A Methodological Exploration of Rule-Based and MLP-Based Agents in Agent-Based Modeling

Lorenzo Gastaldo[1,2,†], Federico Carucci[1,2,†], Leonardo Mascagni[1,2,†] and Francesco Bertolotti[1,3,*,†]

[1]*Intelligence, Complexity and Technology Lab (ICT Lab), University Cattaneo, LIUC, Italy*
[2]*School of Industrial Engineering, University Cattaneo, LIUC, Italy*
[3]*Università Cattolica di Milano, Department of Philosophy, L. Gemelli 1 - 20123 Milano, Italy*

Abstract

This paper presents a methodological framework for comparing rule-based and neural network-based decision-making in ABM by explicitly separating two design dimensions that are often treated jointly in the literature: policy initialisation and policy adaptation. Using a first-principles spatial foraging ABM as a controlled testbed, the study defines an experimental design in which rule-based and MLP-based agents are evaluated under multiple initialisation strategies (random, fixed, or offline-trained) and adaptation mechanisms (fixed inheritance, random offspring, evolutionary mutation, or online learning). In total, 14 agent variants (a fully random baseline, 6 rule-based variants, and 7 MLP-based variants) are tested across five environmental scenarios spanning different levels of ecological difficulty, with performance assessed through extinction and survival outcomes. For neural agents, the framework also includes offline behavioural cloning, online REINFORCE updates, and a systematic comparison across network depths. The paper offers a replicable experimental protocol for studying how alternative behavioural architectures affect agent performance in ABMs, with particular value for research at the intersection of ABM and machine learning.

Keywords

agent-based model, ABM, neural network, machine learning, multi-layer perceptron, MLP

1. Introduction

In the last 30 years, agent-based modelling (ABM) has become a widely used paradigm for modelling complex adaptive systems [1, 2]. In many fields and conditions, it can overcome traditional equation-based models, in which global system behaviour is represented by a small number of differential equations [3, 4]. The underlying intuition is to model the individual components of a system as autonomous agents rather than modeling only aggregate behaviour, and then to observe the emergent behaviour at the macro-level [5, 6]. In such a setting, how agents make decisions is a central design problem. Algorithmic rules are most commonly used because they are computationally cheap and transparent, but they can limit behavioural richness and make calibration difficult [7, 8]. Machine learning agents – in particular neural networks (NN) – offer greater flexibility at the cost of opacity and higher computational cost [9, 10, 11]. Several works have explored ML-ABM integration [12, 13], yet controlled comparisons isolating individual design choices remain scarce [7].

A common source of confusion is the conflation of two design perspectives. A timing-based taxonomy of ML use in ABM has been proposed [8], and we build on it to address this conflation. Building on this taxonomy, we frame the distinction as one between policy initialisation, namely what the agent knows before the simulation begins, and policy adaptation, namely how the policy changes during the simulation. A further distinction cuts across both: whether a procedure is offline, taking place before the simulation starts (for example, supervised training on pre-collected data), or online, taking place during the simulation itself through step-by-step updates driven by reward signals. Existing surveys classify

WOA 2026, the 27th Workshop From Objects to Agents, June 15–17, 2026, Salerno, Italy
*Corresponding author.
†These authors contributed equally.
lo07.gastaldo@stud.liuc.it (L. Gastaldo); fcarucci@liuc.it (F. Carucci); le21.mascagni@stud.liuc.it (L. Mascagni); fbertolotti@liuc.it (F. Bertolotti)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-10-30

CEUR-WS.org/Vol-4307/paper1.pdf

ML-ABM contributions along orthogonal axes such as algorithm family or model-building step [10, 9], so an MLP trained offline and kept fixed is not separated from one that starts from random weights and adapts online via reinforcement [14], yet both are labelled as neural agents. Learned initialisations are also subject to distribution shift, with prediction error compounding quadratically over the deployment horizon [15, 16].

In this paper, the difference between these two dimensions is analyzed through an experiment using a first-principles spatial foraging ABM [17]. Two policy families were evaluated, a rule-based (RB) heuristic and an MLP-based policy, each crossed with multiple initialisation strategies (random, fixed, or offline behavioural cloning) and adaptation operators (fixed inheritance, random offspring, evolutionary mutation, or reinforcement learning). All 14 variants ($V0$–$V13$, where $V0$ is a fully random baseline and $V1$–$V13$ are the rule-based and MLP-based variants under study) were evaluated under five scenarios spanning a survival-probability gradient from near-certain extinction to near-certain survival, using ecological outcomes such as extinction rate and overall population survival as metrics.

The main contributions are as follows. First, the study identifies the conditions under which initialisation, rather than adaptation, is the main determinant of survival for both rule-based and MLP-based populations [18]. It also shows that MLP agents are markedly more sensitive to initial conditions [19]: randomly initialised networks are nearly non-functional in adverse scenarios, while offline-initialised networks approach but do not exceed the best rule-based variants. Second, in MLP agents, learning processes interact asymmetrically with initialisation: they improve randomly initialised networks only in sufficiently favourable scenarios, but degrade offline-initialised ones. Increasing network depth also consistently harms randomly initialised MLPs while leaving offline-initialised variants unaffected. Finally, all surviving variants converge to the same steady-state population equilibrium within each scenario, indicating that policy differences affect the probability of surviving an early bottleneck rather than the long-run carrying capacity.

The remainder of the paper is organised as follows. Section 2 reviews the relevant background. Section 3 describes the model, agent variants, and experimental protocol, while Section 4 presents and discusses the results and Section 5 presents the conclusions.

2. Background

2.1. ABM Integration with Machine Learning

ABM is a bottom-up simulation paradigm in which system-level outcomes emerge from the interaction of heterogeneous autonomous agents operating under local information and behavioural rules [10, 5, 6]. This makes ABM particularly suitable for studying adaptive, decentralised, and nonlinear systems, where aggregate patterns cannot be inferred directly from top-down equations [17]. This often comes at the cost of replicability [20], which standardized protocols such as Overview, Design concepts, Details (ODD) attempt to address by providing a structured description of the model [21].

A common modelling practice is to specify agents through algorithmic decision rules [5, 17]. Rule-based agents are typically used because their behaviour can be traced back to explicit modeller assumptions, offering transparency and controllability [17, 8]. However, behavioural richness is bounded by the structure of the rule set itself [7], and calibration can be difficult [8, 10]. Neural networks, by contrast, can approximate complex nonlinear mappings from perception to action [22, 23, 16]. This approximation power comes at the cost of opacity and reduced interpretability, given that cause-effect relationships are hidden within the network's parameters and cannot be inferred without specific knowledge of the system [17, 9]. The two approaches can be seen as complementary: ABMs capture heterogeneity and local interactions, transforming them into emergent dynamics, while neural networks learn nonlinear patterns from data but risk overfitting and producing dynamics that violate structural constraints [24].

A growing body of work has explored the integration of ML into ABM [7, 8]. An early formalisation describes a two-cycle framework in which the ML algorithm uses the ABM as an environment and reward generator, while the ABM uses the ML algorithm to refine the internal models of its agents [25].

Several surveys have since mapped this integration space: a systematic review of 71 studies documents the use of ML for behavioural modelling, calibration, and computational efficiency across learning paradigms [9]; a broader survey of 190 publications identifies two dominant directions, using ML to build more adaptive agents and using ML to analyse simulation outputs [10]; and a multidisciplinary review organises ML-ABM integration into four canonical scenarios [12]. General frameworks for integrating ML into the ABM pipeline have also been proposed [7], with concrete applications in microbiology [26]. A complementary direction encodes ABM structural principles directly into neural differential equations, obtaining hybrid architectures that preserve the mechanistic constraints of ABMs while retaining the learning capacity of neural networks [24].

2.2. Distinguishing Learning Mechanisms in ML-ABM

In the ML-ABM literature, ML methods are typically classified by algorithm family or by the model-building step they support [10, 9]. A different organising principle is the *moment* at which learning takes place relative to the simulation, as proposed in a timing-based extension of the ODD protocol [8]. This taxonomy distinguishes offline learning, which occurs outside the simulation, from online learning, which occurs during it. Offline learning is in turn split into an a priori form, applied in the design stage to instantiate model components from existing data, and an a posteriori form, applied after the simulation to calibrate parameters towards validation or to analyse the relationship between inputs and outputs [8]. Online learning instead uses data generated during the simulation, typically to let agents derive optimal behaviour through trial and error [8].

Keeping these aspects separate matters because mechanisms that differ in purpose and timing are often grouped under a single label: an MLP trained offline and kept fixed behaves very differently from one that adapts online via REINFORCE [14, 27], yet existing surveys tend to place both under the same "neural agent" heading [10, 9]. Likewise, evolutionary operators [28, 29, 30] and gradient-based updates act on different temporal scales while both being treated as "adaptation", even though evolutionary strategies have been shown to be a competitive alternative to policy gradient methods for optimising neural network policies [30].

An empirical instance of this separation is provided by a system in which neuroevolution operates as an outer loop that refines agent weights across generations through mutation and physiology-driven selection, while a Long Short-Term Memory (LSTM) component enables a distinct inner loop of intra-lifetime adaptation without any weight update [31]. The two mechanisms act at different temporal scales and serve different functional roles [31], illustrating how mechanisms commonly bundled under a single "learning" label can be separated into distinct design choices [8, 31].

