frontiers | Frontiers in Robotics and AI

TYPE Original Research
PUBLISHED 21 January 2025
DOI 10.3389/frobt.2024.1470886

OPEN ACCESS

EDITED BY
Giovanni Iacca,
University of Trento, Italy

REVIEWED BY
José Antonio Becerra Permuy,
University of A Coruña, Spain
Eric Medvet,
University of Trieste, Italy

*CORRESPONDENCE
Paolo Pagliuca,
paolo.pagliuca@istc.cnr.it

RECEIVED 26 July 2024
ACCEPTED 23 December 2024
PUBLISHED 21 January 2025

CITATION
Nolfi S and Pagliuca P (2025) Global progress in competitive co-evolution: a systematic comparison of alternative methods. *Front. Robot. AI* 11:1470886. doi: 10.3389/frobt.2024.1470886

Global progress in competitive co-evolution: a systematic comparison of alternative methods

Stefano Nolfi and Paolo Pagliuca*

Laboratory of Autonomous Robotics and Artificial Life (LARAL), Institute of Cognitive Sciences and Technologies (ISTC), National Research Council (CNR), Rome, Italy

The usage of broad sets of training data is paramount to evolve adaptive agents. In this respect, competitive co-evolution is a widespread technique in which the coexistence of different learning agents fosters adaptation, which in turn makes agents experience continuously varying environmental conditions. However, a major pitfall is related to the emergence of endless limit cycles where agents discover, forget and rediscover similar strategies during evolution. In this work, we investigate the use of competitive co-evolution for synthesizing progressively better solutions. Specifically, we introduce a set of methods to measure historical and global progress. We discuss the factors that facilitate genuine progress. Finally, we compare the efficacy of four qualitatively different algorithms, including two newly introduced methods. The selected algorithms promote genuine progress by creating an archive of opponents used to evaluate evolving individuals, generating archives that include high-performing and well-differentiated opponents, identifying and discarding variations that lead to local progress only (i.e., progress against the opponents experienced and retrogressing against others). The results obtained in a predator-prey scenario, commonly used to study competitive evolution, demonstrate that all the considered methods lead to global progress in the long term. However, the rate of progress and the ratio of progress versus retrogressions vary significantly among algorithms. In particular, our outcomes indicate that the Generalist method introduced in this work outperforms the other three considered methods and represents the only algorithm capable of producing global progress during evolution.

KEYWORDS

competitive co-evolution, evolutionary robotics, local historical and global progress, open-ended evolution, predator-prey robots

1 Introduction

Recent advances in machine learning have demonstrated the importance of using large corpora of training data. In the context of embodied and situated agents, this implies placing the agents in complex and diversified environments. However, manually designing environments of this kind is both challenging and costly. A convenient alternative is constituted by multi-agent scenarios where adaptive agents are situated in environments with

other adaptive agents with conflicting goals—a method known as competitive co-evolution (Rosin and Belew, 1995) or self-play (Bansal et al., 2017). In these settings, learning agents are exposed to continuously varying environmental conditions due to the behavioral changes of other adaptive agents. In other words, these settings allow for the automatic generation of a large corpus of training data.

Competitive settings also offer other important advantages. They can spontaneously produce convenient learning curricula (Rosin and Belew, 1997; Wang et al., 2021) in which the complexity of the training conditions progressively increases while the skills of the agents—and consequently their ability to master complex conditions—also improve. Finally, such settings may naturally generate a form of adversarial learning (Lowd and Meek, 2005), where the training data are shaped to challenge the weaknesses of the adaptive agents.

Unfortunately, competitive co-evolution does not necessarily result in a progressive complexification of agents' skills or environmental conditions. As highlighted by Dawkins and Krebs in the context of natural co-evolution (Dawkins and Krebs, 1979), the evolutionary process can lead to four distinct long-term dynamics: (1) extinction: one side may drive the other to extinction, (2) definable optimum: one side might reach a definable optimum, preventing the other side from reaching its own optimum, (3) mutual local optimum: both sides may reach a mutual local optimum, and (4) endless limit cycle: the race may persist in a theoretically endless limit cycle, in which similar strategies are abandoned and rediscovered over and over again.

Regrettably, pioneering attempts to evolve competing robots consistently yield the last undesirable outcome described by Dawkins and Krebs. Initially, there is true progress, but subsequently agents modify their strategies, resulting in apparent progress. In other words, they improve against the strategies exhibited by their current opponents while retrogressing against other strategies that are later adopted by opponents (Miconi, 2008). Consequently, limit cycle dynamics emerge, where the same strategies are abandoned and rediscovered repeatedly (Sinervo and Lively, 1996; Miconi, 2009; Nolfi, 2012).

The generation of genuine progress requires the usage of special algorithms that: (i) expose the adapting agents to both current and ancient opponents (Rosin and Belew, 1997), (ii) expose the adaptive agents to a diverse range of opponents (De Jong, 2005; Simione and Nolfi, 2021), and (iii) identify and retain variations that lead to true progress only (Simione and Nolfi, 2021). Furthermore, analyzing competitive settings necessitates the formulation of suitable measures to differentiate between true and apparent progress and to evaluate the efficacy of the obtained solutions.

In this article, we present a systematic comparison of alternative competitive co-evolutionary algorithms, including novel variations of state-of-the-art algorithms. We describe the measures that can be used to analyze the co-evolutionary process, discriminate between apparent and true progress, and compare the efficacy of alternative algorithms. Finally, we analyze whether the proposed methods manage to avoid limit-cycle dynamics.

2 Materials and methods

2.1 Measuring progress

In evolutionary experiments in which the evolving individuals are situated in solitary environments, the fitness measured during individuals' evaluation provides a direct and absolute measure of performance. Fitness evaluation typically includes a stochastic factor due to random variations of the initial state of the robot and of the environment. However, these variations are not adversarial in nature. Consequently, their impact can be usually be mitigated by evolving solutions that are robust to these variations and by averaging the fitness obtained during multiple evaluation episodes (Jakobi et al., 1995; Carvalho and Nolfi, 2023; Pagliuca and Nolfi, 2019).

In competitive social settings, instead, the fitness obtained during individuals' evaluation crucially depends on the opponent(s) situated in the same environment. Consequently, the method used to select opponents has a pivotal effect on the course of the co-evolutionary process.

The dependency of the fitness measure on the characteristics of the opponents also affects: (1) the identification of the best solution generated during an evolutionary process, (2) the estimation of the overall effectiveness of a solution, and (3) the comparison of the efficacy of alternative experimental conditions.

The former two problems can be tackled by identifying a specific representative set of opponents known as "champions," which typically include the best opponents produced during independent evolutionary experiments. The third issue can be resolved through a technique called "cross-test." In a cross-test, the best solutions obtained from N independent evolutionary experiments conducted under one experimental condition are evaluated against the best opponents obtained from N different independent evolutionary experiments.

The dependence of the fitness measure on the opponent characteristics impacts the method used to measure evolutionary progress. In non-competitive settings, progress and retrogression can be straightforwardly measured by computing the variation of fitness across generations. Instead, in competitive settings, measuring progress becomes more challenging.

