The Artificial Experimentalist: Discovery and Control of Self-Organizing Phenomena with Autotelic Reinforcement Learning

Marko Cvjetko[1], Benedikt Hartl[2], Michael Levin[2,3], Clément Moulin-Frier[4], Pierre-Yves Oudeyer[1]

[1]Inria Centre at the University of Bordeaux, Bordeaux, France
[2]Allen Discovery Center at Tufts University, Medford, MA, USA
[3]Wyss Institute for Biologically Inspired Engineering at Harvard University, Boston, MA, USA
[4]Inria, INSA Lyon, CITI, UR3720, 69621 Villeurbanne, France
Corresponding Author: marko.cvjetko@inria.fr

Abstract

Existing methods for exploring cellular automata and other complex systems mostly operate in open loop: they set initial conditions, execute a full simulation, and observe the outcome, without intervening during execution. We introduce a closed-loop framework based on autotelic reinforcement learning, in which an agent autonomously samples diverse goals and learns a goal-conditioned policy to intervene in a complex system through minimal, local perturbations. We instantiate this framework on Lenia, a continuous cellular automaton known for life-like self-organizing patterns, in an agentic system we call CARL, and demonstrate three capabilities. First, CARL discovers stable solitons across a wide range of Lenia update rules at a higher rate than heuristic baselines. Second, it learns to steer the movement direction of existing solitons with few interventions, showing that CARL can control self-organizing patterns, not only create them. Third, humans can use trained agents to guide solitons through maze environments in real time by specifying high-level directional commands that the agent translates into low-level interventions. Trained across diverse goals, update rules, and random initial states, the agents acquire policies that generalize zero-shot to various out-of-distribution conditions. These results suggest a path toward artificial experimentalist agents that, autonomously or with human guidance, discover and control emergent phenomena in complex systems.

Submission type: **Full Paper**

Data/Code available at: https://developmentalsystems.org/carl/

Introduction

One of the central endeavors of science is understanding complex systems at all scales of organization, from elementary physics to astronomy, from molecular biology to ecology. Two general goals drive this research: (1) explaining and discovering the diverse phenomena that emerge in complex systems, and (2) learning to control complex systems towards desired states, ideally with minimal effort.

In biomedicine, for instance, the goal is not continuous intervention, but the restoration of healthy, self-sustaining dynamics. Rather than controlling individual components, we aim to guide systems back into stable regimes in which they can maintain their function autonomously. However, identifying such interventions is inherently challenging, as system behavior arises from interactions across many scales. This raises a fundamental question: how can we systematically control systems whose internal dynamics are complex or only partially understood? (Levin, 2025, 2023)

Computational approaches are essential for pursuing these goals, as they enable the simulation of complex systems *in silico*. Cellular automata (CAs) have long served as standard models for studying self-organization, and recent continuous extensions such as Lenia (Chan, 2020) produce increasingly complex and life-like patterns, making them ideal testbeds for this endeavor. However, most existing approaches for exploring CAs operate in open loop: parameters and initial conditions are chosen upfront, with no interaction during rollout (see Related Work for a detailed discussion).

This stands in contrast to how humans typically engage with complex systems: continually observing and interacting with the system in real time to form intuition of its causal dynamics (e.g. a gardener continuously pruning, watering and reshaping a garden as it grows). As systems grow in complexity, however, effective intervention becomes increasingly difficult. Nonlinear interactions, feedback loops, and delayed effects make human intuition prone to systematic biases, and modern challenges — particularly in biomedical, ecological, or economic contexts — quickly reach the limit of human modeling capabilities. (Tversky and Kahneman, 1974)

To address these gaps, we propose an autotelic reinforcement learning (RL) framework as an interaction-driven approach to translate desired experimental outcomes into actionable, causally effective interventions in complex systems. By *autotelic*, we mean a RL agent autonomously

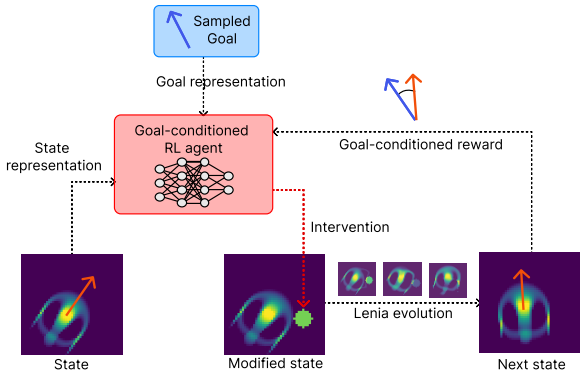

Figure 1: CARL: a goal-conditioned RL agent that learns to drive the evolution of CAs towards desired states. The agent perceives the CA state and the goal, and makes an intervention, after which the CA evolves. This process is repeated multiple times to form an episode.

learning how to achieve diverse goals in the considered complex system. As a concrete instantiation, we introduce **CARL** (for *controlling **C**ellular **A**utomata with **R**einforcement **L**earning*), which learns to intervene on CAs in a closed-loop fashion, continuously observing and shaping their dynamics over time (Fig. 1).

Through a series of experiments using Lenia as a testbed, we demonstrate that CARL can both discover interesting self-organizing phenomena and control their behavior, with limited interventions and in a sample-efficient manner. Moreover, CARL generalises well to out-of-distribution scenarios, such as unseen world dynamics and modified action spaces. Lastly, we show that trained CARL agents can be deployed in real time, enabling humans to specify high-level goals while the agent translates them into low-level interventions for controlling complex systems.

While the framework is demonstrated on Lenia, it is designed to be system-agnostic. Looking ahead, we hope to extend it to increasingly biologically grounded models.

Related Work

AI for Scientific Discovery

AI is playing a growing role in scientific discovery. It has already driven breakthroughs in domains such as protein structure prediction and combinatorial optimization (Jumper et al., 2021; Fawzi et al., 2022), and recent work aims to go further by automating the scientific method end-to-end, from hypotheses to publication (Lu et al., 2026; Zenil et al., 2026). A key question in these efforts is how to design agents that can autonomously decide what to investigate and how. The autotelic AI paradigm (Colas et al., 2022), in which agents set their own goals and learn to achieve them, offers a natural framework for this — capturing the core loop of scientific inquiry: formulating questions and developing the skills to answer them. Autotelic AI has been successfully applied in scientific contexts, including the automated discovery of protocell behaviors (Grizou et al., 2020) and the exploration of gene regulatory networks (Etcheverry et al., 2025).

