The Surprising Effectiveness of Latent World Models for Continual Reinforcement Learning

Anonymous Authors

Abstract

We study the use of model-based reinforcement learning methods, in particular, world models for continual reinforcement learning. In continual reinforcement learning, an agent is required to solve one task and then another sequentially while retaining performance and preventing *forgetting* on past tasks. World models offer a *task-agnostic* solution: they do not require knowledge of task changes. World models are a straight-forward baseline for continual reinforcement learning for three main reasons. Firstly, forgetting in the world model is prevented by persisting existing experience replay buffers across tasks, experience from previous tasks is replayed for learning the world model. Secondly, they are sample efficient. Thirdly and finally, they offer a task-agnostic exploration strategy through the uncertainty in the trajectories generated by the world model. We show that world models are a simple and effective continual reinforcement learning baseline. We study their effectiveness on Minigrid and Minihack continual reinforcement learning benchmarks and show that it outperforms state of the art task-agnostic continual reinforcement learning methods.

1 Introduction

There has been many recent successes in reinforcement learning (RL), such as in games [1], robotics [2] and in scientific applications [3, 4]. However these successes showcase methods for solving individual tasks. Looking beyond, it is conjectured that truly scalable intelligent systems will additionally need to master many tasks in a continual manner [5, 6, 7]. The field of continual reinforcement learning (CRL) aims to develop agents which can solve many tasks, one and then another, continually while retaining performance on all previously seen tasks [8].

This paper aims to explore the possibility of using world models to solve CRL problems. CRL is a multifaceted problem. We showcase a method that satisfies the traditional CL desiderata: avoids *catastrophic forgetting*, achieves high average performance, and is scalable. At the same time it is *task-agnostic*, namely, it does not require external task identification, neither during training nor deployment. This is highly desirable from the practical point of view, and is also an uncommon capability.

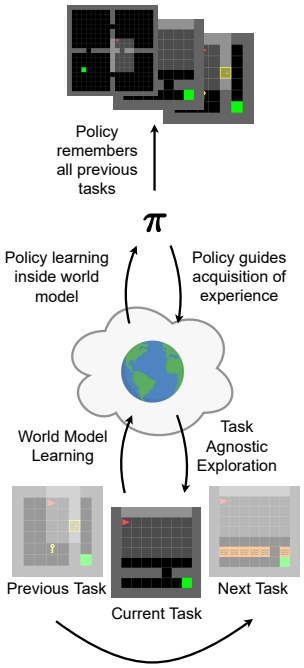

Figure 1: Overview of CRL with world models.

Preprint. Under review.

The proposed method is built on top of DreamerV2 [9]. Due to this, it inherits two important features: excellent sample efficiency due to the model-based paradigm and implicit task detection via the recurrent networks architecture. The latter enables that the method is task-agnostic [10]. The latent model allows processing image observation with a relatively low computational cost. We further propose to use replay memory [11, 12, 13] to reduce forgetting and a task agnostic exploration method [14] to enable solving sparse reward and partially-observable tasks.

Such a method is straightforward to implement and tune. We verify that it successfully solves a challenging suite of Minigrid and Minihack tasks. More generally, our work shows that the model-based paradigm is a viable solution to continual reinforcement learning.

2 Preliminaries

2.1 Reinforcement Learning

The environments we consider are partially observable; the agent does not have access to the environment state $s \in \mathcal{S}$. A Partially Observable Markov Decision Process (POMDP [15]) is the following tuple $(\mathcal{S}, \mathcal{A}, P, R, \Omega, \mathcal{O}, \gamma)$. Here $\mathcal{S}$ and $\mathcal{A}$ are the sets of states and actions respectively, such that for $s_t, s_{t+1} \in \mathcal{S}$ and $a_t \in \mathcal{A}$. $P(s_{t+1}|s_t, a_t)$ is the transition distribution and $R(a_t, s_t, s_{t+1})$ is the reward function. The discount factor is $\gamma \in (0, 1)$. Ω is the set of observations, $\mathcal{O} : \mathcal{S} \times \mathcal{A} \to P(\Omega)$ is an observation function which defines a distribution over observations. Actions a_t are chosen using a policy π that maps observations to actions: $\Omega \to \mathcal{A}$. Let us assume we have access to the states s_t and we are working with a finite horizon H. Then the return from a state is $R_t = \sum_{i=t}^{H} \gamma^{(i\text{-}t)} r(s_i, a_i)$. In RL the objective is to maximize the expected return $J = \mathbb{E}_{a_i \sim \pi}[R_1|s_0]$ given an initial state s_0.

