Hindawi Publishing Corporation
Journal of Robotics
Volume 2011, Article ID 617613, 12 pages
doi:10.1155/2011/617613

Research Article

A Cognitive Model for Generalization during Sequential Learning

Ashish Gupta,[1] Lovekesh Vig,[2] and David C. Noelle[3]

[1] *Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN 37235-1826, USA*
[2] *School of Comptational and Integrative Sciences, Jawaharlal Nehru University, New Delhi 110067, India*
[3] *School of Engineering, University of California, Merced, Merced, CA 95343, USA*

Correspondence should be addressed to Lovekesh Vig, lovekeshvigin@gmail.com

Received 31 May 2011; Revised 2 September 2011; Accepted 20 September 2011

Academic Editor: Ivo Bukovsky

Copyright © 2011 Ashish Gupta et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Traditional artificial neural network models of learning suffer from *catastrophic interference*. They are commonly trained to perform only one specific task, and, when trained on a new task, they forget the original task completely. It has been shown that the foundational neurocomputational principles embodied by the Leabra cognitive modeling framework, specifically fast lateral inhibition and a local synaptic plasticity model that incorporates both correlational and error-based components, are sufficient to largely overcome this limitation during the sequential learning of multiple motor skills. Evidence has also provided that Leabra is able to generalize the subsequences of motor skills, when doing so is appropriate. In this paper, we provide a detailed analysis of the extent of generalization possible with Leabra during sequential learning of multiple tasks. For comparison, we measure the generalization exhibited by the backpropagation of error learning algorithm. Furthermore, we demonstrate the applicability of sequential learning to a pair of movement tasks using a simulated robotic arm.

1. Introduction

Humans acquire many different skills, behaviors, and memories throughout their lifetime. Lack of use of a particular skill, behavior, or memory results in its slow degradation. It is a common observation that reacquisition of any knowledge is typically rapid as compared to its initial acquisition. This retention of knowledge, often in a latent form, is known as *savings*.

Why do we observe a degradation in performance when a particular piece of knowledge fails to be used? The initial acquisition of task knowledge is driven by synaptic plasticity, shaped by experience. This plasticity continues even when that task knowledge is not regularly used. Thus, experience with other activities could shape the neural circuits in a manner that interferes with the initial knowledge that was acquired.

What could be the neural basis of savings? Savings could emerge due to neural specialization. Some of the neurons employed by an initial task might not be reused when learning a subsequent task. To the degree that the sets of neurons associated with different tasks are disjoint, learning one task will not affect the synapses associated with another. Note, however, that when neurons are shared between tasks savings are still possible. Savings could arise from subthreshold residual synaptic weights associated with the initial task—weights that have been driven down by interfering experiences to below their threshold for neural firing, but not all the way down to their initial values. Finally, tasks may share components or "subtasks." To the degree that such components have isolated neural representations, learning a new task may actually reinforce portions of a previously learned task.

Traditional artificial neural network models of human learning, including those based on the powerful *backpropagation of error* learning algorithm, fail to display adequately robust savings when tasks are learned sequentially. A set of synaptic connection weights serve as the neural network's memory, and any task is learned by modifying these weights. This is the strength of neural networks since they can learn almost any possible input-output mapping. However, this is also the source of problems, as the networks have an inherent tendency to abruptly and completely forget previously learned knowledge when presented with new training inputs. This phenomenon is known as catastrophic interference [1]. This prevents artificial neural networks

from exhibiting savings and, therefore, leads to questions about their biological plausibility.

In our previous work, we proposed that the biological constraints imposed by the structure of cortical circuitry may embody the properties that are necessary to promote savings, as observed during human skill acquisition [2]. Specifically, we examined the neurocomputational principles forming the Leabra cognitive modeling framework [3], and we found that these biologically motivated principles give rise to savings without the need for any auxiliary mechanisms. Our findings suggest that the Leabra implementation of fast acting lateral inhibition acts in concert with its synaptic plasticity mechanism in order to produce adequately sparse representations to support skill savings.

Sparse representations involve patterns of neural firing in which only a small fraction of neurons in a "pool" or "layer" are strongly active at any one time. The use of sparse representations results in different sets of neurons being used for the different tasks that the network has been trained to perform. There is good reason to believe that, when learning multiple tasks, humans are able to generalize the common structure, if any, that exists between them. In our previous work, we provided preliminary evidence that Leabra networks, even with a sparse internal representation enforced by a biologically plausible lateral inhibition mechanism, are able to generalize the common subtasks shared by multiple tasks, when doing so leads to appropriate responses. In this paper, we provide a more detailed analysis of the generalization seen in Leabra networks. We show that the generalization seen in a Leabra network when learning two tasks sequentially is comparable to the generalization seen when the two tasks are learned in an interleaved manner. In comparison, a backpropagation-based artificial neural network not only does not show any generalization, but it also does not show any savings either.

Most neural-network-based controllers require substantial training on a particular task and need to be retrained if the network subsequently learns a different task [4, 5]. A network model that exhibits savings offers potential benefits for such applications. The retraining time required for a previously learnt task is considerably reduced if the network exhibits savings. Such a network would be capable of learning multiple tasks sequentially without the need to interleave these tasks.

A wide variety of sequential key pressing tasks have been used to investigate human motor skill learning [6–8], and a number of interesting findings have resulted. There is a period of rapid improvement in performance during the early stages of training. During this stage, learning is effector independent (e.g., switching hands does not substantially degrade performance). Further, interfering with the frontal systems involved in the controlled pathway during this period seriously disrupts performance. Interfering with the automatic pathway, however, does not affect performance during this early period. In this paper, we demonstrate the applicability of our Leabra network model to learning of movement tasks in a simulated robotic manipulator. The network is able to exhibit savings whilst learning arm movement tasks sequentially, thereby substantially reducing the retraining time required.

This paper is organized as follows. In the next section, we provide some background and an overview of related work. Section 3 provides a description of a Leabra-based task learning model and some simulations of the model. Section 4 describes the results of these simulation experiments. We end the paper with a discussion of the results, as well as some conclusions.

2. Background

2.1. Leabra. The Leabra framework offers a collection of integrated cognitive modeling formalisms that are grounded in known properties of cortical circuits while being sufficiently abstract to support the simulation of behaviors arising from large neural systems. It includes dendritic integration using a point-neuron approximation, a firing rate model of neural coding, bidirectional excitation between cortical regions, fast feedforward and feedback inhibition, and a mechanism for synaptic plasticity [3] (refer to the appendices for a more detailed description). Leabra models have successfully illuminated cognitive function in a wide variety of domains, including perception, object recognition, attention, semantic memory, episodic memory, working memory, skill learning, reinforcement learning, implicit learning, cognitive control, and various aspects of language learning and use. Of particular relevance to skill savings are Leabra lateral inhibition formalism and its synaptic learning rule.

In the neocortex, two general patterns of connectivity have been observed involving inhibitory neurons and their interactions with excitatory neurons, namely, *feedforward* and *feedback* inhibition [3]. Feedforward inhibition occurs when the inhibitory interneurons in a cortical region are driven directly by the inputs to that region, producing rapid inhibition of the excitatory neurons in that area. Feedback inhibition occurs when the same neurons that excite nearby inhibitory interneurons are, in turn, inhibited by the cells they excite, producing a kind of negative feedback loop.