As highlighted by Miconi (2009), we need to differentiate between three types of progress: (i) local progress, i.e., progress against current opponents, (ii) historical progress, i.e., progress against opponents of previous generations, and (iii) global progress, i.e., progress against all possible opponents. Local progress can be measured by evaluating agents against opponents from recent preceding generations. Historical progress can be assessed by evaluating agents against opponents from previous generations. These data can be effectively visualized using the "Current Individual against Ancestral Opponents" (CIAO) plots introduced by (Cliff and Miller 1995; Cliff and Miller, 2006). Finally, global progress can be estimated by evaluating agents against opponents generated in independent evolutionary experiments—opponents that differ from those encountered during the evolutionary process. Additionally, an indication of global progress can be obtained by evaluating agents against opponents from future generations. The data obtained by post-evaluating agents against opponents of previous and future generations can be conveniently visualized using the master tournament plots introduced by Nolfi and Floreano (1998).

2.2 Competitive evolutionary algorithms

In this section we will review the most interesting co-evolutionary algorithms described in the literature and the methods that we will compare in our experiments. These algorithms can be applied to the evolution of two species that reciprocally affect each other.

We focus our analysis on evolutionary algorithms attempting to maximize the expected utility, i.e., the expected fitness against a randomly selected opponent or the average fitness against all possible opponents. Other researchers have explored the use of competitive evolution for synthesizing Nash equilibrium solutions (Ficici and Pollack, 2003; Wiegand et al., 2002) and Pareto-optimal solutions (De Jong, 2004).

As previously mentioned, achieving true progress necessitates the utilization of specialized algorithms that: (i) expose adapting agents to both current and ancient opponents, (ii) evaluate adaptive agents against a diverse set of opponents, or (iii) identify and retain variations that lead to genuine progress.

The first method we consider is the Archive algorithm, as introduced by Rosin and Belew (1997), see also (Stolfi et al., 2021). The pseudocode of the Archive algorithm is provided in Algorithm 1. In this algorithm, a copy of the best individual from each generation is stored in a "hall-of-fame" archive. Opponents are then randomly selected from this archive using a uniform distribution. Evaluating evolving agents against opponents from previous generations clearly promotes historical progress. While the production of global progress is not guaranteed, it can be expected as a form of generalization. Indeed, the need to defeat an increasing number of ancient opponents should encourage the development of strategies that generalize to opponents not yet encountered.

The second method that we will consider is the Maxsolve* algorithm (see Algorithm 2), a variation of the original method introduced by De Jong (2005), see also (Samothrakis et al., 2013; Liskowski and Krawiec, 2016). The Maxsolve algorithm

```
1. n_parents = 1, n_offspring = 40, mutation_range
= 0.02, learning_rate = 0.01, max_total_steps =
75 · 10^9
2. initializethe genotype of the parents (predator
and prey) randomly
3. initializethe archives (predator and prey) with
10 genotypes generated randomly
4. whiletot_steps < max_total_steps
5.    selected_opponents = select10 opponents
      randomly from the archive
6.    generateoffspring
7.    fitness = evaluateoffspring against
      selected_opponents
8.    computegradient
9.      updateparent
10.      appendthe fittest offspring to the
        archive
11.      current_generation + = 1
```

Algorithm 1. Archive algorithm.

```
1. n_parents = 1, n_offspring = 40,
max_archive_size = 25000, mutation_range = 0.02,
learning_rate = 0.01, max_total_steps = 75 · 10^9
2. initializethe genotype of the parents (predator
and prey) randomly
3. initializethe archives (predator and prey) with
10 genotypes generated randomly
4. whiletot_steps < max_total_steps
5.    selected_opponents = select10 opponents
      randomly from the archive
6.    generateoffspring
7.    fitness = evaluateoffspring against
      selected_opponents
8.    store fitness_data[agent][opponent]
9.    computegradient
10.    updateparent
11.    appendthe fittest offspring to the archive
12.    current_generation + = 1
13.    ifsize(archive) > 25000
14.      removea dominated agent from the archive
          (see text for details)
```

Algorithm 2. Maxsolve* algorithm.

operates by using an archive containing a predetermined maximum number of opponents. The size of the archive is kept bounded by removing dominated and redundant opponents from it. The domination criterion works for transitive problems, i.e., in problems where agents A outperforming agents B, which in turn outperform agents C, necessarily outperform agents C. To allow the algorithm to operate with non-transitive problems, like the predator and prey task considered in this paper, we implemented the Maxsolve* algorithm, a variation of the original Maxsolve algorithm, which retains in the archive the individuals achieving the highest performance, on average, against the opponents of the 10 preceding phases, where each phase corresponds to $\frac{1}{10}$ of the total number of generations. The method thus attempts to automatically select high-quality champions that can promote the discovery of high-quality solutions and minimize the time spent evaluating agents against poor opponents (for a related approach, see Bari et al., 2018). Clearly, the size of the archive plays an important role in this method. Indeed, the smaller the size of the archive is, the smaller the training data is. On the other hand, the smaller the size of the archive is, the higher the minimization of the time spent against poor opponents is. By systematically varying the size of the archive (data not shown), we observed that the best results are obtained by using relatively large archives, i.e., archives containing up to 25,000 opponents.

The third method is the Archive* algorithm, a novel approach introduced in this paper as a variation of the original Archive algorithm described earlier. The Archive* algorithm operates by evolving N independent populations, each contributing to the generation of a single archive. These evolving populations are then evaluated against opponents selected

```
1. n_parents = 10, n_offspring = 40,
mutation_range = 0.02, learning_rate = 0.01,
max_total_steps = 75 • 10^9
2. initializethe genotype of the parents (predator
and prey) randomly
3. initializethe archives (predator and prey) with
10 genotypes generated randomly
4. whiletot_steps < max_total_steps
5.   forp in range (n_parents)
6.     selected_opponents = select10 opponents
       randomly from the archive
7.     generateoffspring [p]
8.     fitness [p] = evaluateoffspring [p]
       against selected_opponents
9.     computegradient [p]
10.     updateparent [p]
11.     appendthe fittest offspring [p] to
       the archive
12.     current_generation + = 1
```

Algorithm 3. Archive* algorithm.

from the shared archive. The pseudocode of the method is reported in Algorithm 3.

The rationale behind this method lies in the use of multiple populations, which enhances the diversification of opponents included in the archive. Specifically, the Archive* method automatically generates multiple families of diversified champions, representing alternative challenges for the evolving agent. The Archive* algorithm shares similarities with the population-based reinforcement learning method proposed by Jaderberg et al. (2018).

Finally, the fourth method we propose is the Generalist algorithm, introduced by Simione and Nolfi (2021) (see Algorithm 4). In this approach, agents are evaluated against a subset of opponents, while the remaining opponents serve to discriminate between agents who retained variations leading to global progress and those who retained variations leading to local progress. This information guides the preservation or discarding of individuals. The subset of opponents used for agent evaluation is randomly selected at regular intervals (every N generations) to maximize the functional diversity of the subset. Unlike the previous algorithms, the Generalist method does not rely on archives.

Another potential approach involves using randomly generated opponents (Chong et al., 2009; 2012; Jaśkowski et al., 2013). The primary advantage of this technique lies in the direct promotion of global progress because agents are consistently evaluated against new opponents. However, there is a significant drawback: the efficacy of these opponents does not improve over generations. Consequently, this method does not allow for the development of agents capable of defeating strong opponents. In fact, the performance obtained using this approach by Samothrakis et al. (2013) were considerably lower than the performance achieved with a variation of the Maxsolve algorithm described earlier.

