On the Relationships between Synaptic Plasticity and Generative Systems

Paul Tonelli
tonelli@isir.upmc.fr

Jean-Baptiste Mouret
mouret@isir.upmc.fr

ISIR, Université Pierre et Marie Curie-Paris 6, CNRS UMR 7222
4 place Jussieu, F-75252, Paris Cedex 05, France

ABSTRACT

The present paper analyzes the mutual relationships between generative and developmental systems (GDS) and synaptic plasticity when evolving plastic artificial neural networks (ANNs) in reward-based scenarios. We first introduce the concept of synaptic Transitive Learning Abilities (sTLA), which reflects how well an evolved plastic ANN can cope with learning scenarios not encountered during the evolution process. We subsequently report results of a set of experiments designed to check that (1) synaptic plasticity can help a GDS to fine-tune synaptic weights and (2) that with the investigated generative encoding (EvoNeuro), only a few learning scenarios are necessary to evolve a general learning system, which can adapt itself to reward-based scenarios not tested during the fitness evaluation.

Categories and Subject Descriptors

I.2.6 [**Artificial intelligence**]: Learning—*Connectionism and neural nets*

General Terms

Algorithms

Keywords

evolutionary algorithms; neural networks; synaptic plasticity ; generative and developmental systems; neuro-evolution.

1. INTRODUCTION

A major goal of bio-inspired artificial intelligence is to design artificial neural networks (ANNs) with features and abilities similar to those of animal nervous systems. According to the current scientific consensus, the primary process responsible for shaping these complex networks is Darwinian evolution; this suggests that evolution-inspired algorithms are a sensible method to design "artificial nervous systems". Despite the large amount of work in this direction, three striking differences still separate biological nervous systems from most artificially-evolved ones: biological nervous systems are much larger, much more organized (they are modular, regular and hierarchical [8]) and *much more plastic* [1], that is, they can adapt themselves online. Structural challenges (modularity, regularity and scalability) are one of the main focus of current researches with generative and developmental systems (GDS) [7–9, 12, 14, 22]; plastic ANNs were also evolved in several studies, in which Hebbian learning was added to evolved neural networks [7, 15–20, 27]. Yet only a few researchers analyzed the combination of synaptic plasticity and GDS [7, 16, 19], whereas, as the present paper will show, plasticity and GDS are deeply connected.

Indeed, most GDS aim at evolving short descriptions of complex structures by taking advantage of regularities observed in Nature, such as repetition of useful sub-parts, symmetries, symmetries with variation, ... [8,22]. However, these regularities come with a cost: the more an ANN is regular, the more difficult it is to tune a particular connection [4]. To reconcile regularity and fine-tuning, animal brains strongly relies on synaptic plasticity during their lifetime, in particular during their development. It follows that *ANNs evolved via a GDS should similarly benefit from a fine-tuning by synaptic plasticity mechanisms.*

Perhaps less intuitively, evolving plastic ANNs may also need a GDS to scale up to real-world problems without prohibitively long evaluation times. Evolving plastic ANNs currently requires long fitness evaluations because (1) one must must allow enough time for the agent to learn and (2) one must ensure that each possible learning scenario (e.g. different positions of reward) can be learned. This second point is very important because the number of scenarios tends to grow exponentially with the number of alternatives; testing most of them when evaluating the fitness arguably prevents the evolution of plastic ANNs for anything else than toy problems. Besides this computational issue, one of the goal of designing plastic ANNs is to make agents able to react to *unknown situations* which will obviously not be known during the evolutionary process. Put differently, a lot of computation time is employed to encourage the evolutionary process to find *a general learning system* and not specialized adaptation rules. For instance, if an agent must associate stimuli (e.g. lights) with actions (e.g. pushing a lever), the same evolved agent should be able to use a reward to associate a given stimulus (e.g. light 1) with a given action (e.g. lever 1) as easily as any other association (e.g. light 2 with lever 1), *by tuning online a few plastic synapses elicited through evolution.*

GECCO'11, July 12–16, 2011, Dublin, Ireland.

This requirement raises a new question for the evolution of plastic ANNs: *how to evolve plastic ANNs which can adapt themselves to situations that have not been tested during the evolution?* Such a skill will be called *Transitive Learning Ability* (TLA) in the remainder of this paper. At first sight, Nature relies on the long lifetime of animals (compared to the "lifetime" of artificial agents) and on the large size of the populations to obtain a stochastic evaluation of virtually every possible scenarios; however, the encoding and the development process may also play a key role in adapting to situations which have never been encountered before. The present paper investigates this second idea—the importance of the link between sTLA and GDS. Intuitively, a very regular network may repeat the same adaptation structure many times whereas it was only required once by the fitness; it could therefore "propagate" the adaptation structure. Using a carefully designed GDS, one should consequently be able to substantially reduce the number of evaluations required to obtain a general learning system, thus improving the abilities of ANNs to adapt themselves to unforeseen situations.

Following this line of thought, synaptic plasticity and GDS should benefit from each other to evolve ANNs. The present paper looks into this deep relationship which was almost never investigated in the literature. After a short review of related work, we introduce an property that a plastic ANN must possess to ensure it can adapt itself to unforeseen situations. We then propose that ANNs evolved with a map-based encoding (EvoNeuro encoding [14]) possess this property. We subsequently report results of a set of experiments designed to check that (1) synaptic plasticity can help a GDS to fine-tune synaptic weights and (2) with the investigated generative encoding, only a few learning scenarios are necessary to evolve a general learning system, which can adapt itself to reward-based scenarios not tested during the fitness evaluation.

