ADAPTIVE ACTION ADVISING WITH DIFFERENT REWARDS

Yue Guo
Carnegie Mellon University
United States
yueguo@cs.cmu.edu

Zhang Xi-Jia
University of Michigan
United States
ponyz@umich.edu

Simon Stepputtis
Carnegie Mellon University
United States
stepputtis@cmu.edu

Joseph Campbell
Carnegie Mellon University
United States
jcampbell@cmu.edu

Katia Sycara
Carnegie Mellon University
United States
sycara@andrew.cmu.edu

ABSTRACT

Action advising is a critical aspect of reinforcement learning, involving a teacher-student paradigm wherein the teacher, possessing a pre-trained policy, advises the student with the actions calculated from its policy based on the latter's observations, thereby improving the student's task performance. An important requirement is for the teacher to be able to learn to robustly adapt and give effective advice in new environments where the reward is different from the one the teacher has been trained on. This issue has not been considered in the current teacher-student literature, therefore, most of the work require the teacher to be pre-trained with the same reward that the student interacts with and cannot generalize advice that differs from the policy; the reward that the student gained through interaction with the environment is also directly given to the teacher, regardless the exploration process. To fill this gap, our proposed method enhances action advising by allowing the teacher to learn by observing and collecting data from the student and adapting its reward function. We empirically evaluate our method over three environments consisting of a Gridworld, an ALE skiing, and a Pacman, and our method demonstrates improved policy returns and sample efficiency. [1]

1 INTRODUCTION

Reinforcement learning (RL) is often employed in robotics to learn policies enabling embodied agents to perform complex tasks such as autonomous driving (Wu et al., 2023; Osiński et al., 2020; Kiran et al., 2021) and game playing (Silver et al., 2018; 2017; Lample & Chaplot, 2017). Despite its remarkable success, state-of-the-art algorithms suffer from a high degree of sample inefficiency and require vast amounts of training data (Yarats et al., 2021; Scheller et al., 2020). This presents significant problems when attempting to train robots in the real-world, as a) collecting samples is time consuming, and b) poor exploratory actions may result in damage to either the robot or the environment. While many approaches focus on alleviating these issues by overcoming the sim-to-real gap (Abeyruwan et al., 2023; Salvato et al., 2021; Zhao et al., 2020), an alternative paradigm known as *action advising* has been proposed in which one agent directly advises another agent on which actions to take during training. Central to these methods is a teacher-student relationship, where a teacher agent trained in a source task issues advice to a student agent undergoing training in a target task. This advice serves as a form of guided exploration, and has been shown to improve the sample efficiency of the student (Torrey & Taylor, 2013).

One of the challenges inherent to these methods, however, is issuing helpful advice in the presence of reward misalignment between the source and target tasks. A teacher which blindly issues advice without considering task differences can harm the student's learning by actively guiding it to regions of the state space which yield low rewards. Prior works have attempted to address this by identifying the subset of states for which advice which is helpful and only issuing advice in these states, (Anand et al., 2021; Campbell et al., 2023), but this leads to a more conservative teacher and is thus less beneficial to the student. In this work, we instead propose an approach in which the teacher adapts its own advice such that it maximizes the reward associated with the target task. Rather than identifying *which* advice is helpful as in prior methods, our approach adapts the advice such that *all* of it is helpful.

Our proposed teacher, the Adaptive Reward Teaching (ART), performs the following steps, it: a) estimates the unknown target reward function based on the student's observed returns, b) optimizes its own policy with respect to the

[1] Code Released: https://github.com/sophieyueguo/adaptive_action_advising

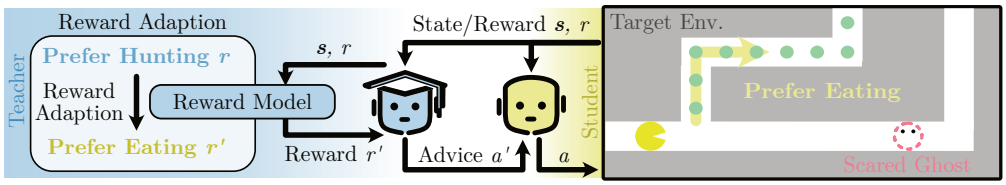


Figure 1: An overview of Adaptive Reward Teaching. The teacher initially issues action advice to the student from a policy trained in the source task – leading to low rewards in the target task due to differences in the reward functions. For example, the teacher could be a ghost chaser while the student is a conservative player that always avoids ghosts and eats food. The teacher has a much higher reward of eating ghosts than the student, however, it would not find this out until examining the data collected by the student. After observing the student's trajectory returns, the teacher adapts its policy using its so as to maximize the student's reward in the target task.

estimated reward function, and c) issues action advice from the updated policy. We assume that our teacher lacks direct access to the student's reward, while the teacher and student are trained with the same transition functions, state space, and action space. The primary contributions of our work are as follows. 1) We introduce Adaptive Reward Teaching, a novel teacher-student method which is robust to differences in rewards between the source and target tasks, leading to improved advice quality. 2) We validate our approach through empirical testing in three environments: a simulated urban search and rescue (USAR) scenario, a skiing game drawn from the Arcade Learning Environment (Machado et al., 2018) and Pacman. We empirically show that our method produces advice of higher quality than existing methods, improving the sample efficiency of reinforcement learning methods.

2 Related Work

Action advising, which, at its core, leverages the knowledge of a *teacher* to expedite the training process of a *student* agent, has a long-standing history in the field of RL (Thomaz & Breazeal, 2008; Kuhlmann et al., 2004; Maclin & Shavlik, 1996). Particularly, in action advising, the teacher provides advice to the student agent by suggesting actions in a given state, helping the student agent discover the correct actions faster than by relying purely on their own exploration. To this end, (Torrey et al., 2005; 2006) proposed methods that map human knowledge to RL tasks by utilizing a user-provided mapping while (Maclin & Shavlik, 1994) discuss giving advice by an external observer.

Human teachers have the remarkable ability to evaluate whether or not their advice should be adjusted given the student's observed experience. Recent work in the domain of action advising that utilize an expert policy in order to provide advice, struggle when the domain of the teacher differs from the domain of the student, as most approaches assume the task to be the same (Omidshafiei et al., 2019; Da Silva et al., 2017). To decide whether or not to give advice, (Torrey & Taylor, 2013) utilizes a heuristic approach while later work formulates the advising task as a separate learning problem (Zimmer et al., 2014). In such setting in which the teacher is sub-optimal, as the domain of the student shifted from the teacher's, the utility of the teacher heavily depends on the quality of its advice (Zhan et al.). This work addresses this problem with a curriculum of teachers that constantly observe and subsequently adapt their behavior in an approach inspired by DAGGER (Ross et al., 2011). However, such an approach requires the existence of multiple teachers. An alternative approach that addresses this domain shift is presented in (Guo et al., 2023), in which the teacher does not only provide the advice itself, but also an explanation as to why the advice was given. While this enables the student to reflect on the given advice, the teacher does not improve its performance and continues to provide fixed advice (Zhu et al., 2020). The work presented in (Campbell et al., 2023) introduces Introspective Action Advising, which allows the teacher to reflect upon its own advice, ultimately learning to withhold advice if the teacher's expected outcome does not align with the student's experience.

