Recurrent Policies Are Not Enough for Continual Reinforcement Learning

Nathan de Lara
nathan.delara@mail.mcgill.ca
McGill University

Veronica Chelu
veronica.chelu@mail.mcgill.ca
McGill University

Doina Precup
dprecup@cs.mcgill.ca
McGill University, DeepMind

Abstract

Continual Reinforcement Learning (CRL) aims to develop algorithms that adapt to non-stationary sequences of tasks. A promising recent approach utilizes Recurrent Neural Networks (RNNs) to learn contextual Markov Decision Process (MDP) embeddings. This enables a reinforcement learning (RL) agent to discern the optimality of actions across diverse tasks. In this study, we examine two critical failure modes in the learning of these contextual MDP embeddings. Specifically, we find that RNNs are prone to catastrophic forgetting, manifesting in two distinct ways: (i) embedding collapse—where agents initially learn a contextual task structure that later collapses to a single task, and (ii) embedding drift—where learning embeddings for new MDPs interferes with embeddings the RNN outputs for previous MDPs in the sequence, leading to suboptimal performance of downstream policy networks conditioned on stale embeddings. We explore the effects of various objective functions and network architectures concerning these failure modes, revealing that one of these modes consistently emerges across different setups.

1 Introduction

The Continual Reinforcement Learning (CRL) class of solution methods encompasses algorithms which deal with a broad range of non-stationary settings in sequential decision making, such as input distribution shift and covariate shift (Berariu et al., 2021; Hadsell et al., 2020; Rolnick et al., 2019). In the CRL setting an agent faces a sequence of Markov Decision Processes (MDPs) $\mathcal{M}_1, \mathcal{M}_2, \ldots$ and is unable to re-attempt previously faced MDPs, unless they reappear later in the sequence. The traditional Reinforcement Learning (RL) goal of maximizing discounted total reward is extended to the goal of *maximizing the expected cumulative reward, both on future yet unseen tasks in the sequence, and on previously faced tasks.* Oftentimes performance on previously faced tasks deteriorates as the sequence progresses, this phenomenon is known as catastrophic forgetting (McCloskey and Cohen, 1989; Nguyen et al., 2019; Cahill, 2011; Chen and Liu, 2018).

The CRL setting can be modeled through the formalism of Partially Observable Markov Decision Processes (POMDPs) (Kaelbling et al., 1998) (see Appendix B for a definition), where the hidden state is a task label that remains constant throughout an episode (Khetarpal et al., 2022; Doshi-Velez and Konidaris, 2016; Choi et al., 1999; Xie et al., 2021). This formulation connects CRL to multi-task RL, which has utilized POMDP frameworks and recurrent networks to optimize MDPs by integrating RNN outputs with observations (Xie et al., 2021; Rakelly et al., 2019; Sodhani et al., 2021). Using Recurrent Neural Networks (RNNs) to generate contextual information for traditional RL algorithms has led to strong agents in several multi-task RL studies (Ni et al., 2022; Zintgraf et al., 2019; Rakelly et al., 2019; Nagabandi et al., 2018). Notable works like Xie et al. (2021) and Caccia et al. (2023) adopt this POMDP perspective and apply recurrent networks to CRL settings.

In this paper, we continue with this line of investigations on applying recurrent networks for learning contextual embeddings in CRL. In particular, we highlight the challenges of applying recurrent policies to CRL focusing particularly on MDP sequences with the following properties: (i) First, they

are non-backtracking, a sequence of MDPs is non-backtracking if it can be divided into segments where all MDPs within a segment are identical, and MDPs in different segments are distinct; (ii) The second property that defines our setting is that the MDPs are unidentifiable from any single transition, but, identifiable given a finite history of transitions. In this setting, one cannot act optimally given only the current state since it lacks enough information to identify the MDP and thus the optimal action. This means applying state-of-the art knowledge distillation, parameter regularization, or experience replay methods to actor-critic (AC) agents will not work since there is not enough information in a single state to optimize every MDP. This setting is different than the ones considered by Xie et al. (2021) as they experiment with sequences where the speed of non-stationarity is so high that the agent sees every unique MDP in the non-stationary sequence at least every 20 timesteps. Our setting is also different than Caccia et al. (2023) as the MDPs they consider are generally identifiable from single states.

