

Effective offline training and efficient online adaptation

Philip J. Ball

Balliol College
University of Oxford

*A thesis submitted for the degree of
Doctor of Philosophy*

Michaelmas 2023

Abstract

Developing agents that behave intelligently in the world is an open challenge in machine learning. Desiderata for such agents are efficient exploration, maximizing long term utility, and the ability to effectively leverage prior data to solve new tasks. Reinforcement learning (RL) is an approach that is predicated on learning by directly interacting with an environment through trial-and-error, and presents a way for us to train and deploy such agents. Moreover, combining RL with powerful neural network function approximators – a sub-field known as “deep RL” – has shown evidence towards achieving this goal. For instance, deep RL has yielded agents that can play Go at superhuman levels, improve the efficiency of microchip designs, and learn complex novel strategies for controlling nuclear fusion reactions.

A key issue that stands in the way of deploying deep RL is poor sample-efficiency. Concretely, while it is possible to train effective agents using deep RL, the key successes have largely been in environments where we have access to large amounts of online interaction, often through the use of simulators. However, in many real-world problems, we are confronted with scenarios where samples are expensive to obtain. As has been alluded to, one way to alleviate this issue is through accessing some prior data, often termed “offline data”, which can accelerate how quickly we learn such agents, such as leveraging exploratory data to prevent redundant deployments, or using human-expert data to quickly guide agents towards promising behaviors and beyond. However, the best way to incorporate this data into existing deep RL algorithms is not straightforward; naïvely pre-training using RL algorithms on this offline data, a paradigm called “offline RL”¹ as a starting point for subsequent learning is often detrimental. Moreover, it is unclear how to explicitly derive useful behaviors online that are positively influenced by this offline pre-training.

With these factors in mind, this thesis follows a 3-pronged strategy towards improving sample-efficiency in deep RL. First, we investigate effective pre-training on offline data. Then, we tackle the online problem, looking at efficient adaptation to environments when operating purely online. Finally, we conclude with hybrid strategies that use offline data to explicitly augment policies when acting online.

¹Under this nomenclature, Standard RL is therefore called “online RL”.

Effective offline training and efficient online adaptation



Philip J. Ball
Balliol College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2023

Acknowledgements

I would first like to express my gratitude to my supervisor Steve, who has been a source of immense support, and has always encouraged and assisted the decisions and directions that were taken throughout the thesis. Without his careful guidance, my DPhil experience would not have been any near as fulfilling as it has been.

I would also like to graciously thank Willowgrove for sponsoring my DPhil. The freedom afforded to me has been truly unique and appreciated, and without their assistance, of course none of this would have been possible.

I'd also like to shout out Jack, who joined MLRG at the same time as me, and through our shared interest in football (and sometimes machine learning) managed to provide some of the most exciting academic work undertaken during the DPhil. I appreciate his endless enthusiasm and belief in me, which has been far greater than my belief in myself at times, and has encouraged and helped me in making my DPhil journey significantly more enjoyable and complete.

On a similar note, I doff my cap to Cong, who has been an immense part of the DPhil. His incredible talent at getting experiments to work and fantastic attitude has been hugely appreciated; I hope we continue to collaborate well into the future, perhaps with saner working hours.

I'd also like to highlight Aldo, whose rigor and talent with theory has been a constant source of inspiration, and has also supported me at critical junctures during the DPhil. I hope that one day I am able to appropriately return the favor.

I'd also like to thank Sergey and the RAIL Group at Berkeley. It was an incredible pleasure getting to know all of you, and thank you for making me feel so welcome so quickly. I'd like to especially highlight my brilliant collaborators Laura and Ilya, from whom I learned an incredible amount about RL, as well as wisdom that goes well beyond it.

Outside of work, I'd like to collectively thank the incredibly talented people in the Oxford University music scene, who have made this return leg to Oxford a significantly richer experience. In particular, my band mates in Green Bean Machine have provided some of the fondest memories during my DPhil. Seeing them develop both as musicians and as people has filled me with an immense sense of pride and inspiration; I wish them all the greatest luck in their future endeavors, music or otherwise.

Finally, I would like to thank my ever loving and ever patient parents. Despite the number of career changes I have taken up to now, they have never wavered in their trust and belief in the decisions I have made, and have supported me above and beyond anything that I could have ever wished for. I hope they can finally relax going forward!

Abstract

Developing agents that behave intelligently in the world is an open challenge in machine learning. Desiderata for such agents are efficient exploration, maximizing long term utility, and the ability to effectively leverage prior data to solve new tasks. Reinforcement learning (RL) is an approach that is predicated on learning by directly interacting with an environment through trial-and-error, and presents a way for us to train and deploy such agents. Moreover, combining RL with powerful neural network function approximators – a sub-field known as “deep RL” – has shown evidence towards achieving this goal. For instance, deep RL has yielded agents that can play Go at superhuman levels, improve the efficiency of microchip designs, and learn complex novel strategies for controlling nuclear fusion reactions.

A key issue that stands in the way of deploying deep RL is poor sample-efficiency. Concretely, while it is possible to train effective agents using deep RL, the key successes have largely been in environments where we have access to large amounts of online interaction, often through the use of simulators. However, in many real-world problems, we are confronted with scenarios where samples are expensive to obtain. As has been alluded to, one way to alleviate this issue is through accessing some prior data, often termed “offline data”, which can accelerate how quickly we learn such agents, such as leveraging exploratory data to prevent redundant deployments, or using human-expert data to quickly guide agents towards promising behaviors and beyond. However, the best way to incorporate this data into existing deep RL algorithms is not straightforward; naïvely pre-training using RL algorithms on this offline data, a paradigm called “offline RL”² as a starting point for subsequent learning is often detrimental. Moreover, it is unclear how to explicitly derive useful behaviors online that are positively influenced by this offline pre-training.

With these factors in mind, this thesis follows a 3-pronged strategy towards improving sample-efficiency in deep RL. First, we investigate effective pre-training on offline data. Then, we tackle the online problem, looking at efficient adaptation to environments when operating purely online. Finally, we conclude with hybrid strategies that use offline data to explicitly augment policies when acting online.

²Under this nomenclature, Standard RL is therefore called “online RL”.

Contents

List of Figures	xi
1 Introduction	1
1.1 Contributions	3
1.1.1 Effective Pre-Training with Offline Data	3
1.1.2 Efficient Online Adaptation to Environments	3
1.1.3 Offline Data Strategies to Augment Online Policy Deployment	4
1.2 Thesis Structure	5
Chapter Bibliography	7
2 Preliminaries	11
2.1 Markov Decision Processes	11
2.1.1 Partially Observable Markov Decision Processes	12
2.2 Online Reinforcement Learning	13
2.2.1 Policy Gradients	14
2.2.2 Approximate Dynamic Programming	15
2.2.3 Model-Based Reinforcement Learning	17
2.3 Offline Reinforcement Learning	18
2.3.1 Deployment-Efficient Reinforcement Learning	19
Chapter Bibliography	20
3 Effective Pre-Training with Offline Data	25
3.1 Revisiting Design Choices in Offline Model-Based RL	26
3.1.1 Introduction	26
3.1.2 Related Work	29
3.1.3 Background	31
3.1.4 Uncertainty Penalty	32
3.1.5 Key Hyperparameters in Offline MBRL	35
3.1.6 Testing the Limits of Current Approaches	39
3.1.7 Conclusion	43
3.1.8 Errata	44
Chapter Bibliography	46

4	Efficient Online Adaption to Environments	51
4.1	Ready Policy One: World Building Through Active Learning	52
4.1.1	Introduction	52
4.1.2	Related Work	54
4.1.3	Background	57
4.1.4	Ready Policy One	61
4.1.5	Experiments	67
4.1.6	Conclusion and Future Work	71
4.1.7	Errata	72
4.2	Stabilizing Off-Policy Deep Reinforcement Learning from Pixels . .	74
4.2.1	Introduction	74
4.2.2	Background	77
4.2.3	Instabilities in TD-Learning from Pixels	78
4.2.4	Counteracting Gradient Discontinuities	86
4.2.5	Performance Evaluation	90
4.2.6	Related Work	94
4.2.7	Conclusion	95
	Chapter Bibliography	97
5	Offline Data Strategies to Augment Online Policy Deployment	107
5.1	Augmented World Models Facilitate Zero-Shot Dynamics Generaliza- tion From a Single Offline Environment	108
5.1.1	Introduction	109
5.1.2	Related Work	111
5.1.3	Preliminaries	113
5.1.4	Augmented World Models with Self Supervised Policy Adap- tation	115
5.1.5	Experiments	118
5.1.6	Conclusion and Future Work	126
5.1.7	Errata	127
5.2	Learning General World Models in a Handful of Reward-Free De- ployments	129
5.2.1	Introduction	129
5.2.2	Problem Statement	132
5.2.3	Coordinated Active Sample Collection	134
5.2.4	Theoretical Motivation	137
5.2.5	Experiments	138
5.2.6	Exploring Worlds	139
5.2.7	Related Work	145

5.2.8	Conclusion	147
5.3	Efficient Online Reinforcement Learning with Offline Data	150
5.3.1	Introduction	150
5.3.2	Related work	153
5.3.3	Preliminaries	154
5.3.4	Online RL with Offline Data	155
5.3.5	Experiments	161
5.3.6	Conclusion	169
5.3.7	Errata	170
	Chapter Bibliography	172
6	Conclusion	191
6.1	Future Work	193
	Chapter Bibliography	197
	Appendices	203
A	Revisiting Design Choices in Offline Model- Based RL: Appendix	205
B	Ready Policy One: Appendix	237
C	Stabilizing Off-Policy Deep Reinforcement Learning from Pixels: Appendix	241
D	Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment: Appendix	265
E	Learning General World Models in a Handful of Reward-Free Deployments: Appendix	269
F	Efficient Online Reinforcement Learning with Offline Data: Appendix	287

List of Figures

3.1	a) The variation of different uncertainty penalties against <i>true</i> dynamics error during a model rollout of Hopper Medium-Expert. The canonical ensemble variance penalty most closely fits the true dynamics error. b) Tuning key hyperparameters (an approach we call Optimized) can lead to large gains over state-of-the-art methods (MOPO) on the D4RL benchmark, as we show in this summary using <code>rliable (?)</code>	27
3.2	Box Plots showing D4RL Medium transferred to Random. We show IQR limits and the median value denoted by the black vertical line. Green = HalfCheetah , Blue = Hopper . Max Aleatoric, Max Pairw. Diff. and LOO KL are unstable w.r.t. ensemble member count. In contrast, ensemble variance and std. are far more stable.	36
3.3	Plot of how error and penalty correlation changes with model number in Hopper across all datasets (i.e., Random through to Expert).	36
3.4	MOPO performance on the Hopper medium-expert environment.	43
4.1	Median performance across 10 seeds. Shaded regions correspond to the Inter-Quartile Range.	68
4.2	Mean one-step policy improvement after a given λ value, for all ten seeds of RP1.	69
4.3	Median performance across 10 seeds. Shaded regions correspond to the Inter-Quartile Range.	70
4.4	Performance of agents in DMC (left) and Atari 100k (right) benchmarks from 10 seeds. A-LIX outperforms previous methods without using image augmentations or auxiliary losses.	75
4.5	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The vertical dashed line shows when augmentations are turned off.	78
4.6	Evidence of overfitting when augmentations are not used. On the left , shaded lines are individual estimates, the solid line represents the median Q-value. On the right , the Q-values Pearson correlation with target values and Monte-Carlo returns (R_{MC}).	79
4.7	TD-loss of offline fixed transitions during training, separated based on having non-zero reward.	80

4.8	Feature maps in the final layer of both augmented (top) and non-augmented (bottom) agent encoders. Non-augmented agents manifest inconsistent, high-frequency feature maps.	83
4.9	Critic loss surface plots of augmented (left) and non-augmented (right) agents after 5,000 steps of offline training.	84
4.10	Instantaneous (red and blue) and accumulated (orange and purple) ND scores for the features gradients from offline (left) and online (right) training in Cheetah Run.	86
4.11	A-LIX's S parameter evolution during training in Cheetah Run (left) and Quadruped Run (right). As the critic targets become more informative, S falls, improving data efficiency and asymptotic performance.	90
4.12	Average performance in 10 seeds for DMC Medium (left) and Hard tasks (right). Shaded regions represent ± 1 SE.	91
4.13	<i>Probability of improvement</i> and <i>performance profiles</i> obtained from the recorded results in DMC (left) and Atari 100k (right). A-LIX displays statistically significant improvements and stochastically dominates most prior algorithms.	93
5.1	An illustration of our approach, each figure shows a transition $P(s, a) \rightarrow s'$. In a) we show the World Model dynamics $\hat{P}$, trained from $\mathcal{D}_{\text{env}}$, a single offline dataset. In b) we show the Augmented World Model, blue represents $\hat{P}$, while each green agent illustrates an instantiation of augmented dynamics, which is sampled at each timestep: $\hat{P}_z, z \sim \mathcal{Z}$. The goal is to approximate c) where we show an unseen test environment, with transition dynamics P^*	110
5.2	Training policies in Augmented World Models. For each state-action pair sampled from the buffer, a new augmentation $z_t \sim \mathcal{Z}$ is sampled to produce an augmentation operator $\mathcal{T}_z$, which is applied to the transition. The policy is then trained with the new tuple of data, with the context concatenated to the state.	115
5.3	Self-supervised policy adaptation via learned context. At each timestep, the state s_t is fed into a linear model (trained online at each timestep) to predict the change in state δ_t , and also passed to the fixed World Model (trained on the offline data) to predict the change in state under $\hat{P}$, $\hat{\delta}_t$. The approximate context is then $z_t = \delta_t / \hat{\delta}_t$, which is concatenated with s_t and passed to the agent to produce an action.	115
5.4	Mean R^2 of the linear model across 20 rollouts.	118
5.5	Mean performance for 5 seeds for MOPO on HalfCheetah environments with varying dynamics. Note the central cell (1,1) corresponds to the in-sample data.	119

5.6	Mean performance for 3 seeds for MBPO with a dynamics oracle on HalfCheetah with $1.5\times$ mass and damping.	120
5.7	Average performance gains of different World Model augmentations over base MOPO for different levels of damping and mass in the HalfCheetah test environment (5 seeds).	121
5.8	Average performance gains of adding contexts over base MOPO for different levels of damping and mass in the HalfCheetah test environment (5 seeds).	122
5.9	Performance under changing dynamics. Top = cumulative returns, Bottom = rolling average single step reward. Both averaged over twenty seeds, shaded area shows $\pm 1\text{std}$	124
5.10	Mean improvement for Augmented World Models over MOPO for the HalfCheetah environment, averaged over five seeds. Top row (blue) = damping scale, bottom row (green) = mass scale. Dotted line is the mean, the same value for both damping and mass.	125
5.11	Mean improvement for Augmented World Models over MOPO for the Walker2d environment, averaged over five seeds. Top row (blue) = damping scale, bottom row (green) = mass scale. Dotted line is the mean, the same value for both damping and mass.	125
5.12	Comparison between MOPO and AugWM on the crippled leg Ant environment. The crippled leg is highlighted in red. The MOPO agent flips over while the AugWM agent is able to adapt to changed dynamics and successfully run. MOPO on the regular Ant environment is provided as a reference.	125
5.13	Overview. Left: CASCADE trains a population of diverse explorers and uses them to collect large batches of reward-free trajectories for learning a general world model that facilitates zero-generalization to novel tasks. Right: To train B exploration agents in parallel, at each training step t , CASCADE first infers a latent state s_t from image observation o_t . It then rolls out latent trajectories $\tau^1, \dots, \tau^B$ in imagination using the current exploration policies $\pi^1, \dots, \pi^B$. The training objective for each policy π^i is to optimize 1) the population diversity estimated by the disagreement of the final states of imagined trajectories sampled from policies $\pi^1, \dots, \pi^i$; 2) the expected per state information gain over all future timesteps $t + 1, \dots, T$, computed as the disagreement of an ensemble of dynamics models.	131
5.14	Motivation for CASCADE: Green areas represent high expected information gain. If we train a population of agents independently, at deployment time they will all follow the trajectory to #1, producing a homogenous dataset. However, if we consider the diversity of the data then we will produce agents that reach both #1 and #2.	131

- 5.15 **MiniGrid Results.** Left: Performance statistics for both **FourRooms** and **MultiRoom**. For reward-free exploration we show both “state coverage”, which corresponds to the the percentage of states visited on a fixed set of levels, and “rewarding episodes” which is the count of total solved episodes found after 1M steps. Finally for zero-shot transfer we show the success rate of a task-specific behavior policy trained with labels provided only at test time. All plots show the mean of ten seeds with SEM shaded. (Random achieves 0 zero-shot success rate). Right: we show the state coverage on a fixed level in **FourRooms**, using the exploration policy π_{EXP} for each deployment (Test time goal state indicated by the red dot). . . . 140
- 5.16 **Atari Zero-Shot Transfer:** Plots show RLiabile metric performance aggregated across all seeds. Note that the ‘Probability of Improvement’ lower CIs all exceed 0.5 for Atari, indicating **CASCADE** provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (?). We show five seeds of zero-shot test performance. 141
- 5.17 **Crafter Score:** Plots show final RLiabile metric performance and training curve over 1M steps aggregated across all seeds. We show ten seeds of Crafter Score, the geomean of skills discovered in a purely reward-free fashion. Again, see that the ‘Probability of Improvement’ lower CIs all exceed 0.5, indicating **CASCADE** provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (?). 142
- 5.18 **Crafter Skills:** Plots show normalized success rate per skill. Note that **CASCADE** not only achieves the highest average success rate but also unlocks the most (16 out of 22) unique skills among all baselines. Skills that are not unlocked by any of the baselines are excluded from the plot. 142
- 5.19 **DMC Qualitative Results:** Each row visualizes behaviors of a population of explorers trained after 15 total deployments by the same world model, deployed at the same time in environments with the same initial state. We can see that **CASCADE** explorers (top) collect data with a diversity of behaviors while **PP2E** agents (bottom) only show less interesting homogenous behaviors. 143
- 5.20 **DMC Zero-Shot Aggregate Results:** Plots show RLiabile metric performance aggregated across all three initial datasets (**random**, **medium**, **expert**) for a total of 30 seeds. Note that the ‘Probability of Improvement’ lower CIs all exceed 0.5, indicating **CASCADE** provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (?), even after a single deployment. 144

5.21	Our approach, RLPD, extends standard off-policy RL and achieves reliable state-of-the-art online performance on a number of tasks using offline data. Here we show the difficult D4RL AntMaze domain (10 seeds, 1 std. shaded), averaged over all 6 tasks. We run RLPD for 300k steps due to early convergence.	151
5.22	Using SAC with our symmetric sampling method can result in instabilities due to diverging Q-values; with LayerNorm in the critic this disappears, improving performance.	156
5.23	We fit data (left) with a two-layer MLP without LayerNorm (center) and with LayerNorm (right). LayerNorm bounds the values and prevents catastrophic overestimation.	158
5.24	RLPD exceeds prior state-of-the-art performance on a number of different popular benchmarks whilst being significantly simpler. Results are aggregated over 21 different environments (10 Seeds, 1 std. shaded). In each case, we compare to the prior best known work (IQL + Finetuning in Adroit and AntMaze, Off2On in Locomotion), and SACfD, a canonical off-policy approach using offline data. . . .	161
5.25	Our approach generalizes to vision-based domains, providing consistent improvements over existing approaches.	165
5.26	Increasing UTD with RLPD greatly improves sample efficiency from pixels.	165
5.27	LayerNorm is crucial for strong performance, particularly when data are limited or narrowly distributed.	166
5.28	Our recommended starting design choices and workflow leads to strong performance on all tasks.	167
5.29	In general, critic ensembling provides the best performance. Dropout performs worse in sparse reward tasks.	167
5.30	Symmetric sampling improves sample efficiency and reduces variance across seeds, and does not work by simply increasing the reward density in a batch.	168
5.31	Initializing the buffer with large amounts of data limits improvement.	168
5.32	RLPD is not sensitive to replay proportion; 50% offers the best compromise between variance, speed of convergence, and asymptotic performance.	169
A.1	Scatter Plots showing HalfCheetah D4RL transfer tasks.	207
A.2	Scatter Plots showing Hopper D4RL transfer tasks.	208
A.3	Scatter Plots showing HalfCheetah D4RL true model-based error calibration.	209
A.4	Scatter Plots showing Hopper D4RL true model-based error calibration.	210
A.5	Box Plots showing HalfCheetah D4RL transfer tasks.	212

A.6	Box Plots showing Hopper D4RL transfer tasks.	213
A.7	Boxplots showing HalfCheetah D4RL true model-based error penalty distributions.	214
A.8	Boxplots showing Hopper D4RL true model-based error penalty distributions.	215
A.9	HalfCheetah Spearman Statistics	216
A.10	HalfCheetah Pearson Statistics	217
A.11	Hopper Spearman Statistics	218
A.12	Hopper Pearson Statistics	219
A.13	HalfCheetah Spearman Statistics	220
A.14	HalfCheetah Pearson Statistics	220
A.15	Hopper Spearman Statistics	220
A.16	Hopper Pearson Statistics	220
A.17	Aggregated True Model-Based correlation statistics over all datasets (i.e., Random through to Expert); Left: HalfCheetah; Right: Hopper	221
A.18	HalfCheetah Transfer.	222
A.19	Hopper Transfer.	223
A.20	HalfCheetah True Model-Based.	224
A.21	Hopper True Model-Based.	224
A.22	Median True Model-Based Errors as a function of rollout timestep . . .	226
A.23	Several Individual Ground Truth Rollouts in Hopper Medium-Expert . .	226
A.24	Comparing OOD dynamics and inputs on a Hopper Medium-Expert trajectory	227
A.25	t-SNE projection of 10,000 Hopper D4RL Medium-Replay and 10,000 exploitative Imagined policy state-action tuples	228
A.26	Hopper Medium-Expert True Model-Based Experiments; Left: Precision v.s. Recall against Ground Truth; Middle: Higher Performing Penalties v.s. Ground Truth MSE in Imagined Rollout; Right: Lower Performing Penalties v.s. Ground Truth MSE in Imagined Rollout	228
A.27	Precision Recall curves on ground truth data.	230
B.1	Ablation study where we consider removing the early stopping mechanism. All results show the median performance across ten seeds.	237
B.2	Ablation study where we consider removing the adaptive mechanism. All results show the median performance across ten seeds.	237
C.1	Performance profiles at 50% (left) and 100% (right) of the total steps in Medium and Hard DMC Tasks.	243
C.2	Medium and Hard DMC Task plots	243

C.3	Probability of Improvement statistics at both 50% (left) and 100% (right) of the total timesteps in Medium and Hard DMC Tasks.	244
C.4	Performance profiles with linear (left) and logarithmic (right) scaling in Atari 100k.	245
C.5	Bootstrapped ranking statistics (left) and probability of improvement plots (right) on Atari 100k.	246
C.6	Q values and Pearson Correlation of the offline Proprioceptive agent on an offline fixed batch.	248
C.7	Pearson Correlation of augmented and non-augmented online agents in Cheetah Run and Quadruped Walk across 5 seeds. Shaded lines represent individual runs, and solid lines represent the median. We see that augmented agents do not immediately overfit to their target networks, and become correlated only after useful signal is learned.	249
C.8	Performance of agents across 4 different seeds of the Cheetah Run environment and their adaptive scalar parameter S . We observe that initially, S is high until agents learn useful behaviors, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.	250
C.9	Performance of agents across 4 different seeds of the Quadruped Run environment and their adaptive scalar parameter S . We observe that as meaningful behaviors are learned in agents towards the end of training, S falls accordingly, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.	250
C.10	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.	251
C.11	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.	255
C.12	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.	256
C.13	An ablation study of A-LIX, showing the contribution of its individual components to ultimate performance in DMC and Atari 100k.	256
C.14	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dotted horizontal line represents mean expert performance.	258
C.15	Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dashed line shows when augmentations are turned off.	259
C.16	Action-Value loss surface plotted with respect two orthogonal random directions sampled from the action space (i.e., $d_r \in A$ and $d_1 \perp d_2$).	260

C.17	Effect of checkerboard artifacts on feature maps and resultant loss sensitivity. We see the non-augmented agent is significantly more sensitive to this high-frequency noise.	261
C.18	Loss surface plotted with respect to <i>Encoder</i> parameters at various stages of training.	261
C.19	Loss surface plotted with respect to <i>Critic MLP</i> parameters at various stages of training.	262
D.1	Performance for Augmented World Models with the DAS augmentation. Each plot shows different values for a and b , the ranges for the sampled noise.	266
E.1	DMC Zero-Shot Results: Plots show the performance by environment and dataset, with the last row being the task performance average over all three initial datasets. Plots show the mean and SEM over 10 seeds. .	270
E.2	From left to right on each row we show a frame from the game, a plot of the cumulative maximum episode reward, the total number of rewarding episodes discovered and zero-shot average return, from 3M training steps (15 deployments). Plots show mean and SEM over 10 seeds. Note that the max episode returns of all methods stabilize after 15 deployments (3M training steps in total).	271
E.3	Trajectories in Montezuma’s Revenge: Each row shows a trajectory from one of the 10 exploration agents of a model after 3M training timesteps (P2E is omitted because it only has 1 agent.) We can see that PP2E’s agents exhibit the most homogeneous behaviors, and result in trajectories that focus on the bottom right of the room, while CASCADE agents manage to explore more of the room collectively.	272
F.1	Full Adroit Results.	287
F.2	Full AntMaze results	288
F.3	SAC with and without Layer Normalization.	289
F.4	RLPD v.s. SAC + LN + Offline Data.	289
F.5	RLPD v.s. IQL + Finetuning. We show offline pre-training steps for IQL using negative indices.	290
F.6	D4RL Ablations.	290
F.7	D4RL Ablation on Expert Data.	291
F.8	VD4RL Ablations.	291
F.9	Visualizations of the environments we consider.	292

1

Introduction

Contents

1.1 Contributions	3
1.1.1 Effective Pre-Training with Offline Data	3
1.1.2 Efficient Online Adaptation to Environments	3
1.1.3 Offline Data Strategies to Augment Online Policy Deployment	4
1.2 Thesis Structure	5
Chapter Bibliography	7

One of the grand challenges in machine learning is developing the capability to deploy autonomous agents that can learn to perform complex tasks from simply interacting with the environment (Stanley et al., 2017). Such an agent could be readily trained with little to no external supervision, and learn novel ways to perform difficult tasks more effectively and efficiently than humans. Having access to agents with these broad capabilities could help us tackle a number of societal challenges, such as those around safety (Wegman, 2017; Schwall et al., 2020), resourcing (Haddad et al., 2022) and discovering solutions to important open-problems (Shukla et al., 2019).

Reinforcement Learning (RL) (Sutton & Barto, 2018) has been presented as a general approach to learning such agents (Silver et al., 2021). Concretely, RL

aims to learn optimal agent behavior through interacting with an environment, and achieves this by maximizing its long term utility, which it learns through trial-and-error. To further improve the ability of such agents to reason in complex environments, as well as operate from a variety of modalities, such as images, neural network function approximators are incorporated into their architecture, resulting in a sub-field referred to as “deep RL” (Tesauro et al., 1995; Mnih et al., 2015). Such agents have shown promising results that suggest the aforementioned goals of learning general capabilities with minimal supervision may be possible, such as playing Go and complex 3D video games at superhuman levels (Silver et al., 2017; Wurman et al., 2022), designing significantly more efficient microchips (Roy et al., 2021; Mirhoseini et al., 2021), and learning complex novel strategies to control nuclear fusion reactions (Degraeve et al., 2022).

However much work is still required to realize these agents across a broader set of tasks, and therefore make them generally applicable to all real-world problems. A key issue that stands in the way of deploying deep RL is that agents often require a significant number of environment interactions before they learn useful behaviors (Botvinick et al., 2019). This is disastrous for many realistic scenarios where such interactions can be costly, such as where human intervention is required to reset an unsuccessful trial. To alleviate this, and following trends in deep learning, we can provide agents with “prior” datasets (also called offline data (Levine et al., 2020)) that contain previous interactions with the environment which, depending on the nature of this data, may help address issues with finding optimal behavior using expert demonstrations (Vecerik et al., 2017), or can front-load the cost of potentially expensive exploration (Agarwal et al., 2020; Laskin et al., 2023). However, offline data is not a canonical part of the standard RL formulation, and its effective incorporation remains an open question. Indeed, naïve approaches that simply pre-train agents on this data and then continue fine-tuning them (an approach common to other branches of machine learning (Devlin et al., 2019; Radford et al., 2021)) can be in fact less sample-efficient than approaches that do not pre-train at all (Nakamoto et al., 2023).

In this thesis we propose tackling the important issue of sample-efficiency in deep RL by breaking the problem into 3 separate parts.

1.1 Contributions

1.1.1 Effective Pre-Training with Offline Data

In Chapter 3 we investigate the important factors in effectively pretraining policies using purely offline data (a branch of RL called “offline RL”). In particular, in Section 3.1 we focus on model-based approaches, and seek to discover the key design choices that enable agents to most effectively utilize offline data when learning intelligent behaviors. We deliver a number of insights into previously neglected design choices, and use these to train agents that perform over 40% better than previous approaches without modifying any underlying algorithms while only using a single set of hyper-parameters. The result of this is recommendations that practitioners can readily use to improve how effectively agents pre-train on the growing number of datasets we have access to.

1.1.2 Efficient Online Adaptation to Environments

In Chapter 4 we investigate efficient adaptation to environments when operating purely online without offline data.

First, in Section 4.1 we describe an agent that can effectively adapt its exploration according to feedback from its own world model by leveraging methods derived from active learning. This results in a novel information theoretic perspective on exploration in RL, and we use this to derive agents that are significantly more sample-efficient than canonical model-based approaches, and moreover we derive insights and approaches that are generally applicable to all model-based methods.

In Section 4.2 we then seek to understand and ameliorate the fundamental issues of training off-policy deep RL approaches on visual domains. Concretely, combining TD-learning with visual domains results in significantly worse performance than their non-visual counterparts, largely due to agents failing to adapt effectively to incoming reward signals and falling into local minima. To this end, simple image

augmentations have been proposed and shown to be effective in preventing this, but paradoxically these image augmentations are not in-distribution with respect to the actual observations the agent sees in the environment. We therefore seek to understand the true mechanism behind why image augmentations help agents to learn and adapt to new data, and subsequently propose a new “visual deadly triad” that exists when using TD-learning with convolutional neural networks (CNNs). Using this insight, we propose a new adaptive network layer that obviates the need for out-of-distribution image augmentations, thus greatly improving performance over canonical approaches while having significantly fewer design choices.

1.1.3 Offline Data Strategies to Augment Online Policy Deployment

In Chapter 5 we investigate open problems when combining offline datasets with methods that require further online deployment, be it either continued learning of optimality, or robustness to unseen changes.

First, in Section 5.1 we investigate scenarios whereby covariate shift occurs between the provided offline dataset and the actual online environment. Concretely, we consider scenarios where the dynamics function changes between offline pre-training and online deployment, which may happen for instance if we experience a sudden mechanical failure of an agent, or we encounter unforeseen weather conditions. We find that existing offline RL approaches are particularly vulnerable to this type of covariate shift, and will incur significant regret compared to an approach that had direct access to the changed dynamics. In order to improve agent robustness to dynamics changes at test time, we introduce a novel data augmentation and training methodology that exploits a learned world model. We then train a meta-policy that can additionally condition on an inferred “dynamics latent variable” to further take advantage of certain dynamics changes, such as a lighter and less dampened agent, in particular under non-stationary dynamics.

Second, in Section 5.2 we ask whether it is possible to improve offline policy learning inside a world model that enables the training of generalist agents. Con-

cretely, we seek to learn a world model in an unsupervised RL setting that is able to train agents that can zero-shot generalize to any downstream task when provided with an appropriate reward function. In order to do this, we propose learning and deploying a large population of agents that can maximally cover an environment to collect as complete a dataset as possible. We propose a novel diversity based behavioral metric that encourages a large population of diverse agents to be learned offline, and then repeat this process to derive a maximally useful world model for any unseen downstream task.

Finally, in Section 5.3 we seek to answer why off-policy approaches cannot effectively utilize offline datasets in deep RL settings, despite the off-policy formulation supporting the natural incorporation of this data. We find that key differences exist between online agents and naïve methods that use offline RL and online fine-tuning, and introduce a novel perspective on the LayerNorm layer as having desirable properties for value learning. This, in combination with several other key design choices, allows us to convert any off-policy deep RL algorithm into a remarkably efficient approach that can naturally incorporate any offline dataset, from those containing a limited number of expert demonstrations in sparse reward domains, to those having large amounts of sub-optimal trajectories in dense reward settings.

1.2 Thesis Structure

In this thesis, we first present preliminary material in Chapter 2 that arms the reader with the key concepts and context required to understand the proceeding material.

In Chapters 3 to 5 we then present the work completed over the duration of this thesis, which, as outlined in the introduction, is separated into three categories. In accordance with the integrated nature of this thesis, the following works are facsimiles of the original transcripts, with the only deviations being formatting changes in order to fit the thesis. We include the appendices for these in the works separately at the end of the thesis. Finally, in Chapter 6 we provide a concluding discussion, and explore opportunities that may merit further research. To further improve the flow of the thesis, separate bibliographies are included for each chapter.

Papers Presented In “Effective Pre-Training with Offline Data”, we present the following work¹:

- Cong Lu*, **Philip J. Ball***, Jack Parker-Holder, Michael A. Osborne and Stephen J. Roberts. Revisiting Design Choices in Offline Model-Based RL. In *International Conference on Learning Representations 2022 (Spotlight)*.

In “Efficient Online Adaptation to Environments”, we present the following work:

- **Philip J. Ball***, Jack Parker-Holder*, Aldo Pacchiano, Krzysztof Choromanski and Stephen J. Roberts. Ready Policy One: World Building Through Active Learning. In *International Conference on Machine Learning 2020*.
- Edoardo Cedin*, **Philip J. Ball***, Stephen J. Roberts and Oya Celiktutan. Stabilizing Off-Policy Deep Reinforcement Learning from Pixels. In *International Conference on Machine Learning 2022*.

Finally, in “Offline Data Strategies to Augment Online Policy Deployment” we present the following work:

- **Philip J. Ball***, Cong Lu*, Jack Parker-Holder, Stephen Roberts. Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment. In *International Conference on Machine Learning 2021*.
- Yingchen Xu*, Jack Parker-Holder*, Aldo Pacchiano*, **Philip J. Ball***, Oleh Rybkin, Stephen J. Roberts, Tim Rocktäschel and Edward Grefenstette. Learning General World Models in a Handful of Reward-Free Deployments. In *Conference on Neural Information Processing Systems 2022*.
- **Philip J. Ball***, Laura Smith*, Ilya Kostrikov* and Sergey Levine. Efficient Online Reinforcement Learning with Offline Data. In *International Conference on Machine Learning 2023*.

¹An asterisk (*) after an author name denotes equal contribution.

Chapter Bibliography

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pp. 104–114. PMLR, 2020.

Matthew Botvinick, Sam Ritter, Jane X Wang, Zeb Kurth-Nelson, Charles Blundell, and Demis Hassabis. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 23(5):408–422, 2019.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL <https://aclanthology.org/N19-1423>.

Lisa M. Haddad, Pavan Annamaraju, and Tammy J. Toney-Butler. *Nursing Shortage*. StatPearls Publishing, Treasure Island (FL), 2022. URL <http://europepmc.org/books/NBK493175>.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, maxime gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=hy0a5MMPUv>.

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

Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim M. Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, Jiwoo Pak, Andy Tong, Kavya Srinivasa, Will Hang, Emre Tuncer, Quoc V. Le, James Laudon, Richard Ho, Roger Carpenter, and Jeff Dean. A graph placement methodology for fast chip design. *Nature*, 594:207 – 212, 2021. URL <https://api.semanticscholar.org/CorpusID:235395490>.

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.

- Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. *arXiv preprint arXiv:2303.05479*, 2023.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.
- Rajarshi Roy, Jonathan Raiman, Neel Kant, Ilyas Elkin, Robert Kirby, Michael Siu, Stuart Oberman, Saad Godil, and Bryan Catanzaro. PrefixRL: Optimization of parallel prefix circuits using deep reinforcement learning. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 853–858. IEEE Press, 2021. doi: 10.1109/DAC18074.2021.9586094. URL <https://doi.org/10.1109/DAC18074.2021.9586094>.
- Matthew Schwall, Tom Daniel, Trent Victor, Francesca Favaro, and Henning Hohnhold. Waymo public road safety performance data. *arXiv preprint arXiv:2011.00038*, 2020.
- Priyadarshi R Shukla, Jim Skea, E Calvo Buendia, Valérie Masson-Delmotte, Hans Otto Pörtner, DC Roberts, Panmao Zhai, Raphael Slade, Sarah Connors, Renée Van Diemen, et al. IPCC, 2019: Climate change and land: an IPCC special report on climate change, desertification, land degradation, sustainable land management, food security, and greenhouse gas fluxes in terrestrial ecosystems. 2019.
- David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. *nature*, 550(7676):354–359, 2017.
- David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artificial Intelligence*, 299:103535, 2021.
- Kenneth O Stanley, Joel Lehman, and Lisa Soros. Open-endedness: The last grand challenge you’ve never heard of. *While open-endedness could be a force for discovering intelligence, it could also be a component of AI itself*, 2017.
- Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Gerald Tesauro et al. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68, 1995.
- Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.
- Fred Wegman. The future of road safety: A worldwide perspective. *IATSS research*, 40(2):66–71, 2017.

Peter R Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, et al. Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022.

2

Preliminaries

Contents

2.1 Markov Decision Processes	11
2.1.1 Partially Observable Markov Decision Processes	12
2.2 Online Reinforcement Learning	13
2.2.1 Policy Gradients	14
2.2.2 Approximate Dynamic Programming	15
2.2.3 Model-Based Reinforcement Learning	17
2.3 Offline Reinforcement Learning	18
2.3.1 Deployment-Efficient Reinforcement Learning	19
Chapter Bibliography	20

2.1 Markov Decision Processes

In this thesis, we consider problems that can be formulated as Markov Decision Processes (MDPs) (Bellman, 1957). An MDP is defined as a tuple $M = (\mathcal{S}, \mathcal{A}, R, P, \rho_0, \gamma)$, where we denote the state space and action space as $\mathcal{S}$ and $\mathcal{A}$ respectively, the transition dynamics as $P(s'|s, a) = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a]$, the reward function as $R(s, a) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a]$, the initial state distribution as $S_0 \sim \rho_0$, and the discount factor as $\gamma \in [0, 1]$ which trades off how much we favor future outcomes over immediate benefits (e.g., $\gamma = 0$ only looks at immediate rewards). This general formalism can be applied to many real-world problems

(White III & White, 1989), and also allows us to develop approaches towards solving sequential decision making problems. To see this, we assume that the MDP describes a discrete time process, such that at a given time step t , we can take an action $a_t \in \mathcal{A}$ while residing in state $s_t \in \mathcal{S}$, and receive reward $r_{t+1} = R(s_t, a_t)$, arriving at a state $s_{t+1} \sim P(s'|s_t, a_t)$ and so on. As we see, our degree of freedom to affect the world lies in the *decisions* on how to take actions a_t . The natural question is then, how do we decide what actions to take? To consider *solving* an MDP (and therefore general decision making processes), we define a “discounted return”, which is described by the following equation: $G_t = \sum_{k=0}^T \gamma^k R_{t+k+1}$ where T is the (potentially infinite) horizon. To evaluate this, we must act in the MDP according to a policy which maps $\mathcal{S} \rightarrow \mathcal{A}$ such that $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$. We can then take expectations over the distributions induced by the reward function, transition dynamics and reward function to get the expected discounted return under that policy, or value function $v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$. To solve an MDP, we wish to maximize the expected discounted return starting from an initial state by finding an optimal policy π^* , i.e., $\pi^* = \arg \max_\pi v_\pi(s_0)$.

Finally we note that a key attribute of MDPs is their being “Markovian”, such that actions and states observed in the past (i.e., the history) do not affect the next state probability distribution (i.e., the future): $\mathbb{P}[S_{t+1}|S_t, A_t] = \mathbb{P}[S_{t+1}|S_t, A_t, S_{t-1}, A_{t-1}, \dots]$. The practical consequence of this is that to take the optimal action a_t^* , the agent only needs access to the current state s_t rather than all the states, actions and rewards preceding s_t , significantly reducing the complexity of the modeling problem.

2.1.1 Partially Observable Markov Decision Processes

In some domains, such as visual domains considered in Section 4.2, or those with hidden attributes in Section 5.1, the assumptions on history independence may not hold. For instance, consider an humanoid agent operating from pixel inputs that is running. If the agent is mid-stride, a single image of this agent is not sufficient to describe the optimal action, as we do not know the velocity of the

agent. Concretely, the action we would take given the agent is in fact falling downwards is very different to if it was moving upwards because it had just pushed off the ground, despite potentially looking the same from a single image. In other words, the problem is no longer Markovian, as we now require the history of how we transitioned into that state.

To resolve this issue, it is possible to frame such a problem as a Partially Observable Markov Decision Process, or POMDP (Åström, 1965). This formulation adds elements into the original MDP to account for the incomplete information in the observations from the environment. A POMDP is defined as a tuple $M = (\mathcal{S}, \mathcal{A}, \mathcal{O}, R, P, \mathcal{Z}, \rho_0, \gamma)$, with deviations from the original MDP in red. To understand how these new elements now reduce the problem back into one that is Markovian, we see that instead of observing states $s \in \mathcal{S}$, we now observe observations $o \in \mathcal{O}$, which may not have sufficient information to fully represent s . As a corollary, the *observation function* $\mathcal{Z}$ is able to map from the Markovian (but unobserved) state space to observation space, i.e., $Z(o|s, a) = \mathbb{P}[O_{t+1} = o | S_{t+1} = s, A_t = a]$. Given we do not have access to the state space, but do have access to observations, rewards and actions, we must instead attempt to model the state space using a *belief* state. A belief state describes a distribution over the true Markovian state space $\mathcal{S}$ given a history of observations, rewards and actions, i.e., $H_t = A_0, O_1, R_1, \dots, A_{t-1}, O_t, R_t$ and the belief state is $b(h) = (\mathbb{P}[S_t = s^1 | H_t = h], \dots, \mathbb{P}[S_t = s^n | H_t = h])$. Consequently, while the observation state is non-Markovian, the belief state *is* Markovian. Therefore we learn to take actions based on the belief state $b(h)$, rather than observations o .

2.2 Online Reinforcement Learning

At this point we have described MDPs, and note that we can solve them by finding an optimal policy; in this section we describe methods to learn such policies.

2.2.1 Policy Gradients

In online RL, a common approach to learning the optimal policy is using policy gradient methods (Sutton et al., 1999), which we utilize in Section 4.1 and Section 5.2. Policy gradients represent the most direct way of optimizing the objective function first discussed in Section 2.1. Recall, we can write the objective as $J(\pi) = v_\pi(s_0)$. Typically we parameterize the policy using weights θ (e.g., the weights of a neural network), so we usually write the objective directly in terms of these weights: $J(\theta) = v_{\pi_\theta}(s_0)$. The question now becomes, how can we find the weights θ that maximize this objective function? The typical approach involves principles from optimization, which involves not directly optimizing the objective $J(\theta)$ (which is usually intractable due to the fact that the quantities associated with agent and MDP are non-convex and have non-analytical forms), but instead finding a maximum by performing gradient ascent (Robbins, 1951). Therefore the question now becomes, how can we evaluate the *gradient* $\nabla_\theta J(\theta)$? At this point we introduce the *action-value function* $q_\pi(s, a) = \mathbb{E}_{\pi, P}[R(s, a) + \gamma v_\pi(s')]$, which allows us to express values with respect to states *and actions*. Following the compact proof in Sutton & Barto (2018), the gradient of interest can be written as follows:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(s) q_{\pi_\theta}(s, a)]. \quad (2.1)$$

Without changing optimality, the term $q_{\pi_\theta}(\cdot, \cdot)$ is usually replaced with an “advantage” function $A_{\pi_\theta}(\cdot, \cdot)$, which helps to reduce the variance associated with this quantity through subtracting a correlated but action-independent function, which is normally chosen to be v_{π_θ} : $A_{\pi_\theta}(s, a) = q_{\pi_\theta}(s, a) - v_{\pi_\theta}(s)$. To recover the modern policy gradient methods used in this thesis, the advantage is estimated and learned using function approximation rather than pure Monte-Carlo samples in order to reduce variance and improve speed of convergence (Schulman et al., 2015; Mnih et al., 2016).

To provide an intuition on how Equation (2.1) allows us to discover optimal behaviors despite not requiring the MDP to be differentiable, consider the different terms inside the expectation. The first term, $\nabla_\theta \log \pi_\theta(s)$, describes a vector that can

be added to the parameters θ to encourage an action $a = \pi_\theta(s)$. In order to determine how much we should encourage action a , we look at the second term, $q_{\pi_\theta}(s, a)$ (or $A_{\pi_\theta}(\cdot, \cdot)$). If we determine that an action will provide a larger return (or advantage), we will consequently nudge θ in that direction, and thus a larger weight will be ascribed to that set of parameter updates. Hence, performing a step of gradient ascent using Equation (2.1) will result in “good” actions (as evaluated under $q_{\pi_\theta}(s, a)$) being more likely to be sampled in the subsequent iteration of the policy (e.g., $\pi_{\theta_{n+1}}$).

2.2.2 Approximate Dynamic Programming

In all other sections, we utilize approximate dynamic programming approaches. These approaches rely on iterating the recursive relationships in learned values present in the *Bellman equations*, and then exploiting *Bellman optimality* to derive the optimal policy (in contrast to policy gradients in 2.2.1, which directly attempt to learn the optimal policy without explicit reference to value learning).

To unpack these terms, we first address the Bellman equations. These equations use the Markovian property to establish a recursive relationship between values in the present and values in the future. Recalling the definition of v and q from earlier, we can expand these by a single timestep and observe the following relationship (Sutton & Barto, 2018):

$$v_\pi(s_t) = \mathbb{E}_{a \sim \pi, s_{t+1} \sim P} [R(s_t, a_t) + \gamma v_\pi(s_{t+1})] \quad (2.2)$$

$$q_\pi(s_t, a_t) = R(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi, s_{t+1} \sim P} [q_\pi(s_{t+1}, a_{t+1})] \quad (2.3)$$

Now we address Bellman optimality equations; recalling Section 2.1, the optimal policy π^* is the policy that maximizes the value function v_π for all states s , and we can write the value function corresponding to the optimal policy as $v^*(s)$, and similarly for the action-value function $q^*(s, a)$. Now we can leverage a key property about optimality in MDPs to write the optimal value (and action-value) functions *without* reference to an actual policy, and can obtain an update rule. Recalling that the optimal value functions must be greater than any other value function for all states, we can relate the optimal value (v^*) and action-value (q^*) functions

through simply noticing that the optimal policy is also a policy that takes the action which *maximizes the expected return* from a state s , and then expand this to obtain the *Bellman optimality equations*:

$$v^*(s) = \max_a q^*(s, a) \quad (2.4)$$

$$= \max_a \mathbb{E}_{\pi^*} [G_t | S_t = s, A_t = a] \quad (2.5)$$

$$= \max_a \mathbb{E}_{\pi^*} [R(s, a) + \gamma G_{t+1} | S_t = s, A_t = a] \quad (2.6)$$

$$= \max_a \mathbb{E}_{s_{t+1} \sim P} [R(s, a) + \gamma v^*(s_{t+1})] \quad (2.7)$$

Similarly, for the action-value function:

$$q^*(s, a) = R(s, a) + \gamma \mathbb{E}_{s_{t+1} \sim P} \left[\max_{a_{t+1}} q^*(s_{t+1}, a_{t+1}) \right] \quad (2.8)$$

Invoking Banach’s fixed point theorem (Szepesvári, 2022), we can in fact treat this recursive relationship as an *update rule*, because it is a guaranteed contraction in the L_∞ norm, and of course that fixed point must be the optimal value (and action-value) function. With this in mind, to perform learning with Equation (2.8), we can parameterize an action-value function using a neural network (i.e., $\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$), and we simply leverage the convergence property by taking the difference between the left and right hand sides, and minimize it with respect to the parameters of the action-value function using gradient descent.

This forms the basis for many of the algorithms that we use in Approximate Dynamic Programming (DP) methods, such as Q-learning (Watkins & Dayan, 1992; Mnih et al., 2015). To provide some clarity on why these *approximate* DP, we note that in deep RL, we do not typically have access to the MDP to evaluate and iterate the Bellman equations for all actions and states, so therefore must approximate the DP procedure by instead using *samples* of actions and states gained while interacting with the MDP during policy rollouts.

As a final note, for continuous control problems, we usually instantiate an actor network π_θ which outputs a continuous action vector, such as in Lillicrap et al. (2015). Confusingly, these methods are also called policy gradient approaches, namely deterministic policy gradients (DPG), but practically bear little resemblance

to the approaches in Section 2.2.1, which are called stochastic policy gradients (SPG). While it can be shown that DPG and SPG are special limiting cases of each other (Silver et al., 2014), DPG instead learns an action-value function, or Critic, using off-policy Expected SARSA (similar to Q-learning) (Sutton & Barto, 2018), and then updates the actor weights using gradients through the learned Critic by invoking the chain-rule¹. This means that the actor training approach more closely resembles that seen in Generative Adversarial Networks (Goodfellow et al., 2014; Pfau & Vinyals, 2016), whereby the the actor attempts to “adversarially attack” the critic using gradients to find actions that produce high expected return. With this in mind, the actor gradient can be written as follows:

$$\nabla_{\theta} J_{\pi_{\theta}} = \mathbb{E}_{s \sim P} \left[\nabla_a q(s, a)|_{a=\pi_{\theta}(s)} \nabla_{\theta} \pi_{\theta}(s) \right]. \quad (2.9)$$

Modern instantiations of DPG-style algorithms include Fujimoto et al. (2018); Haarnoja et al. (2018). While these use a stochastic actor, they both rely on the same Expected SARSA policy evaluation approach, thus practically only require the additional marginalization over the randomness induced by the policy. As in much of deep learning, this expectation is estimated using a single Monte-Carlo sample, so optimization therefore only requires the ability to take such a sample, and if there is a need to learn the sampling distribution itself, we can use pathwise derivatives (Rezende et al., 2014; Kingma & Welling, 2014).

2.2.3 Model-Based Reinforcement Learning

Model-based approaches simultaneously derive a policy *and* learn a dynamics model of the MDP. Concretely, they attempt to derive π^* while also modeling $P(s'|s, a)$. In this thesis, the estimate of the dynamics function is parameterized by weights ω belonging to a neural network, i.e., $\hat{P}_{\omega}(s'|s, a)$.

There are several different approaches for discovering optimal policies within model-based RL. Model Predictive Control (MPC) (Tassa et al., 2012) eschews policy learning entirely, and instead queries the learned model directly. It predicts

¹The action-value update can be seen as a step of Policy Evaluation, and the actor update can be seen as a step of Policy Improvement.

the optimal next action by performing multiple rollouts (planning) inside the learned dynamics model using, for instance, random shooting (sampling actions uniformly) (Nagabandi et al., 2018) and selecting the action that resulted in the highest return. Other approaches train a policy explicitly by using gradients through the model (Werbos, 1988) to determine the direction for greatest policy improvement (Deisenroth & Rasmussen, 2011; Heess et al., 2015; Clavera et al., 2020). Finally, Dyna-style (Sutton, 1990) approaches simply pretend the model is the real world, and use it to train a standard RL policy, without any of the cost associated with sampling the true MDP (Kurutach et al., 2018; Janner et al., 2019).

To avoid confusion, we note several naming conventions that exist in model-based RL. Oftentimes the learned transition dynamics function $\hat{P}_\omega$ is referred to as a *world model*. We also note that policy rollouts that occur within the world model can also be called *imagined* rollouts.

2.3 Offline Reinforcement Learning

Offline Reinforcement Learning (also called Batch Mode RL) (Ernst et al., 2005; Levine et al., 2020) is a topic of research explored in Chapter 3 and Chapter 5 of this thesis. A key difference in offline RL compared to online RL is that the MDP cannot be accessed at all by the policy. Therefore we are fully reliant on a *fixed* offline dataset, which comprises tuples of interactions by some prior behavior policy in the form $\mathcal{D} = \{(s_t^i, a_t^i, r_t^i, s_{t+1}^i)\}$. The goal is therefore to learn the optimal behavior given this fixed dataset $\mathcal{D}$ without any online interaction.

A key difference between offline RL and approaches using behavior cloning (BC) or imitation learning (IL) (Ross & Bagnell, 2010) is that no assumption is made on the quality of the dataset provided. In BC and IL, it is assumed that the data are high quality, hence the learning problem reduces to either distribution matching (Ho & Ermon, 2016) or supervised learning (Ross & Bagnell, 2010) so as to emulate the policy implied by the provided trajectories. Offline RL on the other hand does not make this assumption, and relies on the properties of RL

to derive optimal behavior through the provided interactions, such as through trajectory-stitching (Kostrikov et al., 2022).

Canonical model-free approaches to offline RL usually involve using standard off-policy approaches, such as Q-learning or DPG-style algorithms, and then enforcing a constraint on those algorithms to ensure that value learning and/or policy learning does not stray too far from those in the provided dataset. The reason why constraints are necessary compared with online RL is because an online RL agent is able to receive corrective feedback from the MDP if it accidentally overestimates (Thrun & Schwartz, 1993) part of the state-action space. In offline RL however, there is no such corrective mechanism, so there is a greater possibility of runaway value divergence (Fujimoto et al., 2019), resulting in poor performing agents whose behaviors are based on incorrectly overestimated values. Popular constraints include enforcing a behavior cloning term (Fujimoto & Gu, 2021), ensuring conservatism in the values on actions that do not lie in the offline dataset (Kumar et al., 2020), or discounting values based on uncertainty (An et al., 2021).

Model-based RL can also be applied to solve offline RL problems. Approaches typically first learn a dynamics model from the offline dataset, then train a policy using the learned dynamics model as if it were the real MDP. However, analogous to value divergence issues in the model-free case, policies can learn to “exploit” the learned model where it is inaccurate, resulting in learning actions that appear to perform well in the learned model, but then perform poorly on the real MDP. To resolve this issue, common approaches apply some degree of penalty to the reward function to dissuade sampling parts of the state-action space where the learned dynamics model may make errors, such as through uncertainty on transition dynamics (Yu et al., 2020; Kidambi et al., 2020); indeed we cover such approaches in Section 3.1.

2.3.1 Deployment-Efficient Reinforcement Learning

In Section 5.2, we consider the deployment-efficient setting (Matsushima et al., 2021), which can be considered a middle-ground between online and offline RL.

Concretely, in pure online approaches we do not limit how often we can deploy our agent after each round of training; we could for instance choose to run several steps of gradient descent during one timestep, and therefore deploy a different agent iteration at each timestep of a single rollout (Chen et al., 2021). Offline RL on the other hand only allows a single deployment, which is performed after training. The deployment-efficient setting bridges this gap by setting predefined training (offline) and deployment phases (online). A key difference to pure online RL is that during the deployment phase, we cannot perform any agent training, so we must deploy a single instantiation for a fixed number of timesteps before we are allowed to retrain on the newly gathered data.

Deployment-efficient settings are representative of limitations that may occur the real-world. For instance, re-training the policy may be computationally expensive, so we cannot afford to retrain at each timestep without creating unacceptable latencies during deployment. As another example, if safety is required, we will need to verify an agent after each step of training, thus it cannot be updated once it is deployed.

Chapter Bibliography

- Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. *Advances in Neural Information Processing Systems*, 34:7436–7447, 2021.
- Karl Johan Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.
- Richard Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957. URL <http://www.jstor.org/stable/24900506>.
- Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized ensembled double Q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=AY8zfZm0tDd>.
- Ignasi Clavera, Yao Fu, and Pieter Abbeel. Model-augmented actor-critic: Backpropagating through paths. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=Skln2A4YDB>.
- Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 465–472, 2011.

- Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005.
- Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.
- Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.
- Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, 2014.
- Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.
- Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in Neural Information Processing Systems*, 28, 2015.
- Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in Neural Information Processing Systems*, 29, 2016.
- Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.
- Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOREL: Model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:21810–21823, 2020.
- Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
- Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=68n2s9ZJWF8>.
- Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.
- Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=SJJinbWRZ>.

- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.
- Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=3hGNqpI4WS>.
- 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.
- Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937. PMLR, 2016.
- Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7559–7566. IEEE, 2018.
- David Pfau and Oriol Vinyals. Connecting generative adversarial networks and actor-critic methods. *arXiv preprint arXiv:1610.01945*, 2016.
- Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286. PMLR, 2014.
- Herbert E. Robbins. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951. URL <https://api.semanticscholar.org/CorpusID:16945044>.
- Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 661–668. JMLR Workshop and Conference Proceedings, 2010.
- John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, pp. 387–395. Pmlr, 2014.

- Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pp. 216–224. Elsevier, 1990.
- Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12, 1999.
- Csaba Szepesvári. *Algorithms for reinforcement learning*. Springer Nature, 2022.
- Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913. IEEE, 2012.
- Sebastian Thrun and A. Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of 4th Connectionist Models Summer School*. Erlbaum Associates, June 1993.
- Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.
- Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1(4):339–356, 1988.
- Chelsea C White III and Douglas J White. Markov decision processes. *European Journal of Operational Research*, 39(1):1–16, 1989.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

3

Effective Pre-Training with Offline Data

Contents

3.1	Revisiting Design Choices in Offline Model-Based RL	26
3.1.1	Introduction	26
3.1.2	Related Work	29
3.1.3	Background	31
3.1.4	Uncertainty Penalty	32
3.1.5	Key Hyperparameters in Offline MBRL	35
3.1.6	Testing the Limits of Current Approaches	39
3.1.7	Conclusion	43
3.1.8	Errata	44
	Chapter Bibliography	46

In this chapter, we concentrate on the problem of effective offline reinforcement learning.

As offline data, or prior data, become more readily available, a key problem is how to best pre-train on this data. In this chapter, we tackle this problem using offline RL, and determine how to learn the most optimal behaviors from a huge diversity of offline datasets without any online interaction. Optimal pre-training is an important question because this allows us to maximize utility gained from the information contained in the dataset, and also forms the foundation from which we can learn further behaviors, should we choose to continue training online.

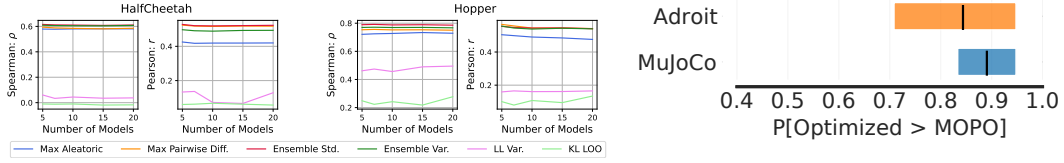
3.1 Revisiting Design Choices in Offline Model-Based RL

Abstract Offline reinforcement learning enables agents to leverage large pre-collected datasets of environment transitions to learn control policies, circumventing the need for potentially expensive or unsafe online data collection. Significant progress has been made recently in offline model-based reinforcement learning, approaches which leverage a learned dynamics model. This typically involves constructing a probabilistic model, and using the model uncertainty to penalize rewards where there is insufficient data, solving for a *pessimistic* MDP that lower bounds the true MDP. Existing methods, however, exhibit a breakdown between theory and practice, whereby pessimistic return ought to be bounded by the *total variation distance* of the model from the true dynamics, but is instead implemented through a penalty based on estimated *model uncertainty*. This has spawned a variety of uncertainty heuristics, with little to no comparison between differing approaches. In this paper, we compare these heuristics, and design novel protocols to investigate their interaction with other hyperparameters, such as the number of models, or imaginary rollout horizon. Using these insights, we show that selecting these key hyperparameters using Bayesian Optimization produces superior configurations that are vastly different to those currently used in existing hand-tuned state-of-the-art methods, and result in drastically stronger performance.

3.1.1 Introduction

In offline (or batch) reinforcement learning (RL) (Ernst et al., 2005; Levine et al., 2020), the goal is to leverage offline datasets of transitions in an environment to train a policy that transfers to an online task. This could have vast implications for using RL in real-world settings, as agents can make use of ever-increasing amounts of data without the need for an accurate simulator, while also avoiding expensive and potentially even unsafe exploration in the environment.

Model-based reinforcement learning (MBRL) has recently shown promise in this paradigm, obtaining state-of-the-art performance on offline RL benchmarks



(a) Uncertainty vs. True Errors

(b) Probability of Improvement

Figure 3.1: **a)** The variation of different uncertainty penalties against *true* dynamics error during a model rollout of Hopper Medium-Expert. The canonical ensemble variance penalty most closely fits the true dynamics error. **b)** Tuning key hyperparameters (an approach we call Optimized) can lead to large gains over state-of-the-art methods (MOPO) on the D4RL benchmark, as we show in this summary using `rliable` (Agarwal et al., 2021).

(Kidambi et al., 2020; Yu et al., 2021), improving upon powerful model-free approaches (e.g. Kumar et al. (2020)). MBRL works by training a dynamics model from the offline data, then optimizing a policy using imaginary rollouts from the model. This allows the agent to learn from on-policy experience, as the model is agnostic to the policy used to generate data, making it possible to achieve high returns using data collected from even a random policy. Furthermore, recent work has demonstrated the utility of world models *beyond* maximizing return, such as generalizing to unseen variations in an environment (Ball et al., 2021), transferring to new tasks (Yu et al., 2020), and learning with safety constraints (Argenson & Dulac-Arnold, 2021). Therefore, the case for MBRL in offline RL is clear: not only does it represent state-of-the-art in terms of performance, but it also provides the opportunity to maximize the signal in the offline data to generalize onto tasks beyond those encoded by the behavior policy. This is crucial for offline RL to be useful for real-world tasks (Dulac-Arnold et al., 2021), where there will inevitably be differences between the data and desired task.

However, a common failure mode of MBRL is when policies exploit the model in parts of the state-action space where the model is inaccurate. Thus, naïve application of MBRL to offline data can result in suboptimal performance. To prevent this, concurrent works (Yu et al., 2020; Kidambi et al., 2020) have approached the problem by training a policy in a *pessimistic* MDP (P-MDP). The P-MDP lower bounds the true MDP, and discourages the policy from regions where there is large discrepancy between the true and learned dynamics; this often provides a theoretical guarantee of improvement over cloning the behavior policy that generated the offline

data. This is made practically possible by adding a penalty correlated with the uncertainty in the dynamics model. However, while these recent successes are similar in principle, in practice they differ in a series of design choices. First and foremost, they make use of different heuristics to measure model uncertainty, in some cases deviating from simpler metrics which are more consistent with the theory.

In this paper, we conduct a rigorous investigation into a series of these design choices. We begin by focusing on the choice of uncertainty metric, comparing both recent state-of-the-art offline approaches (Kidambi et al., 2020; Yu et al., 2020; Rafailov et al., 2020) with additional metrics used in the online setting (Ball et al., 2020; Pan et al., 2020; Cowen-Rivers et al., 2022). We also explore the interaction with a series of other hyperparameters, such as the number of models and imaginary rollout length. Interestingly, the relationship between these variables and model uncertainty varies significantly depending on the choice of uncertainty penalty. Furthermore, we compare these uncertainty heuristics under new evaluation protocols that, for the first time, capture the specific covariate shift induced by model-based RL. This allows us to assess calibration to model exploitation in MBRL, observing that some existing penalties are surprisingly successful at capturing the errors in predicted dynamics, as seen in Fig. 3.1a (see App. A.4 for details). Then, using the insights gained from sections 3.1.4 and 3.1.5, we then achieve a **43%** gain over a previously grid-searched method by using a *single hyperparameter* value across all environments. We then jointly fine-tune our identified key variables using a powerful Bayesian Optimization algorithm (Wan et al., 2021) and find the simpler uncertainty measures can provide state-of-the-art results in continuous control offline benchmarks, and that the chosen optimal hyperparameters continue to align with our analysis. Finally, we rigorously confirm the aggregate improvement of our results using the `rliable` framework (Agarwal et al., 2021) in Fig. 3.1b, and show that the improvements over existing methods are significant (see App. A.8 for details). This work is intended to benefit both researchers and practitioners in offline RL. Our main findings include:

- **Longer horizon rollouts with larger penalties can improve existing methods.** Contrary to common wisdom, conducting significantly longer rollouts inside the model, coupled with larger uncertainty penalties, typically improves performance.
- **Penalties that use canonical forms of uncertainty estimation achieve better correlation with OOD measures.** The uncertainty estimation approach of [Lakshminarayanan et al. \(2017\)](#) often outperforms the penalty from state-of-the-art methods ([Yu et al., 2020](#); [Kidambi et al., 2020](#)). We observe that the ensemble standard deviation is statistically strikingly similar to that used in [Kidambi et al. \(2020\)](#), but has improved correlation and scaling behavior.
- **Uncertainty is more correlated with *dynamics error* than *distribution shift*.** We find that successful penalties measure the discrepancy in dynamics, and can in fact assign high certainty to data far away from the offline data.

3.1.2 Related Work

Two recent works concurrently demonstrated the effectiveness of model-based reinforcement learning (MBRL) in the offline setting. MOPO ([Yu et al., 2020](#)) follows the successful online RL algorithm MBPO ([Janner et al., 2019](#)) but trains inside a *conservative MDP*, penalizing the reward based on the maximum aleatoric uncertainty over the ensemble members. MOREL ([Kidambi et al., 2020](#)) achieves even stronger performance, penalizing the rewards by a penalty based on the maximum pair-wise difference in ensemble member predictions. For pixel-based tasks, LOMPO ([Rafailov et al., 2020](#)) also proposed a novel penalty, using the variance of ensemble log-likelihoods. Outside the offline setting, probabilistic dynamics models leveraging uncertainty have underpinned a series of successes ([Chua et al., 2018](#); [Kurutach et al., 2018](#); [Buckman et al., 2018](#); [Pan et al., 2020](#); [Pacchiano et al., 2021](#)). Uncertainty can also be measured in MBRL without the

use of neural networks (Deisenroth & Rasmussen, 2011), although these methods tend to be harder to scale and thus lack widespread use.

Effective hyperparameter selection in RL has been shown to be crucial to the success of popular algorithms (Engstrom et al., 2020; Andrychowicz et al., 2021). This becomes even more challenging in MBRL with additional hyperparameters/design-choices for the dynamics model. Recent work has shown that carefully optimizing these hyperparameters for online MBRL can significantly improve performance, with the tuned agent breaking the MuJoCo simulator (Zhang et al., 2021). In contrast, we focus on the offline setting, and investigate parameters specifically related to uncertainty estimation. Previous work studied the impact of hyperparameters in offline RL (Paine et al., 2020), finding offline RL algorithms to be brittle to hyperparameter choices. However, unlike our work they only consider model-free approaches, whereas we specifically investigate *model-based* offline algorithms. Abbas et al. (2020) investigates the impact of different uncertainty estimation methods in online MBRL; they too find penalizing with combined aleatoric and epistemic uncertainty improves performance.

Our work also relates to the rich literature on *deep ensembles* (Lakshminarayanan et al., 2017), which train multiple deep neural networks with different initializations and dataset orderings, and generally outperform variational Bayes methods (Mackay, 1992; Blundell et al., 2015). Achieving effective calibration with neural networks is notoriously difficult (Guo et al., 2017; Kuleshov et al., 2018; Maddox et al., 2019), and furthermore we require calibration under co-variate shift (Ovadia et al., 2019), as the policy learned in the model will likely deviate from the behavior policy that generated the offline data. Recent work has highlighted this issue in offline RL (Kumar et al., 2020; Yu et al., 2021) and has reported superior performance when eschewing model uncertainty entirely, and instead performing “conservative” Q-updates. However, it is unclear if this improvement is due to poor uncertainty calibration, implementation details, or a limitation in the pessimistic-MDP formulation.

3.1.3 Background

All of the methods we investigate in this paper model the environment as a Markov Decision Process (MDP), defined as a tuple $M = (\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces respectively, $P(s'|s, a)$ the transition dynamics, $R(s, a)$ the reward function, ρ_0 the initial state distribution, and $\gamma \in (0, 1)$ the discount factor. The goal is to optimize a policy $\pi(a|s)$ that maximizes the expected discounted return $\mathbb{E}_{\pi, P, \rho_0} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$.

