Learning Aligned Local Evaluations For Better Credit Assignment In Cooperative Coevolution

Anonymous Author(s)

ABSTRACT

Cooperative coevolutionary algorithms prove effective in solving tasks that can be easily decoupled into subproblems. When applied to problems with high coupling (where the fitness depends heavily on specific joint actions), evolution is often stifled by the *credit assignment problem*. This is due to each agent evolving their policy using a shared evaluation function that is sensitive to the "noise" of all other agents' actions. Using fitness critics alleviates this problem by approximating a local model of an agent's contribution and using that signal as a fitness function. However, fitness critics suffer when the quality of the local approximation degrades. In this work, we present Global Aligned Local Error (GALE), a loss function that generates better credit-assigning local evaluations that aim to maximize the alignment of the local and global evaluations. In a multiagent exploration domain, we show GALE's ability to learn better credit assignment, which leads to improved teaming behavior.

CCS CONCEPTS

• **Computing methodologies** → **Cooperation and coordination**; **Multi-agent systems**.

KEYWORDS

Cooperative Coevolution, Multiagent Systems, Fitness Approximation

ACM Reference Format:

Anonymous Author(s). 2024. Learning Aligned Local Evaluations For Better Credit Assignment In Cooperative Coevolution. In *Genetic and Evolutionary Computation Conference (GECCO '24), July 14–18, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3638529.3654157

1 INTRODUCTION

Cooperative coevolutionary algorithms effectively train multiagent systems to tackle a variety of cooperative multiagent tasks such as UAV navigation [3, 27, 31], multiagent exploration [10, 19, 33], and robot soccer [13, 15, 20]. However, as the task complexity and coordination requirements increase, it becomes exceedingly difficult to determine how much an individual agent has contributed to the team's performance. This "credit assignment problem" often stifles learning highly cooperative behaviors.

Fitness shaping methods such as difference evaluations approach this credit assignment problem by comparing the evaluation of the team with and without each agent. These different evaluations provide significantly improved feedback to individuals but can be difficult to apply in domains where these evaluations are prohibitively expensive, sparse, or impossible to perform [1]. Prior work has evaluated learning-based approaches to learning local or individual evaluation agents that do not require additional evaluation from the environment, instead using the agents' experiences to learn a model of their contribution to the team as a whole. These methods often learn a lower-quality approximation of an agent's contribution, often promoting less effective behaviors [4]. Fitness critics aim to capture the temporal nature of agent behaviors, analyzing entire agent state trajectories to better assign credit. However, they often fall short because their local approximations are inadequate or fail to capture their true contributions to the team [29].

In this work, we present Global Aligned Local Error, GALE, a loss function to train local evaluations for better credit assignment in cooperative coevolutionary algorithms. Each agent uses a neural network function approximator that learns to map their individual state to a prediction of the global or team performance. These local evaluations are trained in parallel with the evolving agents, so as the approximations improve, so does the performance of the agents. GALE improves overall credit assignment by directly optimizing the alignment of an agent's local evaluation with the global evaluation.

The key insight of this approach is to view the problem of learning better credit assignment as a noise issue caused by other dynamic agents and correct it using a differentiable form of the alignment function from multiagent learning literature. Alignment is a key component of a local evaluation, as optimizing an aligned local evaluation also optimizes team performance using only local information. GALE directly maximizes alignment of our local evaluations to improve their overall approximation quality, which improves the learning of our agents.

The contributions of this work are:

(1) a loss function that promotes aligned local evaluations and,
(2) the incorporation of that loss function within the fitness evaluation step of evaluation during cooperative coevolution.

Experimental results in a simulated multi-agent exploration domain show that GALE can train better teamwork in our agents. We also show that these agents are learning better local approximations of a team's evaluation. This improvement in credit assignment correlates to a roughly 10% increase in teaming performance over the previous state-of-the-art when our teams increased in size.

2 BACKGROUND

2.1 Cooperative Coevolution

Many complex tasks, such as multiagent learning [12, 34, 35], circuit board design [30], cancer screening [14], and others [18] can be decomposed into multiple sub-problems, which can be leveraged to evolve better solutions [24]. Cooperative coevolutionary algorithms

GECCO '24, July 14–18, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0494-9/24/07.
https://doi.org/10.1145/3638529.3654157

solve these problems by dividing the solution into multiple subsolutions, where each subsolution is represented by a population. Subsolutions are then joined to form complete solutions, which are then evaluated using some evaluation function. This evaluation is then assigned to each subsolution used and is repeated until every element of every population has been evaluated [23, 25, 32].

In a cooperative multiagent context, a team would be divided into a series of agents, where each agent is represented by a population of control policies. Each agent seeks to learn a mapping of local states to their local actions that maximize the team outcome. To evaluate the team, we would draw a member from each population, and then have that team cooperate in some environment. At the end of the episode, the team is evaluated based on the global evaluation function. When all agent populations have been evaluated, mutation as selection occurs within each population and the cycle repeats, optimizing overall team performance [11, 34].

However, agents often struggle to optimize their policies as their evaluations are based on team performance rather than individual performance. It is unclear whether changes in performance are due to an agent's mutation or a mutated team member, causing the selection operator to frequently keep lesser policies or discard better ones. To ameliorate this issue of credit assignment, we need evaluation functions that are specific to each agent, describing their contribution to the teams [2, 26].

