# Exploring Multi-Species Multi Agent Predator and Prey Interactions Using RL: Introducing RL BattleGround

**Aditya Somasundaram** [1] **Chiara von Gerlach** [1] **Likhith Ayinala** [1] **Leoric Luo** [1]

## Abstract

We extend traditional single-agent predator-prey models to a multi-species multi agent reinforcement learning (MARL) framework, where cyclic dominance drives population size. Our approach is inspired by the classic game of Rock-Paper-Scissors, where each species is simultaneously predator and prey. These three species interact in a shared environment, with policies learned through deep MARL. This research analyses flexible reward structures where agents learn policies that balance their long and short term objectives. We analyze the stages of learning from a random policy to evolved strategies in the single predator single prey, multi predator single prey, single predator multi prey, and a 3 species system. We hope that the library we introduce, RL BattleGround, helps researchers simulate multi-species multi agent interactions with ease and allows for flexible and more controlled simulations. The code can be found here.

## 1. Introduction

In many societal systems where a precarious balance of power is upheld, arena-style combat is common. For fear of losing what they have to an adversary, entities refrain from overextending their capabilities. This cyclic dominance model is exemplified by the Rock-Paper-Scissors model, in which *rock* loses to *paper*, which in turn loses to *scissor*, which in turn loses to *rock*.

(Hofbauer & Sigmund, 1998) presents a game theoretic approach to the study of population dynamics and evolutionary games. (Yang et al., 2017) talks about using RL agents to simulate predator-prey interactions. The Lotka-Volterra equations and other conventional predator-prey models concentrate on interactions between two species (Berryman, 1992; Wangersky, 1978). Large-scale multi-agent predator-prey systems are thoroughly surveyed in (Yamada et al., 2020). In this work, we develop a library (RL BattleGround) that facilitates multi-species multi-agent predator-prey simulations. We investigate how strategies change in a three-species system where each species (rock, paper, and scissors) influences the other. To comprehend cyclic dominant environments using reinforcement learning, we use MARL to investigate how these three species coexist and modify their tactics over time.

## 2. Related Works

### 2.1. Predator Prey Games

Agents learn optimal behaviour through interactions with their surroundings in reinforcement learning (RL) (Kaelbling et al., 1996; Wang et al., 2022b). RL has affected several fields, such as autonomous driving (Zhu & Zhao, 2021) and protein folding (Jumper et al., 2021), by incorporating deep neural networks as function approximations. Evolutionary games with predator-prey models is a well-studied problem at the nexus of biological ecosystems and reinforcement learning (Polis et al., 1989). The field is particularly interested in how strategies and population change over time because it models a variety of real-world settings (Abrams, 2000).

### 2.2. multi agent Reinforcement Learning

A single policy governs the actions of numerous agents in multi-agent reinforcement learning, an advancement of reinforcement learning (Busoniu et al., 2008). This makes it easier to convert individual strategies into coordinated and targetted ones (Tan & Li, 2021), which results in an intriguing examination of how strategies change over time for various games (Yang & Wang, 2020). In order to comprehend the evolution of strategies using multi agent reinforcement learning (MARL), we examine the three species cyclic predator-prey model (Wang et al., 2022a).

## 3. Methods: Building our Arena

Three species inhabit our arena: Rocks (R), Papers (P), and Scissors (S). As always, R defeats S, S defeats P, and P defeats R. In our setting, when two different entities come in contact with each other (occupy the same cell), the losing entity dies and the winning entity replaces it. This changes the population of the colliding species. For example: if R comes in contact with P, there will be no R left and in its place a new P. The population of R will reduce by 1 and the population of P will increase by 1. The goal of each entity is to increase its population to gain rewards. An important thing to note is that the total population of all 3 species remains constant with time. This setup has some excellent properties:

- R needs to know not to wipe out the S population. If it does and P is not extinct, R will eventually fall to P and go extinct. This tells us that R will need to learn to understand long term gains compared to short term ones, i.e., wait for S to kill all the P and then start its attack.
- If R does learn to wipe out S, it will also need to learn to dodge P. In this case, will the policy controlling R fall into the local optima of capturing all S subsequently learning to run away is an interesting question!
- If all the entities learn at the same time, the optimal scenario will result in a Nash Equilibrium where each entity is running to capture its prey while simultaneously running from its predator.

It is interesting to investigate the evolution of strategies (thereby understand learning) across time. We split our work into 3 sub problems: analysis with • 1 R, 1 P (simple predator prey scenario) • $m$ R, and $n$ P (introduces capture) • 1 R, 1 P, 1 S (having 3 species)

Each agent will be allowed 5 possible actions {“up”, “down”, “left”, “right”, “stay”}. One policy controls all the agents of a species: $\Pi_R, \Pi_P, \Pi_S$. Each policy will take in the entire information of all agents’ positions, along with relative distances, and return the actions of each agent it controls.

### 3.1. The Problem of Changing Population

The number of agents under the policy’s control does not change over time in a typical multi agent setup. New agents are not typically added or removed. Since our configuration calls for a dynamically fluctuating number of entities, we must adapt our neural network to act accordingly.

### 3.2. Method on Learning Optimal Policy

The grid’s state representation differs from the traditional one, which uses an input dimension of $G \times G$. Rather, we use an ID for each type of agent and the location of agents to represent our grid. This facilitates individual learning, which will be discussed later in this section, and lowers the neural network’s complexity and trainable hyperparameters. We employ the Deep Q-network (DQN) (Mnih, 2013) to handle the intricacies of this multi agent setting. All of the agents in a given species are controlled by a single DQN. To let the network know which agent we are looking for, it takes in the positions of agents, a species identification number, and a marker. For instance, if we want to take action for the 2nd S in a scenario with 1 R, 1 P, and 2 S, we would send the input as follows: $(1, x_{R_1}, y_{R_1}, 2, x_{P_1}, 3, x_{S_1}, y_{S_1}, 4, x_{S_2}, y_{S_2})$. In this case, the marker is 4, and the IDs of R, P, and S are 1, 2, and 3, respectively. Simple multi-layer perceptrons (MLPs) with ReLU activations make up the network, along with an output layer that displays the Q-values for the five potential actions.

