An Investigation of Environmental Influence on the Benefits of Adaptation Mechanisms in Evolutionary Swarm Robotics

Andreas Steyven
Edinburgh Napier University
10 Colinton Road
Edinburgh, Scotland, UK
a.steyven@napier.ac.uk

Emma Hart
Edinburgh Napier University
10 Colinton Road
Edinburgh, Scotland, UK
e.hart@napier.ac.uk

Ben Paechter
Edinburgh Napier University
10 Colinton Road
Edinburgh, Scotland, UK
b.paechter@napier.ac.uk

ABSTRACT

A robotic swarm that is required to operate for long periods in a potentially unknown environment can use both evolution and individual learning methods in order to adapt. However, the role played by the environment in influencing the effectiveness of each type of learning is not well understood. In this paper, we address this question by analysing the performance of a swarm in a range of simulated, dynamic environments where a distributed evolutionary algorithm for evolving a controller is augmented with a number of different individual learning mechanisms. The learning mechanisms themselves are defined by parameters which can be either fixed or inherited. We conduct experiments in a range of dynamic environments whose characteristics are varied so as to present different opportunities for learning. Results enable us to map environmental characteristics to the most effective learning algorithm.

CCS CONCEPTS

•**Computing methodologies** → **Mobile agents;**

KEYWORDS

Evolutionary Swarm Robotics, Environment, Learning

ACM Reference format:
Andreas Steyven, Emma Hart, and Ben Paechter. 2017. An Investigation of Environmental Influence on the Benefits of Adaptation Mechanisms in Evolutionary Swarm Robotics . In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071232

1 INTRODUCTION

Recent advances in technology are driving novel research in swarm robotics, envisioning future applications in which swarms might be sent to remote or hazardous environments and in which they will need to survive over long periods of time. As these environments will be unknown to the designer *a priori* and are potentially dynamic, the swarm must be able to continuously adapt its behaviour to ensure it both maintains sufficient energy to survive, and to successfully perform tasks.

The importance of being able to adapt over time has been a subject of research within Evolutionary Robotics for some time [20]. Adaptation often takes one or all of three forms: evolutionary, individual and social learning. In *evolutionary* adaptation, information encoded on the genome adapts through selection and reproductive operators over many generations. In *individual* learning, a robot can adapt its own behaviour during the course of its lifetime, for example, updating weight values in a neural network controller. Finally in *social* learning, robots can exchange information during a lifetime.

The relative benefits of mixing the different types of adaptation have been studied both in simulation [3, 6] and hardware [4, 10–12]. Typically, experiments are conducted in single environment related to a specific task, therefore the role of the environment in influencing the result is not made explicit. An exception is recent work from Haasdjik [5] who explicitly studied the effect of combining conflicting environmental and task requirements in a simulated system. This showed that high selective pressure exerted by a task can outweigh any selective pressure from the environment. However, an arbitrary environment was defined to conduct experiments in, leaving open the question of whether the same effects would be observed in a different environment.

The goal of this paper is to investigate the interplay between evolution, individual learning and environment characteristics. We consider a swarm which undergoes distributed evolution of a neural-network based controller and is augmented with an individual learning mechanism: this modifies the information gleaned from the environment and fed to the controller over the lifetime of a robot. Specifically, we consider a swarm operating in an environment which is unknown *a priori* and which robots must learn relative values of positive and negative energy tokens. Each environment contains n positive and n negative energy tokens. Positive tokens increase the robot's energy by v units of energy, while negative ones reduce it by a fixed amount. As n, v vary, each environment presents different opportunities for learning in that there are a small number of high value tokens, or a large number of low value tokens. In addition, tokens change their nature across 'seasons', i.e. tokens of a specific colour switch value from negative to positive on a cyclical basis. This forces the swarm to have to re-learn the effect of any given colour of token every season. Various settings for individual learning

GECCO '17, Berlin, Germany

are investigated in which the learning mechanism is either fixed or has components that can be simultaneously evolved. The following questions are investigated:

- How do the parameters of the environment (token count, token value) influence the effectiveness of different individual learning settings?
- How does the rate of change of a given environment influence the effectiveness of individual learning mechanisms?
- How does the nature of the individual learning mechanism influence performance in different environments?

We augment a distributed evolutionary algorithm previously described in [9] with mechanisms for individual learning in order to conduct experiments. Note that the goal is not to propose a novel method of either individual learning or evolutionary adaptation but to explore the relationship between the environment and value of different types of adaptation.

2 RELATED WORK

A reasonable body of research exists in relation to combining learning and evolution, and factors that influence this relationship [7, 13, 14]. The relationship of the two methods in a swarm environment in which it is necessary to simultaneously learn behaviours which enable reproduction in addition to task performance is less well studied however. Haasdjik *et al* propose a framework for evolution, individual and social learning in collective systems, and consider the interaction of evolution and individual learning in which the latter is achieved by *reinforcement learning* [19]. Their experiments show that in a collective system, it is possible for learning to counteract evolution. A *hiding-effect* can occur in which individual learning acts to mask the ill-adapted nature of non-optimal agents and is therefore counter-productive. Although a number of environments were investigated which essentially modified the reward system, all environments were static, and the relationship of the learning framework to specific parameterisations of the environmental features was not examined.

