REACTIVE EXPLORATION TO COPE WITH NON-STATIONARITY IN LIFELONG REINFORCEMENT LEARNING

Christian Steinparz [‡] [*] **Thomas Schmied** [†] **Fabian Paischer** [†] **Marius-Constantin Dinu** [†] [‖]
Vihang Patil [†] **Angela Bitto-Nemling** [†] [§] **Hamid Eghbal-zadeh** [†] **Sepp Hochreiter** [†] [§]

[†] ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning, Johannes Kepler University Linz, Austria
[‡] Visual Data Science Lab, Institute of Compute Graphics, Johannes Kepler University Linz, Austria
[‖] Dynatrace Research, Linz, Austria
[§] Institute of Advanced Research in Artificial Intelligence (IARAI), Vienna, Austria

ABSTRACT

In lifelong learning, an agent learns throughout its entire life without resets, in a constantly changing environment, as we humans do. Consequently, lifelong learning comes with a plethora of research problems such as continual domain shifts, which result in non-stationary rewards and environment dynamics. These non-stationarities are difficult to detect and cope with due to their continuous nature. Therefore, exploration strategies and learning methods are required that are capable of tracking the steady domain shifts, and adapting to them. We propose *Reactive Exploration* to track and react to continual domain shifts in lifelong reinforcement learning, and to update the policy correspondingly. To this end, we conduct experiments in order to investigate different exploration strategies. We empirically show that representatives of the policy-gradient family are better suited for lifelong learning, as they adapt more quickly to distribution shifts than Q-learning. Thereby, policy-gradient methods profit the most from Reactive Exploration and show good results in lifelong learning with continual domain shifts. Our code is available at: https://github.com/ml-jku/reactive-exploration.

1 INTRODUCTION

Existing Reinforcement Learning (RL) algorithms are tailored towards the limited setup of stationary environments. That is, considering a task in an episodic manner, with an assumption on stationary components of the underlying Markov Decision Process (MDP). This scenario does not reflect how we humans learn in the real world, which steadily evolves over time. Learning in non-stationary environments requires algorithms that are inherently capable of constantly dealing with domain shifts (Widmer & Kubát, 2004; Adler et al., 2020). In the literature, several topics have been dedicated to study this problem. For example, the paradigm of domain adaptation (Ben-David et al., 2010) comprises a single domain shift from a source to a target domain. In the field of lifelong learning (or continual learning) (Thrun & Mitchell, 1995; Thrun, 1998), however, these domain shifts occur in a distinct continual fashion (Ring et al., 1994). While we humans easily adapt to such continual domain shifts in the real world, existing algorithms are often designed towards specific problem settings, rendering them incapable of handling these continual shifts. In the context of lifelong learning, the horizon is unbounded, and agents must learn from potentially years of experiences under continual domain shifts. The knowledge acquired by agents can then be used to facilitate transfer of the learned skills to various tasks to mitigate performance plateaus (Mitchell et al., 2018). An example of such problems are robots that constantly explore and ideally learn from previous experiences (Wołczyk et al., 2021).

Many potential sources of non-stationarity exist in lifelong RL. One way to induce non-stationarity is to change various components of the MDP over time, i.e. the reward function, or transition probabilities. Non-stationarity can also be introduced via altering observations or the set of actions over time (Khetarpal et al., 2020). These changes can occur gradually and over multiple steps (Besbes et al., 2014), or abruptly. The latter is often referred to as piecewise stationary (Hartland et al., 2007; Garivier & Moulines, 2008; Cao et al., 2019). Approaches that adapt to non-stationary environments have to cope with effects known as catastrophic forgetting or catastrophic interference, which is also referenced as the sensitivity-stability dilemma or stability-plasticity dilemma (Hebb, 1949; Carpenter & Grossberg, 1987; McCloskey & Cohen, 1989; Mermillod et al., 2013; Ehret et al., 2020). A policy learned under

[*] Correspondence to: christian.steinparz@jku.at

such conditions needs to retain valuable information from the past while still being able to quickly adapt to unseen situations. Recent works in the context of lifelong RL also address this as forward transfer and backward transfer of knowledge (Wang et al., 2019). Gaining new knowledge on current tasks may positively impact past beliefs of already learned tasks (backward transfer), or facilitate adaptation to future tasks (forward transfer).

In environments with domain shifts, it is difficult for a learning agent to keep track and react to changes. The detection of such changes in continuous streams of data is often referred to as changepoint detection (Kifer et al., 2004; Adams & MacKay, 2007). Dealing with dynamic changepoints is crucial in coping with non-stationarity. In this work, we propose to leverage Reactive Exploration to efficiently detect dynamic changepoints and facilitate recovery and forward transfer in lifelong learning. Reactive Exploration predicts the expected future outcome using an observations model and a reward model, which enable detecting changes in the environment. If the current input deviates strongly from the intrinsic belief of the agent, it is encouraged to explore, in order to quickly adapt to changes in the environment. Thus, Reactive Exploration assigns credit to exploring parts of the environment that diverge from its internal belief. We demonstrate that Reactive Exploration enables agents to deal with various types of non-stationarities.

In order to evaluate RL agents under these conditions, we systematically induce changes to the environment dynamics, observations, and the reward function. Particularly, we compare Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Deep-Q Networks (DQN) (Mnih et al., 2013), as the most popular representatives of on-policy and off-policy algorithms. First, we investigate the robustness of selected algorithms against steadily and abruptly changing reward functions. We find that while PPO is able to recover in the face of gradually shifting rewards, DQN completely fails. Even for abrupt changes in the reward function, we observe that PPO is more successful at recovery. A thorough analysis of the different modules of DQN suggests that its shortcomings can be mostly attributed to (i) its lack of systematic exploration, and (ii) the replay buffer management. We find that injecting prior knowledge with respect to environmental changepoints into the exploration schedule and replay buffer allocation drastically improves stability of DQN. However, such information is usually not available a-priori. This renders DQN unsuitable for non-stationary problems. Further, we induce non-stationarity in environment observations via different transformations, and find that even PPO completely fails in such a setup. Equipping the agent with Reactive Exploration significantly improves stability and robustness of PPO against the different forms of non-stationarity. In summary, our in-depth analysis provides new insights on the problem of lifelong RL, which result in the following contributions:

1. PPO is more robust to domain shifts over time, compared to DQN.
2. DQN is not suitable for non-stationarity when no additional information is given on occurring shifts.
3. For severe domain shifts, both DQN and PPO fail to adapt.
4. Reactive Exploration helps PPO to recover, even for severe shifts.

2 Related Work

Learning several tasks sequentially is necessary to tackle lifelong learning problems. Agents need to retain already gained knowledge, but also remain pliable to acquire new knowledge. Ideally, this is achieved by leveraging past experiences to effectively build up knowledge representations that are useful across multiple tasks, allowing for forward and backward transfer of knowledge. Early neural network architectures were shown to be unsuitable for continual domain shifts, and incapable of coping with effects of catastrophic forgetting or the stability-plasticity dilemma (French, 1999). The field of lifelong learning has emerged to study general approaches to alleviate these issues.

Some approaches in lifelong learning address continual domain shifts via rehearsal or pseudo-rehearsal methods (Parisi et al., 2018; Hayes et al., 2019), the reduction of representational overlap (Serra et al., 2018; Wortsman et al., 2020) or architectural strategies (Rusu et al., 2016; Fernando et al., 2017; Mallya & Lazebnik, 2018; Xu & Zhu, 2018; Maltoni & Lomonaco, 2019), regularization strategies for consolidating parameters (Kirkpatrick et al., 2017; Zenke et al., 2017), and correcting for domain shifts (Li & Hoiem, 2017; Lee et al., 2017; Zeno et al., 2021). Other approaches consider meta-learning (Schmidhuber, 1987; Bengio et al., 1992), which uses task-specific inner loops and task agnostic outer loops to learn supportive representations that are helpful across tasks (Javed & White, 2019; Gupta et al., 2020). Lifelong learning can also be addressed via multi-task learning (Caruana, 1997), by learning joint embedding spaces over multiple tasks at the same time (Mitchell et al., 2018).

In the context of non-stationarity and lifelong RL specifically, continual domain shifts may occur in state or observations spaces, action spaces, transition dynamics, reward functions, or combinations thereof (Cheung et al., 2020; Igl et al., 2020; Khetarpal et al., 2020). Commonly, non-stationarity is encountered in transition dynamics or reward functions. To cope with such non-stationarities an agent must be equipped with a mechanism that is capable of detecting environmental changes to facilitate its adaptation to a new domain. Padakandla et al. (2019) combine Q-learning

(Watkins & Dayan, 1992) with a changepoint detection algorithm (Prabuchandran et al., 2022) to handle changing environment dynamics. Canonaco et al. (2020) combine policy gradient algorithms with the CUSUM (Basseville & Nikiforov, 1996) approach for changepoint detection. (Alegre et al., 2021) utilizes an ensemble of learned dynamics models to compute change statistics and deploy different policies for different contexts. Instead of detecting changepoints, Flennerhag et al. (2022) meta-learn an exploration schedule to cope with a non-stationary reward function. In contrast, Reactive Exploration is a mechanism that allows for both, detecting changepoints while simultaneously assigning credit to exploration of novel states that have undergone change.

In other settings, such as multi-armed bandit (MAB) problems, Raj & Kalyani (2017) take a Bayesian approach and investigate Thompson sampling and its relation to non-stationarity, Wu et al. (2018) propose a contextual bandit (Lattimore & Szepesvári, 2020) algorithm to detect changes based on its reward estimation confidence, and update the bandit arm selection strategy respectively. Besbes et al. (2014) characterize the regret in MAB problems with non-stationary rewards such that there is a link between reward variation and minimal achievable regret to factor in a variation budget to allow for temporal changes. Zhao et al. (2020) propose a method to overcome non-stationarity by investigating a periodically restarted Upper-Confidence-Bound algorithm, while Russac et al. (2019) propose an algorithm based on discounted linear regression. Hong et al. (2021) focus on abrupt changes in contextual bandits, and develop an off-policy method to learn sets of policies corresponding to a categorical latent state for individual phases of non-stationarity. Besbes et al. (2019) introduce a non-parametric statistical model to track changes in the reward function over time. However, MAB settings do not consider state-transition dynamics.

Additional problems in lifelong RL include credit assignment (Sutton, 1984; Arjona-Medina et al., 2019; Patil et al., 2020; Holzleitner et al., 2020; Widrich et al., 2021; Dinu et al., 2022), abstraction (Shanahan & Mitchell, 2022; Paischer et al., 2022), and exploration (Sutton & Barto, 1998). Several works address the exploration problem by novelty-based (Burda et al., 2019), count-based, (Strehl & Littman, 2008; Bellemare et al., 2016; Ostrovski et al., 2017; Martin et al., 2017; Tang et al., 2017; Machado et al., 2020), curiosity-based (Schmidhuber, 1991; 1990), and model-based (Stadie et al., 2015; Pathak et al., 2017; Achiam & Sastry, 2017) approaches. Additionally, Oudeyer & Kaplan (2007) use intrinsic motivation, and Raileanu & Rocktäschel (2020) include novelty measures for exploration in procedurally generated environments. In the context of hierarchical RL, McGovern & Barto (2001); Goel & Huber (2003) tackle exploration for temporally extended actions. Garcia & Thomas (2019) address the problem of optimizing the exploration strategy in a meta-MDP. Other strategies include memory-based exploration strategies such as episodic memory (Badia et al., 2020) or planning with worst-base planning evolution (Lecarpentier & Rachelson, 2019), Go-Explore (Ecoffet et al., 2019), and optimal experiment design (Cohn, 1994; Storck et al., 1995). Nonetheless, the exploration/exploitation trade-off has been studied little in the context of non-stationary environments. Current state-of-the-art exploration mechanisms such as RIDE (Raileanu & Rocktäschel, 2020) or NovelD (Zhang et al., 2021) rely on episodic counts and are thus not applicable to the infinite horizon setting out-of-the-box. Methods such as Intrinsic Curiosity Module (ICM) (Pathak et al., 2017) and Random Network Distillation (RND) (Burda et al., 2019) do not have this limitation.