Experiences are stored as tuples $(s, a, r, s', d)$, which represent the state before action, the action taken, the reward received, the state after action, and a done flag indicating whether the episode has ended. Each agent type keeps its own experience replay buffer (Zhang & Sutton, 2017). To stabilize learning, a distinct target network is also kept up to date for every type of agent (Fellows et al., 2023). This target network is updated from the weights of the main network on a regular basis.

The learning process entails implementing an $\epsilon$-greedy strategy to strike a balance between exploration and exploitation. At each instance, agents choose actions based on their present situation, and the environment alters the state due to all agents’ actions. Agents update their DQNs during training by periodically sampling mini-batches and storing their experiences in their respective replay buffers. The mean squared error between the target and current Q-values is used to calculate the loss. The Bellman equation is used to determine the target Q-values.

$$Q_{\text{target}} = r + \gamma \max_{a'} Q_{\text{target}}(s', a')$$

where $r$ is the reward received, $\gamma$ is the discount factor, and $Q_{\text{target}}(s', a')$ is the target network’s estimate of the future rewards from the next state $s'$. The exploration rate $\epsilon$ is decayed over time for each agent type to reduce exploration as training progresses, allowing agents to increasingly exploit learned policies. Target networks are also updated periodically to improve learning stability by providing consistent targets during Q-value updates.

### 3.3. What we bring to the table!

Our approach differs significantly from the traditional single-predator single-prey model. In the conventional model, there are only two agents: a single predator and a single prey with straightforward interactions. The predator aims solely to catch the prey, while the prey’s objective is to evade capture. Learning typically involves training a single DQN for the predator, with the prey following a fixed or random policy, resulting in a relatively stable and stationary environment.

In contrast, our approach features multiple agents of different species, leading to complex interactions and emergent behaviours. Agents have dual roles as both predators and prey, necessitating more sophisticated strategies that balance offensive and defensive. The environment becomes highly non-stationary from any single agent's perspective because other agents are also learning and adapting their policies simultaneously. This non-stationarity poses significant challenges for learning algorithms, as the optimal policy for an agent may change as other agents' policies evolve.

Moreover, while the traditional model's state representation often includes only the positions of the predator and prey, and the whole flattened grid, our method uses positions and indexes of all agents, the unique flagged index, and relative distances between agents, providing agents with comprehensive environmental information. This full observability is crucial in multi agent settings, as agents need to be aware of both threats and opportunities in their surroundings to make optimal decisions. The increased complexity in state representation and the necessity for agents to process more information require more sophisticated neural network architectures and training procedures.

Our novel approach introduces a scalable and flexible framework for the simulating arena battles in multi agent reinforcement learning. It enhances realism by simulating environments closer to real-world scenarios where multiple agents interact simultaneously, such as robotic swarms and ecological systems. These complexities of agent interactions in our environment encourages the development of advanced tactics.

### 3.4. Training Environment Setup

We can change important parameters for defining the simulation setup, reward system, and training hyperparameter in our prey-predator environment. Customization of the grid size, agent types, and agent-to-agent interaction rules is possible with our environment setup. For behaviours like winning, losing, moving, catching prey, and interacting with boundaries, the reward system offers flexibility in defining penalties and incentives. The model used is specified by the training hyperparameter, e.g. settings for experience replay and batch size, as well as the neural network architecture, maximum steps, and number of episodes (Deep Q-Network). When combined, these parameters allow for fine-grained control over the learning and simulation processes. See Appendix A for a detailed explanation.

## 4. Analysis of Single Predator Single Prey

### 4.1. Experiment 1

The rewards are set up as: $win = 1, loss = -1, predator\_step = 0, prey\_step = 0.2, distance = -1, boundary = -0.1$. The training is done for 50000 episodes with a network of size 64, 128, 64. The learning process is shown in Figure 1. We see an interesting case where the predator and prey are learning simultaneously.

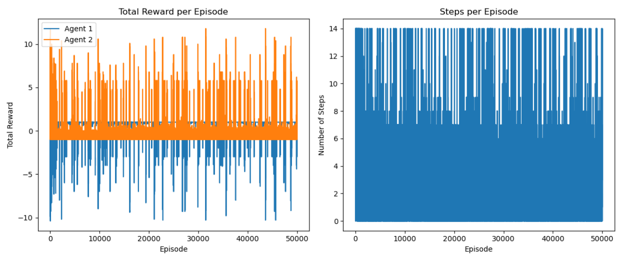

*Figure 1.* The rewards over each episode is shown. Predator is marked by Agent 1 and prey, Agent 2.

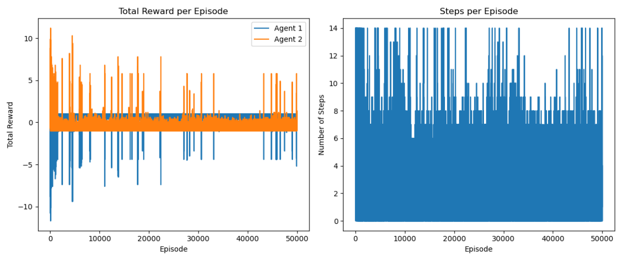

*Figure 2.* The rewards over each episode is shown. Predator is marked by Agent 1 and prey, Agent 2. Note how learning is sparse when compared to Figure 1

### 4.2. Experiment 2

With $predator\_step = -0.1$, we employ the same reward values as those listed in Section 4.1. We observe that the predator learns much more quickly and can outperform the prey. A small number of spikes show that the prey is learning. Nevertheless, Figure 2 has fewer crests and troughs than Figure 1. Our experiments led us to the conclusion that punishing the predator for existing greatly increases the likelihood that it will catch the prey. It can accomplish this in a relatively small number of episodes and quickly adjust the approach.

