fact, the task of the network is not to learn this list of 50 associative pairs (*X*,*Y*) concurrently, but to learn sequentially the 5 consecutive distinct sets of 10 associations forming the whole list, which means that training a given new set begins once the previous one is completely learned to a given criterion.

Initially, the modifiable weights of the network (with 25 hidden units) are set to random values (between -0.5 and 0.5 according to a uniform distribution) and the first set of item pairs (pairs 1 to 10) is trained (without self-refreshing) using the learning algorithms described above, until the following learning criterion is reached for all the 10 item pairs: for each of the five units belonging to the WTA cluster forming the output layer, the error between its activity computed by the network and the corresponding component of the expected target *Y* has to be less than 0.01 (this value being evaluated with the output cluster in the non competing state 'Off '). Then the four other item sets are sequentially learned, each in the same way as the first (i.e., completely learned until the 0.01 criterion), either without or with the use of the self-refreshing mechanism. The network parameters defined above are: $\gamma = 0.05$ for the factor reducing the backward activity from layer *Y* to layer *H*, $\alpha = 0.05$, $\beta = \alpha/\gamma = 1$, $\lambda = 0.1$ for the learning rates, $R_{HY} = 2$ for the number of computing iterations between layers H and Y. In the case where the self-refreshing mechanism is working, $R_{XX} = 20$ for the number of reverberating iterations (from a random seed within the input layer *X*) required to internally generate a pseudo-event, and $N_{PE} = 10$ for the number of pseudo-events trained during each learning presentation of an actual (external) event.

Once a given set of 10 items is completely learned, tests of retroactive interference are performed on each of the previous sequentially learned sets. In test phases, any input *X* presented alone is considered as correctly categorized if the network produces a corresponding output WTA cluster, in the competing state 'On', which is *identical* to the expected category target *Y*. The correctness of a given set is given by the percentage of correct outputs over the 10 item pairs of the set. After the first set of 10 item pairs was initially learned, an entire sequence that checks retroactive interference is performed as follow. Once the second set is learned (pairs 11 to 20), the correctness of the first set is evaluated. Once the third set is learned (pairs 21 to 30), the correctness of the first and the second set are separately evaluated. And so on, until the fifth set being completely learned (pairs 41 to 50), the correctness of each of the four previous sets are separately evaluated. This learning-test sequence

is done for two conditions: one without the use of the self-refreshing mechanism, the second with this mechanism at work. The results are shown in Figure 2 where each curve gives the percentage of correct outputs for both the current last learned set (i.e., 100%) and for each of the sets sequentially learned before. These percentages are obtained from 12 replications (each for the two conditions) performed with 12 distinct lists of 50 item pairs to be learned and different random weights initializing the networks. It can be observed that without the use of the self-refreshing mechanism, retroactive interference is rather severe. In fact, from the third learned set, the previous ones suffer from catastrophic forgetting. Note that the chance level for a correct output is 20 % since the probability that a unit be correct by chance in a WTA cluster with 5 units is 1/5. On the other hand, when the self-refreshing mechanism is working, catastrophic forgetting is avoided and the global retroactive interference remains reasonable.

4. Conclusion

In contrast with earlier papers, in which two complementary networks were required to implement the self-refreshing mechanism avoiding catastrophic forgetting, I show here that it is possible to implement this mechanism within a single network that in addition learns with a more realistic learning rule (CHL) that the backpropagation algorithm. Future studies are needed to specify at a more formal level the precise nature of the pseudo-events entities and to evaluate the relative efficiency between dual and single network architecture as well as their relative appropriateness with regard to neurobiological and behavioral data or practical applications. It is to be noticed that the WTA clusters, not only put in concrete form the undefined notion of 'clamped' state often used in different learning algorithms (in particular in the standard version of the CHL algorithm), but also offer a means to produce internally generated targets allowing a single network to learn its own production, which is typically unachievable in standard gradient descent and CHL algorithms that require explicit external targets. This last property is fundamental because, additionally to the self-refreshing process, it confers also to the memory network a crucial *self-learning* ability, which, in particular, can account for frequency effects related to production without supervisor (for example, everyday reading of more or less frequent words without a teacher giving the desired pronunciation).

Simulation results have shown that catastrophic forgetting could be eliminated within a neural architecture with only one distributed network. The residual retroactive interference mainly results from the arbitrary nature of the learned associative pairs and also from the rather small size of the learning network with therefore limited resources. If the associations to be learned were more structured and the network larger retroactive interference would be a lot lesser. However, some retroactive interference is the price to pay to save the ability to generalize, distinctive and crucial property of highly distributed networks in neural information processing. Moreover, with regard to human long term memory, some degree of gradual forgetting is tolerable. Interestingly about that, one can observe in Figure 2 that the earliest memorized set of events resists to forgetting better than the next ones