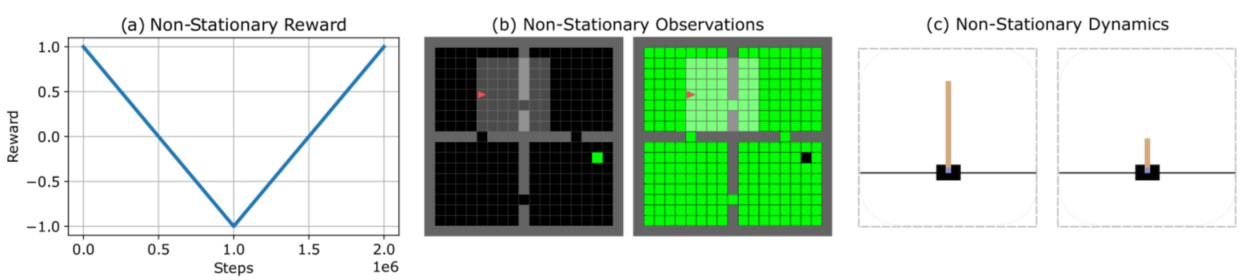


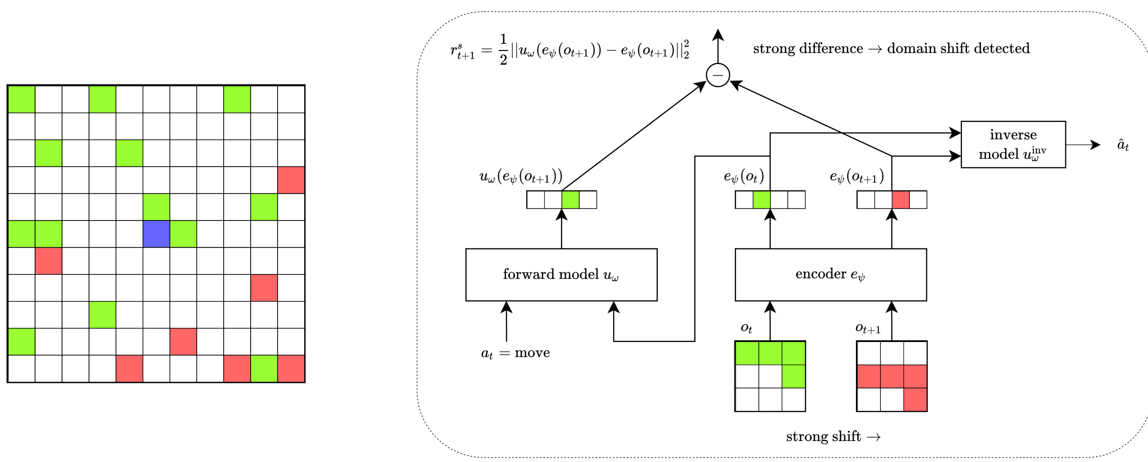
Figure 1: Different types of Non-Stationarity. (a) changing reward function over time; (b) shift in the observation distribution of a gridworld (Chevalier-Boisvert et al., 2018); (c) distribution shift in the dynamics of the CartPole balancing system (Brockman et al., 2016), altering the length of the pole results in different dynamics.

3 Problem Statement

One of the big problems in lifelong RL is the capability to adapt to new distributions. Agents that are not equipped with tools to deal with domain shifts drastically fail to solve a given task in lifelong RL. The problem of dealing with domain shifts can occur in different aspects, and in varying levels of severity and intervals. Some of the most important ones which are at the core of lifelong RL are: (a) changes in the rewards, (b) changes in the observations received from the environment, and (c) changes in the environment dynamics. Each of these introduce a new kind of challenge to the agent. In (a) for example, an agent must discover the new goal to be accomplished and re-evaluate its behavior accordingly, as the reward distribution shifts (Figure 1, left). In (b) the agent must generalize to the observation distribution that presents a new representation of the world (Figure 1, middle), while in (c) the agent must approximate changes occurring in the underlying physics (Figure 1, right). All these options are unique in nature and present a set of challenges that might differ from one another.

We as humans deal with similar situations on a daily basis, and successfully adapt our strategies to the evolving world around us. When facing any of the aforementioned problems, we first perceive the occurrence of change, and then accustom to those changes as we realize the previous strategy no longer works. In other words, we detect the changes, and react accordingly. A necessary component facilitating this capability is our internal model of the world, which stores all necessary information to reliably predict future outcomes (Forrester, 1971). This internal model allows us to act upon internal predictions of how the world will behave in the future (Maus et al., 2013). In turn, we are inherently capable of detecting changepoints when our perception of the world strongly deviates from our internal expectation of how the world will behave in the future. Similarly, we attempt to bridge the gap for changepoint detection and adaptation in the realm of lifelong RL. In this regard, we define Reactive Exploration as a mechanism that intrinsically motivates the agent to explore after a changepoint has been detected. Reactive Exploration equips an agent with the means to detect changes in the distribution of dynamics, observations, or rewards. Furthermore, it forces the agent to explore regions of the state space that have undergone change, immediately after they experienced change.

In the following, we consider the setting of a partially observable Markov Decision Process (POMDP, Kaelbling et al., 1998), which is defined as a 7-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is a set of actions, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{P}(\mathcal{S})$ is the transition function which takes a state $s \in \mathcal{S}$ and an action $a \in \mathcal{A}$ and maps it to a probability distribution over states $\mathcal{S}$. Further, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function which takes a state and an action and maps it to a scalar reward, Ω is the set of observations, $\mathcal{O} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{P}(\Omega)$ is the observation function that yields a distribution over observations given a state-action pair, and $\gamma \in [0, 1)$ is the discount factor. The policy $\pi_\theta : \Omega \mapsto \mathcal{A}$ is parameterized by θ and maps an observation $\boldsymbol{o} \in \mathcal{O}$ to an action $a \in \mathcal{A}$. Moreover, we consider the infinite horizon case, in which the episode horizon is unbounded. In Section 4, we present our Reactive Exploration method, in order to cope with non-stationarities and continual domain shifts in lifelong RL. We show empirical results in Section 5 which illustrate that state-of-the-art RL algorithms struggle to solve tasks under different non-stationarities, and how Reactive Exploration can enable the agents to cope with these issues.



(a) (b)

Figure 2: (a) Layout of the JellyBean-Base environment: image observations are agent-centered, walking over items adds them to the agents' inventory and yields reward. (b) Intrinsic Curiosity Module as one specific instance of Reactive Exploration detecting a distribution shift in the observations as colors are changing. We discuss and apply alternative instantiations in Appendix A.6.

4 Reactive Exploration as Remedy for Non-Stationarity

Exploration strategies can help to track non-stationarities and to cope with their drawbacks. After encountering a changepoint, what an agent has come to believe about certain aspects of the environment may become incorrect. We are interested in the exact point of change and the nature of change. If these were known, the agent could simply resample transitions of the affected parts of the environment. As a result, the agent could then correct its policy accordingly. However, this is usually not the case since agents do not receive any information about changes in the environment. By equipping an agent with an exploration mechanism, it can be encouraged to re-explore parts of the environment that changed over time. Reactive Exploration is a mechanism that enables the agent to react to the novelties observed in the environment after a domain shift occurred. More specifically, in Reactive Exploration, a mechanism is used to detect changes in the environment and controls the amount of exploration directed towards changed regions, via an intrinsic reward signal. We illustrate the effectiveness of Reactive Exploration and its capability to track changes in the environment and encourage exploration towards changed regions.

Our Reactive Exploration is based on the Intrinsic Curiosity Module (Pathak et al., 2017), which consists of a forward and an inverse dynamics model. Both models share an encoder $e_\psi : \Omega \mapsto \mathbb{R}^d$ that maps an observation $o_t \in \Omega$ to a vector of dimension d. The forward model $u_\omega : \mathbb{R}^d \times \mathcal{A} \mapsto \mathbb{R}^d$ maps the encoded observation of the current timestep $e_\psi(\boldsymbol{o}_t) \in \mathbb{R}^d$ and an action $a_t \in \mathcal{A}$ to the encoded next timestep $e_\psi(\boldsymbol{o}_{t+1}) \in \mathbb{R}^d$. The inverse model $u_\omega^{inv} : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{P}(\mathcal{A})$ takes as input the encoded current observation $e_\psi(\boldsymbol{o}_t) \in \mathbb{R}^d$ and the encoded next observation $e_\psi(\boldsymbol{o}_{t+1}) \in \mathbb{R}^d$ and predicts a probability distribution over actions $\mathbb{P}(\mathcal{A})$. Furthermore, we add a reward model $g_\nu : \Omega \times \mathcal{A} \mapsto \mathbb{R}$ that approximates the extrinsic reward r^e_{t+1}, given the current observation $\boldsymbol{o}_t \in \Omega$ and action $a_t \in \mathcal{A}$. The training of the dynamics models is decoupled from training the policy π_θ and can happen on different timescales. In the off-policy case, the entire replay buffer is used for training the dynamics models which contains data collected by the behavioral policy π_β while in the on-policy case the data stems from the current policy π_θ.

The intrinsic reward is defined as the prediction error of the observation prediction model and the reward model. Since these error terms appear in different scales and are in principle unbounded we introduce three weighting factors $\alpha \in [0, 1]$, $\beta \in [0, 1]$, and $\lambda \in [0, 1]$ with the constraint $\lambda = 1 - \alpha - \beta$ to properly combine the extrinsic with the intrinsic rewards. In principle, any novelty-based exploration mechanism may be used. Depending on the task at hand different components may be neglected, i.e. when dealing with shifting dynamics, the reward model does not provide additional useful information. Algorithm 1 outlines the policy update with added Reactive Exploration.

Algorithm 1 Policy Update with Reactive Exploration

Require: Weighting factors $\alpha \in [0, 1]$, $\beta \in [0, 1]$ and $\lambda \in [0, 1]$, policy model π_θ, observation model u_ω, reward model g_ν, parametrized by θ, ω, ν respectively, Buffer $\mathcal{B}$, and environment $\mathcal{E}$
Ensure: $\lambda = 1 - \alpha - \beta$

```
Initialize θ, ω, ν, B ← ∅                                          ▷ Initialize parameters and buffer
repeat
    for k iterations do
        (o_t, a_t, r^e_{t+1}, o_{t+1}) ~ E with action a_t ~ π_θ(· | o_t)   ▷ Sample transition from environment
        B ← B ∪ {o_t, a_t, o_{t+1}, r^e_{t+1}}                          ▷ Add sample to buffer
    end for
    for j update steps do
        (o_t, a_t, o_{t+1}, r^e_{t+1}) ~ B                              ▷ Sample transitions from buffer
        r^s_{t+1} = u_ω(o_t, a_t, o_{t+1})                ▷ Compute intrinsic reward with observation model
        r̂^e_{t+1} = g_ν(o_t, a_t)                          ▷ Predict extrinsic reward with reward model
        r^r_{t+1} = 1/2(r̂^e_{t+1} − r^e_{t+1})^2            ▷ Compute intrinsic reward of the reward model
        r_{t+1} = α r^s_{t+1} + β r^r_{t+1} + λ r^e_{t+1}                  ▷ Compute combined reward
        θ ← UpdatePolicy(π_θ, o_t, a_t, o_{t+1}, r_{t+1})                          ▷ Update policy
        ν ← UpdateRewardModel(g_ν, o_t, a_t, r^e_{t+1})                      ▷ Update reward model
        ω ← UpdateObservationModel(u_ω, o_t, a_t, o_{t+1})              ▷ Update observation model
    end for
until done
```

Algorithm 1: Reactive exploration for detecting non-stationarities and adjust to them. The observation model u_ω can be instantiated with any novelty-based exploration module. The reward model g_ν may be neglected if no shift in the reward function occurs. In the case of on-policy RL the buffer $\mathcal{B}$ is instantiated as rollout buffer, while in the off-policy case it is instantiated as replay buffer (Lin, 1992).

We choose this particular setup as Reactive Exploration mechanism since it provides several benefits when dealing with non-stationarities. First, the forward dynamics model learns to predict the next observation and its prediction error reflects when changes in the environment dynamics occur. After such a shift, it encourages the agent to explore regions of the observation space that yield a high error. Second, the inverse dynamics model partially alleviates the socalled "noisy-TV" problem (Burda et al., 2019) apparent in stochastic environments, which refers to an endless stream of exploration-based rewards being generated by randomly changing aspects of an environment. The inverse dynamics model forces the shared encoder to neglect features that are created randomly and are therefore unpredictable. Thereby, the prediction of the forward dynamics model becomes invariant to random effects in the environment. Moreover, our setup does not rely on episodic information to compute the intrinsic rewards, as compared to other state-of-the-art exploration algorithms (Raileanu & Rocktäschel, 2020; Zhang et al., 2021). Finally, the reward model allows detecting changes in the reward function.

5 Empirical Analysis of Reactive Exploration

We provide compelling evidence which supports the use of Reactive Exploration to cope with domain shifts. In Section 5.1 we elaborate on the environment used for our setup which allows introducing various non-stationarities as discussed in Section 3 in a controlled manner. We train two well-established algorithms (PPO and DQN) in our environment and present empirical evidence in three parts: First, in Section 5.2, we show which types of non-stationarity are especially difficult for agents to cope with by altering the reward function. We find that PPO consistently outperforms DQN when confronted with various domain shifts. Our analysis of the shortcomings of DQN identifies two major reasons for its plummeting performance (Section 5.3). On one hand, the replay buffer management is critical to avoid interference of transitions collected under different non-stationarities. On the other hand, a more informed exploration strategy proves to be extremely useful in the face of non-stationarity. In Section 5.4 we apply transformations to enforce non-stationary observations which are severe enough to break the PPO agents. Following this, in Section 5.5, we add Reactive Exploration to PPO, which drastically improves performance under domain shifts. These results demonstrate the usefulness of Reactive Exploration, which helps PPO to recover and can facilitate forward transfer between stationarities. Details on hyperparameters, as well as the model architectures, can be found in Appendix B.

5.1 Environments

We use Jelly Bean World (JBW) (Platanios et al., 2020), to design environments with different levels and kinds of non-stationarities. JBW is commonly used in evaluating agents in lifelong learning, and supports infinite horizon RL, non-stationarity, multi-task, and multi-modal learning (Ngiam et al., 2011). Environments in JBW are procedurally generated POMDPs, where only a fraction of the created world is visible to the agent. By default, JBW provides multi-modal observations comprising images, as well as a vector representing the simulated diffusion of scent. It allows simple and efficient control over task design, and enables incorporating various types of non-stationarities. In our experiments, we configure the environment to contain two easily reachable types of items. Particularly, the items are represented by green and red pixels, which randomly occur at the same frequency. In the stationary setting, the collection of a green item yields +1.0 reward, while collecting a red item yields -1.0 reward. Collected items are added to the inventory of the agent. We refer to the stationary environment with fixed reward function as JellyBean-Base. A sketch of the JellyBean-Base environment is depicted in Figure 2.