### 4.3. The changes in strategy

The crests and troughs, intuitively, represent shifts in the agents' strategies. At first, the predator learns that it must capture the prey, while the prey learns little because it always prevails. The prey learns over the course of the following episodes that it must get away from the predator. Subsequently, we observe the strategies and witness an intriguing sequence of crests and troughs. The majority of strategies lack intuitiveness and are difficult to explain. A more complex evolution of strategies results from the prey realizing that it can continuously move away from the predator and escape around the $\sim 30000$ episode count, while the predator tries to corner the prey and anticipates where the prey will go next, then moves there. We can clearly see the predator pursuing the prey in Figures 7 and 8, which displays the states at each time step for episode number 15750 and 33750.

## 5. Analysis of Multi Predator Single Prey

By altering certain parameters in our configuration, RL BattleGround extends to the multi predator single prey scenario. This can be thought of as having multiple P and a single R, multiple R and a single S, or multiple S and a single P. In this empirical trial, we focus on 2 predators and 1 prey, adaptable to all combinations of RPS. We modified the number of agents and ran similar experiments as described in section 4.1. Here, we again explore how policy learning for the predators and prey changes with varying hyper parameter values and reward structures.

### 5.1. Experiment 1

The rewards are set up as: $win = 1, loss = -1, step = -0.1, prey\_step = 0.1, distance = -0.1, boundary = -0.1$. The training is conducted over 20,000 episodes using a network architecture of 64, 128, 256, 128, 64. The learning process is shown in Figure 3. We notice the predator and prey learning simultaneously.

In Figure 9, we observe an emergent learning pattern among the predators that we will continue to explore. They form a "group-up" strategy where they organize themselves into a wall-like structure to corner the prey. This behavior suggests that predators are learning to coordinate spatially, likely due to shared policies and consistent rewards.

Even though we can see that the prey is trying to escape the predators by moving away from them, it does not appear to have learned an optimal escape policy. From multiple observations of the roll-out of each episode, we notice a rather reactive movement from the prey, rather than a strategic one. This can be due to (a) *grid size limitations:* The fixed $5 \times 5$ grid reduces the prey's ability to escape effectively, restricting potential strategies, and (b) *shared policy constraints:* The prey may struggle because predators leverage group strategies while the prey operates independently.

Furthermore, as shown in Figure 10, predators appear to behave identically rather than collaboratively. This observation raises an important question: *is the observed "group behavior" true collaboration or an artifact of agents following identical policies?* In future experiments, we suggest introducing heterogeneous policies to test whether agents can learn distinct yet complementary roles.

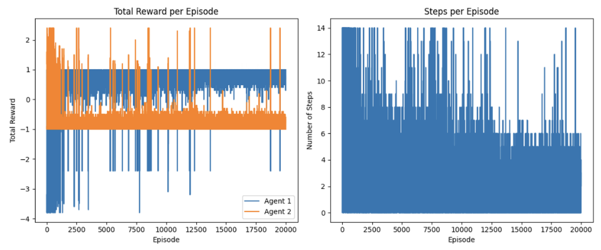

*Figure 3.* The rewards over each episode is shown. Predator is marked by Agent 1 and prey, Agent 2.

When we extended the duration of the training by doubling the episodes, the outcomes remained largely similar. However, in Figure 11, we observe predators beginning to split their formation. This adjustment increases their spatial coverage and enables them to trap the prey more efficiently. Furthermore, this contradicts the idea of predators acting identically, and instead learning to use other agents of its type to capture the prey.

### 5.2. Experiment 2

In this experiment, we maintain a similar reward structure but adjust the distance penalty $distance\_step$ to $-0.3$ and reduce the dimensions of the hidden layer to three layers with deeper neurons: 128, 256, 128. Increasing the magnitude of $distance\_step$ imposes a stronger penalty on predators for being farther from the prey and on the prey for proximity to predators.

The results reveal the following.

**Predator Behavior:** The predators consistently display coordinated movements to corner the prey. The "wall" structure observed earlier persists, suggesting that the increased distance penalty reinforces the need for proximity to the prey.

**Prey Behavior:** Even though the prey shows slight improvements in evading the predators, it remains reactive and suboptimal. From Figure 12, we see that the prey initially attempts to move downward to evade the predators, realizes its proximity to danger, and reverses direction. However, by that point, it is too late as the second predator closes in from the opposite direction, demonstrating the prey's reactive rather than strategic policy and its inability to anticipate strategic movement from the predators.

### 5.3. Strategies in Multi Predator Single Prey

From the empirical trials, several key patterns emerged:

1. **Predator Learning Dynamics:** Predators consistently evolve group-based strategies such as "wall" and "cor-

nering" tactics to trap the prey efficiently. In Figure 11, we observe predators dynamically adjusting their positions to maximize spatial coverage, indicating coordination.

2. **Prey Learning Limitations:** The prey demonstrates slower learning and suboptimal strategies, as observed in frequent reactive movements and occasional failure to move at times.

## 6. Analysis of Single Predator Multi Prey

To illustrate how a predator can most efficiently catch as many prey as possible in the fewest steps, it is also crucial to simulate how a single predator tracks and hunts multiple prey. Here, we present a number of experiments with the same reward structure as Section 4.1.

### 6.1. Experiment 1

Using the same multi predator single prey experiment setup, we run the train over 15,000 episodes with a DQN hidden dimensions of 64, 128, 64. Rewards for both agents is shown in Figure 4. Notice both from Figures 4 and 13, the predator seem to be inactive despite proximity to the prey. On the other hand, the prey learns how to move away from the inactive predator in a single direction.