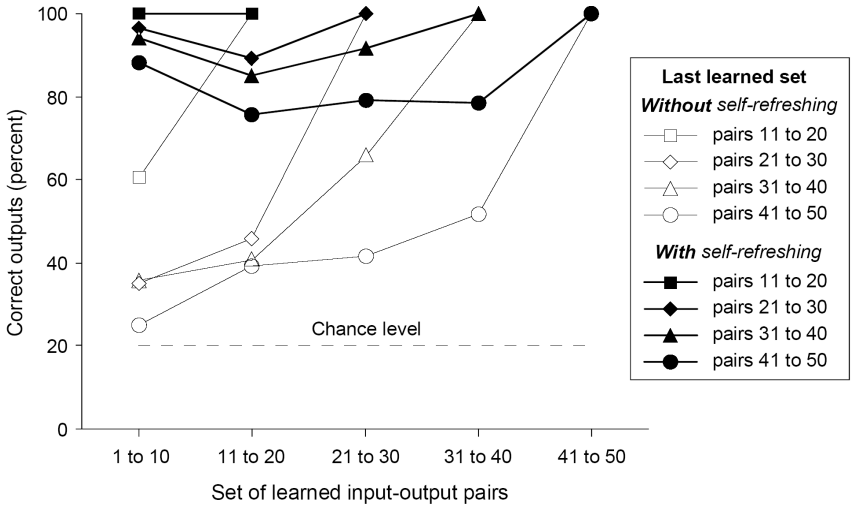

Figure 2. Retroactive interference (percent correct outputs averaged on 12 replications) over sets of input-output pairs sequentially learned before the last learned set. Without the self-refreshing mechanism, retroactive interference is severe and, from the third learned set, the previous ones suffer in fact from catastrophic forgetting. With the self-refreshing mechanism, catastrophic forgetting is avoided.

when the self-refreshing mechanism is working. This simulation result can be compared, of course to some extent and at a very different scale, to behavioral data obtained on autobiographical memory across human lifespan [14] that present the same higher performance of reminiscence occurring for the early events experienced after the childhood amnesia period. Notice that, in the presented simulation, this phenomenon has to be attributed to the self-refreshing mechanism since, when this mechanism does not come into play, forgetting of the earliest memorized events is precisely at a maximum level.

Acknowledgment : This research was supported in part by the French government (CNRS UMR 5105) and by a research grant from the European Commission, "Basic mechanisms of learning and forgetting in natural and artificial systems" (HPRN-CT-1999-00065).

References

[1] M. McCloskey and N.J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *The Psychology of Learning and Motivation, Vol. 24*, Edited by G.H. Bower, Academic Press, New York, pp. 109-165, 1989.

[2] R. Ratcliff, "Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions," *Psychological Review*, Vol. 97, pp. 285-308, 1990.

[3] N.E. Sharkey and A.J.C. Sharkey, "An analysis of catastrophic interference," *Connection Science*, Vol. 7, pp. 301-329, 1995.

[4] R.M. French, "Catastrophic forgetting in connectionist networks," *Trends in Cognitive* Sciences, Vol. 3, pp. 128-135, 1999.

[5] A.V. Robins, "Catastrophic forgetting, rehearsal and pseudorehearsal," *Connection Science*, Vol. 7, pp. 123-146. 1995.

[6] B. Ans and S. Rousset, "Avoiding catastrophic forgetting by coupling two reverberating neural networks," C.R. Académie des Sciences Paris, *Life Sciences*, Vol. 320, pp. 989-997, 1997.

[7] R.M. French, "Pseudo-recurrent connectionist networks: An approach to the 'sensitivity-stability' dilemma," *Connection Scie*nce, Vol. 9, pp. 353-379, 1997.

[8] B. Ans and S. Rousset, "Neural networks with a self-refreshing memory: knowledge transfer in sequential learning tasks without catastrophic forgetting," *Connection Science*, Vol. 12, pp. 1-19, 2000.

[9] B. Ans, S. Rousset, R. M. French, and S. Musca, "Preventing catastrophic interference in multiple- sequence learning using coupled reverberating Elman networks," *Proceedings of the 24th Annual Conference of the Cognitive Science Society,* Lawrence Erlbaumn, Hillsdale NJ, pp. 71-76, 2002.

[10] B. Ans, S. Rousset, R. M. French, and S. Musca, "Self-refreshing memory in artificial neural networks: Learning temporal sequences without catastrophic forgetting," *Connection Science*, Vol. 16, pp. 71-99, 2004.

[11] C. Peterson and J. Anderson, "A mean field theory learning algorithm for neural networks," *Complex Systems*, Vol. 1, pp. 995-1019, 1987.

[12] G.E. Hinton, "Deterministic Boltzmann learning performs steepest descent in weight-space," *Neural Computation*, Vol. 1, pp. 143-150, 1989.

[13] X. Xie and H.S. Seung, "Equivalence of backpropagation and contrastive Hebbian learning in a layered network," *Neural Computation*, Vol. 15, pp. 441-454, 2003.

[14] D.C. Rubin, S.E. Wetzler and R.D. Nebes, "Autobiographical memory across the lifespan," *Autobiographical Memory*, edited by D.C. Rubin, Cambridge University Press, New York, pp. 202-221, 1986.

Bernard Ans is a CNRS researche director currently working at the Psychology and Neurocognition laboratory (CNRS – Pierre Mendes-France University, Grenoble, France, http://www.upmf-grenoble.fr/LPE/). He published works on neural network modeling of basic principles of learning and memory related to memory self-refreshing mechanisms avoiding catastrophic forgetting in distributed neural systems, cognitive transfer in sequential learning tasks, temporal sequence learning and sensorimotor programming, visual illusions, connectionist simulations of reading and acquired dyslexia. Early in his career he published a princeps paper on neural computation of independent component analysis and mathematical models of plant morphogenesis.