Besides action advising, there are various works on transferring knowledge, such as the field of inverse reinforcement learning (IRL) and imitation learning. We share the intuition with the cooperative IRL work (Hadfield-Menell et al., 2016), where one agent has access to the true reward, but not the other; the two agents form a team, so that the learning agent may learn better. However, we assume the agent who advises is trained with another reward, and has no access to the reward of the agent who receives advice. This challenge can be hardly resolved in the framework of IRL or imitation learning, but is promising in the framework of action advising. Another work of a pedagogical IRL (Ho et al., 2016) studies the improvement of sampling demonstrations for teaching over simply doing the tasks. This coincides with the research interest of "what to advise" in action advising, even if the frameworks of action advising and IRL are different. We are also interested in improving the quality of advice, however, we especially focus on how teacher's

policy can be adjusted overall with a different reward. Furthermore, agent adaptation may be productive when the agent has learnt various RL tasks (Bauer et al., 2023) and adapted to an unknown reward. Here in our work, we limit the teacher's capability of only learning its own task, however, the teacher's adaptation happens between different rewards. Some well-known work in transfer learning consider how the catastrophic forgetting can be prevented when a new task is learnt (Rusu et al., 2016; Kirkpatrick et al., 2017). While we are also interested in building a teacher that augments its expertise in various tasks that are learnt sequentially, we only focus on how it adapts to a new task, instead of also preserving its original knowledge simultaneously.

In this work, we directly address the domain shift between the student's and teacher's tasks by quickly adapting the teacher's reward model given the student's observed behavior and returned reward. This allows our teacher to not only withhold advice that is expected to negatively impact the student, but also allows it to go beyond prior work by actively improving the quality of issued advice.

3 Background

We investigate action advising in the context of reinforcement learning, which uses a Markov Decision Process (MDP) to model agent behavior. The MDP consists of a tuple $(\mathcal{S}, \mathcal{A}, R, T, \gamma)$, where $\mathcal{S}$ denotes a collection of states within the environment, $\mathcal{A}$ signifies a set of feasible agent actions, $R : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is a reward function defined as $R(s, a) = \mathbb{E}\left[r_{t+1}|s_t = s, a_t = a\right]$, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ represents a state transition probability given by $T(s', s, a) = P(s_{t+1} = s'|s_t = s, a_t = a)$, and $\gamma \in [0, 1]$ is the discount factor. A stochastic policy is denoted by $\pi(a|s) : \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$, indicating the probability that an agent opts for action a given the state s.

Action advising hinges on the teacher-student paradigm, where a teacher agent aims to advise the student agent by offering guidance in the form of actions based on the student's observations during the training process. The teacher has access to a policy pre-trained in a source task, π^T and provides recommended actions to the student agent attempting to learn a policy π^S in a target task. Formally, let s_t represent the student's perceived state of the environment at time t, and a_t denote the action undertaken by the student.

$$\begin{aligned} a_t = a_t^T \sim \pi^T(a_t|s_t) \text{ if } H(s_t) = 1, \\ \text{or } a_t^S \sim \pi^S(a_t|s_t) \text{ if } H(s_t) = 0, \end{aligned} \tag{1}$$

where $H : \mathcal{S} \mapsto 0, 1$ serves as a binary indicator function which is assigned the value 1 when the teacher provides advice, and 0 in its absence.

4 Methodology

Algorithm 1 Adaptive Reward Teaching (ART)

Input: Trained teacher policy π^T, advising strategy H, adaptation period μ.

Prior to student training
Teacher collects $\mathcal{D}_T = \{(s_t, a_t, r_t)\}_{t=1}^{\omega}$ in its environment.
Teacher trains reward model $\mathcal{M}(\mathcal{D}_T)$.

During student training
if iteration $\leq \mu$ **then**
    Teacher collects β student trajectories $\mathcal{D}_S = \{(s_t, a_t, r_t)\}_{t=1}^{\beta}$
    Reward adaptation: $\mathcal{M}' = \mathcal{M}(\mathcal{D}_S) + \mathcal{E}(\mathcal{D}_S)$
    Policy adaptation: $\pi^T = \text{argmax}_\pi \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \mathcal{M}'(s_t, a_t)\right]$
end if
Teacher issues advice $a_t^T \sim \pi^T(a_t|s_t)$ if $H(s_t) = 1$

In this work, we consider a scenario in which a teacher that has already learned a policy π^T in a source task seeks to advise a student learning a (randomly initialized) policy π^S in a target task. We assume that the state-action spaces of the source and target tasks are the same, however, the tasks exhibit different reward functions. This presents a problem for action advising in that the teacher may issue advice that leads the student to regions of the state space associated

with low rewards in the target task, despite yielding high rewards in the source task. Our goal in this work is to adapt the teacher's policy π^T such that it maximizes the reward associated with the target task, rather than the source task in which it is trained, side-stepping the above problem.

Our approach consists of three main steps: 1) learning a model of the reward function of the source task during teacher training, 2) estimating the unknown reward function associated with the target task, and 3) adapting the teacher's policy via fine-tuning so as to maximize the target task's reward, which can then be used to issue action advice. While step 1 is only performed once prior to student training, steps 2 and 3 are performed throughout the student's training process, leveraging the student's interactions with the environment as training data. Intuitively, our approach can be understood as the teacher observing the student's returns in the target task, estimating how they differ from those in the source task in which it was trained, and then tailor its policy to the target task and afterwards advising the student.

Learning Environment and Reward Dynamics: Prior to advising a student in the target task, the teacher learns an approximation of the source task's reward function $\mathcal{M}$. Formally, for each time step t, the teacher observes its state s_t, takes an action a_t under policy π^T, and receives a reward r_t from the environment. The collected data takes the format of $\mathcal{D}_T = (s_t, a_t, r_t)_{t=1}^{\omega}$, where ω represents the total number of time steps in the teacher's data sets. (Algorithm 1 Line 3). $\mathcal{D}_T$ is utilized to train a reward prediction model, which is a neural network $\mathcal{M}$ parameterized by ϕ (Algorithm 1 Line 4). $\mathcal{M}$ aims to estimate the reward r given a state-action pair (s, a): $\mathcal{M}(s, a; \phi) \rightarrow r$. Thus the training objective for $\mathcal{M}$ is to minimize the difference between the predicted reward and the actual reward:

$$L(\phi) = \mathbb{E}\left[||\mathcal{M}(s_t, a_t; \phi) - r_t||^2\right], \tag{2}$$

Reward Adaptation: Once the student begins training, ART observes the student's interactions with the target environment for β episodes, gathering the student's state-action-reward data, denoted by $\mathcal{D}_S = (s_t, a_t, r_t)$, where $t \in [1, \beta]$ (Algorithm 1 Line 7). To accommodate the differences in the rewards of the source and target tasks, ART introduces an error prediction model $\mathcal{E}$, parameterized by ψ. This model is trained to predict the discrepancy between the rewards estimated by the initial reward model $\mathcal{M}$ and the actual rewards obtained by the student in the target task.

The error prediction model $\mathcal{E}$ aims to estimate the error ϵ_t given a state-action pair (s_t, a_t): $\epsilon_t = \mathcal{E}(s_t, a_t; \psi)$. The error ϵ_t represents the difference between the predicted reward and the actual reward observed by the student: $\epsilon_t = r_t - \mathcal{M}(s_t, a_t; \phi)$. The training objective for $\mathcal{E}$ is to minimize the difference between the predicted error and the actual error:

$$L(\psi) = \mathbb{E}\left[||\mathcal{E}(s_t, a_t; \psi) - (r_t - \mathcal{M}(s_t, a_t; \phi))||^2\right], \text{where} (s_t, a_t, r_t) \in \mathcal{D}_S. \tag{3}$$

Following the training of the error prediction model, the adapted reward at each step is computed as the sum of the predicted reward and the predicted error: $\mathcal{M}'(s_t, a_t) = \mathcal{M}(s_t, a_t; \phi) + \mathcal{E}(s_t, a_t; \psi)$. This adaptation allows the teacher to more accurately align its reward predictions with the actual rewards in the target task, facilitating more effective policy adaptation and advice to the student.