The methods described above are meta-algorithms that should be combined with an evolutionary algorithm to determine how populations of individuals vary across generations. In previous

```
1. n_parents = 80, n_offspring = 40,
mutation_range = 0.02, learning_rate = 0.01,
max_total_steps = 75 • 10^9
2. initializethe genotype of the parents (predator
and prey) randomly
3. whiletot_steps < max_total_steps
4.  every20 generations
5.   performance [n_parents] [n_parents] =
evaluate(all_parents, all_opponents)
6.   selected_parents = select10 parents randomly
7.   selected_opponents = selectthe 10 opponents
with the highest performance against the 10
selected parents
8.   candidate_parents []
create-a-copy-ofselected_parents []
9.  forp in range (candidate_parents)
10.   generateoffspring [p]
11.   fitness [p] = evaluateoffspring [p] against
selected selected_opponents
12.   computegradient [p]
13.   updatecandidate_parent [p]
14.   every20 generations
15.   performance [n_selected_parents] [n_parents]
= evaluate(selected_parents, all_opponents)
16.   replacethe worst parents with the best
candidate_parents which outperform them.
17.   current_generation + = 1
```

Algorithm 4. Generalist algorithm.

studies, standard evolutionary algorithms or evolutionary strategies were employed. Instead, in this work, we utilize the OpenAI-ES algorithm (Salimans et al., 2017), which represents a "modern" evolutionary strategy (Pagliuca et al., 2020). The OpenAI-ES algorithm leverages the matrix of variations introduced within the population and the fitness values obtained by corresponding individuals to estimate the gradient of fitness. It then guides the population's movement in the direction of this gradient using a stochastic optimizer (Kingma and Ba, 2014). Importantly, this algorithm is well-suited for non-stationary environments. This is because the momentum vectors also guide the population in the direction of previously estimated gradients, thereby enhancing the possibility of generating solutions effective against opponents encountered in previous generations.

Below we include the pseudo-code of the algorithms. In all cases the connection weights of the controllers of the robots were evolved by using the OpenAI-ES algorithm (Salimans et al., 2017) and by using the hyperparameters indicated in the reference. More specifically, observation vectors were normalized by using virtual batch normalization (Salimans et al., 2016; Salimans et al., 2017), the connections weights were normalized by using weight decay, the distribution of the perturbations of parameters was set to 0.02, and the step size of the Adam optimizer was set to 0.01. The fitness gradient was estimated by generating 40 offspring, i.e., 40 perturbed versions of the parent. The fitness of the evolving

individuals corresponds to the average fitness obtained during 10 episodes in which they are evaluated against 10 different opponents in 10 corresponding evaluation episodes. The fitness and the perturbation vectors are used to compute the gradient of the expected fitness, which is used to update the parameters of the parent through the Adam (Kingma, and Ba, 2014) stochastic optimizer. The predator and prey robots evolved in parallel. The evolutionary process is continued until the total number of evaluation steps performed exceeds $75 \bullet 10^9$.

In the case of the Archive and Maxsolve* algorithms, the parameters of the evolving robots were generated from a single parent (a parent for the predator and a parent for the prey robots). In the case of the Archive* algorithm, they are generated from 10 parents. The generalist algorithm, instead, uses a population of 80 parents, evolves 10 candidate parents generated by creating a copy of 10 parents selected randomly every 20 generations, and replace the worst parents with the candidate parents outperforming them. The performance of parents and of candidate parents are computed by evaluating the parents against the full set of 80 opponents, i.e., by evaluating the candidate parents also against opponents not encountered in the preceding generations. The steps performed to evaluate performance contribute to increase the total number of steps performed.

The Archive and Archive* algorithms preserve the best robots of each generation in archives that keep increasing in size during the evolutionary process. The Maxsolve* algorithm uses archives that can grow up to a maximum size only. This is realized by: (i) storing in a fitness_data [X][Y] matrix the average fitness obtained by agents of generation X against opponents of generation Y, (ii) computing the average fitness obtained by agents contained in the archive against opponents of 10 subsequent evolutionary phases, and (iii) eliminating one dominated agent selected randomly from the archive in each generation. An agent is dominated when the average fitness obtained against opponents of different phases is consistently equal or lower than the average fitness obtained by another agent included in the archive.

2.3 The predator and prey problem

We chose to compare alternative algorithms using a predator and prey problem because it represents a challenging scenario (Miller and Cliff, 1994). Additionally, this problem is widely used for studying competitive evolutionary algorithms (Miller and Cliff, 1994; Floreano and Nolfi, 1997a; Floreano and Nolfi, 1997b; Floreano et al., 1998; Nolfi and Floreano, 1998; Stanley and Miikkulainen, 2002; Buason and Ziemke, 2003; Buason et al., 2005; Jain et al., 2012; Palmer and Chou, 2012; Ito et al., 2013; Georgiev et al., 2019; Lan et al., 2019; Lee et al., 2021; Simione and Nolfi, 2021; Stolfi et al., 2021).

The predator and prey problem presents extremely dynamic, highly unpredictable, and hostile environmental conditions. Consequently, it necessitates the development of solutions that are rapid, resilient and adaptable. Moreover, agents must exhibit a range of integrated behavioral and cognitive capabilities, including avoiding stationary and moving obstacles, optimizing motion trajectories under multiple constraints, integrating sensory information over time, anticipating opponent behavior, disorienting opponents, and adapting behavior in real time based on the opponent's actions (Humphries and Driver, 1970).

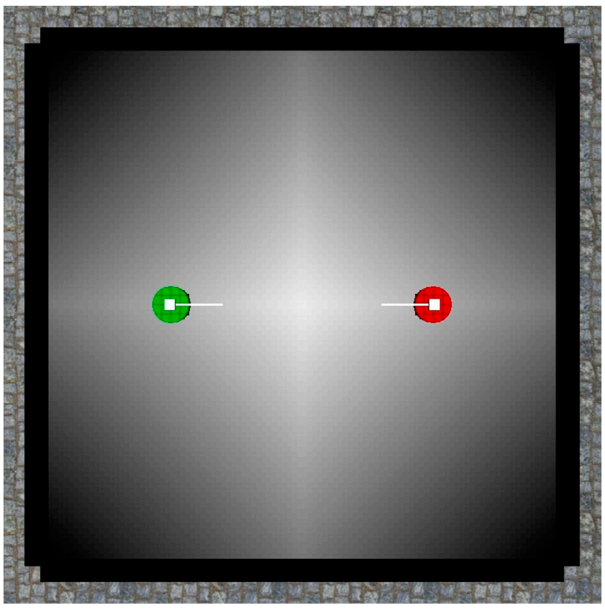

FIGURE 1
The robots and the environment in simulation. The red and green robots correspond to the predator and prey robots, respectively.

The robots in our study are simulated MarXbots (Bonani et al., 2010) equipped with neural network controllers. The connection strengths within the robots' neural networks, which determine their behavior, are encoded in artificial genotypes and evolved. Specifically, predators are evolved to enhance their ability to capture prey (i.e., reach and physically touch the prey) as quickly as possible, while prey are evolved to maximize their ability to avoid being captured for as long as possible. The fitness of predators corresponds to the fraction of time required to capture the prey (see Equation 1). The fitness of preys is the inverse of the fraction of time required by the predator to capture them (see Equation 2).

$$F_{pred} = \frac{NSteps - CStep}{NSteps} \quad (1)$$

$$F_{prey} = 1.0 - F_{pred} \quad (2)$$

In the equations above, *NSteps* denotes the maximum length of the evaluation episode (1000 in our experimental scenarios), *CStep* represents the step in which the predator succeeds in capturing the prey, F_{pred} is the fitness of the predator, while F_{prey} indicates the performance of the prey.