Cellular Automata (CAs)

CAs are dynamical systems consisting of grids of cells whose states are updated based on local neighborhoods. Despite their simplicity, CAs can produce remarkably complex phenomena, making them used both as models of real-world processes (in ecology, urban development, physics) and objects of study in their own right. Foundational contributions include the work of Turing (1952), Von Neumann and Burks (1966), Barricelli (1962, 1963), Langton (1986), and Wolfram (1983), who studied morphogenesis, self-replication, evolution, and complexity. More recently, CAs have seen a revival with prominent continuous models such as Lenia and Neural Cellular Automata (Chan, 2019, 2020; Mordvintsev et al., 2020), and extensions incorporating mass conservation that promote evolutionary phenomena (Plantec et al., 2025; Papadopoulos and Guichard, 2025). These more expressive CAs have garnered broad interest since they give rise to a plethora of self-organizing patterns and dynamics that behave increasingly life-like, resembling artificial organisms and ecosystems (Hartl et al., 2025).

Automated Exploration of Cellular Automata

Over the years, many methods have been used to illuminate the range of possible CA behaviors, including random search, manual tuning, and hand-crafted heuristics. More recently, gradient-based methods have been used to optimize for specific phenomena (Mordvintsev et al., 2020; Miotti et al., 2025; Hamon et al., 2025), while diversity-driven algorithms — including novelty search, quality-diversity, and intrinsically motivated goal exploration processes (IMGEPs) — aim to discover a variety of distinct behaviors (Reinke et al., 2020; Etcheverry et al., 2020; Faldor and Cully, 2024; Khajehabdollahi et al., 2025; Michel et al., 2025). All of these methods, however, operate in open loop, with no ability to intervene during execution.

Some works have moved beyond this limitation. Rainwater (2024) studies how external forces can impact the dynamics of Game of Life. Kumar et al. (2025) optimize CA rules at specific checkpoints during evolution, though these are planned in advance rather than chosen reactively. Sánchez-Fibla et al. (2024) train agents in a CA reproducing forest-fire dynamics to manage resource acquisition with environmental extremes. Earle and Togelius (2024) train embodied agents in evolvable CA-based game environments.

Method

General Framework

We formalize a framework based on autotelic reinforcement learning for discovering and controlling phenomena in com-

plex systems. In the first phase, an autotelic agent is trained to achieve a diversity of goals; in the second, the learned goal-conditioned policy serves as a high-level interface to produce self-organizing patterns and control them in real time. The framework is designed to be general, abstracting away the specifics of any particular system. Instantiating it requires defining three components: a *complex system* whose dynamics are to be studied, an *intervention space* through which a goal-conditioned RL policy can intervene in the system, and a *task specification* that defines the goals the policy must learn to achieve.

Complex system. We define a complex system as a tuple $(\mathcal{S}, F)$, where $\mathcal{S}$ is a state space and $F : \mathcal{S} \to \mathcal{S}$ is an update rule that governs the system's dynamics. At each discrete time step, the system evolves as $\mathbf{s}^{t+1} = F(\mathbf{s}^t)$. We make no assumptions about F beyond the ability to simulate it; it may be deterministic or stochastic, continuous or discrete, and may operate over spatial grids, graphs, particle systems, or other structures.

Intervention space. An intervention is a modification to the system state. We define an intervention function $\alpha : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, where $\mathcal{A}$ is the set of available actions. Each action $a \in \mathcal{A}$ produces a perturbation to the current state. The action space can optionally include a *no-op* action, leaving the system unmodified. Crucially, interventions are intended to be small relative to the system — the agent nudges the system rather than rewriting the whole state.

Task specification. A task is defined by a goal space $\mathcal{G}$ and a goal-conditioned reward function $r : \mathcal{S}^{\leq T} \times \mathcal{G} \to \mathbb{R}$, where $\mathcal{S}^{\leq T}$ denotes sequences of states of length $\leq T$. At the start of each training episode, a goal $g \sim p(\mathcal{G})$ is sampled from a predefined distribution. The reward $r(\mathbf{s}^{0:t}, g)$ measures the degree to which the behavior of the system aligns with the goal. Importantly, goals are not limited to a single state: they can include properties of the trajectory, the system's update rules, and constraints on intervention effort (e.g. adding action costs). By conditioning on goals sampled from this rich space, a policy must generalize across diverse objectives.

The loop. Given a complex system $(\mathcal{S}, F)$, an intervention function α, and a set of tasks $(\mathcal{G}, r)$, we can train a goal-conditioned policy $\pi : \mathcal{S}^{\leq T} \times \mathcal{G} \to \mathcal{A}$ through episodic reinforcement learning (Fig. 1). Each episode of length T proceeds as follows:

The objective is to train a goal-conditioned action policy π able to maximize cumulative reward for any goal $g \in \mathcal{G}$. During inference, the goal can change dynamically $g \mapsto g^t$, which enables a human user to control the complex system in real time by issuing high-level commands that the trained policy translates into low-level interventions.

Algorithm 1 General Framework Loop

for $n = 0, \ldots, n\_episodes - 1$ **do**
  Sample goal $g \sim p(\mathcal{G})$ and initial state $\mathbf{s}^0$
  for $t = 0, \ldots, T - 1$ **do**
    Observe $\mathbf{s}^{t-\Delta t:t}$ and g over a time interval Δt
    Select $a^t \sim \pi(\cdot \mid \mathbf{s}^{t-\Delta t:t}, g)$
    Apply intervention: $\tilde{\mathbf{s}}^t = \alpha(\mathbf{s}^t, a^t)$
    Evolve system for N steps: $\mathbf{s}^{t+1} = F^N(\tilde{\mathbf{s}}^t)$
    Receive reward $r(\mathbf{s}^{0:t+1}, g)$
  end for
  (Optional) Roll out $\mathbf{s}^{T+m} = F(\mathbf{s}^{T+m-1})$ for $m = 1, \ldots, M$ to assess resulting phenomena
end for

Instantiation: Lenia

We instantiate CARL (Fig. 1) on Lenia, a continuous generalization of Conway's Game of Life (Chan, 2019, 2020). Lenia produces a rich variety of self-organizing phenomena from simple update rules, making it an ideal testbed for our framework.