2. RELATED WORK

2.1 Evolving Adaptive Neural Networks

Synaptic plasticity underlies most models of learning, memory and development in animals [1]; this ubiquity makes it one of the most studied topic in neuroscience. It has been described at many levels of detail, but studies on the evolution of plastic ANNs are mainly focused on Hebbian-like adaptation rules, according to which the strength of connection is modified with regard to pre- and post-synaptic activity [2, 26, 27]. A synapse can also be strengthened or weakened as a result of the firing of a third, modulatory inter-neuron (e.g. dopaminergic neurons). To reflect this phenomenon, a modulation factor m can be included in a classic Hebbian rule [17, 19]:

$$\Delta w_{ij} = m \cdot (A \cdot a_i \cdot a_j + B \cdot a_i + C \cdot a_j + D) \qquad (1)$$

where i and j are neurons, Δw_{ij} is the modification of synaptic weight w_{ij}, a_i is the activation of neuron i, m is the sum of the modulating signals received by the post-synaptic neuron and $(A, B, C, D) \in \mathbb{R}^4$ are four parameters of the rule.

Soltoggio et al. [18] successfully used this heterosynaptic rule to evolve plastic ANNs in a simple dynamic, reward-based scenario: a robot is put in a T-maze in which it has to find a reward, whereas this reward is regularly swapped from one end of the T-maze to the other. These authors designed a fitness such that the best individuals are those that manage to switch their behavior as fast as possible when the position of the reward is changed. Enabling heterosynaptic plasticity substantially improved the performance of evolved controllers, thus showing the potential of plastic ANNs. Nevertheless, this setup does not allow to test controllers in unknown situations, because successful controllers necessarily have encountered the two positions of the reward; one may therefore ask whether the evolutionary algorithm designed "controllers that can learn" or, instead, exploited the rich dynamic provided by plasticity to design a network that can select one behavior among two pre-evolved ones. A similar analysis can be drawn for several other papers based on T-maze experiments [16, 17, 20].

Urzelai and Floreano [27] then Blynel and Floreano [2] approached synaptic plasticity from the point of view of behavioral robustness. They first evolved neuro-controllers with plastic synapses to solve a light-switching task *in which there was no reward*; they then investigated whether these controllers were able to cope with four types of environmental changes: new sensory appearances, transfer from simulations to physical robots, transfer across different robotic platforms and re-arrangement of environmental layout. The plastic ANNs were able to overcome these four kind of changes, contrary to a classic ANN with fixed weights. However, as highlighted by the authors, "these behaviors were not learned in the classic meaning of the term because they were not necessarily retained forever". Actually, synaptic weights were continuously changing such that the robot performed several sub-behaviors in sequence; the evolutionary algorithm therefore opportunistically used plasticity to enhance the dynamic power of the ANN and not to change the behavior with regard to a new situation.

In supervised learning, Chalmers [3] assessed how well an evolved plastic ANN can cope with situations never encountered during the evolution. In his experiments, he evolved the learning rule for a small single-layer ANN (5 inputs, 1 output) and his analysis showed that at least 10 sets of input/output patterns (among 30 possible sets) were required to evolve an algorithm that correctly learns on 10 unknown sets. In reinforcement learning, Niv et al. [15] evolved plastic ANNs to solve a bumblebee-inspired foraging task in which simulated bees must select flowers by recognizing their color. To promote general learning abilities, they randomly assigned rewards to colors at each generation and they showed that the resulting ANNs successfully learned unknown color/reward associations. Chalmers and Niv et al. both had to use a large number of the possible scenarios to lead to general learning abilities, but this approach is only possible for very simple domains. Stanley et al. [21] similarly randomized the fitness parameters to avoid overspecialized behaviors, but they show that, surprisingly, plasticity did not help in the task they studied. At any rate, these authors did not discuss about how the encoding and the chosen topology affected their results.

2.2 Generative and Developmental Systems

Inspired by the regularities of natural organisms and human-made designs, many researchers study how these regularities can emerge in both natural and artificial evolution. In the latter, their effort is focused on "generative and developmental systems" (GDS), in which structures are genetically encoded by a compact representation that is then *developed*. In particular, these researches have led to many

indirect encoding for ANNs, inspired by L-systems [9], gene regulatory networks [12], chemical gradients [4, 22], ...

Despite promising results with GDS, fostering regular networks is necessarily to the detriment of fine-tuning individual connections: the more the representation of a network is compact, the more it is difficult to add exceptions to the general pattern. This trade-off was recently investigated by Clune et al. [4] who have shown that the performance of HyperNEAT [22] decreased as problem regularity decreased. To solve this issue, Clune et al. proposed a two-stage algorithm, called HybrID, to combine the benefits of both approaches: in the first stage, the ANN is evolved with HyperNEAT to discover the general patterns; in the second stage, the encoding is switched to a direct encoding to account for irregularities. In their experiments, this hybrid algorithm outperformed both HyperNEAT and the tested direct encoding. Although this procedure seems efficient, tuning individual connections by evolution is unrealistic from a biological point of view and no method are known to compute the ideal switch generation. Instead, the direct encoding phase can be assimilated to a learning procedure, which could be performed on-line with synaptic plasticity.