A dynamically changing reward system was investigated in [1] who proposed mEDEA, a completely distributed evolutionary algorithm for open-ended evolution. Here, efficient adaptation in a changing environment was demonstrated using a set up that switched phases: in the *free-ride* phase, there is no cost to movement therefore a robot only needs to meet a single other robot to pass on its genome, while in the alternating phase the robot is required to harvest energy in order to move and therefore creating opportunties for passing on its genome. Haasdijk *et al* [8] extended mEDEA to add explicit task-selection in the MONEE framework [15]. In [5] they examine in more detail the relative selection pressures induced by task performance and survival in different environments, finding that task performance is optimised even if it reduces the lifetime of robots (and therefore their ability to reproduce). Heinermann *et al* investigate the relationship between evolution, individual and social learning in real swarm [10–12]. Here, the evolutionary part focuses on evolving a suitable sensory layout, while the individual learning runs an evolution strategy to learn the network weights during the robot lifetime. Learnt weight vectors are broadcast to other robots during the social learning phase. The main focus of this work was to investigate the impact of social learning. Individual learning is *required* to learn a controller and hence cannot be omitted.

In contrast to the above, we consider scenarios in which individual learning has the potential to improve evolved behaviours, but is not essential. We investigate the relative benefits of evolution and individual learning using a variety of learning mechanisms and in a range of environments with different features. The goal is to specifically relate the roles of evolution and individual learning performance to features of the environment.

3 OVERVIEW

A swarm operates in an open environment in which there are two types of coloured tokens: driving over one colour increases the robots energy while the other decreases it. Robots should learn to avoid the negative token. However, a "seasonal" change is imposed where the value of the token is reversed, i.e. red becomes positive and blue negative or vice versa. A robot must thus adapt any previously evolved behaviour. All robots in the swarm evolve a neural network that controls their behaviour through a distributed evolutionary algorithm [9] In addition, they can exploit an individual learning mechanism which can potentially learn the *current* value of a given colour of token. This information modifies an input to the evolved neural network. We investigate a number of types of individual learning in which some components of the learning mechanism can be either heritable, fixed or absent.

Experiments are conducted using the Roborobo simulator [2]. The robots have 8 ray-sensors distributed around the body and detect proximity to the nearest object and its type. Each robot is controlled by an evolved Elman recurrent neural network (RNN). The network has 16 sensory inputs and 2 motor outputs (translational and rotational speeds). The 16 inputs comprise of two information of each of the 8 ray-sensors, proximity and whether or not this object is an energy token. Although the colour/type of the object is also detected by the robot, it is not fed into the RNN as an input, but only used in the adaptation mechanism[1].

3.1 mEDEA

Using the inputs and outputs just described, an RNN with 1 hidden layer containing 16 nodes is evolved by a distributed evolutionary algorithm [9]. This algorithm is an extension of *mEDEA* [1], and incorporates a selection mechanism based on relative fitness. In brief, for a fixed period, robots move according to their control algorithm, broadcasting their genome that is received and stored by any robot within range. At the end of this period, a robot uses fitness-proportionate

[1] the information cannot be encoded directly to the network without *a priori* knowledge of the number of potential colours

selection to choose a genome from its list of collected genomes according to a relative fitness value, and applies a variation operator. This takes the form of a Gaussian random mutation operator, inspired from Evolution Strategies. Pseudo-code is given in Algorithm 1.

```
load($currentGenome = randomInitialisedGenome$);
while $iteration \leq maxIterations$ do
    if $hasGenome()$ then
        if $lifetime \leq maxLifetime$ & $energy > 0$ then
            move();
            if $neighbourhood.isNotEmpty()$ then
                $rf$ =calculateRelativeFitness();  // eq.1
                broadcast($currentGenome$,$rf$);
            end
        else
            remove($currentGenome$);
        end
    end
    $genomeList$.addIfUnique($receivedGenomes$);
    if $genomeList.size() > 0$ then
        $genome$ = select$_{roulette-wheel}$($genomeList$);
        load($currentGenome$ =
          applyVariation($genome$));
        $genomeList$.empty();
        $lifetime = 0$;
    end
end
```

Algorithm 1: Pseudo code of the adapted version of the mEDEA algorithm with relative fitness mEDEA$_{rf}$ as introduced in [9] used with roulette-wheel as explicit selection mechanism

