Seminar: Advanced Deep Reinforcement Learning – Adversarially Guided Actor Critic –

Lukas Welzel [1] **Koen Ponse** [1]

Abstract

Reinforcement learning has been a very active subdomain of AI research, but some environments still pose a significant challenge. Here we will report on AGAC, a recently proposed reinforcement learning algorithm for sparse reward environments. We were able to reproduce the reported performance of AGAC, propose an adaptation to include noise in its adversarial agent and also adapt AGAC to include recurrence as its method of memory. Our results show that the recurrence hindered performance massively. However, the inclusion of noise in the adversarial agent showed minor performance improvements in one of our experiments. As the performance did not drop for other environments, one may want to experiments with adding noise to its own agents for a possible minor performance improvement, as it does not require massive changes in any algorithm.

1. Introduction

Reinforcement Learning (RL) has emerged as a powerful approach for solving complex decision-making problems in various domains, including robotics, game-playing, and autonomous systems. However, the success of RL algorithms is often hindered by the challenges of hard exploration, sparse rewards, and partial visibility. Hard exploration refers to the problem of discovering optimal actions in a large state space, where only a small fraction of states are visited during training. Sparse rewards, on the other hand, refer to the difficulty of providing informative feedback to the agent, where the reward signal is infrequent or only available at the end of an episode. Finally, partial visibility refers to the challenge of incomplete or noisy observations of the environment, which limits the agent's ability to learn a comprehensive policy. (Ladosz et al., 2022; Wang et al., 2022; Hare, 2019) In some problems, external rewards may not be easily defined or not available at all, and it may be necessary for the agent to learn from intrinsic rewards that arise from the environment or the task itself. This approach is called intrinsic motivation or curiosity-driven learning, where the agent is motivated to explore the environment and learn new skills for the sake of novelty or curiosity. (Pathak et al., 2017) Without effective strategies for dealing with hard exploration, sparse rewards, and partial visibility, current RL agents fail to learn effective policies and take an impractically long time to converge. (Matheron et al., 2019) The relevance of these challenges to real-life problems is apparent in a wide range of applications, from robotics to finance and healthcare, see e.g. Vecerik et al. (2017); Ibarz et al. (2021), Li (2017), and Yu et al. (2021) respectively. Hence, the further development of methods that can handle sparse or delayed rewards in partially visible and changing environments has the potential to enable significant advances in a wide range of domains. (Ladosz et al., 2022) This work explores a recent approach which adds a new protagonist to the actor-critic algorithm and leverages adversarial surprise to improve exploration (Flet-Berliac et al., 2021).

First, in section 2 we give a short overview of the related work. Then, in section 3 we introduce the new actor critic (AC) method and describe the experimental setup in section 4. We report the results in section 5 and discuss the new method in the context of the existing work in section 6.

2. Background & Related Work

Several methods have been proposed to address the challenges of sparse rewards. In the following an overview of the state-of-the-art for hard-exploration is given. Raileanu & Rocktäschel (2020) propose RIDE which makes use of an intrinsic reward that encourages the agent to take actions that are predicted to change a learned state representation, resulting in a more sample-efficient approach for procedurally-generated environments. In contrast, RAPID regards each episode as a whole and gives an episodic exploration score, and stores highly scored episodes in a small ranking buffer to reproduce past good exploration behaviors (Zha et al., 2021). COUNT (Bellemare et al., 2016) uses density models to measure uncertainty and derives a pseudo-count from an arbitrary density model to transform it into intrinsic rewards, resulting in significantly improved exploration in Atari games. (Burda et al., 2018b) proposes random network distillation (RND) as an exploration bonus for DRL, which uses the error of a fixed randomly initialized neural network that predicts features of the observations to encour-

age the agent to explore new states. Nikulin et al. (2023) show that RND might not be sufficiently discriminative to be used in offline RL and improve RND by introducing Feature-wise Linear Modulation conditioning. Pathak et al. (2017) build an intrinsic curiosity module (ICM). They understand curiosity as the error in predicting the consequence of actions using an inverse dynamics model. Contrary to most follow-up work in RL, Li et al. (2019) propose a simplified version of ICM, called S-ICM, which removes the inverse dynamics model and replaces it with a simple forward model which still outperforms non-IM methods for hard exploration problems. Burda et al. (2018a) survey curiosity-driven learning on Atari games. Their results show a surprising degree of alignment between the IC objectives and extrinsic rewards. They show that learned features are needed for generalizing in procedurally generated, stochastic environments compared to random features. Campero et al. (2020) propose AMIGo which uses adversarially motivated intrinsic goals to train a goal-conditioned policy. The agent proposes increasingly challenging goals which are predicted to be still achievable to train the policy. Colas et al. (2022) propose a framework for goal-conditioned RL and present a typology of the various goal representations used in the literature, and review existing methods to learn to represent and prioritize goals in autonomous systems. Finally, Fickinger et al. (2021) propose Adversarial Surprise for high-dimensional, stochastic environments based on an adversarial game between two policies. One part visits states that surprise the other, which tries to return to predictable states. This dynamic leads to exploring increasingly challenging parts of the environment. Still, sparse rewards and partial observability continues to be a problem in DRL with no method archiving the desired convergence speed or accuracy.

3. Method

Actor-Critic (AC) methods have enjoyed large popularity for reinforcement learning due to their ability to speed up learning due to lower variance and good convergence. AC consist of two parts an actor and critic which are, a policy $\pi(\theta)$ and value function $V(\phi)$ respectively. (Konda & Tsitsiklis, 1999; Konda & Borkar, 1999) Recently, soft-actor-critic (SAC) has been introduced to reduce brittleness and convergence of off-policy DRL methods for stochastic environments. (Haarnoja et al., 2018a;b) Our implementation uses proximal policy optimization (PPO, Schulman et al. (2017)) policy gradients to update the policy parameters θ and the finite time-horizon temporal difference scheme to update the value function parameters ϕ, see Equation 1. We will indicate the deep policy gradient scheme for the actor and critic with subscript *PG* and *V* respectively.

$$V_\phi \leftarrow \min_\phi \sum_{t'=t}^{t+T} \left[V_\phi(t') - A_{t'}^{AGAC} - V_{\phi_{old}}(s_{t'}) \right], \quad (1)$$

where T is the (finite) time horizon, $A_{t'}^{AGAC}$ is a modification of the generalized advantage estimator (GAE, see Equation 10 and Equation 9, Schulman et al. (2015)), and ϕ_{old} is the previous parameter set of the value function.