Surprisingly, despite the large amount of work about GDS for neural networks, only a few authors have included synaptic plasticity into their system. Gruau and Whitley [7] evolved neural networks with the cellular encoding and Hebbian synapses on the connections that feed into output units; they were mostly interested in how adding learning can change the fitness landscape. Soltoggio et al. [19] used the a generative encoding inspired by gene regulatory networks (AGE, [12]) to design neural networks with heterosynaptic plasticity. However, they focused on the benefits brought by heterosynaptic plasticity and, consequently, the generative encoding was only used "as a tool" to evolve a topology. Last, Risi and Stanley [16] extended the HyperNEAT encoding to evolve both synaptic weights and parameters of learning rules. They tested their method on the T-maze experiments (see section 2.1) and thus were not able to test the general learning abilities of evolved ANNs.

2.3 Novelty search

Evolving plastic ANNs raises a technical challenge for any evolutionary algorithm: in most situations, there exists a non-plastic (or non-adaptive) ANN that, despite being non-optimal, solves a significant part of the task [17, 21]. Unfortunately, it is often impossible for the algorithm to add plastic synapses *a posteriori* without significantly degrading the fitness. From the optimization point of view, this makes most fitness functions that reward learning behaviors very deceptive [16, 20]. A direct consequence is that most fitness functions employed to evolve plastic ANNs have to be precisely crafted to make the adaptive behaviors very attractive, whereas an ideal fitness function should be straightforwardly deduced from the task.

To avoid such a deceptiveness, several recent papers proposed to explicitly reward the novelty or the diversity of behaviors [11, 13, 17]. Once a behavioral distance has been designed, it is indeed possible to compute how much each individual differs from those of the previous generations. A new objective can thereafter be defined: maximizing the novelty (or the diversity) of behaviors. Several experiments [11, 17] have shown that this new objective can efficiently replace the fitness to overcome its deceptiveness, leading to an approach called "novelty search". In particular, Risi et al. [17] applied this algorithm to successfully evolve plastic ANNs to solve the T-maze problem. This novelty/diversity objective can also be combined with the fitness function in a Pareto-based multi-objective optimization, so as fitness and novelty/diversity can complete each other [13].

3. TRANSITIVE LEARNING ABILITIES

3.1 Definitions

In the present paper, we are interested in evolving ANNs that learn to select the most rewarding action given some stimuli. For the sake of simplicity, we assume that the reward is never delayed and that there is always a reward, may it be positive or negative. This framework corresponds to many setups from operant conditioning. An ANN with synaptic General Learning Abilities (sGLA) must be capable to learn each possible association of stimulus/action with the same topology, the same learning function but a different reward scheme. More formally, the neural network $N(I, \lambda)$ must adapt several synaptic weights $\lambda \in \mathbb{R}^z$ such that each input pattern $I \in [0,1]^n$ is associated to the best rewarded output vector $K \in [0,1]^m$, that is λ is optimal when $N(I, \lambda) = K$; the adaptation is performed by a learning function such that $\lambda = g(\lambda_r, I, R_{I,K})$, where λ_r is a random vector in $\mathbb{R}^z$ and $R_{I,K}$ the reward function. These notations lead to the following definitions:

Definition 1 (Association). *An association is a pair (I, K) of input/output that leads to the maximum positive reward.*

Definition 2 (Association set). *An association set $A = \{(I_1, K_1), \cdots, (I_n, K_n)\}$ is a list of associations that covers all the possible input patterns. The set of all association sets is denoted $\mathbb{A}$.*

Definition 3 (Learnable set). *Given a suitable reward function $R_{I,K}$, an association set $A \in \mathbb{A}$ is said to be learnable by the neural network N, if and only if $\forall \lambda_r \in \mathbb{R}^z$ and $\forall (I, K) \in A$, $\exists \lambda = g(\lambda_r, I, R_{I,K})$ such that $N(I, \lambda) = K$. The set of all learnable sets for N is denoted $\mathbb{L}_N$.*

Definition 4 (sGLA). *A plastic ANN is said to possess synaptic General Learning Abilities (sGLA) if and only if $\forall A \in \mathbb{A}$, $A = \mathbb{L}_N$.*

To evolve a plastic ANN with sGLA, the typical method is to check the learnability of each association set during the fitness evaluation, as it is often done in the T-maze experiments. However, to cope with unknown situations, a plastic ANN must have sGLA while only a subset of the possible association sets (i.e. a subset of problems from the same problem class) has been used during the evolutionary process. Put differently, it is desirable to work with plastic ANNs for which knowing that a few association sets are learnable is sufficient to know that the ANN possesses sGLA. We call this property "synaptic Transitive Learning Abilities" (sTLA), defined as follows:

Definition 5 (sTLA). *A plastic ANN is said to possess synaptic Transitive Learning Abilities (sTLA) if and only if $\exists \mathbb{T}_N \subset \mathbb{A}$ such that the following implication is true: $\mathbb{T}_N \subset \mathbb{L}_N \Rightarrow \mathbb{L}_N = A$. $p = card(\mathbb{T}_N)$ will be called the "sTLA-level".*

