ADVERSARIALLY GUIDED ACTOR-CRITIC

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite definite success in deep reinforcement learning problems, actor-critic algorithms are still confronted with sample inefficiency in complex environments. Many of those methods consider an actor and a critic, whose respective losses are obtained with different motivations and approaches. In this paper, we introduce a third protagonist, the adversary, whose task is to match the action probability distribution of the actor. While the adversary minimizes the KL-divergence between its action distribution and the actor policy, the actor maximizes the log-probability difference between its action and that of the adversary in combination with maximizing expected rewards. This novel objective stimulates the actor to follow strategies that could not have been correctly predicted from previous trajectories, making its behavior innovative in tasks where the reward is extremely rare. Our experimental analysis shows that the resulting Adversarially Guided Actor-Critic (AGAC) algorithm leads to more exhaustive exploration. Notably, AGAC outperforms current state-of-the-art methods on a set of various hard-exploration and procedurally-generated tasks.

1 INTRODUCTION

Research in deep reinforcement learning (RL) has proven to be successful across a wide range of problems (Silver et al., 2014; Schulman et al., 2016; Lillicrap et al., 2016; Mnih et al., 2016b). Nevertheless, generalization and exploration in RL still represent key challenges that leave most current methods ineffective. First, a battery of recent studies (Farebrother et al., 2018; Zhang et al., 2018a; Song et al., 2019; **?**) indicates that current RL methods fail to generalize correctly even when agents have been trained in a diverse set of environments. Second, exploration has been extensively studied in RL, however most hard-exploration problems use the same environment for training and evaluation. Hence, since a well-designed exploration strategy should maximize the information received from a trajectory about an environment, if that information is memorized then the exploration capabilities may not be properly assessed. In this line of research, we choose to study the exploration capabilities of our method and its ability to generalize to new scenarios. Our evaluation domains will therefore be tasks with sparse reward in procedurally-generated environments.

In this work, we propose Adversarially Guided Actor-Critic (AGAC), which reconsiders the actor-critic framework by introducing a third protagonist: the adversary. Its role is to predict the actor's actions correctly. Meanwhile, the actor must, in addition to finding the optimal actions to maximize the sum of expected returns, also counteract the adversary's predictions. This formulation is inspired by adversarial methods and specifically generative adversarial networks (GANs) (Goodfellow et al., 2014). In this context, the adversary can be interpreted as playing the role of the discriminator by predicting the actions of the actor. As to the actor, it can be seen as playing that of the generator by being led to behave in a way that will fool the predictions of the adversary. Similar to GANs, the advantage of the approach is that the optimization procedure generates a diversity of meaningful data, corresponding to relevant sequences of actions in the context of our method.

AGAC explicitly drives diversity in the behaviors of the agent, while at the same time remaining reward-focused through the combined objective of outsmarting the adversary and maximizing returns. This approach also proves to be useful as a way to adapt to the evolving state space of procedurally-generated environments. There is no such thing as a free lunch, and stability is a legitimate concern which we tackle in the following sections, since specific instances of GANs were shown (Arjovsky & Bottou, 2017) to be prone to instability issues or brittle to hyper-parameter changes.

The contributions of this work are as follow: (i) we propose a novel actor-critic formulation inspired from adversarial learning, (ii) we analyse theoretically and empirically AGAC on key reinforcement learning aspects such as diversity, exploration and stability, (iii) we demonstrate significant gains in performance on several sparse-reward hard-exploration tasks including procedurally-generated tasks.

2 Related Work

The scientific interest for actor-critic methods (Barto et al., 1983; Sutton, 1984) was renewed when Mnih et al. (2016a) proposed to combine deep function approximation and multiple distributed actors to an actor-critic, demonstrating strong results on Atari. Since then, many additions have been proposed, be it architectural improvements (Vinyals et al., 2019), better advantage estimation (Schulman et al., 2016), or the incorporation of transition selection or off-policy elements (Schaul et al., 2015; Wang et al., 2016; Oh et al., 2018; Flet-Berliac & Preux, 2020). A notable research direction is the use of regularization to improve actor-critic methods, either to enforce trust regions (Schulman et al., 2015; 2017; Wu et al., 2017), or to correct for off-policiness (Munos et al., 2016; Gruslys et al., 2017). Incidentally, some works analyzed the impact of regularization from a theoretical standpoint (Geist et al., 2019; Ahmed et al., 2019; Vieillard et al., 2020).

While introduced in the context of supervised learning, adversarial learning was taken advantage of in several RL works. Ho & Ermon (2016) propose an imitation learning method that makes use of a discriminator whose task is to distinguish between expert trajectories and those of the agent while the agent tries to imitate expert behavior to fool the discriminator. Bahdanau et al. (2018) use a discriminator to distinguish goal states from non-goal states based on a textual instruction, and use the resulting model as a reward function. In the goal-oriented setting, Held et al. (2017) use a GAN to produce sub-goals at the right level of difficulty for the current agent, inducing a form of curriculum. In addition, Pfau & Vinyals (2016) provide a parallel between GANs and the actor-critic framework.

Exploration is a key aspect of RL. While exploration is driven in part by the core RL algorithms (Fortunato et al., 2017), it is often necessary to resort to exploration specific techniques. One of these is intrinsic motivation, which complements the rewards from the environment by bonuses that reward explorative behavior from the agent. Among intrinsic motivation methods, some work use state-visitation counts or pseudo-counts as a way to promote exhaustive exploration (Bellemare et al., 2016a), while others reward curiosity, expressed in the magnitude of prediction error from the agent, to push it towards unfamiliar areas of the state space (Burda et al., 2018). Ecoffet et al. (2019) propose a technique akin to tree traversal to explore the environment, while learning to come back to promising areas. Eysenbach et al. (2018) show that creating and maintaining diversity in the policy space helps with exploration, even when there is no reward from the environment.

Generalization in RL is a key challenge. Zhang et al. (2018b) showed that even when the environment is not deterministic, agents can overfit their training distribution, and that it is difficult to distinguish agents that will generalize to new environments from those that will not, based on loss curves. In the same vein, recent work has advocated the use of procedurally-generated environments, in which a new instance of the environment is sampled when a new episode starts, to better assess generalization capabilities (Justesen et al., 2018; Cobbe et al., 2019). Finally, methods based on network randomization (Igl et al., 2019), noise injection (Lee et al., 2020), and credit assignment (Ferret et al., 2020) have been proposed to reduce the generalization gap for RL agents.