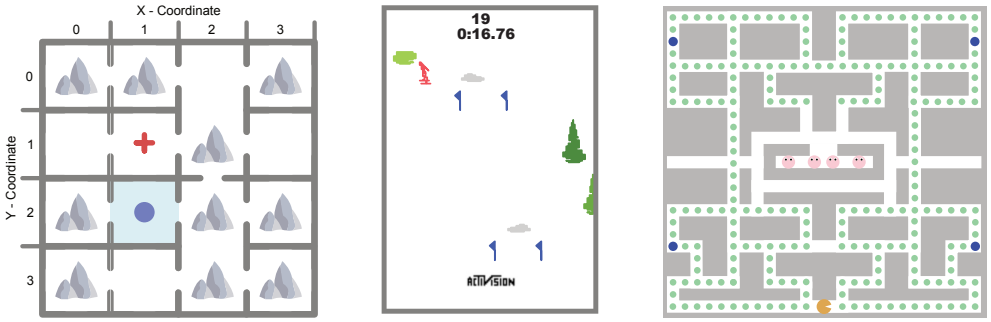
Policy Adaptation: After adjusting its reward prediction model, the teacher uses the updated $\mathcal{M}'$ to perform a limited number of reinforcement learning iterations and improve its policy π^T with respect to the target task (Algorithm 1 Line 9), by modifying the parameters of π^T based on the standard update rule: $\theta' = \theta + \alpha \nabla J'(\theta)$, where $J'(\theta)$ maximizes the expected cumulative reward under the new reward distribution:

$$J'(\theta) = \mathbb{E}\pi^T \left[\sum_{k=0}^{\infty} \gamma^k \mathcal{M}'(s_t, a_t; \phi) \middle| s_t, a_t \right] \tag{4}$$

This adaptation process allows the teacher to improve its policy and may be performed numerous times throughout the student's training process. The updated policy is subsequently used to issue future action advice to the student according to an advising heuristic $H(\cdot)$ – our approach is agnostic of the specific heuristic which is used. The effectiveness of this adaptation step is contingent on how accurate the reward model is. As a result, adaptation steps performed early in the interaction when a limited number of student returns have been observed are likely to be more noisy than those performed later in training.

5 Experiments and Results

We empirically evaluate our proposed adaptive teacher in three separate scenarios. These domains have been selected such that: 1) the teacher and student have different rewards, and this leads to different policies and behaviors after the training convergence; 2) the student performance is better when advised by an adaptive teacher, compared to student's performance with a teacher that never adapts or when the student receives no advice; and 3) the student reward may be inferred from a small batch of data at the beginning of its training.



(a) Search and Rescue (b) Skiing (c) Pacman

Figure 2: Three environments to test our adaptive advising method. (a) The agent (blue circle) navigates to the room where a rubble is removable, and a victim represented by red cross needs to be rescued. (b) The agent (red) needs to avoid the trees (green) and gates (blue). It can pass the gates if it goes through the empty space between gate pairs. (c) The pacman (yellow) eats up food (green) and power pellets (blue). In the default mode, the ghosts (pink) chase the pacman and can eat it up; After the pacman eats up a power pellet, it can eat ghosts. The maximum steps for (a), (b), and (c) are 500, 1500, and 600.

We use the following environments. USAR: A Gridworld which simulates an urban search and rescue environment in which the agent needs to remove rubble and rescue trapped victims. Skiing: The ALE (Machado et al., 2018) Skiing game game where the agent moves left or right or having no actions. The agent always sits on top of the screen and the background keeps rolling up to simulate the agent is moving downwards. Pacman: A variant of the Pacman game with a feature-based state representation.

5.1 Action Advising Comparisons

To evaluate the effectiveness of our adaptive advising method, we first compare the learning performance of the student agent under the action advising of different types of teachers. The types of teachers we compare are:

Adaptive Reward Teaching (ART): The teacher is trained in the source environment and adapts its policy to align with the target task's reward function (used by the student). This adaptation occurs within one or a few iterations of the beginning of student's training, leading to a noticeable improvement or 'jumpstart' in the student's performance, and helps the student to converge faster.

Non-adaptive Teacher: The teacher is trained in the source environment and provides consistent advice throughout the learning process without adapting its policy. As a result, the advice may be suboptimal with respect to the target environment's reward function.

Ideal Teacher: This hypothetical teacher is an optimal policy within the student's target environment. The ideal teacher is included as a baseline to represent the upper limit of teaching quality.

No Advice: The student agent receives no advice and learns from scratch. This approach typically shows high variance in performance and, although it can potentially converge to an optimal solution, there is no guarantee of this outcome.

Additionally, our methodology involves a gradual decay in the probability of advice being given. When the probability goes to zero, it means no advice is given from the teacher, and thus the reward reflects the actual student's performance. Although there are a range of works focusing on when to issue the advice, we utilized the simple heuristic method and focused on adaptability. Key hyperparameters in this decay advising process include the initial probability of advice, the final period when the advice stops, and the rate at which the probability is reduced. We present the best empirical results from various combinations of these parameters to illustrate the effectiveness of our adaptive advising approach in different environment settings. This comparative analysis allows us to robustly evaluate the benefits of adaptive advising in enhancing the student's learning process across various environments. Each of the methods is averaged over 5 episodes with different seeds.

USAR: We apply our method to a simulated search and rescue scenario (Lewis et al., 2019; Freeman et al., 2021) modeled as a Gridworld environment. Shown in Fig. 2, this is a single agent scenario in which the agent (blue circle) must navigate the environment, remove rubble, and rescue victims (red cross). In this variant, there is a single victim and single piece of rubble randomly located within the 14 room layout. The source environment only rewards the agent

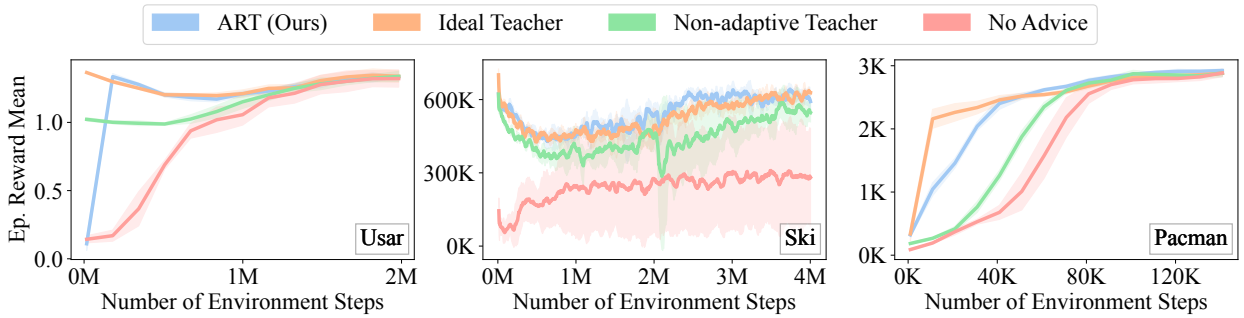


Figure 3: Training curves for a student which receives action advice from our proposed method as well as the action advising variants described in Sec. 5.1. Left-to-right: USAR, Skiing, and Pacman.

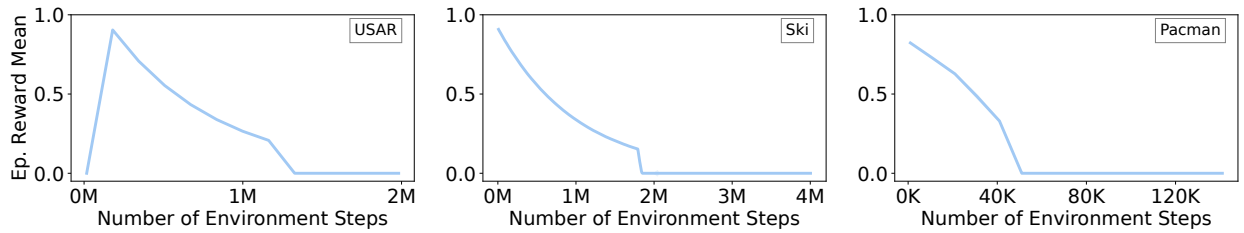


Figure 4: The advising probability of each environment.

for rescuing the victim with a per-timestep penalty – no reward is given for removing rubble. The optimal behavior in the source environment is to ignore rubble and immediately locate and rescue the victim. To induce a change in rewards, the target task rewards the agent for both removing rubble and rescuing victims. Correspondingly, the optimal behavior in the target environment is to remove as much rubble as possible before rescuing the victim.