Specifically, our work is concerned with exploring the use of dynamics and optimality signals for training Recurrent Neural Networks that generate learning embeddings which are additionally concatenated to the standard input of AC networks. We show that while recurrent models greatly enhance the adaptivity and plasticity of AC RL agents, across all the variations considered, two failure modes consistently arise. The first failure mode is embedding collapse, where initial learning creates a clear task structure, followed by a convergence of all embeddings to a single point. The second failure mode resembles classical notions of catastrophic forgetting: the RNN adjusts embeddings for all tasks when encountering a new task, leading to input drift for the policies conditioned on those embeddings corresponding to previously seen tasks. To mitigate these failure modes we explore combining parameter regularization and knowledge distillation methods with recurrent policies.

2 Problem Setting

We adopt the standard CRL setting where an agent faces a sequence of MDPs $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \ldots$. We set the contiguous segments of the non-backtracking sequence such that they are of equal size and long enough for agents to learn optimal policies for each task individually. This allows us to evaluate the adaptivity and stability of the agent across the MDP sequence. All MDPs in the sequence have the same state space, action space, and transition functions, but different reward functions. We model the MDPs as being members of a subclass of POMDPs which have constant hidden states (MDP labels) throughout an episode but varying hidden states between episodes. This is analogous to the Hidden-Parameter MDP (Doshi-Velez and Konidaris, 2016; Fu et al., 2024; Perez et al., 2020; Killian et al., 2017).

3 Recurrent Agents for Continual Reinforcement Learning

We evaluate a general approach using recurrent networks to produce embedding vectors, which are concatenated with observations to generate belief states. These belief states become the inputs to downstream dynamics models, policy networks, and deep Q-networks. Using a vanilla replay buffer causes old tasks to be over-sampled. So, to ensure that the agent is trained on recent tasks we employ a recency buffer which over-samples from recent transitions, the effect of the buffer is shown in Figure 2c (details in Appendix C). We investigate the impact of different architectural variations on performance and embedding disentanglement. Diagrams showing the different compositions of learned models can be found in Appendix C.

Learning Task Embeddings with Recurrent Models The goal is to infer a rich task embedding, b_i, for task $\mathcal{M}_i$. We use a standard isotropic multivariate Gaussian distribution. For a specific episode trajectory $\tau_{1:T}$, we partition it at the K-th time step ($K < T$). We train a conditional task embedding model $p_\psi(\tau_{1:K})$ with a Gated Recurrent Unit (GRU) (Cho et al., 2014) to process $\tau_{1:K}$, followed by a Multilayer Perceptron (MLP) to process the GRU's final hidden state, and separate layers for mean and variance parameters. All components are trained to minimize the loss functions of downstream networks on the entire trajectory $\tau_{1:T}$.

Learning Actor and Critic Networks Following Rakelly et al. (2019); Xie et al. (2021) we extend the Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018) to use a policy and value function conditioned on task embeddings, states, and actions. We consider a policy $\pi_\theta(a_t|s_t, g_\psi(\{(s_l, a_l, r_l)\}_{l=1}^{t-1}))$,

parameterized with parameters θ, receiving the input from a recurrent network g_ψ, with parameter vector ψ defined over a sequence of transitions and returning a vector of fixed size representing the encoded embedding (More details in Appendix E).

Learning Dynamics Networks For All Tasks We also make use of learned dynamics models: a transition model q_ϕ and reward model q_ω, parametrized with parameter vectors ϕ and ω, respectively. To learn the dynamics we use neural networks outputting means and variances of isotropic Gaussian distributions. The distributions give log-likelihoods for every observed transition in our dataset. The dynamics models are trained by minimizing the negative log-likelihood of the agent's observed rewards and next states given task embeddings, states, and actions as conditions. Currently, dynamic networks are used to learn embeddings. Resolving catastrophic forgetting in recurrent networks would enable their use for model-based policy optimization (Janner et al., 2019).

4 Experiments

Benchmarks To evaluate the different approaches and task embeddings, we use two benchmarks. The first is a 2D point-reaching environment where an agent must reach a non-backtracking goal point on the boundary of a disc. The second is a non-stationary variant of the Gymnasium Reacher environment (Towers et al., 2023), featuring a 2-DOF arm that must move its fingertip to a non-stationary goal location. In our variant, the goal location is occluded and moves to create a non-backtracking sequence of MDPs. Hyper-parameters for both environments are detailed in Appendix G.

Models: Here, we introduce our methods for generating task embeddings. In each recurrent agent, the task embeddings are concatenated with the state and sent to actor-critic networks trained using the Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018).

