Robust Offline Learning via Adversarial World Models

Uljad Berdica[*] University of Oxford
Kelvin Li University of Oxford
Michael Beukman University of Oxford
Alexander D. Goldie University of Oxford
Perla Maiolino University of Oxford
Jakob N. Foerster University of Oxford

Abstract

Online reinforcement learning (RL), a setting in which an agent learns directly from interactions with the environment, has shown remarkable improvements with the advent of tools enabling faster and more efficient parallel training. However, learning a policy in an *offline* setting, where the agent is trained from trajectories collected earlier on an environment, has yet to make similar strides. World modelling, where a representation of the underlying environment is trained from offline data, enables an agent to sample trajectories from a learned model without ever interacting with the true environment. Policies learned in world models are often brittle as agents frequently learn to exploit inaccuracies in the world model rather than to behave according to the true underlying dynamics. Methods under the umbrella of Unsupervised Environment Design (UED) address robustness by designing a 'difficult but solvable' autocurriculum for the agent. Unfortunately, UED has been confined to simple environments definable by a set of selected parameters. This paper presents a novel approach that integrates the robustness of UED with the descriptive power of world models, achieving strong test-time performance in a range of environments while training using only offline data. Our approach provides benefits in both sparse and rich data regimes. As such, it offers huge potential for decision making in real-world settings without the need for any expensive real-world sampling.

1 Introduction

Recent advancements in machine learning have demonstrated the importance of exploiting large amounts of data. Generative models like large language models (e.g., (OpenAI et al., 2024; Touvron et al., 2023)), text-to-image models (e.g., (Imagen-Team-Google et al., 2024; Betker et al., 2023)) or text-to-video models (e.g., (Brooks et al., 2024)) have improved drastically in the past few years with the advent of larger models and, importantly, *more data* (Sutton, 2019). Reinforcement learning (Sutton and Barto, 2018, RL) is not an outlier to this phenomenon; the introduction of search, a compute- and data-intensive procedure, was paramount to the success of AlphaGo (Silver et al., 2016a). However, the same scale of success has not yet propagated to the majority of RL research.

One way to leverage data in RL—which generally consist of sequences of observations, actions and rewards tuples—is to train *world models* (Ha and Schmidhuber, 2018). World models are networks trained to be learned simulators of the real environment, attempting to replicate the transitions between sequential states. Importantly, world models can serve as generative models in RL settings. Rather than training a policy by continuously *sampling* from the real environment, which is expen-

[*]Correspondence to `uljadb@robots.ox.ac.uk`.

Conference on Neural Information Processing Systems - Open World Agents Workshop (NeurIPS 2024).

sive and potentially dangerous in real world settings, the policy can be learned by sampling from a world model trained on a pre-collected *offline* dataset.

While increasing the dataset size can improve the fidelity of learned world models, they are rarely perfect recreations of the underlying environment. Ha and Schmidhuber (2018) demonstrate how RL agents frequently learn to exploit discontinuities and edge cases in dynamics to receive large spikes in simulated reward. This drastically limits the transfer capabilities and robustness of an agent from the world model to the true environment.

On a separate note, recent methods under the class of regret-based *Unsupervised Environment Design* (UED) have emerged with the objective of improving an agent's robustness to challenging, or *worst-case*, dynamics (Dennis et al., 2020; Jiang et al., 2021a,b; Parker-Holder et al., 2022). UED is concerned with designing, or selecting, the next *levels* (i.e., environment configurations) an agent should learn on. In theory, UED produces agents that minimize their maximum regret (how far the agent is from an optimal policy on that level) over a space of levels (Dennis et al., 2020).

In our work, we combine the robustness characteristics of UED with the sample efficiency and effectiveness of world modelling. In particular, by training a *number of world models* on an offline RL dataset, each world model exhibits slightly different behaviours and can be adversarially sampled as a level within the UED framework. Using the robustness guarantees from UED (Dennis et al., 2020), and given that the true dynamics are within the support of the world models, our method minimizes the *worst-case* regret, effectively optimizing transfer performance. In addition to theoretical guarantees on the agent's transfer performance, we show that agents learned using our method empirically outperform those trained on world models or domain randomization alone.

2 Background

2.1 Reinforcement Learning

Markov Decision Process We consider a finite horizon Markov Decision Process (MDP). The MDP is defined by the tuple $\langle \mathcal{S}_0, \mathcal{S}, \mathcal{A}, T, \mathcal{R}, I, \mathcal{O} \rangle$. Here $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space and $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics, defining a distribution over next states given an initial state and action.[1] $\mathcal{S}_0$ denotes the initial state distribution. The scalar reward function is $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and the reward at time t for action a_t taken on state s_t is denoted as $r_t = \mathcal{R}(s_t, a_t)$. $\mathcal{I} : \mathcal{S} \to \Delta(O)$ is the observation function that maps states to observations; in a fully observable environment, the state and observation are equivalent. At time t, the agent has the observation o_t and takes action a_t. Thus, the actions-observation history or agent trajectory at time t is $\tau_t = \langle o_0, a_0, ..., o_{t-1}, a_{t-1}, o_t \rangle$.

A policy π maps an observation in $\mathcal{O}$ to an action distribution in $\mathcal{A}$. For a recurrent π, as is used in this work, the action is conditioned on the entire past trajectory such that $a_t \sim \pi(\tau_t)$. The policy is trained to maximize discounted episodic return J_M^π for a given MDP M with fixed episode length T:

$$J_M^\pi := \mathbb{E}_{a_{0:T} \sim \pi, s_0 \sim \mathcal{S}_0, s_{1:T} \sim T} \left[\sum_{t=0}^{T} r_t \right]. \tag{1}$$

Offline Reinforcement Learning This set of RL methods aims to maximize J^π using a dataset $\mathcal{D}$ consisting of a set of transition tuples $\{(s_n, a_n, r_n, s_{n+1})\}_{n=1}^N$ with $s_0 \sim \mathcal{S}_0$ as described above. $\mathcal{D}$ is collected by deploying a behavior policy π_b on the real environment. The behavior policy can have varying levels of expertise and exploration and $\mathcal{D}$ is therefore not guaranteed to have complete coverage over all of the environment states. The offline RL algorithm should always produce policies that achieve meaningful results in the true environment.

2.2 World Models

Definition As defined by Ha and Schmidhuber (2018), world models are compact representations of the dynamics of an environment which are independent of the agent's interactions with it; from the agent's perspective, a trained world model can be interacted with in the same way as the true

[1] $\Delta(\mathcal{X})$ is the set of all probability distributions over the set $\mathcal{X}$.

environment. In this work, we assume a one-step predictive world model. Thus, we make the assumption that the environment dynamics are Markovian. World models are generally represented using a neural network, denoted here as $\mathcal{F}_\theta$ with learnable parameters θ. Therefore, one-step transition dynamics of the environment can be modelled as $\hat{o}_{t+1}, \hat{r}_{t+1} \leftarrow \mathcal{F}_\theta(\hat{o}_t, a_t)$, where the world model predicts *both* the observation transition and the reward of the agent.