3 Background and Notations

We consider Markov Decision Problems (Puterman, 1994) defined by a tuple $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma\}$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the transition kernel, $\mathcal{R}$ is a bounded reward function and $\gamma \in [0, 1)$ is a discount factor. Let $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$denote a stochastic policy. We place ourselves in the infinite-horizon setting, *i.e.*, we seek a policy that optimizes $J(\pi) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$.The value of a state is defined by $V^\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s]$ and the value of a state-action pair $Q^\pi(s, a)$ of performing action a in state s and then following policy π is defined as: $Q^\pi(s, a) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a]$. The advantage function quantifies how better than the average an action a is in state s is defined by $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. Finally, the entropy $\mathcal{H}^\pi$ of a policy is $\mathcal{H}^\pi(s) = \mathbb{E}_{a_t \sim \pi}[-\log \pi(a_t|s_t) | s_0 = s]$.

PP : c'est quoi cette no-ta-tion avec P cal-ligraphié

Actor-Critic and Deep Policy Gradients. An actor-critic algorithm is composed of two main components: a policy and a value predictor. In deep RL, both the policy and the value function are obtained via parametric estimators, we denote θ and ϕ their respective parameter. The policy is updated via policy gradient, while the value is usually updated via temporal difference. In practice, for a sequence of transitions $\{s_t, a_t, r_t, s_{t+1}\}_{t\in[0,N]}$, we use the following policy gradient loss (including the commonly used entropic penalty):

$$\mathcal{L}_{PG} = \frac{1}{N}\sum_{t'=t}^{t+N} A_{t'}\nabla_\theta \log \pi\left(a_{t'}|s_{t'},\theta\right) - \alpha\mathcal{H}^\pi(s_{t'},\theta),$$

where α is the entropy coefficientand A_t is the generalized advantage estimator defined as: $A_t = \sum_{t'=t}^{t+N}(\gamma\lambda)^{t'-t}(r_{t'} + \gamma V_\phi(s_{t'+1}) - V_\phi(s_{t'}))$ with λ a fixed hyper-parameter. To estimate the value function, we solve the non-linear regression problem $\text{minimize}_\phi \sum_{t'=t}^{t+N}(V_\phi(s_{t'}) - \hat{V}_{t'})^2$ where $\hat{V}_t = A_t + V_{\phi_{\text{old}}}(s_{t'})$ with $V_{\phi_{\text{old}}}$ the value function estimator at the previous iteration.

PP : plutôt qu'entropy coefficient qui ne précise rien, je dirais plutôt $\alpha \in$? intervalle ?

4 Method: KL-regularized Adversarial Actor-Critic

The spirit of our approach is to promote more diversified behaviors in the policy gradient by introducing a third protagonist to the actor-critic framework that should correctly predict the actor's actions. The adversary is therefore optimized to *minimize* the discrepancy between the distribution of the actor policy π and the adversary policy π_{adv}. Meanwhile, the actor must, in addition to finding the optimal actions to *maximize* the sum of expected returns, also counteract the adversary's predictions by *maximizing* the latter discrepancy between π and π_{adv} (see Appendix D for an illustration). This discrepancy, used as a form of exploration bonus, is defined in terms of the Kullback–Leibler divergence between the two distributions:

$$D_{\text{KL}}(\pi(\cdot|s)\|\pi_{\text{adv}}(\cdot|s)) = \mathbb{E}_s\left[\log\pi(\cdot|s) - \log\pi_{\text{adv}}(\cdot|s)\right].$$

On the one hand, the adversary is an independent neural network whose weights are updated with a lower learning rate than that of the target for his predictions: the actor network. In this way, the adversary network weights represent a delayed and more stable version of the actor network weights. On the other hand, the actor achieves behavior that optimizes the extrinsic reward function and the log ratio between the sampled actor and adversary distributions. Formally, for each state-action pair (s_t, a_t) in a trajectory, a sampled bonus $\log\pi(a_t|s_t) - \log\pi_{\text{adv}}(a_t|s_t)$ is given to the actor. In addition, the critic includes the KL divergence bonus $D_{\text{KL}}(\pi(\cdot|s_t)\|\pi_{\text{adv}}(\cdot|s_t))$ in the value function estimator. Intuitively, when the policy is not optimal and assuming that the critic has a low estimation error, a positive advantage function in `AGAC` implies that there exist better state-action pairs than expected for a state that will do a better job in (i) maximizing the expected sum of returns and (ii) diverging from the adversarial distribution. In the next section , we discuss the implications of a divergence from π_{adv}. In addition to the parameters θ and ϕ defined above (resp. θ_{old} the parameter of the policy at the previous iteration and ϕ_{old} that of the critic), we denote ψ (resp. ψ_{old}) that of the adversary.

`AGAC` optimizes the following loss:

$$\mathcal{L}_{\texttt{AGAC}} = \mathcal{L}_{\text{PG}} + \beta_V\mathcal{L}_V + \beta_{\text{adv}}\mathcal{L}_{\text{adv}}.$$

In the new objective $\mathcal{L}_{PG} = \frac{1}{N}\sum_{t=0}^{N} A_t^{\texttt{AGAC}}\nabla_\theta\log\pi\left(a_t|s_t,\theta\right) - \alpha\mathcal{H}^\pi(s_t,\theta)$, `AGAC` modifies A_t as:

$$A_t^{\texttt{AGAC}} = A_t + c\left(\log\pi(a_t|s_t,\theta_{\text{old}}) - \log\pi_{\text{adv}}(a_t|s_t,\psi_{\text{old}})\right) \tag{1}$$

with c a hyper-parameter that controls the dependence on the log-action probability difference. $\mathcal{L}_V$ is the objective function of the critic defined as:

$$\mathcal{L}_V = \frac{1}{N}\sum_{t=0}^{N}\left(V_\phi(s_t) - \hat{V}_t - c\, D_{\text{KL}}(\pi(\cdot|s_t,\theta_{\text{old}})\|\pi_{\text{adv}}(\cdot|s_t,\psi_{\text{old}}))\right)^2. \tag{2}$$

Finally, $\mathcal{L}_{\text{adv}}$ is the objective function of the adversary:

$$\mathcal{L}_{\text{adv}} = \frac{1}{N}\sum_{t=0}^{N} D_{\text{KL}}(\pi(\cdot|s_t,\theta_{\text{old}})\|\pi_{\text{adv}}(\cdot|s_t,\psi)). \tag{3}$$

Eq. 1, 2 and 3 are the three equations that our method modifies (red highliths these modifications) in the traditional actor-critic framework. β_V and β_{adv} are fixed hyper-parameters.