Different types of non-stationarities are incorporated into JellyBean-Base by altering the reward function, and observations. First, we introduce a distribution shift in the reward function. To this end, we consider two different properties of non-stationarity: (i) the smoothness of change, and (ii) the interval of change. The former defines whether changes occur gradually over a certain timespan, or abruptly, i.e. piecewise. The interval is defined as the period after which the reward function is fully inverted. In this regard, we design four different non-stationary reward functions: 1) abrupt inversions every $n = 1\text{e}6$ steps, 2) gradual inversions over $n = 1\text{e}6$ steps, 3) abrupt inversions every $n = 1\text{e}5$ steps, and 4) gradual inversions over $n = 1\text{e}5$ steps. All of these inversions occur repeatedly throughout an experiment over 2e6 interaction steps with the environment.

Further, we incorporate two different cases of non-stationarities that affect observations and environment dynamics: (i) rotation of observations by 90 degrees (Rotated), and (ii) color changes of background and items (Color-Swap). The former preserves information prevalent in the environment, such as locations and color of items, while inducing a shift in the environment dynamics. The latter entirely disregards information by altering item and background color. We train our models for a total of 2e6 interaction steps and apply an abrupt shift after every $n = 5\text{e}5$ interaction steps, which would be sufficient for the algorithm to converge in the stationary setup. Additionally, we test our algorithms in a custom version of the CartPole environment, in which we alter the environment dynamics. For more details about the CartPole experiments, see Appendix A.2.

Unless stated otherwise, all experiments are conducted across 5 seeds, and the interquartile range (IQR) of the average reward over 1000 steps is shown. Moreover, we perform statistical significance tests via a Wilcoxon rank-sum test

(Wilcoxon, 1945) at a significance level of $\alpha_{wilcoxon} = 0.05$. In this regard, we gather the average rewards after the first shift occurred and subsample every 10th value to perform pairwise comparisons between algorithms.

5.2 Comparing PPO Against DQN Under Non-Stationarity

We conduct experiments to investigate how both PPO and DQN cope with non-stationarity in the reward function of JellyBean-Base, and which non-stationarities are particularly difficult to cope with. As a point of reference, we additionally show the performance of PPO and DQN trained on the stationary version of JellyBean-Base. The results for the different non-stationarities in the reward function are provided in Figure 3. For a large interval of change and a gradual non-stationarity (2nd column), we observe that PPO is capable of adapting to the changing reward function, exhibiting promising forward transfer. When the change occurs abruptly under the same interval, however (1st column), the performance of PPO deteriorates. The most difficult setting is represented by an abrupt change and a small interval (3rd column), while results are slightly more stable for gradual changes and small intervals for PPO (4th column). Nevertheless, PPO shows promising behavior in adapting to the different types of non-stationary reward functions. In contrast to recent work which showed that the application of the softmax to policy gradient algorithms is error-prone in non-stationary environments (Hennes et al., 2020), we find PPO to be quite stable. A possible reason for that might be the constrained optimization strategy via trust regions, which enforces a small policy gap. In contrast to PPO, DQN completely breaks regardless of the applied shift. We additionally show the performance of a recurrent PPO agent in Figure 13 on the reward shift experiments. We find the recurrent agent to be much less sample efficient and prone to overfitting.

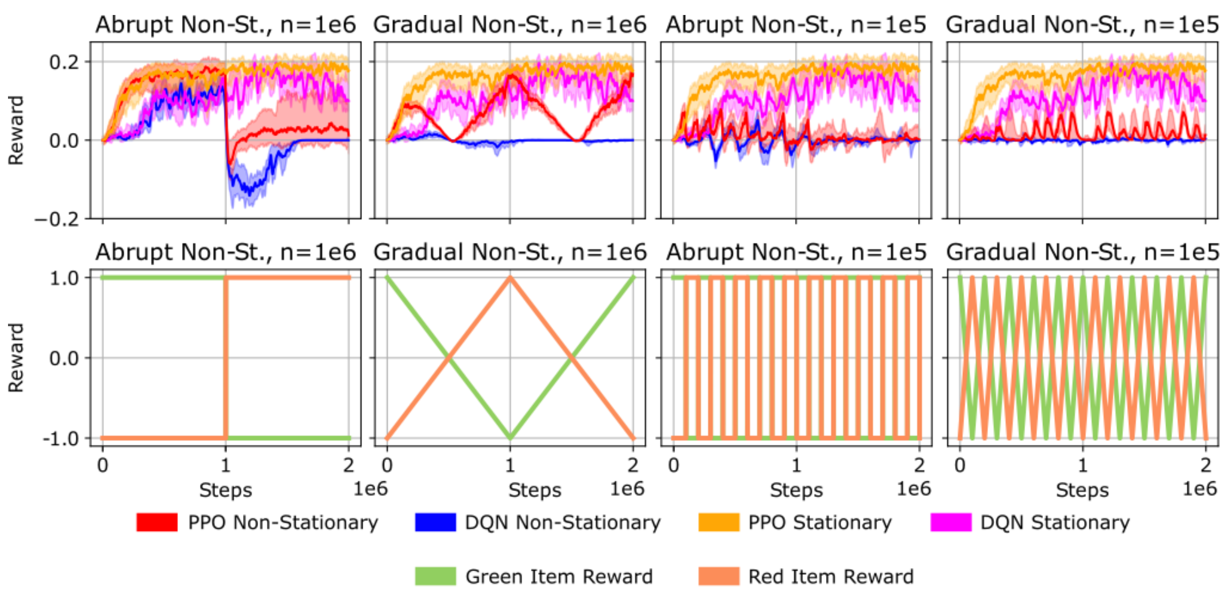


Figure 3: Non-stationary reward functions (bottom) with gradual or abrupt inversions, after or over the intervals (with $n = \{1e5, 1e6\}$), respectively. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing.

5.3 Failure Modes of DQN

As observed in previous experiments, DQN struggles to recover, especially when changes occur abruptly or very frequently. By default, vanilla DQN uses ϵ-greedy exploration with a decaying ϵ, which results in a lower exploration rate in the later stages of training. Furthermore, the replay buffer of DQN may still contain transitions collected under the previous stationarity regime after a domain shift has occurred. Prior work has highlighted the importance of correctly re-using collected experiences in off-policy algorithms when non-stationarities are encountered (Liotet et al., 2021; Chandak et al., 2020; Xie et al., 2021). In order to investigate the effect of the replay buffer and the exploration scheme, we conduct a thorough analysis on the components of DQN by:

1. reducing the replay buffer size with respect to the interval of change
2. injecting prior knowledge about when a change in the environment occurs into the ϵ-greedy schedule (i.e. restarting exploration after a change occurred)

3. prioritizing samples with a high temporal difference (TD) error via prioritized experience replay (PER, Schaul et al., 2016) without bias correction
4. sampling actions from a distribution obtained via a softmax transformation of Q-values instead of the ϵ-greedy schedule to add random exploration (DQN-stochastic)
5. measuring the effect of maximum entropy regularization via Soft Q-learning (SQL) (Haarnoja et al., 2017)

In Figure 7 we show results for various replay buffer sizes on the Color Swap and Rotated environments. We find that flushing the replay buffer with new samples more frequently leads to improved results on Rotation, however it only leads to slightly better performance on Color Swap. Also, reducing the buffer size by too much diminishes performance of DQN, indicating the existence of a sweet spot which depends on the interval of change. Furthermore, if the interval of change is known a-priori, the exploration schedule may be adapted as well to restart exploration after a change in the environment occurred. Figure 4 shows that with a reduced buffer size (left: 1e5, and right: 5e3) and adapted exploration schedule, DQN successfully recovers for abrupt shifts after $n = 1\text{e}6$. However, the information required for properly designing the exploration schedule and the replay buffer size is usually not available. For implementation details, see Appendix A.1.

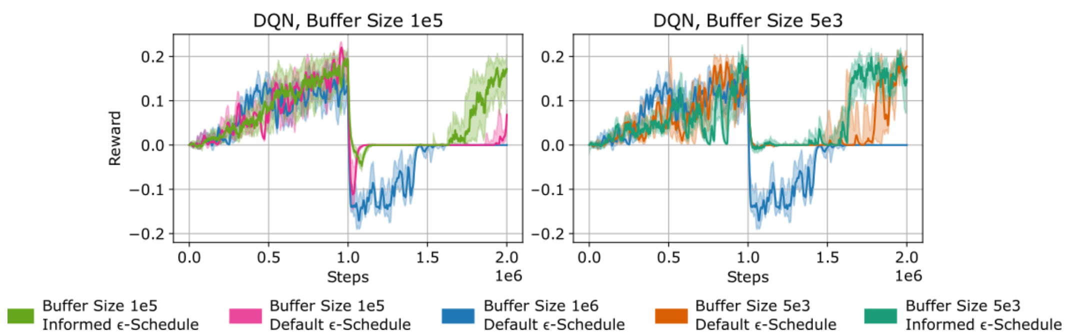


Figure 4: Ablation on DQN buffer sizes $1\text{e}5, 5\text{e}3$, and exploration schedule under abrupt reward inversion after $n = 1\text{e}6$. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing.

To alleviate the dependence on prior knowledge about the interval of change, we conduct additional experiments to improve upon vanilla DQN. Naturally, after a domain shift occurs, newly sampled transitions will lead to a high TD error and can be prioritized via PER. Initial experiments with PER indicate that the bias introduced via prioritized sampling can be beneficial to exploit in one particular distribution. However, it rarely exhibits gains for adapting to new ones (see DQN+PER, Figure 8). Alternative exploration schemes such as DQN-stochastic do not provide any gains over DQN (see Figure 8). Additionally, we compare DQN to SQL, which maximizes entropy while maximizing the return, and thus, is constantly encouraged to explore. Figure 8 shows that while SQL tends to improve over DQN after a shift in observations has occurred, it does not reach the performance of PPO. This is especially apparent in environments with a low interval of change or gradually applied shifts in the rewards. Also, these results highlight that even with the default buffer size, a more sophisticated exploration scheme can improve the performance over vanilla DQN without requiring external knowledge.

Our empirical evidence suggests that tuning the exploration strategy is a decisive factor for robustness in non-stationary environments when using DQN. While injecting prior knowledge about the interval of change improves performance, such information is usually not available a-priori. Even though PPO does not have access to this additional information, it outperforms DQN in environments with very frequent or gradual shifts.

5.4 Breaking PPO with Non-Stationarity

Although PPO is surprisingly resilient to changes in the reward function, we show that severe distribution shifts in the environment dynamics lead to failure cases. Prior experiments have illustrated the capability of PPO to handle gradual shifts in a large interval of change. Thus, in order to enforce failure cases of PPO, we focus exclusively on abrupt changes.

Results for both, Rotated and Color-Swap non-stationarities, are shown in Figure 5. As a baseline, we provide a comparison to a PPO agent trained in a stationary manner on JellyBean-Base and Color Swap/Rotation. We find that in the isolated stationary cases, both scenarios are easily solvable by PPO. However, if they are combined to create a non-stationary scenario, PPO is experiencing difficulties. A change in the rotation of the observations does not cause too much harm, and PPO shows signs of forward transfer. This might be due to the fact that non-rotated observations still contain useful information, such as locations of collectibles, that are transferred to the rotated case. However, severely altering the background color and the color of the items disregards most information that would facilitate forward transfer, resulting in complete failure as soon as the stationarity changes. Remarkably, as soon as the observations transition from the Color-Swap to the initial setting, PPO instantly recovers. This indicates that no catastrophic forgetting occurs in our experimental setup.

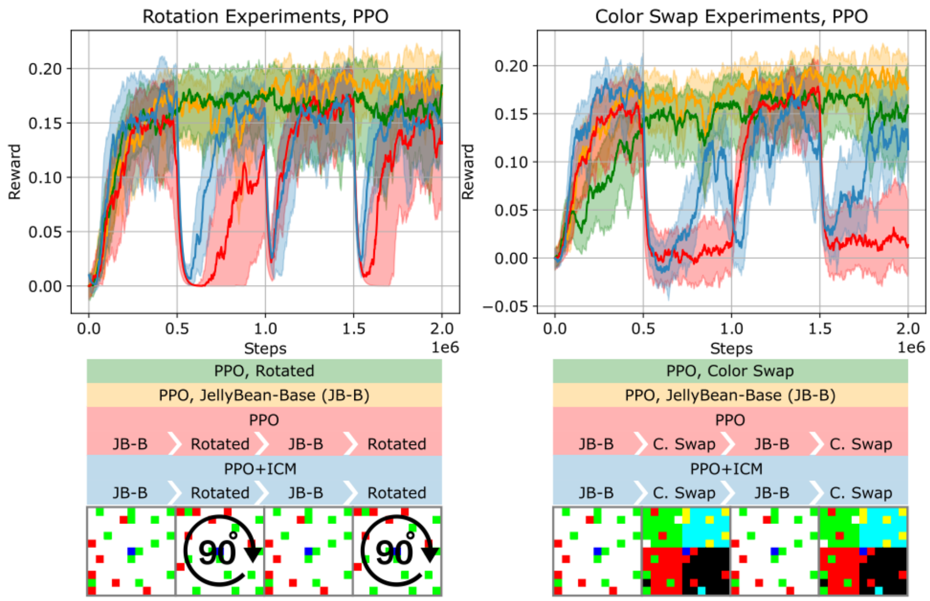


Figure 5: Experiments on non-stationary dynamics with PPO including PPO+ICM. Results are shown on the top row, while we visualize below when non-stationarity occurs due to Rotation or Color-Swap. Average rewards are shown every 1000 steps using an IQR and exponential average smoothing.