- **3RL:** Introduced by Caccia et al. (2023). It uses one RNN for each critic network and one RNN for the policy network. The networks are trained end to end using the SAC algorithm.
- **Recurrent Dynamics (RD):** Uses a single RNN whose output is sent to reward and transition networks. The RNN is trained as a VAE, minimizing the negative log likelihood of the dynamics models' outputs and a KL-divergence with a normal prior.
- **Separate Recurrent Dynamics (SRD):** Uses one RNN for the reward network and another for the transition network. The RNNs are trained to minimize the negative log likelihood of their respective models and a KL-divergence with a normal prior. When samples are taken uniformly from the dataset we denote the algorithm as USRD.
- **SAC:** Introduced by Haarnoja et al. (2018), serves as a baseline for a non-recurrent agent

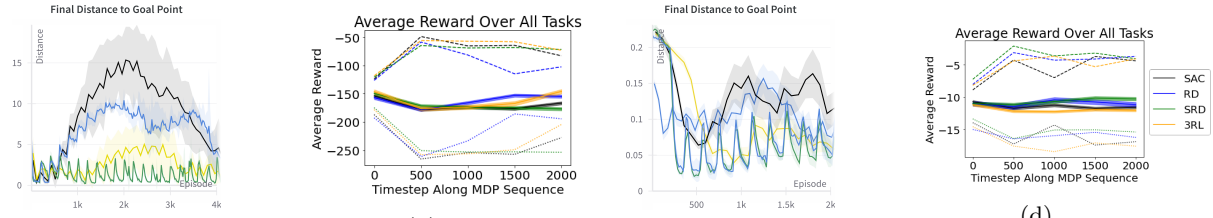
5 Results

5.1 Adaptivity and General Performance

In Figure 1a and 1c, we depict the distance to the goal at the conclusion of each episode across both benchmarks and all models examined. Notably, SAC's performance deteriorates as the sequence unfolds, only improving at the end when the goal point nears the original location. However, introducing recurrent networks enables the agent to consistently refine its performance on the ongoing task. In Figure 1d and 1b we plot how the average reward of the agent averaged over all MDPs in the sequence changes as the agent progresses through the task sequence. We observe that while the agents are able to optimize at least one MDP at every point (from the top dashed line), this comes at the cost of forgetting other MDPs as the performance over all MDPs stays constant. This suggests that recurrence improves adaptability but still suffers from catastrophic forgetting.

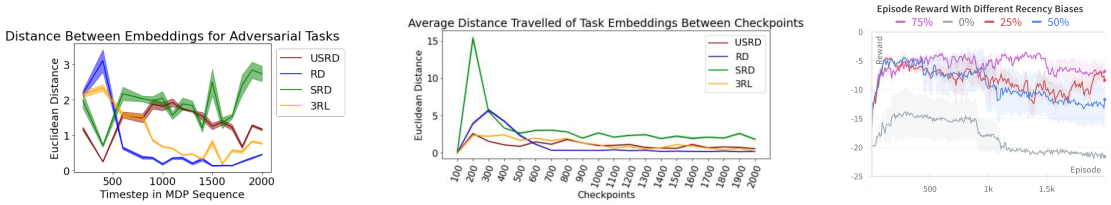
5.2 Failure Mode #1: Embedding Collapse

In Figure 2a, we show how the distance between task embeddings for "adversarial" tasks changes as different agents progress through the MDP sequence. Adversarial tasks are defined as those with goal points farthest apart relative to all tasks in the sequence. We observe that all models see this distance go to zero in the 2D point reacher environment. Furthermore, we see 3RL and RD having this distance go to zero in Figure 2a implying that these models suffer from embedding collapse.



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

Figure 1: Performance across all models, including final distance to goal point (1a,1c) and averaged reward on all MDPs (1b, 1d), in 2D Point Reacher and Non-stationary MuJoCo Reacher respectively. Dotted and dashed lines represent the max and min performance of each model over tasks.



(a) (b) (c)

Figure 2: Distance between embeddings for adversarial tasks (2a) and distance between embedding for same task between checkpoints (2b) in the Non-stationary MuJoCo Reacher benchmark. Figure 2c shows how changing the % of batch samples from a recency buffer impacts performance.

Previous research on representation collapse in reinforcement learning and contrastive learning finds that significant data augmentation or implicit regularization can induce embedding collapse (Kumar et al. (2021), Jing et al. (2021)). Our results extend these findings to the CRL setting, as optimal agents for two different tasks may have almost zero overlap in their state visitation distribution, which can be viewed as data augmentation, while the recency buffer can be seen as implicit regularization. Removing the recency buffer may alleviate collapse, but it has disastrous effects on the agent's experienced reward, as shown in Figure 2c.