4.1 Building Motivation

With this new gradient formulation, the actor log-probability of a sample action sequence is reinforced if (i) the corresponding reward is large or (ii) the adversarial bonus is large, which means that the action could not be predicted from the behavior of the actor in previous trajectories. Mechanistically, our method disfavors transitions whose actions were better predicted than the average of all other actions, *i.e.*, $\log \pi(a|s, \theta_{\text{old}}) - \log \pi_{\text{adv}}(a|s, \psi_{\text{old}}) \leq D_{\text{KL}}(\pi(\cdot|s'_t, \theta_{\text{old}}) \| \pi_{\text{adv}}(\cdot|s'_t, \psi_{\text{old}})$.

In the following we provide an interpretation of the new objective formulation in terms of game of attraction and repulsion between the actor and the adversary. In a simplified case and under the spectrum of policy iteration, let us ignore advantage functions and instead consider q-functions. The quantity of interest is (for simplicity, π_k is analogous to π with parameter θ_{old} in previous notations):

$$\tilde{q}_{\pi_k} = q_{\pi_k} + c(\log \pi_k - \log \pi_{\text{adv}})$$

with π_k the policy at iteration k. The new policy optimizes for (including the entropic penalty):

$$\mathcal{L}_{PI}(\pi) = \mathbb{E}_s \mathbb{E}_{a \sim \pi(\cdot|s)} [\tilde{q}_{\pi_k}(s, a) - \alpha \ln \pi(a|s)]$$

We can rewrite this loss:

$$\begin{aligned}
\mathcal{L}_{PI}(\pi) &= \mathbb{E}_s \mathbb{E}_{a \sim \pi(\cdot|s)} [\tilde{q}_{\pi_k}(s, a) - \alpha \log \pi(a|s)] \\
&= \mathbb{E}_s \mathbb{E}_{a \sim \pi(\cdot|s)} [q_{\pi_k}(s, a) + c(\log \pi_k(a|s) - \log \pi_{\text{adv}}(a|s)) - \alpha \log \pi(a|s)] \\
&= \mathbb{E}_s \mathbb{E}_{a \sim \pi(\cdot|s)} [q_{\pi_k}(s, a) + c(\log \pi_k(a|s) - \log \pi(a|s) + \log \pi(a|s) - \log \pi_{\text{adv}}(a|s)) - \alpha \log \pi(a|s)] \\
&= \mathbb{E}_s \Big[\mathbb{E}_{a \sim \pi(\cdot|s)} [q_{\pi_k}(s, a)] \underbrace{-cD_{\text{KL}}(\pi(\cdot|s)||\pi_k(\cdot|s))}_{\pi_k \text{ is attractive}} \underbrace{+cD_{\text{KL}}(\pi(\cdot|s)||\pi_{\text{adv}}(\cdot|s))}_{\pi_{\text{adv}} \text{ is repulsive}} \underbrace{+\alpha \mathcal{H}(\pi(\cdot|s))}_{\text{enforce stocastic policies}} \Big]
\end{aligned}$$

We describe the behavior of the different parts of the equation above. In this simplified formulation AGAC would compute a greedy policy, maximizing q-values, while at the same time remaining close to the previous policy but far from a form of mixture of the policies that preceded the previous one (*i.e.*, $(\pi_{k'})_{k' \in \{0, \dots, k-1\}}$). Notice that we experimentally observe (*cf.* Section 5.3) that our method performs better when the learning rate of the adversarial network is lower than that of the other networks; nevertheless, it is not a sensitive parameter hence we can consider an ideal (with same learning rates) case where $\pi_{\text{adv}} = \pi_{k-1}$ in which the policy would be repulsive from π_{k-1}.

Thisoptimization problem is strongly convex in π (thanks to the entropy term), and is a Legendre-Fenchel transform. Its solution is given by:

$$\pi_{k+1} \propto \left(\frac{\pi_k}{\pi_{\text{adv}}} \right)^{\frac{c}{\alpha}} \exp \frac{q_{\pi_k}}{\alpha}.$$

This gives us some insight into the behavior of the objective function. Notably, in our example, if π_{adv} is fixed, we recover the framework of a KL-regularization with a modified reward $r - c \log \pi_{\text{adv}}$.

4.2 Implementation

In all the experiments, we use PPO (Schulman et al., 2017) as the base algorithm and build on it to incorporate our method. We choose PPO for its leading performance on a set of popular tasks and its algorithmic simplicity. Hence,

$$\mathcal{L}_{PG} = \frac{1}{N} \sum_{t'=t}^{t+N} \min \left(\frac{\pi(a_{t'}|s_{t'}, \theta)}{\pi(a_{t'}|s_{t'}, \theta_{\text{old}})} A_{t'}^{\texttt{AGAC}}, \text{clip}\Big(\frac{\pi(a_{t'}|s_{t'}, \theta)}{\pi(a_{t'}|s_{t'}, \theta_{\text{old}})}, 1 - \epsilon, 1 + \epsilon \Big) A_{t'}^{\texttt{AGAC}} \right),$$

with $A_{t'}^{\texttt{AGAC}}$ given in Eq. 1, N the temporal window considered for one update of parameters and ϵ the usual PPO clipping parameter. Similar to RIDE, we also discount PPO by episodic state visitation counts, except for Vizdoom (*cf.* Section 5.1). The actor, critic and adversary use the convolutional architecture of the Nature paper of DQN (Mnih et al., 2015) with different hidden

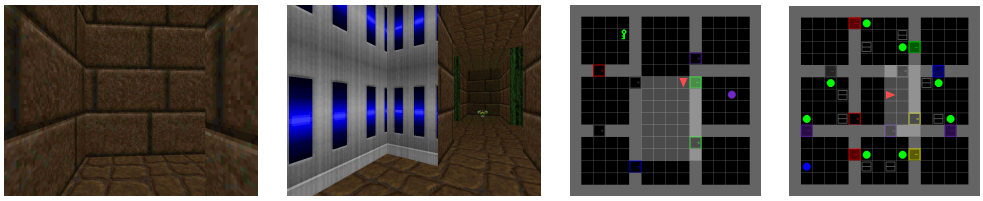
sizes (see Appendix C for architecture details). The three neural networks are optimized using Adam (Kingma & Ba, 2015). Our method does not use RNNs in its architecture; instead, in all our experiments, we use frame stacking. Indeed, Hausknecht & Stone (2015) interestingly demonstrate that although recurrence is a reliable method for processing state observation, it does not confer any systematic advantage over stacking observations in the input layer of a CNN. Note that the parameters are not shared between the policy, the critic and the adversary and that we did not observe any noticeable difference in computational complexity when using AGAC compared to the baseline. We direct the reader to Appendix B for a list of hyper-parameters. In particular, the c coefficient behind the adversarial bonus is linearly annealed.