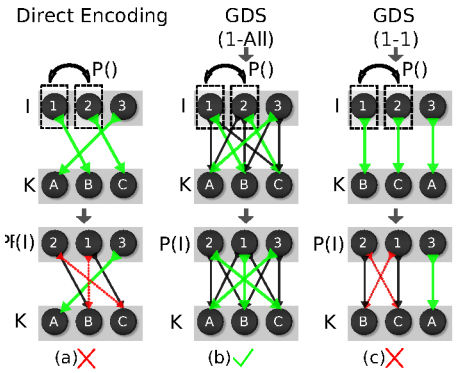

Figure 1: (a) When two neural maps are connected with irregular plastic connections, some patterns cannot be learned, for instance if two input neurons are swapped; (b) if a GDS generate networks in which neural maps are connected by one-to-all connections, swapping two inputs have no effect on the learnability of the association; (c) one-to-one connections can prevent the ANN from learning new associations.

As far as we know, the concept of sTLA is original. A sTLA-level means that fewer experiments have to be carried out to check if an ANN have sGLA. Consequently, the smaller the sTLA-level is, the faster evolving an ANN with sGLA will be. In the ideal case, the sTLA-level could be as small as 1: checking only one association set would be enough to be sure that the ANN possesses sGLA.

DEFINITION 6 (OPTIMAL-STLA). *A plastic ANN is said to possess Optimal synaptic Transitive Learning Abilities (sTLA) if and only if it possesses sTLA and* $card(\mathbb{T}_N) = 1$.

3.2 Map-based Neural Networks

A well-chosen list of association sets could lead to sGLA for evolved plastic ANNs. However, it seems unlikely that only one or two associations sets will suffice to get sGLA in a fairly complex task. In the present paper, we focus on an alternative approach: ensuring optimal-sTLA (p equals 1) or near-optimal-sTLA by constraining topologies.

ANNs with irregular topologies, for instance those evolved with a direct encoding, encounter difficulties to learn all the association sets because only the existing connections can be tuned by synaptic plasticity mechanisms (figure 1(a)), whereas some new connections may be required for a particular association; this is why many association sets (if not all) have to be tested to know if such networks have sGLA. By contrast, if two maps of identical neurons are fully connected by plastic connections, then any input can be swapped without harming the learnability of the association set (figure 1(b)). Interestingly, ANNs designed in computational neuroscience (and most of those used in machine learning) are also based on a network of neural maps or layers (e.g. [6]) with regular connection schemes between maps. They rely on two main connection schemes: one-to-all connections (with identical or plastic synaptic weights) and one-to-one connections [14]. Neural maps with these two connections schemes allow to model a large number of brain structures, from basal ganglia [6,14] to colliculus [25].

Let's now suppose that we are interested in the simplest associations sets, in which each input is associated to exactly one output and in which only one input/desired output can be activated at the same time. Any association set can be selected except the "direct" set which can be trivially solved with a single one-to-one connection between outputs and outputs. For instance, here is a typical association set for 3 inputs and 3 outputs: $\{[0,0,1] \rightarrow [1,0,0]; [0,1,0] \rightarrow [0,1,0]; [1,0,0] \rightarrow [0,0,1];\}$. Since non-plastic connections between two maps all have the same synaptic weights, evolving a network of neural maps that perform this kind of association is a complex task without plasticity. If plasticity is enabled, there exist no reasonably simple topology that would make Hebbian learning work for only this association: connections cannot be crossed as with a direct encoding and nothing distinguishes two neurons of a map except their position, consequently there must exist in the ANN a Hebbian one-to-all connection modulated by the reward signal. Since we did not assume anything about the association set, there is a high probability that the same plastic one-to-all connection can be tuned for other similar associations sets. This line of thought lead to the following proposition:

PROPOSITION 1. *If the associations are restricted to "one input is associated to one output" then any network of maps connected together by either one-to-all or one-to-one connections is sTLA-optimal* ($p = 1$).

In the future, it may be possible to formally prove this proposition. The sTLA-level of a family of topologies can also be empirically evaluated by a succession of evolutionary experiments: (1) select p association sets; (2) evolve ANNs that successfully learns the p association sets; (3) check the sGLA of optimal ANNs; (4) if optimal ANNs do not possess sGLA, then increase p and start again.

3.3 Evolving Plastic, Map-Based ANNs

Weight Modulation Mechanism.

We opt here for a basic neuro-modulated Hebbian rule (equation 1) in which coefficients B, C, D are set to 0. This rule was chosen because we succeeded to reproduce several classic adaptive models using it (such as neuron-based actor critic [24]) while it remains simple. Following the work by Soltoggio [18], we distinguish two types of neurons: "standard neurons" and "modulatory neurons". To use the modulation factor m of equation 1, inputs of each neuron are divided into modulatory inputs I_m and standard I_s inputs. The output a_i of a neuron i then defined as follows:

$$a_i = \varphi_1\Big(\sum_{j \in I_s} w_{ij} a_j + b_i\Big) \tag{2}$$

where i is the identifier of a neuron, a_i its output, b_i its bias, $\varphi_1(x)$ a sigmoid on $[0,1]$, w_{ij} the synaptic weight between neurons i and j. Each non-modulatory synaptic weight w_{ij} is modified with regards to the sum of modulatory inputs and a constant coefficient η:

$$m_i = \varphi_2\Big(\sum_{j \in I_m} w_{ij} a_j\Big) \tag{3}$$

$$\Delta w_{ij} = \eta \cdot m_i \cdot a_i \cdot a_j \tag{4}$$

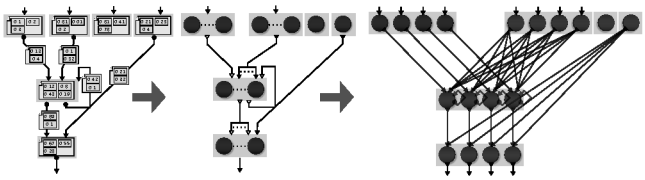

Figure 2: Example of a neural network encoded with a map-based encoding (EvoNeuro [14]). (left) the genotype is a labeled graph. (center) this graph describes a network of neural maps. (right) final developed phenotype with maps and connections.

$$w_{ij}(t+\delta t) = \begin{cases} \min(\max(w_{ij}(t) + \Delta w_{ij}, 0), 30) \\ \text{ if } w_{ij}(t) \geq 0 \\ \min(\max(w_{ij}(t) - \Delta w_{ij}, -30), -10^{-5}) \\ \text{ if } w_{ij}(t) < 0 \end{cases}$$

Generative Encoding.

A network of neural maps can be efficiently represented as a labeled graph whose labels describe the properties of each map and of each connection [14] (figure 2). Three parameters are associated to each vertex of the graph: (1) neural map / single neuron (Boolean); (2) modulatory neuron(s) / standard neuron(s) (Boolean); and (3) bias of the activation function(s) (real value). In the case of a vertex which is a modulatory map, each neuron in the map is modulatory. Each edge of the graph is labeled with two parameters: (1) synaptic weight (real value); and (2) connection type ("one to one" or "one to all") (Boolean).

Such graphs are evolved with a classic direct encoding, broadly inspired by the NEAT encoding [23]. Mutations act directly on the graph and cross-over is not employed; three mutations are possible: (1) add/remove a node; (2) add/remove/change a connection; (3) mutate labels.

The size of maps is not evolved and is set as a parameter of the experimental setup (the same graph can therefore be interpreted for different map sizes). The topology of the graph is not restricted; in particular, recurrent connections are possible. More details about this map-based encoding are available in [14].

Control experiments: direct encoding.

The previously described evolvable labeled graph can also be employed in a more classic fashion to directly define an ANN. This leads to a direct encoding of ANNs. In this case, each node describes a neuron (instead of a map), labels used to describe neurons are the same as those used to describe maps (bias) and connections are labeled with a single real number interpreted as the synaptic weight. For these control experiments, each input/output map is replaced with N neurons (each of them with its own parameters).

4. EXPERIMENTAL SETUP

Two main hypotheses have been introduced in the previous sections: (1) an ANN evolved with a GDS should benefit from synaptic plasticity to counterbalance its regularity and, (2) plastic ANNs evolved with a map-based encoding should have better transitive learning abilities than ANNs evolved with a direct encoding. To further investigate these hypotheses, we designed a set of experiments in a simulation of a setup used to study operant conditioning with animals, commonly called a "Skinner box". In this type of experiment a caged animal, usually a mouse or a rat, is presented with stimuli which can be either sound, light or a combination of the previous elements. When the stimuli are presented, the animal must perform a specific action to obtain a reward. The typical action is the activation (push) of a lever and the reward is usually food or water. If the action is performed without the right stimulus, a punishment is given (electric shock). An example is shown in figure 3.

This task was selected for two main reasons. Firstly, it purposely does not involve delayed reward, a well known problem in reinforcement learning [24] which requires more complex mechanisms than basic neuro-modulation [10]. Secondly, this task presents a choice with many possible combinations for the answers. It is therefore different from a switch between only two alternatives, as done by most previous authors [17,18]. Furthermore, as the number of stimuli and possible actions can be changed, the task's complexity can be arbitrarily increased or decreased. This feature will allow us to test the synaptic Transitive Learning Abilities (sTLA) of the evolved networks.

4.1 Formalization

An agent in a skinner box can be formalized as a system which associates an input vector of size N_s to a particular action among N_a possible actions. To simplify the use of neural maps, we consider here that $N_s = N_a$. The goal of the agent is to learn a set of associations that links each possible N_s to the action that maximizes its reward. To make the task easier, we restricted ourselves to vectors N_s in which only one input is active at a given time. Such a constraint corresponds to classic reinforcement learning setups where an input is equivalent to a state and in which two inputs cannot be active at the same time. Depending on N_s, there are $N_s^{N_s}$ possible association sets. If $N_s = 4$, there are 256 association sets.