Each robot estimates its fitness in terms of its ability to survive based on the balance between energy lost and energy gained, denoted (δ_E): this term is initialised to 0 at $t = 0$ (when the current genome was activated) and is decreased according an energy-model described below that accounts for both movement and the cost of communicating for evolution, and increased by E_{token} if it crosses an energy token. Given δ_E, a robot calculates a fitness value which is relative to those robots in the neighbourhood of range r. according to equation 1, where f_i' is the relative fitness of robot i at time t, $mean_{sub_i}$ is the mean δ_E of the robots within the subpopulation defined by all robots in range r of robot i, and sd_{sub_i} is the standard deviation of the δ_E of the subpopulation.

$$f_i'(t) = \frac{\delta_i(t) - mean_{sub_i}(t)}{sd_{sub_i}(t)} \tag{1}$$

There is a fixed cost to living of 0.5 units per timestep, regardless of whether the robot moves or not. A robot moving consumes an amount of energy that is related to its rotational speed v_{rot}, translational speed v_{trans}, and their respective maximum values $v_{\text{rot-max}}$ and $v_{\text{trans-max}}$

$$E_{\text{step}} = 0.5 + \left(\frac{v_{\text{rot}}}{v_{\text{rot-max}}} + \frac{v_{\text{trans}}}{v_{\text{trans-max}}}\right)/4 \tag{2}$$

The amount of energy spent on communication E_{com} is calculated using equation 3, where i and j are the number of genomes received and transmitted respectively. The values $a_{\text{rx}} = 0.0305$, $a_{\text{tx}} = 0.01379$ and $a_{\text{tx-amp}} = 0.000614$ were determined based on the method described by [18]; the reader is referred to this publication for a description of their approach.

$$E_{\text{com}} = \sum_{k=0}^{i} a_{\text{rx}} + \sum_{k=0}^{j} \left(a_{\text{tx}} + b_{\text{tx-amp}} \times d^2\right) \tag{3}$$

Equation 4 shows the change in energy at each simulation step, where n is the number of tokens that have been collected in that step.

$$E(t+1) = E(t) - E_{\text{step}} - E_{\text{com}} + (n_{\text{token}} \times E_{\text{token}}) \tag{4}$$

3.2 Environment

In Evolutionary Robotics, it is often unclear exactly how parameterisation of a given environment might influence the emergence of particular behaviours. Often, the focus of reported studies is on algorithm performance, without serious consideration of how the choice of environment may influence results. This is particularly important for an open-ended distributed algorithm such as *mEDEA* in which survival of robots is crucial for evolution to occur. To counter this, Steyven *et al* [17] recently proposed a technique by which preliminary experimentation could be used to generate a surface-plot, highlighting regions of the parameter space in which the environment provides the right balance between facilitating survival and exerting sufficient pressure for new behaviours to emerge. This enables a researcher to select appropriate settings for experimentation. For example, for a given task, on the one hand, there will be regions in which the characteristics of the environment are such that robots find survival to be trivial (e.g. food supplies are unlimited and easy to find), and hence there is little pressure to evolve specialised behaviours. On the other hand, environmental characteristics which are harsh enough to cause individual robots to die prematurely and therefore prevent any effective evolution are also identified.

Using the algorithm described above, we conducted experiments in an environment parameterised by two variables: the *number* of energy tokens available, and the *value* of the energy token. In each environment tested, there are n positive tokens with value v, and n negative tokens with value -400. The delta-energy δ_E, i.e. difference between start and end energy is recorded for multiple points in the parameter space, resulting in the plot shown in figure 1. From this plot, we identify three points to conduct experiments along the *energy neutral line*, i.e the region in which the robot expends as much energy as it acquires. This represents a region in which selection-pressure from the environment to survive is neither too small or too large to mask the behaviours we are

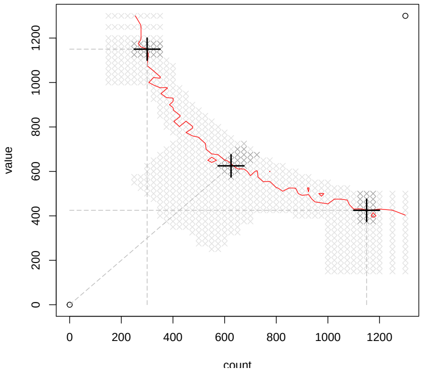

Figure 1: Overview of newly created surface landscape. The red line shows the Neutral Line, the line where the surface plot crosses a plane drawn at delta-energy (δ_E)=0.

*

Table 1: Environmental configurations: description refers to the prevalence of energy tokens within the environment.

Number of tokens	*Value per token*	*Description*
300	1150	Scarce
625	625	Balanced
1150	425	Abundant

interested in investigating. The points identified are specified in table 1.

4 INDIVIDUAL LEARNING

The neural network described above has a set of binary inputs (one for each sensor ray of the robot) that denote the presence (1) or absence (0) of a token (independent of its type). Therefore, in an environment in which there are multiple types of tokens, the only way for an individual to distinguish between them is to pick up the token and observe the change in energy. If the environment in which the robot operates is known *a priori*, then clearly, the neural network could be designed in order to include relevant information about each token type. However, if the environment is unknown, then the robot must learn to adapt to the different types and values of tokens it may encounter.