At each training step, we perform a stochastic optimization step to minimize $\mathcal{L}_{\texttt{AGAC}}$ using stop-gradient:

$$\theta \leftarrow \text{Adam}\left(\theta, \nabla_\theta \mathcal{L}_{PG}, \eta_1\right)$$
$$\phi \leftarrow \text{Adam}\left(\phi, \nabla_\phi \mathcal{L}_{V}, \eta_1\right)$$
$$\psi \leftarrow \text{Adam}\left(\psi, \nabla_\psi \mathcal{L}_{\text{adv}}, \eta_2\right).$$

5 Experiments

In this section, we describe our experimental study of AGAC in which we investigate: (i) whether the adversarial bonus alone is sufficient to outperform other methods in Vizdoom, a sparse-reward task with high-dimensional observations, (ii) whether AGAC succeeds in partially-observable and procedurally-generated environments with extremely sparse rewards, compared to other methods, (iii) how well AGAC is capable of exploring in environments without extrinsic reward, (iv) the training stability of AGAC. In all plots, lines are average performances and shaded areas represent one standard deviation around the average. We include the code for the method in the Supplementary Material.



(a) (b) (c) (d)

Figure 1: Illustrations of the tasks being considered: (a,b) frames from the 3-D navigation task VizdoomMyWayHome. (c) MiniGrid-KeyCorridorS6R3. (d) MiniGrid-ObstructedMazeFull.

Environments. To carefully evaluate the performance of AGAC, its ability to develop robust exploration strategies and its generalization to unseen states, we choose tasks that have been used in prior works, which are tasks with high-dimensional observations, sparse reward and procedurally-generatedenvironments (see Fig. 1 for some examples). In **Vizdoom** (Kempka et al., 2016), the agent must learn to move along corridors and through rooms without any reward feedback from the environment. The **MiniGrid** environments (Chevalier-Boisvert et al., 2018) are a set of partially-observable and extremely-sparse-reward gridworlds. In this type of procedurally-generated environments, memorization is impossible due to the huge size of the state space, so the agent must learn to generalize across the different layouts of the environment. Each gridworld has different characteristics: in the MultiRoom tasks, the agent is placed in the first room and should reach a goal that is located in the most distant room. In the KeyCorridor tasks, the agent must navigate to pick up an object located in a room locked by a door whose key is in another room. Finally, in the ObstructedMaze tasks, the agent must pick up a box that is located in a corner of a 3x3 maze in which the doors are also locked, the keys are hidden in boxes and the doors are obstructed by balls. These environments are available as part of OpenAI Gym (Brockman et al., 2016).

Baselines. To assess AGAC, we compare its performance to those of some of the most prominent methods specialized in hard-exploration tasks: **RIDE** (Raileanu & Rocktäschel, 2019), based on an intrinsic reward associated with the magnitude of change between two consecutive state representations and state visitation, **Count** as Count-Based Exploration (Bellemare et al., 2016b), which we

PP : c'est bizarre de citer cette tâche ici : pourquoi la citer ? plutôt parler de ce qui fait qu'on l'a choisie.

PP : AGAC ?

PP : comprends pas ctete phrase : je ne vois pas de code dans le

couple with IMPALA (Espeholt et al., 2018), **RND** (Burda et al., 2018) in which an exploration bonus is positively correlated to the error of predicting features from the observations and **ICM** (Pathak et al., 2017) where an Intrinsic Curiosity Module only predicts the changes in the environment that are produced by the actions of the agent. Finally, we compare AGAC to most of the recent and best performing methods at the time of writing in procedurally-generated environments: **AMIGo** (Campero et al., 2020) in which a goal-generating teacher provides count-based intrinsic goals. While avoiding unnecessary use of computational resources, we make the fairest comparison with other methods by using, when available, the scores reported in Raileanu & Rocktäschel (2019) and Campero et al. (2020). In the performance graphs presented below: the vertical dotted line corresponds to the moment when the performance of the corresponding method reaches its point of convergence and the continuous horizontal line corresponds to its point of maximum performance in the considered plot. Note that all of the corresponding curves can be consulted in Fig. 3 of Raileanu & Rocktäschel (2019) and Fig. 4 of Campero et al. (2020). For the reader convenience, we include the screenshots in Appendix E.

5.1 Adversarially-based Exploration (No Episodic Count)

In this section, we assess the benefits of using an adversarially-based exploration bonus and examine how AGAC performs without the help of count-based exploration. In order to provide a comparison to state-of-the-art methods, we choose Vizdoom, a hard-exploration problem used in prior works. In this task, 9 are rooms connected by corridors and 270 steps separate the initial position of the agent and the goal under an optimal policy. An episode is terminated either when the agent finds the goal or if the episode exceeds 2100 teps. Importantly, while other algorithms (Raileanu & Rocktäschel, 2019; Campero et al., 2020) benefit from count-based exploration, AGAC does not use any such counts. Results in Fig. 2 indicate that AGAC clearly outperforms other methods in sample efficiency. Then, with nearly 2x more transitions, only ICM and RIDE match the score of AGAC. These results clearly support the usefulness of the adversarial bonus and show that it can, on its own, lead to significant gains in performance. Now, we want to investigate the capabilities of generalization to new states, by using MiniGrid.

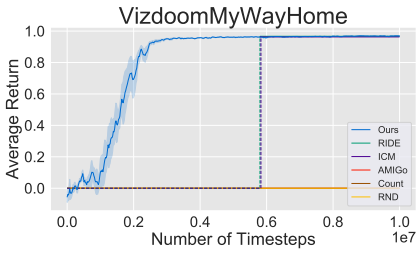


Figure 2: Average return in Vizdoom.

