Data-Efficient Exploration with Self Play for Atari

Michael Laskin [1] **Catherine Cang** [1] **Ryan Rudes** [1] **Pieter Abbeel** [1 2]

Abstract

Most reinforcement learning (RL) algorithms rely on hand-crafted extrinsic rewards to learn skills. However, crafting a reward function for each skill is not scalable and results in narrow agents that learn reward-specific skills. To alleviate the reliance on reward engineering it is important to develop RL algorithms capable of efficiently acquiring skills with no rewards extrinsic to the agent. While much progress has been made on reward-free exploration in RL, current methods struggle to explore efficiently. Self-play has long been a promising approach for acquiring skills but most successful applications have been in multi-agent zero-sum games with extrinsic reward. In this work, we present SelfPlayer, a data-efficient single-agent self-play exploration algorithm. SelfPlayer samples hard but achievable goals from the agent's past by maximizing a symmetric KL divergence between the visitation distributions of two copies of the agent, Alice and Bob. We show that SelfPlayer outperforms prior leading self-supervised exploration algorithms such as Go-Explore and Curiosity on the data-efficient Atari benchmark.

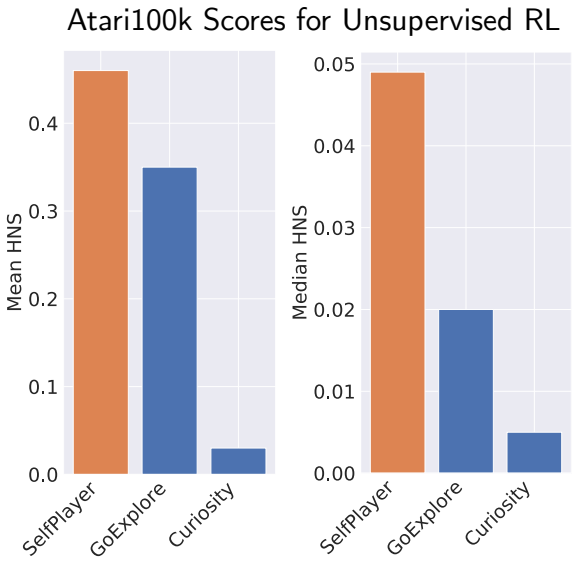

Figure 1. Mean and median human normalized scores (HNS) of unsupervised RL algorithms – SelfPlayer (ours), GoExplore, and Curiosity – on the data-efficient Atari benchmark (Atari100k) (Kaiser et al., 2020).

1. Introduction

Deep Reinforcement Learning (RL) has seen a number of breakthrough advances over the last decade. From learning to play Atari video games from pixels (Mnih et al., 2015), to mastering the game of Go (Silver et al., 2017b), to learning robotic locomotion (Schulman et al., 2017) and manipulation (Kalashnikov et al., 2018) skills, to competing at grandmaster level on large scale multi-agent video games (Vinyals et al., 2019). All of these breakthroughs were based on training RL algorithms to maximize an extrinsic reward that was provided by the human designers of these environments, and indeed most RL algorithms today are developed with the assumption that an extrinsic reward function is provided. However, dense reward shaping is a manual and cumbersome process and many real-world tasks that humans solve on a daily basis either have delayed sparse rewards or no apparent rewards at all. How can we enable artificial agents to explore their worlds efficiently in a self-supervised manner without access to task-specific extrinsic rewards?

Perhaps the most common approach to self-supervised exploration in RL is intrinsic motivation (Oudeyer & Kaplan, 2009) where an intrinsic reward function is specified that does not depend on the extrinsic reward. The RL agent still optimizes a reward function but the reward function does not use any data extrinsic to the agent. A popular example of an agent trained with an intrinsic motivation objective is Curiosity (Pathak et al., 2017) though many other instrinsic motivation objectives exist (Bellemare et al., 2016; Osband et al., 2016; Eysenbach et al., 2019). A less commonly used but alternate approach is through novel goal discovery where an agent sets progressively harder goals for itself dur-

[*]Equal contribution [1]University of California, Berkeley [2]Covariant. Correspondence to: Michael Laskin <mlaskin@berkeley.edu>.

International Conference on Machine Learning, Unsupervised Reinforcement Learning Workshop, 2021. Copyright 2021 by the author(s).

ing training and uses a goal-reaching policy to achieve them. A recent example of this approach is GoExplore (Ecoffet et al., 2020), which uses a count-based heuristic to score previous goals achieved and select challenging ones.

This work falls into the latter category of exploration methods with a focus of automatic goal discovery in the single-agent setting through self-play. Self-play has long been utilized for effective exploration in multi-agent settings. It was a key component that enabled state-of-the-art performance in large-scale multi-agent experiments (Silver et al., 2017a; Vinyals et al., 2019; Berner et al., 2019). However, given that self-play is defined as a game between two or more agents, it has been more challenging to employ in the single-agent setting.

Recently, it was shown (Sukhbaatar et al., 2018; OpenAI et al., 2021) that self-play can also be utilized for single-agent problems. In asymmetric self-play, the single agent maintains two copies of itself - Alice and Bob - that play an asymmetric game where Alice tries to achieve goals that are hard for Bob to clone and Bob in turn tries to copy Alice. Through this adversarial game, an automatic goal-reaching curriculum is built. While promising, the main shortcoming of this approach is that the self-play reward requires Monte-Carlo rollouts for Alice's policy, which results in extremely sample inefficient training. As such, ASP has only been shown to work beyond toy domains in large-scale experiments from OpenAI that required over 30B training steps to learn effective exploration policies.