In *offline RL*, the policy is not deployed in the environment until test time. Instead, the algorithm only has access to a static dataset $\mathcal{D}_{env} = \{(s_j, a_j, r_j, s_{j+1})\}_{j=1}^J$, collected by one or more behavioral policies π_b . Following the notation in Yu et al. (2020) we refer to the distribution from which $\mathcal{D}_{env}$ was sampled as the *behavioral distribution*. The canonical approach in offline MBRL is to train an ensemble of N probabilistic dynamics models (Nix & Weigend, 1994). These usually learn to predict both the next state s_{t+1} and reward r_t from a state-action pair, and are trained on $\mathcal{D}_{env}$ using supervised learning. Concretely, each of the N models output a Gaussian $\hat{P}_{\phi}^i(s_{t+1}, r_t | s_t, a_t) = \mathcal{N}(\mu_{\phi}^i(s_t, a_t), \Sigma_{\phi}^i(s_t, a_t))$ parameterized by ϕ . The resulting learned dynamics model $\hat{P}$ and reward model $\hat{R}$ define a *model MDP* $\hat{M} = (\mathcal{S}, \mathcal{A}, \hat{P}, \hat{R}, \rho_0, \gamma)$. To train the policy, we use k -step rollouts inside $\hat{M}$ to generate trajectories (Sutton, 1991).

To prevent policy exploitation in a model, a pessimistic MDP (P-MDP) is constructed by lower bounding the true-expected return, $\eta_M(\pi)$, using some error between the true and estimated models. For instance, in Yu et al. (2020), the authors show that a lower bound on the return can be established by penalizing the reward by a measure that corresponds to estimated model error:

$$\eta_M(\pi) \geq \mathbb{E}_{(s,a) \sim \rho_P^{\pi}} [R(s, a) - \gamma |G_M^{\pi}(s, a)|] \quad (3.1)$$

where ρ_P^{π} represents transitioning under the dynamics model $\hat{P}$ and policy π . Several potential choices for $|G_M^{\pi}(s, a)|$ are proposed, including an upper bound based on the total variation distance between the learned and true dynamics. However, for their practical algorithm, the authors elect to use a heuristic based on impressive empirical

results. Concurrent to MOPO, MOREL (Kidambi et al., 2020) in theory constructs a P-MDP by augmenting a standard MDP with a negative valued absorbing state that is transitioned to when total variation distance between true and learned dynamics is exceeded. They show that a policy learned in this P-MDP exceeds simple behavior cloning. However, while dynamics-based total variation distance has desirable theoretical properties, the practical algorithm relies on another heuristic to approximate this quantity. This motivates the study of penalties used, as well as other under-used candidates, and their overall effectiveness.

3.1.4 Uncertainty Penalty

The key idea underpinning recent success in offline MBRL is the introduction of a P-MDP, penalized by some uncertainty penalty. The theory dictates this should be some distance measure between the true and predicted dynamics. Of course, this cannot be truly estimated without access to an oracle, so a proxy for this quantity is constructed instead based on uncertainty heuristics. In this paper, we compare the following uncertainty heuristics, from recent works in both offline and online MBRL:

Max Aleatoric (Yu et al., 2020): $\max_{i=1,\dots,N} \|\Sigma_\phi^i(s, a)\|_F$, which corresponds to the maximum aleatoric error, computed over the variance heads of the model ensemble.

Max Pairwise Diff (Kidambi et al., 2020): $\max_{i,j} \|\mu_\phi^i(s, a) - \mu_\phi^j(s, a)\|_2$, which corresponds to the pairwise maximum difference of the ensemble predictions.

LL Var (Log-Likelihood Variance) (Rafailov et al., 2020):

$\text{Var}(\{\log \hat{P}_\phi^i(s'|s, a), i = 1, \dots, N\})$, where s' is a next state sampled from a single ensemble member. We evaluate its log-likelihood under each ensemble member and take the variance.

LOO KL (Leave-One-Out KL Divergence) (Pan et al., 2020):

$D_{\text{KL}}[\hat{P}_{\phi_i}(\cdot|s, a) \parallel \hat{P}_{\phi_{-i}}(\cdot|s, a)]$, which corresponds to the KL divergence between the Gaussian parameterized by a randomly selected ensemble member, and the aggregated Gaussian of the remaining ensemble members.

Ensemble Standard Deviation/Variance (Lakshminarayanan et al., 2017):

The variance is given as: $\Sigma^*(s, a) = \frac{1}{N} \sum_i^N ((\Sigma_\phi^i(s, a))^2 + (\mu_\phi^i(s, a))^2) - (\mu^*(s, a))^2$ where μ^* is the mean of the means ($\mu^*(s, a) = \frac{1}{N} \sum_i^N \mu_\phi^i(s, a)$). This corresponds to a combination of epistemic and aleatoric model uncertainty. This is surprisingly under-utilized in offline MBRL, and is a canonical method of uncertainty estimation used in the Bayesian inference literature (Ovadia et al., 2019; Filos et al., 2019; Scalia et al., 2020). We choose to evaluate both standard deviation (the square root of the above) and variance, as this will provide intuition about the importance of penalty distribution *shape*.

These can all be computed using the output from an ensemble of probabilistic dynamics models (Lakshminarayanan et al., 2017), so we are able to compare them in a controlled manner.

How Well Do Ensemble Penalties Detect Out of Distribution Errors?

We begin by assessing how well uncertainty penalties correlate with next state MSE (we justify the MSE under deterministic dynamics in App. A.1.2). This is crucial in penalizing the policy from visiting parts of the state-action space where the model is inaccurate, and therefore exploitable. Using D4RL (Fu et al., 2021a), we train models on each dataset, then evaluate them on *other datasets* from the same environment, but collected under *different* policies. These form our “Transfer” experiments as they directly measure the ability of uncertainty penalties at detecting errors on *unseen* data. We compare the penalties against true MSE for a variety of settings in App. A.1.3, and summarize this in the “Transfer” column of Table 3.1. We measure Spearman rank (ρ) and Pearson bivariate (r) correlations, and justify their use in App. A.1.1. Full details of all experiments and hyperparameters are given in App. A.7. We will analyze these results in detail in the next section, after introducing a novel protocol for assessing our penalties under the out-of-distribution (OOD) data induced by model exploitation.

Table 3.1: Correlation statistics of penalties against true mean-sq. model error, averaged over all datasets (i.e., Random through to Expert) showing ± 1 SD over 12 seeds. The best in each column is **bolded**. The ensemble penalties generally perform best.

Penalty	Transfer				True Model-Based			
	HalfCheetah		Hopper		HalfCheetah		Hopper	
	ρ	r	ρ	r	ρ	r	ρ	r
Max Aleatoric	0.78 $\pm$ 0.00	0.55 $\pm$ 0.01	0.71 $\pm$ 0.01	0.41 $\pm$ 0.01	0.58 $\pm$ 0.01	0.42 $\pm$ 0.01	0.73 $\pm$ 0.03	0.48 $\pm$ 0.01
Max Pairwise Diff.	0.79 $\pm$ 0.01	0.62 $\pm$ 0.00	0.77 $\pm$ 0.00	0.57 $\pm$ 0.00	0.58 $\pm$ 0.01	0.52 $\pm$ 0.01	0.75 $\pm$ 0.02	0.55 $\pm$ 0.02
Ens. Std.	0.82 $\pm$ 0.01	0.64 $\pm$ 0.01	0.79 $\pm$ 0.00	0.56 $\pm$ 0.00	0.61 $\pm$ 0.01	0.52 $\pm$ 0.00	0.79 $\pm$ 0.02	0.55 $\pm$ 0.02
Ens. Var.	0.82 $\pm$ 0.01	0.67 $\pm$ 0.00	0.79 $\pm$ 0.00	0.59 $\pm$ 0.00	0.60 $\pm$ 0.01	0.49 $\pm$ 0.01	0.77 $\pm$ 0.02	0.55 $\pm$ 0.02
LL Var.	0.13 $\pm$ 0.05	0.14 $\pm$ 0.02	0.36 $\pm$ 0.04	0.12 $\pm$ 0.02	0.04 $\pm$ 0.16	0.07 $\pm$ 0.06	0.50 $\pm$ 0.02	0.16 $\pm$ 0.02
LOO KL	0.03 $\pm$ 0.02	0.11 $\pm$ 0.02	0.11 $\pm$ 0.02	0.08 $\pm$ 0.02	-0.02 $\pm$ 0.12	0.06 $\pm$ 0.06	0.22 $\pm$ 0.02	0.10 $\pm$ 0.02

How Do These Perform During an Imaginary Rollout?

We additionally design an experiment aimed at capturing the OOD data *generated by the actual offline MBRL process*, which we call our “True Model-Based” experiments. First, we train a set of policies with 4 different starting seeds *without* a penalty inside the model for 500 iterations. We then measure the difference between the return predicted by the model over a rollout, and the true return in the real environment. We define a policy to be “exploitative” if the model significantly *over-estimates* the return compared to the true return. It is these exploitative policies that induce the types of extrapolation errors which cause MBRL methods to fail in the offline setting. It is therefore important that the penalty is able to accurately determine when the model is being exploited in this way. We use a subset of the 5 most exploitative policies to generate trajectories in the model, and record the uncertainty predicted by each penalty at every time step. To generate the True Model-Based data, we then “replay” these trajectories in the true environment, loading the state and action taken in the model into the environment, and record the “true” next state according to the MuJoCo simulator (Todorov et al., 2012). True Model-Based therefore calculates the MSE between the *predicted* and *actual* next states. Table 3.1 summarizes the results from both the Transfer and True Model-Based experiments. Additional details are provided in App. A.4 along with full correlation plots in App. A.1.3.

We are now in a position to analyze the results in Table 3.1. It is immediately obvious that the LOO KL and LL Var penalties have very weak correlation with MSE. We believe this is because LL Var relies on likelihood statistics, which are notoriously sensitive; it was designed for use with a KL-regularized latent

state space model which has well-behaved dynamics. Regarding LOO KL, we note that this penalty was designed for the online setting with significantly less data, and becomes quite uncorrelated in this larger data setting. This advocates penalties that are less reliant on distributional information concerning the separate Gaussians in the ensemble, as such penalties appear sensitive to the quality of their estimated distributions. We observe that Max Aleatoric, Max Pairwise Diff and the Ensemble penalties perform broadly similarly despite their different analytical forms; Ensemble measures do however exhibit noticeably higher rank correlation. We also observe a significant performance loss between the Transfer and True Model-Based HalfCheetah settings, with the latter being relatively poor. This implies further work is needed to develop penalties that can successfully detect the type of dynamics discrepancies that actually arise in offline MBRL. Finally, we observe that despite the similar rank correlations ρ , the bivariate correlations r can vary considerably, and observe from the scatter plots that Max Aleatoric exhibits low kurtosis, having large penalty values “bunched” at its extreme; we provide 3rd and 4th order moment statistics to facilitate shape comparisons in App. A.3.

3.1.5 Key Hyperparameters in Offline MBRL

How Many Models Do We Need?

At present the number of models used has not been discussed since MBPO, which trains seven probabilistic dynamics models of the same architecture (with different initializations), using only the top five models based on validation accuracy (referred to as “Elites” in the Evolutionary community, e.g. Mouret & Clune (2015)). The reason or justification for this is not discussed, but it has seemingly been adopted in the wider MBRL setting (Shen et al., 2020; Omer et al., 2021; Pineda et al., 2021). However, offline RL is a totally different paradigm, where it is possible that access to compute is less of a bottleneck and it may be preferable to use more models to extract the most signal possible from the static dataset. Inevitably, many of the ensemble penalties are dependent on the number of models; for example, it is easy to see that the Max Aleatoric value could scale poorly with more models.

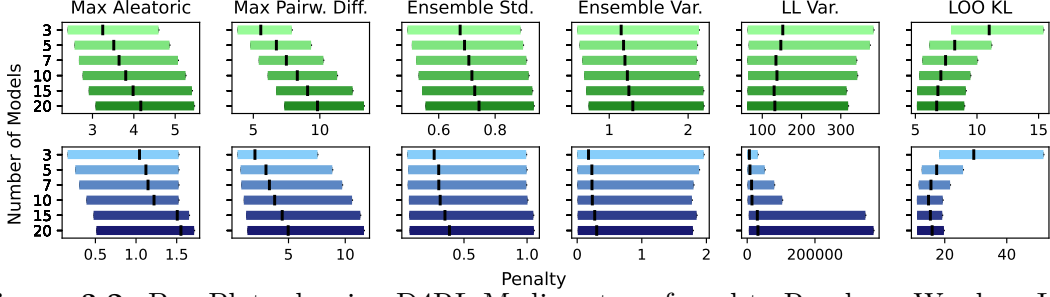


Figure 3.2: Box Plots showing D4RL Medium transferred to Random. We show IQR limits and the median value denoted by the black vertical line. **Green = HalfCheetah**, **Blue = Hopper**. Max Aleatoric, Max Pairw. Diff. and LOO KL are unstable w.r.t. ensemble member count. In contrast, ensemble variance and std. are far more stable.

How Does Penalty Distribution Change with Model Count? We now vary the number of models used in the calculation of the penalties and plot their respective distributions; an illustrative example is shown in Fig. 3.2 with full results in App. A.2. The scaling of penalties relying on max over sets is most affected with increasing the number of models due to admitting more extreme values, and we observe that the distribution shape of Max Aleatoric changes significantly as we admit more models, which we validate in App. A.3. This impacts the tuning of this hyperparameter, as we have to contend with a changing distribution along with calibration quality (which we explore in the next section). Finally, we observe that the Ensemble penalties change the least with differing model count, highlighting their ease of tuning; this is clearly a desirable property for designing such metrics going forward.

How does Penalty Performance Scale with Model Count? Empirically, there exists an optimal number of models to use in an ensemble for model-based RL (Kurutach et al., 2018; Matsushima et al., 2021). Up to now, heuristics have been used to select how many models we use for uncertainty estimation, despite it being possible to

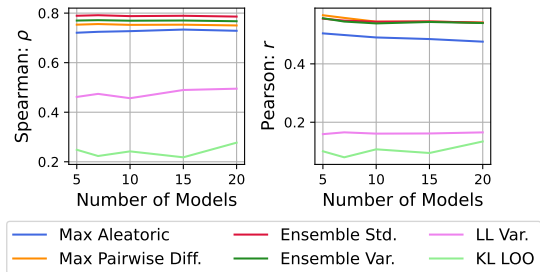


Figure 3.3: Plot of how error and penalty correlation changes with model number in Hopper across all datasets (i.e., Random through to Expert).

use a different number of models for dynamics prediction and uncertainty estimation. For instance, in MOPO, transitions are generated with five Elite models, but all seven models are used to calculate the penalty. In MOREL, four models are used for both transitions and penalty prediction. Therefore, we wish to understand if there is merit to using a larger number of models for uncertainty estimation compared with next state prediction. We provide a snapshot in Fig. 3.3, showing the aggregated results on the True Model-Based data in Hopper, with full results in App. A.2. We see there is no clear consensus, and that the optimal number of models is highly dependent on environment, the behavior data, and penalty type, with some settings showing improved calibration with model count and vice-versa. This clearly justifies treating the number of models as a hyperparameter that is important to tune, especially on transfer tasks. Interestingly, we observe that it is possible to simultaneously improve rank (ρ) correlation, but reduce bivariate (r) correlation, especially with the MOPO penalty. This again suggests that the number of models not only affects the quality of the estimation, but *also its distributional shape*.

The Weight of Uncertainty λ

To weight penalty against reward, MOPO introduces a parameter λ that trades off between the two terms. In their paper, the authors sweep over $\lambda \in \{1, 5\}$ for each environment. However, the optimal values may lie outside this region. Furthermore, we have shown this value will need to drastically change to account for using a different penalty or even number of models.

The Rollout Horizon h

The horizon h of the rollouts plays a crucial role in offline RL. Longer horizon rollouts increase the likelihood of errors in the transitions (we verify this intuition in App. A.4), but conversely can improve performance when errors are properly managed (Janner et al., 2019; Pan et al., 2020). Furthermore, as highlighted in Fig. 3.1a, the model can generalize, and dynamics error does not necessarily increase with drift away from the offline dataset. Instead, we observe spikes, and note it is possible to recover from these to valid states and transitions. It is therefore

Table 3.2: Performance of different penalties as OOD event detectors averaged over all datasets in Hopper and HalfCheetah (i.e., Random through to Expert) showing ± 1 SD over 12 seeds. AUC is “Area Under Curve” and AP is “Average Precision”. The best (highest) in each column is highlighted in **bold**.

Penalty	Percentile											
	90th				95th				99th			
	Dynamics		Distribution		Dynamics		Distribution		Dynamics		Distribution	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
Max Aleatoric	0.89 $\pm$ 0.01	0.50 $\pm$ 0.02	0.76 $\pm$ 0.01	0.35 $\pm$ 0.01	0.89 $\pm$ 0.00	0.35 $\pm$ 0.02	0.80 $\pm$ 0.01	0.27 $\pm$ 0.01	0.92 $\pm$ 0.00	0.20 $\pm$ 0.04	0.89 $\pm$ 0.03	0.16 $\pm$ 0.04
Max Pairwise Diff.	0.90 $\pm$ 0.00	0.54 $\pm$ 0.01	0.77 $\pm$ 0.01	0.34 $\pm$ 0.01	0.91 $\pm$ 0.00	0.40 $\pm$ 0.02	0.81 $\pm$ 0.01	0.28 $\pm$ 0.01	0.93 $\pm$ 0.00	0.26 $\pm$ 0.01	0.89 $\pm$ 0.02	0.15 $\pm$ 0.02
Ensemble Std.	0.90 $\pm$ 0.00	0.55 $\pm$ 0.01	0.79 $\pm$ 0.01	0.38 $\pm$ 0.01	0.91 $\pm$ 0.00	0.40 $\pm$ 0.02	0.83 $\pm$ 0.01	0.31 $\pm$ 0.01	0.93 $\pm$ 0.00	0.25 $\pm$ 0.02	0.90 $\pm$ 0.02	0.18 $\pm$ 0.02
Ensemble Var.	0.90 $\pm$ 0.00	0.56 $\pm$ 0.01	0.78 $\pm$ 0.01	0.35 $\pm$ 0.01	0.91 $\pm$ 0.00	0.42 $\pm$ 0.02	0.82 $\pm$ 0.01	0.29 $\pm$ 0.01	0.93 $\pm$ 0.00	0.27 $\pm$ 0.01	0.89 $\pm$ 0.02	0.16 $\pm$ 0.02
LL Var.	0.66 $\pm$ 0.03	0.33 $\pm$ 0.00	0.74 $\pm$ 0.02	0.33 $\pm$ 0.00	0.67 $\pm$ 0.02	0.21 $\pm$ 0.02	0.76 $\pm$ 0.02	0.25 $\pm$ 0.02	0.73 $\pm$ 0.03	0.09 $\pm$ 0.01	0.81 $\pm$ 0.02	0.11 $\pm$ 0.01
LOO KL	0.59 $\pm$ 0.03	0.21 $\pm$ 0.01	0.68 $\pm$ 0.00	0.24 $\pm$ 0.02	0.60 $\pm$ 0.02	0.12 $\pm$ 0.00	0.70 $\pm$ 0.01	0.14 $\pm$ 0.02	0.65 $\pm$ 0.03	0.04 $\pm$ 0.00	0.72 $\pm$ 0.02	0.05 $\pm$ 0.00

imperative that a penalty captures these spikes over the course of an entire model rollout with horizon h , and down-weights the reward accordingly.

Using this observation, we design a novel experiment that treats these spikes as “positive” labels, and normalize each penalty to $[0, 1]$. This converts the penalties into a probabilistic classifier, and we evaluate how well they classify these events that occur increasingly under longer h . This is precisely the intuition behind the LOO KL and LL Var approaches, whereby the penalty acts as an anomaly detector, removing detrimental transitions that lie above a threshold. This is the regime we focus on here, where binary detection is more important than correlation. Finally, we assess two “True Model-Based” errors: the dynamics error as before, and introduce the distance from the offline distribution trained on, which we calculate as the 2-norm between a state-action tuple and its nearest point in the offline data (Dadashi et al., 2021); these are called “Dynamics” and “Distribution” respectively. We provide precision-recall curves and more details on this experiment in App. A.4 and A.5.

We observe in Table 3.2 that the penalties are powerful at identifying dynamics discrepancy, but not as accurate at identifying when the world-model data is out-of-distribution with respect to the offline data. This is a well-known phenomenon in deep neural networks and has been recently investigated in terms of feature collapse (Van Amersfoort et al., 2020), where latent representations of points far away in the input space get mapped close together. On the other hand, this shows an important distinction between the regularization induced by MBRL uncertainty and explicit state-action regularization in model-free approaches, such as Kumar et al. (2020); Wu et al. (2021). In the latter approaches, policies are penalized for

taking out of distribution actions w.r.t. the offline dataset, but this is not always the case with policies trained under MBRL and uncertainty penalties. The success of MBRL methods in RL may therefore lie in the generation of state-action samples that are **OOD but represent accurate dynamics**, thus facilitating dynamics generalization in policies; recent work has shown that augmenting dynamics improves offline RL policy generalization (Ball et al., 2021). We believe future work understanding the implications of this property is vitally important.

3.1.6 Testing the Limits of Current Approaches

Given our previous analysis, in this section we seek to answer the following question: how well can existing methods perform with a more optimal selection of the discussed hyperparameters? To answer this, we consider, first, a naïve selection of one hyperparameter set across all environments (based on our previous analysis), and then more definitively, tuning the configuration for each individual D4RL MuJoCo environment using a state-of-the-art Bayesian Optimization (BO) algorithm (Wan et al., 2021). Our first set of results show that following our analysis can provide significant gains over existing baselines, whilst the second beats the current SoTA. Note, previous analysis focused on HalfCheetah and Hopper environments, so we extend our evaluation to Walker2d as a held-out test.

General applicability of our insights. Two of our main takeaways in Sections 3.1.4 and 3.1.5 are that we should favor the canonical Ensemble penalties and longer rollout horizons. To test these claims, we design an experiment where we fix $h = 20$ for the horizon (c.f. $h = 5$ in MOPO at most), and only use Ensemble Std. as our penalty (see App. A.7 for details). Since tuning the penalty weight λ per environment is unrealistic, we employ an automatic penalty tuning scheme, analogous to the automatic entropy tuning used in Haarnoja et al. (2018). We tune the penalty weight on-the-fly to a constraint value of $\Lambda = 1$, meaning we *use only a single hyperparameter across all environments*. Full details on the penalty weight tuning are provided in App. A.9. With this approach, we get an average

Table 3.3: Best hyperparameters discovered by our BO algorithm, followed by a comparative evaluation on the D4RL benchmark suite against other model-based RL algorithms. We use D4RL v0 datasets. The raw score for Optimized[†] and MOPO[†] was taken to be the average over the last 10 iterations of policy training, averaged over 4 seeds and showing ± 1 SD. Results of MOPO and COMBO were taken from the COMBO paper. Results for MOREL were taken from its paper. $\star$ indicates $p < 0.05$ for Welch’s t-test for gain over MOPO. [†]Run on our codebase. [‡]Authors’ reported scores. [°]Authors used D4RL v2, which has more [performant](#) offline data.

Environment		Discovered Hyperparameters				Optimized [†]	MOPO [†]	MOPO [‡]	MOREL [°]	COMBO
		N	λ	h	Penalty					
HalfCheetah	random	10	6.64	12	Ensemble Std	31.7 $\pm$ 1.5	32.7 $\pm$ 1.7	35.4	25.6	38.8
	mixed	11	0.96	37	Ensemble Var	58.0 $\pm$ 2.5	52.8 $\pm$ 1.1	53.1	40.2	55.1
	medium	12	5.92	6	Ensemble Var	45.7 $\pm$ 2.6	46.5 $\pm$ 0.7	42.3	42.1	54.2
	med.-exp.	7	4.56	5	Max Aleatoric	104.2 $\pm$ 5.7 $\star$	67.6 $\pm$ 23.6	63.3	53.3	90.0
Hopper	random	6	4.46	47	Ensemble Std	12.1 $\pm$ 0.2 $\star$	4.2 $\pm$ 1.5	11.7	53.6	17.8
	mixed	7	5.90	5	Max Aleatoric	90.8 $\pm$ 11.1 $\star$	66.7 $\pm$ 27.8	67.5	93.6	73.1
	medium	7	37.28	42	Ensemble Std	69.3 $\pm$ 15.2 $\star$	17.3 $\pm$ 6.3	28.0	95.4	94.9
	med.-exp.	12	39.08	43	Max Aleatoric	105.8 $\pm$ 1.2 $\star$	24.9 $\pm$ 5.5	23.7	108.7	111.1
Walker2d	random	10	0.21	12	Ensemble Var	21.7 $\pm$ 0.1 $\star$	13.6 $\pm$ 1.4	13.6	37.3	7.0
	mixed	13	2.48	47	Ensemble Std	65.8 $\pm$ 17.4 $\star$	37.6 $\pm$ 20.6	39.0	49.8	56.0
	medium	8	5.28	14	Ensemble Std	79.7 $\pm$ 2.3 $\star$	-0.1 $\pm$ 0.0	17.8	77.8	75.5
	med.-exp.	12	0.99	37	Ensemble Std	97.1 $\pm$ 4.9 $\star$	46.2 $\pm$ 27.0	44.6	95.6	96.1
Average Score		-	-	-	-	65.2 $\pm$ 5.4 $\star$	34.2 $\pm$ 9.8	36.7	64.4	64.1

reward of **49.0** in the D4RL locomotion test suite (Fu et al., 2021a), an increase of **43%** over MOPO, which was grid-searched per environment.

This clearly shows that applying the findings from our analysis provides large performance gains generally. This result is the best we know for a single hyperparameter setup, and is particularly significant as other offline MBRL algorithms tune many hyperparameters per environment. This ‘zero-shot’ hyperparameter restriction is also the most realistic application of offline RL to real world problems. If we were to allow ourselves to take the maximum over just 2 hyperparameter setups (the second setup being $h = 10$, $\Lambda = 0.5$), we achieve an average reward of **57.8**, an increase of **69%** over MOPO. We show the full results in Table A.4 in App. A.9 with improvement probabilities.

Testing the limits of current approaches. Next, we wish to further validate that our earlier theoretical analysis can correspond to strong empirical performance gains by performing BO over the key hyperparameters. Details on the BO algorithm are listed in App. A.7. We define our search space over hyperparameters most related to *uncertainty quantification*:

- **Penalty type (categorical):** taking values over {Max Aleatoric, Max Pairwise Diff, LOO KL, LL Var, Ensemble Std, Ensemble Variance}.
- **Penalty scale λ (continuous):** taking values over $[1, 100]$.
- **h (integer):** taking values over $\{1, 2, \dots, 50\}$.
- **Models N (integer):** taking values over $\{1, 2, \dots, 15\}$.

Table 3.3 shows the optimal hyperparameters under BO. We note that Ensemble penalties are mainly selected, corroborating the findings in our analysis that these are most correlated with model error. We observe that Max Pairwise Diff is not chosen, likely because ensemble penalties are better correlated with true dynamics error, and are more stable under tuning since their scaling changes less with model number; we know that Max Pairwise Diff has very similar shape statistics to Ensemble Std. (App. A.3). Finally, we also observe these solutions have lower performance variance than MOPO.

The selection of Max Aleatoric is also explainable; we observe it displays significantly lower skew and kurtosis than all other metrics (App. A.3), while still maintaining strong rank correlation. We also found that in all Hopper experiments, Ensemble Var. never achieved high performance, despite the only difference with Ensemble Std. being its distributional shape. Interestingly, in HalfCheetah, the opposite is true, with Ensemble Var. delivering significant performance gains. This implies that distributional shape may play as important a role as calibration, and advocates the learning of *meta-parameters* that control for this. Finally, in Walker2d, the well-grounded ensemble penalties win in all cases. We note that values of the rollout horizon h and penalty weight λ differ greatly from those chosen in the original MOPO paper, which chooses both from $\{1, 5\}$. Notably, the Hopper and Walker2d environments can prefer a much longer rollout length and higher penalty weight, even accounting for penalty magnitudes. Again this is backed up by our analysis; along a single rollout, dynamics errors do not necessarily accumulate, they simply become more likely to occur. Therefore, as long as we penalize errors

appropriately, we can handle longer rollouts and, as a result, generate more on-policy data. The number of models used to compute the uncertainty estimates can also differ greatly from the standard 7. This again aligns with our findings that using more models for uncertainty estimation can be beneficial, but is dependent on environment, data, and penalty.

Table 3.3 also demonstrates how these unconventional hyperparameter choices fare against state-of-the-art offline MBRL algorithms. We spent considerable effort ensuring that our implementation of MOPO matched the authors’ results using the same hyperparameters. We note the two are very similar¹, thereby allowing us to make a faithful comparison when modifying hyperparameters. Our approach, labeled “Optimized[†]”, achieves *statistically significant* improvements over MOPO on 9 out of 12 environments, validating our prior analysis over the key design choices. As an additional bonus, and it is not the stated aim of this work, our approach achieves state-of-the-art performance on five HalfCheetah and Walker2d environments by a considerable margin. Further notable results include the Hopper mixed and Hopper medium-expert environments, in which we show we are able to tune the MOPO-like method up to the performance of COMBO (Yu et al., 2021) and MOREL. The importance of good uncertainty estimation and hyperparameter selection is shown visually in Fig. 3.4 where we improve MOPO performance by over 5× whilst obtaining a stable solution.

As aforementioned, we found our policies are more stable than previous works and consequently *do not need to cherry-pick* high performing checkpoints². Instead, we report the average performance over our final 10 policy-improvement iterations. It should be noted that stability during training (Chan et al., 2020) is paramount for successful policy deployment in offline RL, and we should therefore prioritize hyperparameters that ensure this. We further confirm the reliability of our evaluation using the `rliable` framework (Agarwal et al., 2021) in Fig. 3.1b, showing that

¹There was a disparity in Walker2d-medium, but this was also noted in Ball et al. (2021)

²It is unclear what procedure is used in some prior work (indeed `issues` have been `raised` about this).

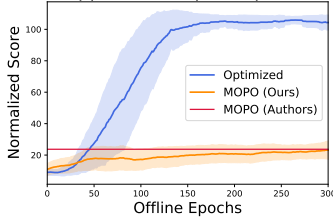


Figure 3.4: MOPO performance on the Hopper medium-expert environment.

Environment		MOPO [†]	Optimized [†]	CQL
pen	cloned	5.4 $\pm$ 10.8	23.0 $\pm$ 4.2	39.2
	human	6.2 $\pm$ 7.8	19.0 $\pm$ 7.9	37.5
	expert	15.1 $\pm$ 9.7	50.6 $\pm$ 10.5	107.0
hammer	cloned	0.2 $\pm$ 0.1	5.2 $\pm$ 1.5	2.1
	human	0.2 $\pm$ 0.0	0.5 $\pm$ 0.8	4.4
	expert	6.2 $\pm$ 8.4	23.3 $\pm$ 4.1	86.7

Table 3.4: Comparative evaluation on the D4RL Adroit v0 dataset against Model Free CQL

the improvement over MOPO (with 95% bootstrap CIs shaded) is clear in both MuJoCo and Adroit.

Results on Adroit dexterous hand manipulation tasks. We present results in Table 3.4 on the Adroit Pen and Hammer environments which, as far as we are aware, *have not previously been used in offline MBRL*, and present very different challenges to the locomotion tasks. These tasks feature sparse rewards, real human demonstrations and narrow data distributions. We compare against the current state-of-the-art *model-free* algorithm (CQL, Kumar et al. (2020)) and find that offline MBRL can learn useful policies in the Adroit domains, providing the best performance seen so far on the hammer-cloned setting. Best found penalties and hyperparameters are listed in App. A.10, and mirror the findings in the locomotion experiments. We believe issues with the world model not accurately capturing sparse rewards may account for any major performance difference. Our work is therefore an important step towards bridging the gap between model-based and model-free methods for sparse reward tasks, especially in the offline setting where exploration is not possible. We define MOPO to be the best performance with the Max Aleatoric penalty, searching λ , h in $\{1, 5\}^2$.

3.1.7 Conclusion

In this paper, we rigorously evaluated the impact of various key design choices on offline MBRL, comparing for the first time a number of different uncertainty penalties used in the literature. By proposing novel evaluation protocols, we have also gained key insights into the nature of uncertainty in offline MBRL that we believe benefits the RL community. We demonstrated the impact of this analysis

by significantly improving upon existing offline MBRL by using vastly different key hyperparameters, obtaining statistically significant performance improvements in almost all benchmarks.

Going forward, we are excited by developments in offline evaluation (Chen et al., 2021; Fu et al., 2021b) to accurately assess agent performance without querying the environment. This would open the door for population-based training methods (Jaderberg et al., 2017; Parker-Holder et al., 2020), which have shown great success in online MBRL (Zhang et al., 2021). Furthermore, throughout the paper we have highlighted potential areas of interest, from better understanding the generalization provided by world models, through to the development of meta-parameters controlling penalty distribution shape. We also highlight key issues in implementation in App. A.6 as we strongly believe this is a vital frontier for disentangling the effect of algorithmic innovations from code-level details. Finally, Offline MBRL so far has only focused on deterministic environments; the ensemble penalties we investigate support the modeling of stochastic dynamics, and the novel tools for analysis we develop here can be readily applied to such settings.

3.1.8 Errata

Mackay (1992) is referred to as an example of a variational Bayesian approach to modeling deep neural networks in Section 3.1.2. This is in fact incorrect; the work in question uses a Laplace approximation with the Hessian of the parameters to approximate the posterior distribution of the neural network weights.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Revisiting Design Choices in Offline Model-Based Reinforcement Learning
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Cong Lu*, Philip Ball*, Jack Parker-Holder, Michael Osborne, Stephen J. Roberts, Revisiting Design Choices in Offline Model Based Reinforcement Learning, <i>International Conference on Learning Representations 2022</i>

Student Confirmation

Student Name:	Philip Ball		
Contribution to the Paper	Proposed the research topic, wrote the offline model-based RL codebase from scratch, majority writing of paper, performed all analysis (and did analysis plots), proposed Bayesian Optimization approach.		
Signature 	Date	05/Oct/2023	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter Bibliography

- Zaheer Abbas, Samuel Sokota, Erin Talvitie, and Martha White. Selective Dyna-style planning under limited model capacity. In *ICML*, pp. 1–10, 2020. URL <http://proceedings.mlr.press/v119/abbas20a.html>.
- Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.
- Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters for on-policy deep actor-critic methods? a large-scale study. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=nIAxjsniDzg>.
- Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=0MNB1G5xzd4>.
- Philip Ball, Jack Parker-Holder, Aldo Pacchiano, Krzysztof Choromanski, and Stephen Roberts. Ready policy one: World building through active learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML*. 2020.
- Philip J Ball, Cong Lu, Jack Parker-Holder, and Stephen Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 619–629. PMLR, Jul 2021. URL <http://proceedings.mlr.press/v139/ball21a.html>.
- Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1613–1622, Lille, France, 07–09 Jul 2015. PMLR. URL <http://proceedings.mlr.press/v37/blundell15.html>.
- Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *Advances in Neural Information Processing Systems*. 07 2018.
- Stephanie C.Y. Chan, Samuel Fishman, Anoop Korattikara, John Canny, and Sergio Guadarrama. Measuring the reliability of reinforcement learning algorithms. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=SJlpYJBKvH>.
- Yutian Chen, Liyuan Xu, Caglar Gulcehre, Tom Le Paine, Arthur Gretton, Nando de Freitas, and Arnaud Doucet. On instrumental variable regression for deep offline policy evaluation, 2021.

- Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems 31*, pp. 4754–4765. 2018.
- Alexander Cowen-Rivers, Daniel Palenicek, Vincent Moens, Mohammed Abdullah, Aivar Sootla, Jun Wang, and Haitham Bou Ammar. SAMBA: safe model-based & active reinforcement learning. *Machine Learning*, pp. 1–31, 01 2022. doi: 10.1007/s10994-021-06103-6.
- Robert Dadashi, Leonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal Wasserstein imitation learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=TtYSU29zgR>.
- Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 465–472, 2011.
- Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, pp. 1–50, 2021.
- Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020.
- Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(18):503–556, 2005. URL <http://jmlr.org/papers/v6/ernst05a.html>.
- Angelos Filos, Sebastian Farquhar, Aidan N. Gomez, Tim G. J. Rudner, Zachary Kenton, Lewis Smith, Milad Alizadeh, Arnoud de Kroon, and Yarin Gal. A systematic comparison of Bayesian deep learning robustness in diabetic retinopathy tasks, 2019.
- Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2021a.
- Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, Ziyu Wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, Sergey Levine, and Thomas Paine. Benchmarks for deep off-policy evaluation. In *International Conference on Learning Representations*, 2021b. URL <https://openreview.net/forum?id=kWSeGEeHvF8>.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 06–11 Aug 2017. URL <http://proceedings.mlr.press/v70/guo17a.html>.
- Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.

- Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M. Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, Chrisantha Fernando, and Koray Kavukcuoglu. Population based training of neural networks, 2017.
- Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*. 2019.
- Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOREL : Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*. 2020.
- Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2796–2804. PMLR, 10–15 Jul 2018. URL <http://proceedings.mlr.press/v80/kuleshov18a.html>.
- Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*. 2020.
- Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018.
- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS’17, pp. 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.
- David John Cameron Mackay. *Bayesian Methods for Adaptive Models*. PhD thesis, USA, 1992. UMI Order No. GAX92-32200.
- Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL <https://proceedings.neurips.cc/paper/2019/file/118921efba23fc329e6560b27861f0c2-Paper.pdf>.
- Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=3hGNqpI4WS>.
- Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites, 2015.

- D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pp. 55–60 vol.1, 1994.
- Muhammad Omer, Rami Ahmed, Benjamin Rosman, and Sharief F. Babikir. Model predictive-actor critic reinforcement learning for dexterous manipulation. In *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEE)*, pp. 1–6, 2021. doi: 10.1109/ICCCEE49695.2021.9429677.
- Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL <https://proceedings.neurips.cc/paper/2019/file/8558cb408c1d76621371888657d2eb1d-Paper.pdf>.
- Aldo Pacchiano, Philip Ball, Jack Parker-Holder, Krzysztof Choromanski, and Stephen Roberts. Towards tractable optimism in model-based reinforcement learning. In *Uncertainty in Artificial Intelligence*. 2021.
- Tom Le Paine, Cosmin Paduraru, Andrea Michi, Çağlar Gülçehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *CoRR*, abs/2007.09055, 2020. URL <https://arxiv.org/abs/2007.09055>.
- Feiyang Pan, Jia He, Dandan Tu, and Qing He. Trust the model when it is confident: Masked model-based actor-critic. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 10537–10546. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/77133be2e96a577bd4794928976d2ae2-Paper.pdf>.
- Jack Parker-Holder, Vu Nguyen, and Stephen J Roberts. Provably efficient online hyperparameter optimization with population-based bandits. In *Advances in Neural Information Processing Systems*, volume 33, pp. 17200–17211. Curran Associates, Inc., 2020.
- Luis Pineda, Brandon Amos, Amy Zhang, Nathan O. Lambert, and Roberto Calandra. MBRL-Lib: A modular library for model-based reinforcement learning. *Arxiv*, 2021. URL <https://arxiv.org/abs/2104.10159>.
- Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Offline Reinforcement Learning Workshop at Neural Information Processing Systems*, 2020.
- Gabriele Scalia, Colin A Grambow, Barbara Pernici, Yi-Pei Li, and William H Green. Evaluating scalable uncertainty estimation methods for deep learning-based molecular property prediction. *Journal of Chemical Information and Modeling*, 60(6):2697–2717, 2020.

- Jian Shen, Han Zhao, Weinan Zhang, and Yong Yu. Model-based policy optimization with unsupervised model adaptation. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2823–2834. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/1dc3a89d0d440ba31729b0ba74b93a33-Paper.pdf>.
- Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bull.*, 2(4):160–163, July 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL <https://doi.org/10.1145/122344.122377>.
- Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.
- Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9690–9700. PMLR, 13–18 Jul 2020.
- Xingchen Wan, Vu Nguyen, Huong Ha, Binxin Ru, Cong Lu, and Michael A. Osborne. Think global and act local: Bayesian optimisation over high-dimensional categorical and mixed search spaces. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10663–10674. PMLR, Jul 2021.
- Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. In *To Appear: The International Conference on Learning Representations (ICLR)*. 2021.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*. 2020.
- Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2021.
- Baohe Zhang, Raghu Rajan, Luis Pineda, Nathan Lambert, André Biedenkapp, Kurtland Chua, Frank Hutter, and Roberto Calandra. On the importance of hyperparameter optimization for model-based reinforcement learning. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, 2021.

4

Efficient Online Adaption to Environments

Contents

4.1 Ready Policy One: World Building Through Active Learning	52
4.1.1 Introduction	52
4.1.2 Related Work	54
4.1.3 Background	57
4.1.4 Ready Policy One	61
4.1.5 Experiments	67
4.1.6 Conclusion and Future Work	71
4.1.7 Errata	72
4.2 Stabilizing Off-Policy Deep Reinforcement Learning from Pixels	74
4.2.1 Introduction	74
4.2.2 Background	77
4.2.3 Instabilities in TD-Learning from Pixels	78
4.2.4 Counteracting Gradient Discontinuities	86
4.2.5 Performance Evaluation	90
4.2.6 Related Work	94
4.2.7 Conclusion	95
Chapter Bibliography	97

In this chapter, we focus on the problem of online reinforcement learning, and specifically how to improve the sample-efficiency of algorithms by improving their ability to adapt to new information gathered during exploration.

We specifically focus on ways of improving how quickly agents can respond to

information gathered during environment exploration. Agents that cannot adapt their behavior quickly based on new information may waste valuable time acquiring redundant and potentially expensive samples. We propose two ways of addressing this; first we specifically encourage agents to continually gather novel data over their entire lifetime, and secondly we discover and ameliorate a key issue with TD-learning from pixels that causes agents to get stuck in local minima.

4.1 Ready Policy One: World Building Through Active Learning

Abstract Model-Based Reinforcement Learning (MBRL) offers a promising direction for sample efficient learning, often achieving state of the art results for continuous control tasks. However many existing MBRL methods rely on combining greedy policies with exploration heuristics, and even those which utilize principled exploration bonuses construct dual objectives in an ad hoc fashion. In this paper we introduce Ready Policy One (RP1), a framework that views MBRL as an active learning problem, where we aim to improve the world model in the fewest samples possible. RP1 achieves this by utilizing a hybrid objective function, which crucially adapts during optimization, allowing the algorithm to trade off reward v.s. exploration at different stages of learning. In addition, we introduce a principled mechanism to terminate sample collection once we have a rich enough trajectory batch to improve the model. We rigorously evaluate our method on a variety of continuous control tasks, and demonstrate statistically significant gains over existing approaches.

4.1.1 Introduction

Reinforcement Learning (RL) considers the problem of an agent learning to construct of actions that result in an agent receiving high rewards in a given environment. This can be achieved in various ways such as: learning an explicit mapping (*a policy*) from states to actions that maximizes expected return (policy gradients),

or inferring such a mapping by calculating the expected return for a given state-action pair (TD-control methods). Model-Based Reinforcement Learning (MBRL) seeks to improve the above by learning a model of the dynamics (from agent’s interactions with the environment) that can be leveraged across many different tasks (transferability) and for planning, which is substantially less expensive than a real environment (which plays crucial role in robotics applications).

A series of recent work (Ha & Schmidhuber, 2018; Hafner et al., 2019; Kurutach et al., 2018; Clavera et al., 2018; Chua et al., 2018; Kaiser et al., 2020; Schrittwieser et al., 2019) illustrate the benefits of MBRL-approaches that allow us to decouple learning task-dependent policy and task-agnostic dynamics. With recent advances, MBRL approaches often outperform model-free methods (Wang et al., 2019). However, these results are often overly sensitive to heuristics.

In particular, many of these methods lack a principled mechanism to acquire data for training the model. This issue is circumvented in Ha & Schmidhuber (2018), since they only consider environments which can be explored with random policies. Other model-based approaches, such as Kurutach et al. (2018); Feinberg et al. (2018); Hafner et al. (2020), rely on stochastic policies to aid exploration, and inevitably acquire redundant data which reduces sample efficiency. Such issues have been highlighted previously (Schmidhuber, 1991; Sun et al., 2011), and motivate the design of our algorithm. Concretely, we reduce the cost incurred from data collection by using active learning methods, and introduce an early stopping mechanism to address the issue of redundancy.

Efficient exploration is a challenge for existing RL algorithms, and is a core focus in model-free RL (Chentanez et al., 2005; Houthoofd et al., 2016; Brafman & Tenenbholz, 2003; Lopes et al., 2012). Despite often considering a principled objective, these methods generally contain a fixed temperature parameter, thus requiring hand engineering to determine the optimal degree of exploration. Our approach adjusts this parameter in an online manner from the collected trajectories, and we provide an information theoretic motivation for our exploration objective.

In this paper, we introduce a novel approach to acquiring data for training world models through exploration. Our algorithm, Ready Policy One (RP1), includes principled mechanisms which acquire data for model-based RL through the lens of Online Active Learning. Crucially, we continue to jointly optimize our policies for both reward and model uncertainty reduction, since we wish to avoid focusing on stochastic or challenging regions of the state space which have no impact on the task at hand. Therefore policies used for data collection also perform well in the true environment, and means we can use these policies for evaluation. Consequently, a separate ‘exploit’ agent does not need to be trained (Henaff, 2019).

To summarize, our key contributions are as follows:

- Inspired by Active Learning, we train policies in a learned world model with the objective of acquiring data that most likely leads to subsequent improvement in the model.
- We introduce a novel early-stopping criteria for real-environment samples, reducing redundancy in expensive data collection.
- We adapt the objective function as learning progresses using an Online Learning mechanism.

The paper is structured as follows. In Section 4.1.2 we discuss related work. In Section 4.1.3 we describe our RL setting and introduce basic concepts. In Section 4.1.4 we introduce our method and related theory. Finally we demonstrate the effectiveness of our approach across a variety of continuous control tasks in Section 4.1.5 before concluding in Section 4.1.6, where we also mention some exciting future work.

4.1.2 Related Work

The Dyna algorithm (Sutton, 1991) is a canonical approach for model based reinforcement learning (MBRL), based on the principle of ‘trying things in your head’, using an internal model of the world. In its original form, Dyna contained an exploration bonus for each state-action pair, proportional to this uncertainty

measure (Sutton, 1990). A decade later, principled approaches were proposed for exploration (Brafman & Tenenbholz, 2003), yet their guarantees were only possible in discrete settings with a finite number of states.

In recent times there has been great deal of progress in MBRL, with success in Atari games (Kaiser et al., 2020; Schrittwieser et al., 2019), and dexterous manipulation (Nagabandi et al., 2019), while progress has been made on continuous control benchmarks (Kurutach et al., 2018; Chua et al., 2018; Clavera et al., 2018; Janner et al., 2019). We note that our approach is orthogonal to these in the following ways: 1) our methods, or some subset of the methods we introduce, can be incorporated into existing MBRL algorithms; 2) we adhere to a strict Dyna style framework, and our methods are aimed at incorporating active learning into this paradigm. Other recent work (Ha & Schmidhuber, 2018) shows that Dyna can be extended to latent-state dynamics models. We note that our framework could be extended to this setting, however we instead focus on efficient data acquisition through active learning.

Active approaches (i.e., acquiring data in a principled manner) in MBRL have been considered previously, for example in Shyam et al. (2019); Pathak et al. (2019); Henaff (2019). Usually, ensembles of models are maintained, and an intrinsic reward, defined as some difference measure (i.e., KL-divergence, total variation) across the output of different models in the ensemble drives exploration. Such exploration might be ineffective however, as the policy may visit regions of the state space which have no relation to solving the task. This may also lead to unsafe exploration if deployed on a real robot. Henaff (2019) bears similarities to our work in that it aims to improve model generalization through exploration, and has a criteria to trade-off exploration and exploitation. However their approach to exploration is purely novelty-seeking, and they collect data until they discover a model (or subset of models) that can fully model the MDP in question (i.e., when novelty falls below a predefined value). Once a model is discovered, they implement a policy (through search) which exploits it to maximize performance. This has

drawbacks mentioned above concerning wasted exploration in task-irrelevant states as well as unsafe exploration.

Also similar to our work is [Akiyama et al. \(2010\)](#), who explicitly use an active learning approach for solving an RL task; a robot arm hitting a ball. Our work differs in three significant ways: 1) our policies are trained inside a model, not on the true environment; 2) we actively limit the number of trajectories per collection phase based on the data (they introduce a heuristic); 3) we do not have access to an off-line measure of generalization, and therefore must introduce an online-learning mechanism to seek out the optimal setting of the exploration parameter.

Approaches that produce Bayes-optimal exploration and exploitation with a model ([Deisenroth & Rasmussen, 2011](#); [Sajid et al., 2021](#)) are also of relevance, however these methodologies do not scale well to high dimensional tasks ([Wang et al., 2019](#)).

Efficient exploration in environments with very sparse rewards also represents a relevant area of research. In such settings an agent follows its *curiosity*, quantified by either: 1) rewarding areas of the state-space that reduce uncertainty in some internal model (i.e., inverse or forward dynamics models) ([Chentanez et al., 2005](#); [Mohamed & Rezende, 2015](#); [Houthoofd et al., 2016](#); [Pathak et al., 2017](#); [Burda et al., 2019](#)); 2) rewarding un-visited areas of the state-space ([Brafman & Tennenholtz, 2003](#); [Lopes et al., 2012](#); [Ostrovski et al., 2017](#)). Our approach to exploration leverages a model ensemble, and sits in the former category.

There has also been a great deal of work on using maximum entropy principles as a means for exploration in model-free RL ([Haarnoja et al., 2018a](#)). The aim is to find rewarding behavior whilst maximizing some measure of entropy. We differ from these works in both what entropy is maximized (action entropy v.s. model prediction entropy) and by not having task-specific, fixed temperature parameter that trades off reward and entropy/surprise. In later maximum entropy work, temperature selection has been formulated as a constrained optimization problem, such that performance is maximized subject to some minimum level of policy entropy

(Haarnoja et al., 2018b). In contrast, we select this parameter in an online manner that optimally improves our internal models.

4.1.3 Background

RL Policies & Markov Decision Processes

A Markov Decision Process (MDP, (Bellman, 1957)) is a tuple $(\mathcal{S}, \mathcal{A}, P, R)$. Here $\mathcal{S}$ and $\mathcal{A}$ stand for the sets of states and actions respectively, such that for $s_t, s_{t+1} \in \mathcal{S}$ and $a_t \in \mathcal{A}$: $P(s_{t+1}|s_t, a_t)$ is the probability that the system/agent transitions from s_t to s_{t+1} given action a_t and $R(a_t, s_t, s_{t+1})$ is a reward obtained by an agent transitioning from s_t to s_{t+1} via a_t .

A policy $\pi_\theta : \mathcal{S} \rightarrow \mathcal{A}$ is a (possibly randomized) mapping (parameterized by $\theta \in \mathbb{R}^d$, e.g. weight of the neural network) from $\mathcal{S}$ to $\mathcal{A}$. Policy learning is the task of optimizing parameters θ of π_θ such that an agent applying it in the environment given by a fixed MDP maximizes total (expected/discounted) reward over given horizon H . In this paper we consider MDPs with finite horizons.

In most practical applications the MDP is not known to the learner. In MBRL, we seek to use a dataset $\mathcal{D} = \{(s_t, a_t), s_{t+1}\}_{t=1}^N$ of observed transitions to train a dynamics/world model $\hat{f}_\phi$ parameterized by ϕ to approximate the true dynamics function $f(s_{t+1}|s_t, a_t)$ such that $\hat{f}_\phi(s_{t+1}|s_t, a_t) \cong f(s_{t+1}|s_t, a_t)$.

We aim to construct rich $\mathcal{D}$ s for learning accurate enough models $\hat{f}_\theta$, but only in those regions that are critical for training performant policies.

Sequential Model Based Optimization

Consider a black box function $F : \mathbb{R}^d \rightarrow \mathbb{R}$ over some domain $\mathcal{X}$, whereby the goal is to find $x^* \in \mathcal{X}$ such that

$$x^* = \arg \max_{x \in \mathcal{X}} F(x) \quad (4.1)$$

Sequential Model Based Optimization (SMBO (Hutter et al., 2011)) is a model-based black box optimization method which seeks to learn a surrogate model $\hat{F}$, within the true model F . Using the surrogate model, it is possible to determine

which data should be collected to discover the optimum point of the real black box function F . The surrogate model is sequentially updated with the data collected in order to obtain better estimates of the true F , and this process is repeated until convergence or limited to a set number of iterations.

Many MBRL algorithms follow this regime, by trying to model a true black box function F using a world model $\hat{f}_\phi$ parameterized by ϕ as follows:

$$F(\theta) \cong \sum_{t=0}^H \gamma_t R(s_t, \pi_\theta(s_t), \hat{f}_\phi(\pi_\theta(s_t), s_t)) \quad (4.2)$$

where $\gamma_t \in [0, 1]$ is a discount factor, actions are $\pi_\theta(s_t) = a_t$, and next states are generated by $s_{t+1} = \hat{f}_\phi(\pi_\theta(s_t), s_t)$. Subsequently, they seek to find a policy π parameterized by θ that maximizes the acquisition function, which is just $\max F(\theta)$, then they will generate rollouts in the environment by using this policy and adding noise to the actions, such as via a stochastic policy.

Taking this view, essentially the majority of existing MBRL algorithms are conducting a form of SMBO with a greedy acquisition function. Any exploration conducted is simply a post hoc addition of noise in the hope of injecting sufficient stochasticity, but there is no principled mechanism by which the model may escape local minima.

Our approach is markedly different. We seek to conduct *Active Learning*, whereby the goal is not just to find an optimal solution to F , but to learn an optimal surrogate model $\hat{F}^*$ through the world model $\hat{f}_{\phi^*}$. To do this, we train policies to maximize an acquisition function that trades-off reward and information, rather than just greedily maximizing reward. Given how the acquisition function in SMBO is an important component for finding optimal performance (Jones et al., 1998), it therefore makes sense to carefully design how we acquire data in the MBRL setting.

Active Learning

Active Learning considers the problem of choosing new data $\mathcal{D}' = \{x'_i, y'_i\}_{i=1}^m$, which is a subset of a larger dataset i.e., $\mathcal{D}' \in \mathcal{D} = \{x_i, y_i\}_{i=1}^n$, such that a model $\mathcal{M}$ is most improved.

In traditional supervised learning, we usually have access to the entirety of $\mathcal{D}$ for training, but in the active learning setting we only have access to its feature vectors $\{x_i\}_{i=1}^n$, and need to query an oracle to obtain the corresponding labels $\{y_i\}_{i=1}^n$ which incurs a cost. Active learning aims to reduce this cost by iterating through the set of unlabeled feature vectors $\{x_i\}_{i=1}^n$, and determining which subset $\{x'_i\}_{i=1}^m$ would produce the greatest improvement in the performance of $\mathcal{M}$ should we train our models on the subset $\mathcal{D}' = \{x'_i, y'_i\}_{i=1}^m$. The ultimate goal is to achieve the highest performance with the fewest number of queries (i.e., at the lowest cost).

In reinforcement learning, there is a key difference to the standard active learning setting; we do not have direct access to a dataset $\mathcal{D}$, and must instead generate $\mathcal{D}$ through placing a ‘sampling’ policy π_s in the environment, producing trajectories. We then assess these trajectories generated by π_s and determine how to best train this policy to obtain trajectories in some oracle/true environment that will benefit performance. Framing the problem this way is done in [Akiyama et al. \(2010\)](#), where it is cheap to generate and evaluate trajectories, but expensive to obtain labels.

In the Dyna-style approach ([Sutton, 1991](#)) that we adopt, this means training our sampling policy π_s in the world model exclusively, then using this policy to collect samples in the real world. Through the lens of active learning, it is therefore important to train π_s in our world model such that the resultant trajectories maximize our world model performance, therefore maximizing our final policy performance. This is because it is free, from a data collection perspective, to train in the world model, but acquiring data in the real environment incurs an expense.

Concretely, we wish to train a policy that performs well in the real environment in as few samples as possible, and we identify that having a robust and generalizable world model as being paramount to achieving this. We observe that state-action pairs which cause high uncertainty/disagreement in the world model are likely to be parts of the state space that our world model is poor at modeling (as observed in [Freeman et al. \(2019\)](#), where solving ‘difficult’ environment dynamics is important for learning optimal policies).

However we cannot simply target regions of disagreement; we are primarily concerned with maximizing policy performance, so wish to explore regions of disagreement that are also critical to solving the task. This becomes a classic explore/exploit dilemma; how much do we want our sampling policy to explore (i.e., find states with poor generalization) or exploit (i.e., visit known high value states) when acquiring new real samples. In order to manage this trade-off, we leverage an online active learning approach similar to [Osugi et al. \(2005\)](#), where model generalization feedback from the gathered trajectories can be used to determine the degree to which we explore or exploit in the future.

Online Learning

Online Learning is a family of methods that is used when a learner tackles a decision-making task by taking actions $a \in \mathcal{A}$, whilst learning from a sequence of data $z_1, z_2, \dots, z_N$ that arrive in incremental rounds n . The learner must take an action at the beginning of each round, and the aim is to minimize cumulative regret, which is usually defined as the difference in some loss $\ell(a, z_n)$ between the actual action taken and the optimal action that could have been taken at that round:

$$\sum_{n=0}^N \left(\ell(a_n, z_n) - \min_{a \in \mathcal{A}} \ell(a, z_n) \right).$$

At each round n the learner does not know what the consequence of some action a will be, and only receives feedback after submitting its chosen action a_n . Based on this feedback the learner then updates how it selects actions in the future. In our approach we consider the set of actions $\mathcal{A}$ to be the degree to which our policy explores, and the loss ℓ to be a normalized generalization error from the data collected. The aim is to therefore ensure that generalization error (i.e., RMSE on the new data) is maximized at each round. Since the task of maximizing generalization is both noisy and stochastic (i.e., the optimal degree of exploration may vary as we collect data), careful design of this algorithm is required.

4.1.4 Ready Policy One

Here we introduce our main algorithm, *Ready Policy One* (RP1). The key differences between RP1 and existing state of the art MBRL methods are as follows:

1. By taking an Active Learning approach rather than focusing on greedy optimization, RP1 seeks to directly learn the best *model*, rather than learning the best *policy*, and indirectly learning the best model to achieve this objective.
2. We introduce a principled Online Learning-inspired framework, allowing RP1 to adapt the level of exploration in order to optimally improve the model in the fewest number of samples possible.
3. We introduce a mechanism to stop gathering new samples in any given collection phase when the incoming data resembles what we have already acquired during that phase.

The algorithm begins in a similar fashion to other MBRL methods, by sampling initial transitions with a randomly initialized policy. In the Dyna framework, a policy is then trained inside $\hat{f}_\phi$, and then subsequently used to gather new data. Typically, random noise is added to the policy to induce exploration. Other methods consider a hybrid objective. In RP1, consider training a *sampling policy*, parameterized by θ_t , to optimize the following objective:

$$\pi_{\theta_t} = \max[\mathbb{E}_{\tau \sim \pi_{\theta_t}} [(1 - \lambda)\mathcal{R}(\tau) + \lambda\sigma(\mathcal{R}(\tau))]] \quad (4.3)$$

where $\mathcal{R}(\tau) = \sum_{i=0}^H r_i$, $r_i = R(s_i, a_i, s_{i+1})$ and $\sigma(\mathcal{R}(\tau)) = \sum_{i=1}^H \sqrt{\frac{\sum_{j=1}^M (r_i^j - \bar{r}_i)^2}{M-1}}$. This λ value is chosen before training the policy in the model, and is selected using an online learning algorithm mechanism detailed in 4.1.4. Hence λ defines the relative weighting of reward and reward variance, with $\lambda = 0$ training a policy that only maximizes expected return (model-guided exploitation)¹, and $\lambda = 1$ training a policy that only maximizes variance/disagreement per time step (model-guided

¹ $\lambda = 0$ corresponds to the same objective as in prior MBRL work, such as Kurutach et al. (2018).

exploration). In reality we limit λ to $[0.0, 0.5]$ as we wish any exploration to be guided by the reward signal. As a consequence, there is no need to train a separate ‘exploit’ policy, since we find policies trained in this way provide significant gains over commonly used approaches. This mirrors the fact that MaxEnt strategies obtain high-performance in deterministic environments (Eysenbach & Levine, 2019).

Information Gain in the Model as Maximum Entropy

Our objective in Equation 4.3 is inspired by curiosity driven exploration via model disagreement (Pathak et al., 2019). When shown a new unlabeled feature x' comprising a state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$ pair, the model has an existing ‘prior’ $p(r|x', \mathcal{D})$ where r is the predicted reward and $\mathcal{D}$ is the data seen so far. After obtaining the label s' we have a new datapoint D' (i.e., $D' = (s, a, s')$), and can update the world model, producing a ‘posterior’ $p(r|s, a, \{\mathcal{D} \cup D'\})$. In our case, reward is a deterministic function of this triple (i.e., $r = R(D')$, see Appendix B.2 for details). As such, we define the *Information Gain* (IG) in the reward as the KL divergence between the model posterior after observing D' and its’ respective prior at x' , as follows:

$$\mathcal{IG}(r; x') = D_{\text{KL}}[p(r|x', \{\mathcal{D} \cup D'\}) || p(r|x', \mathcal{D})] \quad (4.4)$$

$$= \int p(r|x', \{\mathcal{D} \cup D'\}) \log \frac{p(r|x', \{\mathcal{D} \cup D'\})}{p(r|x', \mathcal{D})} dr. \quad (4.5)$$

We observe that to maximize $\mathcal{IG}(r; x')$, we must sample x' appropriately, which is our only degree freedom (through policy training). Because we cannot tractably calculate this quantity for all D' , one approach to maximize information gain is to ensure that the prior assigns low mass to all regions of r (i.e., minimize the denominator over all r). In order to do this we would select the improper prior over the continuous variable r ². In our setting however, the model takes the form of an empirical Gaussian distribution, formed from the sample mean and variance of

²This follows the proposal in MacKay (2002) when selecting priors in the face of uncertainty.

the individual models in the ensemble. Therefore we would like to choose $p(r|x', \mathcal{D})$ such that the following is minimized:

$$D_{\text{KL}}[p(r|x', \mathcal{D})|p_0(r)] \quad (4.6)$$

where $p_0(r)$ is an improper prior. The only way that Equation 4.6 is minimized is when the differential entropy of $p(r|x', \mathcal{D})$ is maximized. The differential entropy of a Gaussian is well known, given by $h(x) = \ln(\sqrt{2\pi}\sigma) + \frac{1}{2}$. Therefore to maximize the entropy of $p(r|x', \mathcal{D})$, and maximize information gain in the face of uncertainty, we need to maximize its variance. This is achieved by training policies that generate trajectory tuples $x' = (s, a)$ which cause high variance in the model, and is analogous to information based acquisition functions in active learning (MacKay, 1992).

This motivates the second term of Equation 4.3, where we show the objective function for our exploration policies. Essentially, we are seeking to maximize information gain *in the model* through maximizing model entropy over the reward. This is in contrast to other maximum entropy approaches, which seek to maximize entropy over the action distribution (Haarnoja et al., 2018a), aiming to succeed at the task while acting as randomly as possible.

It is also possible to maximize information gain for next state predictions (as opposed to rewards), and this is similar to the approach in Pathak et al. (2019). However in practice we find that maximizing reward variance results in better performance (see Section 4.1.5).

Online Learning Mechanism

We use the Exponential Weights framework to derive an adaptive algorithm for the selection of λ . In this setup we consider k experts making recommendations at the beginning of each round. After sampling a decision $i_t \in \{1, \dots, k\}$ from a distribution $\mathbf{p}^t \in \Delta_k$ with the form $\mathbf{p}^t(i) \propto \exp(\ell_t(i))$ the learner experiences a loss $l_{i_t}^t \in \mathbb{R}$. The distribution $\mathbf{p}^t$ is updated by updating ℓ_t as follows:

$$\ell_{t+1}(i) = \begin{cases} \ell_t(i) + \eta \frac{l_{i_t}^t}{\mathbf{p}^t(i)} & \text{if } i = i_t \\ \ell_t(i) & \text{o.w.} \end{cases} \quad (4.7)$$

For some step size parameter η .

In our case we consider the case when the selection of λ_t is thought as choosing among k experts which we identify as the different values $\{\lambda_i\}_{i=1}^k$. The loss we consider is of the form $l_{i_t} = \hat{G}_{\phi_t}(\theta_{t+1})$, where $G_{\phi_t}(\theta_{t+1})$ is the RMSE of the model under parameters ϕ_t on data collected using $\pi_{\theta_{t+1}}$, and θ_{t+1} is the parameter of the policy trained under the choice λ_{i_t} , after incorporating into the model the data collected using the previous policy π_{θ_t} . We then perform a normalization of G (see Appendix B.2 for details), hence $\hat{G}$. Henceforth we denote by $\mathbf{p}_\lambda^t$ the exponential weights distribution over λ values at time t .

Our version of Exponential Weights algorithm also known as Exponentially Weighted Average Forecaster (Cesa-Bianchi & Lugosi, 2006) is outlined in Algorithm 1.

Algorithm 1 Online Learning Mechanism

Input: step size η , number of timesteps T .

Initialize: $\mathbf{p}_\lambda^1$ as a uniform distribution.

for $t = 1, \dots, T - 1$ **do**

1. Select $i_t \sim \mathbf{p}_\lambda^t$ and $\lambda_t = \lambda_{i_t}$.

2. Use Equation 4.7 to update $\mathbf{p}_\lambda^t$ with

$$l_{i_t}^t = \hat{G}_{\phi_t}(\theta_{t+1})$$

In practice and in order to promote more effective exploration over λ values we sample from a mixture distribution where $\mathbf{p}_\lambda^t$ is not proportional to $\exp(\ell_t)$ but it is a mixture between this exponential weights distribution and the uniform over $[k]$. In other words, let $\epsilon > 0$ be a small parameter. With probability $1 - \epsilon$ the produce i_t as a sample from an exponential weights distribution proportional to $\exp(\ell_t)$, and with probability ϵ it equals a uniform index from $1, \dots, k$.

Diverse Sample Collection

Consider the problem of online data acquisition from a policy in an environment. At each timestep we receive a set of datapoints $\{x_1, \dots, x_H\} \sim \pi_\theta$ corresponding to the concatenation of each state and action in a trajectory. At timestep t we have

a dataset $X_t = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, where $X_t \in \mathbb{R}^{d \times n}$ sampled from the sampling policy. We represent this data in the form of the Singular Value Decomposition (SVD) of the symmetric matrix, $\text{Cov}_t = \frac{1}{n} X_t X_t^\top = \mathbf{Q}_t^\top \Sigma_t \mathbf{Q}_t \in \mathbb{R}^d$.

Equipped with this representation, we take the top k eigenvalues λ_i of Cov_t , where k is smallest such that: $\sum_{i=1}^k \lambda_i \geq (1 - \delta) \sum_{i=1}^d \lambda_i$ for some parameter $\delta > 0$, and take $n_t = k$. Next we take the corresponding eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_k \in \mathbb{R}^d$ and let $\mathbf{U} \in \mathbb{R}^{d \times k}$ be obtained by stacking them together. We define the Active Subspace (Constantine et al., 2014) $\mathbf{U}^{\text{act}} \in \mathbb{R}^{d \times k}$ as $\mathcal{L}_{\text{active}} \stackrel{\text{def}}{=} \text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_k\}$. $\mathbf{U}^{\text{act}}$ is an orthonormal basis of $\mathcal{L}_{\text{active}}$.

We use $\mathbf{U}^{\text{act}}$ to evaluate new data. After we collect n' new samples $V_{t+1} \in \mathbb{R}^{d \times n'}$, we form a covariance matrix with this new data as $\frac{1}{n'} V_{t+1} V_{t+1}^\top \in \mathbb{R}^{d \times d}$ and project it onto $\mathbf{U}^{\text{act}}$. We define the residual at timestep t , r_t , as follows:

$$r_t = \frac{\text{tr}(V_{t+1} V_{t+1}^\top - \mathbf{U} \mathbf{U}^\top V_{t+1} V_{t+1}^\top \mathbf{U} \mathbf{U}^\top)}{\text{tr}(V_{t+1} V_{t+1}^\top)} \quad (4.8)$$

Where tr denotes the trace operator. After evaluating r_t we append the new data V_{t+1} to X_t to form $X_{t+1} \in \mathbb{R}^{d \times (n+n')}$. Intuitively, r_t tells us how much of the new data could not be explained by the principal components in the data collected thus far. We stop collecting data and proceed to retrain the model once $r_t < \alpha$, where α is a proxy for δ . The full procedure is presented in Algorithm 2.

Let q_t be the probability that at timestep t , step 4. of Algorithm 2 is executed (i.e., $q_t = \text{P}(r_t < \alpha)$). The evolution of q_t operates in roughly two phases. First, the algorithm tries to collect data to form an accurate estimate of the covariance matrix Cov_t and a stable estimator $\mathbf{U}^{\text{act}}$. During this phase, q_t is small as it is roughly the probability of two random samples $X, X' \in \mathbb{R}^{d \times n'}$ aligning. After t_0 steps when the algorithm has built a stable estimator of $\mathbf{U}^{\text{act}}$, the stopping probability stabilizes to a value q^* that solely depends on the trajectories intrinsic noise. Both the length of t_0 and the magnitude of q^* scale with the trajectories' noise. If the trajectories have little noise both t_0 and q^* are small. On the other hand, if the trajectories have high noise, the early stopping mechanism will take longer to trigger.