After analysis, the primary causes of the predator's "dormant" behaviour could be: 1. *Oversimplified Model:* In a multi agent setting, the DQN is too basic to assist the predator in learning any chasing behaviour. Although, the same network showed learning in the single predator single prey and multi predator single prey setting. 2. *Target Network Update Frequency:* The DQN target model may be updated too infrequently per episode for the model to learn. 3. *The network:* The hyperparameter that were chosen may not be optimal. A larger model and additional hyper parameter tuning may facilitate learning. We determine that the primary cause of our issue is the target network's update frequency after investigating each of these elements using various trails. Better learning is demonstrated in the following experiment when the target model update is done every three steps rather than every episode.

### 6.2. Experiment 2

The reward structure from Section 6.1 is applied after the target model update frequency has been changed. 10,000 episodes of training are conducted using the same hidden dimension size. Figure 5 displays the steps and rewards plotted against the number of episodes. In contrast to the flat line seen in the majority of experiment 1's episodes (Figure 4), the graph (Figure 5) clearly shows that the predator and prey are acting against the other with alternating wins and losses. Combined with the animation, the results reveal the

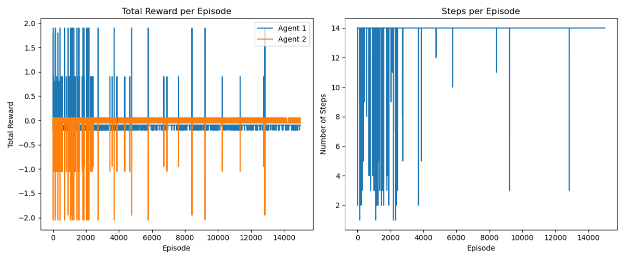

*Figure 4.* The rewards over each episode is shown. Predator is marked by Agent 1 and prey, Agent 2. Note that the learning stagnates after 4000 episodes. In depth analysis of this shows a suboptimal policy from the predator network.

following.

**Predator Behaviour:** As can be seen from the animation, the predator continuously exhibits improved hunting behaviour in subsequent training episodes; however, it seems to assume that prey are near one another. Thus, despite using the step penalty, it lingers around attempting to sniff the other prey after successfully hunting the first one (Figures 14 and 15). This demonstrates an intriguing fact: compared to directly pursuing the prey, using a neighbour-searching approach to look for possible prey actually yields greater rewards.

**Prey Behaviour:** The prey doesn't show optimal learning due to similar reasons mentioned in the previous section. In later episodes, the prey monotonously move toward the same edge without escape when predator comes close. This might due to their early learning that moving to the edge will survive when predator has not be able to acquire any learning.

### 6.3. Strategies in Single Predator Multi Prey

From our experiments, these key features emerged:

1. **Predator Learning Dynamics:** Predators expect their prey to be nearby, and when they are, they hunt with great skill. As the animation illustrates, it also demonstrates the ability to hunt the prey with a greater degree of freedom (on edge rather than in corner).

2. **Prey Learning Limitations:** As evidenced by their unidirectional movement to the edge and inability to evade the predator's pursuit, prey exhibit limited learning.

## 7. Analysis of a 3 species system

Finally, we introduce our work on the 3 species system: 1 R, 1 P, 1 S. Each agent is simultaneously both: a predator and a prey. The complexities of this system arise in multiple

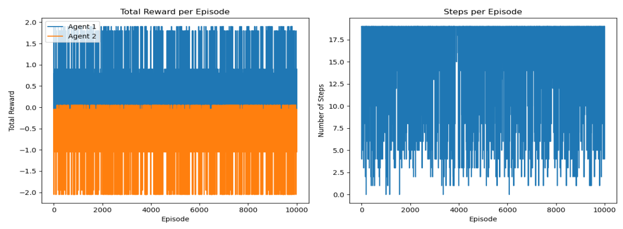

*Figure 5.* The rewards over each episode is shown. Predator is marked by Agent 1 and prey, Agent 2. More fluctuations indicated better learning, unlike Figure 4.

real world situations: ecological balances in the jungle, democracies and coalitions in politics, countries forming allies in wars, and many more.

The reward values are the same as Section 4.1 and an episode count of 50000. From Figure 6, we see that the learning is skewed. Agents 1 and 2 perform much better than Agent 3 throughout. Ideally, one would expect the learning process to be symmetric, as the problem is same from each agent's perspective. However, all the agents learn a simple strategy of choosing a particular direction and running away from its predator (see Figure 16). Although we do expect emergence of complex strategies as mentioned in Section 3, a larger DQN with hyper parameter tuning is required to visualize it.

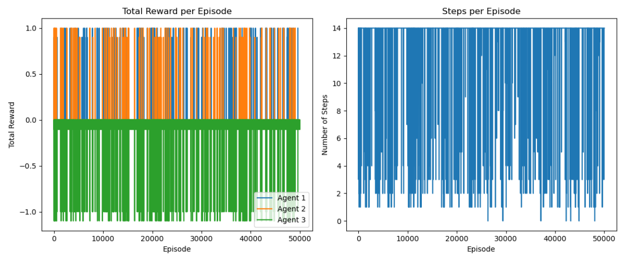

*Figure 6.* The rewards over each episode is shown. Every agent is simultaneously a predator and a prey. Agent 1 and 2 perform much better than Agent 3.

## 8. Limitations

Although change in population per species is modelled in RL BattleGround, most of our experiments are not complex enough to emulate that. While we did expect to see evolved strategies in the 3 species setting, the agents resolved to a simpler strategy of evasion without hunting. Given more compute resources, we aim to simulate these experiments till an optimal policy is attained and the best strategies which show the Nash Equilibrium is achieved.

## 9. Discussions and Future Work

While originally we were aiming to observe emergent and complex behaviours in a multi-species multi agent setting, we show that there is no simple and straight forward answer. Through hundreds of experiments, we analyse different reward incentives, hyper parameter tuning, and various other parameters built in RL BattleGround such as grid size, experience replay, and agent interaction mechanisms. Given more compute resources and compute time, we believe that with a deeper network we can observe evolution of strategies in the single predator multi prey scenario as well as the multi-species multi agent setting. The library we introduce allows for huge flexibility with ease and for complex and controlled experiments.