2.2 Fitness Shaping and Alignment

Fitness shaping presents one approach to the credit assignment problem, where a shaping term is added to the fitness evaluation to better evaluate an individual's contribution to the global evaluation function [6, 22]. This global evaluation function G, evaluates the joint state of all of the agents, z [1]. This joint state consists of a list of all states observed and actions taken by each agent in the system. Difference evaluations, shown in equation 1, are a fitness shaping method that better assigns credit through the use of counterfactual global evaluations [1, 8]. Here we evaluate the gain from the team evaluated with agent i ($G(z)$) and the joint state of the team evaluated with agent i removed (z_{-1}) and replaced with a counterfactual state, c_i. Difference evaluations effectively measure the change in team performance resulting from the replacement of a single agent in the team.

$$D_i(z) = G(z) - G(z_{-i} \cup c_i) \tag{1}$$

Difference evaluations can be difficult to apply in domains where this counterfactual evaluation is infeasible, such as highly stochastic domains. Additionally, these evaluations provide limited credit assignment in domains where the evaluation function is sparse, providing mostly fitness scores of zero [5]. Consider an example of robot soccer where a robot passes the ball to the correct team member but the team member doesn't score a point. All counterfactual actions for the passing robot do not lead to the team scoring, thus it will always receive an evaluation of zero, despite having a positive on the team. In these sparsely evaluated tasks, difference evaluations often provide only marginal benefit.

For a local evaluation function to be effective it must have two properties: high sensitivity and high alignment. Sensitivity measures how much an evaluation changes in response to changes in a single agent [5]. Global evaluations have low sensitivity as they measure the impact of all agents. As previously discussed difference evaluations have low sensitivity in sparsely evaluated environments. Alignment measures how aligned a local evaluation is with the global evaluation [1]. Equation 2 defines two functions ($G(z)$ and $g(z)$) being aligned if the signs of their derivatives are equal.

$$sign(\frac{dG(z)}{dz_i}) = sign(\frac{dg(z)}{dz_i}) \tag{2}$$

Alternatively, we can measure the degree of alignment between the global evaluation G and the local agent-specific evaluation g using equation 3, where u is the unit step function and z, z' represent two points in the joint state space. If two functions are completely aligned, then their alignment value will be 1.0. If they are completely misaligned, they will measure 0.0 alignment. Alignment is a key property of a local evaluation function as maximizing an aligned local evaluation also maximizes the global evaluation, resulting in quality cooperative teaming behavior. Additionally, this continuous measure of alignment is helpful when learning local evaluations as we can determine their overall degree of alignment with the team evaluation [1].

$$Alignment = \frac{\int\int u[(G(z) - G(z'))((g(z) - g(z'))]dz_i dz'_i}{\int\int dz_i dz'_i} \tag{3}$$

2.3 Learning Credit Assignment

Learning-based approaches to the credit assignment attempt to learn a local approximation of an agent's contribution, without requiring additional team evaluations. In these methods, we often treat credit assignment as a regression problem, where a function approximation for agent i, f_i parameterized by some θ, optimizes a mapping from an agent's local state to the expected global evaluation using Mean Squared Error (MSE) of j samples, as shown in equation 4. This is learned from the populations' interactions with the environment, where agents receive local state information along with a global evaluation. However, learning these mappings is difficult, often learning a rough approximation of an agent's contribution to the team outcome [4].

$$MSE_i = \frac{1}{n}\sum_{j}^{n}(G(z^j) - f_i(z_i^j, \theta))^2 \tag{4}$$

Fitness critics extend prior credit assignment methods by evaluating not just the end-state of the system, but every state during an episode T. The highest evaluation ($FC_i(z_i, \theta)$) from this set is then assigned to an agent as its fitness[28, 29], as shown in equation 5.

$$\max_{z \in T} FC_i(z_i, \theta) \tag{5}$$

This method provides an optimistic evaluation of an agent, promoting partially good solutions that may not be apparent from the end state of the system. In doing so, the fitness critic can better leverage the information provided by the rough approximation of the function approximation, often leading to better-learned performance by the agent. However, biasing evaluation using the max operator does not guarantee improved performance and can lead to a reduction in overall learned teaming behavior [7, 9].

3 GLOBAL ALIGNED LOCAL ERROR

In this work, we present Global Aligned Local Error (GALE), a loss function for neural networks that promotes alignment of the learned local, agent-specific evaluations with the team-wide evaluation function. This agent-specific feedback improves cooperative coevolution by significantly reducing the impact of the credit assignment problem that arises from optimizing a centralized fitness function. We extend previous local fitness modeling using MSE by introducing an alignment term that helps each agent in our system better align their local evaluation with the team evaluation, providing an easier signal to optimize for cooperative coevolution. Strong alignment is a desirable property of any local evaluation function and can be directly optimized using a differentiable form of alignment as a loss function. In this work, we optimize alignment by minimizing negative alignment, which we refer to as minimizing misalignment of our local evaluations. We present the functional form of Misalignment Error (MAE) for agent i, our alignment penalty, in 6.

$$MAE_i = \frac{-\sigma[(G(Z) - G(Z)^T) \odot (f_i(Z_i;\theta) - f_i(Z_i;\theta)^T)]}{n^2} \quad (6)$$

This loss function is a differentiable, sampling-based version of equation 3. Here, we evaluate a batch of n joint states, concatenated to form the $n\ x\ m$ matrix Z_i where m is the length of an agent's state vector. The function f_i represents the agent's learned local evaluation function, likely a neural network, that is parameterized by its weights θ. The goal of this function is to learn a mapping of an agent's local states to an evaluation aligned with the team evaluation $G(Z)$.

Evaluating $f_i(Z_i;\theta)$ produces an $n\ x\ 1$ matrix of evaluations, one for each local state the agent observes. Subtracting the transpose of this matrix produces an $n\ x\ n$ matrix where each element represents the difference between the agent's evaluation of two states, equivalent to $g(z) - g(z')$ from equation 3. The evaluation of $G(Z)$ and the subtraction of its transpose also produces an $n\ x\ n$ matrix that represents the difference between the team evaluations at the same states.