5.3 Failure Mode #2: Embedding Drift

When scaling up to the Non-stationary MuJoCo Reacher, we observe a new failure mode causing catastrophic forgetting. We hypothesize this occurs when gradient updates to the recurrent network adjust task embeddings for all tasks, moving them away from the inputs that downstream dynamics networks were previously optimized on. Figure 2b visualizes the Euclidean distances between task embeddings every 200 episodes. The consistently above-zero change in embeddings for SRD implies the encoder is continuously adjusting embeddings. Similar reasons for failure have been studied before in parameter regularization methods like EWC (Kirkpatrick et al., 2017) showing promising future directions using a combination of these methods.

6 Discussion & Future Directions

This paper investigates the application of recurrent networks for learning contextual embeddings in Continual Reinforcement Learning (CRL), specifically in non-backtracking sequences of Markov Decision Processes (MDPs) which cannot be identified from single transitions. Our findings demonstrate that while recurrent models significantly enhance the adaptivity and plasticity of reinforcement learning agents, they also introduce two persistent failure modes: embedding collapse and continual forgetting. These issues underscore the complexities involved in maintaining performance across evolving tasks within CRL settings. Future research should focus on integrating parameter regularization and knowledge distillation methods with recurrent policies to mitigate these failure modes. Additionally, exploring hybrid architectures that combine the strengths of different learning signals could further enhance the robustness and effectiveness of CRL agents in dynamic environments.

References

Tudor Berariu, Wojciech Czarnecki, Soham De, Jorg Bornschein, Samuel Smith, Razvan Pascanu, and Claudia Clopath. A study on the plasticity of neural networks. *arXiv preprint arXiv:2106.00042*, 2021.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8. URL https://www.sciencedirect.com/science/article/pii/S0079742108605368.

Cuong V. Nguyen, Alessandro Achille, Michael Lam, Tal Hassner, Vijay Mahadevan, and Stefano Soatto. Toward understanding catastrophic forgetting in continual learning. *CoRR*, abs/1908.01091, 2019. URL http://arxiv.org/abs/1908.01091.

Andy Cahill. *Catastrophic forgetting in reinforcement-learning environments*. PhD thesis, University of Otago, 2011.

Zhiyuan Chen and Bing Liu. *Continual Learning and Catastrophic Forgetting*, pages 55–75. Springer International Publishing, Cham, 2018. ISBN 978-3-031-01581-6. doi: 10.1007/978-3-031-01581-6_4. URL https://doi.org/10.1007/978-3-031-01581-6_4.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semi-parametric regression approach for discovering latent task parametrizations. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 1432–1440. AAAI Press, 2016. ISBN 9781577357704.

Samuel Choi, Dit-Yan Yeung, and Nevin Zhang. Hidden-mode markov decision processes. 12 1999.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11393–11403. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/xie21c.html.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9767–9779. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/sodhani21a.html.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16691–16723. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/ni22a.html.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2018.

Massimo Caccia, Jonas Mueller, Taesup Kim, Laurent Charlin, and Rasool Fakoor. Task-agnostic continual reinforcement learning: Gaining insights and overcoming challenges, 2023.

Haotian Fu, Shangqun Yu, Michael Littman, and George Konidaris. Model-based lifelong reinforcement learning with bayesian exploration. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9781713871088.

Christian Perez, Felipe Petroski Such, and Theofanis Karaletsos. Generalized hidden parameter mdps: Transferable model-based rl in a handful of trials. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5403–5411, 2020.

Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. *Advances in neural information processing systems*, 30, 2017.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179.

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

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *CoRR*, abs/1906.08253, 2019. URL http://arxiv.org/abs/1906.08253.

Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. Gymnasium, March 2023. URL https://zenodo.org/record/8127025.

Aviral Kumar, Rishabh Agarwal, Tengyu Ma, Aaron Courville, George Tucker, and Sergey Levine. Dr3: Value-based deep reinforcement learning requires explicit regularization. In *Deep RL Workshop NeurIPS 2021*, 2021.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *International Conference on Learning Representations*, 2021.

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.

Martin L. Puterman. *Markov Decision Processes*. Wiley, 1994. ISBN 978-0471727828. URL http://books.google.com/books/about/Markov_decision_processes.html?id=Y-gmAQAAIAAJ.