5.2 Hard-Exploration with Partially-Observable Policy

We now evaluate AGAC on multiple hard-exploration procedurally-generated tasks. From Fig. 3, it is clear that AGAC dramatically outperforms the baselines on these tasks. We also successfully outperform the current state-of-the-art method, AMIGo, although ituses fully-observable policies. In all the considered tasks, because they are procedurally-generated, the agent must learn to generalize across a very large state space. We consider three main arguments to explain why AGAC is so successful: (i) AGAC relies on partially-observable policies: in this context, the adversary has a harder time predicting the actor's actions; nevertheless the first one's mistakes benefit the other in the form of an exploration bonus, which pushes the agent to explore further in order to better deceive the adversary, (ii) intrinsic reward does not dissipate in AGAC compared to most other methods, as observed in Fig. 9 in Appendix A.3, (iii) AGAC does not make assumptions about the environment dynamics (*e.g.*, changes in the environment produced by an action as in Raileanu & Rocktäschel (2019)) since this can hinder learning when the state space changes induced by an action is too large (such as the action of moving a block in a grid as in ObstructedMaze).

In Appendix A.2, we also include experiments in two environments with extremely sparse reward signal: KeyCorridorS8R3 and ObstructedMazeFull. In these environments, the agent still manages to find rewards and can perform well by taking advantage of the diversified behaviour induced by AGAC. No other method ever succeeded to perform well (> 0 average return) in those tasks. We believe that with more computing time, AGAC's score could go higher.

5.3 Training Stability

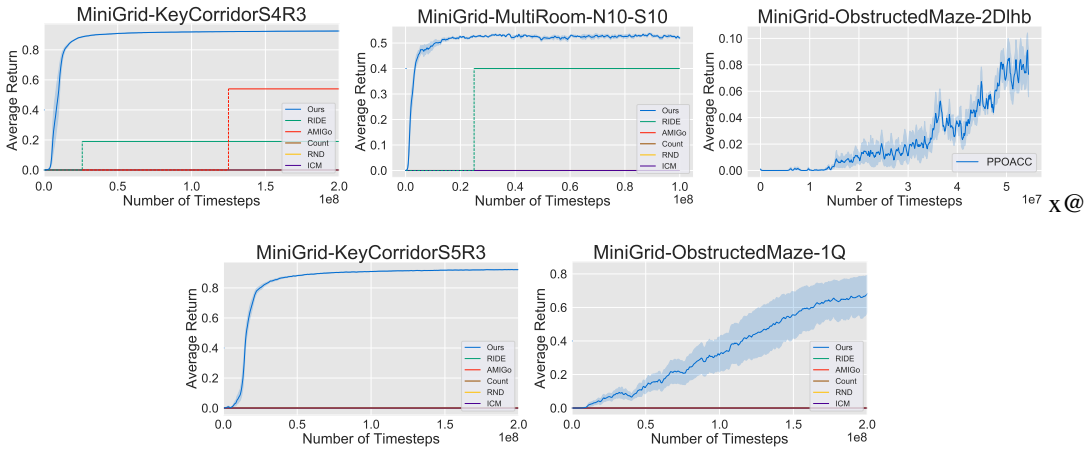


Figure 3: Performance evaluation of AGAC ("Ours" in the plots) compared to RIDE, AMIGo, Count, RND and ICM on several hard-exploration problems.

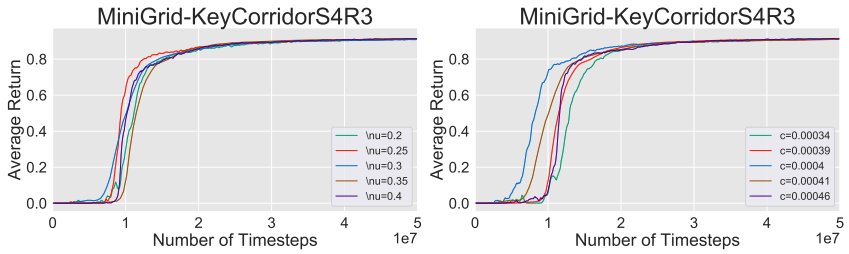


Figure 4: Ablation analysis of AGAC in KeyCorridorS4R3.

Here we want to analyse the stability of AGAC when changing hyper-parameters. The most important parameters in AGAC are c, the coefficient of the adversarial bonus, and the learning rates ratio $\nu = \frac{\eta_2}{\eta_1}$. We choose KeyCorridorS4R3 as the evaluation task because among all the tasks under consideration, its difficulty is at a medium level. Fig. 4 shows the learning curves. For readability, we plot the average return only; the standard deviation is about the same for all curves. We observe that deviating from the hyper-parameter values found using grid-search results in a slower training. Moreover, c appears to have more sensitivity than ξ.

5.4 Exploration in Reward-free Environment

To better understand the effectiveness of AGAC and investigate how the agent collects rewards that would not otherwise be achievable by simple exploration heuristics or other methods, we analyze the performance of AGAC in another (although procedurally-generated) moderately challenging environment, MultiRoomN10S6, when there is no reward signal (no extrinsic reward). Beyond the good performance of AGAC when extrinsic rewards are given to the agent, Fig. 5 indicates that the exploration induced by AGAC allows to reach the goal of the task a significant number of times. Note that in the configuration of NoExtrinsicReward, the reward signal is not given hence the actor is not optimized for it, confirming that the agent exhaustively covers the environment.

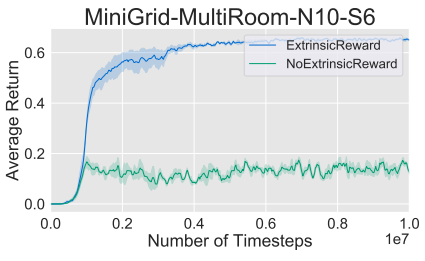


Figure 5: Average return on MultiRoomN10S6 with ExtrinsicReward or without NoExtrinsicReward.