2.3. Neural Networks for Behavioural Modelling in ABM

Several studies have proposed neural networks as a way to replace or complement hand-coded behavioural rules in ABM [23, 16, 31]. A general framework trains a neural network to map local agent perceptions to actions and deploys it inside the simulation; when tested on Sugarscape, it reproduces core macro-level patterns without hand-coded movement rules [16]. The quality of the deployed policy, however, depends strongly on the state distribution observed during data collection: if training data come from regions of the state space that differ from those encountered during deployment, predictive accuracy can remain high while behavioural performance in simulation remains poor [16], a problem amplified in ABM, where each agent's actions reshape the future state distribution [16]. The same framework has been extended to public goods games, showing that agents with identical value parameters but different training experiences develop distinct strategies, whereas strict reinforcement learning produces uniform behaviour [32].

At ecological scale, LSTM networks have been deployed as the decision-making architecture in a large non-episodic multi-agent foraging environment, in which evolvable recurrent networks must sustain the population under continuously evolving ecological conditions without environment or population resets [31].

2.4. Offline Learning, Distribution Shift, and Online Adaptation

A typical approach to offline policy learning is behavioural cloning (BC), in which a neural network is trained via supervised learning on state-action pairs collected from a reference policy [33, 15]. The problem of distribution shift was first identified empirically in the context of autonomous road following: the network performed well on training conditions but lacked any mechanism to recover from deviations not encountered during training [33]. Although BC is straightforward to implement, it is particularly vulnerable to distribution shift: a policy with classification error ε under the expert's state distribution can incur up to $T^2\varepsilon$ expected cost over a T-step deployment, a quadratic growth that makes offline imitation learning unreliable on long horizons [15]. An iterative algorithm called DAgger has subsequently been proposed that addresses this by retraining the policy on states it actually visits, recovering near-linear regret [34].

This theoretical concern has been confirmed empirically in the ABM setting: more training data do not necessarily improve behavioural performance if the underlying experience distribution is not representative of deployment conditions [16]. The problem is compounded in multi-agent environments, where an agent trained on trajectories generated by another policy class may face states whose frequency and structure diverge from the training set once it begins to interact with other agents inside the ABM [15, 16].

2.5. Foraging Environments and Ecological Fitness

Foraging models provide a natural testbed for comparing alternative cognitive architectures because agent success can be measured through architecture-neutral ecological outcomes such as survival, energy balance, and reproductive success [17]. A major reference point is Sugarscape, where agents move on a grid, harvest renewable resources, consume energy, and reproduce or die depending on their energetic state [35]. This family of models is particularly well suited for studying behavioural adaptation because the mapping from local decisions to long-term fitness is indirect, nonlinear, and shaped by population interactions [17]. A prominent empirical application is the Long House Valley model, in which household agents governed by simple demographic and nutritional rules reproduce the observed dynamics of the Kayenta Anasazi over five centuries of occupation [36].

This class of models has been extended to study eco-evolutionary dynamics at larger scales: a non-episodic common-pool resource environment with spatially varying regrowth and freely fluctuating population size produces oscillatory dynamics reminiscent of Lotka-Volterra patterns. [31]. ABM simulations have also been used to study the evolutionary origins of risk sensitivity, showing that risk-averse strategies emerge under selection pressure in small populations, with the degree of risk aversion shaped by effective population size and the fitness impact of individual decisions [37].

3. Methods

3.1. Model description

At a high level, the environment models a population of agents foraging for food on a finite spatial grid. Each agent must balance movement against energy management: moving to a richer neighboring cell costs energy and resets the agent's local view, while staying conserves energy but risks depleting the local resource. Agents reproduce when their energy exceeds a threshold and die when it reaches zero. The only active decision at each step is where to move (or whether to stay), making this a single-action testbed that deliberately excludes memory, planning, and strategic interaction. This simplicity is a feature, not a limitation: it isolates the effects of policy initialisation and adaptation without confounding factors, providing a clean and reproducible basis for comparing behavioural architectures.

In this model, agents are located on an $l \times l$ resource grid, with at most one agent per cell. Agents can perform only a small set of actions: forage, which means collecting food from the cell they occupy and storing it; eat, consuming part of the stored energy; move to a neighboring cell; reproduce, spawning a new agent; and die. Of these actions, only movement is a decision, while the others are metabolic

reactions to specific internal or external states. Each simulation begins with n agents A_i placed at random locations with energy e_i sampled uniformly in an interval $[e_{min}, e_{max}]$; food f_j in each cell j is initialised uniformly in $[f_{min}, f_{\text{cap}}]$ per cell.

At each discrete step: (i) food regenerates by a scenario-dependent amount up to a maximum of f_{cap} units per cell; (ii) each agent collects up to fc_{max} food units from its cell and pays a metabolic cost; (iii) the agent observes its local state and selects one of five actions (move N/S/E/W or stay). Invalid moves, such as out-of-bounds or occupied, are treated as stay. A successful move transfers ft of pre-move energy to the vacated cell. Reproduction is attempted when energy $\geq re$ with probability p_r, but only if a free Von-Neumann neighbour exists. In this case, the parent splits its energy equally with the offspring. Agents at or below zero energy die and are removed.

Every agent receives the same 10-dimensional percept at each time-step t: food in the four neighbouring cells, occupancy of those cells, food in the current cell, and current energy. Out-of-bounds occupancy is encoded as 1; out-of-bounds food as -1.

Table 1
Environment variables.

Name	Description	Value
$l \times l$	spatial resource grid, at most one agent per cell	15×15
f_j	food in cell j	—
$[f_{\text{min}}, f_{\text{cap}}]$	range of initial food and cell capacity	$[0, 8]$
fr	food regrowth per step	scenario-dependent

Table 2
Agent variables common to all agents.

Name	Description	Value
n	initial population size, placed at random locations	50
A_i	i-th agent	—
e_i	current energy of agent A_i	—
$[e_{\text{min}}, e_{\text{max}}]$	range of initial energy, sampled uniformly	$[1, 5]$
bm	base metabolism, energy cost paid per step	scenario-dependent
fc_{max}	max food units collected per step from current cell	3
re	reproduction energy threshold	5.0
p_r	probability of reproducing when eligible and a free neighbour exists	0.5
ft	fraction of pre-move energy transferred to the vacated cell	0.20
f_h	food in the current cell	—
f_{best}	highest food level among free neighbouring cells	—
`percept`	10-dim observation at step t: food and occupancy of 4 neighbours, f_h, e_i	—

3.2. Agent decision-making

In this work, agent decision-making can follow one of three approaches: fully random, which serves as a baseline; rule-based; and MLP-based. This subsection specifies how the rule-based and MLP-based behaviours are implemented.

Rule-based agents. The rule-based policy of each A_i is controlled by a scalar risk propensity α_i, such that the agent moves to the free neighbor with the highest food level f_{best} if and only if

$$f_{\text{best}} \cdot \alpha > f_h + e_i \cdot ft, \tag{1}$$

where f_h is food in the current cell, e_i is current energy, and ft is the fraction of pre-move energy transferred to the vacated cell, acting as the movement cost in the decision rule.[1] If the condition is

[1]Food units and energy units are treated as equivalent (1:1) throughout the model.

not satisfied, the agent does not move. Because α represents how agents handle the uncertainty of moving to a different cell, $\alpha < 1$ encodes risk aversion, $\alpha = 1$ a rational cost-benefit threshold, and $\alpha > 1$ risk seeking. Moving also carries an informational cost: the agent observes a largely different neighbourhood after each move, since only two cells overlap with the previous view.

In the random-initialisation setup, $\alpha_i \sim \mathcal{U}[\alpha_{min}, \alpha_{max}]$, while fixed variants start at $\alpha_i = 1.0$ and are therefore risk neutral. Behavioural adaptation can then occur through several mechanisms. In the simplest case, the offspring inherits the parent's α_i exactly, so that no further modification is introduced across generations. A second possibility is complete re-initialisation, in which each offspring receives a newly sampled value of α_i, independently of the parent. Third, offspring can undergo gradual evolutionary change by inheriting the parental value with a Gaussian perturbation of standard deviation σ, with the resulting value clipped to the admissible interval $[\alpha_{min}, \alpha_{max}]$. Finally, adaptation can also take place within the lifetime of the agent through online learning. The risk-propensity gene is reparameterised as $\tilde{\alpha}_i = \alpha_i / \alpha_{\max} \in [0, 1]$ and updated at each step by a Bernoulli-style policy-gradient rule inspired by REINFORCE in logit space,

$$\operatorname{logit}(\tilde{\alpha}_{i,t+1}) = \operatorname{logit}(\tilde{\alpha}_{i,t}) + \eta\,(r_t - b_t)\,\big(a_{i,t} - \tilde{\alpha}_{i,t}\big), \tag{2}$$

where $a_{i,t} \in \{0, 1\}$ indicates whether the agent moved, η is the learning rate, and b_t is an exponential baseline with smoothing parameter α_b. The gene is then recovered as $\alpha_{i,t+1} = \alpha_{\max}/(1 + e^{-\operatorname{logit}(\tilde{\alpha}_{i,t+1})})$.

MLP agents. The neural family uses a feed-forward MLP with n_i inputs, n_o outputs, and hl_{min}–hl_{max} hidden layers of x ReLU units. Non-learning agents act greedily according to an argmax logit, while online-learning agents sample from a softmax distribution. MLP agents are initialised in different ways: random variants employ random weights, while offline variants are initialised by BC on perception-action pairs as specified in the experimental design.

