Multi-Agent Exploration with Random Network Distillation

Philip Raeisghasem

Abstract—A significant part of reinforcement learning is ensuring adequate exploration. An agent cannot learn how good a new strategy is if the strategy is never tried. For multi-agent tasks, the space of possible configurations grows exponentially with the number of agents. This larger space is much harder to explore. In this paper, we apply the recently proposed Random Network Distillation exploration method to a multi-agent setting. We compare this approach to other exploration methods in the multi-agent setting. We do this both by explicitly quantifying amount of exploration and by analyzing downstream performance.

Index Terms—reinforcement learning, multi-agent, exploration, random network distillation.

1 Introduction

Reinforcement learning is a paradigm of machine learning alongside supervised and unsupervised learning. It is most applicable to problems that can be framed as an agent acting within an environment to achieve some goal. Like supervised learning, reinforcement learning happens through a feedback mechanism during an initial training phase. Unlike in supervised learning, this feedback is not in the form of 'labeled' data. There are no simple 'correct' answers to directly compare the model's performance against. Instead, the feedback loop in reinforcement learning is between the agent and the environment by way of a reward function. Broadly speaking, instead of learning whether its actions are 'right' or 'wrong', the agent learns how good its actions are.

This difference in feedback mechanism is important for allowing reinforcement learning models to learn from self-generated experience. Thanks to the advent deep neural networks, the space of problems solvable by artificial intelligence and machine learning has exploded in size. Many of these problems, however, require much more data to solve. In supervised learning, vast datasets that require significant preprocessing are the norm, and a human is needed to label every single training example. In contrast, a human often need only specify the reward function for a given environment in reinforcement learning.

While for many problems it can be challenging to design a reward function that will lead to desired behavior, a large class of problems are amenable to very simple and sparse reward functions. If the problem is a two-player game, for example, the agent could only be rewarded at the very end, depending on whether it won the game. Using these sparse reward functions comes at the cost of the agent receiving less feedback. That is, there is a tradeoff between human effort embedding prior knowledge and difficulty. This tradeoff is not unique to reinforcement learning, but it has risen to unique methods in reinforcement learning for addressing the issue.

In particular, using a sparse reward function exacerbates the existing issue of exploration. While learning, an agent must always be deciding how confident it is in its current strategy. If it is very confident in the information it has received so far, it will choose to act in ways that have given it large rewards in the past. The agent is said to be exploiting its current knowledge. If the agent is not very confident, it may decide to try something new in order to gather information. The agent is then said to be exploring the environment. A sparse reward function lessens the effectiveness of exploration by limiting the number of times an agent is given any feedback after exploratory actions.

The difficulty of exploration depends on more

• *Philip Raeisghasem is with Louisiana Tech University, Department of Cyberspace Engineering.*
E-mail: par019@latech.edu

than just the sparsity of the reward function, of course. Other important factors are the size and complexity of the environment as well as the number of agents acting concurrently. This last factor, the number of agents, is especially impactful. An environment's reward is a function of the states occupied and the actions taken at any given time step. If there are $|S|$ states and $|A|$ actions available in an environment with n agents, then the input space of the reward function is of size $|S|^n|A|^n$. For problems with large state or action spaces, exploration in a multi-agent setting quickly becomes a daunting task.

In this paper, we investigate the application of exploration by random network distillation (RND) as proposed by Burda et al to the multi-agent setting. This method is one way of encouraging an agent to seek out states that are "surprising". Specifically, an intrinsic reward bonus is given to the agent for visiting states that it is not able to predict. This intrinsic reward, when combined with the (potentially sparse) extrinsic environment reward, has been shown to result in a more thorough exploration of the environment by a single agent.

We analyze the simulated data generated by multiple agents concurrently exploring with this method. The environment we test with is a simple 2D environment with obstacles and consumable objects. The metrics we gather include the total percentage of the state-action space visited over time as well as the total reward received over time. We also collect these metrics for other exploration methods for the sake of comparison.