Algorithm 2 Early Stopping Mechanism**Input:** thresholds α, δ , maximum number of samples T .**Initialize:** training set $X = \emptyset$ Collect initial samples $X_1 = \{x_1, \dots, x_H\}$.**for** $t = 2, \dots, T - 1$ **do**

1. Compute Active Subspace $\mathbf{U}^{\text{act}}$ as the result of stacking together some orthonormal basis of $\mathcal{L}_{\text{active}} \stackrel{\text{def}}{=} \text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_r\}$ where the vectors $\mathbf{u}_i$ correspond to the top k eigenvalues of the covariance matrix Cov_t .
2. Produce samples V_{t+1} via the sampling policy
3. Calculate the residual r_t using Equation 4.8.
4. Stop collecting data if $r_t < \alpha$

This dynamic approach to determining the effective ‘batch size’ of the incoming labeled data is similar to [Chakraborty et al. \(2011\)](#), whereby feedback from unsupervised learning is used to control the amount of data collected per batch. However, we do this in a more instantaneous fashion, leveraging data collected so far to determine when to stop.

The Algorithm

We now present our algorithm: Ready Policy One (RP1). At each iteration we select a policy objective (λ) to maximize sample utility, and train a policy on this objective. The policies are trained using the original PPO ([Schulman et al., 2017](#)) loss function, but we use the training approach in ([Schulman et al., 2015](#)) as this combination delivered more robust policy updates.³

Once the policy has been trained inside the model, we use it to generate samples in the real environment. These samples continue until our early stopping mechanism is triggered, and we have sufficiently diverse data to retrain the model. The full procedure is outlined in Algorithm 3.

The overall aim is to therefore determine which part of the model space is both high value and unknown, so that our trained sampling policy can obtain enough data samples pertaining to those regions of the environment.

³Full implementation details can be found in Appendix B.2.

Algorithm 3 RP1: Ready Policy One

Input: Number of initial samples N_0 , number of ongoing samples N_t , number of policies in the ensemble M , number of time steps T .

Initialize: Initial World Model $\hat{f}_0$ comprised of M models.

Collect N_0 samples with a random policy and initialize data set $\mathcal{D}_0 = \{(s_t, a_t), s_{t+1}\}_{t=1}^{N_0}$.

for $t = 1, \dots, T - 1$ **do**

1. Train $\hat{f}_{t-1}$ with $\mathcal{D}_t$ to derive $\hat{f}_t$.
 2. Select λ using Algorithm 1.
 3. Train exploration policy π_{ϕ_t} using Equation 4.3 in $\hat{f}_t$.
 4. Collect new samples $\mathcal{D}_{new} = \{(s_t, a_t), s_{t+1}\}_{t=1}^{N_t}$ in the environment with π_{ϕ_t} , where N_t is defined as the number of time steps required for Algorithm 2 to return.
 5. $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup \mathcal{D}_{new}$
 6. Set $\alpha_t = \mathcal{L}(\hat{f}_t(\mathcal{D}_{new}))$
-

4.1.5 Experiments

The primary goal of our experiments is to evaluate whether our active learning approach for MBRL is more sample efficient than existing approaches. In particular, we test RP1 on a variety of continuous control tasks from the OpenAI Gym (Brockman et al., 2016), namely: HalfCheetah, Ant, Swimmer and Hopper, which are commonly used to test MBRL algorithms. For specifics, see Appendix B.2. In order to produce robust results, we run all experiments for ten random seeds, more than typically used for similar analyses (Henderson et al., 2017).

Rather than individual algorithms, we compare against the two approaches most commonly used in MBRL:

- **Greedy:** We train the policy to maximize reward in the model, and subsequently add noise to discover previously unseen states. This is the approach used in ME-TRPO (Kurutach et al., 2018).
- **Variance + Reward (V+R):** We train the policy with $\lambda = 0.5$, producing a fixed degree of priority for reward and model entropy. This resembles methods with exploration bonuses such as Houthoofd et al. (2016).

We note that these baselines are non-trivial. In particular, ME-TRPO is competitive with state of the art in MBRL. In fact, for two of the tasks considered (Swimmer and Hopper) it outperformed all other approaches in a recent paper benchmarking MRBL methods (Wang et al., 2019).

We also compare against the same policy gradients algorithm as a model free baseline, which we train for 10^6 , 5×10^6 and 10^7 timesteps. This provides an indication of the asymptotic performance of our policy, if trained in the true environment.

Table 4.1: Median best performance at a given timestep for ten seeds. Bold indicates the best performing algorithm. T1 corresponds to the t-stat for RP1 vs. Greedy, T2 corresponds to the t-stat for RP1 vs. V+R. * indicates $p < 0.05$.

	Timesteps	Greedy	V+R	RP1	T1	T2
HalfCheetah	10^4	-0.95	-1.1	100.51	5.39*	4.34*
Ant	10^4	94.72	95.07	113.63	4.02*	3.08*
Swimmer	10^4	1.08	1.07	3.24	1.2	1.98
Hopper	10^4	76.5	139.03	322.22	4.32*	3.42*
HalfCheetah	10^5	260.92	283.27	390.49	3.89*	2.62*
Ant	10^5	186.36	217.08	238.4	3.83*	3.11*
Swimmer	10^5	61.76	62.89	64.19	-0.11	-1.09
Hopper	10^5	487.25	570.09	619.73	3.52*	2.21

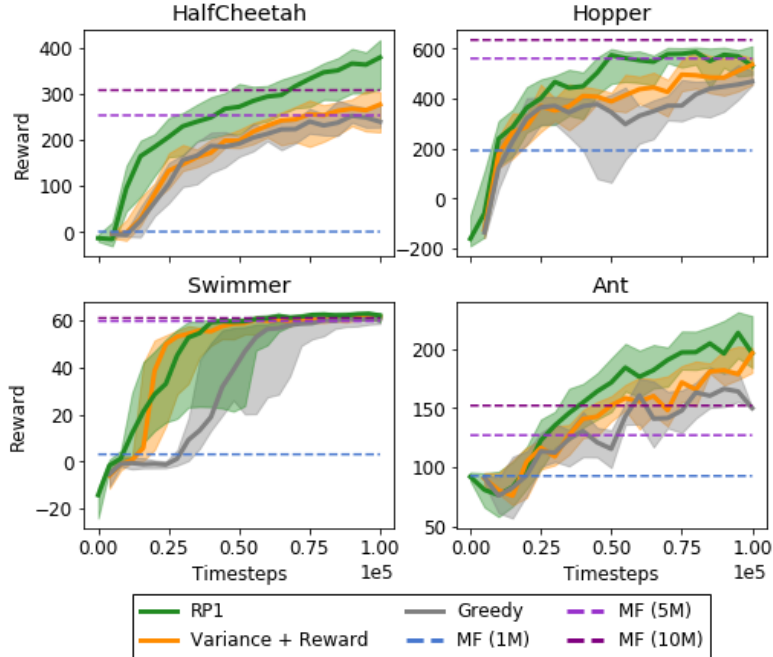


Figure 4.1: Median performance across 10 seeds. Shaded regions correspond to the Inter-Quartile Range.

Table 4.1 and Fig 4.1 show the main results, where RP1 outperforms both the greedy baseline and the fixed variance maximizing (V+R) approach. Furthermore, we perform Welch’s unequal variances t-test and see that in most cases the results are statistically significant, aside from Swimmer at 10^5 timesteps where all three methods have converged to the optimal solution. In addition, we observe that RP1 is able to achieve strong performance vs. model-free in fewer timesteps than the existing baselines.

Interestingly, we see that simply adding a fixed entropy term (V+R) into the reward function gives improved performance over the baseline greedy approach. This corroborates with findings in Freeman et al. (2019), where there is, in most tasks, a correlation between regions that are difficult to predict and regions of high reward. However our findings also suggest that this is not always the case, and having the ability to adjust how much we focus on such regions of disagreement is vital. We hypothesize that the fixed V+R approach may collect too many high-variance samples for certain tasks, since we do not tune λ , nor limit batch size. As a result, the trajectories gathered do not necessarily result in strong policy performance, unlike RP1, which aims to maximize policy performance through the data collected.

We support this hypothesis with Fig. 4.2 where, we show the normalized change in reward for different λ values in each task. In particular, we observe for HalfCheetah, Swimmer and Ant, a greater focus on uncertainty appears positively related to faster learning. However, the opposite is true for Hopper. The benefit of our mechanism is the ability to dynamically learn this preference, and thus adapt to the current optimization landscape.



Figure 4.2: Mean one-step policy improvement after a given λ value, for all ten seeds of RP1.

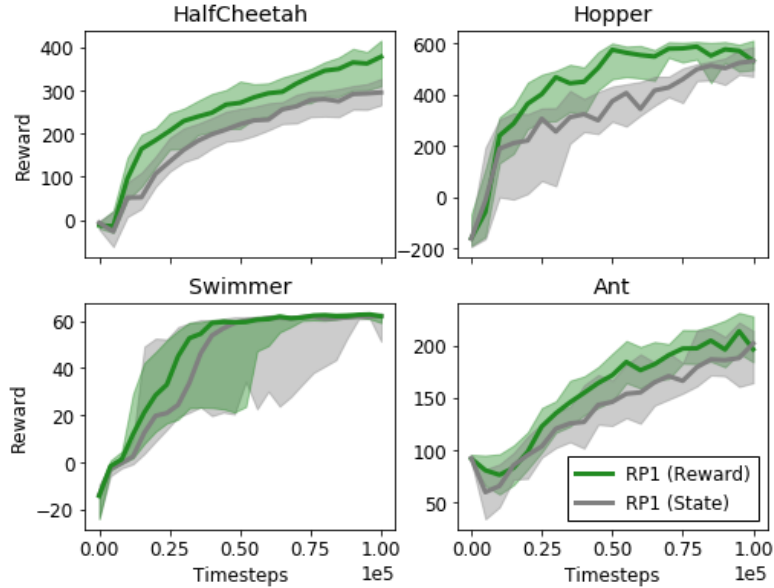


Figure 4.3: Median performance across 10 seeds. Shaded regions correspond to the Inter-Quartile Range.

Next we study the choice of model variance used in the reward function. Other work, such as Pathak et al. (2019) use the variance over the next state prediction, whereas RP1 uses the variance over the reward. Fig 4.3 and Table 4.2 show that next state variance is a less effective approach. This is likely due to over-emphasis of regions of the state space that are inherently hard to predict, but do not impact the ability to solve the task (Schmidhuber, 2010).

Table 4.2: Study for the choice of error to maximize. Results show the median best performance at 10^5 timesteps for ten seeds. The highest performing value for each environment is bolded.

	HalfCheetah	Ant	Swimmer	Hopper
State	319.41	214.33	63.47	549.19
Reward	390.49	238.4	64.19	619.73

Finally, we consider the individual components of the RP1 algorithm. We evaluate two variants: RP1 ($\lambda = 0$), where we remove the online learning mechanism and train a greedy policy, and RP1 (No EarlyStop), where we remove the early stopping mechanism and use a fixed batch size. Results are shown in Table 4.3, and Figs B.1 and B.2 in Appendix B.1.

We observe that the improvements attributed to RP1 are not down to any single design choice, and the individual components complement each other to provide significant overall gains. For example, by conducting purely noise-based exploration

Table 4.3: Ablation study for the key components of RP1. Results show the median performance at 10^5 timesteps for 10 seeds. The highest performing value for each environment is bolded.

	HalfCheetah	Ant	Swimmer	Hopper
Baseline	283.27	186.36	62.89	487.25
RP1 ($\lambda = 0$)	319.14	223.21	41.14	603.52
RP1 (No EarlyStop)	247.82	197.95	41.04	595.77
RP1	390.49	238.4	64.19	619.73

($\lambda = 0$), we lose the flexibility to target specific regions of the state-space. On the other hand, by removing our early stopping mechanism (No EarlyStop), we acquire a trajectory dataset for our model that has too much redundant data, reducing sample efficiency. Nonetheless, we believe adding either of these components to existing MBRL methods, which either have a fixed temperature parameter (λ) or fixed data collection batch size, would lead to performance gains.

For further details on our experiments, see the open sourced repo at <https://github.com/fiorenza2/ReadyPolicyOne>.

4.1.6 Conclusion and Future Work

We presented Ready Policy One (RP1), a new approach for Model-Based Reinforcement Learning (MBRL). RP1 casts data collection in MBRL as an active learning problem, and subsequently seeks to acquire the most informative data via an exploration policy. Leveraging online learning techniques, the objective function for this policy adapts during optimization, allowing RP1 to vary its focus on the often fruitful reward function. We showed in a variety of experiments that RP1 significantly increases sample efficiency in MBRL, and we believe it can lead to new state of the art when combined with the latest architectures.

We are particularly excited by the many future directions from this work. Most obviously, since our method is orthogonal to other recent advances in MBRL, RP1 could be combined with state of the art probabilistic architectures (Chua et al., 2018), or variational autoencoder based models (Ha & Schmidhuber, 2018; Hafner et al., 2019).

In addition, we could take a hierarchical approach, by ensuring our exploration policies maintain core behaviors but maximize entropy in some distant unexplored

region. This would require behavioral representations, and some notion of distance in behavioral space (Co-Reyes et al., 2018), and may lead to increased sample efficiency as we could better target specific state action pairs.

4.1.7 Errata

There is an error in Equation (4.3). The equality should relate to the arg max of the function values, i.e.,:

$$\pi_{\theta_t} = \arg \max_{\pi_{\theta_t}} [\mathbb{E}_{\tau \sim \pi_{\theta_t}} [(1 - \lambda)\mathcal{R}(\tau) + \lambda\sigma(\mathcal{R}(\tau))]]$$

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Ready Policy One: World Building Through Active Learning
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Philip Ball, Jack Parker-Holder, Aldo Pacchiano, Krzysztof Choromanski, Stephen Roberts, Ready Policy One: World Building Through Active Learning, <i>International Conference on Machine Learning 2020</i>

Student Confirmation

Student Name:	Philip Ball
Contribution to the Paper	Wrote majority of codebase, performed main analysis, wrote majority of manuscript, produced proofs
Signature 	Date 5/October/2023

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

4.2 Stabilizing Off-Policy Deep Reinforcement Learning from Pixels

Abstract Off-policy reinforcement learning (RL) from pixel observations is notoriously unstable. As a result, many successful algorithms must combine different domain-specific practices and auxiliary losses to learn meaningful behaviors in complex environments. In this work, we provide novel analysis demonstrating that these instabilities arise from performing temporal-difference learning with a convolutional encoder and low-magnitude rewards. We show that this new *visual deadly triad* causes unstable training and premature convergence to degenerate solutions, a phenomenon we name *catastrophic self-overfitting*. Based on our analysis, we propose A-LIX, a method providing adaptive regularization to the encoder’s gradients that explicitly prevents the occurrence of *catastrophic self-overfitting* using a dual objective. By applying A-LIX, we significantly outperform the prior state-of-the-art on the DeepMind Control and Atari 100k benchmarks without any data augmentation or auxiliary losses.

4.2.1 Introduction

One of the core challenges in real world Reinforcement Learning (RL) is achieving stable training with sample-efficient algorithms (Dulac-Arnold et al., 2019). Combining these properties with the ability to reason from visual observations has great implications for the application of RL to the real world (Kalashnikov et al., 2018; Zhu et al., 2020). Recent works utilizing *temporal-difference* (TD-) learning have made great progress advancing sample-efficiency (Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2018a; Cetin & Celiktutan, 2021). However, stability has remained a key issue for these off-policy algorithms (Sutton, 1988; Duan et al., 2016; Van Hasselt et al., 2018; Buşoniu et al., 2018), making their general applicability limited as compared to their on-policy counterparts (Schulman et al., 2017; Cobbe et al., 2021). At the same time, using pixel observations has been another orthogonal source of instabilities, with several successful approaches relying on pre-training instead of end-to-end learning (Finn et al., 2015; Dwibedi et al.,

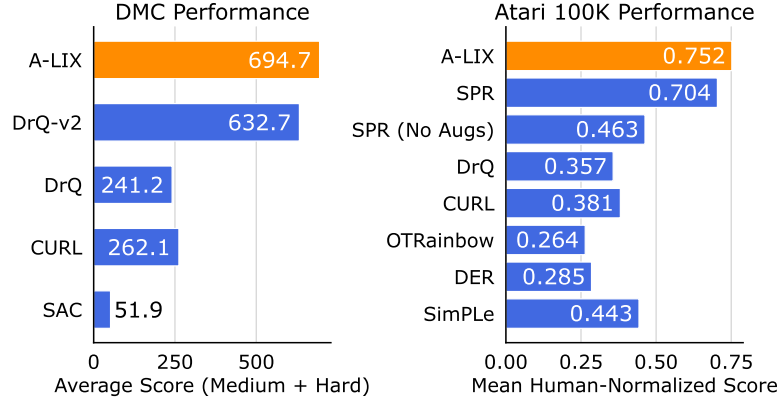


Figure 4.4: Performance of agents in DMC (**left**) and Atari 100k (**right**) benchmarks from 10 seeds. A-LIX outperforms previous methods without using image augmentations or auxiliary losses.

2018). In fact, alternative optimization objectives, large amounts of simulation data, and symbolic observations have been common factors in most contemporary large-scale RL milestones (Silver et al., 2017; Vinyals et al., 2019; Berner et al., 2019).

In this work, we provide novel insights behind why applying successful off-policy RL algorithms designed for proprioceptive tasks to pixel-based environments is generally underwhelming (Lee et al., 2019; Yarats et al., 2021). In particular, we provide evidence that three key elements strongly correlate with the occurrence of detrimental instabilities: *i) Exclusive* reliance on the *TD-loss*. *ii) Unregularized* end-to-end learning with a *convolutional encoder*. *iii) Low-magnitude sparse rewards*. Using this framework, we are able to motivate the effectiveness of auxiliary losses (Laskin et al., 2020b; Schwarzer et al., 2020; Yarats et al., 2021) and many domain-specific practices (Hessel et al., 2018; Laskin et al., 2020a) by explaining how they address elements of this new *visual deadly triad*.

We focus our analysis on the popular DeepMind Control suite (DMC) (Tassa et al., 2018), where the introduction of random shift augmentations has played a key role in recent advances (Laskin et al., 2020a; Kostrikov et al., 2021; Yarats et al., 2022). In this domain, we observe that the presence of the *visual deadly triad* results in the TD-loss gradients through the convolutional encoder’s feature maps having high spatial frequencies. We find these gradients are *spatially inconsistent* and result in degenerate optimization landscapes when backpropagated

Table 4.4: Practices from recent pixel-based TD-learning methods to mitigate elements of the visual deadly triad. [†]DrQ uses 10-step returns on Atari. *CURL uses 20-step returns on Atari.

Algorithm	Visual Deadly Triad Mitigation		
	TD-Loss	CNN Overfit	Low-Density Reward
DrQ/RAD	-	Shift/Jitter Augmentations	10-step returns [†]
DrQ-v2	-	Shift Augmentations	3-step returns
SAC-AE	VAE Loss	-	-
SPR	Model-Based Loss	Shift/Jitter Augmentations	10-step returns
DER	-	Non-Overlapping Strides	20-step returns
CURL	Contrastive Loss	Shift Augmentations	20-step returns*

to the encoder’s parameters. Furthermore, repeatedly updating the convolutional encoder with these gradients consistently leads to early convergence to degenerate feature representations causing the critic to fit high-variance erroneous targets, a phenomenon we name *catastrophic self-overfitting*. As a way of identifying the direct implications of the visual deadly triad in the gradient signal, we propose a new measure called the *Normalized Discontinuity (ND)* score and show how its value precisely correlates with agent performance. Thus, we explain the effectiveness of shift augmentations by recognizing that they regularize the gradient signal by providing an *implicit* spatial smoothing effect.

Based on our analysis, we propose **Adaptive Local Signal MiXing (A-LIX)** a novel method to prevent catastrophic self-overfitting with two key components: *i)* A new parameterized layer (LIX) that *explicitly* enforces smooth feature map gradients. *ii)* A dual objective that ensures learning stability by adapting the LIX parameters based on the estimated *ND* scores. We show that integrating A-LIX with existing off-policy algorithms achieves state-of-the-art performance in both DeepMind Control and Atari 100k benchmarks without requiring image augmentations or auxiliary losses and significantly fewer heuristics. We open-source our code to facilitate reproducibility and future extensions⁴.

Our main contribution can be summarized as follows:

⁴<https://github.com/Aladoro/Stabilizing-Off-Policy-RL>

- We conjecture the existence of a **visual deadly triad** as a principal source of instability in reinforcement learning from pixel observations and provide clear empirical evidence validating our hypothesis.
- We show these instabilities affect the gradient signal causing **catastrophic self-overfitting**, a phenomenon that can severely harm TD-learning. As a result, we design the **normalized discontinuity score** to explicitly anticipate its occurrence.
- We propose **A-LIX**, a new method that adaptively regularizes convolutional features to prevent catastrophic self-overfitting, achieving state-of-the-art results on two popular pixel-based RL benchmarks.

4.2.2 Background

We consider problem settings described by Markov Decision Processes (MDPs) (Bellman, 1957), defined as the tuple $(S, A, P, p_0, r, \gamma)$. This comprises a state space S , an action space A , transitions dynamics given by P and p_0 , and a reward function r . The RL objective is then for an agent to recover an optimal policy π^* , yielding a distribution of trajectories $p_\pi(\tau)$ that maximizes its expected sum of discounted future rewards, $\pi^* = \arg \max_\pi \mathbb{E}_{p_\pi(\tau)} [\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. In off-policy RL, this objective is usually approached by learning a *critic* function to evaluate the effectiveness of the agent’s behavior. A common choice for the critic is to parameterize the policy’s Q-function $Q^\pi : S \times A \rightarrow \mathbb{R}$, that quantifies the agent’s performance after performing a particular action: $Q^\pi(s, a) = \mathbb{E}_{p_\pi(\tau|s_0=s, a_0=a)} [\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. Most off-policy algorithms entail storing trajectories in a buffer D , and learning parameterized Q-functions by iteratively minimizing a squared temporal difference (TD-) loss:

$$\begin{aligned} J_Q(\phi) &= \mathbb{E}_{(s,a,s',r) \sim D} \left[(Q_\phi^\pi(s, a) - y)^2 \right], \\ y &= r + \gamma \mathbb{E}_{a \sim \pi(s')} \left[\hat{Q}_{\phi'}^\pi(s', a) \right]. \end{aligned} \tag{4.9}$$

Here, the TD-targets y are computed from a 1-step bootstrap operation with a slowly-changing *target Q-function* $\hat{Q}_{\phi'}^\pi$. In continuous action spaces, we also learn a separate parameterized policy to exploit the information in the critic. This practically

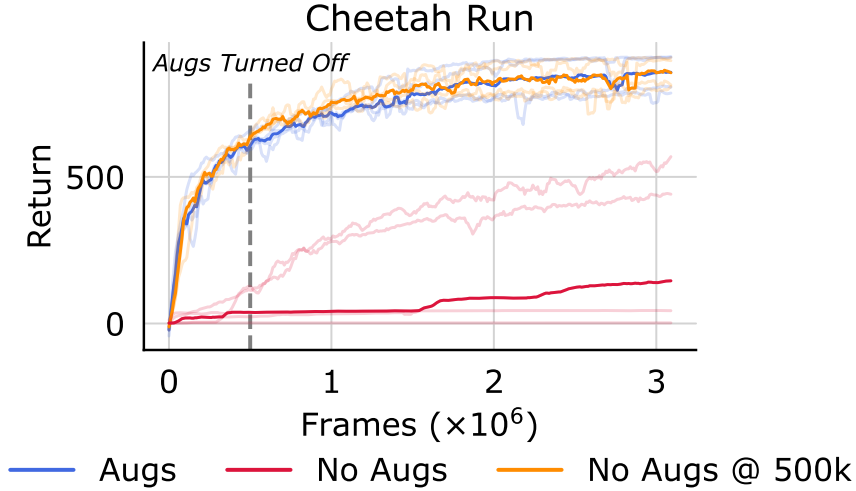


Figure 4.5: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The vertical dashed line shows when augmentations are turned off.

results in alternating TD-learning with maximizing the Q-function’s expected return predictions, following the policy gradient theorem (Sutton et al., 2000).

4.2.3 Instabilities in TD-Learning from Pixels

Unlike proprioceptive observations, off-policy RL from pixel observations commonly requires additional domain-specific practices to ensure stability. In this section, we provide a novel analysis of this phenomenon by focusing on the DeepMind Control Suite (Tassa et al., 2018). In this benchmark, the introduction of random shift data augmentations has been a core component of recent advances in pixel-based off-policy RL (Laskin et al., 2020a; Yarats et al., 2022), allowing us to isolate and reproduce stable and unstable training regimes. Our analysis suggests the existence of specific elements that cause instabilities and strives to explain their implications on learning dynamics. We validate our findings via thorough empirical experimentation showing numerous results corroborating our hypotheses. Based on our discoveries, in Section 4.2.4 we provide a new interpretation of random shifts and propose a new improved method to isolate and counteract instabilities.

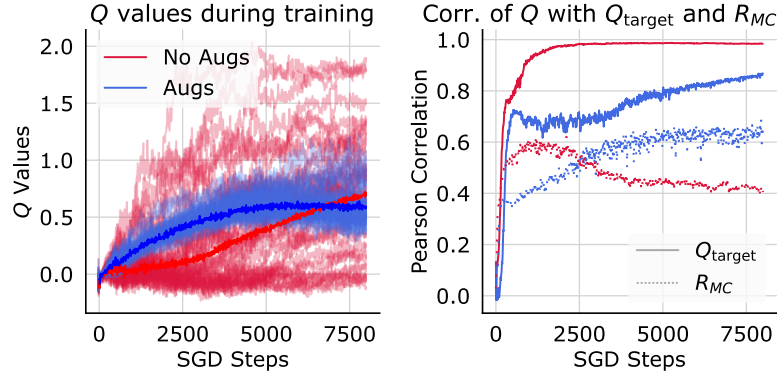


Figure 4.6: Evidence of overfitting when augmentations are not used. On the **left**, shaded lines are individual estimates, the solid line represents the median Q-value. On the **right**, the Q-values Pearson correlation with target values and Monte-Carlo returns (R_{MC}).

Why Do Augmentations Help?

The underlying mechanism behind the effectiveness of random shifts is not immediately clear. While this augmentation may appear to assist generalization by encoding an invariance (Shorten & Khoshgoftaar, 2019), we note that all the environments from DMC employ a camera that is fixed relative to the agent’s position. Hence, robustness to shifts does not appear to introduce any useful inductive bias about the underlying tasks. Moreover, prior work successfully learned effective controllers without augmentations (Hafner et al., 2020; Yarats et al., 2021), suggesting that shift generalization might not be the primary benefit of this method. We analyze the effect of random shifts by training a DrQ-v2 agent (Yarats et al., 2022) on Cheetah Run but turning off augmentations after an initial 500,000 time-steps learning phase. As shown in Fig. 4.5, while training without any shift augmentation fails to make consistent progress, turning off augmentations after the initial learning phase actually appears to slightly improve the performance of DrQ-v2. This result is a clear indication that augmentations are not needed for asymptotic performance, and are most helpful to counteract instabilities present in the *earlier stages of learning*, which we now focus on analyzing (see App. C.6.1-C.6.2 for complementary experiments validating these claims).

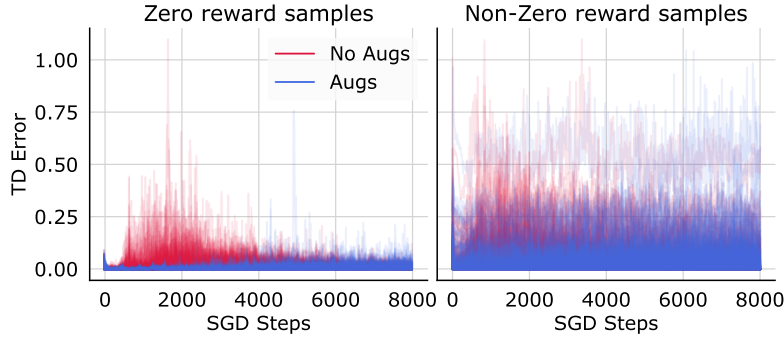


Figure 4.7: TD-loss of offline fixed transitions during training, separated based on having non-zero reward.

Identifying a New Deadly Triad

To reduce confounding factors and to disentangle the origin of these instabilities, we design an offline RL experiment (Levine et al., 2020). This experiment isolates three distinct elements affecting off-policy RL: exploration, policy evaluation, and policy improvement. First, we gather a set of 15,000 transitions with pixel observations using a random policy in Cheetah Run. This allows us to ground exploration and analyze learning from fixed data resembling the early stages of online training (when augmentations appear most helpful). We then isolate policy evaluation by training both critic and encoder using SARSA (Rummery & Niranjan, 1994) until convergence on this fixed data. Finally, we run policy improvement, training an actor to maximize the expected discounted return as predicted by the converged critic (see App. C.2.1 for details). Interestingly, we find that *turning on* augmentations exclusively during exploration or policy improvement has no apparent effect on stability and final performance. Hence, we focus on the effects that augmentations have on TD-learning and analyze applying augmentations only during policy evaluation.

As shown in Table 4.5, applying augmentations during policy evaluation enables us to learn policies that achieve a return of 86.5, despite the best trajectory in the offline data achieving only 10.8. In contrast, without augmentations we consistently recover near 0 returns, resembling the failures observed in the online experiments. On the left of Fig. 4.6 we show the evolution of the predicted Q-values for both agents on a fixed batch of offline data. In particular, when performing policy evaluation without augmentations, these predictions display extremely high variance across

Table 4.5: Performance and training statistics of different agent types in the offline experiments from 15,000 random transitions.

Agent	Final TD-Loss	Final Policy Loss	Return
Augmented	0.021	-0.99	86.5 ± 11.3
Non-Augmented	0.002	-1.05	9.2 ± 12.1
Proprioceptive	0.012	-1.14	79.1 ± 7.7
Frozen CNN (random)	0.023	-0.95	43.6 ± 20.2
Frozen CNN (pre-trained)	0.012	-0.99	77.6 ± 18.5
Non-Augmented (norm r)	18.616	3.86	38.6 ± 16.5
Non-Augmented (10-step returns)	0.003	-1.24	36.5 ± 20.3

different state-action pairs. In Table 4.5 we further show that the non-augmented agent displays significantly lower loss, despite having higher average Q-values than the augmented agent (Schaul et al., 2021). We argue this is a clear indication of the occurrence of *overfitting*. We corroborate our claim by analyzing the evolution of the Pearson bi-variate correlation between the estimated Q-values and target Q-values on the right of Fig. 4.6. These results show that the non-augmented agent displays near-perfect correlation with its own target Q-values throughout training, indicating that it immediately *learns to fit its own noisy, randomly-initialized predictions*. We also record the correlation with the actual discounted Monte-Carlo returns, which represent the *true targets* the Q-values should ideally approximate during policy evaluation. For these results, we observe that the relationship between applying augmentations and the recorded correlation is reversed, with the non-augmented agent displaying significantly lower correlation. This dichotomy appears to indicate that fitting the noisy targets severely affects learning the *useful* training signal from the collected transitions regarding the experienced rewards. We confirm this phenomenon by splitting the data into non-zero and zero reward transitions, where the only learning signal propagated in the TD-loss is from the initially random target values. In Fig. 4.7 we illustrate that the non-augmented agents initially experience much higher TD-errors on zero reward transitions, confirming that they focus on fitting uninformative components of the TD-objective.

In Table 4.5 we provide the results of additional experiments that indicate that TD-learning is not the *only* cause for the observed instabilities. First, we confirm

that the observed overfitting appears to be exclusive to performing *end-to-end* TD-learning with convolutional neural network (CNN) encoders. Concretely, we run the same offline experiment without *training an encoder* in three different settings. First, we consider performing policy evaluation directly from non-augmented *proprioceptive* observations with a fully-connected critic network. Moreover, we also consider *freezing* the encoder weights either to their initial *random values* or to *pre-trained* values from the augmented agent experiments. In all three cases, we attain largely superior performance, almost matching the augmented agent’s performance for both the proprioceptive and pre-trained experiments. In addition, we also find that the observed overfitting phenomenon is diminished when simply increasing the magnitude of the reward signal in the TD-loss. We test this through two additional experiments which consider *normalizing* the collected rewards before policy evaluation and incorporating *large n -step returns* (Sutton, 1988). As reported, both modifications considerably improve the non-augmented agent’s performance. However, we note that both practices introduce further unwanted variance in the optimization, failing to yield the same improvements as augmentations (see App. C.3.2).

Taken together, our results appear to strongly indicate that instabilities in off-policy RL from pixel observation come from three key conditions, which we refer to as the *visual deadly triad*: i) Exclusive reliance on the TD-loss; ii) Unregularized learning with an expressive convolutional encoder; iii) Initial low-magnitude sparse rewards. Further evidence arises when considering the ubiquity of particular practices employed in pixel-based off-policy RL. In particular, as summarized in Table 4.4, most popular prior algorithms feature design choices that appear to counteract at least two elements of this triad, either directly or implicitly. Furthermore, we show these instabilities result in the non-augmented critics focusing on learning their own noisy predictions, rather than the actual experienced returns. We observe this ultimately leads to convergence to erroneous and high-variance Q-value predictions, a phenomenon we name *catastrophic self-overfitting*.

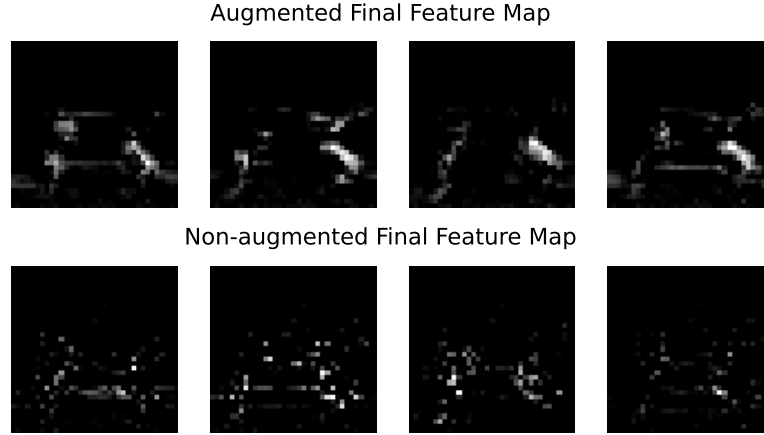


Figure 4.8: Feature maps in the final layer of both augmented (**top**) and non-augmented (**bottom**) agent encoders. Non-augmented agents manifest inconsistent, high-frequency feature maps.

Anticipating Catastrophic Self-Overfitting

We now attempt to unravel the links that connect the *visual deadly triad* with *catastrophic self-overfitting*. We start by observing that catastrophic self-overfitting comes with a significant reduction of the critic’s sensitivity to changes in action inputs, implying that the erroneous high-variance Q-value predictions arise primarily due to changes in the observations (see App. C.6.3 for action-value surface plots). Hence, we focus on analyzing the feature representations of the pixel observations, computed by the convolutional encoder, $z \in \mathbb{R}^{C \times H \times W}$. In particular, we wish to quantify the *sensitivity* of feature representations to small perturbations in the input observations. To measure this, we evaluate the Jacobians of the encoder across a fixed batch of offline data for the augmented and non-augmented agents. We then calculate the Frobenius norm of each agent’s Jacobians, giving us a measure of how quickly the encoder feature representations are changing locally around an input (see App. C.2.2 for details). Our results show a stark difference, with the feature representations of the non-augmented agents being on average **2.85** times more sensitive. This suggests that overfitting is driven by the CNN encoder’s *representations* learning *high-frequency* information about the input observations and, thus, breaking useful inductive biases about this class of models (Rahaman et al., 2019).

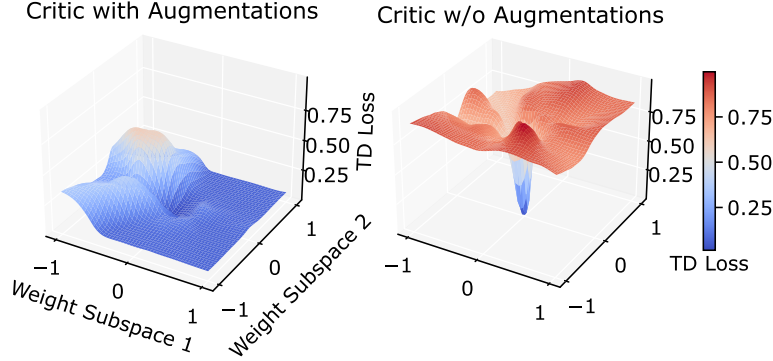


Figure 4.9: Critic loss surface plots of augmented (**left**) and non-augmented (**right**) agents after 5,000 steps of offline training.

In App. C.5.1 we demonstrate that lower sensitivity to random noise, while desirable for optimization (Rosca et al., 2020), is actually a *byproduct* of a stable feature representations, and not its defining factor. Furthermore, observing the actual feature maps of different observations in Fig. 4.8, we see that augmentations make the encoder produce features that are *spatially consistent*, aligned with common understandings of how natural representations should appear (Alsallakh et al., 2021; Allen-Zhu & Li, 2021). In contrast, the non-augmented agents display high-frequency and *discontinuous* feature maps that do not reflect the spatial properties of their inputs. Hence, our evidence suggests that catastrophic self-overfitting specifically follows from the same learning process that produces highly-sensitive and discontinuous encoder feature maps. Therefore, we turn our focus to analyzing the *gradients* backpropagated to the encoder’s features maps and observe one key property: the gradients of the output feature maps *consistently reflect the same spatial properties* of their resulting features. In particular, the *gradients of the feature maps* appear *spatially consistent* for the augmented agent, and *discontinuous* for the non-augmented agent. This optimization property reflects intuitive understandings of backpropagation since discontinuous gradients should push the encoder’s weights to encode discontinuous representations. To provide further complementary evidence that discontinuous gradients are the direct cause of catastrophic self-overfitting, we analyze the normalized loss surfaces *when backpropagating these discontinuous gradients* to the encoder’s parameters (following Li et al. (2018)). In Fig. 4.9, we

see that gradient discontinuities in the non-augmented agent yield extreme peaks in its encoder’s loss surface, clearly suggestive of overfitting (Keskar et al., 2017)⁵.

To quantify the level of discontinuity in the features and their gradients, we propose a new metric that encodes the aggregated immediate spatial ‘unevenness’ of each feature location within its relative feature map. In particular, we define $D(z) \in \mathbb{R}^{C \times H \times W}$ as the expected squared local discontinuity of z in any spatial direction, i.e.:

$$D(z)_{ijc} \approx \mathbb{E}_{v \sim S^1} \left[\left(\frac{\partial z_{ijc}}{\partial v} \right)^2 \right], \quad (4.10)$$

practically estimated via sampling. We then normalize each value in $D(z)$ by its squared input and average over all the feature positions. We name this metric the *normalized discontinuity (ND)* score:

$$ND(z) = \frac{1}{C \times H \times W} \sum_{c=1}^C \sum_{j=1}^H \sum_{i=1}^W \frac{D(z)_{ijc}}{z_{ijc}^2}. \quad (4.11)$$

Intuitively, this score reflects how locally discontinuous z is expected to be at any spatial location. In Fig. 4.10, we show how the ND score of ∇z evolves during training in the offline and an online setting for both augmented and non-augmented agents. We see that augmented agents experience considerably less discontinuous gradients through their features, and that recordings of lower ND scores also appear to be highly correlated with performance improvements. We additionally show an accumulated ND score, using an exponential moving average of ∇z in each spatial position to calculate this metric. Interestingly, we observe that the ND score over accumulated gradients is almost identical to the instantaneous ND score, showing that similar gradient discontinuities are propagated *persistently* through training in each position of the feature map. This property confirms that the discontinuities are not smoothed by the stochastic sampling of different consecutive training batches, in which case we would expect to observe lower accumulated ND scores. Thus, it suggests that self-overfitting emerges in the non-augmented agents due to repeated gradient steps towards *persistent* feature map discontinuities.

⁵Instead, the loss surface with respect to the fully-connected weights is smoother (App. C.6.5).

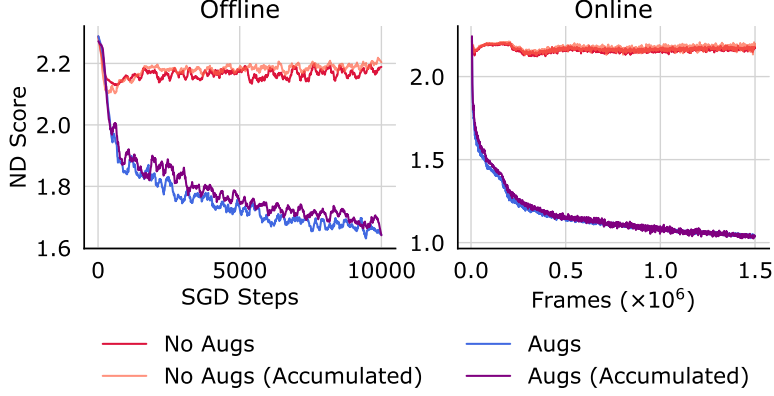


Figure 4.10: Instantaneous (red and blue) and accumulated (orange and purple) ND scores for the features gradients from offline (left) and online (right) training in Cheetah Run.

4.2.4 Counteracting Gradient Discontinuities

Gradient Smoothing and Random Shifts

As analyzed in Section 4.2.3, catastrophic self-overfitting occurs when the gradients in the convolution layers are locally discontinuous. As a result, we argue that the efficacy of random shifts arises from their downstream effect on feature gradient computation, which counteracts these discontinuities during backpropagation. In particular, while random shifts do not act directly on the latent representation or their respective gradients, they do affect *how* the latent representations are computed. This has an impact on how *persistent* discontinuous components of the gradient are backpropagated to the encoder’s parameters during learning. From the approximate shift invariance of convolutional layers, we can view a convolutional encoder as computing each of the feature vectors $[z_{1ij}, \dots, z_{Cij}]^T$ with the same parameterized function, V_ϕ , that takes as input a subset of each observation $O \in \mathbb{R}^{C' \times H' \times W'}$. This subset corresponds to a local neighborhood around some reference input coordinates i', j' . Thus, the only factor differentiating features in the same feature map (e.g., z_{cij} and z_{ckl}) is some implicit function $f(i, j) = i', j'$ translating each of the output features coordinate into the relative reference input coordinate, i.e. $z_{cij} = V_\phi(O, i', j')_c$ (determined by kernel sizes, strides...). Therefore, random shifts are approximately equivalent to further translating each reference coordinate by adding some uniform random variables δ'_x, δ'_y :

$$z_{cij} \approx V_\phi(O, i' + \delta'_x, j' + \delta'_y)_c,$$

where $\delta'_x, \delta'_y \sim U[-s', s'], \quad f(i, j) = i', j'.$

Due to the employed strides from the convolutional architectures used in DrQ-v2 (Yarats et al., 2022), the difference in reference coordinates of adjacent features in a feature map is less than the maximum allowable shift employed in the augmentations, i.e., $(i + 1)' - i', (j + 1)' - j' < s'$ (where s' is the maximum allowable shift). Consequently, shift augmentations effectively turn the deterministic computation graph of each feature z_{cij} into a random variable, whose sample space comprises the computation graphs of all nearby features within its feature map. Hence, applying different random shifts to samples in a minibatch makes the gradient of each feature ∇z_{cij} backpropagate to a random computation graph, sampled from a set that extends the set of non-augmented computation graphs for all features in a local neighborhood of coordinates i, j . Therefore, aggregating the parameter gradients produced with different δ'_x, δ'_y , provides a *smoothing effect* on how each discontinuous component of ∇z affects learning, and prevents persistent discontinuities from accumulating. Hence, random shifts break the second condition of the visual deadly triad, by providing effective *implicit* regularization of the convolutional encoder’s learning process. At the same time, this minimally disrupts the information content of the resultant features, since discarded observation borders almost exclusively comprise background textures that are irrelevant for performing the task. This interpretation of random shifts aligns with the analysis in Section 4.2.3, showing that implicitly smoothing over the backpropagated gradient maps consistently prevents catastrophic self-overfitting.

Local Signal Mixing

We extrapolate our hypotheses about catastrophic self-overfitting and random shifts by proposing a technique that aims to enforce gradient smoothing regularization *explicitly*. We propose **Local S**IGNAL **M**iXing, or **LIX**, a new layer specifically designed to prevent catastrophic self-overfitting in convolutional reinforcement

learning architectures. LIX acts on the features produced by the convolutional encoder, $z \in \mathbb{R}^{C \times H \times W}$, by randomly mixing each component z_{cij} with its neighbors belonging to the same feature map. Hence, LIX outputs a new latent representation with the same dimensionality $\hat{z} \in \mathbb{R}^{C \times H \times W}$, whose computation graph minimally disrupts the information content of each feature z_{cij} while smoothing discontinuous components of the gradient signal during backpropagation.

LIX is a regularization layer that acts as a simple random smoothing operation, reducing the expected magnitude of gradient discontinuities by preventing higher frequency signals to persist. In the forward pass, LIX produces a new latent representation where for each of the C feature maps, $\hat{z}_{cij}$ is computed as a randomly weighted average of its spatial neighbors around coordinates i, j . We further *parameterize* this stochastic operation using some maximum range radius S , and consequently sample two uniform *continuous* random variables $\delta_x, \delta_y \sim U[-S, S]$, representing shifts in the x and y coordinates respectively. Correspondingly, we define $\tilde{i} = i + \delta_x$ and $\tilde{j} = j + \delta_y$ and perform the weighted averaging as a bilinear interpolation with weights determined by the random shifts:

$$\begin{aligned} \hat{z}_{cij} = & z_{c[\tilde{i}][\tilde{j}]}(\lceil \tilde{i} \rceil - \tilde{i})(\lceil \tilde{j} \rceil - \tilde{j}) + z_{c[\tilde{i}][\tilde{j}]}(\lceil \tilde{i} \rceil - \tilde{i})(\tilde{j} - \lfloor \tilde{j} \rfloor) \\ & + z_{c[\tilde{i}][\tilde{j}]}(\tilde{i} - \lfloor \tilde{i} \rfloor)(\lceil \tilde{j} \rceil - \tilde{j}) + z_{c[\tilde{i}][\tilde{j}]}(\tilde{i} - \lfloor \tilde{i} \rfloor)(\tilde{j} - \lfloor \tilde{j} \rfloor). \end{aligned}$$

Since nearby features in a convolutional feature map are computed with very similar receptive fields, the mixing effect of LIX should have a trivial effect on the information the encoder can convey in its latent representations. In addition, LIX should have a direct regularization effect on the gradients by acting on the feature maps themselves. In particular, since LIX computes each output feature from a weighted average of its neighbors, backpropagation will split each gradient $\nabla \hat{z}_{cij}$, to a random local combination of features within the same feature map, $\{\nabla z_{c[\tilde{i}][\tilde{j}]}, \nabla z_{c[\tilde{i}][\tilde{j}]}, \nabla z_{c[\tilde{i}][\tilde{j}]}, \nabla z_{c[\tilde{i}][\tilde{j}]}\}$. Thus, LIX should mostly preserve the consistent component of ∇z , while randomly smoothing its discontinuous component.

There are multiple key differences between the regularization from LIX and random shifts. LIX provides a local smoothing effect over the gradients explicitly

and exactly, without having to deal with the implications of padding and strided convolutions breaking shift-invariance assumptions. Moreover, LIX smooths the gradient signal not only across different inputs but also within each feature map. In addition, by applying its operation solely at the feature level, the encoder can still learn to entirely circumvent LIX’s smoothing effect on the information encoded in the latent representations, given enough capacity. This means that LIX does not forcibly preclude any input information from affecting the computation. Consequently, LIX also does not have to enforce learning invariances which might not necessarily reflect useful inductive biases about the distribution of observations. In contrast, random shifts need to exploit the particular uninformative nature of the observations borders to avoid disrupting the features’ information content.

A-LIX

LIX introduces a single key parameter: the range radius S used for sampling δ_x and δ_y . Intuitively, this value should reflect how much we expect gradients to be locally consistent for a given architecture and task. Therefore, we argue that the value of S should ideally decrease throughout training as the useful learning signal from the TD-loss becomes stronger. This is consistent with the results illustrated in Figure 4.5, showing that turning off random shift augmentations after the TD-targets become informative can improve learning. Hence, we propose an adaptive strategy to learn S throughout training. Utilizing the *normalized discontinuity* (ND) score in Section 4.2.3, we set up a dual optimization objective to ensure a minimum value of local smoothness in the representation gradients, $\overline{ND}$. However, computing the ND score of the gradient signal involves a ratio between potentially very small values. As a result, estimation of these values from a batch of gradient samples can lead to outliers having an extreme impact on this average measure, translating into large erroneous updates of S . To overcome this, we propose using a slightly modified version of the ND score with increased robustness to outliers (see App. C.3.1 for further details):

$$\widetilde{ND}(\nabla \hat{z}) = \sum_{c=1}^C \sum_{j=1}^H \sum_{i=1}^W \log \left(1 + \frac{D(\nabla \hat{z})_{cij}}{\nabla \hat{z}_{ijc}^2} \right). \quad (4.12)$$

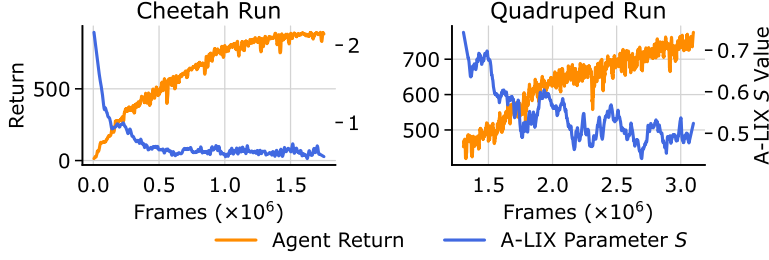


Figure 4.11: A-LIX’s S parameter evolution during training in Cheetah Run (**left**) and Quadraped Run (**right**). As the critic targets become more informative, S falls, improving data efficiency and asymptotic performance.

In practice, we set up a dual optimization objective similar to the automatic temperature adjustment from Haarnoja et al. (2018b). This entails alternating the optimization of the TD-learning objectives described in Section 4.2.2 with minimizing a dual objective loss:

$$\arg \min_S -S \times \mathbb{E}_{\hat{z}} \left[\widetilde{ND}(\nabla \hat{z}) - \overline{ND} \right], \quad (4.13)$$

approximating dual gradient descent (Boyd et al., 2004). Hence, we call this new layer **Adaptive LIX** (A-LIX). In Fig. 4.11 we show that A-LIX effectively anneals S as the agent escapes its unstable regimes, in line with our intuition.

4.2.5 Performance Evaluation

We evaluate the effectiveness of A-LIX in pixel-based reinforcement learning tasks in two popular and distinct domains featuring a diverse set of continuous and discrete control problems. We integrate A-LIX with existing popular algorithms and compare against current state-of-the-art model-free baselines. We provide further details of our integration and full hyperparameters in App. C.4. We also extend this section by providing more granular evaluation metrics in App. C.1. Furthermore, we provide ablation studies analyzing different components of A-LIX in App. C.5.

DeepMind Control Evaluation

We first evaluate the effectiveness of A-LIX for pixel-based RL on continuous control tasks from the DeepMind Control Suite (DMC) (Tassa et al., 2018). Concretely, we integrate A-LIX with the training procedure and network architecture from DrQ-v2 (Yarats et al., 2022), but *without using image augmentations*. To show

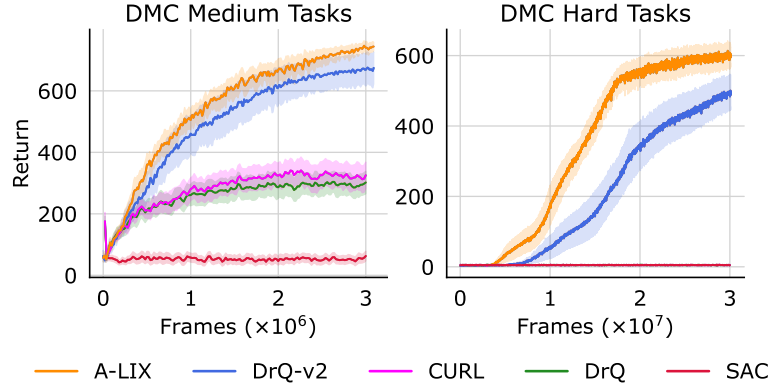


Figure 4.12: Average performance in 10 seeds for DMC Medium (**left**) and Hard tasks (**right**). Shaded regions represent ± 1 SE.

the generality of our method we *do not modify* any of the environment-specific hyperparameters from DrQ-v2 and simply add our A-LIX layer after each encoder nonlinearity. For simplicity, we optimize a shared S for all the A-LIX layers with the dual objective in Eq. 4.13. Hence, this introduces a single additional parameter and negligible computational overhead. We compare A-LIX to DrQ-v2, which represents the current state-of-the-art on this benchmark. We also compare against three further baselines: the original DrQ (Kostrikov et al., 2021), which foregoes n -step returns and includes an entropy bonus; CURL (Laskin et al., 2020b), which includes an auxiliary contrastive objective; an extension of SAC (Haarnoja et al., 2018b) with the encoder from Yarats et al. (2021). These last three baselines have been performant on a prior DMC benchmark that considers fewer tasks with high action repeats, as described by Hafner et al. (2019). Instead, we evaluate on the more challenging ‘Medium’ and ‘Hard’ benchmarks from Yarats et al. (2022), comprising 15 tasks with low action repeats.

Results. We summarize the results in Figure 4.12, showing the mean performance curves for both medium and hard benchmark tasks. We provide further details and the full list of results across all 15 environments in App. C.1.1. Overall, A-LIX surpasses all prior methods with clear margins, both in terms of efficiency and final performance. This is particularly notable in the more complex ‘Hard’ tasks. As highlighted in prior work (Cetin & Celiktutan, 2021), DrQ-v2 appears to yield inconsistent results on some of the harder exploration tasks with sparse

Table 4.6: Results summary for the Atari 100k benchmark. The reported performance of A-LIX is from 10 seeds.

<i>Metrics</i>	SimPLe	DER	OTRainbow	CURL	DrQ	SPR	A-LIX
<i>Norm. Mean</i>	0.443	0.285	0.264	0.381	0.357	0.704	0.753
<i>Norm. Median</i>	0.144	0.161	0.204	0.175	0.268	0.415	0.411
<i># SOTA</i>	7	1	1	1	1	4	11
<i># Super</i>	2	2	1	2	2	7	7
<i>Average Rank</i>	3.92	5.00	5.21	3.92	4.85	2.88	2.21

rewards. This likely indicates that the gradient regularization induced by random shifts (described in Section 4.2.4) is unable to consistently prevent catastrophic self-overfitting in scenarios where the initial learning signal from TD-learning is particularly low. Finally, DrQ, CURL, and SAC fail to make consistent meaningful progress on this harder benchmark. This performance gap corroborates the third component of the visual deadly triad, showing how lower magnitude rewards due to harder exploration and lower action-repeats further destabilize TD-learning based algorithms, and explains the gains seen in DrQ-v2 when incorporating n-step returns. We believe these results emphasize the challenge of overcoming the visual deadly triad in continuous control problems and the particular effectiveness of A-LIX to counteract its direct implications.

Atari 100k Evaluation

We perform a second set of experiments in an entirely different setting, discrete control. We make use of the popular Atari Learning Environment (ALE) (Bellemare et al., 2013) and consider the 100k evaluation benchmark from Kaiser et al. (2020). In particular, this benchmark comprises evaluating performance for 26 tasks after only two hours of play-time (100k interaction steps), following the evaluation protocol in Machado et al. (2018). We integrate A-LIX with Data-Efficient Rainbow (DER) (van Hasselt et al., 2019), a simple extension to Rainbow (Hessel et al., 2018) with improved data-efficiency. We would like to note that our integration has key differences to DER, designed to highlight the generality of our method in tackling the visual deadly triad. In particular, we reduce the n-step returns to 3 (from 20), and we

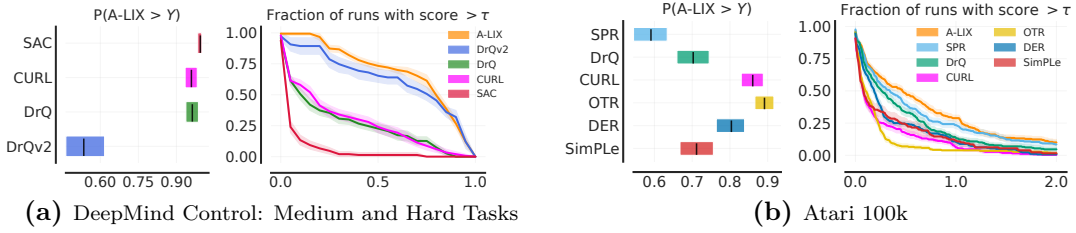


Figure 4.13: *Probability of improvement and performance profiles obtained from the recorded results in DMC (left) and Atari 100k (right). A-LIX displays statistically significant improvements and stochastically dominates most prior algorithms.*

maintain the *same* encoder architecture as in DrQ-v2. To speak to the latter point, this means we do not require the highly regularized encoders with large convolutional filters and strides, used ubiquitously in off-policy learning for Atari environments. Instead, to stabilize learning we simply apply our A-LIX layer after the final encoder nonlinearity. We compare against three algorithms that, like A-LIX, do not employ data-augmentation: Data-Efficient Rainbow (DER); Overtrained Rainbow (OTRainbow) (Kielak, 2019); and Simulated Policy Learning (SimPLe) (Kaiser et al., 2020) (model-based). Moreover, we also compare with additional state-of-the-art off-policy baselines that make use of data augmentations: the aforementioned CURL and DrQ; and Self-Predictive Representations (SPR) (Schwarzer et al., 2020), the current state-of-the-art TD-learning based algorithm on this benchmark. SPR combines data augmentation with numerous additional algorithmic design choices, such as an auxiliary self-supervised loss for learning a latent dynamics model.

Results. We summarize the results in Table 4.6, showing the mean and median human-normalized scores together with the number of environments where each algorithm either achieves state-of-the-art or super-human performance. We include the full per-environment results in App. C.1.2. Remarkably, A-LIX obtains a substantially higher human-normalized mean performance than all other considered algorithms. While the recorded normalized median performance is slightly inferior to SPR, we argue that such difference is not particularly significant since this metric depends on the performance obtained in just two environments. Moreover, A-LIX achieves super-human performance in 7 games (the same as SPR), and state-of-the-art performance in 11 games, considerably more than all other algorithms. These

results corroborate how tuned architectures, data augmentation, and auxiliary losses used on ALE mostly serve the purpose of counteracting the direct implications of the visual deadly triad and show that A-LIX enables us to learn powerful models without relying on these design choices.

Statistical Significance

To validate the significance of our improvements, we statistically analyze our results using the *Rliable* tools and practices from Agarwal et al. (2021). We summarize some of our key findings in Fig. 4.13, showing the *probability of improvements* of A-LIX over prior methods (computed with the Mann-Whitney U statistic (Mann & Whitney, 1947)) and the relative normalized *performance profiles* (Dolan & Moré, 2002). The ranges correspond to 95% *stratified bootstrap confidence intervals* (Efron, 1992). In both DMC and Atari benchmarks, we find that our improvements are statistically significant (lower confidence intervals >0.5) and observe ‘stochastic dominance’ of our algorithm against almost all considered baselines (Dror et al., 2019). We provide further results and details of the employed statistical analysis in App. C.1.1 and App. C.1.3 respectively.

4.2.6 Related Work

Previous works have characterized several optimization issues related to performing RL via TD-learning (Baird & Moore, 1998; Baird, 1999). In this work, we instead focus on the *empirical analysis* of modern TD-learning algorithms, specific to the pixel-based RL setting. We also observe links with recent work studying *observational overfitting* (Song et al., 2020). Our work differs by focusing on memorization effects particular to the combination of CNNs and TD-learning. There are also connections with existing feature-level augmentation work, such as Dropout (Srivastava et al., 2014) and DropBlock (Ghiasi et al., 2018). In particular, the latter also applies structured transformations directly to the feature maps and introduces a heuristic to adjust this transformation over training, validating our findings on the utility of adaptivity. Outside RL, there is a rich body of work on

implicit regularization and memorization in CNNs (Keskar et al., 2017; Neyshabur et al., 2017; Arpit et al., 2017; Liu et al., 2020; Maennel et al., 2020). Rahaman et al. (2019) show that higher frequency data manifolds cause CNNs to learn higher spectral frequency terms, aligning with our analysis of higher frequency representations. Chatterjee (2020) show generalization arises when similar examples induce similar gradients during learning (i.e., coherence). Their work supports our findings since inconsistent feature gradients are a manifestation of non-coherence, explaining their poor generalization. Finally, our dual objective falls under automatic tuning methods in RL (AutoRL) (Parker-Holder et al., 2022). These approaches have been applied very successfully to manage non-stationary trade-offs, such as exploration and exploitation (Ball et al., 2020) and optimism (Moskowitz et al., 2021; Cetin & Celiktutan, 2021). Finally, we note links with recent work concerning implicit regularization in TD-learning (Kumar et al., 2021). However, while Kumar et al. (2021) observe an implicit ‘underfitting’ phenomenon in later training stages, we analyze an opposed ‘overfitting’ phenomenon occurring during the first training steps, which we find to be specific to learning from visual inputs.

4.2.7 Conclusion

In this work, we provide a novel analysis demonstrating that instabilities in pixel-based off-policy RL come specifically from performing TD-learning with a convolutional encoder in the presence of a sparse reward signal. We show this *visual deadly triad* affects the encoder’s gradients, causing the critic to *catastrophically self-overfit* to its own noisy predictions. Therefore, we propose **Adaptive Local Signal MiXing** (A-LIX), a powerful regularization layer to explicitly counteract this phenomenon. Applying A-LIX enables us to outperform prior state-of-the-art algorithms on popular benchmarks without relying on image augmentations, auxiliary losses, or other notable design choices.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Stabilizing Off-Policy Deep Reinforcement Learning from Pixels
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Edoardo Cetin*, Philip J. Ball*, Steve Roberts, Oya Celiktutan, Stabilizing Off-Policy Deep Reinforcement Learning from Pixels, <i>International Conference on Machine Learning 2022</i>

Student Confirmation

Student Name:	Philip Ball		
Contribution to the Paper	Proposed research direction, performed all analysis, established root cause, wrote majority of paper, produced all plots		
Signature 	Date	5/October/2023	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter Bibliography

- Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.
- Takayuki Akiyama, Hirotaka Hachiya, and Masashi Sugiyama. Efficient exploration through active learning for value function approximation in reinforcement learning. *Neural Networks*, 23(5):639–648, 2010. ISSN 0893-6080. doi: <https://doi.org/10.1016/j.neunet.2009.12.010>. URL <http://www.sciencedirect.com/science/article/pii/S0893608010000031>.
- Zeyuan Allen-Zhu and Yuanzhi Li. Feature purification: How adversarial training performs robust deep learning, 2021.
- Bilal Alsallakh, Narine Kokhlikyan, Vivek Miglani, Jun Yuan, and Orion Reblitz-Richardson. Mind the pad – CNNs can develop blind spots. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=m1CD7tPubNy>.
- Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 233–242. Pmlr, 06–11 Aug 2017. URL <https://proceedings.mlr.press/v70/arpit17a.html>.
- Leemon Baird. Reinforcement learning through gradient descent. Technical report, Carnegie-Mellon University, Department of Computer Science, 1999.
- Leemon Baird and Andrew Moore. Gradient descent for general reinforcement learning. *Advances in Neural Information Processing Systems*, 11, 1998.
- Philip Ball, Jack Parker-Holder, Aldo Pacchiano, Krzysztof Choromanski, and Stephen Roberts. Ready policy one: World building through active learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML*. 2020.
- Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.
- Richard Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957. URL <http://www.jstor.org/stable/24900506>.
- Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.
- Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

- Ronen I. Brafman and Moshe Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.*, 3(null):213–231, March 2003. ISSN 1532-4435. doi: 10.1162/153244303765208377. URL <https://doi.org/10.1162/153244303765208377>.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.
- Yuri Burda, Harrison Edwards, Deepak Pathak, Amos J. Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. URL <https://openreview.net/forum?id=rJNwDjAqYX>.
- Lucian Buşoniu, Tim de Bruin, Domagoj Tolić, Jens Kober, and Ivana Palunko. Reinforcement learning for control: Performance, stability, and deep approximators. *Annual Reviews in Control*, 46:8–28, 2018.
- Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.
- Edoardo Cetingin and Oya Celiktutan. Learning pessimism for robust and efficient off-policy reinforcement learning. *arXiv preprint arXiv:2110.03375*, 2021.
- Shayok Chakraborty, Vineeth Balasubramanian, and Sethuraman Panchanathan. Dynamic batch mode active learning. In *2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011*, pp. 2649–2656, 2011. ISBN 9781457703942. doi: 10.1109/cvpr.2011.5995715.
- Satrajit Chatterjee. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. *arXiv preprint arXiv:2002.10657*, 2020.
- Nuttapong Chentanez, Andrew G. Barto, and Satinder P. Singh. Intrinsically motivated reinforcement learning. In L. K. Saul, Y. Weiss, and L. Bottou (eds.), *Advances in Neural Information Processing Systems 17*, pp. 1281–1288. MIT Press, 2005. URL <http://papers.nips.cc/paper/2552-intrinsically-motivated-reinforcement-learning.pdf>.
- Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 4754–4765. 2018.
- Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, volume 87 of *Proceedings of Machine Learning Research*, pp. 617–629. Pmlr, 2018. URL <http://proceedings.mlr.press/v87/clavera18a.html>.
- John D. Co-Reyes, Yuxuan Liu, Abhishek Gupta, Benjamin Eysenbach, Pieter Abbeel, and Sergey Levine. Self-consistent trajectory autoencoder: Hierarchical reinforcement learning with trajectory embeddings. In *ICML*, 2018.

Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pp. 2020–2027. Pmlr, 2021.

Paul G. Constantine, Eric. Dow, and Qiqi. Wang. Active subspace methods in theory and practice: Applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4):A1500–a1524, 2014.

Marc Peter Deisenroth and Carl Edward Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML’11, pp. 465–472, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Elizabeth D Dolan and Jorge J Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

Rotem Dror, Segev Shlomov, and Roi Reichart. Deep dominance-how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2773–2785, 2019.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. Pmlr, 2016.

Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

Debidatta Dwibedi, Jonathan Tompson, Corey Lynch, and Pierre Sermanet. Learning actionable representations from visual observations. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1577–1584. Ieee, 2018.

Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in Statistics*, pp. 569–593. Springer, 1992.

Benjamin Eysenbach and Sergey Levine. If MaxEnt RL is the answer, what is the question?, 2019.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I. Jordan, Joseph E. Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *CoRR*, abs/1803.00101, 2018. URL <http://arxiv.org/abs/1803.00101>.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *arXiv preprint arXiv:1509.06113*, 25, 2015.

Daniel Freeman, David Ha, and Luke Metz. Learning to predict without looking ahead: World models without forward prediction. In *Advances in Neural Information Processing Systems 32*, pp. 5380–5391. Curran Associates, Inc., 2019. URL <http://papers.nips.cc/paper/8778-learning-to-predict-without-looking-ahead-world-models-without-forward-predict.pdf>.

- Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *ICML*, pp. 1582–1591, 2018. URL <http://proceedings.mlr.press/v80/fujimoto18a.html>.
- Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. DropBlock: A regularization method for convolutional networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL <https://proceedings.neurips.cc/paper/2018/file/7edcfb2d8f6a659ef4cd1e6c9b6d7079-Paper.pdf>.
- David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NeurIPS’18, pp. 2455–2467, Usa, 2018. Curran Associates Inc. URL <http://dl.acm.org/citation.cfm?id=3327144.3327171>.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018a. Pmlr. URL <https://proceedings.mlr.press/v80/haarnoja18b.html>.
- Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018b. URL <http://arxiv.org/abs/1812.05905>.
- Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565. Pmlr, 2019.
- Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=S110TC4tDS>.
- Mikael Henaff. Explicit explore-exploit algorithms in continuous state spaces. In *Advances in Neural Information Processing Systems 32*, pp. 9372–9382. Curran Associates, Inc., 2019. URL <http://papers.nips.cc/paper/9135-explicit-explore-exploit-algorithms-in-continuous-state-spaces.pdf>.
- Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *CoRR*, abs/1709.06560, 2017. URL <http://arxiv.org/abs/1709.06560>.
- Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

- Rein Houthooft, Xi Chen, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems 29*. 2016.
- Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In Carlos A. Coello Coello (ed.), *Learning and Intelligent Optimization*, pp. 507–523, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-25566-3.
- 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>.
- Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.
- Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłoś, Błażej Osipiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for Atari. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=S1xCPJHtDB>.
- Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.
- Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL <https://openreview.net/forum?id=H1oyRlYgg>.
- Kacper Piotr Kielak. Do recent advancements in model-based deep reinforcement learning really improve data efficiency? 2019.
- Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*. 2021.
- Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=09bnihsFfXU>.
- Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=SJJinbWRZ>.

- Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems 33*. 2020a.
- Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020b.
- Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019.
- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Neural Information Processing Systems*, 2018.
- Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- Shengchao Liu, Dimitris Papailiopoulos, and Dimitris Achlioptas. Bad global minima exist and SGD can reach them. *Advances in Neural Information Processing Systems*, 33, 2020.
- Manuel Lopes, Tobias Lang, Marc Toussaint, and Pierre yves Oudeyer. Exploration in model-based reinforcement learning by empirically estimating learning progress. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 206–214. Curran Associates, Inc., 2012.
- Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.
- D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, July 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.4.590.
- David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, Usa, 2002. ISBN 0521642981.
- Hartmut Maennel, Ibrahim M Alabdulmohsin, Ilya O Tolstikhin, Robert Baldock, Olivier Bousquet, Sylvain Gelly, and Daniel Keysers. What do neural networks learn when trained with random labels? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19693–19704. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/e4191d610537305de1d294adb121b513-Paper.pdf>.

- H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947. doi: 10.1214/aoms/1177730491. URL <https://doi.org/10.1214/aoms/1177730491>.
- Shakir Mohamed and Danilo J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, Nips’15, pp. 2125–2133, Cambridge, MA, USA, 2015. MIT Press.
- Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael I. Jordan. Tactical optimism and pessimism for deep reinforcement learning, 2021. URL <https://openreview.net/forum?id=a4WgjcLeZIn>.
- Anusha Nagabandi, Kurt Konoglie, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation, 2019.
- Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Nips’17, pp. 5949–5958, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
- Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML’17, pp. 2721–2730. JMLR.org, 2017.
- Thomas Osugi, Deng Kun, and Stephen Scott. Balancing exploration and exploitation: A new algorithm for active machine learning. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, Icdm ’05, pp. 330–337, Usa, 2005. IEEE Computer Society. ISBN 0769522785. doi: 10.1109/icdm.2005.33. URL <https://doi.org/10.1109/ICDM.2005.33>.
- Jack Parker-Holder, Raghu Rajan, Xingyou Song, André Biedenkapp, Yingjie Miao, Theresa Eimer, Baohe Zhang, Vu Nguyen, Roberto Calandra, Aleksandra Faust, Frank Hutter, and Marius Lindauer. Automated reinforcement learning (AutoRL): A survey and open problems, 2022.
- Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 2778–2787, 2017. URL <http://proceedings.mlr.press/v70/pathak17a.html>.
- Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 5062–5071, 2019. URL <http://proceedings.mlr.press/v97/pathak19a.html>.
- Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural

- networks. In *International Conference on Machine Learning*, pp. 5301–5310. Pmlr, 2019.
- Mihaela Rosca, Theophane Weber, Arthur Gretton, and Shakir Mohamed. A case for new neural networks smoothness constraints. In *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop*, 2020. URL https://openreview.net/forum?id=_b-uT9wCI-7.
- G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report Tr 166, Cambridge University Engineering Department, Cambridge, England, 1994.
- Noor Sajid, Philip J. Ball, Thomas Parr, and Karl J. Friston. Active Inference: Demystified and Compared. *Neural Computation*, 33(3):674–712, 03 2021. ISSN 0899-7667. doi: 10.1162/neco_a_01357. URL https://doi.org/10.1162/neco_a_01357.
- Tom Schaul, Georg Ostrovski, Iurii Kemaev, and Diana Borsa. Return-based scaling: Yet another normalisation trick for deep RL, 2021.
- J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, Sep. 2010. ISSN 1943-0612. doi: 10.1109/tamd.2010.2056368.
- Jürgen Schmidhuber. Curious model-building control systems. *[Proceedings] 1991 IEEE International Joint Conference on Neural Networks*, pp. 1458–1463 vol.2, 1991.
- Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model, 2019.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pp. 1889–1897, 2015.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <http://arxiv.org/abs/1707.06347>.
- Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.
- Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.
- Pranav Shyam, Wojciech Jaskowski, and Faustino Gomez. Model-based active exploration. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 5779–5788, 2019. URL <http://proceedings.mlr.press/v97/shyam19a.html>.

- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.
- Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL <http://jmlr.org/papers/v15/srivastava14a.html>.
- Yi Sun, Faustino Gomez, and Jürgen Schmidhuber. Planning to be surprised: Optimal Bayesian exploration in dynamic environments. In *Proceedings of the 4th International Conference on Artificial General Intelligence*, Agi’11, pp. 41–51, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 9783642228865.
- Richard Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 08 1988. doi: 10.1007/bf00115009.
- Richard S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *In Proceedings of the Seventh International Conference on Machine Learning*, pp. 216–224. Morgan Kaufmann, 1990.
- Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bull.*, 2(4):160–163, July 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL <https://doi.org/10.1145/122344.122377>.
- Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1063, 2000.
- Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.
- Hado Van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.
- Hado P van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? *Advances in Neural Information Processing Systems*, 32: 14322–14333, 2019.
- Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

- Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *CoRR*, abs/1907.02057, 2019. URL <http://arxiv.org/abs/1907.02057>.
- Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10674–10681, May 2021. URL <https://ojs.aaai.org/index.php/AAAI/article/view/17276>.
- Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.
- Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real-world robotic reinforcement learning. *arXiv preprint arXiv:2004.12570*, 2020.

5

Offline Data Strategies to Augment Online Policy Deployment

Contents

5.1	Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment	108
5.1.1	Introduction	109
5.1.2	Related Work	111
5.1.3	Preliminaries	113
5.1.4	Augmented World Models with Self Supervised Policy Adaptation	115
5.1.5	Experiments	118
5.1.6	Conclusion and Future Work	126
5.1.7	Errata	127
5.2	Learning General World Models in a Handful of Reward-Free Deployments	129
5.2.1	Introduction	129
5.2.2	Problem Statement	132
5.2.3	Coordinated Active Sample Collection	134
5.2.4	Theoretical Motivation	137
5.2.5	Experiments	138
5.2.6	Exploring Worlds	139
5.2.7	Related Work	145
5.2.8	Conclusion	147
5.3	Efficient Online Reinforcement Learning with Offline Data	150
5.3.1	Introduction	150
5.3.2	Related work	153
5.3.3	Preliminaries	154

5.3.4	Online RL with Offline Data	155
5.3.5	Experiments	161
5.3.6	Conclusion	169
5.3.7	Errata	170
Chapter Bibliography		172

In this chapter, we investigate different approaches and problems associated with incorporating prior offline data when further online deployment is required. As aforementioned, the inclusion of offline data is not part of the canonical online RL formulation. As a result, there are many additional considerations when training with such data, in particular when considering the use of neural-network function approximators.

The work presented in this chapter seeks to answer 3 vital questions on this relatively nascent topic: 1) What if there is a co-variate shift which occurs between the training data and the testing environment?; 2) What is the optimal way to train a population of agents on an offline dataset to encourage subsequent improvement?; 3) Why does standard off-policy RL fail to learn useful behaviors when supplied with offline data?

5.1 Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment

Abstract Reinforcement learning from large-scale offline datasets provides us with the ability to learn policies without potentially unsafe or impractical exploration. Significant progress has been made in the past few years in dealing with the challenge of correcting for differing behavior between the data collection and learned policies. However, little attention has been paid to potentially changing dynamics when transferring a policy to the online setting, where performance can be up to 90% reduced for existing methods. In this paper we address this problem with *Augmented World Models* (AugWM). We augment a learned dynamics model with simple transformations that seek to capture potential changes in physical properties

of the robot, leading to more robust policies. We not only train our policy in this new setting, but also provide it with the sampled augmentation as a context, allowing it to adapt to changes in the environment. At test time we *learn the context* in a self-supervised fashion by approximating the augmentation which corresponds to the new environment. We rigorously evaluate our approach on over 100 different changed dynamics settings, and show that this simple approach can significantly improve the zero-shot generalization of a recent state-of-the-art baseline, often achieving successful policies where the baseline fails.

5.1.1 Introduction

Offline reinforcement learning (RL) describes the problem setting where RL agents learn policies solely from previously collected experience without further interaction with the environment (Fujimoto et al., 2019; Levine et al., 2020). This could have tremendous implications for real-world problems (Dulac-Arnold et al., 2019), with the potential to leverage rich datasets of past experience where exploration is either not feasible (e.g. a Mars Rover) or unsafe (e.g. in medical settings). As such, interest in offline RL has surged in recent times.

This work focuses on model-based offline RL, which has achieved state-of-the-art performance through the use of uncertainty penalized updates (Yu et al., 2020; Kidambi et al., 2020). However, existing work only addresses the issue of transferring from different behavior policies in the *same environment*, ignoring any possibility of distribution shift. Consider the case where it is expensive to collect data, and we have access to a single dataset from a robot. Using existing methods, we would be unable to make any changes that impact the dynamics, such as using a newer model of the robot or deploying it in a different room.

A related setting is the Sim2Real problem, which considers transferring an agent from a simulated environment to the real world. A popular recent approach is domain randomization (Tobin et al., 2017; James et al., 2017), the process of randomizing non-essential regions of the observation space to make agents robust to ‘observational overfitting’ (Song et al., 2020). Indeed, methods seeking to generalize

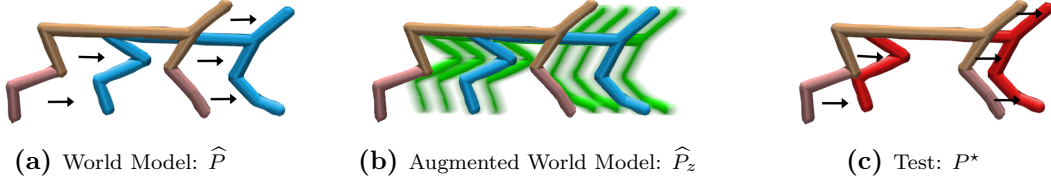


Figure 5.1: An illustration of our approach, each figure shows a transition $P(s, a) \rightarrow s'$. In a) we show the World Model dynamics $\hat{P}$, trained from $\mathcal{D}_{\text{env}}$, a single offline dataset. In b) we show the Augmented World Model, blue represents $\hat{P}$, while each green agent illustrates an instantiation of augmented dynamics, which is sampled at each timestep: $\hat{P}_z, z \sim \mathcal{Z}$. The goal is to approximate c) where we show an unseen test environment, with transition dynamics P^* .

to novel dynamics have also shown promise (Peng et al., 2018), by randomizing physical properties such as the mass of the agent. A significant limitation of these approaches is the requirement for a simulator, which may not be available.

In this work we take inspiration from Sim2Real to generalize solely from an offline dataset, in a learned simulator or *World Model* (WM). We therefore describe our problem setting as follows: an agent must learn to generalize to unseen test-time dynamics whilst having access to offline data from *only a single environment*; we call this “dynamics generalization from a single offline environment”.

In this paper we concentrate on the zero-shot performance of our agents to unseen dynamics, as it may not be practical (nor safe) to perform multiple rollouts at test time. To tackle this problem, we propose a novel form of data augmentation: rather than augment observations, we focus on *augmenting the dynamics*. We first learn a world model of the environment, and then augment the transition function at policy training time, making the agent train under different imagined dynamics. In addition, our agent is given access to the augmentation itself as part of the observations, allowing it to consider the context of modified dynamics.

At test time we propose a simple, yet surprisingly effective, self-supervised approach to learning an agent’s augmentation context. We learn a linear dynamics model which is then used to approximate the dynamics augmentation induced by the modified environment. This context is then given to the agent, allowing it to adapt on the fly to the new dynamics within a single episode (i.e. zero-shot). We show that our approach is capable of training agents that can vastly outperform existing Offline RL methods on the “dynamics generalization from a single offline

environment” problem. We also note that this approach does *not* require access to environment rewards at test time. This facilitates application to Sim2Real problems whereby test time rewards may not be available.

Our contributions are twofold: 1) As far as we are aware, we are the first to propose **dynamics augmentation for model based RL**, allowing us to generalize to changing dynamics despite only training on a single setting. We do this without access to any environment parameters or prior knowledge. 2) We propose a simple **self-supervised context adaptation** reward-free algorithm, which allows our policy to use information from interactions in the environment to vary its behavior in a single episode, increasing zero-shot performance. We believe both of these approaches are not only novel, but offer significant improvement v.s. state-of-the-art methods, improving generalization and providing a promising approach for using offline RL in the real world.

5.1.2 Related Work

In this work we focus on Model Based RL (MBRL). A key challenge in MBRL is that an inaccurate model can be exploited by the policy, leading to behaviors that fail to transfer to the real environment. As such, a swathe of recent works have made use of model ensembles to improve robustness (Kurutach et al., 2018; Chua et al., 2018; Clavera et al., 2018; Janner et al., 2019; Ball et al., 2020). With increased accuracy, MBRL has recently been shown to be competitive with model free methods in continuous control (Ha & Schmidhuber, 2018; Chua et al., 2018; Janner et al., 2019) and games (Schrittwieser et al., 2019; Kaiser et al., 2020). We make use of an ensemble of probabilistic dynamics models, first introduced in Lakshminarayanan et al. (2017) and subsequently used in Chua et al. (2018).

In this paper we focus on Model-Based *offline* RL, where MOPO (Yu et al., 2020) and MOReL (Kidambi et al., 2020) have recently demonstrated the effectiveness of learned dynamics models, using model uncertainty to constrain policy optimization. We build upon this approach for zero-shot dynamics generalization from offline data. There have also been successes in off policy methods for offline RL (Wu

et al., 2019; Fujimoto et al., 2019; Kumar et al., 2020; Rudner et al., 2021) and context based approaches (Ajay et al., 2021), although these works only consider tasks within the support of the offline dataset. Finally, MBOP (Argenson & Dulac-Arnold, 2021) addresses the problem of goal-conditioned zero-shot transfer from offline datasets. However, their goal-conditioning relies on unchanged dynamics in the test environment.

In online RL, recent work has achieved strong dynamics generalization with a learned model (Seo et al., 2020). However, this required training under varied dynamics, assigning different experiences to models. In addition, this work used MPC whereas we train a policy inside the model, which is significantly faster at deployment time. Also related are Clavera et al. (2019); Nagabandi et al. (2019), where the model is trained to quickly adapt to new dynamics $P(s'|s, a)$, however both these works place more emphasis on model-adaption rather than zero-shot policy performance. Furthermore, access to an underlying task distribution is required, something we do not have in our offline setting. Also similar to our work is the recently proposed Policy Adaptation during Deployment (PAD, (Hansen et al., 2021)) approach. Our approach differs in that we learn a context, whereas PAD uses an auxiliary objective to adapt its features. In addition, PAD considers the *online model free* setting, while our method is *offline and model based*.

Sim2Real is the setting where an agent trained in a simulator must transfer to the real world. A common approach to solve this problem is through domain randomization (Tobin et al., 2017; James et al., 2017), whereby parameters in the simulator are varied during training. This has shown to be effective for dynamics generalization (Andrychowicz et al., 2020; Antonova et al., 2017; Peng et al., 2018; Yu et al., 2017; Zhou et al., 2019; OpenAI et al., 2019), but requires access to a simulator which we do not have. Another form of domain randomization, data augmentation, has proved to be effective for training RL policies (Laskin et al., 2020a;b; Kostrikov et al., 2021; Raileanu et al., 2020), resulting in improved efficiency and generalization. So far, these works have focused on online model free methods, and used data

augmentation on the *state* space, reducing *observational* overfitting (Song et al., 2020). In contrast, we focus on offline MBRL and instead augment the *dynamics*.

We also note clear links to contextual MDPs (Hallak et al., 2015; Modi et al., 2018) and hidden parameter MDPs (HiP-MDP) (Doshi-Velez & Konidaris, 2016; Killian et al., 2017; Zhang et al., 2021a) settings, whereby our self-supervised dynamics embedding can be considered as a context/hidden parameter. However, in these settings the embedding is chosen at the beginning of each episode and is fixed throughout, whereas our embedding varies per timestep.

We are not the first to propose data augmentation in the MBRL setting, Pitis et al. (2020) proposed Counterfactual data augmentation for improving performance in the context of locally factored tasks. Approaches to ensuring adversarial robustness can include data augmentations that assist with out-of-domain generalization, as opposed to observational overfitting. In Volpi et al. (2018) this is done without a simulator and from a single source of data, however they only work on supervised learning problems and require an adversary to be learned, adding computational complexity. Finally, Wellmer & Kwok (2021) concurrently explore the idea of augmenting world model dynamics for improved test-time transferability, however they focus on in-domain generalization, and do not infer context at test time.

5.1.3 Preliminaries

We consider a Markov Decision Process (MDP), defined as a tuple $M = (\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state space and action space respectively, $P(s'|s, a)$ the transition dynamics, $R(s, a)$ the reward function, ρ_0 the initial state distribution, and $\gamma \in (0, 1)$ the discount factor. The goal in RL is to optimize a policy $\pi(a|s)$ that maximizes the expected discounted return $\mathbb{E}_{\pi, P, \rho_0} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$. The value function $V^\pi(s) := \mathbb{E}_{\pi, P} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s]$ gives the expected discounted return under π when starting from state s . In *offline RL*, the policy is not deployed in the environment until test time. Instead, the algorithm only has access to a static dataset $\mathcal{D}_{env} = \{(s, a, r, s')\}$, collected by one or more behavioral policies π_b . We

borrow notation from Yu et al. (2020) and refer to the distribution from which $\mathcal{D}_{env}$ was sampled as the *behavioral distribution*.

When training a model, we follow MBPO (Janner et al., 2019) and MOPO (Yu et al., 2020) and train an ensemble of N probabilistic dynamics models (Nix & Weigend, 1994). Each model learns to predict both next state s' and reward r from a state-action pair, using $\mathcal{D}_{env}$ in a supervised fashion. Furthermore, each model outputs a Gaussian $\hat{P}_i(s_{t+1}, r_t | s_t, a_t) = \mathcal{N}(\mu(s_t, a_t), \Sigma(s_t, a_t))$. The resulting model $\hat{P}$ defines a *model MDP* $\hat{M} = (\mathcal{S}, \mathcal{A}, \hat{P}, \hat{R}, \rho_0, \gamma)$, where $\hat{R}$ refers to the learned reward model.

To train the policy, we use k step rollouts inside $\hat{M}$, adding experience to a replay buffer $\mathcal{D}_{\hat{env}}$ to learn an action-value function and a policy, using Soft Actor Critic (SAC (Haarnoja et al., 2018b)). SAC alternates between a soft policy evaluation step, which estimates $Q^\pi(s, a) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t (R(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot | s_t))) | s_0 = s, a_0 = a]$ using Bellman backups (where α is a temperature parameter for policy entropy $\mathcal{H}$), and a policy improvement step which learns a policy by minimizing the expected KL divergence $J_\pi(\phi, \mathcal{D}) = \mathbb{E}_{s_t \sim \mathcal{D}_{\hat{env}}} [D_{KL}(\pi \| \exp\{\frac{1}{\alpha}\{Q^\pi - V^\pi\}\})]$. Note that the SAC algorithm is unchanged from the model-free setting, aside from the environment being a learned model, and the rollout horizon k being truncated. Perhaps surprisingly, this approach alone produces strong results in the offline setting. However, MOPO (Yu et al., 2020) and MOREL (Kidambi et al., 2020) show improvement in performance by penalizing rewards in regions of the state space where the ensemble of probabilistic models is less certain. This implicitly reduces reliance on samples which deviate beyond the support of $\mathcal{D}_{env}$.

While MOPO and MOREL have addressed the issue of training a policy in $\mathcal{D}_{env}$, and transferring to the true environment M , they only consider where the data in $\mathcal{D}_{env}$ is actually drawn from P . However, sometimes this may not be sufficient for deployment. For example, a robot could fail to walk when learning from data that was collected by a different version of the robot (with different mass), or if the same robot collected data but in a different room to deployment (with varied friction). It is this setting, where dynamics may vary at test time,

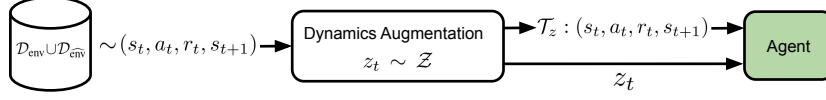


Figure 5.2: Training policies in Augmented World Models. For each state-action pair sampled from the buffer, a new augmentation $z_t \sim \mathcal{Z}$ is sampled to produce an augmentation operator $\mathcal{T}_z$, which is applied to the transition. The policy is then trained with the new tuple of data, with the context concatenated to the state.

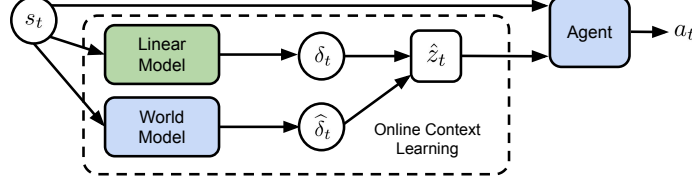


Figure 5.3: Self-supervised policy adaptation via learned context. At each timestep, the state s_t is fed into a linear model (trained online at each timestep) to predict the change in state δ_t , and also passed to the fixed World Model (trained on the offline data) to predict the change in state under $\hat{P}$, $\hat{\delta}_t$. The approximate context is then $z_t = \delta_t / \hat{\delta}_t$, which is concatenated with s_t and passed to the agent to produce an action.

that is the focus of our work. To learn successfully, we propose a novel approach to training robust context-dependent policies.

5.1.4 Augmented World Models with Self Supervised Policy Adaptation

In this section we introduce our algorithm: Augmented World Models (AugWM). We first discuss our training procedure (Fig. 5.2) before moving onto our self-supervised approach to online context learning (Fig. 5.3).

Augmented World Models

Rather than seeking to transfer our policy from $\hat{M}$ to M , we instead wish to transfer from $\hat{M}$ to M^* , where $M^* = (\mathcal{S}, \mathcal{A}, P^*, R^*, \rho_0, \gamma)$ is an unseen environment with *different dynamics*. Thus, our emphasis shifts to producing experience inside the world model such that our agent is able to generalize to unseen, out of distribution dynamics. We approach this problem by training our policy in an *Augmented World Model*. Formally, we denote an augmentation as $z_t \sim \mathcal{Z}, z \in \mathbb{R}^{|\mathcal{S}|}$, which is sampled at each timestep to produce an augmentation operator $\mathcal{T}_z$. $\mathcal{T}_z$ is applied to (s, a, r, s') tuples from the dataset $\mathcal{D}$, and when used with tuples sampled from $\mathcal{D}_{\widehat{\text{env}}}$ indirectly induces an augmented distribution $\hat{P}_z$. In principle, we wish to produce

augmentations such that the true modified environment dynamics P^* lies in the support of the distribution of augmented dynamics, i.e.

$$\inf_z D(\hat{P}_z(s, a) \| P^*(s, a)) \leq \epsilon \quad (5.1)$$

for all s, a , some small $\epsilon > 0$, and suitable distance/divergence metric D . We consider several augmentations, beginning with existing works before moving to new approaches which specifically target the problem of dynamics generalization. We begin with **Random Amplitude Scaling** as in Laskin et al. (2020a), which we refer to as RAD. RAD scales both s_t and s_{t+1} as follows:

$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (z \odot s_t, a_t, r_t, z \odot s_{t+1}) \quad (5.2)$$

for $z \sim \text{Unif}([a, b]^{|S|})$. Given that our focus is on changing dynamics (vs. observational overfitting), we also propose to scale only the next state, i.e., **Random Amplitude Nextstate Scaling** (RANS):

$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (s_t, a_t, r_t, z \odot s_{t+1}) \quad (5.3)$$

for $z \sim \text{Unif}([a, b]^{|S|})$. Note that while RANS is more focused on augmenting dynamics than RAS, it still suffers from a dependence on the magnitude of s_{t+1} . As such, we further propose a more targeted augmentation, which we call **Dynamics Amplitude Sampling** (DAS). Rather than directly scale the state, DAS scales the *change in the state* which we denote with $\delta_t = s_{t+1} - s_t$, as follows:

$$\mathcal{T}_z : (s_t, a_t, r_t, s_{t+1}) \mapsto (s_t, a_t, r_t, s_t + z \odot \delta_t) \quad (5.4)$$

for $z \sim \text{Unif}([a, b]^{|S|})$. The full training procedure is shown in Algorithm 4.

Algorithm 4 Augmented World Models: Training

Input: Offline data $\mathcal{D}_{\text{env}}$, Penalty λ , horizon h , batchsize B , augmentation $\mathcal{Z}$.

Initialize: Ensemble of N dynamics models $\hat{P}$, policy π . Replay buffer $\widehat{\mathcal{D}}_{\text{env}}$

1. Train $\hat{P}$ in a supervised fashion using $\mathcal{D}_{\text{env}}$.

for $epoch = 1, 2, \dots$ **do**

- (i) Sample initial states: $\{s_1^1, \dots, s_1^B\} \sim \mathcal{D}_{\text{env}}$
- (ii) Rollout policy (in parallel), using λ -penalized reward, storing all data in $\widehat{\mathcal{D}}_{\text{env}}$
- (iii) Train policy using $\mathcal{D}_{\text{env}} \cup \widehat{\mathcal{D}}_{\text{env}}$. For each (s, a, r, s') , sample $z \sim \mathcal{Z}$ and apply $\mathcal{T}_z$.

Return: Policy π

One crucial addition to our method is the use of context. Concretely, when we are optimizing the policy using a batch of data, we concatenate the next state with the augmentation vector z . This allows our policy to be informed of the specific augmentation that was applied to the environment and thus behave accordingly. However, at test time we do not know z , so what can we use? Next we propose a solution to this problem, learning the context on the fly.

Self-Supervised Policy Selection

In the meta-learning literature there have been many recent successes making use of a *learned context* to adapt a policy at test time to a new environment (Rakelly et al., 2019; Zintgraf et al., 2020; 2021), typically using a blackbox model with a latent state. Crucially, these approaches require several episodes to adapt at test time, making them unfeasible in our zero-shot setting. What makes our setting unique is we explicitly know what the context represents: a linear transformation of s_t , or δ_t . Using this insight, we are able to learn an effective context on the fly at test time. Concretely, we observe that from a state s_t drawn from M^* , we can sample an action $a_t \sim \pi$ and then compute an approximate $\hat{s}_{t+1}$ using our model $\hat{P}$. With $\hat{s}_{t+1}$, we have a sample estimate of the *state change under $\hat{P}$* , i.e. $\hat{\delta}_t = \hat{s}_{t+1} - s_t$. We can make this approximation of the next state without interacting with the environment, but once we do take the action a_t in the environment, we then receive the *true next state* s_{t+1} and can store the true difference $\delta_t = s_{t+1} - s_t$. Using the DAS augmentation, we can approximate z as $\delta_t/\hat{\delta}_t$.

Algorithm 5 Augmented World Models: Testing

Input: Initial state s_1 , horizon H , policy π , world model $\hat{P}$, initial context $\hat{z} = \mathbf{1}^{|S|}$

Initialize: Linear model f_ψ , dataset $\mathcal{D} = \emptyset$, return $R_0 = 0$.

for $step = 1, 2, \dots H$ **do**

 Select action: $a_t \sim \pi(s_t, \hat{z}_t)$

 Take action: $s_{t+1} \sim P^*(s_t, a_t)$

 Update return: $R_{t+1} = R_t + R^*(s_t, a_t, s_{t+1})$

 Update Dataset: $\mathcal{D} \cup (s_t, \delta_t)$.

 Update Linear model by minimizing $\mathcal{L}_{\text{MSE}}(\psi, \mathcal{D})$.

 Predict new context using $\hat{z}_t = \delta_t/\hat{\delta}_t$.

Return: R_T

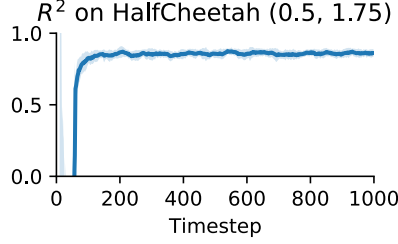


Figure 5.4: Mean R^2 of the linear model across 20 rollouts.

This however is retrospective; we can only approximate z having already seen the next state, by which time our agent has already acted. Furthermore, we believe under changed dynamics the true z likely depends on s , thus we cannot use a previous z for future timesteps. Therefore, we learn a forward dynamics model using the data collected *during* the test rollout. After h timesteps in the environment, we have the following dataset: $\mathcal{D} = \{(s_1, s_2), \dots, (s_{h-1}, s_h)\}$. This allows us to learn a simple dynamics model $f_\psi : (s_t) \mapsto \delta_t = s_{t+1} - s_t$, by minimizing the mean squared error $\mathcal{L}_{\text{MSE}}(\psi, \mathcal{D})$. Notably, since we never actually plan with this model, it does not need to be as accurate as a typical dynamics model in MBRL. Instead, it is crucial that the model learns quickly enough such that we can use it in a zero-shot evaluation. Thus, we choose to use a simple linear model for f . To show the effectiveness of this, in Fig. 5.4 we show the mean R-squared of linear models learned on the fly during evaluation rollouts.

We observe that in less than 100 timesteps the linear model achieves high accuracy on the test data. Subsequently, equipped with f_ψ , we can approximate δ_t , and predict the augmentation as $\hat{z}_t = \delta_t / \hat{\delta}_t$. We then provide the agent with $\hat{z}_t$ to compute action a_t . The full procedure is shown in Algorithm 5.

5.1.5 Experiments

In our experiments, we aim to investigate the effectiveness of our approach for zero-shot dynamics generalization from a single offline dataset. To assess this, we will answer a series of questions, beginning with a question on the necessity of our method:

Do we really need to develop methods specifically for dynamics generalization?

To answer this, we train MOPO using offline data from a single environment, and test it under changed dynamics. We consider the HalfCheetah environment from the OpenAI Gym (Brockman et al., 2016), using offline data from D4RL (Fu et al., 2020). We train a MOPO agent using the mixed dataset, using our own implementation of the algorithm (but using the same hyperparameters as the original authors). To test the trained policy, we vary both the mass of the agent and damping coefficient by a multiplicative factor.¹ In this work we consider a grid of the following values for HalfCheetah: $\{0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75\}$ and $\{0.5, 0.75, 1.0, 1.25, 1.5\}$ for Walker2d, representing a significant out-of-distribution shift.

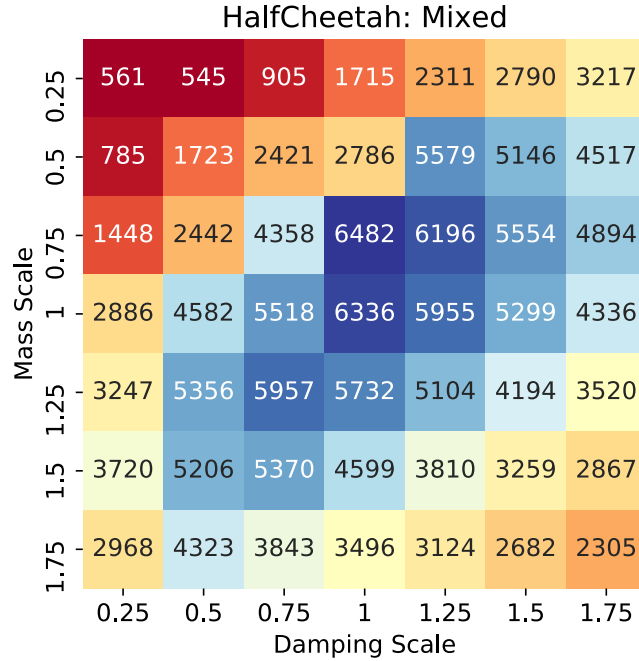


Figure 5.5: Mean performance for 5 seeds for MOPO on HalfCheetah environments with varying dynamics. Note the central cell (1,1) corresponds to the in-sample data.

The results (Fig. 5.5), show that MOPO performance is clearly impacted by changing dynamics. We see in the central cell, that performance for our version of MOPO matches the author results (Yu et al., 2020), and in some cases we even see small gains (e.g. mass = 0.75, damping = 1.0). However, on the top left we see dramatically weaker performance, often below 1k, indicating the robot is failing to

¹The standard environment (both in Gym and D4RL) corresponds to both these values being set to 1.0.

achieve locomotion. Before evaluating AugWM, we first test whether training with the “correct” augmentation improves generalization performance. In short, we ask:

Is augmenting dynamics even worthwhile?

To answer this, we train SAC for 1×10^5 steps and save the states visited in the ‘true’ environment. We then use these starting states to train a policy using an offline MBPO² approach with AugWM. However, instead of sampling $z_t \sim \mathcal{Z}$, we provide the actual $z = \delta^*/\hat{\delta}$ as we have access to the ‘true’ and ‘modified’ environments; we refer to this as an *oracle* version of our method, and is designed to assess the viability of our approach. Note that we *do not* augment the ‘true’ environment rewards. We consider two baselines: a) offline MBPO in the ‘true’ environment; b) online SAC in the ‘modified’ environment. We train MBPO until convergence, and SAC for 1×10^5 steps.

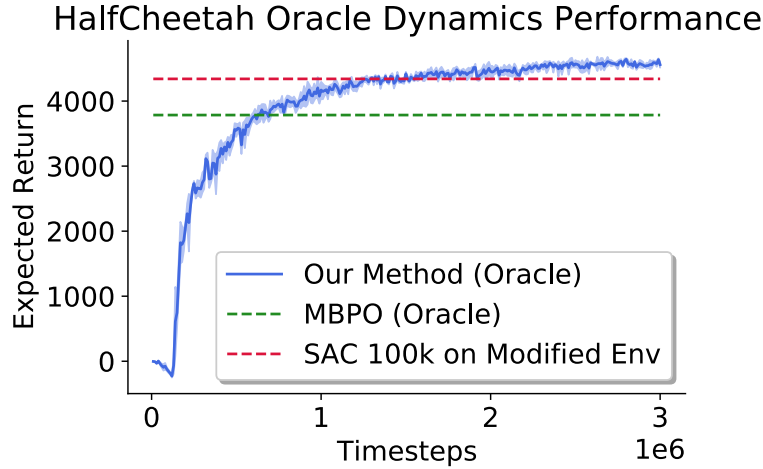


Figure 5.6: Mean performance for 3 seeds for MBPO with a dynamics oracle on HalfCheetah with $1.5\times$ mass and damping.

As shown Fig. 5.6, when provided with the true z , AugWM outperforms both baselines. The SAC result is surprising: the baseline agent was trained directly on the ‘modified’ environment for the same number of steps as the policy that generated our oracle starting states. One explanation is the greater exploration induced by the ‘easier’ dynamics of the ‘true’ environment. This validates our approach; if we augment the dynamics $\hat{P}$ from a model correctly, we can generalize to unseen

²Since we have access to the true environment, there is no need for the MOPO penalty.

dynamics. In other words, neither the starting states nor rewards need to be from the ‘modified’ environment. With this in mind, our next question is a simple one:

Which augmentation strategy is most effective?

To test this, we train as in Algorithm 4, *without* context, to isolate the effectiveness of the training process. We use the HalfCheetah Mixed dataset and train a MOPO agent, augmenting either both s and s' (RAD), just s' (RANS) or just δ (DAS).

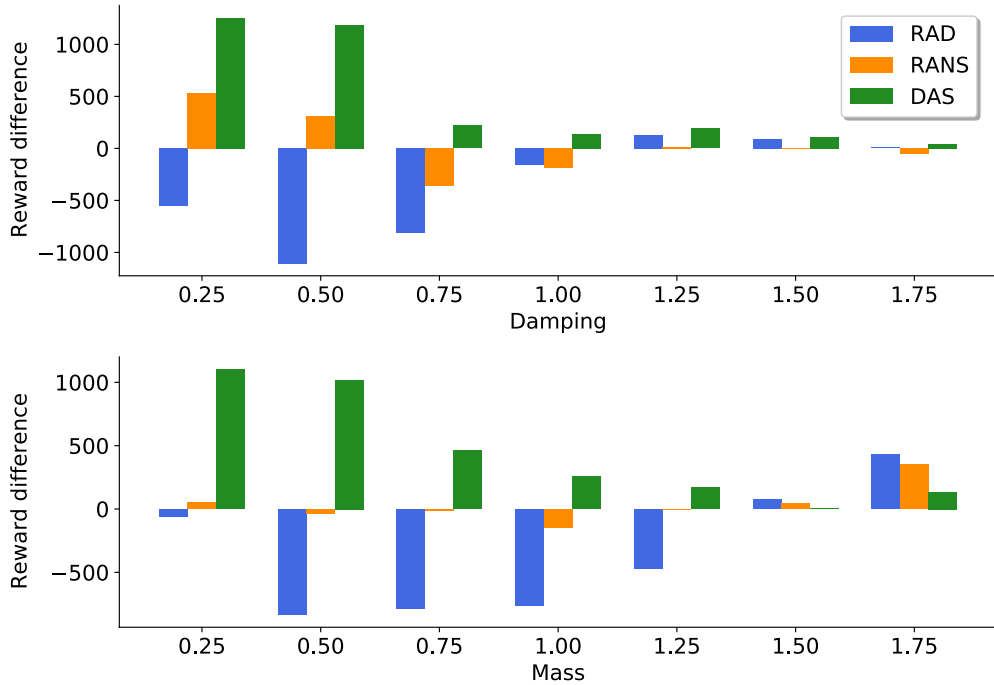


Figure 5.7: Average performance gains of different World Model augmentations over base MOPO for different levels of damping and mass in the HalfCheetah test environment (5 seeds).

The results are shown in Fig. 5.7. As we see, the RAD augmentation fails to improve dynamics generalization, actually leading to worse performance overall. RANS does improve performance on unseen dynamics, as we are influencing the *dynamics*, not just the observation. However, DAS clearly provides the strongest performance. As a result, we use DAS for AugWM. Our final algorithm design question is as follows:

Does training with context improve performance?

To answer this question, we return to the HalfCheetah Mixed setting from Fig. 5.7, taking the policy trained with DAS. We now train two additional agents: 1) Default Context: at *train* time the agent is provided with the DAS augmentation z as context, at *test* time it is provided with a vector of ones, $\mathbf{1}^{|S|}$; 2) Learned Context: trained as in 1), but context is learned online using Algorithm 5.

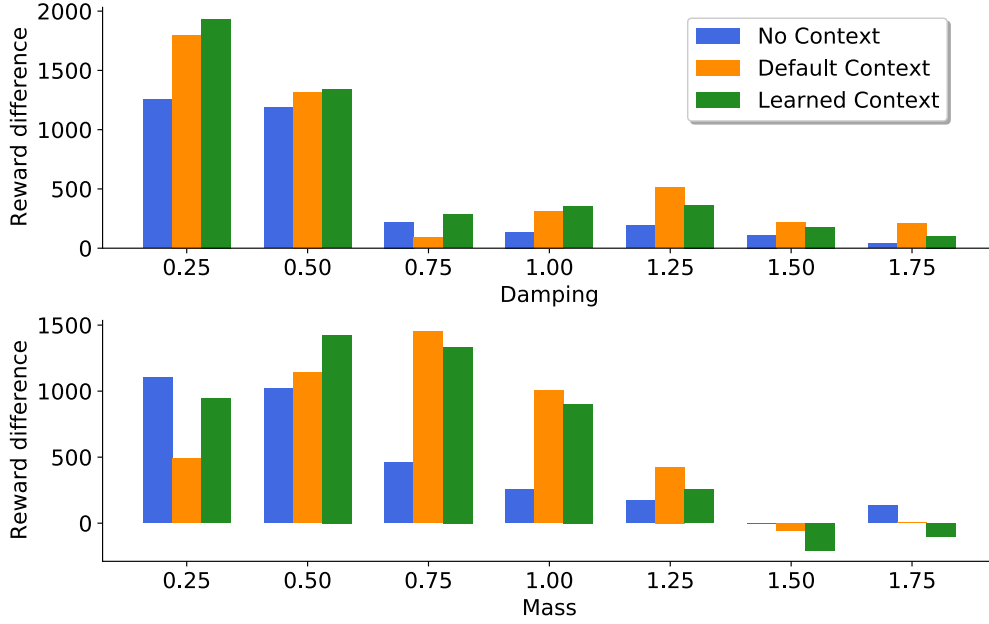


Figure 5.8: Average performance gains of adding contexts over base MOPO for different levels of damping and mass in the HalfCheetah test environment (5 seeds).

The results are shown in Fig. 5.8. We observe that training with context (orange) improves performance on average, while adapting the context on the fly (green) leads to further gains (+80 on average). These methods combine to produce our AugWM algorithm. We are now ready for the final question:

Can Augmented World Models improve zero-shot generalization?

To answer this question we perform a rigorous analysis, using multiple benchmarks from the D4RL dataset (Fu et al., 2020). Namely, we consider the Random, Medium, Mixed and Medium-Expert datasets for both Walker2d and HalfCheetah. In each setting, we compare AugWM against base MOPO on zero-shot performance, training entirely on the data provided, but not seeing the test environment until evaluation. The results are shown as a change v.s. MOPO, averaged over one dimension

in Fig. 5.11, and as a total return number averaged over both dimensions in Table 5.1. For additional implementation details (e.g., hyperparameters) see Appendix D.2. AugWM provides *statistically significant* improvements in zero-shot performance v.s. MOPO in many cases, achieving successful policies where MOPO fails.

Table 5.1: Each entry for HalfCheetah is the mean of 49 different dynamics, while for Walker2d it is over 25 dynamics. Results are mean $\pm$ 1std. $\star$ indicates $p < 0.05$ for Welch’s t-test for gain over MOPO (5 seeds).

Dataset Type	Environment	MOPO	AugWM (Ours)
Random	HalfCheetah	2303 $\pm$ 112	2818 $\pm$ 197 $\star$
Random	Walker2d	569 $\pm$ 103	706 $\pm$ 139
Mixed	HalfCheetah	3447 $\pm$ 218	3948 $\pm$ 122 $\star$
Mixed	Walker2d	946 $\pm$ 95	1317 $\pm$ 206 $\star$
Medium	HalfCheetah	2954 $\pm$ 89	2967 $\pm$ 106
Medium	Walker2d	1477 $\pm$ 337	1614 $\pm$ 440
Med-Expert	HalfCheetah	1590 $\pm$ 766	2885 $\pm$ 432 $\star$
Med-Expert	Walker2d	1062 $\pm$ 334	2521 $\pm$ 316 $\star$

By now we have provided significant evidence that AugWM can significantly improve performance for HalfCheetah and Walker2d with varied mass and damping. However, this is only a small subset of possible dynamics changes. We next consider several significantly harder settings. We test increased dimensionality, using the Ant environment from MOPO (Yu et al., 2020), and consider additional types of dynamics changes (e.g., Ant with crippled legs, HalfCheetah with changed limb sizes from Henderson et al. (2017)). We illustrate the impact of the crippled leg Ant environment on baseline agent performance, and the improvement provided by AugWM, in Fig. 5.12. We show the mean results over each of these factors of variation in Table 5.2, where again AugWM provides a non-trivial improvement over a strong baseline. For more details, see Appendix D.2.

Table 5.2: Mean performance for MOPO, AugWM with the default context, and AugWM with learned context (LM). Entries are mean zero-shot reward for all dynamics. Bold = highest (5 seeds).

Setting	MOPO	AugWM (Default)	AugWM (LM)
Ant: Mass/Damp	1634	1715	1804
Ant: One Crippled Leg	1370	1572	1680
Ant: Two Crippled Legs	700	697	795
HalfCheetah: Big	4891	5194	4968
HalfCheetah: Small	5151	5488	5263

Finally, we note that dynamics may change *during* an episode; consider a robot that suffers a motor fault, reducing the power delivered to its joints. Evidently the underlying dynamics have been altered, and being robust to such changes when only training from a single dataset of offline experience is challenging. To illustrate this, we perform a 1500 step rollout in the HalfCheetah environment, starting with offline dynamics (mass/damping = 1), before changing to mass = 0.75, damping = 0.5 after 500 steps; performance is shown in Fig. 5.9. Observe that after 500 steps, MOPO performance is dramatically reduced. This is because the agent continues to apply the same force and thus falls forward with lighter mass. For our AugWM agent, performance initially drops, then when the new context is learned we achieve *higher* performance than before, making use of the lighter torso.

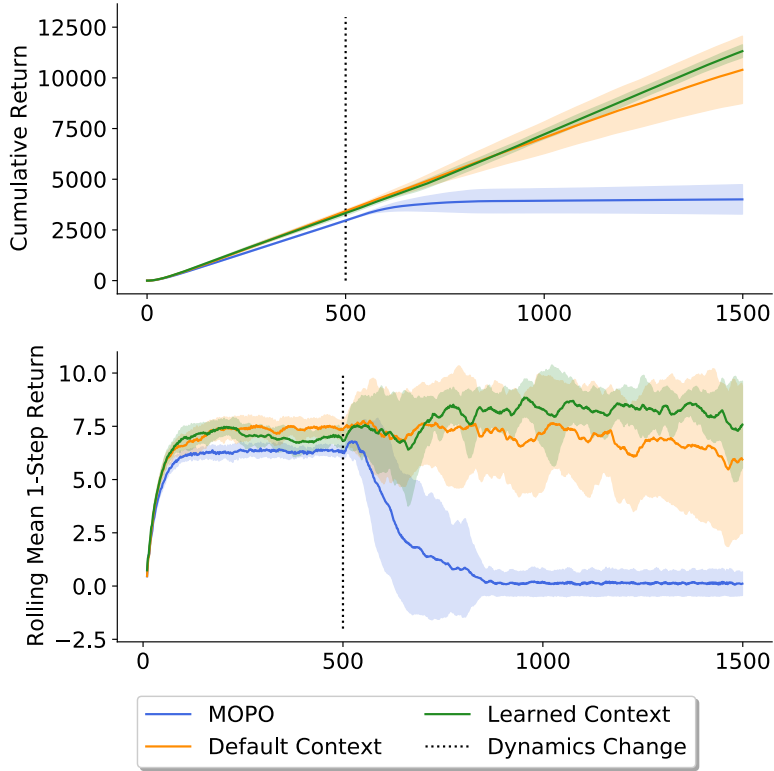


Figure 5.9: Performance under changing dynamics. Top = cumulative returns, Bottom = rolling average single step reward. Both averaged over twenty seeds, shaded area shows $\pm 1\text{std}$.

Discussion We believe that our experiments provide significant support to the claim that training with AugWM improves zero-shot dynamics generalization. In a

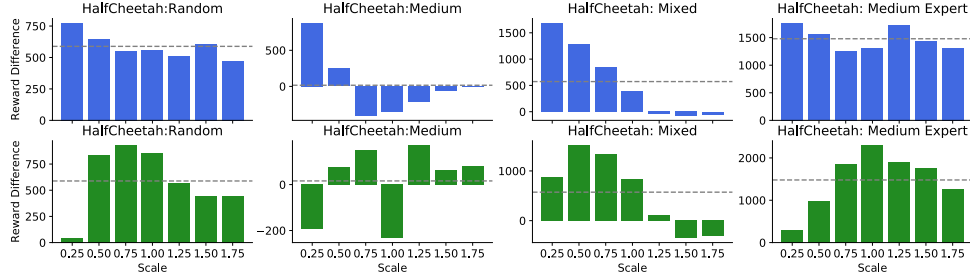


Figure 5.10: Mean improvement for Augmented World Models over MOPO for the HalfCheetah environment, averaged over five seeds. Top row (blue) = damping scale, bottom row (green) = mass scale. Dotted line is the mean, the same value for both damping and mass.

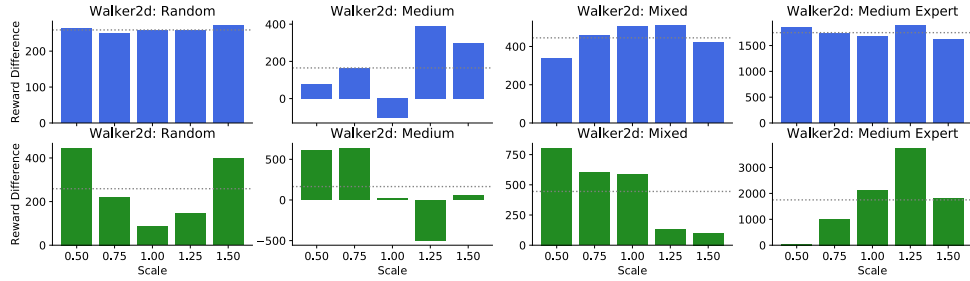
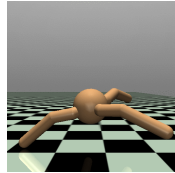
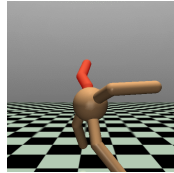


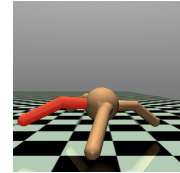
Figure 5.11: Mean improvement for Augmented World Models over MOPO for the Walker2d environment, averaged over five seeds. Top row (blue) = damping scale, bottom row (green) = mass scale. Dotted line is the mean, the same value for both damping and mass.



(a) MOPO on the regular Ant environment



(b) MOPO on the crippled leg Ant environment



(c) AugWM on the crippled leg Ant environment

Figure 5.12: Comparison between MOPO and AugWM on the crippled leg Ant environment. The crippled leg is highlighted in red. The MOPO agent flips over while the AugWM agent is able to adapt to changed dynamics and successfully run. MOPO on the regular Ant environment is provided as a reference.

broad set of commonly used datasets, and with a wide range of out-of-distribution dynamics, our algorithm learns good policies where a state-of-the-art baseline fails.³ This is due to a number of novel contributions: 1) using *dynamics augmentation* rather than *observation augmentation*; 2) training and testing with a context-based policy. Regarding limitations, we note that training inside the WM with context

³For videos see: <https://sites.google.com/view/augmentedworldmodels/>

generally takes longer to converge (Appendix D.2). Furthermore, in more nonlinear settings such as the HalfCheetah modified body part setting, we saw a reduced performance for the learned context. This could be because the dynamics changes are out of the distribution of DAS augmentations (violating Eqn. 5.1), or due to the difficulty of modeling the task with a linear model. We note that linear models have achieved success in planning (Gu et al., 2016) and meta learning (Peng et al., 2021), and are effective in our case due to their data efficiency, but can be replaced by more flexible models to deal with different augmentations. Indeed, given our work is the first of its kind, we believe significant improvements are possible, such as using more complex and problem-specific augmentations.

5.1.6 Conclusion and Future Work

In this paper we propose Augmented World Models (AugWM), which we show sufficiently simulates changes in dynamics such that agents can generalize in a zero-shot manner. We believe that we are the first to propose this problem setting, and our results show a significant improvement over existing state-of-the-art methods which ignore this problem.

A promising line of future work would be to meta-train a policy over AugWM such that it can quickly adapt to new dynamics in the few-shot setting. There is evidence that data augmentation can improve robustness in meta-learning (Rajendran et al., 2020), and could extend to strong performance in out-of-distribution tasks. We also wish to consider varying goals at test time, and other potential sources of non-stationarity which may impact policy performance (Igl et al., 2021). It may also be possible to extend AugWM to pixel-based tasks, which have received a great deal of recent attention (Hafner et al., 2019; 2020). We believe that our transition based augmentations will be applicable to a latent representation, as commonly used in state-of-the-art vision MBRL approaches. Thus, we think that extending our work to this setting, while a considerable feat of engineering, should not require significant methodological changes.

5.1.7 Errata

The shaded regions in Figure 5.4 and Figure 5.6 were not defined in the original manuscript. These are, in accordance with the rest of the figures, representative of ± 1 standard deviation across seeds.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Philip J. Ball, Cong Lu, Jack Parker-Holder, Stephen Roberts. Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment. <i>International Conference on Machine Learning 2022</i> .

Student Confirmation

Student Name:	Philip Ball		
Contribution to the Paper	Came up with main idea, wrote the offline model-based codebase from scratch, created the main techniques used, ran the analysis, majority of the writing.		
Signature 	Date	06/October/2023	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

5.2 Learning General World Models in a Handful of Reward-Free Deployments

Abstract Building generally capable agents is a grand challenge for deep reinforcement learning (RL). To approach this challenge practically, we outline two key desiderata: 1) to facilitate generalization, exploration should be task agnostic; 2) to facilitate scalability, exploration policies should collect large quantities of data without costly centralized retraining. Combining these two properties, we introduce the *reward-free deployment efficiency* setting, a new paradigm for RL research. We then present CASCADE, a novel approach for self-supervised exploration in this new setting. CASCADE seeks to learn a world model by collecting data with a population of agents, using an information theoretic objective inspired by Bayesian Active Learning. CASCADE achieves this by specifically maximizing the *diversity of trajectories* sampled by the population through a novel *cascading* objective. We provide theoretical intuition for CASCADE which we show in a tabular setting improves upon naïve approaches that do not account for population diversity. We then demonstrate that CASCADE collects diverse task-agnostic datasets and learns agents that generalize zero-shot to novel, unseen downstream tasks on Atari, MiniGrid, Crafter and the DM Control Suite. Code and videos are available at <https://ycxuyingchen.github.io/cascade/>

5.2.1 Introduction

Reinforcement learning (RL (Sutton & Barto, 1998)) has achieved a number of impressive feats over the past decade, with successes in games (Mnih et al., 2015; Berner et al., 2019; Silver et al., 2016), robotics (Kalashnikov et al., 2018; OpenAI et al., 2018), and the emergence of real world applications (Bellemare et al., 2020; Degraeve et al., 2022). Indeed, now that RL has successfully mastered a host of individual tasks, the community has begun to focus on the grand challenge of building *generally capable* agents (Reed et al., 2022; Team et al., 2021; Mendonca et al., 2021; Ahn et al., 2022).

In this work, we take steps towards building *generalist agents at scale*, where we outline two key desiderata. First, for agents to become generalists that can adapt to novel tasks, we eschew the notion of restricting agent learning to task-specific reward functions and focus on the *reward-free* problem setting instead (Pathak et al., 2017; Eysenbach et al., 2019), whereby agents must discover novel skills and behaviors without supervision.⁴ Consider the problem of learning to control robotic arms, where we may already have some expert offline data to learn from. In many cases this data will cover only a subset of the entire range of possible behaviors. Therefore, to learn additional general skills, it is imperative to collect additional novel and diverse data, and to do so without a pre-specified reward function.

Second, to ensure scalability, we should have access to a large fleet of robots that we can *deploy* to gather this data for a large number of timesteps (Levine et al., 2018), without costly and lengthy centralized retraining during this crucial phase (Matsushima et al., 2021). This has recently been referred to as *deployment efficiency*, falling between the typical online/offline RL dichotomy. Limiting deployments not only reduces the overhead in retraining exploration policies, but also limits the potential costs and risks present when deploying new policies (Tan, 1993; Kober et al., 2013), an important consideration in many real world settings, such as robotics (Kalashnikov et al., 2018), education (Mandel et al., 2014) and healthcare (Fatemi et al., 2021).

Combining these two desiderata, we introduce the *reward-free deployment efficiency* setting, a new paradigm for deep RL research. To tackle this new problem we train a population of exploration policies to collect large quantities of useful data via *world models* (Chua et al., 2018; Ha & Schmidhuber, 2018; Hafner et al., 2019). World models allow agents to plan and/or train policies without interacting with the true environment. They have already been shown to be highly effective for deployment efficiency (Matsushima et al., 2021), offline RL (Yu et al., 2020; Kidambi et al., 2020; Argenson & Dulac-Arnold, 2021; Rafailov et al., 2021) and self-supervised exploration (Shyam et al., 2019; Sekar et al., 2020).

⁴Indeed, designing reward functions to learn behaviors can have unintended consequences (Ng et al., 1999; Amodei & Clark, 2016).

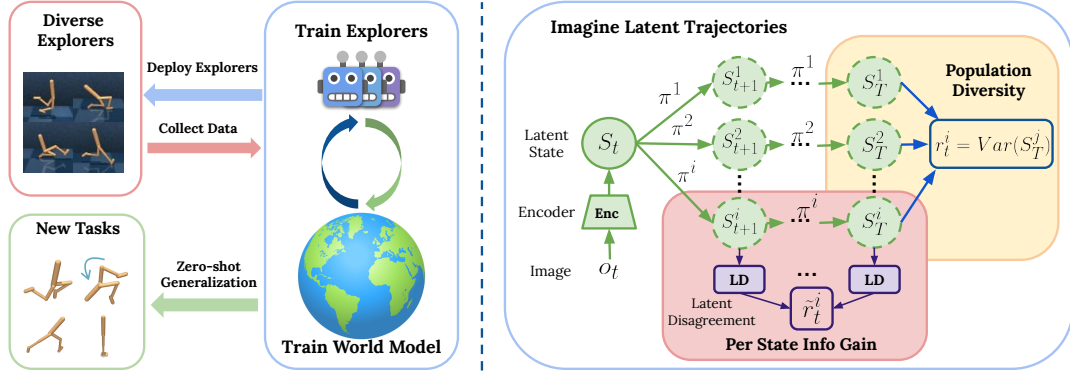


Figure 5.13: Overview. Left: CASCADE trains a population of diverse explorers and uses them to collect large batches of reward-free trajectories for learning a general world model that facilitates zero-generalization to novel tasks. Right: To train B exploration agents in parallel, at each training step t , CASCADE first infers a latent state s_t from image observation o_t . It then rolls out latent trajectories $\tau^1, \dots, \tau^B$ in imagination using the current exploration policies $\pi^1, \dots, \pi^B$. The training objective for each policy π^i is to optimize 1) the population diversity estimated by the disagreement of the final states of imagined trajectories sampled from policies $\pi^1, \dots, \pi^i$; 2) the expected per state information gain over all future timesteps $t + 1, \dots, T$, computed as the disagreement of an ensemble of dynamics models.

Furthermore, world models offer the potential for increasing agent generalization capabilities (Anand et al., 2022; Hafner, 2022; Mendonca et al., 2021; Byravan et al., 2019; Argenson & Dulac-Arnold, 2021; Clavera et al., 2019; Ball et al., 2021; Lee et al., 2020), one of the frontiers of RL research (Packer et al., 2018; Kirk et al., 2021). However, since existing self-supervised methods for learning world models are designed to collect only a few transitions with a single exploration policy, they likely produce a *homogenous dataset* when deployed at scale, which does not optimally improve the model.

Instead, drawing analogies from Bayesian Active Learning (Houlsby et al., 2011; Kirsch et al., 2019), we introduce a new information theoretic objective that maximizes the information gain from an *entire dataset* collected by a *population* of exploration agents (see Figure 5.14). We call our method *Coordinated Active Sample Collection via Diverse Explorers* or CASCADE (Figure 5.13).

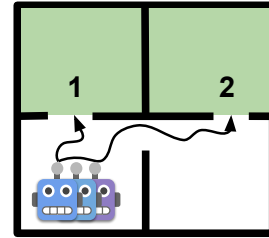


Figure 5.14: Motivation for CASCADE: Green areas represent high expected information gain. If we train a population of agents independently, at deployment time they will all follow the trajectory to #1, producing a homogenous dataset. However, if we consider the diversity of the data then we will produce agents that reach both #1 and #2.

We provide theoretical justification for CASCADE, which emphasizes the importance of collecting data with diverse agents. In addition, we provide a rigorous empirical evaluation across four challenging domains that shows CASCADE can discover a rich dataset from a handful of deployments. We see that CASCADE produces general exploration strategies that are equally adept at both “deep” exploration problems and diverse behavior discovery. This makes it possible to train agents capable of zero-shot transfer when rewards are provided at test time in a variety of different settings.

To summarize, our contributions are as follows: 1) We introduce a novel problem setting, *Reward-Free Deployment Efficiency*, designed to train generalist agents in a scalable fashion; 2) We propose CASCADE, a theoretically motivated model-based RL agent designed to gather diverse, highly informative data, inspired by Bayesian Active Learning; 3) We provide analysis that shows CASCADE theoretically improves sample efficiency over other naïve methods that do not ensure sample diversity, and demonstrate that CASCADE is capable of *improved zero-shot transfer* in four distinct settings, ranging from procedurally generated worlds to continuous control from pixels.

5.2.2 Problem Statement

Reinforcement learning (RL) considers training an agent to solve a Markov Decision Process (MDP), represented as a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, P, R, \rho, \gamma\}$, where $s \in \mathcal{S}$ and $a \in \mathcal{A}$ are the set of states and actions respectively, $P(s'|s, a)$ is a probability distribution over next states given a previous state and action, $R(s, a, s') \rightarrow r$ is a reward function mapping a transition to a scalar reward, ρ is an initial state distribution and γ is a discount factor. A policy π acting in the environment produces a trajectory $\tau = \{s_1, a_1, \dots, s_H, a_H\}$ for an episode with horizon H . Since actions in the trajectory are sampled from a policy, we can then define the RL problem as finding a policy π that maximizes expected returns in the environment, i.e. $\pi^* = \arg \max_{\pi} \mathbb{E}_{\tau \sim \pi}[R(\tau)]$.

We seek to learn policies that can transfer to *any* MDP within a family of MDPs. This can be formalized as a *Contextual* MDP (Kirk et al., 2021), where observations,

dynamics and rewards can vary given a context. In this paper we consider settings where only the reward varies, thus, if the test-time context is unknown at training time we must collect data that *sufficiently covers the space of possible reward functions*. Finally, to facilitate scalability, we operate in the deployment efficient paradigm (Matsushima et al., 2021), whereby policy learning and exploration are completely separate, and during a given *deployment*, we gather a large quantity of data without further policy retraining (c.f. online approaches like DER (van Hasselt et al., 2019), which take multiple gradient steps *per* exploration timestep in the real environment). Taken together, we consider the *reward-free deployment efficiency* problem. This differs from previous work as follows: 1) unlike previous deployment efficiency work, our exploration is task agnostic; 2) unlike previous reward-free RL work, we cannot update our exploration policy π_{EXP} during deployment. Thus, the focus of our work is on how to train π_{EXP} offline such that it gathers heterogeneous and informative data which facilitate zero-shot transfer to unknown tasks.

In this paper we make use of model-based RL (MBRL), where the goal is to learn a model of the environment (or *world model* (Schmidhuber, 1990)) and then use it to subsequently train policies to solve downstream tasks. To do this, the world model needs to approximate both P and R . Typically, the model will be a neural network, parameterized by ψ , hence we denote the approximate dynamics and reward functions as P_ψ and R_ψ , which produces a new “imaginary” MDP, $\mathcal{M}_\psi = (\mathcal{S}, \mathcal{A}, P_\psi, R_\psi, \rho)$. We focus on Dyna-style MBRL (Sutton, 1991), whereby we train a policy (π_θ parameterized by θ) with model-free RL solely using “imagined” transitions inside $\mathcal{M}_\psi$. Furthermore, we can train the policy on a single GPU with parallelized rollouts since the simulator is a neural network (Kurutach et al., 2018). The general form of all methods in this paper is shown in Algorithm 6, with the key difference being step 5: We aim to update π_{EXP} in the new imaginary MDP $\mathcal{M}_\psi$ such that it continues to collect a large, diverse quantity of reward-free data. Note that π_{EXP} need not be a single policy, but could also refer to a collection of policies that we can deploy (either in parallel or in series), such that $\pi \in \pi_{\text{EXP}}$.

Algorithm 6 Reward-Free Deployment Efficiency via World Models

```

1: Input: Initial exploration policy  $\pi_{\text{EXP}}$ 
2: for each deployment do
3:   Deploy  $\pi_{\text{EXP}}$  to collect a large quantity of reward-free data.
4:   Train world model on all existing data.
5:   Update  $\pi_{\text{EXP}}$  in new imaginary MDP  $\mathcal{M}_\psi$ .
6: end for

```

We focus on learning world models from high dimensional sensory inputs such as pixels (Ha & Schmidhuber, 2018; Oh et al., 2015; Kaiser et al., 2020), where at each timestep we are given access to an observation o_t rather than a state s_t . A series of recent works have shown tremendous success by mapping the observation to a compact latent state z_t (Hafner et al., 2019; 2020; 2021). In this paper we will make use of the model from DreamerV2 (Hafner et al., 2021), which has been shown to produce highly effective policies in a variety of high dimensional environments. The primary component of DreamerV2 is a Recurrent State Space Model (RSSM) that uses a learned latent state to predict the image reconstruction, reward r_t and discount factor γ_t . Aside from the reward head, all components of the model are trained jointly, in similar fashion to variational encoders (VAEs, (Kingma & Welling, 2014; Rezende et al., 2014)). For zero-shot evaluation, we follow Sekar et al. (2020) and only train the reward head at test time when provided with labels for our pre-collected data, which is then used to train a behavior policy offline. Thus, it is critical that our dataset is sufficiently diverse to enable learning novel, unseen behaviors.

5.2.3 Coordinated Active Sample Collection

The aim of this work is to train a population of B exploration policies $\{\pi_{\text{EXP}}^{(i)}\}_{i=1}^B$ such that they collectively acquire data which maximally improves the accuracy of a *world model*. To achieve this, we take inspiration from the information theoretic approach in Plan2Explore (Sekar et al., 2020), but crucially focus on maximizing information gain over *entire trajectories* rather than per state-action, and hence drop the conditional dependence on state and action (see App. E.3.1 for why

this distinction is important):

$$\pi_{\text{EXP}} = \arg \max_{\pi} \mathcal{I}(d_{\mathcal{M}_{\psi}}^{\pi}; \mathcal{M}_{\psi}) = \mathcal{H}(d_{\mathcal{M}_{\psi}}^{\pi}) - \mathcal{H}(d_{\mathcal{M}_{\psi}}^{\pi} | \mathcal{M}_{\psi}) \quad (5.5)$$

where $d_{\mathcal{M}_{\psi}}^{\pi}$ is the distribution of states visited by the policy π in the imaginary MDP $\mathcal{M}_{\psi}$. This objective produces π_{EXP} , a policy whose visitation distribution has a high entropy when computed over model samples, but has low entropy for each individual MDP model (i.e., high epistemic/reducible uncertainty). We think of each model $\mathcal{M}_{\psi}$ as sampled from a posterior distribution over models given the data. A good exploration policy has low entropy on individual models but large entropy across models, i.e. it is intent in visiting regions of the space where there is large uncertainty about the model transitions. To make this objective more general, we represent the trajectory data collected by a policy with a “summary” embedding space (Pacchiano et al., 2020; Moskovitz et al., 2021a). Let $\Phi : \Gamma \rightarrow \Omega$ be a summary function mapping trajectories into this embedding space. $\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]$ denotes the embedding distribution generated by policy π in imaginary MDP $\mathcal{M}_{\psi}$. We can now write the objective from Eq. 5.5 as follows:

$$\pi_{\text{EXP}} = \arg \max_{\pi} \mathcal{I}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]; \mathcal{M}_{\psi}) = \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]) - \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi}) \quad (5.6)$$

This more general framework allows us to consider multiple representations for trajectories, in a similar fashion to behavioral characterizations in Quality Diversity algorithms (Pugh et al., 2016). For the rest of this discussion, we will use the final state embedding as our summary representation, whereby $\Phi(\tau) = h_H$, since in the case of the RSSM, the final latent is a compact representation of the entire trajectory collected by the policy, analogous to the final hidden state in an RNN (Rumelhart et al., 1986; Sutskever et al., 2014).