We then take the Hadamard product of the two resulting square matrices to produce another square matrix, which we will denote as the alignment matrix. In this matrix, positive values correlate to the alignment of the local objective function with the global objective, while negative values indicate misalignment. Effectively, positive values indicate that for those two states, equation 2 holds true, while negative values do not.

The alignment matrix is transformed by the sigmoid function, σ, as this is a differentiable replacement for the unit step function from the original alignment equation. This gives a pair-wise count of the number of state-pair evaluations that are locally aligned with the global evaluation. We can then divide by the total number of local state pairs, n^2, to produce an average differentiable alignment value between 0 and 1. Finally, we append a negative sign to the front of the equation to produce a metric of misalignment. By minimizing misalignment, we can train functions that are strongly aligned with the global evaluation.

3.1 GALE: Handling Sparse and Discrete Evaluations

It should be noted that in the case of sparse or discrete global evaluation functions, there will be many repeating values of $G(z)$. The differences in these values will lead to zeros largely populating the alignment matrix, which will provide no useful gradient information. This problem becomes worse in these settings when the batch size is large, causing many of the values in the alignment matrix to be zero, skewing the average value, and providing little alignment information. In many cases, we cannot define a more continuous global evaluation, and this problem of evaluation sparsity cannot be avoided through MAE optimization alone.

To correct this issue, we present GALE, a loss function consisting of Mean Squared Error (MSE) plus MAE as shown in equation 7. Using MSE provides a dense loss evaluation unless the expected values perfectly match the actual global evaluation. Including this term provides information to the network when the evaluation is sparse or dense. Additionally, learning a local approximation of $G(z)$ using MSE alone is difficult, due to the high degree of noise injected into the global evaluation from the other agent. This misalignment term helps ensure that this approximation will maintain alignment with the global evaluation. The resulting loss function takes two different approaches to learning local evaluations that mitigate each other's deficiencies, providing a highly learnable signal for generating local evaluations for coevolution. In summary, GALE trains a high-quality mapping of an agent's local state to the expected team outcome, which is used to determine the individualized fitness signal in cooperative coevolution each generation.

$$GALE_i = MSE_i + MAE_i \quad (7)$$

While this loss function may appear to be a scalarization of two objectives, this loss function is not a multiobjective optimization problem. Theoretically, functions learned using MSE should be fairly aligned as it is learned from the global evaluation function. The addition of the MAE term better ensures alignment. While there are an infinite number of functions that are aligned with $G(z)$, incorporating MSE effectively collapses optimization to a single function, an aligned local approximation of $G(z)$. As a result, when $G(z)$ is sparse or discrete, then MSE will provide a dense signal to optimize, while MAE will act as an alignment corrective term. Practically, it should be noted that the magnitude of the global evaluations will result in disproportionate MSEs. In these cases, a scaling term may need to be added so that these errors are optimized equally. In this work, we do not scale the errors as they were of similar orders of magnitude.

3.1.1 Motivating Example: A Two Agent Cooperation Problem. Let us assume that we have an evaluation function for a two-agent system: $G(z_0, z_1) = (z_0 * z_1)^{100}$ on the domain of $z \in [0, 1]$. This represents a cooperative problem, where two agents must move to the correct state together to solve the task. However, optimizing this signal is difficult as nearly all values will be close to zero, except points close to $z = (1, 1)$. Additionally, one agent may be at $z = 1$, the correct local solution, but receive a score of zero due to the other agent being in the wrong state.

To more easily solve this problem, we assign each agent a function approximator, $f_i(z_i, \theta)$ that approximates their contribution to

the task. After training this function using GALE, each agent would learn an approximation: $f_i(z_i, \theta) = z_i^{100}$. This function is aligned with the global evaluation and models the global function at a local level. This decouples an agent evaluation from the team evaluation, transforming the task of training the team of agents into parallel agent-optimizing tasks. As a result, each agent can identify that the state $z_i = 1$ is the state that each agent must occupy to solve the problem.

4 MULTIAGENT EXPLORATION DOMAIN

We evaluate GALE on a simulated multiagent exploration domain known as the Rover Domain in prior literature [1]. In this domain, multiple agents must explore their environment and observe valuable Points Of Interest (POI). Some of the POIs are more valuable than others and the agents must identify and move to the most valuable POI to be successful. We increase the difficulty of this task by requiring multiple simultaneous observations of each POI to be considered successfully observed.

To sense their environment, the rovers have four pairs of sensors where each pair consisting of a rover sensor and a POI sensor. The sensor model for the other rovers is the sum of inverse distances between the agent i and the other agents i' in the system, denoted by equation 8.

$$S_{robot} = \sum_{i'} \frac{1}{\delta(i, i')} \tag{8}$$

The sensor models for the POIs are defined similarly to the rover ones as shown in equation 9. This sensor model incorporates the distance between agent i and POI j and is also scaled by the value of the POI V_j, so that rovers can sense which POI are more valuable.

$$S_{POI} = \sum_{j} \frac{V_j}{\delta(i, j)} \tag{9}$$

Each pair of sensors can sense rovers and POIs in a 90-degree arch, with four pairs of sensors being equally spaces around the rover to fully view their surroundings. This results in a state vector for each agent of size eight.

At every time step, each rover can take two continuous actions: one for forward and backward movement, and another for turning. Each time step the rover can move $[-1, 1]$ units forward and turn $[-45, 45]$ degrees. This action is chosen by the agent's control policy using its state information.