The baseline comparison result is shown in Figure 3 (left). The student receiving no advice learned the slowest and had the most variance but ultimately converged to the optimal reward. The student advised by the non-adaptive teacher received sub-optimal advice, yet it was still beneficial and resulted in an initial performance improvement over the baseline student with no advice. Despite the sub-optimality, this student was able to converge to the optimal reward due to the decaying probability of issuing advice. Our proposed method, ART, achieved performance comparable to the student advised by an ideal teacher. We note that the teacher issued no advice in the first 256K training steps, as observations were collected only to adapt the teacher's reward function.

Skiing: In the Atari (Machado et al., 2018) Skiing environment shown in Figure 2 (b), the primary objective is to navigate a downhill course as quickly as possible while avoiding obstacles such as trees and navigating through gates. The source environment uses the default reward structure from ALE, where the agent receives $+10000$ score for passing through a gate and -1000 for hitting an obstacle, with a per-timestep penalty. The optimal behavior results in the agent passing through gates near the agent's position while avoiding obstacles. Empirically, this results in an agent trained in the source environment to pass through 58% of the gates on average. The target environment increases the reward for passing through gates by 10X, thereby encouraging the agent to focus on passing through every gate while avoiding collisions. This necessitates more conservative behavior as gates are worth relatively more compared to the per-timestep penalty. The resulting optimal agent behavior passes through 87% of the gates on average.

Empirical results are shown in Figure 3 (middle). The student receiving no advice exhibited training instability and often failed to converge to optimal behavior. The student with the non-adaptive teacher did converge to optimal behavior, however, suffered a noticeable drop in performance once the advising probability fully decayed to 0 and advice was no longer issued. This was not an issue for the student advised by the adaptive or ideal teacher, where the reward growth is consistent and smooth even when advising stopped.

Pacman: Pacman is shown in Figure 2 (c), which is a widely used benchmark for reinforcement learning. The reward encourages the agent to eat food which increases the agent's score as well as power pellets, which cause the ghosts to enter a "frightened" state after which they can be eaten for a large increase in score. The source environment highly

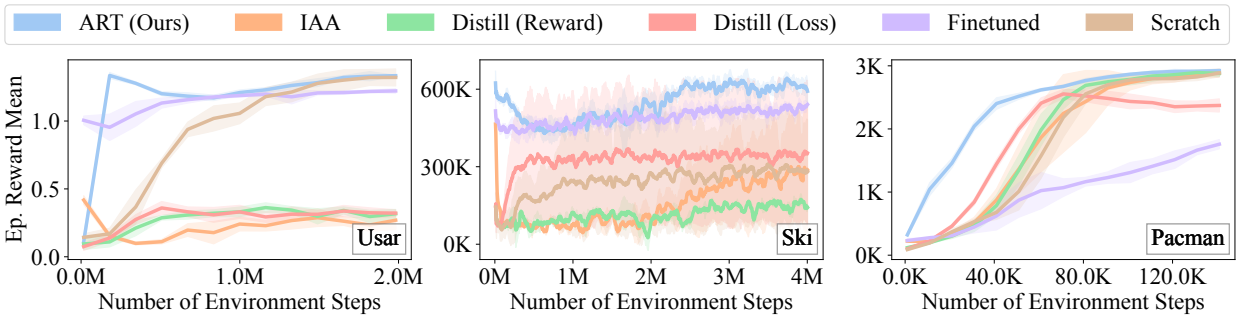


Figure 5: Training curves for a student which receives action advice from our proposed method as well as the methods defined in Sec. 5.2. Left-to-right: USAR, Skiing, and Pacman.

rewards the agent for consuming ghosts, resulting in an optimal behavior that is characterized by a strong preference for chasing ghosts, sometimes leading to risky situations or hesitation between pursuing ghosts and consuming more pellets. On average, an optimal agent trained in the source environment consumed approximately 6.4 ghosts and 45% of the available food.

The target environment, in contrast, rewards the agent for collecting food while avoiding ghosts, as the agent is not rewarded for eating ghosts even when they are in the "frightened" state. On average, the optimal agent trained in the target environment consumed 1.4 ghosts and 85% of the food.

The results are shown in Figure 3 (right). Our experimental results reveal significant performance disparities between students advised by adaptive and non-adaptive teachers. The non-adaptive teacher, though helpful in directing the student to eat food, demonstrates a starkly inferior performance compared to the adaptive teacher as it encouraged chasing ghosts. In such cases, the student's performance aligns more closely with an untrained, random agent, often getting quickly eaten by ghosts with minimal food consumption. Notably, the difference in performance between students trained with adaptive and non-adaptive teachers in the Pacman environment was more pronounced than in the USAR or Skiing scenarios. While the adaptive teacher's performance gradually converges towards that of an ideal teacher, it demonstrates a clear advantage over other baseline approaches. Our experiments, also additionally indicated that an initial advising probability less than 1.0 (specifically 0.8 in our case) was optimal. This allowed for sufficient exploration by the student while still benefiting from the teacher's guidance. Overall, the Pacman environment is more challenging for the teacher to adapt since there is a large shift in dynamics after a power pellet is eaten. This challenge is further exacerbated in our scenario, given the difference in reward between the source and target environments.

The advising probability over time-steps is shown in Figure 4. This balances the student's exploring and following teacher's advice. We empirically choose the best hyper-parameters for each of the environment. The details are reported in the Appendix A.

5.2 Transfer Learning Comparisons

We compare our approach to three state-of-the-art transfer learning methods shown in Figure 5. All methods use the same teacher policy which has been pre-trained with the reward in the source environment and used to transfer to a student learning from scratch with the reward in the target environment.

Introspective Action Advising (IAA)(Campbell et al., 2023): Action advising method in which the teacher identifies helpful advice based on the expected state-value in the target task.

Policy Distillation (Loss)(Schmitt et al., 2018): Also known as kickstarting, this is a policy distillation method which adds an auxiliary loss term based on the cross-entropy between the student and teacher policies.

Policy Distillation (Reward)(Czarnecki et al., 2019): Policy distillation method which introduces a reward shaping term that captures the difference in the teacher's critic between the previous and current timesteps.

We also evaluate against several simple ablations.

Finetune: Student policy initialized with the teacher's policy and trained in the target task.

Random: A randomly initialized policy trained from scratch in the target task.

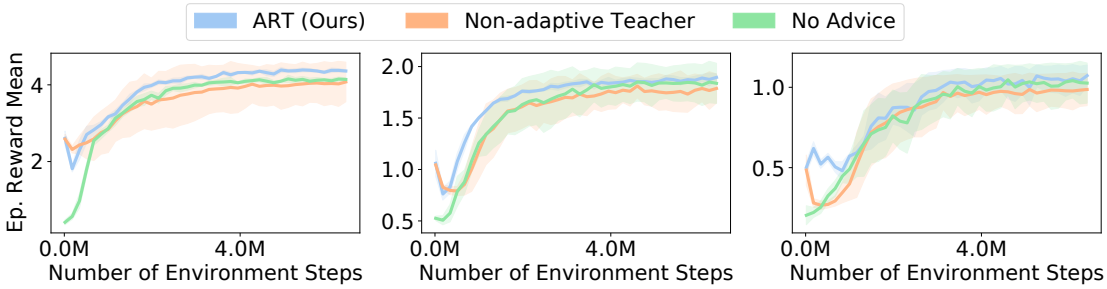


Figure 6: Training curves for our proposed method as well as a student advised by a non-adaptive teacher and a student with no advising. The reward for removing each piece of rubble varies from 0.5 (left), 0.25 (middle), and 0.125 (right).