We add a third player, called the adversary, that increases the diversity of sampled trajectories. While the task of the actor is to select good actions under the local policy $\pi(\theta)$, and the critic assesses actions through its value function $V(\phi)$, the adversary predicts the actions taken by the actor under the current policy π. This means that the adversary learns a delayed policy $\pi_{adv}(\psi) \leftarrow \pi(\theta_{old})$ with parameters ψ. This policy is used to differentiate the actor from the adversary by giving the actor an action dependent bonus of $\log \pi(a_t \mid s_t) - \log \pi_{adv}(a_t \mid s_t)$ via the advantage function, and an action independent D_{KL} bonus, see Equation 2, for the value function and the adversary evaluated on the divergence of its policy π_{adv} and an old actor policy $\pi(\theta_{old})$. In the following we will continue to indicate old (policy) parameterizations and the adversary with an *old* and *adv* subscript respectively.

$$D_{\mathrm{KL}}\left[\pi(\cdot \mid s) \| \pi_{\mathrm{adv}}(\cdot \mid s)\right] \equiv D_{\mathrm{KL}}\left(\pi \| \pi_{\mathrm{adv}}, s\right) \quad (2)$$

$$= \pi(\cdot \mid s)\left[\log\{\pi(\cdot \mid s) \cdot \pi_{adv}^{-1}(\cdot \mid s)\}\right] \quad (3)$$

$$= \mathbb{E}_{\pi(\cdot|s)}\left[\log \pi(\cdot \mid s) - \log \pi_{adv}(\cdot \mid s)\right], \quad (4)$$

the Kullback-Leibler divergence over all actions, $(\cdot \mid s)$ or for the critic and adversary. Note the abbreviation introduced in the first line of Equation 2. The AGAC minimizes the composite loss in Equation 5, where $\mathcal{L}$ are the respective losses, and β is a constant weighting factor for the critic and adversary loss. For clarity we will indicate function parameterizations with subscripts in the following so that e.g. $\pi(\theta) \equiv \pi_\theta$.

$$\mathcal{L}_{AGAC} = \mathcal{L}_{PG} + \beta_V \mathcal{L}_V + \beta_{adv} \mathcal{L}_{adv} \quad (5)$$

We show the dependence of AGAC functions on the different (policy and value function) parametrizations in Table 1.

3.1. Actor

The policy gradient surrogate objective $\mathcal{L}_{PG}$, in Equation 6, is modified from the clipped policy gradient loss to include the AGAC bonus in the new advantage A_t^{AGAC} in Equation 9.

$$\mathcal{L}_{PG} = -\frac{1}{T}\sum_{t=0}^{T}\Big[A_t^{AGAC} \log\{\pi_\theta(a_t \mid s_t)\} + \alpha \mathcal{H}^{\pi_\theta}(s_t)\Big], \quad (6)$$

Table 1. Parametrizations used by selected functions in AGAC. The parameters that are associated with the (sub-) objectives are indicated in bold.

Function	$\mathcal{L}_{PG}$	$\mathcal{L}_V$	$\mathcal{L}_{adv}$	A_t^{AGAC}	A_t	$\mathcal{H}$
Parameters	$\boldsymbol{\theta}$: θ_{old}, ϕ_{old}, ψ_{old}	$\boldsymbol{\phi}$: θ_{old}, ϕ_{old}, ψ_{old}	$\boldsymbol{\psi}$: θ_{old}	θ_{old}, ϕ_{old}, ψ_{old}	ϕ_{old}	θ

where α is the entropy coefficient, and $\mathcal{H}^{\pi_\theta}(s) = \mathbb{E}_{\pi(\cdot|s)}\left[-\log\{\pi(\cdot \mid s)\}\right]$ is the entropy of policy $\pi(\theta)$ at state s. In practice the parallel, clipped PPO loss $\mathcal{L}_{PG}^{CLIP}$ is used, see Equation 7, Equation 8, where the surrogate loss estimated the objective using N actors take T-step trajectories. (Schulman et al., 2017; Wang et al., 2020)

$$\mathcal{L}_{PG}^{CLIP} = -\hat{A}_t \min\left[r_t(\theta), \text{clip}\left|r_t(\theta)\right|_{1-\epsilon}^{1+\epsilon}\right] \tag{7}$$

$$= -\frac{1}{T}\sum_{t'=t}^{t+T} A_{t'}^{AGAC} \min\left[\frac{\pi^{\theta}\left(a_{t'} \mid s_{t'}\right)}{\pi^{\theta_{old}}\left(a_{t'} \mid s_{t'}\right)}, \text{clip}\left|\frac{\pi^{\theta}\left(a_{t'} \mid s_{t'}\right)}{\pi^{\theta_{old}}\left(a_{t'} \mid s_{t'}\right)}\right|_{1-\epsilon}^{1+\epsilon}\right], \tag{8}$$

where ϵ is the clipping parameter introduced to increase the stability of the surrogate objective, which is a conservative policy iteration objective for AGAC. (Kakade & Langford, 2002) With the GAE A_t (Schulman et al., 2015) in Equation 10, the AGAC advantage A_t^{AGAC} in Equation 9 gets a D_{KL} bonus scaled by an annealed hyperparameter c.

$$A_t^{AGAC} = A_t + c\left[\log\{\pi^{\theta_{old}}\left(a_t \mid s_t\right)\} - \log\{\pi_{adv}^{\psi_{old}}\left(a_t \mid s_t\right)\}\right] \tag{9}$$

$$A_t = \sum_{t'=t}^{t+T} (\gamma\lambda)^{[t'-t]}\left[r_{t'} + \gamma V_{\phi_{\text{old}}}\left(s_{t'+1}\right) - V_{\phi_{\text{old}}}\left(s_{t'}\right)\right], \tag{10}$$

where $r_t \in \mathcal{R}$ is the reward at step t, $\gamma \in [0, 1)$ is the discount factor, and $\lambda \in [0, 1]$ is the GAE bias-variance trade-off parameter.

