Provably Sub-Linear Two-Timescale NeuroEvolution with Online Plasticity

Shishen Lin[1] , **Yixin Chen**[2]
[1]Department of Statistics, University of Warwick, Coventry, United Kingdom
[2]Biozentrum, University of Basel, Basel, Switzerland
shishen.lin.1@warwick.ac.uk, chenyx1119@gmail.com

Abstract

NeuroEvolution of Augmenting Topologies (NEAT) is a widely used neuroevolution algorithm for learning neural network architectures and weights for control tasks. However, standard offline optimisation searches for connection strengths directly, which can scale poorly in high-dimensional weight spaces and more difficult continuous control problems. Hybrid methods that combine neuroevolution with online learning can address this challenge, but their theoretical properties remain underexplored. This paper gives the first regret analysis for a general NeuroEvolutionary Online Learning (NEOL) framework, which decouples learning into two timescales: an outer loop for architecture search and an inner loop for online weight adaptation via reward-modulated plasticity. Under mild conditions, we prove that NEOL achieves sublinear regret. Empirically, under fixed interaction budgets on four standard control benchmarks, a NEAT-based NEOL implementation achieves higher final fitness and lower variance than pure NEAT, and is competitive with strong reinforcement learning (RL) baselines on several tasks. The results are supported by Wilcoxon rank-sum tests and ablation studies. Overall, the findings show that online plasticity can improve the sample efficiency and robustness of two-timescale neuroevolution. Code is available at https://github.com/boobaa2001/NeuroEvolution_Online_Learning_NEOL.

1 Introduction

NeuroEvolution (NE) employs evolutionary operators, rather than gradient descent, to optimise neural network architectures and parameters [Miikkulainen, 2025; Miikkulainen *et al.*, 2024; Stanley *et al.*, 2019; Khan *et al.*, 2010; Yao, 1999; Angeline *et al.*, 1994]. It has been successfully applied to biologically inspired models for lifelong learning [Kudithipudi *et al.*, 2022], reinforcement learning (RL) tasks [Xue *et al.*, 2024; Chalumeau *et al.*, 2023; Co-Reyes *et al.*, 2021; Khadka and Tumer, 2018; Stanley and Miikkulainen, 2002a], and domain-specific challenges (i.e., optimisation of land-use planning policies for carbon reduction [Young *et al.*, 2025]).

A key limitation of pure neuroevolution is that weight optimisation in high-dimensional spaces relies on mutation-based perturbations. These often provide weak fitness assignment, leading to unstable optimisation dynamics and premature convergence in complex continuous-control problems [Stanley *et al.*, 2009]. Furthermore, many neuroevolutionary algorithms are offline: a population is evolved on a task, and the policy is fixed at deployment. This approach is often inadequate in settings requiring real-time adaptation under sequential interaction and non-stationary environments [Agogino *et al.*, 2000; Bellman, 1966; Sutton and Barto, 1998]. These challenges motivate hybrid methods that retain evolutionary local search for structure and network, while introducing direct mechanisms for online weight adaptation [Agogino *et al.*, 2000; Stanley *et al.*, 2003; Peng *et al.*, 2018].

Combining population-based search with lifelong learning is a foundational goal for adaptive intelligence [Schmidhuber, 1987; Holland, 1992; Miikkulainen, 2025]. While early attempts to evolve synaptic plasticity rules directly struggled as genetic search spaces for learning rules expanded [Agogino *et al.*, 2000; Stanley *et al.*, 2003], the introduction of neuromodulation, using Hebbian updates with reward signals, has enabled networks to solve dynamic control tasks more efficiently than were previously intractable for both fixed-weight and non-modulated plastic architectures [Soltoggio *et al.*, 2008; Soltoggio *et al.*, 2018; Najarro and Risi, 2020]. This established a robust two-timescale loop where an outer evolutionary process designs an inner online learner.

Despite these empirical successes, theoretical analysis of neuroevolution is limited. Existing studies focus entirely on the runtime analysis of evolutionary neural architecture search for discrete optimisation tasks [Fischer *et al.*, 2023; Lv *et al.*, 2024; Lv *et al.*, 2025], building an important block for a rigorous understanding of neuroevolution for discrete search spaces. Although they are an important first step, these theoretical analyses cannot directly be applied to neuroevolution with online learning, especially in a continuous search space. To the best of our knowledge, a formal regret analysis of neuroevolution or its online variants remains unexplored. This paper addresses this gap by introducing a general NeuroEvolutionary Online Learning (NEOL) framework. As a reasonable starting point, we employ an exponential weight

update mechanism, in which the algorithm selects the architecture via softmax selection (we will explain further in Section 3, A6), using a simplified model for the selection process rather than more complicated selection algorithms. We do not model the full nonconvex MDP dynamics of MuJoCo (Multi-Joint dynamics with Contact)/Box2D (2D game-like RL environments) or other control tasks, as NEOLs (e.g., NEAT) might be highly intractable due to their design. Instead, we isolate the algorithmic roles of: (i) outer search over network topology, and (ii) inner online weight adaptation from interaction feedback via local plasticity (NEOL-style reward-modulated, including Hebb/Oja/BCM updates), Therefore, this paper aims to answer:

(1) Can we provide a regret guarantee for NeuroEvolutionary Online Learning (NEOL) methods?

(2) How can we effectively decompose policy learning into structural evolution and online weight adaptation?

1.1 Contribution

Our contributions are primarily in three folds. First, we introduce a general NEOL framework and provide a first regret analysis of this framework, showing that under mild conditions and exponential-weights selection over neural architectures, NEOL with reward-modulated plasticity rules (BCM, Hebb, and Oja) achieves sublinear regret $O(\sqrt{T})$, i.e., implying asymptotically optimal average performance. Second, we propose NEAT-NEOL, a practical implementation that decouples topology (network architecture) evolution and online weight adaptation via plasticity rules. Third, we conduct extensive experiments on four standard control benchmarks (CartPole, Lunar Lander, Hopper, and Bipedal Walker) under fixed interaction budgets. We also compare NEAT-NEOL with standard NEAT and strong RL baselines (PPO and SAC), and validate the results through Wilcoxon rank-sum tests and ablation studies. These findings highlight the potential of online plasticity in improving the performance (best fitness) and robustness of two-timescale NeuroEvolution in interactive environments.