One approach to maximizing expected return is to use a *model-free* approach to learn a policy $\pi_\phi : \mathcal{S} \to \mathcal{A}$ with a neural network parameterized by ϕ guided by an action-value function $Q_\theta(s_t, a_t)$ with parameters θ. Instead of learning a policy directly from experience we can employ *model-based* RL (MBRL) and learn an intermediate model f, for instance a transition model $s_{t+1} = f(s_t, a_t)$ from experience and learn our policy with additional experience generated from the model f [16]. Instead of working with the actual state s_t our methods consider the observations o_t and recurrent action-value functions and models to help better estimate states s_t [17].

2.2 Continual Supervised Learning

Continual Learning (CL) is a setting whereby a model must master a set of tasks sequentially while maintaining performance across all previously learned tasks. In supervised CL, the model is sequentially shown T tasks, denoted $\mathcal{T}_\tau$ for $\tau = 1, \ldots, T$. Each task, $\mathcal{T}_\tau$, is comprised of a dataset $\mathcal{D}_\tau = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_\tau}$ which a neural network is required to learn from, to perform predictions. More generally, a task is denoted by a tuple comprised of the conditional and marginal distributions, $\{p_\tau(y|\mathbf{x}), p_\tau(\mathbf{x})\}$. After task τ, the model will lose access to the training dataset for $\mathcal{T}_\tau$, however its performance will be continually evaluated on all tasks $\mathcal{T}_i$ for $i \leq \tau$. For a comprehensive review of CL scenarios see [18, 19].

2.3 Continual Reinforcement Learning

In continual RL the agent will have a budget of N interactions with each task environment $\mathcal{T}_\tau$. The agent is then required to learn a policy to maximize rewards in this environment, before interacting with a new environment and having to learn a new policy. Each task is defined as a new POMDP, $\mathcal{T}_\tau = (\mathcal{S}_\tau, \mathcal{A}_\tau, P_\tau, R_\tau, \Omega_\tau, \mathcal{O}_\tau, \gamma_\tau)$. The agent is continually evaluated on all past and present tasks and so it is desirable for the agent's policy to transfer to new tasks while not forgetting how to perform past tasks. CRL is not a new problem setting [20], however its definition has evolved over time and some settings will differ from paper to paper, we employ the setting above which is related to previous recent work in CRL [21, 22, 13, 23, 24, 10, 7].

3 Related Work

3.1 Continual Supervised Learning

One approach to CL, referred to as *regularization approaches* regularizes a NN's weights to ensure that optmizing for a new task finds a solution which is "close" to the previous task's [21, 25, 26]. Working with functions can be easier than with NN weights and so task functions can be regularized to ensure that learning new function mappings are "close" across tasks [27, 28, 29]. By contrast, *expansion approaches* add new NN components to enable learning new tasks while preserving components for specific tasks [30, 31]. *Memory approaches* replay data from previous tasks when learning the current task. This can be performed with a generative model [32]. Or samples from previous tasks (*memories*) [33, 34, 35].

3.2 Continual Reinforcement Learning

Seminal work in CRL, EWC [21] enables DQN [36] to be able to continually learn how to play different Atari games with limited forgetting. EWC learns new Q-functions by regularizing the L2 distance between the new task's optimal weights and previous task's optimal weights. EWC requires additional supervision informing it of task changes to update its objective, select specific Q-function head and select a task specific ϵ-greedy exploration schedule. Progress and Compress [22] applies a regularization to policy and value function feature extractors for an actor-critic approach. Alternatively LPG-FTW [37] learns an actor-critic that factorizes into task specific parameters and shared parameters. Both methods require task supervision and make use of task-specific parameters and shared parameters. Task agnostic methods like CLEAR [13] do not require task information to perform CRL. CLEAR leverages experience replay buffers [11] to prevent forgetting: by using an actor-critic with V-trace importance sampling [38] of past experiences from the replay buffer. Model-based RL approaches to CRL have been demonstrated where the model weights are generated from a hypernetwork which itself is conditioned a task embedding [39]. Recent work demonstrates that recurrent policies for POMDPs can obtain good overall performance on continuous control CRL benchmarks [10].

A number of previous works have studied transfer in multi-task RL settings where the goals within an environment change [40, 41, 42]. In particular by incorporating the task definition directly into the value function [41] and combining this with off-policy learning allows a CRL agent to solve multiple tasks continually, and generalize to new goals [43].

