LEARNING TO ACQUIRE NOVEL COGNITIVE TASKS WITH EVOLUTION, PLASTICITY AND META-META-LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A hallmark of intelligence is the ability to autonomously learn new flexible, cognitive behaviors - that is, behaviors where the appropriate action depends not just on immediate stimuli (as in simple reflexive stimulus-response associations), but on memorized contextual information. Such cognitive, memory-dependent behaviors are by definition meta-learning ("learning to learn") tasks. Artificial agents can learn a given cognitive task with external, human-designed meta-learning algorithms. By contrast, animals are able to pick up such cognitive tasks automatically, from stimuli and rewards alone, through the operation of their own internal machinery: evolution has endowed animals with the ability to automatically acquire novel cognitive tasks, including tasks never seen during evolution. Can we harness this process to generate artificial agents with such abilities? Here we evolve neural networks, endowed with plastic connections and neuromodulation, over a sizable set of simple meta-learning tasks based on a framework from computational neuroscience. The resulting evolved networks can automatically modify their own connectivity to acquire a novel simple cognitive task, never seen during evolution, from stimuli and rewards alone, through the spontaneous operation of their evolved neural organization and plasticity system. Our results suggest that attending to the multiplicity of loops involved in natural learning may provide useful insight into the emergence of intelligent behavior.

1 INTRODUCTION

1.1 OBJECTIVE

An important feature of intelligent behavior is the ability to learn not just simple, reactive tasks (associating a stimulus with a response), but also more complex, *cognitive* tasks. While "cognition" in general is difficult to define precisely, here we operationally define "cognitive" tasks as those that require storing and manipulating a certain unpredictable piece of information for each new instance of the task - that is, tasks that require working memory. Under this definition, acquiring a new cognitive task is necessarily a case of "learning to learn", or *meta-learning*. Meta-learning, that is, the acquisition of memory-guided tasks, has been studied quantitatively in animals; examples include Harlow's seminal study of memory-guided choice among pairs of objects (Harlow, 1949), or more recently, mice learning to perform memory-guided navigation in virtual reality mazes (Morcos and Harvey, 2016).

Many meta-learning algorithms exist for artificial agents (Thrun and Pratt, 1998; Schmidhuber, 1993a; Hochreiter et al., 2001; Finn et al., 2017; Wang et al., 2016; Duan et al., 2016; Bengio et al., 1991; Floreano and Urzelai, 2000; Ruppin, 2002; Soltoggio et al., 2008; Miconi, 2016; Miconi et al., 2018). However, these hand-designed algorithms meta-train an agent for one specific domain (bandits, maze-solving, foraging, etc.) Here we seek something different. Rather than develop an external algorithm to train an agent for this or that meta-task, we would like to build an agent capable of *automatically* acquiring novel, cognitive (meta-)tasks, including tasks never seen before in the agent's lifetime (or during its initial design), through the operation of its own internal machinery, from stimuli and rewards alone - much like animals in the experiments mentioned above.

We propose to use an evolutionary process to design a self-contained network endowed with plastic connections and reward-modulated plasticity. We expect that this evolved agent, when exposed to many episodes of a novel cognitive task, will automatically refashion its own connectivity in order to adequately extract, store and manipulate task-relevant information during each new episode of the task. Importantly, we do not want the agent to merely be able to perform tasks from a pre-defined fixed set; rather, we want an agent that can automatically *acquire* novel cognitive tasks, including tasks never encountered during evolution.

Training such a system clearly requires a significant number of different meta-learning tasks. We would like to find a framework that could provide a sizable variety of different, simple meta-learning tasks, each reasonably tractable computationally, while at the same time being biologically relevant. We turn to the computational neuroscience literature and use the formalism of Yang et al. (2019), which implements a large number of tasks from the animal neuroscience literature (memory-guided saccades, comparing two successive stimuli, etc.) in a common format. In this paper, due to limited computational resources, we restrict this framework to only use binary stimuli and responses (see Methods).

Crucially, these tasks are actually meta-learning tasks: they require memorizing, storing and manipulating unpredictable items of information to produce adequate responses at each trial. The meta-learning aspect of these tasks is confirmed by noticing that one of these tasks is exactly equivalent to a Harlow task with two objects and two trials per episode (see Appendix E). Furthermore, these tasks also represent a simple, fast type of meta-learning, in which every trial is a separate episode (much like in the Omniglot task, a common meta-learning benchmark). The simple stimuli abstract away the need for perceptual processing, allowing us to concentrate on cognitive operations. The framework can of course be extended to include any task that can be formulated as a series of stimuli-delay-response(s) trials. Thus, this framework combines computational efficiency with biological relevance.

Because we operationally define cognitive tasks as meta-learning task, evolving such a cognitive learner necessarily involves at least three nested learning loops. The multiplicity of nested learning loops in nature, and the potential usefulness of extending typical meta-learning with additional levels, have already been pointed out before (Wang, 2021; Miconi et al., 2019). Our results demonstrate the feasibility of this approach, and also show that carefully considering the multiple levels of learning involved in an experiment may actually have great practical impact on performance (see Discussion).

2 METHODS

2.1 OVERVIEW

Here we first provide a brief summary of the experiment. A complete description is provided in the following subsections. See also Figure 1 for an overview of the entire process. All code is available at [URL withheld for anonymity].

We are evolving a plastic (that is, self-modifying), fully-connected recurrent network to be able to autonomously acquire novel simple meta-learning tasks. The whole experiment is composed of three nested loop. The outermost loop is the Evolutionary loop, which loops over lifetimes; the middle loop, or Task loop, constitutes the lifetime of the agent and loops over trials/episodes of a given task; and the inner loop (the Episode loop) loops over timesteps.

During each trial/episode, the network observes and memorizes two stimuli in succession, produces a response, then receives a reward signal (which depends on the task, stimuli and response for this trial), and applies reward-modulated synaptic plasticity to its own connections. The synaptic plasticity rule is guided by the evolved, innate parameters of the network, namely, its innate ("birth") weights $\mathbf{W}$ and plasticity coefficients $\mathbf{\Pi}$ (as detailed below). The network is exposed to 400 such trials of a given task, constituting the Task loop, during which it is expected that the network will learn the task through the operation of reward-modulated plasticity, as guided by its evolved innate weights and plasticity parameters.

Notice that these two loops (Episode loop and Task loop) are homologous to the "inner loop" and "outer loop" of a typical meta-learning experiment. The difference is that, in typical meta-learning experiment, the network modification occurring in the outer loop is determined by a fixed, hand-

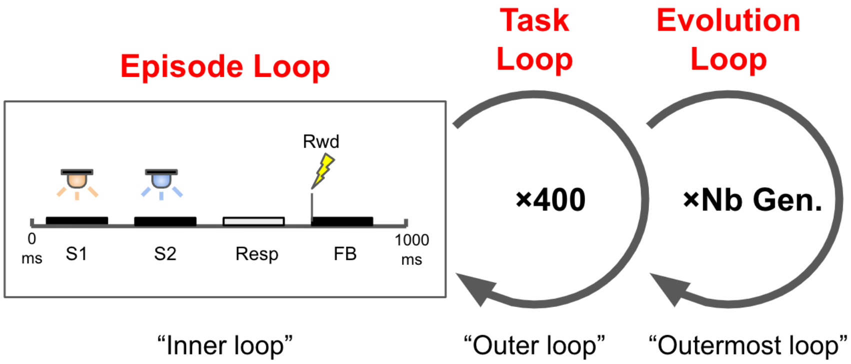

Figure 1: Overall organization of the experiment. During the Episode loop (which loops over time steps), the network stores and processes episodic information (stimuli S1 and S2) in its recurrent activations, provides a response (Resp) and receives a neuromodulatory reward and a feedback signal. In the Task loop, the network is modified by reward-modulated plasticity between episodes, storing task-solving information in plastic connections. These two loops correspond to the "inner loop" and "outer loop" of typical meta-learning experiments. An additional Evolutionary loop then optimizes the innate connectivity and plasticity structure of the network (that is, its genome), ensuring that the neuromodulated plasticity being applied after each trial results in proper lifetime learning.

designed external algorithm. Here however, network modification is determined by an evolved, self-contained plasticity process, guided by the innate structure of the network, which is optimized by evolution in an additional loop. Specifically, the process described above is iterated (with a randomly chosen task from the training set for each pass) over a third, Evolutionary loop, in which evolution optimizes the network's structural parameters (innate weights $\mathbf{W}$ and plasticity parameters $\mathbf{\Pi}$) in order to improve the network's task learning ability (as estimated by the network's performance over the last 100 trials of the Task loop).

