A Neural Framework for Robot Motor Learning based on Memory Consolidation

Heni Ben Amor[1], Shuhei Ikemoto[2], Takashi Minato[2], Bernhard Jung[1] and Hiroshi Ishiguro[2]

[1] VR and Multimedia Group, TU Bergakademie Freiberg, Freiberg, Germany
{amor,jung}@informatik.tu-freiberg.de
[2] Department of Adaptive Machine Systems, Osaka University, Osaka, Japan
{ikemoto,minato,ishiguro}@ed.ams.eng.osaka-u.ac.jp

Abstract. Neural networks are a popular technique for learning the adaptive control of non-linear plants. When applied to the complex control of android robots, however, they suffer from serious limitations such as the moving target problem, i.e. the interference between old and newly learned knowledge. However, in order to achieve lifelong learning, it is important that robots are able to acquire new motor skills without forgetting previously learned ones. To overcome these problems, we propose a new framework for motor learning, which is based on consolidation. The framework contains a new rehearsal algorithm for retaining previously acquired knowledge and a growing neural network. In experiments, the framework was successfully applied to an artifical benchmark problem and a real-world android robot.

1 Introduction

Neural networks have proved to be powerful and popular tools for learning motor control of robots with many degrees of freedom. However, some critics arouse about the use of sigmoidal neural networks (neural networks with sigmoidal kernel) for robot control. For example, Vijayakumar and colleagues [11] identified the following drawbacks of sigmoidal neural networks:

1. The need for careful network structure
2. The problem of catastrophic forgetting
3. The need for complex off-line retraining

The first point refers to the complex relationship between the network structure and it's ability to learn particular functions. It is well known in the machine learning community that particular functions (for instance XOR) cannot be learned if the network structure is not appropriately chosen. However, predefining the network structure also means limiting the amount of learnable data. The problem of catastrophic forgetting refers to the interference between previously and newly learned knowledge. When been sequentially trained on two problems A and B, it is often found that the network will have forgotten most of it's knowledge of problem A after training on problem B. However, if we really want to achieve lifelong learning robots [10], it is important that such robots are able to acquire new skills without forgetting previously learned ones. According to Vijayakumar et al.[11] complex off-line learning is needed to overcome the problem of catastrophic forgetting, which "from practical point, (this approach) is hardly useful". In this paper, we propose a new framework inspired by the architecture of the human brain and the process of *consolidation*. Main components of this framework are two neural networks, representing the short- and longtime memory respectively. In order to avoid that the longtime memory forgets any precious information, we employ cascade-correlation learning [1] and an improved rehearsal mechanism. Cascade-correlation adapts the network structure according to the problem difficulty, while rehearsal makes sure that old knowledge is retained.

2 Consolidation Learning

All problems of sigmoidal neural networks mentioned in the introduction can interestingly also be found in humans. In [6] it was demonstrated that two motor tasks may be learned and retained, only if the training sessions are seperated by an interval of approximately 6 hours. Otherwise, learning the second task leads to an unlearning of the internal model of the first task. However, if sufficient temporal distance is taken between the tasks, then the corresponding motor skills are retained for a long time (at least 5 months after trainig). The reason for this is a process called *consolidation*. During training, the acquired information is first stored in the prefrontal regions of the cortex. In this phase the information is still fragile and sucseptible to behavioral interference. Then, approx. 6 hours after trainig, consolidation shifts the information to the premotor, posterior and cerebellar cortex in which the information is stored for a long period of time without being disrupted. In this process the brain engages new regions to perform the task at hand.

2.1 A Computational Framework for Consolidation Learning

In this section we propose a new computational framework for motor learning based on consolidation. Figure 1 shows the components and the working of this new framework.

An important feature of the proposed consolidation framework is the separation between short-time memory (represented by the small neural network) and the long-time memory (represented by the large neural network) of the system. Each new task or subtask is first learned **online** by the small neural network. Learning is continued until the network becomes an expert in that particular task. Once this online phase is finished the expert network teaches newly acquired knowledge to the large neural network in the consolidation phase. This process is done **offline**. To ensure that the network has enough memory capacity

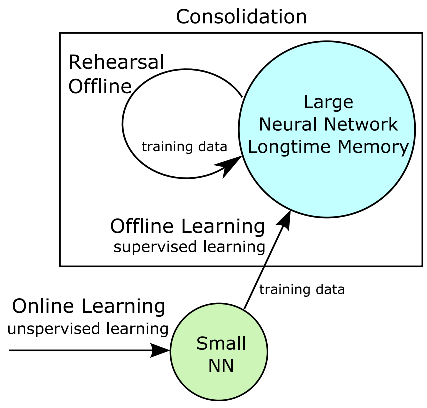

Fig. 1. Consolidation learning framework. A new problem is first learned by a small neural network. The information gained from this NN is then used to teach a large neural network in an offline process involving rehearsal of old information.

to learn the new knowledge, learning is performed using cascade-correlation. At the same time, old knowledge contained in the large neural network is rehearsed in order to avoid catastrophic forgetting.