Lenia $(\mathcal{S}, F)$**.** The state is a grid $\mathbf{X}^t \in [0,1]^{H \times W}$ with periodic boundary conditions. The update rule is:

$$\mathbf{X}^{t+dt} = \left[\mathbf{X}^t + dt\, \varphi(\mathbf{K} * \mathbf{X}^t)\right]_0^1 \tag{1}$$

where $\mathbf{K}$ is a convolutional kernel, φ is an element-wise growth function, and dt is the step size.

The kernel is defined over a disk of radius ρ, partitioned into a number of $b = |\boldsymbol{\beta}|$ concentric rings of equal width with peak values $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_b)$. For a point at normalized distance r, the ring index is $i = \min(\lfloor b \cdot r \rfloor, b - 1)$ and the local coordinate is $r' = (b \cdot r) \bmod 1$. The unnormalized kernel is:

$$\tilde{K}(r) = \beta_i \cdot \left(4\, r'(1 - r')\right)^4 \tag{2}$$

and the final kernel is normalized: $\mathbf{K} = \tilde{K} / \sum \tilde{K}$. The growth function maps the convolution output to $[-1, 1]$ via a Gaussian bump:

$$\varphi(u) = 2 \exp\left(-\frac{(u - \mu)^2}{2\sigma^2}\right) - 1 \tag{3}$$

A key phenomenon of interest in Lenia are ***solitons***: localized patterns that persist and often move across the grid. We design our experiments with the intent of showing that CARL can discover new solitons and control their behavior across a diversity of update rules, across a range of action costs, and from randomized initial states. To detect solitons, we apply a simple filter inspired by prior work (Hamon et al., 2025; Faldor and Cully, 2024): after running the Lenia update rule for 5000 steps without intervention, we classify the resulting state as containing a soliton if its total mass is non-zero and is below 10% of the grid capacity.

We omit additional checks used in prior work — such as temporal stability and robustness testing — since our goal is to find cases when CARL solves the task by creating local, persistent patterns of any kind.

Interventions over Lenia $(\mathcal{A}, \alpha)$**.** An action is a tuple $a^t = (x^t, y^t, \delta^t)$, where (x^t, y^t) are spatial coordinates and $\delta^t \in \{-1, 0, +1\}$ indicates whether to remove mass, take no action, or add mass. An action modifies all cell values within a radius R_a around (x^t, y^t) by $(\delta^t \cdot M_a)$, with fixed hyperparameters R_a and M_a. Values are clipped to $[0, 1]$. In all experiments, we use $R_a = 5$ and $M_a = 0.3$ unless stated otherwise.

Policy Architecture and Training

We train policies using Double Deep Q-Networks van Hasselt et al. (2016). The network architecture is a U-Net Ronneberger et al. (2015), a fully convolutional network that produces dense Q-value maps over the grid for each action type (add, remove, no-op). This architecture naturally mirrors the spatial structure of CAs: it treats each cell locally while maintaining receptive fields large enough to perceive most or all of the grid. Similar architectures have been used in robotics (Zeng et al., 2018; Wu et al., 2020).

Observation consists of last 4 stacked Lenia grid states. Additional context — including the goal, action cost coefficient, current episode time step, and Lenia update rule parameters — is provided through FiLM conditioning layers Perez et al. (2018).

We chose an off-policy algorithm for its sample efficiency. Full details of the RL algorithm hyperparameters and network architecture are provided in the code repository.

Experiments

We demonstrate CARL through three sets of experiments. First, we show that it can create stable solitons across a wide range of Lenia update rules and from procedurally generated initial states, and that trained agents generalize zero-shot to unseen update rules and modified action spaces. Second, we train an agent to steer the movement direction of existing solitons, demonstrating that CARL can control self-organizing patterns, not only create them. Third, we show that humans can interact with complex systems through CARL agents by modifying their goals in real time. We illustrate this by having users navigate solitons through a maze using the movement direction agent.

We invite the reader to follow experimental results on the companion website, which contains many video examples.

Soliton Creation Task

Rather than searching for solitons directly — which would require defining what constitutes a soliton within the reward signal — we train CARL on a simpler proxy task: maintaining a target mass on the Lenia grid, for a given update rule and action cost. When actions are costly, the agent faces a choice between constantly intervening to hold mass at the target, or finding a self-sustaining configuration that matches it. Action costs tip the balance toward the latter, making soliton creation an emergent byproduct of reward maximization rather than an explicit objective.

Goal space and reward. The goal space is defined as $\mathcal{G} = \mathcal{T} \times \mathcal{C} \times \Omega$, where $\mathcal{T}$ is the set of target masses, $\mathcal{C}$ the set of action costs, and Ω the set of update rules. At the start of each episode, CARL samples a goal $g = (\tau, c, \omega)$ uniformly, with $\tau \in [0, 200]$, $c \in [0, 0.2]$, and ω drawn from a set of training update rules. Sampling the action cost per episode rather than fixing it serves two purposes: it produces a single budget-adaptive policy that can operate across a spectrum of intervention regimes at deployment, and we speculate it also acts as an exploration mechanism during training — low-cost episodes allow the agent to freely discover viable configurations, while high-cost episodes pressure it to find self-sustaining ones. The reward at each step is:

$$r = -\sqrt{\frac{|M^t - \tau|}{N}} - c \cdot \mathbf{1}[\delta^t \neq 0], \tag{4}$$

where $M^t = \sum_{i,j} X^t_{i,j}$ is the total mass of the grid at time t, N is the total number of grid cells, and $\delta^t \in \{-1, 0, +1\}$ is the action type selected by the agent. The first term penalizes deviation from the target mass, while the second penalizes non-trivial interventions, weighted by the sampled action cost c. A single policy must therefore learn to act across diverse combinations of target masses, action costs, update rules, and initial conditions.