The simulated MarXbots are circular robots with a 17 cm diameter. They are equipped with a differential drive motion system, a ring of 24 color LEDs, 24 infrared sensors, 4 ground sensors, an omnidirectional camera, and a traction sensor. In the experiments, the LEDs of predator robots are set in red, while the LEDs of prey robots are set in green. The robots were placed in a 3 × 3 m square arena surrounded by black walls. The arena floor was grayscale, varying from white to black from the center to the periphery (see Figure 1).

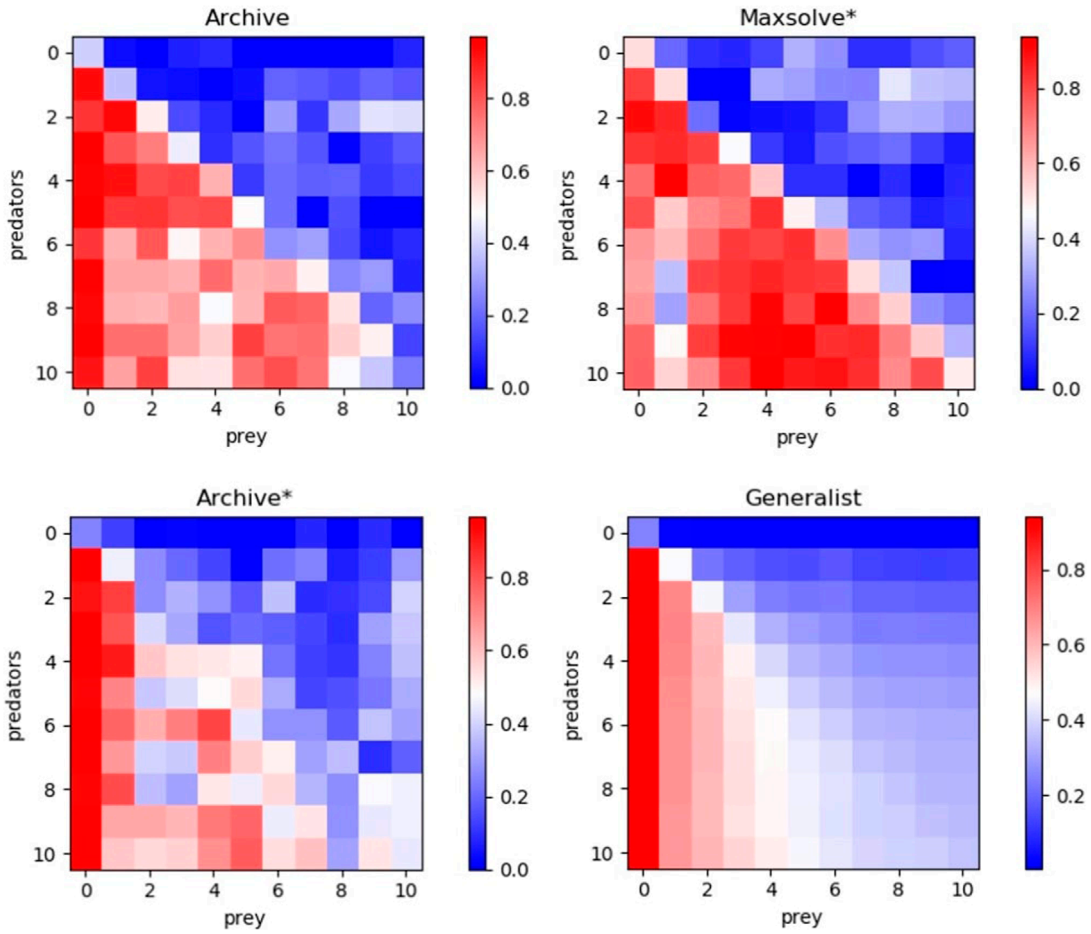

FIGURE 2
Performance of predators of different evolutionary phases evaluated against opponents of all phases. Results obtained with the Archive, Maxsolve*, Archive*and Generalist algorithms. Data was collected by evaluating the predators of phase 0 (generation 0) and the subsequent 10 phases against prey from phase 0 (generation 0) and the following 10 phases. The vertical and horizontal axes represent the phases of the predator and prey, respectively. The color of each cell indicates the average performance of the predator against prey from the corresponding phases. The performance of the prey is the inverse of the predator's performance, i.e., it corresponds to 1.0 minus the predator's performance (see Equation 2). The phases are separated by $\frac{1}{10}$ of the total generations. The results are averaged over 10 evolutionary experiments. Generally, the performance of predators from any given phase (shown in any given row) improves (becomes redder) against opponents from previous generations (displayed in the first columns) and declines (becomes bluer) against opponents from successive generations (displayed in the last columns). Similarly, the performance of prey from any given phase (displayed in any given column) improves (becomes bluer) against opponents from previous phases (shown in the first rows) and worsens (becomes less blue) against opponents from successive phases (displayed in the last rows).

The maximum wheel speed that the wheels of the differential drive motion system could assume was 10 rad/s for the prey and 8.5 rad/s for the predators respectively. The relative speed of the two robot types was adjusted to balance the overall complexity of the problem faced by predators and preys, i.e., preventing one species from reaching maximum or minimum fitness. The environment state, robot sensors, motors, and neural network were updated at a frequency of 10 Hz.

The neural network controller of the robot consists of a LSTM (Long Short-Term Memory, see Hochreiter and Schmidhuber, 1997; Gers and Schmidhuber, 2001) recurrent neural network with 23 sensory neurons, 25 internal units, and 2 motor neurons. The sensory layer includes 8 sensory neurons encoding the average activation state of eight groups of three adjacent infrared sensors, 8 neurons that encode the fraction of green or red light perceived in the eight 45° sectors of the visual field of the camera, 1 neuron that encodes the average amount of green or red light detected in the entire visual field of the camera, 4 neurons encoding the state of the four ground sensors, 1 neuron that encodes the average activation of the four ground sensors, 1 neuron that encodes whether the robot collides with an obstacle (i.e., whether the traction force detected by the traction sensor exceeds a threshold). The sensory neuron states are normalized in the range [0.0, 1.0]. The motor layer includes two neurons encoding the desired translational and rotational motion of the robot in the range [0.0, 1.0].

To compare the relative effectiveness of the considered methods, we continued the evolutionary process until a total of 75 billion evaluation steps were performed. This ensures that the comparison of the different techniques is fair.