2 Preliminaries and Notation

Notation. $|A|$ is the cardinality of a finite set A. We use the standard Euclidean inner product on $\mathbb{R}^d$: $\langle w, u\rangle := \sum_{j=1}^{d} w_j\, u_j, \|w\|_2 := \sqrt{\langle w, w\rangle}$. We define Euclidean projection of $x \in \mathbb{R}^d$ on space $\mathcal{W}$ by $\Pi_{\mathcal{W}}(x) := \arg\min_{w\in\mathcal{W}} \frac{1}{2}\|w - x\|_2^2$. For a convex function $f : \mathbb{R}^d \to \mathbb{R}$, the subdifferential at x is $\partial f(x) := \{g \in \mathbb{R}^d : f(y) \geq f(x) + \langle g, y - x\rangle\ ,\ \forall\, y \in \mathbb{R}^d\}$. Any $g \in \partial f(x)$ is called a subgradient of f at x. An algorithm is no-regret if its regret is sublinear $\mathbb{E}[R_T] = o(T)$, i.e., its average performance converges asymptotically to the optimal policy[1]. The natural filtration, $(\mathcal{G}_n)_{n\geq 0}$ associated with the stochastic process $(X_n)_{n\geq 0}$ on probability space $(\Omega, \mathcal{F}, \Pr)$ is given by $\mathcal{G}_n = \sigma(X_0, \ldots, X_n)$ (sigma-algebra generated by a family of r.v.s..) for $n \geq 0$.

[1]Note that different no-regret bounds can differ in different convergence rates. "Optimal" here does not mean the convergence rate is the best [Orabona, 2019].

Online Learning via Synaptic Plasticity. Online learning follows a sequential protocol: a learner repeatedly acts, observes (reward) feedback, and updates its parameters immediately, to minimise cumulative loss over time [Shalev-Shwartz, 2011]. In biological systems, life cycle adaptation is often attributed to synaptic plasticity, in which synaptic strength varies with changes in local neural activity [Martin *et al.*, 2000; Takeuchi *et al.*, 2014]. Importantly, many plasticity mechanisms are gated by a third factor, such as neuromodulatory signals correlated with reward, yielding three-factor rules that support behaviourally relevant credit assignment without backpropagating gradients through time [Frémaux and Gerstner, 2016; Gerstner *et al.*, 2018; Florian, 2007]. Motivated by this, we use reward-modulated synaptic updates as the inner-loop learning mechanism in NEOL. Let x denote presynaptic activity, y postsynaptic activity, w a synaptic weight, $\eta > 0$ a step size, r the scalar reward, $\tau \in \mathbb{N}$ denote the time step in the ROLLOUT and $\beta > 0$ a reward scaling factor. We study three standard local rules.

Reward-modulated Hebbian rule. Following Hebb's rule [Hebb, 1949], correlated pre- and postsynaptic activities strengthen a synapse. We use a reward-gated three-factor form whose local correlation defines $e_\tau = x_\tau y_\tau$: $\Delta w_\tau = \eta\,\beta\, r_\tau\, e_\tau = \eta\,\beta\, r_\tau\, x_\tau\, y_\tau$. This rule is maximally local and simple, but can be unstable without additional constraints [Caporale and Dan, 2008].

Reward-modulated Oja rule. Oja's rule varied from Hebbian learning with an additional activity-dependent negative feedback normalisation that prevents divergence [Oja, 1982]. Using the same reward-gating yields $e_\tau = y_\tau\big(x_\tau - y_\tau w_\tau\big), \Delta w_\tau = \eta\,\beta\, r_\tau\, e_\tau = \eta\,\beta\, r_\tau\, y_\tau\big(x_\tau - y_\tau w_\tau\big)$. This stabilises the dynamics and yields a principal-component interpretation under stationary inputs.

Reward-modulated BCM rule. The BCM rule introduces a sliding threshold θ_τ that separates depression from potentiation and supports selectivity with homeostasis [Bienenstock *et al.*, 1982]. In reward-modulated form, $e_\tau = y_\tau\big(y_\tau - \theta_\tau\big)x_\tau, \Delta w_\tau = \eta\,\beta\, r_\tau\, e_\tau = \eta\,\beta\, r_\tau\, y_\tau\big(y_\tau - \theta_\tau\big)x_\tau$. The threshold θ_τ is updated on a slower timescale to track recent postsynaptic activity.

3 NeuroEvolutionary Online Learning

In this section, we present a NeuroEvolutionary Online Learning (NEOL) framework. We begin with a high-level overview, then describe the decoupled update strategy for weights and topology in Section 3.1, and finally detail the main algorithm in Section 3.2.

3.1 Decoupling Updates for Weight and Topology

We maintain a population of network architectures, each encoded as a genome. The flowchart in Fig. 1 shows a generational loop in which evolution and evaluation are interleaved to progressively improve solutions. Unlike standard NEAT, which mutates both topology and weights offline between episodes and evaluates policies with fixed weights during rollouts, NEOL decouples the two update processes. During each rollout, synaptic plasticity performs online weight

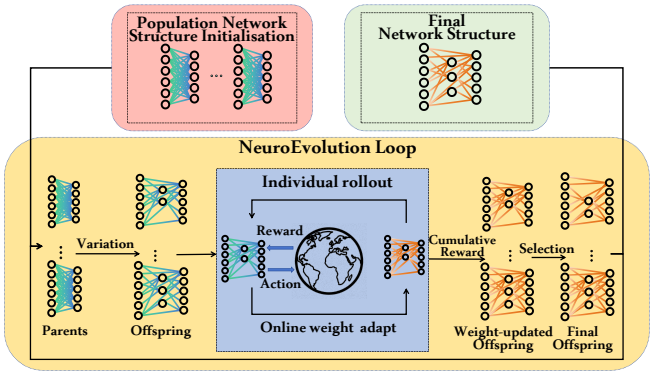

Figure 1: Overview of the NEOL framework. The procedure begins by initialising the network structures population. In each generation, the variation operators generate offspring from the current parent generation. Each offspring is evaluated in a separate rollout, during which its weights are adapted online via reward-modulated plasticity (illustrated connectivity lines colour change from blue to orange). The cumulative reward from the rollout is used as the fitness for selection to form the next generation. The best individual in the population determines the final network structure, including its adapted weights and evolved topology.

adaptation once the reward feedback is observed and then, based on the current policy, the agent chooses an action to interact with the environment until the inner loop phase ends. Then, the outer loop uses an evolutionary variation to modify the topology/architecture of networks and select the final offspring based on final fitness, which is the cumulative reward of the agent interacting with the environment. Such a separation reserves the structural innovation and adapts to real-time feedback, thereby improving the performance of the algorithm (in the sense of fitness) and stabilising the search process (see Section 5).