Errors and Hacking An agent trained entirely in a world model can also learn to take advantage of out-of-distribution or discontinuous areas of the world model's state space (Levine et al., 2020; Ha and Schmidhuber, 2018). These discontinuities *do not* exist in the true underlying environment, and thus the agent learns a policy which does not perform well in practice. Learned world models, much like other model-based RL methods, are also subject to compounding error where the difference between the world model outputs and the environment outputs diverge further as the episode proceeds Saleh et al. (2022). As a result, agents trained in fully offline long-horizon rollouts often lack robustness in transfer to the real environment. Our method empirically demonstrates strong performance when training on full-length episodes *entirely* inside world models.

2.3 Unsupervised Environment Design

Unsupervised environment design (UED) is a class of autocurriculum methods for RL, where an adversary proposes tasks (commonly referred to as *levels*) for an agent to train on. In one commonly-used paradigm (Dennis et al., 2020), environments are modelled as an Underspecified Partially Observable Markov Decision Process (UPOMDP) $\langle \mathcal{S}_0, \mathcal{S}, \mathcal{A}, T, \mathcal{R}, I, \mathcal{O}, \Theta \rangle$.

In a UPOMDP, Θ represents the set of *free parameters* of an environment which can be adjusted to change an environment's characteristics; θ denotes a specific instance of parameters, called a *level*, e.g., Θ could refer to block placement in a grid (Chevalier-Boisvert et al., 2023) with θ being their precise locations. Each θ instantiates a concrete POMDP.

While there are multiple ways to choose these tasks (Tobin et al., 2017; Matiisen et al., 2020; Florensa et al., 2018; Portelas et al., 2019), the recent approach of Minimax Regret (MMR) UED has emerged as a promising way to train robust agents (Dennis et al., 2020; Jiang et al., 2021a,b; Parker-Holder et al., 2022). Here, the adversary chooses levels that maximise the agent's *regret*, defined as: $U_\theta(\pi^*_\theta) - U_\theta(\pi)$, where U denotes the discounted sum of returns of a particular policy on the level θ, and π^*_θ is the optimal policy on θ.

Dennis et al. (2020) posed the UED setting as a two-player, zero-sum game between the adversary and the policy. Furthermore, they showed that if the adversary aims to maximize regret, and it is in Nash equilibrium with the policy, the policy satisfies the following equation:

$$\pi \in \arg\min_{\pi \in \Pi} \{ \arg\max_{\theta \in \Theta} \{ \text{Regret}_\theta(\pi) \} \}. \tag{2}$$

Therefore, the policy's minimizes its worst-case regret. This confers a degree of robustness to the policy, as its regret in any level $\theta \in \Theta$ must be below this bound.

2.4 Prioritized Level Replay

Prioritized Level Replay (Jiang et al., 2021a) is an empirically successful curriculum method that relies on curating high-scoring levels. In practice, PLR maintains a buffer of previous high-scoring levels, and either samples from this buffer, or samples new levels. The agent is then rolled out on these new levels, and they are scored depending on how the agent performed. High-scoring levels are then also added to the buffer, and the agent trains on the collected experience. It is not necessary to sample new levels, however; in some cases, the level set may be predefined and fixed (Tzannetos et al., 2024).

Scoring functions The original PLR scores each level θ_i using a time-averaged L_1 value loss of each agent's last trajectory on the level (Jiang et al., 2021a). In order to achieve *minimax robustness*, a scoring function should account for regret as described in Section 2.3. Jiang et al. (2021b) propose different scoring functions that more closely approximate the regret. Ultimately, the choice of a scoring function is a design choice depending on the nature of the environment. We further elaborate on the scoring function choices in section 3.

3 Prioritized World-Model Replay: Robustness Through Adversity

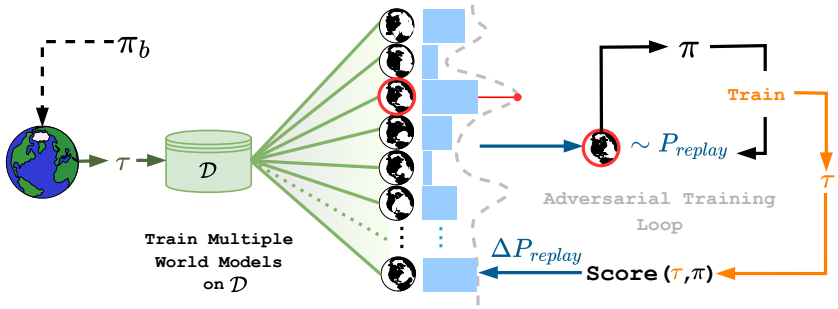


Figure 1: An overview of Prioritized world-model replay. Dataset $\mathcal{D}$ is collected by training a behavior policy in the real world (left). The data is used to train multiple world models. A probability distribution P_{replay} (middle) over the different world models is updated as different world models are sampled during the fully offline adversarial training loop of policy π (right).

For our method illustrated in Figure 1, we start by training a collection of world models consistent with the provided data. We then treat these models as *levels* and apply UED to train a robust, transferable policy. We make the empirically justified assumption that our neural network architecture is expressive enough to capture the true environment dynamics, enabling minimax regret robustness with respect to these dynamics.

3.1 Training Multiple World Models

Given some finite dataset of transitions from the environment, there are multiple possible world models θ that fit this data reasonably well (say, e.g., with a loss $L_2 < \epsilon$) and, by our assumption, the true dynamics is also within this set; we denote this as θ^*. We leverage this by training multiple distinct world models, using different random initializations of the neural network, and training on different permutations of the data. Due to the variability of stochastic gradient descent used to update the weights of the world model, each of these will have slightly different dynamics (Amari, 1993). However, an agent trained in any one of these world models is not guaranteed to transfer well to the real environment, and it is this problem we tackle through UED where the world models effectively serve as our level buffer.