3.3 Continual Adaptation

Instead of focusing on remembering how to perform all past tasks, another line of research investigates quick adaptation to changes in environment. This can be captured by using a latent variable and off-policy RL [44]. Alternatively, one can meta-learn a model such that it can then adapt quickly on new changes in environment [45]. All these works use small environment changes such as reward function changes or changes in gravity or mass of certain agent limbs for instance. The environments which we consider contain different $\mathcal{A}$, $\mathcal{S}$ and reward functions such as opening doors with keys or avoiding lava or crossing a river which is quite different in comparison. Continual exploration strategies which use curiosity [46] can be added as an intrinsic reward in the face of non-stationary environments in infinite horizon MDPs [47]. Our proposed model uses Plan2Explore which has been shown to outperform curiosity based methods [14].

Another related area of research is open-ended learning which aims to build agents which generalized to unseen environments through a curriculum which starts off with easy tasks and then progresses to harder tasks thereby creating agents which can generalize [48, 49, 50].

4 World Models for Continual Reinforcement Learning

We leverage world models for learning tasks sequentially without forgetting. We use DreamerV2 [9] which introduces a discrete stochastic and recurrent world model that is state of the art on numerous single-GPU RL benchmarks. This is a good choice for CRL since the world model is trained by reconstructing state, action and reward trajectories from experience, we can thus leverage experience

replay buffers which persist across tasks to prevent *forgetting* in the world model. Additionally, we can train a policy in the imagination or in the generated trajectories of the world model, similar to generative experience replay methods in supervised CL which remember previous tasks by replaying generated data [32]. Thus, using a world model is also sample efficient. Also, world models are *task-agnostic* and do not require external supervision, without signaling to the agent that it is interacting with a new task. Additionally, by generating rollouts in the world model's imagination the uncertainty in the world model's predictions, more specifically the disagreement between predictions can be used as a task-agnostic exploration bonus. To summarize, we propose using using model-based RL with recurrent world models as a viable method for CRL, see Algorithm 1 for an overview, which we instantiate using DreamerV2.

4.1 Learning the World Model

DreamerV2 learns a recurrent (latent) state-space world model (RSSM) which predicts the forward dynamics of the environment. At each time step t the world model receives an observation o_t and is required to reconstruct the observations $\hat{o}_t$ conditioned on the previous actions $a_{<t}$ (in addition to reconstructing rewards and discounts). The forward dynamics are modeled using an RNN, $h_t = \text{GRU}(h_{t-1}, z_t, a_t)$ [51] where h_t is the hidden state z_t are the discrete probabilistic latent states which condition the observation predictions $p(o_t|z_t, h_t)$. Trajectories are sampled from an experience replay buffer and so persisting the replay buffer across different tasks should alleviate forgetting in the world model [13].

4.2 Policy Learning inside the World Model

The policy π is learned inside the world model by using an actor-critic [52] while freezing the weights of the RSSM world model. At each step t of the dream inside the RSSM world model a latent state z_t is sampled, z_t and the RNN hidden state condition the actor $\hat{a}_t \sim \pi(\,\cdot\,|z_t, h_t)$. The reward $\hat{r}_{t+1}$ is predicted by the world model. The policy, π is then used to obtain new trajectories in the real environment. These trajectories are added to the experience replay buffer. An initial observation o_1 is used to start generating rollouts for policy learning. This training regime ensures that the policy generalizes to previously seen environments through the world model which imagines trajectories.

4.3 Task Agnostic Exploration

The policy learns using the imaged trajectories from the RSSM world model, so world model's predicted rewards are used as a signal for the agent's policy and critic. The policy is also used to gain experience inside the real environment. So trajectories which the world model is uncertain how to predict indicate regions of the state and action space which should be prioritized by the policy when exploring the true environment. Hence, the uncertainty in the world model's trajectory prediction can be used as an additional intrinsic reward. This idea underpins Plan2Explore [14] which naturally fits with DreamerV2.

The world model quantifies the uncertainty in the next latent state prediction by using a deep ensemble; multiple neural networks with independent weights. Deep ensembles are a surprisingly robust baseline for uncertainty quantification [53] and the ensemble's variance is used as an intrinsic reward. The exploration neural networks in the ensemble are trained to predict the next RSSM latent features $[z_{t+1}, h_{t+1}]$. The world model is frozen while the ensemble is trained.