A Cascading Objective with Diverse Explorers

We now consider a population-based version of Equation. 5.6, using B agents:

$$\{\pi_{\text{EXP}}^{(i)}\}_{i=1}^B = \arg \max_{\pi^B \in \Pi^B} \mathcal{I}\left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^{\Phi}[\mathcal{M}_{\psi}]; \mathcal{M}_{\psi}\right) = \mathcal{H}\left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^{\Phi}[\mathcal{M}_{\psi}]\right) - \mathcal{H}\left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi}\right) \quad (5.7)$$

where $\boldsymbol{\pi}^B = \pi^{(1)}, \dots, \pi^{(B)}$ and $\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi]$ is the product measure of the policies' embedding distributions in $\mathcal{M}_\psi$. By definition, the conditional entropy factorizes as:

$$\mathcal{H}\left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi\right) = \sum_{i=1}^B \mathcal{H}\left(\mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi\right). \quad (5.8)$$

It is now possible to show that maximum information gain is achieved with a *diverse* set of agents:

Lemma 1. *When all models $\mathcal{M}_\psi$ in the support of the model posterior are deterministic and tabular, and the space of policies Π consists only of deterministic policies, there always exists a solution $\{\pi_{\text{EXP}}^{(i)}\}_{i=1}^B$ satisfying $\pi_{\text{EXP}}^{(i)} \neq \pi_{\text{EXP}}^{(j)} \forall i \neq j$. Moreover, there exists a family of tabular MDP models, such that the maximum cannot be achieved by setting $\pi_{\text{EXP}}^{(i)} = \pi$ for a fixed π .*

The proof of Lemma 1 is in Appendix E.3.1. Since the mutual information objective 5.7 is submodular, a greedy algorithm yields a $(1 - \frac{1}{e})$ approximation of the optimum (where e is Euler's number) (Nemhauser et al., 1978). Leveraging this insight, let us assume that we already have a set of policies $\pi^{(1)}, \dots, \pi^{(i-1)}$; we then select the next policy $\pi^{(i)}$ based on the following greedy objective:

$$\pi^{(i)} = \arg \max_{\tilde{\pi}^{(i)} \in \Pi} \mathcal{I}\left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi]; \mathcal{M}_\psi \middle| \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1\right) \quad (5.9)$$

$$= \mathcal{H}\left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi] \middle| \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1\right) - \mathcal{H}\left(\prod_{j=1}^i \mathbb{P}_{\pi^{(j)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi, \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1\right) \quad (5.10)$$

Which can be factorized in similar fashion to Equation 5.8 (See Appendix E.3).

A Tractable Objective for Deep RL

Inspired by Sekar et al. (2020), we make a couple of approximations to derive a tractable objective for $\{\pi_{\text{EXP}}\}_{i=1}^B$ in the deep RL setting. First, we assume that the final state embedding distributions are Gaussian with means that depend on the policies and sampled worlds, and variances that depend on the worlds, i.e. $\mathbb{P}_\pi^\Phi[\mathcal{M}_\psi = w] = \mathcal{N}(\mu(w, \pi), \Sigma(w))$. In this case, $\mathcal{H}(\mathbb{P}_\pi^\Phi[\mathcal{M}_\psi] | \mathcal{M}_\psi = w) = \rho(w)$, and Eq. 5.9 reduces to solving $\pi^{(i)} = \arg \max_{\tilde{\pi}^{(i)} \in \Pi} \mathcal{H}\left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi] \middle| \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1\right)$

for a policy that maximizes the resulting joint entropy of the embedding distribution when added to the policy population. This produces the following surrogate objective, maximizing a quadratic *cascading* disagreement:

$$\text{PopDiv}^\Phi(\pi|\pi^{(1)}, \dots, \pi^{(i-1)}) = \mathbb{E}_{\tau \sim \mathbb{P}^\pi[\mathcal{M}_\psi]} \left[\frac{1}{|\mathcal{D}^{(i-1)}| - 1} \sum_{\tilde{\tau} \in \mathcal{D}^{(i-1)}} \|\Phi(\tau) - \Phi(\tilde{\tau})\|^2 \right]$$

where $\mathcal{D}^{(i-1)}$ is a dataset of imagined trajectories sampled from policies $\pi^{(1)}, \dots, \pi^{(i-1)}$ in the model, and **PopDiv** is short for Population Diversity. Finally, following [Sekar et al. \(2020\)](#); [Ball et al. \(2020\)](#), we also add a per state information gain component to each policy’s reward to encourage a richer landscape for data acquisition: $\text{InfoGain}(\pi) = \mathbb{E}_{\tau \sim \mathbb{P}^\pi[\mathcal{M}_\psi]} \left[\sum_{(s,a) \in \tau} \sigma(s, a) \right]$ where $\sigma(\cdot, \cdot)$ is the variance across the ensemble latent state predictions (for details see App. E.2).

Taken together, these objectives form our approach, which we call *Coordinated Active Sample Collection via Diverse Explorers* or CASCADE. CASCADE trains agents to optimize: 1) a diversity term (**PopDiv**) that takes into account the behaviors of the other agents in the population; 2) an information gain term (**InfoGain**) that encourages an individual agent to sample states that maximally improve the model:

$$\pi^{(i)} = \arg \max_{\pi \in \Pi} \left[\lambda \text{PopDiv}^\Phi(\pi|\{\pi_{\text{EXP}}^{(j)}\}_{j=1}^{i-1}) + (1 - \lambda) \text{InfoGain}(\pi) \right] \quad (5.11)$$

where λ is a weighting hyperparameter that trades off whether we favor individual model information gain or population diversity. Finally, we train the B agents in *parallel* using (policy) gradient descent over θ , which makes it possible to achieve the same wall clock time as training a single agent ([Cui et al., 2021](#)).

5.2.4 Theoretical Motivation

We now seek to provide a tabular analogue to CASCADE which provides a theoretical grounding for our approach. In App. E.4 we outline the pseudo-code of CASCADE-TS, a greedy Thompson Sampling algorithm ([Agrawal & Goyal, 2012](#)) that produces the i -th exploration policy in a tabular environment using “imaginary” data gathered by running policies $\pi^{(1)}, \dots, \pi^{(i-1)}$ in the *model*. We can then show the following:

Lemma 2. *For the class of Binary Tree MDPs, the CASCADE-TS algorithm satisfies,*

$$T(\epsilon, \text{Sequential}) \leq T(\epsilon, \text{CASCADE-TS}) \leq T(\epsilon, \text{SinglePolicyBatch})$$

where $T(\epsilon, \cdot)$ are the expected number of rounds of deploying a population of B policies necessary to learn the true model up to ϵ accuracy; SinglePolicyBatch plays a fixed policy B times in each round; Sequential does not have a population, and instead interleaves updates and executions of a single policy B times within each round.

The proof is in App. E.4. Indeed, we see that CASCADE-TS achieves provable efficiency gains over a naïve sampling approach that does not ensure diversity in its deployed agents. This provable gain is achieved by discouraging the i -th policy away from imaginary state-action pairs sampled by the previous $i - 1$ policies by using imaginary counts.

Now returning to CASCADE, we can see the importance of leveraging the imaginary data gathered in the model by the previous $i - 1$ policies when training the i -th exploration policy. Concretely, encouraging policy $\pi^{(i)}$ to induce high disagreement with the embeddings produced by $\{\pi^{(1)}, \dots, \pi^{(i-1)}\}$ (i.e. the PopDiv term) is analogous to the imaginary count bonus term of CASCADE-TS in Line 10 of Alg. 8, which avoids redundant data collection during deployment.

5.2.5 Experiments

In our experiments we test whether CASCADE facilitates the learning of generalist agents by evaluating their zero-shot transfer to unseen tasks, given a limited number of reward-free deployments. In all experiments, agents cannot train online in the environment, and instead execute a fixed exploration policy to gather new data during each deployment. We test against three baselines. The first is random exploration, which is used in the majority of RL publications. Our next baseline is Plan2Explore (Sekar et al., 2020) (P2E), which trains a single exploration policy that optimizes for expected information gain inside a world model. We then include a population-based version of P2E, which we call *Population Plan2Explore*

(PP2E). PP2E trains a set of randomly initialized agents that independently seek to maximize expected information gain. All methods make use of a DreamerV2 world model (Hafner et al., 2021) and use the same hyperparameters for model and agent training (more details in App. E.2).

We test all methods in two separate settings: Firstly *Exploring Worlds* (Sec 5.2.6), where we seek to collect data in challenging exploration environments. We test how well the explorers cover the state space and discover sparse rewards (also known as “deep exploration” (Osband et al., 2019; O’Donoghue et al., 2020)). Secondly, we consider *Exploring Behaviors* (Sec 5.2.6) where we consider collecting data in continuous control environments. In both settings we test zero-shot transfer as follows: 1) we provide reward labels to the model; 2) we train a separate reward head; 3) we train a task specific behavior policy and test it in the environment. This tests whether our model is general enough to facilitate learning downstream policies for previously unknown tasks (Sekar et al., 2020; Lambert et al., 2022; Yarats et al., 2022a).

5.2.6 Exploring Worlds

In many cases we may not know a priori which states in an environment are highly rewarding. Thus, if we wish to use our world model to subsequently train agents to solve tasks, it is crucial that we accurately model as much of state space as possible. Here we consider exploring worlds in three different environments: MiniGrid (Chevalier-Boisvert et al., 2018), Atari (Bellemare et al., 2012) and Crafter (Hafner, 2022). In all three settings, we start with a *random* deployment before beginning reward-free exploration.

We begin with MiniGrid, a set of sparse-reward, partially observable navigation environments commonly used to test state-of-the-art exploration methods (Raileanu & Rocktäschel, 2020; Campero et al., 2021; Flet-Berliac et al., 2021; Zhang et al., 2021c). MiniGrid environments are procedurally generated, which provides an additional generalization challenge (Justesen et al., 2018; Risi & Togelius, 2020). We consider the canonical `FourRooms` and the more challenging `MultiRoom`. In

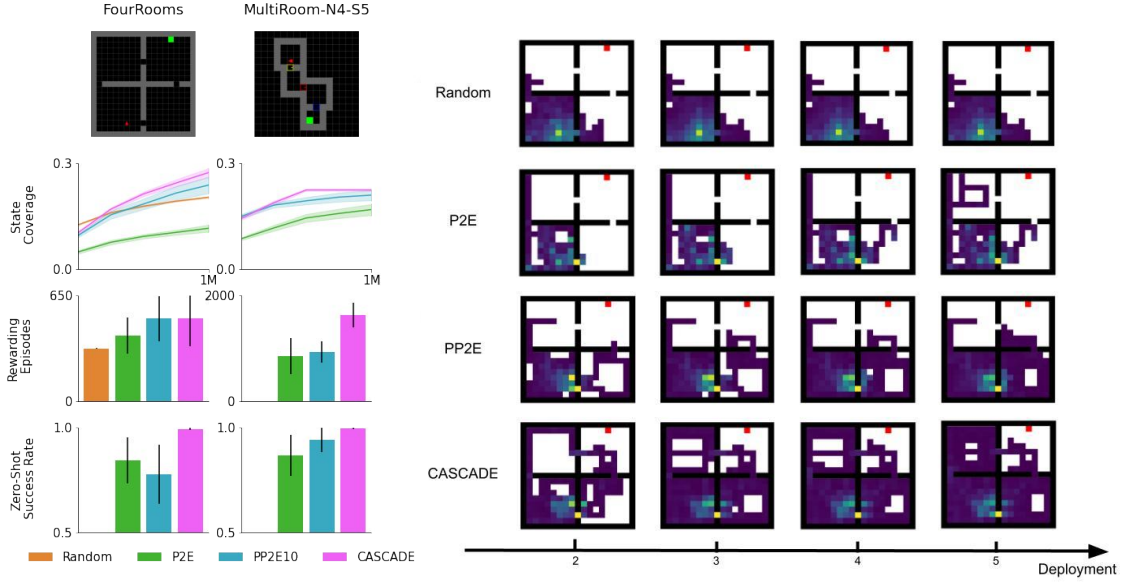


Figure 5.15: MiniGrid Results. Left: Performance statistics for both **FourRooms** and **MultiRoom**. For reward-free exploration we show both “state coverage”, which corresponds to the the percentage of states visited on a fixed set of levels, and “rewarding episodes” which is the count of total solved episodes found after 1M steps. Finally for zero-shot transfer we show the success rate of a task-specific behavior policy trained with labels provided only at test time. All plots show the mean of ten seeds with SEM shaded. (Random achieves 0 zero-shot success rate). Right: we show the state coverage on a fixed level in **FourRooms**, using the exploration policy π_{EXP} for each deployment (Test time goal state indicated by the red dot).

each case we are limited to 5 deployments of 200k transitions, for a total of 1M environment interactions.⁵ In order to gauge the effectiveness of the reward-free exploration, we use two metrics: 1) state coverage, where we evaluate the percentage of the states visited by the exploration policies after each deployment on the same set of held out test levels; and 2) rewarding episodes, where we show the number of rewarding episodes collected during the training process. Note that these two objectives are not useful in isolation, as solved levels may only include a narrow set of observations, while state coverage alone could be maximized without performing deep, goal-seeking exploration in the environment. However, in combination they provide a proxy for exploration.

We show the results in Figure 5.15, where we see the CASCADE exploration policies cover the state space far more effectively than all baselines. In **FourRooms**,

⁵Zero-shot success rates of P2E, PP2E and CASCADE increase with more training steps, and CASCADE achieves a near 100% success rate at 1M steps in both **FourRooms** and **MultiRoom**.

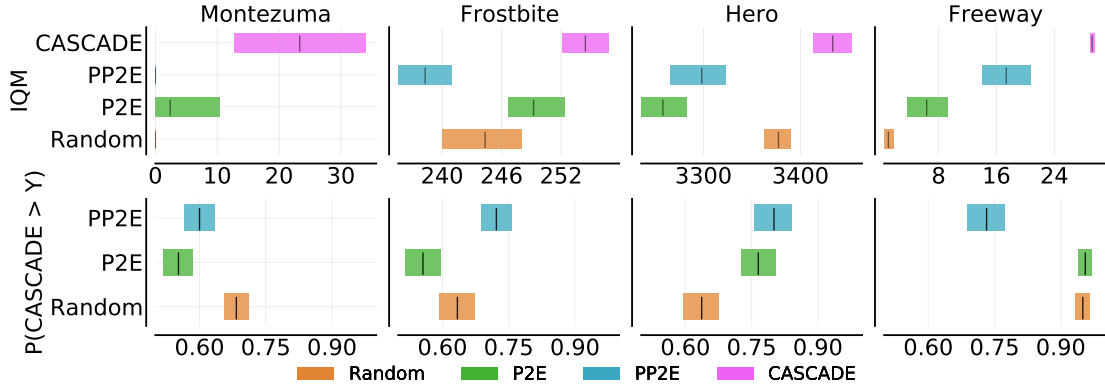


Figure 5.16: Atari Zero-Shot Transfer: Plots show RLiabile metric performance aggregated across all seeds. Note that the ‘Probability of Improvement’ lower CIs all exceed 0.5 for Atari, indicating CASCADE provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (Agarwal et al., 2021). We show five seeds of zero-shot test performance.

both CASCADE and PP2E find the goal an equal number of times, while CASCADE finds almost double the number of rewarding episodes in the **MultiRoom** environment. We also test zero-shot performance, where CASCADE achieves almost 100% success rate. Interestingly, in both settings the Random policy does cover a large proportion of the state space, but finds fewer rewarding episodes and subsequently does not have a sufficient quantity to train a policy to solve the task (and accordingly achieves 0% success rate).

Next we consider four Atari games: **Montezuma’s Revenge**, **Frostbite**, **Hero** and **Freeway**, often used as barometers for exploration capabilities (Ecoffet et al., 2021). We use 15 reward-free deployments, each consisting of 200k steps, and use the final model to train a behavior policy with task reward. We test these agents in a zero-shot fashion, with results shown in the first three columns of Figure 5.16. We see that CASCADE gets statistically significantly stronger zero-shot returns compared to the baselines. Furthermore, the baseline methods display no clear trend, with random exploration sometimes offering the strongest performance. On the other hand, we see that CASCADE performs consistently well, providing further evidence of it being a strong general exploration strategy. See App. E.1.2 for more detailed analysis.

Finally, we consider the **Crafter** environment (Hafner, 2022), a procedurally generated grid world based on the game of Minecraft. This more complex setting requires the agent to master a variety of compositional skills to fully explore all

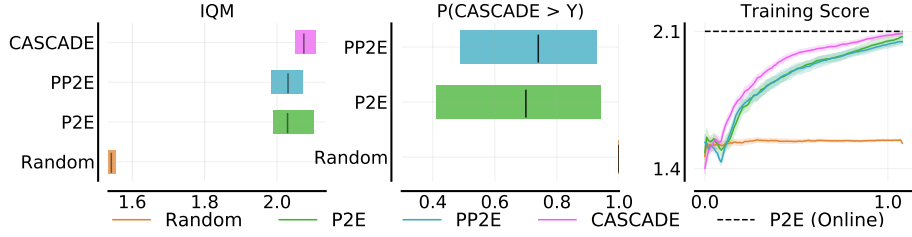


Figure 5.17: Crafter Score: Plots show final RL-liable metric performance and training curve over 1M steps aggregated across all seeds. We show ten seeds of Crafter Score, the geomean of skills discovered in a purely reward-free fashion. Again, see that the ‘Probability of Improvement’ lower CIs all exceed 0.5, indicating CASCADe provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (Agarwal et al., 2021).

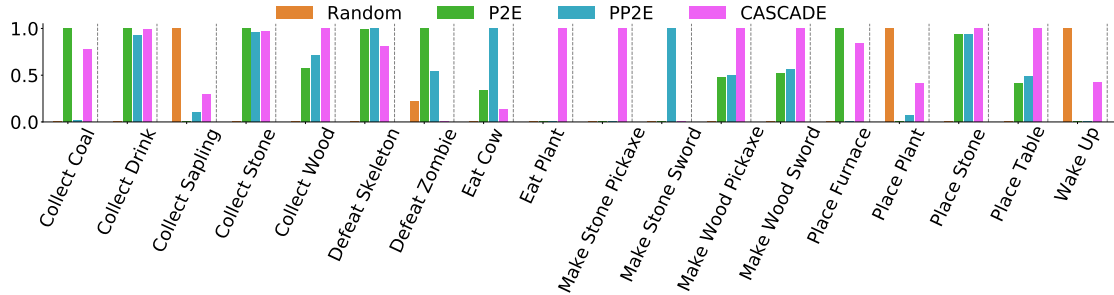


Figure 5.18: Crafter Skills: Plots show normalized success rate per skill. Note that CASCADe not only achieves the highest average success rate but also unlocks the most (16 out of 22) unique skills among all baselines. Skills that are not unlocked by any of the baselines are excluded from the plot.

possible behaviors. Online P2E represents the state of the art reward-free agent, achieving a “Crafter Score” of 2.1. Making the problem more challenging, we deploy only 20 distinct exploration policies, each collecting 50k timesteps, which leads to weaker performance for P2E vs. the reported result in Hafner et al. (2021). However, as we see in Figure 5.17, adding a population of diverse explorers recovers the majority of this performance. We find this result encouraging as the behaviors required to achieve a higher Crafter score are non-trivial. Moreover, as shown in Figure 5.18, CASCADe achieves the highest average success rate and unlocks the most number of unique Crafter skills among all baselines, including Making a Stone Pickaxe. We believe this opens the door to further gains with more active tuning of the behavioral representation used in CASCADe.

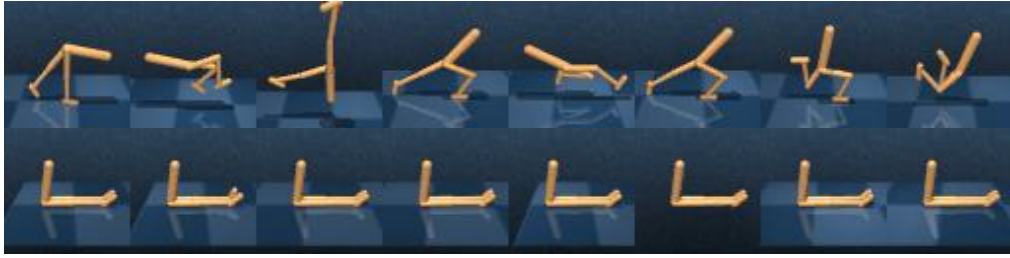


Figure 5.19: DMC Qualitative Results: Each row visualizes behaviors of a population of explorers trained after 15 total deployments by the same world model, deployed at the same time in environments with the same initial state. We can see that CASCADE explorers (top) collect data with a diversity of behaviors while PP2E agents (bottom) only show less interesting homogenous behaviors.

Exploring Behaviors

Next we wish to test is whether our agents sufficiently explore a range of behaviors in a continuous control setting, using the **walker** environment from the DM Control Suite (Tunyasuvunakool et al., 2020) (DMC). In these experiments, we operate from pixels and assume access to an initial dataset of 200k transitions. Then, we conduct 14 further reward-free deployments, each collecting 200k transitions. We consider the possibility of improving generality from arbitrary offline data, using common datasets from the offline RL literature: **random**, **medium** and **expert**. Using random data represents learning from scratch. In Figure 5.19 we show the behaviors deployed by CASCADE and PP2E in a single deployment. The CASCADE explorers all display diverse, useful behaviors, while PP2E agents are homogenous.

We then test the generality of the world model after each deployment by training a separate agent from scratch for each of the four individual tasks: **stand**, **walk**, **run**, **flip** proposed in URLB (Laskin et al., 2021). This demonstrates whether the world model is capable of producing an imaginary MDP that facilitates learning specific behaviors. To compare the performance of CASCADE and the baselines, in Figure 5.20 we show the aggregated statistics from the RLiable library (Agarwal et al., 2021) for a given number of deployments. To do this, we combine the results across all datasets (**random**, **medium**, **expert**) and all downstream tasks (**stand**, **walk**, **run**, **flip**) and compute the following: 1) the Inter-Quartile Mean (IQM) of the normalized scores (the robust statistic recommended in Agarwal et al. (2021));

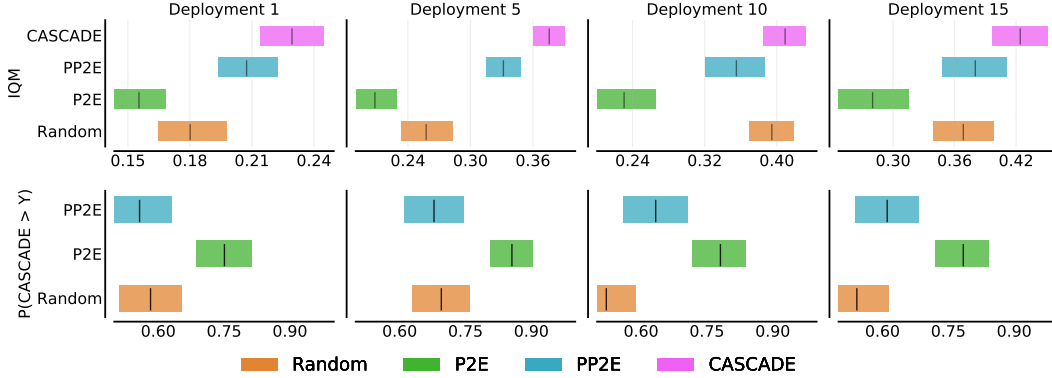


Figure 5.20: DMC Zero-Shot Aggregate Results: Plots show RLiablie metric performance aggregated across all three initial datasets (`random`, `medium`, `expert`) for a total of 30 seeds. Note that the ‘Probability of Improvement’ lower CIs all exceed 0.5, indicating CASCADE provides a statistically significant improvement (under the Mann-Whitney U test) over all baseline methods (Agarwal et al., 2021), even after a single deployment.

2) the Probability of Improvement, the likelihood of one method outperforming another on a new, unseen task. As we see, CASCADE shows clear and consistent gains over the next best baseline (PP2E) throughout 15 deployments, showing statistical significance under the Mann-Whitney U test (Mann & Whitney, 1947). Note that here the single agent P2E performs poorly, likely due to only collecting data with a single mode of exploratory behavior. See Figure E.1 in the Appendix for a full breakdown of the performance.

Discussion and Limitations

We have shown that CASCADE is the most effective method in training generally capable agents across a wide range of environments. Indeed, CASCADE provides the best of both worlds: deep exploration by seeking the frontier of the current knowledge (via InfoGain), while covering the space of possible behaviors (via PopDiv). Each of our baselines lacked in one of these two key properties. Random exploration fared poorly when exploring worlds, which requires deep exploration. However, it can be a highly effective method for exploring behaviors. This is likely why boosting entropy in the action space is so effective for robotics tasks (Haarnoja et al., 2018b), but less so in deep exploration tasks (O’Donoghue et al., 2020). Furthermore, the information theoretic baselines were effective in Exploring Worlds,

but the disappointing performance for P2E in DMC is likely caused by a lack of data diversity, while the PP2E populations were not sufficiently heterogeneous to match the coverage of CASCADE.

In terms of limitations, the main weakness in current methods is the absolute level of performance. Despite outperforming baselines in DMC, the policies learned by CASCADE are far from optimal. This is likely due to the challenge of the setting (reward-free deployment efficiency) and we believe that future work in the community will be crucial to further improve these methods. Finally, CASCADE does require training additional agents that may provide additional computational expense. In this setting we assume the time between deployments is arbitrary, but this may not be the case in all settings and future work could consider ways to make this process more efficient and scalable.

5.2.7 Related Work

At the core of our paper is the notion of training generalist agents, which have recently been of increased interest in RL (Reed et al., 2022; Mendonca et al., 2021; Team et al., 2021; Touati & Ollivier, 2021). In particular, we focus on the *reward-free* paradigm, since in many cases it may not be possible to know a priori all possible tasks we may wish our agent to solve (Jin et al., 2020). With this in mind, there has been a surge of interest in methods that can learn in a self-supervised fashion, to build representations that lead to fast (or even zero-shot) adaptation to future tasks. These methods range from competence-based (i.e. learning “skills”) (Achiam et al., 2018; Gregor et al., 2017; Eysenbach et al., 2019; Lee et al., 2019; Liu & Abbeel, 2021a; Strouse et al., 2022) to data-based that aim to maximize data diversity (Liu & Abbeel, 2021b; Yarats et al., 2021).

In this work we take inspiration from *knowledge-based* approaches, which typically seek to learn a world model (Ball et al., 2020; Pathak et al., 2017; 2019; Filos et al., 2021; Svidchenko & Shpilman, 2021; Burda et al., 2019; Sharma et al., 2020). In particular we build on Plan2Explore (Sekar et al., 2020) which trains an exploration agent by maximizing an objective resembling information gain inside the world

model. Plan2Explore was the first work that showed it is possible to transfer a model learned without rewards in a zero-shot fashion. However, it differed from our work in that Plan2Explore solely considers the online RL setting, deploying a new exploration policy every timestep. Other recent work examined the effectiveness of reward-free RL for collecting datasets for subsequent offline RL (Yarats et al., 2022a; Lambert et al., 2022). Once again however, this work trains the exploration agent online. By contrast, we only deploy a handful of exploration agents and collect a larger quantity of data with each one.

Our setting resembles a reward-free version of the *deployment efficiency* paradigm, first introduced in Matsushima et al. (2021). By contrast, their method focuses on policy optimization in the *supervised* RL paradigm, whereas we focus on exploration and zero-shot generalization in the unsupervised RL setting. Follow up work has remained focused on supervised policy optimization (Su et al., 2022), whereas we are focused on training generalist agents in the reward-free paradigm. Recent work has also shown the importance of frequent policy updates in ensuring diverse exploration (Schaul et al., 2022), further supporting the need for a diverse population of explorers when operating in the deployment-efficient setting.

To address the challenges of this novel problem setting, we draw inspiration from a variety of fields. In particular, several works have considered exploration with a population of agents (Sutton et al., 2011; Dimakopoulou et al., 2018; Liu et al., 2021), but these have only considered task-specific online RL. Our choice of a diverse population of agents takes inspiration from the field of Quality Diversity (QD, (Pugh et al., 2016; Cully et al., 2015)), where it has been shown that diversity can boost exploration in RL (Conti et al., 2018). Recently, Lim et al. (2022) showed it is possible to train diverse populations inside a world model to acquire new skills. As far as we are aware, we are the first to consider a cascading objective for training diverse populations, which is typically done either synchronously (Parker-Holder et al., 2020) or sequentially (Zhou et al., 2022; Zahavy et al., 2022).

Also related are works that specifically train policies that are able to maximally cover the state space of a given MDP (Agarwal et al., 2020; Modi et al., 2021).

However, these works focus on the more theoretically amenable (and less general) low-rank MDP setting, and hence do not demonstrate any experiments on larger scale problems. Furthermore, we aim to achieve state-space coverage by encouraging *policy diversity*, which is distinct from the approaches taken in these works. We make use of an information theoretic objective, which has been shown to be highly effective for exploration in RL (Mohamed & Rezende, 2015; Houthoofd et al., 2016; Hazan et al., 2019). Most similar to our work, Zhang et al. (2021b) consider the reward free RL setting, and show that maximizing entropy can lead to improved exploration that can facilitate transfer to arbitrary reward functions. By contrast we consider the limited deployment setting and use model-based RL. Finally, our work relates to collective intelligence (Ha & Tang, 2021; Yahya et al., 2017), since each member of the population contributes a small piece to the puzzle that produces a general world model.

5.2.8 Conclusion

We introduced the *reward-free deployment efficiency* paradigm—an important problem setting for learning generally capable agents in a scalable fashion. To address the challenges of this setting, we proposed CASCADE, a theoretically motivated approach that leverages a diverse population of self-supervised exploration agents. We showed in a variety of experiments that CASCADE produces *general exploration strategies*, adept at both deep exploration *and* gathering sufficiently heterogeneous datasets that facilitate the learning of diverse downstream behaviors. We believe this work provides a new approach for collecting large, diverse datasets in a self-supervised fashion, inducing a potentially open-ended process for training general world models at scale.

For future work, we could consider alternative means for computing diversity, for example using optimization landscapes (Parker-Holder et al., 2020) or explicit behavioral diversity (Eysenbach et al., 2019; Parker-Holder et al., 2020). These could also be combined with representations that directly incorporate the behaviors of the policy (Bharadhwaj et al., 2022) to further improve the design of the “summary”

embedding space. Furthermore, we would be interested in considering transformer models (Vaswani et al., 2017) for trajectory modelling (Chen et al., 2021a; Janner et al., 2021) that ought to scale more gracefully with larger batches of data (Radford et al., 2019). Recent work has shown that this class of models can compose unseen complex multi-task behaviors at test time from fixed offline datasets (Furuta et al., 2022), as well as leverage significantly large and diverse offline datasets to improve behavioral modeling (Reid et al., 2022), highly relevant to both aforementioned desiderata for the reward-free deployment efficiency paradigm.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Learning General World Models in a Handful of Reward-Free Deployments
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Yingchen Xu, Jack Parker-Holder, Aldo Pacchiano, Philip J. Ball, Oleh Rybkin, Stephen J. Roberts, Tim Rocktäschel, Edward Grefenstette. Learning General World Models in a Handful of Reward-Free Deployments. <i>Neural Information Processing Systems 2022</i>

Student Confirmation

Student Name:	Philip Ball		
Contribution to the Paper	Co-created main idea and problem setting, created the cascading loss term, significant proportion of writing.		
Signature 	Date	6/October/2023	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

5.3 Efficient Online Reinforcement Learning with Offline Data

Abstract Sample efficiency and exploration remain major challenges in online reinforcement learning (RL). A powerful approach that can be applied to address these issues is the inclusion of offline data, such as prior trajectories from a human expert or a sub-optimal exploration policy. Previous methods have relied on extensive modifications and additional complexity to ensure the effective use of this data. Instead, we ask: *can we simply apply existing off-policy methods to leverage offline data when learning online?* In this work, we demonstrate that the answer is yes; however, a set of minimal but important changes to existing off-policy RL algorithms are required to achieve reliable performance. We extensively ablate these design choices, demonstrating the key factors that most affect performance, and arrive at a set of recommendations that practitioners can readily apply, whether their data comprise a small number of expert demonstrations or large volumes of sub-optimal trajectories. We see that correct application of these simple recommendations can provide a **2.5×** improvement over existing approaches across a diverse set of competitive benchmarks, with no additional computational overhead. We have released our code here: github.com/ikostrikov/rldp.

5.3.1 Introduction

Deep reinforcement learning (RL) has achieved success in a number of complex domains, such as Atari (Mnih et al., 2015) and Go (Silver et al., 2016), as well as real-world applications like Chip Design (Mirhoseini et al., 2021) and Human Preference Alignment (Ouyang et al., 2022). In many of these settings, strong RL performance is predicated on having large amounts of online interaction with an environment, which is usually made feasible through the use of simulators. In real-world problems, however, we are often confronted with scenarios where samples are expensive, and furthermore, rewards are sparse, often exacerbated by high dimensional state and action spaces.

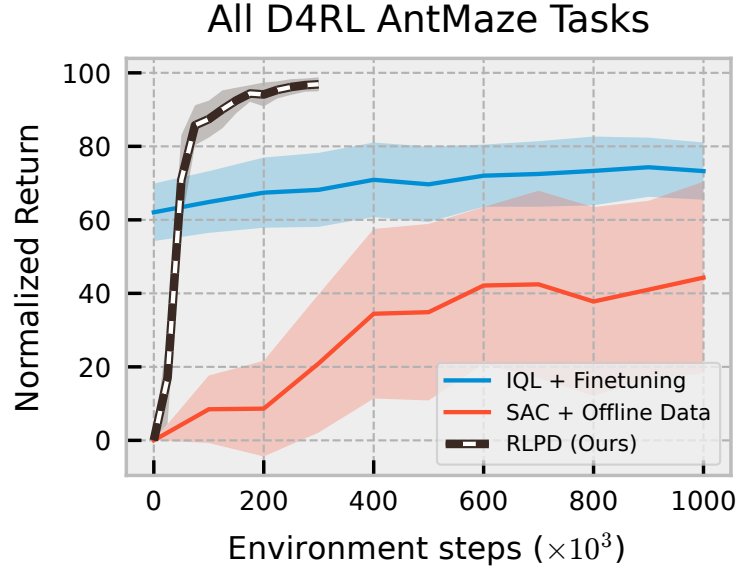


Figure 5.21: Our approach, RLPD, extends standard off-policy RL and achieves reliable state-of-the-art online performance on a number of tasks using offline data. Here we show the difficult D4RL AntMaze domain (10 seeds, 1 std. shaded), averaged over all 6 tasks. We run RLPD for 300k steps due to early convergence.

One promising way to resolve this issue is via the inclusion of data generated by a previous policy or human expert when training deep RL algorithms (often referred to as *offline data* (Levine et al., 2020)), as evidenced theoretically (Wagenmaker & Pacchiano, 2022; Song et al., 2023) and in real-world examples by Cabi et al. (2019); Nair et al. (2020); Lu et al. (2021). This can alleviate challenges due to sample efficiency and exploration by providing the algorithm with an initial dataset to “kick-start” the learning process, either in the form of high-quality expert demonstrations, or even low-quality but high-coverage exploratory trajectories. This also provides us with an avenue to leverage large pre-collected datasets in order to learn useful policies.

Some prior work has focused on using this data through pre-training, while other approaches introduce constraints when training online to handle issues with distribution shift. However, each approach has its drawbacks, such as additional training time and hyperparameters, or limited improvement beyond the behavior policy respectively. Taking a step back, we note that *standard off-policy algorithms* should be able to take advantage of this offline data, and furthermore issues with distribution shift should be alleviated in this setting, as we can explore the

environment online. Thus far however, such methods have seen limited success in this problem setting. Therefore, in this work, we ask the following question: *can we simply apply existing off-policy methods to leverage offline data when learning online, without offline RL pre-training or explicit imitation terms that privilege the prior offline data?*

Through a set of thorough experiments on a collection of widely studied benchmarks, we show that the answer to this question is yes. However, naïvely applying existing online off-policy RL algorithms can result in comparatively poor performance, as we see in Figure 5.21 comparing ‘SAC + Offline Data’ with ‘IQL + Finetuning’. Instead, a minimal set of key *design choices* must be taken into consideration to ensure their success. Concretely, we first introduce a remarkably simple approach to sampling the offline data, which we call “symmetric sampling”, that performs well over a large variety of domains with no hyperparameter tuning. Then, we see that in complex settings (e.g., sparse reward, low volume of offline data, high dimensionality, etc.), it is vital that value functions are prevented from over-extrapolation. To this end, we provide a novel perspective on how Layer Normalization (Ba et al., 2016) implicitly prevents catastrophic value over-extrapolation, thereby greatly improving sample-efficiency and stability in many scenarios, while being a minimal modification to existing approaches. Then, to improve the rate at which the offline data are utilized, we incorporate and compare the latest advances in sample-efficient model-free RL, and find that large ensembles are remarkably effective across a variety of domains. Finally, we identify and provide evidence that key design choices in recent RL literature are in fact environment sensitive, showing the surprising result that environments which share similar properties in fact require entirely different choices, and recommend a *workflow for practitioners* to accelerate their application of our insights to new domains.

We demonstrate that our final approach, which we call **RLPD** (**R**einforcement **L**earning with **P**rior **D**ata) outperforms previously reported results, as we see in Figure 5.21, on many competitive domains, sometimes by $2.5\times$. Crucially, as our changes are minimal, we maintain the attractive properties of online algorithms,

such as ease of implementation and computational efficiency. Furthermore, we see the generality of our approach, which achieves strong performance across a number of diverse offline datasets, from those containing limited expert demonstrations, through to data comprised of high-volume sub-optimal trajectories.

We believe that our insights are valuable to the community and practitioners. We show that online off-policy RL algorithms can be remarkably effective at learning with offline data. However, we show their reliable performance is predicated on several key design choices, namely the way the offline data are sampled, a crucial way of normalizing the critic update, and using large ensembles to improve sample efficiency. While the individual ingredients of RLPD are refreshingly simple modifications on existing RL components, we show that their combination delivers state-of-the-art performance on a number of popular online RL with offline data benchmarks, exceeds the performance of significantly more complex prior methods, and generalizes to a number of different types of offline data, whether it be expert demonstrations or sub-optimal trajectories. We have released RLPD here: github.com/ikostrikov/rlpd.

5.3.2 Related work

Offline RL pre-training. We note connections to offline RL (Ernst et al., 2005; Fujimoto et al., 2019; Levine et al., 2020); many prior works perform offline RL, followed by online fine-tuning (Hester et al., 2018; Kalashnikov et al., 2018; Nair et al., 2020; Lee et al., 2021; Kostrikov et al., 2022). Notably, Lee et al. (2021) also considers large ensembles and multiple gradient-step per timestep regimes when learning online. However, our approach uses a significantly simpler sampling mechanism with no hyperparameters and does not rely on costly offline pre-training, which introduces yet *additional* hyperparameters. We also emphasize that our normalized update is *not an offline RL method*—we do not perform any offline pre-training but run online RL from scratch with offline data included in a replay buffer.

Constraining to prior data. An alternative to the offline RL pre-training paradigm is to explicitly constrain the online agent updates such that it exhibits behavior that resembles the offline data (Levine & Koltun, 2013; Fox et al., 2016; Hester et al., 2018; Nair et al., 2018; Rajeswaran et al., 2018; Rudner et al., 2021). Particularly relevant to our approach is work by Rajeswaran et al. (2018), which augments a policy gradient update with a weighted update that explicitly includes *demonstration* data. In contrast, we use a sample-efficient off-policy paradigm, and *do not* perform any pre-training. Also similar to our work is that by Nair et al. (2018), who also use an off-policy algorithm with a fixed offline replay buffer. However, we do not restrict the policy using a behavior cloning term, and do not reset to demonstration states. Moreover, we note that these approaches generally require the offline data to be high quality (i.e., ‘learning from demonstration data’ (Asada & Hanafusa, 1979; Schaal, 1996)), while our approach is, importantly, agnostic to the quality of the data.

Unconstrained methods with prior data. Prior work has also considered ways of incorporating offline data without any constraints. Some methods focus on initializing a replay buffer with offline data (Večerík et al., 2017; Hester et al., 2018), while other works have utilized a balanced sampling strategy to handle online and offline data (Nair et al., 2018; Kalashnikov et al., 2018; Hansen et al., 2022; Zhang et al., 2023). Most recently, Song et al. (2023) presented a theoretical analysis of such approaches, showing that balanced sampling is important both in theory and practice. In our experiments, we also show that balanced sampling helps online RL with offline data; however, directly using this approach on a range of benchmark tasks is insufficient, and other design decisions that we present are critical in achieving good performance across all tasks.

5.3.3 Preliminaries

We consider problems that can be formulated as a Markov Decision Process (MDP) (Bellman, 1957), described as a tuple $(\mathcal{S}, \mathcal{A}, \gamma, p, r, d_0)$ where $\mathcal{S}$ is the state space,

$\mathcal{A}$ is the action space and $\gamma \in (0, 1)$ is the discount factor. The dynamics are governed by a transition function $p(s'|s, a)$; there is a reward function $r(s, a)$ and initial state distribution $d_0(s)$. The goal of RL is then to maximize the expected sum of discounted rewards: $\mathbb{E}_\pi [\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t)]$.

In this work, we focus on RL while having access to offline datasets $\mathcal{D}$ (Levine et al., 2020), a collection of (s, a, r, s') tuples generated from a particular MDP. A key property of offline datasets is they usually do not provide complete state-action coverage, i.e., $\{s, a \in \mathcal{D}\}$ is a small subset of $\mathcal{S} \times \mathcal{A}$. Due to this lack of on-policy coverage, methods using function approximation may over-extrapolate values when learning on this data, leading to a pronounced effect on learning performance (Fujimoto et al., 2019).

5.3.4 Online RL with Offline Data

As outlined in Section 5.3.3, we consider the standard RL setting with the addition of a pre-collected dataset. In this work, we aim to design a general approach that is agnostic to the *quality and quantity* of this pre-collected data. For instance, this data could take the form of a handful of human demonstrations, or swathes of sub-optimal, exploratory data. Furthermore, we wish to make recommendations that are agnostic to the nature of the problem setting, such as whether the observations are state or pixel-based, or whether the rewards are sparse or dense.

To this end, we present an approach based on off-policy model-free RL, without pre-training or explicit constraints, which we call **RLPD** (**R**einforcement **L**earning with **P**rior **D**ata). As discussed in Section 5.3.4, we base our algorithm design on SAC (Haarnoja et al., 2018a;b), though in principle these design choices may improve other off-policy RL approaches. First, we propose a simple mechanism for incorporating the prior data. Then, we identify a pathology that exists when naïvely applying off-policy methods to this problem setting, and propose a simple and minimally invasive solution. After, we improve the rate the offline data are utilized by incorporating the latest approaches in sample-efficient RL. Finally, we

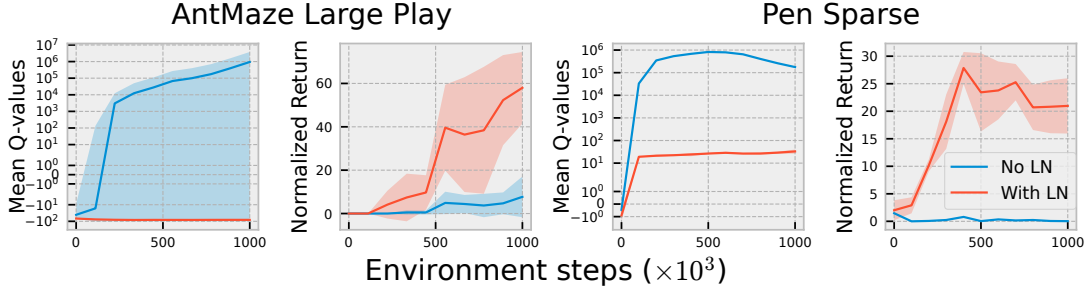


Figure 5.22: Using SAC with our symmetric sampling method can result in instabilities due to diverging Q-values; with LayerNorm in the critic this disappears, improving performance.

highlight common design choices in recent deep RL that are in fact environment sensitive, and should be adjusted accordingly by practitioners.

Design Choice 1: A Simple and Efficient Strategy to Incorporate Offline Data

We start with a simple approach that incorporates prior data which adds no computational overhead, yet is agnostic to the nature of the offline data. We call this ‘symmetric sampling’, whereby for each batch we sample 50% of the data from our replay buffer, and the *remaining 50% from the offline data buffer*, resembling the scheme used by Ross & Bagnell (2012). As we will see in later sections, this sampling strategy is surprisingly effective across a variety of scenarios, and we extensively ablate various elements of this scheme (see Section 5.3.5). However, applying this approach to canonical off-policy methods, such as SAC (Haarnoja et al., 2018a), does not yield strong performance, as we see in Figure 5.21, and further design choices must be taken into consideration.

Design Choice 2: Layer Normalization Mitigates Catastrophic Overestimation

Standard off-policy RL algorithms query the learned Q-function for out-of-distribution (OOD) actions, which might not be defined during learning. Consequently, there can be significant overestimation of actual values due to the use of function approximation (Thrun & Schwartz, 1993). In practice, this phenomenon leads to training instabilities and possible *divergence* when the critic is trying to catch up with a constantly increasing value.

In particular, we find this to be the case when naïvely applying our symmetric sampling approach for complex tasks (see Figure 5.22). Critic divergence is a well-studied problem, particularly in the *offline* regime, where the policy cannot generate new experience. In our problem setting, however, we *can* sample from the environment. Therefore, instead of creating a mechanism that explicitly discourages OOD actions, which can be viewed as anti-exploration (Rezaeifar et al., 2022), we instead need to simply ensure that the learned functions do not extrapolate in an unconstrained manner. To this end, we show that Layer Normalization (LayerNorm) (Ba et al., 2016) can bound the extrapolation of networks but, crucially, *does not* explicitly constrain the policy to remain close to the offline data. This in turn does not discourage the policy from exploring unknown and *potentially valuable* regions of the state-action space. In particular, we demonstrate that LayerNorm bounds the values and empirically prevents catastrophic value extrapolation. Concretely, consider a Q-function Q parameterized by θ, w , applying LayerNorm and intermediate representation $\psi_\theta(\cdot, \cdot)$. For any a and s we can say⁶:

$$\begin{aligned} \|Q_{\theta,w}(s, a)\| &= \|w^T \text{relu}(\psi_\theta(s, a))\| \\ &\leq \|w\| \|\text{relu}(\psi_\theta(s, a))\| \leq \|w\| \|\psi(s, a)\| \\ &\leq \|w\| \end{aligned}$$

Therefore, as a result of *Layer Normalization*, the Q-values are bounded by the norm of the weight layer, even for actions outside the dataset. Thus, the effect of erroneous action extrapolation is greatly mitigated, as their Q-values are unlikely to be significantly greater than those already seen in the data. Indeed, referring back to Figure 5.22, we see that introducing LayerNorm into the critic greatly improves performance through mitigating critic divergence.

To illustrate this, we generate a dataset with inputs x distributed in a circle with radius 0.5 and labels $y = \|x\|$. We study how a standard two-layer MLP with ReLU activations (common in deep RL) extrapolates outside of the data distribution, and

⁶For simplicity, we consider LayerNorm without bias terms. This does not change the analysis, as it is a constant.

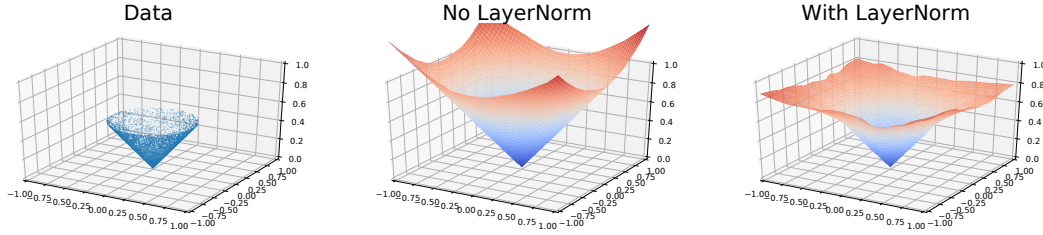


Figure 5.23: We fit data (left) with a two-layer MLP without LayerNorm (center) and with LayerNorm (right). LayerNorm bounds the values and prevents catastrophic overestimation.

the effect of adding LayerNorm. In Figure 5.23, the standard parameterization leads to unbounded extrapolation outside of the support, while LayerNorm bounds the values, greatly reducing the effect of uncontrolled extrapolation.

Design Choice 3: Sample Efficient RL

We now have an online approach leveraging offline data that also suppresses extreme value extrapolation, whilst maintaining the freedom of an unconstrained off-policy method. However, a benefit of offline and constrained approaches is that they have an explicit mechanism to efficiently incorporate prior data, such as through pre-training (Hester et al., 2018; Lee et al., 2021), or an auxiliary supervision term (Nair et al., 2018; Rudner et al., 2021) respectively. In our case, the incorporation of prior data is implicit through the use of online Bellman backups over offline transitions. Therefore, it is imperative that these Bellman backups are performed as sample-efficiently as possible.

One way to achieve this is to increase the number of updates we perform per environment step (also referred to as update-to-data (UTD) ratio), allowing the offline data to become “backed-up” more quickly. However, as highlighted in recent literature in online RL, this can create issues in the optimization process and ironically *reduce sample efficiency*, due to statistical over-fitting (Li et al., 2022). To ameliorate this, prior work has suggested a number of regularization approaches, such as simple L2 normalization (Večerík et al., 2017), Dropout (Srivastava et al., 2014; Hiraoka et al., 2022) and random ensemble distillation (Chen et al., 2021b). In this work, we settle on the latter approach of random ensemble distillation;

we will demonstrate through ablations that this performs strongest, particularly on sparse reward tasks.

We also note that value over-fitting issues exist when performing TD-learning from images (Cetin et al., 2022). Therefore in these settings, we further include random shift augmentations (Kostrikov et al., 2021; Yarats et al., 2022b).

Per-Environment Design Choices

Having highlighted the 3 key design choices for our approach that can be applied generally to all environments and offline datasets, we now turn our attention to design choices that are commonly taken for granted, but can in fact be environment-sensitive. It is well documented that deep RL algorithms are sensitive to implementation details (Henderson et al., 2018; Engstrom et al., 2020; Andrychowicz et al., 2021; Furuta et al., 2021). As a result, many works in deep RL require per-environment hyperparameter tuning. Given the huge variety of tasks we consider in our experiments, we believe it is important to contribute to this discourse, and highlight that certain design choices, which are often simply inherited from previous implementations, should in fact be *carefully reconsidered*, and may explain why off-policy methods have not been competitive thus far on the problems we consider. We therefore take a view that, given the well-documented sensitivity of deep RL, it is important to demonstrate a critical path of design choices to consider when assessing new environments, and provide a *workflow* to simplify this process for practitioners.

Clipped Double Q-Learning (CDQ). Value-based methods combined with function approximation and stochastic optimization suffer from estimation uncertainty, which, in combination with the maximization objective of Q-learning, leads to value overestimation (van Hasselt et al., 2016) (as also discussed in Section 5.3.4). In order to mitigate this issue, Fujimoto et al. (2018) introduce Clipped Double Q-Learning (CDQ) which involves taking a minimum of an ensemble of two Q-functions for computing TD-backups. They define the targets for updating the

critics as follows:

$$y = r(s, a) + \gamma \min_{i=1,2} Q_{\theta_i}(s', a') \text{ where } a' \sim \pi(\cdot|s').$$

However, this corresponds to fitting target Q-values that are 1 std. below the actual target values, and recent work (Moskovitz et al., 2021b) suggests that this design choice may not be universally useful as it can be too conservative. Therefore, this is important to reconsider, especially outside of the domains for which it was originally designed, such as sparse reward tasks prevalent in our problem setting.

Maximum Entropy RL. MaxEnt RL augments the traditional return objective with an entropy term:

$$\max_{\pi} \mathbb{E}_{s \sim \rho^{\pi}, a \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t (r_t + \alpha \mathcal{H}(\pi(a|s))) \right].$$

This corresponds to maximizing the discounted reward and expected policy entropy at each time step. The motivation for these approaches is centered around robustness and exploration (i.e., maximize reward while behaving as randomly as possible). Approaches relying on this objective are empirically impressive (Haarnoja et al., 2018a;b; Chen et al., 2021b; Hiraoka et al., 2022). We, therefore, believe this design choice is of interest in the context of online fine-tuning, where rewards are often sparse and require exploration.

Architecture. Network architecture can have a significant impact on deep RL performance, and the same architecture that can be optimal in one environment can be sub-optimal in another (Furuta et al., 2021). To simplify the search space, we consider the impact of having 2 or 3 layers in the actor and critic, which have been shown to affect performance, even on canonical tasks (Ball & Roberts, 2021).

RLPD: Approach Overview

Here we present pseudo-code for our approach, highlighting in **Green** elements that are important to our approach, and in **Purple**, environment-specific design choices.

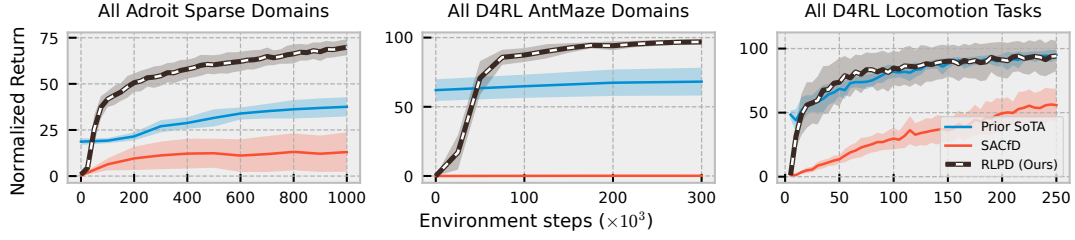


Figure 5.24: RLPD exceeds prior state-of-the-art performance on a number of different popular benchmarks whilst being significantly simpler. Results are aggregated over **21 different environments** (10 Seeds, 1 std. shaded). In each case, we compare to the prior best known work (IQL + Finetuning in Adroit and AntMaze, Off2On in Locomotion), and SACfD, a canonical off-policy approach using offline data.

The key factors of RLPD reside in lines 1 and 13 of Algorithm 7 with adopting LayerNorm, large ensembles, sample efficient learning and a symmetric sampling approach to incorporate online and offline data. For environment specific choices, we recommend the following as a starting point:

- *Line 3:* Subset 2 critics
- *Line 16:* Remove entropy
- *Line 1:* Utilize a *deeper* 3 layer MLP

As a pragmatic workflow, we recommend ablating these **Purple** design choices first, and in the order stated above.

5.3.5 Experiments

We design our experiments to not only demonstrate the importance of our design choices, but also provide the insights that allow practitioners to quickly adapt RLPD to their problems. As such, we aim to answer the following questions:

1. Is RLPD competitive with prior work despite using *no pre-training nor having explicit constraints*?
2. Does RLPD transfer to *pixel-based* environments?
3. Does LayerNorm *mitigate value divergence*?
4. Does the proposed workflow around environment-specific design choices lead to *reliable performance*?

Algorithm 7 Online RL with Offline Data (RLPD)

```

1: Select LayerNorm, Large Ensemble Size  $E$ , Gradient Steps  $G$ , and architecture.

2: Randomly initialize Critic  $\theta_i$  (set targets  $\theta'_i = \theta_i$ ) for  $i = 1, 2, \dots, E$  and Actor
    $\phi$  parameters. Select discount  $\gamma$ , temperature  $\alpha$  and critic EMA weight  $\rho$ .
3: Determine number of Critic targets to subset  $Z \in \{1, 2\}$ 
4: Initialize empty replay buffer  $\mathcal{R}$ 
5: Initialize buffer  $\mathcal{D}$  with offline data
6: while True do
7:   Receive initial observation state  $s_0$ 
8:   for  $t = 0, T$  do
9:     Take action  $a_t \sim \pi_\phi(\cdot | s_t)$ 
10:    Store transition  $(s_t, a_t, r_t, s_{t+1})$  in  $\mathcal{R}$ 
11:    for  $g = 1, G$  do
12:      Sample minibatch  $b_R$  of  $\frac{N}{2}$  from  $\mathcal{R}$ 
13:      Sample minibatch  $b_D$  of  $\frac{N}{2}$  from  $\mathcal{D}$ 
14:      Combine  $b_R$  and  $b_D$  to form batch  $b$  of size  $N$ 
15:      Sample set  $\mathcal{Z}$  of  $Z$  indices from  $\{1, 2, \dots, E\}$ 
16:      With  $b$ , set
          
$$y = r + \gamma \left( \min_{i \in \mathcal{Z}} Q_{\theta'_i}(s', \tilde{a}') \right), \quad \tilde{a}' \sim \pi_\phi(\cdot | s')$$

17:      Add entropy term  $y = y + \gamma \alpha \log \pi_\phi(\tilde{a}' | s')$ 
18:      for  $i = 1, E$  do
19:        Update  $\theta_i$  minimizing loss:
          
$$L = \frac{1}{N} \sum_i (y - Q_{\theta_i}(s, a))^2$$

20:      end for
21:      Update target networks  $\theta'_i \leftarrow \rho \theta'_i + (1 - \rho) \theta_i$ 
22:    end for
23:    With  $b$ , update  $\phi$  maximizing objective:
          
$$\frac{1}{E} \sum_{i=1}^E Q_{\theta_i}(s, \tilde{a}) - \alpha \log \pi_\phi(\tilde{a} | s), \quad \tilde{a} \sim \pi_\phi(\cdot | s)$$

24:  end for
25: end while

```

For (1), we compare RLPD to those which have been designed to use offline data to accelerate online learning. For (2), we consider an additional suite of tasks to study RLPD's applicability to vision-based domains. Then, for (3) we perform analysis to demonstrate the importance of using LayerNorm. Lastly, for (4) we demonstrate

our proposed workflow on the most challenging tasks (see Subsection 5.3.4).

How does RLPD compare? We consider the following **21 tasks** from established benchmarks:

- **Sparse Adroit** (Nair et al., 2020). These 3 dexterous manipulation tasks—pen-spinning, door-opening, ball relocation—are challenging, sparse-reward tasks. The offline data are multi-modal, with a small set of human demonstrations and a large set of trajectories from a behavior-cloned policy trained on this human data. We follow the *rigorous evaluation criteria* of Kostrikov et al. (2022), whereby performance is based on completion speed, rather than success rate. IQL + Finetuning represents the strongest prior work (Kostrikov et al., 2022).
- **D4RL AntMaze** (Fu et al., 2020). These 6 sparse reward tasks require an Ant agent to learn to walk and navigate through a maze to reach a goal. The offline data comprise *only sub-optimal* trajectories that can in principle be “stitched” together. Again, IQL + Fine-tuning represents the strongest prior work (Kostrikov et al., 2022).
- **D4RL Locomotion** (Fu et al., 2020). Lastly, we have 12 dense reward, locomotion tasks featuring offline data with varying levels of expertise. Off2On (Lee et al., 2021) has state-of-the-art performance on this suite of tasks.

For evaluation, we first include **SACfD**, a baseline studied in prior work (Večerík et al., 2017; Nair et al., 2020), which, similar to RLPD, is an off-policy approach that incorporates offline data during training. However, SACfD simply *initializes* the online replay buffer with the offline data. We implement this baseline using SAC without the additional design decisions discussed in Section 5.3.4. Then, since there is no *single* prior method that achieves the best performance across all groups of tasks, we compare to the state-of-the-art method specific to each group (as listed above). We refer to this comparison in our plots as **Prior SoTA**. For all experiments in this work, we report the mean and standard deviation across *10 seeds* and aggregate the results across tasks within the three groups

listed. For full detailed results broken down by task, see Appendix F.1, and for more environment details, see Appendix F.2.1.

We see that RLPD performs strongly, either matching or *significantly exceeding* the best known prior work on these challenging benchmarks (see Figure 5.24). We reiterate that we present results for RLPD and SACfD without doing *any pre-training*, unlike the Prior SoTA methods. So, while prior work (shown in blue) may achieve strong initial performance, the online improvement is more modest. On the other hand, our method reaches or surpasses this performance in the order of just 10k online samples. Notably, we outperform the best reported performance on the Sparse Adroit ‘Door’ task by $2.5\times$. Moreover, to our knowledge, our method is the first to effectively ‘solve’ *all* AntMaze tasks. Furthermore, we are able to do so in *less than a third* the time-step budget allocated to prior methods.

Does RLPD transfer to pixels? Here we consider the medium and expert locomotion tasks in V-D4RL (Lu et al., 2022), an offline dataset with *only pixel observations*. V-D4RL is particularly challenging as the behavior policies are state-based, which means the data is partially observable in pixel-space. This is most obvious in ‘Humanoid Walk’, where body parts are visually occluded, thus behavior cloning (BC) can struggle to achieve strong performance. For evaluation, as our method seeks to accelerate online learning with offline data, we focus on data-efficiency—we introduce a challenge that we call “10%DMC”, which involves training policies using only 10% of the total timesteps recommended by Yarats et al. (2022b). As we use the medium and expert data, we include a **BC** baseline. To evaluate RLPD’s ability to efficiently use offline data to boost online learning, we compare to a baseline approach that does not use the offline data. To isolate the effect of the utilization of offline data, we use the same architecture and policy optimizer as our method and label this baseline as **Online** in our plots. Then, to evaluate how this compares to the state-of-the-art sample-efficient RL method from pixels, we compare to **DrQ-v2** (Yarats et al., 2022b).

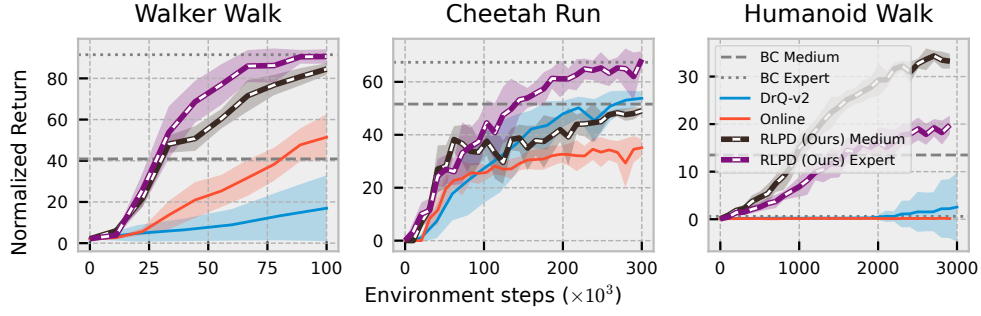


Figure 5.25: Our approach generalizes to vision-based domains, providing consistent improvements over existing approaches.

We see in Figure 5.25⁷ that RLPD provides consistent improvements over purely online approaches, and in many cases greatly improves over a BC baseline. We see through the difference in RLPD (dashed black and purple lines) and the Online baseline that RLPD *effectively* utilizes the offline data to bootstrap learning. This conclusion holds as we compare to the SoTA vision-based RL method (blue).

Lastly, we test increasing UTD to 10 on one of the tasks—Cheetah Run with Expert offline data. Figure 5.26 shows a remarkable improvement in performance when learning with the offline dataset. To our knowledge, this is the first demonstration of a high UTD approach improving model-free pixel-based continuous control.

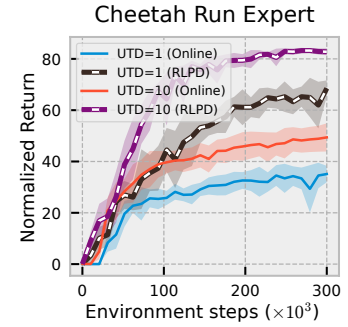


Figure 5.26: Increasing UTD with RLPD greatly improves sample efficiency from pixels.

RLPD Analysis and Ablation Study

Here, we address (3) and (4) by quantifying the effect of LayerNorm, and demonstrating the reliability of our proposed workflow (see Subsection 5.3.4).

Does LayerNorm mitigate value divergence? We now illustrate the importance of LayerNorm in mitigating catastrophic value divergence, and focus on tasks where overestimation is particularly prone. In Figure 5.27, we see LayerNorm is crucial for strong performance in the Adroit domain; excluding LayerNorm results in significantly higher variance across seeds and reduces mean performance.

To more clearly illustrate this effect, we construct a dataset of *only the expert human demonstration data* from the Adroit Sparse tasks (see “Expert Adroit Sparse

⁷V-D4RL normalized return is episode return divided by 10.

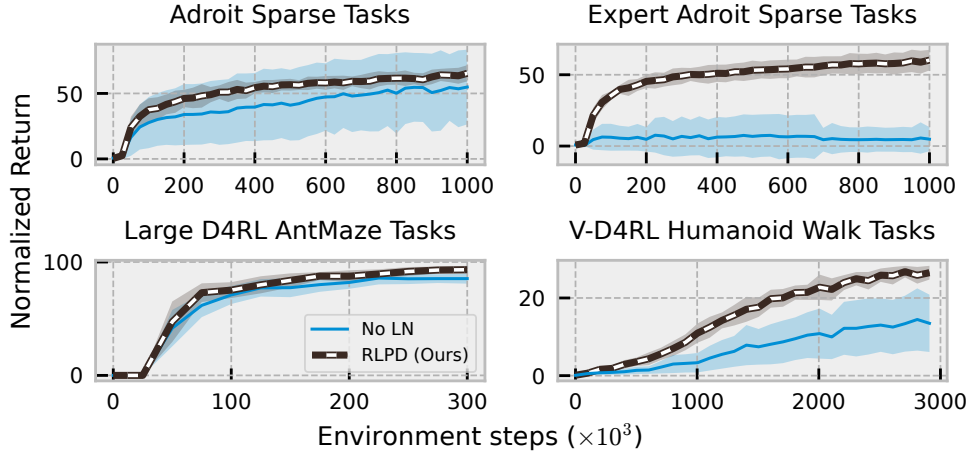


Figure 5.27: LayerNorm is crucial for strong performance, particularly when data are limited or narrowly distributed.

Tasks” in Figure 5.27). This subset comprises just **22** of the 500 trajectories in the original dataset and is much more narrowly distributed by nature—representing a task with sparse rewards, limited demonstrations, and narrow offline data coverage—likely to exacerbate value divergence. Here we see a remarkable result: RLPD still *exceeds* prior work, despite significant restrictions in data. Moreover, removing LayerNorm now results in collapsed performance, with no progress made on any task. We further observe improvements in sample efficiency through the inclusion of LayerNorm in AntMaze and Humanoid Walk through reducing excessive extrapolation. Additional experiments and results are in Appendix F.1.

Design choice workflow. We now motivate our workflow in Section 5.3.4, demonstrating the importance of these design choices. We focus on the hardest tasks, namely ‘Relocate’ in Adroit Sparse, ‘Large Diverse’ in AntMaze, and ‘Humanoid Expert’ in V-D4RL, as we found these to be most sensitive. We provide full results on all tasks in Appendix F.1, noting that optimal decisions in the harder domains also produce optimal results in the easier domains.

We see in Figure 5.28 that with the recommended environment-specific design choices provide strong performance; we see using entropy backups and smaller networks always results in worse performance. In ‘AntMaze Large Diverse’ however we see that with CDQ, performance deteriorates. Following our workflow, and ablating this by subsetting 1 critic is crucial to recovering strong performance. The

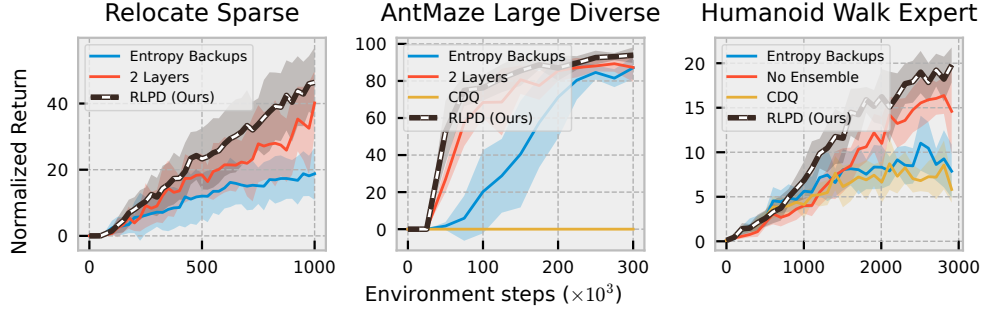


Figure 5.28: Our recommended starting design choices and workflow leads to strong performance on all tasks.

same applies to ‘Humanoid Walk Expert’, whereby we surprisingly see that CDQ is detrimental to performance, despite its popularity in recent implementations. We also show the surprising positive effect of larger ensembles in pixel-based tasks, with the standard 2 member critic ensemble performing worse than the 10 member ensemble we use by default in RLPD.

Lastly, we conduct additional ablations to understand the importance of the design choices that we propose for our method, showing that, though the individual design decisions are simple, they are vital for good performance.

Critic regularization. Here we examine the effects of critic regularization on performance. We compare 3 approaches: weight-decay⁸ (Večerík et al., 2017), Dropout (Hiraoka et al., 2022) and ensembling (Chen et al., 2021b). We choose a subset of the prior experiments to evaluate on from the AntMaze, Adroit, and Locomotion sections.

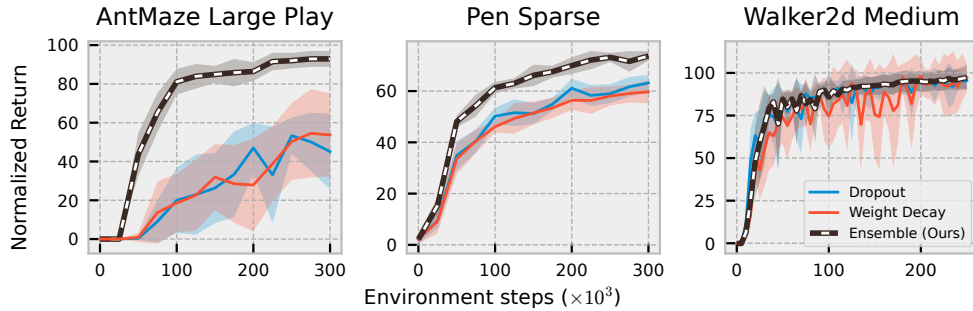


Figure 5.29: In general, critic ensembling provides the best performance. Dropout performs worse in sparse reward tasks.