3.2. Critic

The objective for the critic $\mathcal{L}_V$ is given by Equation 11.

$$\mathcal{L}_V = \frac{1}{T}\sum_{t=0}^{T}\left[V_\phi\left(s_t\right) - V_{\phi_{old}}\left(s_t\right) - A_t - cD_{\text{KL}}\left(\pi^{\theta_{old}} \| \pi_{\text{adv}}^{\psi_{old}}, s_t\right)\right]^2 \tag{11}$$

3.3. Adversary

The objective for the adversary $\mathcal{L}_{adv}$ is given by Equation 12 which is simply the estimated KL-divergence of the policies.

$$\mathcal{L}_{\text{adv}} = \frac{1}{T}\sum_{t=0}^{T} D_{\text{KL}}\left(\pi^{\theta_{old}} \| \pi_{\text{adv}}^{\psi}, s_t\right). \tag{12}$$

3.4. Random Adversary

We extend the AGAC architecture by a random action predictor. This agent will take random actions that the actor needs to avoid. The bonus for avoiding the randomly predicted actions for the actor is the same as for the learned adversary. In this way we aim to push the actor away from uniform action distributions while not otherwise altering the shape of the overall objective. We expect the impact to be similar to injecting randomness in the action selection via Gibbs sampling and help the actor escape flat regions in the reward space by pushing it away from any uniform equilibrium. (Yu, 2018) Nevertheless, while we expect this to in effect decrease the bias of a single trace, this is probably traded against an increase in variance over many runs. We did not explore the interaction of this randomness injection with the entropy maximization term from SAC, due to the expected increase in training time when entropy maximization is removed. (Haarnoja et al., 2018a)

3.5. Frame-stacking or Recurrence

Since the environment is causal, temporally coherent and partially observable we expect that a form of memory is essential for good performance. Flet-Berliac et al. (2021) implement simple frame stacking of depth 4, however we extend this idea to use recurrence so that the agent can learn a memory efficient history of its exploration. We implement this using LSTM blocks. (Gruslys et al., 2017; Muskardin et al., 2022)

3.6. Interpretation of AGAC

The approach of AGAC differs from the other adversarial learning approaches in DRL, such as 1) the follow-up paper by Flet-Berliac & Basu (2022) where the components of AC are altered to include an adversary actor and critic, with the adversary critic estimating failure probability to make the idae of AGAC suitable for safe RL, 2) classic adversaries in e.g. Pan et al. (2019); Bucher et al. (2021)

that counteract the actor by periodically taking control, and 3) recent work on adversarial surprise by Fickinger et al. (2021) that simultaneously avoid low-entropy regions and seek novel transitions. This means that AGAC, together with AMIGo Campero et al. (2020), are poorly understood in theory and thorough interpretations are missing from their papers. Flet-Berliac et al. (2021), find that AGAC favours transitions which are less accurately predicted than average, see Equation 13, and traded against expected reward where the equilibrium point is annealed during learning. Furthermore they separate the objective in a policy iteration scheme and find that dynamics neatly separate, see Equation 14, however their analysis does not consider the PPO implementation.

$$c\left[\log \pi^{\theta_{old}} - \log \pi^{\psi_{old}}_{adv}\right] \geq D_{\mathrm{KL}}\left(\pi^{\theta_{old}} \| \pi^{\psi}_{\mathrm{adv}}\right) \tag{13}$$

$$\begin{aligned}\mathcal{J}_{\mathrm{PI}}(\pi) = \mathbb{E}_s \Bigg[& \underbrace{-cD_{\mathrm{KL}}\left(\pi(\cdot \mid s) \| \pi_k(\cdot \mid s)\right)}_{\pi_k \text{ is attractive}} \\ & \underbrace{+cD_{\mathrm{KL}}\left(\pi(\cdot \mid s) \| \pi_{adv}(\cdot \mid s)\right)}_{\pi_{adv} \text{ is repulsive}} \\ & \underbrace{+\alpha \mathcal{H}(\pi(\cdot \mid s))}_{\text{enforces stochastic policies}} \Bigg]\end{aligned} \tag{14}$$

3.6.1. Information-Theoretic Interpretation of AGAC Objectives

Recently Aubret et al. (2023) performed a survey of IM in RL from an information-theoretic background in which they introduce a scheme to categorize IM and intrinsic rewards (IR) in the context of information-theory. The entropy $H(X)$ or Shannon information measure of a random variable $x \sim X$ with density $p(X)$ is given by Equation 15 and its conditional entropy $H(X \mid Y)$ on jointly distributed random variable $y \sim Y$, with joint $p(x, y)$ by Equation 16, where we consider the general case. The information required to infer X based on knowledge of Y is given by this conditional entropy. (Bell & Sejnowski, 1995; Williams, 2013)

$$H(X) = -\mathbb{E}_{x\sim X}\left[\log p(x)\right] \tag{15}$$

$$H(X \mid Y) = -\mathbb{E}_{x,y\sim X,Y}\left[\log p(x \mid y)\right] \tag{16}$$

The mutual information $I(X;Y)$ of the two variables, defined as Equation 17, is a measure of the statistical independence of the variables. Similar to Equation 16, the conditional mutual information $I(X;Y \mid Z)$ in Equation 18 is a measure of the information that can be inferred from one random variable about another one given knowledge of a third random variable (here Z). (Aubret et al., 2019) This (conditional) mutual information forms the basis for the scheme by Aubret et al. (2023).