The team of agents is evaluated using the global objective function defined in equation 10. This equation represents the sum of the values of POIs that are successfully observed. The value is also scaled by the distance of the closest agent i'' to POI j to ensure that agents move close to the POIs. The indicator function $I(j)$ returns 1.0 if POI j has been successfully observed and 0.0 otherwise. For a POI to be successfully observed, two rovers must be within the observation radius of the POI at the end of the episode.

$$G = \sum_{j} \frac{V_j I(j)}{max(1, \delta(i'', j))} \tag{10}$$

Overall, this domain presents a challenge for cooperative coevolution as all agents contribute to a very sparsely evaluated task, that only defines success at a team level. This lack of clear agent-specific feedback highlights the difficulties of the credit assignment problem, which leads to difficulty in optimizing agent populations.

4.1 Experiments

We evaluate GALE in the Rover Domain, with a POI configuration described by figure 1. Here, agents start randomly in the center of a 30 by 30 unit map, shown by the stars, and must observe the various POI, denoted by blue triangles. The values of the POIs are also indicated next to each POI. This configuration was chosen as it provides a sufficient challenge for the agents to explore. By having the higher-valued POIs separated from each other, the agents cannot simply move as a group and randomly discover the best POI. To sufficiently solve this task, the agents need to divide into groups of two, the number of agents needed to observe a POI, and move towards the higher valued POIs. Multiple configurations of this domain were evaluated, showing similar trends. Results are reported using figure 1 as it was the most challenging of those tested. Additionally, we focus on a single domain to provide a more in-depth analysis of the evolved agents and their local learned fitness functions.

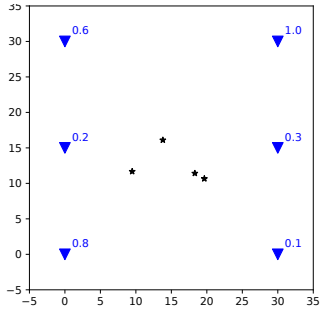


Figure 1: The rover domain configuration used in each experiment. Agents (stars) start in the center and must observe the POI (triangles) with varying values denoted above them. The axes portray the scale of the environment, with the POI distributed in a 30-unit by 30-unit area.

In the first set of experiments, we use cooperative coevolution to train four, six, and eight agents to solve this task. As the number of agents in the system increases, the problem of credit alignment becomes increasingly significant as more agents are contributing to the global objective. It should also be noted that increasing the number of agents effectively changes the overall team task. For every two agents added, one more POI can be observed, resulting in more subtasks within the environment that the agents need to learn to divide amongst themselves.

In this first set of experiments, we compare GALE to six baseline evaluation functions for cooperative coevolution. The first two baselines are the *global evaluation* function and *difference evaluations*. These highlight the difficulties in training from the shared global information and the agent-specific shaped version. The next two baselines consist of training our local evaluations using *MSE* and *Misalignment Error*. Training local evaluations represents prior

work in fitness modeling. The two methods also represent an ablation study on the relative impacts of these two error terms on the overall learned outcome. The last two baselines represent two versions of *fitness critics*, the previous state-of-the-art in learned local evaluations. The first version of the fitness critic is the default version, described in prior work, and the second version incorporates the GALE loss function.

In the second set of experiments, we compare the evaluation functions learned by each agent using fitness critics and using GALE. In these experiments, we vary the states of one agent at a time, to determine how the changes of each agent affect their evaluations.

4.2 Experimental Parameters

We train our system using cooperative coevolution where each agent is given a population of size 32. Each agent is represented as a neural network with a hidden layer of size 20, an input of 8, and an output of 2. The hidden layer has a ReLU activation function and the output has a Tanh activation [17]. Agents are selected using binary tournament selection. Mutation only occurs on the parameters of the network, with a 5% change of each parameter having its value replaced using one from a standard Cauchy distribution. Agents are trained for a total of 2000 generations.

All local evaluation neural networks share the same architecture and training structure, only varying loss functions. Each network contains two hidden layers of size 80 with Tanh activation functions. The output is of size 1 and is left linear. We optimize these networks using ADAM with a learning rate of 5e-4 [16]. Each agent population is assigned a neural network that maps that agent's local state to the expected global evaluation, defined by their type of local evaluation. Note that the performance is minorly affected by the architecture of the network, with each domain needing minor network architectural tuning. We store every agent's end state and team evaluation in an agent-specific dequeue of size 3e4 tuples. We sample 100 batches of size 24 between each generation to train these networks. The dequeue acts as a method of stabilizing the training of these networks, as shown in deep reinforcement learning literature [21]. For the fitness critics, state trajectories are stored in the dequeue in place of the end state.

Experiments consist of 36 statistical trials. The learning curves presented show the mean team performance and standard error produced by these trials. Statistical significance was determined using a one-sided t-test of the performance means with a confidence interval of 95%.

5 RESULTS

5.1 GALE And Baseline Teaming Performance

In this first experiment, we train a team of four agents in the Rover Domain using GALE and compare it to our six baselines, shown in figure 2. We see that agents trained using the global evaluation function struggle to learn meaningful policies due to the problem of credit assignment. Additionally, we only see a minor improvement provided by difference evaluations due to the sparsity of the global evaluation. As a result, different valuations can better assign credit, but the overall learning signal is still too sparse to optimize effectively.

Looking at the learning-based local evaluations, we see that learning from alignment only provides better-learned policies, but not better than other learned local evaluations. We also see a much later increase in performance compared to the other methods. With the sparse global evaluation, learning aligned local evaluations is difficult, as the alignment matrix is largely populated by zeros, providing infrequent useful information and causing these networks to learn much more slowly than the other neural networks. Despite these shortcomings, we notice only a minor reduction in teaming performance.