3.2 World Models as levels for adversarial training

If we treat each world model θ as a *level*, we can apply standard minimax regret algorithms to our setting. More formally, we consider the two-player game between an adversary G and student policy π, such that the adversary generates a level (i.e., a world model) $\theta \in \Theta$ that minimizes the agent's regret, and the agent trains as normal on the provided levels. Note, we define $\Theta \doteq \{\theta : L_2(\theta, \mathcal{D}) < \epsilon\}$ to be the set of all world models that have loss over the dataset $\mathcal{D}$ of less than some threshold ϵ. At Nash equilibrium of this game, Dennis et al. (2020) showed that the policy satisfies Equation (2). In other words, the policy's maximum regret on any $\theta \in \Theta$ is bounded by $W \doteq \min_{\pi \in \Pi}\{\max_{\theta \in \Theta}\{\text{Regret}_\theta(\pi)\}\}$. Since we have assumed that $\theta^* \in \Theta$, *this bound further applies to the true environment dynamics*. Moreover, since the adversary is constrained to only choose levels within Θ, i.e., those that have loss less than a certain value, it cannot be overly adversarial and provide totally unrealistic dynamics to train the agent on.

In order to make this procedure practical, we use the high-performing PLR algorithm, treating different world models θ as levels. Despite PLR not guaranteeing convergence to a Nash equilibrium, it generally results in improved zero-shot generalisation to out-of-distribution tasks. Jiang et al. (2021b) show that the original L_1 value loss on Jiang et al. (2021a) can bias towards high variance policies and propose alternative scoring functions that instead approximate regret. Since regret for a given world model is not always known, we use the standard regret approximations of Positive Value Loss for level θ_i: $S_i = \frac{1}{T}\sum_{t=0}^{T} \max\left(\sum_{k=t}^{T}(\gamma\lambda)^{k-t}\delta_k, 0)\right)$.

4 Experimental Setup

This section outlines our experimental setup, covering offline data collection, world model training, RL process details, and baseline descriptions.

4.1 Dataset Curation and World Model Training

One factor guiding our dataset curation strategy is the notion of *state coverage* due to the reasons states in 2.2. Any one particular behaviour policy π_b may only explore a small subset of the full state space. Therefore, we use multiple behaviour policies to collect the data (illustration on A.2). In particular, we train an agent in the real environment using Proximal Policy Optimization (PPO) (Schulman et al., 2017), and periodically checkpoint it throughout training, and use these checkpoints as our behaviour policies. This means that the data in our dataset is collected from a range of policies, spanning from randomly initialized to solving the task at hand.

In order to test our method for different numbers of collected transitions, we uniformly subsample the transitions dataset $\mathcal{D}$ to varying sizes. We then use different seeds to initialize the world model network, and shuffle the order the data is passed through it. More training details can results can be found on Table 1 in Appendix A.3.

4.2 Reinforcement Learning Training

The agent is implemented as a recurrent actor-critic network following the template from Pure-JaxRL (Lu et al., 2022). The agent's actions depend on the current observation and episode trajectory, implemented as the recurrent state of the actor-critic network. We use the recurrent state to test the agent's ability to perform system identification across the world models it is trained on. This is also done to verify that the world models have distinct dynamics. Hidden state visualization can be found in A.5.

We implement a suite of adversarial methods to train on the world models: Prioritized Level Replay (**PLR**) as described in section 2.4 with an L_1 value loss function, PLR with a Positive Value Loss (**PLR_PVL**) scoring as described in 3.2 to approximate regret, Domain Randomization (**DR**) implemented by randomly selecting a new world model θ from a uniform distribution over the world model buffer Θ_{train} and **DR_STEP** where we change θ_i at every individual timestep of the agent in a fixed length episode instead of only doing it at every reset. We use training on only *a single world model* (**WM**) without any adversity or curriculum curation as a baseline for the aforementioned methods.

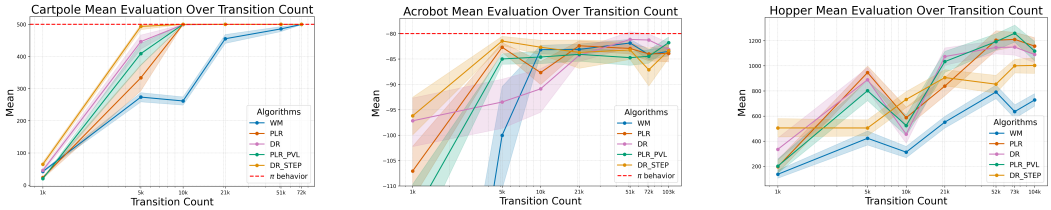
The PLR implementations are based on JaxUED (Coward et al., 2024). We evaluate the policies on the same environment the world model training dataset was collected in and use RLiable library as presented by Agarwal et al. (2021) to measure the performance. The entire pipeline, from data collection to wold model and subsequent policy training is implemented in the JAX Ecosystem (DeepMind et al., 2020) to leverage hardware acceleration and speed up training.

5 Results

In this section we show the most notable results that elucidate important aspect of our approach. A complete compilation of the results can be found in the Appendix. We collect data from and evaluate on environments from the Gymnax (Lange, 2022) and Brax (Freeman et al., 2021) suites. All the evaluations are performed on full trajectories across 10 random seeds on the corresponding real environments.

In Figure 2a, all the methods trained using the adversarial world model selection reach the highest episodic return possible on Cartpole in less than half the transitions counts than the single world model. Training on multiple world models beats the single world models baseline in a simple environment. Figure 2b shows our methods consistently outperform training on a single world model for sparser data and even achieve returns higher than the behavior policy that was learned online. The effectiveness of our approach is further supported by the results on Hopper (Figure 2c). PLR_PVL on top of other adversarial training methods perform better than single world models across transi-

tion counts. This shows that for increasingly realistic tasks and higher dataset sizes, methods using regret approximation for robustness are the most promising.



(a) Evaluation on Cartpole (b) Evaluation on Acrobot (c) Evaluation on Hopper

Figure 2: Comparison of Evaluation Metrics Across Different Environments

Additionally, we demonstrate signs of overall policy improvement on Pendulum-v1 through three common metrics in RL evaluation with their respective 95 % confidence intervals. On $19 \cdot 10^3$ transitions, the interquartile mean (IQM) of the episodic returns is already higher than that of the behavior policy π_b in Figure 3. As the number of transitions increases, the regret-based prioritization performs best and becomes more likely to match or exceed the performance of the online-trained behavior policy indicated by the red line.

Our claim is that the world models have sufficiently distinct dynamics and can therefore serve as levels in the UED framework. If true, regret-based training should help the agent adapt to all these dynamics. We demonstrate this by deploying our agent across multiple world models and on the real environment. We then train a classifier on the recurrent states and achieve an average of 62% accuracy on the **DR**, 60% on **PLR** and 45% on **PLR_PVL**; all above the 10% random prediction accuracy. More in A.6,A.5.