At regular intervals, we test the network on a withheld "test" meta-task, never seen during evolution, to assess the network's performance on the overall objective: automatic acquisition of *unseen* meta-learning tasks. In all experiments in the main text, the test task is "Delayed Match to Sample" (DMS: are the two successive stimuli identical?). Results with other test tasks are reported in the Appendix.

2.2 OUTERMOST LOOP: EVOLUTION

The evolved parameters (the "genome" of the agent) consist of a network's baseline ("innate") weights $\mathbf{W}$ and plasticity parameters $\mathbf{\Pi}$, jointly denoted as $\theta = \{\mathbf{W}, \mathbf{\Pi}\}$. $\mathbf{W}$ represents the initial weights at birth for the network, and $\mathbf{\Pi}$ represents the amount of plasticity at each connection. ($\mathbf{\Pi}$ is equivalent to α in Miconi (2016); Miconi et al. (2018)).

The overall evolutionary algorithm is an evolution strategy similar to Salimans et al. (2017): at each generation, the current candidate network θ_0 is copied and randomly mutated many times, resulting in a population of mutated copies $\theta_i = \theta_0 + \sigma_i$ where each σ_i is a Gaussian vector of the same size as θ; each of these mutated copies is evaluated, as described below, returning a loss L_i; and the sum of all mutation vectors, weighted by their losses, is used as a gradient estimate $\hat{\nabla}_\theta L = \sum_i L_i \sigma_i$. As in Salimans et al. (2017), we use antithetic sampling, which means that for every mutation vector σ_i, we also include the opposite mutation vector $-\sigma_i$ in the population (this considerably improves performance). We pass this gradient estimate $\hat{\nabla}_\theta L$ to the Adam optimizer (Kingma and Ba, 2015) to produce actual parameter changes for θ, resulting in a new θ_0 for the next generation. We iterate this process for a fixed number of generations.

2.3 MIDDLE LOOP: LIFETIMES/TASKS

We evaluate each network by exposing it to a block of 400 trials of a given task. The task is chosen at random for every generation and is the same for the whole population, but the stimuli being presented during each trial are independently sampled for each individual at each trial (except that antithetic pairs receive the same sequence of stimuli). At the start of each block, a network is initialized by having its plastic weights (see below) and neural activities set to zero; the initial weights and plasticity parameters are taken from the mutated parameters θ_i. Plastic weights, unlike innate weights, are updated over the whole block, without reinitialization, according to the plasticity rule described below.

We stress that this loop corresponds to the *outer loop* of a typical meta-learning experiment, during which the specific meta-task at hand (that is, the appropriate way to store and process task-relevant information during each inner-loop episode) is slowly learned across many episodes.

2.4 INNER LOOP: TRIALS/EPISODES

During each trial, the network observes two stimuli in succession. These stimuli are fed to the network, one after the other, by clamping a subset of neurons (the "input" neurons) to specific values for a fixed amount of time. Then, after a short delay, the network's response is recorded. This response is the output of another subset of neurons (the "output" neurons), accumulated over a period of time. Then, the network receives a binary signal, indicating whether its response was correct or not; again, this is provided by clamping a different neuron to the corresponding value for a period of time. The network's error, determined by comparing the accumulated response over time with the expected response for this trial (in accordance with the specific task), is also used as a neuromodulatory signal to perform synaptic plasticity updates at this time, according to the plasticity rule described below. This process is iterated over a fixed number of trials, constituting a block. Then, the mean error over the last 100 trials of both lifetime blocks is used as the loss L_i for each individual for this generation, guiding the evolutionary process as described above.

2.5 NETWORK OPERATION AND PLASTICITY

Lifetime plasticity is implemented by the node-perturbation rule(Fiete et al., 2007), a reward-modulated Hebbian rule that models the influence of dopamine on plasticity in the brain. Conceptually, node perturbation consists in occasionally and randomly perturbing the activation of neurons, then applying a reward-modulated Hebbian update to each synaptic weight, consisting of the product of the perturbation, the input at that synapse at the time of the perturbation, and the reward: $\Delta w_{i\to j}(t) \propto y_i(t)\Delta y_j(t)R$. The crucial element is that the "output" factor in this Hebbian rule is not the neuron's full output $y_j(t)$, but the (random) perturbation $\Delta y_j(t)$.

We chose this rule because it can be implemented in a reasonably biologically plausible manner (Miconi, 2017), yet comes with strong theoretical guarantees, since it is largely equivalent to the REINFORCE algorithm (Williams, 1992) (the algorithm in section 5 of Williams (1992) is a node perturbation algorithm for stochastic spiking neurons). Furthermore, because it is a (reward-modulated) Hebbian rule, it fits easily with the differentiable-plasticity approach that we use in the evolutionary loop.

More precisely, at any time, the fully-connected recurrent network acts according to the following equations:

$$\tau \frac{d\mathbf{x}(t)}{dt} = (\mathbf{W} + \mathbf{\Pi} \odot \mathbf{P}(t))\mathbf{r}(t) - \mathbf{x}(t) \quad (1)$$

$$\mathbf{r}(t) = f(\mathbf{x}(t)) \quad (2)$$

Here $\mathbf{x}$ is the vector of neural activations (the linear product of inputs by weights), $\mathbf{r}$ is the neural responses (activations passed through a nonlinearity), $\mathbf{W}$ and $\mathbf{\Pi}$ are the innate weights and plasticity parameters (fixed and unchanging during a lifetime and optimized across lifetimes by evolution, as mentioned above), $\mathbf{P}$ is the *plastic* weights (changing over a lifetime according to the plasticity rule described below), τ is the network time constant, f is a nonlinear function and $\odot$ represents the pointwise (Hadamard) product of two matrices. Note that these equations are just the standard continuous-time recurrent neural network equations, except that the total weights are the sum of innate weights and plastic weights multiplied by the plasticity parameters.

Random perturbations $\Delta\mathbf{x}(t)$ are occasionally applied to the activations $\mathbf{x}$. Furthermore, each connection maintains a so-called Hebbian eligibility trace $\mathbf{H}(t)$, which is a running decaying average of the product of modulations by inputs:

$$\tau_H \frac{d\mathbf{H}(t)}{dt} = \Delta\mathbf{x}(t)\mathbf{r}(t)^{\mathsf{T}} - \mathbf{H}(t) \tag{3}$$

Here τ_H is the time constant of the eligibility trace, which is significantly longer than the neural time constant τ. Finally, whenever a reward signal R is applied (once per trial, after each response; see previous section), the current value of the Hebbian trace is multiplied by R and added to the plastic weights $\mathbf{P}$:

$$\mathbf{P}(t) \leftarrow \mathbf{P}(t) + \eta R\mathbf{H}(t) \tag{4}$$

Here η is the lifetime plasticity rate. We reiterate that $\mathbf{P}(t)$ is initialized to 0 at the beginning of each block of trials ("lifetime") and changes according to the above equations (without any reinitialization) during a whole block.

2.6 TASKS

In this paper, we simplify the framework of Yang et al. (2019) by only using binary stimuli and responses. The set of all possible tasks is the set of all 16 mappings from two successive binary stimuli to a binary response. Each trial consists of two binary stimuli shown in succession, each for a period of time; a response period, during which the network's response is recorded; and a period during which a binary signal is provided to indicate whether the response was correct. There are two output neurons, representing responses '0' and '1' respectively; the response of the network, for each trial, is determined by which of the two output neuron had the largest accumulated response over the response period. The error signal used for neuromodulation (as $-R$ in the previous section) is the mean square of the difference between the responses of the two output neurons and their target for this trial, over the response period.

One task is withheld as the "test" task, which is never seen during evolution, and on which the evolved network is periodically tested to assess performance in the overall objective, that is, ability to acquire a *novel* meta-learning task. In the main text, our main withheld test task is the delayed-match-to-sample task (DMS, i.e. "are the two successive stimuli identical or different?"). Results with different withheld test tasks are shown in the Appendix. We chose the DMS task because it is the most difficult task to acquire in the set (see Appendix), and also because it is actually used in animal studies. Note that the DMS task requires a strongly nonlinear processing of the two successive stimuli (being the negation of the exclusive OR). Thus, DMS is a simple, but not trivial task, with some degree of biological relevance.