$$I(X;Y) = H(X) - H(X \mid Y) \tag{17}$$

$$\begin{aligned} I(X;Y \mid Z) &= H(X \mid Z) - H(X \mid Y,Z) && (18)\\ &= H(Y \mid Z) - H(Y \mid X,Z) && (19)\\ &= D_{KL}[p(X,Y \mid Z) \| p(X \mid Z)p(Y \mid Z)] && (20)\end{aligned}$$

$$D_{KL}\left[p(X) \| p(Y)\right] = H(X,Y) - H(X) \tag{21}$$

For IM the reward typically tries to quantify information *gain* over some model conditioned on a dataset $\mathcal{D}$. The information gain (I_G) over a true, forward or density model is Equation 22. (Little & Sommer, 2013; Aubret et al., 2023)

$$IG(\mathcal{D},\mathcal{A},\mathcal{S}',\mathcal{S},\theta) = I(\mathcal{S}';\theta \mid \mathcal{D},\mathcal{A},\mathcal{S}) \tag{22}$$

$$\begin{aligned} &= \mathop{\mathbb{E}}_{\substack{(s,a)\sim p(\cdot \mid \mathcal{D}),\\ s'\sim p(\cdot \mid s,a,\mathcal{D})}} D_{KL}(p(\theta \mid \mathcal{D},s,a,s') || p(\theta \mid \mathcal{D})) \\ &\approx \mathop{\mathbb{E}}_{s'\sim p(\cdot \mid s,a,\mathcal{D},\theta_T)} D_{KL}(p(\theta \mid \mathcal{D},s,a,s') \| p(\theta \mid \mathcal{D}))\end{aligned} \tag{23}$$

where θ_T is the true parametrization, assuming a single one exists, of the environment or a distribution over such parametrizations otherwise. If such a true model exists, Equation 22 can be written as the expected information gain over the true model in Equation 25. This relation requires knowledge of the true model which is generally not known.

$$I(\mathcal{S}';\theta_T \mid \mathcal{D},\mathcal{A},\mathcal{S}) = H(\theta_T \mid \mathcal{D},\mathcal{A},\mathcal{S}) - H(\theta_T \mid \mathcal{D},\mathcal{A},\mathcal{S},\mathcal{S}') \tag{24}$$

$$= \mathop{\mathbb{E}}_{\substack{(s,a)\sim p(\cdot \mid \mathcal{D}),\\ \theta_T\sim p(\cdot)}} \log p(s' \mid s,a,\mathcal{D},\theta_T) - \log p(s' \mid s,a,\mathcal{D}) \tag{25}$$

Nevertheless, for a forward model Equation 22 can be rewritten as Equation 26 (Aubret et al., 2023) which shows that the AGAC PG loss $\mathcal{L}_{PG}$ together with the adversary loss $\mathcal{L}_{adv}$ means that AGAC uses a hybrid objective in the IM scheme, however it is not clear how this is changed by using the clipped PG loss $\mathcal{L}^{CLIP}_{PG}$ since the relation to π^{θ} is changed with the log-term in Equation 27. Nevertheless, the clipped loss terms should still be an approximation of Equation 26. (Schulman et al., 2017)

$$\begin{aligned} &I(\mathcal{S}';\theta \mid \mathcal{D},\mathcal{A},\mathcal{S}) \\ &= H(\mathcal{S}' \mid \mathcal{D},\mathcal{A},\mathcal{S}) - H(\mathcal{S}' \mid \mathcal{A},\theta,\mathcal{S},\mathcal{D})\end{aligned} \tag{26}$$

$$\begin{aligned} \mathop{\mathbb{E}}_{\substack{\theta\sim p(\cdot \mid \mathcal{D},s,a,s'),\\ \theta_T\sim p(\cdot)\\ (s,a)\sim p(\cdot \mid \mathcal{D}),\\ s'\sim p(\cdot \mid s,a,h,\theta_T)}} -\log \sum_{\vartheta\in\theta} \Bigg[& p(s' \mid \theta,\mathcal{D},s,a)p(\theta \mid \mathcal{D}) \\ & + \log p(s' \mid s,a,\theta,\mathcal{D}) \Bigg],\end{aligned} \tag{27}$$

where ϑ is a forward model parametrization. AGAC and the method proposed by Fickinger et al. (2021) are not examined by Aubret et al. (2023), however they also highlight that the information perspective for exploration in RL is cur-

rently underdeveloped and several proofs are left for future work. This includes showing if the convexity of the D_{KL} for density models ρ in $p\left(\rho \mid \mathcal{D}, s'\right)$ and $p(\rho \mid \mathcal{D})$ leads to a uniform distribution in s' which Flet-Berliac et al. (2021) rely on for convergence.[1] While AGAC does not fit neatly into the scheme proposed by Aubret et al. (2023) (see also Jarrett et al. (2022)) it helps explain parts of the AGAC behaviour by comparing its objectives to Equation 22. From Equation 6 and Equation 9 the AGAC actor relates to information gain since it attempts to maximize over its current model (see the surrogate in Equation 8), however the mirrored D_{KL} term in Equation 9, see Equation 28 from Flet-Berliac et al. (2021) for $\lambda = 1$, mean that the actor is attracted to regions where dynamics are poorly understood by the model or that are stochastic, while being repelled by regions where the transition model predicts that the novelty of these states will trade-off unfavourably with the expected reward.

$$A_t^{AGAC} \propto \log \pi^{\theta}\left(a_t \mid s_t\right) - \log \pi_{adv}\left(a_t \mid s_t\right) - \hat{D}_{KL}^{\phi_{old}}\left(\pi^{\theta} \| \pi_{adv}^{\phi}, s_t\right), \tag{28}$$

where $\hat{D}_{KL}$ is the expected value of the KL divergence. Comparing the AGAC objective with information gain over a density model, see Equation 29, as suggested by Bellemare et al. (2016) which is directly used for the adversary and is related to count-based novelty estimation. (Bellemare et al., 2016)

$$IG\left(\mathcal{D}, \mathcal{A}, \mathcal{S}', \mathcal{S}, \mathcal{P}\right) \tag{29}$$