In this paper, we propose an alternate approach to single-agent self-play that does not rely on Monte-Carlo rollouts for reward computation. Our key insight is to leverage previous data collected by Alice and Bob to set goals instead of relying on Monte-Carlo rollouts of the goal-generating agent. In our formulation, Alice and Bob are symmetric goal-conditioned agents, and they receive goals from an independent goal-setter. The goal-setter is rewarded for setting goals for Alice that are hard for Bob to achieve and vice versa. To formalize this idea, we introduce a new objective function for self-play based on the symmetric KL divergence between Alice and Bob's goal distributions. To summarize, we present our contributions below:

1. We introduce SelfPayer a new algorithm for single-agent self-play that instantiates two copies of the same agent, Alice and Bob, and samples challenging goals in hindsight for each agent based on the agent's prior experience.
2. SelfPlayer leverages a new objective function for reward-free exploration that is expressed in terms of the symmetric KL divergence between the visitation distributions of different copies of the agent.
3. We demonstrate that this formulation explores Atari games more efficiently than leading exploration approaches for Atari such as Curiosity (Burda et al., 2019) and GoExplore (Ecoffet et al., 2020) by evaluating performance on the efficient Atari (Atari100k) benchmark (Kaiser et al., 2020).

2. Related Work

Exploration through Intrinsic Motivation: In recent years, a number of works in RL have aimed to answer this question. The most common approaches to exploration in RL have been motivated by the psychological concept of intrinsic motivation (Oudeyer & Kaplan, 2009) wherein an agent seeks to maximize a task-agnostic and hence intrinsic reward function that, if maximized, will help the agent explore its environment more effectively. Proposals for intrinsic reward heuristics have included state-based counting (Bellemare et al., 2016; Machado et al., 2020), prediction error (Pathak et al., 2017; Burda et al., 2019), uncertainty (Osband et al., 2016; Pathak et al., 2019), and mutual information-based option discovery (Eysenbach et al., 2019; Achiam et al., 2018; Sharma et al., 2020) to name a few. While intrinsic motivation approaches enable agents to autonomously discover short-horizon skills such as jumping, hopping, and flipping, a limitation is that, in practice, they fail to discover long-horizon skills. This is particularly evident in hard exploration games like Montezuma's Revenge where exploration schemes that rely solely on intrinsic motivation fail to discover far apart rooms (Burda et al., 2019).

Goal-conditioned Exploration: A less frequently employed approach to exploration that overcomes the limitations of intrinsic motivation is exploring through automatic goal discovery. The automatic goal discovery family of approaches trains a goal-conditioned agent jointly with a goal-setting policy that sets incrementally harder goals over time. Automatically increasing the goal difficulty induces a curriculum and allows the goal-conditioned agent to explore over long-horizons and discover far apart states. Examples of automatic goal discovery algorithms are GoExplore (Ecoffet et al., 2020), which utilizes a hand-crafted state counting heuristic to set goals, goal generating algorithms (Florensa et al., 2018; Campero et al., 2021; Zhang et al., 2020), and self-play approaches (Sukhbaatar et al., 2018; OpenAI et al., 2021), which pit two or more agents against each other to compete in a way that induces an automatic goal-reaching curriculum.

Reinforcement Learning with Self-Play: Self-play has been a core feature of several advances in supervised Reinforcement Learning, particularly in two-player game algorithms such as AlphaZero (Silver et al., 2017a) and large-scale multiplayer game algorithms such as AlphaStar (Vinyals et al., 2019) and Dota5 (Berner et al., 2019). While

a key component of many leading multi-player supervised RL algorithms, self-play has not been adopted to single-agent settings until recently when Asymmetric Self-Play (ASP) (Sukhbaatar et al., 2018) provided a framework for casting single-agent exploration as a multi-agent problem. In ASP, two copies of the same agent - Alice and Bob - compete against each other to achieve progressively harder goals. Alice is a state-conditioned agent that is rewarded for finding paths that are hard for Bob to copy. Bob is a state and goal-conditioned agent that aims to copy the goals achieved during Alice's rollout. A recent extension of ASP (OpenAI et al., 2021) demonstrated that it can scale to simulated robotics manipulation settings but requires enormous compute resources that are only available to a small handful of industrial labs. To date, no self-supervised self-play exploration approaches have been demonstrated in the Atari setting.

3. Method

3.1. SelfPlayer: Data-Efficient Single-Agent Self-Play

In this paper, we consider the standard setting for goal-conditioned RL where an agent acts within a Markov Decision Process, defined as the tuple $(\mathcal{O}, \mathcal{G}, \mathcal{A}, P, \gamma)$, with the following components: observations $o \in \mathcal{O} = R^n$, goals $s \in \mathcal{G} = R^n$, actions $a \in \mathcal{A}$, and state transition distribution, $P = P(o', r|o, a)$, which defines the task mechanics and rewards. Actions can either be discrete ($\mathcal{A} = J$) or continuous ($\mathcal{A} = R^m$), though we evaluate on the Atari benchmark (Bellemare et al., 2013) where the action space is discrete. Without prior knowledge of P, the RL agent's goal is to use experience to maximize expected rewards, $R = \sum_{t=0}^{\infty} \gamma^t r_t$, under discount factor $\gamma \in [0, 1)$. In Atari the agent receives image-based observations which are a high-dimensional indirect representations of the underlying state. The above describes the general RL problem setting. Since this work is concerned with self-supervised exploration, we assume that we cannot access the extrinsic reward function and must explore given only extrinsic reward-independent information in the MDP. The MDP must still have a reward function but the reward is computed with information intrinsic to the agent such as achieved observations, goals, or actions. Additionally, in this work goals are defined as embeddings of observations. Let $z \in \mathcal{Z}$ be a latent space of image embeddings and $\mathcal{G} = \mathcal{Z}$. Our MDP then consists of the following tuple $(\mathcal{O}, \mathcal{Z}, \mathcal{A}, R_{\text{unsup}}, P, \gamma)$.