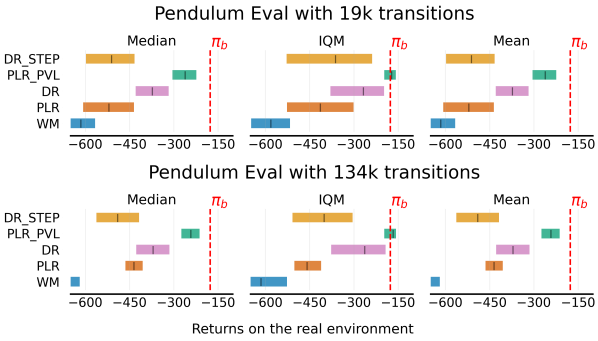


Figure 3: Interquartile Mean (IQM), Mean, and Median of the offline trained policy evaluated on the real environment

6 Future Work and Limitations

Regardless of our theoretical guarantees and empirical results showing the scaling of performance with larger datasets on more difficult environments, our problem setting requires more rigorous theoretical analysis of the underlying assumptions. We assume that a neural network is sufficiently expressive to model the real dynamics but lack a more specific definition of such 'modeling capacity'.

Moreover, this work would benefit from a more principled and interpretable method of sampling the possible world models from Θ set – as defined in 3.2 – other than simply changing the shuffling and initialization seeds. A natural extension is that of level generation to have an expanding buffer of available levels during the adversarial training.

The results in physical engines like Brax should be extended to *real* physical platforms and address the engineering challenges posed by the sim2real gap, especially in sensitive settings where online training can be physically hazardous.

7 Conclusion

In this work we present a novel way to guarantee transfer robustness to the real environment over world models fitted on offline data. To the best of our knowledge, this is the first work that performs adversarial training under this specific fully offline and fully paramteric constraint. Our method naturally lends itself to other architectures and vectors of adversity and can blaze the trails towards meaningful deployment of state-of-the-art RL algorithms into the *real* world.

References

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, March 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023.

Imagen-Team-Google, Jason Baldridge, Jakob Bauer, Mukul Bhutani, Nicole Brichtova, Andrew Bunner, Kelvin Chan, Yichang Chen, Sander Dieleman, Yuqing Du, Zach Eaton-Rosen, Hongliang Fei, Nando de Freitas, Yilin Gao, Evgeny Gladchenko, Sergio Gómez Colmenarejo, Mandy Guo, Alex Haig, Will Hawkins, Hexiang Hu, Huilian Huang, Tobenna Peter Igwe, Christos Kaplanis, Siavash Khodadadeh, Yelin Kim, Ksenia Konyushkova, Karol Langner, Eric Lau,

Shixin Luo, Soňa Mokrá, Henna Nandwani, Yasumasa Onoe, Aäron van den Oord, Zarana Parekh, Jordi Pont-Tuset, Hang Qi, Rui Qian, Deepak Ramachandran, Poorva Rane, Abdullah Rashwan, Ali Razavi, Robert Riachi, Hansa Srinivasan, Srivatsan Srinivasan, Robin Strudel, Benigno Uria, Oliver Wang, Su Wang, Austin Waters, Chris Wolff, Auriel Wright, Zhisheng Xiao, Hao Xiong, Keyang Xu, Marc van Zee, Junlin Zhang, Katie Zhang, Wenlei Zhou, Konrad Zolna, Ola Aboubakar, Canfer Akbulut, Oscar Akerlund, Isabela Albuquerque, Nina Anderson, Marco Andreetto, Lora Aroyo, Ben Bariach, David Barker, Sherry Ben, Dana Berman, Courtney Biles, Irina Blok, Pankil Botadra, Jenny Brennan, Karla Brown, John Buckley, Rudy Bunel, Elie Bursztein, Christina Butterfield, Ben Caine, Viral Carpenter, Norman Casagrande, Ming-Wei Chang, Solomon Chang, Shamik Chaudhuri, Tony Chen, John Choi, Dmitry Churbanau, Nathan Clement, Matan Cohen, Forrester Cole, Mikhail Dektiarev, Vincent Du, Praneet Dutta, Tom Eccles, Ndidi Elue, Ashley Feden, Shlomi Fruchter, Frankie Garcia, Roopal Garg, Weina Ge, Ahmed Ghazy, Bryant Gipson, Andrew Goodman, Dawid Górny, Sven Gowal, Khyatti Gupta, Yoni Halpern, Yena Han, Susan Hao, Jamie Hayes, Amir Hertz, Ed Hirst, Tingbo Hou, Heidi Howard, Mohamed Ibrahim, Dirichi Ike-Njoku, Joana Iljazi, Vlad Ionescu, William Isaac, Reena Jana, Gemma Jennings, Donovon Jenson, Xuhui Jia, Kerry Jones, Xiaoen Ju, Ivana Kajic, Christos Kaplanis, Burcu Karagol Ayan, Jacob Kelly, Suraj Kothawade, Christina Kouridi, Ira Ktena, Jolanda Kumakaw, Dana Kurniawan, Dmitry Lagun, Lily Lavitas, Jason Lee, Tao Li, Marco Liang, Maggie Li-Calis, Yuchi Liu, Javier Lopez Alberca, Peggy Lu, Kristian Lum, Yukun Ma, Chase Malik, John Mellor, Inbar Mosseri, Tom Murray, Aida Nematzadeh, Paul Nicholas, João Gabriel Oliveira, Guillermo Ortiz-Jimenez, Michela Paganini, Tom Le Paine, Roni Paiss, Alicia Parrish, Anne Peckham, Vikas Peswani, Igor Petrovski, Tobias Pfaff, Alex Pirozhenko, Ryan Poplin, Utsav Prabhu, Yuan Qi, Matthew Rahtz, Cyrus Rashtchian, Charvi Rastogi, Amit Raul, Ali Razavi, Sylvestre-Alvise Rebuffi, Susanna Ricco, Felix Riedel, Dirk Robinson, Pankaj Rohatgi, Bill Rosgen, Sarah Rumbley, Moonkyung Ryu, Anthony Salgado, Sahil Singla, Florian Schroff, Candice Schumann, Tanmay Shah, Brendan Shillingford, Kaushik Shivakumar, Dennis Shtatnov, Zach Singer, Evgeny Sluzhaev, Valerii Sokolov, Thibault Sottiaux, Florian Stimberg, Brad Stone, David Stutz, Yu-Chuan Su, Eric Tabellion, Shuai Tang, David Tao, Kurt Thomas, Gregory Thornton, Andeep Toor, Cristian Udrescu, Aayush Upadhyay, Cristina Vasconcelos, Alex Vasiloff, Andrey Voynov, Amanda Walker, Luyu Wang, Miaosen Wang, Simon Wang, Stanley Wang, Qifei Wang, Yuxiao Wang, Ágoston Weisz, Olivia Wiles, Chenxia Wu, Xingyu Federico Xu, Andrew Xue, Jianbo Yang, Luo Yu, Mete Yurtoglu, Ali Zand, Han Zhang, Jiageng Zhang, Catherine Zhao, Adilet Zhaxybay, Miao Zhou, Shengqi Zhu, Zhenkai Zhu, Dawn Bloxwich, Mahyar Bordbar, Luis C. Cobo, Eli Collins, Shengyang Dai, Tulsee Doshi, Anca Dragan, Douglas Eck, Demis Hassabis, Sissie Hsiao, Tom Hume, Koray Kavukcuoglu, Helen King, Jack Krawczyk, Yeqing Li, Kathy Meier-Hellstern, Andras Orban, Yury Pinsky, Amar Subramanya, Oriol Vinyals, Ting Yu, and Yori Zwols. Imagen 3, August 2024.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. Improving Image Generation with Better Captions. 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