When comparing fitness critic performance to GALE,we do not see any statistically significant difference in performance, with a p-value of 0.331. This is because GALE only evaluates the end state of the system, unlike the fitness critic which evaluates every state in an episode. The fitness critic can better leverage its lower-quality approximation of an agent's contribution to a team, to achieve decent performance. Conversely, GALE relies solely on its ability to model an agent's contribution to the team to achieve its high performance. Additionally, the inclusion of MAE into the fitness critic does not appear to improve performance due to the way that fitness critics bias the agent's evaluations.

Agents trained using local evaluations trained on the MSE of the global evaluation appear to perform slightly worse than most other learning-based evaluations. This is expected as there are no mechanisms included to improve either the quality or temporal aspects of credit assignment.

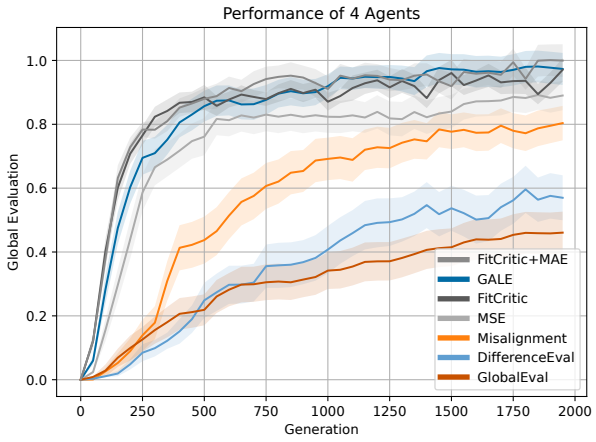


Figure 2: We compare the learned performance of agents evolved using GALE and alignment to other baseline local evaluations. We train four agents in each team, resulting in a maximum score of 1.8. The legend is sorted by average final team performance.

In the next experiment, we increase the number of agents to six, further increasing the need for effective credit assignment. The performance curves of the agents trained in this setting are shown in figure 3. As a result of increasing the number of agents, we see improved performance from difference evaluations, as this function is designed to assign credit better than the global evaluation. Similarly to the previous experiment, using alignment as a loss function does not perform as well as the other methods, but the difference

narrows. Even though training using only alignment is difficult due to signal sparsity, we are still able to get a relatively decent performance that is nearly on par with the fitness critic.

Looking at GALE, we see that the agents trained using this learned evaluation function exhibit the best performance with a p-value of 0.013. Here, agents more effectively learn a local approximation of their contribution to a team by leveraging both parts of GALE's loss function. Unlike the alignment portion which struggles with the global evaluation sparsity, the MSE term can train the local evaluation to predict the global evaluation based on a single agent's state. However, using only MSE is still prone to errors introduced by the noise from the other agents in the system. By adding this misalignment term, we can correct these errors and correct the predictions from the network that are misaligned with the global evaluation. The result is an improved approximation of an agent's contribution to a team. With this cleaner learning signal, each agent within cooperative coevolution can evolve with reduced difficulty.

Figure 3: We train six agents in each team, using GALE and the baselines, leading to a maximum score of 2.4. Again, the legend is sorted by average final team performance, with GALE providing the highest quality policies through improved credit assignment.

In the last experiment from this set, we further increase the number of agents to eight, providing a task with the strongest need for quality credit assignment. In figure 4, local evaluations optimized using alignment appear to perform equally to fitness critics and many other learned local evaluations. As we increase the number of agents, alignment in the local evaluation becomes necessary to train effective teamwork in agents. This is also highlighted by the performance improvement provided by difference evaluation, an evaluation that is completely aligned with the global evaluation.

The fitness critic shows the lowest relative performance in the eight-agent experiment. Agents sometimes benefit from the temporal aspect that is captured by the evaluation of the whole trajectory of action performed by the fitness critic. However, biasing the evaluation with the max operator provides an optimistic view of an agent's trajectory, which can either help or hurt the agent's performance. In this case, this type of biasing impedes agent learning.

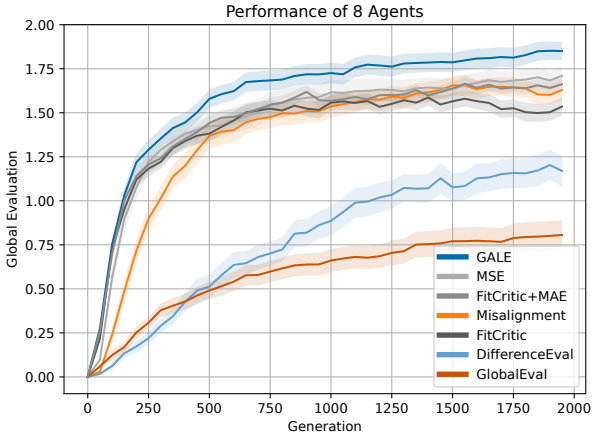


Figure 4: We train eight agents in each team, leading to a maximum score of 2.7. Here, the problem of credit assignment is most relevant, leading to the largest separation in performance between GALE and baselines.

GALE shows the clearest delineation from other methods in the eight-agent experiment with a p-value of 0.022. In this experiment, the problem of credit assignment is the most pronounced, which GALE is designed to handle. Again, by promoting alignment, agents can produce more accurate estimations of their impact on the team outcome, improving learning.

5.2 Visualizing Learned Local Evaluation Functions

In this second set of experiments, we perform a post hoc analysis of the learned local evaluation functions of agents that learned to complete the four-agent task as shown in figure 5. We compare the functional mappings learned by fitness critics, agents trained using MSE of the global evaluation, those trained purely for alignment, and GALE to the global evaluation function.