To evaluate the performance of an agent, the ANN is simulated with inputs which are not subject to random processes or noise other than the weight initialization. This makes it difficult for the network to have an exploration behavior, whereas we know that this kind of exploration behavior is necessary to test the different solutions. The controller also cannot perform multiple actions at once (if we take the analogy of the rat in a cage, it cannot push more than one lever at a time). The most elegant solution we found to solve both of these problems is to add a probabilistic selection method to choose to final output of the ANN. We rely on a method commonly used in reinforcement learning [24] called "softmax" according to which the probability of selecting an output is linked to the output level of this output compared to the others as follows:

$$P(i) = \frac{\exp(\beta a_i)}{\sum_{j=1}^{n} \exp(\beta a_j)} \quad (5)$$

where $P(i)$ is the probability of selecting output i, a_i is the activity of output i and β is a fixed coefficient. From previous experiments, we also observed that if we use only a max function on the outputs instead of the softmax, the evolutionary algorithm usually finds solutions where the difference between outputs is around 10^{-2} for a total range of

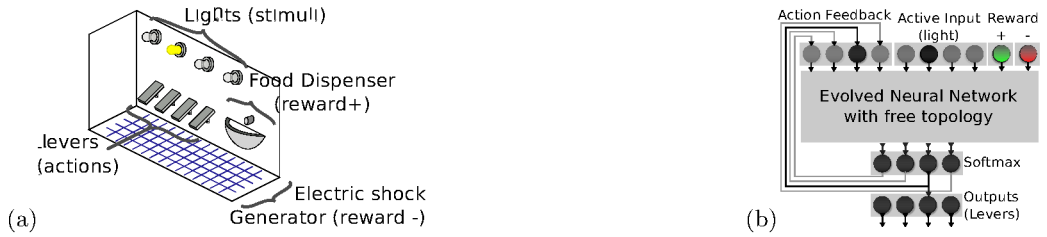

Figure 3: (a) Example of a typical Skinner box used in operant conditioning. (b) Formalization using an evolved neural network setup.

1. Using the softmax mechanism, we constrain the network to differentiate its outputs by a significant value.

The softmax mechanism selects an output based on both the output values and a random process, therefore, if we want the network to adapt, we must add inputs that provides to the network the decision of the softmax and the associated reward (such a feedback loop is implicit in reinforcement learning models). The inputs of the networks can therefore be organized in three groups : (1) the input pattern from the rule (the map encoding is used, it is a map of size N_s, otherwise there are N_s separate inputs); (2) the feedback pattern from the last softmax choice (either a map of size N_s or N_s separate inputs); (3) two additional inputs which represent the positive and negative rewards (both are single neurons). The overall network topology is described on figure 3(b).

The ANNs that solve this task may seem trivial at first sight. However, we were unable to hand-design solutions that did not require at least one hidden layer of neurons with non-trivial connectivity. The challenge raised by this problem is threefold: (1) identifying and correctly connecting the reward inputs, (2) gating the reward with the softmax choice to modify only the connections corresponding to the chosen action, and (3) applying the resulting reinforcement to a link between the inputs and the output.

4.2 Objectives

Fitness Objective.

The evaluation of each controller follows the diagram displayed in figure 4. Each controller is evaluated for p association sets, each of them consisting in N_s different associations. For each association set, the modulated weights are randomly initialized because we only test the ability to learn and not ability to forget then learn. The fitness is the number of associations that the ANN answered correctly.

For in each set, each association is presented in a pseudo random order to the network. Each presentation is done in two parts: (1) the pattern is presented to the network while the other inputs are at O; after k_1 update cycle (to let the ANN converge), the output of the evolved ANN is read and fed into the softmax mechanism, which selects the active output; (2) the input pattern and the softmax output are kept on the input layer, while the reward input are set. After k_2 cycles, another association is selected. For each association, the fitness score of the network is updated with regard to the output corresponding to the last presentation.

Diversity Objective.

As explained in section 2.3, using the behavioral differences to force the EA to explore can improve results with deceptive fitness functions. In the present setup, we describe the behavior of a network with a vector containing the post-learning output values of the network for each tested association set. This both partially reduces the softmax mechanism impact on the measure and reduces the importance of the random initialization of the modulated weights. The distance between behaviors is computed with an Euclidean distance between these behavioral vectors. The diversity score of each individual is the average distance from its 15 nearest neighbors. We use the diversity score as a second objective in a multi-objective EA.

Objectives are shown in equation 6, where the first objective is the performance with δ being equal to 1 if the given answer (A_g) is identical to the correct answer (A_c), else 0, while k is the total number of associations tested. The second objective is the diversity, which is the sum of the distance between behaviors with respect to the 15 nearest neighbors.

$$\text{Maximize} \begin{cases} Fit(x) = \frac{1}{k}\sum_{i=1}^{k} \delta(A_g, A_c) \\ Div(x) = \sum_{j=1}^{15} d(B_x, B_j) \end{cases} \quad (6)$$

5. EXPERIMENTS AND RESULTS

The same objectives, experimental setups and parameters were used in two batches of experiments (section 4.2), each one testing one of our hypotheses. To successfully learn each input/output pattern, a network must have tested each output to find the one providing the highest reward. To enable a network with a random output to select every correct association at least once with a probability superior to 0.99, we perform 90 learning cycles during which we randomly select an association from the set and present it to the network (using $N_s = N_a = 4$).

We chose NSGA-II [5], a state-of-the-art multiobjective evolutionary algorithm, to optimize the two objectives (main fitness and novelty). Each experiment is launched 30 times to obtain statistically significant results.

For further details, please check the source code associated to this paper on the website: `http://www.isir.fr/evorob_db`

5.1 Counterbalancing regularity

Experiment. In this batch of experiments, we evolved plastic and non plastic ANNs that outputs perfect answers for a particular (arbitrarily chosen) association set, with a direct encoding and a generative encoding.