Yash Chandak, Geo rgios Theocharous, Shiv Shankar, Martha White, Sridhar Mahadevan, and Philip S. Thomas. Optimizing for the future in non-stationary mdps. *CoRR*, abs/2005.08158, 2020. URL https://arxiv.org/abs/2005.08158.

Appendices

A MDP Definition

Markov Decision Processes (MDP) A discrete-time infinite-horizon discounted MDP ((Puterman, 1994)) is a 6-tuple $\{\mathcal{S}, \mathcal{A}, T, R, P, \gamma\}$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition function T, reward function R, and discount factor $\gamma \in [0, 1)$.

B POMDP Definition

Partially Observable MDP (POMDP) A POMDP is an extension of MDPs (MDP definition in Appendix A) to include hidden state components (Kaelbling et al., 1998). It includes a hidden state that influences transitions and rewards. In a POMDP the full state has an observable and hidden component which both evolve as the MDP progresses.

C Model Layouts

The non-stationary setting makes for an unbalanced input distribution, such that more recent observations are more relevant to future tasks. For all models we include a recency buffer which contains the last 500 transitions, 500 was chosen as it empirically led to the best results. We simultaneously sample from both the recency buffer and general buffer according to a chosen balance β. Figure 2c shows the impact changing β has on the reward RD attains as it progresses through the MDP sequence.

In all the following graphics solid lines indicate the flow of information while dashed lines indicate how losses backpropagate between learned models.

C.1 Layout of RD Model

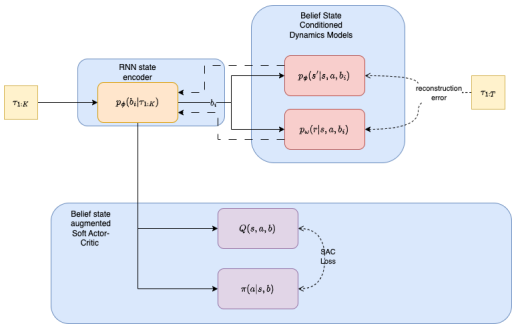


Figure 3: Layout of our RD Model

C.2 Layout of SRD Model

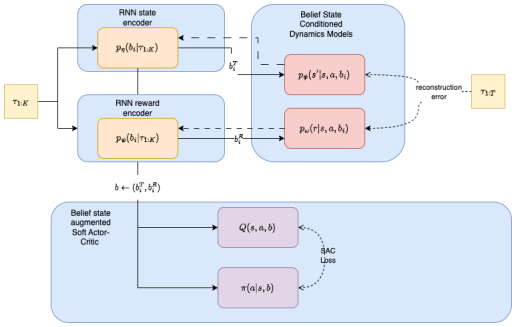


Figure 4: Layout of our SRD Model

C.3 Layout of 3RL

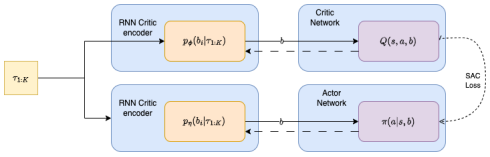


Figure 5: Layout of our 3RL Model

D Related Work

Belief State Based Approaches to Reinforcement Learning Prior methods have use variational inference to learn posteriors over belied states given the current trajectory. A recurrent network learns to take a trajectory through the BAMDP and returns a posterior belief distribution over the current MDP, $p(b|\tau_{1:t})$. The policy is then conditioned on a sample from this posterior, and both the policy and recurrent network are trained to maximize within-task performance conditioned on the sampled belief. Xie et al. (2021) attempts to apply this approach to continual learning, assuming that the task embeddings evolve according to some stationary distribution over episodes. In their experiments the rate of non-stationarity is such that after only 20 episodes all MDPs in the non-stationary sequence have been seen at least once and will be seen again with that same frequency. As we are focused on the ability of these models to extrapolate to unseen tasks their findings are peripheral to the findings we present.

Continual Reinforcement Learning Task embeddings represented via recurrent networks have been previously explored in CRL by Caccia et al. (2023), demonstrating superior performance of the task-conditional policies even in comparison to policies conditioned on the ground truth task, for certain task sequences. The authors show that the emerging task structure acquires semantic meaning. Importantly, the authors do not learn dynamics networks alongside the policy. Chandak et al. (2020) assume the task sequence changes smoothly or impose more stringent assumptions on the task non-stationarity process. When it comes to accounting for how the task will change in the next coming episodes of the sequence Pro-WLS and LiLAC Xie et al. (2021) are two existing approaches proposed under varying assumptions different than our case of study. Pro-WLS Chandak et al. (2020) assumes that the loss of a policy with respect to its objective is smooth over time and implements an importance sampling approach for approximating future gradients by weighting the gradients of previous episodes. Both Pro-WLS and LiLAC impose strong assumptions on the sequence of tasks that agents face that show prospective optimization to be possible in more restricted cases than ours.