Importantly, we also remove the logical negation of the test task from the training set (for DMS, that is the "delayed non-match to sample" task), since the response for the logical negation of a task is simply the mirror image of the test task. This is to ensure that the networks are truly able to learn the structure of the task at test time, rather than acquiring it through evolution and merely adapting the sign of the response at test time. This results in a training set of 14 different tasks for all experiments.

2.7 QUANTITATIVE DETAILS AND TERMINOLOGY

Each generation is a batch of 1000 individuals. Each block is composed of 400 trials, each of which lasts 1000 ms. Following Yang et al. (2019), we use $\tau = 100$ ms and simulation timesteps of 20 ms. Perturbations occur independently for each neuron with a probability of 0.1 at each timestep; perturbations are uniformly distributed in the $[-0.5, 0.5]$ range. We set $\tau_H = 1000$ ms, $\eta = 0.03$. At generation 0, $\mathbf{W}$ is initialized with Gaussian weights with mean 0 and standard deviation $1.5/\sqrt{N}$, where $N = 70$ is the number of neurons in the network (this distribution ensures self-sustaining spontaneous dynamics in the initialized network (Sompolinsky et al., 1988)), while all values of $\mathbf{\Pi}$ are initialized to 0.5. Evolution runs over 1000 generations for DMS. We feed evolutionary gradients

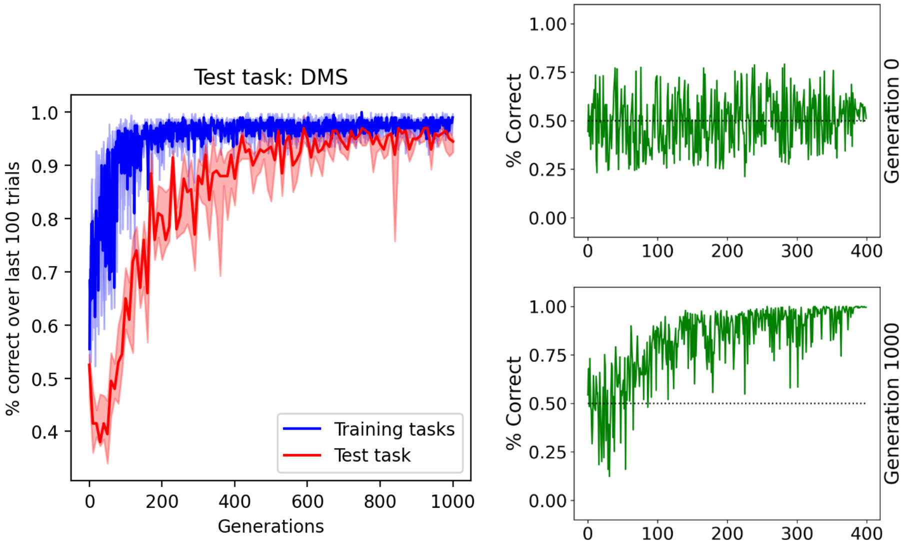

Figure 2: Left: Median (across all runs) proportion of correct trials, over the last 100 trials of each generation, shown separately for training tasks (blue, solid) and withheld task (red, dashed). Filled areas indicate inter-quartile range across runs. Right: Proportion (across a batch of 500 individuals) of correct trials over the course of a full block of the withheld task, shown separately at Generation 0 (i.e. randomly initialized network with uniform plasticity, top) and Generation 1000 (after evolving innate weights and plasticity of each connection, bottom). Dotted line: 50% chance level. The evolved network successfully acquires the task over many trials/episodes. We stress that the 400 episodes shown here are equivalent to the *outer loop* of a typical meta-learning experiment.

to the Adam optimizer, with a learning rate of 0.003. The whole experiment completes in about 10 hours on a single GPU accessed through Google Colab.

Regarding terminology: we use the words “task”, “meta-task” and “cognitive task” interchangeably. An individual, or agent, means an individual network within the batch (“batch” and “population” are used interchangeably). Due to the nature of our tasks, we also use “trial” and “episode” interchangeably: each trial/episode loop consists of the successive presentation of two stimuli, the network’s response, and the feedback returned to the network, as shown in the “Episode Loop” section of Figure 1. Each agent ’s lifetime thus consists of one block of 400 such trials/episodes of one randomly chosen task from the set (in the present design, each lifetime only contains one task).

3 RESULTS

3.1 PERFORMANCE

To assess meta-meta-learning progress, we track the evolutionary loss (that is, the total lifetime loss over the last 100 trials of each successive lifetime) as a function of the number of generations. Every 10 generations, we test the current candidate genotype θ_0 on a withheld task, which does not result in any weight modification. We report both training loss (on the tasks used for evolutionary training) and test loss (on the withheld task), over 6 runs with different random seeds.

The main result is that the system successfully evolves an architecture that can automatically acquire a novel, unseen meta-learning task, as seen by increasing withheld-task performance over generations in Figure 2 (left panel, red dashed curve). Unsurprisingly, initial performance at generation 0 (with random $\mathbf{W}$ and uniform $\mathbf{\Pi}$) is poor. This indicates that, despite the fact that all connections undergo reward-modulated synaptic plasticity, this lifetime plasticity alone does not suffice to acquire the task in a randomly initialized network, as expected. This can also be observed by tracking the lifetime performance (that is, the correctness of the response for each successive trial over a block) of the initial genotype at generation 0, which indicates little lifetime learning (Figure 2, right panel, top).

By contrast, after evolution has sculpted the innate parameters of the network, the candidate genotype now supports competent lifetime acquisition of the unseen meta-learning task. Again, this is confirmed by observing the lifetime performance of the evolved candidate, which indicates successful lifetime learning based on incoming correctness signals and synaptic plasticity (Figure 2, right panel, bottom).

3.2 EVOLUTIONARY OUTCOMES

Importantly, the increase in performance is not simply due to a mere indiscriminate increase in overall plasticity. In a representative run, values in $\mathbf{\Pi}$, starting from a uniform value of 0.5, evolved to a final range of 0.0 to 1.83 (median 0.47). Evolved values of $\mathbf{W}$ cover a larger range than their initialization values: in a representative run, weights went from an initial range of $[-0.71, 0.65]$ (median -0.00, median absolute value 0.11) to a range of $[-1.60, 1.77]$ (median 0.00, median absolute value 0.30). The learned plastic weights $\mathbf{P}$ after lifetime learning (in a fully evolved individual) cover a smaller range than innate weights: at the end of the last block of a representative run (on the withheld task), values in $\mathbf{P}$ ranged from -0.11 to 0.11 (median 0.00).

We did not detect any obvious structure in the evolved weights or plasticity parameters, or in the learned plastic weights (Figure A4). They do not seem to show specially different values for output or input neurons (whether on input or output connections), and the matrix of plasticity components shows no vertical, horizontal or diagonal structure. Evolved plasticity components look largely random besides the relatively constant overall level.

3.3 IMPORTANCE OF PLASTICITY

Because the network receives error signals, and because recurrent networks are universal approximators, it is theoretically possible that a non-plastic network might learn to solve the problems solely through its fixed weights and activities, as shown in typical meta-learning studies based on optimizing recurrent networks Wang et al. (2016); Duan et al. (2016). To test this possibility, we ran the exact same experiment, but disabling plasticity (i.e. removing the $\mathbf{\Pi} \odot \mathbf{P}(t)$ term in Equation 1). As expected given the small size of the network, the large number of tasks, and the multi-level nature of the experiment, this two-level system fails to acquire either the training tasks or the withheld task (Figure A2). This confirms that all three learning systems (evolution, plasticity and recurrence) are necessary for this model to succeed, and the importance of considering all loops of learning that occur in a given experiments (see Appendix C).