To enable self-play in a single agent setting, we instantiate two copies of the agent Alice and Bob similar to the setup in ASP (OpenAI et al., 2021). The primary aim of self-play approaches is to induce a curriculum that allows the agent to achieve incrementally harder goals at each iteration. We wish to set goals for Alice and Bob that are possible to achieve yet challenging in order to induce this curriculum. Given the two agents Alice A and Bob B, a goal g, a goal-reaching policy $\pi_g^{(j)}$ where $j = A, B$ and a success function $F(\pi_g^{(j)}, g) = \{0, 1\}$ that determines whether Alice or Bob's policy $\pi_g^{(j)}$ can successfully achieve a goal g, we wish to maximize the following reward function for each agent:

$$R_i = 1 \;\text{ IF }\; F(\pi_g^{(i)}, g) = 1 \;\text{ AND }\; F(\pi_g^{(j)}, g) = 0 \quad (1)$$

$$\text{ELSE } \; 0 \; , \quad i = A, B, j = B, A \quad (2)$$

In equation 1, Alice is rewarded if she can reach a goal that Bob cannot and vice versa. To motivate this objective, we first study its properties and show that the Nash equilibrium according to eq. 1 corresponds to a fully explored MDP. Our primary contribution is not the use of a self-play objective but rather a new approach to approximating it which we describe later in this section.

Definition 1. *The MDP is considered fully explored if all states of the MDP have been visited, meaning $\nexists s \in \mathcal{S}$ such that $s \in \mathcal{D}$ where $\mathcal{D}$ is the replay buffer of the exploration agent.*

Definition 2. *Given players $i = 1, \ldots, N$ and value functions for a particular agent's policy v_π^i, a game is considered to be in a Nash equilibrium if all agents' value functions are optimal $v_\pi^i = v_{\pi^*}^i \forall i$. This definition is taken from (Pérolat et al., 2017).*

Theorem 1. *If Alice and Bob are in a Nash equilibrium according to equation 1, then the environment has been fully explored.*

Proof. We provide a full proof in the Appendix and provide a sketch of the proof here. We prove this theorem by contradiction. Suppose that Alice and Bob are in Nash equilibrium but the MDP is not fully explored. We can show that there is always a goal $\exists \hat{g} \in \mathcal{G}$ that Alice can reach but Bob cannot (or vice versa). Therefore, Alice's value function v_π^A upon selecting $\hat{g}$ for Bob is greater than her current value function, but Alice is in a Nash equilibrium meaning her current value function is optimal. This is a contradiction. Therefore the MDP must be fully explored. □

We've shown that the Nash equilibrium of eq. 1 corresponds to a fully explored MDP suggesting that this is an appropriate objective for exploration. We now focus on how to optimize this objective efficiently. In this work, rather than relying on Monte-Carlo rollouts, we instead rely on Alice and Bob's memory to sample goals from each agent's past that are hard for the other agent to achieve. Given Alice and Bob's replay buffers $\mathcal{D}_A$ and $\mathcal{D}_B$ we introduce a goal-setter that provides goals for Alice and Bob conditioned on their replay buffers. The goal-setter is rewarded for setting goals for Alice that are hard for Alice to achieve but easy for Bob

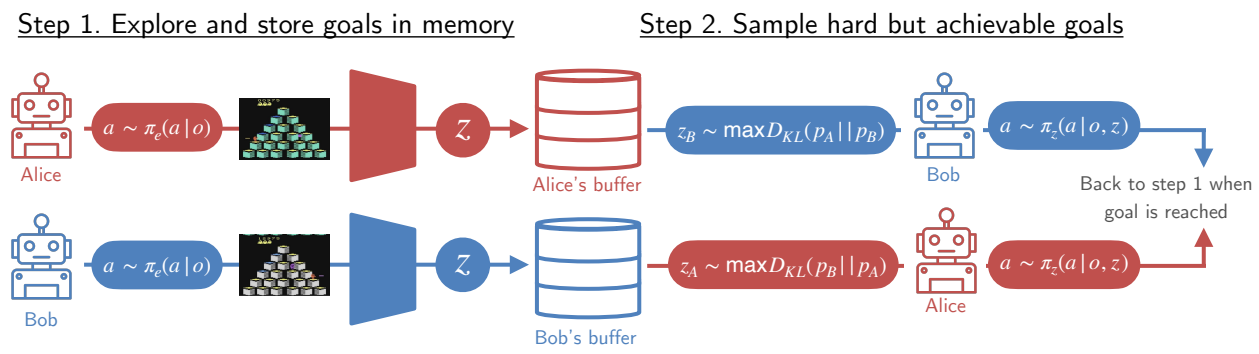

Figure 2. The SelfPlayer algorithm is comprised of two steps - (i) exploring stochastically and (ii) returning to goals. During the exploration phase, Alice and Bob store observations in their respective replay buffers. Any policy can be used for exploration and we opt for simple random exploration with sticky actions. During the goal returning phase, hard but achievable goals are sampled for Alice and Bob using the self-play objective in eq. 3. Our goal space, defined in 3.1, consists of embeddings of observations o which we denote as $z \in \mathcal{Z}$. After sampling, a goal-conditioned policy is used to reset Alice and Bob at their sampled goals. Once a goal is reached, Alice and Bob go back to exploring.

and vice versa, hence the goal-setter's total reward after one rollout for Alice and Bob is $R = \sum_i R_i = R_A + R_B$.