For MLP-based agents, behavioural change can arise through different inheritance and adaptation mechanisms. In the fixed setting, offspring receive an exact copy of the parent's weights, so that the policy is transmitted unchanged across generations. In the random-offspring setting, each offspring is instead assigned a newly sampled set of weights, independent of the parent. A third mechanism introduces evolutionary variation by adding Gaussian noise to each inherited weight with probability p_m and standard deviation σ. Finally, adaptation can also occur during the agent's lifetime through online learning. In this case, the policy is updated at each step using a one-step REINFORCE algorithm with entropy regularisation [14], according to

$$\mathscr{L}_t = -\left[\log \pi_\theta(a_{t,i} \mid s_t)\,(r_t - b_t) + \beta_H\,\mathscr{H}(\pi_\theta(\cdot \mid s_t))\right], \tag{3}$$

where b_t is an exponential moving-average baseline updated as

$$b_{t+1} = (1 - \alpha_b)\,b_t + \alpha_b\,r_t, \quad b_0 = 0, \tag{4}$$

with $\alpha_b = 0.15$ and $\beta_H = 0.01$ controlling the strength of entropy regularisation; optimisation is performed with Adam. To improve numerical stability, the gradient norm is clipped. Under this online-learning scheme, offspring inherit the current weights of the parent, including any updates accumulated during its lifetime.

Both online-learning families share the same reward, the clipped sum of four contributions evaluated between consecutive steps: an energy-gain term $r_t^{\mathrm{e}} = \alpha_e(e_{t+1} - e_t)/e^*$, a reproduction bonus $r_t^{\mathrm{r}} = b_b\,\mathbf{1}[\text{reproduction at } t]$, an energy-deficit penalty $r_t^{\mathrm{d}} = -\beta_{\mathrm{def}}\,(e_c - e_{t+1})/e_c \cdot \mathbf{1}[e_{t+1} < e_c]$, and a survival bonus $r_t^{\mathrm{s}} = b_{\mathrm{surv}}$, combined as

$$r_t = \operatorname{clip}\big(r_t^{\mathrm{e}} + r_t^{\mathrm{r}} + r_t^{\mathrm{d}} + r_t^{\mathrm{s}},\ -r_{\mathrm{clip}},\ r_{\mathrm{clip}}\big). \tag{5}$$

Here $\mathbf{1}[\,\cdot\,]$ denotes the indicator function, equal to 1 when its argument is true and 0 otherwise; $e^* = re$ is the reproduction threshold (Table 2); $e_c = k \cdot m$ is a scenario-scaled critical energy level, with $m = bm$ the base metabolic cost (Table 1); α_e weights the energy-gain term; b_b is the reproduction bonus; β_{def}

scales the deficit penalty; b_{surv} is the per-step survival bonus; and r_{clip} bounds the reward magnitude. Default values used in all experiments are $\alpha_e = 1.0$, $b_b = 0.2$, $\beta_{\text{def}} = 0.6$, $k = 1.5$, $b_{\text{surv}} = 0.02$, $r_{\text{clip}} = 2.0$. The energy gain is normalised by e^* to keep the signal comparable across scenarios; the deficit penalty scales with m so the agent receives the same warning lead time regardless of scenario severity. For surviving agents, the reward for step t is applied after the nutrition phase of step $t+1$ so that e_{t+1} reflects both movement cost and nutritional intake; if the agent dies in step t the update is applied immediately.

3.3. Variant summary and experimental design

The parameter values used in this experiment are listed in Table 3. All variant-specific parameters were fixed empirically through preliminary testing; the conclusions should therefore be interpreted conditional on these values. Table 4 lists all evaluated variants.

Table 3
Variant-specific parameters (not common to all agents).

Name	Description	Value
α_i	risk-propensity gene (RB only); <1 risk-averse, =1 rational, >1 risk-seeking	$[0, 2]$
σ	Gaussian variation of evolutionary mutation	0.05 (0.1 for MLP)
η	learning rate	0.05
α_b	exponential baseline parameter	0.15
p_m	probability of mutation in MLP evolution	0.05
β_H	strength of entropy regularisation	0.01
n_i	number of inputs of the MLP agents	10
n_o	number of outputs of the MLP agents	5
hl_{min}	minimum number of hidden layers in the MLP	1
hl_{max}	maximum number of hidden layers in the MLP	5
x	number of ReLU units per hidden layer	16

Table 4
Summary of all evaluated agent variants.

ID	Full name	Family	Initialisation	Adaptation
$V0$	Random	—	—	—
$V1$	RB_Rand_Rand	RB	Random	Random offspring
$V2$	RB_Rand_Evo	RB	Random	Evolutionary
$V3$	RB_Rand_Learn	RB	Random	Online learning
$V4$	RB_Fix_Fix	RB	Fixed (α=1.0)	Fixed
$V5$	RB_Fix_Evo	RB	Fixed (α=1.0)	Evolutionary
$V6$	RB_Fix_Learn	RB	Fixed (α=1.0)	Online learning
$V7$	MLP_Rand_Fix	MLP	Random	Fixed
$V8$	MLP_Rand_Rand	MLP	Random	Random offspring
$V9$	MLP_Rand_Evo	MLP	Random	Evolutionary
$V10$	MLP_Rand_Learn	MLP	Random	Online learning
$V11$	MLP_Offline_Fix	MLP	Offline (BC)	Fixed
$V12$	MLP_Offline_Evo	MLP	Offline (BC)	Evolutionary
$V13$	MLP_Offline_Learn	MLP	Offline (BC)	Online learning

Environmental scenarios. Five scenarios ($S1$–$S5$) were selected along a linear transect of the Pareto front in the (fr, bm) parameter space such that $V2$ achieves survival probabilities of approximately 5%, 25%, 50%, 75%, and 100% respectively, spanning the full range from near-certain extinction to near-certain survival. Figure 1 shows the full parameter space coloured by $V2$ survival probability and the five selected scenario points. Each combination of variant, scenario, and (for MLP variants) number of hidden layers was replicated 500 times with independent random seeds.

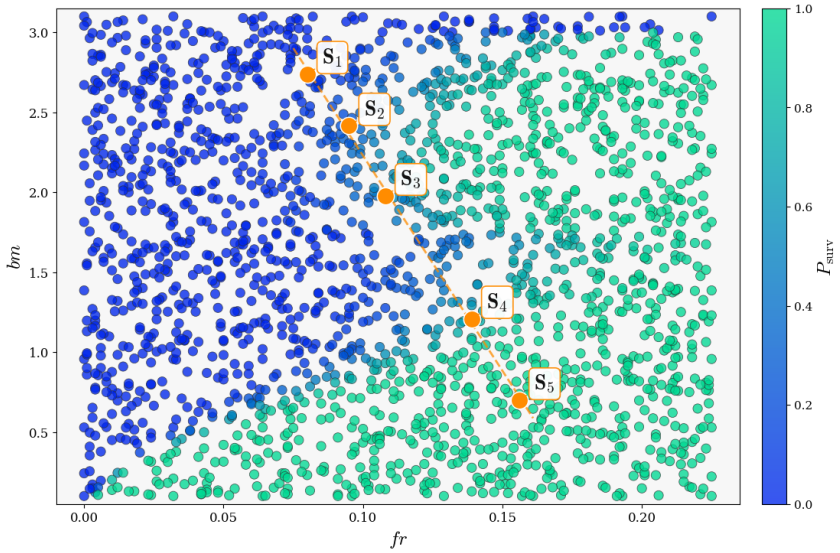

Figure 1: Survival map of the (food regeneration, base metabolism) parameter space. X-axis: food regeneration rate (fr, units per cell per step). Y-axis: base metabolic cost (bm, energy units per step). Each point is one parameter combination, coloured by $V2$ survival probability. The five labelled orange points ($S1$–$S5$) mark the scenarios selected along a linear transect.

Offline training procedure. Offline MLP weights were obtained by BC on perception-action pairs from $V4$ agents. $V4$ was chosen because its fixed initialisation ($\alpha = 1.0$) and fixed inheritance produce a fully deterministic decision rule: given identical percepts, every $V4$ agent selects the same action, yielding a coherent and unambiguous training signal. A variant with heterogeneous α values (e.g. $V2$) would generate contradictory labels for the same state, making supervised training ill-posed. This is a limitation of the present comparison: the offline MLP imitates one specific rule-based teacher that is not necessarily the strongest available, and training against alternative teachers is left to future work (see Section 5).

Each scenario-specific dataset contained at least 500,000 examples from rollouts of at most 2,000 steps. Training used cross-entropy loss, Adam [38] (lr $= 10^{-3}$), 5-fold stratified cross-validation (batch 256/512 train/val), early stopping (patience 10, max 50 epochs). The checkpoint with the lowest validation loss per fold was retained; reported accuracy is the mean across folds. For each scenario, 15 independent training runs with different random seeds were conducted. Each trained model was evaluated over 500 simulation replications.

The simulation code and analysis scripts used for all experiments are publicly available at: https://github.com/LoriGas/A-Methodological-Exploration-of-Rule-Based-and-MLP-Based-Agents-in-Agent-Based-Modeling.[2]

4. Results and discussion

4.1. Rule-based agent-based model

Sensitivity of extinction rate to the risk propensity gene. Figure 2 shows extinction rate as a function of a fixed, homogeneous $\alpha \in [0.1, 2.0]$ for each scenario.