3.4 VISUALIZING EVOLVED AND LEARNED REPRESENTATIONS WITH CROSS-TEMPORAL DECODING

How does the evolved network perform its task? To investigate how evolved networks process information during each trial, we use cross-temporal decoding, a standard method in computational neuroscience Meyers et al. (2008); King and Dehaene (2014); Stokes et al. (2013); Miconi (2017). For every pair of instants t_1, t_2 within a trial (1000 ms), we try to decode task-relevant information from neural activities at time t_2, based on a decoder trained on neural activities from time t_1. This allows us to determine whether the network encodes this information at any given time (by evaluating decoding performance when $t_1 = t_2$), but also to estimate whether the network uses a stable encoding, or a dynamic, time-varying representation of task information: if the same decoder trained on neural data from time t_1 can successfully decode task information from neural data at time t_2, this implies a similar encoding of this information at both time points (see Appendix D for implementation details). We perform this analysis both for at generation 0 and generation 1000 (demonstrating the effects of evolution), as well as at the first and last trial of each generation (demonstrating the effects of lifetime learning). In Figure 3, we show decoding performance for target response (i.e. whether the two stimuli are identical or not), using data from the same run described in Section 3.2. We see that generation-0 networks do not reliably encode this information at any time during the trial. By contrast, after evolution (generation 1000), the network robustly encodes this information, even though it was not part of its evolutionary training (high decoding performance along the diagonal, $t_1 = t_2$). Furthermore, comparing first-trial and last-trial data, we observe a stabilization of target response representation around the time of the response period (dotted blue lines): the region of high performance bulges out into a squarish shape at this point. Interestingly, we observe quite different dynamics when trying to decode stimulus identity rather than target response (see Appendix D).

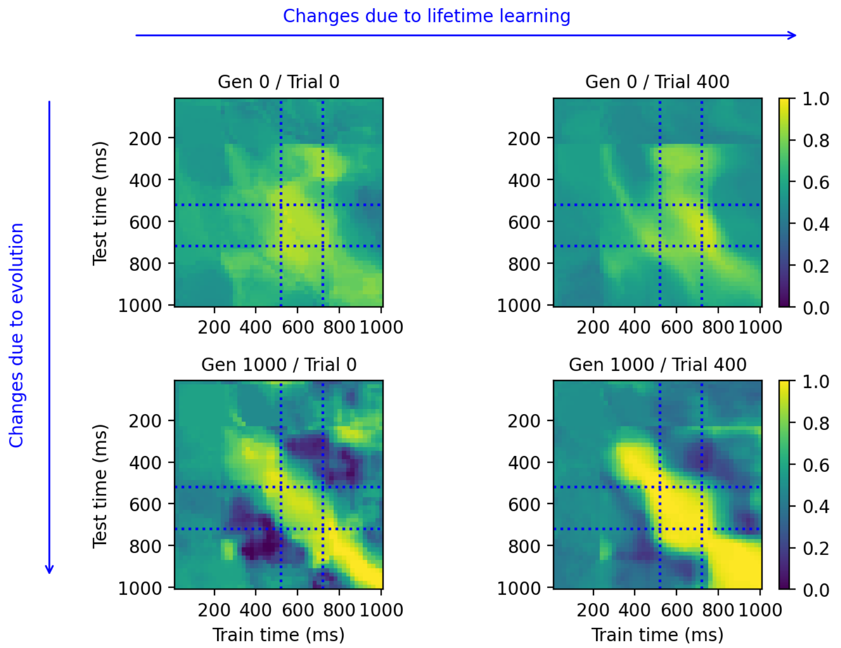

Figure 3: Cross-temporal decoding of target response for the (withheld) DMS task. Each matrix represents a full trial, at generations 0 and 1000 (top and bottom), and first and last trial of each generation (left and right). Dotted lines represent the response period, i.e. the 500-700ms interval. Each data point x, y represents decoding performance at time y, using a decoder trained on neural activities from time x. The network does not reliably encode target response ("are the stimuli equal?") at generation 0, but does after evolution; furthermore, lifetime learning leads to a broadening of stable representations around response time (bottom-right plot). See also Appendix D for similar plots for decoding identity of first and second stimulus.

4 DISCUSSION

4.1 A MULTI-LOOP MODEL FOR THE EMERGENCE OF AUTONOMOUS COGNITIVE LEARNING

In this work we sought to evolve a self-contained learning system, capable of acquiring novel simple cognitive tasks from mere exposure to stimuli and rewards. Because we operationally restrict our definition of "cognitive" tasks as those that require working memory, acquiring such a task is necessarily a meta-learning problem, and thus evolving or otherwise optimizing an agent with this capacity necessarily involves at least three learning loops. We showed that existing methods could be combined to successfully design a self-contained agent capable of acquiring novel cognitive tasks.

We emphasize that the tasks used here, despite their simplicity, are truly meta-learning tasks: new items of information must be acquired, stored and exploited appropriately at each episode. As discussed in Appendix E, the DMS task used above is actually exactly equivalent to a Harlow task with two trials per episode. Simple extensions would allow this framework to include the actual Omniglot tasks (by allowing more than two stimuli per episode and making the stimuli multi-dimensional) and reinforcement learning tasks like bandit tasks or maze-solving (by allowing more than one response per trial/episode). Thus, despite its simplicity, the formalism can potentially encompass all forms of meta-learning, including meta-supervised learning (Hochreiter et al., 2001) and meta-reinforcement learning (Wang et al., 2016; Duan et al., 2016).

Our model relies on multiple nested loops of learning, with each loop designing the learning algorithm of the next one, extending the standard two-loop model of meta-learning. The multiplicity of learning loops in nature has been observed before. Wang (2021) pointed out the existence of "multiple nested scales of learning in nature", with each building priors to constrain (and thus accelerate) learning at the level below (a generalization of the Baldwin effect Hinton et al. (1987)). Miconi et al. (2019)

proposed that an additional level of “meta-meta-learning” could learn the weight-modifying algorithm in the outer loop of a meta-learning system. Our results demonstrate the feasibility of this process.

Importantly, our results show that the concept of multiple learning loops is not merely academic, but has practical importance: paying attention to the multiple levels of learning in an experiment can greatly impact performance, and be the difference between success and failure (Figure 2 vs Figure A2). These results emphasize the importance of thinking carefully and systematically about the multiple loops and levels of learning, in nature or artificial experiments (see Appendix G).

Although our initial results confirm the basic validity of the approach, it is clear that the present experiment is essentially a starting point. Building upon this platform, several avenues of investigation that may greatly enhance the abilities of the system immediately suggest themselves (see Appendix F). One important avenue of future research is the incorporation of an additional learning loop that is currently not included in the model, namely the *lifetime experience* loop: In the current form of the model, each lifetime is devoted to one single task. This is of course unrealistic. In reality, animals acquire a considerable amount of knowledge from their lifetime experience, accumulated across many different tasks, which greatly improves their performance in mature adulthood. In other words, the model described here deliberately excludes an additional learning loop, separate from the three loops considered here: the lifetime experience loop. This loop would be crucial in studying the emergence of mechanisms that support robust *continual learning* over a lifetime (forward and backward transfer, robustness against catastrophic forgetting, etc.). See Appendix F.4 for a discussion of this point.

4.2 The “Bitter Lesson”, AI-GAs, inateness, and the many loops of learning

Why would one want to evolve a self-contained cognitive learner, rather than simply apply an off-the-shelf meta-learning algorithm to any new cognitive task? First, for the same reason that we use meta-learning in the first place: the meta-meta-learning process may extract common structure in the environment and generic concepts useful across cognitive tasks, facilitating individual task learning (Wang, 2021). This may be evidenced here by the remarkably fast learning of new cognitive tasks (reaching high performance within $\tilde{2}$00 episodes), as shown in Figure 2, when compared to the many thousands of episodes used in the outer loop of typical meta-learning experiments; however, the simplified stimuli (which remove the need for perceptual processing) certainly play a role in this efficiency.

More generally, Sutton (2019) has pointed out the “bitter lesson” that approaches that leverage learning and mass computation have consistently overtaken approaches based on hand-designed features and expert knowledge. Clune (2019) pushes this argument further by proposing that the learning systems themselves (both their architecture and their algorithms) should be learned rather than designed, resulting in so-called “AI-Generating Algorithms”. The present model applies these principles further by putting as much as possible of the system under the control of optimization rather than human design.

In the other direction, Marcus and Davis (2019) have emphasized the importance of innate structure and “common sense” knowledge in the performance of human intelligence. The present model exemplifies that when the multiplicity of learning levels and loops is duly considered, the two aspects may become more complementary: one level’s innateness is another level’s learning, and one loop’s structural knowledge is another loop’s slowly acquired information (as pointed out by (Wang, 2021)).