Richard Sutton. The Bitter Lesson. http://www.incompleteideas.net/IncIdeas/BitterLesson.html, March 2019.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0-262-03924-9.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016a. ISSN 1476-4687. doi: 10.1038/nature16961.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in neural information processing systems*, 33:13049–13061, 2020.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *International Conference on Machine Learning*, pages 4940–4950. PMLR, 2021a.

Minqi Jiang, Michael Dennis, Jack Parker-Holder, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Replay-guided adversarial environment design. *Advances in Neural Information Processing Systems*, 34:1884–1897, 2021b.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. In *Proceedings of the International Conference on Machine Learning*, pages 17473–17498. PMLR, 2022. URL https://proceedings.mlr.press/v162/parker-holder22a.html.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Esra' Saleh, John D Martin, Anna Koop, Arash Pourzarabi, and Michael Bowling. Should models be accurate? *arXiv preprint arXiv:2205.10736*, 2022.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems 36, New Orleans, LA, USA*, December 2023.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *International Conference on Intelligent Robots and Systems*, pages 23–30. IEEE, 2017. doi: 10.1109/IROS.2017.8202133. URL https://doi.org/10.1109/IROS.2017.8202133.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-student curriculum learning. volume 31, pages 3732–3740, 2020. doi: 10.1109/TNNLS.2019.2934906. URL https://doi.org/10.1109/TNNLS.2019.2934906. Classification=important; Field=ued; Tags=ued.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *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*, pages 1514–1523. PMLR, 2018. URL http://proceedings.mlr.press/v80/florensa18a.html.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep RL in continuously parameterized environments. In *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*, volume 100 of *Proceedings of Machine Learning Research*, pages 835–853. PMLR, 2019. URL http://proceedings.mlr.press/v100/portelas20a.html.

Georgios Tzannetos, Parameswaran Kamalaruban, and Adish Singla. Proximal curriculum with task correlations for deep reinforcement learning. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 5027–5036. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/556. URL https://doi.org/10.24963/ijcai.2024/556. Main Track.

Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5 (4-5):185–196, 1993.

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

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 35:16455–16468, 2022.

Samuel Coward, Michael Beukman, and Jakob Foerster. Jaxued: A simple and useable ued library in jax. *arXiv preprint arXiv:2403.13091*, 2024.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020.

Robert Tjarko Lange. gymnax: A JAX-based reinforcement learning environment library, 2022.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016b.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

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.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pages 1282–1289. PMLR, 2019.

Ishita Mediratta, Qingfei You, Minqi Jiang, and Roberta Raileanu. The generalization gap in offline reinforcement learning. *arXiv preprint arXiv:2312.05742*, 2023.

Leonard J Savage. The theory of statistical decision. *Journal of the American Statistical association*, 46(253):55–67, 1951.

Samuel Kessler, Mateusz Ostaszewski, MichałPaweł Bortkiewicz, Mateusz Żarski, Maciej Wolczyk, Jack Parker-Holder, Stephen J Roberts, Piotr Mi, et al. The effectiveness of world models for continual reinforcement learning. In *Conference on Lifelong Learning Agents*, pages 184–204. PMLR, 2023.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Marc Rigter, Minqi Jiang, and Ingmar Posner. Reward-free curricula for training robust world models. *arXiv preprint arXiv:2306.09205*, 2023.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International conference on machine learning*, pages 8583–8592. PMLR, 2020.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning, 2020.

Aviral Kumar, Anikait Singh, Stephen Tian, Chelsea Finn, and Sergey Levine. A workflow for offline model-free robotic reinforcement learning. *arXiv preprint arXiv:2109.10813*, 2021.

Matthew Thomas Jackson, Michael Tryfan Matthews, Cong Lu, Benjamin Ellis, Shimon Whiteson, and Jakob Foerster. Policy-guided diffusion. *arXiv preprint arXiv:2404.06356*, 2024.

Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *Advances in neural information processing systems*, 31, 2018.

A Appendix

A.1 Related Work

Reinforcement Learning has achieved impressive results, some of the most notable ones being Go (Silver et al., 2016b), Starcraft (Vinyals et al., 2019), Atari (Mnih et al., 2015) and more recent advances focusing on multi-task generalizations (Bruce et al., 2024; Hafner et al., 2023). Despite these impressive results, RL methods fail to generalize to settings even slightly different than the training environments (Cobbe et al., 2019; Mediratta et al., 2023), indicating that the generalization to real world settings remains an open challenge.

One promising way to increase the generalization capabilities of an agent trained via RL is to ensure that the agent is exposed to a sufficiently diverse set of environments in training time. The Unsupervised Environment Design (UED) (Dennis et al., 2020; Jiang et al., 2021b) line of work achieves this by relaxing the definition of the environment to a combinatorially large set of possible configurations captured by a set of parameters, commonly referred to as *levels*. The choice of the parameter space is specifically tailored to the general task domain also known as the underspecified environments (e.g. a maze environment is parameterized by the placement of the walls, start and goal position whereas a one dimensional bipedal environment is parameterized by the roughness of the terrain). UED uses Minimax regret (Savage, 1951) to make the agent robust to the most challenging environment configurations without prior knowledge of which set of parameters it will act in. While these approaches are meant to exemplify deployment in challenging situations, they remain reliant on semantically informed choices of parameters that capture useful *levels* of difficulty (the color of the background is not as useful in curating the training of a bipedal walker as the roughness of the train). Jiang et al. (2021b) was very helpful in bridging the intuitive algorithm of Prioritized Level Replay with new regret approximations that provide minimax guarantees.

World models (Ha and Schmidhuber, 2018) propose a different approach where the agent is equipped with a compact representation of the real environments trained using a dataset of transitions in said environment. More recent work shows that world models can serve as task-agnostic Continual Reinforcement Learning baselines (Kessler et al., 2023) or used in online RL to achieve human-level performance on Atari (Hafner et al., 2020). In principles, world modelling does not hinge on task-specific heuristics and only relies on increasing the robustness of the agent by tuning the uncertainty inside the world model.