The policy π observes the reward $r = \alpha_i r_i + \alpha_e r_e$, where r_e is the extrinsic reward predicted by the world model, r_i is the intrinsic reward, the latent disagreement between the next latent state predictions. The coefficients α_i and α_e are $\in [0, 1]$. Hence the policy π can be trained inside the world model to seek regions in the state action space which the world model struggles to predict and hence when the policy is deployed in the environment it will seek these same regions in the state-action space to obtain new trajectories to train the RSSM world model. The exploration strategy is significant for CRL since it is not task dependent unlike using DQN where each task needs an ϵ-greedy schedule [21, 23] or SAC [54] which needs an entropy regularizer per task [7].

Algorithm 1 Continual Reinforcement Learning with World Models

Input: Tasks (environments) $\mathcal{T}_{1:T}$, world model M, policy π, experience replay buffer $\mathcal{D}$.
for $\mathcal{T}_1$ **to** $\mathcal{T}_T$ **do**
  Train world model M on $\mathcal{D}$.
  Train π inside world model M.
  Execute π in task $\mathcal{T}_\tau$ to gather episodes and append to $\mathcal{D}$.
end for

5 Experiments

To test the performance of DreamerV2 as a CRL method we consider a set of challenging problems. Firstly we use 3 Minigrid tasks [55]. We also consider one CRL benchmark from the CORA suite [24]: 8 Minihack tasks [56]. Code is available at `https://anonymous.4open.science/r/dv24crl-C594`. We use two primary baselines. First, Impala which is a powerful deep RL method not designed for CRL [38]. Second, we consider CLEAR [13] which uses Impala as a base RL algorithm and leverages experience replay buffers to prevent forgetting and is task agnostic.

We evaluate our methods by measuring success rates over the course of learning, Fig. 2 and Fig. 3. We also can use average performance, average forgetting and average forward transfer metrics [7] to assess the effectiveness of our proposed baseline.

Average Performance. This measures how well a CRL method performs on all tasks at the end of the task sequence. The task performance is $p_\tau(t) = [-1, 1]$ for all $\tau < T$. Since we have a reward of $+1$ for completing the task and -1 for being killed by a monster or falling into lava. If each task is seen for N environment steps and we have T tasks and the τ-th task is seen over the interval of steps $[(\tau - 1) \times N, \tau \times N]$. The average performance metric for our continual learning agent is defined as:

$$p(t_f) = \frac{1}{T} \sum_{\tau=1}^{T} p_\tau(t_f), \tag{1}$$

where $t_f = N \times T$ is the final timestep.

Forgetting. The average forgetting is the performance difference after interacting with a task versus the performance at the end of the final task. The average forgetting across all tasks is defined as:

$$F = \frac{1}{T} \sum_{\tau=1}^{T} F_\tau \quad \text{where} \quad F_\tau = p_\tau(\tau \times N) - p_\tau(t_f). \tag{2}$$

By definition the forgetting of the final T-th task is $F_T = 0$. If a CRL agent has better performance at the end of the task sequence compared to after τ-th task at time-step $\tau \times N$ then $F_\tau < 0$.

Forward Transfer. The forward transfer is the difference in task performance during continual learning compared to the single task performance. The forward transfer is defined as:

$$FT = \frac{1}{T} \sum_{\tau=1}^{T} FT_\tau \quad \text{where} \quad FT_\tau = \frac{\text{AUC}_\tau - \text{AUC}_{\text{ref}_\tau}}{1 - \text{AUC}_\tau} \tag{3}$$

$$\text{AUC}_\tau = \frac{1}{N} \int_{(\tau-1)\times N}^{\tau \times N} p_\tau(t) dt \quad \text{and} \quad \text{AUC}_{\text{ref}_\tau} = \frac{1}{N} \int_0^N p_{\text{ref}_\tau}(t) dt. \tag{4}$$

$FT_\tau > 0$ means that the CRL agent achieves better performance on task τ during continual learning versus in isolation. So this metric measures how well a CRL agent transfers knowledge from previous tasks when learning a new task.