Rationale. Evolving an ANN to match a single association set does not require any plasticity mechanism: to make sure that input i is associated to output j, an ANN only needs a connection between neuron i and neuron j. As a

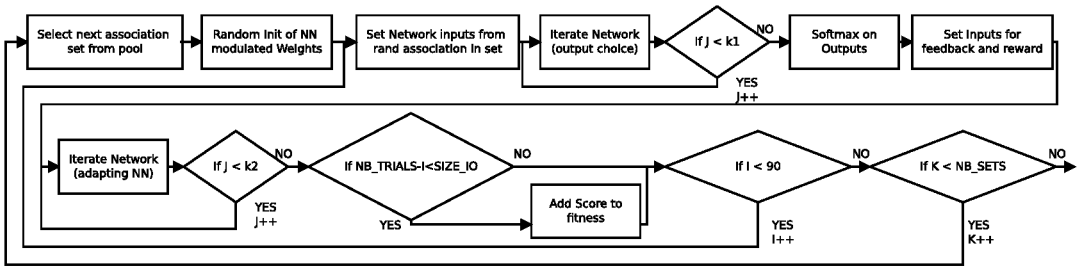

Figure 4: Flowchart of the evaluation procedure.

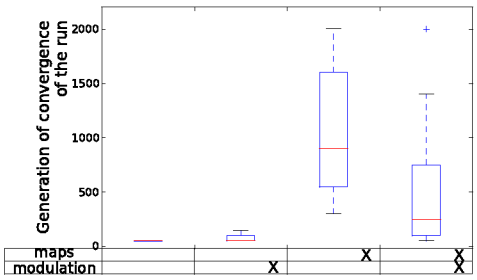

Figure 5: Comparison of median convergence generation on a simple task requiring no adaptation.

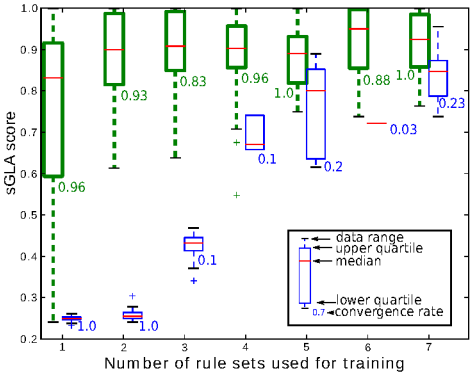

Figure 6: Success rate on learning association sets different from the ones used in evolution with (bold boxes) and without using EvoNeuro. Only successful runs ($Fit(x) = 1$) are taken into account.

consequence, this task should be trivial to solve with a direct encoding. Nevertheless, it should be especially difficult to solve in a network encoded as a network of neural maps—a highly regular generative encoding—, because all synaptic weights of a one-to-all or on a one-to-one connection are exactly the same. Hence, we expect that synaptic plasticity will be required to evolve an ANN with the EvoNeuro encoding for this task, whereas a direct encoding should solve it easily without plasticity.

Results. Figure 5 confirms our hypothesis: while the direct encoding easily solved the problem in less than 50 generations, the non-plastic GDS required about 900 generations (median). However, adding modulation to the GDS divided the median convergence generation by four, hence showing that adding modulation to a GDS with regularities improves its abilities to generate specific connectivity patterns.

We wondered how the GDS could solve the problem without modulation, as we initially thought it should not have been possible. In these experiments, the evolved networks relied on dynamic loops which can also provide adaptive behaviors.

5.2 Synaptic transitive learning abilities

Experiment. We launched 7 batches of 30 experiments in which plastic ANNs (with both the generative encoding and a direct encoding) were evolved with a different number of association sets: one batch with one set, one batch with two sets, etc. In a second step, we tested the sGLA of the optimal ANN of each run and deduced an empiric estimation of the sTLA-level with each encoding.

Rationale. As highlighted in section 3.2, a plastic ANN evolved with the map-based encoding should require to be evolved with only one association set to be able learn any other association set. By contrast, ANNs evolved with a direct encoding should require many more association sets to possess general learning abilities.

Results. When using three to seven assocation sets, most runs using the direct encoding did not manage to evolve ANNs with the perfect fitness (about 10% reached it). We initially thought that this results came from the increased size of the search space (a all-to-all projection between two maps is equivalent to 16 connections); however, we also evolved plastic ANNs with a direct encoding and a fitness that tested each network on all the possible association sets[1]: most runs (4 out of 5) obtained the optimal fitness (and logically, a sGLA score of 1). It therefore appears that the low convergence rates originates more from a deceptive fitness function than from the search space size.

When we analyze the sGLA of ANNs with a perfect fitness, figure 6 confirms that only one association set is sufficient to obtain good sGLA with our generative encoding, whereas 7 association sets were required to obtain similar sGLA with a direct encoding. Nevertheless, these results show that some runs with the generative encoding have low sGLA while they had a perfect fitness. This unexpected variability stems from the random selection performed by the softmax function: when a network outputs sufficiently contrasted values, the softmax function provides a near-deterministic output; however, if the contrast between two outputs is low, there is a similar probability of choosing each

[1] Unfortunately, the experiments with all the association sets were too computationally intensive to obtain enough runs to include them in figure 6.

output; as a consequence, if one of this output is the right one, then there is a significant probability to obtain the right output for each test performed during the fitness evaluation, whereas the network is actually not reliably solving the task. Since this issue is related to the softmax and not to the encoding, a similar problem arose when the direct encoding was used. We are still investigating why the variability of the sGLA scores is higher when a single set is used.