The effects of inhibitory interneurons tend to be strong and fast in the cortex. This allows inhibition to act in a regulatory role, mediating the positive feedback of bidirectional excitatory connections between brain regions. Simulation studies have shown that a combination of fast feedforward and feedback inhibition can produce a kind of "set-point dynamics," where the mean firing rate of cells in a given region remains relatively constant in the face of moderate changes to the mean strength of inputs. As inputs become stronger, they drive inhibitory interneurons as well as excitatory pyramidal cells, producing a dynamic balance between excitation and inhibition. Leabra implements this dynamic using a *k-Winners-Take-All (kWTA)* inhibition function that quickly modulates the amount of pooled inhibition presented to a layer of simulated cortical neural units, based on the layer level of input activity. This results in a roughly constant number of units surpassing their firing threshold. The amount of lateral inhibition within a layer

can be parameterized in a number of ways, with the most common being the percentage of the units in the layer that are expected, on average, to surpass threshold. A layer of neural units with a small value of this k parameter (e.g., 10–25%) will produce sparse representations, with only a small fraction of the units being active at once.

With regard to learning, Leabra modifies the strength of synaptic connections in two primary ways. An error-correction learning algorithm changes synaptic weights so as to improve network task performance. Unlike the backpropagation of error algorithm, Leabra error-correction scheme does not require the biologically implausible communication of error information backward across synapses. In addition to this error-correction mechanism, Leabra also incorporates a Hebbian correlational learning rule. This means that synaptic weights will continue to change even when task performance is essentially perfect. This form of correlational learning allows Leabra to capture certain effects of overlearning.

2.2. Catastrophic Interference. Many past studies have shown that artificial neural networks suffer from a kind of catastrophic interference that is uncharacteristic of human performance. The seminal example of catastrophic interference is the experiment performed by McCloskey and Cohen [1], in which the authors tried to employ a standard backpropagation network to perform the AB-AC list-learning task. In this task, the network begins by learning a set of paired associates (A-B) consisting of a nonword and a real word (e.g., "pruth-heavy"). Once, this learning was completed, they trained the network to associate a new real word with each of the original nonwords (A-C). The authors found that as soon as the training on the AC list started, the network completely forgot the AB list.

Since the original observation of catastrophic interference in artificial neural networks, a number of computational mechanisms have been proposed to overcome it. Most of these involve segregating the neural units associated with different skills in order to avoid the damage caused by "reuse" of synaptic weights [9]. For example, forcing layers of neural units to form sparse representations reduces the probability that a given unit will be active while performing a given skill, thereby reducing the probability of interference when learning multiple skills in sequence. Leabra offers a biologically justified mechanism for producing sparse representations. With a low k parameter, Leabra kWTA lateral inhibition implementation limits the overlap between the neural representations used for different tasks. This has been shown to improve performance on the AB-AC list learning task [3].

One extreme form of segregation between neurons devoted to different tasks involves isolating them into discrete modules. Modular artificial neural network architectures have been proposed in which differences between tasks are explicitly detected during learning, and a "fresh" module of neural units is engaged to learn the task, protecting previously trained modules from interference [10, 11]. Importantly, overlearning of a task can strengthen its consolidation in a module, increasing resistance to interference, as is observed in humans [12, 13]. While such modular models can exhibit robust savings (and appropriately limited forms of interference), the biological plausibility of a reserve of untrained neural modules awaiting assignment when a new task is to be learned is questionable. Even if we assume the existence of unused modules, questions still remain about their granularity—do we need a different module even for different variants of the same task? It also poses a question concerning the total number of available modules. Some modular networks do show limited forms of generalization through combining the outputs from multiple modules [11], but they still need to use a fresh module for most cases [14].

Modular approaches of this kind should be distinguished from the hypothesis that the hippocampus and the neocortex form distinct learning systems [15]. This hypothesis asserts that catastrophic interference is alleviated through the use of a fast hippocampal learning system that uses sparse representations. While neocortical systems are assumed to use less sparse representations, making them more vulnerable to interference, problems are avoided through a hippocampally mediated process of consolidation, where neocortical networks receive interleaved "virtual" practice in multiple skills. Through the mediation of the hippocampus, multiple skills continue to be essentially learned "together," rather than sequentially, one after the other.

One successful computational learning strategy that is similar in nature to hippocampally mediated consolidation involves the use of "pseudopatterns" [16]. Trained artificial neural network models are used to generate virtual training experiences—pseudopatterns—which are similar to previously learned patterns. These pseudopatterns are mixed with the new patterns, corresponding to new learning experiences, and the network is given interleaved training on all of these patterns. It has been observed that the use of pseudopatterns alleviates the problem of catastrophic interference substantially [17, 18]. However, the biological mechanisms giving rise to the generation and use of these patterns have yet to be fully explicated. It is also difficult to imagine the generation of pseudopatterns for the maintenance of all kinds of knowledge. This is particularly true if hippocampus is believed to be the sole site for the maintenance of mixed training pattern sets. All kinds of knowledge and skill acquisition do not depend upon the hippocampus. For example, there is evidence that humans can continue to learn new motor skills even after complete removal of the hippocampus [19].

There are many other factors that contribute to savings [2, 20]. Contextual cues help in disambiguating between different tasks and could also lead to the use of different sets of neurons for their representation. Overlearning a particular task results in a sharpening of its representation, which is more resistant to perturbation. Finally, it is possible that synaptic changes during the learning of an interfering task may drive certain neurons associated with a previously learned task below their firing threshold—but just below that threshold, allowing them to recover quickly once practice of the previous task is resumed.

We have shown, in previous work, that the sparse representations enforced by Leabra lateral inhibition mechanism,

in conjunction with its synaptic learning rule, causes Leabra simulations of cortical circuits to escape the most dire pitfalls of catastrophic interference when those circuits are required to sequentially learn multiple temporally extended motor trajectories [2].

2.3. Generalization. The use of sparse representation results in the use of different sets of neurons for different tasks. This implies that the network learns the tasks in a highly specialized manner. Hence, the network may not be able to generalize the common substructure existing between different tasks. There is good reason to believe that humans are able to make such generalizations, even if the tasks are learned sequentially. Also, artificial neural network models of human learning show generalization when multiple tasks are learned in an interleaved manner [21]. Such generalization also emerges from networks that generate pseudopatterns of previously learned tasks [17, 18].

Is it possible for a network with a sparse representation scheme to still exhibit generalization when tasks are learned sequentially? We know that the use of a small value for the k parameter for internal hidden layers creates the possibility of using different units for different tasks. A contextual cue layer increases this probability by providing inputs to distinct hidden layer neurons for the different tasks. Contextual cues are biologically justified, and they improve savings significantly. However, if the cue signal is too strong, it enforces a separate representation even for the common subtask, thus hindering generalization. We found that the strength of the cues can be set to an optimum value, such that the network continues to show significant savings by enforcing orthogonal representation [22], while still allowing the use of the same neurons for the common subtask, thus enabling generalization as well.

2.4. Neural Network Controllers. The neural-network-based control technique has been widely used to solve problems commonly encountered in control engineering. The most useful property of neural networks in control is their ability to approximate arbitrary linear or nonlinear mappings through learning. It is because of the above property that many neural-network-based controllers have been developed for the compensation of the effects of nonlinearities and system uncertainties in control systems so that the system performance such as the stability and robustness can be improved.