In these experiments, our approach uses a heuristic function such that the probability the advice is issued decays over time, and finally the student has learnt to perform tasks without teacher's advice.

The Adaptive Teacher strategy demonstrated superior performance in the USAR and Skiing scenarios when contrasted with other benchmarks. This advantage, however, was less pronounced in the more complex Pacman environment, though it still maintained a lead. The efficacy of the Introspective Action Advising (IAA) approach was particularly notable in environments where differences could be assessed through value function estimations. This allowed for more advanced advising strategies, as the teacher's insights were grounded in a robust understanding of the environment's dynamics. However, in scenarios where the reward changes leading to policy differences, the value-based insights provided by IAA became less effective. In such cases, the performance of IAA aligned more closely with that of traditional policy distillation methods, which rely on mimicking teacher behaviors either through reward or loss minimization. This convergence highlights the challenges posed by environments with altered rewards and underscores the need for adaptable advising.

5.3 Adaptability with respect to Policy Variance

We further explored the impact and limits of an adaptive teacher on a student's learning performance in a variant of the USAR environment which contained 10 rubble and 1 victim. The agent's task was to find the victim and clear the rubble, with the episode ending either upon finding the victim or after 500 steps. For the teacher, rewards were given for finding victims (+1) and removing rubble (+1), and these rewards decayed over time. The student's reward differed from that of the teacher, with varied levels of difference. While finding the victim always yielded a reward of 1.0 for the teacher and all the student policies, we varied the reward for rubble removal between 0.0625 and 0.5 for different students. This necessitated the agent to balance the trade-off between conserving steps to maximize the reward for finding the victim as soon as possible and exploring more rooms to locate all the rubble and remove them. As the teacher received high reward of removing rubble and would remove all of them before finding the victim, the varied smaller reward for removing rubble by the student indicated the priority of removing rubble decreased for different students. This setup introduced more variability in behavior and required longer times for the agent's policy to converge. We conducted experiments with the teacher adapting to each of the different student policies and advising them. Each scenario was tested over five episodes, and the results were evaluated with 1,000 trajectories. Instead of presenting each of the learning processes, we evaluated the performance to demonstrate adaptability, compared to no advice and non-adaptive decay advise. The performances are evaluated after the student's training has been converged, meanwhile the teacher has already finished its advising and the evaluation is done for the students' policies.

First, the adaptive teacher improved the student's policy convergence as shown in Figure 6. This improvement was particularly noticeable when the reward for rubble removal was set at moderate values (0.25 and 0.5), which is closer to the teacher's 0.1. The complexity of the task required the agents to balance between exploring and quickly finding the victim. For students where the reward was smaller, the adaptive teacher's impact was less pronounced. The non-adaptive teacher also helped improving convergence in general, but less so than the adaptive teacher.

Intuitively, as shown in Figure 7 (left), the decreased reward of removing rubble would lead to a decreased number of rubble removal of the converged student's policy. Therefore, from the general trend, we observed an increase of rubble removal as expected. Notably, when the reward for rubble removal was equal to that for finding the victim (1.0), the teacher and student achieved same performances after convergence. Both the adaptive and non-adaptive teacher would not affect the number of rubble removal after convergence at the end. However, the adaptive teacher consistently outperformed both the non-adaptive teacher and scenarios with no advice in helping the agent find victims

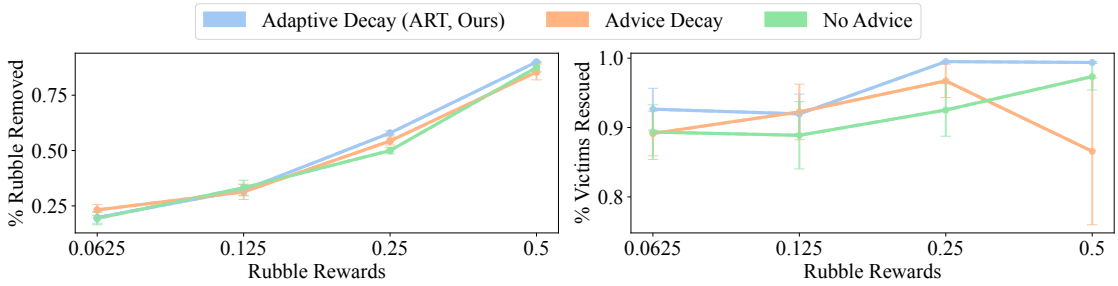


Figure 7: USAR 10 Rubble: Percentage of removed rubble (out of 10) and triaged victims (out of 1) over 1000 rollouts across five student seeds each.

as shown in Figure 7 (right). It guided all the students to find the victims and reduced the errors of the victims not being found. This was especially evident with reward settings of 0.5, where the trade-off of exploring and quickly finding the victims was the hardest to resolve. The non-adaptive teacher, which always prioritizing removing rubble, led to a high error and unstable performance of the students, while the adaptive teacher's guidance led to more stable and efficient victim-finding strategies. In summary, the experiment demonstrated the efficacy of an adaptive teacher in guiding an agent through varied students' rewards. The adaptive teacher not only enhanced the agent's ability efficiently but also expedited the policy convergence process, particularly in scenarios with moderate reward discrepancies.

5.4 Reward Similarity

Given two reinforcement learning tasks with the same state-action space, we quantify the average reward difference across all state-action pairs. For each state-action pair (s, a), let $D(s, a) = |R_1(s, a) - R_2(s, a)|$ denote the absolute difference in rewards between the two tasks. The average reward difference D is defined as:

$$D = \frac{\sum_{(s,a)\in S\cdot A} D(s, a)}{|S \cdot A|}$$

This formula calculates D by summing the reward differences for all state-action pairs and dividing by the total number of pairs, yielding an average of the reward discrepancies. A more robust approach to compute reward difference could be found in literature (Gleave et al., 2020), which assumes the reward functions could be unknown, and those with difference values may also be considered equivalent if they leads to similar behaviors without actually training optimal policies with them. However, in our case the situation is slightly different as we have access to the reward functions, don't have distribution assumptions, and don't consider the challenge of different values leading to similar behaviors. As we are interested in the teacher's adaptability, training is inevitable, which does not utilize the superiority of evaluation with no training. Furthermore, we also would like to compare across environments. These factors lead to our choice of the most straightforward similarity measurement.

To facilitate a normalized comparison of reward differences across various environments, we define D_{norm} as:

$$D_{\text{norm}} = \frac{D}{\max_{(s,a)\in S\cdot A} D(s, a)}$$

Here, $\max D$ is the maximum possible average difference, which is determined by the structure of the tasks' rewards. Normalizing D to D_{norm} scales the average difference to a range between 0 and 1.

Generally, the Pacman environment has a bigger reward difference compared to the other two, where critical objects such as rubble or gates are static, while eating ghosts can happen anywhere in the maze. The computation details are shown in Appendix B. In order to further quantify the influence, we calculate the performance difference between the teacher and the trained student with ART, when evaluated in the student's environment.

$$\text{Performance Improvement} = \frac{\text{perf}_{\text{ART}} - \text{perf}_{\text{teacher}}}{\text{perf}_{\text{teacher}}}$$

Figure 8 shows the result where the performance improvement is over the normalized reward difference. The Pacman who has the highest reward difference also yields a higher performance improvement. This means our ART is more helpful when the reward difference is bigger, and may also explain the small difference between ART and the finetune

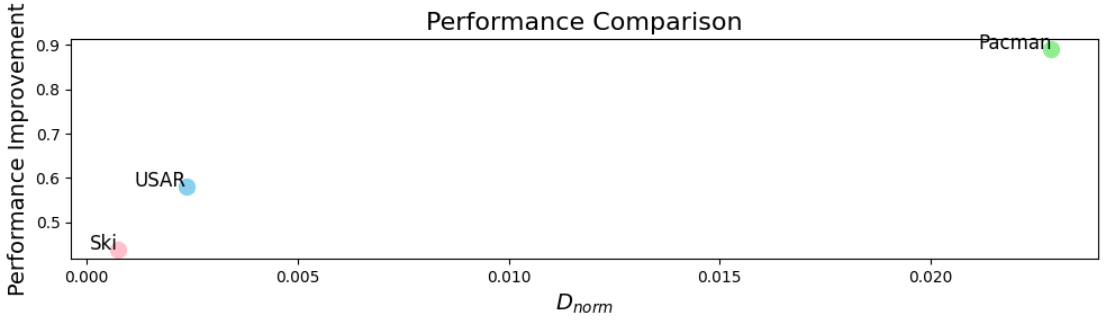


Figure 8: Improvement over the reward difference.

methods in Ski and USAR, because in these two scenarios the reward difference is smaller, and there is limited room to for the teacher to improve and provide the student with better advice. However, we also don't anticipate the performance improvement keeps growing when the reward difference becomes bigger. We acknowledge our study majorly focuses on the small D_{norm} space. The boundary exploration will be saved for future work.