A situation in which several predators vie for the same prey species is quite feasible. By extending *Rock-Paper-Scissors* to *Rock-Paper-Scissors-Lizard-Spock* with a five species interaction on larger grids with more agents, we hope to analyse more intricate interactions. Additionally, we want to expand RL BattleGround to support various DQN learning methods and boost our computational capacity during the demanding experiments.

Overall, we present a predator-prey environment with intricate relationships and interactions between multiple species. Our setup more accurately replicates real-world interactions than previous studies, producing a variety of complex strategies that may find use in practical settings. With this work, we hope to improve modelling of the tangled interplay that is common in today's world.

## Acknowledgements

We thank Prof. Chong Li and Prof. Chonggang Wang, whose discussions and insightful observations were invaluable during this research.

## References

Abrams, P. A. The evolution of predator-prey interactions: theory and evidence. *Annual Review of Ecology and Systematics*, 31(1):79–105, 2000.

Berryman, A. A. The orgins and evolution of predator-prey theory. *Ecology*, 73(5):1530–1535, 1992.

Busoniu, L., Babuska, R., and De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

Fellows, M., Smith, M. J., and Whiteson, S. Why target networks stabilise temporal difference methods. In *International Conference on Machine Learning*, pp. 9886–9909. PMLR, 2023.

Hofbauer, J. and Sigmund, K. *Evolutionary games and population dynamics*. Cambridge university press, 1998.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Kaelbling, L. P., Littman, M. L., and Moore, A. W. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Mnih, V. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Polis, G. A., Myers, C. A., and Holt, R. D. The ecology and evolution of intraguild predation: potential competitors that eat each other. *Annual review of ecology and systematics*, pp. 297–330, 1989.

Tan, Z. and Li, K. Differential evolution with mixed mutation strategy based on deep reinforcement learning. *Applied Soft Computing*, 111:107678, 2021.

Wang, X., Lu, Y., Shi, L., and Park, J. The effect of territorial awareness in a three-species cyclic predator–prey model. *Scientific Reports*, 12(1):1821, 2022a.

Wang, X., Wang, S., Liang, X., Zhao, D., Huang, J., Xu, X., Dai, B., and Miao, Q. Deep reinforcement learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):5064–5078, 2022b.

Wangersky, P. J. Lotka-volterra population models. *Annual Review of Ecology and Systematics*, 9:189–218, 1978.

Yamada, J., Shawe-Taylor, J., and Fountas, Z. Evolution of a complex predator-prey ecosystem on large-scale multi-agent deep reinforcement learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2020.

Yang, Y. and Wang, J. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.

Yang, Y., Yu, L., Bai, Y., Wang, J., Zhang, W., Wen, Y., and Yu, Y. A study of ai population dynamics with million-agent reinforcement learning. *arXiv preprint arXiv:1709.04511*, 2017.

Zhang, S. and Sutton, R. S. A deeper look at experience replay. *arXiv preprint arXiv:1712.01275*, 2017.

Zhu, Z. and Zhao, H. A survey of deep rl and il for autonomous driving policy learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14043–14065, 2021.

## Contributions

### Aditya Somasundaram

Introduced the novelty in this work after an extensive literature review and discussions with Prof. Chong Li and Prof. Chonggang Wang. As advised by Prof. Li, I initially built a Monte Carlo learning setup without neural networks on a small scale to show proof of concept. Created the skeleton version code to showcase learning using DQNs. Conducted over a 100 single prey single predator experiments, tweaking and guiding the parameters which were used in subsequent sections. Also conducted experiments on the 3 species setup. Contributed and collaborated on Sections 1, 2, and 3. Independently authored Sections 4, 7, 8, and 9.

### Chiara von Gerlach

Contributed to Sections 1, 2, and 3 by researching the relevant work that directed our research and organizing the write-up for our research paper. I conducted single prey single predator experiments to reveal optimal reward structures for agents to learn and scaled to multi predator single prey experiments. Thoroughly researched optimal reward function for multi predator single prey and resulting experimentation to lead to thoughtful analysis. Additionally, I wrote Section 5 where I highlight how my reward functions lead to significant learning and joint coordination in predator behaviour.

### Likhith Ayinala

Built the library that we use to run experiments from scratch to accommodate our unique experiment. This includes the environment scripts, training scripts and the model scripts. Additionally, I regularly updated the code to fix any new bugs that emerged during the experiments, and implemented any new features that were added to the experiment as the research demanded. Conducted a few experiments for single predator single prey to determine some optimal values.

### Leoric Luo

Assisted in the development and debugging of the model, including regular evaluation of updated code, performing trial tests, and identifying potential issues. Conducted experiments involving single predator single prey interactions as well as single predator multi prey dynamics, with the results documented in Section 6. Additionally contributed to the paper by authoring the Methods section.

## A. Experiment Setup

We can conduct a variety of experiments with our new library with ease. A thorough configuration that includes settings for the simulation environment, agent behaviours, reward systems, and training procedures defines the experimental setup for the prey-predator environment.

### A.1. Values used in the DQN

The DQN has the following values throughout our analysis. While we did vary these during our experiments, the following were found to be a good selection:

$$\gamma = 0.99, \epsilon_{\text{init}} = 1.0, \epsilon_{\text{decay}} = 0.995, \epsilon_{\text{min}} = 0.01$$

The length of the memory buffer was set to be 1000. Learning using the DQN involved the Adam optimizer with a learning rate of 0.005 with the Mean Square Error (MSE) loss.

### A.2. Environment Parameters

- **Grid Size:** Specifies the dimensions of the simulation grid.
- **Agents:** Defines the number and types of agents participating in the environment.
- **Prey-Predator Interaction Rules:** Determines which agent types can capture or be captured by others.
- **Capture Toggle:** Enables or disables the capture and conversion of agents.