Experimental setup. Initial states are procedurally generated by randomly applying 20 actions to an empty grid without rolling out the CA in between, producing diverse unstructured configurations. Each episode step consists of an agent's action followed by a single Lenia update step ($N_{\text{steps}} = 1$). Episodes last 150 steps on a 64×64 grid. The model is trained on 2 000 000 episode step transitions. The training set of update rules consists of 85 hand-selected rules supporting diverse solitons discovered by Hudcová et al. (2026)[1]. A detailed description of the hyperparameters and included update rules will be found in the code repository.

Mass tracking evaluation We evaluate the agent on the Cartesian product of target masses $\tau \in \{0, 25, 50, \ldots, 400\}$, action costs $c \in \{0, 0.05, 0.10, \ldots, 0.40\}$, and all 85 training update rules $\omega \in \Omega_{\text{train}}$. For each triplet (τ, c, ω), we run 16 episodes, yielding a three-dimensional evaluation grid. Both τ and c extend to twice their training range. Fig. 2 shows projections onto the (τ, c) plane, with metrics averaged over update rules.

[1] Companion website of Hudcová et al. (2026)

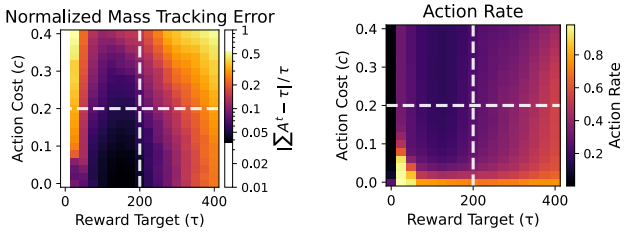

Figure 2: Agent behavior across different action costs and mass targets. Each cell is averaged over all training update rules. The bottom-left quadrants represent τ and c values seen during training. Left: mass tracking error. Right: action rate (fraction of steps where the agent intervenes).

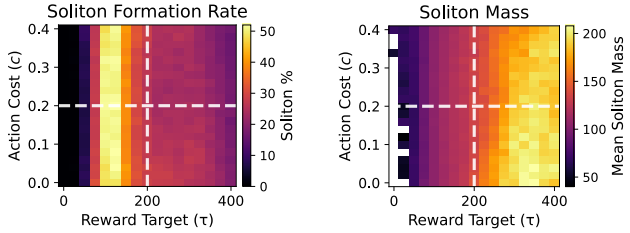

Figure 3: Soliton creation metrics across different target masses and action costs, averaged over all training update rules. The bottom-left quadrants represent and c values seen during training. Left: percentage of episodes resulting in solitons. Right: average mass of the generated solitons.

The agent tracks the target mass reliably across most of the evaluation range (Fig. 2, left). Performance degrades at boundary values of τ, particularly when action costs are high. This is expected: at both extremes, the system lacks stable self-sustaining configurations — mass dissipates at low τ and grows unboundedly at high τ — forcing the agent into costly continuous intervention. The action cost also shapes the behavior as intended: when $c > 0$, the agent acts less frequently and relies more on the intrinsic dynamics of the system (Fig. 2, right).

Soliton creation We now turn to the central question: does the agent produce solitons? To test this, we take the final Lenia grid state from each evaluation episode above, roll it out for $M = 5000$ CA steps without any agent intervention, and apply the soliton filter.

We observe that the agent produces solitons at a high rate, especially for target masses between $\tau = 100$ and 150 (Fig. 3, left), and that the mass of created solitons correlates well with the target (Fig. 3, right). For very low target masses, almost no episodes yield solitons, consistent with the observation above that such masses cannot persist without constant intervention. For high targets, the agent discovered an interesting strategy: creating several independent solitons whose combined mass matches the target. We observe that action costs have little impact on the soliton formation rate, however, we speculate that this mechanism was crucial during training.

Comparison with baselines. We compare CARL against several heuristic baselines across all training update rules, with a fixed action cost of $c = 0.1$ and mass targets $\tau \in \{50, 100, 150, 200\}$. The baselines include: **No-op** (always selects no-op), **Random** (random action type and location), **Mass-based** (adds/removes mass toward target, placed randomly or at locations of existing mass), and **Mass-based with deadzone** (same, but only acts when mass deviates by more than 10% from target). Both mass-based variants are tested with random and proximity placement.

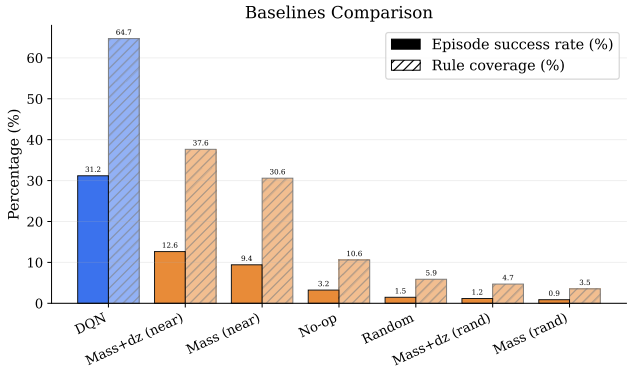

Figure 4: CARL compared to baseline methods for soliton creation, averaged across all training update rules with action cost $c = 0.1$ and mass targets $\tau \in \{50, 100, 150, 200\}$.

CARL outperforms all baselines both in the overall soliton creation rate and in the number of update rules for which at least one soliton is generated (Fig. 4). The gap is particularly notable against the mass-based heuristics, which have access to the same mass information as CARL but lack spatial awareness: they cannot learn where to place mass to seed a viable pattern. The no-op baseline confirms that solitons rarely arise from random initial conditions alone, underscoring that the agent's interventions are essential.