5.5 FIXING PPO WITH REACTIVE EXPLORATION

Reactive Exploration helps detect environment changes and to cope with them. Our previous experiments have shown that severe domain shifts considerably affect the performance of agents trained with PPO. As discussed in Section 4, we incorporate Reactive Exploration in the form of intrinsic curiosity, and investigate its capabilities in detecting dynamically occurring changepoints. Figure 6 shows the loss of the different components of the curiosity module. We find that both, the reward model and the forward dynamics model, accurately detect changes in the reward function and in the observations, respectively. The predictions of the forward dynamics model (right) appear to be slightly more noisy, which is expected since the task of predicting the next observation is inherently more difficult than predicting a single scalar value. More details on the predicted reward of items are given in Section A.3.

We repeat the Color-Swap experiment for which PPO failed, and incorporate the intrinsic rewards from the curiosity module. Specifically, we set $\alpha = 0.85$ and $\beta = 0$ since no changes in the reward occur in this particular setup. Equipped with Reactive Exploration, the agent manages to re-use previously acquired knowledge and quickly adapts to the new condition (see Figure 5 PPO+ICM). PPO+ICM significantly outperforms PPO on the Rotation ($p = 4.44\text{e}-9$) and Color-Swap environments ($p = 3.44\text{e} - 12$). Experiments for additional hyperparameter settings can be found in Figure 19 and a discussion on our hyperparameter choices can be found in Appendix B.1.2. In general, too low values of α results in too little or no exploration at all, since the extrinsic rewards outweigh their intrinsic counterpart. Vice-versa, too high values of α lead to exploration only. Hence, to facilitate recovery, intrinsic and extrinsic rewards should be balanced to be roughly in the same range. In Appendix A.2 and Appendix A.3, we provide additional evidence on the usefulness of Reactive Exploration on a different environment under shifting environment dynamics, and on the reward-shift experiments presented in Section 5.2, respectively.

Figure 6: Results for changepoint detection via Reactive Exploration. On the left, the loss of the reward model (top) is depicted when trained on prediction of a non-stationary reward function (bottom). On the right, the loss of the novelty-based observation model (top) is shown for predicting the next observation (bottom). Reactive Exploration successfully detects various points of change. Reward (right) and loss (left) are shown every 1000 steps using an IQR and exponential average smoothing.

6 Conclusion and Future Directions

In lifelong RL an agent learns via interaction with a constantly evolving environment. Such a scenario requires the agent to detect and cope with various domain shifts that arise over time, which most existing RL algorithms are not designed for. In this paper, we propose to leverage Reactive Exploration to track and react to continual domain shifts in lifelong RL. We conduct an empirical study comparing two popular RL algorithms, namely PPO and Q-learning, under a lifelong learning scenario when they face various types of non-stationarities. We find that PPO is quite durable in the face of non-stationarity, while DQN drastically fails in the most simple cases, when only the reward function changes. We conduct additional experiments to shed light on the failures of DQN, revealing that a more informative exploration mechanism can improve DQN. For drastic changes in the observations, PPO eventually fails to recover after a shift is encountered. As a remedy for such drastic shifts, we equip the PPO agent with Reactive Exploration in the form of an Intrinsic Curiosity Module, that helps to detect distribution changes, and to cope with them. Equipped with Reactive Exploration, the agent successfully solves tasks under heavy domain shifts where performance has plummeted before, and shows promising signs of forward transfer. Overall, our analysis yields the insight that exploration mechanisms are vital for detecting and consequently dealing with continually occurring domain shifts.

Naturally, there is a multitude of ideas to extend our work. Firstly, a more in-depth analysis of the effect of various non-stationarities on different algorithms potentially yields more insights on shortcomings and how to alleviate them. Since our work draws a connection to the model-based RL literature, an interesting direction would be investigating how model-based algorithms perform in our framework. Finally, building on our findings that stacking of various non-stationarities improves generalization (see Section A.9) we would like to thoroughly investigate this effect.

ACKNOWLEDGEMENTS

The ELLIS Unit Linz, the LIT AI Lab, the Institute for Machine Learning, are supported by the Federal State Upper Austria. IARAI is supported by Here Technologies. We thank the projects AI-MOTION (LIT-2018-6-YOU-212), AI-SNN (LIT-2018-6-YOU-214), DeepFlood (LIT-2019-8-YOU-213), Medical Cognitive Computing Center (MC3), INCONTROL-RL (FFG-881064), PRIMAL (FFG-873979), S3AI (FFG-872172), DL for GranularFlow (FFG-871302), AIRI FG 9-N (FWF-36284, FWF-36235), ELISE (H2020-ICT-2019-3 ID: 951847). We thank Audi.JKU Deep Learning Center, TGW LOGISTICS GROUP GMBH, Silicon Austria Labs (SAL), FILL Gesellschaft mbH, Anyline GmbH, Google, ZF Friedrichshafen AG, Robert Bosch GmbH, UCB Biopharma SRL, Merck Healthcare KGaA, Verbund AG, Software Competence Center Hagenberg GmbH, TÜV Austria, Frauscher Sensonic, AI Austria Reinforcement Learning Community and the NVIDIA Corporation.

REFERENCES

Joshua Achiam and Shankar Sastry. Surprise-based intrinsic motivation for deep reinforcement learning. *CoRR*, abs/1703.01732, 2017.

Ryan Prescott Adams and David J. C. MacKay. Bayesian online changepoint detection, 2007.

Thomas Adler, Johannes Brandstetter, Michael Widrich, Andreas Mayr, David Kreil, Michael Kopp, Günter Klambauer, and Sepp Hochreiter. Cross-domain few-shot learning by representation fusion. *arXiv preprint arXiv:2010.06498*, 2020.

Lucas Nunes Alegre, Ana L. C. Bazzan, and Bruno C. da Silva. Minimum-delay adaptation in non-stationary reinforcement learning via online high-confidence change-point detection. In Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé (eds.), *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pp. 97–105. ACM, 2021.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martín Arjovsky, Alexander Pritzel, Andew Bolt, et al. Never give up: Learning directed exploration strategies. *arXiv preprint arXiv:2002.06038*, 2020.

Michèle Basseville and Igor V. Nikiforov. Detection of abrupt changes: Theory and application. *Autom.*, 32(8): 1235–1236, 1996. doi: 10.1016/0005-1098(96)82332-6.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29:1471–1479, 2016.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Mach. Learn.*, 79(1-2):151–175, 2010. doi: 10.1007/s10994-009-5152-4.

Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. On the optimization of a synaptic learning rule. *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, 2, 1992.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Optimal exploration–exploitation in a multi-armed bandit problem with non-stationary rewards. *Stochastic Systems*, 9(4):319–337, 2019.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Yuri Burda, Harrison Edwards, Amos J. Storkey, and Oleg Klimov. Exploration by random network distillation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Giuseppe Canonaco, Marcello Restelli, and Manuel Roveri. Model-free non-stationarity detection and adaptation in reinforcement learning. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang (eds.), *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pp. 1047–1054. IOS Press, 2020. doi: 10.3233/FAIA200200.

Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 418–427. PMLR, 16–18 Apr 2019.

Gail A. Carpenter and Stephen Grossberg. Art 2: self-organization of stable category recognition codes for analog input patterns. *Appl. Opt.*, 26(23):4919–4930, Dec 1987. doi: 10.1364/AO.26.004919.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Yash Chandak, Georgios Theocharous, Shiv Shankar, Martha White, Sridhar Mahadevan, and Philip S. Thomas. Optimizing for the future in non-stationary mdps. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1414–1425. PMLR, 2020.

Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. Reinforcement learning for non-stationary markov decision processes: The blessing of (more) optimism. *CoRR*, abs/2006.14389, 2020.

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym, 2018.

David A. Cohn. Neural network exploration using optimal experiment design. In J. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems 6*, pp. 679–686. Morgan Kaufmann, 1994.

Marius-Constantin Dinu, Markus Hofmarcher, Vihang P. Patil, Matthias Dorfer, Patrick M. Blies, Johannes Brandstetter, Jose A. Arjona-Medina, and Sepp Hochreiter. *XAI and Strategy Extraction via Reward Redistribution*, pp. 177–205. Springer International Publishing, Cham, 2022. ISBN 978-3-031-04083-2. doi: 10.1007/978-3-031-04083-2_10.

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

Benjamin Ehret, Christian Henning, Maria R Cervera, Alexander Meulemans, Johannes von Oswald, and Benjamin F Grewe. Continual learning in recurrent neural networks with hypernetworks. *arXiv preprint arXiv:2006.12109*, 2020.

Chrisantha Fernando, Dylan S. Banarse, Charles Blundell, Yori Zwols, David R. Ha, Andrei A. Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *ArXiv*, abs/1701.08734, 2017.

Sebastian Flennerhag, Yannick Schroecker, Tom Zahavy, Hado van Hasselt, David Silver, and Satinder Singh. Bootstrapped meta-learning. In *International Conference on Learning Representations*, 2022.

Jay W. Forrester. Counterintuitive behavior of social systems. *Theory and Decision*, 2(2):109–140, December 1971. ISSN 1573-7187. doi: 10.1007/BF00148991.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Francisco M. Garcia and Philip S. Thomas. A meta-mdp approach to exploration for lifelong reinforcement learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 5692–5701, 2019.

Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems, 2008.

Sandeep Goel and Manfred Huber. Subgoal discovery for hierarchical reinforcement learning using learned policies. In Ingrid Russell and Susan M. Haller (eds.), *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference, May 12-14, 2003, St. Augustine, Florida, USA*, pp. 346–350. AAAI Press, 2003.

Gunshi Gupta, Karmesh Yadav, and Liam Paull. La-maml: Look-ahead meta learning for continual learning. *arXiv preprint arXiv:2007.13904*, 2020.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, pp. 1352–1361. PMLR, 2017.

Cédric Hartland, Nicolas Baskiotis, Sylvain Gelly, Michèle Sebag, and Olivier Teytaud. Change Point Detection and Meta-Bandits for Online Learning in Dynamic Environments. In *CAp 2007 : 9è Conférence francophone sur l'apprentissage automatique*, pp. 237–250, Grenoble, France, July 2007.

Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9769–9776. IEEE, 2019.

Donald O. Hebb. *The organization of behavior: A neuropsychological theory*. Wiley, New York, 1949. ISBN 0-8058-4300-0.

Daniel Hennes, Dustin Morrill, Shayegan Omidshafiei, Rémi Munos, Julien Perolat, Marc Lanctot, Audrunas Gruslys, Jean-Baptiste Lespiau, Paavo Parmas, Edgar Duèñez Guzmán, and Karl Tuyls. Neural replicator dynamics: Multiagent learning via hedging policy gradients. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, pp. 492–501, Richland, SC, 2020. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Markus Holzleitner, Lukas Gruber, Jose A. Arjona-Medina, Johannes Brandstetter, and Sepp Hochreiter. Convergence Proof for Actor-Critic Methods Applied to PPO and RUDDER. *CoRR*, abs/2012.01399, 2020. arXiv: 2012.01399.

Joey Hong, Branislav Kveton, Manzil Zaheer, Yinlam Chow, and Amr Ahmed. Non-stationary off-policy optimization. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2494–2502. PMLR, 13–15 Apr 2021.

Maximilian Igl, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson. The impact of non-stationarity on generalisation in deep reinforcement learning. *CoRR*, abs/2006.05826, 2020.

Khurram Javed and Martha White. *Meta-Learning Representations for Continual Learning*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Leslie P. Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101:99–134, 1998.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *CoRR*, abs/2012.13490, 2020.

Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In Mario A. Nascimento, M. Tamer Özsu, Donald Kossmann, Renée J. Miller, José A. Blakeley, and K. Bernhard Schiefer (eds.), *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, pp. 180–191. Morgan Kaufmann, 2004. doi: 10.1016/B978-012088469-8.50019-X.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Erwan Lecarpentier and Emmanuel Rachelson. Non-stationary markov decision processes a worst-case approach using model-based reinforcement learning. In *NeurIPS*, 2019.

Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Long Ji Lin. Self-Improving Reactive Agents Based On Reinforcement Learning, Planning and Teaching. *Mach. Learn.*, 8:293–321, 1992. doi: 10.1007/BF00992699.

Pierre Liotet, Francesco Vidaich, Alberto Maria Metelli, and Marcello Restelli. Lifelong hyper-policy optimization with multiple importance sampling regularization. *CoRR*, abs/2112.06625, 2021.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. Count-based exploration with the successor representation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 5125–5133. AAAI Press, 2020.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Davide Maltoni and Vincenzo Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, 2019.

Jarryd Martin, Suraj Narayanan Sasikumar, Tom Everitt, and Marcus Hutter. Count-based exploration in feature space for reinforcement learning. In Carles Sierra (ed.), *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 2471–2478. ijcai.org, 2017. doi: 10.24963/ijcai.2017/344.

Gerrit W. Maus, Jason Fischer, and David Whitney. Motion-dependent representation of space in area mt+. *Neuron*, 78:554–562, 2013.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pp. 109–165. Academic Press, 1989. doi: 10.1016/S0079-7421(08)60536-8.