We use an adaptation mechanism which enables a robot to modify the value input to the RNN corresponding to a token sensor: instead of simply having a binary input, the robot uses a learned/evolved multiplier to adapt the token input to a continuous value between -1 and 1.

Each time a previously unseen type of token is encountered (detected by a sensor ray or through consumption[2]), a new multiplier is added to the multiplier set. As tokens are usually detected before they are consumed, no information regarding a new token's value is known: the robot therefore randomly initialises a value to associate with the type (x) of detected token. Following consumption, the resulting change of energy is detected by the robot and its learning mechanism can modify the corresponding multiplier value (m_x).

All multiplier values are adjusted every time a token is consumed according to equation 5:

$$m'_x = m_x + LS \times \left(LR - \frac{C_x}{C_{\text{total}}} \right) \times \left(\frac{V_x}{V_{\max} - V_{\min}} \right) \qquad (5)$$

where m_x is the current value for the multiplier for type x; $C_x$is the number of tokens of type x collected; C_{total} is the total number of all tokens collected; V_x is the value of the token that has just been consumed and is therefore now known to the robot (being equivalent to the change in energy); $V_{\max}, V_{\min}$ define the minimum and maximum values of all tokens encountered so far. LR is a learning rate that controls the magnitude of the change, and LS is either -1 or $+1$ and simply inverts the direction of change; this is required to adjust the learning mechanism to the internal value notation of the neural network and can be adapted via evolution. The learning mechanism is shown in Algorithm 2.

```
if $token_x$ is unknown then
    $multipliers$.add($token_x$);
end
if $token_x$ is consumed then
    $tokenCounter_x$.update($token_x$);
    $totalTokenCount$.update();
    $tokenValue_x$.update($\delta_E(t) - \delta_E(t-1)$);
    $totalValueRange$.update();
    for $m_x$ in multipliers do
        $m_x$.update();                          // eq. 5
    end
end
```

Algorithm 2: Pseudo code of the steps carried out to update all multipliers every time a token is encountered.

Three factors influence the learning mechanism: the initial value assigned to a token V_x, the learning rate LR and the associated sign LS. These factors can be randomly assigned, fixed to some specific value, or can themselves be subject to evolution. Allowing the learning sign to co-evolve enables the learning mechanism to self-adapt to the internal value convention of the neural network. Finally, enabling the robot to evolve an appropriate starting value for each type of token based on its experience may speed up learning in some circumstances. Even though token values change over seasons, inheriting a good starting value may be beneficial, likely dependent on the rate of change of the environment.

Table 2 defines four variants of the learning algorithm that we investigate in conjunction with the three environments described in section 3.2. Note that in no case is any Lamarkian evolution used, i.e. although the multiplier starting values

[2] The sensor rays of the robot are not evenly distributed around the robot body. This can lead to the situation in which a robot drives over the token before any of the sensor rays detected it.

Table 2: Learning scenarios investigated showing heritability of information

	Initial Value of Multiplier	LR	LS
Baseline	1 (all tokens)	none	n/a
IL	random	fixed	evolved
EVO	evolved	none	n/a
EVO+IL	evolved	evolved	evolved

Table 3: Simulation and Experimental Parameters for all experiments

Simulation parameters	
Arena size	1024 px × 1024 px
Max. robot lifetime	2500 iterations
Token re-spawn time	500 iterations
Sensor range	196 pixel
Max. communication range $r_{\max}$	128 pixel
Experimental parameters	
Number of independent runs	30
Number of robots	100
Max. iterations	100,000
Start energy	500

are adapted over the course of a lifetime, they are *never* written back to the genome and are therefore not inherited.

4.1 Experiments

An experiment is defined by a tuple <*environment, seasonal change rate, algorithm*>. Three environments (see section 3.2) and three different rates of seasonal change are investigated: 0 (no change, i.e. static environment), every 5000 iterations, and every 15000 iterations. Note that the maximum lifetime of a genome before it is replaced is 2500 iterations, so every robot should go through at least one evolutionary generation during the shorter (5000 iterations) season and at least 5 times in the 15000 season. In practice, as robots tend to die before their maximum lifetime, more evolutionary cycles are likely to occur.

Four algorithms are investigated as detailed in table 2. Note that in the baseline experiments, all tokens have a fixed multiplier of 1 and therefore the robots cannot distinguish between tokens of different types. Thus, in total 36 (=3x3x4) experiments are conducted. In each experiment, we record the *totalTokenRatio* at the end of the season. This value is the ratio of the number of collected token with positive value divided by the sum of all collected token within that season. A ratio of 0.5 shows that an equal amount of positive and negative token was collected, below 0.5 more negative and above more positive token, respectively.