$S1$ remains at ≈97–100% extinction and $S5$ at 0% across the full range, confirming the calibration of the scenarios. $S3$ and $S4$ exhibit a clear non-monotonic pattern with a local extinction maximum near $\alpha \approx 0.7$–0.8: agents move just enough to incur costs without reliably reaching better cells. $S3$ peaks at ≈88% before declining monotonically to ≈41% at α=2.0; $S4$ peaks at ≈58% and declines to ≈38%, making risk propensity consequential but not monotonic even in moderately favourable environments. $S2$

[2]https://github.com/LoriGas/A-Methodological-Exploration-of-Rule-Based-and-MLP-Based-Agents-in-Agent-Based-Modeling

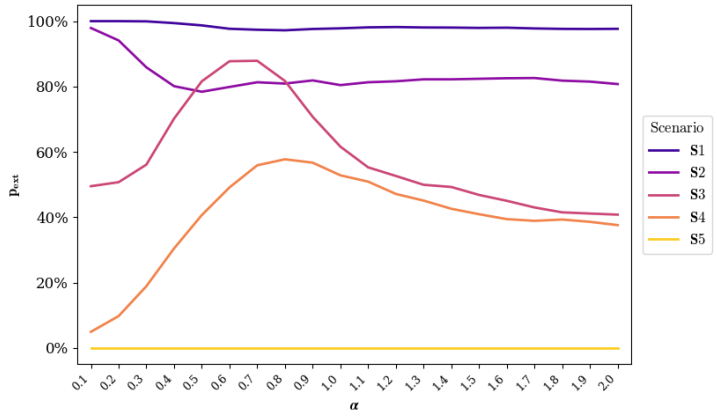

Figure 2: Extinction rate as a function of the risk-propensity gene value α, for each scenario $S1$–$S5$. X-axis: fixed homogeneous α value, ranging from 0.1 to 2.0 in steps of 0.1. Y-axis: extinction rate (fraction of the 500 replications ending in extinction). Each line corresponds to one scenario; each point is a population in which all agents share the same fixed α.

instead follows a different pattern: extinction starts at ≈98% for α=0.1, reaches a minimum of ≈78–80% at $\alpha \approx 0.4$–0.5, and stabilises around 80–82% for $\alpha > 1.0$, without the pronounced intermediate peak of $S3$ and $S4$. No single α is universally optimal: the minimum extinction lies at low α in $S4$, at intermediate α in $S2$, and at high α in $S3$, making risk propensity scenario-dependent rather than monotonically tied to environmental favourability. The non-monotonic shape shared by $S3$ and $S4$ (peak extinction at $\alpha \approx 0.7$–0.9) likely reflects the fact that the movement cost scales with energy ($f_t \cdot e_i$): at intermediate α, high-energy agents are held back by their own accumulated reserves and rarely move, while low-energy agents move but cannot consistently recover the transfer cost. Both consistently sedentary and consistently mobile populations avoid this trade-off.

Extinction rate across variants. Figure 3 shows extinction rates for $V0$–$V6$ across scenarios.

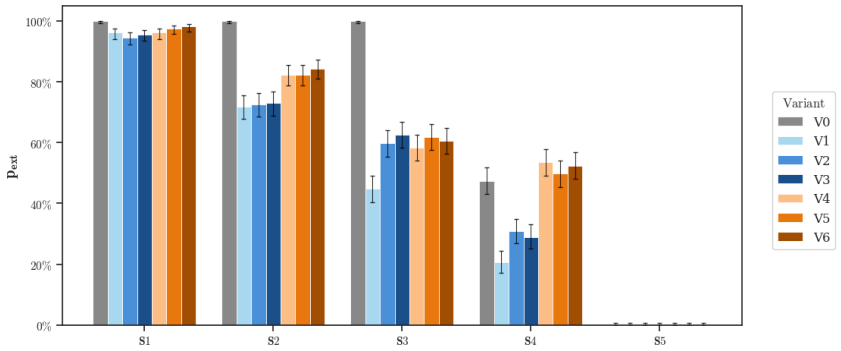

Figure 3: Rule-based agents: fraction of runs ending in extinction for each variant ($V0$–$V6$) across scenarios $S1$–$S5$. X-axis: scenario ($S1$–$S5$, grouped). Y-axis: extinction rate. Bars are coloured by initialisation class (one hue for randomly initialised variants $V1$–$V3$, another for fixed-initialisation variants $V4$–$V6$); shade intensity within each hue distinguishes the adaptation operator. This colour convention is maintained throughout all subsequent figures. Error bars indicate 95% Wilson confidence intervals (n = 500 replications per condition).

$S1$ and $S5$ are saturated at the two extremes, with all variants reaching ≈95–100% extinction in $S1$ and 0% in $S5$, so they carry little discriminative information. In the intermediate scenarios, initialisation matters more than adaptation, though the effect is not uniform. In $S2$, randomly initialised variants $V1$–$V3$ outperform the fixed-initialisation group $V4$–$V6$ by roughly ten points (≈71–73% versus ≈82–84%). In $S3$ the picture is more nuanced: $V1$ (≈44%) substantially outperforms all other variants, while $V2$–$V3$ (≈59–62%) are comparable to or slightly worse than $V4$–$V6$ (≈58–61%), showing that the benefit of

random initialisation is not guaranteed across adaptation operators. The gap widens and becomes more systematic in $S4$, where $V1$ drops to ≈20% and $V2$–$V3$ reach ≈29–31%, while $V4$–$V6$ remain at ≈49–53% and even the policy-free baseline $V0$ sits at ≈47%. Within the random group, $V1$ clearly outperforms $V2$–$V3$ because directional adaptation gradually reduces α diversity, whereas $V1$'s stochastic offspring keep replenishing it. That $V0$ matches or beats $V4$–$V6$ in $S4$ shows that what matters is behavioural diversity: $V0$ achieves it through stochastic decisions, $V1$–$V3$ through heterogeneous α, while $V4$–$V6$ are locked at α=1.0, suboptimal given that Figure 2 shows extinction drops further for $\alpha > 1.3$.

Population dynamics over time. Figure 4 shows mean population trajectories for surviving runs only.

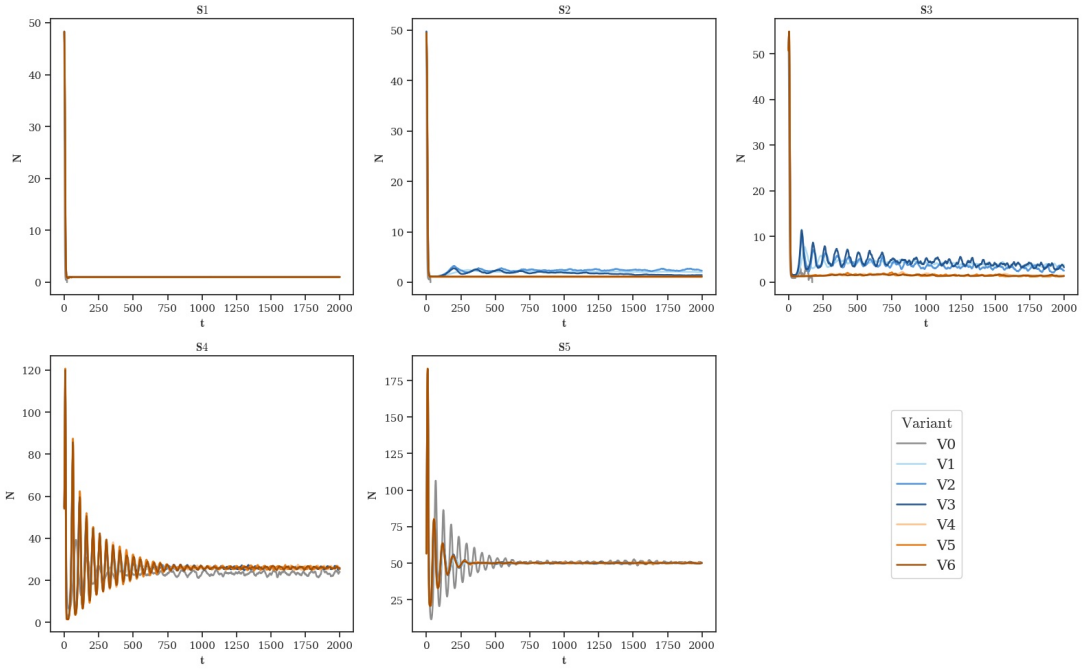

Figure 4: Rule-based agents: mean population over time for surviving runs only. X-axis: simulation step t (0–2,000). Y-axis: mean population size N, averaged across surviving replications (note that the y-axis range differs across panels to accommodate the very different equilibrium sizes). Each panel corresponds to one scenario ($S1$–$S5$); lines are variants $V0$–$V6$, coloured according to the convention introduced in Figure 3.

The trajectories fall into three regimes. In $S1$–$S2$, populations collapse to 1–3 agents within tens of steps and stabilise there: at such low density, competition for resources vanishes and survivors maintain a positive energy balance. In $S3$, an initial overshoot is followed by oscillatory decay to ≈3–5 agents. In $S4$–$S5$, pronounced oscillations damp over ≈500 steps to equilibria of ≈25 and 50 agents, as the population gradually matches the carrying capacity through cycles of overexploitation and recovery. All variants converge to the same equilibrium within each scenario: the steady-state is set by food regrowth and grid size, so policy affects whether a run survives the bottleneck, not the equilibrium density.