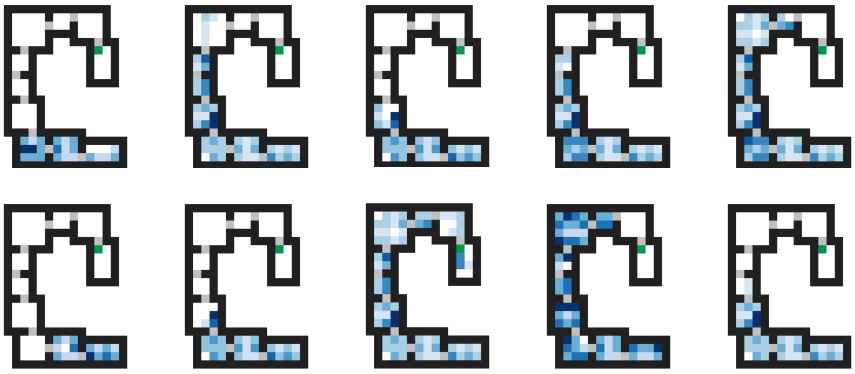


Figure 6: State visitation heatmaps of the last ten episodes of an agent trained in a procedurally-generated environment without extrinsic reward for 10M timesteps. The agent is continuously engaging into new strategies.

5.5 Promoting Diversity

In this section, we study the behavior of the agent. Fig. 6 presents the state visitation heatmaps (darker areas correspond to more visits) of ten episodes in the same singleton environment. We consider an agent trained on a procedurally-generated environment from the MultiRoomN10S6 task without extrinsic reward. The heatmaps correspond to the behavior of the resulting policy and still trained policy. Looking at the figure, we can see that the strategies vary at each update with, for example, back-and-forth and back-to-start behaviors. From one update to the next, although there are no extrinsic reward, the strategies seem to diversify. Fig. 7 in Appendix A.1 shows the state visitation heatmaps in a different configuration: when the agent has been trained on the same singleton environment from the MultiRoomN10S6 task without extrinsic reward. Looking at the figure we can make essentially the same observations as previously, with a noteworthy behavior in the fourth heatmap of the bottom row where it appears the agent went to the fourth room to remain inside it. Those episodes indicate that, although the agent sees the same environment over and over again, the successive updates force it to change behavior by trying new strategies.

6 Discussion

In this paper, we have introduced AGAC, where an adversary added to the traditional actor-critic couple. This adversary network is added to the policy gradient objective. The mechanics of `AGAC` have been discussed from a policy iteration point of view where we provide theoretical insight on the mechanics of the proposed algorithm. The adversary forces the agent to remain close to the previous policy while moving away from the old ones. In turns, the influence of the adversary makes the actor *conservatively diversified*.

In the experimental study, we have evaluated the adversarially-based bonus in Vizdoom and empirically demonstrated its effectiveness and superiority compared to other relevant methods (some benefiting from count-based exploration). Then, we have also conducted performance experiments using `AGAC` in its full potential (*i.e.*, by including an episodic state count) and have shown a significant performance improvement over some of the most popular exploration methods (Count (Bellemare et al., 2016a), ICM (Pathak et al., 2017), RND (Burda et al., 2018), RIDE (Raileanu & Rocktäschel, 2019)) on a set of various challenging tasks from MiniGrid. These environments, whose characteristic is to be procedurally-generated, have served a dualpurpose which is to validate the capacity of our method to generalize to unseen scenarios. In addition, the training stability of our method has been studied showing a greater but acceptable sensitivity for c, the adversarial bonus coefficient. Finally, we have investigated the exploration capabilities of `AGAC` in a reward-free setting where the agent demonstrated exhaustive exploration and exhibited a number of strategic choices, further supporting that the adversary successfully promotes diversity in the behavior of the actor.

PP
:
quid
des
temps
de
cal-
cul
d'AGAC?

PP
:
why
dual?

PP : on n'a aucune piste pour poursuivre le travail ?

REFERENCES

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, pp. 151–160, 2019.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Representation Learning*, 2017.

Dzmitry Bahdanau, Felix Hill, Jan Leike, Edward Hughes, Arian Hosseini, Pushmeet Kohli, and Edward Grefenstette. Learning to understand goal specifications by modelling reward. *arXiv preprint arXiv:1806.01946*, 2018.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5): 834–846, 1983.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pp. 1471–1479, 2016a.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in neural information processing systems*, pp. 1471–1479, 2016b.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2018.

Andres Campero, Roberta Raileanu, Heinrich Küttler, Joshua B. Tenenbaum, Tim Rocktäschel, and Edward Grefenstette. Learning with amigo: Adversarially motivated intrinsic goals. *arXiv preprint arXiv:2006.12122*, 2020.

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Representation Learning*, 2016.

Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pp. 1407–1416, 2018.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Jesse Farebrother, Marlos C Machado, and Michael Bowling. Generalization and regularization in dqn. *arXiv preprint arXiv:1810.00123*, 2018.

Johan Ferret, Raphaël Marinier, Matthieu Geist, and Olivier Pietquin. Self-attentional credit assignment for transfer in reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pp. 2655–2661, 2020.

Yannis Flet-Berliac and Philippe Preux. Only relevant information matters: Filtering out noisy samples to boost rl. In *International Joint Conference on Artificial Intelligence*, pp. 2711–2717, 2020.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. *arXiv preprint arXiv:1901.11275*, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Audrunas Gruslys, Will Dabney, Mohammad Gheshlaghi Azar, Bilal Piot, Marc Bellemare, and Remi Munos. The reactor: A fast and sample-efficient actor-critic agent for reinforcement learning. *arXiv preprint arXiv:1704.04651*, 2017.

Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents*, 2015.

David Held, Xinyang Geng, Carlos Florensa, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. *arXiv preprint arXiv:1705.06366*, 2017.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pp. 4565–4573, 2016.

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. *arXiv prepint arXiv:1910.12911*, 2019.

Niels Justesen, Ruben Rodriguez Torrado, Philip Bontrager, Ahmed Khalifa, Julian Togelius, and Sebastian Risi. Illuminating generalization in deep reinforcement learning through procedural level generation. *arXiv preprint arXiv:1806.10729*, 2018.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*, pp. 1–8. IEEE, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Representation Learning*, 2015.

Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*, 2020.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016a.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016b.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 1054–1062, 2016.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. *arXiv preprint arXiv:1806.05635*, 2018.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pp. 2778–2787, 2017.

David Pfau and Oriol Vinyals. Connecting generative adversarial networks and actor-critic methods. *arXiv preprint arXiv:1610.01945*, 2016.

Martin L. Puterman. *Markov Decision Processes*. Wiley, 1994. ISBN 978-0471727828.