Experimental and simulation parameters are given in table 3. Parameters associated with the learning mechanism are given in table 4. The values for LR_{initial} and $LR_{\max}$ where selected following limited empirical exploration.

Table 4: Learning parameter with their initial values and ranges in which they can change during runtime of the experiment.

Parameter	**Init. Value**	**Value Range**
Learning rate, LR	1.02	$[LR_{\min}, LR_{\max}]$
Min. LR, $LR_{\min}$	1	fixed
Max. LR, $LR_{\max}$	1.5	fixed
Multiplier of type x, m_x	random	$[-1, 1]$
Learning sign, LS	random	$[-1, 1]$

The positive value of an energy token is determined by the environment. In seasons when a token is negative, the value is fixed -400 which is 80% of a robot's initial energy.

Following 30 runs of each experiment, statistical analysis was conducted based on the method in [16] using a significance level of 5%. The distributions of two results were checked using a Shapiro-Wilk test. If both followed a Gaussian distribution then Levene's test for homogeneity of variances was perfomed. For equal variances the p-value was determined using an ANOVA test, otherwise using a Welch test. A Kruskal-Wallis rank sum test was perfomed to determine the p-value if one of the results followed a non-Gaussian distribution.

5 RESULTS

This section provides summarised results: detailed experimental data is available as supplementary material. Table 5 shows the median totalTokenRatio for each of three individual learning mechanisms (EVO, EVO+IL, IL) in each of the 3 environments and for each value of seasonal change. The values are compared to the result from the baseline experiment each case, and statistical significance is indicated in the table.

The EVO method (which evolves multiplier values but has no adaption during a lifetime) outperforms the baseline method in all three static environments (season change = 0). Here, evolution is able to determine appropriate values for each multiplier type. However, in the dynamic environment, evolving the multiplier value is detrimental. In the first season, evolution can find appropriate multiplier values (particularly in a long season). However, as soon as the season changes, these become irrelevant; if these values have spread sufficiently through the population it may take considerable time for evolution to reverse this change, while in the meantime, the robot will continue to collect negative tokens.

The IL method (fixed learning rate and random initialisation of values) never outperforms the baseline method in the static environment, and is worse than the baseline in the dynamic environments. The magnitude of the effect is highest in the seasonal change=5000 environment for a balanced environment. It appears that the learning rate is not sufficient to adapt a randomly initialised multiplier to a suitable value while the randomness can actually bias the

Table 5: Showing median of end values by seasonal change and Experiment for *totalTokenRatio* over generation 199 to 200 (N:30). ↓, ↔, ↑ indicate whether the value is lower, not different or higher respectively compared to the baseline experiment. The number of arrows corresponds to the magnitude level of the effect size based on a Vargha and Delaney A test. (1 = small, 2 = medium, 3 = large)

Experiment		Evo			IL			Evo + IL		
	count	300	625	1150	300	625	1150	300	625	1150
Season	value	1150	625	425	1150	625	425	1150	625	425
0		↑↑↑ 0.5301	↑↑↑ 0.5411	↑↑↑ 0.5662	↔ 0.5034	↔ 0.5056	↓ 0.4997	↑↑↑ 0.5306	↑↑↑ 0.5388	↑↑↑ 0.5705
5k		↔ 0.4995	↓ 0.4982	↓ 0.4989	↓ 0.5006	↓↓ 0.4975	↔ 0.5023	↔ 0.5029	↑↑ 0.5134	↑↑↑ 0.5191
15k		↓↓ 0.496	↓↓ 0.495	↓ 0.4981	↓ 0.4973	↓ 0.4993	↓ 0.5011	↔ 0.4981	↑↑ 0.5136	↑↑↑ 0.5121

robot towards collecting a particular type. On average, this is worse than the baseline case in which the robot has equal preference for both types.

In contrast, with the exception of the two *dynamic and scarce* environments, the EVO+IL method which evolves the LR, LS and the multiplier values and also adjusts the latter during lifetime, a significant improvement is observed with respect to the baseline method. In the *scarce* environments, the robots have little information available to them to inform learning, as there are few tokens. When the environment is changing rapidly this is particularly detrimental. In the other environments, there are more tokens to learn from. When this is coupled with the ability to both evolve useful multiplier values *and* adapt them at a appropriate rate, the swarm learns to adapt to the changing environments and improves its behaviour in the static environment.

5.1 Influence of environmental parameters

Next, we examine the first question posed in section 1 in more depth: *under what environmental conditions is augmenting evolution with an individual learning mechanism beneficial?*

Table 6 provides a pairwise comparison of environments for *totalTokenRatio* obtained at the end of each experiment. In this table and subsequent ones, the symbols =, <,> indicate whether the median values for totalTokenRatio are not significantly different, significantly smaller or larger respectively. p-values below the significance level of 0.05 are written in bold.