Survival curves. Figure 5 shows the fraction of runs still alive at each step.

In $S1$–$S2$, nearly all extinctions occur within the first ~50 steps and the surviving fraction plateaus almost immediately: these scenarios act as an early bottleneck that kills or spares a variant before adaptation has time to play a role. $S3$ starts behaving differently: $V1$ retains the highest surviving fraction at step 2,000, because directional operators gradually drive α toward a single value, whereas $V1$'s stochastic offspring keep reintroducing variance. The clearest separation appears in $S4$, where survival fractions spread widely: $V1$ stabilises at ≈79%, $V2$–$V3$ at ≈70–72%, the policy-free baseline $V0$ at ≈55%, and $V4$–$V6$ at just ≈48–52%. That the gap persists across 2,000 steps suggests α heterogeneity

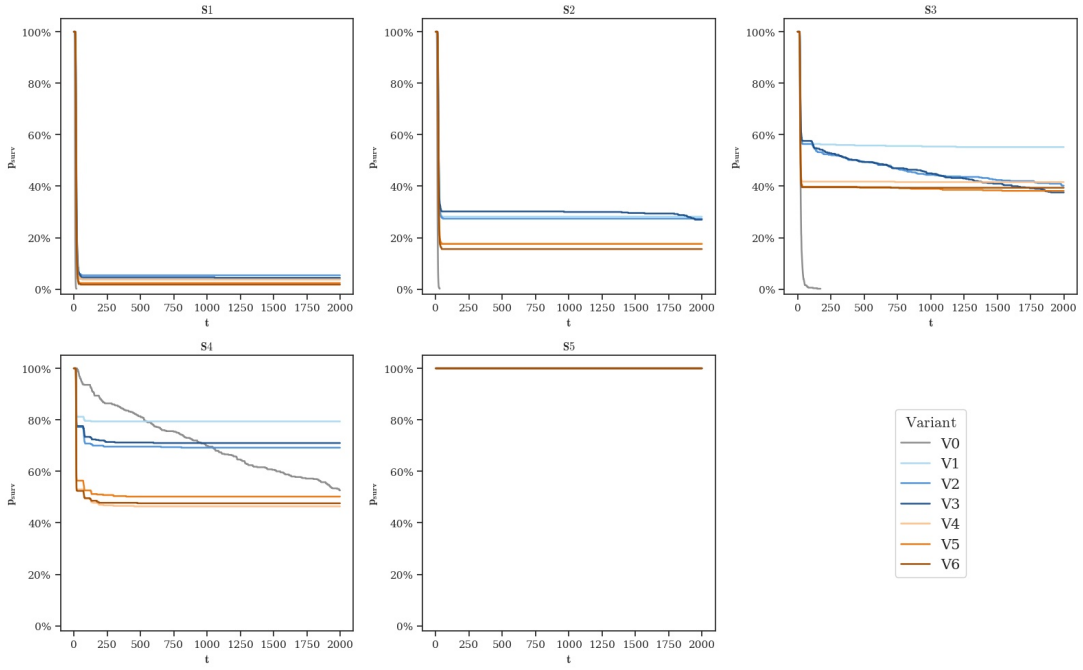

Figure 5: Rule-based agents: fraction of runs still alive at each step, across scenarios and variants. X-axis: simulation step t (0–2,000). Y-axis: surviving fraction (fraction of the 500 replications in which the population is still non-extinct at step t). Each panel corresponds to one scenario ($S1$–$S5$); lines are variants $V0$–$V6$, coloured according to the convention introduced in Figure 3.

keeps hedging the population against local resource fluctuations well beyond the founding phase. $S5$ is flat at 100%, as expected when survival pressure is absent.

Evolution of risk propensity. Figure 6 shows how mean α evolves during surviving runs.

$V4$ remains at α=1.0 throughout; in $S4$, $V6$ drifts upward to ≈1.25 while $V5$ drifts only slightly to ≈1.07. The randomly initialised variants drift more strongly, but unevenly: $V3$ reaches ≈1.48 and $V2$ ≈1.36, while $V1$ drifts least, to ≈1.18, as its random-offspring operator keeps resampling α toward the prior mean rather than concentrating it. Higher α is universally selected for among randomly initialised populations, while fixed-initialisation variants drift modestly in $S4$ and not at all in other scenarios. Notably, in both initialisation groups the online-learning variant ($V3$, $V6$) reaches higher α values than its evolutionary counterpart ($V2$, $V5$), indicating that gradient updates track the selection pressure more aggressively than evolutionary mutation over the simulation horizon. The mechanism is the interaction between movement frequency and resource dynamics: since agents consume at most 3 units per step while food accumulates up to 8, mobile populations routinely land on cells with at least 3 available units, while sedentary agents exhaust their current cell before moving.

The large cross-run standard deviation visible in $S1$–$S3$ should not be read as genuine within-population diversity: with only one to three agents surviving in each run, the apparent spread is just stochastic variation across isolated individuals rather than a coexistence of strategies. The situation changes in $S4$, where populations of around 25 agents are large enough to sustain true phenotypic coexistence, with different α values persisting side by side within the same run. In $S5$, instead, randomly initialised populations ($V1$–$V3$) converge to $\alpha \approx 1.24$–1.28 with substantial residual variance, while fixed-initialisation variants ($V4$–$V6$) remain close to 1.0: when resources are abundant, survival pressure is effectively absent and selection can no longer drive α upward from the initial value.

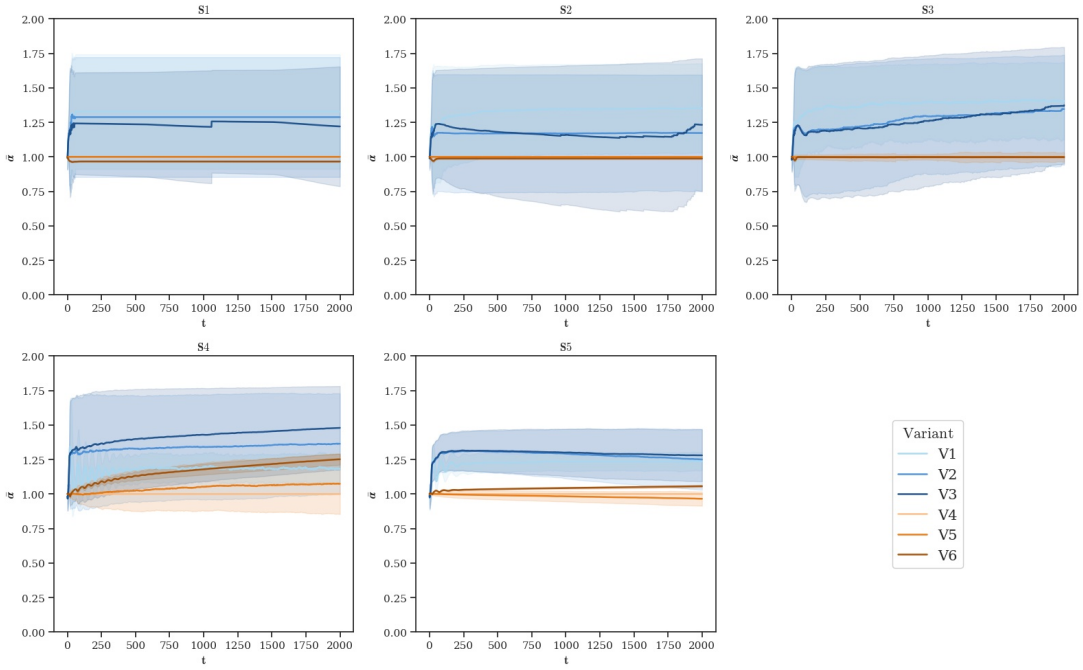

Figure 6: Rule-based agents: evolution of mean population risk propensity α over time for surviving runs. X-axis: simulation step t (0–2,000). Y-axis: cross-run mean of the population-mean α value (range 0–2). Shaded regions are cross-run standard deviations. Each panel corresponds to one scenario ($S1$–$S5$); lines are variants $V1$–$V6$, coloured according to the convention introduced in Figure 3 ($V0$ is omitted as it has no α parameter).

4.2. MLP-based agent-based model

Extinction rate. Figure 7 reports extinction rates for $V7$–$V13$ across 1–5 hidden layers. The results split sharply between randomly initialised ($V7$–$V10$) and offline-initialised ($V11$–$V13$) variants.

In $S1$–$S3$, randomly initialised variants $V7$–$V10$ reach ≈100% extinction regardless of depth, confirming that random MLP weights are essentially non-functional when the environment leaves no margin for error. The offline family behaves differently: in $S2$, $V11$–$V13$ reduce extinction mainly at lower depths (≈76–94%) but approach saturation by L=5; in $S3$, $V11$–$V12$ fall to ≈65–70%, while $V13$ remains higher at ≈77–88%. This suggests that online gradient updates can destabilise an already competent policy rather than refine it. $V10$ performs no better than $V7$–$V9$, consistent with online learning needing survival time to act, which $S1$–$S3$ provide only weakly.