3.2 General Outer-Inner Framework for NEOL

This section provides more detailed pseudocode for NEOL. Algorithm 1 consists of a two-timescale learning phase. At the outer loop ($t = 1, \dots, T$), the algorithm selects an architecture $h_t \in \mathcal{H}$ using an outer selector state Θ_t (which might be a probability distribution over candidates, or a population of architectures in evolutionary algorithms such as NeuroEvolution of Augmenting Topologies (NEAT, a neuroevolution method that evolves both neural network topology and connection weights in an offline manner [Stanley and Miikkulainen, 2002b].). The selected architecture is then evaluated by an inner ROLLOUT learning procedure (Algorithm 2), which adapts the weights online through reward-modulated plasticity and returns an estimated fitness $\widehat{J}_t$. "Env" denotes the task environment, modelled as an episodic MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho_0)$ with horizon $T_{\max}$, where $\mathcal{S}$ is state space, $\mathcal{A}$ is action space, $P(\cdot \mid s, a)$ is transition kernel, $r(s, a)$ is a reward function, and ρ_0 is the initial-state distribution. Finally, the outer state Θ_t is updated based on this evaluation, favouring future selections toward higher-performing architectures.

Algorithm 2 evaluates a fixed architecture h by performing an online adaptation through reward-modulated plasticity. For each of the N episodes, the environment is reset,

Algorithm 1 General OUTER-INNER framework

Input: Candidate space $\mathcal{H}$ (architectures/genomes); outer rounds T. Outer selector state Θ (e.g., a distribution p_t on $\mathcal{H}$, or a NEAT population $\mathcal{P}_t$). Inner loop: learning rule $\mathcal{L}$; step size η; reward scale β; episodes N; horizon $T_{\max}$; env.

Output: Final architecture and weights $(h_T, w_T^{(h_T)})$.

$\Theta_1 \leftarrow$ OUTERINIT$(\mathcal{H})$ ▷ e.g., uniform $p_1(h) = 1/|\mathcal{H}|$ or random initial population $\mathcal{P}_1$
for $t = 1, \dots, T$ **do** ▷ Outer: architecture selection
  $h_t \leftarrow$ OUTERSELECT(Θ_t)
  $(w_t^{(h_t)}, \widehat{J}_t) \leftarrow$ ROLLOUT$(h_t, \mathcal{L}, \eta, \beta, N, T_{\max}, env)$
  $\ell_t(h_t) \leftarrow$ LOSS$(\widehat{J}_t)$ ▷ e.g., $\ell_t = -\widehat{J}_t$
  $\Theta_{t+1} \leftarrow$ OUTERUPDATE$(\Theta_t, h_t, \ell_t(h_t))$
end for
return $(h_T, w_T^{(h_T)})$

Algorithm 2 ROLLOUT: reward-modulated plasticity

Input: Fixed architecture h; learning rule $\mathcal{L}$; step size η; reward scale β; episodes N; inner horizon $T_{\max}$; env.

Output: Adapted weights $w^{(h)}$; average return $\widehat{J}(h, w^{(h)})$.

1: RETURNS $\leftarrow []$
2: $w^{(h)} \leftarrow$ null
3: **for** $e = 1, \dots, N$ **do**
4: Instantiate net from h; initialise plastic weights $w_1^{(h)}$
5: $s \leftarrow env.reset(), R \leftarrow 0$
6: **for** $\tau = 1, \dots, T_{\max}$ **do**
7: $a \leftarrow \pi_{w_\tau^{(h)}}(s)$
8: $(s', r, \text{done}) \leftarrow env.step(a)$
9: $u_\tau^{(h)} \leftarrow$ LOCALTRACE(s, a, s')
10: $w_{\tau+1}^{(h)} \leftarrow$ PLASTICUPDATE$(w_\tau^{(h)}, u_\tau^{(h)}, r; \mathcal{L}, \eta, \beta)$
11: $R \leftarrow R + r, s \leftarrow s'$
12: **if** done **then break**
13: **end if**
14: **end for**
15: RETURNS.APPEND(R)
16: $w^{(h)} \leftarrow w_{\tau+1}^{(h)}$
17: **end for**
18: $\widehat{J}(h, w^{(h)}) \leftarrow \frac{1}{N} \sum_{e=1}^{N}$ RETURNS$[e]$
19: **return** $(w^{(h)}, \widehat{J}(h, w^{(h)}))$

and the network weights are initialised uniformly at random over candidate architectures h (line 4). At every interaction step $\tau \leq T_{\max}$, the current policy $\pi_{w_\tau^{(h)}}$ produces an action a, the environment returns the next state and reward, and a local synaptic trace $u_\tau^{(h)}$ is computed from pre/post-synaptic activity (line 9). The weights are then updated online via a reward-gated plasticity rule (Hebb/Oja/BCM) with step size η and reward scaling β (line 11). Finally, the episode return is accumulated as fitness, and the final output is the average return $\widehat{J}(h, w^{(h)})$ across episodes together with the adapted weight vector $w^{(h)}$.

This paper presents a simplified theoretical lens for the

outer-loop with inner-loop structure in Algorithms 1–2. We show that under standard boundedness assumptions, the hybrid admits a sublinear regret bound. For simplicity of both theoretical and empirical analysis, we consider episode $N = 1$ in this paper.

3.3 Simplified protocol and regret definition

Let $\mathcal{H}$ be a finite set of candidate genomes/architectures (e.g., a finite pool of NEAT topologies), with $M := |\mathcal{H}| \in \mathbb{N}$. Each $h \in \mathcal{H}$ induces a parameter space $\mathcal{W}_h \subset \mathbb{R}^{d_h}$. We consider an online interaction protocol over outer rounds $t = 1, \dots, T$.

1. At outer round t, the environment defines an evaluation instance (e.g., initial state distribution).
2. The outer loop selects an architecture $h_t \in \mathcal{H}$ according to $p_t \in \Delta(\mathcal{H})$.
3. Given h_t, the inner loop runs for N episodes at most $T_{\max}$ iterations using reward-modulated plasticity, generating adaptive weights $w_t^{(h_t)} \in \mathcal{W}_{h_t}$ and a (stochastic) average return $J_t\big(h_t, w_t^{(h_t)}\big) \in [0,1]$ (normalised).
4. Define the (stochastic) loss at round t as $\ell_t\big(h_t, w_t^{(h_t)}\big) := 1 - J_t\big(h_t, w_t^{(h_t)}\big) \in [0,1]$.

As a simplified mode, in our analysis, we consider $N = 1$ and $T_{\max}$ is constant with respect to the outer iteration T.