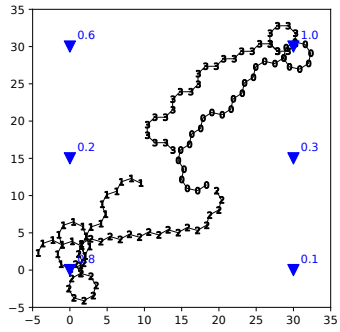
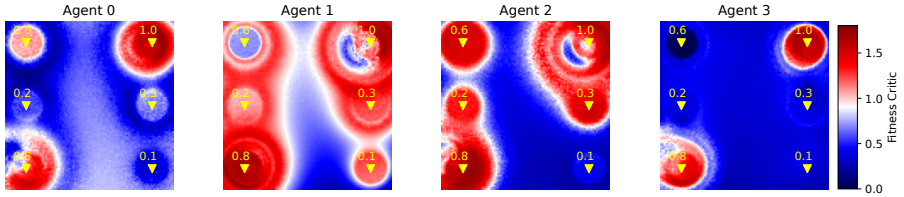
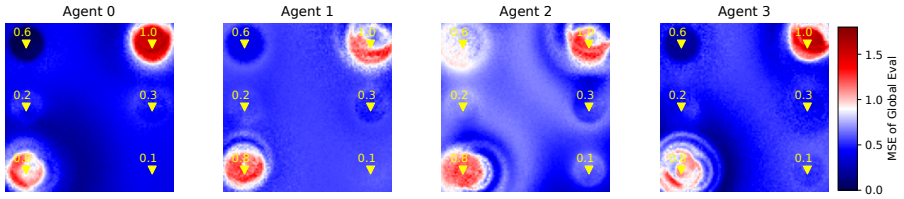


Figure 5: An example of paths taken by agents that successfully complete the task of observing the two highest valued POIs. Agent path numbers differentiate paths for agents 0 through 3.

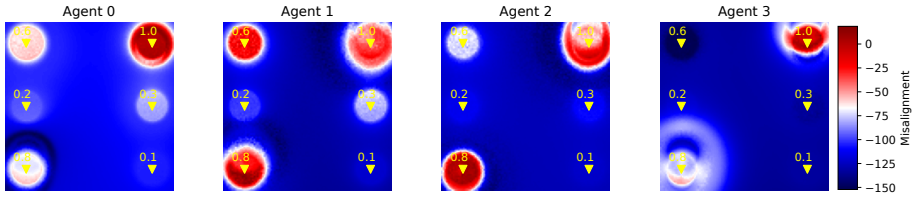
For each agent, we randomly sample states for that agent on a fine grid of coordinates, while keeping the other agents in the



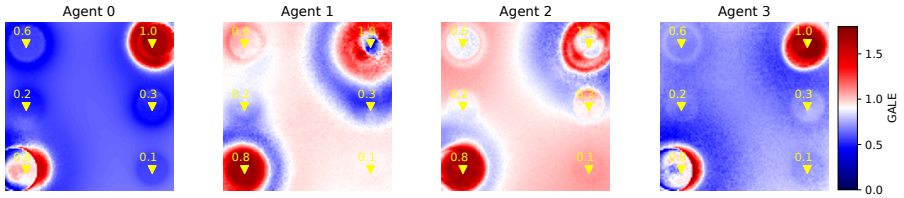
(a) Fitness Critics provide temporally skewed credit assignment.



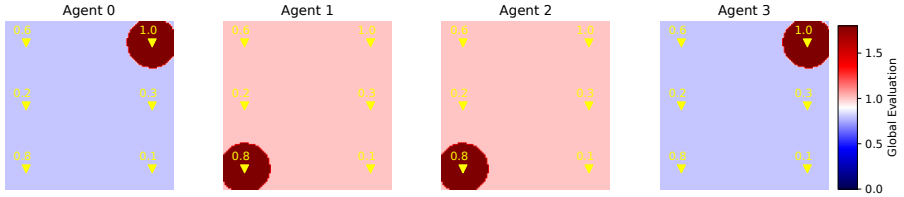
(b) Global value approximation produces noisy functions.



(c) Alignment optimization provides improved credit assignment but is inconsistent to train.



(d) GALE produces the highest quality credit assignment, most closely matching the global evaluation.



(e) The global evaluation provides ground truth as to how an agent affects team performance.

Figure 6: We plot and compare the evaluation functions learned by Fitness critics (a), MSE of the Global Evaluation (b), Misalignment Error (c) and GALE (d), and compare them to the global evaluation function (e) for each agent in the system using randomly samples states. Note that GALE learns a high-quality local approximate of the global evaluation for *all* agents on the team. This largely removes the noise from the other agents in the system, providing local evaluations that are easier to optimize.

system in their ending states. We then evaluate each function using this sampled state, and then average the resulting local evaluation for that grid coordinate. This results in the function evaluation maps shown in figure 6. We perform this analysis for each agent on each team and number the agents by similar function within the team. To summarize, agents labeled 0 and 3 move to observe the top right POI while agents labeled 1 and 2 observe the bottom left POI, achieving a maximal score.

Figure 6e represents how the global evaluation changes with the changing behavior of each agent. This can be viewed as similar to difference evaluations in that it measures how the global evaluation changes as a function of the changes to individual agents. Figure

6e represents an idealized view of the global objective. In reality, the agents do not have access to this degree of information, as the global evaluation is sampled by the whole team of dynamic, learning agents at once, where each agent injects their contribution into the signal. Ideally, each agent would learn to approximate this function at a local level, without requiring information from other agents.

The fitness critic can learn a rough approximation of an agent's contribution, as shown in figure 6a. Here, we see that the highest valued states are the ones within the observation radius of the POI observed by each agent, similar to the global evaluation. However, a majority of the other state evaluations do not appear to be aligned with the global evaluation. This lower-quality approximation can sometimes be compensated for through the evaluation of the agent's full state trajectory but is not guaranteed.