There has been considerable research towards the development of intelligent or self-learning neural-based controller architectures in robotics.Riedmiller and Janusz [23] proposed a neural self-learning controller architecture based on the method of asynchronous dynamic programming [24]. The capabilities of the controller are shown on two typical sequential decision problems. Johnson et al. [25] were the first to utilize a backpropagation network to train a robotic arm. The backprop was able to develop the inverse kinematics relationship for the arm after being trained on the arm data. Hesselroth et al. [26] employ a neural map algorithm to control a five-joint pneumatic robot arm and gripper through feedback from two video cameras. The pneumatically driven robot arm (SoftArm) employed in this investigation shares essential mechanical characteristics with skeletal muscle systems. To control the position of the arm, 200 neurons formed a network representing the three-dimensional workspace embedded in a four-dimensional system of coordinates from the two cameras and learned a three-dimensional set of pressures corresponding to the end-effector positions.

Bouganis and Shanahan [27] present a spiking neural network architecture that autonomously learns to control a 4-degrees-of-freedom robotic arm after an initial period of motor babbling. Its aim is to provide the joint commands that will move the end-effector in a desired spatial direction, given the joint configuration of the arm. Vaezi and Nekouie [28] utilize a new neural networks and time series prediction-based method to control the complex nonlinear multivariable robotic arm motion system in 3D environment without engaging the complicated and voluminous dynamic equations of robotic arms in the controller design stage.

2.5. Sequential Learning. The implications and advantages of sequential training of partially unconnected tasks on a given network architecture remain mostly unexplored, unclear or have even been seen negatively in form, for example, catastrophic interference [20, 29]. In robotics and machine learning, though, a number of attempts have been made in analysing enhanced learning through a sequence of training environments [30–35]. Saal et al. [36] consider the problem of tactile discrimination, with the goal of estimating an underlying state parameter in a sequential setting. They present a framework that uses active learning to help with the sequential gathering of data samples, using information theoretic criteria to find optimal actions at each time step. Kruger explores the effects of sequential training in the form of shaping in the cognitive domain. He considers abstract, yet neurally inspired, learning models and proposes extensions and requirements to ensure that shaping is beneficial using a long term memory model. While this model can learn sequentially and mitigate catastrophic interference, it is reliant on an explicit memory model whereas our model avoids catastrophic interference by simulating the biological processes in real neurons such as lateral inhibition. Kruger [37] explores the effects of sequential training in the form of shaping in the cognitive domain. He considers abstract, yet neurally inspired, learning models and proposes extensions and requirements to confirm that shaping is beneficial for sequential learning. However, to the best of our knowledge no system has yet been designed to explicitly utilize the phenomenon of savings in sequential environments by utilizing generalization of neural models in a cognitive framework.

2.6. RoboSim Robotic Arm Simulator. The RoboSim simulated robotic manipulator [38] was used to demonstrate the applicability of our approach to learning of movement sequences. RoboSim is a simulation system for a 6-degrees-of-freedom robot manipulator. For our experiments we used only three joints and kept the other 3 fixed to result in

Table 1: Similarities between various tasks as compared to Base Task.

Task	Similarity
Nothing Same Task	Nothing in common with Base Task
5 Patterns Same Task	5 of the 18 patterns same as in Base Task
10 Patterns Same Task	10 of the 18 patterns same as in Base Task

a manipulator with 3 degrees of freedom. The manipulator may be moved either in joint coordinates or in cartesian coordinates. RoboSim allows joint weights to be altered, in order to favor movement of individual joints versus others. The simulation system includes both forward and inverse manipulator kinematics. The system also allows the user to teach the manipulator a sequence of consecutive movements.

3. Method

3.1. Simulated Tasks with the Leabra Network. Four different tasks were used in our simulations. Each task contained 18 different input-output pairs, with the input and output vectors generated randomly. While the output vectors for each task differed, the input vectors were kept identical for all tasks. This resulted in different input-output mappings being learnt by the network for different tasks.

After training on the Base Task, the network was trained on one of the interfering tasks. Table 1 briefly describes the similarity between the Base Task and the interfering tasks. After training on the interfering task, retained performance on the Base Task was assessed. Each of the input and output patterns was encoded in the Leabra network over a pool of 36 neural units.

3.2. The Network. Figure 1 shows the Leabra network used in our simulations. At each step, the network was provided with a 36-unit input vector, encoding one of the 18 patterns comprising a task. Complete interconnections from this input layer to a 100-unit hidden layer produced an internal representation for the current pattern, with the sparsity of this representation controlled by lateral inhibition within the hidden layer (i.e., by its k parameter). Complete bidirectional excitatory connections map this internal representation to an output layer that is intended to encode the output pattern. During training, the output layer was also provided with a target signal, indicating the correct output. The context layer contained two units, each corresponding to one of the two learned tasks, indicating which of the two tasks was to be produced by the network. This layer was connected to the hidden layer with an 80% chance of interconnection between a given context layer unit and any given hidden layer unit. This random interconnection structure was expected to increase the chances of orthogonal representations for different tasks.

Most of the network parameters used in the simulations were Leabra default values. Connection weights between units were randomly initialized with a mean of 0.5 and a variance of 0.25. The minimum weight value for any interconnection was 0 and the maximum was 1. The activation of any node could range between 0 and 1. Leabra uses a mixture of the GeneRec learning algorithm and a form of Hebbian learning. GeneRec is an error-driven learning algorithm [3]. Hebbian learning was strengthened in our simulations, as compared to the Leabra default, contributing to 1% of synaptic weight changes rather than the default 0.1%. The learning rate for training was set to 0.01. Error tolerance was set to 0.25. This means that any output unit could be within ± 0.25 of its desired activity without prompting error-correction learning. In order to facilitate a sparse internal representation, a value of $k = 10$ was used for the hidden layer. For comparison, a backpropagation of error network (BP), matching the Leabra network in structure and size, was also examined.

There are two common measures of savings: *exact recognition* and *relearning* [9]. The exact recognition measure assesses the percentage of the original task that the network performs correctly after it has learned a second task. The relearning measure examines how long it takes the network to relearn the original task. The two measures are usually correlated. We used an exact recognition measure to assess savings. In particular, we measured the sum-squared error (SSE) of the network output on the first task after training on the second task. In order to contrast this SSE value with "complete forgetting" of the first task, the SSE was also recorded prior to the first task training, and we report the ratio of SSE after interference training to SSE of the untrained network. A value of one or more for this ratio indicates complete forgetting of the initial task, while lower values indicate savings.

To measure the similarity of the hidden layer representation for the common subtask between the two tasks, two different measures were used. First, we computed the root-mean-square (RMS) value of the difference between the hidden layer activities for the common subtask using the following formula:

$$\mathrm{RMS} = \sqrt{\frac{\sum_{i=1}^{N} \left(A_i^1 - A_i^2\right)^2}{N}}. \tag{1}$$

Here, A_i^1 is the activity of the ith hidden layer unit for the common subtask after the network has been trained on the Base Task. Similarly, A_i^2 is the activity of the ith hidden layer unit for the common subtask after the network has been trained on the interfering task. N is the total number of units in the layer. The minimum RMS value is 0. This would occur if the hidden layer activity is exactly the same for the two tasks. Since the activity of the units is restricted to the range between 0 and 1, the maximum RMS value is 1. This would occur if the hidden layer activity is completely orthogonal and every unit is firing at its maximum possible activation value for one of the tasks and is not firing at all for the other task.