We define the (expected) regret of the hybrid method as

$$\mathbb{E}[R_T] := \mathbb{E}\left[\sum_{t=1}^{T} \ell_t\big(h_t, w_t^{(h_t)}\big) - \min_{h\in\mathcal{H}} \min_{w\in\mathcal{W}_h} \sum_{t=1}^{T} \ell_t(h,w)\right],$$

where $\ell_t(h,w)$ denotes the loss that would be incurred on round t if the candidate (h,w)that is in round t if deployed on the same evaluation instance, and the expectation is taken from all the randomness in evaluation instances, rollouts, and the algorithm. Regret quantifies the cost of online learning. The use of the Regret metric $R(T)$ is motivated by the need to balance structural exploration (finding the right NEOL topology) with parametric exploitation (optimising weights for a given topology). A sub-linear regret bound, $R(T)/T \to 0$ as $T \to \infty$, guarantees that the learner's cumulative reward is asymptotically as good as the best possible architecture-weight pair (h^*, w^*) in the candidate set.

3.4 Assumptions

Most NEOL algorithms, including NEAT and its variants, are highly intractable. It is sensible to make some assumptions before conducting regret analysis.

A1 Finite architecture. $\mathcal{H}$ is finite with $|\mathcal{H}| = M < \infty$.

A2 Bounded rewards/losses. Rewards are normalised so that $r_t(h,w) \in [0,1]$ for all t, h, w, hence $\ell_t(h,w) \in [0,1]$.

A3 Bounded activities and bounded parameter domain. For each architecture h, the (vectorised) weights lie in a convex and compact set $\mathcal{W}_h$ with diameter at most D: $\|w - w'\|_2 \le D, \forall w, w' \in \mathcal{W}_h$. Moreover, for each h and each round t, the network induces bounded local synaptic traces (e.g., pre-/post-synaptic activity products) denoted by $z_t := x_t y_t \in \mathbb{R}^{d_h}$, there exists a constant Z such that $\|z_t\|_2 \le Z$ for all t, h. For BCM, we additionally assume a bounded sliding threshold $\theta_t \in [-1,1]$ and bounded post-synaptic activity $y_t \in [-1,1]$.

A4 NEOL inner update is a reward-modulated local plasticity rule. For each fixed architecture h, the within-lifetime update (Algorithm 2, line 9-10) is a local rule driven by synaptic traces and reward, implemented with projection to keep weights in $\mathcal{W}_h$:

$$w_{t+1}^{(h)} = \Pi_{\mathcal{W}_h}\big(w_t^{(h)} + \eta\, u_t^{(h)}\big), \tag{1}$$

where $u_t^{(h)}$ depends only on local quantities and the reward. Assumption 4 does not assume backpropagation or policy gradients. It only assumes that the inner-loop update is a bounded local synaptic rule of the form (1), which holds for the reward-modulated Hebb/Oja/BCM updates below when weights are clipped/projected.

$$u_t^{(h)} = \begin{cases} r_t z_t & \text{Hebb,} \\ r_t\left(z_t - \alpha_t w_t^{(h)}\right) & \text{Oja,} \\ r_t y_t (y_t - \theta_t)\, x_t & \text{BCM.} \end{cases}$$

Here $\alpha_t \ge 0$ is bounded (e.g., $\alpha_t = y_t^2$ with $y_t \in [-1,1]$), and y_t, θ_t are bounded as in Assumption 3. The pre-synaptic activity x_t can be absorbed into z_t if desired. We say the plasticity rule is reward-modulated if $u_t^{(h)} = r_t \cdot$ local trace, where r_t is the reward.

A5 Surrogate domination almost certainly. For each architecture $h \in \mathcal{H}$ and each round t, define a linear surrogate $L_t^{(h)}(w) := -\langle w, u_t^{(h)} \rangle$. We assume there exists a constant $C \ge 1$ such that for all $w \in \mathcal{W}_h$,

$$\ell_t\big(h, w_t^{(h)}\big) - \ell_t(h,w) \;\le\; C\,\left(L_t^{(h)}\big(w_t^{(h)}\big) - L_t^{(h)}(w)\right)$$

holds almost surely. An informal intuition for Assumption 5 is: if $\ell_t(h,\cdot)$ is locally smooth and the reward-modulated plasticity direction $u_t^{(h)}$ is positively aligned with the descent direction of $\ell_t(h,\cdot)$ at $w_t^{(h)}$, then the decrease in the true loss can be controlled by the decrease in the linear surrogate,

A6 Selection as a soft and fitness-based reweighting. At each outer round t, NEOL produces a population of genomes and applies selection based on their fitness values. For theoretical tractability, we model this selection as maintaining a probability distribution p_t over a finite candidate set $\mathcal{H}$ of genomes/architectures, and updating it by a monotone fitness-based reweighting rule of exponential-weights form:

$$p_{t+1}(h) \;=\; \frac{p_t(h)\exp\big(-\gamma\,\widehat{\ell}_t(h)\big)}{\sum_{h'\in\mathcal{H}} p_t(h')\exp\big(-\gamma\,\widehat{\ell}_t(h')\big)}, \tag{2}$$

where $\widehat{\ell}_t(h)$ is an empirical estimate of $\ell_t(h, w_t^{(h)})$. The exponential weight update follows the classical Hedge algorithm [Freund and Schapire, 1997]. This abstraction captures that genomes with smaller estimated loss (higher fitness) become more likely to be selected, while ignoring NEAT-specific mechanisms such as speciation, explicit fitness sharing, and structural mutations. Exponential weights are a standard setting and starting point for such selection dynamics, helping us conduct a clear outer-loop regret analysis.

A7 Full-information evaluation over the candidate pool. At each outer round t, we assume the loss values are revealed for all candidates in the pool $\mathcal{H}$: the outer update has access to the entire loss vector $\big(\ell_t(h, w_t^{(h)})\big)_{h\in\mathcal{H}}$ at round t.

4 Regret Analysis of a Simplified NEOL

First, we consider the inner-loop regret for reward-modulated local updates.