In $S4$, $V7$–$V9$ show high extinction already at shallow depth and approach saturation as depth increases: $V7$ and $V8$ rise from ≈85–92% at 1 layer toward ≈100% by 3–5 layers, while $V9$ follows a similar trajectory, increasing from ≈86% to ≈99–100%. $V10$ is the exception among randomly initialised variants: it remains around ≈60–70%, with a slight worsening at higher depths rather than a monotonic rise. This suggests that, in the more favourable $S4$ environment, online learning can partly rescue an initially incoherent random policy, whereas fixed inheritance, random re-initialisation, and evolutionary mutation do not provide the same protection against the added difficulty introduced by depth. Among the offline family, $V11$, $V12$, and $V13$ perform comparably across depths (≈40–50%), with differences of a few percentage points and no consistent pattern: at 1 layer $V13$ is slightly the best (≈46%), while at other depths the ordering fluctuates. Depth has little effect within the offline family.

$S5$ reverses the pattern: $V11$–$V13$ achieve 0% extinction at all depths, while randomly initialised variants show strongly depth-dependent behaviour. $V7$ deteriorates steadily from ≈55% at 1 layer to ≈99% at 5 layers, consistent with a fixed lineage becoming increasingly likely to inherit a dysfunctional random founder policy as depth grows. An instructive crossover occurs between $V8$ and $V9$: at 1–2 hidden layers, $V8$ (random offspring) outperforms $V9$ (evolutionary) (≈8% vs ≈20% at L=1; ≈39% vs ≈48% at L=2), because in a small parameter space fresh random weights can still be viable. From 3 layers onward, the ordering reverses and $V9$ overtakes $V8$ (≈76% vs ≈85% at L=3), as random re-initialisation

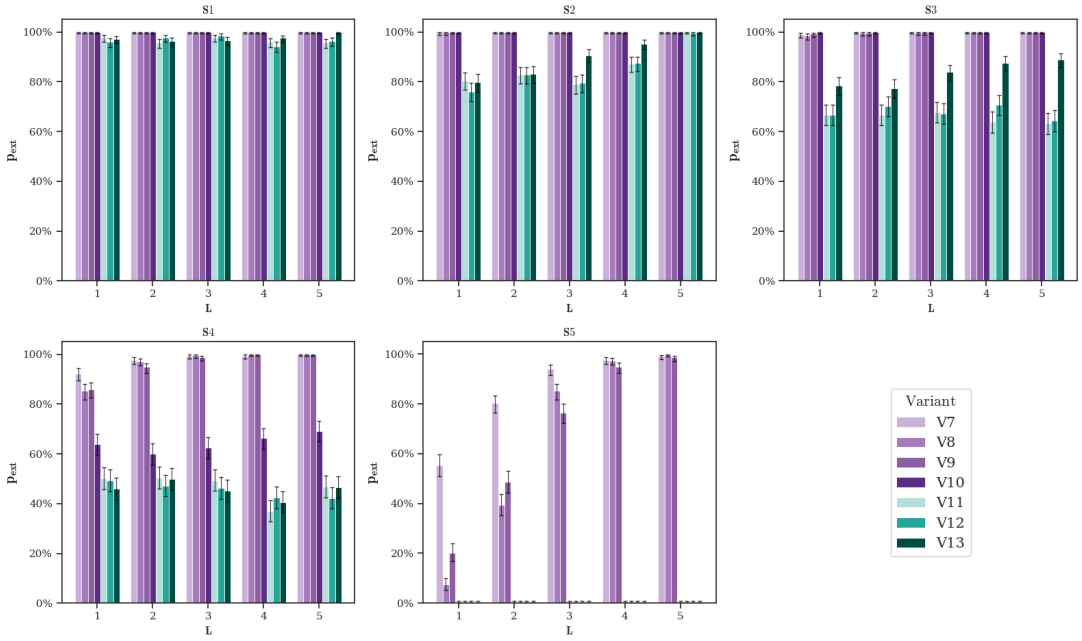

Figure 7: MLP agents: extinction rate by variant and number of hidden layers across all scenarios $S1$–$S5$. X-axis: number of hidden layers L (1–5). Y-axis: extinction rate. Each panel corresponds to one scenario ($S1$–$S5$); bars are coloured by initialisation class (one hue for randomly initialised variants $V7$–$V10$, another for offline-initialised variants $V11$–$V13$), with shade intensity distinguishing the adaptation operator. Error bars indicate 95% Wilson confidence intervals (n = 500 replications per condition).

becomes almost certainly lethal while evolutionary mutation preserves parental structure. The crossover suggests that random exploration is competitive in low-dimensional spaces but collapses in higher-dimensional ones. $V10$, by contrast, maintains near-zero extinction at all depths: the favourable environment appears to grant enough survival time for gradient updates to move the policy into a viable region, regardless of initial placement.

Relationship between offline accuracy and simulation performance. Figure 8 plots validation accuracy against extinction rate for each of the 15 independent BC training runs of $V11$ with 1 hidden layer.

Classification accuracy and ecological performance are largely disconnected. In $S2$–$S3$, accuracy exceeds 99.9% because "stay" dominates the dataset (≈99.8% in $S2$, ≈97.4% in $S3$). This is misleading: the network has learned the majority class, not the teacher's policy. A model always outputting "stay" would reach ≈99.8% accuracy in $S2$ while being indistinguishable from a frozen agent. The rare movement decisions, those that determine survival, are the ones the model is least reliable on, explaining high extinction (≈76–84% in $S2$, ≈62–73% in $S3$) despite near-perfect accuracy. In $S4$–$S5$ the dataset is more balanced ("stay" at ≈47% and ≈68%), accuracy reflects actual policy learning more faithfully, and extinction is lower; yet no correlation between accuracy and extinction is visible across runs.

$S1$ is the opposite extreme: accuracy drops to ≈63%, well above the majority-class baseline (≈27%) but far below the other scenarios. The task is intrinsically harder, energy is persistently low and uniform, and the teacher breaks ties randomly when neighbours share the highest food level, injecting irreducible label noise. The action distribution is nearly uniform (≈22–27% per direction, ≈2% stay), making it a genuine five-way classification. Across the 15 runs, no relationship between accuracy and extinction is visible.

In heavily imbalanced datasets the network may achieve high accuracy by learning to guess the majority class, with little gradient signal for the rare survival-critical decisions. The training data may also be consistent with multiple risk propensities: in $S2$–$S3$, any α below the movement threshold

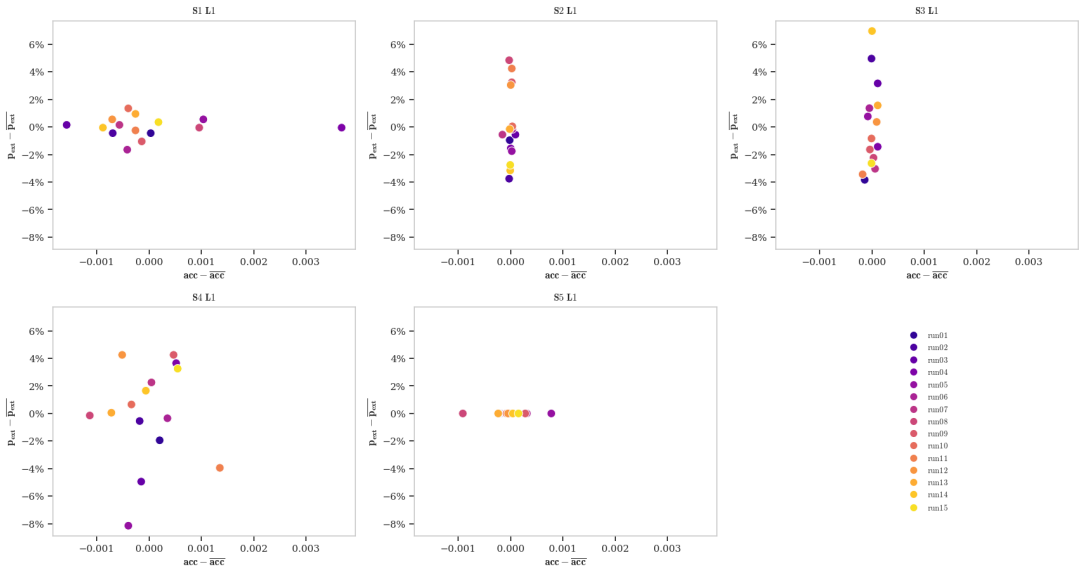

Figure 8: Accuracy of the offline-trained BC model versus extinction rate for $V11$ (1 hidden layer) across scenarios $S1$–$S5$. X-axis: deviation of validation accuracy from the per-scenario mean. Y-axis: deviation of extinction rate from the per-scenario mean. Each point is one of the 15 independent BC training runs, evaluated over 500 simulation replications; colours distinguish the individual runs (run01–run15). Each panel corresponds to one scenario ($S1$–$S5$).

generates "stay" on nearly all states; in $S1$, a high-α policy achieves similar accuracy to the teacher's cost-benefit rule. The dataset under-determines the policy, and different training runs can converge on functionally different strategies despite equivalent accuracy, explaining the scatter in Figure 8.

To avoid this confound, the offline checkpoint used to initialise the offline MLP variants ($V11$–$V13$) in the cross-variant comparisons was drawn at random from the corresponding set of BC training runs, rather than selected by in-simulation extinction rate. Selecting the best-performing run would have rewarded models that happened to approximate a coincidentally favourable risk propensity (Figure 2), conflating behavioural-cloning quality with luck in the learned strategy.

Population dynamics over time. Figure 9 shows mean population trajectories for surviving runs only. Only the 1-hidden-layer case is shown to keep the number of curves per panel manageable: as Figure 7 shows, depth matters mainly for randomly initialised variants, whose depth-dependent behaviour is already documented there.

