Learning to Cooperate with Unseen Agents Through Meta-Reinforcement Learning

Rujikorn Chararkorn, Poramate Manoonpong, Nat Dilokthanakul
Vidyasirimedhi Institute of Science and Technology (VISTEC)

Introduction

In this work, we posit that a cooperative agent needs the ability to ***adapt*** its policy to the dynamics of ***unseen agents*** and this can be achieved *with* ***meta-reinforcement learning***. Meta-learning paradigm considers learning an adaptive behaviour using data.

Method

We consider two-player cooperative tasks, where our agent will have to cooperate with a set of agents from a pool, ***P***, of partner agents. Similar to previous work our meta-RL agent is implemented with an RNN and trained with a distribution of partners ***P***. We use two cooperative environments with different cooperative circumstances: lever game and speaker-listener. The objective of the meta-RL agent is to maximise the expected return $E_{p\sim \mathbf{P},\tau\sim\pi}[\sum_{t=0}^{H}\gamma^t r_t]$.

Neural network models

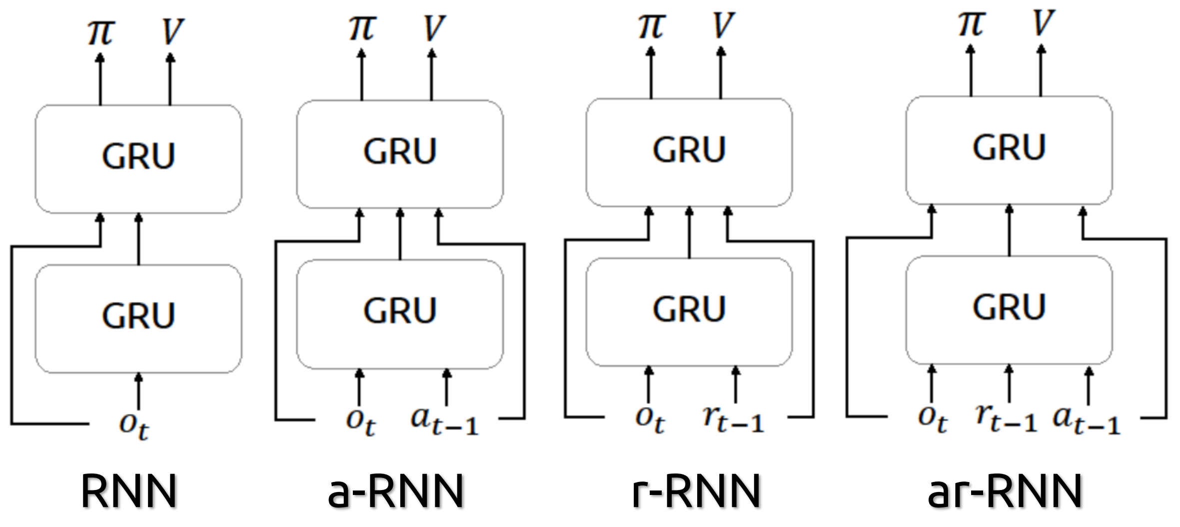

Figure 1: Architecture variations. Four variations of architectures considered in this work. Each architecture has different input features. These architectures are tested in the experiments to study the impact of each component of meta-RL.

Experimental results

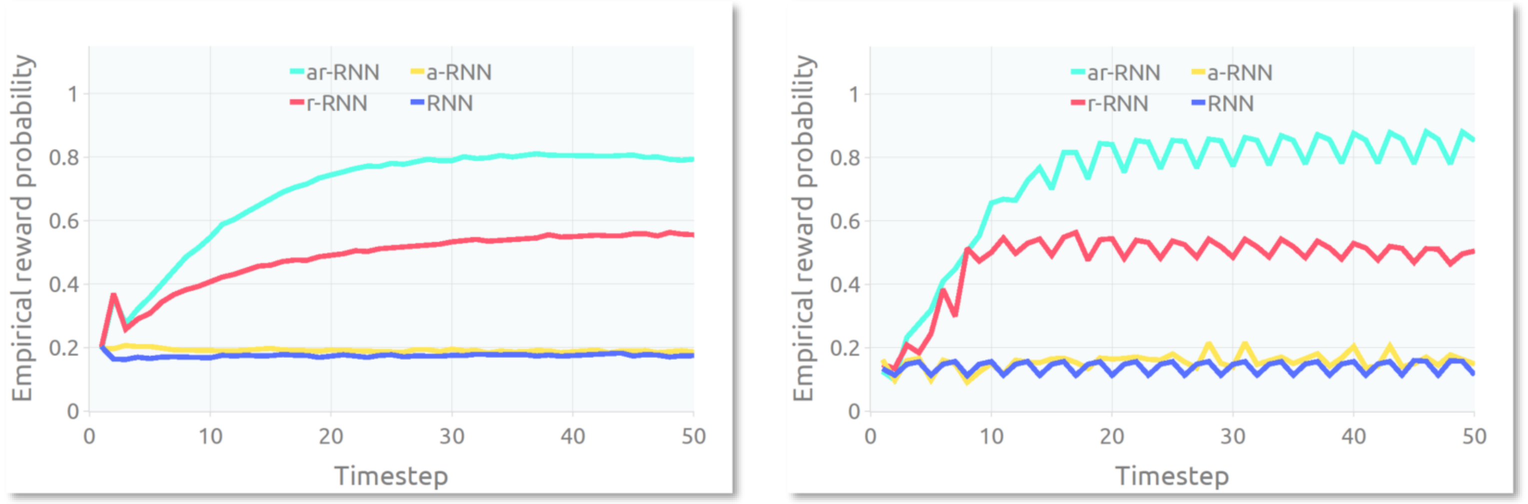

Figure 2: Chance of getting a reward within a trajectory. The graph shows the chance of getting a reward at each timestep in a trajectory from lever game (left) and speaker-listener (right).