But how do we determine Alice and Bob's success functions $F(\pi_A, g)$ and $F(\pi_B, g)$ without relying on estimates from Monte-Carlo rollouts? Our main insight is that goals that were frequently achieved in the past are likely to be achieved again and goals that were rarely achieved or not achieved at all in the past are likely to be challenging for the agent's current policy. This enables us to approximate the self-play reward in terms of distributions over Alice and Bob's past visitations. Specifically, we fit distributions over Alice and Bob's visitation histories $p_A \equiv p(g|\mathcal{D}_A)$ and $p_B \equiv p(g|\mathcal{D}_B)$. Therefore, the expected log likelihood of an agent's $j = A, B$ success is $E_{g\sim p_j}[\log p_j]$. We can then express the new cumulative reward function in terms of the $p_{j=A,B}$ distributions.

$$
\begin{aligned}
R &= E_{g\sim p_A}[\log p_A - \log p_B] && (3)\\
&+ E_{g\sim p_B}[\log p_B - \log p_A] && (4)\\
&= D_{KL}(p_A||p_B) + D_{KL}(p_B||p_A) && (5)
\end{aligned}
$$

which is the symmetric KL divergence between p_A and p_B. In this framework, Alice and Bob are normal goal-conditioned agents who aim to achieve goals given by a goal-setter.

3.2. Relation to Mutual Information and Entropy

The objective eq. 3 can be expressed in terms of Alice and BoB's entropy and mutual information.

$$
\begin{aligned}
R &= D_{KL}(p_A||p_B) + D_{KL}(p_B||p_A) && (6)\\
&= H(p_A, p_B) - H(p_B) + H(p_B, p_A) - H(p_B) && (7)\\
&= H(p_B|p_A) + H(p_A|p_B) && (8)\\
&= H(p_A) + H(p_B) - 2I(p_A; p_B) && (9)
\end{aligned}
$$

The goal-setter is therefore trying to set goals that maximize Alice and Bob's individual entropy while minimizing the mutual information between Alice and Bob's achieved goal distributions. From eq. 6 it would appear that Alice and Bob will eventually drift off into different mutually exclusive parts of the MDP's state space, but this is not the case. When the goal-setter samples goals for Alice those goals are sampled from Bob's replay buffer and vice versa. For this reason, each time a goal is set for Alice, the mutual information between Alice's achieved goal distribution and Bob's achieved goal distribution increases. The goal-setter's objective is thus to sample goals that will minimally increase $I(p_A, p_B)$ while maximally increasing $H(p_A)$ and $H(p_B)$.

3.3. Practical Implementation

Sampling goals with self-play: We observe that since Alice and Bob are goal-conditioned agents, the goal-setter is the only part of the algorithm that is responsible for maximizing the self-play objective. Moreover, the goal-setter's MDP only has one timestep since it sets a goal and observes its total reward in one step. This means that we can directly maximize the self-play objective by sampling goals according to eq. 3. The goal-setter thus functions as a critic. In our practical implementation, we store discrete representations

Figure 3. State visitation heatmaps for a random policy, GoExplore, and SelfPlayer in a gridworld Tunnel environment after 10k exploration steps.

z of the input observations o as goals in $\mathcal{D}_j$, fit categorical visitation probability distributions to $\mathcal{D}_j$ where $j = A, B$, and score the goals in Alice and Bob's replay buffers according to eq. 3 for each agent. We then sample goals from $\mathcal{D}_j$ according to the scoring function. The full method is shown in Algorithm 1.

Reaching goal states and exploring: There are a number of ways to reach goal states once a goal is sampled for Alice and Bob. If given a simulator, the goal-reaching policy can be a reset function that resets the agent at its desired goal. Since Atari is a simulated environment, this is the strategy used to report the main results in GoExplore (Ecoffet et al., 2020) and we do so as well. A more general approach for goal reaching is to dispatch a goal-conditioned policy that starts at the current observed state and travels to the desired goal. Prior works have explored how to craft such goal-reaching policies in the context of Atari by leveraging previous goal-reaching sequences as demonstrations (Hester et al., 2018; Guo et al., 2020; Ecoffet et al., 2020). We provide an implementation of an imitation policy in our code and visualizations of goal-reaching rollouts in the supplementary material. Finally, for exploration we use the same stochastic sticky-action policy used in GoExplore (Ecoffet et al., 2020). We also tried more sophisticated exploration policies such as Curiosity but found that random exploration with sticky actions performed better (see appendix).

Encoding Visual Observations: Our method relies on the existence of discrete representations of the underlying goal space in order to fit a categorical distribution for sampling goals. Since we evaluate on the Atari benchmark where observations and goals are pixel images, we wish to encode images into discrete latent codes. In prior work (Ecoffet et al., 2020) it was observed that downsampling the input pixel image from $(64, 64, 3) \rightarrow (11, 8, 1)$ dimensions with numerical values discretized to 8 values is sufficient for Atari. Another, more general approach, is to learn a discrete representation space with a Vector Quantized Variational Autoencoder (VQVAE) (van den Oord et al., 2018). In this work, we examine self-play with both discrete codes generated through downsampling as well as those learned with VQVAE, opting for the VQVAE2 (Razavi et al., 2019) due to its superior performance on vision tasks than its predecessor. We run pixel downsampling experiments on CPUs and VQVAE2 experiments on GeForce RTX Nvidia GPUs (see appendix for details and hyperparameters).

3.4. Motivational Experiment: Exploring in PointMaze

We motivate SelfPlayer by investigating its performance in a simple gridworld environment that contains a pointmass agent in a maze (PointMaze). We choose this setting because mazes are closed and hence the underlying MDPs are finite meaning that maze coverage directly measures the exploration capabilities of the agent. We compare SelfPlayer to a random exploration policy and a count-based one (GoExplore) in the data-efficient regime by providing only 10k exploration steps. To score Alice and Bob's replay buffers according to eq. 3, we fit a categorical distribution to the discrete gridworld states. We find that on large Tunnel maze, SelfPlayer, GoExplore, and a random policy achieve $35\%, 21\%$ and 4% coverage respectively when evaluated over 10 seeds suggesting that in a simple environment SelfPlayer explores more efficiently. We show qualitative visitation heatmaps in Fig. 3.