Second, we measured the percentage of total active units (at least 0.05 activation) that were common for the common subtask at the two relevant points in training. This percentage is a good measure of similarity in representation for the

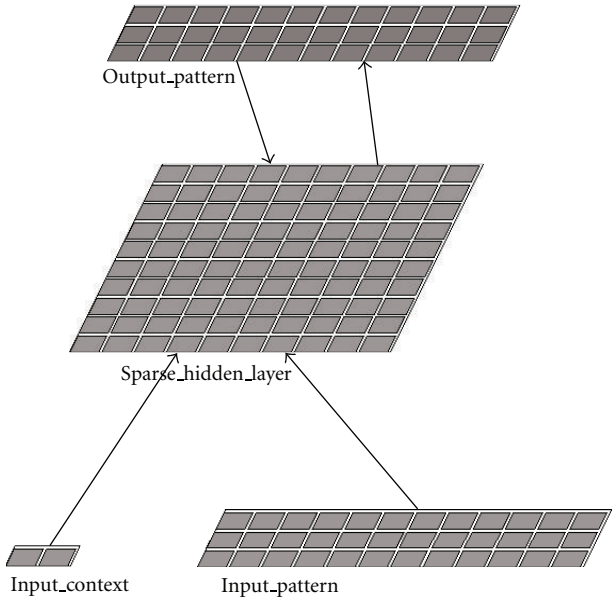


Figure 1: The Leabra network.

Leabra network, since roughly only 10 units are active at any time (since $k = 10$ for the hidden layer). However, this measure is not appropriate for the BP network, since BP uses almost all the hidden units to represent a task.

We repeated each experiment five times in order to deal with stochastic variations (arising from such factors as random initial weights and the random order of pattern presentation) in our simulations. We report the average of these repetitions.

4. Results

4.1. Generalization. A randomly initialized network was trained on the Base Task. Then, the network was trained on an interfering task. Using the "Nothing Same Task" as the second task; the variation of the SSE ratio as a function of the hidden layer kWTA parameter is shown in Figure 2.

We found a regular decrease in the error ratio with decrease in k. The reason for this decrease is the decrease in overlap between the Base Task and Nothing Same Task hidden layer activation patterns as representations became sparser. We measured the percentage of total active units common for the two tasks. This percentage was large for dense representation and decreased considerably for sparse representations. Thus, increasing lateral inhibition produced more distinct internal representations between the tasks and resulted in improved savings. The network showed complete forgetting (SSE ratio greater than 1) of the base task for k greater than 40. This suggests that the savings exhibited by the Leabra network are due to the kWTA mechanism and not due to its recurrent architecture. It was found that the BP network exhibited no savings at all. This was as expected since there is no explicit mechanism to facilitate nonoverlapping hidden layer representation in the BP network.

Table 2: Mean and standard error of SSE ratios for Base Task, when different interfering tasks were used. The first row gives the ratio for a BP network, the second row for a Leabra network with contextual cues of strength 1.0, the third row for a Leabra network with contextual cues of strength 0.35, and the fourth row for a Leabra network trained in an interleaved manner for the two tasks. A smaller ratio signifies more savings.

Network	5 Patterns Same	10 Patterns Same
BP	1.269 (± 0.0242)	0.841 (± 0.0425)
Context = 1.0	0.196 (± 0.0606)	0.144 (± 0.0457)
Context = 0.35	0.147 (± 0.0350)	0.108 (± 0.0168)
Interleaved	0.000 (± 0.0000)	0.000 (± 0.0000)

Next, we fixed $k = 10$ for the hidden layer and performed similar experiments with all of the other interfering tasks. Table 2 shows the SSE ratio for the various cases. It was found that the SSE ratio was close to 1 for the BP network. This suggests that the BP network consistently experienced catastrophic interference. On the other extreme was the Leabra network that was given interleaved training for the two tasks. In this case, the network learned both of the tasks completely. The Leabra network with contextual cues (of activation magnitude 1.0 and 0.35) that was given sequential training on the two tasks displayed significant savings, represented by a very small SSE ratio.

To test if the networks were able to come up with a generalized representation for the common subtask, we compared the hidden layer activity for the common subtask between the learning of the two tasks. Table 3 shows the percentage of active units that were common across the two tasks. As explained before, this percentage has been computed only for the Leabra networks.

We found that the Leabra network with contextual cues of strength 0.35 shows generalization comparable to the Leabra network that is trained on the two tasks in an interleaved manner. On the other hand, the Leabra network with contextual cues of strength 1.0 shows very little generalization. For comparison, we also measured the percentage of common active units when the "Nothing Same Task" was used as the interfering task and the strength of contextual cues was 1.0. We found that this percentage was zero, signifying completely orthogonal internal representations.

To compare the results of generalization seen in the Leabra networks with that seen in the BP network, we computed the root-mean-square (RMS) difference in the hidden layer activity for the common subtask between the two tasks. Table 4 shows the RMS values for the different cases. Once again, it was observed that the RMS difference is comparable for the Leabra network with contextual cues of strength 0.35 and the Leabra network that was given interleaved training. Also, the RMS value for the Leabra network with contextual cues of strength 1.0 is comparable to that of a BP network. For comparison, we also measured the RMS value when the "Nothing Same Task" was used as the interfering task. The RMS value for the BP network was 0.3279 ± 0.0098, and for the Leabra network it was 0.3517 ± 0.0027.

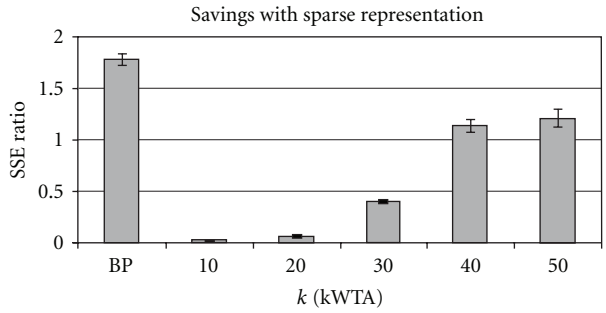


Figure 2: Savings as a function of sparsity. An SSE ratio of one or more indicates no savings, while lower values indicate retention of Base Task knowledge. Nothing Same Task was used as the interfering task. The k parameter roughly equals the percentage of active units in the hidden layer. Error bars display standard errors of the mean.

Table 3: Mean and standard error of percentage of common units for the common subtask between Base Task and the interfering task. The first row gives the ratio for a Leabra network with contextual cues of strength 1.0, the second row for a Leabra network with contextual cues of strength 0.35, and the third row for a Leabra network trained in an interleaved manner for the two tasks.