Lemma 1. *Fix any $h \in \mathcal{H}$. Under Assumptions A2–4, define the (convex) surrogate inner loss: $L_t^{(h)}(w) := -\langle w, u_t^{(h)}\rangle$, where $u_t^{(h)}$ is the local update direction from* (1)*. Then the projected update* (1) *satisfies*

$$\sum_{t=1}^{T} L_t^{(h)}\big(w_t^{(h)}\big) - \min_{w\in\mathcal{W}_h} \sum_{t=1}^{T} L_t^{(h)}(w) \le \frac{D^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T} \|u_t^{(h)}\|_2^2.$$

Moreover, if $\|u_t^{(h)}\|_2 \le U$ for all t (implied by A2–4), then

$$\sum_{t=1}^{T} L_t^{(h)}\big(w_t^{(h)}\big) - \min_{w\in\mathcal{W}_h} \sum_{t=1}^{T} L_t^{(h)}(w) \;\le\; \frac{D^2}{2\eta} + \frac{\eta U^2 T}{2}.$$

Choosing $\eta = D/(U\sqrt{T})$ yields

$$\sum_{t=1}^{T} L_t^{(h)}\big(w_t^{(h)}\big) - \min_{w\in\mathcal{W}_h} \sum_{t=1}^{T} L_t^{(h)}(w) \;\le\; DU\sqrt{T}.$$

Lemma 1 shows that under the bounded assumptions for architecture and bounded local synaptic update, we can bound the inner regret by $DU\sqrt{T}$, where the weight vectors lie in a convex and compact set $\mathcal{W}_h$ with diameter D and the 2-norm of the bounded local synaptic traces is at most U. This implies that the inner regret can be controlled by the size of the space of weights and local synaptic update in a linear sense.

Next, we consider the regret of the outer-loop, assuming the architecture uses exponential weights/Hedge.

Lemma 2. *Let $\{\mathcal{F}_t\}_{t\ge1}$ be a natural filtration generated by all randomness and loss observations up to the end of round $t-1$. If $h_t \sim p_t$ is drawn after observing $\mathcal{F}_t$ and under Assumptions 1,2,6,7 with losses in $[0,1]$, the exponential-weights update* (2) *satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_{h_t\sim p_t}\Big[\ell_t\big(h_t, w_t^{(h_t)}\big) \mid \mathcal{F}_t\Big] - \min_{h\in\mathcal{H}} \sum_{t=1}^{T} \ell_t\big(h, w_t^{(h)}\big) \le \frac{\log M}{\gamma} + \frac{\gamma T}{8}.$$

Choosing $\gamma = \sqrt{8\log M/T}$ yields

$$\sum_{t=1}^{T} \mathbb{E}_{h_t\sim p_t}\Big[\ell_t\big(h_t, w_t^{(h_t)}\big) \mid \mathcal{F}_t\Big] - \min_{h\in\mathcal{H}} \sum_{t=1}^{T} \ell_t\big(h, w_t^{(h)}\big) \le \sqrt{\frac{T\log M}{2}}.$$

Lemma 2 shows that we can bound the outer regret conditioning on the current observation by $\sqrt{(T\log M)/2}$, where M is the size of the set of architectures. This implies that the outer regret can also be controlled by the size of the set of architectures in a square-root sense.

Finally, using Lemmas 1 and 2, we obtain:

Theorem 3. *Under Assumptions 1–7, with $\eta = D/(U\sqrt{T})$ in Lemma 1 and $\gamma = \sqrt{8\log M/T}$ in Lemma 2, the expected regret is bounded as*

$$\mathbb{E}[R_T] \;\le\; CDU\sqrt{T} \;+\; \sqrt{\frac{T\log M}{2}} \;=\; O\Big(\sqrt{T}\Big).$$

Proof Sketch. Theorem 4 decomposes the regret into an inner-loop term $CDU\sqrt{T}$ (cost of weight adaptation) and an outer-loop term $\sqrt{T\log M/2}$ (cost of architecture search). Both are sublinear, so their sum is sublinear. □

In our analysis, we model Algorithms 1 and 2 with reward-gated synaptic updates as an inner-loop online procedure of the form (1) (A3–4), where the update direction $u_t^{(h)}$ depends only on local traces and the scalar reward. For theoretical tractability, we consider the outer selection as an exponential-weights update over a finite architecture pool $\mathcal{H}$ (A1 and A7). In practice, Algorithm 1 might evaluate each genome in the population via repeated online rollouts and then apply selection pressure through reproduction/speciation, which we investigate in Section 5. Moreover, we consider a full-information outer update in which losses for all $h \in \mathcal{H}$ are available (A7). Under these assumptions, the hybrid outer/inner-loop method admits a sublinear $O(\sqrt{T})$ regret guarantee, supporting this interpretation as an effective combination of evolutionary architecture search and online learning via plasticity updates.

5 Experiments

If we choose the learning rule $\mathcal{L} \in \{\text{Hebb, Oja, BCM}\}$ in Algorithm 1, then the general NEOL framework becomes a NEAT-based NEOL method (we give the pseudocode in the Appendix). In this section, we evaluate this method (namely, NEAT-NEOL) and compare it with standard NEAT and two strong reinforcement learning baselines. We test whether online weight updates from the plasticity rules help, and how the results compare with well-tuned PPO and SAC in practice.

First, we provide a formal formulation of sequential decision making by using the Markov Decision Process (MDP). Given an MDP defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, T\rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R} : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ is the reward function and $\mathcal{P} : \mathcal{S}\times\mathcal{A} \to \mathcal{S}$ is the transistion function. In this paper, we consider an online RL setting where the agent can interact with the environment repeatedly until a certain time horizon T by using a policy $\pi : \mathcal{S} \to \mathcal{A}$. Such an agent's policy is usually represented by a neural net. Then, the goal of the entire learning process is to find an optimal policy π^* such that it can maximise the expected rewards:

$$\pi^* \in \arg\max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \mathcal{R}\left(s_t, a_t\right) \mid s_0, a_0\right].$$

As a special setting for NeuroEvolution, this paper directly refers to the cumulative reward as the fitness of the policy at time horizon T, i.e., $f(\pi, T) := \sum_{t=0}^{T} \mathcal{R}\left(s_t, a_t\right)$.

5.1 Experiment setups

Benchmark Environments. We evaluate algorithms on four control tasks from the OpenAI gym benchmark suite [Brockman *et al.*, 2016], spanning diverse reward structures and action spaces (detailed description deferred to Appendix).

Baselines. We compare NEAT-NEOL with two baseline families. (1) NEAT [Stanley and Miikkulainen, 2002b]: we use the same outer-loop evolutionary search as in NEAT-NEOL, but remove online synaptic plasticity, i.e., synaptic weights are fixed during each evaluation rollout. (2) Proximal Policy Optimisation (PPO) [Schulman *et al.*, 2017] and Soft Actor-Critic (SAC) [Haarnoja *et al.*, 2018]: PPO serves as a strong on-policy baseline and is used on `CartPole`; for continuous-control tasks (`LunarLander`, `BipedalWalker`, `Hopper`), we evaluate both PPO and SAC to reflect standard practice for continuous action spaces.