Algorithm 1 Pseudo-code for consolidation learning $i = 0$

```
1: Learn new task $t_i$ unsupervised learning (online)
2: Create set $A$ with $n * \beta$ supervised data from small neural network
3: Create set $B$ with $n * (1 - \beta)$ supervised data from large neural network
4: while $e_n > desired\_error$ do
5:    Learn $A$ and rehearse $B$
6:    Compute network error $e_n$
7: end while
8: $i \leftarrow i + 1$
9: GOTO 1
```

Algorithm 1 illustrates the coarse flow of consolidation learning. Here, the set $T = \{t_1, ..., t_k\}$ refers to the k subtasks to be sequentially learned. The number β is a biasing parameter, which biases learning towards new or towards old information. If β equals zero, then the network will mainly try to remember old knowledge without learning new information. Empirically we found out that setting this value to 0.5 results in a good tradeoff. Finally, the number n represents the size of the training set to be used during consolidation.
A critical point in this algorithm is the reinstatement, i.e. the creation of supervised data used for rehearsal. One way to create the supervised data would be to store the old input vectors which were previously used for learning. By com-

puting the output of the large neural network for each of these input vectors, we get input-output pairs which can be used as a training set during rehearsal. The drawback of this approach is that it requires storing input data of all training phases so far. Another approach to reinstatement, called “pseudorehearsal”, is to use random input vectors when computing the rehearsal training set from the large neural network. This has the advantage of getting rid of additional data to be saved. On the other hand, it might detoriate the performance of consolidation. A random input might lie in an area of the input space, for which the long-time memory did not get any training data so far. In such a case, the rehearsed pattern might conflict with a newly learned pattern from the small neural network. When using growing techniques, such a conflict will result in more and more expert neurons being added to the network. The consequence is increasing error, learning time and network complexity. The following rehearsal algorithm aims at avoiding such situations.

2.2 Stimulation Based Pseudorehearsal

Previous rehearsal and pseudorehearsal techniques create a buffer which combines a number of new (from set A) and old patterns (from set B). Training the neural network with this buffer ensures that new information is learned and old information retained. In contrast to this, our new rehearsal algorithm called *stimulation based pseudorehearsal* explicitly distinguishes between a 'pure' rehearsal step and a step for learning new knowledge. These steps are executed alternately. Learning is executed until either the network error falls below the desired error or until convergence. In the latter case the problem is not successfully solved and thus new neurons will be trained and recruited to the network. The neurons are trained to maximize correlation between their output and the set A of new patterns. New neurons which are added to the network should act as experts for the newly acquired knowledge solely while old expert neurons represent previously acquired knowledge. These expert neurons are stimulated every second epoch, so they don't forget their information. After insertion of a new neuron into the network, rehearsal and learning are repeated. Algorithm 2 shows a pseudo-code implementation of this rehearsal technique.

In order to verify the strength of this pseudorehearsal technique, we conducted a set of experiments. The experiments were performed using both the original pseudorehearsal algorithm (for short: PR1) as described in [4] and stimulation based pseudorehearsal (SBPR). As benchmark we chose a problem which was already used in [5] and at the time could not be solved by a 3-layer sigmoidal feedforward neural network. The goal is to approximate the function $y = sin(2x) + 2exp(-16x^2) + N(0, 0.16)$ in two steps. In the first step a set of 130 data points which are uniformly distributed around in $x \in [-2.0, 0.5]$ are presented to the network. After learning these data points, the network is then presented with a new set of 70 new data points in $x \in [0.5, 2.0]$. It was shown that a neural network would forget the knowledge about the first part of the function after it has learned the second set of data points.

Algorithm 2 Pseudo-code for stimulation based rehearsal

```
 1: Learn $B$ for one epoch
 2: Learn $A$ for one epoch and get network error $e_n$
 3: if $e_n < desired\_error$ then
 4:     Learning finished
 5: else if not converged then
 6:     GOTO 1
 7: else
 8:     Create new neuron and maximize correlation with $A$
 9:     Add new neuron to network
10:     GOTO 1
11: end if
```