Fig 2. shows that r-RNN and ar-RNN become more likely to get reward as the episode goes on. Whereas, RNN and a-RNN do not show this adaptive behaviour. These results indicate **that ar-RNN and r-RNN can learn about its partner** as they interact. Hence, they cooperate effectively. We interpret the results as follow:

- **The reward signal is necessary** for the emergence of cooperative skills. This is because the agent needs to know whether or not its current strategy is suited to the current partner.
- **The previous action input helps** the agent to cooperate quicker when used in combination with the reward signal because this feature can be used by the RNN to correlate the action with the reward. This makes it easier for the RNN to identify what is the correct action during adaptation.

Continual adaptation

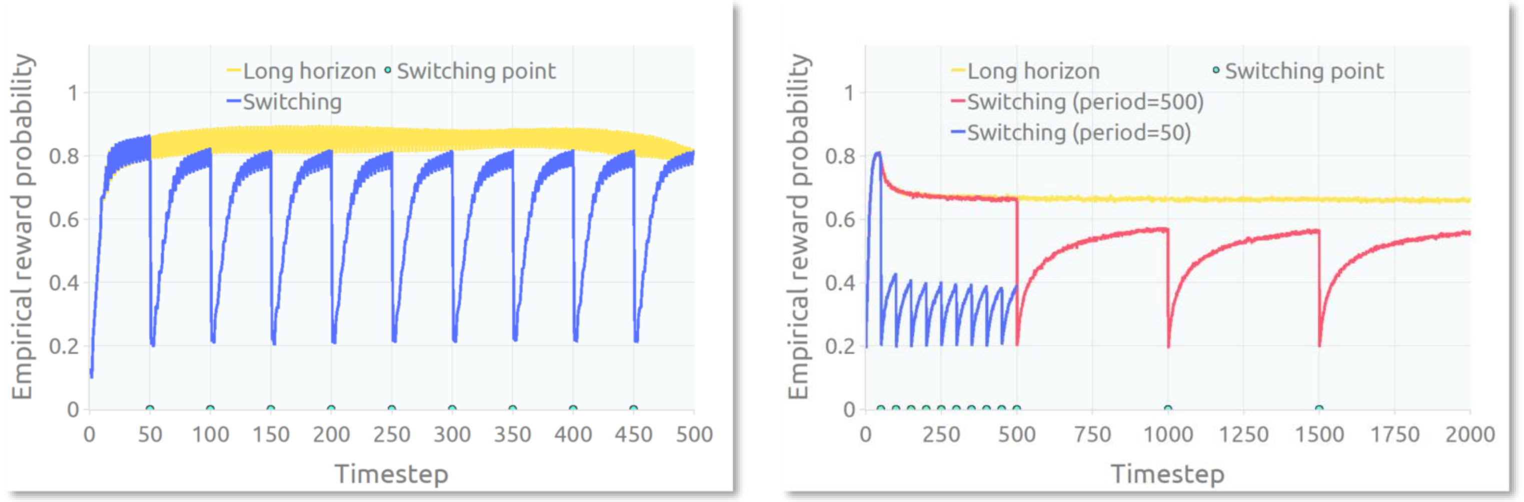

Figure 3: Continual adaptation. We test meta-RL agents in episodes with longer horizon and partner switching. The agents have stable performance when put into extremely long episodes. Also, they can adapt to multiple partners when the partner is changed periodically within an episode.

When an agent is deployed into the real world, **test-time scenarios might differ** from the ones that are used during the training. In this section, we examine the ability of a meta-RL agent to extrapolate under **unexpected situations** including working under **longer horizon** and **partner switching**. We find that the meta-RL agent is **robust when it performs to longer horizo**n length in both environments. The performance is stable throughout the entire trajectory.

Also, the meta-RL **agent can adapt flexibly** even though it has been trained to adapt with only one partner per episode. We see different adaptation behaviours when the agent is already adapted to one partner. Specifically, it adapts much faster to the first partner compared to later partners.

Limitation

We also study the **impact of the number of training partners**. We consider the number of training partners from the set of {5,10,15,20}. We observe that the ad hoc teamwork performance and cooperation **get better as we increase the number of training partners**. Next, we study the **impact of diversity of the training partners**. Instead of randomly selecting training partners from the pool of all possible agents, we select the training partners such that they only come from a specific part of a behaviour space. This is similar to **out-of-distribution testing** in supervised learning. The training score of meta-RL agent does not deteriorate when trained under this skewed distribution. However, we notice that **the test-time performance reduced significantly** in both environments.