6 Conclusion

In this work, we have introduced Adaptive Reward Teaching – an action advising method which produces a teacher capable of adapting its advice to a target task in the face of changing rewards. We empirically show that our approach results in significant policy improvement with respect to the target task's reward function in a far more sample efficient manner when compared to a number of state-of-the-art transfer learning approaches. In the future, we plan to further explore the adaptability of the teacher and quantify the distribution shift of the student's data from teacher's.

7 Acknowledgement

This work has been funded in part by DARPA under grant HR001120C0036, the Air Force Office of Scientific Research (AFOSR) under grants FA9550-18-1-0251, the Army Research Laboratory (ARL) under Grant W911NF-2320007 and Honda Research Institute award number 58629.1.1012949. We thank discussions from Honda Research Institute with Vaishnav Tadiparthi, Behdad Chalaki, Hossein Nourkhiz Mahjoub, Jovin Dsa, Ehsan Moradi Pari, and Kwonjoon Lee.

References

Saminda Wishwajith Abeyruwan, Laura Graesser, David B D'Ambrosio, Avi Singh, Anish Shankar, Alex Bewley, Deepali Jain, Krzysztof Marcin Choromanski, and Pannag R Sanketi. i-sim2real: Reinforcement learning of robotic policies in tight human-robot interaction loops. In *Conference on Robot Learning*, pp. 212–224. PMLR, 2023.

Daksh Anand, Vaibhav Gupta, Praveen Paruchuri, and Balaraman Ravindran. An enhanced advising model in teacher-student framework using state categorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 6653–6660, 2021.

Jakob Bauer, Kate Baumli, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, Vibhavari Dasagi, Lucy Gonzalez, et al. Human-timescale adaptation in an open-ended task space. In *International Conference on Machine Learning*, pp. 1887–1935. PMLR, 2023.

Joseph Campbell, Yue Guo, Fiona Xie, Simon Stepputtis, and Katia Sycara. Introspective action advising for interpretable transfer learning. In *Conference on Lifelong Learning Agents*, 2023.

Wojciech M Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling policy distillation. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1331–1340. PMLR, 2019.

Felipe Leno Da Silva, Ruben Glatt, and Anna Helena Reali Costa. Simultaneously learning and advising in multiagent reinforcement learning. In *Proceedings of the 16th conference on autonomous agents and multiagent systems*, pp. 1100–1108, 2017.

Jared T Freeman, Lixiao Huang, Matt Woods, and Stephen J Cauffman. Evaluating artificial social intelligence in an urban search and rescue task environment. In *AAAI 2021 Fall Symposium, 4-6 Nov 2021*. AAAI, 2021.

Adam Gleave, Michael D Dennis, Shane Legg, Stuart Russell, and Jan Leike. Quantifying differences in reward functions. In *International Conference on Learning Representations*, 2020.

Yue Guo, Joseph Campbell, Simon Stepputtis, Ruiyu Li, Dana Hughes, Fei Fang, and Katia Sycara. Explainable action advising for multi-agent reinforcement learning. In *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Mark K Ho, Michael Littman, James MacGlashan, Fiery Cushman, and Joseph L Austerweil. Showing versus doing: Teaching by demonstration. *Advances in neural information processing systems*, 29, 2016.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Gregory Kuhlmann, Peter Stone, Raymond Mooney, and Jude Shavlik. Guiding a reinforcement learner with natural language advice: Initial results in robocup soccer. In *The AAAI-2004 workshop on supervisory control of learning and adaptive systems*. San Jose, CA, 2004.

Guillaume Lample and Devendra Singh Chaplot. Playing fps games with deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Michael Lewis, Katia Sycara, and Illah Nourbakhsh. Developing a testbed for studying human-robot interaction in urban search and rescue. In *Human-Centered Computing*, pp. 270–274. CRC Press, 2019.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Richard Maclin and Jude W Shavlik. *Incorporating advice into agents that learn from reinforcements*. University of Wisconsin-Madison. Computer Sciences Department, 1994.

Richard Maclin and Jude W Shavlik. Creating advice-taking reinforcement learners. *Machine Learning*, 22(1-3): 251–281, 1996.

Shayegan Omidshafiei, Dong-Ki Kim, Miao Liu, Gerald Tesauro, Matthew Riemer, Christopher Amato, Murray Campbell, and Jonathan P How. Learning to teach in cooperative multiagent reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 6128–6136, 2019.

Błażej Osiński, Adam Jakubowski, Paweł Ziecina, Piotr Miłoś, Christopher Galias, Silviu Homoceanu, and Henryk Michalewski. Simulation-based reinforcement learning for real-world autonomous driving. In *2020 IEEE international conference on robotics and automation (ICRA)*, pp. 6411–6418. IEEE, 2020.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Erica Salvato, Gianfranco Fenu, Eric Medvet, and Felice Andrea Pellegrino. Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning. *IEEE Access*, 9:153171–153187, 2021.

Christian Scheller, Yanick Schraner, and Manfred Vogel. Sample efficient reinforcement learning through learning from demonstrations in minecraft. In *NeurIPS 2019 Competition and Demonstration Track*, pp. 67–76. PMLR, 2020.

Simon Schmitt, Jonathan J Hudson, Augustin Zidek, Simon Osindero, Carl Doersch, Wojciech M Czarnecki, Joel Z Leibo, Heinrich Kuttler, Andrew Zisserman, Karen Simonyan, et al. Kickstarting deep reinforcement learning. *arXiv preprint arXiv:1803.03835*, 2018.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Andrea L Thomaz and Cynthia Breazeal. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence*, 172(6-7):716–737, 2008.

Lisa Torrey and Matthew Taylor. Teaching on a budget: Agents advising agents in reinforcement learning. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 1053–1060, 2013.

Lisa Torrey, Trevor Walker, Jude Shavlik, and Richard Maclin. Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pp. 412–424. Springer, 2005.

Lisa Torrey, Jude Shavlik, Trevor Walker, and Richard Maclin. Skill acquisition via transfer learning and advice taking. In *Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings 17*, pp. 425–436. Springer, 2006.

Jingda Wu, Zhiyu Huang, Zhongxu Hu, and Chen Lv. Toward human-in-the-loop ai: Enhancing deep reinforcement learning via real-time human guidance for autonomous driving. *Engineering*, 21:75–91, 2023.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10674–10681, 2021.

Yusen Zhan, Haitham Bou Ammar, and Matthew E Taylor. Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer.

Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pp. 737–744. IEEE, 2020.

Changxi Zhu, Yi Cai, Ho-fung Leung, and Shuyue Hu. Learning by reusing previous advice in teacher-student paradigm. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1674–1682, 2020.

Matthieu Zimmer, Paolo Viappiani, and Paul Weng. Teacher-student framework: a reinforcement learning approach. In *AAMAS Workshop Autonomous Robots and Multirobot Systems*, 2014.