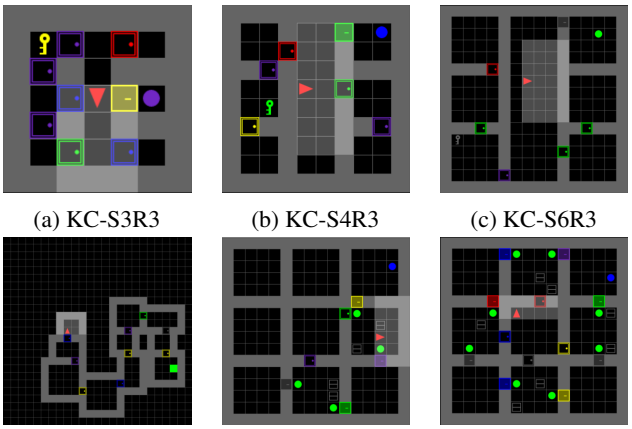
$$\approx \mathop{\mathbb{E}}_{\substack{(s,a)\sim p(\cdot|h),\\ \rho_T \sim p(\cdot)}} D_{KL}\left(p\left(\rho \mid h, s'\right) \| p(\rho \mid h)\right), \tag{30}$$

where $\mathcal{P}$ is the transition function. Naturally, the AGAC objective also includes the direct entropy term from SAC, which pushes the agent to novel states. (Hazan et al., 2019) From an information theory perspective AGAC combines several of the most performant objectives and uses information gain of the actor over a mix of its old policies and the adversary policy as a density model to construct a composite objective that should outperform each of its parts by combining their advantage; good exploration and robustness around stochastic or poorly-understood regions by exploiting the conditional mutual information gain of actor over adversary on their knowledge of the transition dynamics at the cost of memory and compute. Finally, the information gain perspective helps identify the prospect of examining other repulsive terms for mutual information between the actor and adversary like Jensen-Rényi divergence to help diversify AGAC in flat mutual information regions. (Osorio et al., 2022; Yang et al., 2021) This would be similar to Shyam et al. (2019) for model-based RL exploration.

[1] Compare §4.1-4.2 in Flet-Berliac et al. (2021) and §5.4 in Aubret et al. (2023).

4. Experiments

This section will first describe the MiniGrid (Chevalier-Boisvert et al., 2018), the environment in which we performed our experiments. We follow up with reproduction results of the original work (Flet-Berliac et al., 2021). In Section 4.3, we elaborate on our proposed extensions to AGAC, which are previously introduced in Sections 3.4 and 3.5.



(a) KC-S3R3 (b) KC-S4R3 (c) KC-S6R3

(d) MR-N10-S6 (e) OM-2Q (f) OM-Full

Figure 1. Various examples of procedurally generated MiniGrid environments in different challenges. Figures 3d, 3c, 3f show the KeyCorridor challenge, in which the agent is spawned in a hallway with rooms on either side. The goal is to pick up the ball, which may be blocked by closed doors for which a key is found in some other room. Each of the three figures depicts an increase in difficulty, in which the rewards are even sparser due to the increased grid size. Figure 1d depicts the MultiRoom environment in which the agent simply has to navigate through a number of rooms within a maximum amount of timesteps. The reward is obtained only when reaching the final goal. Figures 1e and 1f show the ObstructedMaze challenge in which the agent has to find a goal that may be hidden in a box. Doors are also blocked by balls. These environments represent the greatest challenge as reaching the goal may require a lot of different and very deliberate steps.

4.1. Environment

The MiniGrid library (Chevalier-Boisvert et al., 2018) is a collection of grid-world environments with a Gym-like API, perfectly suited for research in Reinforcement learning. The library consists of a wide variety of grid world challenges in which an agent (depicted as a triangle) has to navigate a 2D grid world and reach a final goal within a given amount of timesteps. Extrinsic rewards are only obtained if the final goal is reached, but the agent has to navigate its way through various obstacles (closed doors, walls, balls) while performing small sub-tasks which may or may not lead to the final goal (picking up keys, opening doors, moving balls). Each of the challenges is tune-able in their difficulty, which is useful for curriculum learning or simply comparing

performance over a wider variety of algorithms. A few examples of different environments are shown in Figure 1. in the first three images (3d, 3c, 3f), we can see the same challenge (KeyCorridor, KC) in three different difficulties. As the difficulty increases, the rewards get sparser.

During training and testing, the MiniGrid environments are procedurally generated according to the rules and difficulty of the challenge. Furthermore, in our tests (and that of Flet-Berliac et al. (2021)), the agent only observes a maximum of 7 squares in front of it. As such, it may benefit to somehow remember information gained in previous states.

4.2. Reproduction

The AGAC paper (Flet-Berliac et al., 2021) showed a gigantic leap in performance sparse reward environments. Earlier work (Raileanu & Rocktäschel, 2020; Bellemare et al., 2016; Burda et al., 2018b; Pathak et al., 2017) often did not even perform experiments on some of the more difficult environments of MiniGrid which AGAC seemingly was able to solve without much issue. AGAC was also able to beat the reported performance of AMIGO (Campero et al., 2020) which was, at the time of publication, the best-performing algorithm. It should be noted however, that true comparisons could not be made as the results of AMIGO were not reproducible by us or the authors of AGAC.

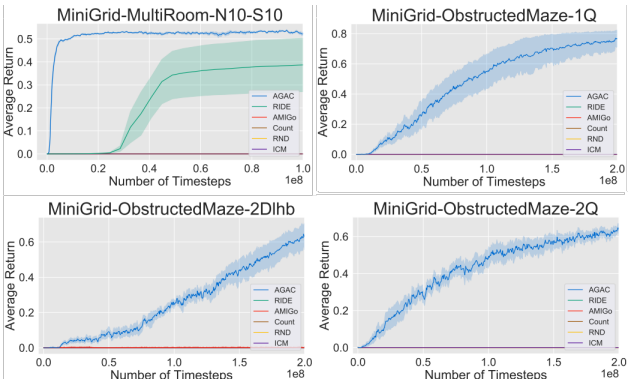


Figure 2. Some of the original results on various MiniGrid environments as reported by the authors of AGAC (Flet-Berliac et al., 2021). These results show a very large performance increase compared to the earlier state-of-the-art. Oftentimes, AGAC is the only algorithm managing to obtain any rewards. All of these experiments are performed by the authors themselves, as the original authors had usually not included any results on these difficulty settings for these environments. A few more original results on different environments (KeyCorridor) are shown in Figure 3.