Table 6 clearly indicates that for the methods that include an evolutionary component with the learning algorithm, then in the static environment, *abundant* > *balanced* > *scarce*. In contrast, when only a fixed individual learning mechanism is used with no adaptation of learning rate, then the reverse appears true; the token ratio is higher in the *balanced* and *scare* environments is higher than in the *abundant* environment, with no significant difference between *balanced* and *abundant*.

In the slow changing environment (15k), the general trend is that *abundant* > *balanced* > *scarce* for *all* three mechanisms. In the rapidly changing environment, a mixed picture emerges. For the EVO+IL mechanism, it is clear that *abundant* > *balanced* > *scarce*. For EVO, the *scarce* environment does *not* provide significantly different results to the other two, whereas for IL, both *scarce* and *balanced* prove harder than *abundant*, but *scarce* outperforms *balanced.*

5.2 Influence of Environmental Change

Table 7 illustrates how the rate of change of a given environment influences the interaction between environmental parameters and learning mechanisms. In 21/27 pairwise comparisons, statistically significant results are observed.

In the *scarce* environments, there is a general pattern that in terms of rate of change, *static* > $5k$ > $15k$ for *all* mechanisms. In the *balanced* environments, the same general pattern is observed, with the exception that for the IL and EVO+IL mechanisms, no statistical differences are noted between the 5k and 15k environments. In the *abundant* environments, we also note the same general pattern as above, except that for IL, the only significant result shows that 5k>15k significant, while in contrast, for EVO, 5k<15k.

5.3 Influence of learning mechanism

Table 8 provides a pairwise comparison of learning mechanisms within different environments. 22/27 comparisons are significant.

For the *scarce* environment, general pattern that EVO+IL outperforms the other two methods in 4/6 cases, with no statistical difference in the other two cases. In the balanced environment, EVO+IL also clearly dominates both EVO and IL. EVO dominates IL in the static and 5k experiments. Finally, in the *abundant* environment, we again observe the supremacy of EVO+IL, while IL dominates EVO in both of the dynamic environments.

5.4 Analysis

The previous section showed that the EVO+IL clearly outperforms IL and EVO in all parameterisations of the environment and for all rates of change. We examine its behaviour more closely by plotting the normalised difference between the number of positive tokens (p) and the number of negative tokens (n) collected per season over time (i.e. p-n). This is shown in figure 2 for the (scarce, balanced, abundant) environments for the two cases in which the values of the tokens change dynamically with seasons. The solid lines on the graph represent this value combined over both seasons, while the dashed and dotted lines represent the value in season 0 and season 1 respectively. All lines are smoothed over the

Table 6: p-values of pairwise comparison of environments for *totalTokenRatio* (row vs. column) over generation 199 to 200

	Experiment		**Evo**		**IL**		**Evo + IL**	
	count		**625**	**1150**	**625**	**1150**	**625**	**1150**
Season		**value**	**625**	**425**	**625**	**425**	**625**	**425**
0	**300**	**1150**	**< 1.24e-07**	**< 3.02e-27**	= 5.78e-01	**> 7.33e-05**	**< 3.27e-02**	**< 1.44e-40**
	625	**625**		**< 9.37e-16**		**> 7.87e-11**		**< 1.33e-32**
5k	**300**	**1150**	= 4.55e-01	= 3.08e-01	**> 7.89e-05**	**< 1.99e-02**	**< 3.87e-19**	**< 1.5e-32**
	625	**625**		**> 1.22e-03**		**< 1.81e-17**		**< 5.88e-04**
15k	**300**	**1150**	**< 4.78e-02**	**< 1.58e-04**	= 6.51e-01	**< 4.11e-04**	**< 1.94e-16**	**< 9.56e-30**
	625	**625**		**< 1.09e-08**		**< 1.44e-12**		= 2.61e-01

Table 7: Showing p-values of pairwise comparison of seasonal change for *totalTokenRatio* (row vs. column) over generation 199 to 200

Environment		**count:300 value:1150**		**count:625 value:625**		**count:1150 value:425**	
Experiment	**Season**	**5k**	**15k**	**5k**	**15k**	**5k**	**15k**
Evo	**0**	**> 9.09e-69**	**> 6.76e-55**	**> 1.2e-131**	**> 4.34e-99**	**> 6.91e-120**	**> 9.94e-88**
	5k		**> 1.89e-02**		**> 1.67e-05**		**< 6.04e-03**
IL	**0**	**> 2.54e-05**	**> 6.93e-09**	**> 4.05e-27**	**> 4.43e-20**	= 1.03e-01	= 7.08e-01
	5k		= 7.61e-02		= 8.51e-02		**> 8.33e-03**
Evo + IL	**0**	**> 2.82e-35**	**> 1.37e-31**	**> 1.76e-32**	**> 4.15e-32**	**> 3.41e-178**	**> 3.54e-118**
	5k		**> 1.7e-02**		= 4.38e-01		= 7.19e-01