Amy McGovern and Andrew G. Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In Carla E. Brodley and Andrea Pohoreckyj Danyluk (eds.), *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pp. 361–368. Morgan Kaufmann, 2001.

Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, 4:504, 2013.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. Never-ending learning. *Communications of the ACM*, 61(5): 103–115, 2018. doi: 10.1145/3191513.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. arXiv, 2013.

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.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *ICML*, 2011.

Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2721–2730. PMLR, 2017.

Pierre-Yves Oudeyer and Frédéric Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers Neurorobotics*, 1:6, 2007. doi: 10.3389/neuro.12.006.2007.

Sindhu Padakandla, Prabuchandran K. J., and Shalabh Bhatnagar. Reinforcement learning in non-stationary environments. *CoRR*, abs/1905.03970, 2019.

Fabian Paischer, Thomas Adler, Vihang Patil, Angela Bitto-Nemling, Markus Holzleitner, Sebastian Lehner, Hamid Eghbal-zadeh, and Sepp Hochreiter. History compression via language models in reinforcement learning. 2022. doi: 10.48550/ARXIV.2205.12258.

German I. Parisi, Jun Tani, Cornelius Weber, and Stefan Wermter. Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization. *Frontiers in neurorobotics*, pp. 78, 2018.

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

Vihang P Patil, Markus Hofmarcher, Marius-Constantin Dinu, Matthias Dorfer, Patrick M Blies, Johannes Brandstetter, Jose A Arjona-Medina, and Sepp Hochreiter. Align-rudder: Learning from few demonstrations by reward redistribution. *arXiv preprint arXiv:2009.14108*, 2020.

Emmanouil Antonios Platanios, Abulhair Saparov, and Tom Mitchell. Jelly bean world: A testbed for never-ending learning. *arXiv preprint arXiv:2002.06306*, 2020.

K. J. Prabuchandran, Nitin Singh, Pankaj Dayama, Ashutosh Agarwal, and Vinayaka Pandit. Change point detection for compositional multivariate data. *Appl. Intell.*, 52(2):1930–1955, 2022. doi: 10.1007/s10489-021-02321-6.

Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3, 2019.

Roberta Raileanu and Tim Rocktäschel. RIDE: rewarding impact-driven exploration for procedurally-generated environments. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Vishnu Raj and Sheetal Kalyani. Taming non-stationary bandits: A bayesian approach. *arXiv preprint arXiv:1707.09727*, 2017.

Suraj Regmi. Infinite steps cartpole problem with variable reward, 2020. Accessed: 2022-03-09.

Mark Bishop Ring et al. Continual learning in reinforcement environments. 1994.

Yoan Russac, Claire Vernade, and Olivier Cappé. Weighted linear bandits for non-stationary environments. *arXiv preprint arXiv:1909.09146*, 2019.

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

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Jürgen Schmidhuber. Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook. *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, 1987.

Jürgen Schmidhuber. Making the world differentiable: On using fully recurrent self-supervised neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical Report FKI-126-90 (revised), Institut für Informatik, Technische Universität München, November 1990.

Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In J. A. Meyer and S. W. Wilson (eds.), *Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 222–227. MIT Press/Bradford Books, 1991.

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

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

Murray Shanahan and Melanie Mitchell. Abstraction for deep reinforcement learning. *CoRR*, abs/2202.05839, 2022.

Bradly C. Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *CoRR*, abs/1507.00814, 2015.

Jan Storck, Sepp Hochreiter, and Jürgen Schmidhuber. Reinforcement driven information acquisition in non-deterministic environments. *International Conference on Artificial Neural Networks (ICANN)*, 1995.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for markov decision processes. *J. Comput. Syst. Sci.*, 74(8):1309–1331, 2008. doi: 10.1016/j.jcss.2007.08.009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press, 1998.

Richard Stuart Sutton. *Temporal credit assignment in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 1984.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 2753–2762, 2017.

Sebastian Thrun. Lifelong learning algorithms. In Sebastian Thrun and Lorien Y. Pratt (eds.), *Learning to Learn*, pp. 181–209. Springer, 1998. doi: 10.1007/978-1-4615-5529-2_8.

Sebastian Thrun and Tom M Mitchell. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46, 1995.

Hao Wang, Bing Liu, Shuai Wang, Nianzu Ma, and Yan Yang. Forward and backward knowledge transfer for sentiment classification. *arXiv preprint arXiv:1906.03506*, 2019.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, May 1992. ISSN 1573-0565. doi: 10.1007/BF00992698.

Gerhard Widmer and Miroslav Kubát. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69–101, 2004.

Michael Widrich, Markus Hofmarcher, Vihang Prakash Patil, Angela Bitto-Nemling, and Sepp Hochreiter. Modern Hopfield Networks for Return Decomposition for Delayed Rewards. In *Deep RL Workshop NeurIPS 2021*, 2021.

Frank Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. Publisher: [International Biometric Society, Wiley].

Maciej Wołczyk, Michał Zając, Razvan Pascanu, Lukasz Kucinski, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020.

Qingyun Wu, Naveen Iyer, and Hongning Wang. Learning contextual bandits in a non-stationary environment. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 495–504, 2018.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11393–11403. PMLR, 2021.

Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task-agnostic continual learning using online variational bayes with fixed-point updates. *Neural Computation*, 33:3139–3177, 2021.

Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu, Kurt Keutzer, Joseph E. Gonzalez, and Yuandong Tian. Noveld: A simple yet effective exploration criterion. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Peng Zhao, Lijun Zhang, Yuan Jiang, and Zhi-Hua Zhou. A simple approach for non-stationary linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pp. 746–755. PMLR, 2020.

SUPPLEMENTARY MATERIAL

Table of Contents

A EXTENDED RESULTS

A.1 DQN FAILURES - EXPERIMENT DETAILS

In this section, we elaborate on the limiting factors of DQN in more detail. In particular, we conduct additional experiments with ϵ-exploration schedules, reduced buffer sizes, Prioritized Experience Replay, Soft Q-Learning, and DQN-Stochastic.

Vanilla DQN with reduced buffer size and ϵ-exploration schedule. We repeat the experiment with abrupt changes after a long interval of change of $n = 1e6$ steps from Section 5.2. In this setting, we add an ϵ-exploration schedule in such a way that a second exploration surge is repeated exactly when the reward function changes in order to restart exploration. A similar schedule is imposed by Reactive Exploration, as highlighted in Figure 6.

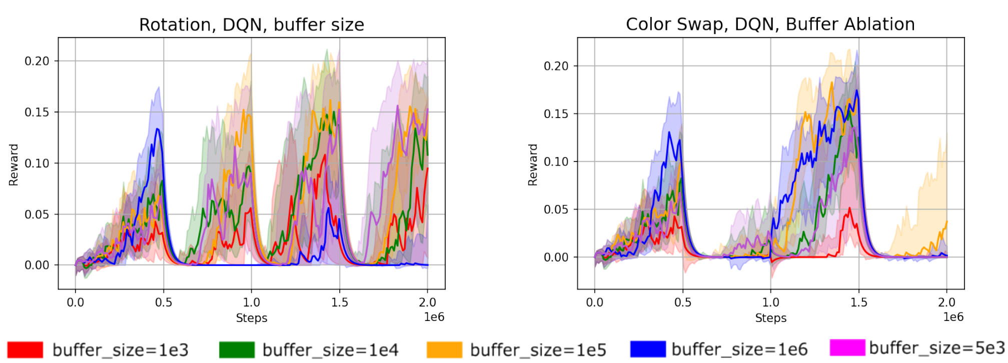


Figure 7: DQN buffer size ablation. Experiments for non-stationary dynamics. We investigate the effect using different buffer sizes with DQN. The default buffer size is 1e6.

The default ϵ-schedule is initialized at $\epsilon = 1.0$, which resembles purely random behavior in the beginning, and decaying to $\epsilon = 0.05$ over the first 10% of interaction steps. For the informative schedule, we repeat this ϵ schedule every time a domain shift occurs. We call this schedule "informed" since it assumes prior knowledge about the occurring shifts. Additionally, we limit the buffer size to contain either $1e5$, or $5e3$ sampled transitions. We create four experiments for all combinations of default versus informed ϵ-schedule and buffer size $1e5$ versus $5e3$, and contrast them to the default $1e6$ with the default ϵ-schedule. The results of these experiments can be seen in Figure 4. In addition, we provide an ablation over the buffer sizes $\{1e3, 5e3, 1e4, 1e5\}$ for the observation shift environments in Figure 7. Reducing the buffer size results in quick recovery in the Rotation environment as compared to the default buffer size of $1e6$. In the Color-Swap environment, a smaller buffer results in marginal gains, but a promising trend in terms of recovery.

DQN + PER. We use the default value for $\alpha_{PER} = 0.6$. Our initial suspicion is that the bias introduced via prioritization of samples with a high TD-error might have a positive impact in the non-stationary setup. This is due to the fact that transitions after a shift will exhibit a higher TD-error than others in the buffer and will be prioritized. Therefore, we set $\beta_{PER} = 0$. We show in Figure 8 that while PER improves performance for one type of stationarity, agents only recover rarely and there is no significant improvement upon DQN in terms of recovery.

Soft Q-Learning In Figure 8 we additionally show the learning curve for SQL on all environments. SQL maximizes entropy while maximizing the return and, thus, is constantly encouraged to explore the environment. As described in Section 5.3, While SQL tends to improve over DQN after a shift in observations has occurred, it does not reach the performance of PPO.

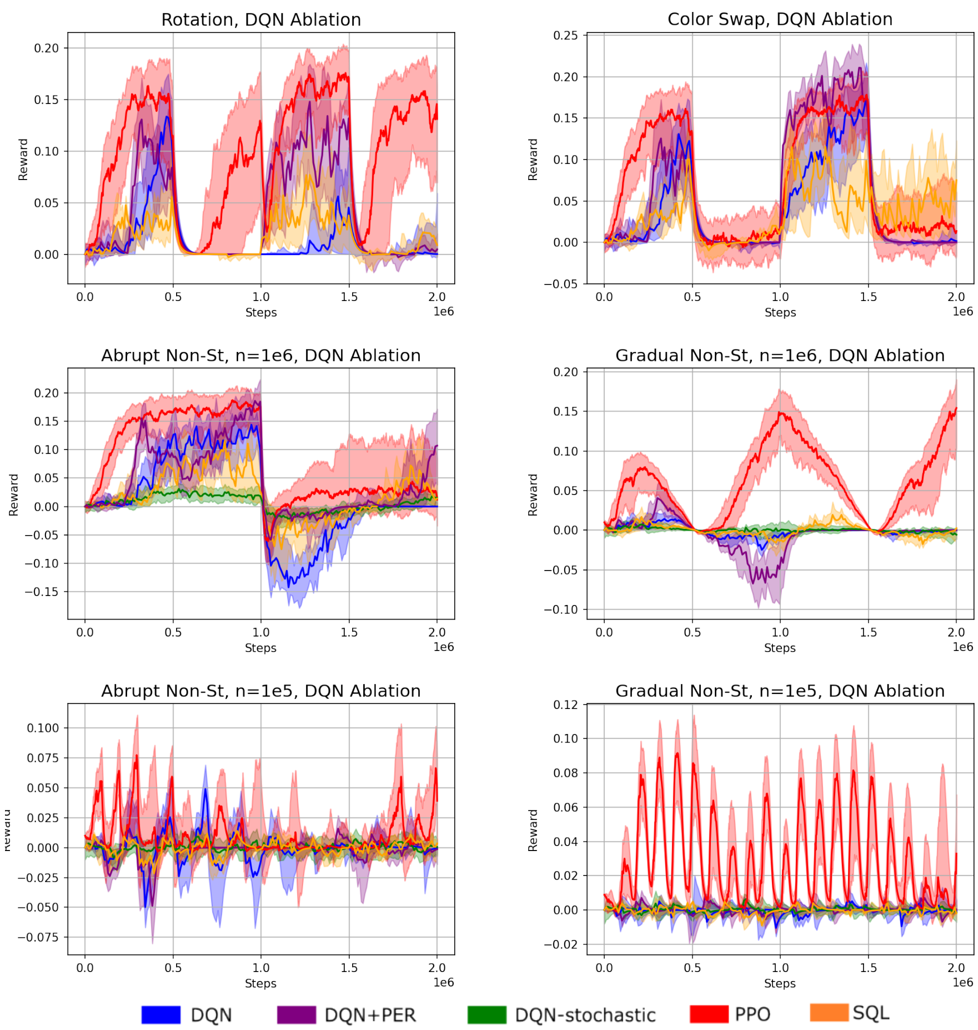


Figure 8: Experiments for non-stationary dynamics (top row) and non-stationary reward functions (middle, bottom row). We investigate the effect of DQN+PER and SQL and compare them to the PPO and DQN baselines.

Stochastic DQN. Figure 8, presents the performance of DQN-stochastic. DQN-stochastic is a form of randomized exploration which is enforced via a probability distribution over predicted Q-values. We observe that this results in inferior performance due to small differences between Q-values, which results in an almost uniform action distribution, and thus almost fully random exploration and little to no exploitation.

A.2 Repeating Experiments in CartPole

We repeat several of our experiments using a custom lifelong version of the popular CartPole environment, in which we can easily induce non-stationarity in environment dynamics. First, we elaborate on the environment, followed by experiments with PPO and DQN. Further, we design difficult non-stationarity to break PPO, and finally, show promising behavior in terms of recovery after adding Reactive Exploration.