Network	5 Patterns Same	10 Patterns Same
Context = 1.0	12.748 (±1.695) %	12.724 (±1.975) %
Context = 0.35	71.246 (±2.036) %	66.702 (±2.750) %
Interleaved	68.882 (±1.785) %	67.340 (±2.300) %

Table 4: Mean and standard error of RMS difference in the hidden layer activation for the common subtask between Base Task and the interfering task. The first row gives the RMS value for a BP network, the second row for a Leabra network with contextual cues of strength 1.0, the third row for a Leabra network with contextual cues of strength 0.35, and the fourth row for a Leabra network trained in an interleaved manner for the two tasks.

Network	5 Patterns Same	10 Patterns Same
BP	0.324 (±0.0080)	0.381 (±0.0067)
Context = 1.0	0.299 (±0.0044)	0.329 (±0.0060)
Context = 0.35	0.159 (±0.0060)	0.147 (±0.0052)
Interleaved	0.156 (±0.0053)	0.156 (±0.0083)

4.2. Savings. To uncover the degree to which our model exhibits savings, we conducted simulations to record how the network changes during training and extinction. Animals are faster to reacquire an extinguished behavior, as compared to initial acquisition, and they are faster to extinguish a reacquired behavior, as compared to initial extinction [39–44]. A randomly initialized network was trained to respond upon the presentation of a pattern. Once this training reached criterion, the network was trained to not respond upon the presentation of the pattern. This process was repeated 5 times. Figure 3 shows the number of trials required for successive acquisition and extinction trainings. Note that the required time quickly decreases. The model predicts that the required number of trials will asymptote to a small value after just a few acquisition-extinction iterations.

The network starts with small initial synaptic weights. Hence, a large change in weights is required for success during the first acquisition training session. During the first extinction training session, the weights to the acquisition neurons start decreasing and the weights to the extinction neurons start increasing. As soon as the extinction neurons win the inhibitory competition, the acquisition neurons tend to fall below their firing threshold. At this stage, the weights to the acquisition neurons stop decreasing, as these neurons are no longer contributing to erroneous outputs. Hence, a significant amount of acquisition-related association strength is retained through the extinction process. During the reacquisition training, the weights to the acquisition neurons increase once again and the weights to the extinction neurons decrease. Once again, the weights stop changing as soon as the extinction neurons lose the inhibitory competition. Hence, most of extinction-related plasticity is retained through the acquisition process. In this manner, subsequent acquisition and extinction trainings require a very small change in weights (Figure 4). Effectively, acquisition and extinction associations are maintained side by side in the network, allowing for the rapid switching between them based on recent training feedback.

4.2.1. Savings in Robotic Arm Movements. We have used our model to simulate the learning of arm movement sequences. Our model controls a simulated 3-joint planar arm which moves over a 3-dimensional space, as shown in Figure 5. The state of the arm at any point in time is represented by the vector (q1, q2, q3), where q1, q2, and q3 are the three joint angles. The joint angles range between -180° and 180°. Two unrelated trajectories are taught to the manipulator with each trajectory represented as a sequence of arm states over the successive time steps. During training, the arm is essentially guided along the desired trajectory, with differences between the motor output of the arm controller and the configuration of the arm, as specified by the guide, acting as a measure of error to drive synaptic weight change.

Figure 6 shows the Leabra network used for our simulations. The Sensory_Input layer provides the current state of the arm as input to the network and the Motor_Output layer is to produce the desired arm state for the next time step. Each joint angle is encoded over a pool of 36 neural units. Each of the 36 units has a preferred angle, ranging from -180° to 180°, in 10° increments. To encode a given joint angle, the closest unit with regard to preference, as well as its two neighbors, is set to its maximal firing rates. Similarly, patterns of activity over each row of 36 units in the Motor_Output are decoded by identifying the preferred angle of the unit in the middle of the three adjacent units that are all active. Other patterns of activity in the Motor_Output layer are considered to be ill-formed. With each joint angle encoded over 36 units in this way, the complete arm configuration is encoded over 108 units. The context layer was used to encode contextual information related to the tasks.

The tasks consist of 20 consecutive arm positions that the manipulator was trained to achieve in order. After the manipulator had learnt the movements associated with

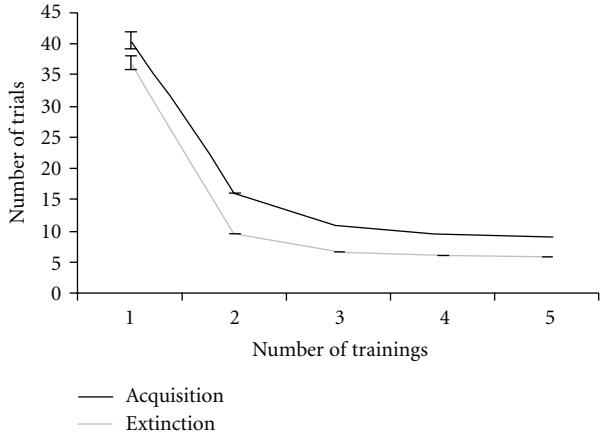


Figure 3: The number of training trials required to reach criterion (y-axis) decreases as the number of prior acquisition and extinction training sessions (x-axis) increases. Error bars report standard errors of the mean.

Table 5: SSE ratio and retraining epochs.

Task	SSE ratio	Training epochs	Retraining epochs
Task 1	0.522 (±0.0033)	10.8 (±0.24)	4.2 (±0.20)
Task 2	0.467 (±0.0023)	12.2 (±0.24)	4.2 (±0.20)
Task 3	0.413 (±0.0097)	15.4 (±0.24)	2.4 (±0.48)
Task 4	0.328 (±0.0049)	13.8 (±0.24)	2.4 (±0.48)
Task 5	—	13.6 (±0.24)	—

the first task, the network was subsequently trained on the second task. The tasks were generated to ensure that there were no common patterns between the tasks. The manipulator was then tested and retrained on the first task. The manipulator was able to accurately remember the training for 15 out of 20 movements for the first tasks with an SSE ratio of 0.234. More importantly, the training time for relearning the first time was just 2 epochs as opposed to 15 epochs for the untrained network.

To further investigate the applications of savings, the manipulator network was sequentially trained on five manipulator tasks (task 1 to task 5). The SSE ratio for all the tasks was shown in Table 5. As expected, the SSE ratio is the lowest for the most recently trained tasks, indicating greater savings for these tasks. More importantly, the retraining time for all tasks was significantly less than the original training time. Figure 7 shows the variation in savings for successively trained tasks with different values of hidden layer kWTA. As the kWTA parameter is increased, the network is able to retain less information about previously trained tasks.

5. Conclusion and Discussion

We have shown that the neurocomputational principles embodied by the Leabra modeling framework are sufficient to produce generalization, whilst exhibiting significant savings during the sequential learning of multiple tasks. No auxiliary computational mechanisms are needed. Generalization

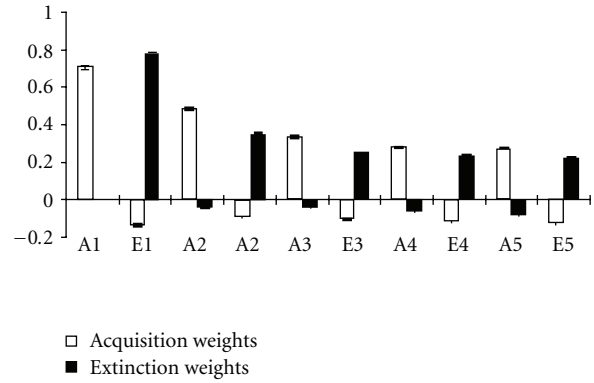


Figure 4: This graph plots the change in the summed connection weights in the acquisition pathway and in the extinction pathway (y-axis) during the successive acquisition and extinction trainings (x-axis). The change in weights decreases in both the pathways as the number of prior acquisitions and extinctions training sessions increases. There seems to be a slow upward going trend in the weights in both the pathways, which appears to be a quirk of the simulator.