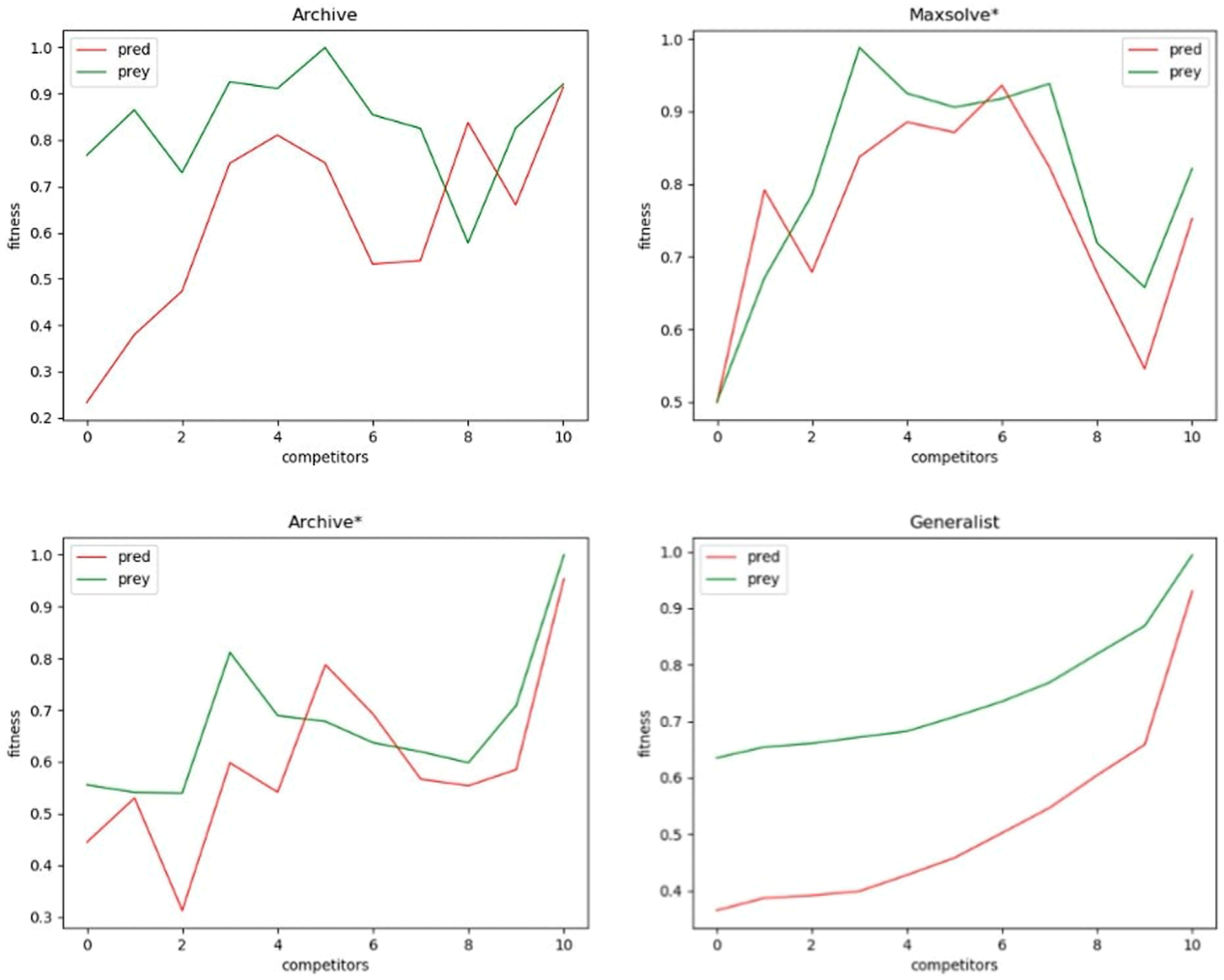

FIGURE 3
Performance of predators (red lines) and prey (green lines) from the last generation against opponents from the same generation (0) and opponents from the 10 preceding phases (1–10), where 10 corresponds to opponents from the initial generation. The 10 phases are separated by $\frac{1}{10}$ of the total generations. The vertical axes represent the average performance while the horizontal axes represent the preceding phase of the opponents, where 0 correspond to opponents from the last generation and 10 corresponds to opponent from generation 0. Results were obtained using the Archive, Maxsolve*, Archive*, and Generalist algorithms. Each plot represents the average results of 10 evolutionary experiments. These plots display the same data as shown in the last row and last column of the matrices presented in Figure 2.

The experiments included in this article can be replicated using the Evorobotpy2 tool which is available from the Github repository https://github.com/snolfi/evorobotpy2. The source code of the co-evolutionary algorithms and of the associated experiments is available from the Github repository https://github.com/snolfi/competitive-evolution.

3 Results

In this section, we present the results obtained using the Archive, Maxsolve*, Archive*, and Generalist algorithms. The results include data collected from 10 replication experiments conducted with each algorithm, resulting in a total of 40 experiments.

Figure 2 displays master tournament data, i.e., the performance of predator and prey robots of different generations evaluated against opponents of previous, current, and future generations. The results demonstrate that all considered methods exhibit historical progress overall. Notably, the robots perform better against opponents from previous generations than against those from successive generations in most cases. Additionally, these data provide an indication of global progress, as the robots of generation X + N perform better against opponents from future generations than robots of generation X in most of the cases. However, it is worth noting that only the Generalist algorithm consistently produces progress across all phases, resulting in monotonically better robots. The other algorithms also exhibit retrogressions, albeit less frequently than progress.

This qualitative difference is confirmed by Figure 3, which illustrates the performance of the robots from last generation against opponents from both the current and previous generations. The Archive and Maxsolve* robots of the last generations consistently

TABLE 1 The cross-test of champion agents conducted using the Archive, Maxsolve*, Archive*, and Generalist algorithms.

Predators				
	Archive	Maxsolve*	Archive*	Generalist
Archive				
MaxSolve*	0.05, p = .270			
Archive*	**0.15, p = .001**	0.04, p = .112		
Generalist	**0.12, p = .007**	**0.13, p = .001**	**0.14, p = .001**	
Preys				
	Archive	Maxsolve*	Archive*	Generalist
Archive				
MaxSolve*	0.04, p = .933			
Archive*	**0.17, p = .003**	0.12, p = .044		
Generalist	**0.27, p = .001**	**0.23, p = .001**	**0.24 p = 0.001**	

In each row, we evaluate the performance of agents evolved under the condition indicated in that row against opponents evolved under the condition indicated in the corresponding column. We then subtract the performance obtained by evaluating the agents against the opponents indicated in the same row. Positive values indicate that the condition indicated in the row outperforms the condition indicated in the column. The numbers denoted by "p = " represent the probability that the performance obtained against the two sets of opponents belongs to the same distribution. Values in bold indicate cases where the difference in performance is statistically significant (Mann–Whitney U-test with Bonferroni correction, p-value <0.0167). The table is divided into two parts: the top section displays cross-tests using predators as agents and preys as opponents, while the bottom section reverses the roles.

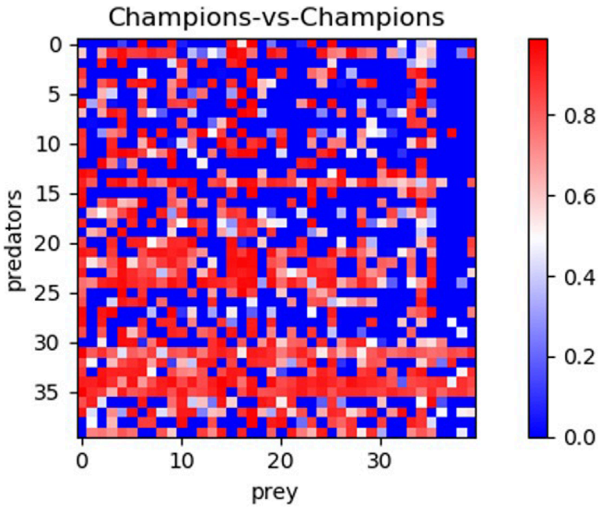

FIGURE 4
Post-evaluation of the 40 predator champions obtained using four different methods against the 40 prey champions obtained using the same methods. The table consists of four sets of 10 rows and columns, representing the performance of predator and prey champions from the Archive (rows 0–9 and columns 0–9), Maxsolve*(rows 10–19 and columns 10–19), Archive*(rows 20–29 and columns 20–29), and Generalist (rows 30–39 and columns 30–39) methods. Each champion is selected from a corresponding replication. Each pixel's color indicates the predator's performance, while the prey's performance corresponds to the inverse of the predator's one.

exhibit better and better performance against opponents of the last four preceding phases only. The Archive*robots of the last generations consistently display better and better performance against opponents of the last 3 phases (for predator robots) and the last 5 phases (for prey robots) only. Only robots evolved with the Generalist algorithm consistently exhibit improved performance against older opponents across all phases.