E Mathematical Formalization of Recurrent Policies

We define a recurrent policy as a set of parameters θ, ψ which parameterize an RNN $g_\psi : (\mathcal{S} \times \mathcal{A} \times \mathbb{R})^T \to \mathbb{R}^{d_z}$ and a policy network $\pi_\theta : \mathcal{S} \times \mathbb{R}^{d_z} \times \mathcal{A} \to [0, 1]$ such that π_θ defines a conditional probability measure over $\mathcal{A}$ when conditioned on a state $s \in \mathcal{S}$ and embedding $z \in \mathbb{R}^{d_z}$ and d_z is a pre-defined dimension size for the learned task embeddings. Given this definition for the two models that make our recurrent policy we give an action probability for an action a_t, conditioned on state s_t and previous history $\{s_k, a_k, r_k\}_{k=1}^{t-1}$ by: i) passing $\{s_k, a_k, r_k\}_{k=1}^{t-1}$ to g_ψ to get embedding vector z_t, ii) then passing s_t, and z_t to π_θ to get the mean μ and covariance matrix Σ for an isotropic multivariate Gaussian distribution, iii) finally we calculate the log likelihood of a_t under the distribution parameterized by μ, Σ. We can write the above evaluation and composition of models as $\pi_\theta(a_t|s_t, g_\psi(\{s_k, a_k, r_k\}_{k=1}^{t-1}))$.

F Minimization Functions

F.1 Actor Loss

$$\mathcal{J}_\pi = \mathbb{E}_{\substack{s,a,r,s',\tau\sim\mathcal{D} \\ z\sim p_\psi(\tau)}}[KL(\pi(a|s,z)||\frac{exp(Q(s,a,z)}{Z(s)})],$$

F.2 Critic Loss

$$\mathcal{J}_Q = \mathbb{E}_{\substack{s,a,r,s',\tau\sim\mathcal{D} \\ z\sim p_\psi(\tau) \\ a'\sim\pi(a'|s',z)}}[(Q(s,a,z) - r - \bar{Q}(s',a',z))^2]$$

G Benchmark Hyper-parameters

G.1 2D Point Reacher

The 2D Point Reacher benchmark consists of an agent and a goal point configured as 2D points existing on a disk of fixed radius α. The agent can move 1 unit in any direction on the disk with actions $a = (a_1, a_2)$ that correspond directly to adding a_1 to the first dimension and a_2 to the second dimension of the agents location. We place the goal point on the boundary of the disc. In our experiment we set the disk to have radius 10 and rotate it by $\theta = \pi/2000$ degrees every 100 episodes. If an agent chooses an action that takes it outside the disk we re-normalize it's new position to have a norm of size at most α.

G.2 Non-Stationary MuJoCo Reacher

The Reacher environment (Towers et al., 2023) is a 2-DOF arm planted on the middle of a plane that must move its fingertip to a specific goal location. We show a picture of the Reacher robot and environment with 4 sample goal points in Figure 6. To create the Non-stationary MuJoCo Reacher benchmark we occlude the goal point and move it by a constant degree $\theta = \pi/1000$ around the object every 100 episodes for a total of 2000 episodes. This choice of θ and rotation speed means that the goal returns to the starting position at the end of the MDP sequence. For the starting state of the robot, the 2 degrees of freedom correspond to the angle of the base of the arm makes and the second degree is the angle between the two straight components of the arm. we initialize the agent to be in the same configuration for every episode setting the first degree of freedom to be 0 and second degree of freedom to be π causing the fingertip to be in the center.

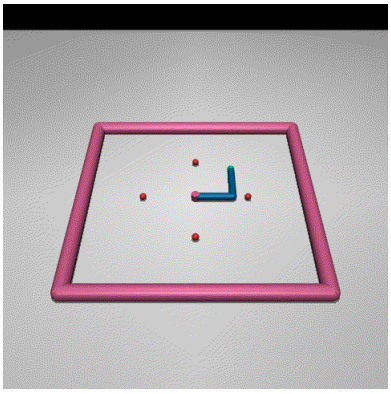


Figure 6: Visualization of the Reacher arm with 4 sample goal points around it