5.1 Minigrid

Minigrid [55] is a challenging image based, partially observable and sparse reward environment. The agent, in red, will get a reward of $+1$ when it gets to the green goal, Fig. 2. The agent sees a small region of the Minigrid environment as observation o_t. We use 3 different tasks from Minigrid:

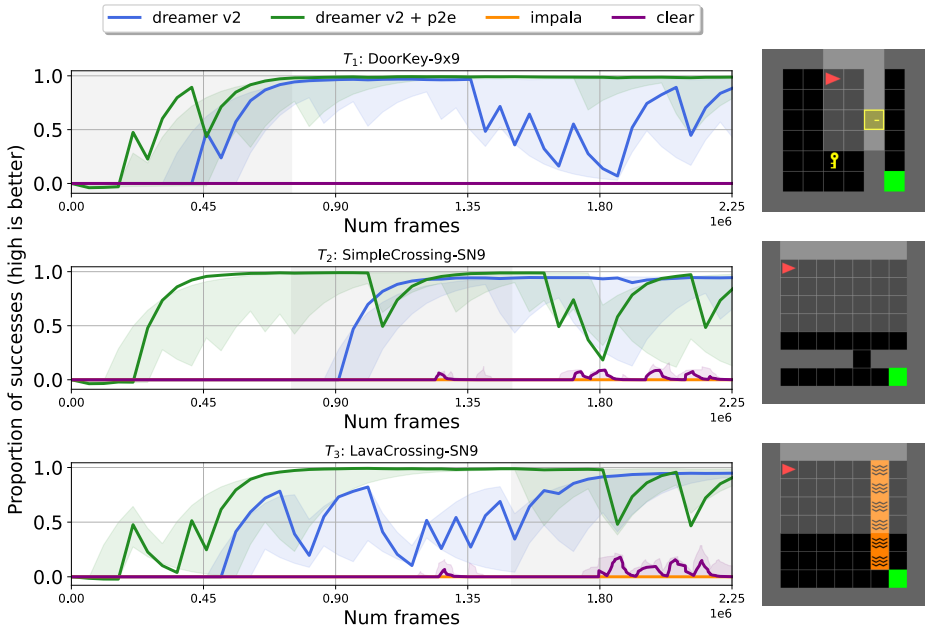

Figure 2: Performance of CRL agents on 3 Minigrid tasks. Grey shaded regions indicate the environment which the agent is currently interacting with. All learning curves are a median and inter-quartile range across 5 seeds. For Impala and CLEAR we use 10 seeds. On the right we pick a random instantiation of the Minigrid environments that are being evaluated.

`DoorKey-9x9`, `SimpleCrossing-SN9` and `LavaCrossing-9x9`. Each environment has a different obstacle and so the tasks are diverse. Each method interacts with each task for 0.75M environment interactions, as previously proposed in [23].

We continuously evaluate CRL agents on all tasks, see Fig. 2 for the success rates. The results indicate that DreamerV2 is able to solve difficult exploration tasks like the `DoorKey-9x9` which involves the agent having to pick up a key and then use the key to open a door before accessing the goal. Additionally, since DreamerV2 trains its policy inside the world model it is more sample efficient than powerful baselines like CLEAR which need $\times 10$ more environment interactions to be able to solve the easier Minigrid tasks `SimpleCrossing-SN9` and `LavaCrossing-9x9`, Table 1. The addition of Plan2Explore enables DreamerV2 to solve these environments even more quickly, see Fig. 2. DreamerV2 does exhibit some forgetting of the `DoorKey-9x9` task and this indicates that additional mechanisms to prevent forgetting might be needed.

From the metrics in Table 1 we can see that DreamerV2 has strong forward transfer. From the learning curves for individual tasks Fig. 4 we can see that DreamerV2 struggles on independent task learning over the course of 0.75M environment steps. In contrast, when learning continually DreamerV2 is able to solve all tasks indicating that it transfer knowledge from previous tasks. This is not entirely surprising since the levels look similar and so the world model will already be able to reconstruct certain observations from one task to the next.

For DreamerV2 we use the model and hyperparameters from [9] with an experience replay buffer for world model learning of size 2M. For DreamerV2 + Plan2Explore we set the reward coefficients to $\alpha_i = \alpha_e = 0.9$ which was found by grid search of various single task Minihack environments over $\alpha_i = \alpha_i = \{0.1, 0.5, 0.9\}$ we use the same policy for exploration and evaluation and learn world model by observation reconstruction only, rather than observation, reward and discount reconstruction. We explore these design decisions using the Minihack benchmark in Appendix B.1. For CLEAR we use an experience replay buffer size of 1M.