6. CONCLUSION

These experiments empirically show that (1) the difficulties of a generative encoding (in this case, EvoNeuro [14]) with irregular domains can be overcome with synaptic plasticity, and (2) using a generative encoding makes it easier to obtain plastic artificial neural networks that can cope with situations not encountered during the evolution. We employed the EvoNeuro encoding in these particular experiments, but similar properties should be observed with other generative encodings. The newly introduced concept of synaptic Transitive Learning Abilities (sTLA) should help to perform such an analysis.

7. ACKNOWLEDGMENTS

This project was funded by the ANR, project ANR-09-EMER-005-01 and a Monaco Ph.D. scholarship.

8. REFERENCES

[1] L. F. Abbott and S. B. Nelson. Synaptic plasticity: taming the beast. *Nature neuroscience*, 3:1178–83, 2000.

[2] J. Blynel and D. Floreano. Levels of dynamics and adaptive behavior in evolutionary neural controllers. In *Proc. of SAB*, 2002.

[3] D. Chalmers. The evolution of learning: An experiment in genetic connectionism. *Connectionist Models Summer School,*, 1990.

[4] J. Clune, K. O. Stanley, R. T. Pennock, and C. Ofria. On the Performance of Indirect Encoding Across the Continuum of Regularity. *IEEE Transaction On Evolutionary Computation*, 2011.

[5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation*, 6(2):182–197, 2002.

[6] B. Girard, N. Tabareau, A. Berthoz, and J.-j. Slotine. Selective amplification using a contracting model of the basal ganglia. *NeuroComp06*, pages 30–33, 2006.

[7] F. Gruau and D. Whitley. Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect. *Evolutionary computation*, 1(3):213–233, 1993.

[8] G. S. Hornby. Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In *Proc. of GECCO*, 2005.

[9] G. S. Hornby and J. B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246, 2002.

[10] H. Kimura and S. Kobayashi. An Analysis of Actor-Critic Algorithms Using Eligibility Traces: Reinforcement Learning with Imperfect Value Functions. *Journal of Japanese Society for AI*, 15(2):267–275, 2000.

[11] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 2011.

[12] C. Mattiussi and D. Floreano. Analog Genetic Encoding for the Evolution of Circuits and Networks. *Evolutionary Computation*, 11(5):596–607, 2007.

[13] J.-B. Mouret and S. Doncieux. Using behavioral exploration objectives to solve deceptive problems in neuro-evolution. In *Proc. of GECCO*, 2009.

[14] J.-B. Mouret, S. Doncieux, and B. Girard. Importing the computational neuroscience toolbox into neuro-evolution—application to basal ganglia. *Proc. of GECCO*, 2010.

[15] Y. Niv, D. Joel, I. Meilijson, and E. Ruppin. Evolution of Reinforcement Learning in Uncertain Environments: A Simple Explanation for Complex Foraging Behaviors. *Adaptive Behavior*, 10(1):5–24, 2002.

[16] S. Risi and K. O. Stanley. Indirectly Encoding Neural Plasticity as a Pattern of Local Rules. *Proc. of SAB*, 2010.

[17] S. Risi, S. D. Vanderbleek, C. E. Hughes, and K. O. Stanley. How novelty search escapes the deceptive trap of learning to learn. *Proc. of GECCO*, 2009.

[18] A. Soltoggio, J. Bullinaria, C. Mattiussi, P. Dürr, and D. Floreano. Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. *Artificial Life*, 11:569, 2008.

[19] A. Soltoggio, P. Dürr, C. Mattiussi, and D. Floreano. Evolving neuromodulatory topologies for reinforcement learning-like problems. *Proc. of CEC*, 2007.

[20] A. Soltoggio and B. Jones. Novelty of behaviour as a basis for the neuro-evolution of operant reward learning. *Proc. of GECCO*, 2009.

[21] K. O. Stanley, B. D. Bryant, and R. Miikkulainen. Evolving adaptive neural networks with and without adaptive synapses. *Proc. of CEC*, 2003.

[22] K. O. Stanley, D. D'Ambrosio, and J. Gauci. A Hypercube-Based Indirect Encoding for Evolving Large-Scale Neural Networks. *Artificial Life*, 15(2):185–212, 2009.

[23] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[24] R. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. The MIT press, Sept. 1998.

[25] T. P. Trappenberg, M. C. Dorris, D. P. Munoz, and R. M. Klein. A model of saccade initiation based on the competitive integration of exogenous and endogenous signals in the superior colliculus. *Journal of Cognitive Neuroscience*, 13(2):256–271, 2001.

[26] E. Tuci and M. Quinn. Behavioral Plasticity in Autonomous Agents: a Comparison Between Two Types of Controller. *Proc. of Applications of evolutionary computing*, 2003.

[27] J. Urzelai and D. Floreano. Evolution of adaptive synapses: Robots with fast adaptive behavior in new environments. *Evolutionary Computation*, 9(4):495–524, 2001.