In figure 6b, we see the functions trained using MSE on the global evaluation. These evaluations attempt to approximate and predict the team outcome at a local level. However, they struggle to learn an effective model of this function due to the noise injected by the other dynamic agents. This noise can be seen in figure 6b, where the values of the useful POIs are noisy and lower than expected. Additional areas of underestimation are seen in the areas outside of the POIs for agents 1 and 2. These errors are due to approximation difficulties presented by agent noise.

When we observe the local evaluations learned using Misalignment Error (figure 6c), we notice clearer separation between useful states near POIs and those outside of the observation radius. However, the values produced by these aligned functions are far outside of the ranges presented by the other local and global evaluations. Unlike the other methods, Misalignment Error does not try to model a local approximation of the global evaluation. Instead, it only tries to find a function that is maximally aligned with the global objective. As there are an infinite number of aligned functions, the network often learns functions with large ranges. The scale of the evaluation function is not problematic as it does not change the selection process's ability to differentiate quality policies. The function only needs to highly value useful states, as defined by the global objective, and penalize less useful ones. We see that indeed, the aligned value function learns to value the POIs each agent needs to move to to achieve the maximum evaluation.

The highest quality approximation of an agent's contribution to the global objective is learned using GALE, shown in figure 6d. This function learns the cleanest delineation between the value of the POI the agent needs to move to versus the other POIs. As seen in the global evaluation function, if an agent moves to any other POI than the one it ends on, the team receives a lower score. This is because all other POIs have either zero agents observing them or exactly enough agents observing the POI. For these POIs, adding an observing agent leads to either too few agents for successful observation or an excessive number of agents viewing the POI. The function learned using GALE, was able to learn these correlations, producing fairly uniform regions of value outside of their observed POIs. The other models often significantly overestimate the values within the radius of observation of the other POIs and underestimate the values outside of these radii. Overall, GALE learned the best local approximation. As a result, the agents have a higher-quality local learning signal, which explains the increase in performance seen in the first set of experiments.

While GALE can learn a quality local approximation of the global evaluation, it still ignores the temporal nature of an agent's policy. Fitness critics attempt to learn credit assignment along with the temporal aspects of the policy. Learning both inhibits the learning of credit assignment in favor of capturing these temporal correlations. When credit assignment becomes more relevant (when the number of agents in the system increases), we see GALE perform better.

In summary, we show one method of incorporating alignment into local evaluations via GALE. By including optimization of MSE of the global evaluation, we effectively collapse the infinite number of aligned functions to an aligned local model of the global evaluation. While this method is effective in training agents, it likely does not produce the optimal local evaluation function for learning. As there are an infinite number of aligned functions, there are likely aligned local evaluations that are easier for an agent to optimize. However, identifying the learnability of a function remains a non-trivial task, and optimizing local evaluations for both alignment and learnability remains future work.

6 CONCLUSION

In this work, we presented GALE, a loss function for improved credit assignment, especially in environments with sparse evaluations, for cooperative coevolution. We show that alignment can be represented as a differentiable function that our local evaluation functions can optimize directly. Incorporating alignment into MSE regression of the global objective produces high-quality assignment of credit to each agent in the system, improving learning. In a two-dimensional exploration domain, we observed improved performance from agents trained using these learned local evaluations, especially in tasks with a higher number of agents.

While we see significantly improved credit assignment using GALE, it does not solve the problem of credit assignment as a whole. As this method focuses on evaluating the end-state of the system, it ignores the information provided by the states leading up to the final state. Future work will look into incorporating sequential modeling mechanisms such as recurrent architectures or attention mechanisms. These network extensions should help capture the temporal aspect of many multiagent and cooperative problems, and further help assign credit in more complex domains.

Another potential research direction involves improving the learnability of the local evaluations using alignment. Learning aligned functions from sparse or discrete evaluations leads to relatively discrete local evaluations that are difficult to optimize. Introducing a penalty for such discrete local evaluations may potentially produce more continuous local evaluations. These local evaluations may be easier to optimize for our agents as they provide a continuous gradient for our agents to follow to reach useful states, which may improve convergence rates and overall team performance.

REFERENCES

[1] Adrian K Agogino and Kagan Tumer. 2008. Analyzing and Visualizing Multiagent Rewards in Dynamic and Stochastic Domains. *Autonomous Agents and Multi-Agent Systems* 17, 2 (2008), 320–338.

[2] Yu-Han Chang, Tracey Ho, and Leslie P Kaelbling. 2004. All Learning is Local: Multi-agent Learning in Global Reward Games. In *Advances in neural information processing systems*. 807–814.

[3] Mitchell Colby, Jen Jen Chung, and Kagan Tumer. 2015. Implicit Adaptive Multi-Robot Coordination in Dynamic Environments. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5168–5173.

[4] Mitchell Colby, Theodore Duchow-Pressley, Jen Jen Chung, and Kagan Tumer. 2016. Local Approximation of Difference Evaluation Functions. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. 521–529.

[5] Mitchell K Colby, Sepideh Kharaghani, Chris HolmesParker, and Kagan Tumer. 2015. Counterfactual Exploration for Improving Multiagent Learning.. In *AAMAS*. 171–179.

[6] Mitchell K Colby and Kagan Tumer. 2012. Shaping Fitness Functions for Coevolving Cooperative Multiagent Systems.. In *AAMAS*, Vol. 1. 425–432.

[7] Joshua Cook, Tristan Scheiner, and Kagan Tumer. 2023. Multi-Team Fitness Critics For Robust Teaming. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 2406–2408.

[8] Joshua Cook and Kagan Tumer. 2022. Fitness Shaping For Multiple Teams. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 332–340.