### A.3. Reward System

- **Win/Lose Rewards:** Defines the incentives or penalties for achieving the end goal of the simulation.
- **Step-Based Penalties:** Includes penalties for each movement step taken by the agents or prey.
- **Distance Penalty:** Specifies penalties based on the distance between agents to encourage strategic positioning.
- **Boundary Penalty:** Penalizes agents for hitting the grid boundary.
- **Capture/Elimination Rewards:** Configures the rewards or penalties for capturing prey or being eliminated.

### A.4. Training hyperparameter

- **Number of Actions:** If set to 4, the the action "stay" is disregarded.
- **Episodes and Steps:** Defines the total number of episodes and the maximum steps allowed per episode.
- **Model Type:** Specifies the learning model used for training agents (e.g., reinforcement learning algorithms).
- **Hidden Layer Dimensions:** Describes the architecture of the neural network used in the model.
- **Experience Replay:** Toggles whether past experiences are stored and replayed during training.
- **Batch Size:** Determines the batch size for training.

### A.5. Position and Distance Settings

- **Random Positioning:** Configures whether agents start at random positions.
- **Distance Input:** Indicates whether the distance between agents is used as input.
- **Distance Type:** Specifies how distance-related data is used in the loss function(e.g., average or last position). Setting this to average would lead to the averaging of distance between set of agents over the entire epoch, whereas setting it to last would lead to using the distances between the set of agents at the last step.

This configuration enables a high degree of flexibility for studying agent interactions, behaviour, and learning strategies within the prey-predator environment.

## B. Animations

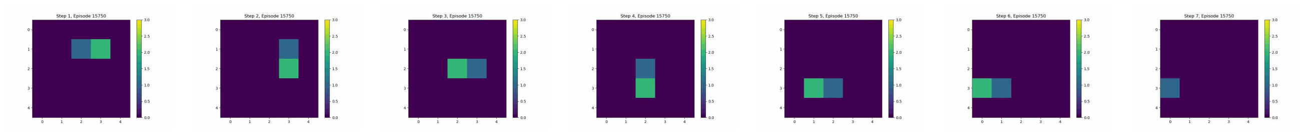

*Figure 7.* Animation for a single predator single prey without penalizing the predator at episode 15750.

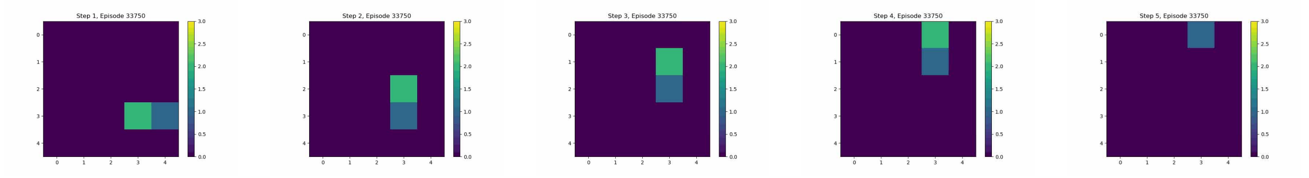

*Figure 8.* Animation for a single predator single prey with penalizing the predator at episode 33750.

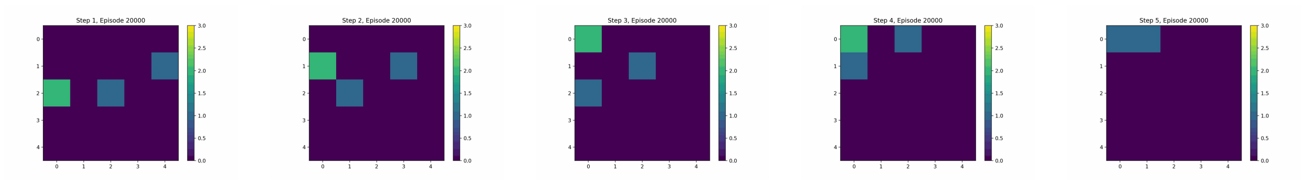

*Figure 9.* Animation for a multi predator single prey without penalizing the predator at episode 20000. Notice how the predators team up and catch the prey in the corner.

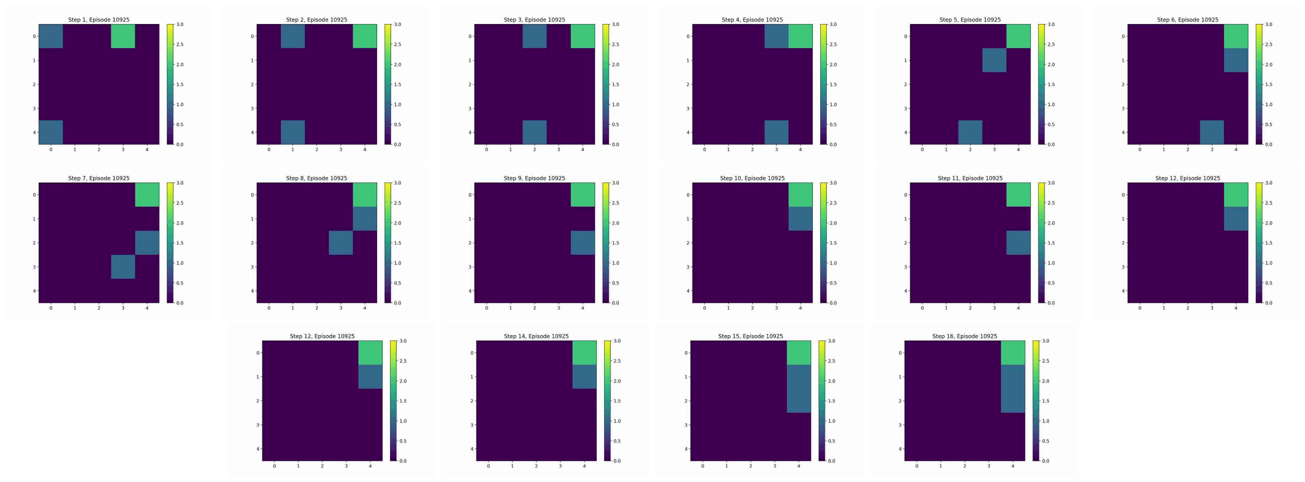

*Figure 10.* Animation for a multi predator single prey without penalizing the predator at episode 10925. Notice how the predators struggle to catch the prey.