⁸We found a weight decay value of 0.01 worked best across a number of settings.

In Figure 5.29, we see that ensembling is the strongest form of regularization. Notably, while Dropout performs well in the Locomotion domain ‘walker2d-medium-v0’, as affirmed by Hiraoka et al. (2022), it does not generalize to challenging sparse reward environments. We also see that weight-decay regularization is less performant in all domains.

Buffer initialization. In this section, we compare symmetric sampling with that of one which relies on *initializing* a replay buffer with offline data.

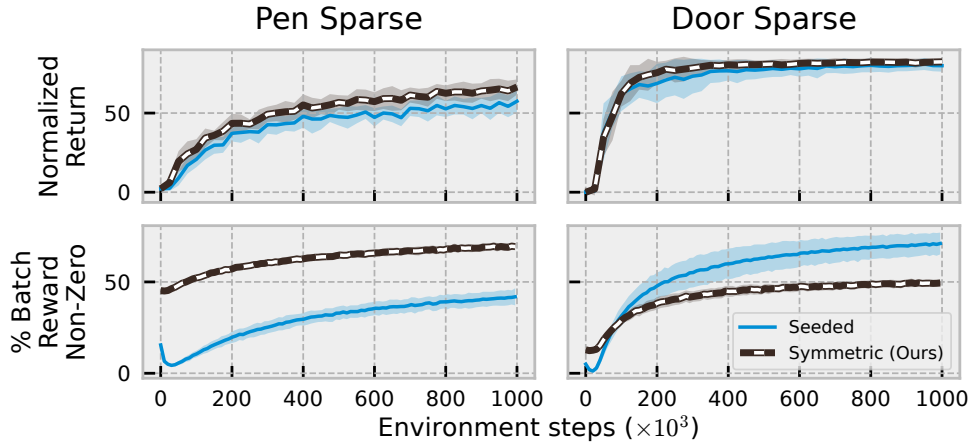


Figure 5.30: Symmetric sampling improves sample efficiency and reduces variance across seeds, and does not work by simply increasing the reward density in a batch.

First returning to the challenging Human Expert Demonstrations setting in Adroit, in Figure 5.30 we show two contrasting examples that demonstrate symmetric sampling effectively trades-off between replay and offline data. We observe that in the Pen environment, symmetric sampling clearly improves exploration by ensuring increasing reward density within mini-batches. In contrast, we see that although the buffer initialization

approach explores just as well in the Door environment, symmetric sampling has the important effect of improving stability and decreases variance due to relying less on the higher-variance data generated by the online policy.

We now consider a situation whereby we have abundant sub-optimal offline data, and see that again, our balanced sampling approach improves sample-efficiency. In

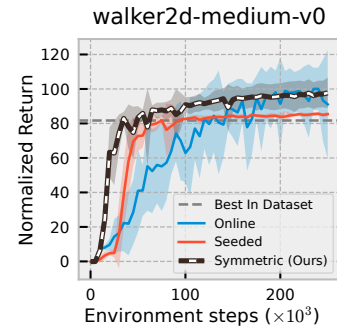


Figure 5.31: Initializing the buffer with large amounts of data limits improvement.

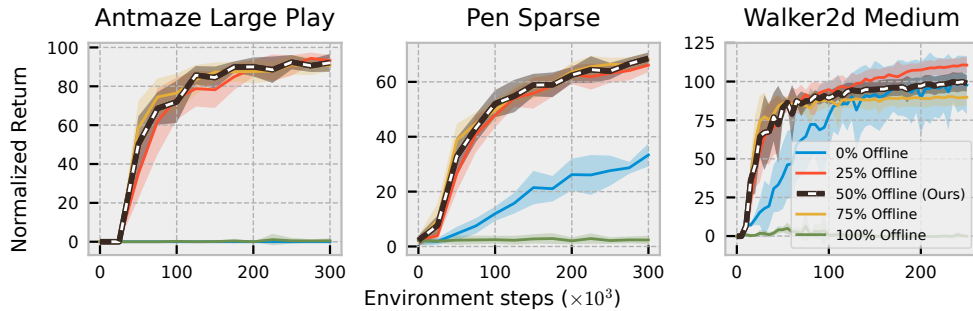


Figure 5.32: RLPD is not sensitive to replay proportion; 50% offers the best compromise between variance, speed of convergence, and asymptotic performance.

Figure 5.31, initializing the buffer with high volumes of medium quality locomotion data gives initial improvement over online performance, but struggles asymptotically, likely due to a lack of on-policy data sampling, vital for online improvement. On the other hand, our symmetric sampling approach improves sample efficiency and matches asymptotic performance while reducing variance.

Sampling proportion sensitivity. We assess the sensitivity of our sampling approach away from the “symmetric” ratio of 50% online/offline.

As we see in Figure 5.32, RLPD is not very sensitive to sampling proportion. While sampling 25% offline can marginally help asymptotic performance in ‘walker2d-medium-v0’, this comes at the expense of variance and sample efficiency in sparse reward tasks. We also affirm with our 100% offline results that RLPD is not an offline method, and key to its success is how it controls for divergence *without restricting exploration or behavior learning*.

5.3.6 Conclusion

In this work, we show that off-policy approaches can be adapted to leverage offline data when training online. We see that with careful application of key design choices, RLPD can attain remarkably strong performance, and demonstrate this on a total of **30 different tasks**. Concretely, we show that the unique combination of symmetric sampling, LayerNorm as a value extrapolation regularizer, and sample efficient learning is key to its success, resulting in our outperforming prior work by up to **2.5×** on a large variety of competitive benchmarks. Moreover, our recommendations have

negligible impact on computational efficiency compared to pure-online approaches, and are simple, allowing practitioners to easily incorporate the insights of this work into existing approaches. To further improve adoptability, we recommend and demonstrate a workflow for practitioners that improves performance on a wide variety of tasks, and thus demonstrate that certain canonical design choices should be reconsidered when applying off-policy methods. Finally, to facilitate future research, we have released the RLPD codebase here: github.com/ikostrikov/rlpd, which features highly optimized off-policy algorithms for proprioceptive and pixel-based tasks in a single codebase written in JAX (Bradbury et al., 2018).

5.3.7 Errata

The original manuscript did not define what the shaded regions represented. The shading featured in all figures of this work refer to ± 1 standard deviation across seeds.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Efficient Online Reinforcement Learning with Offline Data
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Philip J. Ball, Laura Smith, Ilya Kostrikov, Sergey Levine. Efficient Online Reinforcement Learning with Offline Data. <i>International Conference on Machine Learning 2023</i> .

Student Confirmation

Student Name:	Philip Ball		
Contribution to the Paper	Contributed to main idea, ran all analysis, came up with proposed workflow, ran all experiments, wrote majority of the paper, contributed to the codebase, discovered utility of the key concepts introduced.		
Signature 	Date	06/October/2023	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Stephen Roberts		
Supervisor comments I confirm that Phil made substantive first-author contributions to the paper		
Signature 	Date	6/10/23

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter Bibliography

Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational option discovery algorithms. *CoRR*, abs/1807.10299, 2018.

Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. FLAMBE: Structural complexity and representation learning of low rank mdps. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 20095–20107. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/e894d787e2fd6c133af47140aa156f00-Paper.pdf>.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Shipra Agrawal and Navin Goyal. Analysis of Thompson Sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pp. 39–1. JMLR Workshop and Conference Proceedings, 2012.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as I can and not as I say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. OPAL: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

Dario Amodei and Jack Clark. Faulty Reward Functions in the Wild. <https://blog.openai.com/faulty-reward-functions/>, 2016.

Ankesh Anand, Jacob C Walker, Yazhe Li, Eszter Vértés, Julian Schrittwieser, Sherjil Ozair, Theophane Weber, and Jessica B Hamrick. Procedural generalization by planning with self-supervised world models. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=FmBegXJToY>.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters for on-policy deep actor-critic methods? a large-scale study. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=nIAxjsniDzg>.

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray,

- Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.
- Rika Antonova, Silvia Cruciani, Christian Smith, and Danica Kragic. Reinforcement learning for pivoting task. *arXiv preprint arXiv:1703.00472*, 2017.
- Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=OMNB1G5xzd4>.
- Haruhiko Asada and Hideo Hanafusa. Playback control of force teachable robots. *Transactions of the Society of Instrument and Control Engineers*, 15(3):410–411, 1979. doi: 10.9746/sicetr1965.15.410.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL <https://arxiv.org/abs/1607.06450>.
- Philip Ball, Jack Parker-Holder, Aldo Pacchiano, Krzysztof Choromanski, and Stephen Roberts. Ready policy one: World building through active learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML, 2020*.
- Philip J. Ball and Stephen J. Roberts. Offcon³: What is state of the art anyway?, 2021. URL <https://arxiv.org/abs/2101.11331>.
- Philip J Ball, Cong Lu, Jack Parker-Holder, and Stephen Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 619–629, 2021.
- Marc Bellemare, Salvatore Candido, Pablo Castro, Jun Gong, Marlos Machado, Subhodeep Moitra, Sameera Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588:77–82, December 2020.
- Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *CoRR*, abs/1207.4708, 2012.
- Richard Bellman. A Markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957. ISSN 0022-2518.
- Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019.
- Homanga Bharadhwaj, Mohammad Babaeizadeh, Dumitru Erhan, and Sergey Levine. Information prioritization through empowerment in visual model-based RL. In *International Conference on Learning Representations*, 2022.

- James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL <http://github.com/google/jax>.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.
- Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019.
- Arunkumar Byravan, Jost Tobias Springenberg, Abbas Abdolmaleki, Roland Hafner, Michael Neunert, Thomas Lampe, Noah Siegel, Nicolas Manfred Otto Heess, and Martin A. Riedmiller. Imagined value gradients: Model-based policy optimization with transferable latent dynamics models. In *CoRL*, 2019.
- Serkan Cabi, Sergio Gomez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott E. Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg O. Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyun Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *Robotics: Science and Systems XVI*, 2019.
- Andres Campero, Roberta Raileanu, Heinrich Kuttler, Joshua B. Tenenbaum, Tim Rocktäschel, and Edward Grefenstette. Learning with AMIGo: Adversarially motivated intrinsic goals. In *International Conference on Learning Representations*, 2021.
- Edoardo Cetin, Philip J Ball, Stephen Roberts, and Oya Celiktutan. Stabilizing off-policy deep reinforcement learning from pixels. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2784–2810. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/cetin22a.html>.
- Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021a. URL <https://openreview.net/forum?id=a7APmM4B9d>.
- Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized ensembled double Q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021b. URL <https://openreview.net/forum?id=AY8zfZm0tDd>.
- Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for OpenAI Gym. <https://github.com/maximecb/gym-minigrid>, 2018.
- Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems 31*, pp. 4754–4765, 2018.

- Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, volume 87 of *Proceedings of Machine Learning Research*, pp. 617–629. PMLR, 2018. URL <http://proceedings.mlr.press/v87/clavera18a.html>.
- Ignasi Clavera, Anusha Nagabandi, 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*, 2019. URL <https://openreview.net/forum?id=HyztsoC5Y7>.
- Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Improving exploration in Evolution Strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, pp. 5032–5043, USA, 2018. Curran Associates Inc.
- Brandon Cui, Hengyuan Hu, Luis Pineda, and Jakob Nicolaus Foerster. K-level reasoning for zero-shot coordination in hanabi. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.
- Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521:503–507, 2015.
- Jonas Degraeve, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de las Casas, Craig Donner, Leslie Fritz, Cristian Galperti, Andrea Huber, James Keeling, Maria Tsimpoukelli, Jackie Kay, Antoine Merle, Jean-Marc Moret, Seb Noury, Federico Pesamosca, David Pfau, Olivier Sauter, Cristian Sommariva, Stefano Coda, Basil Duval, Ambrogio Fasoli, Pushmeet Kohli, Koray Kavukcuoglu, Demis Hassabis, and Martin Riedmiller. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602:414–419, February 2022.
- Maria Dimakopoulou, Ian Osband, and Benjamin Van Roy. Scalable coordinated exploration in concurrent reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
- Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI’16*, pp. 1432–1440. AAAI Press, 2016. ISBN 9781577357704.
- Gabriel Dulac-Arnold, Daniel J. Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *CoRR*, abs/1904.12901, 2019.
- Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return then explore. *Nature*, 590 7847:580–586, 2021.

- Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=r1etN1rtPB>.
- Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(18):503–556, 2005. URL <http://jmlr.org/papers/v6/ernst05a.html>.
- Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019.
- Mehdi Fatemi, Taylor W Killian, Jayakumar Subramanian, and Marzyeh Ghassemi. Medical dead-ends and learning to identify high-risk states and treatments. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 4856–4870. Curran Associates, Inc., 2021.
- Angelos Filos, Eszter V rt s, Zita Marinho, Gregory Farquhar, Diana Borsa, Abram Friesen, Feryal Behbahani, Tom Schaul, Andr  Barreto, and Simon Osindero. Model-value inconsistency as a signal for epistemic uncertainty. *arXiv preprint arXiv:2112.04153*, 2021.
- Yannis Flet-Berliac, Johan Ferret, Olivier Pietquin, Philippe Preux, and Matthieu Geist. Adversarially guided actor-critic. In *International Conference on Learning Representations*, 2021.
- Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *32nd Conference on Uncertainty in Artificial Intelligence (UAI)*, 2016.
- Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2020.
- Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *ArXiv*, abs/1802.09477, 2018.
- Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 09–15 Jun 2019. URL <https://proceedings.mlr.press/v97/fujimoto19a.html>.
- Hiroki Furuta, Tadashi Kozuno, Tatsuya Matsushima, Yutaka Matsuo, and Shixiang Gu. Co-adaptation of algorithmic and implementational innovations in inference-based deep reinforcement learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vLyI__SoeAe.
- Hiroki Furuta, Yutaka Matsuo, and Shixiang Shane Gu. Generalized decision transformer for offline hindsight information matching. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CAjxVodl_v.

- Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.
- Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep Q-learning with model-based acceleration. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2829–2838. PMLR, 20–22 Jun 2016. URL <http://proceedings.mlr.press/v48/gu16.html>.
- David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS’18, pp. 2455–2467, 2018.
- David Ha and Yujin Tang. Collective intelligence for deep learning: A survey of recent developments. *CoRR*, abs/2111.14377, 2021. URL <https://arxiv.org/abs/2111.14377>.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference on Machine Learning (ICML)*, 2018a.
- Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2018b. URL <https://arxiv.org/abs/1812.05905>.
- Danijar Hafner. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations*, 2022.
- Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565, 2019.
- Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.
- Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=0oabwyZb0u>.
- Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes, 2015.
- Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=o_V-MjyyGV_.

- Nicklas Hansen, Yixin Lin, Hao Su, Xiaolong Wang, Vikash Kumar, and Aravind Rajeswaran. MoDem: Accelerating visual model-based reinforcement learning with demonstrations. *arXiv preprint*, 2022.
- Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2681–2691. PMLR, 09–15 Jun 2019. URL <https://proceedings.mlr.press/v97/hazan19a.html>.
- P. Henderson, W.-D. Chang, F. Shkurti, J. Hansen, D. Meger, and G. Dudek. Benchmark environments for multitask learning in continuous domains. *ICML Lifelong Learning: A Reinforcement Learning Approach Workshop*, 2017.
- Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.11694. URL <https://ojs.aaai.org/index.php/AAAI/article/view/11694>.
- Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Leibo, and Audrunas Gruslys. Deep Q-learning from demonstrations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.11757. URL <https://ojs.aaai.org/index.php/AAAI/article/view/11757>.
- Takuya Hiraoka, Takahisa Imagawa, Taisei Hashimoto, Takashi Onishi, and Yoshimasa Tsuruoka. Dropout Q-functions for doubly efficient reinforcement learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=xCVJMsPv3RT>.
- Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *CoRR*, abs/1112.5745, 2011.
- Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: Variational information maximizing exploration. *Advances in Neural Information Processing Systems*, 29, 2016.
- Maximilian Igl, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson. Transient non-stationarity and generalisation in deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=Qun8fv4qSby>.
- Stephen James, Andrew J. Davison, and Edward Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In *1st Conference on Robot Learning*, 2017.
- Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

- Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In *Advances in Neural Information Processing Systems*, 2021.
- Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In *International Conference on Machine Learning*, pp. 4870–4879. PMLR, 2020.
- Niels Justesen, Ruben Rodriguez Torrado, Philip Bontrager, Ahmed Khalifa, Julian Togelius, and Sebastian Risi. Procedural level generation improves generality of deep reinforcement learning. *CoRR*, abs/1806.10729, 2018.
- Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for Atari. In *International Conference on Learning Representations*, 2020.
- Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto (eds.), *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 651–673. PMLR, PMLR, 29–31 Oct 2018. URL <https://proceedings.mlr.press/v87/kalashnikov18a.html>.
- Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel : Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.
- Taylor Killian, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. URL <https://ojs.aaai.org/index.php/AAAI/article/view/11065>.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.
- Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021.
- Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. BatchBALD: Efficient and diverse batch acquisition for deep Bayesian active learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013. doi: 10.1177/0278364913495721. URL <https://doi.org/10.1177/0278364913495721>.
- Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=GY6-6sTvGaf>.

- Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=68n2s9ZJWF8>.
- Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf>.
- Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018.
- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL <https://proceedings.neurips.cc/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf>.
- Nathan Lambert, Markus Wulfmeier, William F. Whitney, Arunkumar Byravan, Michael Bloesch, Vibhavari Dasagi, Tim Hertweck, and Martin A. Riedmiller. The challenges of exploration for offline reinforcement learning. *CoRR*, 2022.
- Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems 33*, 2020a.
- Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020b.
- Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. URLB: Unsupervised reinforcement learning benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, 2020.
- Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. 2019.
- Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic Q-ensemble. In *5th Annual Conference on Robot Learning*, 2021. URL <https://openreview.net/forum?id=AlJXhEI6J5W>.
- Sergey Levine and Vladlen Koltun. Guided policy search. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1–9, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL <https://proceedings.mlr.press/v28/levine13.html>.

- Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Robotics Res.*, 37(4-5):421–436, 2018.
- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL <https://arxiv.org/abs/2005.01643>.
- Qiyang Li, Aviral Kumar, Ilya Kostrikov, and Sergey Levine. Efficient deep reinforcement learning requires regulating statistical overfitting. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022. URL <https://openreview.net/forum?id=Jwfa-oyQduy>.
- Bryan Lim, Luca Grillotti, Lorenzo Bernasconi, and Antoine Cully. Dynamics-aware quality-diversity for efficient learning of skill repertoires. In *ICRA*, 2022.
- Hao Liu and Pieter Abbeel. APS: Active pretraining with successor features. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6736–6747, 2021a.
- Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. In *Advances in Neural Information Processing Systems*. 2021b.
- Iou-Jen Liu, Unnat Jain, Raymond A Yeh, and Alexander Schwing. Cooperative exploration for multi-agent deep reinforcement learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6826–6836. PMLR, 18–24 Jul 2021.
- Cong Lu, Philip J. Ball, Tim G. J. Rudner, Jack Parker-Holder, Michael A Osborne, and Yee Whye Teh. Challenges and opportunities in offline reinforcement learning from visual observations. In *Workshop on Learning from Diverse, Offline Data*, 2022. URL <https://openreview.net/forum?id=bPOBIKaqLba>.
- Yao Lu, Karol Hausman, Yevgen Chebotar, Mengyuan Yan, Eric Jang, Alexander Herzog, Ted Xiao, Alex Irpan, Mohi Khansari, Dmitry Kalashnikov, and Sergey Levine. AW-Opt: Learning robotic skills with imitation and reinforcement at scale. In *5th Annual Conference on Robot Learning*, 2021.
- Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS ’14, pp. 1077–1084, Richland, SC, 2014. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450327381.
- H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947. doi: 10.1214/aoms/1177730491. URL <https://doi.org/10.1214/aoms/1177730491>.

- Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *International Conference on Learning Representations*, 2021.
- Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. In *Advances in Neural Information Processing Systems*. 2021.
- Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim M. Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, Jiwoo Pak, Andy Tong, Kavya Srinivasa, Will Hang, Emre Tuncer, Quoc V. Le, James Laudon, Richard Ho, Roger Carpenter, and Jeff Dean. A graph placement methodology for fast chip design. *Nature*, 594 7862:207–212, 2021.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dhharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL <http://dx.doi.org/10.1038/nature14236>.
- Aditya Modi, Nan Jiang, Satinder Singh, and Ambuj Tewari. Markov decision processes with continuous side information. In *Proceedings of Algorithmic Learning Theory*, volume 83 of *Proceedings of Machine Learning Research*, pp. 597–618. PMLR, 07–09 Apr 2018. URL <http://proceedings.mlr.press/v83/modi18a.html>.
- Aditya Modi, Jinglin Chen, Akshay Krishnamurthy, Nan Jiang, and Alekh Agarwal. Model-free representation learning and exploration in low-rank MDPs, 2021. URL <https://arxiv.org/abs/2102.07035>.
- Shakir Mohamed and Danilo J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’15, pp. 2125–2133, Cambridge, MA, USA, 2015. MIT Press.
- Ted Moskovitz, Michael Arbel, Ferenc Huszar, and Arthur Gretton. Efficient Wasserstein natural gradients for reinforcement learning. In *International Conference on Learning Representations*, 2021a.
- Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael Jordan. Tactical optimism and pessimism for deep reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12849–12863. Curran Associates, Inc., 2021b. URL <https://proceedings.neurips.cc/paper/2021/file/6abcc8f24321d1eb8c95855eab78ee95-Paper.pdf>.
- Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based RL. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=HyxAfnA5tm>.

- Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299. IEEE, IEEE Press, 2018. doi: 10.1109/ICRA.2018.8463162. URL <https://doi.org/10.1109/ICRA.2018.8463162>.
- Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv*, June 2020.
- George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming*, 14(1):265–294, 1978.
- Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *In Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287. Morgan Kaufmann, 1999.
- D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pp. 55–60 vol.1, 1994.
- Brendan O’Donoghue, Ian Osband, and Catalin Ionescu. Making sense of reinforcement learning and probabilistic inference. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=S1xitgHtvS>.
- Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in Atari games. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS’15*, pp. 2863–2871, Cambridge, MA, USA, 2015. MIT Press.
- OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub W. Pachocki, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *CoRR*, abs/1808.00177, 2018.
- OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik’s cube with a robot hand. *CoRR*, abs/1910.07113, 2019.
- Ian Osband, Benjamin Van Roy, Daniel J. Russo, and Zheng Wen. Deep exploration via randomized value functions. *Journal of Machine Learning Research*, 20(124):1–62, 2019.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models

- to follow instructions with human feedback. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=TG8KACxEON>.
- Aldo Pacchiano, Jack Parker-Holder, Yunhao Tang, Anna Choromanska, Krzysztof Choromanski, and Michael I. Jordan. Learning to score behaviors for guided policy optimization. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.
- Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krähenbühl, Vladlen Koltun, and Dawn Song. Assessing generalization in deep reinforcement learning. *arXiv preprint arXiv:1810.12282*, 2018.
- Jack Parker-Holder, Luke Metz, Cinjon Resnick, Hengyuan Hu, Adam Lerer, Alistair Letcher, Alexander Peysakhovich, Aldo Pacchiano, and Jakob Foerster. Ridge rider: Finding diverse solutions by following eigenvectors of the Hessian. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 753–765. Curran Associates, Inc., 2020.
- Jack Parker-Holder, Aldo Pacchiano, Krzysztof Choromanski, and Stephen Roberts. Effective diversity in population-based reinforcement learning. In *Advances in Neural Information Processing Systems 33*, 2020.
- Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML’17, pp. 2778–2787. JMLR.org, 2017.
- Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5062–5071. PMLR, 09–15 Jun 2019.
- Matt Peng, Banghua Zhu, and Jiantao Jiao. Linear representation meta-reinforcement learning for instant adaptation. *CoRR*, 2021.
- Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation, ICRA*, 2018.
- Silviu Pitis, Elliot Creager, and Animesh Garg. Counterfactual data augmentation using locally factored dynamics. In *Advances in Neural Information Processing Systems*, 2020.
- Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016. ISSN 2296-9144. doi: 10.3389/frobt.2016.00040. URL <https://www.frontiersin.org/article/10.3389/frobt.2016.00040>.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

- Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pp. 1154–1168. PMLR, 2021.
- Roberta Raileanu and Tim Rocktäschel. RIDE: Rewarding impact-driven exploration for procedurally-generated environments. In *International Conference on Learning Representations*, 2020.
- Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in deep reinforcement learning. *CoRR*, abs/2006.12862, 2020.
- Janarthanan Rajendran, Alex Irpan, and Eric Jang. Meta-learning requires meta-augmentation. In *Advances in Neural Information Processing Systems*, 2020.
- Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.
- Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97, pp. 5331–5340. PMLR, 2019.
- Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent, 2022.
- Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning?, 2022. URL <https://arxiv.org/abs/2201.12122>.
- Shideh Rezaeifar, Robert Dadashi, Nino Vieillard, Léonard Hussenot, Olivier Bachem, Olivier Pietquin, and Matthieu Geist. Offline reinforcement learning as anti-exploration. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7): 8106–8114, Jun. 2022. doi: 10.1609/aaai.v36i7.20783. URL <https://ojs.aaai.org/index.php/AAAI/article/view/20783>.
- Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286. PMLR, 2014.
- Sebastian Risi and Julian Togelius. Increasing generality in machine learning through procedural content generation. *Nature Machine Intelligence*, 2, 08 2020. doi: 10.1038/s42256-020-0208-z.
- Stéphane Ross and J. Andrew Bagnell. Agnostic system identification for model-based reinforcement learning. In *Proceedings of the 29th on International Conference on Machine Learning, ICML’12*, pp. 1905–1912, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

- Tim G. J. Rudner, Cong Lu, Michael Osborne, Yarin Gal, and Yee Whye Teh. On pathologies in KL-regularized reinforcement learning from expert demonstrations. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=sS8rRmgAatA>.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. doi: 10.1038/323533a0.
- Stefan Schaal. Learning from demonstration. In M.C. Mozer, M. Jordan, and T. Petsche (eds.), *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996. URL <https://proceedings.neurips.cc/paper/1996/file/68d13cf26c4b4f4f932e3eff990093ba-Paper.pdf>.
- Tom Schaul, André Barreto, John Quan, and Georg Ostrovski. The phenomenon of policy churn, 2022. URL <https://arxiv.org/abs/2206.00730>.
- Jürgen Schmidhuber. Making the world differentiable: On using self-supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical report, 1990.
- Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *CoRR*, abs/1911.08265, 2019.
- Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pp. 8583–8592. PMLR, 2020.
- Younggyo Seo, Kimin Lee, Ignasi Clavera, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel. Trajectory-wise multiple choice learning for dynamics generalization in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.
- Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *International Conference on Learning Representations*, 2020.
- Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International Conference on Machine Learning*, pp. 5779–5788. PMLR, 2019.
- David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016. doi: 10.1038/nature16961.

- Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.
- Yuda Song, Yifei Zhou, Ayush Sekhari, Drew Bagnell, Akshay Krishnamurthy, and Wen Sun. Hybrid RL: Using both offline and online data can make RL efficient. In *International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=yyBis80iUuU>.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL <http://jmlr.org/papers/v15/srivastava14a.html>.
- DJ Strouse, Kate Baumli, David Warde-Farley, Volodymyr Mnih, and Steven Stenberg Hansen. Learning more skills through optimistic exploration. In *International Conference on Learning Representations*, 2022.
- DiJia Su, Jason D. Lee, John Mulvey, and H. Vincent Poor. MURO: Deployment constrained reinforcement learning with model-based uncertainty regularized batch optimization, 2022. URL <https://openreview.net/forum?id=eWNpRVcfzi>.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. 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. URL <https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf>.
- Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bull.*, 2(4):160–163, July 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL <https://doi.org/10.1145/122344.122377>.
- Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.
- Richard S. Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M. Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '11, pp. 761–768, Richland, SC, 2011. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 0982657161.
- Oleg Svidchenko and Aleksei Shpilman. Maximum entropy model-based reinforcement learning. *CoRR*, abs/2112.01195, 2021.
- Ming Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *Mach. Learn.*, 13(1):7–33, Oct 1993. ISSN 0885-6125. doi: 10.1023/A:1022679428250. URL <https://doi.org/10.1023/A:1022679428250>.
- Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michaël Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta

- Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. Open-ended learning leads to generally capable agents. *CoRR*, abs/2107.12808, 2021.
- Sebastian Thrun and A. Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of 4th Connectionist Models Summer School*. Erlbaum Associates, June 1993.
- J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23–30, 2017. doi: 10.1109/IROS.2017.8202133.
- Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13–23, 2021.
- Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. DM Control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: <https://doi.org/10.1016/j.simpa.2020.100022>. URL <https://www.sciencedirect.com/science/article/pii/S2665963820300099>.
- Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30 (1), Mar. 2016. doi: 10.1609/aaai.v30i1.10295. URL <https://ojs.aaai.org/index.php/AAAI/article/view/10295>.
- Hado van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? In *Advances in Neural Information Processing Systems*, pp. 14322–14333, 2019.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.
- Matej Večerík, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv*, July 2017.
- Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *Advances in Neural Information Processing Systems*, volume 31, pp. 5334–5344. Curran Associates, Inc., 2018. URL <https://proceedings.neurips.cc/paper/2018/file/1d94108e907bb8311d8802b48fd54b4a-Paper.pdf>.
- Andrew Wagenmaker and Aldo Pacchiano. Leveraging offline data in online reinforcement learning. *arXiv preprint arXiv:2211.04974*, 2022.

- Zac Wellmer and James T Kwok. Dropout’s dream land: Generalization from learned simulators to reality. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021.
- Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.
- Ali Yahya, Adrian Li, Mrinal Kalakrishnan, Yevgen Chebotar, and Sergey Levine. Collective robot reinforcement learning with distributed asynchronous guided policy search. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 79–86, 2017. doi: 10.1109/IROS.2017.8202141.
- Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 11920–11931, 2021.
- Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don’t change the algorithm, change the data: Exploratory data for offline reinforcement learning. *CoRR*, abs/2201.13425, 2022a.
- Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=_SJ-_yyes8.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, 2020.
- Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. In Nancy M. Amato, Siddhartha S. Srinivasa, Nora Ayanian, and Scott Kuindersma (eds.), *Robotics: Science and Systems XIII*, 2017.
- Tom Zahavy, Yannick Schroecker, Feryal Behbahani, Kate Baumli, Sebastian Flennerhag, Shaobo Hou, and Satinder Singh. Discovering policies with DOMiNO: Diversity optimization maintaining near optimality. *arXiv preprint arXiv:2205.13521*, 2022.
- Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block MDPs. In *International Conference on Learning Representations*, 2021a. URL <https://openreview.net/forum?id=fm00I2a3tQP>.
- Chuheng Zhang, Yuanying Cai, Longbo Huang, and Jian Li. Exploration by maximizing renyi entropy for reward-free RL framework. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 10859–10867. AAAI Press, 2021b.

- Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=-Y34L45JR6z>.
- 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*, 2021c.
- Wenxuan Zhou, Lerrel Pinto, and Abhinav Gupta. Environment probing interaction policies. In *International Conference on Learning Representations*, 2019.
- Zihan Zhou, Wei Fu, Bingliang Zhang, and Yi Wu. Continuously discovering novel strategies via reward-switching policy optimization. In *International Conference on Learning Representations*, 2022.
- 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. In *International Conference on Learning Representations*, 2020.
- Luisa M Zintgraf, Leo Feng, Cong Lu, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and Shimon Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 12991–13001. PMLR, 18–24 Jul 2021.

6

Conclusion

In this thesis, we have sought to improve sample-efficiency in deep reinforcement learning by separating the problem into three sub-areas.

In Chapter 3, we sought to maximally utilize offline data during pre-training. In particular, in Section 3.1 we sought to achieve this by investigating the key design choices that impact agent performance when training inside a world model of the environment. We discovered novel insights around rollout horizon, penalty calibration, and made general recommendations that any practitioner can leverage when training such agents going forward.

In Chapter 4, we investigated the problem of agent learning in the online setting. In particular, in Section 4.1, we showed that reframing model-based reinforcement learning as an active learning problem resulted in agents that were significantly more sample efficient than canonical Dyna-style agents. This framework also allowed us to develop a principled information-theoretic exploration bonus, and we also introduced multi-armed bandits as a way to balance the degree of agent exploration and exploitation. We then showed in Section 4.2 that image augmentations, a common approach to improving pixel-based off-policy algorithms, are in fact an approach to help adaptability by preventing a phenomenon we described as “self-overfitting”, and identified a novel “visual deadly triad” that arises when combining TD-learning with visual environments. Furthermore, we noticed that once useful values were learned in

the critic networks, turning off augmentations in fact improved performance. Given these insights, we developed a new network layer for convolutional neural networks called LIX, and adaptively controlled its regularization during training through a gradient divergence metric. Through this, we developed a simple model-free method that attained state-of-the-art performance on several competitive domains, and notably outperformed model-based approaches on the Atari 100k benchmark with significantly fewer design choices.

Finally, in Chapter 5 we investigated the problem of leveraging offline data for online deployment. First, in Section 5.1 we investigated the impact of deployment-time covariate shift in the offline RL setting. We observed a significant decrease in performance when a dynamics covariate shift occurs during deployment, and therefore proposed a novel data augmentation technique called “DAS” to imbue agents with robustness to such test-time changes. To further improve adaptability online, we treated the changed dynamics as a POMDP problem, and therefore also conditioned our learned policies on a latent dynamics vector, which was learned during offline training. We found that combining “DAS” with a learned latent dynamics vector produced agents that were not only significantly more robust to deployment-time changes in dynamics, but were also able to respond and adapt to non-stationary environments where dynamics may change during the rollout. Then in Section 5.2, we investigated the problem of learning generalist agents using fixed offline datasets that could adapt to any downstream task. To this end, we proposed a novel model-based approach that leveraged a population of agents trained to maximize information gain and diversity simultaneously. With this approach, we were able to obtain significantly better data for world model training, and thus developed a strategy to train better generalist agents that adapt zero-shot to unseen tasks. Finally, in Section 5.3 we investigated and addressed an important open problem in deep RL, which is understanding why deep off-policy methods fail to successfully leverage offline datasets. We identified value divergence as being the root cause for this issue, and proposed a novel use for LayerNorm to address this, which has the desirable properties of not inhibiting exploration, unlike offline RL

approaches. We also identified the importance of constantly sampling the offline data in each batch, and proposed “symmetric sampling” as a general strategy to incorporate offline data into online training. This allows us to turn off-policy methods into approaches that can readily incorporate any form of offline data.

6.1 Future Work

Developing Complex and Realistic Domains for RL As stated in the introduction, a grand challenge is to create agents that can be deployed in any general setting. A key drawback that current deep RL approaches have is that their progress is evaluated on common benchmarks that are often significantly less complex than actual domains we may wish to consider. Moreover, in some domains we may already achieve strong performance using simpler approaches, such as behavior cloning (Emmons et al., 2022), which calls into question when RL is necessary.

Therefore there is a need to develop complex domains for which RL can be shown to be useful. Moreover, these domains must strongly mirror actual useful downstream tasks that we may wish to solve. Indeed, recent progress in language modeling (Radford et al., 2019; Devlin et al., 2019) and generative modeling (Ramesh et al., 2022) has been driven by having access to a lucid and clear set of canonical benchmarks and datasets that are directly related to relevant downstream use-cases.

There has been progress towards developing such benchmarks that move away from the canonical RL tasks, such as Habitat 2.0 (Szot et al., 2021), which aims to provide a significantly more challenging and representative training environments for the goal of learning *embodied agents* (Smith & Gasser, 2005). Furthermore, to aid generalization, such environments have been combined with procedural generation (Deitke et al., 2022). However, currently the best performing methods trained on these environments do not use RL. It might be the case that existing deep RL methods are still reliant on large sample sizes, as recent work shows positive results for learning embodied agents with RL using high-throughput environments by simplifying physics calculations and massive deployment parallelism (Truong et al., 2023; Berges et al., 2023). One avenue towards tractable end-to-end RL

on real-world environments could be a curriculum-based approach extending the Sim2Real paradigm (Sadeghi & Levine, 2016), whereby we begin training on quicker simplified diverse environments, and then progressively fine-tuning on computationally intensive higher fidelity environments until we achieve deployment in the real world.

Other proposed use cases for RL have also included interacting with the internet (Shi et al., 2017; Humphreys et al., 2022), which has strong links with popular recent use cases of combining RL with large language models, Reinforcement Learning from Human Feedback (RLHF) (Ziegler et al., 2019). In the current formulation of RLHF, RL is performed in a closed-loop fashion and resembles offline model-based RL, whereby an agent learns a reward function of preferences from a fixed dataset, and attempts to optimize its actions (e.g., text outputs) against it through a learned generative model. It is therefore certainly plausible to consider making such a system open-loop, and imbuing agents with the ability to also take actions to navigate the internet to acquire their own datasets, and explore in a way that maximizes human preferences. Moreover, with the rise in multi-modal VLM models (Alayrac et al., 2022; Brohan et al., 2023), it should be feasible to imbue agents with the ability to interact with the internet using more than just text (Nakano et al., 2021), and instead learn to browse the internet freely using actions such as clicking analogous to humans. In order to feasibly develop such agents, additional research is required into developing potentially internet-like multimodal environments (such as extending MiniWOB++ (Liu et al., 2018)), whereby the primary challenge is gathering relevant data to optimize a provided reward function such as in Section 4.1, or perhaps some unknown downstream language modeling task, therefore relying on unsupervised RL methodologies such as in Section 5.2.

High Fidelity World Models Much of the work discussed in this thesis relies on the having a world model of the environment. However, as we venture towards agents that can learn on increasingly realistic and complex domains, we must learn world models that are equally capable of modeling their corresponding dynamics

function. A key difference between model-free and model-based approaches is the former only needs to learn the mapping $\mathcal{S} \rightarrow \mathcal{A}$ whereas model-based methods must model the significantly more complex $\mathcal{S}, \mathcal{A} \rightarrow \mathcal{S}', \mathcal{R}$.

Existing work has already shown promise scaling towards pixel-based domains (Hafner et al., 2021), as we have also demonstrated in Section 5.2. However, these domains are relatively trivial compared to the complexity and diversity of environments encountered by, for instance, embodied agents mentioned earlier. Therefore we need to continue developing modeling approaches that can cope with the increasingly complex domains that can be tackled by model-based RL.

Recent work in Neural Radiance Fields (NeRFs) (Mildenhall et al., 2020) presents an interesting direction for creating higher fidelity world models. NeRFs learn a mapping from camera position and orientation to RGB images from a sparse set of inputs, and represent state-of-the-art for synthesizing novel views of scenes. Recent work has shown that the representations learned by NeRFs are useful for downstream RL tasks (Driess et al., 2022), however the learned NeRF was not used for planning, resulting in an approach that closely mirrors that seen in Ota et al. (2020); Schwarzer et al. (2021), whereby modeling properties of the environment is treated as an auxiliary loss. Initial work has shown promise for learning standard route planning in NeRFs (Adamkiewicz et al., 2022), however it is plausible to further imbue these with additional dynamics awareness and model transition functions that maintain incredibly high parity with the real world, and providing a route to modeling complex realistic scenes for the purposes of policy optimization.

We also note Diffusion Processes (Sohl-Dickstein et al., 2015; Ho et al., 2020) provide an interesting avenue for learning high fidelity world models, as these have shown the capacity to learn high quality generative models across a number of modalities (Ramesh et al., 2022; Ho et al., 2022; Poole et al., 2023). Recent work has shown that diffusion models can be applied to decision making problems by learning entire trajectory distributions, and optimizing behaviors through a classifier-guidance inspired reward model mechanism (Janner et al., 2022). Moreover, it has been shown that diffusion models can be integrated with canonical RL by implicitly

learn high-fidelity transition dynamics despite only training on joint distribution experience tuples, providing evidence that diffusion models have the potential to learn a dynamics function (Lu et al., 2023). It might be possible to train these models using classifier-free guidance (Ho & Salimans, 2021), such that a conditional (dynamics) model (i.e., learn $p(s', r|s, a)$) and full joint distribution (as in Lu et al. (2023), learning $p(s', r, s, a)$) are learned to provide high quality samples across a number of different domains and modalities.

The Growing Relevance of Continual Learning A key difference between RL and other disciplines of machine learning is the increased non-stationarity of the training process itself. This non-stationarity can manifest in a number of different ways, such as a non-static dataset due to acquiring new data points during rollouts, or TD-learning causing growing values in critic functions, thus resulting in non-stationary dependent variables (i.e., target values), even in fixed data settings such as offline RL.

Not accounting for this non-stationarity has been shown to significantly reduce the capacity for agents to adapt to new information; indeed in Section 4.2 we attempt to mitigate this capacity reduction by preventing self-overfitting. While prior work has also investigated this problem on simple settings, such as policy resets (Nikishin et al., 2022), weight resets (Sokar et al., 2023), and implicit regularization through self-distillation (Kumar et al., 2021), more work needs to be done to understand if such approaches generalize to increasingly complex domains, such as where non-stationarity may exist over the task distribution itself.

One potential avenue to resolving this issue is framing RL as a continual learning problem (Abel et al., 2023). Continual learning aims to learn a model that can handle non-stationarity in task distribution without “forgetting” previously learned information about those tasks. In this case, we can view RL as a continual learning task in two different ways. For instance, a single task may require multiple “phases” to be learned before in order to successfully achieve optimal behavior, such as learning to stand, then walk then run in Humanoid domains in MuJoCo, or multiple distinct

sub-tasks that need to be achieved in hard exploration tasks, such as Montezuma’s Revenge. Secondly, with a growing interest in learning generalist agents (Reed et al., 2022) and open-endedness (Bauer et al., 2023), we can view the task distributions themselves as having non-stationarity, and agents may be expected to recall how to optimally perform some earlier task over its entire lifetime (in the same way humans can retain knowledge on how to ride a bike after learning to drive).

It is well known that the aforementioned examples already present issues in deep reinforcement learning (Kirkpatrick et al., 2017; Fedus et al., 2020), and strides towards learning more complex agents will exacerbate this issue. Existing work has shown promise in borrowing approaches in continual learning, such as learning “heads” per task (Kessler et al., 2022), or knowledge distillation (Wolczyk et al., 2022), but questions remain over whether these approaches can scale to complex domains. Therefore given the growing complexity of tasks and use-cases for RL, leveraging insights from continual learning may represent a promising approach towards resolving such issues when they arise.

Chapter Bibliography

David Abel, André Barreto, Benjamin Van Roy, Doina Precup, Hado van Hasselt, and Satinder Singh. A definition of continual reinforcement learning. *arXiv preprint arXiv:2307.11046*, 2023.

Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Jakob Bauer, Kate Baumli, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, Vibhavari Dasagi, Lucy Gonzalez, Karol Gregor, Edward Hughes, Sheleem Kashem, Maria Loks-Thompson, Hannah Openshaw, Jack Parker-Holder, Shreya Pathak, Nicolas Perez-Nieves, Nemanja Rakicevic, Tim Rocktäschel, Yannick Schroecker, Satinder Singh, Jakub Sygnowski, Karl Tuyls, Sarah York, Alexander Zacherl, and Lei M Zhang. Human-timescale adaptation in an open-ended task space. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume

- 202 of *Proceedings of Machine Learning Research*, pp. 1887–1935. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/bauer23a.html>.
- Vincent-Pierre Berges, Andrew Szot, Devendra Singh Chaplot, Aaron Gokaslan, Roozbeh Mottaghi, Dhruv Batra, and Eric Undersander. Galactic: Scaling end-to-end reinforcement learning for rearrangement at 100k steps-per-second. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13767–13777, 2023.
- Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.
- Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-scale embodied AI using procedural generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=4-bV1bi74M>.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL <https://aclanthology.org/N19-1423>.
- Danny Driess, Ingmar Schubert, Pete Florence, Yunzhu Li, and Marc Toussaint. Reinforcement learning with neural radiance fields. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=3SLW-YIw7tX>.
- Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. RvS: What is essential for offline RL via supervised learning? In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=S874XAIpkR->.
- William Fedus, Dibya Ghosh, John D Martin, Marc G Bellemare, Yoshua Bengio, and Hugo Larochelle. On catastrophic interference in Atari 2600 games. *arXiv preprint arXiv:2002.12499*, 2020.
- Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=0oabwyZbOu>.
- Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL <https://openreview.net/forum?id=qw8AKxfYbI>.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

- Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv:2204.03458*, 2022.
- Peter C Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy Lillicrap. A data-driven approach for learning to control computers. In *International Conference on Machine Learning*, pp. 9466–9482. PMLR, 2022.
- Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/janner22a.html>.
- Samuel Kessler, Jack Parker-Holder, Philip Ball, Stefan Zohren, and Stephen J Roberts. Same state, different task: Continual reinforcement learning without interference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7143–7151, 2022.
- 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.
- Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021.
- Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations (ICLR)*, 2018. URL <https://arxiv.org/abs/1802.08802>.
- Cong Lu, Philip J. Ball, Yee-Whye Teh, and Jack Parker-Holder. Synthetic experience replay. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=6jNQ1AY1Uf>.
- Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.
- Evgenii Nikishin, Max Schwarzer, Pierluca D’Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International Conference on Machine Learning*, pp. 16828–16847. PMLR, 2022.

- Kei Ota, Tomoaki Oiki, Devesh Jha, Toshisada Mariyama, and Daniel Nikovski. Can increasing input dimensionality improve deep reinforcement learning? In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7424–7433. PMLR, 13–18 Jul 2020. URL <https://proceedings.mlr.press/v119/ota20a.html>.
- Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2D diffusion. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=FjNys5c7VyY>.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.
- Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maroon, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL <https://openreview.net/forum?id=1ikK0kHjvj>. Featured Certification, Outstanding Certification.
- Fereshteh Sadeghi and Sergey Levine. CAD2RL: Real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.
- Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=uCQfPZwRaUu>.
- Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. World of bits: An open-domain platform for web-based agents. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3135–3144. PMLR, 06–11 Aug 2017. URL <https://proceedings.mlr.press/v70/shi17a.html>.
- Linda Smith and Michael Gasser. The development of embodied cognition: Six lessons from babies. *Artificial Life*, 11(1-2):13–29, 2005.
- Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.
- Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume

202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/sokar23a.html>.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 251–266. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/021bbc7ee20b71134d53e20206bd6feb-Paper.pdf.

Joanne Truong, Max Rudolph, Naoki Harrison Yokoyama, Sonia Chernova, Dhruv Batra, and Akshara Rai. Rethinking Sim2Real: Lower fidelity simulation leads to higher Sim2Real transfer in navigation. In *Conference on Robot Learning*, pp. 859–870. PMLR, 2023.

Maciej Wolczyk, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Disentangling transfer in continual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:6304–6317, 2022.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Appendices



Revisiting Design Choices in Offline Model-Based RL: Appendix

A.1 Calibration

A.1.1 Choice of calibration metrics

We consider both the Spearman rank (ρ) correlation and Pearson bivariate (r) correlation. We believe that the former better represents the actual statistical power of the metric compared to the true distributional shift value, as it is robust to outliers and isn't impacted by distributional shape (i.e., skewness, kurtosis). After all, we do not know if some 'true' $|G_M^\pi(s, a)|$ is even linearly correlated with the MSE values that we report, so naïvely comparing based on bivariate correlation may result in incorrect assessment of penalty efficacy. However, we do also include the Pearson bivariate correlation to gain insight into how the penalty distribution shape changes with design choices. For instance, consider two metrics that have identical Spearman coefficients, but vastly different Pearson coefficients—this implies they have significantly different distributional shapes whilst having the same statistical ranking power. The two correlation coefficients have the further advantage that they are unaffected by the scale of the uncertainty penalty, which can vary widely. Furthermore, algorithms such as MOPO and MOREL will often scale the penalty by some coefficient λ and thus the raw unscaled value is hard to interpret.

A.1.2 The use of MSE as the Ground Truth for Deterministic Dynamics

Following Yu et al. (2020), it is possible upper bound the expected performance η_M of a policy π in the true MDP M under training in a world model MDP $\hat{M}$ as follows:

$$\eta_M(\pi) \geq \mathbb{E}_{(s,a) \sim \rho_P^\pi} [R(s, a) - \gamma |G_M^\pi(s, a)|] \quad (\text{A.1})$$

where $R(\cdot, \cdot)$ is the reward function, ρ_P^π represents transitioning under the world model dynamics $\hat{P}$ and policy π . The quantity $|G_M^\pi(s, a)|$ can be upper-bounded by an integral probability metric (IPM):

$$|G_M^\pi(s, a)| \leq \sup_{f \in \mathcal{F}} |\mathbb{E}_{s' \sim \hat{P}(s, a)}[f(s')] - \mathbb{E}_{s' \sim P(s, a)}[f(s')]| =: d_{\mathcal{F}}(\hat{P}(s, a), P(s, a)) \quad (\text{A.2})$$

where $\mathcal{F}$ is some set of functions mapping $\mathcal{S}$ to $\mathbb{R}$, and P is the dynamics under the true MDP M . As noted in Yu et al. (2020), making assumptions over the functional form of $\mathcal{F}$ induces different distance measures. Restricting $\mathcal{F}$ to the set of 1-Lipschitz functions results in an IPM with the following form:

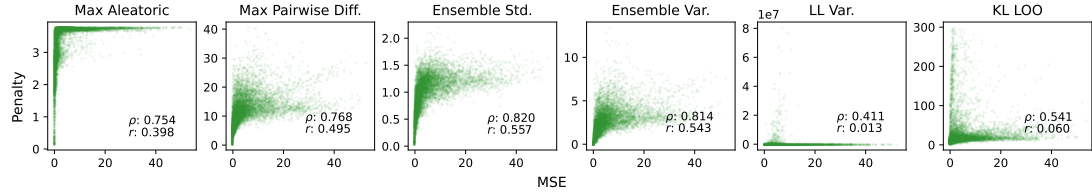
$$|G_M^\pi(s, a)| \leq cW_1(\hat{P}(s, a), P(s, a)) \quad (\text{A.3})$$

which is the 1-Wasserstein distance, where the constant c is the Lipschitz constant of the value function V_M^π with respect to a norm $\|\cdot\|$. Recalling that the environments we evaluate have deterministic dynamics (Todorov et al., 2012), this means the dynamics distributions P and $\hat{P}$ in Eq. A.3 are Dirac delta functions. In this case, the 1-Wasserstein distance simply reduces to the 2-norm between some ‘true dynamics’ $T(s, a)$ and the ‘estimated dynamics’ $\hat{T}(s, a)$. This justifies the use of MSE between the oracle dynamics (as detailed in Sec. 3.1.4 and App. A.4.1) and the world model dynamics as the *ground truth* measure under which we assess calibration.

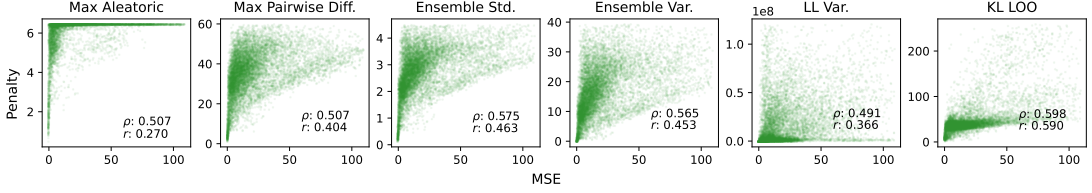
A.1.3 Offline Dataset Transfer Calibration

We present the full calibration scatter plots described in Sec. 3.1.4. Concretely, we plot penalty values on the y-axis, and ground-truth MSE on the x-axis. First, we present the transfer performance of training sets onto offline datasets. Then, we present the results for all training datasets under the True Model-Based experiment under the adversarial policies.

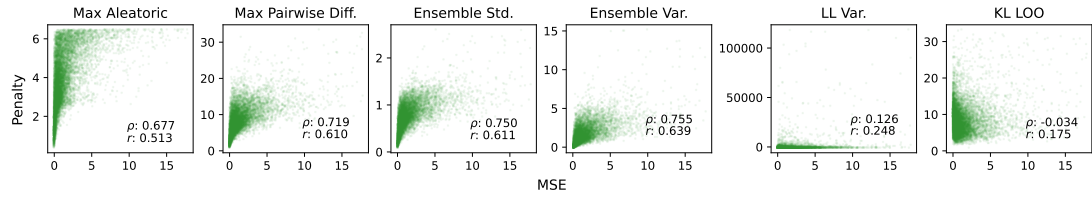
A.1.4 HalfCheetah



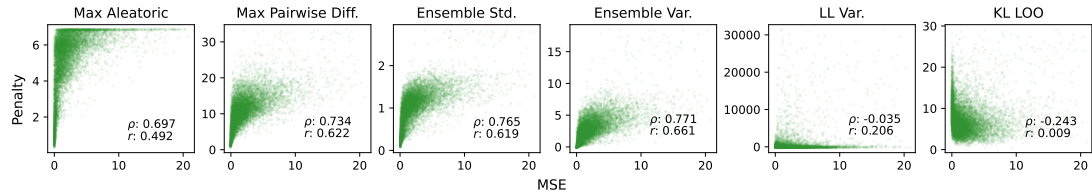
(a) Random transferred to Expert



(b) Medium transferred to Expert



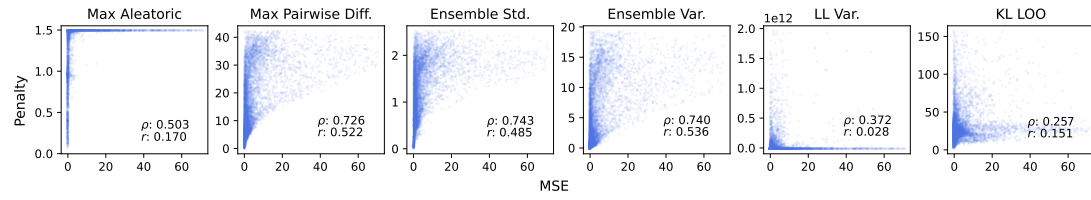
(c) Medium transferred to Random



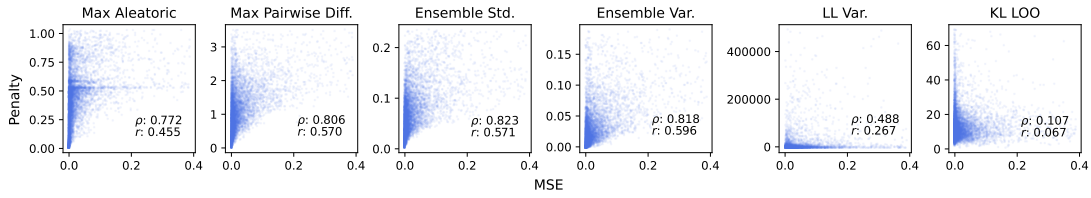
(d) Expert transferred to Random

Figure A.1: Scatter Plots showing HalfCheetah D4RL transfer tasks.

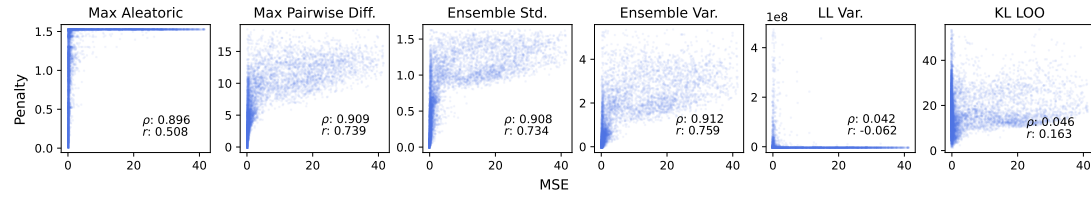
A.1.5 Hopper



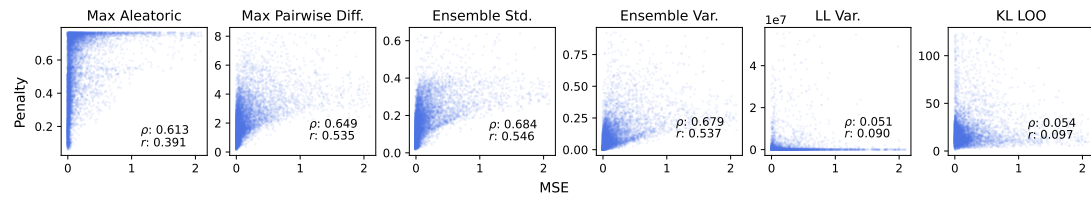
(a) Random transferred to Expert



(b) Medium transferred to Expert



(c) Medium transferred to Random

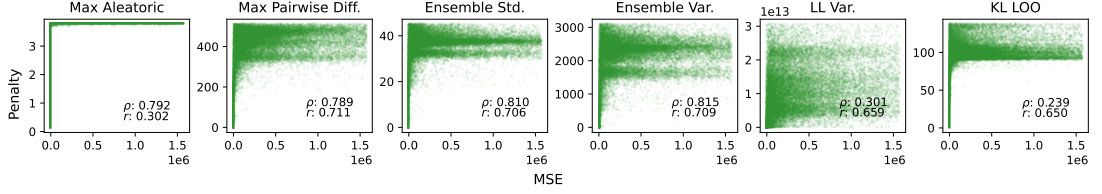


(d) Expert transferred to Random

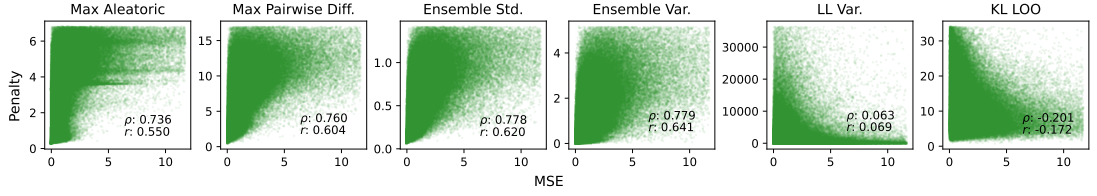
Figure A.2: Scatter Plots showing Hopper D4RL transfer tasks.

A.1.6 True Model-Based Error Calibration

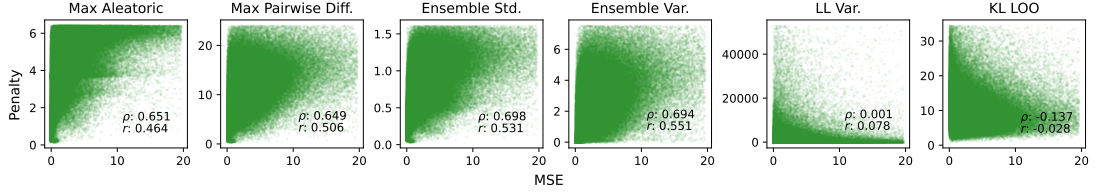
HalfCheetah



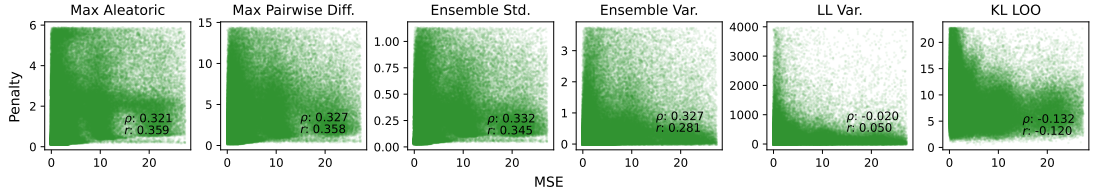
(a) Random



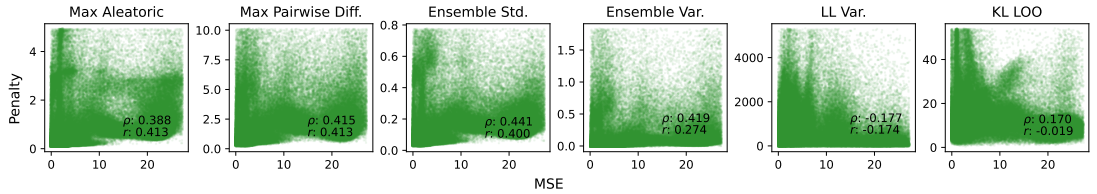
(b) Mixed



(c) Medium



(d) Medium Expert



(e) Expert

Figure A.3: Scatter Plots showing HalfCheetah D4RL true model-based error calibration.

Hopper

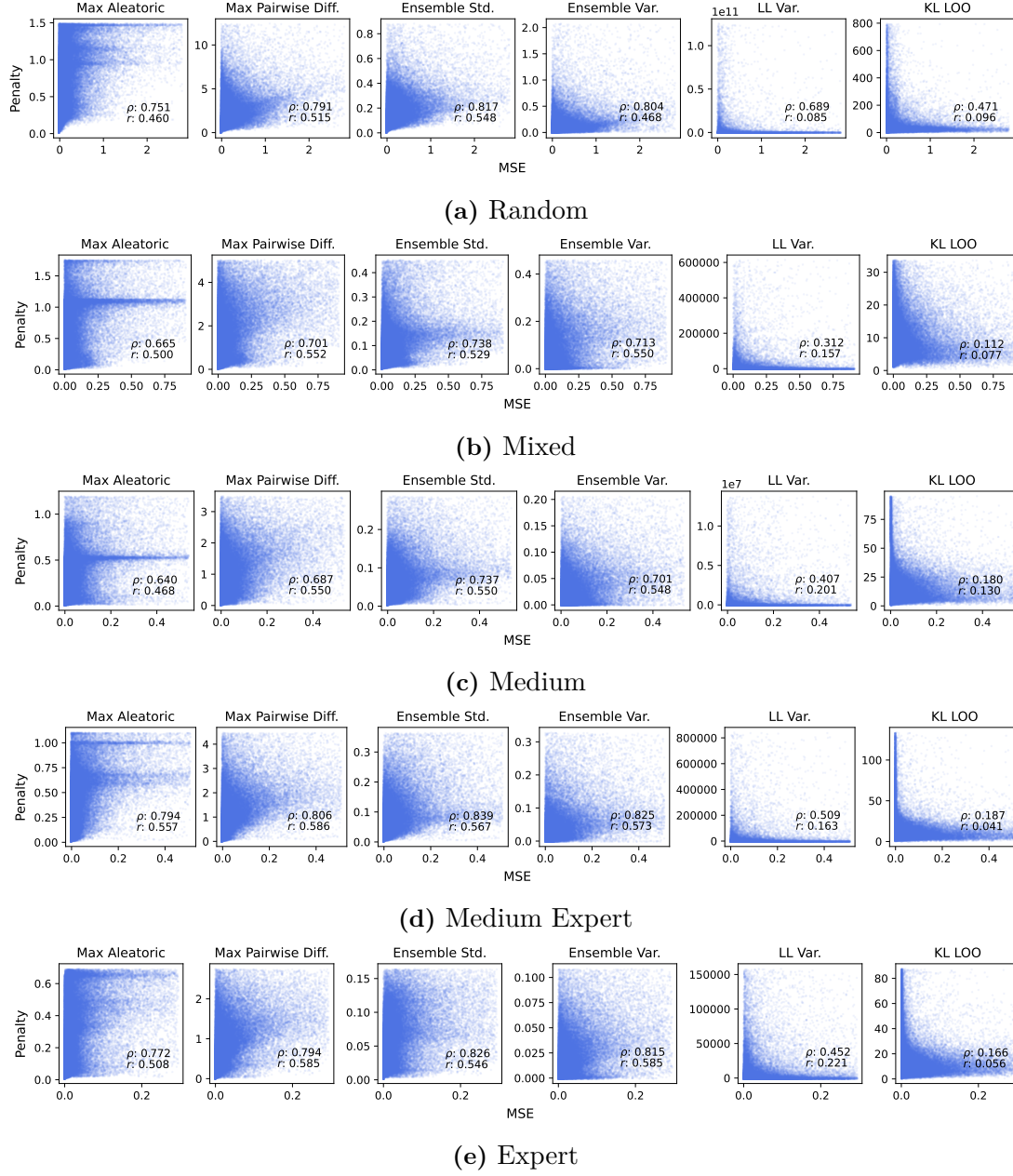


Figure A.4: Scatter Plots showing Hopper D4RL true model-based error calibration.

A.1.7 Additional Log Probability Correlation Analysis

Table A.1 shows correlation between reward penalties and the negative log-likelihood of the true data under the model in the Transfer experiments.

Table A.1: Correlation statistics of penalties against model negative log-likelihood of the true data, averaged over all datasets (i.e., Random through to Expert) showing ± 1 SD over 12 seeds. The best in each column is **bolded**.

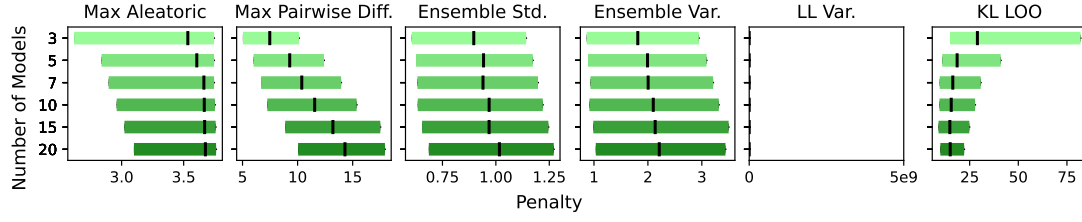
Penalty	Transfer			
	HalfCheetah		Hopper	
	ρ	r	ρ	r
Max Aleatoric	0.87 $\pm$ 0.00	0.82 $\pm$ 0.01	0.81 $\pm$ 0.01	0.57 $\pm$ 0.01
Max Pairwise Diff.	0.79 $\pm$ 0.01	0.62 $\pm$ 0.00	0.79 $\pm$ 0.01	0.51 $\pm$ 0.00
Ens. Std.	0.93 $\pm$ 0.00	0.86 $\pm$ 0.01	0.89 $\pm$ 0.01	0.61 $\pm$ 0.01
Ens. Var.	0.90 $\pm$ 0.01	0.74 $\pm$ 0.01	0.82 $\pm$ 0.00	0.59 $\pm$ 0.00
LL Var.	0.04 $\pm$ 0.07	0.07 $\pm$ 0.03	0.25 $\pm$ 0.03	0.10 $\pm$ 0.01
LOO KL	-0.04 $\pm$ 0.06	-0.02 $\pm$ 0.04	0.08 $\pm$ 0.03	0.05 $\pm$ 0.01

A.2 Full Results Increasing Models

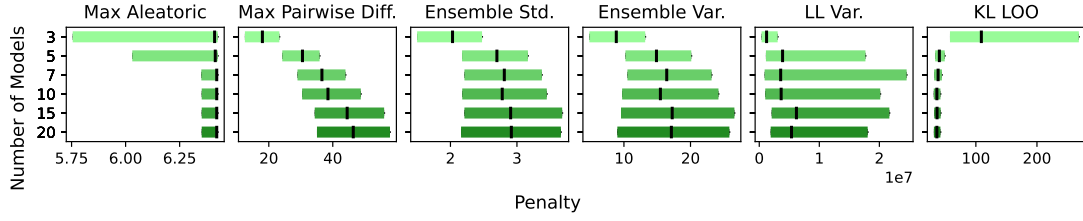
A.2.1 Penalty Distribution

In this section we provide the full set of results showing the impact of increasing model count on the distribution quantile statistics as introduced in Sec. 3.1.5. We show inter-quartile range and the median (the latter being denoted by a black vertical line) of each penalty as a function of increasing model number across all training domains and test settings. First, we present the transfer performance of all training sets onto all offline datasets. Then, we present the results for all training datasets under the True Model-Based experiment under the adversarial policies.