Generalization The results above show that CARL reliably creates solitons under training conditions. We now assess how robust this capability is by testing three axes of generalization: modified action parameters, rescaled update rule kernels, and entirely novel update rules.

Modified action parameters. We evaluate the agent when the action hyperparameters — R_a and M_a — are changed at test time. Importantly, the agent does not observe these hyperparameters. We fix $\tau = 125$ and $c = 0.1$, conditions that produce solitons reliably under training settings. The agent adapts well to combinations where $R_a \cdot M_a \approx R_a^{\text{train}} \cdot M_a^{\text{train}}$,

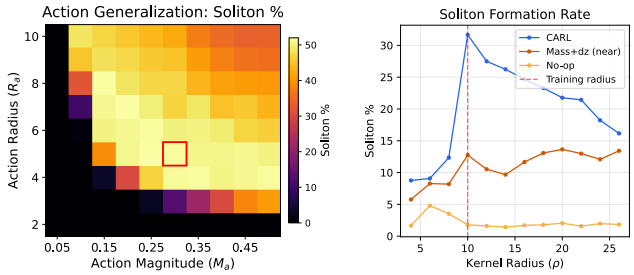

Figure 5: Soliton creation rate under modified parameters. Left: varying action radius and magnitude; the red square marks training conditions. Right: Soltion creation rate for varying kernel radius ρ contrasting CARL with baselines.

i.e. when the integrated effect of each action matches that of the training conditions (Fig. 5, left), and degrades gracefully away from this curve. Performance drops to zero only for low-impact hyperparameters, where individual actions are too weak to seed or sustain any mass on the grid.

Rescaled kernel radius. We evaluate whether the agent can create solitons when the update rule kernel radius ρ is rescaled. This is a particularly challenging form of generalization: rescaling the kernel does not simply scale the emerging patterns, but can fundamentally alter their behavior due to discretization effects. The same update rule at different kernel radii can produce solitons with different shapes, sizes, movement patterns, and robustness.

We deploy the agent across kernel radii $\rho \in \{4, 6, 8, \ldots, 26\}$, all training update rules, target masses $\tau \in \{50, 100, 150, 200\}$, and action costs $c \in \{0.1\}$. When rescaling ρ, we proportionally adjust the action radius and grid size (linearly) and the target masses (quadratically), keeping the ratio between action scale, target mass, and pattern size roughly constant. This isolates the effect of rescaled dynamics on agent performance. All rescaled settings produce out-of-distribution values. Since the policy networks are fully convolutional, they can be deployed on different grid sizes without modification.

The agent adapts well to scaled Lenia worlds, particularly for up-scaled kernels (Fig. 5, right). Although performance drops compared to the training radius, the agent still creates solitons at a relatively high rate, even for kernels double or half the training size. Across all tested radii, the agent's soliton creation rate remains well above the no-op baseline and comparable to or above the best heuristic baseline evaluated at the training radius. The success rate drops more sharply for down-scaled kernels, likely due to discretization effects.

Novel update rules. Finally, we deploy CARL on unseen convolutional kernels. We select *seven* convolutional kernels $\mathbf{K}$ not included in the training set, and sweep across growth function parameters $\mu \in \{0.2, 0.205, \ldots, 0.4\}$ and $\sigma \in \{0.02, 0.022, \ldots, 0.06\}$ with $\tau \in \{50, 100, 150, 200, 250\}$ and $c = 0.1$. We conduct these grid searches for kernel radii $\rho \in \{10, 14, 18\}$, rescaling settings as was done in the previous evaluation.

We aggregate results and visualize the likelihood of creating solitons across the update rule spaces (Fig. 6).

The results show that the trained agents can efficiently map novel update rule spaces, identifying which regions of (μ, σ) space support soliton formation. The agent discovers solitons across a range of unseen kernels, though the success rate varies considerably depending on the kernel — some kernels admit large regions of soliton-supporting parameters, while others are more restrictive. This suggests that a trained CARL agent can serve as a practical tool for rapid exploration of new Lenia update rules for downstream tasks.

Soliton Direction Task

To showcase that CARL can control self-organizing phenomena, we create a task of directing soliton towards a target direction. Each episode is initialized with a uniformly sampled target direction vector, an action cost (as before), and a soliton (from a set of 48 solitons found by the mass agent during the kernel scaling generalization test, for $\rho = 18$). The solitons are placed on a grid and randomly rotated. The reward combines two terms: (1) cosine similarity between the center-of-mass displacement over last 4 timesteps and the target direction, and (2) a mass penalty as in the previous task, where the target mass is that of the initial state. We train the agent for 1 000 000 episode step transitions.

We evaluate generalisation along two axes: *solitons* and *directions*. The 48 solitons are split into 24 training and 24 held-out (stratified across (μ, σ)), and the unit circle of target directions is partitioned into four quadrants, of which only two opposite ones are used during training. This yields a 2×2 generalisation grid with 128 episodes per cell.

Fig. 7 shows the mean cosine similarity between the soliton's center-of-mass displacement and the target direction, averaged over all 200 episode steps across the four conditions: On training solitons and training directions, the agent achieves a mean cosine similarity of 0.91 ± 0.14, indicating strong directional alignment. Performance degrades gracefully under generalisation: to 0.82 ± 0.14 for unseen directions (training solitons), 0.91 ± 0.10 for unseen solitons (training directions), and 0.76 ± 0.15 when both are held out. Notably, generalisation to unseen solitons within training directions is nearly lossless, suggesting that the steering policy captures direction-dependent strategies that transfer across soliton morphologies. The larger drop for held-out directions indicates that the mapping from direction to intervention pattern is partially learned.