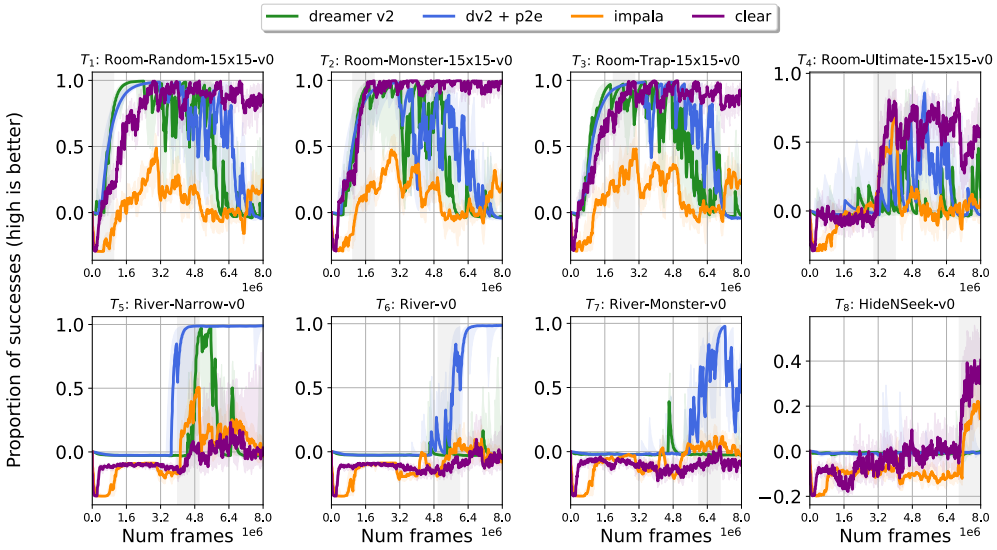

Figure 3: Performance of various CRL agents on 8 Minihack tasks. Grey shaded regions indicate the environment which the agent is currently interacting with. All learning curves are a median and inter-quartile range across 5 seeds, for Impala and CLEAR we use 10 seeds.

	Avg. Performance (↑)	Avg. Forgetting (↓)	Avg. Forward Transfer (↑)
Impala	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
CLEAR	0.03 ± 0.04	0.02 ± 0.02	0.01 ± 0.01
Impala$\times$10	0.16 ± 0.16	0.06 ± 0.13	-
CLEAR$\times$10	0.64 ± 0.20	0.00 ± 0.00	-
DreamerV2	0.75 ± 0.11	0.01 ± 0.14	0.45 ± 0.63
DreamerV2 + P2E + $\pi_{exp} = \pi_{eval}$	0.74 ± 0.02	-0.02 ± 0.03	1.06 ± 0.83

Table 1: Results on 3 Minigrid tasks. All metrics are an average and standard errors over 5 seeds. For CLEAR and Impala we use 10 seeds. We use 0.75M interactions for each task, and 7.5M in methods marked with $\times 10$. ↑ indicates better performance with higher numbers, and ↓ the opposite.

5.2 Minihack

To test the limits of DreamerV2's ability to remember previous tasks and its exploration mechanism we look at a longer set of tasks which are harder to solve in the from of Minihack [56]. The effect of using an additional exploration strategy is apparent as DreamerV2 + Plan2Explore is able to solve harder tasks than the baselines. Additionally DreamerV2 + Plan2Explore is also able to achieve large forward transfer similarly with the Minigrid experiments.

Minihack is a set of diverse image based, sparse reward tasks based on the game of Nethack [57]. We test DreamerV2 performance on 8 tasks from Minihack [56]. In particular we consider the following tasks `Room-Random-15x15-v0`, `Room-Monster-15x15-v0`, `Room-Trap-15x15-v0`, `Room-Ultimate-15x15-v0` `River-Narrow-v0`, `River-v0`, `River-Monster-v0` and `HideNSeek-v0`, which are a subset of the 12 Minihack tasks from the CORA CRL benchmark [24]. Each task is seen once and has a budget of 1M environment interactions.