To identify the best performing method, we conducted cross-tests comparing the champions of each algorithm against those of each other algorithms. Each champion was selected from the best predators and prey of the last 40 generations of the corresponding replication. Consequently, we have 10 champion predators and 10 champion prey for each experimental condition. The cross-tests were conducted by comparing the performance of a set of 10 champion agents evolved under one experimental condition against champion opponents evolved under the same or a different experimental condition. More specifically, cross-tests values were computed according to Equation 3:

$$c_{12} = \sum_{i=1,j=1}^{10} F\left(A_i^1 \mid O_j^1\right) - F\left(A_i^1 \mid O_j^2\right) \quad (3)$$

where c is the cross-test value, *1* and *2* denote the experimental conditions being compared, $F()$ indicates the performance (fitness), A represents the agents, O represents the opponents, i and j are indices for the 10 champion agents and opponents, respectively. The notation (x |y) indicates the evaluation of an individual x against the opponent y. The symbol x_i^k denotes the *i-th* individual (x) evolved under the experimental condition k. Positive and negative cross-test values indicate the superiority and inferiority, respectively, of the first experimental condition over the second.

Table 1 presents the results of cross-tests conducted among the four experimental conditions. Notably, agents evolved using the Archive* method significantly outperform those evolved with the Archive method, for both predators and preys. Furthermore, agents evolved using the Generalist method significantly outperform agents from the other three methods, again for both predators and preys.

To further validate the effectiveness of the alternative methods and assess the generality of the solutions, we conducted an additional analysis. Specifically, we evaluated the 40 champion predators (obtained from each method in corresponding 10 replications) against the 40 champion preys. The results, displayed in Figures 4, 5 and Table 2, reveal significant differences. Notably, Generalist predator and prey champions outperform the champions obtained with all other methods (Mann–Whitney U-test with Bonferroni correction, p-value <0.0167). Moreover, Archive* predator and prey champions outperform Archive predators and preys (Mann–Whitney U-test with Bonferroni correction, p-value <0.0167). Instead, the performances of Maxsolve* and Archive predator and prey champions do not significantly differ (Mann–Whitney U-test with Bonferroni correction, p-value >0.0167).

As shown in Figure 4, the predators and preys obtained using the Generalist method (displayed in rows and columns 30–39,

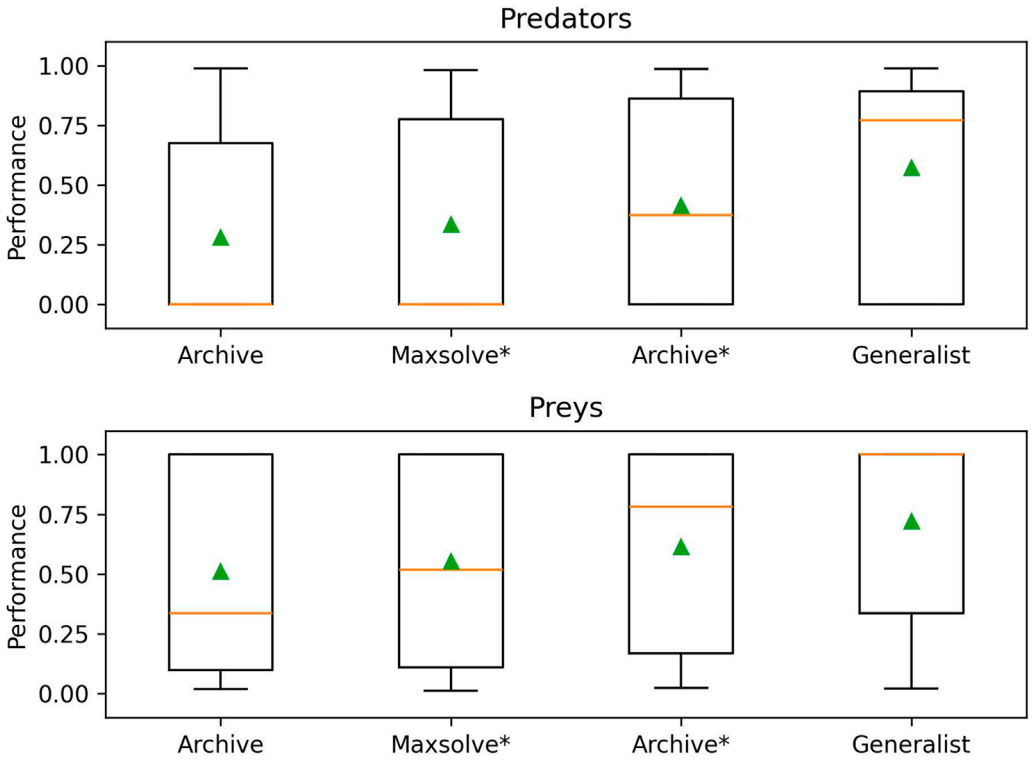

FIGURE 5
The performance achieved by the 40 predator champions (top figure) and the 40 prey champions (bottom figure), evolved using the four different methods across 10 replications of the experiments. The data refer to the post-evaluation phase (see also Figure 4). Boxes denote the interquartile range of the data. The horizontal line inside the boxes represents the median value, while the green triangle in each box indicates the average fitness value (see also Table 2). The whiskers extend to the most extreme data points within 1.5 times the interquartile range from the.

TABLE 2 Average performance achieved by the champion agents evolved with the four different methods.

Predators			
Archive	Maxsolve*	Archive*	Generalist
0.282 [0.382]	0.335 [0.401]	0.415 [0.414]	**0.572 [0.381]**
Preys			
Archive	Maxsolve*	Archive*	Generalist
0.511 [0.415]	0.552 [0.419]	0.613 [0.403]	**0.720 [0.369]**

Data refer to the cross-test (green triangles in Figure 4). Data within squared brackets denote the standard deviations. Bold values indicate the best results. The table is split in two parts: the top section shows the performance obtained by the predator champions, while the bottom one displays the fitness achieved by the prey champions.

respectively) achieve the best performance. Notably, the fifth and sixth predator champions obtained with the Generalist algorithm (Figure 4, rows 34 and 35) achieve a performance of at least 0.5 against 29 out of the 30 prey champions obtained using the other three methods. Additionally, the seventh and eighth prey champions obtained with the generalist algorithm (Figure 4, columns 36 and 37) also achieve a performance of at least 0.5 against 29 out of 30 predator champions obtained using the other three methods.

The analysis of the behavior displayed by the champions reveals their acquisition of sophisticated behavioral skills. Specifically, some of the champions demonstrate the ability to move both toward the front and rear directions, skillfully alternating their direction of motion based on the circumstances (as shown in Video 1, Appendix). They exhibit the capability to capture and evade a wide range of opponents. Moreover, they remain robust against adversarial behaviors exhibited by opponents in most cases; in other words, they are rarely fooled by opponent strategies, despite those strategies have been fine-tuned against them. The ability to alternate their direction of motion depending on the circumstances is more commonly observed in agents evolved using the Generalist algorithm.