Table 8: Showing p-values of pairwise comparison of learning mechanism for *totalTokenRatio* (row vs. column) over generation 199 to 200

Environment		**count:300 value:1150**		**count:625 value:625**		**count:1150 value:425**	
Season	**Experiment**	**IL**	**Evo + IL**	**IL**	**Evo + IL**	**IL**	**Evo + IL**
0	**Evo**	**> 6.04e-35**	= 6.64e-01	**> 8.51e-77**	= 4.32e-01	**> 4.44e-82**	**< 1.5e-03**
	IL		**< 3.6e-26**		**< 6.66e-59**		**< 6.7e-150**
5k	**Evo**	= 8.45e-01	**< 3.18e-05**	**> 4.42e-06**	**< 3.36e-55**	**< 4.22e-15**	**< 9.94e-122**
	IL		**< 1.27e-04**		**< 3.6e-70**		**< 2.9e-80**
15k	**Evo**	= 7.84e-02	**< 8.55e-04**	**< 7.4e-03**	**< 1.09e-41**	**< 1.28e-08**	**< 3.11e-77**
	IL		= 5.27e-02		**< 5.83e-42**		**< 2.23e-72**

relevant points. The continuous improvement in this metric is clearly identified for EVO+IL, showing a generally robust response to the changes in token value (i.e. an upward trend). The *abundant* environment proves most straightforward to learn in: having a large *quantity* of information of low-value outweighs the situation in which a small quantity of high-value information is available. In contrast, in the baseline experiment in which *no* information is available as to token value, the (p-n) metric continuously cycles. In this case, the best that evolution can do is learn a token-avoidance behaviour, as there is no means of distinguishing between tokens.

6 CONCLUSION

We have investigated the performance of a number of adaptation mechanisms that augment evolution of a neural network controller. Adaptation mechanisms that included heritable and fixed components were analysed in three different environments in which both the number of learning opportunities and the impact of the learning opportunity varied.

We show that an adaptation mechanism in which all components evolve and are heritable (EVO+IL) copes well in static and dynamic environments, and is able to learn to distinguish between tokens of different value. In dynamic environments, the greatest effect is observed when the environment contains a large number of small learning opportunities. The fewer the learning opportunities, the less effective the mechanism becomes, despite the fact that the opportunities provide more energy and therefore more information to the learning mechanism.

In contrast, the EVO and IL mechanisms both prove to be detrimental in a changing environment when compared to the baseline scenario. No clear pattern emerges however in terms of the magnitude of the effect with respect to the number of learning opportunities present. The IL method

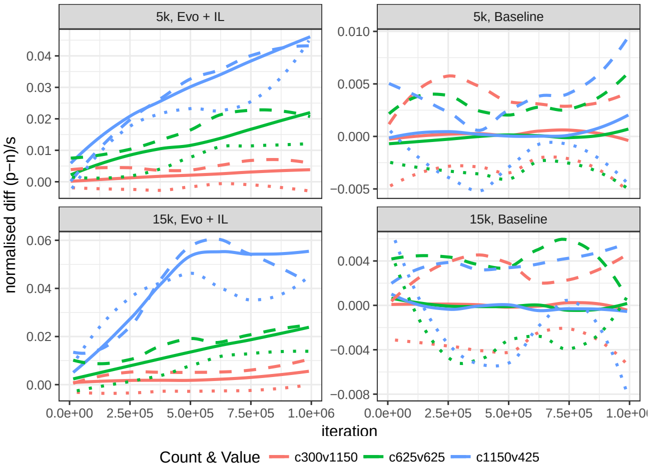

Figure 2: Normalised difference between positive and negative tokens collected. Solid line is value combined over all seasons, dashed = season 0, dotted = season 1

never outperforms the baseline experiments, whereas EVO is beneficial only in a static environment. In the latter case, performance is greatest in the environment with most tokens, and decreases as the number of tokens decreases.

The results clearly demonstrate the interaction between the learning mechanism and environmental parameters. This is of particular relevance for distributed algorithms such as mEDEA in which environmental pressure influences reproductive abilities. The huge variety of behaviour that were displayed in different environments highlight how fundamental it is to not just select parameters at random, but to perform a more thorough analysis. The emerging behaviour using a single set of algorithmic parameter varied from giving a massice advantage, to showing no difference, to even being counter productive. Future work will extend the analysis to other mechanisms for adding individual learning and/or adaptation, as well as considering social learning, recently demonstrated by [11, 12] to be effective in some scenarios.

REFERENCES

[1] Nicolas Bredeche and Jean-Marc Montanier. 2010. Environment-driven embodied evolution in a population of autonomous agents. In *Parallel Problem Solving from Nature, PPSN XI*, Robert Schaefer, Carlos Cotta, Joanna Kołodziej, and Günter Rudolph (Eds.), Vol. 6239. Springer Berlin Heidelberg, Krakov, Poland, 290–299.