Roberta Raileanu and Tim Rocktäschel. Ride: Rewarding impact-driven exploration for procedurally-generated environments. In *International Conference on Learning Representations*, 2019.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

Xingyou Song, Yiding Jiang, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. *arXiv preprint arXiv:1912.02975*, 2019.

Richard Stuart Sutton. *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 1984.

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leverage the average: an analysis of regularization in rl. *arXiv preprint arXiv:2003.14089*, 2020.

Oriol Vinyals, Igor Babuschkin, Wojciech Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John Agapiou, Max Jaderberg, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575, 11 2019.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in Neural Information Processing Systems*, pp. 5279–5288, 2017.

Amy Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018a.

Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018b.

A ADDITIONAL EXPERIMENTS

A.1 STATE VISITATION HEATMAPS IN SINGLETON ENVIRONMENT WITH NO EXTRINSIC REWARD

In this section, we provide additional state visitation heatmaps. The agent has been trained on a singleton environment from the MultiRoomN10S6 task without extrinsic reward. These ten episodes suggest that the agent sees the same environment over and over again, the updates force it to change behavior by trying new strategies.

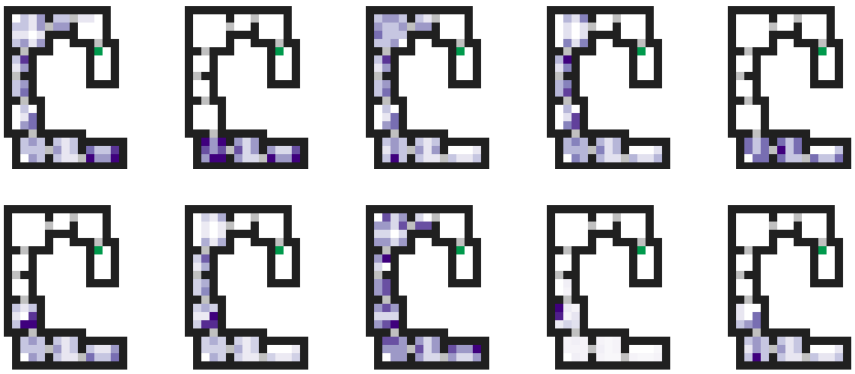


Figure 7: State visitation heatmaps of the last ten episodes of an agent trained in a singleton environment with no extrinsic rewards 10M timesteps. The agent is continuously engaging into new strategies.

A.2 (EXTREMELY) HARD-EXPLORATION WITH PARTIALLY-OBSERVABLE POLICY

In this section, we include additional experiments on one of the hardest tasks available in MiniGrid. The first, KeyCorridorS8R3, where the size of the rooms has been increased. In it, the agent has to pick up an object which is behind a locked door: the key is hidden in another room and the agent has to explore the environment to find it. The second, ObstructedMazeFull, similar to ObstructedMaze4Q, where the agent has to pick up a box which is placed in one of the four corners of a 3x3 maze: the doors are locked, the keys are hidden in boxes and the doors are obstructed by balls. In those difficult tasks, only our method succeeds in exploring well enough to find rewards.

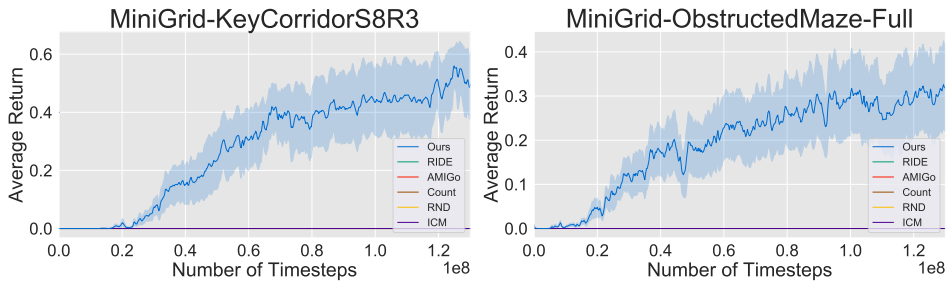


Figure 8: Performance evaluation (6 seeds) of `AGAC` compared to RIDE, AMIGo, Count, RND and ICM on extremely hard-exploration problems.

A.3 MEAN INTRINSIC REWARD

In this section, we report the mean intrinsic reward computed for an agent trained in MultiRoomN12S10 to conveniently compare our results with that of Raileanu & Rocktäschel (2019).

We observe in Fig. 9 that the intrinsic reward in consistently larger for our method and that, contrary to other methods, does not converge to low values (a comparison to the comparing results published in Raileanu & Rocktäschel (2019) is found in Fig. 12c). Please note that, in all considered experiments, the adversarial bonus coefficient c in Eq. 2 and 3 is linearly annealed throughout the training since it is mainly useful at the beginning of learning when the rewards have not yet been met. In the long run, this coefficient may prevent the agent from solving the task by forcing it to always favour exploration over exploitation.

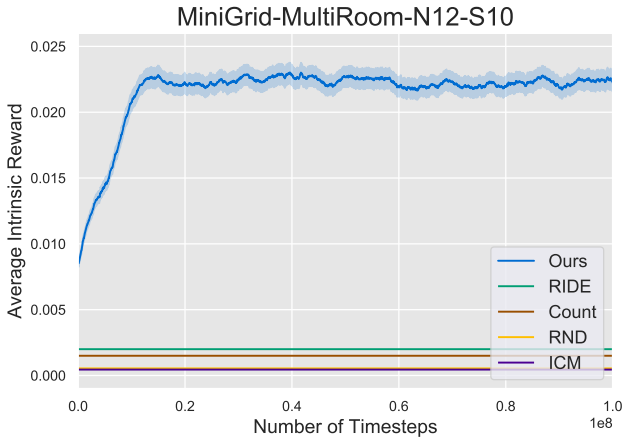


Figure 9: Average intrinsic reward for different methods trained in MultiRoomN12S10.

B Experimental Details and Hyper-parameters

In all experiments, we train six different instances of our algorithm with different random seeds. In Table 1, we report the list of hyper-parameters.

Table 1: Hyper-parameters used in `AGAC`.