An illustrative example of an agent vulnerable to specific opponent behavior is the 8th champion predator obtained using the Generalist method. While this predator is generally effective, it proves fragile when confronted with the adversarial strategy employed by the 7th champion prey. This prey displays an oscillatory behavior that triggers a harmless spinning-in-place response from the predator (see Video 2, Appendix). Another instance of an agent vulnerable to specific opponent behavior is the predator shown in Video 3 (Appendix). The 4th prey champion, obtained through the Archive*algorithm, effectively neutralizes this specific predator by moving counterclockwise around it,

consistently eliciting the same avoidable attacking behavior in the opponent.

4 Conclusion

In this article, we delve into the conditions that drive competitive evolution toward genuine progress, i.e., toward solutions that become better and better against all possible opponents. Specifically, we introduced a set of methods for measuring historical and global progress, we discussed factors that facilitate genuine progress, and we compared the efficacy of four algorithms.

The methods considered were the follows: (1) the Archive algorithm (Rosin and Belew, 1997) that promotes global progress by maintaining an archive of the best individuals from previous generations. This permits to evaluate evolving individuals against opponents of current and previous generations. (2) The Maxsolve* algorithm, i.e., a variation of the original De Jong's algorithm (De Jong, 2005) adapted for both transitive and non-transitive problems. This method also relies on an archive that, however, is used to preserve only diversified individuals. (3) The Archive* algorithm, introduced in this paper, which extends the vanilla Archive method by leveraging multiple evolving populations. This extension allows for the inclusion of more diverse individuals in the archive. (4) The Generalist algorithm (Simione and Nolfi, 2021) that does not use an archive but incorporates a mechanism for identifying and discarding variations leading to local progress only. To analyze the long-term dynamics of the co-evolutionary process, the methods were compared by performing long-lasting experiments.

The results obtained in a predator-prey scenario, commonly used to study competitive co-evolution, demonstrate that all the considered methods lead to global progress in the long term. However, the rate of progress and the ratio of progress versus retrogressions vary significantly.

The Generalist method outperforms the other three methods and is the only one capable of producing solutions that consistently score better and better across generations against previous and future opponents in successive evolutionary phases. The other three methods also exhibit retrogression phases, although less frequently than progression phases. Additionally, the Archive* algorithm, introduced in this paper, outperforms both the vanilla Archive Algorithm and the MaxSolve* algorithm. Overall, our results demonstrate the utilization of proper methods can prevent the convergence of the evolutionary process in limit-cycle dynamics, which jeopardizes the appealing properties of competitive co-evolution.

The superiority of the Generalist algorithm is also demonstrated through visual comparisons of the behavior exhibited by the evolving robots. Indeed, the ability to move bi-directionally and appropriately alternate the direction of motion depending on the circumstances, providing a significant advantage, is more commonly observed among the robots evolved with the Generalist algorithm than among the robots evolved with other algorithms.

Future research should verify whether our results generalize to other competitive settings and whether the continuation of evolutionary progress can lead to an open-ended dynamic in which the efficacy of the evolved solutions keeps increasing in an unbounded manner. The remarkable results recently achieved with Large Language Models also open the possibility to leverage the knowledge acquired by these systems and their online learning capabilities to select useful opponents. Pioneering attempts in this direction are reported in Jorgensen et al. (2024) and Zala et al. (2024).

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

SN: Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Software, Supervision, Validation, Writing–original draft, Writing–review and editing. PP: Data curation, Formal Analysis, Investigation, Methodology, Software, Validation, Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. We acknowledge financial support from PNRR MUR project PE0000013-FAIR.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Bansal, T., Pachocki, J., Sidor, S., Sutskever, I., and Mordatch, I. (2017). Emergent complexity via multi-agent competition. *arXiv Prepr. arXiv:1710.03748*. doi:10.48550/arXiv.1710.03748

Bari, A. G., Gaspar, A., Wiegand, R. P., and Bucci, A. (2018). "Selection methods to relax strict acceptance condition in test-based coevolution," in *2018 IEEE congress on evolutionary computation (CEC)* (IEEE), 1–8.

Bonani, M., Longchamp, V., Magnenat, S., Rétornaz, P., Burnier, D., Roulet, G., et al. (2010). "The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research," in *IEEE/RSJ 2010 international conference on intelligent robots and systems, IROS 2010 - conference proceedings*, 4187–4193.

Buason, G., Bergfeldt, N., and Ziemke, T. (2005). "Brains, bodies, and beyond: competitive co-evolution of robot controllers, morphologies and environments," in *Genetic programming and evolvable machines* (Springer Science + Business Media, Inc. Manufactured in The Netherlands), 25–51.

Buason, G., and Ziemke, T. (2003). Co-evolving task-dependent visual morphologies in predator-prey experiments. *Lect. Notes Comput. Sci.* 2723, 458–469. doi:10.1007/3-540-45105-6_58

Carvalho, J. T., and Nolfi, S. (2023). The role of morphological variation in evolutionary robotics: maximizing performance and robustness. *Evol. Comput.*, 1–18. doi:10.1162/evco_a_00336

Chong, S. Y., Tiňo, P., Ku, D. C., and Yao, X. (2012). Improving generalization performance in co-evolutionary learning. *IEEE Trans. Evol. Comput.* 16, 70–85. doi:10.1109/tevc.2010.2051673

Chong, S. Y., Tiňo, P., and Yao, X. (2009). Relationship between generalization and diversity in coevolutionary learning. *IEEE Transaction Comput. Intell AI Games* 1, 214–232. doi:10.1109/TCIAIG.2009.2034269

Cliff, D., and Miller, G. F. (1995). "Tracking the red queen: measurements of adaptive progress in co-evolutionary simulations," in *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Springer Verlag), 200–218.

Cliff, D., and Miller, G. F. (2006). Visualizing coevolution with CIAO plots. *Artif. Life* 12, 199–202. doi:10.1162/artl.2006.12.2.199

Dawkins, R., and Krebs, J. R. (1979). Arms races between and within species. *Proc. R. Soc. B Biol. Sci.* 205, 489–511. doi:10.1098/rspb.1979.0081

De Jong, E. (2004). The incremental pareto-coevolution archive. *Lect. Notes Comput. Sci.* 3102, 525–536. doi:10.1007/978-3-540-24854-5_55

De Jong, E. (2005). "The MaxSolve algorithm for coevolution," in *Genetic and evolutionary computation (GECCO 2005), lecture notes in computer science*, 483–489.

Ficici, S. G., and Pollack, J. B. (2003). "A game-theoretic memory mechanism for coevolution," in *Genetic and evolutionary computation (GECCO-2003), lectures notes on computer sciences* (Chicago, USA: Springer Verlag), 286–297.

Floreano, D., and Nolfi, S. (1997a). "Adaptive behavior in competing co-evolving species," in *Proceeding of the fourth European conference on artificial life*, 378–387.

Floreano, D., and Nolfi, S. (1997b). "God save the red queen! Competition in co-evolutionary robotics," in *Genetic programming 1997: proceedings of the second annual conference*, 398–406.

Floreano, D., Nolfi, S., and Mondada, F. (1998). "Competitive Co-evolutionary robotics: from theory to practice," in *Proc. Of the fifth international conference on simulation of adaptive behavior (SAB), from animals to animats* (Zürich: ETH).