[9] Joshua Cook, Kagan Tumer, and Tristan Scheiner. 2023. Leveraging Fitness Critics To Learn Robust Teamwork. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 429–437.

[10] Ping-an Gao, Zi-xing Cai, and Ling-li Yu. 2009. Evolutionary Computation Approach to Decentralized Multi-Robot Task Allocation. In *2009 Fifth International Conference on Natural Computation*, Vol. 5. IEEE, 415–419.

[11] Jorge Gomes, Miguel Duarte, Pedro Mariano, and Anders Lyhne Christensen. 2016. Cooperative Coevolution of Control for a Real Multirobot System. In *International Conference on Parallel Problem Solving from Nature*. Springer, 591–601.

[12] Jorge Gomes, Pedro Mariano, and Anders Lyhne Christensen. 2015. Cooperative Coevolution of Partially Heterogeneous Multiagent Systems. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*. International Foundation for Autonomous Agents and Multiagent Systems.

[13] Jorge Gomes, Pedro Mariano, and Anders Lyhne Christensen. 2017. Dynamic Team Heterogeneity in Cooperative Coevolutionary Algorithms. *IEEE Transactions on Evolutionary Computation* 22, 6 (2017), 934–948.

[14] Shan He, Guanbo Jia, Zexuan Zhu, Daniel A Tennant, Qiang Huang, Ke Tang, Jing Liu, Mirco Musolesi, John K Heath, and Xin Yao. 2016. Cooperative Co-Evolutionary Module Identification with Application to Cancer Disease Module Discovery. *IEEE Transactions on Evolutionary Computation* 20, 6 (2016), 874–891.

[15] Andreas Kallinteris, Stavros Orfanoudakis, and Georgios Chalkiadakis. 2022. The Performance Impact of Combining Agent Factorization with Different Learning Algorithms for Multiagent Coordination. In *Proceedings of the 12th Hellenic Conference on Artificial Intelligence*. 1–10.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Yuanzhi Li and Yang Yuan. 2017. Convergence Analysis of Two-Layer Neural Networks With ReLU Activation. *Advances in neural information processing systems* 30 (2017).

[18] Xiaoliang Ma, Xiaodong Li, Qingfu Zhang, Ke Tang, Zhengping Liang, Weixin Xie, and Zexuan Zhu. 2018. A Survey on Cooperative Co-Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation* 23, 3 (2018), 421–441.

[19] Xin Ma, Qin Zhang, and Yibin Li. 2007. Genetic Algorithm-Based Multi-Robot Cooperative Exploration. In *2007 IEEE International Conference on Control and Automation*. IEEE, 1018–1023.

[20] P Mariano, AL Christensen, and J Gomes. 2014. Avoiding Convergence in Cooperative Coevolution with Novelty Search. *Avoiding convergence in cooperative coevolution with novelty search* (2014), 1149–1156.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari With Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602* (2013).

[22] Andrew Y Ng, Daishi Harada, and Stuart Russell. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Icml*, Vol. 99. 278–287.

[23] Liviu Panait, R Paul Wiegand, and Sean Luke. 2004. A Sensitivity Analysis of a Cooperative Coevolutionary Algorithm Biased for Optimization. In *Genetic and Evolutionary Computation Conference*. Springer, 573–584.

[24] Mitchell A Potter and Kenneth A De Jong. 1994. A Cooperative Coevolutionary Approach to Function Optimization. In *International Conference on Parallel Problem Solving from Nature*. Springer, 249–257.

[25] Mitchell A Potter and Kenneth A De Jong. 2000. Cooperative Coevolution: An Architecture For Evolving Coadapted Subcomponents. *Evolutionary computation* 8, 1 (2000), 1–29.

[26] A. Rahmattalabi, J. J. Chung, M. Colby, and K. Tumer. 2016. D++: Structural Credit Assignment in Tightly Coupled Multiagent Domains. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*. 4424–4429.

[27] Carrie Rebhuhn, Ryan Skeele, Jen Jen Chung, Geoffrey A Hollinger, and Kagan Tumer. 2015. Learning to Trick Cost-Based Planners Into Cooperative Behavior. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4627–4633.

[28] Golden Rockefeller, Shauharda Khadka, and Kagan Tumer. 2020. Multi-level fitness critics for cooperative coevolution. In *AAMAS Conference proceedings*.

[29] Golden Rockefeller, Patrick Mannion, and Kagan Tumer. 2019. Fitness Critics for Multiagent Learning. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 222–224.

[30] Michaela Sikulova, Gergely Komjathy, and Lukas Sekanina. 2014. Towards Compositional Coevolution in Evolutionary Circuit Design. In *2014 IEEE International Conference on Evolvable Systems*. IEEE, 157–164.

[31] Pedro Trueba, Abraham Prieto, and Francisco Bellas. 2017. Embodied Evolution Versus Cooperative Coevolution in Multi-Robot Optimization: A Practical Comparison. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 79–80.

[32] Eiji Uchibe, Masateru Nakamura, and Minoru Asada. 1999. Cooperative and Competitive Behavior Acquisition for Mobile Robots Through Co-evolution. In *Proc. of the Genetic and Evolutionary Computation Conference*. Citeseer, 1406–1413.

[33] Connor Yates, Reid Christopher, and Kagan Tumer. 2020. Multi-Fitness Learning for Behavior-Driven Cooperation. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 453–461.

[34] Chern Han Yong and Risto Miikkulainen. 2001. *Cooperative Coevolution of Multi-Agent Systems*. Technical Report. Citeseer.

[35] Chern Han Yong and Risto Miikkulainen. 2009. Coevolution of Role-based Cooperation in Multiagent Systems. *IEEE Transactions on Autonomous Mental Development* 1, 3 (2009), 170–186.