Parameter	Value
Horizon T	2048
Nb. epochs	4
Nb. minibatches	8
Nb. frames stacked	4
Nonlinearity	ELU (Clevert et al., 2016)
Discount γ	0.99
GAE parameter λ	0.95
PPO clipping parameter ϵ	0.2
β_V	0.5
c	$4 \cdot 10^{-4}$ ($4 \cdot 10^{-5}$ in Vizdoom)
c anneal schedule	linear
β_{adv}	$4 \cdot 10^{-5}$
Adam stepsize η_1	$3 \cdot 10^{-4}$
Adam stepsize η_2	$9 \cdot 10^{-5} = 0.3 \cdot \eta_1$

C Implementation Details

In Fig. 10 is depicted the architecture of our method.

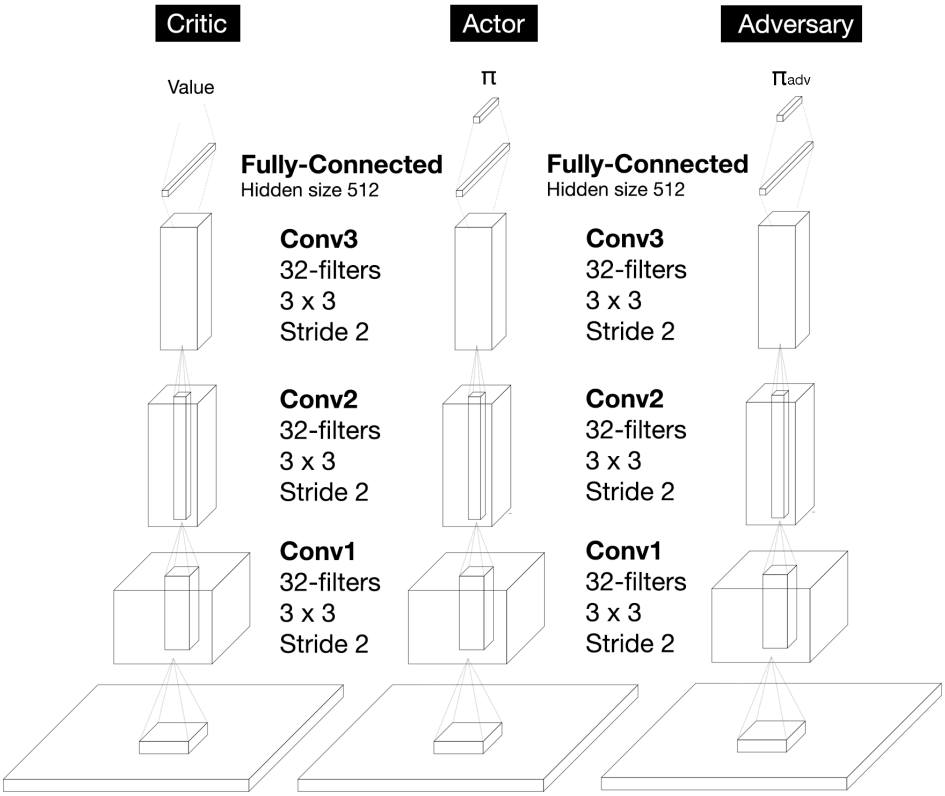


Figure 10: Artificial neural architecture of the critic, the actor and the adversary.

D Illustration of AGAC

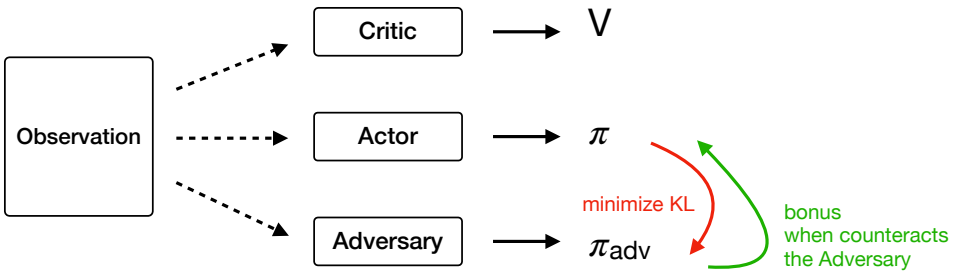
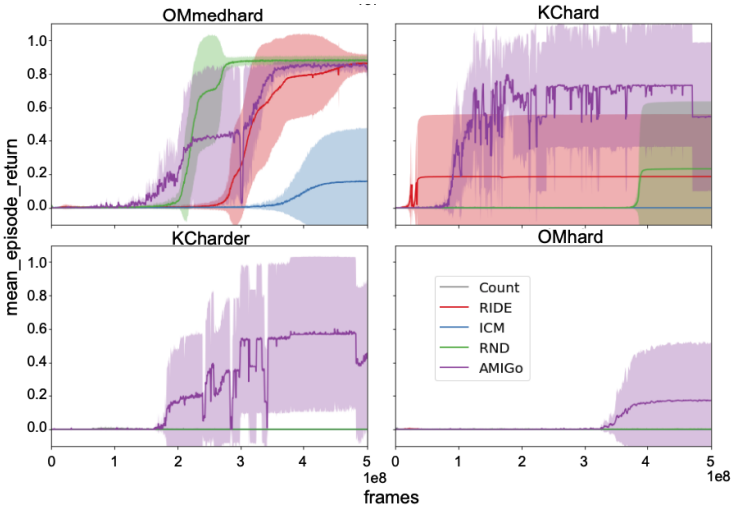


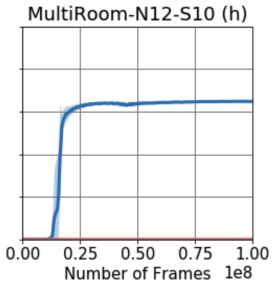
Figure 11: A simple schematic illustration of our method.

E EXTERNAL ADDITIONAL EXPERIMENTS

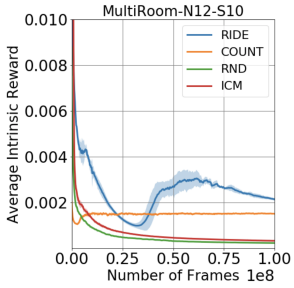
In this section, we attach screenshots from the experimental results in Fig. 3 of Raileanu & Rocktäschel (2019) and Fig. 4 of Campero et al. (2020).



(a)



(b)



(c)

Figure 12: (a) Screenshot of the performance reported in Campero et al. (2020). (b) Screenshot of the performance (X-axis: average return) reported in Raileanu & Rocktäschel (2019). (c) Screenshot of the average intrinsic reward reported in Raileanu & Rocktäschel (2019).