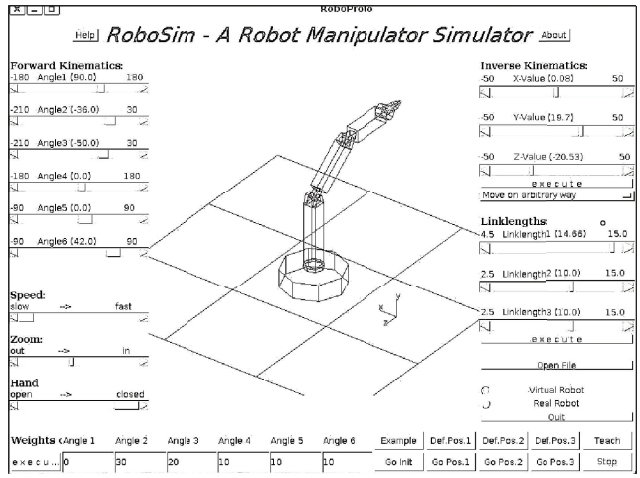


Figure 5: The RoboSim Robotic Arm Simulator. The simulator allows for both forward and inverse kinematics, learning of a sequence of movements, varying degrees of freedom and link lengths, and speed control.

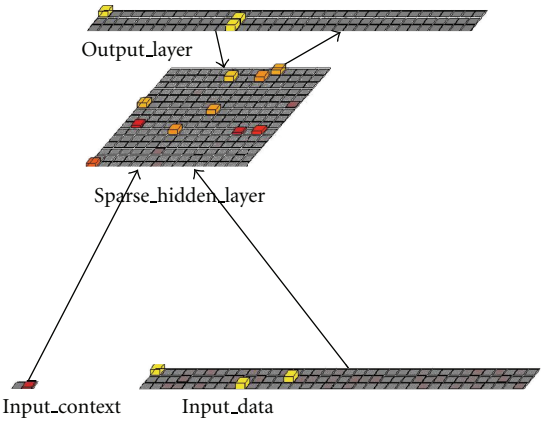


Figure 6: The Leabra network for robot manipulator simulations.

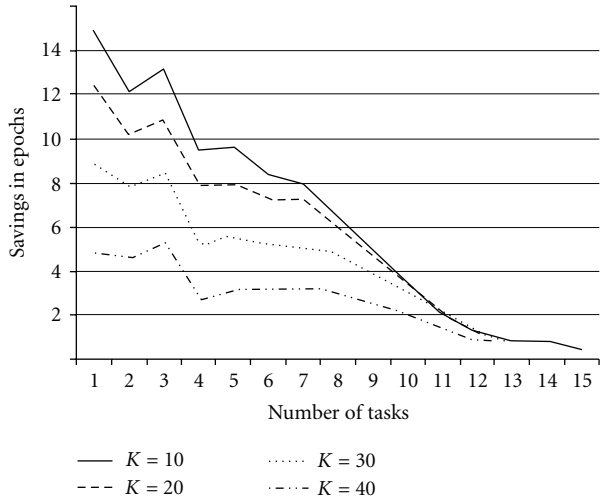


Figure 7: Savings in successive tasks.

is found to be sensitive to the strength of the contextual cues input. Certain neurons develop strong connection weights for the common subtask during the base task training. Due to strong weights, these neurons receive stronger input for the common subtask even during the interfering task training. Hence, they have a greater chance of firing. Contextual cues, however, provide strong inputs to certain other neurons in the hidden layer that compete with the shared neurons. If the cue signal is too strong, these other neurons win the competition, and the network fails to generalize the common subtask. We have shown that the cue strength can be set to an optimum value such that cues are strong enough to enforce savings through the generation of orthogonal representations, while still allowing the emergence of similar representation for the common subtasks between different tasks. The generalization observed during sequential learning of the tasks is comparable to the generalization observed when the two tasks are learned in an interleaved manner.

While our previous work [2] demonstrated how to overcome catastrophic interference via generalization on a Leabra Network, the current paper demonstrates the application of savings to a simulated robotic arm. We have shown that the application of biologically plausible savings during the sequential learning of robotic manipulator tasks has the additional benefit of retention of previously learnt tasks. We demonstrated that with a minimal amount of retraining the manipulator is able to perform the original task accurately. This highlights the advantages of designing a network which is capable of learning multiple tasks in sequence. It saves the user the trouble of having to interleave the tasks or generating artificial pseudopatterns. While human motor skill learning is considerably more nuanced, this framework offers insight into the rapid reacquisition after extinction exhibited by subjects during conditioning experiments. Many more applications of such networks may exist.

Even though biological plausibility has been the focus of our work, research in this direction is also significant for many engineering domains that use artificial neural networks. In many real-world scenarios, the full range of training data is not available up front. In such cases, traditional neural networks need to be updated as new information arrives. The techniques described in this paper can be used to improve the efficiency of such systems by enabling the networks to learn tasks in a sequential manner.

Appendices

A. Dendritic Integration

A fundamental function of neurons is the transformation of incoming synaptic information into specific patterns of action potential output. An important component of this transformation is synaptic integration, the combination of voltage deflections produced by a myriad of synaptic inputs into a singular change in membrane potential. Leabra simulates this integration at the dendrite of the neuron via a weighted summation of all the input activations followed by functional transformation (normally sigmoidal) of the sum.

B. Point Neuron Approximation

Leabra uses a point neuron activation function that models the electrophysiological properties of real neurons, while simplifying their geometry to a single point. This function is nearly as simple computationally as the standard sigmoidal activation function, but the more biologically based implementation makes it considerably easier to model inhibitory competition, as described below. Further, using this function enables cognitive models to be more easily related to more physiologically detailed simulations, thereby facilitating bridge-building between biology and cognition.

C. Lateral Inhibition

The processes involved in lateral inhibition are particularly relevant to the model presented in this paper. Lateral inhibition allows for competition between neurons involved in the encoding of stimuli. Along with the mechanisms of synaptic learning, this competition separates the neurons that associate the stimulus with responding, or acquisition neurons, from those which associate the stimulus with nonresponding, called extinction neurons. The class of in-hibitory functions that Leabra adopts are known as k-winners-take-all (kWTA) functions. A kWTA function ensures that no more than k units out of a total of n in a layer are active at any given point in time. This is attractive from a biological perspective because it captures the *setpoint* property of inhibitory interneurons, where the activity level is maintained through negative feedback at a roughly constant level (i.e., k).