In addition to these utilitarian concerns, we suggest that the emergence of an agent capable of autonomously acquiring novel simple *cognitive* tasks, through its own internal machinery, is of interest in and by itself. This represents a step toward the goal of “banishing the homunculus” (in the expression of Hazy et al. (2006)), that is, to eliminate the need for an artificial, human-designed external “executive”, and endow the system with truly autonomous learning. Obviously this goal is still some way off: as mentioned above, the current system still includes many design restrictions. Nevertheless, we believe the success of this experiment demonstrates the potential of the approach.

References

Harry F. Harlow. The formation of learning sets. *Psychol. Rev.*, 56(1):51–65, January 1949.

Ari S Morcos and Christopher D Harvey. History-dependent variability in population dynamics during evidence accumulation in cortex. *Nature neuroscience*, 19(12):1672–1681, 2016.

Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. In Sebastian Thrun and Lorien Pratt, editors, *Learning to Learn*, pages 3–17. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

Jürgen Schmidhuber. Reducing the ratio between learning complexity and number of time varying variables in fully recurrent nets. In *ICANN '93: Proceedings of the International Conference on Artificial Neural Networks*, pages 460–463. 1993a.

Sepp Hochreiter, A Younger, and Peter Conwell. Learning to learn using gradient descent. *Artificial Neural Networks—ICANN 2001*, pages 87–94, 2001.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv e-prints*, art. arXiv:1611.05763, November 2016.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. 2016. URL `http://arxiv.org/abs/1611.02779`.

Yoshua Bengio, Samy Bengio, and Jocelyn Cloutier. Learning a synaptic learning rule. In *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on*, volume 2, pages 969–vol. IEEE, 1991.

Dario Floreano and Joseba Urzelai. Evolutionary robots with on-line self-organization and behavioral fitness. *Neural Networks*, 13(4-5):431–443, 2000.

Eytan Ruppin. Evolutionary autonomous agents: A neuroscience perspective. *Nature Reviews Neuroscience*, 3(2):132–141, 2002.

Andrea Soltoggio, John A Bullinaria, Claudio Mattiussi, Peter Dürr, and Dario Floreano. Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In *Proceedings of the 11th international conference on artificial life (Alife XI)*, number LIS-CONF-2008-012, pages 569–576. MIT Press, 2008.

Thomas Miconi. Backpropagation of hebbian plasticity for continual learning. In *NIPS Workshop on Continual Learning*, 2016.

Thomas Miconi, Jeff Clune, and Kenneth O. Stanley. Differentiable plasticity: training plastic networks with gradient descent. In *Proceedings of the 35th International Conference on Machine Learning*, 2018. URL `https://arxiv.org/abs/1804.02464`.

Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.

Jane X Wang. Meta-learning in natural and artificial intelligence. *Current Opinion in Behavioral Sciences*, 38:90–95, 2021.

Thomas Miconi, Aditya Rawal, Jeff Clune, and Kenneth O Stanley. Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity. In *International Conference on Learning Representations*, 2019.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*. 2015. URL `https://arxiv.org/abs/1412.6980`.

Ila R Fiete, Michale S Fee, and H Sebastian Seung. Model of birdsong learning based on gradient estimation by dynamic perturbation of neural conductances. *J. Neurophysiol.*, 98(4):2038–2057, October 2007.

Thomas Miconi. Biologically plausible learning in recurrent neural networks reproduces neural dynamics observed during cognitive tasks. *Elife*, 6, February 2017.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3-4):229–256, 1992.

Haim Sompolinsky, Andrea Crisanti, and Hans-Jurgen Sommers. Chaos in random neural networks. *Physical review letters*, 61(3):259, 1988.

Ethan M Meyers, David J Freedman, Gabriel Kreiman, Earl K Miller, and Tomaso Poggio. Dynamic population coding of category information in inferior temporal and prefrontal cortex. *Journal of neurophysiology*, 100(3):1407–1419, 2008.

Jean-Rémi King and Stanislas Dehaene. Characterizing the dynamics of mental representations: the temporal generalization method. *Trends in cognitive sciences*, 18(4):203–210, 2014.

Mark G Stokes, Makoto Kusunoki, Natasha Sigala, Hamed Nili, David Gaffan, and John Duncan. Dynamic coding for cognitive control in prefrontal cortex. *Neuron*, 78(2):364–375, 2013.

Geoffrey E Hinton, Steven J Nowlan, et al. How learning can guide evolution. *Complex systems*, 1 (3):495–502, 1987.

Rich Sutton. The bitter lesson. 2019. URL `http://www.incompleteideas.net/IncIdeas/BitterLesson.html`.

Jeff Clune. AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence. *arXiv preprint arXiv:1905.10985*, 2019.

Gary Marcus and Ernest Davis. *Rebooting AI: Building artificial intelligence we can trust*. Vintage, 2019.

Thomas E Hazy, Michael J Frank, and Randall C O'Reilly. Banishing the homunculus: making working memory work. *Neuroscience*, 139(1):105–118, 2006.

Jane X Wang, Zeb Kurth-Nelson, Dharshan Kumaran, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Demis Hassabis, and Matthew Botvinick. Prefrontal cortex as a meta-reinforcement learning system. *Nature neuroscience*, 21(6):860, 2018.

D Ackley and M Littman. Interactions between learning and evolution. In *Artificial Life II*. Addison-Wesley, 1991.

Andrea Soltoggio and Jochen J Steil. Solving the distal reward problem with rare correlations. *Neural Comput.*, 25(4):940–978, April 2013.

Andrea Soltoggio, Kenneth O. Stanley, and Sebastian Risi. Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks. 2017. URL `http://arxiv.org/abs/1703.10371`.

James M. Baldwin. A new factor in evolution. *American Naturalist*, 30:441–451, 1896.

Conrad H Waddington. Canalization of development and the inheritance of acquired characters. *Nature*, 150(3811):563–565, 1942.

Jürgen Schmidhuber. A neural network that embeds its own meta-levels. In *IEEE International Conference on Neural Networks*, pages 407–412. IEEE, 1993b.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Functional requirements for reward-modulated spike-timing-dependent plasticity. *J. Neurosci.*, 30(40):13326–13337, October 2010.

Bence P Ölveczky, Aaron S Andalman, and Michale S Fee. Vocal experimentation in the juvenile songbird requires a basal ganglia circuit. *PLoS biology*, 3(5):e153, 2005.

Xingyou Song, Wenbo Gao, Yuxiang Yang, Krzysztof Choromanski, Aldo Pacchiano, and Yunhao Tang. Es-maml: Simple hessian-free meta learning. *arXiv preprint arXiv:1910.01215*, 2019.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. October 2014.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack Rae, Piotr Mirowski, Joel Z Leibo, Adam Santoro, et al. Unsupervised predictive memory in a goal-directed agent. *arXiv:1803.10760*, 2018.

A EXTENDED RELATED WORK

A.1 TWO FORMS OF META-LEARNING

As mentioned in the introduction, many meta-learning algorithms exist(Thrun and Pratt, 1998; Schmidhuber, 1993a; Hochreiter et al., 2001; Finn et al., 2017; Wang et al., 2016; Duan et al., 2016; Bengio et al., 1991; Floreano and Urzelai, 2000; Ruppin, 2002; Soltoggio et al., 2008; Miconi, 2016; Miconi et al., 2018). Interestingly, many (but not all) of these algorithms fall within one of two broad categories, supported by very different interpretations:

1. Algorithms where the inner-loop stores episodic information in the time-varying neural *activities* of a recurrent network, while the outer loop slowly modified in order to optimize within-episode learning (Hochreiter et al., 2001; Wang et al., 2016; Duan et al., 2016). A biological interpretation of this method is that the inner loop represents the within-episode self-sustaining activity of cerebral cortex, while the outer loop represents lifetime sculpting of neural connections by reward-modulated synaptic plasticity, as it occurs in the brain under the effect of neuromodulators such as dopamine (this interpretation is made explicit by Wang et al. (2018)).
2. Algorithms where the inner-loop stores episodic information in the synaptic *connections* of the network, through some kind of parameterized synaptic plasticity algorithm, while the outer loop optimizes the innate structure of the network and/or the parameters of synaptic plasticity(Schmidhuber, 1993a; Bengio et al., 1991; Floreano and Urzelai, 2000; Ruppin, 2002; Soltoggio et al., 2008; Miconi, 2016; Miconi et al., 2018). In a biological interpretation, the inner loop represents lifetime learning by the brain's synaptic plasticity, while the outer loop represents the *evolutionary* design of this built-in synaptic learning system over many lifetimes (even though the actual outer loop algorithm may not be evolutionary (Schmidhuber, 1993a; Miconi et al., 2018)).