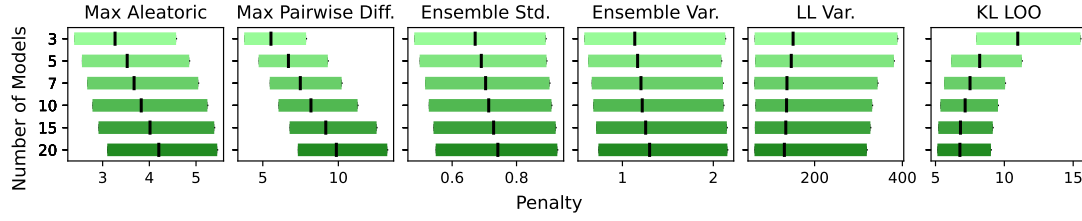
A.2.2 Offline Dataset Transfer Distribution



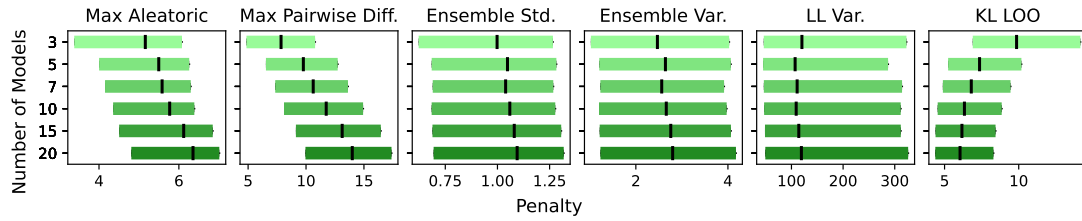
(a) Random transferred to Expert



(b) Medium transferred to Expert

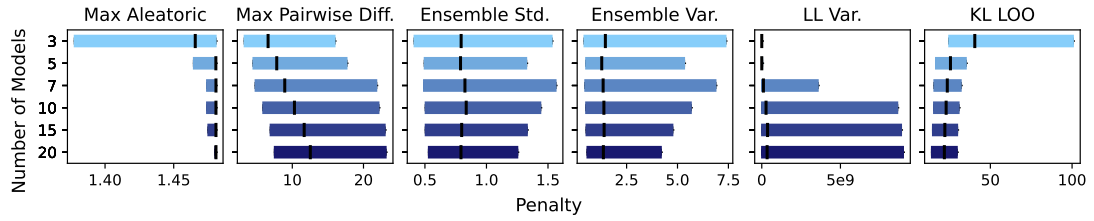


(c) Medium transferred to Random

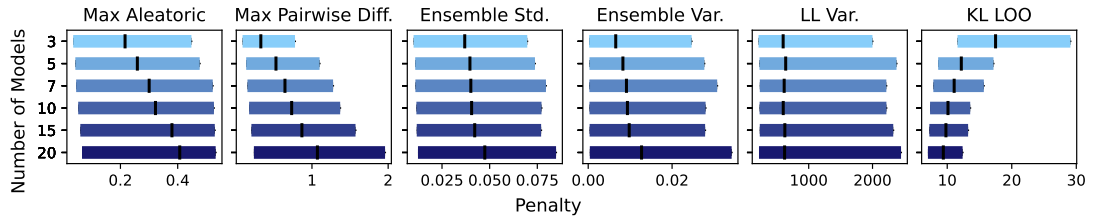


(d) Expert transferred to Random

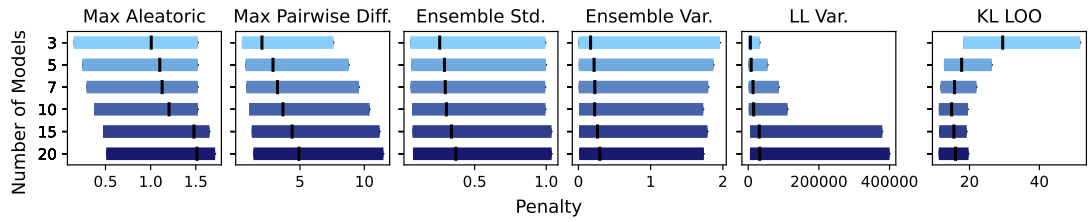
Figure A.5: Box Plots showing HalfCheetah D4RL transfer tasks.



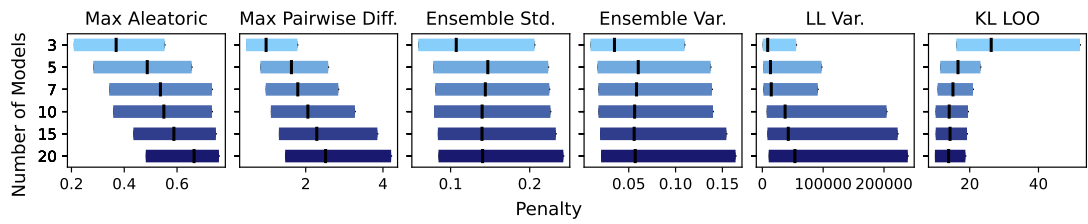
(a) Random transferred to Expert



(b) Medium transferred to Expert



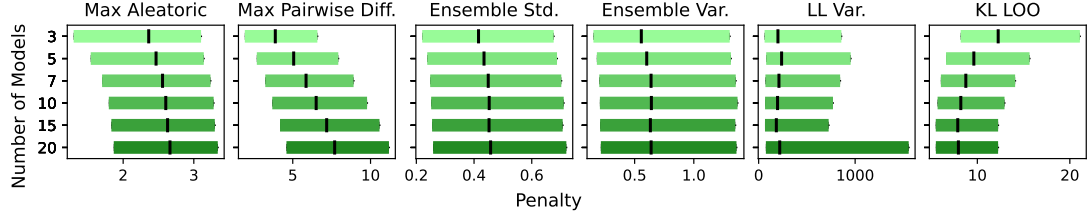
(c) Medium transferred to Random



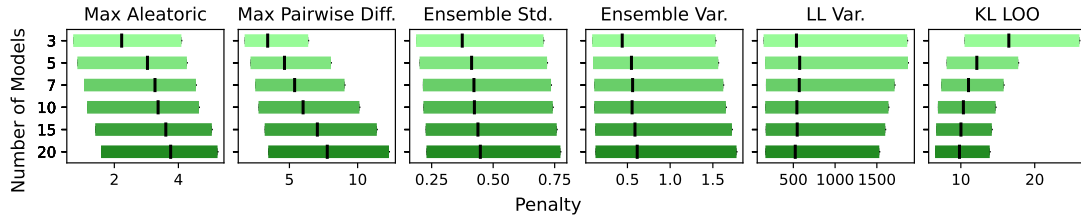
(d) Expert transferred to Random

Figure A.6: Box Plots showing Hopper D4RL transfer tasks.

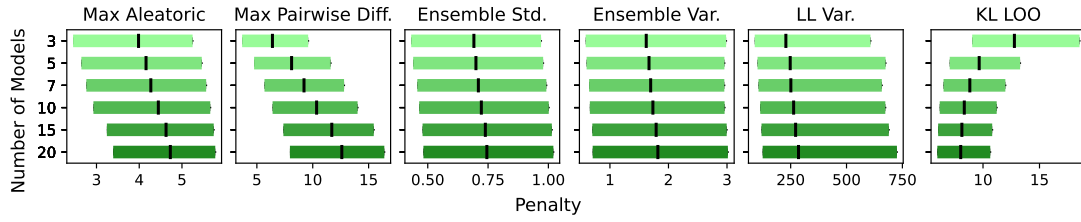
A.2.3 True Model-Based Error Distribution



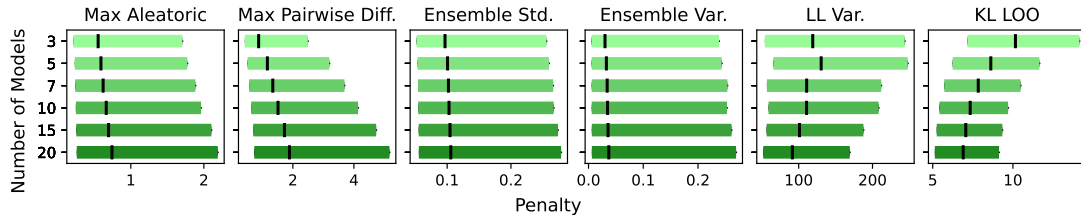
(a) Random



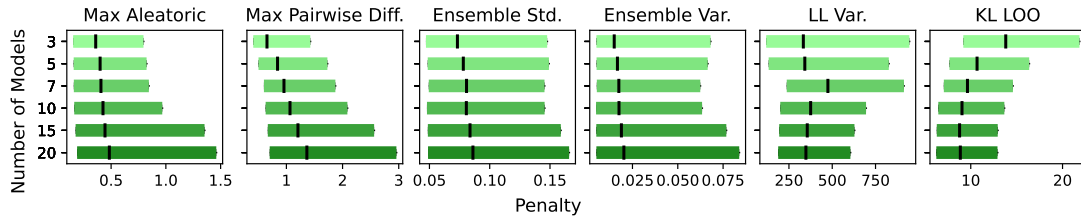
(b) Mixed



(c) Medium

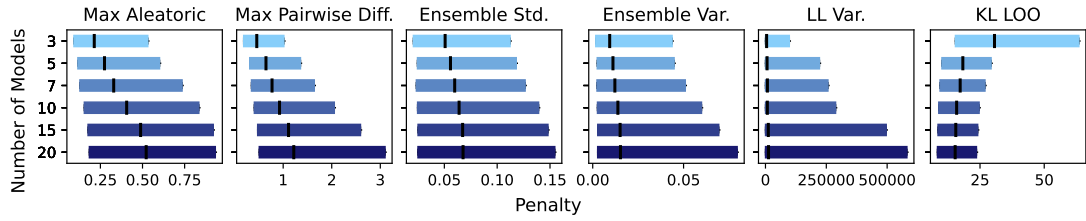


(d) Medium Expert

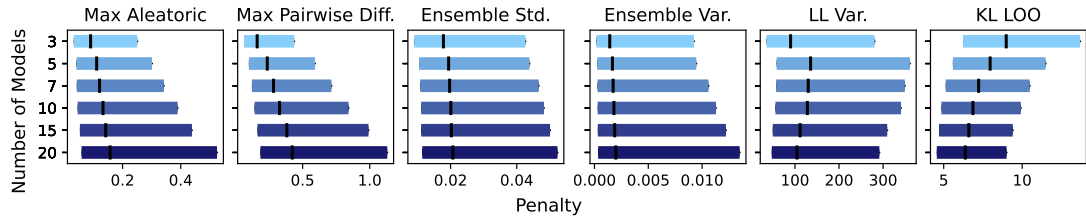


(e) Expert

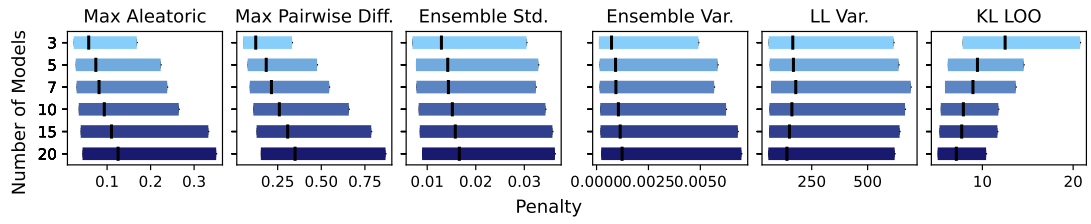
Figure A.7: Boxplots showing HalfCheetah D4RL true model-based error penalty distributions.



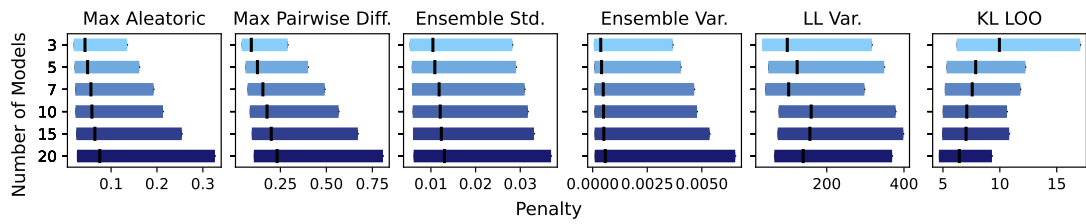
(a) Random



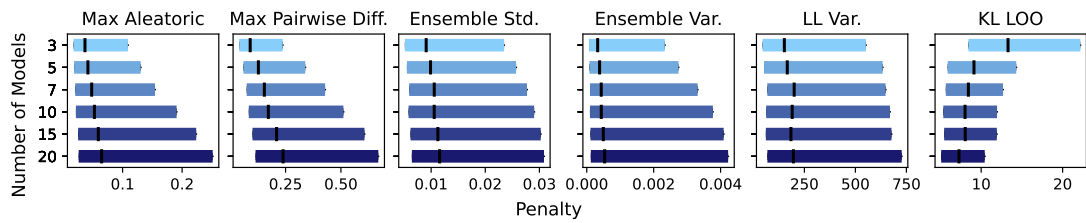
(b) Mixed



(c) Medium



(d) Medium Expert



(e) Expert

Figure A.8: Boxplots showing Hopper D4RL true model-based error penalty distributions.

A.2.4 Penalty Performance

In this section, we provide the full set of results showing the impact of increasing model count on the correlation statistics of each penalty, as described in Sec. 3.1.5. We show the Spearman and Pearson correlation between penalty and ground truth MSE for all training datasets. First, we present the transfer performance of all training sets onto all offline datasets. Then, we present the results for all training datasets under the True Model-Based experiment under the adversarial policies.

A.2.5 HalfCheetah D4RL: Transfer

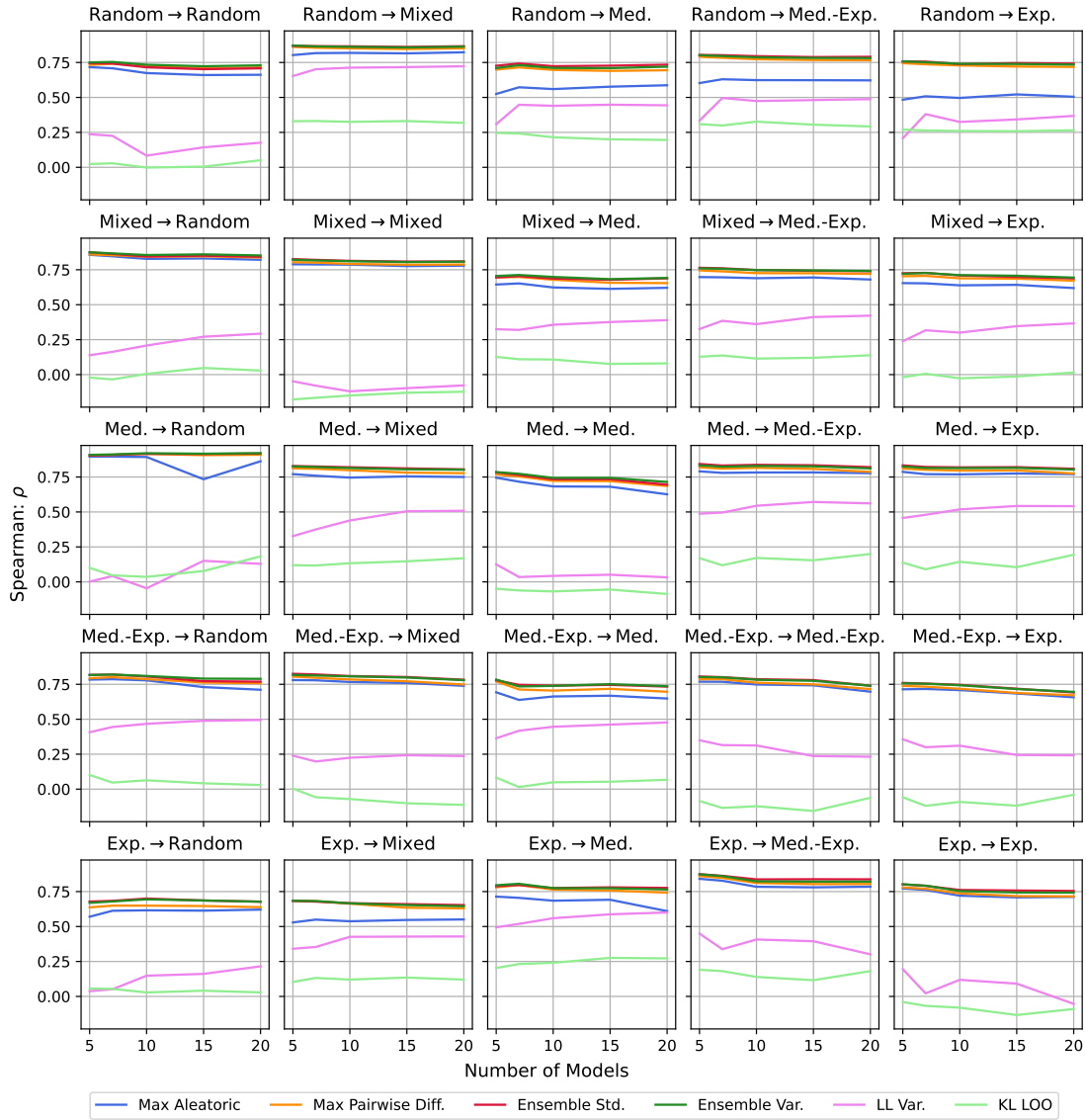


Figure A.9: HalfCheetah Spearman Statistics

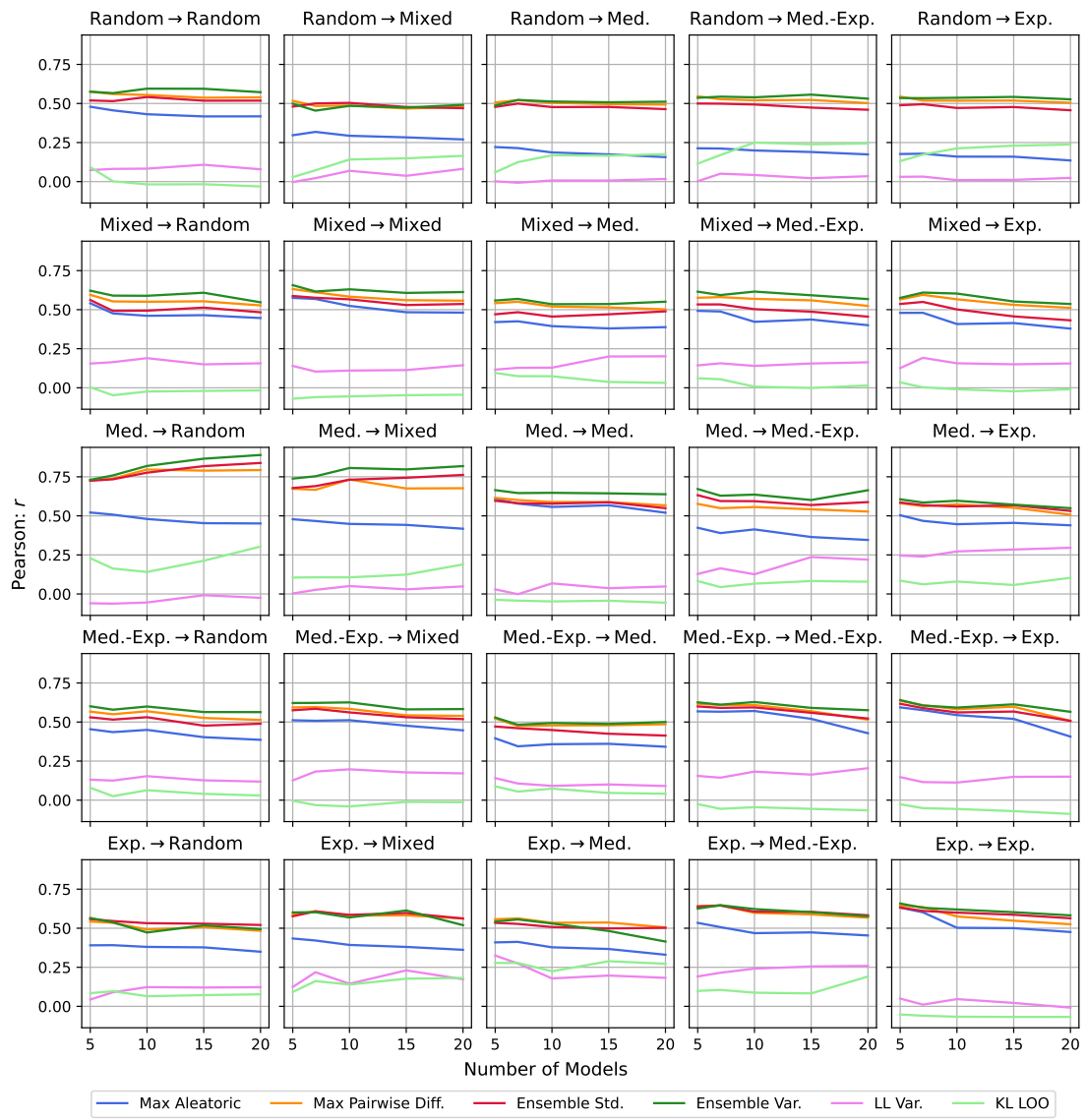


Figure A.10: HalfCheetah Pearson Statistics

A.2.6 Hopper D4RL: Transfer

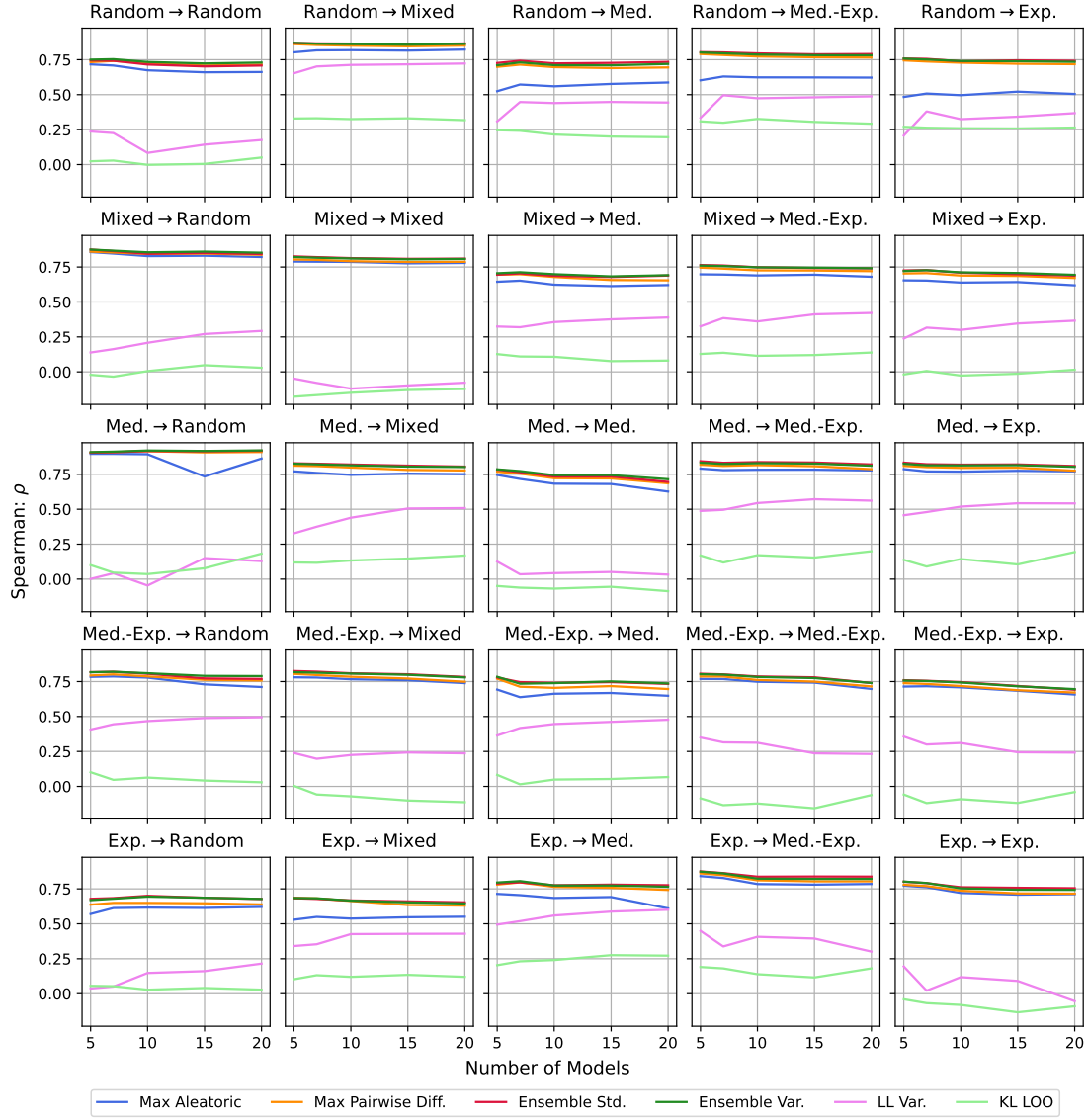


Figure A.11: Hopper Spearman Statistics

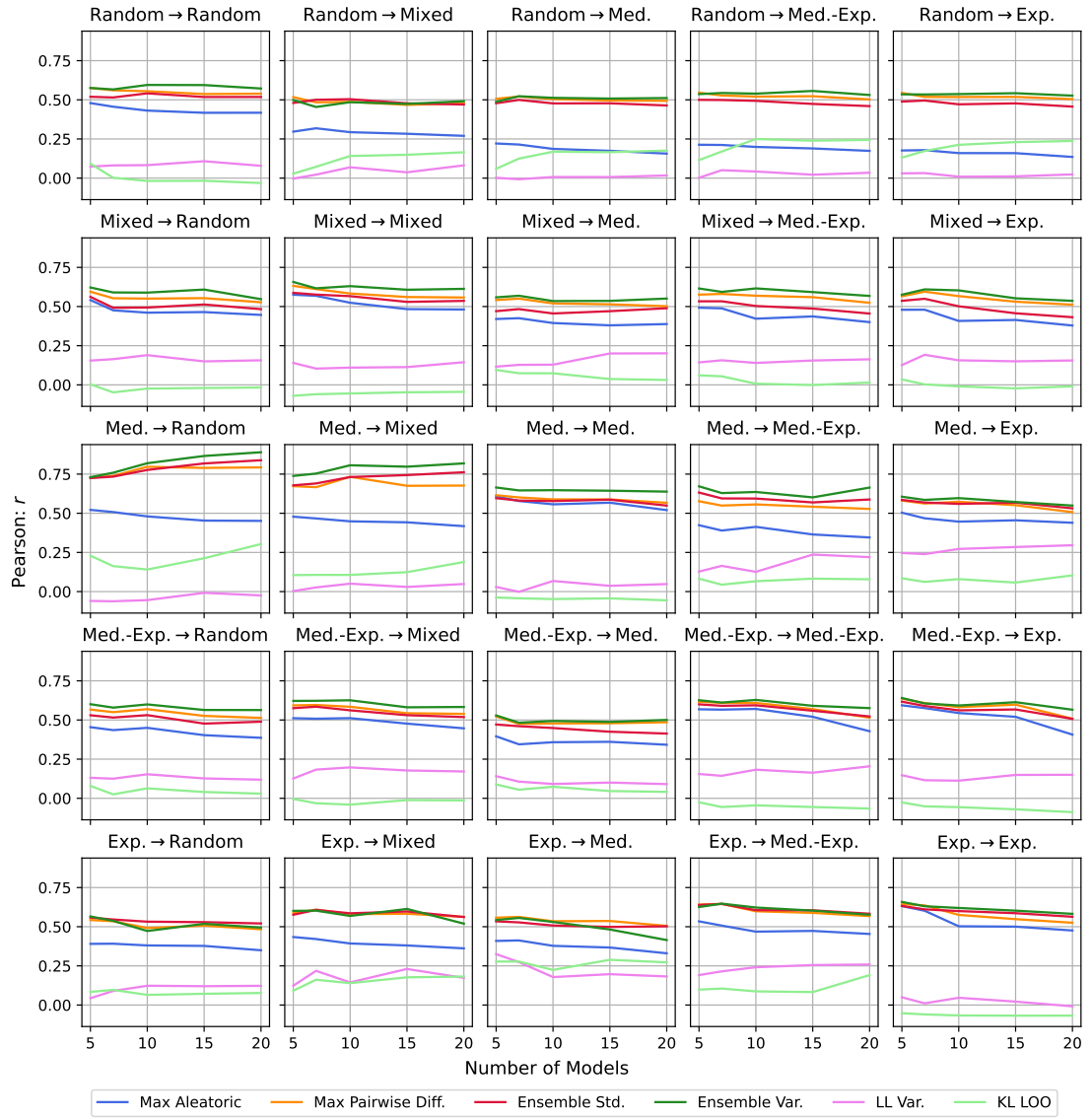


Figure A.12: Hopper Pearson Statistics

A.2.7 HalfCheetah D4RL: True Model-Based Error

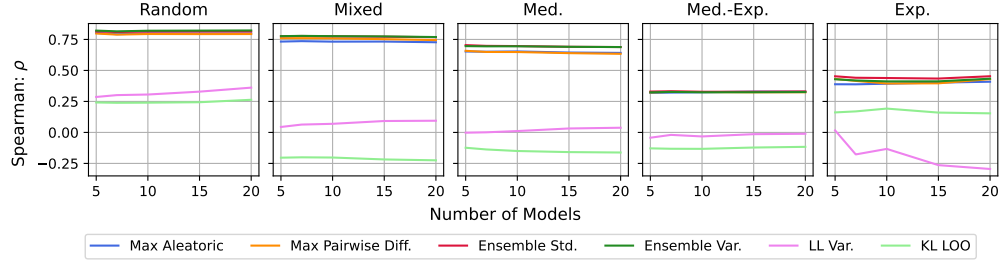


Figure A.13: HalfCheetah Spearman Statistics

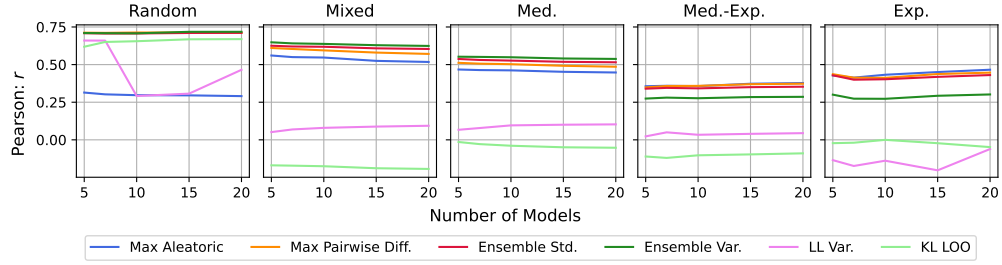


Figure A.14: HalfCheetah Pearson Statistics

A.2.8 Hopper D4RL: True Model-Based Error

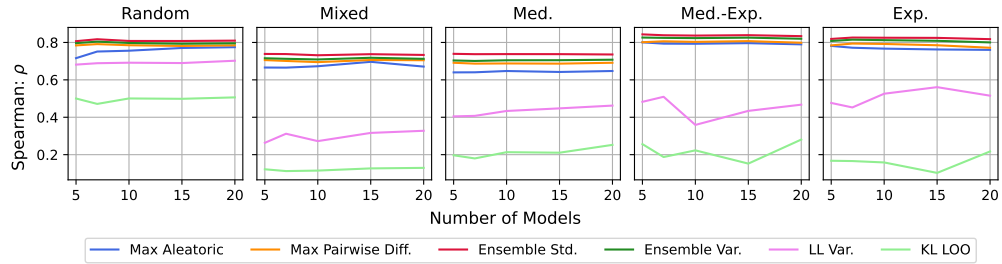


Figure A.15: Hopper Spearman Statistics

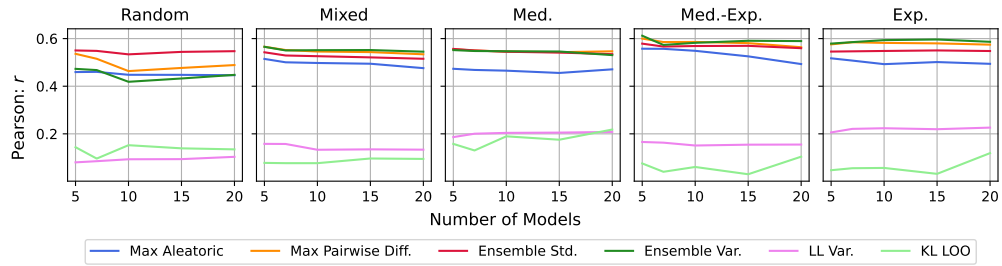


Figure A.16: Hopper Pearson Statistics

A.2.9 All Aggregated

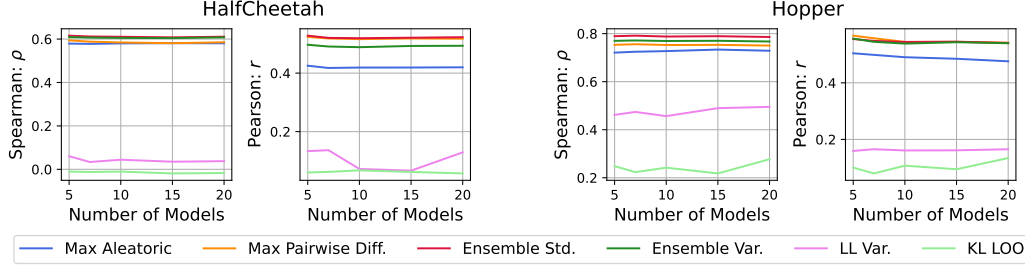


Figure A.17: Aggregated True Model-Based correlation statistics over all datasets (i.e., Random through to Expert); **Left:** HalfCheetah; **Right:** Hopper

A.3 Skewness and Kurtosis Comparisons

A.3.1 Skewness and Kurtosis Overall

Here we present the 3rd and 4th order statistics (skew and kurtosis respectively) of each penalty, illustrating that even with identical model counts, the shape statistics between penalties are vastly different.

Table A.2: Skew (γ_1) and Kurtosis (γ_2) statistics of all experiments averaged over all datasets (i.e., Random through to Expert) using the MOPO Default of 7 models.

Penalty	Transfer				True Model-Based			
	HalfCheetah		Hopper		HalfCheetah		Hopper	
	γ_1	γ_2	γ_1	γ_2	γ_1	γ_2	γ_1	γ_2
Max Aleatoric	-0.010	0.580	0.689	1.377	0.671	0.920	1.873	2.864
Max Pairwise Diff.	0.919	0.957	1.967	4.578	1.661	3.081	2.571	7.465
Ensemble Std.	0.794	0.806	2.136	6.560	1.656	3.178	2.739	9.061
Ensemble Var.	1.823	4.830	3.436	15.983	2.612	8.800	4.517	25.380
LL Var.	6.893	114.843	10.920	180.716	5.100	37.865	14.415	251.705
LOO KL	1.778	5.729	3.729	29.606	1.840	4.600	4.008	28.089

A.3.2 Skew and Kurtosis Scaling with Model Count

We omit LL Var. and LOO KL due to the fact that their changes were so significant as to obfuscate the changes of the more performant penalties.

We choose 7 models, as in Table A.2, to act as our 'baseline' (following the default MOPO setting), and we measure the change in the skew and kurtosis relative to this, hence 7 models always has a 0% change in our graphs. For brevity, in the

transfer experiments, we average over all ‘transferred to’ environments, e.g., Random, Medium, etc.; the graph title refers to the data that the model was trained on.

Again, we observe the environment *and* setting dependency of these metrics, sometimes having increasing skewness and kurtosis with model count, and other times decreasing. This further justifies using a ranking metric to compare penalties, as the overall penalty shape can vary hugely and unpredictably w.r.t. co-dependent hyperparameters. We do observe however in the True Model-Based experiments that ensemble standard deviation appears to be most robust to scaling with models. We also observe that the Max Aleatoric penalty can change shape significantly w.r.t. model count, and no penalties are fully immune to this. This further advocates the use of shape meta-parameters to control for changing distribution properties when adjusting the number of models as a hyperparameter, as well as selecting penalties that are relatively invariant to model count to make tuning easier.

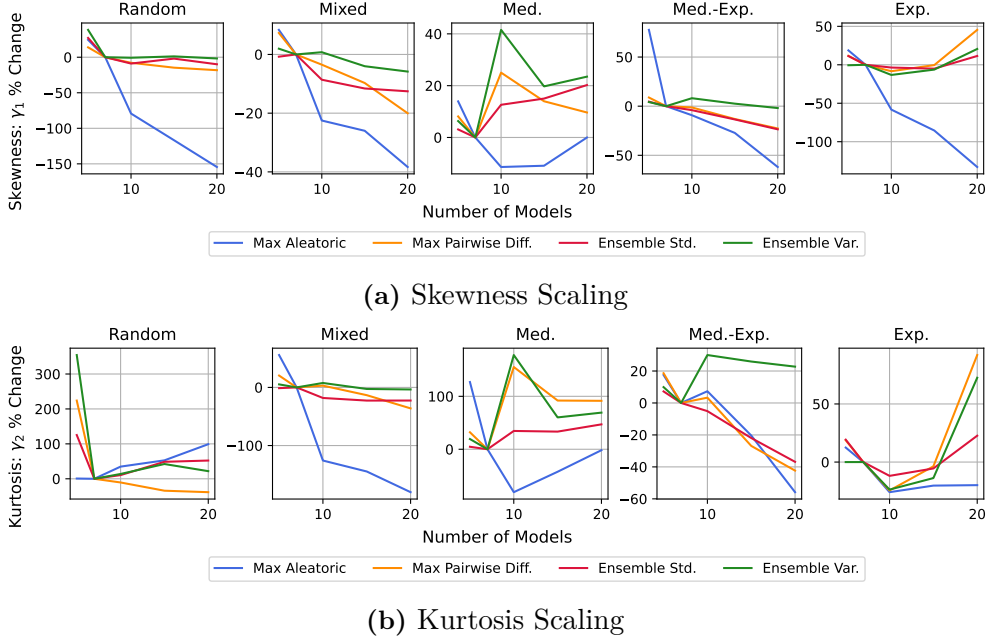
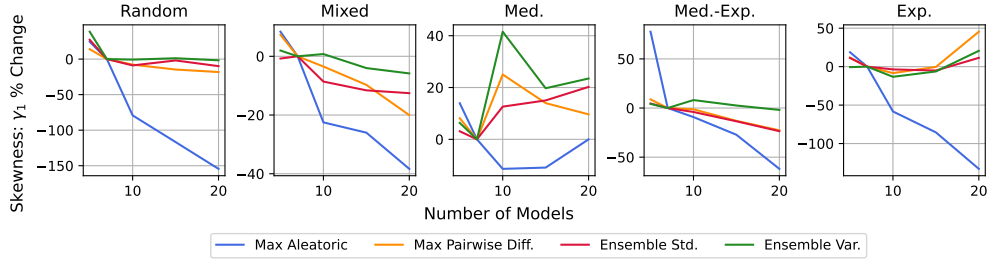
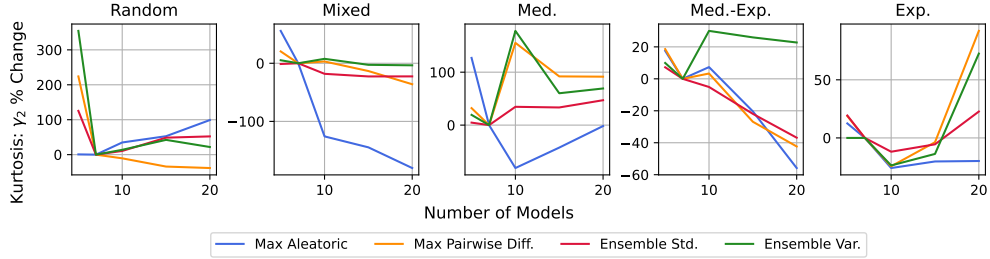


Figure A.18: HalfCheetah Transfer.



(a) Skewness Scaling



(b) Kurtosis Scaling

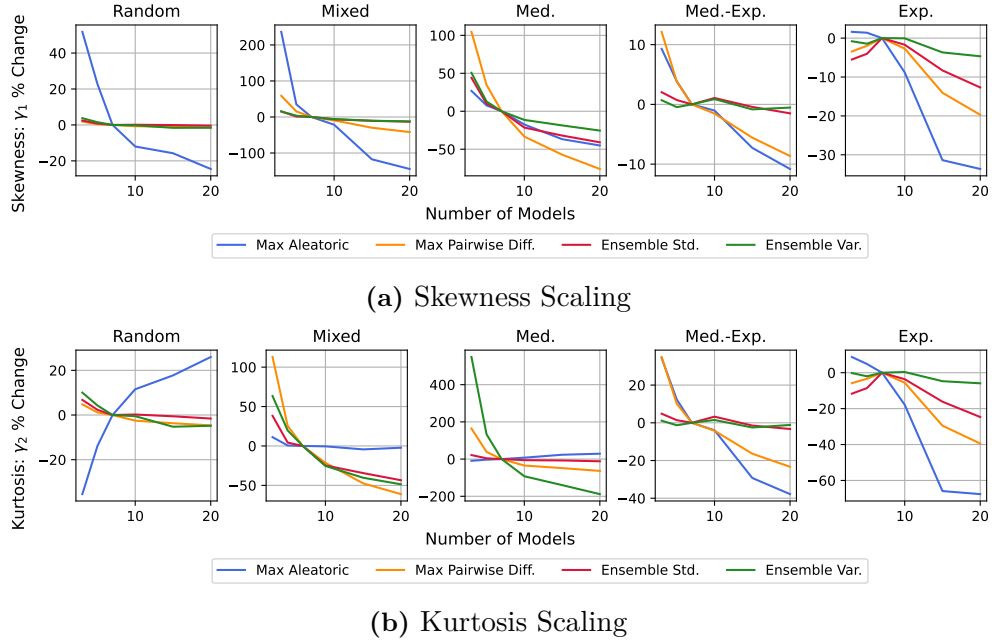
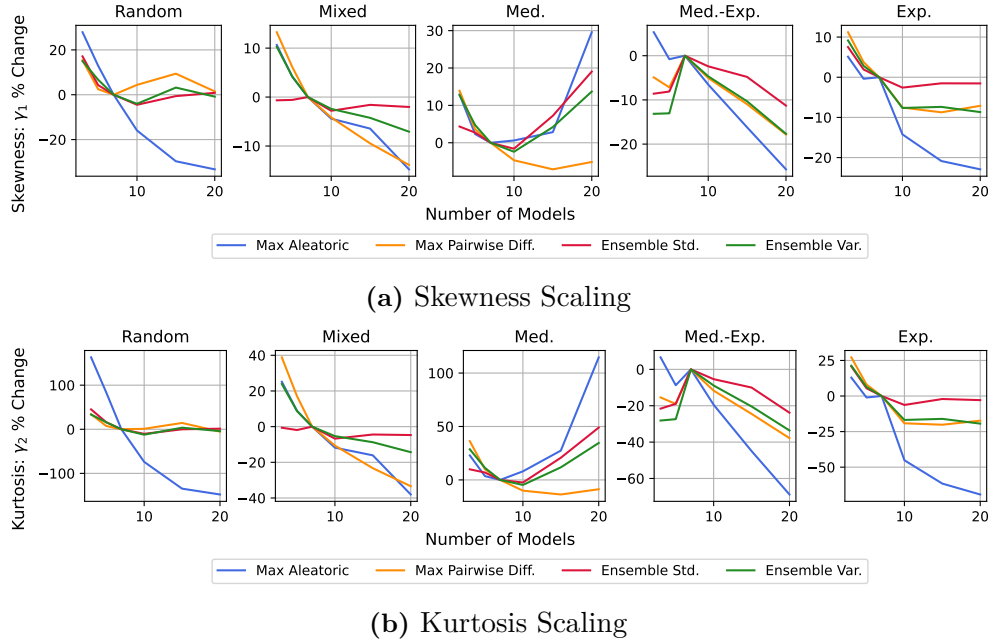
Figure A.19: Hopper Transfer.

A.4 Further details on True Model-Based experiments

A.4.1 Methodological Details

We leverage the MuJoCo (Todorov et al., 2012) simulator to provide us with ground truth dynamics that we can use to compare against our model predictions and penalties. This is done by providing the state and action inputs given to the model to the simulator through the `set_state` method in the simulator API. It must be noted that this method also requires an additional ‘displacement’ value which is not modelled by the world models (nor is it provided in the D4RL data), however we found in practice this did not affect the dynamics predicted by the simulator, and simply setting this to 0 was sufficient to generate ground truth predictions.

This makes it possible to provide the simulator the hallucinated model states,

**Figure A.20:** HalfCheetah True Model-Based.**Figure A.21:** Hopper True Model-Based.

and provide a true proxy to the *dynamics* discrepancy. We note that since the states are ‘hallucinated’ by the model, it might be the case that they may not be admissible under the true environment, but in reality the simulator was able to process almost any combination of state and action, barring settings that featured

anomalously large magnitudes. To handle such cases, we found it necessary to clip the model states to the range $[-10, 10]$.

In order to assess the permissibility of states, as well as measure the accuracy of the penalties as OOD input detectors, we provide an alternative distance measure based on the distance away from the training set. We use this measure for our analysis in Section 3.1.5, and is calculated as the distance from the offline training dataset, which we define to be the 2-norm between a given state-action tuple and its nearest point in the offline data, a similar metric to those used in recent works on imitation learning (Dadashi et al., 2021). We describe this quantity henceforth as ‘Distribution Error’.

A.4.2 On the nature of OOD data along hallucinated trajectories

Here we discuss the nature of OOD data along a single hallucinated trajectory (in the model) in offline MBRL, analyzing the inductive bias that some ‘error’ increases with increasing rollout length in the model. We find that there is merit to this assumption, and show this in Fig. A.22 for all HalfCheetah and Hopper environments in D4RL. Here, we plot the median error at each time-step across 30,000 aggregated trajectories in the model. Note that for all plots, we re-normalize all penalties by subtracting their mean and dividing by their standard deviation to facilitate comparison; this normalization was also applied in the analysis performed in Fig. 3.1a. Concretely, each time step corresponds to the normalized median value of 30,000 data-points.

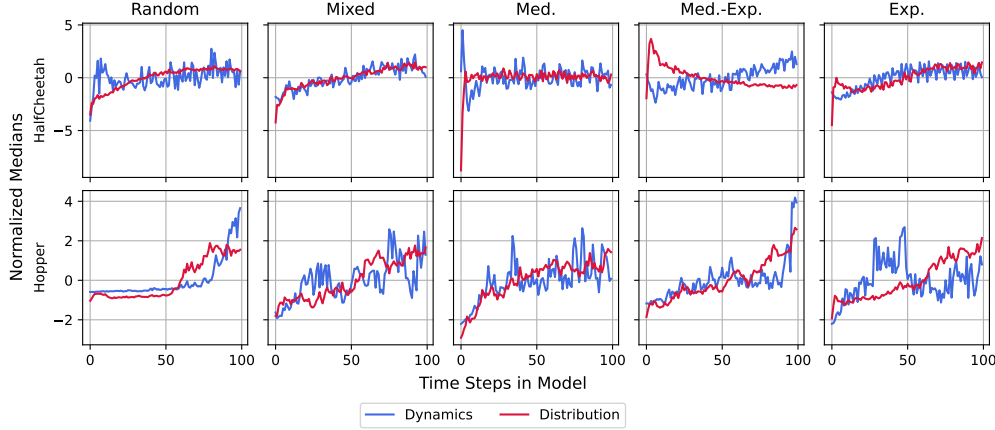


Figure A.22: Median True Model-Based Errors as a function of rollout timestep

We observe indeed that both median dynamics and distribution errors increase with increasing time step in the model. The only real exception is HalfCheetah Medium-Expert, which we believe to be due to our trained policy not being able to successfully exploit this environment.

The above analysis captures *overall trends* in the error over a large number of trajectories. However, the way errors manifest during an *individual rollout* is not so straightforward. To illustrate this, observe Fig. A.23, where we plot a random subset of 5 individual rollouts from the Hopper Medium-Expert data we generated.

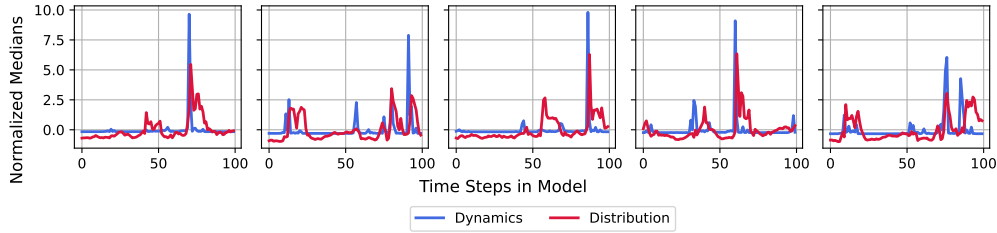


Figure A.23: Several Individual Ground Truth Rollouts in Hopper Medium-Expert

We observe that errors along any single trajectory tend to manifest as ‘spikes’, and that it is entirely possible to recover from these, returning to either admissible dynamics, or parts of the state-action space that have been seen in the data. This speaks to the nature of how we ought to penalize policies for accessing regions of inaccuracy/uncertainty, and may justify a hybrid MOPO/MOREL approach, whereby we penalize individual transitions along a trajectory, but do not stop rollouts early. Indeed, this is similar to the approach taken in M2AC (non-stop),

albeit they choose to ‘mask’ uncertain transitions, not penalize them. We leave the design of such an algorithm to future work.

Finally, we address the issue of comparing OOD dynamics and inputs. As already observed in Fig. A.23, these two errors are not necessarily always the same, and oftentimes it is possible that one quantity is large, whilst the other is small. We revisit Fig. 3.1a to explore this, now also plotting the Distribution Error in Fig. A.24.

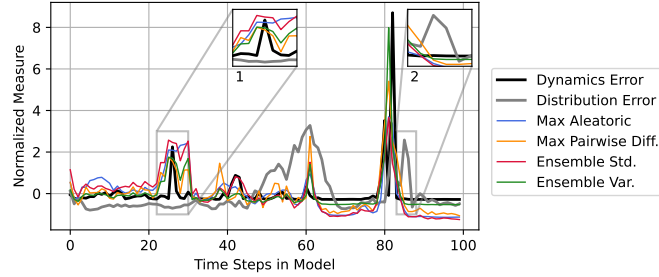


Figure A.24: Comparing OOD dynamics and inputs on a Hopper Medium-Expert trajectory

We first speak to the inset annotated ‘1’. Here we observe that the transitions generated in fact closely resemble the data that our model was trained on, however the predicted dynamics are incorrect, and cause an aforementioned ‘spike’. This is the opposite of what is observed in the inset annotated ‘2’; where we actually predict accurate dynamics, however the resultant state-action tuples do not closely resemble the data that our model was trained on. We generally observe that regions of high Distribution Error tend to be preceded by ‘spikes’ pertaining to high Dynamics Error, and this present an exciting avenue for future work understanding how these quantities are related.

A.4.3 On the diversity of exploitative policies

It is possible that training policies purely to exploit the world models may result in generating state-action tuples that are low in diversity, as the policy could discover "pockets" in the model that provide consistently high return. To prevent this, we train multiple policies inside the model from different seeds, with the aim of inducing different modes of exploitation. To validate this induces diverse trajectories in the world models, we visualize the state-action manifold using a t-SNE projection

(van der Maaten & Hinton, 2008) of: 1) the D4RL Hopper Mixed-Replay (which contains diverse samples); 2) the imagined rollouts inside the model from the exploitative policies in Fig. A.25. The policies were trained to exploit a model that itself was trained on the Hopper Mixed-Replay data.

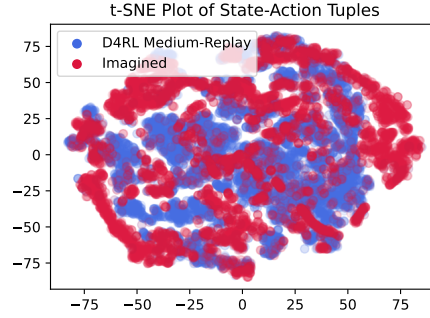


Figure A.25: t-SNE projection of 10,000 Hopper D4RL Medium-Replay and 10,000 exploitative Imagined policy state-action tuples

We observe that the induced policies inside the model displays some overlap with the D4RL data, but also resides in parts of the manifold where there is little coverage from the D4RL data, likely representing regions of exploitation. Importantly, the exploitative WM trajectories display strong state-action tuple diversity, comparable to that of the offline data it was trained on.

A.5 Using metrics as OOD event detectors

A.5.1 Measuring Statistics

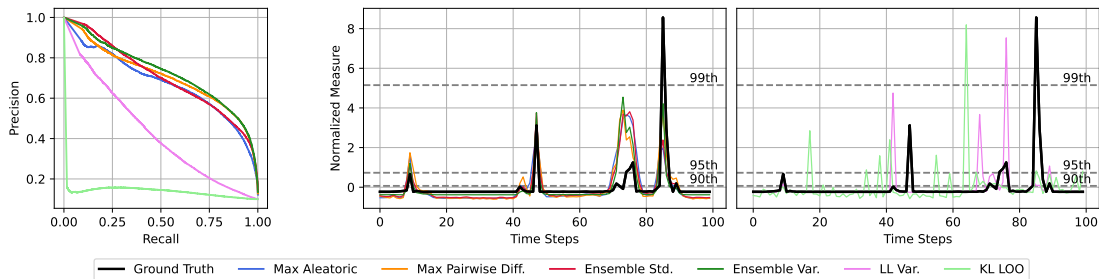


Figure A.26: Hopper Medium-Expert True Model-Based Experiments; **Left:** Precision v.s. Recall against Ground Truth; **Middle:** Higher Performing Penalties v.s. Ground Truth MSE in Imagined Rollout; **Right:** Lower Performing Penalties v.s. Ground Truth MSE in Imagined Rollout

As noted previously, different penalties have varying scales and distribution profiles, so we need a way of standardizing the method of assessment. Using our observation that errors manifest as ‘spikes’ during a rollout, we propose treating each penalty as a classifier. Concretely, our test set consists of the ground truth data labeled by whether or not they exceed a certain percentile at a particular time step. Each penalty may be then be treated as a ‘classifier’ by normalizing its range to lie in $[0, 1]$. We can then use standard classification quality measures, such as AUC, to determine the effectiveness of these penalties at capturing these spikes, whilst sidestepping the issue of the different distributional profiles identified previously.

Fig. A.26 shows how our proposed method may be used to compare the effectiveness of each metric at capturing OOD events. In the figure, we plot a single rollout in the model, and the resultant ground truth MSE between the predicted next state and the true next state in black. We then superimpose the 90th, 95th and 99th percentile MSEs across the entire imagined trajectories onto the figure in gray dashed lines. To construct our OOD labels, we label any point below the percentile line as being ‘False’, and any point above that line as being ‘True’. Finally, we normalize the uncertainty metrics as previously described into values in the range $[0, 1]$, allowing us to construct precision-recall graphs and calculate classifier statistics.

A.5.2 Precision Recall Curves

In this section we present the Precision-Recall curves described in App. A.5.1.

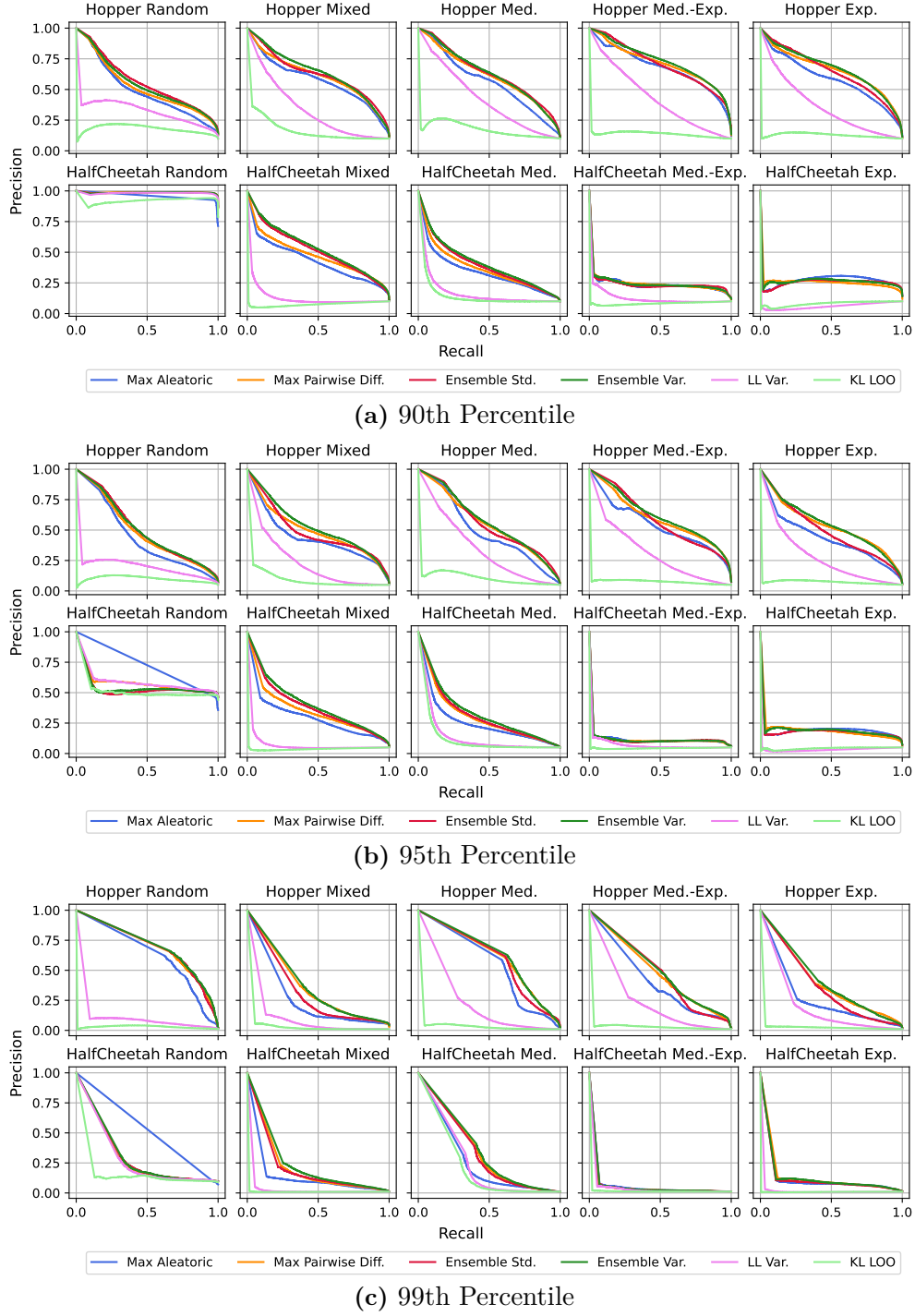


Figure A.27: Precision Recall curves on ground truth data.

A.6 Implementation Details

This paper extensively discusses key *hyperparameters* specific to current offline MBRL algorithms. However, there are significant *code-level* implementation details which are often critical for strong performance and make it hard to disambiguate between algorithmic and implementation improvements. Worryingly, many of these details are not mentioned in their respective papers, or are different between the authors’ code and paper. We detail clear examples of this below. We believe further investigation of these code-level implementation details represents important future work, as has already been done for policy gradients (Engstrom et al., 2020; Andrychowicz et al., 2021). Indeed – it is unclear if the improvement of MOREL over MOPO is due to its different P-MDP formulation, or if it is successful *in spite of* this formulation, due to a superior policy optimization strategy or dynamics model design. We believe that this paper takes a significant first step in tackling this issue by directly comparing a number of key design choices, and understanding their individual impact. Now we summarize key differences between the paper and code for the MOPO and MOREL algorithms which we compare against that are crucial to achieve the same reported performance.

In MOPO,

- Each layer in the model neural network has a different level of weight-decay
- The authors’ code uses different objectives for training (log-likelihood) and validation (MSE).
- The authors use elites, but only for next state prediction (as discussed previously).

In MOREL,

- There is a difference in the authors’ code about how the penalty threshold is calculated and tuned, and isn’t provided as a hyperparameter in the appendix.
- The absorbing HALT state does not appear in the authors’ code.

- The negative halt penalty appears significantly different between code and paper.
- There is a minimum trajectory steps parameter (hard-coded to 4) not mentioned in the paper.
- The reward function appears to be hard-coded in the authors’ implementation, not learned as stated in the paper.
- The policy architecture is different in the authors’ code (64,64 hidden layers) and the paper (32,32 hidden layers)
- It is not clear when the optional behavior cloning initialization step is applied.

A.7 Hyperparameters and Experiment Details

The D4RL (Fu et al., 2021) codebase and datasets used for the empirical evaluation is available under the CC BY 4.0 Licence. As stated in the main text, we choose to use ‘v0’ experiments as these are more challenging for Hopper due to having low return trajectories (Kostrikov et al., 2021), and we clearly state when other benchmarks use the ‘v2’ experiments, which have offline trajectories with higher returns on Hopper.

The remaining hyperparameters for the MOPO algorithm that we do not vary by Bayesian Optimization were taken from the original MOPO paper (Yu et al., 2020), apart from we fix the number of policy epochs/iterations to 1,000 for all experiments. This means our implementation uses the same probabilistic dynamics models (with unchanged hyperparameters) and policy optimizer (SAC (Haarnoja et al., 2018)) as MOPO, differing from MOREL, which uses Natural Policy Gradient (Kakade, 2002).

The hyperparameters used for the BO algorithm, CASMOPOLITAN, are listed in Table A.3. We use the batch-mode of CASMOPOLITAN, where multiple hyperparameters settings are proposed and evaluated concurrently.

Each BO iteration is run for 300 epochs on a single seed, and the full optimization over an offline dataset took ~200 hours on a NVIDIA GeForce GTX 1080 Ti GPU taken up predominantly by MOPO training.

Table A.3: CASMOPOLITAN Hyperparameters

Parameter	Value
Number of parallel trials	4
Number of random initializing points	20
ARD	False
Acquisition Function	Thompson Sampling
Global BO	True
Kernel	CoCaBo Kernel (Ru et al., 2020)

Unless specified otherwise, plots and reported statistics are completed with 7 models in the ensemble, as this is the number chosen in the original MOPO paper used with the Max Aleatoric penalty.

A.8 **rliable** framework for performance evaluation

Throughout this work, we choose to adopt the **rliable** framework introduced in Agarwal et al. (2021) to evaluate the performance of our approaches. **rliable** advocates for computing aggregate performance statistics and probability of improvement across many tasks in a benchmark suite; indeed we take this approach when reporting the values in the analysis performed in Sections 3.1.4 and 3.1.5. This is important when the number of tasks become large, and also prevents outliers from dominating mean statistics. Furthermore, this allows us to make clear statements about improvements given the relatively low number of seeds that are used in deep RL; indeed, Agarwal et al. (2021) show that even using high seed counts does not ameliorate the variance issues experienced when training such algorithms. For normalization, we use the standard D4RL return scaling.

A.9 Evaluation using automatic constraint tuning

We show the full tabulated results from Sec. 3.1.6 with statistical significance using the **rliable** framework in Table A.4, using the ‘Probability of Improvement’ metric in Agarwal et al. (2021).

To evaluate our claims in Sections 3.1.4 and 3.1.5 without needing to laboriously tune the penalty weight λ per environment, we employ an automatic penalty tuning scheme, analogous to the automatic entropy tuning used in Haarnoja et al. (2018). Concretely, given a constraint value Λ , at each epoch we minimize:

$$J(\lambda) = \mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \mathcal{D}} [\log \lambda (\Lambda - \lambda \cdot u(\mathbf{s}_t, \mathbf{a}_t))] \quad (\text{A.4})$$

We start from an initial weight $\lambda = 1$. We observe that the penalty weight found by automatic tuning tends to converge within the first 50 epochs and then remains stable throughout training.

Table A.4: Improvement over grid-searched MOPO through restricted hyperparameter choices (e.g., one single choice, or an arg max between two) on the D4RL MuJoCo benchmark. The single and two setup approaches both use the Ensemble Std. penalty and $N = 10$.

Algorithm	Average Score	$\mathbb{P}[\text{Improvement over MOPO}]$
MOPO (default hyperparameters)	34.2	-
Single setup: ($h = 20, \Lambda = 1$)	49.0 (+43%)	73.96%
Two setups: $\arg \max\{(h = 10, \Lambda = 0.5), (h = 20, \Lambda = 1)\}$	57.8 (+69%)	80.20%
Optimized MOPO (ours, Table 3)	65.2 (+91%)	89.06%

A.10 Best found Adroit hyperparameters

We present the best found hyperparameters under our BO procedure for the D4RL Adroit tasks in Table 3.4. We see similar trends as in our main evaluation in Table 3.3 favoring higher rollout lengths and the Ensemble penalties.

Table A.5: Best discovered hyperparameters using BO for Adroit

Environment		Discovered Hyperparameters			
		N	λ	h	Penalty
pen	cloned	10	6.64	12	Ensemble Std
	human	11	0.96	37	Ensemble Var
	expert	7	4.56	5	Max Aleatoric
hammer	cloned	10	0.21	12	Ensemble Var
	human	13	2.48	47	Ensemble Std
	expert	12	0.99	37	Ensemble Std

Chapter Bibliography

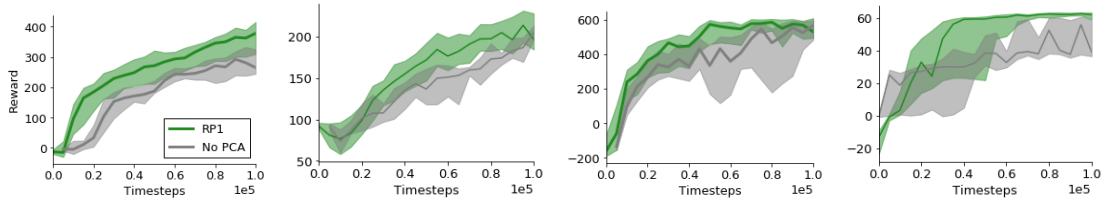
- Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.
- Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters for on-policy deep actor-critic methods? a large-scale study. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=nIAxjsniDzg>.
- Robert Dadashi, Leonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal Wasserstein imitation learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=TtYSU29zgR>.
- Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020.
- Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2021.
- Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.
- Sham M Kakade. A natural policy gradient. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002. URL <https://proceedings.neurips.cc/paper/2001/file/4b86abe48d358ecf194c56c69108433e-Paper.pdf>.
- Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning, 2021.
- Binxin Ru, Ahsan Alvi, Vu Nguyen, Michael A Osborne, and Stephen Roberts. Bayesian optimisation over multiple continuous and categorical inputs. In *International Conference on Machine Learning*, pp. 8276–8285. PMLR, 2020.
- Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.
- Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL <http://jmlr.org/papers/v9/vandermaaten08a.html>.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*. 2020.

B

Ready Policy One: Appendix

B.1 Additional Ablation Studies

In this section we show the full results from the ablation study in Table 4.3.



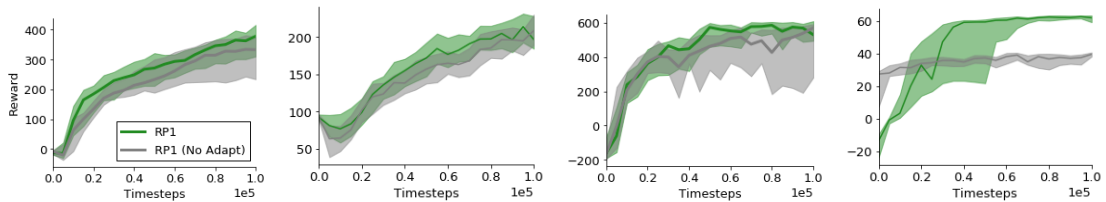
(a) HalfCheetah

(b) Ant

(c) Hopper

(d) Swimmer

Figure B.1: Ablation study where we consider removing the early stopping mechanism. All results show the median performance across ten seeds.



(a) HalfCheetah

(b) Ant

(c) Hopper

(d) Swimmer

Figure B.2: Ablation study where we consider removing the adaptive mechanism. All results show the median performance across ten seeds.

B.2 Implementation Details

In terms of approach, we follow ME-TRPO (Kurutach et al., 2018) with some adjustments. Instead of using TRPO loss (Schulman et al., 2015), we leverage the

first-order approximation loss in PPO (Schulman et al., 2017). We do not apply parameter noise, and we modify policy training slightly by ensuring at least 10 updates are performed before termination is considered, which we find helps to improve convergence. We do not apply GAE nor the overall training approach in Schulman et al. (2017), as this introduced instabilities. We instead found that generating large batch sizes gave better and more consistent performance, which corroborates the findings in Ilyas et al. (2018) with respect to true policy gradients.

We augment each environment with an additional state which contains velocity information. This is also done in the ‘FixedSwimmer’ environment in Wang et al. (2019), and allows us to infer the reward from the states directly. It must also be noted that in the original ‘rllab’ (Duan et al., 2016) environments used in Kurutach et al. (2018), one of the observable states was the velocity state used to calculate rewards, and we therefore mirror this in our OpenAI Gym (Brockman et al., 2016) implementation; we do not anticipate there to be any problem integrating reward prediction with our framework. Furthermore, we provide this state in both the model-free and model-based benchmarks to ensure there is no advantage, and do not notice any noticeable improvement in the model-free setting when this is provided; we hypothesize that some close proxy to the true velocity state already exists in the original state-space. We remove contact information from all environments, and instead of ‘Swimmer-v2’ we use the aforementioned ‘FixedSwimmer’, since this can be solved by our policy in a model-free regime. We remove early stopping from Hopper since we found it was necessary for convergence, but left the early stopping in for Ant since it was possible to train performant policies. We train the policy for 100 time steps in HalfCheetah and Ant, and for 200 time steps in Swimmer and Hopper. In experiments without the early stopping mechanism, data collection defaults to 3,000 timesteps per iteration. Full hyperparameter values can be found in Table B.2.