In the Cartpole environment, the agent must balance a pole which is place on a cart that it can freely move. Usually, the environment is reset after the pole leans towards one side with a certain angle. We reset the pole when it drops, but allow the cart to move infinitely in a single episode. In this regard, we set the cart's velocity and acceleration to 0 when it hits the edges of the box, which turns the problem to an infinite horizon problem. This setup resembles the most similar to the episodic case, since restricting the cart movement inside its box prevents a velocity buildup that would not be possible in the episodic setting.

In addition, we apply a variable reward function (Regmi, 2020), in which a perfectly balanced pole yields $+0.5$ reward, and the perfectly centered cart yields $+0.5$ reward, totaling $+1.0$. A sum of 0 reward is yielded exactly at the point at which both the cart position x and the pole angle θ are at their original thresholds. If the pole drops further, the reward drops drastically below 0. Formally, the reward function is defined as:

$$r = 1 - 11.52 \cdot x^2 - \frac{\theta^2}{288}$$

Creating settings with modified dynamics can be achieved by changing the properties of the environment, which are based on physics calculations. The default settings, which we use as a baseline for comparison against our modifications, are defined by a gravity of 9.8, a cart mass of 1.0, a pole mass of 0.1, a pole length of 0.5, and a force magnitude of 10.0. The sign of the force magnitude controls the direction of the cart movement.

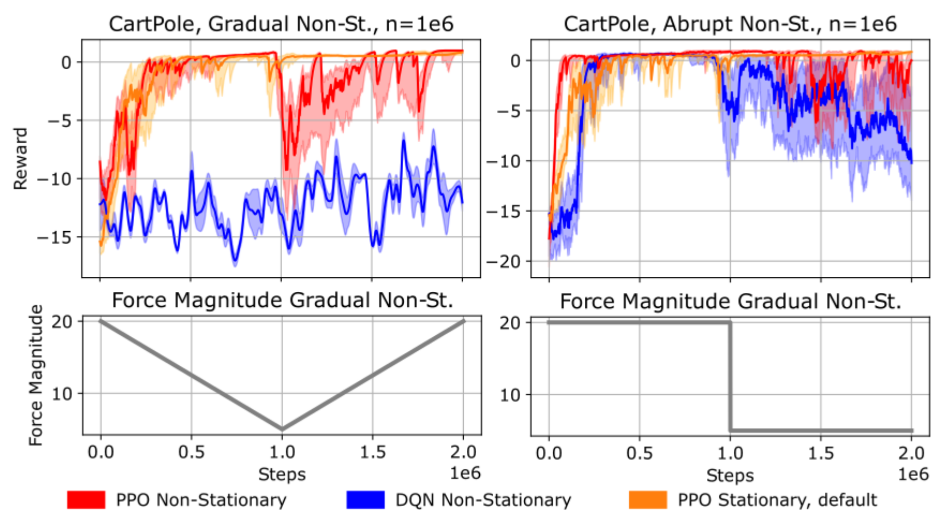


Figure 9: Experiments for non-stationary dynamics in CartPole, comparing DQN and PPO agents. The top shows the results, and below we visualize how the non-stationary dynamics occur. Average rewards are logged every 1000 steps and shown using an IQR and exponential average smoothing.

We create abrupt and gradual non-stationarity in the environment dynamics. To this end, we change the force magnitude from 20 to 5 either gradually over $1e6$ steps and back again over the next $1e6$ steps, or abruptly every $1e6$ steps. Again, we add the performance of PPO and DQN agents trained on the stationary environment under default parameters as a baseline (see Figure 9). In the results for the gradual shift (left), we can observe how DQN is not capable of solving the task at all. This is likely due to the fact that adapting to a changing force magnitude in order to keep the balance is too difficult with outdated knowledge in the replay buffer. PPO, on the other hand, manages to learn how to balance the pole quite well. However, we can observe that once the force starts to increase instead of decrease, PPO agents become more unstable. In the abrupt case (right), both algorithms initially learn to balance as expected. Once the environment dynamics change, PPO agents adapt well with minor instabilities, while DQN again struggles more severely.

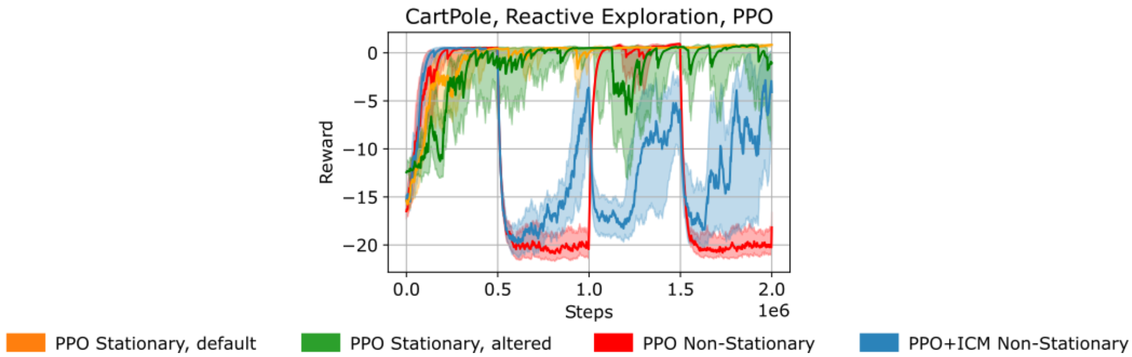


Figure 10: Results for experiments on difficult non-stationarity in CartPole to break PPO and Reactive Exploration to alleviate the difficulties. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing. For the default settings, we use a cart mass of 1.0, a pole length of 0.5, and a force magnitude of 10.0. When switching to the altered setting, these turn into a pole length of 1.0, a cart mass of 2.0, and a force magnitude of -10.0 (inverting the direction of movement).

Next, we explore domain shifts that reliably render PPO unable to recover. As we experienced in prior experiments, simple modifications such as increasing or decreasing the force magnitude only introduces minor instabilities. We instead focus on changing the sign of the force magnitude, resulting in an inversion between left and right movement of the cart. Additionally, we set the environment parameters to a pole length of 1.0, and a cart mass of 2.0. Further, we apply the same interval of change as used in our JBW experiments at $5e5$ steps until force magnitude is inverted. A stationary baseline is trained on the default as well as the altered dynamics and shown as a reference. The results show that both the default and the new dynamics are learnable in the stationary case, but when switching between them, PPO is unable to recover (see Figure 10, red line).

Finally, we apply our Reactive Exploration, including the forward and inverse dynamics models, as well as the reward model. For scaling of the intrinsic reward, we observe the magnitude of the model rewards in early experiments, scale down r_t^r by a factor of 0.2, and set the parameters $\alpha = 0.15$, and $\beta = 0.15$. Results (see Figure 10, blue line) show that agents equipped with Reactive Exploration successfully recover after encountering the shift in the environment dynamics. For additional details on hyperparameter selection, see Appendix B.

A.3 Reward Model Predictions

We visualize the predictions of the reward model for the abrupt inversion of the reward function every $n = 5e5$ steps (Section 5.5). Figure 11 shows that the reward model almost perfectly predicts the true reward function, as the model immediately adapts to the changes. Thus, a reward model can effectively detect and respond to non-stationarities in the reward function.

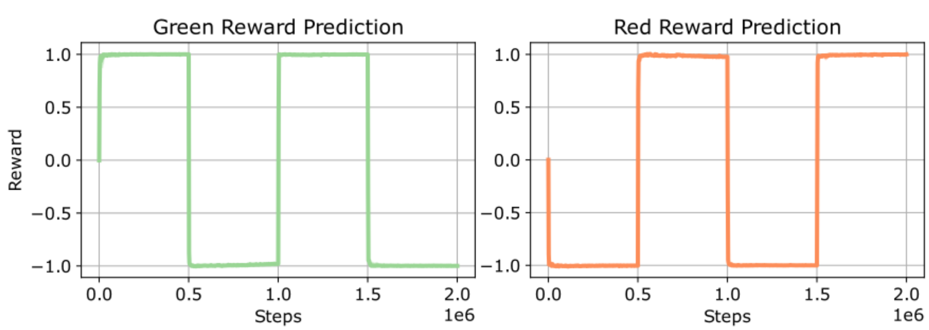


Figure 11: Predicted reward for picking up green and red reward using the reward model, which perfectly represent the true rewards yielded by green and red items as they are inverted every $n = 5e5$ steps between $+1.0$ and -1.0. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing.

A.4 Reactive Exploration for Reward Shift Experiments

In Figure 5 (Section 5.4), we showed how Reactive Exploration mitigates the impact of observation shifts on the agent's performance. For completeness, we provide experiments with Reactive Exploration on environments with reward shifts (Figure 3). In Figure 12, we show the learning curves on abrupt (left) and gradual (right) reward shifts over $n = 1e6$ (top) and $n = 1e5$ (bottom) steps for the agent equipped with Reactive Exploration (PPO+ICM, green) and compare to PPO (red) and DQN (blue) baselines. For all four tasks, we use an intrinsic reward weight $\alpha = 0.85$ for PPO+ICM.

On experiments comprising abrupt shifts, PPO+ICM significantly outperforms PPO for an interval of change of $n = 1e6$ (top left, $p = 8.28e-9$). Even for frequently occurring reward shifts ($n = 1e5$, bottom left), PPO+ICM significantly outperforms PPO ($p = 7.87e-9$). On gradual shifts, PPO without exploration already performs close to optimal. Thus, there is not much gain in adding Reactive Exploration.

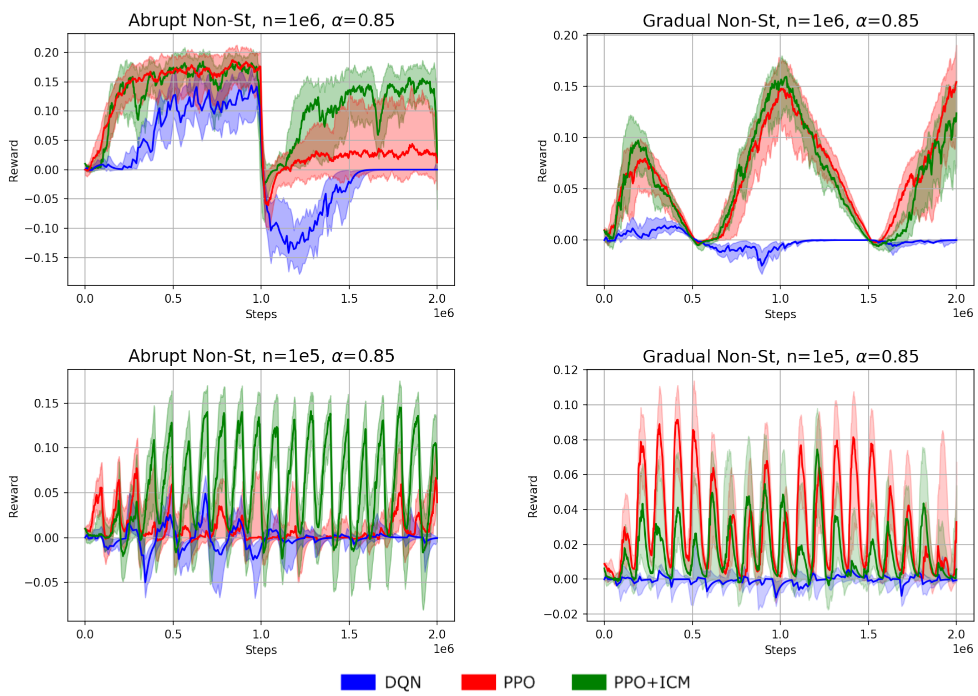


Figure 12: Experiments for non-stationary reward functions with abrupt (left) or gradual (right) reward shifts. We compare the baselines PPO and DQN to PPO with Reactive Exploration (ICM, $\alpha = 0.85$).

A.5 Recurrent PPO Ablation

To investigate the robustness of a recurrent architecture to the different domain shifts, we add an experiment where we use an LSTM (Hochreiter & Schmidhuber, 1997) on top of the convolutional backbone (PPO-LSTM). Since the task the agent is required to solve is not dependent on memory, we sample sequence chunks of length 8 and reset hidden and cell state for computing the gradients. Figure 13 shows the performance of PPO-LSTM compared to DQN and PPO. We observe that a recurrent agent is much less sample efficient than a Markovian policy. This is clearly visible for an abrupt shift and a large interval of change ($n = 1e6$), where PPO-LSTM shows a promising trend in terms of recovery after being trained sufficiently long (top left). However, for gradual shifts (right column) or frequent shifts ($n = 1e5$, bottom left) PPO-LSTM performs poorly. This is due to the fact that it frequently overfits to avoid collecting any item, since it is not capable of adapting fast enough to the new stationarity.

A.6 Possible Instantiations for Reactive Exploration

As mentioned in Section 4 we instantiate our Reactive Exploration based on ICM (Pathak et al., 2017). However, any other exploration mechanism suited for the infinite horizon setup may be used. In this regard, we investigate different instantiations of Reactive Exploration.