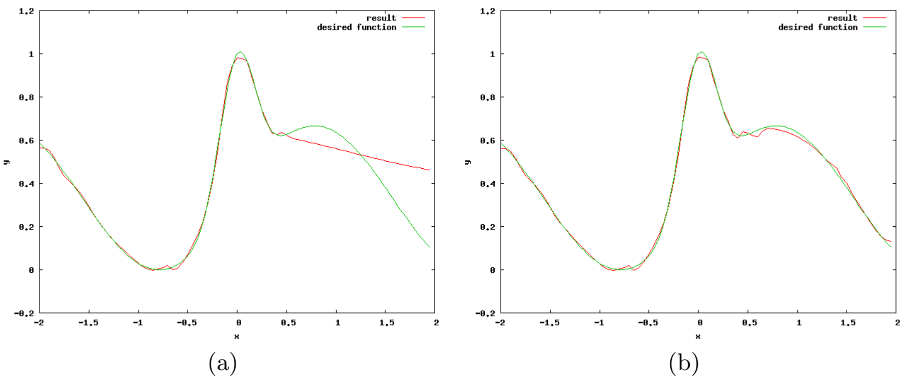

Fig. 2. Two phases of consolidation learning with stimulation based pseudorehearsal: the function $y = sin(2x) + 2exp(-16x^2) + N(0, 0.16)$ (green) is to be learned in two steps. In Figure (a) the first part of the function was approximated. In Figure (b) the second part was approximated, while information of the first part was rehearsed.

Figure 2 (a) shows the result of the training after learning the first set of data points. The neural network successfully learned to approximate the function in the range $x \in [-2.0, 0.5]$. In the next step the network was provided with points from $x \in [0.5, 2.0]$ and SBPR was used in order to make sure that new knowledge is learned without forgetting the old. Figure 2 (b) clearly shows, that the problem was successfully solved. Table 1 presents a comparison between result of PR1 and SBPR. From the table we can see, that the new variant leads to lower network error and network size. The mean squared error is about 0.0024 with SBPR, while it is approximately 6 times as high with PR1. The average number of neurons with SBPR is only 12.9 neurons compared to 26.4 neurons with PR1. In our experiments we also found out that on average the SBPR algorithm runs three times as fast as PR1.

Method	Mean Sq. Error	Std. Dev. Error	Average Num. Neurons	Std. Dev. Neurons
PR1	0.0147	0.0199	26.4	11.5
SBPR	**0.0024**	**0.0027**	**12.9**	**2.8**

Table 1. Performance of the original pseudorehearsal algorithm (PR1) and the stimulation based pseudorehearsal algorithm (SBPR) on an incremental approximation problem.

3 Experiments & Result

We evaluated consolidation learning on the pneumatic arm of an android robot called Repliee Q2. The android system adopts small pneumatic actuators and long, thin air tubes to realize a compact mechanism because the android system must have a very humanlike appearance. This results in strong non-linearity and provides a large response lag, including a large dead time. Controlling the motion of such a system can be difficult. One well-known method for learning desired motion is feedback error learning (FEL) [2]. However, it has been reported that such a large dead time makes feedback control stabilization more difficult [3]. This may disturb the feedback error learning applied to the motor learning of the android. Another learning technique which was shown to achieve state-of-the-art results in control tasks is Neuro Evolution on Augmenting Topologies (NEAT) [8]. NEAT uses a genetic algorithm to evolve appropriate neural networks. In our evaluation consolidation learning is applied to the control of a one-DoF air driven manipulator. Three different reference trajectories were sequentially used for learning. The robot arm had to learn to follow these reference trajectories by minimizing the error. After sequentially learning the three reference trajectories, the resulting neural network was tested for generalization and forgetting. For this, a validation set was created which included the three reference trajectories and two new (not learned) trajectories. The tests were performed using FEL, NEAT and consolidation learning. The dimension of the input vectors was 31. For FEL the number of hidden neurons was set to 20.

3.1 Results

In Figure 4 we see two out of five validation trajectories (red), and the tracking performed by consolidation learning (green) and by FEL (blue). Trajectory (a) was also used for learning, while trajectory (b) was only used for validation. It can be seen that the network learned with consolidation learning was able to follow both trajectories. The network was also able to solve both trajectories that have been seen and new trajectories. In contrast to this, the tracking achieved by FEL did not appropritately follow the reference trajectories; e.g it oscillated around trajectory (a), which lead to unnatural movements and friction between the robot parts. We also computed the mean squared error performed by the different learning algorithms in this setting. Consolidation learning performed best followed by NEAT and then FEL with a mean squared error of **0.044**, **0.055** and **0.056** respectively.

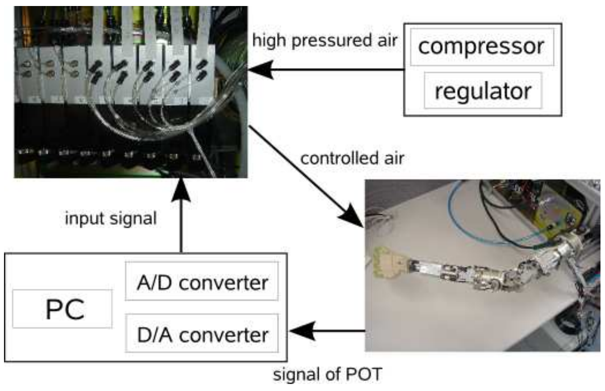

Fig. 3. The setting of the experiment for testing consolidation: An android arm is actuated by air pressure coming from a compressor.