In $S1$, all variants collapse to ≈1 agent. In $S2$–$S3$, among the surviving runs, $V13$ reaches ≈8–10 agents in $S2$ and ≈3–8 in $S3$, while $V11$ and $V12$ stabilise at 1–2; these conditional densities should not be read as policy quality indicators, since $V13$'s survival probability in $S2$–$S3$ is below 35% (Figure 10). In $S4$–$S5$, $V13$ overshoots sharply (peak ≈115 in $S4$, ≈175 in $S5$) while $V11$ and $V12$ converge without overshoot; all offline variants then damp to the scenario-specific equilibrium reached by the rule-based family.

Survival curves. Figure 10 shows the fraction of runs still alive at each step (again, 1-hidden-layer case).

In $S1$–$S3$, only $V11$–$V13$ maintain a non-negligible surviving fraction. In $S4$, $V11$–$V12$ stabilise at ≈51–54%; $V10$ declines more gradually than $V7$–$V9$, from an initial ≈80% at early steps to ≈38% at t=2,000, as REINFORCE updates accumulate stochastic drift. $V11$ (frozen weights) and $V12$ (small evolutionary perturbations) do not suffer this problem. $V7$–$V9$ collapse to ≈10–15%. In $S5$, offline variants and $V10$ hold near 100%; $V7$ declines most steeply to ≈45%, because with random initialisation and fixed inheritance, a dysfunctional founder can lock the lineage into a poor policy, a failure mode avoided by online learning ($V10$).

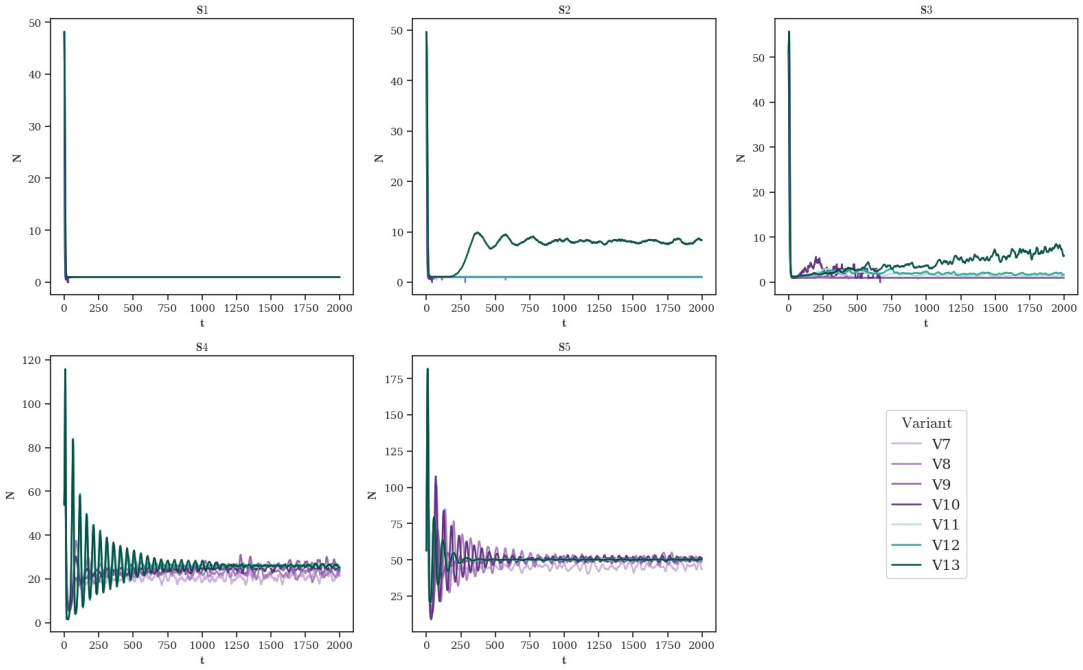

Figure 9: MLP agents: mean population over time for surviving runs (1 hidden layer only). X-axis: simulation step t (0–2,000). Y-axis: mean population size N, averaged across surviving replications (y-axis range differs across panels). Each panel corresponds to one scenario ($S1$–$S5$); lines are variants $V7$–$V13$, coloured according to the convention introduced in Figure 7.

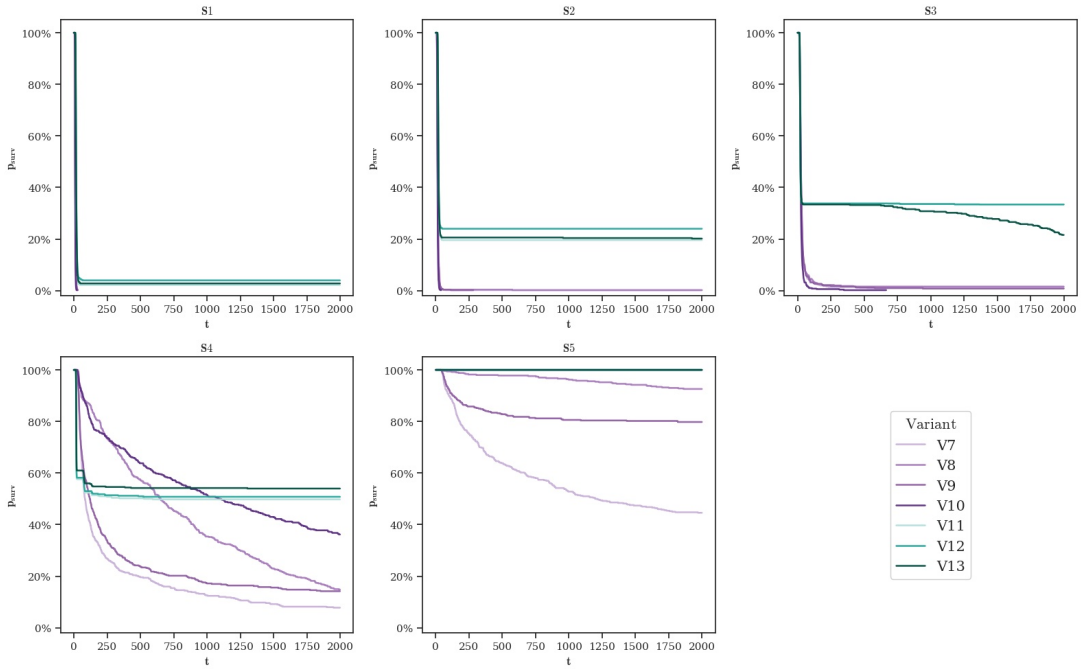

Figure 10: MLP agents: fraction of runs still alive at each step (1 hidden layer only), across scenarios and variants. X-axis: simulation step t (0–2,000). Y-axis: surviving fraction. Each panel corresponds to one scenario ($S1$–$S5$); lines are variants $V7$–$V13$, coloured according to the convention introduced in Figure 7.

4.3. Difference between rule-based and MLP-based agents

Initialisation dominates adaptation in both families, but through different mechanisms. For rule-based agents, "good initialisation" means population-level diversity: a random spread of α values provides bet-hedging, while a monomorphic start does not. For MLP agents, it means individual-level competence:

offline BC gives each agent a functional policy, whereas random weights in even a shallow MLP are far more likely to be lethal than a randomly drawn scalar α. As a result, MLPs exhibit a wider performance range and greater sensitivity to initialisation quality than rule-based agents, at a computational cost roughly 10–25× higher per agent-step and exceeding 100× for online-learning variants[3].

Offline MLP variants approach but do not exceed the best rule-based variant ($V1$), whose advantage stems from population-level heterogeneity in α, unavailable to a homogeneously trained MLP population. The comparison is asymmetric: offline MLPs imitate $V4$, a fixed teacher weaker than $V1$. The reported gap therefore reflects imitation of this specific teacher and does not establish an upper bound on offline BC performance.

Online learning also differs across families. In rule-based agents, gradient updates drive α upward slightly more aggressively than evolutionary mutation (Figure 6), yet this has little impact on extinction rates since $V2$ and $V3$ perform comparably (Figure 3): both random-initialised online and evolutionary variants converge on similarly adaptive populations. In MLP agents, the effect depends on initialisation: gradient updates partially rescue random weights in favourable scenarios ($V10$ at L=1) but destabilise offline-calibrated weights ($V13$). Network depth degrades randomly initialised MLPs while leaving offline variants unaffected. Two implications follow. First, online learning is not a general-purpose fix but a mechanism whose sign depends on where the policy starts: it helps when the alternative is random behaviour, and hurts when the starting policy is already competent. Second, the benefit of a larger architecture only materialises if initialisation places the network in a viable region of parameter space; otherwise, added depth simply enlarges the space of dysfunctional policies that inheritance and mutation must navigate.

5. Conclusions

Comparing rule-based and MLP-based policies across five foraging scenarios shows the empirical importance of separating initialisation from adaptation, a distinction under-explored in ML-ABM. Initialisation dominates: rule-based agents benefit from bet-hedging through heterogeneous α, while MLPs require competence from offline BC; random networks are nearly non-functional in adverse scenarios and worsen with depth. Gradient updates partially rescue randomly initialised MLPs in favourable scenarios but destabilise offline-initialised policies during the early bottleneck. One-step REINFORCE updates are noisy but immediate; accumulating rewards would reduce variance but could delay the first update beyond the agent's expected survival time. All surviving variants reach the same scenario-specific equilibrium, linking policy differences to bottleneck survival rather than carrying capacity. The deliberately simple environment enables a controlled comparison, but its lack of memory, planning, and strategic interaction may limit online adaptation.