Each of these methods can be applied to acquire one single meta-learning task, after which the agent can automatically perform new instances (episodes) of this same task. However, note that these two types of methods have a clear point of junction: the outer loop of the former corresponds to the inner loop of the former (synaptic plasticity). This suggests a combined overall method, in which an outermost "evolutionary" process would design a self-contained, reward-based plastic network over many lifetimes; in turn, this evolved plasticity machinery, guided by within-episode rewards, would automatically sculpt the network's own connections over many episodes of any given memory-guided task (including new such tasks, never seen before), and thus refashion the network's connectivity to be able to extract, store and manipulate task-relevant information from the environment during each episode. The overall result would be the evolution of an agent capable of automatically acquiring novel meta-learning tasks, which we have argued here covers an important type of "cognitive" tasks. This is the purpose of the present paper.

A.2 INTERACTIONS OF EVOLUTION AND LIFETIME LEARNING

Evolving agents capable of lifelong learning has a long history (Ackley and Littman, 1991; Hinton et al., 1987; Floreano and Urzelai, 2000; Soltoggio et al., 2008; Soltoggio and Steil, 2013) (see Soltoggio et al. (2017) for a review). The interaction of evolution and lifetime learning gives rise to

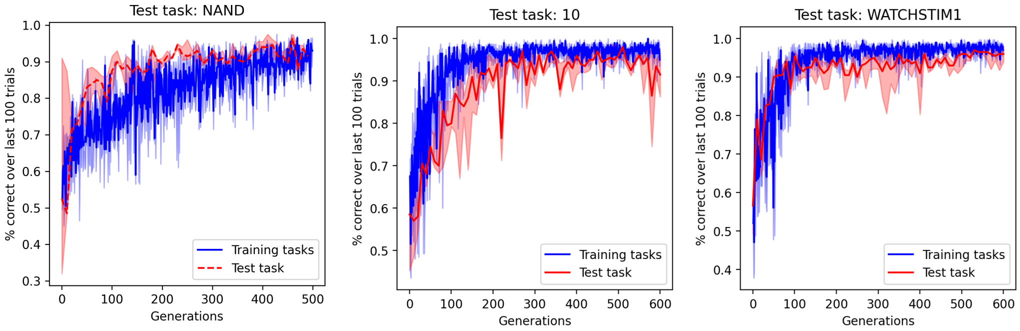

Figure A1: Performance over the course of evolution for different test tasks. Conventions are as in Figure 2, left panel.

complex dynamics. One example is the Baldwin effect (Baldwin, 1896; Hinton et al., 1987), under which learning can guide evolution: beneficial features that are initially learned during each lifetime become increasingly incorporated into the genome, as natural selection favors individuals "born closer" to the eventual, beneficial phenotype[1]. This enhances the reach of evolution, allowing it to produce phenotypes that it would have been unlikely to hit upon alone (Hinton et al., 1987). It also makes lifetime learning faster and more reliable - but also more constrained and less flexible (i.e. "canalized" (Waddington, 1942)).

A.3 MULTIPLE LOOPS OF LEARNING

Meta-learning, or "learning to learn", generally involves two nested learning loops: an "inner loop" which stores and process relevant episodic information during an episode of the task, and an "outer loop" which optimizes inner-loop learning and processing over may episodes. Several authors have proposed extending this hierarchy. Schmidhuber (1993b) (also in many later works) suggested that a recurrent network capable of outputting its own weight changes could in principle "change not only themselves but also the way they change themselves, and the way they change the way they change themselves, etc.". Miconi et al. (2019) suggested that meta-learning by weight optimization of recurrent networks (Hochreiter and Schmidhuber, 1997; Wang et al., 2016; Duan et al., 2016) could be extended with an additional level that would design the weight-modifying algorithm itself (rather than using a standard RL algorithm), pointing out the evolutionary design of humans (who are capable of meta-learning) as a real-life example of such"meta-meta-learning". Wang (2021) explicitly describes three nested "learning loops" in natural learning (evolution, meta-tasks, and specific tasks), with higher loops each providing structure and priors to optimize learning in the lower ones.

Here we posit that such a meta-meta-learning process is the natural way to design an agent capable of autonomously learning novel cognitive tasks. Using the framework of (Yang et al., 2019) as a source of multiple computationally efficient simple meta-tasks with some biological relevance, we successfully demonstrate this process. Furthermore, we show that the concept of multiple learning loops is not merely academic, but has real practical importance, as failing to account for the multiplicity of learning levels prevents successful optimization (see Figure A2, Section A.3 and Appendix C).

B RESULTS WITH DIFFERENT TEST TASKS

In Figure A1, we show results when using different tasks as the withheld test task. We report results with the NAND task, the "report first stimulus" task (respond with the value of the first stimulus, which requires memorizing this first stimulus in a way that is robust to the distractor second stimulus),

---

[1] In the words of Baldwin (1896): "This principle secures by survival certain lines of determinate phylogenetic [i.e. evolved] variation in the directions of the determinate ontogenetic [i.e. learned] adaptations of the earlier generation . . . So there is continual phylogenetic progress in the directions set by ontogenetic adaptation . . . This complete disposes of the Lamarkian [*sic*] factor".

and the “1-0” task (only respond positively if the two successive stimuli are exactly 1 and 0). In all cases, as above, we also withhold the logical negation of the test task.

All of these tasks turn out to be much easier to acquire (as withheld test tasks) than the DMS task, with test performance occasionally higher than for the training set tasks (including DMS) that were used to guide evolution . This difference may reflect the inherent difficulty of the DMS task, which requires a non-linearly separable integration between the two successive stimuli.

C RESULTS WITH DISABLED PLASTICITY: THE IMPORTANCE OF MULTIPLE LEARNING SYSTEMS

In Figure A2, we show the result of running the exact same experiment, but with disabled lifetime plasticity (removing the $\mathbf{\Pi} \odot \mathbf{P}(t)$ term in Equation 1). In addition, we use an easier-to-acquire test task (logical NAND between the two successive stimuli).

Note that this control experiment only involves two levels of learning: evolution in the outer loop, and RNN operation in an undifferentiated inner loop over many trials. As such, it is exactly equivalent to standard, two-loop meta-learning methods based on optimizing recurrent networks, such as RL^2 (Duan et al., 2016) or Learning to Reinforcement Learn (Wang et al., 2016). Note that we use evolution in the outer loop instead of gradient descent, because it is not feasible to backpropagate gradients through twenty thousand time steps at each pass.

As expected, this two-loop system fails to acquire either the training tasks or the withheld task (Figure A2). This is in contrast with the success of our three-systems model, as shown in Figure 2. This confirms that all three learning systems (evolution, plasticity and recurrence) are necessary for this model to succeed.

Our hypothesis is that this negative result results from a mismatch between the two levels of learning in the system, and the three levels of learning required by the experiment. Since our “cognitive” tasks (tasks that require memorizing and manipulating information) are meta-learning task, learning an agent capable of acquiring such tasks necessarily incvlves three levels of learning (see Figure A5). By trying to shoehorn this process on only two learning levels (evolution and RNN activations), we are asking the recurrent networks to solve two levels of learning by its operation: the within-episode storage and processing of episodic data, and the between-episodes learning of the actual task at hand through rewards. Evidently this is beyond the capacities of such simple networks. By contrast, by acknowledging the three learning loops and introducing an additional learning system n the hierarchy (the plasticity system, which is optimized by evolution between lifetimes, and in turn optimizes the RNN between episodes), the same networks now become capable of acquiring these tasks. This emphasizes the practical importance of considering the multiple loops of learning that occur in a given experiment (Wang, 2021).