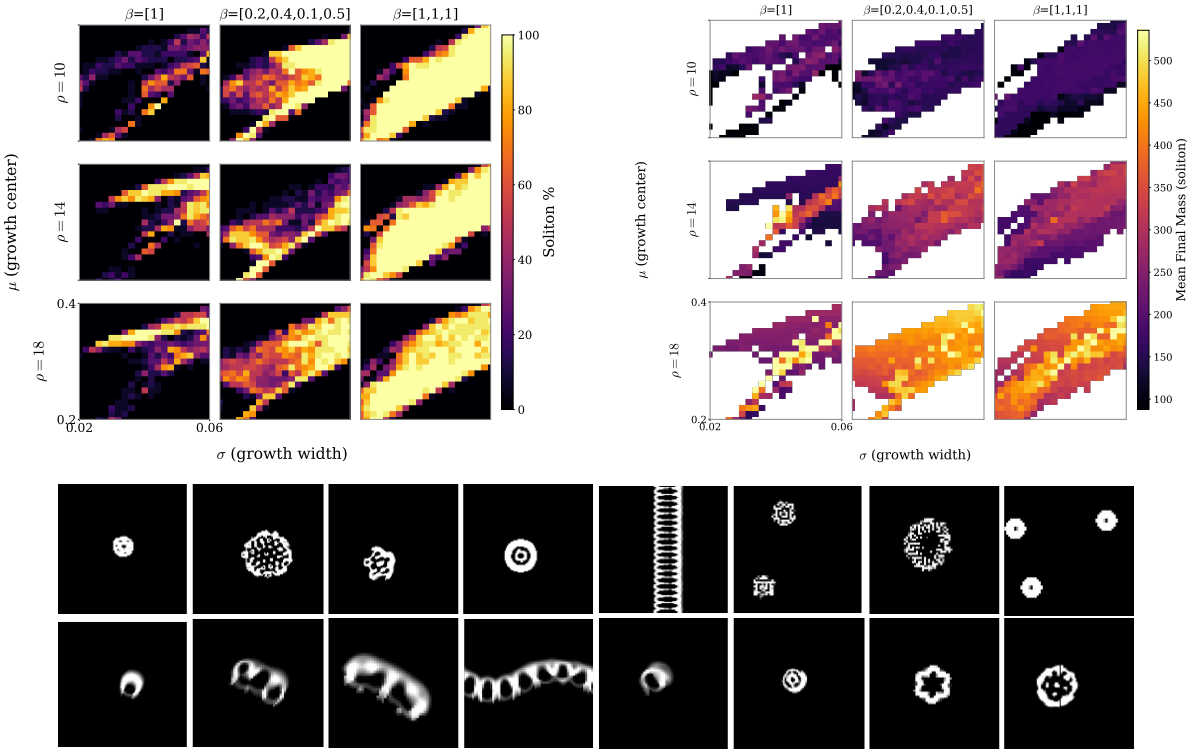

Figure 6: Soliton formation rate maps for novel update rules. Left: likelihood of soliton creation as a function of growth function parameters (μ, σ), for a fixed kernel β and radius ρ. Right: mean soliton mass under the same conditions. Bottom: examples of solitons discovered in novel update rules.

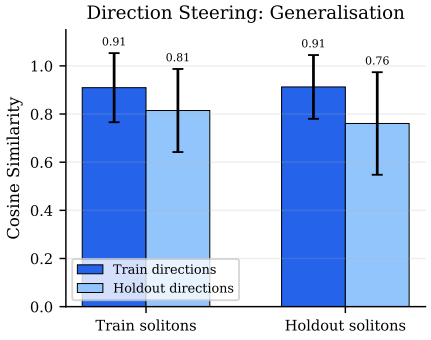

Figure 7: Generalisation of the direction-steering agent. Mean cosine similarity between soliton movement and target direction, evaluated on a 2×2 grid of {train, holdout} solitons $\times$ {train, holdout} directions. Error bars show one STD across 128 episodes per cell.

Human-in-the-Loop

We demonstrate how trained CARL agents can serve as real-time interfaces for human control. We extend the direction environment with procedurally generated mazes, where walls are regions in which cell values are fixed to zero. A soliton is placed in the maze and the user can modify the agent's perceived target direction and action cost in real time, steering the soliton through the maze (Fig 8).

The agent has no explicit representation of the maze — it perceives only the single-channel Lenia grid, identical to its training setting. Furthermore, the agent was never trained with dynamically changing goals, yet it successfully redirects solitons multiple times within a single episode while preserving their coherent shape. Reducing the action cost makes the agent intervene more frequently and advance faster, giving the user an intuitive speed-precision trade-off.

Despite generalizing well, several failure modes emerge. Wall collisions can cause the soliton to disintegrate or explode, though some solitons are robust to contact. High action costs can also lead to failure, as interventions become too weak to maintain the soliton's shape after perturbations, and the agent generally cannot recover a disrupted pattern.

Discussion

We introduced a general framework for autonomous discovery and control of self-organizing phenomena in complex systems, based on autotelic RL. Rather than setting initial conditions and passively observing outcomes, a goal-conditioned policy learns to observe the evolving state of a system and apply minimal, local perturbations to steer it toward diverse goals. We instantiated this framework on

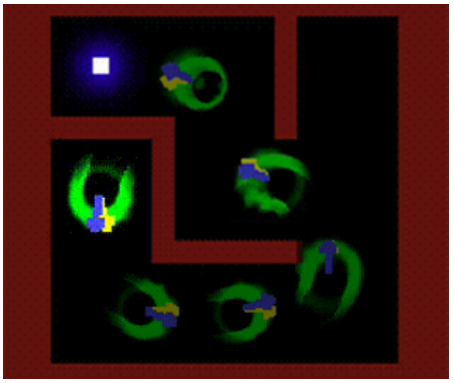

Figure 8: A timelapse of a soliton being guided through a maze by a human controlling the goals of a movement direction agent in real-time. The yellow and blue arrows represent target and current movement directions, respectively. Note that the soliton preserves well its original shape throughout the episode.

Lenia, a continuous cellular automaton, in a system named CARL. We showed that CARL discovers solitons across a wide range of update rules and procedurally generated initial states, generalizes zero-shot to various out-of-distribution conditions, and can efficiently map novel update rule spaces to identify regions that support soliton formation. Beyond discovery, CARL agents can also learn to control solitons by steering their movement direction. Finally, we demonstrate that trained agents can serve as real-time interfaces, enabling human users to guide solitons through maze environments with simple directional commands.