Experimental Configuration for NEAT and NEAT-NEOL. We ran NEAT and NEAT-NEOL with population sizes $P \in \{50, 100, 200, 300\}$. Each run lasted for $G = 500$ generations. In each generation, we evaluated all P individuals on one rollout, so each run contained $P \times G$ rollouts in total (and $P \times G \times T_{max}$, where T_{max} is the rollout length in environment steps). For NEAT-NEOL, we tuned the inner-loop learning rate `lr` by grid search over $\{2.5\times10^{-4}, 2.5\times10^{-3}, 2.5\times10^{-2}, 2.5\times10^{-1}\}$ and $T_{max} = 10^3$. For each configuration, we used 30 independent random seeds (See Figures 2 and Figures 6-9 in Appendix).

Experimental Configuration for PPO and SAC. For PPO and SAC, we trained each run for up to 10^7 environment steps per seed. To make a fair comparison under the same interaction budget as the evolutionary runs, we also report results at $P\times G\times T_{max} = 10^7$ environment steps for NEAT and NEOL (i.e., $P \in \{50, 100\}, G \in \{100, 200\}$ and $T_{max} = 10^3$). We used 30 random seeds and report boxplots of the best fitness (episode return) across seeds (See Figure 3).

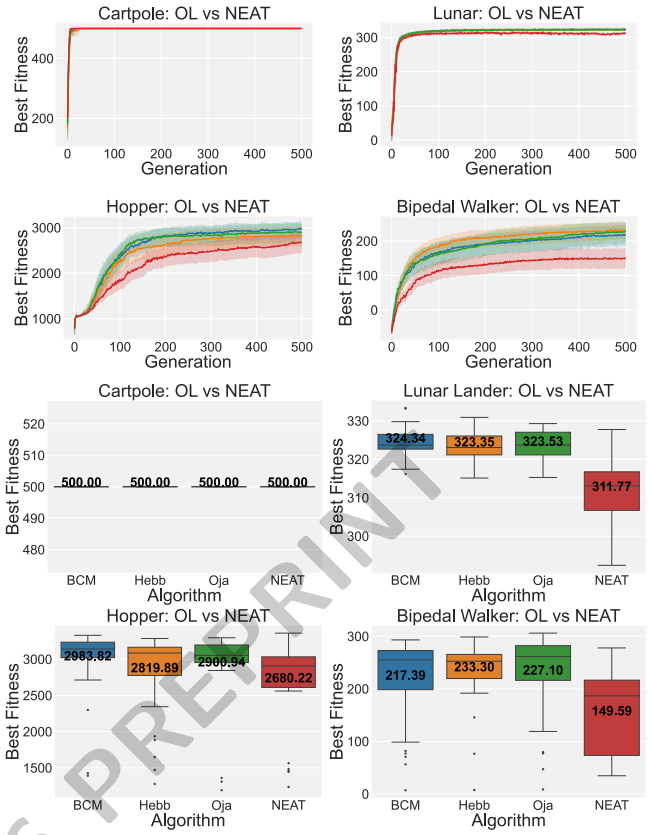

Figure 2: Fitness comparison of NEAT-NEOL with NEAT across four control tasks in 30 independent runs, each task is under a fixed interaction budget $P \times G \times T_{\max}$ environment steps. BCM, Hebb, and Oja learning rules are shown in blue, orange, and green, respectively, while standard NEAT is shown in red.

5.2 NEAT-NEOL vs NEAT vs RL Baselines

To systematically evaluate the role of online neural plasticity in evolutionary processes, we conducted a comprehensive comparison against NEAT. This comparison pitted the standard NEAT method against our NEOL framework, which integrates online plasticity rules (Hebb, Oja, and BCM). The results clearly demonstrate that the NEOL framework exhibits substantial advantages across multiple test environments.

In the simpler CartPole task, all methods rapidly converge to the maximum fitness, serving as a successful sanity check. However, in the more complex environments, significant performance disparities emerge. As shown in the convergence plots (Figure 2, top two rows), the NEOL variants consistently achieve higher final fitness scores than standard NEAT. The boxplots (Figure 2, bottom two rows) and standard deviations (Table 4) further reveal that while standard NEAT struggles, often resulting in a high variance and numerous low-performing outliers, the NEOL methods achieve a superior median performance. Notably, in three continuous control tasks, NEOL not only outperforms NEAT but also exhibits a tighter fitness distribution, indicating higher learning reliability in terms of smaller variance.

The observed performance gains are not coincidental. We confirmed their statistical significance using a one-sided Wilcoxon rank-sum test. The null hypothesis is that the two configurations yield samples from the same distribution; the alternative hypothesis is that the OL method tends to achieve the larger best-fitness than NEAT. As detailed in Table 5, we reject the null hypothesis at a significance level of $p < 0.05$ for all NEOL variants across all three tasks apart from Cartpole. This provides strong evidence that the integration of online learning improves the performance of algorithms.

Next, we compare NEOL with PPO and SAC. As shown in Figure 3, under the same interaction budget of 10 million (10M) timesteps, NEOL and NEAT outperform PPO and SAC on CartPole and LunarLander, achieving higher fitness with smaller variance, and hence better sample efficiency. On Hopper and BipedalWalker, PPO and SAC remain strong and outperform both NEAT and NEOL. However, NEOL is still more competitive against PPO than standard NEAT. Overall, these results highlight the potential of NEOL, especially on tasks where fast within-episode adaptation is beneficial.

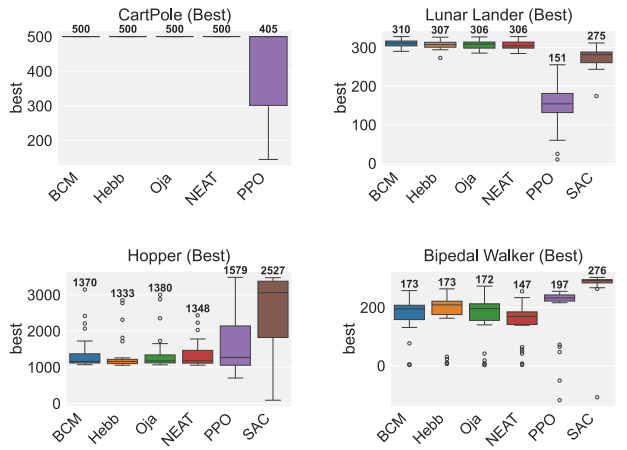

Figure 3: Fitness comparison of NEAT-NEOL with NEAT and RLs across four control tasks in 30 independent runs under a fixed interaction budget of 10 million environment steps per seed. For neuroevolution methods, the total budget for each configuration equals $P \times G \times T_{\max} = 10^7$; for PPO/SAC, it equals the total training timesteps. Boxplots show episode return aggregated over seeds.