A recent combination of the world model and *Minimax Regret* approach by Rigter et al. (2023) trains a world model that can derive robust policies. This is done through an exploration policy seeking maximal model uncertainty, similar to the self-supervised world model methods by Sekar et al. (2020). These methods require sufficient exploration of states that can be physically dangerous to the agent and disrupt operation altogether (Kumar et al., 2020, 2021).

Offline RL work has provided a useful signal on the importance of using offline datasets (Kumar et al., 2020, 2021), the common challenges that arise form the distribution shift between the behavior and learned policy (Levine et al., 2020) and model error (Saleh et al., 2022) alongside the most common workarounds like truncated rollouts (Jackson et al., 2024). Model-based offline (Rigter et al., 2022) and online (Chua et al., 2018) RL methods have served as useful blueprints to manage uncertainty through *multiple* dynamic models.

The work of Li and Liang (2018) and the foundational work of Amari (1993) have paved the intuition that shuffling the data and most importantly, changing the initializations, would be effective in training sufficiently distinct models on a shared offline dataset.

A.2 Dataset Curation

The checkpointing frequency and number of trajectories collected per checkpoint is heuristically determined by the environment's episode length and the monotonicity of the training and evaluation curves. Figure 4 illustrates the frequency at which the behavior policy trajectories are collected for the Hopper environment to capture different levels of agent's abilities.

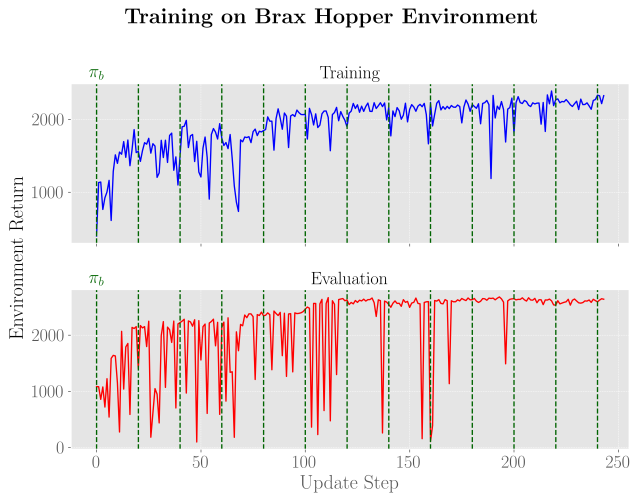


Figure 4: Data collection using different π_b checkpoints marked by the vertical lines are used to collect trajectories for $\mathcal{D}$

A.3 World Model Training Results

The world models are trained on the same data with an L_2 test loss but show different final test losses and slightly different dynamics.

We set the number of gradient updates for each world model to be the same to ensure consistent levels of training across dataset sizes. The world models in our experiments are implemented as fully connected networks with a concatenated input of actions and observations and an output of the concatenated next observations and reward. At inference time, we add Gaussian Noise to the outputs equal to the square root of the L_2 test loss, to represent the fact that the world models are modelling a distribution over next states.

Table 1: Training results for different $\mathcal{D}$ ratios across environment

Environment	% **of** $\lvert\mathcal{D}\rvert$	**Train Mean**	**Train Median**	**Test Mean**	**Test Median**
Pendulum-v1	1	$1.201 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$	$5.87 \cdot 10^{-4}$	$5.83 \cdot 10^{-4}$
	5	$2.20 \cdot 10^{-6}$	$2.19 \cdot 10^{-6}$	$5.93 \cdot 10^{-5}$	$5.91 \cdot 10^{-5}$
	10	$4.28 \cdot 10^{-6}$	$4.39 \cdot 10^{-6}$	$3.02 \cdot 10^{-5}$	$3.01 \cdot 10^{-5}$
	20	$6.85 \cdot 10^{-6}$	$6.90 \cdot 10^{-6}$	$1.87 \cdot 10^{-5}$	$1.86 \cdot 10^{-5}$
	50	$9.35 \cdot 10^{-6}$	$9.34 \cdot 10^{-6}$	$1.33 \cdot 10^{-5}$	$1.34 \cdot 10^{-5}$
	70	$3.99 \cdot 10^{-1}$	$1.02 \cdot 10^{-5}$	$4.08 \cdot 10^{-1}$	$1.28 \cdot 10^{-5}$
	100	$3.99 \cdot 10^{-1}$	$1.11 \cdot 10^{-5}$	$4.08 \cdot 10^{-1}$	$1.23 \cdot 10^{-5}$
Acrobot	1	$8.86 \cdot 10^{-7}$	$9.11 \cdot 10^{-7}$	$1.20 \cdot 10^{-2}$	$1.20 \cdot 10^{-2}$
	5	$7.53 \cdot 10^{-6}$	$7.35 \cdot 10^{-6}$	$2.55 \cdot 10^{-3}$	$2.57 \cdot 10^{-3}$
	10	$1.71 \cdot 10^{-5}$	$1.69 \cdot 10^{-5}$	$1.17 \cdot 10^{-3}$	$1.18 \cdot 10^{-3}$
	20	$3.37 \cdot 10^{-5}$	$3.37 \cdot 10^{-5}$	$5.05 \cdot 10^{-4}$	$5.05 \cdot 10^{-4}$
	50	$7.60 \cdot 10^{-5}$	$7.60 \cdot 10^{-5}$	$3.01 \cdot 10^{-4}$	$3.02 \cdot 10^{-4}$
	70	$9.14 \cdot 10^{-5}$	$9.09 \cdot 10^{-5}$	$2.67 \cdot 10^{-4}$	$2.66 \cdot 10^{-4}$
	100	$1.40 \cdot 10^{-4}$	$1.39 \cdot 10^{-4}$	$2.81 \cdot 10^{-4}$	$2.81 \cdot 10^{-4}$
Cartpole	1	$1.95 \cdot 10^{-8}$	$1.86 \cdot 10^{-8}$	$3.57 \cdot 10^{-5}$	$3.60 \cdot 10^{-5}$
	5	$2.97 \cdot 10^{-7}$	$2.89 \cdot 10^{-7}$	$4.20 \cdot 10^{-6}$	$4.15 \cdot 10^{-6}$
	10	$4.86 \cdot 10^{-7}$	$4.85 \cdot 10^{-7}$	$2.22 \cdot 10^{-6}$	$2.23 \cdot 10^{-6}$
	20	$6.49 \cdot 10^{-7}$	$6.47 \cdot 10^{-7}$	$1.52 \cdot 10^{-6}$	$1.52 \cdot 10^{-6}$
	50	$8.05 \cdot 10^{-7}$	$8.03 \cdot 10^{-7}$	$1.15 \cdot 10^{-6}$	$1.14 \cdot 10^{-6}$
	70	$8.61 \cdot 10^{-7}$	$8.61 \cdot 10^{-7}$	$1.08 \cdot 10^{-6}$	$1.08 \cdot 10^{-6}$
	100	$8.98 \cdot 10^{-7}$	$8.98 \cdot 10^{-7}$	$1.05 \cdot 10^{-6}$	$1.04 \cdot 10^{-6}$
Hopper	1	$7.51 \cdot 10^{-7}$	$7.33 \cdot 10^{-7}$	$2.03 \cdot 10^{-2}$	$2.01 \cdot 10^{-2}$
	5	$1.22 \cdot 10^{-4}$	$1.22 \cdot 10^{-4}$	$1.19 \cdot 10^{-2}$	$1.18 \cdot 10^{-2}$
	10	$4.21 \cdot 10^{-4}$	$4.13 \cdot 10^{-4}$	$9.79 \cdot 10^{-3}$	$9.77 \cdot 10^{-3}$
	20	$8.29 \cdot 10^{-4}$	$8.29 \cdot 10^{-4}$	$7.66 \cdot 10^{-3}$	$7.66 \cdot 10^{-3}$
	50	$1.37 \cdot 10^{-3}$	$1.37 \cdot 10^{-3}$	$6.33 \cdot 10^{-3}$	$6.33 \cdot 10^{-3}$
	70	$1.72 \cdot 10^{-3}$	$1.77 \cdot 10^{-3}$	$5.68 \cdot 10^{-3}$	$5.68 \cdot 10^{-3}$
	100	$2.27 \cdot 10^{-3}$	$2.27 \cdot 10^{-3}$	$5.06 \cdot 10^{-3}$	$5.06 \cdot 10^{-3}$
Mountaincar-Discrete	1	$5.68 \cdot 10^{-8}$	$4.76 \cdot 10^{-8}$	$1.15 \cdot 10^{-6}$	$1.10 \cdot 10^{-6}$
	5	$4.44 \cdot 10^{-8}$	$4.49 \cdot 10^{-8}$	$1.44 \cdot 10^{-7}$	$1.40 \cdot 10^{-7}$
	10	$1.03 \cdot 10^{-7}$	$1.02 \cdot 10^{-7}$	$1.51 \cdot 10^{-7}$	$1.50 \cdot 10^{-7}$
	20	$1.19 \cdot 10^{-7}$	$1.16 \cdot 10^{-7}$	$1.27 \cdot 10^{-7}$	$1.24 \cdot 10^{-7}$
	50	$1.28 \cdot 10^{-7}$	$1.28 \cdot 10^{-7}$	$1.25 \cdot 10^{-7}$	$1.26 \cdot 10^{-7}$
	70	$1.26 \cdot 10^{-7}$	$1.20 \cdot 10^{-7}$	$1.23 \cdot 10^{-7}$	$1.18 \cdot 10^{-7}$
	100	$1.20 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$	$1.08 \cdot 10^{-7}$	$1.08 \cdot 10^{-7}$