A HYPER-PARAMETERS

The hyper-parameters most relevant to the performance of ART are as follows:

1. **Initial Follow Teacher Probability (IFTP)** This is the same as the initial advising probability that the teacher issues advice. If IFTP=1.0, the teacher always gives advice, otherwise the teacher advises with a probability and the student has more freedom exploring its own environment. In our case, the student always follows the teacher's advice when issued.
2. **Burn In (BI)** This is the beginning duration where the teacher does not give advice but instead lets the student explore the environment, and collects the student's reward data.

3. **Advise Decay Rate (ADR)** The teacher gradually lowers its probability of giving advice, until there is no advice at all given to the student. ADR is the decay rate γ where $Prob_{T+1} = \gamma * Prob_T$, where T is the time-steps of a training batch.
4. **Advise Decay Rate (ADS)** Similar as ADR, ADS is the decay step σ where $Prob_{T+1} = Prob_T - \sigma$, where T is the time-steps of a training batch.

Overall, the freedom that the student initially explores the environment, the amount of data the teacher collects before its fine-tuning, and the amount of advice issued empirically affect the performance of ART the most. Figure 9 shows an example of the teacher advising the student with the reward of removing rubble of 0.9 in the 10 rubble USAR scenario that we have shown in the experiment section. It can be inferred that with a smaller IFTP, longer BI, bigger ADR and bigger ADS, the advice given to the student is less, and thus the student is less affected by the teacher's advice. It may also allow the teacher to be more accurate when estimating the reward in the student's environment as it collects more data based on the student's exploration. However, the downside of advising less is that if the teacher's policy is beneficial, the student's own exploration maybe unnecessary, especially in this example where the teacher's advice on removing all the rubble aligns a lot with the student's reward.

As shown in Figure 9, from left to right, the student's performance with ART becomes more and more similar to the regular RL training where there is no advice. It is common in action advising that advising too much (left-most columns) may limit the student's exploration and when advising stops, the student's performance drops a lot and takes time to learn from the unexplored space. This issue empirically may be fixed with a smaller ADS or ADR, and a lower IFTP, depending on the specific tasks. Additionally, certain combinations of these important hyper-parameters may not work, as some outlier exist that limit the student's learning. We avoid the combinations that produce outliers when selecting important hyper-parameters.

The important hyper-parameters we used in the experiment section are listed as in the following table. When the advising probability is reduced close to 0.2, stopping advising empirically does not cause the student performance to drop since the student has already learnt, therefore in the figure sharp drops to zero can be observed.

Environment	IFTP	BI	ADR	ADS
USAR	0.9	100000	0.9999985	N/A
Ski	0.9	0	0.999999	N/A
Pacman	0.8	0	N/A	0.000015

Table 1: Important Hyper-parameters

There are other hyper-parameters such as when the teacher stops training, i.e. teacher training stop timestep (TTST), the teacher training frequency, i.e. inner loop teacher frequency (ILTF), the teacher training iterations (TTI), and basic RL training hyper-parameters for both teacher and student. All the experiments use the algorithm of PPO and the framework of Torch in RLlib. We report the teacher's training details in the following table, and the remaining could be identified in our code, which will be released upon acceptance. The other reported hyper-parameters are common RL training hyper-parameters: learning rate (LR), vf clip param (VFCP), vf loss coefficient (VFLC), kl coefficient (KLC), entropy coefficient (EC), train batch size (TBS), stochastic gradient descent mini-batch size (SGDS), and number of stochastic gradient descent iteration (NSGDI).

Environment	TTST	ILTF	TTI	LR	VFCP	VFLC	KLC	EC	TBS	SDGS	NSGDI
USAR	100000	5000	150	0.002	10	0.5	0.5	0.01	8192	256	4
Ski	100000	5000	500	0.001	10	0.5	0.5	0.01	4196	256	4
Pacman	10000	500	150	0.001	10	0.5	0.5	0.012	1000	100	10
USAR 10 Rubble	100000	5000	150	0.002	10	0.5	0.5	0.01	8192	256	4

Table 2: RL Training Hyper-parameters

B Reward Similarity Computation

For USAR, each room is composed of 3 by 3 grids, the connection between rooms take 1 grid, rubble, victim, and the agent all occupy 1 grid. Moving to the wall grid, removing non-rubble grid, and healing non-victim grid are considered

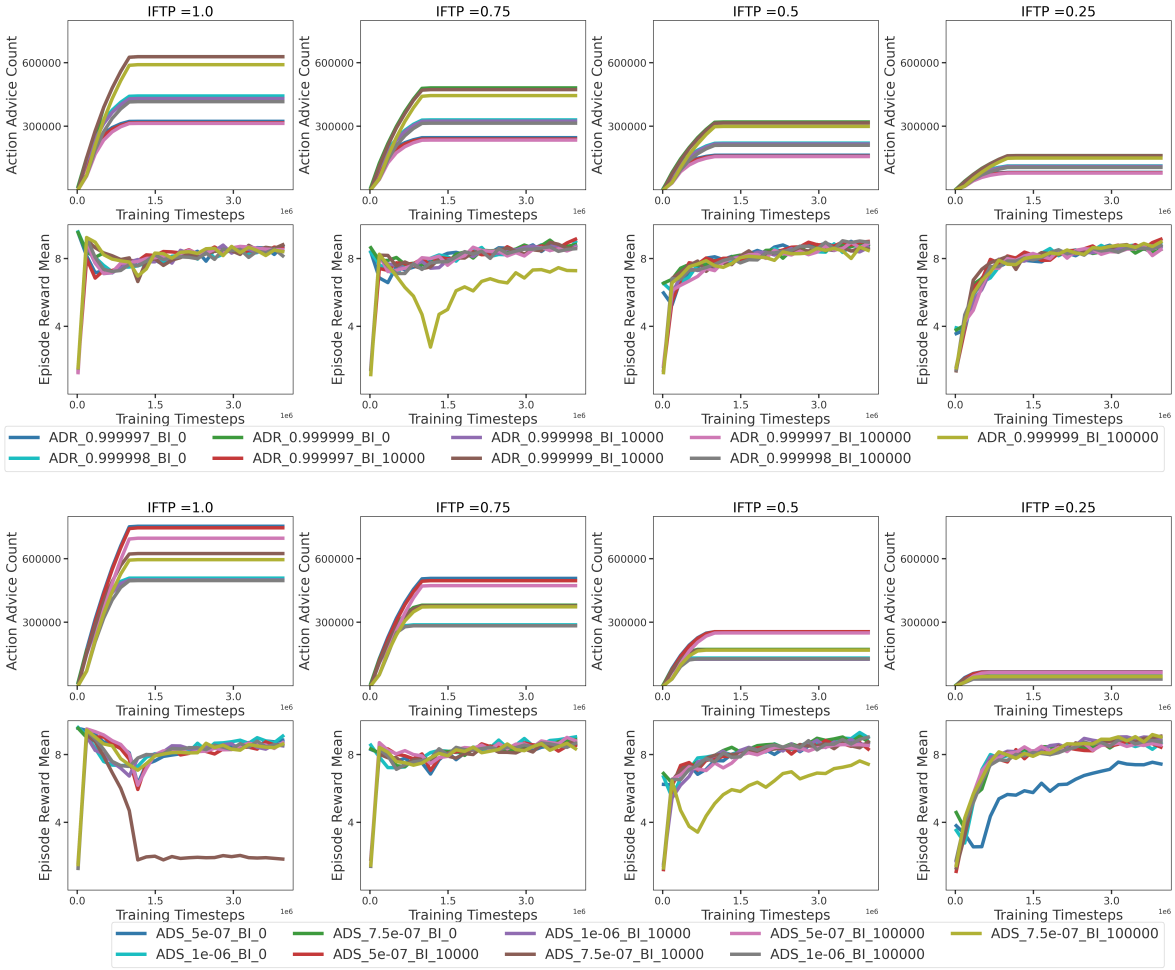


Figure 9: Hyper-parameters

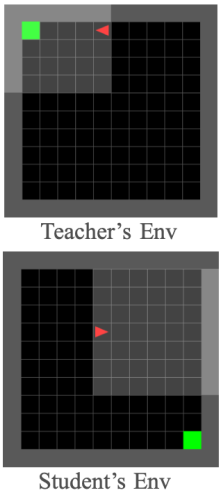


Figure 10: Environments of the teacher and student in the bidirectional navigation task.