D. kWTA Function Implementation

The k active units in a kWTA function are the ones receiving the most excitatory input (g_e). Each unit in the layer computes a layer-wide level of inhibitory conductance (g_i) while updating its membrane potential such that the top k units

will have above threshold equilibrium membrane potentials with that value of g_i, while the rest will remain below firing threshold. The function computes the amount of inhibitory current g_i^θ that would put a unit just at threshold given its present level of excitatory input, where θ is the threshold membrane potential value. Computing inhibitory conductance at the threshold (g_i^θ) yields

$$g_i^\theta = \frac{g_e^* \bar{g_e}(E_e - \theta) + g_l \bar{g_l}(E_l - \theta)}{\theta - E_i}, \tag{D.1}$$

where g_e^* represents the excitatory input minus the contribution from the bias weight and $g_l \bar{g_l}, g_e \bar{g_e}$ are the total conductances from the potassium and sodium channels, respectively. E_l and E_e are the equilibrium potentials for the the potassium and sodium channels, respectively [3]. g_i is computed as an intermediate value between the g_i^θ values for the kth and $k+1$th units as sorted by level of excitatory conductance (g_e). This ensures that the $k+1$th unit remains below threshold, while the kth unit is above it. Expressed as a formula this is given by

$$g_i = g_{k+1}^\theta + q\left(g_i^\theta(k) - g_i^\theta(k+1)\right), \tag{D.2}$$

where $0 < q < 1$ determines where the inhibition lies between the k and $k+1$th units.

E. Leabra Learning Algorithms

Leabra provides for a balance between Hebbian and error-driven learning. Hebbian learning is performed using a conditional principal component analysis (CPCA) algorithm. Error-driven learning is performed using GeneRec, which is a generalization of the recirculation algorithm and approximates Almeida-Pineda recurrent backpropagation.

F. Hebbian Learning

The objective of the CPCA learning rule is to modify the weights for a given input unit (x_i) to represent the conditional probability that the input unit (x_i) is active when the corresponding receiving unit (y_j) is also active. This is expressed as

$$w_{ij} = P(x_i = 1 \mid y_i = 1) = P\left(x_i \mid y_j\right). \tag{F.1}$$

In (F.1) the weights reflect the frequency with which a given input is active across the subset of input patterns, represented by the receiving unit. If an input pattern occurs frequently with such inputs, then the resulting weights from it will be relatively large. On the other hand if the input pattern occurs rarely across such input patterns, then the resulting weights will be small. The following weight update rule achieves the CPCA conditional probability objective represented by (F.1)

$$\Delta w_{ij} = \epsilon\left[y_j x_i - y_j w_{ij}\right] = \epsilon y_j\left(x_i - w_{ij}\right), \tag{F.2}$$

where ϵ is the learning rate parameter. The weights are adjusted to match the value of the sending unit activation x_i, weighted in proportion to the activation of the receiving unit (y_j). Thus inactivity of the receiving unit implies that no weight modification will occur. Conversely, if the receiving unit is very active (near 1), the update rule modifies the weight to match the input unit's activation. The weight will eventually come to approximate the expected value of the sending unit when the receiver is active (consistent with (F.1)).

G. Error-Driven Learning

GeneRec implements error backpropagation using locally available activation variables thereby making such a learning rule biologically plausible. The algorithm incorporates the notion of plus and minus activation phases. In the *minus phase*, the outputs of the network represent the expectation or response of the network, as a function of the standard activation settling process in response to a given input pattern. Then, in the *plus phase*, the environment is responsible for providing the outcome or target output activations.

The learning rule for all units in the network is given by

$$\Delta w_{ij} = \epsilon\left(y_j^+ - y_j^-\right)x_i^- \tag{G.1}$$

for a receiving unit with activation y_i and sending unit with activation x_i. The rule for adjusting the bias weights is just the same as for the regular weights, but with the sending unit activation set to 1:

$$\Delta \beta_{ij} = \epsilon\left(y_j^+ - y_j^-\right). \tag{G.2}$$

The difference between the two phases of activation is an indication of the unit contribution to the overall error signal. Bidirectional connectivity allows output error to be communicated to a hidden unit in terms of the difference in its activation states during the plus and minus states. $(y_j^+ - y_j^-)$.

Acknowledgments

The authors would like to thank three anonymous reviewers for their valuable feedback which helped to significantly improve the quality of this paper.

References

[1] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: the sequential learning problem," in *Psychology of Learning and Motivation*, G. H. Bower, Ed., vol. 24, pp. 109–164, Academic Press, New York, NY, USA, 1989.

[2] A. Gupta and D. C. Noelle, "The role of neurocomputational principles in skill savings," in *Proceedings of the 27th Annual Conference of the Cognitive Science Society*, pp. 863–868, 2005.

[3] R. C. O'Reilly and Y. Munakata, *Computational Explorations in Cognitive Neuroscience*, MIT Press, 2000.

[4] Y. Zhao and C. C. Cheah, "Position and force control of robot manipulators using neural networks," in *Prroceedings of the IEEE Conference on Robotics, Automation and Mechatronics*, pp. 300–305, December 2004.

[5] M. J. Er and Y. Gao, "Robust adaptive control of robot manipulators using generalized fuzzy neural networks," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 3, pp. 620–628, 2003.

[6] R. S. Bapi, K. Doya, and A. M. Harner, "Evidence for effector independent and dependent representations and their differential time course of acquisition during motor sequence learning," *Experimental Brain Research*, vol. 132, no. 2, pp. 149–162, 2000.

[7] O. Hikosaka, K. Nakamura, K. Sakai, and H. Nakahara, "Central mechanisms of motor skill learning," *Current Opinion in Neurobiology*, vol. 12, no. 2, pp. 217–222, 2002.

[8] M. K. Rand, O. Hikosaka, S. Miyachi et al., "Characteristics of sequential movements during early learning period in monkeys," *Experimental Brain Research*, vol. 131, no. 3, pp. 293–304, 2000.

[9] R. M. French, Catastrophic interference in connectionist networks, Macmillan Encyclopedia of the Cognitive Sciences, 2011.

[10] T. Brashers-Krug, R. Shadmehr, and E. Todorov, "Catastrophic interference in human motor learning," *Advances in Neural Information Processing Systems*, vol. 7, pp. 19–26, 1995.

[11] M. Haruno, D. M. Wolpert, and M. Kawato, "MOSAIC model for sensorimotor learning and control," *Neural Computation*, vol. 13, no. 10, pp. 2201–2220, 2001.

[12] T. Brashers-Krug, R. Shadmehr, and E. Bizzi, "Consolidation in human motor memory," *Nature*, vol. 382, no. 6588, pp. 252–255, 1996.

[13] R. Shadmehr and H. H. Holcomb, "Neural correlates of motor memory consolidation," *Science*, vol. 277, no. 5327, pp. 821–825, 1997.

[14] C. Miall, "Modular motor learning," *Trends in Cognitive Sciences*, vol. 6, no. 1, pp. 1–3, 2002.

[15] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly, "Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory," *Psychological Review*, vol. 102, no. 3, pp. 419–457, 1995.

[16] A. Robins, "Catastrophic forgetting, rehearsal and pseudorehearsal," *Connection Science*, vol. 7, pp. 123–146, 1995.

[17] B. Ans, "Sequential learning in distributed neural networks without catastrophic forgetting: a single and realistic self-refreshing memory can do it," *Neural Information Processing*, vol. 4, no. 2, pp. 27–32, 2004.

[18] B. Ans, S. Rousset, R. M. French, and S. Musca, "Preventing catastrophic in- terference in multiple-sequence learning using coupled reverberating elman networks," in *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, 2002.