A.4 Hyperparameters

Table 2: Hyperparameters for the world model training

Hyperparameter	Value
Learning Rate	$1 \cdot 10^{-4}$
Batch Size	64
Hidden Size	256
Gradient Updates	$5 \cdot 10^{3}$

Table 3: Hyperparameter Ranges Considered for Each RL Environment

Hyperparameter	Acrobot	CartPole	Hopper	MountainCar	Pendulum
Learning Rate	$5 \cdot 10^{-4}$	$2.5 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$1 \cdot 10^{-3}$	$1 \cdot 10^{-3}$
Number of Environments	16	4	512	16	32
Total Timesteps	$5 \cdot 10^{5}$	$5 \cdot 10^{5}$	$5 \cdot 10^{8}$	$5 \cdot 10^{5}$	$1 \cdot 10^{7}$
PPO Update Epochs	4	4	4	64	4
Number of Minibatches	4	4	32	4	4
Gamma	0.99	0.99	0.99	0.99	0.99
GAE Lambda	0.95	0.95	0.95	0.95	0.95
Clip EPS	0.2	0.2	0.2	0.2	0.2
Entropy Coefficient	0.01	0.01	0.0	0.003	0.01
Value Function Coef	0.5	0.5	0.5	0.5	0.5
Max Grad Norm	1	0.5	0.5	1	1.0
Activation Function	tanh	tanh	tanh	tanh	tanh
Anneal Learning Rate	true	true	false	true	true
Number of Eval Envs	1	1	1	1	1
Eval Frequency	4	4	100	4	4

A.5 Hidden States Visualization

The PCA for RNN states of different agents trained with different algorithms

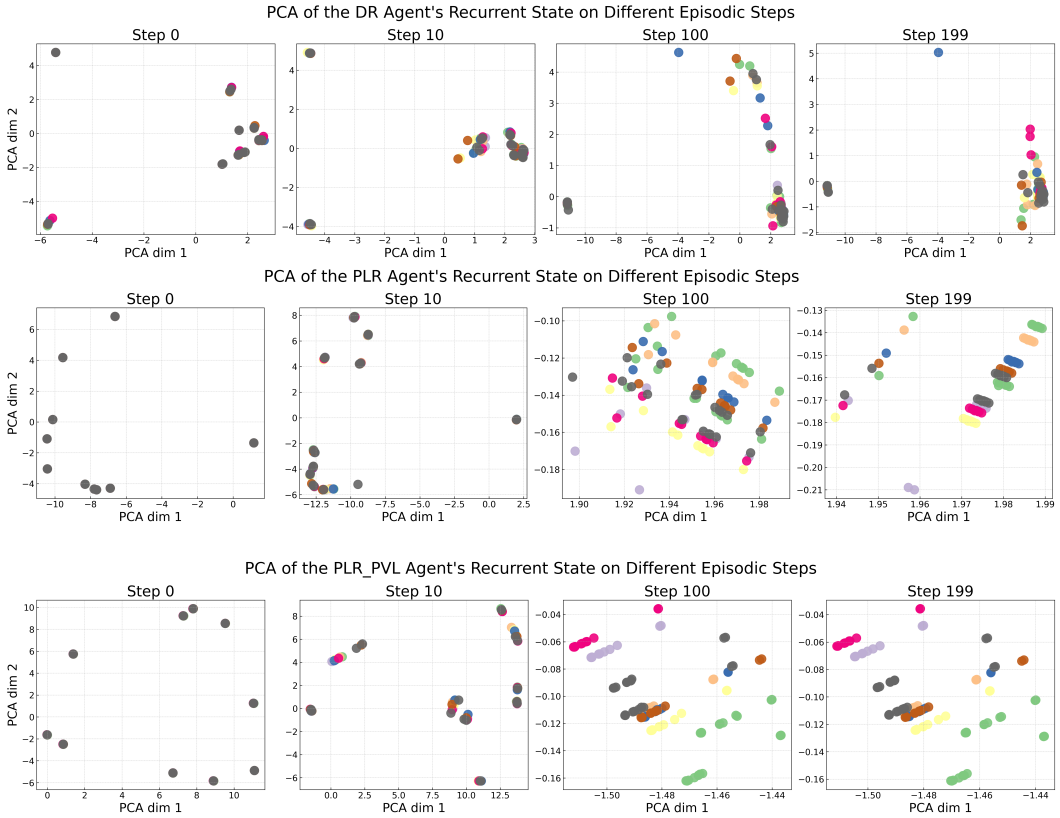


Figure 5: PCA of the hidden recurrent state for agents trained on different algorithms