A key design choice is the use of action costs, which penalize every intervention and force the agent to act sparsely, reflecting the principle that effective control of self-organizing systems should work alongside the system's intrinsic dynamics, not against them. In the mass tracking task, action costs lead the agent to discover self-sustaining solitons as a side effect of reward maximization — the cheapest way to maintain a target mass is to find a configuration that maintains itself. In the direction task, they produce a similar effect: instead of continuously micromanaging the soliton's trajectory, the agent learns to apply a brief perturbation that redirects it, then withdraws, allowing the soliton to continue along the new heading unassisted.

CARL demonstrates strong generalization to out-of-distribution conditions across variations in mass targets, action parameters, system dynamics, and novel updates rules. This suggests the policies capture transferable system dynamics rather than overfitting. As a result, trained policies can be reused and composed to solve tasks beyond their original training objective. We demonstrate this through a maze-navigation task, requiring both soliton creation and adaptive redirection. A human user modifies the agent's goal (e.g., desired movement direction), while the policy handles the low-level control to achieve it in real time. This compositional reuse points toward functional integration, where distinct capabilities can be combined to solve increasingly complex tasks. Such integration suggests a path toward hierarchical control, in which higher-level agents or processes set subgoals for lower-level controllers. We see CARL as a first step toward artificial experimentalist frameworks, where agents not only learn how to autonomously act on complex systems, but also how to structure and combine those actions symbiotically through combinatorial repurposing—autonomously deciding *what* to investigate through self-generated goals and *how* to achieve it.

A key limitation is that instantiating the framework requires domain expertise: the reward function, action space, and observation design all encode knowledge about what makes a given system interesting. While the mass tracking objective sidestepped the need to define solitons explicitly, it still reflects a designer's intuition about Lenia. Domain expertise is inherent to scientific inquiry, but when the goal is to uncover phenomena we cannot yet characterize or anticipate, more open-ended approaches, such as intrinsic reward signals or LLM-driven environment and task design, could reduce this dependence and broaden the scope of discovery. More fundamentally, Lenia offers favorable conditions — full observability, determinism, and a simple action space — that will not hold in the biological and biomedical domains where closed-loop control of self-organizing systems could have the greatest impact. Extending to partial observability, stochasticity, and high-dimensional action spaces is the central challenge for future work.

This work has relevance to regenerative medicine and bioengineering in two ways. First, biological pattern formation is generally irreversible, posing an inverse problem: what low-level interventions produce a desired large-scale anatomical outcome, such as organ regeneration, cancer reprogramming, or aging (Lobo et al., 2014; Pio-Lopez et al., 2025)? Our results provide a proof-of-concept computational pipeline that infers effective stimuli to achieve target outcomes in a complex emergent system (Davies and Levin, 2023). Second, the control of soliton behavior connects to an emerging perspective on health and disease (Levin, 2025). Most biomedical efforts focus on tangible targets (proteins, genes, circuits) but the body's multi-scale hierarchy also comprises persistent patterns in excitable media (bioelectric, mechanical, metabolic) that can move, grow, resist erasure, and reshape their surroundings (Fields and Levin, 2025). Prior formalisms for such dynamical objects may not capture their roles as active information patterns (Mathews et al., 2023). The solitons that CARL creates and controls are concrete instances: localized, self-maintaining, and capable of movement, yet existing only as dynamic configurations of an underlying medium. Learning to control these entities in silico could lay groundwork for doing so with their biological counterparts.

Acknowledgements

We thank Barbora Hudcová for insightful discussions and guidance in navigating the Lenia Explorer dataset. We thank members of the Flowers AI and CogSci lab and the Levin Lab for helpful discussions. We gratefully acknowledge support for this work provided through a sponsored research agreement with Astonishing Labs and from the Templeton World Charity Foundation, Inc. (Grant ID: TWCF-2021-20606). The opinions expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agencies.

References

Barricelli, N. A. (1962). Numerical testing of evolution theories: part i theoretical introduction and basic tests. *Acta Biotheoretica*, 16(1-2):69–98.

Barricelli, N. A. (1963). Numerical testing of evolution theories: part ii preliminary tests of performance. symbiogenesis and terrestrial life. *Acta Biotheoretica*, 16(3-4):99–126.

Chan, B. W.-C. (2019). Lenia: Biology of Artificial Life. *Complex Systems*, 28(3).

Chan, B. W.-C. (2020). Lenia and Expanded Universe. In *ALIFE 2020: The 2020 Conference on Artificial Life*, pages 221–229. MIT Press.

Colas, C., Karch, T., Sigaud, O., and Oudeyer, P.-Y. (2022). Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey. *J. Artif. Int. Res.*, 74.

Davies, J. A. and Levin, M. (2023). Synthetic morphology with agential materials. *Nature Reviews Bioengineering*, 1:46–59.

Earle, S. and Togelius, J. (2024). Autoverse: An Evolvable Game Language for Learning Robust Embodied Agents.

Etcheverry, M., Moulin-Frier, C., and Oudeyer, P.-Y. (2020). Hierarchically Organized Latent Modules for Exploratory Search in Morphogenetic Systems. In *Advances in Neural Information Processing Systems*, volume 33, pages 4846–4859. Curran Associates, Inc.

Etcheverry, M., Moulin-Frier, C., Oudeyer, P.-Y., and Levin, M. (2025). AI-driven automated discovery tools reveal diverse behavioral competencies of biological networks. *eLife*, 13:RP92683.

Faldor, M. and Cully, A. (2024). Toward Artificial Open-Ended Evolution within Lenia using Quality-Diversity. In *ALIFE 2024: Proceedings of the 2024 Artificial Life Conference*. MIT Press.

Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., R. Ruiz, F. J., Schrittwieser, J., Swirszcz, G., Silver, D., Hassabis, D., and Kohli, P. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53.

Fields, C. and Levin, M. (2025). Thoughts and thinkers: On the complementarity between objects and processes. *Physics of Life Reviews*, 52:256–273.

Grizou, J., Points, L. J., Sharma, A., and Cronin, L. (2020). A curious formulation robot enables the discovery of a novel protocell behavior. *Science Advances*, 6(5):eaay4237.