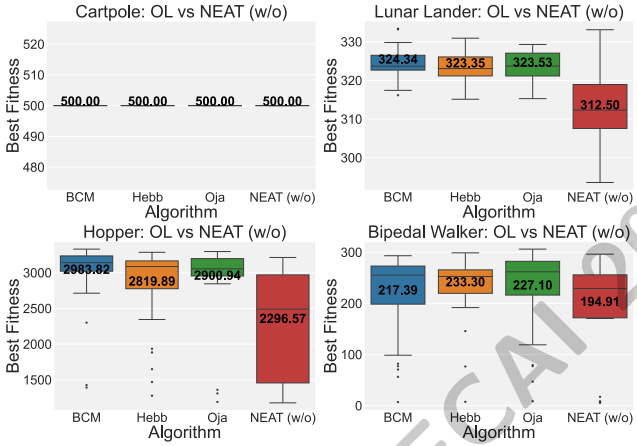

Figure 4: Ablation study comparing NEAT-NEOL with NEAT (w/o) across four control tasks in 30 independent runs.

5.3 Ablation Studies

We ablate the online learning component by disabling weight modulation in NEAT ("NEAT (w/o)"), and all other settings remain unchanged. Figure 4 compares the final best-fitness distributions over 30 independent runs on four control tasks. On CartPole, all methods quickly reach the performance ceiling 500 as expected, so disabling online learning has no clear effect. On Lunar Lander, all NEOL variants (BCM, Hebb, Oja) achieve clearly higher final fitness than NEAT (w/o), with tighter distributions. On Hopper, removing online learning results in a significant drop in performance. NEAT (w/o) has a much lower median and a wider spread, while the NEOL variants remain stronger overall. On Bipedal Walker, NEOL again improves the final fitness compared with NEAT (w/o), and the results are consistent across seeds. In three continuous control tasks, the boxplots further indicate higher empirical means, higher medians, and tighter interquartile ranges, which support the improvement of online learning via synaptic plasticity. Overall, these results isolate the contribution of online synaptic plasticity: removing it impairs both the gained fitness and the robustness of learning.

Summary. In our two-timescale setup, the outer evolutionary loop searches for good network topologies, while the inner loop uses plasticity rules to adjust weights during online interaction. This gives a simple, gradient-free method to assign fitness and fine-tune behaviour within an agent's lifetime. The experiment also shows a relatively competitive performance against some strong RL baselines across several tasks. Moreover, the ablation study supports this across tasks: enabling online plasticity improves final fitness, reduces instability across runs, and leads to better sample efficiency. In contrast, removing it (NEAT (w/o)) generally results in weaker final performance and less robust learning. Taken together, these findings suggest that online learning via plasticity can improve the performance of NeuroEvolution and make it more robust.

6 Conclusion and Discussion

We proposed a general NeuroEvolutionary Online Learning (NEOL) framework, a two-timescale design that separates (i) outer-loop topology/architecture search from (ii) inner-loop, within-episode weight adaptation via reward-modulated local plasticity. Using biologically motivated rules (Hebb, Oja, and BCM), NEOL provides a simple, gradient-free mechanism for online weight adaptation. Our theoretical analysis shows that the resulting outer-inner loop guarantees an expected sublinear regret under standard boundedness and selection assumptions. This serves as a first step towards a more rigorous understanding of these hybrid online learning methods. Empirically, NEAT-NEOL improves over pure NEAT across four control benchmarks and is relatively competitive with strong RL baselines (PPO/SAC) on several tasks, with notably tighter performance distributions. Additional ablation studies and Wilcoxon rank-sum tests further confirm that online plasticity is a key opponent of the improvement.

Despite these results, NEOL has limitations. Theoretically, we only consider the exponential weight update selection for architecture search. Extending the analysis to broader classes of algorithms presents an exciting and challenging avenue. Moreover, current theoretical analysis is restricted to the local plasticity rules, including Hebb, Oja and BCM. We conjecture that sublinear regret is attained for more general local plasticity rules with bounded step sizes. Empirically, testing on more diverse environments can strengthen our empirical analysis. It is not clear how robust our online weighted adaptation is for more complex environments. Also, this paper only uses fixed learning rules. As the performance algorithm remains sensitive to hyperparameters and task structure, adaptation of learning rules might be the next possible direction. Together, these future theoretical and empirical extensions would clarify when and why reward-modulated plasticity most effectively enhances neuroevolution, and would move NEOL closer to robust adaptive agents.

Ethics Statement

We have no ethical concerns to declare.

Acknowledgments

We thank the anonymous reviewers for their helpful reviews. Shishen is supported by UKRI via grant EP/Y014650/1, as part of the ERC Synergy project OCEAN. The computations were performed using the University of Warwick high-performance computing (Avon HPC) service.

Contribution Statement

Shishen Lin and Yixin Chen contributed equally to this work. Shishen Lin contributed to the theoretical analysis, experiment design, empirical evaluation and manuscript writing. Yixin Chen contributed to the algorithm design, implementation, empirical evaluation, and manuscript writing.

References

[Agogino *et al.*, 2000] Adrian Agogino, Kenneth Stanley, and Risto Miikkulainen. Online interactive neuro-evolution. *Neural Processing Letters*, 11(1):29–38, 2000.

[Angeline *et al.*, 1994] Peter J. Angeline, Gregory M. Saunders, and Jordan B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(1):54–65, 1994.

[Bauschke and Combettes, 2017] Heinz H. Bauschke and Patrick L. Combettes. *Correction to: Convex analysis and monotone operator theory in Hilbert spaces*, pages C1–C4. Springer International Publishing, Cham, 2017.

[Bellman, 1966] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[Bienenstock *et al.*, 1982] Elie L. Bienenstock, Leon N. Cooper, and Paul W. Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience*, 2(1):32–48, 1982.

[Brockman *et al.*, 2016] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

[Caporale and Dan, 2008] Natalia Caporale and Yang Dan. Spike timing–dependent plasticity: a hebbian learning rule. *Annual Review of Neuroscience*, 31:25–46, 2008.

[Chalumeau *et al.*, 2023] Felix Chalumeau, Raphael Boige, Bryan Lim, Valentin Macé, Maxime Allard, Arthur Flajolet, Antoine Cully, and Thomas Pierrot. Neuroevolution is a competitive alternative to reinforcement learning for skill discovery. In *ICLR*, 2023.

[Co-Reyes *et al.*, 2021] John D. Co-Reyes, Yingjie Miao, Daiyi Peng, Esteban Real, Quoc V. Le, Sergey Levine, Honglak Lee, and Aleksandra Faust. Evolving reinforcement learning algorithms. In *ICLR*, 2021.

[Fischer *et al.*, 2023] Paul Fischer, Emil Lundt Larsen, and Carsten Witt. First steps towards a runtime analysis of neuroevolution. In *FOGA*, pages 61–72, 2023.