D CROSS-TEMPORAL DECODING

Following an established method in the neuroscience literature, we use a correlation-based classifier for decoding task-relevant information from neural activities Meyers et al. (2008); Stokes et al. (2013); King and Dehaene (2014); Miconi (2017) . Each trial is performed in parallel by 250 networks. We divide this batch into 125 training runs and 125 testing runs. We separately average the network activities of the training set for each of the two possible target responses; for each time point, this gives us two average vectors of neural activities, each of which represents the “prototype” neural activity at this point for one of the two target responses. Then, for any pair of time points t_1, t_2 during the trial, we compute the correlation between the vector of neural activities at time t_2 in each testing run with either of the two response-averaged training activity patterns (stereotypes) at time t_1, and pick the one with the highest correlation as our decoded estimate of the target response for this testing run and this time point. We then compute the average accuracy of this decoding across all 125 test runs. This gives us the decoding performance value which is shown at point t_1, t_2 of the decoding matrices in the figure.

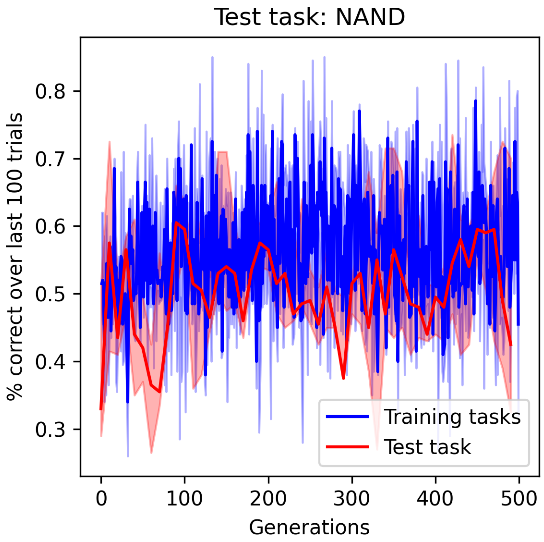

Figure A2: Same as Figure 2 (Left), but with plasticity disabled. The network fails to learn either the training tasks or the withheld task, demonstrating the necessity of all three learning systems.

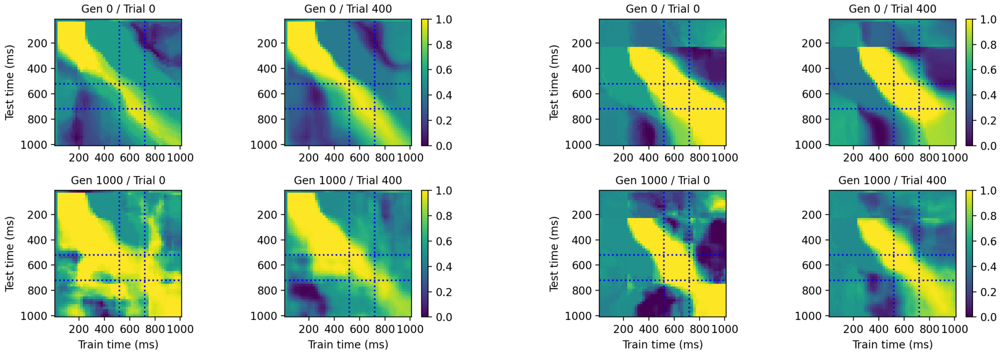

Figure A3: Cross-temporal decoding of first stimulus value (left) and second stimulus value (right) from neural activities. Conventions are as in Figure 3. See text for details.

Note that correlation-based classifiers capture the information that is contained in the collective pattern of the entire network, rather than isolated individual neurons, unlike (say) regression-based classifiers which might focus on specially informative individual neurons.

In Figure A3, we show matrices of cross-temporal decoding for the identity of the first stimulus, and the identity of the second stimulus. We observe a different pattern than for the target response in the DMS task (Figure 3). First, even the generation-0, initialized network encodes this information to some extent, though the encoding is unstable and fleeting (top row). Furthermore, we see that after evolution, lifetime learning does not reinforce or stabilize encoding of stimulus identities during the response period; if anything, the encoding of stimulus identity during the response period seems somewhat less reliable after lifetime learning. This is what we would expect from task-appropriate learning, since the identity of each stimulus is irrelevant to producing the correct response (which depends on equality or inequality of the stimuli, independently of their specific values).

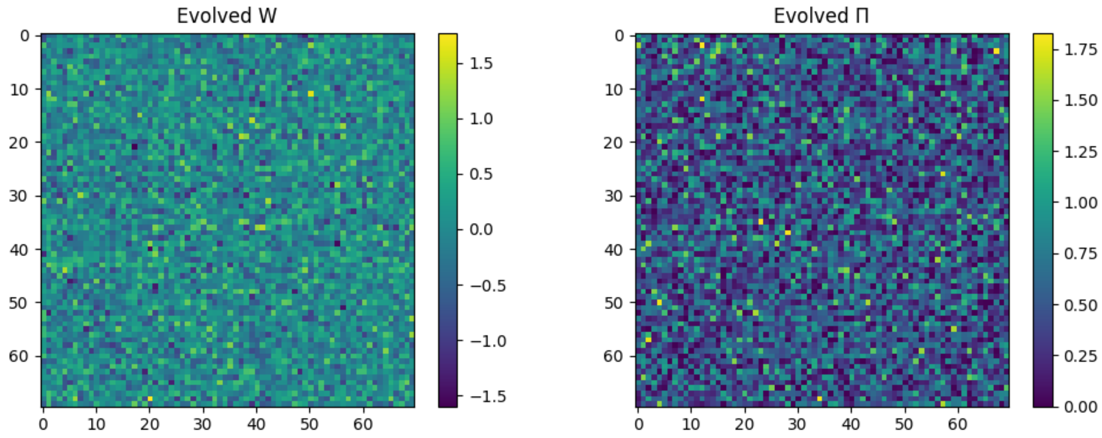

Figure A4: Final $\mathbf{W}$ and $\mathbf{\Pi}$ matrices (initial weights and plasticity coefficients of each connection) after evolution (data from the same run as in Figures 3, A3).

E EQUIVALENCE OF THE DMS TASK AND THE HARLOW TASK

Because the tasks used here involve simple stimuli and operations, it is easy to overlook that they constitute actual meta-learning tasks. To emphasize this point, here we show that the delayed-match-to-sample task ("same or different?") is exactly identical to a simplified version of the Harlow task with two trials per episode.

In the seminal Harlow task Harlow (1949), during each episode, a monkey must determine (by successive trial and error) which of two objects contains food. After enough episodes, monkeys are able to solve new episodes in just one trial, since the information received in the first trial immediately identifies which of the two objects contains the food.

Consider a version of this task where each episode covers only two trials (with the same two objects are used in all episodes). Suppose also that the first choice is passive, that is, the object to be uncovered in the first trial is picked at random by the experimenter under the monkey's view, rather than by the monkey itself. In the second trial, the monkey chooses one of the two objects and gets to consume the food (if he chose correctly) as reward, as in the standard Harlow task. Note that these modifications do not affect the fundamental meta-learning nature of the task.

Yet this task is not just analogous, but exactly identical to our delayed match-to-sample task. The first binary stimulus represents which of the two objects is uncovered in the first trial. The second binary stimulus represents whether or not some food is present under that object. And the agent's response represents which object is to be selected at the second trial, receiving reward (food) if it chose the right object. As the reader can verify, the correct response is exactly identical to that of the DMS task: to receive reward, the agent must produce response 1 if it saw stimuli 1 and then 1 or 0 and then 0, and response 0 otherwise. (Note that in this view, each episode represents two trials rather than just one.)

F EXTENSIONS OF THE MODEL

Here we describe ways in which the system can be extended to improve the resulting network's learning abilities.

F.1 TASKS

For computational efficiency reasons, the framework described above is a simplification of the one introduced by Yang et al. (2019). In particular, we only use binary stimuli and response, as opposed to the real-valued (circular) stimuli and responses of the original study. Yang et al. (2019) also include many more tasks in their experiments, which have the added advantage of biological relevance (many of these tasks being classical tasks from the neuroscience literature). Thus, a straightforward way of improving our model is simply to make it more similar to the original framework of Yang et al. (2019).

We emphasize that the tasks used here, despite their simplicity, are truly meta-learning tasks: new items of information must be acquired, stored and exploited appropriately at each episode. The fact that each trial is a full self-contained episode is similar to the Omniglot task (association between arbitrary stimuli and specific responses). In fact, a homologue of the Omniglot task itself can be implemented in this framework, if we extend the trials to include several, non-binary stimuli: show two arbitrary "cue" stimuli; then the expected responses to each of these cues (arbitrarily chosen for each trial); then show one of the two cues; and finally record the network's response (which should be identical to the expected response for this particular cue, as shown previously).