Hamon, G., Etcheverry, M., Chan, B. W.-C., Moulin-Frier, C., and Oudeyer, P.-Y. (2025). Discovering sensorimotor agency in cellular automata using diversity search. *Science Advances*, 11(44):eadp0834.

Hartl, B., Levin, M., and Pio-Lopez, L. (2025). Neural cellular automata: Applications to biology and beyond classical AI.

Hudcová, B., Dušek, F., Tuccio, M., and Hongler, C. (2026). Visualizing the Structure of Lenia Parameter Space. In *ALIFE 2025: Ciphers of Life: Companion Proceedings of the Artificial Life Conference 2025*. MIT Press.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589.

Khajehabdollahi, S., Hamon, G., Cvjetko, M., Oudeyer, P.-Y., Moulin-Frier, C., and Colas, C. (2025). Expedition & Expansion: Leveraging Semantic Representations for Goal-Directed Exploration in Continuous Cellular Automata. In *ALIFE 2025: Ciphers of Life: Proceedings of the Artificial Life Conference 2025*. MIT Press.

Kumar, A., Lu, C., Kirsch, L., Tang, Y., Stanley, K. O., Isola, P., and Ha, D. (2025). Automating the Search for Artificial Life With Foundation Models. *Artificial Life*, 31(3):368–396.

Langton, C. G. (1986). Studying artificial life with cellular automata. *Physica D: nonlinear phenomena*, 22(1-3):120–149.

Levin, M. (2023). Darwin's agential materials: evolutionary implications of multiscale competency in developmental biology. *Cellular and Molecular Life Sciences*, 80(6).

Levin, M. (2025). The multiscale wisdom of the body: Collective intelligence as a tractable interface for next-generation biomedicine. *BioEssays*, 47(3):e202400196.

Lobo, D., Solano, M., Bubenik, G. A., and Levin, M. (2014). A linear-encoding model explains the variability of the target morphology in regeneration. *Journal of the Royal Society, Interface*, 11(92):20130918.

Lu, C., Lu, C., Lange, R. T., Yamada, Y., Hu, S., Foerster, J., Ha, D., and Clune, J. (2026). Towards end-to-end automation of AI research. *Nature*, 651(8107):914–919.

Mathews, J., Chang, A. J., Devlin, L., and Levin, M. (2023). Cellular signaling pathways as plastic, proto-cognitive systems: Implications for biomedicine. *Patterns*, 4(5):100737.

Michel, T., Cvjetko, M., Hamon, G., Oudeyer, P.-Y., and Moulin-Frier, C. (2025). Exploring Flow-Lenia Universes with a Curiosity-driven AI Scientist: Discovering Diverse Ecosystem Dynamics. In *ALIFE 2025: Ciphers of Life: Proceedings of the Artificial Life Conference 2025*. MIT Press.

Miotti, P., Niklasson, E., Randazzo, E., and Mordvintsev, A. (2025). Differentiable Logic Cellular Automata: From Game of Life to Pattern Generation. In *ALIFE 2025: Ciphers of Life: Proceedings of the Artificial Life Conference 2025*. MIT Press.

Mordvintsev, A., Randazzo, E., Niklasson, E., and Levin, M. (2020). Growing neural cellular automata. *Distill*, 5(2):e23.

Papadopoulos, V. and Guichard, E. (2025). MaCE: General Mass Conserving Dynamics for CAs. In *ALIFE 2025: Ciphers of Life: Proceedings of the Artificial Life Conference 2025*. MIT Press.

Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. (2018). FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, pages 3942–3951, New Orleans, Louisiana, USA. AAAI Press.

Pio-Lopez, L., Hartl, B., and Levin, M. (2025). Aging as a loss of goal-directedness: An evolutionary simulation and analysis unifying regeneration with anatomical rejuvenation. *Advanced Science*, 12(46).

Plantec, E., Hamon, G., Etcheverry, M., Chan, B. W.-C., Oudeyer, P.-Y., and Moulin-Frier, C. (2025). Flow-Lenia: Emergent Evolutionary Dynamics in Mass Conservative Continuous Cellular Automata. *Artificial Life*, 31(2):228–248.

Rainwater, J. H. (2024). Self-Organization and Phase Transitions in Driven Cellular Automata | Artificial Life | MIT Press.

Reinke, C., Etcheverry, M., and Oudeyer, P.-Y. (2020). Intrinsically Motivated Discovery of Diverse Patterns in Self-Organizing Systems. In *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Sánchez-Fibla, M., Moulin-Frier, C., and Solé, R. (2024). Cooperative control of environmental extremes by artificial intelligent agents. *Journal of The Royal Society Interface*, 21(220):20240344.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 237(641):37–72.

Tversky, A. and Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131.

van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double Q-Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2094–2100, Phoenix, Arizona. AAAI Press.

Von Neumann, J. and Burks, A. W. (1966). Theory of self-reproducing automata.

Wolfram, S. (1983). Cellular automata. *Los Alamos Science*, pages 09–01.

Wu, J., Sun, X., Zeng, A., Song, S., Lee, J., Rusinkiewicz, S., and Funkhouser, T. (2020). Spatial Action Maps for Mobile Manipulation. In *Robotics: Science and Systems XVI*, volume 16.

Zeng, A., Song, S., Welker, S., Lee, J., Rodriguez, A., and Funkhouser, T. (2018). Learning Synergies Between Pushing and Grasping with Self-Supervised Deep Reinforcement Learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4238–4245, Madrid, Spain. IEEE Press.

Zenil, H., Tegnér, J., Abrahão, F. S., Lavin, A., Kumar, V., Frey, J. G., Weller, A., Soldatova, L., Bundy, A. R., Jennings, N. R., Takahashi, K., Hunter, L., Dzeroski, S., Briggs, A., Gregory, F. D., Gomes, C. P., Rowe, J., Evans, J., Kitano, H., and King, R. (2026). The future of fundamental science led by generative closed-loop artificial intelligence. *Frontiers in Artificial Intelligence*, 9.