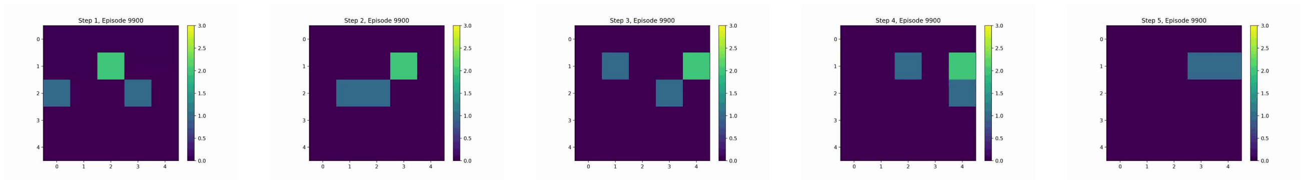

*Figure 11.* Animation for a multi predator single prey without penalizing the predator at episode 9900.

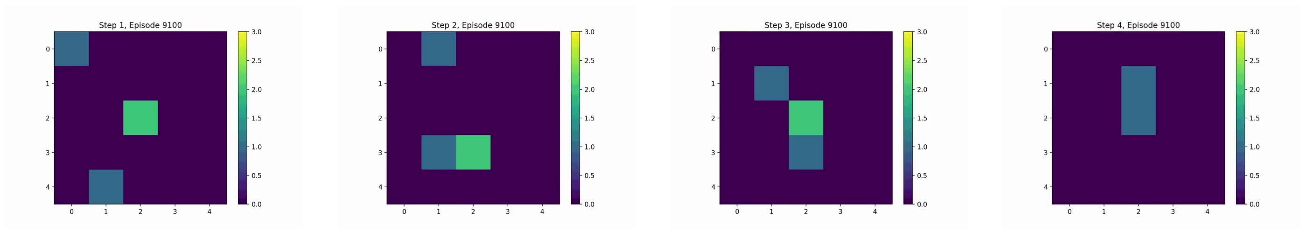

*Figure 12.* Animation for a multi predator single prey with penalizing the predator at episode 9100.

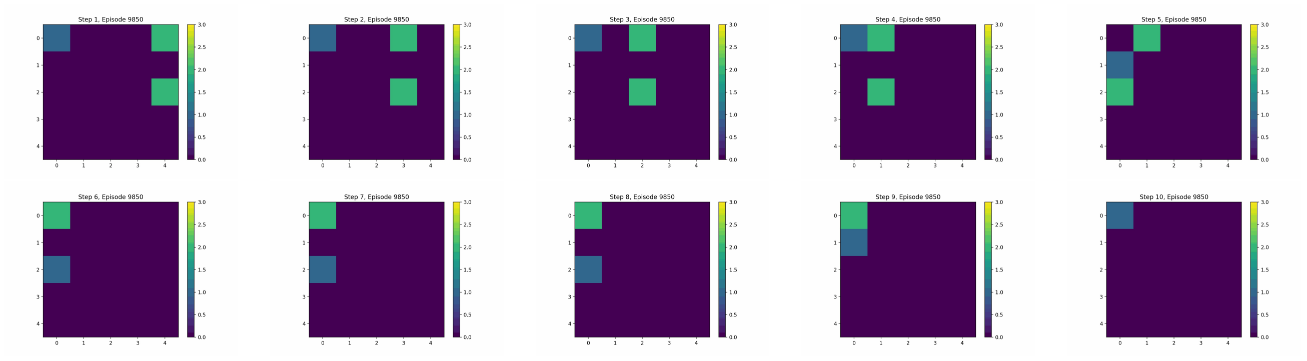

*Figure 13.* Animation for a single predator multi prey with faster target network update, episode 9850.

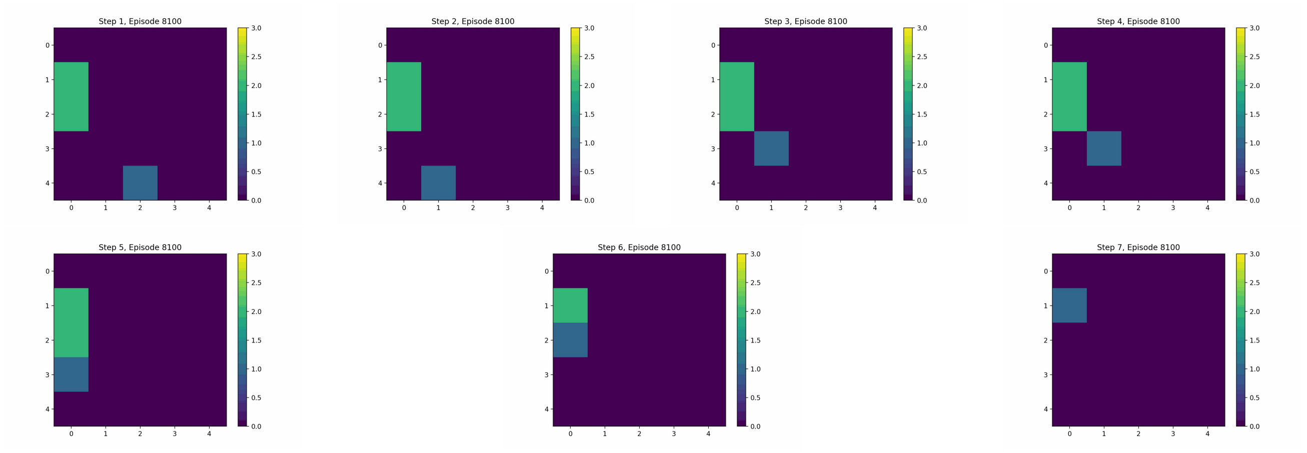

*Figure 14.* Animation for a single predator multi prey with faster target network update showing better performance when prey are close to each other at episode 8100