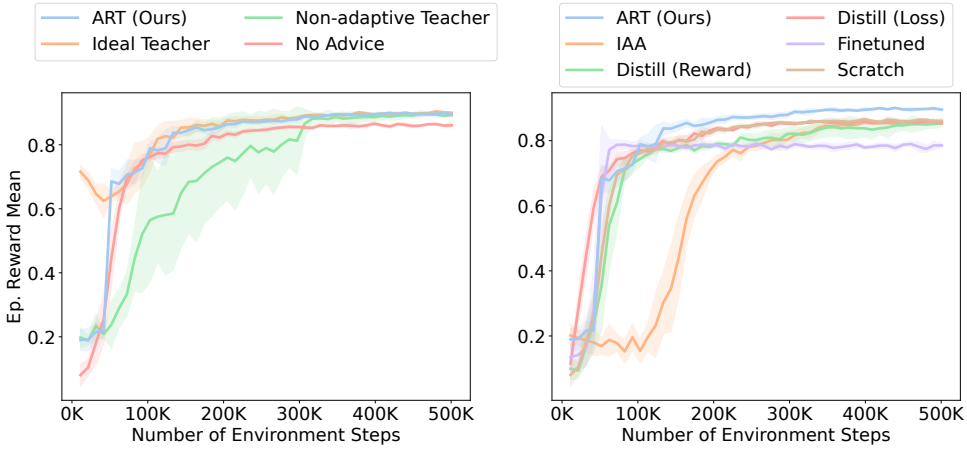


Figure 11: Baseline and benchmark results of the bidirectional navigation task

invalid. For each time the environment is reset, rubble, victim, and the agent are all relocated randomly. Therefore, $D_{(USAR)} = 0.5/422 = 0.0011848$ and $D_{\text{norm}_{(USAR)}} = 1/422 = 0.0023697$.

For Ski, there are 8 gates total, and the student received 10X reward for passing the gate compared to the teacher's +10000. The agent for each time step may choose left or right actions till the borderlines of 10 steps horizontally, and vertically it takes 600 steps. Therefore, $D_{(Ski)} = 720000/10800 = 66.67$ and $D_{\text{norm}_{(Ski)}} = 8/10800 = 0.000741$.

For Pacman, the teacher is an aggressive ghost eater, which means eating ghosts with a reward of 5000 may compensate other potential negative effects such as not finishing eating up foods or eaten by the ghosts when timer counts down to zero. There are 280 non-wall grids. The pacman may move only to non-wall grids, and can eat ghosts when has eaten one of the power pellets within 50 steps. We calculate with a probabilistic approach for this complex dynamics, where the teacher eats 6.4 ghosts on average. This leads to $D_{(Pacman)} = 5000 * 6.4/280 = 114$ and $D_{\text{norm}_{(Pacman)}} = 6.4/280 = 0.02286$.

C BIDIRECTIONAL NAVIGATION

We further present the results of an additional environment. In a grid environment, a square room is partitioned into 10x10 grids, serving for a navigation learning task. The source task involves a teacher agent navigating towards the top-left corner of the grid room. Conversely, the target task requires the student agent to reach an opposite goal, the bottom-right corner. The agent is initialized randomly in the room whenever the environment is reset to resample. This setup highlights a challenge when the source and target tasks have similar forms of reward functions that reaching the goal with a positive reward, but the ideal behaviors are completely opposite. Figure 10 shows the environment, where the green square is the target for the agent to navigate to, and the red triangle is the agent, whose angle points to the direction facing with. For both environments, the only positive reward of 1 is given when the agent reaches the target, discounted over steps.

The teacher agent excels within its native environment, requiring 15 steps on average to reach its goal, achieving a 100% success rate and a discounted reward of 0.89 over 100 trials. This performance contrasts starkly with its performance in the student's environment when not adapted, leading to an average of 85 steps to reach the target and a reduced average reward of 0.14. This elucidates the challenges as the ideal behaviors are completely opposite. Even though the goal is on the bottom right corner, the non-adaptive teacher still keeps navigating towards its previous goal on the top left corner.

Similar to what we have done for the other three environments of USAR, Ski, and Pacman, we plot the baseline and benchmark results of this task. The baseline of the result in shown in Figure 11 (left). As the teacher's advice is for

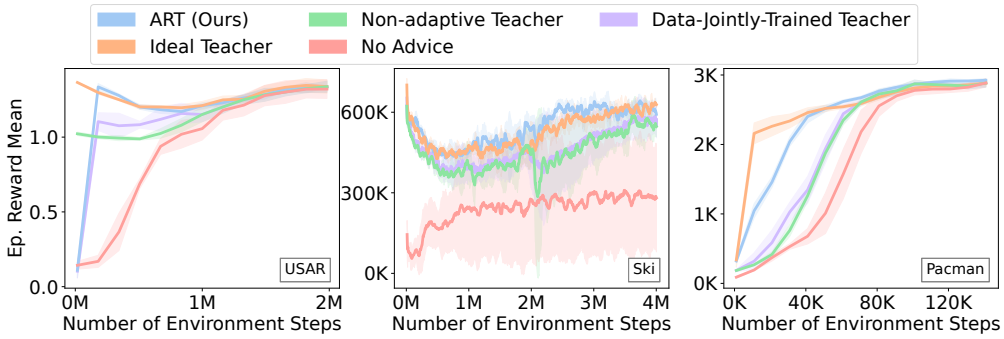


Figure 12: Compare with teacher using both its data and the collected student data to train its reward network.

navigating to the opposite direction of the student's target, the teacher's advice can be harmful to the student's training. Therefore, the student advised by the non-adaptive teacher is worse than no advice for a long time, until finally the teacher stops advising and the student is trained by its own to exceed no-advice. Notably, the teacher's advice is not completely useless, as during the beginning of the student's training, student advised by the teacher who has not improved is still better than the student exploring with no advice. Additionally, student advised by any type of the teacher finally is slightly better than the student with no advice. For our ART method, the teacher quickly finetunes its policy to reach the performance of the ideal teacher, and then avoids giving the misleading advice anymore. Therefore, the student with ART is always better than the student with no advice over training.

The benchmark result is shown in Figure 11 (right). The difficulty of transfer can be further assured with the method of simple finetune, as the agent with a policy network that leads to the opposite behavior can be hardly changed. Thus the performance of the finetune method is the worst. Other benchmark methods may avoid this issue, however, our ART method has the best performance. This shows our ART is more robust in this scenario where teacher and student have opposite rewards that lead to opposite ideal behaviors.

D Joint Data Training

We further conduct an experiment that trains a reward model jointly on teacher-collected data and student-collected data. For this data-jointly trained teacher, no error prediction model is used. We plot the results over the Figure 12. As shown in the figure, in all three environments, the new setting only allows the student to be trained slightly better than being advised by a non-adaptive teacher.

This is because the newly collected student data can hardly override the teacher data. To be more specific, in the USAR scenario, the teacher receives 0 as its reward when removing rubble, but when it collects student data, this becomes 0.5 instead. In the Ski scenario, the reward is 10x bigger when collected by the student, but from what the teacher has previously, it is not that significant. In the Pacman scenario, the reward of eating ghost as received by the teacher is much bigger than with the student. With data jointly trained, the reward network can hardly discern that is expected by the student, and handle the conflicts of reward values. Therefore, the teacher is only slightly improved, as the reward it estimates is still close to its own reward. This is no doubt worse than our method of ART where the estimation of the student reward is more accurate.