Georgiev, M., Tanev, I., Shimohara, K., and Ray, T. (2019). Evolution, robustness and generality of a team of simple agents with asymmetric morphology in predator-prey pursuit problem. *Information* 10 (2), 72. doi:10.3390/info10020072

Gers, F. A., and Schmidhuber, J. (2001). LSTM recurrent networks learn simple context free and context sensitive languages. *IEEE Trans. Neural Netw.* 12 (6), 1333–1340. doi:10.1109/72.963769

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9 (8), 1735–1780. doi:10.1162/neco.1997.9.8.1735

Humphries, D. A., and Driver, P. M. (1970). Protean defence by prey animals. *Oecologia* 5, 285–302. doi:10.1007/BF00815496

Ito, T., Pilat, M. L., Suzuki, R., and Arita, T. (2013). ALife approach for body-behavior predator–prey coevolution: body first or behavior first? *Artif. Life Robotics* 18 (1), 36–40. doi:10.1007/s10015-013-0096-y

Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., et al. (2018). Human-level performance in first-person multiplayer games with population-based deep reinforcement learning. *Arxiv. arXiv Prepr. arXiv:1807.01281*. doi:10.48550/arXiv.1807.01281

Jain, A., Subramoney, A., and Miikulainen, R. (2012). "Task decomposition with neuroevolution in extended predator-prey domain," in *Alife 2012: the thirteenth international conference on the synthesis and simulation of living systems* (MIT Press), 341–348.

Jakobi, N., Husbands, P., and Harvey, I. (1995). "Noise and the reality gap: the use of simulation in evolutionary robotics," in *Advances in artificial life: third European conference on artificial life* (Springer Berlin Heidelberg), 704–720.

Jaśkowski, W., Liskowski, P., Szubert, M., and Krawiec, K. (2013). "Improving coevolution by random sampling," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, 1141–1148.

Jorgensen, S., Nadizar, G., Pietropolli, G., Manzoni, L., Medvet, E., O'Reilly, U. M., et al. (2024). "Large Language model-based test case generation for GP agents," in *Proceedings of the genetic and evolutionary computation conference*, 914–923.

Kingma, D. P., and Ba, J. (2014). Adam: a method for stochastic optimization. *arXiv Prepr. arXiv:1412.6980*. doi:10.48550/arXiv.1412.6980

Lan, G., Chen, J., and Eiben, A. E. (2019). "Evolutionary predator-prey robot systems: from simulation to real world," in *Proceedings of the genetic and evolutionary computation conference companion*, 123–124.

Lee, T., Kim, S., and Shim, Y. (2021). Simulation of sustainable co-evolving predator-prey system controlled by neural network. *J. Korea Soc. Comput. Inf.* 26 (9), 27–35. doi:10.9708/jksci.2021.26.09.027

Liskowski, P., and Krawiec, K. (2016). "Online discovery of search objectives for test-based problems," in *Proceedings of the 2016 on genetic and evolutionary computation conference companion*, 163–164.

Lowd, D., and Meek, C. (2005). "Adversarial learning," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (New York, USA: ACM Press), 641–647.

Miconi, T. (2008). Evolution and complexity: the double-edged sword. *Artif. Life* 14, 325–344. doi:10.1162/artl.2008.14.3.14307

Miconi, T. (2009). "Why coevolution doesn't "Work": superiority and progress in coevolution," in *Proceedings of the 12th European conference on genetic programming, lecture notes in computer science* (Berlin: Springer Verlag), 49–60.

Miller, G., and Cliff, D. (1994). "Protean behavior in dynamic games: arguments for the co-evolution of pursuit-evasion tactics," in *From animals to animats III: proceedings of the third international conference on simulation of adaptive behavior*. Editors D. Cliff, P. Husbands, J. R. Meyer, and S. W. Wilson (Cambridge, MA: MIT Press-Bradford Books).

Nolfi, S. (2012). Co-evolving predator and prey robots. *Adapt. Behav.* 20, 10–15. doi:10.1177/1059712311426912

Nolfi, S., and Floreano, D. (1998). Co-evolving predator and prey robots: do 'arms-races' arise in artificial evolution? *Artif. Life* 4, 1–26. doi:10.1162/106454698568620

Pagliuca, P., Milano, N., and Nolfi, S. (2020). Efficacy of modern neuro-evolutionary strategies for continuous control optimization. *Front. Robotics AI* 7, 98. doi:10.3389/frobt.2020.00098

Pagliuca, P., and Nolfi, S. (2019). Robust optimization through neuroevolution. *PloS one* 14 (3), e0213193. doi:10.1371/journal.pone.0213193

Palmer, M. E., and Chou, A. K. (2012). "An artificial visual cortex drives behavioral evolution in co-evolved predator and prey robots," in *Proceedings of the 14th annual conference companion on Genetic and evolutionary computation*, 361–364.

Rosin, C. D., and Belew, R. K. (1995). "Methods for competitive co-evolution: finding opponents worth beating," in *Proceedings of the 6th international conference on genetic algorithms* (Pittsburgh, PA, USA: Morgan Kaufmann), 1995, 373–381.

Rosin, C. D., and Belew, R. K. (1997). New methods for competitive coevolution. *Evol. Comput.* 5, 1–29. doi:10.1162/evco.1997.5.1.1

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *Adv. neural Inf. Process. Syst.* 29. doi:10.5555/3157096.3157346

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. (2017). Evolution strategies as a scalable alternative to reinforcement learning. *arXiv:1703*, 03864v0382. doi:10.48550/arXiv.1703.03864

Samothrakis, S., Lucas, S., Runarsson, T. P., and Robles, D. (2013). Coevolving game-playing agents: measuring performance and intransitivities. *IEEE Trans. Evol. Comput.* 17, 213–226. doi:10.1109/TEVC.2012.2208755

Simione, L., and Nolfi, S. (2021). Long-term progress and behavior complexification in competitive coevolution. *Artif. Life* 26, 409–430. doi:10.1162/artl_a_00329

Sinervo, B., and Lively, C. M. (1996). The rock-paper-scissors game and the evolution of alternative male strategies. *Nature* 380, 240–243. doi:10.1038/380240a0

Stanley, K. O., and Miikkulainen, R. (2002). The dominance tournament method of monitoring progress in coevolution. *GECCO 2002 Proc. Bird. a Feather Work Genet. Evol. Comput. Conf.*, 242–248.

Stolfi, D. H., Brust, M. R., Danoy, G., and Bouvry, P. (2021). UAV-UGV-UMV multi-swarms for cooperative surveillance. *Front. Robotics AI* 8, 616950. doi:10.3389/frobt.2021.616950

Wang, X., Chen, Y., and Zhu, W. (2021). A survey on curriculum learning. *IEEE Trans. Pattern Analysis Mach. Intell.* 44, 4555–4576. doi:10.1109/tpami.2021.3069908

Wiegand, R. P., Liles, W. C., and De Jong, K. A. (2002). "Analyzing cooperative coevolution with evolutionary game theory," in *Proceedings of the 2002 congress on evolutionary computation, CEC 2002*. (IEEE Press), 1600–1605.

Zala, A., Cho, J., Lin, H., Yoon, J., and Bansal, M. (2024). EnvGen: generating and adapting environments via LLMs for training embodied agents. *arXiv Prepr. arXiv:2403.12014*. doi:10.48550/arXiv.2403.12014

Appendix

Video 1. Available from https://youtube.com/shorts/prpFwtN-v3Y?feature=share

Video 2. Available from https://youtube.com/shorts/77mYMT6CKnI?feature=share

Video 3. Available from https://youtube.com/shorts/hvrs2rcJVdU?feature=share