[Florian, 2007] Răzvan V. Florian. Reinforcement learning through modulation of spike-timing–dependent synaptic plasticity. *Neural Computation*, 19(6):1468–1502, 2007.

[Freund and Schapire, 1997] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[Frémaux and Gerstner, 2016] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity and theory of three-factor learning rules. *Frontiers in Neural Circuits*, 9:85, 2016.

[Gerstner *et al.*, 2018] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. *Frontiers in Neural Circuits*, 12:53, 2018.

[Haarnoja *et al.*, 2018] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint*, 2018.

[Hausknecht *et al.*, 2014] Matthew Hausknecht, Joel Lehman, Risto Miikkulainen, and Peter Stone. A neuroevolution approach to general atari game playing. *IEEE Transactions on Computational Intelligence and AI in Games*, 6(4):355–366, 2014.

[Hebb, 1949] Donald O. Hebb. *The organization of behavior: a neuropsychological theory*. John Wiley & Sons, New York, 1949.

[Hoeffding, 1963] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[Holland, 1992] John H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Complex Adaptive Systems. MIT Press, Cambridge, MA, 1992.

[Khadka and Tumer, 2018] Shauharda Khadka and Kagan Tumer. Evolution-guided policy gradient in reinforcement learning. *NeurIPS*, 31, 2018.

[Khan *et al.*, 2010] Maryam Mahsal Khan, Gul Muhammad Khan, and Julian F. Miller. Evolution of neural networks using cartesian genetic programming. In *17th IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[Kohl and Miikkulainen, 2009] Nate Kohl and Risto Miikkulainen. Evolving neural networks for strategic decision-making problems. *Neural Networks*, 22(3):326–337, 2009.

[Kudithipudi *et al.*, 2022] Dhireesha Kudithipudi, Mario Aguilar-Simon, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

[Lv *et al.*, 2024] Zeqiong Lv, Chao Bian, Chao Qian, and Yanan Sun. Runtime analysis of population-based evolutionary neural architecture search for a binary classification problem. In *GECCO*, GECCO '24, pages 358–366, New York, NY, USA, 2024. Association for Computing Machinery.

[Lv *et al.*, 2025] Zeqiong Lv, Chao Qian, Yun Liu, Jiahao Fan, and Yanan Sun. Runtime analysis of evolutionary NAS for multiclass classification. In *ICML*, volume 267 of *Proceedings of Machine Learning Research*, pages 41648–41666. PMLR, 13–19 Jul 2025.

[Martin *et al.*, 2000] Stephen J. Martin, Paul D. Grimwood, and Richard G. M. Morris. Synaptic plasticity and memory: an evaluation of the hypothesis. *Annual Review of Neuroscience*, 23(1):649–711, 2000.

[Miconi *et al.*, 2018] Thomas Miconi, Kenneth O. Stanley, and Jeff Clune. Differentiable plasticity: training plastic neural networks with backpropagation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3559–3568. PMLR, 2018.

[Miikkulainen *et al.*, 2024] Risto Miikkulainen, Jason Liang, et al. Evolving deep neural networks. In *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 269–287. Elsevier, 2024.

[Miikkulainen, 2025] Risto Miikkulainen. Neuroevolution insights into biological neural computation. *Science*, 387(6735):eadp7478, 2025.

[Najarro and Risi, 2020] Elias Najarro and Sebastian Risi. Meta-learning through hebbian plasticity in random networks. In *NeurIPS*, 2020.

[Oja, 1982] Erkki Oja. A simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, 1982.

[Orabona, 2019] Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[Peng *et al.*, 2018] Yiming Peng, Gang Chen, Harith Singh, and Mengjie Zhang. Neat for large-scale reinforcement learning through evolutionary feature learning and policy gradient search. In *GECCO*, pages 490–497, New York, NY, USA, 2018. ACM.

[Schmidhuber, 1987] Jürgen Schmidhuber. Evolutionary principles in self-referential learning: on learning how to learn. Diploma thesis, Technische Universität München, Institut für Informatik, Munich, Germany, 1987.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint*, 2017.

[Shalev-Shwartz, 2011] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

[Soltoggio *et al.*, 2008] Andrea Soltoggio, Kenneth O. Stanley, and Risto Miikkulainen. Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In *ALIFE*, 2008.

[Soltoggio *et al.*, 2018] Andrea Soltoggio, Kenneth O. Stanley, and Sebastian Risi. Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks. *Neural Networks*, 108:48–67, 2018.

[Stanley and Miikkulainen, 2002a] Kenneth O. Stanley and Risto Miikkulainen. Efficient reinforcement learning through evolving neural network topologies. In *GECCO*, pages 569–577, 2002.

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

[Stanley *et al.*, 2003] Kenneth O. Stanley, Bobby D. Bryant, and Risto Miikkulainen. Evolving adaptive neural networks with and without adaptive synapses. In *CEC*, 2003.

[Stanley *et al.*, 2009] Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009.

[Stanley *et al.*, 2019] Kenneth O. Stanley, Jeff Clune, Joel Lehman, and Risto Miikkulainen. Designing neural networks through neuroevolution. *Nature Machine Intelligence*, 1(1):24–35, 2019.

[Sutton and Barto, 1998] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. MIT Press, Cambridge, MA, 1998.

[Takeuchi *et al.*, 2014] Tatsuya Takeuchi, Adrian J. Duszkiewicz, and Richard G. M. Morris. The synaptic plasticity and memory hypothesis: encoding, storage and persistence. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1633):20130288, 2014.

[Whiteson *et al.*, 2005] Shimon Whiteson, Peter Stone, Kenneth O. Stanley, Risto Miikkulainen, and Nate Kohl. Automatic feature selection in neuroevolution. In *GECCO*, pages 1225–1232, 2005.

[Xue *et al.*, 2024] Ke Xue, Ren-Jian Wang, Pengyi Li, Dong Li, Jianye Hao, and Chao Qian. Sample-efficient quality-diversity by cooperative coevolution. In *ICLR*, 2024.

[Yao, 1999] Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[Young *et al.*, 2025] Daniel Young, Olivier Francon, et al. Discovering effective policies for land-use planning with neuroevolution. *Environmental Data Science*, 4:e30, 2025.

[Zarantonello, 1971] Eduardo H. Zarantonello. Projections on convex sets in hilbert space and spectral theory: Part i. projections on convex sets: Part ii. spectral theory. In Eduardo H. Zarantonello, editor, *Contributions to Nonlinear Functional Analysis*, pages 237–424. Academic Press, 1971.

[Zinkevich, 2003] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.