Due to these extremely high-performing reported results, which are shown in Figure 2, it is key that they are indeed reproducible. As such, we set out to validate some of the reported results before conducting our own experiments.

All code for the original AGAC implementation is available online on GitHub[2]. Both TensorFlow and PyTorch versions are made available, but the original implementation was created in TensorFlow, which is the version we will use for the reproduction experiments. We chose to reproduce the experiments on a subset of the environments on which the AGAC results are presented. As is the case for the original experiments, we averaged our results over a total of 6 repetitions. In most environments, we used a lower amount of total timesteps as it was clear AGAC had reached its maximum potential and stopping early saved us days worth of compute time.

4.3. Noisy agents

As discussed in Section 3.4, we propose to extend the AGAC architecture by including randomness in the adversary. We had originally set out to alter the algorithm to include a third, completely random, adversary. Small-scale experiments made it apparent however, that this implementation likely only hindered performance and we lacked the theoretical basis on how to create a new loss function to properly make use of this extra adversary. However, during the experiments, it turned out that adding noise to the original adversary was seemingly quite fruitful.

We experimented with different amounts of noise added to the output of the adversary, both uniformly or normally distributed around the original mean of the output. We also tried replacing the adversary output with random noise entirely, as how our third adversary would have functioned. While we did not expect this completely random method to work at all, we were surprised to see the algorithm still managing to obtain rewards in environments in which previous state-of-the-art algorithms were not able to generate any. Nevertheless, this random method was performing substantially worse than our methods that added uniformly or normally generated noise to the original output. As such, we decided to perform larger-scale experiments on a wider variety of Minigrid environments for these methods. The uniformly generated noise was generated between $-1e^{-3}$ and $1e^{-3}$ and the results can be found in Section 5.2.

4.4. Recurrence and Framestacking

As previously described in Section 3.5, learning in the Minigrid environments likely benefits from some sort of memory in the model. For instance, in our Minigrid experiments, the agent can only observe a 7x7 grid of squares in front of itself. This introduces the need to remember previous observed states. For example, it would be beneficial to remember the location of a *blue door*, so that the agent may return to its location after finding the *blue key*.

[2] https://github.com/yfletberliac/adversarially-guided-actor-critic

The original AGAC implementation solves this problem by stacking the last 4 states as input to the CNN model (frame stacking). The 4 frames used by AGAC may not always be enough in an environment such as MiniGrid. Rather than simply increasing the amount of stacked frames, it may instead be more efficient to store selected information (doors, keys, balls) over all the information in previous frames. To this end, we propose to include recurrent layers in the model, which we will implement using LSTM blocks. This may yield a more optimal result as LSTM blocks can selectively store important information in its hidden states. The results of the experiments with the new recurrent layers can be found in Section 5.3, where we also experimented with different sizes of framestacking.

5. Results

5.1. Reproduction

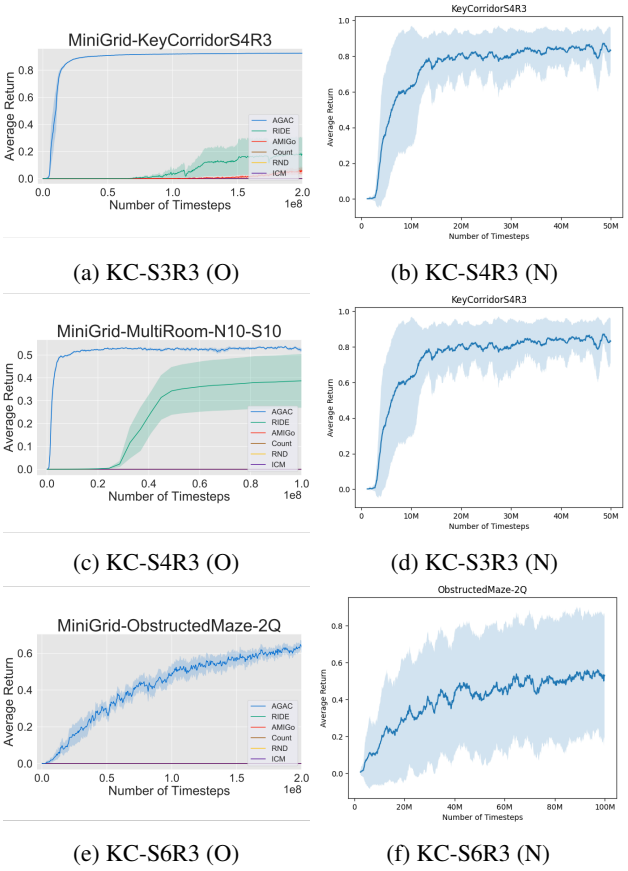
Our reproduction results can be found in Figure 3. While the original results possibly seemed too good to be true, we were able to successfully reproduce the AGAC results in a variety of environments. Only on rare occasions we observed AGAC to not gain any rewards, but this was likely due to unlucky seeds, as these poor results are washed out by the average over multiple runs.

5.2. Noisy agents

Figure 4 displays our final results with two methods of adding noise to the agent in different environments, averaged over 6 repititions and compared to the original method. We observe that adding uniformly generated noise can hurt performance, as shown in Figures 4a and 4c with the orange line performing worse. However, using normally distributed noise (normally distributed around the original output by the AGAC adversary) we can see a small performance increase. Unfortunately, while we see some performance improvement in *KC-S4R3*, we see none in all other environments. As such, we can at best conclude that adding normally distributed noise will at best aid performance slightly and will most likely not hinder it. We further observe a very large standard deviation in these experiments across the different runs. This may have been what led us to believe during smaller-scale experiments that there was indeed a consistent performance improvement for the noise-added method.