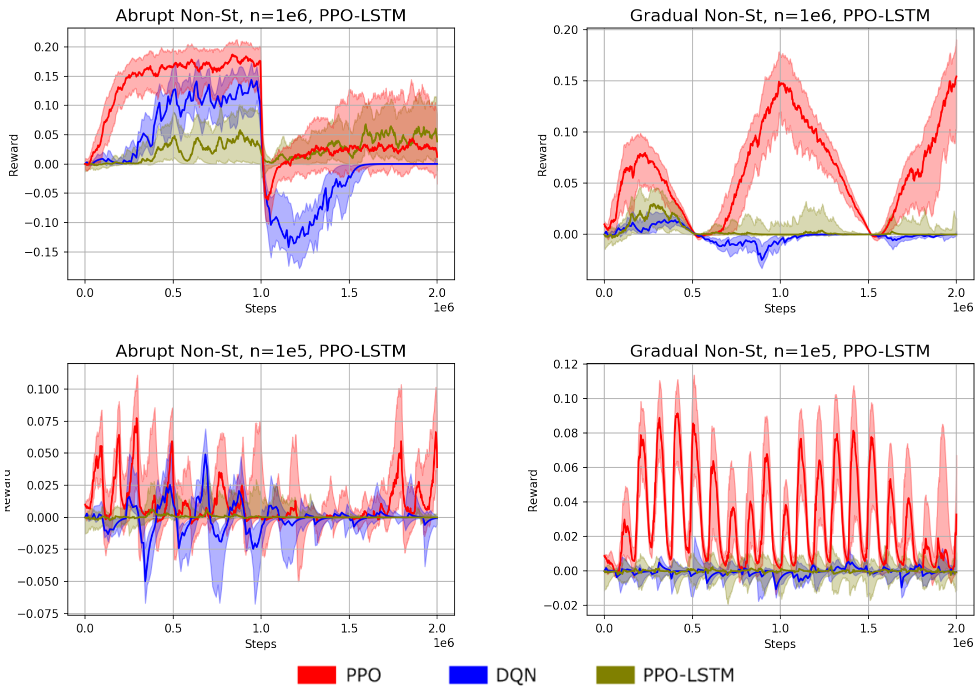


Figure 13: PPO-LSTM ablation. Experiments for non-stationary reward functions with abrupt (left) or gradual (right) reward shifts. We compare the baselines PPO and DQN to PPO-LSTM.

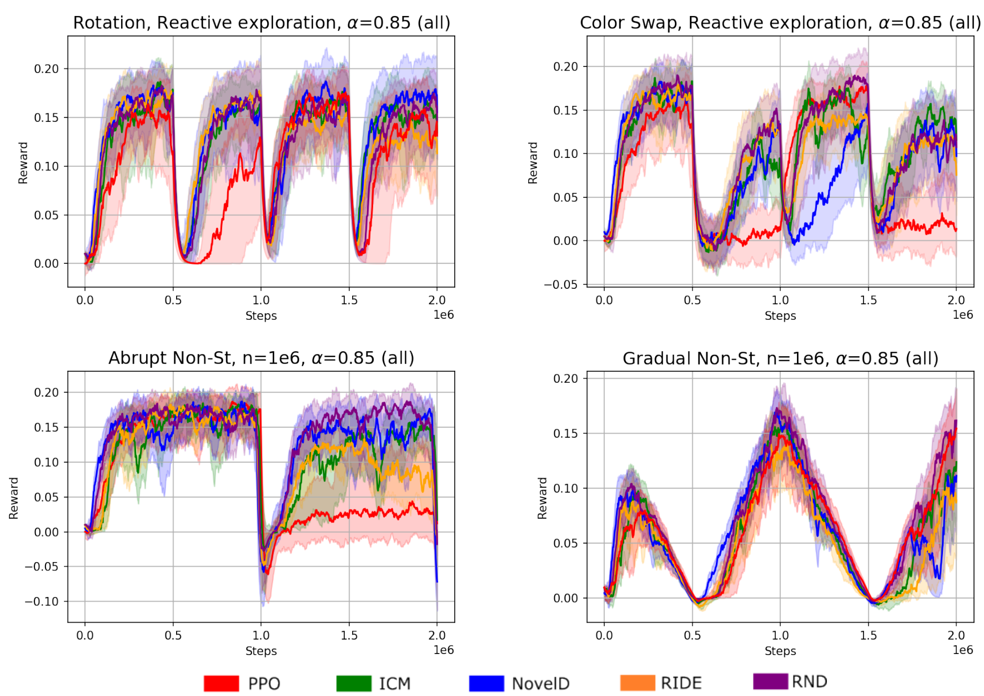


Figure 14: Experiments for non-stationary dynamics (top row) and non-stationary reward functions (bottom row). We investigate the effect of three additional exploration methods (RND, RIDE and NovelD) and compare them to the PPO baseline. We use $\alpha = 0.85$ for all experiments.

Specifically, we consider three other prominent exploration methods: RND (Burda et al., 2019), RIDE (Raileanu & Rocktäschel, 2020), and NovelD (Zhang et al., 2021). Similar to ICM, these three additional methods assign a curiosity-based reward bonus for exploration. RND maintains two networks that receive observations from the environment: (i) a fixed, randomly initialized target network and (ii) a predictor network, that aims to predict the outputs of the target network. The intrinsic reward is computed as the prediction error of the predictor network. Unlike ICM and RND, both RIDE and NovelD compute the intrinsic rewards as the novelty difference of two consecutive states. To quantify the novelty of states, RIDE and NovelD rely on ICM and RND, respectively. In contrast to ICM and RND, however, RIDE and NovelD were designed for episodic environments and maintain a lookup table of state visitation counts per episode. RIDE normalizes the exploration bonus by the root of the episodic state visitation count, and NovelD only assigns the intrinsic reward the first time a state was visited in the current episode. As such, in their original form, they cannot be applied to infinite horizon problems. Therefore, we make slight adjustments, and only maintain lookup tables per rollout of PPO (2048 steps), not per episode.

In Figure 14 we present results for different instances of Reactive Exploration. We conduct experiments on two environments with observation-shifts (Rotation and Color-Swap, top row) as well as two environments with reward shifts (abrupt/gradual shift after/over 1e6 steps, bottom row). In line with experiments presented in Section B.1.2, all agents use the intrinsic reward weight $\alpha = 0.85$. We show the learning curves over eight seeds and compare them to the PPO baseline. Overall, we observe that RND (purple), RIDE (yellow) and NovelD (blue) perform on-par with our instantiation of ICM. Notably, the only method that significantly outperforms the PPO baseline on gradual reward shift (bottom right) is RND ($p = 1.02\text{e}-5$). On all other experiments, all methods significantly outperform the PPO baseline. These results suggest, that other state-of-the-art exploration mechanisms may be used for Reactive Exploration in the lifelong RL setup.

A.7 Comparing Reactive Exploration and Entropy Maximization

To shed more light on what types of exploration are most efficient in facilitating recovery of PPO, we conduct another set of experiments. Particularly, we compare our curiosity-based exploration (PPO+ICM) against the standard maximum entropy regularization of PPO (PPO+ME) with $\epsilon = 0.01$. In addition, we show reference results for PPO, and for combining both Reactive Exploration via ICM and entropy regularization (PPO+ICM+ME).

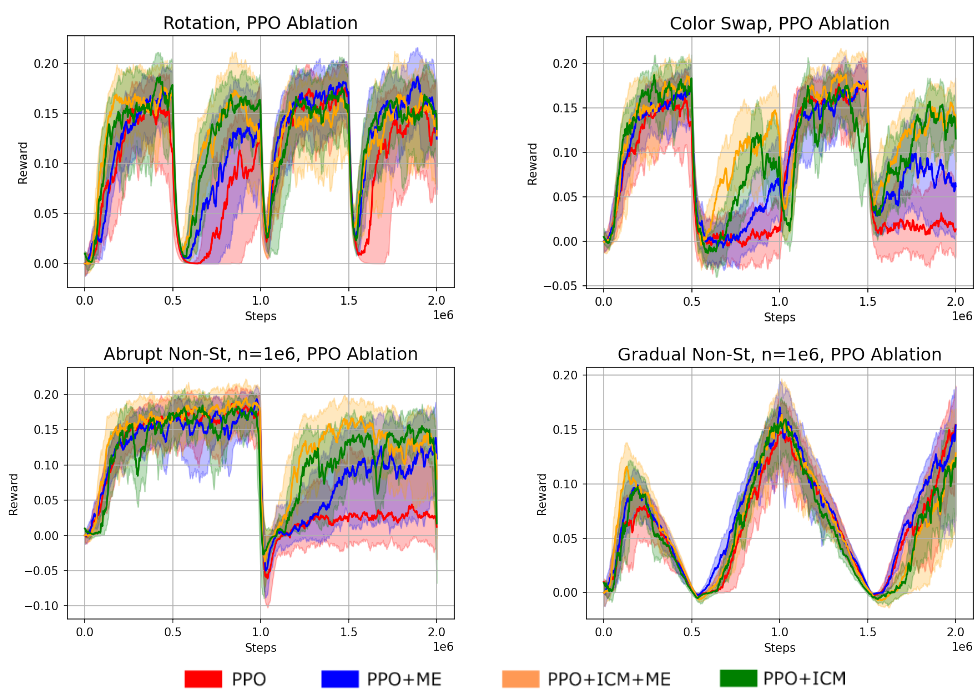


Figure 15: Experiments for non-stationary dynamics (top row) and non-stationary reward functions (bottom row). We compare the effect of PPO+ME and PPO+ICM+ME and to PPO+ICM and the PPO baseline.

Figure 15 shows learning curves for two observation shifts (Rotation, Color-Swap, top row), and two reward shifts (Abrupt $n = 1\text{e}6$, Gradual $n = 1\text{e}6$, bottom row). As in previous experiments, on the gradual reward shift all methods perform on-par. However, on the Color-Swap and abrupt reward shift experiments, we do observe notable differences. We observe significant improvements of PPO+ME over PPO in Color-Swap ($p = 6.59\mathrm{e}-7$), and Rotation ($p = 8.64\mathrm{e}-6$), however not for experiments where a shift in the reward distribution occurs. This effect is expected, since essential information about the task is discarded (i.e. color of item changes) in the case of Color-Swap, and the agent needs to re-learn the task, for which a simple form of exploration is already beneficial. For reward shifts, the agent only needs to update what items to collect after having already learned how to collect them, therefore behaving more randomly does not provide any gains. A more informative form of exploration as for PP0+ICM yields significant gains over PPO ($p = 8.28e-9$), and PPO+max_ent ($p = 1.83\mathrm{e}-8$) for abrupt reward shifts, since the reward model actively steers the policy towards behavior that leads to unexpected reward. Similarly, PPO+ICM significantly outperforms PPO+ME on Color-Swap ($p = 6.62\mathrm{e}-4$), as it enforces exploration of the changed observation space to facilitate re-learning the task at hand. Combining the two different exploration mechanisms further leads to significant improvements in Color-Swap ($p = 4.87\mathrm{e}-2$), however we did not observe the same effect in any other experiment. On Rotation, PPO+ME, PPO+ICM, and PPO+ICM+ME significantly outperform PPO ($p = 8.64\mathrm{e}-6$, $p = 4.44\mathrm{e}-9$, $p = 7.41\mathrm{e}-8$, respectively). In this setup, no significant differences between the different forms of exploration can be observed, indicating that any form of exploration helps. This is due to the fact that Rotation does not discard essential information about the task, since it preserves color and location of items. Our results provide compelling evidence that directed exploration mechanisms are more useful for proper reaction to changing environments.

A.8 Complete Recovery after Domain Shifts

Prior experiments (e.g. Figure 5) may suggest that agents using Reactive Exploration are not capable of recovering to peak performance after a domain shift has occurred. We show an additional experiment setup in which demonstrates that this is not the case and agents are capable to fully recover given sufficient interaction steps with the environment. In Figure 16 we present results for agents learning for 5e6 steps where we apply the Color-Swap shift after 1e6 steps. The performance of agents converges at first, before the domain shift causes a drop in performance. This is followed by recovery and eventually reaching the same performance as before the shift.

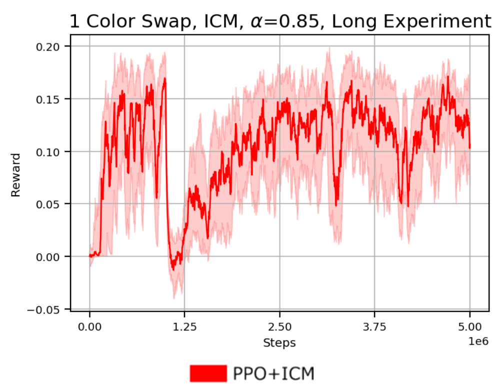


Figure 16: Experiment for a single abrupt shift after 1M steps (Color-Swap) and another 4M steps without further shifts. PPO+ICM manages to recover all the way, i.e., the achievable reward after a shift is not limited.

A.9 Generalization to Non-Stationarity

As a follow-up experiment, we investigate the generalization behavior of the agent after stacking different domain shifts. In this regard, we again consider a simple color swap experiment in which the colors green and white (color of collectible which yields positive reward, and background) are exchanged (GW-Swap). This shift appears every $n = 5\mathrm{e}5$ interaction steps. Figure 17 (left) shows the performance of the stationary versions of PPO and the non-stationary version which encounters GW-Swap. The PPO agent shows signs of forward transfer and is capable of recovering. Surprisingly, when combining GW-Swap with an additional transformation (GW-Swap+Rotation), the PPO agent reaches a higher average reward and recovers more quickly (Figure 17, right). Also, this particular domain shift appears to facilitate more forward transfer than the single GW-Swap. As a reference point, we also provide the results for PPO trained in the stationary regime.