4 Related Work

In [4] Robins discussed different methods for rehearsal and proposed the "pseudorehearsal" approach, which is also adopted in this paper. Tani [9] showed that a consolidation can help to reduce catastrophic forgetting in recurrent neural networks. In his approach, however, he uses a database to store each seen input pattern. Depending on the application, this can lead to high memory consumption. In [7] Poirier and Silver present a framework for Multiple Task Learning (MTL) based on ideas from consolidation. Although this approach shares some concepts with our work, it is mainly focused on the MTL domain. Because backpropagation learning is used, a great deal of effort has to be put into the design of network architecture. Additionally, this means that the amount of learnable knowledge is limited. In contrast to this, the neural networks in our paper scale up to the amount of knowledge to be learned, by inserting new neurons when needed.

5 Conclusion

In this paper we presented a new computational framework for robot motor learning based on a process called consolidation. Consolidation is inspired by neurobiological findings in humans and animals. The framework was evaluated on an android robot and found to be successfull in avoiding catastrophic forgetting and achieving high generalization. Furthermore, it removes the need for specifying the network structure prior to learning. For the future we aim at investigating how well this framework scales up to larger problems and how many times learning can be performed without detoriating knowledge which was learned at the beginning. Another interesting point for further investigation, is the comparison of the influence of different growing techniques on the performance of our framework. Finally, we aim at applying the framework to a more

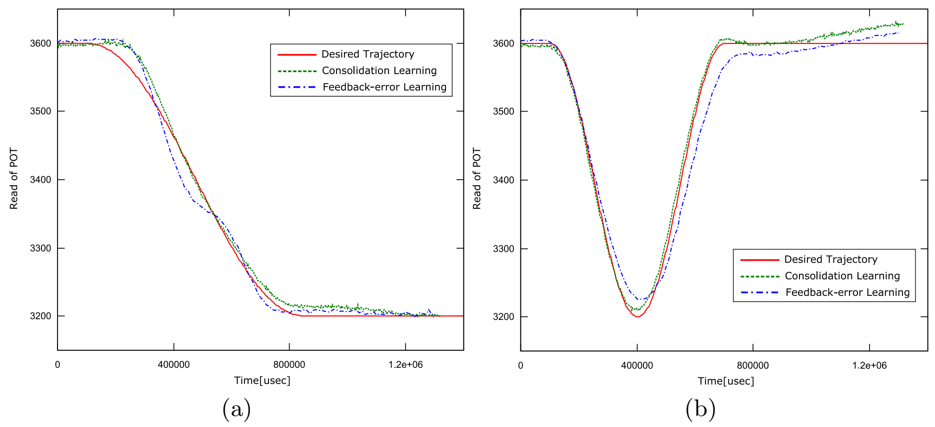

Fig. 4. Different desired and executed trajectories of the android arm after consolidation learning is finished. Time is given in microseconds. The trajectory (a) was also trained with, while trajectory (b) was only used in the validation phase.

sophisticated, new android robot and to simulated humans in virtual reality environments.

References

1. S. Fahlman and C. Lebiere. The cascade-correlation learning architecture. Technical Report CMU-CS-90-100, Carnegie Mellon, University, Pittsburgh, 1991.
2. M. Kawato. Feedback-error-learning neural network for supervised motor learning. *Advanced Neural Computers*, pages 365–472, 1990.
3. Y. Li and T. Asakura. Occurence of trajectory chaos and it's stabilizing control due to dead time of a pneumatic manipulator. *JSME International Journal Series C*, 48(4):640–648, July 2005.
4. A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connect. Sci.*, 7(2):123–146, 1995.
5. S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Comput.*, 10(9):2047–2084, 1998.
6. R. Shadmehr and H. Holcomb. Neural correlates of motor memory consolidation. *Science*, 277:821–825, 1997.
7. D. L. Silver and R. Poirier. Sequential consolidation of learned task knowledge. In *Canadian Conference on AI*, pages 217–232, 2004.
8. K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127, 2002.
9. J. Tani. An interpretation of the 'self' from the dynamical systems perspective: A constructivist approach. *Journal of Cosciousness Studies*, 5(5-6), 1998.
10. S. Thrun. *Lifelong Learning Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1998.
11. S. Vijayakumar and S. Schaal. Fast and efficient incremental learning for high-dimensional movement systems. In *Proc. IEEE Int'l Conf. Robotics and Automation*, pages 1894–1899. IEEE Press, Piscataway, N.J., 2000.