Algorithm 1 SelfPlayer Algorithm

Initialize: Denote Alice as A and Bob as B. The environment provides image observations $o \in \mathcal{O}$. We are also given a scoring function R according to Eq. 3. Given $j = A, B$, initialize replay buffers $\mathcal{D}_j$, goal reaching polices $\pi_g(a|o, g)$, exploration policies $\pi_e(a|o)$, and goal probability distributions $p_j(\cdot|\mathcal{D}_j)$.
for N epochs **do**
  for j in $\{A, B\}$ **do**
    for K trajectories **do**
      Sample g from $\mathcal{D}_A$ or $\mathcal{D}_B$ from a categorical distribution weighed according to Alice or Bob's scoring function in eq. 3. Run π_g to reach g.
      Collect exploration rollouts with $a \sim \pi_e \rightarrow (o_t, a_t, o_{t+1}, g_t)$ of length h. Note $g_t = z_t f(o_t)$ where f is an encoder function and $z \in \mathcal{Z}$ is a latent representation.
      $\mathcal{D}_j \leftarrow \mathcal{D}_j \cup (o_t, a_t, o_{t+1}, g_t)$
  update: goal distributions $p_j((\cdot|\mathcal{D}_j)$ and encoder $f(o)$.

4. Experimental setup and baselines

Environments In this work, we focus on efficient exploration of MDPs. The Atari arcade environment (Bellemare et al., 2013) is perhaps the most common benchmark for evaluating the performance of RL agents. Recently, the efficient Atari benchmark (Atari100k) (Kaiser et al., 2020) was introduced in an effort to develop more sample efficient supervised RL algorithms. Atari100k evaluates performance after 100k agent steps or equivalently 400k frames of game-

play since the baselines in this benchmark use an action repeat of 4 (Kaiser et al., 2020). Evaluation on the Atari100k benchmark has dramatically improved the sample-efficiency of pixel-based supervised RL algorithms (Kaiser et al., 2020; Srinivas et al., 2020; Yarats et al., 2021; Schwarzer et al., 2021). We utilize the same benchmark to evaluate the data efficiency of unsupervised (reward-free) exploration agents and evaluate performance after 400k frames of gameplay for each method. For SelfPlayer this means that Alice and Bob individually interact with the environment 200k times for a total of 400k frames. We evaluate performance on the 48 games used in Curiosity (Burda et al., 2019).

Baselines For baselines we compare to two leading self-supervised exploration algorithms for Atari – GoExplore (Ecoffet et al., 2020) and Curiosity (Pathak et al., 2017; Burda et al., 2019).

Curiosity: Curiosity learns a dynamics model $\hat{T}(s_{t+1}|s_t, a_t)$ and incentives the agent to explore with an intrinsic reward that is proportional to the prediction error $\epsilon = |\hat{s}_{t+1} - s_{t+1}|$ where $\hat{s}_{t+1} \sim \hat{T}$. The dynamics model is learned from data collected from by the agent's policy which is optimized with proximal policy optimization (PPO) (Schulman et al., 2017). The agent therefore tends to explore parts of the state space it has not seen before. We report performance on the Atari100k benchmark based on data provided by the authors (Burda et al., 2019).

GoExplore: GoExplore is a leading count-based algorithm for exploration in Atari that, like SelfPlayer, is sepa rated into exploration and goal-reaching phases. During the exploration phase GoExplore stores discrete representations of the input images in a replay buffer and uses a hand-designed count-based score function to sample goal states from the replay buffer. During the goal-reaching phase, a goal is sampled from the replay buffer and a goal-reaching policy is used to reach the goal state. GoExplore has two variants - one with human-specified features to define states and one operating solely on image inputs. We compare to the variant that operates solely in image input. To generate Atari100k scores, we used the code provided by the authors (Ecoffet et al., 2020) which we ported over to our codebase with a few optimizations for faster runtimes.

Note that to make the fairest comparison to the baselines, we use the same exploration and goal-reaching strategies presented in GoExplore (Ecoffet et al., 2020) as well as the same image downsampling procedure. The only difference is the self-play goal sampling strategy as well as the inclusion of two agents, Alice and Bob. One drawback of GoExplore is that the pixel downsampling procedure may not be general. For this reason, we show in the ablations section that SelfPlayer operating on discrete VQVAE (Razavi et al., 2019) embeddings is competitive with SelfPlayer operating on downsampled pixels, which alleviates generality concerns in terms of the discrete embedding structure.

5. Results

5.1. Main results

A full list of Atari100k scores for SelfPlayer and the baselines is shown in Table 1. In line with prior work on Atari (Mnih et al., 2015; Kaiser et al., 2020; Hessel et al., 2017), we use Human Normalized Score (HNS) as our aggregate statistic for evaluation. With this in mind, we provide a summary of our findings:

On the Atari100k benchmark SelfPlayer is the state-of-the-art unsupervised RL algorithm. SelfPlayer achieves the best score in 26 games, compared to 7 for the leading unsupervised baseline. SelfPlayer has a **1.34x higher mean** and **2.5x higher median HNS** than the next leading unsupervised method and achieves superhuman performance on some games (Breakout, JamesBond, and WizardOfWor). We also find that without any access to extrinsic reward, SelfPlayer outperforms common supervised RL algorithms such as PPO (Schulman et al., 2017) and Rainbow DQN (Hessel et al., 2017), an is competitive with the data-efficient supervised model-based RL algorithm that introduced the Atari100k benchmark (Kaiser et al., 2020) (see Fig. 6). Finally, SelfPlayer does not necessarily rely on Atari-specific downsampled pixel representations. SelfPlayer with discrete embeddings from a VQVAE2 (Razavi et al., 2019) trained jointly using Alice and Bob's replay buffers (VQ-SelfPlayer) is competitive with SelfPlayer trained on downsampled pixels (see Fig. 6).