DreamerV2 is able to solve the easier first three Room environments `Room-Random-15x15-v0` `Room-Monster-15x15-v0` `Room-Trap-15x15-v0` however struggles to solve the harder later tasks and is susceptible to forgetting. In contrast DreamerV2 + Plan2Explore is able to solve the harder `River-Narrow-v0`, `River-v0` and `River-Monster-v0` tasks and is also susceptible to forgetting of the initial Room tasks. CLEAR in contrast is really stable to remembers how to solve the Room tasks with little forgetting, however is unable to solve the more difficult River tasks. For DreamerV2

	Avg. Performance (↑)	Avg. Forgetting (↓)	Avg. Forward Transfer (↑)
Impala	0.14 ± 0.05	0.14 ± 0.04	0.22 ± 0.09
CLEAR	0.51 ± 0.07	-0.05 ± 0.05	1.05 ± 0.09
DreamerV2	0.09 ± 0.07	0.37 ± 0.07	0.56 ± 0.86
DreamerV2 + Plan2Explore	0.38 ± 0.03	0.22 ± 0.05	0.76 ± 0.25

Table 2: Results on 8 Minihack tasks. All metrics are an average and standard error over 5 seeds. For CLEAR and Impala we use 10 seeds. ↑ indicates better performance with higher numbers, and ↓ the opposite.

and DreamerV2 + Plan2Explore we use the same design choices as described for the Minigrid experiments in Section 5.1. For CLEAR we use a replay buffer size of 1M transitions only. The entire task sequence is 8M steps so CLEAR is effective in preventing forgetting with a relatively small experience replay buffer.

From Table 2 we see that CLEAR has good performance, this can be explained from the learning curves in Fig. 3 where only the simplest Minihack Room tasks are solved and remembered. CLEAR like DreamerV2 + Plan2Explore is also similarly able to effectively transfer knowledge from other tasks when learning a new task with impressive forward transfer. On the other hand DreamerV2 and DreamerV2 + Plan2Explore achieve worse average performance since it is more susceptible to forgetting however they still have high forward transfer. An instance of forward transfer can be seen by looking at the single task performance on `Room-Trap-15x15-v0` `Room-Monster-15x15-v0` and `River-Monster-v0` in Fig. 5 which isn't as high as in the CRL setting. One simple design choice we can make to alleviate forgetting further would simply be to increase the size of the experience replay buffer, this decreases forgetting to 0.01 ± 0.11 and increases the average performance to 0.48 ± 0.09 for a replay buffer size of 8M, but at the same time this decreases forward transfer Fig. 6.

6 Discussion and Future Works

We have explored the use of world models as a CRL baseline. World models can be powerful CRL agents as they train the policy inside the world model and can thus be sample efficient. World models are trained by using experience replay buffers and so we can prevent forgetting of past tasks by persisting the replay buffer from the current task to a new task and so on. Importantly, the world model's prediction uncertainty can be used as an additional intrinsic task agnostic reward to help exploration and solve difficult tasks in a task-agnostic fashion [14]. Previous CRL exploration strategies in the literature all require task information to be effective. We use DreamerV2 as the world model [9] and we show that DreamerV2 is a powerful CRL method on two different difficult CRL benchmarks.

We show that world models can be a strong baseline for CRL problems compared to state of the art methods such as CLEAR [13]. DreamerV2 with Plan2Explore outperforms CLEAR on Minigrid. It also can achieve comparable performance with CLEAR on Minihack. On the one hand it exhibits more forgetting than CLEAR, on the other hand it can solve harder and more difficult exploration tasks than CLEAR. Future work, will explore making world model less susceptible to forgetting.

References

[1] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *CoRR*, abs/1911.08265, 2019.

[2] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub W. Pachocki, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *CoRR*, abs/1808.00177, 2018.

[3] V Nguyen, SB Orbell, Dominic T Lennon, Hyungil Moon, Florian Vigneau, Leon C Camenzind, Liuqi Yu, Dominik M Zumbühl, G Andrew D Briggs, Michael A Osborne, et al. Deep

reinforcement learning for efficient measurement of quantum devices. *npj Quantum Information*, 7(1):1–9, 2021.

[4] Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

[5] Mark Bishop Ring. *Continual learning in reinforcement environments*. PhD thesis, University of Texas at Austin, 1994.

[6] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245 – 258, 2017.

[7] Maciej Wolczyk, Michal Zajac, Razvan Pascanu, Lukasz Kucinski, and Piotr Milos. Continual world: A robotic benchmark for continual reinforcement learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 28496–28510, 2021.

[8] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *arXiv preprint arXiv:2012.13490*, 2020.

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

[10] Massimo Caccia, Jonas Mueller, Taesup Kim, Laurent Charlin, and Rasool Fakoor. Task-agnostic continual reinforcement learning: In praise of a simple baseline. *arXiv preprint arXiv:2205.14495*, 2022.

[11] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. Learn.*, 8(3–4):293–321, May 1992.