Each row illustrates the episodic progression, with Figure 5 depicting the 2-dimensional Principal Component Analysis (PCA) of the 256-dimensional hidden states. These hidden states are collected from 10 differently initialized rollouts of the same agent. The rollouts are performed across 9 different world models and the real environment, ensuring a fair and balanced classification dataset. Notably, no pattern of stability emerges with the **DR**-trained agent. However, the **PLR** and **PLR_PVL** agents exhibit stabilization midway through the episode, within a smaller range on the principal components compared to the PCA of their initial state. While this warrants further investigation, we can intuitively infer that the agent learns to act optimally across all world models, and that this optimal behavior tends to become increasingly similar—**though still distinct**—across the different world models and environments.

A.6 Hidden States Classification

The confusion matrix for the classification of the world model using the agent's recurrent state from all the steps of the episode.

Table 4: Classification accuracy of 9 world models and the real environment. The ones below the 10% random guessing are marked in red

% of $\lvert\mathcal{D}\rvert$	DR	PLR	PLR_PVL
1	0.68325	0.10875	0.47375
5	0.4115	0.65275	0.67200
10	0.62325	0.68325	0.39975
20	0.67425	0.67075	0.08525
50	0.7595	0.66225	0.35775
70	0.67875	0.58225	0.36850
100	0.54025	0.85000	0.79000

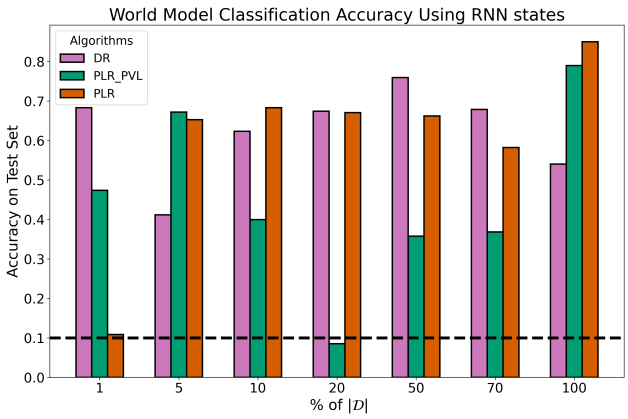
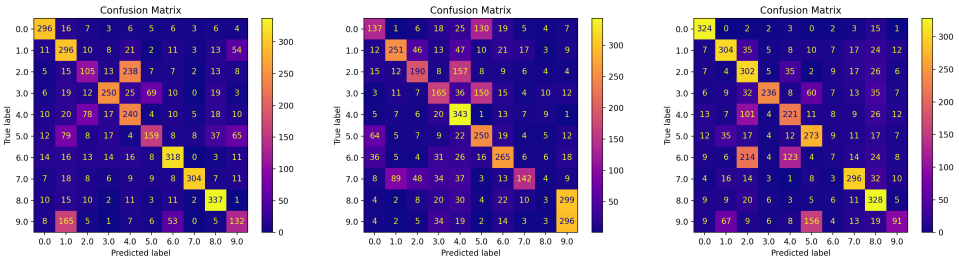


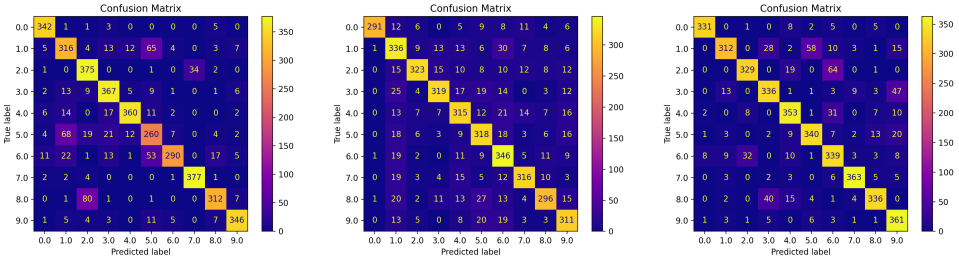
Figure 6: Classification accuracy of the hidden states from agents trained with DR, PLR, and PLR_PVL for a dataset of trajectories from 9 world models and the real environment. The dashed black line is the random prediction accuracy for the 10 classes.



(a) DR on 5% of $|\mathcal{D}|$

(b) DR on 50% of $|\mathcal{D}|$

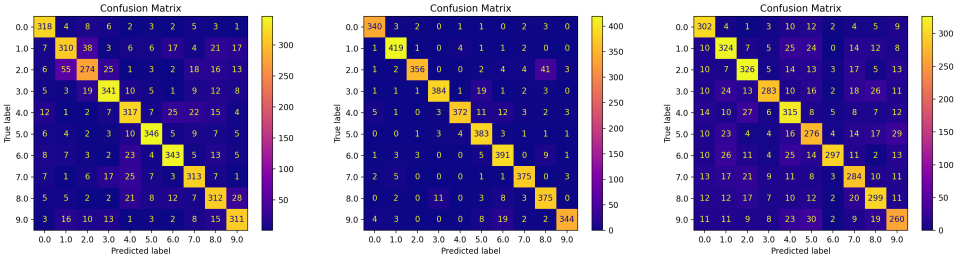
(c) DR on 100% of $|\mathcal{D}|$



(d) PLR on 5% of $|\mathcal{D}|$

(e) PLR on 50% of $|\mathcal{D}|$

(f) PLR on 100% of $|\mathcal{D}|$



(g) PLR_PVL on 5% of $|\mathcal{D}|$

(h) PLR_PVL on 50% of $|\mathcal{D}|$

(i) PLR_PVL on 100% of $|\mathcal{D}|$

Figure 7: Confusion Matrix for classifying 10 different levels or training environments using the RNN hidden states. Label 0 corresponds to the real **Pendulum** environment. Every row is a different training method where, DR is Domain Randomization, PLR is Prioritized Level Replay with an L_1 value loss score function and PLR_PVL refers to Prioritized Level Replay with an Positive Value Loss score function.