[19] I. H. Jenkins, D. J. Brooks, P. D. Nixon, R. S. J. Frackowiak, and R. E. Passingham, "Motor sequence learning: a study with positron emission tomography," *Journal of Neuroscience*, vol. 14, no. 6, pp. 3775–3790, 1994.

[20] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in Cognitive Sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[21] M. Botvinick and D. C. Plaut, "Doing without schema hierarchies: a recurrent connectionist approach to normal and impaired routine sequential action," *Psychological Review*, vol. 111, no. 2, pp. 395–429, 2004.

[22] R. M. French, "Dynamically constraining connectionist networks to produce distributed, orthogonal representations to reduce catastrophic interference," in *Proceedings of the 16th Annual Cognitive Society Conference*, 1994.

[23] M. Riedmiller and B. Janusz, "Using neural reinforcement controllers in robotics," in *Proceedings of the 8th Australian Conference on Artificial Intelligence*, 1995.

[24] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," *Artificial Intelligence*, vol. 72, no. 1-2, pp. 81–138, 1995.

[25] J. Johnson, R. Challoo, R. A. McLauchlan, and S. I. Omar, "A multi-neural network intelligent path planner for a robot arm," in *Proceedings of the Artificial Neural Networks in Engineering (ANNIE '96)*, 1996.

[26] T. Hesselroth, K. Sarkar, P. P. Van der Smagt, and K. Schulten, "Neural network control of a pneumatic robot arm," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 1, pp. 28–38, 1994.

[27] A. Bouganis and M. Shanahan, "Training a spiking neural network to control a 4-dof robotic arm based on spike timing-dependent plasticity," in *Proceedings of the IEEE World Congress on Computational Intelligences (WCCI '10)*, 2010.

[28] M. Vaezi and M. A. Nekouie, "Adaptive control of a robotic arm using neural networks based approach," *International Journal of Robotics and Automation*, vol. 1, no. 5, pp. 87–99, 2011.

[29] M. McCloskey, "Catastrophic interference in connectionist networks: the sequential learning problem," *Psychology of Learning and Motivation*, vol. 24, pp. 164–169, 1989.

[30] S. P. Singh, "Transfer of learning by composing solutions of elemental sequential tasks," *Machine Learning*, vol. 8, no. 3-4, pp. 323–339, 1992.

[31] L. M. Saksida, S. M. Raymond, and D. S. Touretzky, "Shaping robot behavior using principles from instrumental conditioning," *Robotics and Autonomous Systems*, vol. 22, no. 3-4, pp. 231–249, 1997.

[32] M. Dorigo and M. Colombetti, "Robot shaping: developing situated agents through learning," Tech. Rep. TR-92-040, International Computer Science Institute, 1993.

[33] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the 21st International Conference on Machine Learning (ICML '04)*, pp. 1–8, July 2004.

[34] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th International Conference On Machine Learning (ICML '09)*, pp. 41–48, June 2009.

[35] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: a survey," *Journal of Machine Learning Research*, vol. 10, pp. 1633–1685, 2009.

[36] H. Saal, J. Ting, and S. Vijayakumar, "Active sequential learning with tactile feedback," *Journal of Machine Learning Research*, vol. 9, pp. 677–684, 2010.

[37] K. A. Kruger, *Sequential learning in the form of shapingas a source of cognitive exibility*, Ph.D. thesis, Gatsby Computational Neuroscience Unit, University of London, 2011.

[38] R. Pollak, J. Schuetznerz, and T. Braunl, "Robot Manipulator Simulation," 1996, http://robotics.ee.uwa.edu.au/robosim/.

[39] C. Balkenius and J. Moren, "Computational models of classical conditioning: a comparitive study," Tech. Rep. LUCS 62, 1998.

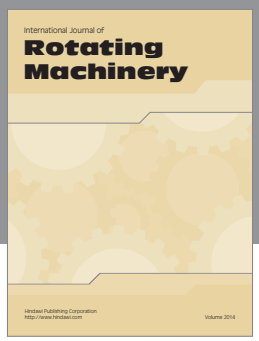
[40] R. A. Rescorla, "Retraining of extinguished Pavlovian stimuli," *Journal of Experimental Psychology: Animal Behavior Processes*, vol. 27, no. 2, pp. 115–124, 2001.

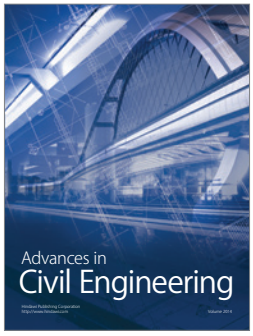
[41] R. A. Rescorla, "Savings tests: separating differences in rate of learning from differences in initial levels," *Journal of Experimental Psychology: Animal Behavior Processes*, vol. 28, no. 4, pp. 369–377, 2002.

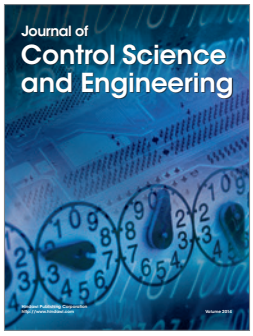
[42] R. A. Rescorla, "Comparison of the rates of associative change during acquisition and extinction," *Journal of Experimental Psychology: Animal Behavior Processes*, vol. 28, no. 4, pp. 406–415, 2002.

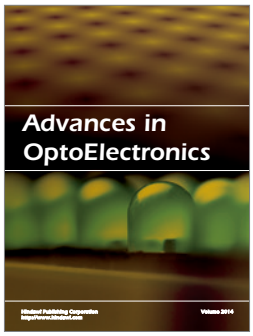
[43] R. A. Rescorla, "More rapid associative change with retraining than with initial training," *Journal of Experimental Psychology: Animal Behavior Processes*, vol. 29, no. 4, pp. 251–260, 2003.

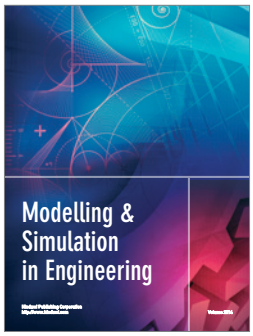
[44] G. S. Reynolds, *A Primer of Operant Conditioning*, Scott, Foresman and Company, 1975.

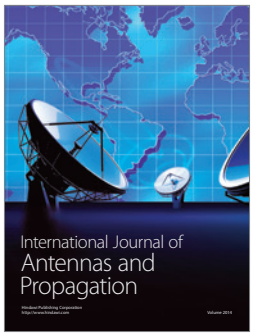


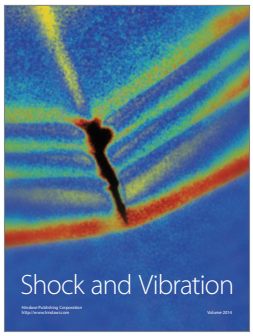


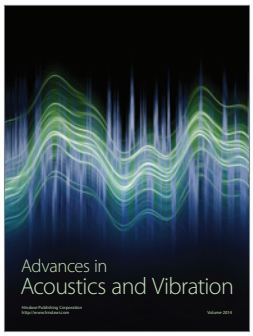












Hindawi

Submit your manuscripts at
http://www.hindawi.com