5.2. Ablations

To further investigate the SelfPlayer algorithm we perform a series of ablations to answer the following questions.

Q1: Why does SelfPlayer achieve higher extrinsic reward than prior unsupervised RL methods?

Since neither SelfPlayer, GoExplore, or Curiosity have access to extrinsic reward, we investigate why SelfPlayer achieves higher performance in terms of extrinsic reward on Atari100k. To do so, we compare the number of distinct discrete representations (cells) in a subset of SelfPlayer and GoExplore runs that have identical downsampling factors. Cell count is therefore the Atari equivalent of coverage in gridworld, which we previously investigated in Fig. 3. Cell count plots shown in Fig. 4 show that, as with the gridworld, SelfPayer discovers substantially more cells than GoExplore which means it has discovered more states in the underlying MDP. Since Atari games are designed to reward human player for advancing further in the game, the higher scores for SelfPlayer are likely a result of more efficient

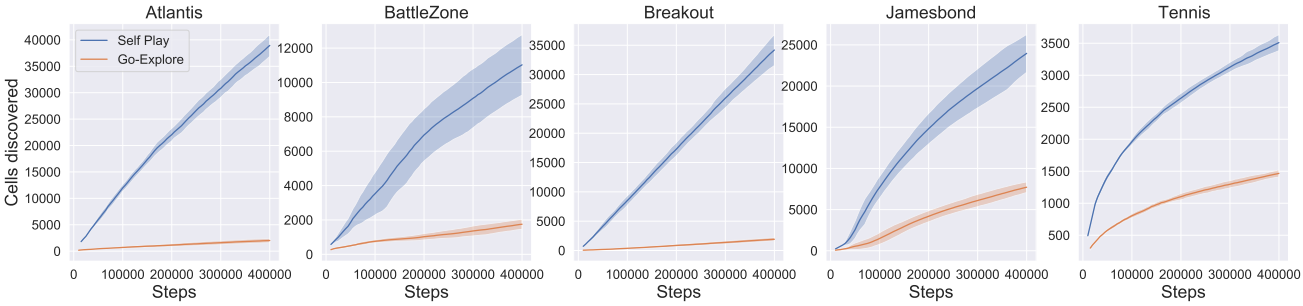

Figure 4. To investigate why SelfPlayer achieves higher mean and median scores on Atari100k than prior unsupervised RL approaches, we count the total number on unique discrete representations (cells) discovered by SelfPlayer compared to GoExplore. We select a subset of environments that have the same discrete latent space dimension and plot cells discovered throughout training. It is evident that SelfPlayer discovers substantially more cells than GoExplore. New cell discovery correlates higher extrinsic reward since Atari games and many video games in general are designed in part to reward exploration.

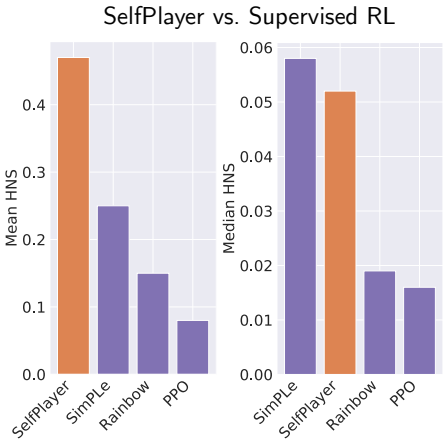

Figure 5. Mean and median HNS evaluated over the 36 games reported in SimPLe (Kaiser et al., 2020). SelfPlayer outperforms common supervised RL baselines such as PPO (Schulman et al., 2017) and Rainbow DQN (Hessel et al., 2017) and is competitive with with data-efficient supervised MBRL (SimPLe) (Kaiser et al., 2020).

exploration.

Q2: How does SelfPlayer compare with superivsed RL on Atari100k?

Given the efficiency gains achieved by SelfPlayer compared to unsupervised exploration algorithms, we investigate whether SelfPlayer is competitive with supervised RL approaches. To do so, we compare to the supervised RL baselines presented the original Atari100k benchmark paper (Kaiser et al., 2020) which introduced a data-efficient supervised model-based RL algorithm (SimPLe). The supervised baselines include PPO (Schulman et al., 2017), Rainbow DQN (Hessel et al., 2017), and SimPLe. We extract the scores from all the across the 36 games presented in (Kaiser et al., 2020) for each baseline, compute the median HNS, and show results in Fig. 6. We find that despite not having access to extrinsic reward, SelfPlayer and VQ-SelfPlayer substantially outerform PPO and Rainbow DQN and are competitive with SimPLe.

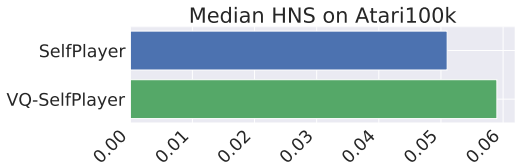

Figure 6. Median HNS for SelfPlayer using a jointly trained VQ-VAE2 for discrete embeddings (VQ-SelfPlayer) versus downsampled pixel representations on 16 randomly selected Atari games.

Q3: Does SelfPlayer rely on Atari specific discrete representations?

A concern with downsampling pixel observations as discrete encodings of the input image for both SelfPlayer and GoExplore is that it is an Atari-specific discretization. To investigate whether SelfPlayer would work with more general representations we implement a variant of SelfPlayer that operates on discrete embeddings produced by a VQVAE2 (Razavi et al., 2019) (VQ-SelfPlayer), which is a general method for encoding images from any domain. To ensure generality, we train the VQVAE2 online as the SelfPlayer agent on data collected by Alice and Bob during exploration and refresh the discrete encodings in the replay buffers after a fixed number of rollouts. Due to compute constraints, we randomly subsample 16 Atari games and report the median HNS in Fig. 6. We find that VQ-SelfPlayer achieves a slightly higher median HNS than SelfPlayer suggesting that SelfPlayer works comparably well when using more general discrete encodings.