We take this line of experiments a step further and additionally induce a shift in the reward function. This shift occurs after $n = 1\mathrm{e}6$ interaction steps with the environment, and inverts the reward function, as explained in Section 5.2. Figure 18, left, shows the performance for the PPO agent after the shift in the reward function. When combining this distribution shift with GW-Swap+Rotation, we find that the PPO agent successfully recovers after the induced change. It appears that certain types of non-stationarities positively influence the ability of the agent to generalize. We leave a further investigation of this phenomenon to future work.

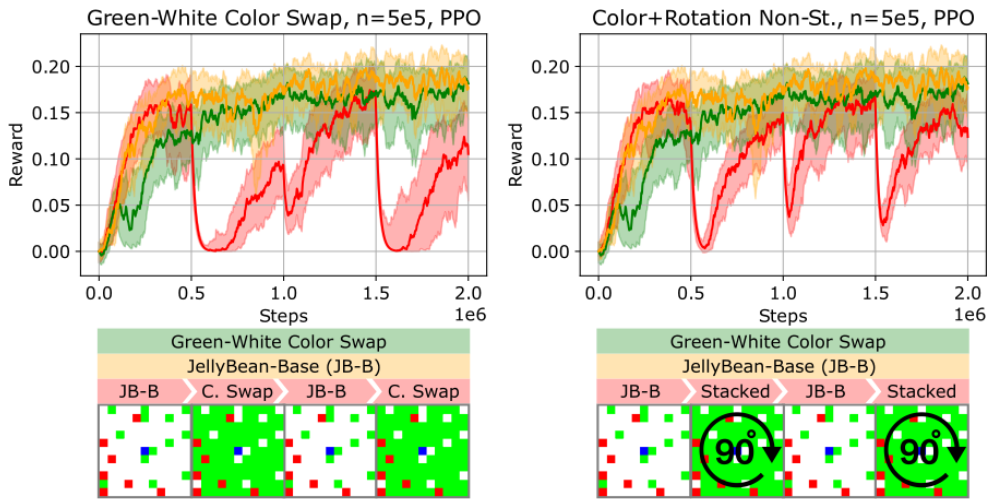


Figure 17: Experiments for non-stationary dynamics via swapping the color of a collectible with the background (GW-Swap, left) versus stacking GW-Swap and rotation (GW-Swap+Rotation, right). PPO agents recover more quickly for the GW-Swap+Rotation. The bottom row visualizes how the non-stationarity occurs. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing across 8 seeds.

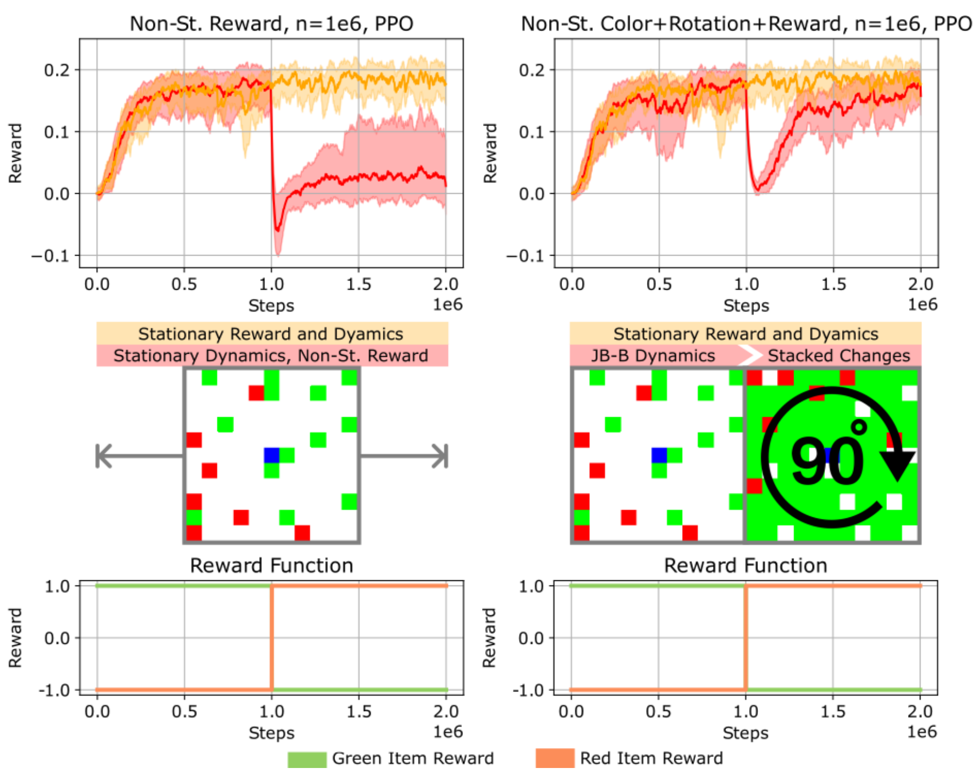


Figure 18: The slow and abrupt non-stationarity experiment from Section 5.2 (left) versus stacking reward shift with GW-Swap+Rotation (right). The bottom row shows the different kinds of non-stationarity. Average rewards every 1000 steps are shown using an IQR and exponential average smoothing across 8 seeds each.

B Experiment Setup

B.1 Hyperparameters

B.1.1 Default DQN and PPO

The hyperparameters used in this work are based on prior work on PPO (Schulman et al., 2017) and DQN (Mnih et al., 2013; 2015). These settings are provided as default in the `Stable-Baselines3` (Raffin et al., 2019) package. We experienced that these hyperparameter choices out of the box resulted in reasonable performance for both algorithms, even in the non-stationary case.

Table 1: PPO Hyperparameters

Hyperparameter	Value
Learning rate	3e-4
Rollout collection steps	2048
Batch size	64
Epochs	10
γ	0.99
GAE λ	0.95
Clip range	0.2
Clip range value function	None
Normalize advantage	True
Entropy coefficient	0
Value function coefficient	0.5
Maximum gradient clipping	0.5
Resample noise matrix	False
Limit on Target KL divergence	None

Table 2: DQN Hyperparameters

Hyperparameter	Value
Learning rate	1e-4
Buffer size	1e6
Learning starts after	5e4
Batch size	32
Soft update coefficient τ	1.0
γ	0.99
Update frequency	4
Gradient steps after rollout	1
Target network update interval	1e4
ϵ exploration fraction	0.1
ϵ initial value	1.0
ϵ final value	0.05
Maximum gradient clipping	10

B.1.2 Mixture of Intrinsic and Extrinsic Rewards

We use hyperparameters α, to weight intrinsic rewards originating from the forward dynamics model, and β to weight intrinsic rewards originating from the reward model. These parameters control the ratio of intrinsic reward, while $1-\alpha-\beta$ yields the ratio of extrinsic reward received by the agent. In addition, it is important to consider the different scales of intrinsic and extrinsic rewards, as the intrinsic rewards generated by the model might be far lower (e.g. 0.01) than the extrinsic rewards of the environment (e.g. 0.15). This discrepancy can be accounted for in the choice of α and β. In certain experiments (i.e. observation shift, Section 5.4) we set $\beta = 0$, as predicting the reward does not provide additional useful information.

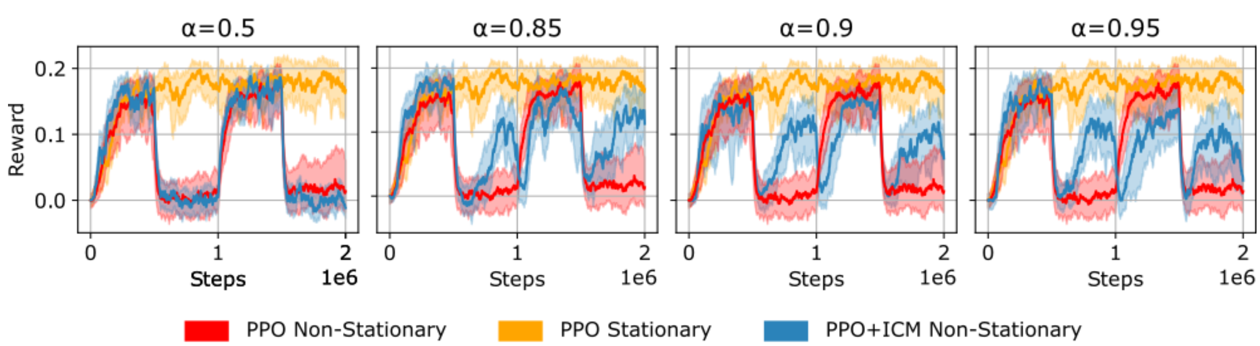


Figure 19: Results for repeating our Color-Swap experiment with added Reactive Exploration. We show different hyperparameter settings for the mixture of intrinsic curiosity reward and extrinsic environment reward over 5 seeds.

In Figure 19 we present four different settings for α. For the first value of $\alpha = 0.5$, the extrinsic reward clearly overshadows the intrinsic reward (roughly 15 times as large), which naturally leads to no Reactive Exploration by the agent, and yields similar results to the default PPO algorithm. For the additionally considered settings of $\alpha = \{0.85, 0.9, 0.95\}$ the agent manages to recover from the domain shifts with almost identical behavior. The ranges for intrinsic and extrinsic rewards are obtained by a preliminary experiment in which we track the range of the prediction error of the forward dynamics/reward model, and the range of the extrinsic rewards. In our particular case, the extrinsic rewards are in an interval of $[0.75; 2.65]$ times as large as intrinsic rewards. Accordingly, we choose the above-mentioned value for α. A simple and alternative way of ensuring a similar range for intrinsic and extrinsic rewards would be to normalize and scale them, leading to the necessity of a different hyperparameter for mixing intrinsic and extrinsic reward.

B.2 Model Architectures

We apply a custom CNN feature extractor which is small enough to support the ($11 \times 11 \times 3$) observation space of JBW and additionally contains a linear layer of size 128. The CNN feature extractor for both, DQN and PPO, is listed in Table 3. For our CartPole experiments, we use the default MLP policy setting in SB3.

Table 3: CNN feature extractor used with Stable-Baselines3 PPO and DQN for JBW.

Layer: Type	Parameters	Connected to
0: Conv2D	in=obs_channels, out=32, kernel_size=(3, 3), stride=2	1: ReLU
1: ReLU		2: Conv2D
2: Conv2D	in=32, out=64, kernel_size=(3, 3), stride=2	3: ReLU
3: ReLU		4: Flatten
4: Flatten		5: Linear
5: Linear	in=n_flattened, out=128	6: ReLU
6: ReLU		

For the Intrinsic Curiosity Module (ICM Pathak et al., 2017), that is, our observation model, we list the architecture for JBW in Table 4, and for CartPole in Table 5. The corresponding reward model architecture for JBW is listed in Table 6, and that for CartPole in Table 7.

Table 4: ICM observation model architecture for JBW consisting of inverse and forward model which share an observation encoder.

Layer: Type	Parameters	Connected to
enc_0: Conv2D	in=obs_channels, out=32, kernel_size=(3, 3), stride=2	enc_1: ReLU
enc_1: ReLU		enc_2: Conv2D
enc_2: Conv2D	in=32, out=64, kernel_size=(3, 3), stride=2	enc_3: ReLU
enc_3: ReLU		enc_4: Flatten
enc_4: Flatten		enc_5: Linear
enc_5: Linear	out=128	enc_6: ReLU
enc_6: ReLU		
forw_0: Concatenate	input action, input observation	forw_1: Linear
forw_1: Linear	action_len + observation_latent_features, hdashline	forw_2: ReLU
forw_2: ReLU		forw_3: Linear
forw_3: Linear	in=128, out=128	forw_4: ReLU
forw_4: ReLU		forw_5: Linear
forw_5: Linear	in=128, out=128	
inv_0: Linear	in=2 · observation_latent_features, out=128	inv_1: ReLU
inv_1: ReLU		inv_2: Linear
inv_2: Linear	in=128, out=128	inv_3: ReLU
inv_3: ReLU		inv_4: Linear
inv_4: Linear	in=128, out=action_len	

Table 5: ICM observation model encoder for CartPole, reusing the forward and inverse models described in Table 4.

Layer: Type	Parameters	Connected to
enc_0: Linear	in=observation_len, out=128	enc_1: ReLU
enc_1: ReLU		enc_2: Conv2D
enc_2: Linear	in=128, out=128	enc_3: ReLU
enc_3: ReLU		enc_4: Flatten
enc_4: Linear	in=128, out=128	enc_5: Linear

Table 6: Reward model architecture for JBW.

Layer: Type	Parameters	Connected to
0: Conv2D	in=obs_channels, out=32, kernel_size=(3, 3), stride=2	1: ReLU
1: ReLU		2: Conv2D
2: Conv2D	in=32, out=64, kernel_size=3, stride=2	3: ReLU
3: ReLU		4: Flatten
4: Flatten		5: Concat
5: Concat	action_len, observation_latent_features	6: Linear
6: Linear	in=action_len + n_flatten, out=128	7: ReLU
7: ReLU		8: Linear
8: Linear	in=128, out=128	9: ReLU
9: ReLU		10: Linear
10: Linear	in=128, out=1	

Table 7: Reward model architecture for CartPole.

Layer: Type	Parameters	Connected to
0: Concat	action, observation	1: Linear
1: Linear	in=action_len+observation_len, out=128	2: ReLU
2: ReLU		3: Linear
3: Linear	in=128, out=128	4: ReLU
4: ReLU		5: Linear
5: Linear	in=128, out=128	6: ReLU
6: ReLU		7: Linear
7: Linear	in=128, out=1	