[2] Nicolas Bredeche, Jean-Marc Montanier, Berend Weel, and Evert Haasdijk. 2013. Roborobo! a Fast Robot Simulator for Swarm and Collective Robotics. *CoRR* abs/1304.2 (apr 2013). arXiv:1304.2888

[3] Kai Ellefsen. 2013. Balancing the Costs and Benefits of Learning Ability. In *Advances in Artificial Life, ECAL 2013*, Pietro Liò, Orazio Miglino, Giuseppe Nicosia, Stefano Nolfi, and Mario Pavone (Eds.). MIT Press, Taomina, 292–299.

[4] Jorge Gomes, Miguel Duarte, Pedro Mariano, and Anders Lyhne Christensen. 2016. *Cooperative Coevolution of Control for a Real Multirobot System.* Springer International Publishing, Cham, 591–601.

[5] Evert Haasdijk. 2015. Combining Conflicting Environmental and Task Requirements in Evolutionary Robotics. In *2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems*. IEEE, 131–137.

[6] Evert Haasdijk, Agoston Endre Eiben, and Alan Frank Thomas Winfield. 2013. Individual, Social and Evolutionary Adaptation in Collective Systems. In *Handbook of Collective Robotics - Fundamentals and Challenges* (2013 ed.), Serge Kernbach (Ed.). Pan Stanford, Germany, Chapter 12, 411–469.

[7] Evert Haasdijk, P. A. Vogt, and Agoston Endre Eiben. 2008. Social learning in Population-based Adaptive Systems. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 1386–1392.

[8] Evert Haasdijk, Berend Weel, and Agoston Endre Eiben. 2013. Right on the MONEE. In *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, Christian Blum (Ed.). ACM New York, NY, USA, Amsterdam, The Netherlands, 207–214.

[9] Emma Hart, Andreas Steyven, and Ben Paechter. 2015. Improving Survivability in Environment-driven Distributed Evolutionary Algorithms through Explicit Relative Fitness and Fitness Proportionate Communication. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15*, Sara Silva (Ed.). ACM Press, New York, New York, USA, 169–176.

[10] Jacqueline Heinerman, Dexter Drupsteen, and Agoston Endre Eiben. 2015. Three-fold Adaptivity in Groups of Robots: The Effect of Social Learning. In *Proceedings of the 17th annual conference on Genetic and evolutionary computation.* ACM Press, New York, New York, USA, 177–183.

[11] Jacqueline Heinerman, Massimiliano Rango, and Agoston Endre Eiben. 2015. Evolution, Individual Learning, and Social Learning in a Swarm of Real Robots. In *2015 IEEE Symposium Series on Computational Intelligence.* IEEE, 1055–1062.

[12] Jacqueline Heinerman, Alessandro Zonta, Evert Haasdijk, and Agoston Endre Eiben. 2016. On-line Evolution of Foraging Behaviour in a Population of Real Robots. Springer, Cham, 198–212.

[13] Giles Mayley. 1996. Landscapes, Learning Costs, and Genetic Assimilation. *Evolutionary Computation* 4, 3 (sep 1996), 213–234.

[14] Stefano Nolfi and Dario Floreano. 1999. Learning and evolution. *Autonomous robots* 7, 1 (1999), 89–113.

[15] Nikita Noskov, Evert Haasdijk, Berend Weel, and Agoston Endre Eiben. 2013. MONEE: Using Parental Investment to Combine Open-Ended and Task-Driven Evolution. In *Applications of Evolutionary Computation*, A. I. Esparcia-Alcázar (Ed.), Vol. 7835. Springer, Berlin Heidelberg, 569–578.

[16] Carlos Segura, Carlos A. Coello Coello, Eduardo Segredo, and Arturo Hernandez Aguirre. 2016. A Novel Diversity-Based Replacement Strategy for Evolutionary Algorithms. *IEEE Transactions on Cybernetics* 46, 12 (dec 2016), 3233–3246.

[17] Andreas Steyven, Emma Hart, and Ben Paechter. 2016. Understanding Environmental Influence in an Open-Ended Evolutionary Algorithm. In *Parallel Problem Solving from Nature PPSN XIV*, Julia Handl et al. (Eds.). Vol. 9921 LNCS. Springer International Publishing AG, Chapter 86, 921–931.

[18] Andreas Steyven, Emma Hart, and Ben Paechter. 2015. The Cost of Communication: Environmental Pressure and Survivability in mEDEA. In *Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference - GECCO Companion '15*, Sara Silva (Ed.). ACM Press, New York, New York, USA, 1239–1240.

[19] R.S. Sutton and A.G. Barto. 1998. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 9, 5 (sep 1998), 1054–1054.

[20] Joanne H. Walker, Simon M. Garrett, and Myra S. Wilson. 2006. The balance between initial training and lifelong adaptation in evolving robot controllers. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* 36, 2 (apr 2006), 423–32.