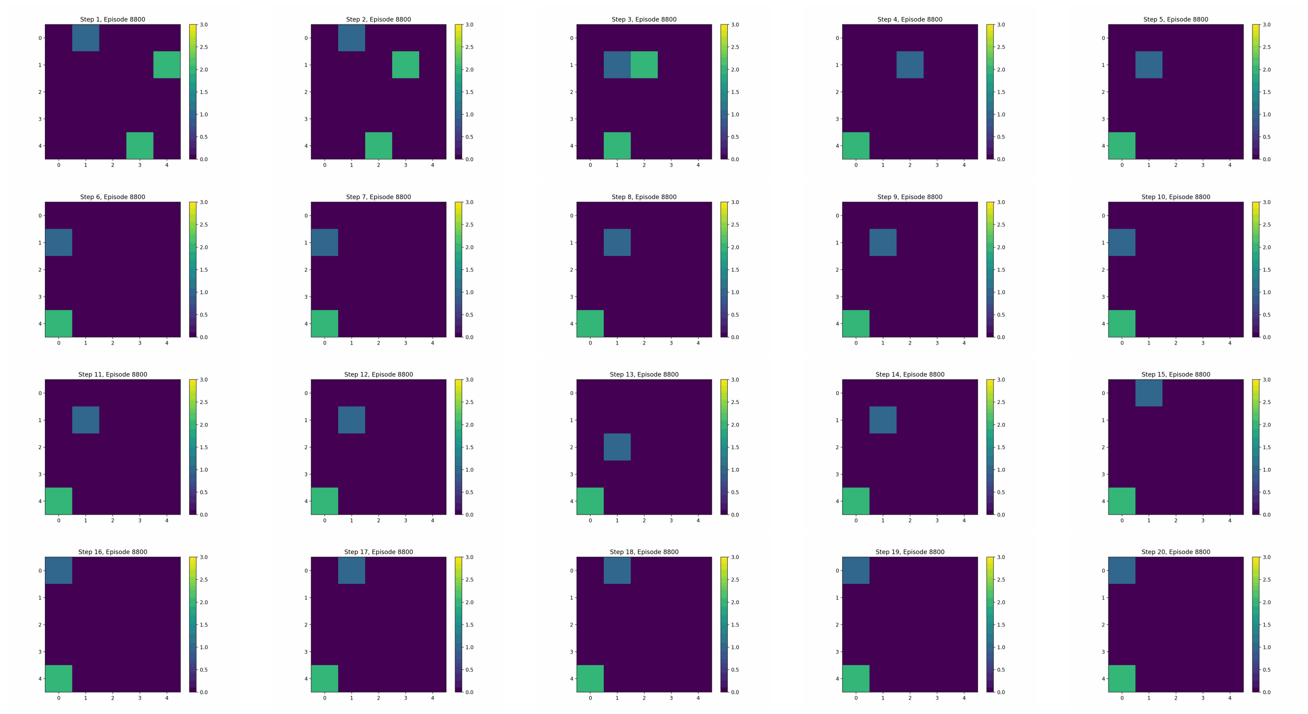

*Figure 15.* Animation for a single predator multi prey with faster target network update, showing the predator's lingering effect around captured prey.

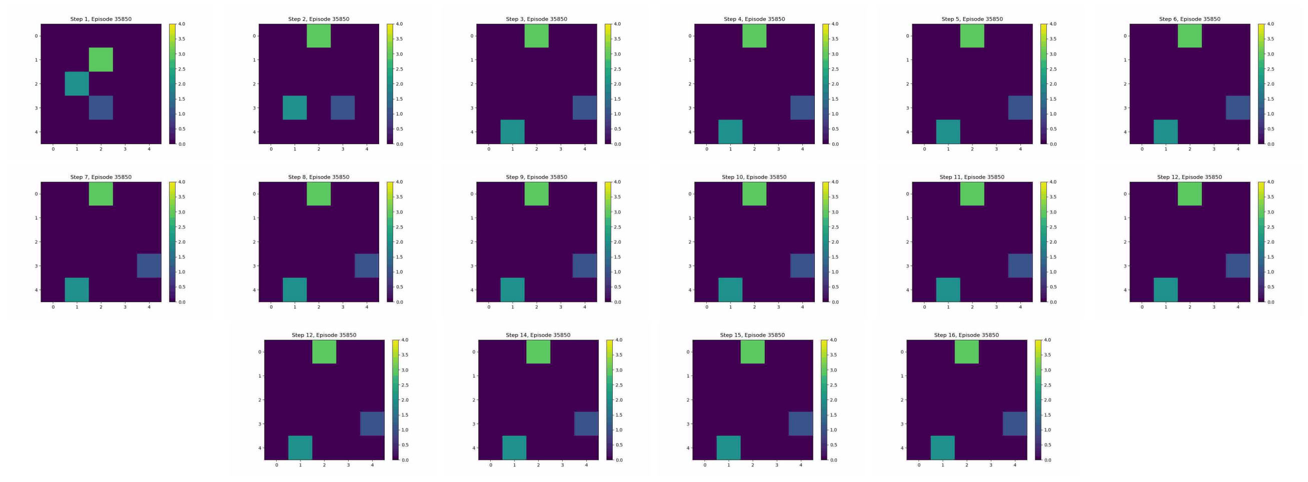

*Figure 16.* Animation for a multi-species predator prey scenario. Notice how each agent prioritizes running away in a certain direction over captures.