Furthermore, if we allow for more than one response (and return signal) per trial, we can also include reinforcement learning tasks, such as bandit tasks, in our task set, with each such "trial" still being a complete self-contained episode. Thus, despite its simplicity, the formalism can encompass all forms of meta-learning, including meta-supervised learning (Hochreiter et al., 2001) and meta-reinforcement learning (Wang et al., 2016; Duan et al., 2016).

F.2 LIFETIME PLASTICITY

Perhaps the most simplistic aspect of our model is the handling of lifetime synaptic plasticity. While node perturbation is a powerful model of reward-modulated plasticity Fiete et al. (2007); Miconi (2017), it is obviously not meant to model the totality of synaptic plasticity processes; for example, it cannot implement simple Hebbian learning (under zero or constant rewards, the expected weight change is 0, which is indeed a requirement for reward-modulated Hebbian learning to perform successful reinforcement learning (Frémaux et al., 2010)). Furthermore, in the current system the modulatory signal R is externally applied by the algorithm. A more realistic (and potentially more powerful) method would be to have the modulatory signal under control of the network itself, that is, making $R(t)$ an output of the network.

In addition, in the model described above, the random perturbations of neural activity that support the node perturbation method are applied uniformly with a fixed probability. However, in nature, such sources of randomness are thought to be under control of dedicated brain structures, for example in models of bird song learning Ölveczky et al. (2005). Putting randomness under control of the network is thus another possibility for future work.

F.3 NON-BIOLOGICAL ALTERNATIVES

Our model makes use of biologically inspired methods throughout (evolution, neuromodulated synaptic plasticity, recurrent neural networks, etc.). We chose biologically-based methods both because of intrinsic interest, and because they allow wide flexibility in parametrizing the various systems involved, placing as much as possible under the control of optimization and learning. However, non-biological methods can also be incorporated into this framework.

As an example, alternative that we have considered would be to use backpropagation and gradient descent as the lifetime learning rule, in replacement of synaptic plasticity (assuming a supervised learning signal is available). In this setting, evolution would guide the initialization of the network, while bthousands of timesteps). We note that such a setting would essentially make the two upper loops identical to Evolutionary MAML (as in e.g. Song et al. (2019)), with the bottom two loops being identical to L2RL / RL^2 (Wang et al., 2016; Duan et al., 2016). Such a choice would trade off biological relevance and flexibility again potentially higher performance, at least for some supervised tasks.

Separately, while here we only consider memory mechanisms based on synaptic plasticity and recurrent activations, there are other, non-biological forms of long-term lifetime memory. In particular, information can be stored in explicit banks of embedded key-value pairs, accessed through some attentional mechanism, as in Neural Turing Machines (Graves et al., 2014) and the MERLIN architecture (Wayne et al., 2018). The framework described here can be applied to such explicit forms of memory, in replacement of, or in conjunction with, synaptic plasticity, which might allow

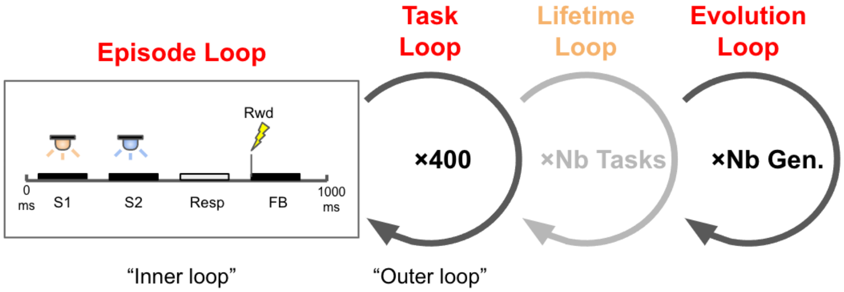

Figure A5: Overall organization of the experiment, with the (unimplemented) "Lifetime loop" added.

Loop:	*Loops over:*	*Learns:*	*Substrate:*	*Algorithm:*
Episode	Timesteps	Episode data	Recurrent activations	Network operation
Task	Episodes	Task structure	Plastic connections (+RA)	Plasticity & Neuromod.
Lifetime	Tasks	Cross-task knowledge	Plastic connections	Plasticity & Neuromod.
Evolution	Lifetimes	Task-learning ability	Genome (Innate $\mathbf{W}$ and $\mathbf{P}$)	Evolution

Figure A6: A classification table of learning loops, including their attributes.

experimenters to assess the relative strengths of both approaches for the purpose of autonomous cognitive learning.

F.4 THE MISSING LOOP: LIFETIME EXPERIENCE

In the current form of the model, each lifetime is composed of one block of trials. Each individual is initialized, performs one task, then is reinitialized again. This is of course unrealistic. In reality, animals acquire a considerable amount of knowledge from their lifetime experience, accumulated across many different problems, which greatly improves their performance in mature adulthood. In animal literature, examples of early life learning include the tuning of sensory cortex, the learning of songs in some bird species, etc. This long-term acquisition and transfer of lifetime experience is particularly developed in humans, in which it forms an important component of so-called "common sense" knowledge, that is thought to be critical to intelligent human behavior (Marcus and Davis, 2019).

In other words, the model described here deliberately excludes an additional learning loop, separate from the three loops considered here: the lifetime experience loop, set between the evolutionary loop and the task loop (see Figure A5). This choice results from a desire for simplicity (and computational efficiency). However, the present model can implement this additional loop by not reinitializing the networks between tasks, and adding more tasks in each lifetime. This may allow us to study the emergence of mechanisms that support robust *continual learning* over a lifetime, including both forward transfer of information across tasks within a lifetime ("life experience", including "common sense" - by adding more tasks per lifetime without reinitialization), and prevention of interference between learned tasks ("catastrophic forgetting" - by repeating previously encountered tasks within each lifetime).

G LEARNING LOOPS: A FUNDAMENTAL COMPONENT IN THE EMERGENCE OF INTELLIGENT BEHAVIOR?

One objective of this paper is to emphasize that multiple learning loops, nested into each other in a hierarchy where each loop optimizes the learning of the previous one, govern the emergence of intelligent agents. This hierarchy extends the traditional dichotomy between "inner loop" and "outer loop" that is commonly observed in meta-learning, as suggested previously (Miconi et al., 2019; Wang, 2021).

As with standard meta-learning, thinking in terms of loops, and asking what is learned in each loop, how is it acquired, and how it is stored, can be helpful in managing the complexities of the process.

One potential concern is the apparent risk of arbitrariness: whether a process is implemented as a loop is often an arbitrary decision by the designed of the experiment. For example, in a standard bandit meta-task, the meta-learning loops over episodes, while the episodes loop over trials (i.e. arm pulls). But what if the programmer decides to implement trials as temporally extended processes (moving towards the arm, grasping the arm, pulling it, etc.), looping over time steps? Does this arbitrary implementation decision add another learning loop to the process, and another "meta-" to the learning?

We found the following definitions useful:

- A loop constitutes an actual learning loop if during each pass of the loop, some new, unpredictable piece of information is acquired, stored and exploited *within the same pass* of the loop.
- Two loops are in a hierarchy (one is the "meta" of the other) if what is learned within one loop is expected to improve the learning in the other loop.

Note that these definitions address the example mentioned above: while the arm pulls may be implemented as "loops", what is learned during each such pull (an instantaneous return from one arm) can only be exploited in *future* trials, or equivalently, in the same pass of the upper-level loop (i.e. the episode loop). Thus, in terms of learning, trials, even though they are implemented as loops, act as mere "macro-timesteps" of their upper-level loop (the episode loop), do not constitute an actual learning loop, and do not add a "meta-" to the process.

The fact that actual learning loops can be defined in a principled way, making them independent of arbitrary implementation choices, suggests that these loops are not mere incidental appearances, but constitute genuine, fundamental, objectively quantifiable components in the emergence of intelligent behavior.

Based on this framework, we can characterize each loop by asking specific questions about it, including:

- What is being looped over?
- What is being learned?
- What is the substrate (that is, where is the thing to be learned stored)?
- What is the algorithm?

We found these questions useful in preventing confusion between the multiple loops involved in natural learning. An example of this classification is shown in Figure A6.