5.3. Recurrence and Framestacking

Here we altered the original model to include recurrence, which we implemented with LSTM blocks. Unfortunately, we observed an extreme drop in performance and were unable to obtain any rewards during our tests on KCS6R3 after 40 million timesteps of training. Our implementation may have simply required more training steps, but as we also observed a large increase in computation time we do not believe a recurrent variant of AGAC would be a reasonable adaptation, due to the already high computational requirements by the original implementation.



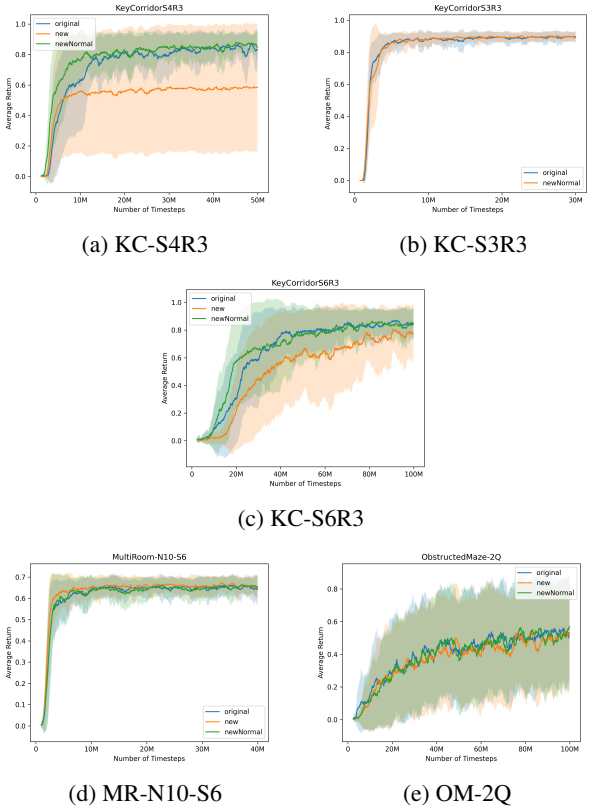
(a) KC-S3R3 (O) (b) KC-S4R3 (N)

(c) KC-S4R3 (O) (d) KC-S3R3 (N)

(e) KC-S6R3 (O) (f) KC-S6R3 (N)

Figure 3. The original results presented in the AGAC paper on the left (O) and our reproduction of their experiments on the same environments on the right (N). We trained for a substantial amount of fewer timesteps as it was already clear the performance was as reported. We can see our learning curves follow a similar trajectory to the original curves, meaning we can validate the original results. In our graphs, the transparent fill represents the standard deviation of our 6 runs. The original AGAC paper, while not stated explicitly, likely uses this same fill to represent the uncertainty of the mean.

We also experimented with different framestack sizes, for which the results can be found in Figure 5. As expected, a single frame (no memory) scores worse, but does reasonably well on this environment. As the single-frame approach does beat previous state-of-the-art performance, we can conclude that, in these environments, the method of exploration is more important compared to the method of memory. Using two frames achieves similar results compared to the original and may be preferred because of the smaller input (and thus less computational requirements). However, more difficult environments may need the extra frames, so we cannot apply these results to all situations. Adding frames beyond 4 hurt



(a) KC-S4R3 (b) KC-S3R3

(c) KC-S6R3

(d) MR-N10-S6 (e) OM-2Q

Figure 4. Various Minigrid environments comparing original AGAC to our method with noise added uniformly between $-1e^{-3}$ and $1e^{-3}$ (new) and our method with the noise generated normally distributed around the original output (newNormal). We observe that the uniformly added noise hurts performance, but the normally distributed noise actually increases performance in KC-S4R4 (a). However, this is not the case for any of the other environments.

performance, but again, more complex environments may benefit from the extra frames. For MiniGrid, the original number 4 looks like a good compromise.

6. Discussion

AGAC is clearly a significant improvement over other SOTA methods for large MiniGrid environments. Only the adverserial method by Fickinger et al. (2021) exceeds/matches AGAC for less complex versions of MiniGrid. This is because AGAC fuses several information gain measures that complement each other, see subsubsection 3.6.1. Other methods using IG over density models are not robust to changing environments and are computationally expensive (E.g. using Bayesian NN to find the density model). AGAC builds its density model jointly with its actor and is robust against procedural environments due to using a bonus proportional to IG over a forward model. To test this interpretation AGAC should be ablated in stochastic environments to show the impact of the different IG measures.

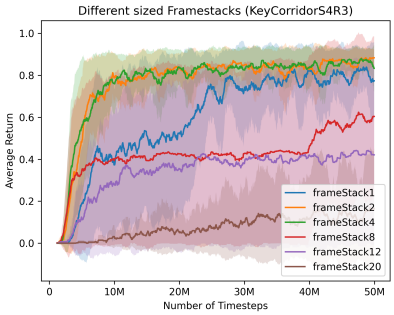


Figure 5. AGAC with different framestack sizes on the KCS4R3 environment. The original number of 4 frames seems to be a good compromise. We observe that, in this particular environment, two frames would be enough, but an increase hurts performance. As expected, no frame stacking (value of 1) hurts performance.

The random adversary is a mechanism to trade bias against variance, as can be seen from Figure 4, however this is probably a disadvantageous trade due to the high computational cost of training AGAC. Furthermore, the random adversary does not lead to IG, and the associated (count-based) novelty gain is likely negligible over entropy maximization. Yet, to our surprise a completely random adversary still achieved better performance than prior state-of-the-art algorithms. Future work may want to investigate this further, but should be warned of the large amount of time required to perform enough experiments. As highlighted in subsubsection 3.6.1 the adversary implicitly estimates of how *safe* exploration of poorly understood regions is against a predicted reward. This makes the work of Flet-Berliac & Basu (2022) a natural extension of AGAC for safe RL.

7. Conclusion

AGAC showed a gigantic leap in performance in Minigrid, a RL environment with sparse rewards and partial observability. We were able to successfully reproduce the results reported in the original paper (Flet-Berliac et al., 2021). We extended the work by introducing noise to the AGAC adversary and found minimal performance increases in some environments if the noise was normally distributed around the original output. Performance drops were not observed and as such it may be worthwhile to experiment with introducing noise to algorithms similar to AGAC, but this will likely only yield minimal benefits at best.