[12] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[13] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems 32*, pages 350–360. 2019.

[14] 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.

[15] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[16] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

[17] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *2015 aaai fall symposium series*, 2015.

[18] Yen-Chang Hsu, Yen-Cheng Liu, Anita Ramasamy, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

[19] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[20] Sebastian Thrun and Tom M. Mitchell. Lifelong robot learning. *Robotics and Autonomous Systems*, 15(1):25–46, 1995. The Biology and Technology of Intelligent Autonomous Agents.

[21] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016.

[22] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework

for continual learning. In Jennifer G. Dy and Andreas Krause, editors, *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 4535–4544. PMLR, 2018.

[23] Samuel Kessler, Jack Parker-Holder, Philip Ball, Stefan Zohren, and Stephen J Roberts. Same state, different task: Continual reinforcement learning without interference. *arXiv preprint arXiv:2106.02940*, 2021.

[24] Sam Powers, Eliot Xing, Eric Kolve, Roozbeh Mottaghi, and Abhinav Gupta. Cora: Benchmarks, baselines, and metrics as a platform for continual reinforcement learning agents. *arXiv preprint arXiv:2110.10067*, 2021.

[25] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[26] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual Learning Through Synaptic Intelligence. In *International Conference on Machine Learning*, 2017.

[27] Zhizhong Li and Derek Hoiem. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[28] Ari S Benjamin, David Rolnick, and Konrad P Kording. Measuring and Regularizing Networks in Function Space. In *International Conference on Learning Representations*, 2019.

[29] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020.

[30] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

[31] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. *arXiv preprint arXiv:2001.00689*, 2020.

[32] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual Learning with Deep Generative Replay. In *Advances in Neural Information Processing Systems*, 2017.

[33] David Lopez-Paz and Marc ' Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. In *Advances in Neural Information Processing Systems*, 2017.

[34] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, 2019.

[35] Arslan Chaudhry, Marcus Rohrbach Facebook, A I Research, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip H S Torr, and Marc ' Aurelio Ranzato. On Tiny Episodic Memories in Continual Learning. *arxiv.org:1902.10486*, 2019.

[36] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

[37] Jorge A Mendez, Boyu Wang, and Eric Eaton. Lifelong Policy Gradient Learning of Factored Policies for Faster Training Without Forgetting. In *Advances in Neural Information Processing Systems*, 2020.

[38] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning*. 2018.

[39] Yizhou Huang, Kevin Xie, Homanga Bharadhwaj, and Florian Shkurti. Continual model-based reinforcement learning with hypernetworks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 799–805. IEEE, 2021.

[40] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[41] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

[42] André Barreto, Diana Borsa, Shaobo Hou, Gheorghe Comanici, Eser Aygün, Philippe Hamel, Daniel K Toyama, Jonathan J Hunt, Shibl Mourad, David Silver, et al. The option keyboard: Combining skills in reinforcement learning. 2019.

[43] Daniel J Mankowitz, Augustin Žídek, André Barreto, Dan Horgan, Matteo Hessel, John Quan, Junhyuk Oh, Hado van Hasselt, David Silver, and Tom Schaul. Unicorn: Continual learning with a universal, off-policy agent. *arXiv preprint arXiv:1802.08294*, 2018.

[44] Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst lifelong non-stationarity. *arXiv preprint arXiv:2006.10701*, 2020.

[45] Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

[46] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[47] Christian Steinparz, Thomas Schmied, Fabian Paischer, Marius-Constantin Dinu, Vihang Patil, Angela Bitto-Nemling, Hamid Eghbal-zadeh, and Sepp Hochreiter. Reactive exploration to cope with non-stationarity in lifelong reinforcement learning. *arXiv preprint arXiv:2207.05742*, 2022.

[48] Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O Stanley. Paired open-ended trailblazer (poet): Endlessly generating increasingly complex and diverse learning environments and their solutions. *arXiv preprint arXiv:1901.01753*, 2019.

[49] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michael Mathieu, et al. Open-ended learning leads to generally capable agents. *arXiv preprint arXiv:2107.12808*, 2021.

[50] Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. *arXiv preprint arXiv:2203.01302*, 2022.

[51] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[52] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[53] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[54] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

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

[56] Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Kuttler, Edward Grefenstette, and Tim Rocktäschel. Minihack the planet: A sandbox for open-ended reinforcement learning research. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[57] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The NetHack Learning Environment. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[58] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013.