Future work could test whether these findings persist in non-stationary, socially interactive environments or tasks requiring memory and planning. Further directions include comparing LLM-based agents' behaviour [39]; formalising the gradient-quality/survival-time trade-off; cloning alternative teachers (risk-averse, risk-seeking, or heterogeneous evolutionary populations); and broadening statistical metrics and behavioural indicators.

Declaration on Generative AI

During the preparation of this work, the authors used OpenAI Codex for code assistance, grammar and spelling checks, rewording, and formatting assistance. The authors reviewed and edited the outputs as needed and take full responsibility for the publication's content.

[3]All experiments were run on a MacBook Air M4 (16 GB RAM).

References

[1] S. Abar, G. K. Theodoropoulos, P. Lemarinier, G. M. O'Hare, Agent based modelling and simulation tools: A review of the state-of-art software, Computer Science Review 24 (2017) 13–33. doi:10.1016/j.cosrev.2017.03.001.

[2] F. Bertolotti, N. Kadera, L. Pasquino, L. Mari, An epidemiological extension of the el farol bar problem, Frontiers in Big Data 8 (2025) 1519369.

[3] H. Rahmandad, J. Sterman, Heterogeneity and network structure in the dynamics of diffusion: Comparing agent-based and differential equation models, Management science 54 (2008) 998–1014. doi:10.1287/mnsc.1070.0787.

[4] S. Roman, F. Bertolotti, Towards a new ai winter? how diffusion of technological innovation on networks leads to chaotic boom-bust cycles, Frontiers in Artificial Intelligence 8 (2025) 1671917.

[5] C. M. Macal, M. J. North, Tutorial on agent-based modelling and simulation, Journal of Simulation 4 (2010) 151–162. doi:10.1057/jos.2010.3.

[6] V. Grimm, S. F. Railsback, Agent-Based and Individual-Based Modeling: A Practical Introduction, Princeton University Press, Princeton, NJ, 2011.

[7] Y. Turgut, C. E. Bozdag, A framework proposal for machine learning-driven agent-based models through a case study analysis, Simulation Modelling Practice and Theory 123 (2023) 102707. doi:10.1016/j.simpat.2022.102707.

[8] A. Platas-Lopez, A. Guerra-Hernández, M. Quiroz-Castellanos, N. Cruz-Ramírez, Agent-based models assisted by supervised learning: a proposal for model specification, Electronics 12 (2023) 495. doi:10.3390/electronics12030495.

[9] M. Ale Ebrahim Dehkordi, J. Lechner, A. Ghorbani, I. Nikolic, É. Chappin, P. Herder, Using machine learning for agent specifications in agent-based models and simulations: A critical review and guidelines, Journal of Artificial Societies and Social Simulation 26 (2023) 9. doi:10.18564/jasss.5016.

[10] J. Dahlke, K. Bogner, M. Mueller, T. Berger, A. Pyka, B. Ebersberger, Is the juice worth the squeeze? machine learning (ml) in and for agent-based modelling (abm), arXiv preprint arXiv:2003.11985 (2020). doi:10.48550/arXiv.2003.11985.

[11] F. Bertolotti, S. Roman, F. Carucci, G. Buonanno, L. Mari, An llm-enhanced agent-based model of a sustainability game, in: Proceedings of the 26th Workshop From Objects to Agents (WOA2025), 2025, pp. 02–05.

[12] W. Zhang, A. Valencia, N.-B. Chang, Synergistic integration between machine learning and agent-based modeling: A multidisciplinary review, IEEE Transactions on Neural Networks and Learning Systems 34 (2021) 2170–2190. doi:10.1109/TNNLS.2021.3106777.

[13] C. Angione, E. Silverman, E. Yaneske, Using machine learning as a surrogate model for agent-based simulations, Plos one 17 (2022) e0263150. doi:10.1371/journal.pone.0263150.

[14] R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine Learning 8 (1992) 229–256. doi:10.1007/BF00992696.

[15] S. Ross, J. A. Bagnell, Efficient reductions for imitation learning, in: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), volume 9 of *Proceedings of Machine Learning Research*, PMLR, 2010, pp. 661–668. URL: https://proceedings.mlr.press/v9/ross10a.html.

[16] G. Jäger, Using neural networks for a universal framework for agent-based models, Mathematical and Computer Modelling of Dynamical Systems 27 (2021) 162–178. doi:10.1080/13873954.2021.1889609.

[17] D. L. DeAngelis, S. G. Diaz, Decision-making in agent-based modeling: A current review and future prospectus, Frontiers in Ecology and Evolution 6 (2019) 237. doi:10.3389/fevo.2018.00237.

[18] S. Roman, F. Bertolotti, Global history, the emergence of chaos and inducing sustainability in networks of socio-ecological systems, Plos one 18 (2023) e0293391.

[19] F. Bertolotti, A. Locoro, L. Mari, Sensitivity to initial conditions in agent-based models, in: European Conference on Multi-Agent Systems, Springer International Publishing Cham, 2020, pp.

501–508.

[20] U. Wilensky, W. Rand, Making models match: Replicating an agent-based model, Journal of Artificial Societies and Social Simulation 10 (2007) 2. URL: https://www.jasss.org/10/4/2.html.

[21] V. Grimm, S. F. Railsback, C. E. Vincenot, U. Berger, C. Gallagher, D. L. DeAngelis, B. Edmonds, J. Ge, J. Giske, J. Groeneveld, et al., The odd protocol for describing agent-based and other simulation models: A second update to improve clarity, replication, and structural realism, Journal of Artificial Societies and Social Simulation 23 (2020) 7. doi:10.18564/jasss.4259.

[22] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2016.

[23] G. Jäger, Replacing rules by neural networks: A framework for agent-based modelling, Big Data and Cognitive Computing 3 (2019) 51. doi:10.3390/bdcc3040051.

[24] N. Antulov-Fantulin, Towards agent-based-model informed neural networks, EPJ Data Science (2026). doi:10.1140/epjds/s13688-025-00616-z.

[25] W. Rand, Machine learning meets agent-based modeling: When not to go to a bar, in: D. Sallach, C. M. Macal, M. J. North (Eds.), Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects, University of Chicago and Argonne National Laboratory, Chicago, IL, 2006, pp. 51–58.

[26] S. H. Chen, P. Londoño-Larrea, A. S. McGough, A. N. Bible, C. Gunaratne, P. A. Araujo-Granda, J. L. Morrell-Falvey, D. Bhowmik, M. Fuentes-Cabrera, Application of machine learning techniques to an agent-based model of pantoea, Frontiers in Microbiology 12 (2021) 726409. doi:10.3389/fmicb.2021.726409.

[27] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed., MIT Press, Cambridge, MA, 2018.

[28] K. O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies, Evolutionary Computation 10 (2002) 99–127. doi:10.1162/106365602320169811.

[29] J. H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.

[30] T. Salimans, J. Ho, X. Chen, S. Sidor, I. Sutskever, Evolution strategies as a scalable alternative to reinforcement learning, arXiv preprint arXiv:1703.03864 (2017). doi:10.48550/arXiv.1703.03864.

[31] G. Hamon, E. Nisioti, C. Moulin-Frier, Eco-evolutionary dynamics of non-episodic neuroevolution in large multi-agent environments, in: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, 2023, pp. 143–146. doi:10.1145/3583133.3590703.

[32] G. Jäger, D. Reisinger, Can we replicate real human behaviour using artificial neural networks?, Mathematical and Computer Modelling of Dynamical Systems 28 (2022) 95–109. doi:10.1080/13873954.2022.2039717.

[33] D. A. Pomerleau, ALVINN: An autonomous land vehicle in a neural network, in: D. S. Touretzky (Ed.), Advances in Neural Information Processing Systems 1, Morgan Kaufmann, San Francisco, CA, 1988, pp. 305–313.

[34] S. Ross, G. Gordon, D. Bagnell, A reduction of imitation learning and structured prediction to no-regret online learning, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, volume 15 of *Proceedings of Machine Learning Research*, PMLR, 2011, pp. 627–635. URL: https://proceedings.mlr.press/v15/ross11a.html.

[35] J. M. Epstein, R. L. Axtell, Growing Artificial Societies: Social Science from the Bottom Up, MIT Press, Cambridge, MA, 1996.

[36] R. L. Axtell, J. M. Epstein, J. S. Dean, G. J. Gumerman, A. C. Swedlund, J. Harburger, S. Chakravarty, R. Hammond, J. Parker, M. Parker, Population growth and collapse in a multiagent model of the kayenta anasazi in long house valley, Proceedings of the National Academy of Sciences 99 (2002) 7275–7279. doi:10.1073/pnas.092080799.

[37] A. Hintze, R. S. Olson, C. Adami, R. Hertwig, Risk sensitivity as an evolutionary adaptation, Scientific reports 5 (2015) 8242. doi:10.1038/srep08242.

[38] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, 2015.

doi:10.48550/arXiv.1412.6980.

[39] R. Scantamburlo, M. Mezzanzana, G. Buonanno, F. Bertolotti, Semantic delta: An interpretable signal differentiating human and llms dialogue, arXiv preprint arXiv:2603.19849 (2026).