We further experimented with recurrent layers in the original AGAC model, instead of its default framestacking approach. This lead to the model no longer obtaining any rewards and a further slowdown of the training process. Adjusting the number of frames used by the framestacking approach yielded no benefits either in our experiments. Whilst simpler environments may find it enough to only include two frames, the originally used number of 4 looks like a good default.

References

Aubret, A., Matignon, L., and Hassas, S. A survey on intrinsic motivation in reinforcement learning. *arXiv preprint arXiv:1908.06976*, 2019.

Aubret, A., Matignon, L., and Hassas, S. An information-theoretic perspective on intrinsic motivation in reinforcement learning: a survey. *Entropy*, 25(2):327, 2023.

Bell, A. J. and Sejnowski, T. J. An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159, 1995.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Bucher, B., Schmeckpeper, K., Matni, N., and Daniilidis, K. An adversarial objective for scalable exploration. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2670–2677. IEEE, 2021.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018a.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018b.

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., and Grefenstette, E. Learning with amigo: Adversarially motivated intrinsic goals. *arXiv preprint arXiv:2006.12122*, 2020.

Chevalier-Boisvert, M., Willems, L., and Pal, S. Minimalistic gridworld environment for gymnasium, 2018. URL https://github.com/Farama-Foundation/Minigrid.

Colas, C., Karch, T., Sigaud, O., and Oudeyer, P.-Y. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022.

Fickinger, A., Jaques, N., Parajuli, S., Chang, M., Rhinehart, N., Berseth, G., Russell, S., and Levine, S. Explore and control with adversarial surprise. *arXiv preprint arXiv:2107.07394*, 2021.

Flet-Berliac, Y. and Basu, D. Saac: Safe reinforcement learning as an adversarial game of actor-critics. *arXiv preprint arXiv:2204.09424*, 2022.

Flet-Berliac, Y., Ferret, J., Pietquin, O., Preux, P., and Geist, M. Adversarially guided actor-critic. *arXiv preprint arXiv:2102.04376*, 2021.

Gruslys, A., Dabney, W., Azar, M. G., Piot, B., Bellemare, M., and Munos, R. The reactor: A fast and sample-efficient actor-critic agent for reinforcement learning. *arXiv preprint arXiv:1704.04651*, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018a.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Hare, J. Dealing with sparse rewards in reinforcement learning. *arXiv preprint arXiv:1910.09281*, 2019.

Hazan, E., Kakade, S., Singh, K., and Van Soest, A. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pp. 2681–2691. PMLR, 2019.

Ibarz, J., Tan, J., Finn, C., Kalakrishnan, M., Pastor, P., and Levine, S. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5):698–721, 2021.

Jarrett, D., Tallec, C., Altché, F., Mesnard, T., Munos, R., and Valko, M. Curiosity in hindsight. *arXiv preprint arXiv:2211.10515*, 2022.

Kakade, S. and Langford, J. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 267–274, 2002.

Konda, V. and Tsitsiklis, J. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Konda, V. R. and Borkar, V. S. Actor-critic–type learning algorithms for markov decision processes. *SIAM Journal on control and Optimization*, 38(1):94–123, 1999.

Ladosz, P., Weng, L., Kim, M., and Oh, H. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 2022.

Li, B., Lu, T., Li, J., Lu, N., Cai, Y., and Wang, S. Curiosity-driven exploration for off-policy reinforcement learning methods. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1109–1114. IEEE, 2019.

Li, Y. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

Little, D. Y. and Sommer, F. T. Learning and exploration in action-perception loops. *Frontiers in neural circuits*, 7: 37, 2013.

Matheron, G., Perrin, N., and Sigaud, O. The problem with ddpg: understanding failures in deterministic environments with sparse rewards. *arXiv preprint arXiv:1911.11679*, 2019.

Muskardin, E., Tappler, M., Aichernig, B. K., and Pill, I. Reinforcement learning under partial observability guided by learned environment models. *arXiv preprint arXiv:2206.11708*, 2022.

Nikulin, A., Kurenkov, V., Tarasov, D., and Kolesnikov, S. Anti-exploration by random network distillation. *arXiv preprint arXiv:2301.13616*, 2023.

Osorio, J. K. H., Skean, O., Brockmeier, A. J., and Giraldo, L. G. S. The representation jensen-rényi divergence. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4313–4317. IEEE, 2022.

Pan, X., Seita, D., Gao, Y., and Canny, J. Risk averse robust adversarial reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8522–8528. IEEE, 2019.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Raileanu, R. and Rocktäschel, T. Ride: Rewarding impact-driven exploration for procedurally-generated environments. *arXiv preprint arXiv:2002.12292*, 2020.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shyam, P., Jaśkowski, W., and Gomez, F. Model-based active exploration. In *International conference on machine learning*, pp. 5779–5788. PMLR, 2019.

Vecerik, M., Hester, T., Scholz, J., Wang, F., Pietquin, O., Piot, B., Heess, N., Rothörl, T., Lampe, T., and Riedmiller, M. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

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*, 2022.

Wang, Y., He, H., and Tan, X. Truly proximal policy optimization. In *Uncertainty in Artificial Intelligence*, pp. 113–122. PMLR, 2020.

Williams, L. Geometric and probabilistic methods in computer science, 2013. https://www.cs.unm.edu/~williams/cs530f15.html.

Yang, Z., Qu, H., Fu, M., Hu, W., and Zhao, Y. A maximum divergence approach to optimal policy in deep reinforcement learning. *IEEE Transactions on Cybernetics*, 2021.

Yu, C., Liu, J., Nemati, S., and Yin, G. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

Yu, Y. Towards sample efficient reinforcement learning. In *IJCAI*, pp. 5739–5743, 2018.

Zha, D., Ma, W., Yuan, L., Hu, X., and Liu, J. Rank the episodes: A simple approach for exploration in procedurally-generated environments. *arXiv preprint arXiv:2101.08152*, 2021.