Table 1. Scores on the efficient Atari (Atari100k) benchmark, which evaluates scores after 100k agent steps with action repeat of 4 or equivalently 400k frames. We report mean scores and standard errors evaluated over 10 random seeds for SelfPlayer and GoExplore. Curiosity scores were provided by the authors of the original paper (Burda et al., 2019). Scores are considered ties if the means are within 5% of each other.

ENV	SELFPLAYER	GOEXPLORE	CURIOSITY	RANDOM	HUMAN
ALIEN	**551.41** ± 41.8	**550.32** ± 54.5	217.0	227.8	7127.7
AMIDAR	**270.84** ± 17.9	166.92 ± 11.9	18.1	5.8	1719.5
ASSAULT	**305.88** ± 17.0	180.67 ± 14.6	214.0	222.4	742.0
ASTERIX	743.43 ± 315.0	**1007.14** ± 47.8	223.0	210.0	8503.3
ASTEROIDS	718.69 ± 101.5	217.78 ± 14.0	**953.0**	719.1	47388.7
ATLANTIS	6990.91 ± 1011.0	1220.63 ± 195.3	**17400.0**	12850.0	29028.1
BANKHEIST	**68.99** ± 5.8	26.19 ± 3.9	12.9	14.2	753.1
BATTLEZONE	**10679.01** ± 604.4	6412.7 ± 636.6	2560.0	2360.0	37187.5
BEAMRIDER	294.26 ± 14.8	285.08 ± 27.1	311.0	**363.9**	16926.5
BOWLING	**40.53** ± 3.0	20.22 ± 1.2	20.2	23.1	160.7
BREAKOUT	**87.31** ± 39.2	9.9 ± 0.8	3.3	1.7	30.5
CENTIPEDE	**4024.76** ± 373.1	3308.33 ± 576.1	2100.0	2090.9	12017.0
CHOPPER	**850.0** ± 65.1	719.05 ± 51.8	597.0	**811.0**	7387.8
CRAZYCLIMBER	3574.75 ± 337.0	1326.98 ± 59.4	9690.0	**10780.5**	35829.4
DEMONATTACK	**455.33** ± 33.0	371.9 ± 26.2	161.0	152.1	1971.0
DOUBLEDUNK	-23.64 ± 0.3	-15.87 ± 0.7	-18.0	-18.6	-16.4
ENDURO	0.38 ± 0.1	**0.46** ± 0.2	0.0	0.0	860.5
FISHINGDERBY	-88.62 ± 0.5	-73.65 ± 1.9	-92.3	-91.7	-38.7
FREEWAY	4.77 ± 0.2	**5.16** ± 0.3	0.73	0.0	29.6
FROSTBITE	**403.09** ± 59.0	86.03 ± 8.9	149.0	65.2	4334.7
GOPHER	142.96 ± 14.3	53.97 ± 7.8	**397.0**	257.6	2412.5
GRAVITAR	**276.54** ± 22.0	103.97 ± 25.4	251.0	173.0	3351.4
ICEHOCKEY	-15.02 ± 0.3	-14.02 ± 0.7	-13.1	-11.2	0.9
JAMESBOND	**5483.89** ± 148.4	4663.49 ± 155.2	38.2	29.0	302.8
KANGAROO	**866.67** ± 127.5	**892.06** ± 234.8	59.3	52.0	3035.0
KRULL	83.52 ± 5.0	37.56 ± 11.2	814.0	**1598.0**	2665.5
KUNGFUMASTER	77.78 ± 33.9	17.46 ± 6.5	**400.0**	258.5	22736.3
MONTEZUMA	**1369.14** ± 140.4	26.98 ± 6.3	0.0	0.0	4753.3
MSPACMAN	**1381.73** ± 79.9	1071.11 ± 78.5	246.0	307.3	6951.6
NAMETHISGAME	1274.32 ± 64.2	1120.63 ± 62.9	2070.0	**2292.3**	8049.0
PITFALL	-446.57 ± 107.3	-430.62 ± 44.3	-39.1	-229.4	6463.7
PONG	-20.21 ± 0.2	-11.71 ± 0.7	-19.1	-20.7	14.6
PRIVATEEYE	**2694.78** ± 764.7	**2565.24** ± 893.4	-608.0	24.9	69571.3
QBERT	**854.01** ± 168.7	687.7 ± 47.7	334.0	163.9	13455.0
RIVERRAID	**2149.63** ± 203.9	1016.98 ± 54.7	1490.0	1338.5	17118.0
ROADRUNNER	**2271.6** ± 312.5	1103.17 ± 239.7	60.0	11.5	7845.0
ROBOTANK	**4.61** ± 0.3	3.62 ± 0.3	2.67	2.2	11.9
SEAQUEST	**252.89** ± 26.3	179.52 ± 45.1	237.0	68.4	42054.7
SPACEINVADERS	**330.06** ± 29.1	304.13 ± 29.2	191.0	148.0	1668.7
STARGUNNER	146.67 ± 7.6	19.05 ± 5.7	533.0	**664.0**	10250.0
TENNIS	-23.08 ± 0.1	-15.79 ± 0.5	-23.5	-23.8	-8.3
TIMEPILOT	1122.22 ± 271.7	120.63 ± 17.5	2520.0	**3568.0**	5229.2
TUTANKHAM	**16.2** ± 2.0	5.94 ± 1.1	7.41	11.4	167.6
UPNDOWN	1213.11 ± 114.5	1176.83 ± 107.3	**1510.0**	533.4	11693.2
VENTURE	23.33 ± 15.8	15.87 ± 8.9	**83.3**	0.0	1187.5
VIDEOPINBALL	10555.21 ± 1434.2	10454.87 ± 1549.6	**35300.0**	0.0	17667.9
WIZARDOFWOR	**7813.33** ± 1012.8	1206.35 ± 66.1	817.0	563.5	4756.5
ZAXXON	**391.11** ± 41.4	184.13 ± 34.8	1.68	32.5	9173.3
# ENVS BEST	**26**	6	7	7	-
AVERAGE HNS	**.47**	.35	0.03	-	-
MEDIAN HNS	**.05**	.02	0.01	-	-

Broader Impact and Limitations

The primary benefits of data-efficient exploration methods is enabling autonomous agents to learn skills quickly without supervision. This could make training of robotic agents, digital assistants, and other decision making systems more scalable than the current paradigm where a reward function

is hand-designed for each task and domain. Additionally, efficient exploration could enable advances in other scientific problems with large search spaces such as molecular docking in drug discovery. However, since exploration methods like SelfPlayer, GoExplore, and Curiosity are unconstrained a negative potential impact and limitation is the lack of safety guarantees. For instance, deploying SelfPlayer onto a real robot could have harmful and unintended consequences for the environment and other agents since the algorithm is designed only to explore. Subsequently the main limitation of our proposed approach SelfPlayer and other exploration methods is that they are most useful in simulated environments as opposed to real world ones. Another limitation specific to our approach is the reliance on discrete representations, which are can be challenging to train. In future work, it would be promising to investigate continuous representation learning approaches as well as other ways of optimizing the self-play objective.

References

Achiam, J., Edwards, H., Amodei, D., and Abbeel, P. Variational option discovery algorithms. *CoRR*, abs/1807.10299, 2018. URL http://arxiv.org/abs/1807.10299.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. In Lee, D. D., Sugiyama, M., von Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 1471–1479, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/afda332245e2af431fb7b672a68b659d-Abstract.html.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., de Oliveira Pinto, H. P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019. URL http://arxiv.org/abs/1912.06680.

Burda, Y., Edwards, H., Pathak, D., Storkey, A. J., Darrell, T., and Efros, A. A. Large-scale study of curiosity-driven learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJNwDjAqYX.

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., and Grefenstette, E. Learning with amigo: Adversarially motivated intrinsic goals, 2021.

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., and Clune, J. First return then explore. *CoRR*, abs/2004.12919, 2020. URL https://arxiv.org/abs/2004.12919.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic goal generation for reinforcement learning agents. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1514–1523. PMLR, 2018. URL http://proceedings.mlr.press/v80/florensa18a.html.

Guo, Y., Choi, J., Moczulski, M., Feng, S., Bengio, S., Norouzi, M., and Lee, H. Memory based trajectory-conditioned policies for learning from sparse rewards. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/2df45244f09369e16ea3f9117ca45157-Abstract.html.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning, 2017.

Hester, T., Vecerík, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Osband, I., Dulac-Arnold, G., Agapiou, J. P., Leibo, J. Z., and Gruslys, A. Deep q-learning from demonstrations. In McIlraith, S. A. and Weinberger, K. Q. (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial*

Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pp. 3223–3230. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16976.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model-based reinforcement learning for atari. In *International Conference on Learning Representations*, 2020.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, 2018.

Machado, M. C., Bellemare, M. G., and Bowling, M. Count-based exploration with the successor representation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 5125–5133. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/5955.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

OpenAI, Plappert, M., Sampedro, R., Xu, T., Akkaya, I., Kosaraju, V., Welinder, P., D'Sa, R., Petron, A., de Oliveira Pinto, H. P., Paino, A., Noh, H., Weng, L., Yuan, Q., Chu, C., and Zaremba, W. Asymmetric self-play for automatic goal discovery in robotic manipulation. *CoRR*, abs/2101.04882, 2021. URL https://arxiv.org/abs/2101.04882.

Osband, I., Blundell, C., Pritzel, A., and Roy, B. V. Deep exploration via bootstrapped DQN. In Lee, D. D., Sugiyama, M., von Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 4026–4034, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/8d8818c8e140c64c743113f563cf750f-Abstract.html.

Oudeyer, P.-Y. and Kaplan, F. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1:6, 2009. ISSN 1662-5218. doi: 10.3389/neuro.12.006.2007. URL https://www.frontiersin.org/article/10.3389/neuro.12.006.2007.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787. PMLR, 2017. URL http://proceedings.mlr.press/v70/pathak17a.html.

Pathak, D., Gandhi, D., and Gupta, A. Self-supervised exploration via disagreement. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5062–5071. PMLR, 2019. URL http://proceedings.mlr.press/v97/pathak19a.html.

Pérolat, J., Strub, F., Piot, B., and Pietquin, O. Learning nash equilibrium for general-sum markov games from batch data. In Singh, A. and Zhu, X. J. (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pp. 232–241. PMLR, 2017. URL http://proceedings.mlr.press/v54/perolat17a.html.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with VQ-VAE-2. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 14837–14847, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/5f8e2fa1718d1bbcadf1cd9c7a54fb8c-Abstract.html.

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

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. Dynamics-aware unsupervised discovery of skills. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgLZR4KvH.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T. P., Simonyan, K., and Hassabis, D. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017a. URL http://arxiv.org/abs/1712.01815.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017b.

Srinivas, A., Laskin, M., and Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, 2020.

Sukhbaatar, S., Lin, Z., Kostrikov, I., Synnaeve, G., Szlam, A., and Fergus, R. Intrinsic motivation and automatic curricula via asymmetric self-play. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=SkT5Yg-RZ.

van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Yarats, D., Kostrikov, I., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

Zhang, Y., Abbeel, P., and Pinto, L. Automatic curriculum learning through value disagreement, 2020.