We use the following approach to normalize G to produce $\hat{G}$; for convenience, we write $\hat{G}_{\phi_t}(\theta_{t+1})$ as $\hat{G}_t$.

$$\hat{G}_t = \frac{G_t - \frac{1}{5} \sum_{\tau=t-5}^{t-1} (G_\tau)}{l_{\text{val}}} \quad (\text{B.1})$$

where l_{val} is the final model validation loss from the iteration $t - 1$.

Attention should be drawn to the α parameter used to determine early stopping in Algorithm 2. For the tasks we test on, we choose to fix this to 0.0005, and therefore do not tune it to be task specific. We found that at this setting of α , the early stopping mechanism generally collects significantly fewer than the default 3,000 samples (which acts as an upper bound in RP1), but can still expand the batch size collected to the full amount where appropriate (i.e., under policies that provide non-homogenous trajectories).

All experiments were run on a virtual machine with an 8-core Intel Skylake microarchitecture CPU, 32GB of RAM, and an NVIDIA Tesla P100 GPU.

Table B.1: Hyperparameters used in the Policy

Hyperparameter Name	Value
Optimizer	Adam
Learning Rate	3e-4
Loss	PPO
Discount Factor	0.99
Batch Size	50,000
Epochs per Batch	10
ϵ -clip	0.2
Default Action σ	0.5
Hidden Layer Size	32
Number of Hidden Layers	2
Activation Function	ReLU

Table B.2: Hyperparameters used in the World Model

Hyperparameter Name	Value
Optimizer	Adam
Learning Rate	1e-3
Train/Validation Split	2:1
Number of Models	5
Batch Size	1,024
Hidden Layer Size	1,024
Number of Hidden Layers	2
Activation Function	ReLU
Early Stopping α in PCA	0.0005

Chapter Bibliography

- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.
- Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. *CoRR*, abs/1604.06778, 2016. URL <http://arxiv.org/abs/1604.06778>.
- Andrew Ilyas, Logan Engstrom, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Are deep policy gradient algorithms truly policy gradient algorithms? *CoRR*, abs/1811.02553, 2018. URL <http://arxiv.org/abs/1811.02553>.
- Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=SJJinbWRZ>.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pp. 1889–1897, 2015.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <http://arxiv.org/abs/1707.06347>.
- Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *CoRR*, abs/1907.02057, 2019. URL <http://arxiv.org/abs/1907.02057>.



Stabilizing Off-Policy Deep Reinforcement Learning from Pixels: Appendix

C.1 Detailed Results

C.1.1 DMC Medium and Hard Tasks

In Table C.1, we show the performance in each of the evaluated 15 DMC environments by reporting the mean and standard deviations over the cumulative returns obtained midway and at the end of training for the medium and hard benchmark tasks, respectively. A-LIX attains state-of-the-art performance in the majority of the tasks at *both* reported checkpoints, while still closely matching DrQ-v2’s performance on the remaining tasks. On the other hand, DrQ-v2 struggles to consistently solve some of the harder exploration tasks such as Cartpole Swingup Sparse and Humanoid Run, as shown by the high standard deviations. Interestingly, unlike in the simpler DMC benchmark from [Hafner et al. \(2019\)](#) with higher action repeat, CURL appears have a slight edge over DrQ. In particular, the self-supervised signal from CURL appears to aid precisely in the sparse reward environments where DrQ-v2 struggles. Hence, this appears to suggest that including an additional self-supervised signal to the TD-loss, lessens the hindering effects of a lower-magnitude reward signal. We interpret this result as additional evidence showing how addressing

any individual component of the deadly triad helps counteracting the catastrophic self-overfitting phenomenon.

We also test the significance of our results by performing a Wilcoxon signed-rank test (Wilcoxon, 1945) between A-LIX and DrQ-v2. We perform a paired rank test across both seeds and tasks, allowing us to obtain an p -value that takes into account both population size and relative performance gains across all tasks. The choice of Wilcoxon signed-rank test also does not presume normality in the distributions of performance which we believe is a more appropriate assumption than for instance a paired t -test (Student, 1908), despite a potential loss of statistical power. To ensure correct population pairing, A-LIX and DrQ-v2 seeds were identical, resulting in the same initially collected data and network initialization. Performing this test over all 15 tasks and 5 seeds, we achieve a p -value of **0.0057** at 50% total frames (1.5M and 15M for Medium and Hard respectively) and **0.0053** at 100% total frames (3.0M and 30M for Medium and Hard Respectively), much lower than the typical rejection criteria of $p > 0.05$. We therefore believe this shows clear evidence that our results in DMC are *strongly statistically significant*.

Table C.1: Full results for the DeepMind Control Suite benchmark. Each displayed return is averaged over 10 random seeds and from 10 evaluation runs collected at each experience checkpoint.

	1.5M frames					3.0M frames				
	SAC	CURL	DrQ	DrQv2	A-LIX (Ours)	SAC	CURL	DrQ	DrQv2	A-LIX (Ours)
<i>Medium tasks</i>										
<i>Acrobot Swingup</i>	8±9	6±5	24±27	256±47	270±99	12±11	6±5	28±25	442±64	402±100
<i>Cartpole Swingup Sparse</i>	118±233	479±329	318±389	485±396	718±250	185±295	499±349	316±389	505±412	742±250
<i>Cheetah Run</i>	9±8	507±114	788±59	792±29	806±78	7±8	590±95	835±45	873±60	864±78
<i>Finger Turn Easy</i>	190±137	297±150	199±132	854±73	546±101	200±155	309±176	216±158	934±54	901±109
<i>Finger Turn Hard</i>	79±73	174±106	100±63	491±182	587±109	100±78	146±95	86±70	902±77	906±101
<i>Hopper Hop</i>	0±0	184±127	268±91	198±102	287±48	0±0	224±135	285±96	240±123	372±48
<i>Quadruped Run</i>	68±72	164±91	129±97	419±204	528±107	63±45	175±104	130±59	523±271	759±107
<i>Quadruped Walk</i>	75±65	134±53	144±149	591±256	776±37	48±32	168±49	142±67	920±36	900±37
<i>Reach Duplo</i>	1±1	8±10	8±12	220±7	212±3	2±3	7±10	9±9	228±2	221±3
<i>Reacher Easy</i>	52±64	707±142	600±201	971±4	887±19	115±98	667±182	612±181	940±50	966±19
<i>Reacher Hard</i>	3±2	463±196	320±233	727±172	720±83	10±23	678±350	397±273	935±49	855±83
<i>Walker Run</i>	26±4	379±234	474±148	571±276	691±10	25±3	447±224	547±143	616±297	756±10
Average score	52.28	291.73	281.03	547.96	585.67	63.80	326.45	300.27	671.40	720.30
	15.0M frames					30.0M frames				
	SAC	CURL	DrQ	DrQv2	A-LIX (Ours)	SAC	CURL	DrQ	DrQv2	A-LIX (Ours)
<i>Hard tasks</i>										
<i>Humanoid Walk</i>	7±3	5±3	3±2	243±162	476±79	4±3	4±3	5±3	675±86	754±79
<i>Humanoid Stand</i>	5±3	6±3	4±3	167±159	519±94	6±3	6±2	6±2	588±63	781±94
<i>Humanoid Run</i>	5±3	6±2	5±3	22±30	122±59	3±3	4±3	4±2	170±122	242±59
Average score	5.64	5.74	4.02	144.16	372.78	4.30	4.89	4.90	477.74	592.48

We now compare our results using the *Rliable* framework introduced in Agarwal et al. (2021) (see App. C.1.3 for a detailed explanation about the metrics introduced).

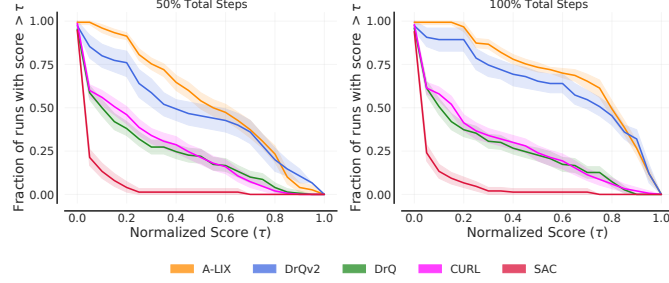
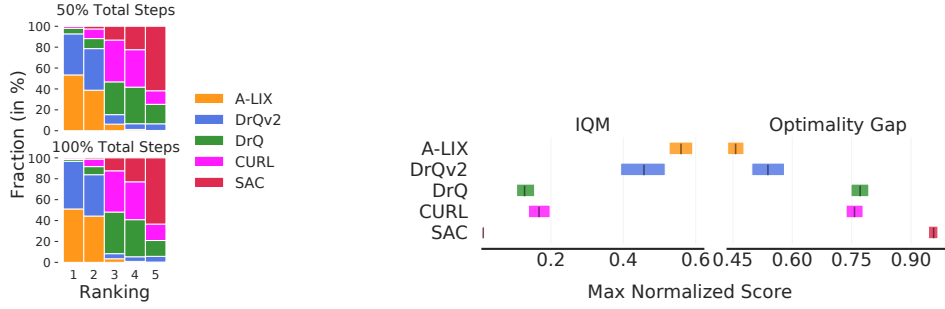


Figure C.1: Performance profiles at 50% (**left**) and 100% (**right**) of the total steps in Medium and Hard DMC Tasks.

We plot performance profiles in Fig. C.1 at both 50% and 100% the total training steps in DMC, which aim to represent sample efficiency and asymptotic performance respectively. We see that in almost all cases, A-LIX improves upon DrQ-v2.



(a) Overall ranking statistics at 50% (**top**) and 100% (**bottom**) of the total steps in Medium and Hard DMC Tasks.

(b) Aggregate IQM (**left**) and Optimality Gap (**right**) metrics at 50% of the total steps in Medium and Hard DMC Tasks.

Figure C.2: Medium and Hard DMC Task plots

We plot ranking statistics in Fig. C.1 at both 50% and 100% the total training steps in DMC. We see that A-LIX clearly appears most in the 1st ranked column, and rarely appears in lower ranked (i.e., > 3), suggesting strong performance across all environments in DMC Medium and Hard. We also provide a further aggregated statistics plot in Fig. C.2b (this time at 50% the total steps), which shows A-LIX is particularly sample-efficient and consistent (i.e., low error bars) across all environments.

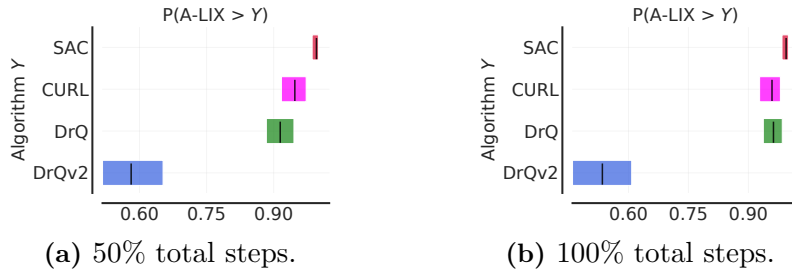


Figure C.3: Probability of Improvement statistics at both 50% (**left**) and 100% (**right**) of the total timesteps in Medium and Hard DMC Tasks.

In Fig. C.3 we observe that A-LIX likely improves over prior work, and note that whilst the improvement probability over DrQ-v2 may seem slightly low at $\sim 60\%$, we note that this value is in line with statistics in prior works that achieve significant gains (as seen in Agarwal et al. (2021)), and furthermore it does not take into account absolute performance values, and instead only compares relative values, which explains why the gains of A-LIX appear larger when evaluated under IQM and OG. Furthermore, the lower CI for 50% total steps does not fall below 0.5, which means improvements are indeed statistically significant.

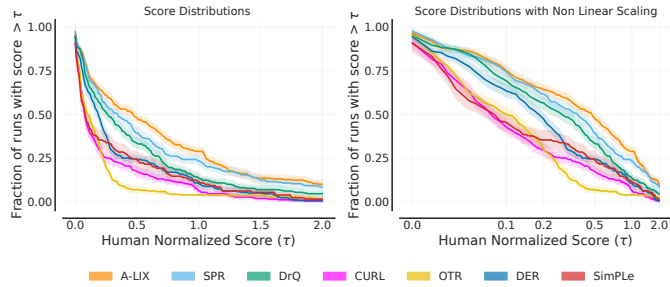
C.1.2 Atari 100k

In Table C.2, we show the final average performance for all the evaluated algorithms in each of the twenty-six tasks in the Atari 100k benchmark. A-LIX outperforms SPR, the previous state-of-the-art off-policy algorithm on this benchmark, on 16 out of 26 tasks. Moreover, it attains comparatively similar performance on most of the remaining tasks despite using no augmentation, auxiliary losses, or model-based elements.

Table C.2: Full results for the Atari 100k benchmark, following the evaluation protocol from Machado et al. (2018). We report the results collected from 10 random seeds.

<i>Tasks</i>	Random	Human	SimPLe	DER	OTRainbow	CURL	DrQ	SPR	A-LIX (Ours)
<i>Alien</i>	227.80	7127.70	616.9	739.9	824.7	558.2	771.2	801.5	902
<i>Amidar</i>	5.80	1719.50	88	188.6	82.8	142.1	102.8	176.3	174.27
<i>Assault</i>	222.40	742.00	527.2	431.2	351.9	600.6	452.4	571	660.53
<i>Asterix</i>	210.00	8503.30	1128.3	470.8	628.5	734.5	603.5	977.8	809.5
<i>Bank Heist</i>	14.20	753.10	34.2	51	182.1	131.6	168.9	380.9	639.4
<i>Battle Zone</i>	2360.00	37187.50	5184.4	10124.6	4060.6	14870	12954	16651	14470
<i>Boxing</i>	0.10	12.10	9.1	0.2	2.5	1.2	6	35.8	21.5
<i>Breakout</i>	1.70	30.50	16.4	1.9	9.8	4.9	16.1	17.1	23.52
<i>Chopper Command</i>	811.00	7387.80	1246.9	861.8	1033.3	1058.5	780.3	974.8	747
<i>Crazy Climber</i>	10780.50	35829.40	62583.6	16185.3	21327.8	12146.5	20516.5	42923.6	53166
<i>Demon Attack</i>	152.10	1971.00	208.1	508	711.8	817.6	1113.4	545.2	888.15
<i>Freeway</i>	0.00	29.60	20.3	27.9	25	26.7	9.8	24.4	31.04
<i>Frostbite</i>	65.20	4334.70	254.7	866.8	231.6	1181.3	331.1	1821.5	1845.7
<i>Gopher</i>	257.60	2412.50	771	349.5	778	669.3	636.3	715.2	500.6
<i>Hero</i>	1027.00	30826.40	2656.6	6857	6458.8	6279.3	3736.3	7019.2	7185.85
<i>Jamesbond</i>	29.00	302.80	125.3	301.6	112.3	471	236	365.4	341.5
<i>Kangaroo</i>	52.00	3035.00	323.1	779.3	605.4	872.5	940.6	3276.4	6507
<i>Krull</i>	1598.00	2665.50	4539.9	2851.5	3277.9	4229.6	4018.1	3688.9	4884.04
<i>Kung Fu Master</i>	258.50	22736.30	17257.2	14346.1	5722.2	14307.8	9111	13192.7	16316
<i>Ms Pacman</i>	307.30	6951.60	1480	1204.1	941.9	1465.5	960.5	1313.2	1258.4
<i>Pong</i>	-20.70	14.60	12.8	-19.3	1.3	-16.5	-8.5	-5.9	6.03
<i>Private Eye</i>	24.90	69571.30	58.3	97.8	100	218.4	-13.6	124	100
<i>Qbert</i>	163.90	13455.00	1288.8	1152.9	509.3	1042.4	854.4	669.1	2974
<i>Road Runner</i>	11.50	7845.00	5640.6	9600	2696.7	5661	8895.1	14220.5	17471
<i>Seaquest</i>	68.40	42054.70	683.3	354.1	286.9	384.5	301.2	583.1	654.6
<i>Up N Down</i>	533.40	11693.20	3350.3	2877.4	2847.6	2955.2	3180.8	28138.5	5011.7
<i>Human Norm. Mean</i>	0.000	1.000	0.443	0.285	0.264	0.381	0.357	0.704	0.753
<i>Human Norm. Median</i>	0.000	1.000	0.144	0.161	0.204	0.175	0.268	0.415	0.411
<i># SOTA</i>	N/A	N/A	7	1	1	1	1	4	11
<i># Super</i>	N/A	N/A	2	2	1	2	2	7	7
<i>Average Rank</i>	N/A	N/A	3.92	5.00	5.21	3.92	4.85	2.88	2.21

We now present additional evaluations under the *Rliable* framework, continuing on from the analysis in Fig. 4.13b.

**Figure C.4:** Performance profiles with linear (left) and logarithmic (right) scaling in Atari 100k.

In Fig. C.4 A-LIX performs noticeably better than previous work, and performs at least as well as SPR over all settings of normalized scores.

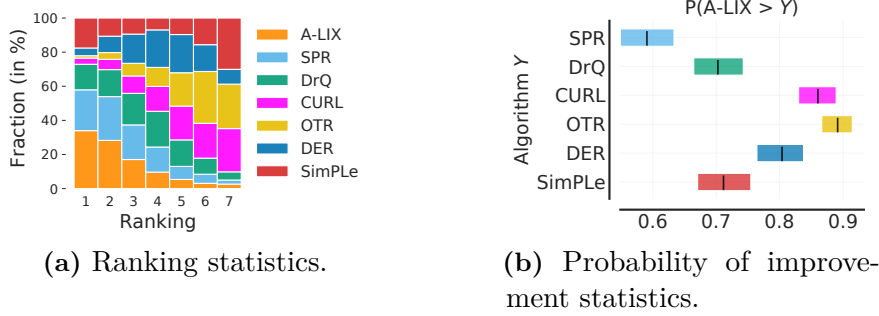


Figure C.5: Bootstrapped ranking statistics (**left**) and probability of improvement plots (**right**) on Atari 100k.

In Fig. C.5a A-LIX constitutes the majority of the algorithms ranked in 1st, and shows far fewer instances of being ranked in lower positions (i.e., > 4). In Fig. C.5b we observe A-LIX likely improves upon prior work. Similar to Fig. C.3, while the $\sim 60\%$ improvement value over SPR may seem low, this is justified due to shortcomings in this metric, such as not taking into account actual performance values, and instead relative improvements. Furthermore, the lower CI does not fall below 0.5, which means improvements due to A-LIX are statistically significant.

C.1.3 Rliable: A Primer

In addition to providing traditional methods of evaluation (e.g., performance tables, significance testing), we use robust metrics and evaluation strategies introduced in *Rliable* (Agarwal et al., 2021). *Rliable* advocates for computing aggregate performance statistics not just across many seeds, but also across the many tasks within a benchmark suite.

We give details on how these metrics achieve reliable performance evaluation in RL, denoting number of seeds as N and number of tasks as M . We follow Agarwal et al. (2021) as closely as possible; please refer to their paper for further details.

Seed and Task aggregation

In order to aggregate performances across different tasks in the same benchmark suite, we must first normalize each benchmark to the same range. In Atari, this is usually done by normalizing scores with respect to those achieved by humans, and

in DMC this is done with respect to the maximum achievable score (i.e., 1,000). We refer to this normalized score as τ .

IQM and OG

Interquartile Mean (IQM) takes the middle 50% of the runs across seeds and benchmarks (i.e., $[NM/2]$) and then calculates its mean score, improving outlier robustness whilst maintaining statistical efficiency. Optimality Gap (OG) calculates the proportion of performances (NM) that fail to meet a minimum threshold γ , with the assumption that improvements beyond γ are not important. In both cases, stratified bootstrap sampling is used to calculate confidence intervals (CIs).

Performance Profiles

Performance profiles are a form of empirical CDF, but with stratified bootstrap sampling to produce confidence bands that account for the underlying variability of the score. We can also establish ‘stochastic dominance’ by observing whether one method’s performance profile is consistently above another’s for all normalized performance values τ .

Ranking

Ranking shows the proportion of times a given algorithm ranks in a given position across *all* tasks, with distributions produced using stratified bootstrap sampling having 200,000 repetitions.

Probability of Improvement

Probability of improvement is calculated by calculating the Mann-Whitney U-statistic (Mann & Whitney, 1947) across all M tasks. The distribution is then plotted as a boxplot, and if the lower CI > 0.5 , the improvement is *statistically significant*.

C.2 Experiments Description

C.2.1 Offline Experiments

We follow the original training hyperparameters of DrQ-v2, and run policy evaluation and policy improvement until we saw convergence in the TD-loss, which would occur at similar points in all agents (i.e., between 10-20k and 5-10k steps of SGD in policy evaluation and policy iteration respectively). For the proprioceptive experiments, we keep everything consistent, except the input to the critic and actor MLP layers are now the proprioceptive states from the DMC simulator, not the latent representation z from the encoder. That is to say we do not modify the MLP architectures nor their learning rates in the interests of a fair comparison. Furthermore, for any given seed of the offline experiment, we also instantiate all networks in the agents identically and train on the same random offline data, with minibatches presented in the same order.

We also note that a similar algorithm is described in [Brandfonbrener et al. \(2021\)](#), but in the context of minimizing extrapolation errors.

Now we present some additional analysis to provide further context to our offline experiments. First, we see that the proprioceptive statistics mirror those of the augmented agent, further illustrating the crucial role of CNN regularization for successful TD-learning from pixels:

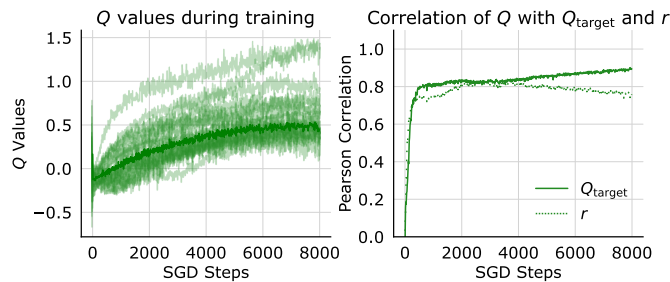


Figure C.6: Q values and Pearson Correlation of the offline Proprioceptive agent on an offline fixed batch.

Secondly, we observe that the exact same self-overfit also manifests in the online setting by plotting the Pearson correlation values over the initial stages of training in 5 seeds, confirming that phenomena of our offline analysis applies to the online RL problem:

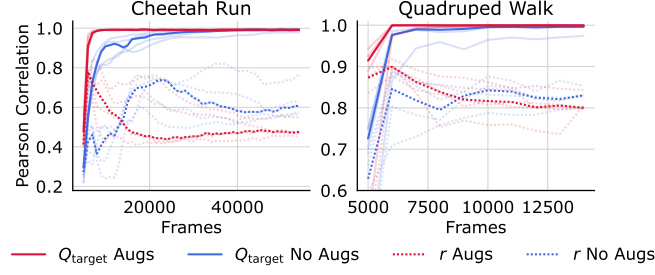


Figure C.7: Pearson Correlation of augmented and non-augmented online agents in Cheetah Run and Quadraped Walk across 5 seeds. Shaded lines represent individual runs, and solid lines represent the median. We see that augmented agents do not immediately overfit to their target networks, and become correlated only after useful signal is learned.

C.2.2 Jacobian Analysis

In order to measure local sensitivity, we linearize the encoder around its input using a Taylor series expansion. Consider an N -dimensional input $\mathbf{x} \in \mathbb{R}^N$ and perturbation $\epsilon \in \mathbb{R}^N$, an M -dimensional output $\mathbf{y} \in \mathbb{R}^M$, and a function $\mathbf{F} : \mathbb{R}^N \rightarrow \mathbb{R}^M$. Now, performing a Taylor series expansion around $\tilde{\mathbf{x}}$:

$$\mathbf{F}(\tilde{\mathbf{x}} + \epsilon) = \mathbf{F}(\tilde{\mathbf{x}}) + \epsilon \mathbf{F}(\tilde{\mathbf{x}}) \nabla^T|_{\mathbf{x}=\tilde{\mathbf{x}}} + \frac{\epsilon^2}{2} \nabla^2 \mathbf{F}(\tilde{\mathbf{x}}) \nabla^T|_{\mathbf{x}=\tilde{\mathbf{x}}} + \dots \quad (\text{C.1})$$

$$\approx \mathbf{F}(\tilde{\mathbf{x}}) + \mathbf{J}(\tilde{\mathbf{x}}) \epsilon \quad (\text{C.2})$$

$$= \tilde{\mathbf{y}} \quad (\text{C.3})$$

where we make the approximation in the second line by dropping the second order/Hessian and higher terms under the assumption the perturbation vector ϵ is small. This allows us to write $\mathbf{F}$ in the form of a local linear system: $\mathbf{y} = \mathbf{F}(\mathbf{x}) + \mathbf{J}(\mathbf{x}) \epsilon$. It is straightforward to see that if the entries of the Jacobian matrix $\mathbf{J}$ are larger, then small perturbations ϵ will cause larger changes in the output $\mathbf{y}$. To measure the magnitude of the Jacobian entries, we take the Frobenius norm:

$$\|\mathbf{J}(\mathbf{x})\|_F = \sum_n \sum_m \left(\frac{\partial \mathbf{F}_m(\mathbf{x})}{\partial \mathbf{x}_n} \right)^2 \quad (\text{C.4})$$

where $\mathbf{x}_n$ is the ' n 'th entry of $\mathbf{x}$ and $\mathbf{F}_m$ is the ' m 'th entry of the codomain of $\mathbf{F}$. The calculation of the Jacobian is trivial through the use of an automatic differentiation framework.

In our analysis we calculate the Jacobians of both agents on a fixed batch of 128 frame stacked images taken from the offline training dataset, and compare

the corresponding ratios of their Frobenius norms, and take this average ratio over the batch across 4 seeds.

C.3 Additional Analysis

C.3.1 Adaptive ND Dual Objective Optimization

The alternative ND score with increased outlier robustness, $\widetilde{ND}$, proposed in Section 4.2.4 is inspired by recordings of signal-to-noise ratio measurements. In particular, by passing the individual normalized $D(z)$ terms through a $\log(1+x)$ smoothing function we downweight the effect that large individual outliers might have on this aggregated metric. We would like to remark that since we set up the optimization of S with a dual objective, changes in the actual target value relating to some appropriate smoothness constraint are mostly irrelevant when considering the optimization’s dynamics. Therefore, we argue that tuning S with the actual ND should not considerably diverge from tuning S based on a re-scaled appropriate target for $\widetilde{ND}$.

We provide further plots comparing agent performance and respective adaptive parameter S during training:

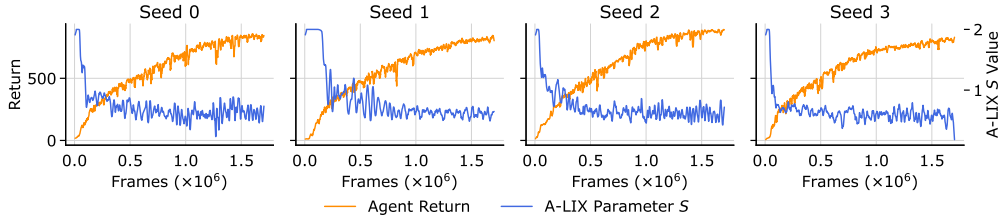


Figure C.8: Performance of agents across 4 different seeds of the Cheetah Run environment and their adaptive scalar parameter S . We observe that initially, S is high until agents learn useful behaviors, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.

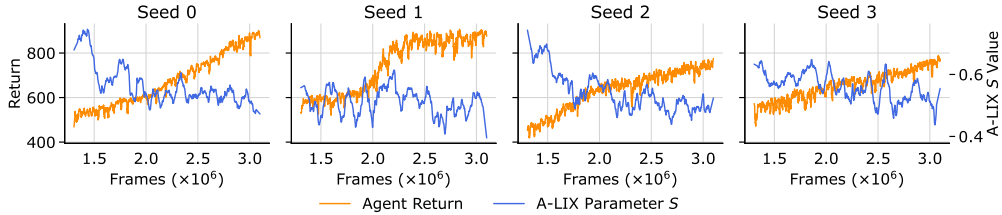


Figure C.9: Performance of agents across 4 different seeds of the Quadruped Run environment and their adaptive scalar parameter S . We observe that as meaningful behaviors are learned in agents towards the end of training, S falls accordingly, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.

We see the same effect in these two contrasting environments; in Cheetah Run, where learning is more stable due to more predictable initializations and fewer degrees of freedom, we see the A-LIX parameter S drop almost immediately as the TD-targets quickly become more accurate. In the less stable Quadruped Run, we also notice this annealing effect, however this occurs later on in training, when the agent can consistently recover from poor initializations.

C.3.2 N-Step Returns

Large n-step rewards have become an important part of many algorithms that use TD-learning from visual observations. As motivated in Section 4.2.3, large n-step rewards can help towards mitigating self-overfitting by densifying the reward and downweighting the contribution of the inaccurate target critic, especially early in training; indeed as shown in Yarats et al. (2022), using 1-step learning has a significant negative impact on performance. However, it is known that there is a bias-variance trade-off with multi-step approaches (Kearns & Singh, 2000), and furthermore, almost all approaches using this method do not apply off-policy bias correction when sampling from a replay buffer. While we motivate the use of n-step returns as a way to mitigate self-overfitting through incurring fewer 0 reward tuples (especially common in sparse reward environments early in training), we believe there is evidence to show that this introduces bias when n is sufficiently large, despite prior work suggesting this is not the case (Hernandez-Garcia & Sutton, 2019).

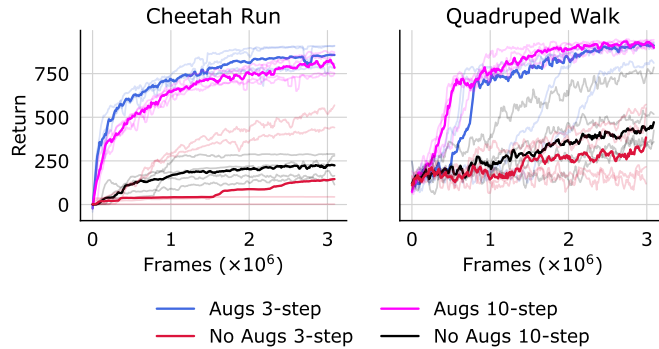


Figure C.10: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

We show in Fig. C.10 that 10-step (as is commonly done in algorithms used to solve Atari) returns can mitigate failure seeds as predicted under the visual deadly triad framework (indeed in Cheetah Run there are no seeds that completely flat-line when 10-step returns are used). However, we also see evidence that applying 10-step returns can have negative impacts on convergence and asymptotic performance in Cheetah Run when the deadly triad is sufficiently managed, such as using augmentations; in Quadraped Run we see moderate benefit initially, but note that asymptotically the 10-step and 3-step agents converge to the same performance. We also provide further evidence in App. C.5.3, where applying 10-step returns to an A-LIX agent generally has a largely negative impact on performance. Finally, we note that trying 20-step returns, as is done in some algorithms that solve Atari (Laskin et al., 2020), caused significant performance reductions in DMC. In conclusion, this provides evidence that we should consider using lower values of ‘n’ in multi-step returns, and achieve this through addressing other elements of the deadly triad.

C.4 Implementation Details

In Tables C.3 and C.4 we provide the full list of hyperparameters used in our implementations for DMC and Atari 100k, respectively. We show significant differences from standard practices in **bold**. In particular, A-LIX uses *the same encoder architecture and n-step returns* for both benchmarks, highlighting its lower reliance to environment-specific heuristics. Moreover, unlike prior state-of-the-art algorithms it *does not employ any data augmentation or auxiliary loss function*. These factors show the effectiveness of our adaptive method in counteracting instabilities from the visual deadly triad without any additional help, highlighting its applicability.

Table C.3: Full hyperparameters list used for the DeepMind Control A-LIX experiments. **Bolded** values represent significant differences from canonical implementations.

DDPG-integration hyperparameters (following (Yarats et al., 2022))	
Replay data buffer size	1000000 (100000 for <i>Quadruped Run</i>)
Batch size	256 (512 for <i>Walker Run</i>)
Minimum data before training	4000
Random exploration steps	2000
Optimizer	<i>Adam</i> (Kingma & Ba, 2014)
Policy/critic learning rate	medium: 0.0001 hard: 0.00008
Policy/critic β_1	0.9
Critic UTD ratio	0.5
Policy UTD ratio	0.5
Discount γ	0.99
Polyak coefficient ρ	0.99
N -step returns	3 (1 for <i>Walker Run</i>)
Hidden dimensionality	1024
Feature dimensionality	medium: 50 hard: 100
Nonlinearity	ReLU
Exploration stddev. clip	0.3
Exploration stddev. schedule	medium: linear: 1 $\rightarrow$ 0.1 in 500000 steps hard: linear: 1 $\rightarrow$ 0.1 in 2000000 steps
Augmentations	OFF
A-LIX-specific hyperparameters	
Initial maximum sampling shift S	1.0
Normalized discontinuity targets $\overline{ND}$	0.635
Maximum sampling shift learning rate	0.003
Maximum sampling shift β_1	0.5

Table C.4: Full hyperparameters list used for the Atari 100k A-LIX experiments. **Bolded** values represent significant differences from canonical implementations.

DER-integration hyperparameters	
Gray-scaling	True
Down-sampling	84×84
Frames stacked	4
Action repetitions	4
Reward clipping	$[-1, 1]$
Max episode frames	108000
Replay data buffer size	100000
Replay period every	1
Batch size	32
Minimum data before training	1600
Random exploration steps	1600
Optimizer	<i>Adam</i> (Kingma & Ba, 2014)
Critic learning rate	0.0001
Critic β_1	0.9
Critic ϵ	0.000015
Max gradient norm	10
Critic UTD ratio	2
Discount γ	0.99
Target update period	1
N-step returns	3
Feature maps	32, 32, 32
Filter sizes	$3 \times 3, 3 \times 3, 3 \times 3$
Strides	2, 1, 1
Hidden dimensionality	256
Feature dimensionality	50
Nonlinearity	ReLU
Exploration noisy nets parameter	0.1
Augmentations	OFF
A-LIX-specific hyperparameters	
Initial maximum sampling shift S	1.0
Normalized discontinuity targets $\overline{ND}$	0.75
Maximum sampling shift learning rate	0.0001
Maximum sampling shift β_1	0.5

C.5 Additional Ablations

C.5.1 Smoothness Regularization through Spectral Normalization

To distinguish between general smoothness constraints in convolutional features, and the smoothness that arises as a result spatial consistency, we apply spectral normalization (Miyato et al., 2018) to the final convolutional layer in the encoder to represent the former class of constraints. Spectral normalization operates on the parameters of a network and constrains its outputs to be 1-Lipschitz and has shown benefits in prior work (Gogianu et al., 2021), but does not explicitly enforce a *spatial regularization* in the features. We train agents without augmentations using spectral normalization.

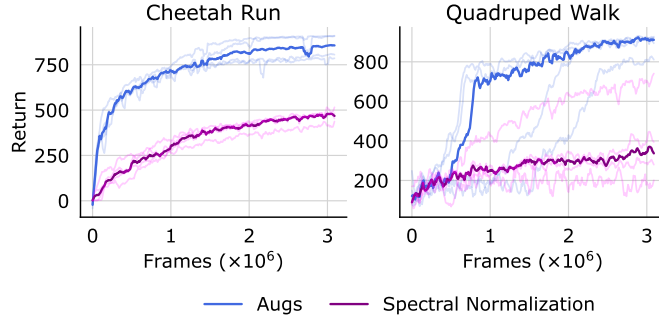


Figure C.11: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

We see that whilst there is clear improvement above the original non-augmented agents in some cases, the performance is still lower than agents that use spatial consistency regularization, such as random shift augmentations.

C.5.2 Is Gradient Smoothing All We Need?

Following the argument in Section 4.2.4, we can view augmentations as a gradient smoothing regularizer. This naturally leads us to ask the following: can we replace the stochastic shifting mechanism with a fixed smoothing mechanism? To test this, we instead apply a Gaussian smoothing kernel to the feature gradients in the CNN, and utilize our ND score to vary the width of the kernel adaptively through training; we call this method A-Gauss (**A**daptive **G**aussian Feature Gradient Kernel).

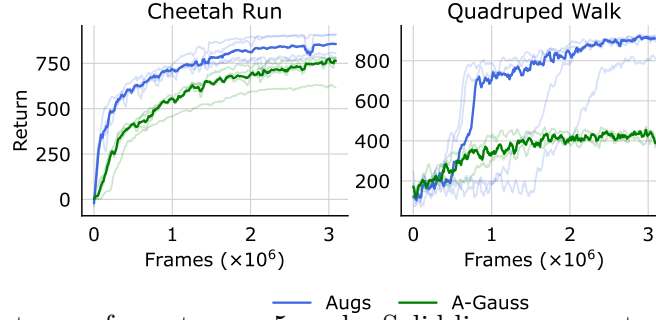
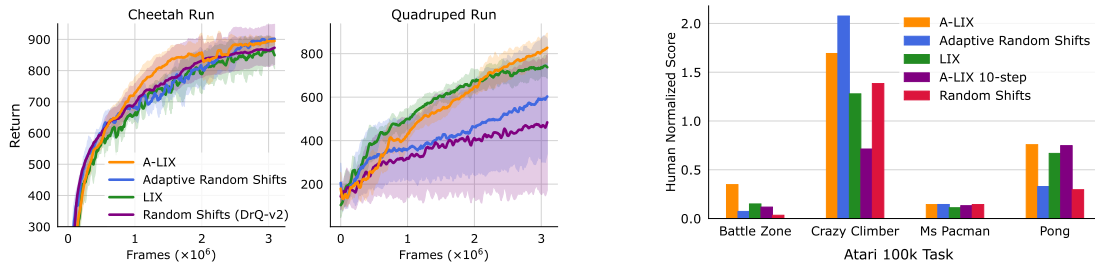


Figure C.12: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

We see that while there is improvement over non-augmented agents, overall performance is still lower than even simple non-adaptive augmentation. We believe this is due to the Gaussian kernel having too significant an effect on the information contained in the feature gradients during backpropagation, causing information to be lost. We believe this explains the effectiveness of shift-augmentations in reinforcement learning, which is that they effectively balance the information contained in the gradients, as well as ensuring their smoothness to reduce overfitting.

C.5.3 Ablations to A-LIX

We now provide a set of ablations on both DMC and Atari, assessing the impact of individual components in A-LIX.



(a) DMC Control ablations in Cheetah Run (left) and Quadruped Run (right) evaluated over 4 seeds.

(b) Atari 100k ablations evaluated over 4 seeds in 4 different Atari 100k tasks.

Figure C.13: An ablation study of A-LIX, showing the contribution of its individual components to ultimate performance in DMC and Atari 100k.

In Fig. C.13a we choose the following ablations for DMC:

- A-LIX

- Adaptive Random Shifts (where the magnitude of the random shift image augmentation is adjusted using the dual ND objective)
- LIX
- Random Shifts (i.e., DrQ-v2)

While we see a slight asymptotic performance improvement in Cheetah Run by using LIX layers instead of random shifts, we notice significant differences in the less stable Quadruped Run environment. Concretely, we see much greater stability in both LIX approaches compared with image augmentation approaches, with the former having no failure seeds. Furthermore, we observe stronger asymptotic performance with the inclusion of the adaptive dual objective for both approaches. As motivated in Fig. C.9, this is likely a result of reducing the shift parameter as the signal in the target values increases.

In Fig. C.13b, we choose the following ablations for Atari 100k on a subset of environments that represent a diverse set of tasks and performances with baseline algorithms:

- A-LIX
- Adaptive Random Shifts (as before)
- LIX
- A-LIX with 10-step returns
- Random Shifts

We see that A-LIX performs consistently strongly across the environments tested, always placing in the top 2 with regards to Human Normalized Score. We also notice that generally, LIX layer methods outperform random shift methods apart from in Crazy Climber, where the opposite is true. We believe this may be due to random shift augmentations actually reflecting the inductive biases concerning generalization in this environment, and believe this merits further investigation. Finally, we observe that using 10-step returns instead of 3 generally harms performance with A-LIX, with justification given in App. C.3.2.

C.6 Additional Offline Experiment Analysis

C.6.1 Behavior Cloning without Augmentations

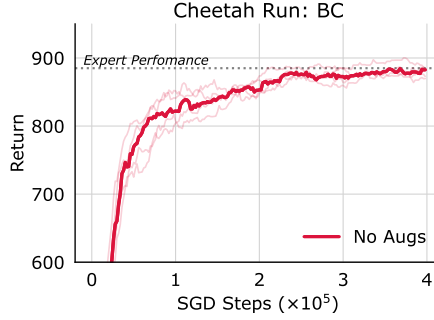


Figure C.14: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dotted horizontal line represents mean expert performance.

To illustrate that test time shift invariance is not required, we show that it is possible to learn a policy through supervised learning. To do this, we generate a pixel-based dataset of 500,000 timesteps under an expert policy in Cheetah Run, and jointly train a CNN encoder and policy using behavior cloning/supervised learning by minimizing the loss $\mathcal{L} = (a - \pi(o))^2$ until convergence, where o follows the stacked frame image inputs of (Mnih et al., 2013). We see that the pixel-based policy performs as well as the behavior agent, despite using both higher dimensional data and fewer than half the samples compared to existing expert offline RL benchmarks from *proprioceptive* states (Fu et al., 2021).

This provides clear evidence that shift invariance is not required at test time, and motivates us to find an alternative explanation for why random shift augmentations help the learning process in TD-learning. An alternative perspective is that when the learning signal is strong, as is the case for supervised learning (and later stages during online learning when target values are more accurate), the natural bias of CNNs to learn lower order representations acts as an implicit regularizer (Rahaman et al., 2019) that results in test-time generalization.

C.6.2 Turning Off Augmentations

We present more evidence showing that augmentations benefit learning the most at the beginning of training. In Fig. C.15 we show the effect of turning off

augmentations at 200,000 steps in Cheetah Run, and at 500,000 in Quadruped Walk. In both instances, we see large improvements over not augmenting at all, and both nearly converge to the same value as DrQ-v2, showing further evidence that stability initially in learning is vital. We posit that turning off augmentations here did not yield similar benefits to Fig. 4.5 due to the fact that there is still high-frequency information in the targets (consider that the augmentations in Cheetah Run are switched off significantly earlier) that cause a marginal amount of self-overfitting, reducing the rate of learning due to feature space degeneration.

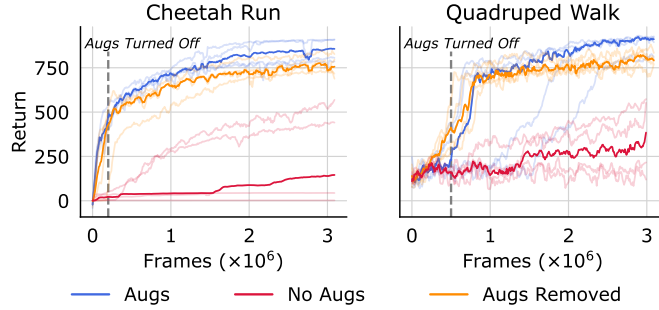


Figure C.15: Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dashed line shows when augmentations are turned off.

C.6.3 Action-Value Surfaces

Here we show the action-value surfaces of the offline agents’ critics at various tuples sampled from the data. This provides us with an intuition over the loss landscape that the policies will be optimizing during the policy improvement, as accordingly the policy under the deterministic policy gradient (Silver et al., 2014) updates its own weights towards maximizing the action-values defined by the critic through the chain rule:

$$\nabla_{\phi} J_{\pi} \approx \mathbb{E}_{s \sim E} \left[\nabla_a Q_{\theta}(s, a)|_{a=f_{\phi}(s)} \nabla_{\phi} f_{\phi}(s) \right] \quad (\text{C.5})$$

where ϕ and θ are policy and critic weights respectively. We hypothesize that self-overfitting reduces the sensitivity of the critic to actions, discarding important information regarding the causal link between actions and expected returns. To evaluate this, we sample state-action pairs from our replay buffer, and then visualize the action-value surface by sampling two random *orthogonal* direction vectors from

the action space A . We then normalize the direction vectors to have a 2-norm of 1, and then multiply each direction vector by scalars $\alpha, \beta \in [-2, 2]$ respectively. We then plot the action-value surface as a result of adding the random vectors multiplied by their respective scalars onto the action sampled from offline dataset, giving us a 3-D surface. We clip actions to $a \in [-1, 1]^{|A|}$ as actions are squashed to this range in the policy through a truncated normal distribution.

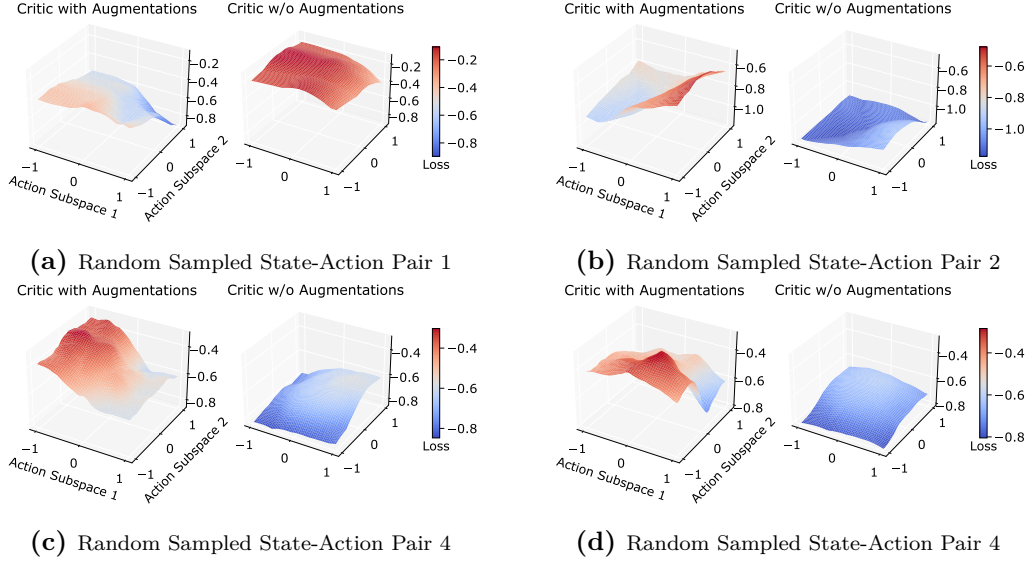


Figure C.16: Action-Value loss surface plotted with respect two orthogonal random directions sampled from the action space (i.e., $d_r \in A$ and $d_1 \perp d_2$).

We see that the critics learned by the augmented agents are more sensitive to changes in action. We believe this is due to the non-augmented agents overfitting to the observations, thus ignoring the lower-dimensional action inputs. To validate this, we sampled 128 random state-action tuples from the offline buffer, and calculated the average variance across the loss surfaces. We see a significant difference, with the augmented agent having an average loss surface variance of **0.0129**, whereas the non-augmented agent has an average loss surface variance of **0.0044**, suggestive of lower sensitivity.

C.6.4 Evidence of Critic MLP Overfitting from High-Frequency Features

We provide further evidence that measuring high-frequency features through the ND score is vital to understanding overfit by showing how overfitting is able to occur

in the fully-connected critic layers, which are usually stable under proprioceptive observations (see Table 4.5). To do this, we construct a pattern containing high frequency checkerboard noise $c \in \mathbb{R}^{H \times W}$, and produce as many patterns as there are channels C in the final layer. To ensure consistency across each individual feature map, we normalize each checkerboard pattern by the maximum value in its respective feature map, and then divide by the width of the checkerboard. We then add this pattern multiplied by a scalar α onto each feature map.

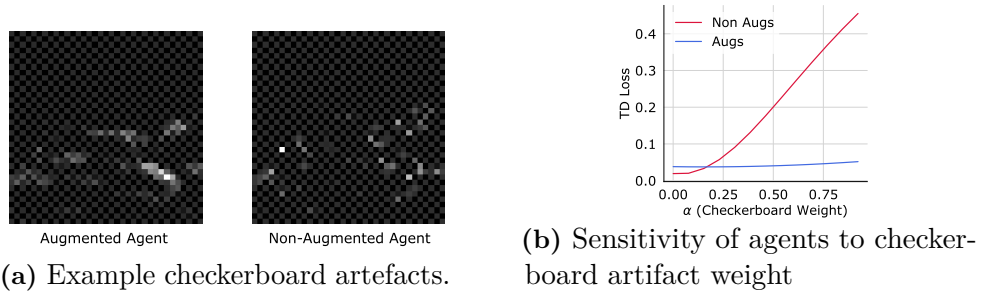


Figure C.17: Effect of checkerboard artifacts on feature maps and resultant loss sensitivity. We see the non-augmented agent is significantly more sensitive to this high-frequency noise.

As we see, the loss is significantly more sensitive to high-frequency perturbations in the non-augmented agent, justifying its reliance on high-frequency patterns in the feature maps to enable self-overfitting.

C.6.5 Additional Loss Surfaces

Here we show the loss surfaces of the offline agents under policy evaluation with at 1,000, 5,000, and 10,000 training steps. We also show the surfaces respect to only the MLP layers, again following the normalization approach of Li et al. (2018).

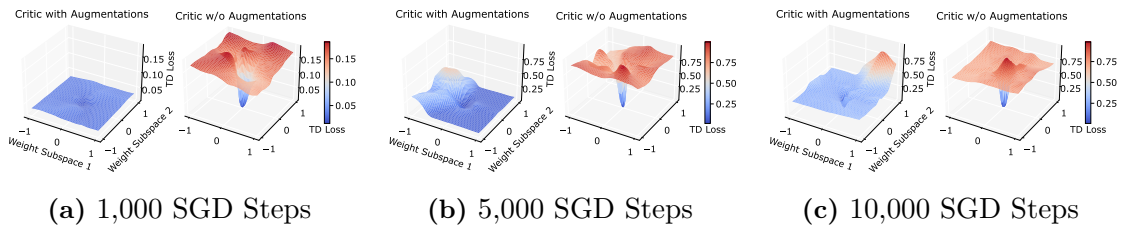


Figure C.18: Loss surface plotted with respect to *Encoder* parameters at various stages of training.

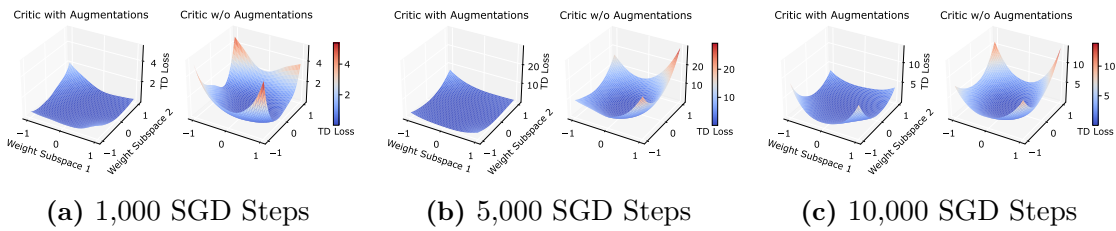


Figure C.19: Loss surface plotted with respect to *Critic MLP* parameters at various stages of training.

As we see, the loss surface with respect to the MLP parameters is significantly less sharp, lending further evidence that self-overfitting is predominately a result of the flexibility of the CNN layers to learn high-frequency features.

Chapter Bibliography

- Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.
- David Brandfonbrener, William F Whitney, Rajesh Ranganath, and Joan Bruna. Offline RL without off-policy evaluation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=LU687itn08w>.
- Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2021.
- Florin Gogianu, Tudor Berariu, Mihaela C Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: An optimisation perspective. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3734–3744. Pmlr, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/gogianu21a.html>.
- Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565. Pmlr, 2019.
- J. Fernando Hernandez-Garcia and Richard S. Sutton. Understanding multi-step deep reinforcement learning: A systematic study of the DQN target, 2019.
- Michael J. Kearns and Satinder P. Singh. Bias-variance error bounds for temporal difference updates. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, *Colt '00*, pp. 142–147, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 155860703x.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

- Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Neural Information Processing Systems*, 2018.
- Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.
- H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947. doi: 10.1214/aoms/1177730491. URL <https://doi.org/10.1214/aoms/1177730491>.
- Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=B1QRgziT->.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
- Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. Pmlr, 2019.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. 2014.
- Student. The probable error of a mean. *Biometrika*, 6(1):1–25, 1908. ISSN 00063444. URL <http://www.jstor.org/stable/2331554>.
- Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6): 80–83, 1945. ISSN 00994987. URL <http://www.jstor.org/stable/3001968>.
- Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

D

Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment: Appendix

D.1 Additional Experiments

In this section we show the performance for Augmented World Models with different training ranges for the DAS augmentation (z train in Table D.2). We train with adaptive context on the HalfCheetah mixed dataset, and present the results in Fig. D.1. As we see, $[0.75, 1.25]$ and $[0.5, 1.5]$ perform the best. Based on this, we use $[0.5, 1.5]$ for our experiments as we believe this helps us sample a wider set of dynamics, helping us generalize better across all environments and data sets.

D.2 Implementation Details

D.2.1 Hyperparameters

Our algorithm is based on MOPO (Yu et al., 2020) with values for the rollout length h and penalty coefficient λ shown in Table D.1.

¹We follow the original MOPO hyperparameters for all datasets except for walker2d-medium where we found (1, 1) worked better for both MOPO and our method than (5, 5).

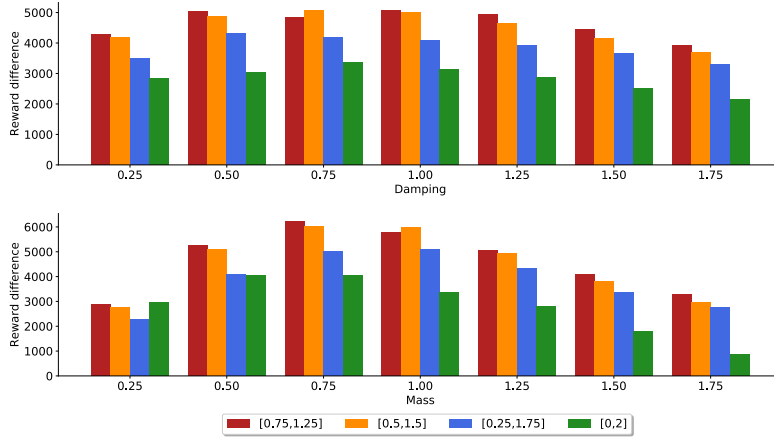


Figure D.1: Performance for Augmented World Models with the DAS augmentation. Each plot shows different values for a and b , the ranges for the sampled noise.

Table D.1: Hyperparameters used in the D4RL datasets.

Dataset Type	Environment	MOPO (h, λ)
random	halfcheetah	5, 0.5
random	walker2d	1, 1
medium	halfcheetah	1, 1
medium	walker2d	1, 1 ¹
mixed	halfcheetah	5, 1
mixed	walker2d	1, 1
med-expert	halfcheetah	5, 1
med-expert	walker2d	1, 2

AugWM specific hyperparameters are listed in Table D.2. For each evaluation rollout, we clear the buffer of stored true modified environment transitions to measure zero-shot performance. We adapt using the context after a set number of steps, k , in the environment to train the linear model. The two ranges used for the context z during training and test time are different. At test time, the estimated context is clipped to remain within the given bounds.

Table D.2: AugWM Hyperparameters

Parameter	Value
evaluation rollouts	5
MOPO offline epochs	400
AugWM offline epochs	900
k , steps for adaptation	300
z train range	[0.5, 1.5]
z test range	[0.93, 1.07]

D.2.2 D4RL dataset

We evaluate our method on D4RL (Fu et al., 2020) datasets based on the MuJoCo continuous control tasks (halfcheetah and walker2d). The four dataset types we evaluate on are:

- **random:** roll out a randomly initialized policy for 1M steps.
- **medium:** partially train a policy using SAC, then roll it out for 1M steps.
- **mixed:** train a policy using SAC until a certain (environment-specific) performance threshold is reached, and take the replay buffer as the batch.
- **medium-expert:** combine 1M samples of rollouts from a fully-trained policy with another 1M samples of rollouts from a partially trained policy or a random policy.

This gives us a total of 8 experiments.

D.2.3 Ant Environment

For the Ant experiments, we follow the Ant Changed Direction approach in MOPO (Yu et al., 2020). Since this offline dataset is not provided in the authors’ code, nor is it in the standard D4RL library (Fu et al., 2020), we were required to generate our own offline Ant dataset. Since the authors’ did not outline certain details in their experiment, we found the following was required to match their performance with our codebase: 1) Training our SAC policy for 1×10^6 timesteps in the Ant

environment provided by the authors’ code in [Yu et al. \(2020\)](#); 2) relabelling each reward in the buffer using the new direction, without the living reward; 3) training a world model over this offline dataset; 4) training a policy in the world model, adding in living reward post-hoc; 5) evaluating the policy with the living reward.

D.2.4 HalfCheetah Modified Agent

We use the modified HalfCheetah environments from [Henderson et al. \(2017\)](#). In each setting one body part of the agent is changed, from following set: {Foot, Leg, Thigh, Torso, Head}. The body part can either be “Big” or “Small”, where Big bodyparts involve scaling the mass and width of the limb by 1.25 and Small bodyparts are scaled by 0.75. In [Table 5.2](#) we show the mean over each of these five body parts, for agents trained on each of the D4RL datasets, repeated for five seeds.

Chapter Bibliography

- Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning, 2020.
- P. Henderson, W.-D. Chang, F. Shkurti, J. Hansen, D. Meger, and G. Dudek. Benchmark environments for multitask learning in continuous domains. *ICML Lifelong Learning: A Reinforcement Learning Approach Workshop*, 2017.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, 2020.

E

Learning General World Models in a Handful of Reward-Free Deployments: Appendix

E.1 Additional Experimental Results

E.1.1 DMC

Here we show additional experimental results, beginning with a full breakdown of the DMC results for each dataset. In Figure [E.1](#) we see the zero-shot transfer performance for models trained with `random`, `medium` and `expert` initial datasets, and 14 subsequent deployments. Interestingly, we see that the `medium` dataset proves to be the most effective for achieving high performance, this may be due to being more diverse than the `expert` dataset, which is a narrow distribution of high performing episodes.

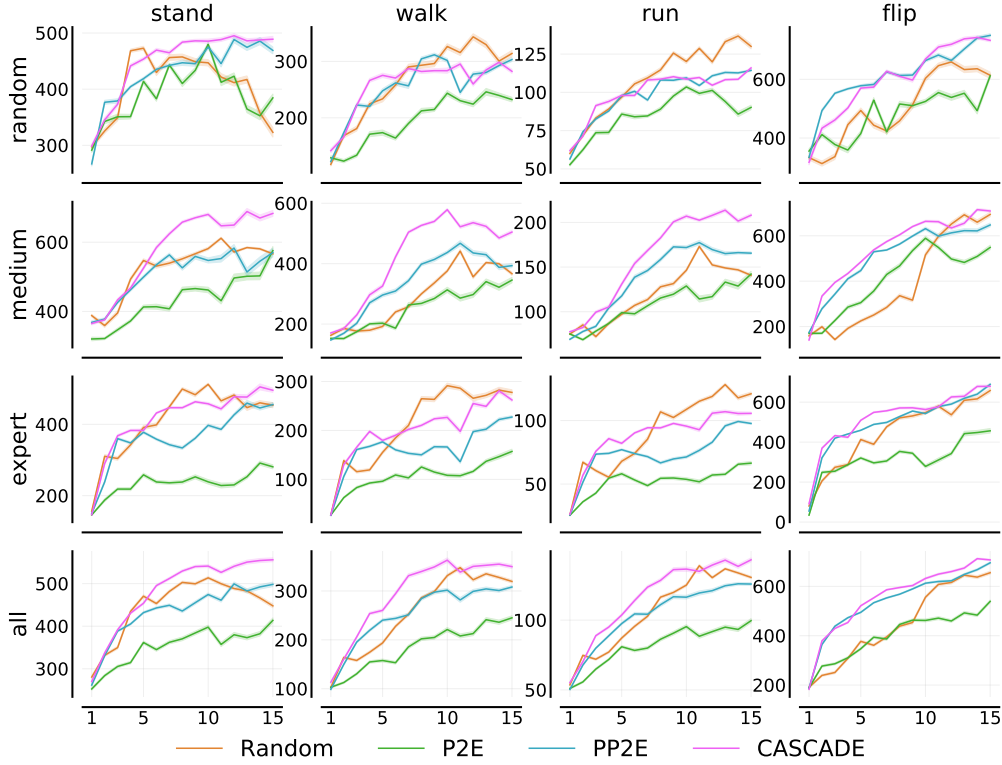


Figure E.1: DMC Zero-Shot Results: Plots show the performance by environment and dataset, with the last row being the task performance average over all three initial datasets. Plots show the mean and SEM over 10 seeds.

E.1.2 Atari Tasks

Next we show detailed results in four Atari games: Montezuma’s Revenge, Frostbite, Hero and Freeway in Fig. E.2. We see that generally CASCADE discovers higher rewarding episodes (almost double the next best baseline in Montezuma’s Revenge), and also discovers more rewarding episodes on average. This translates to stronger zero-shot performance than all baselines in all four environments. On the other hand, the other baselines achieve worse zero-shot performance because they either (1) find an occasional high reward episode (Max Episode Return), but not in a sufficient quantity (Rewarding Episodes), or (2) collect abundant low rewarding episodes that are less helpful for learning good behavior policies. For instance, the random baseline is able to find Rewarding Episodes more frequently in Frostbite, but these are relatively low quality, hence the lower curve when assessing its Max Episode Return.

To understand why CASCADE performs well, in Figure E.3 we plot trajectories

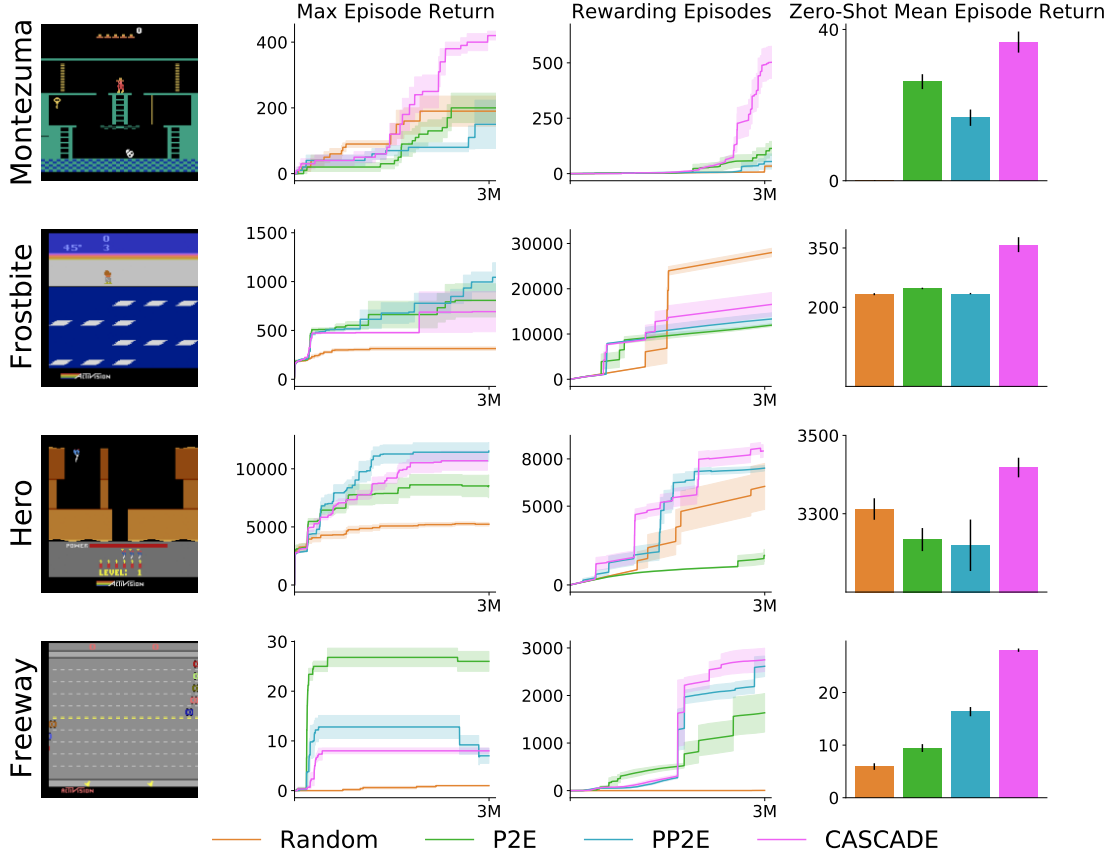


Figure E.2: From left to right on each row we show a frame from the game, a plot of the cumulative maximum episode reward, the total number of rewarding episodes discovered and zero-shot average return, from 3M training steps (15 deployments). Plots show mean and SEM over 10 seeds. Note that the max episode returns of all methods stabilize after 15 deployments (3M training steps in total).

from Montezuma’s Revenge. We see that the inclusion of a diversity-inducing objective does indeed lead to more diverse behaviors in the environment, with each policy exploring different regions of the room. In contrast, for PP2E, agents 3,4,5,6,7,9 all exhibit very similar behavior, likely indicating that random initialization alone is not sufficient to induce diversity when all agents optimize for the same objective.

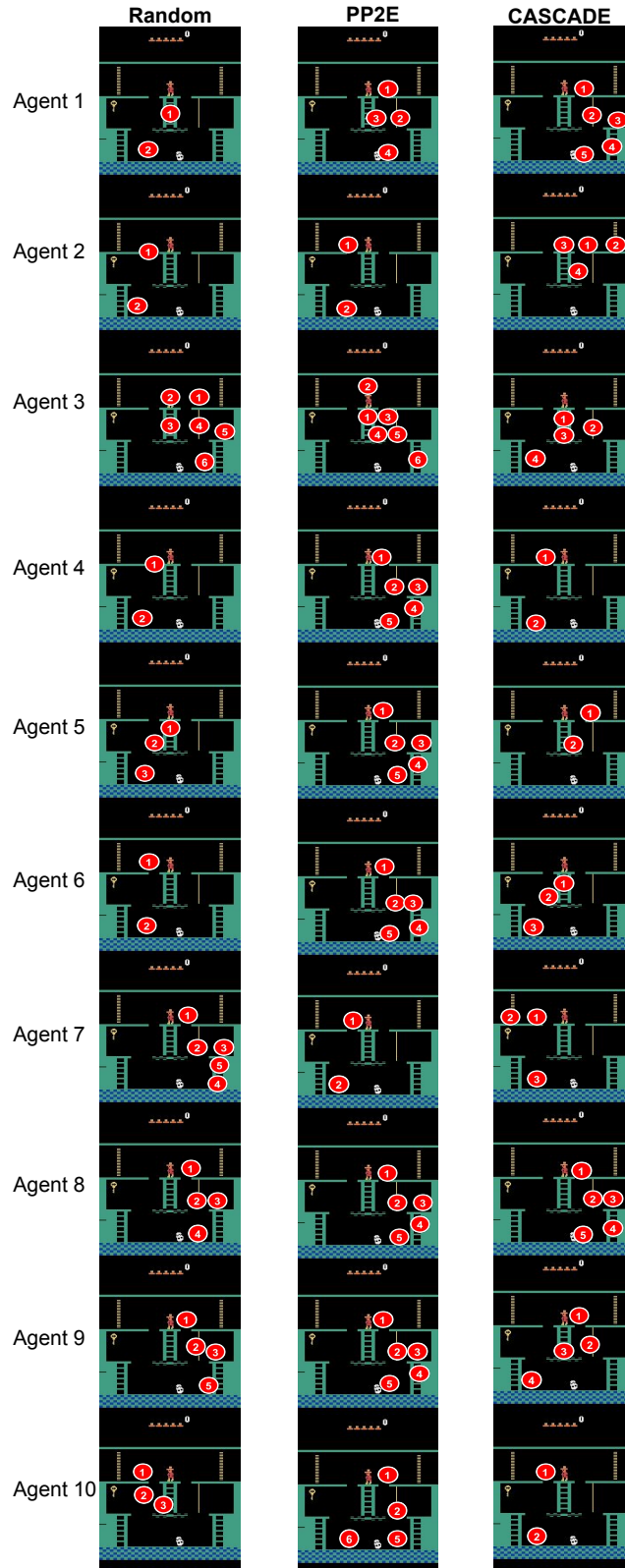


Figure E.3: Trajectories in Montezuma's Revenge: Each row shows a trajectory from one of the 10 exploration agents of a model after 3M training timesteps (P2E is omitted because it only has 1 agent.) We can see that PP2E's agents exhibit the most homogeneous behaviors, and result in trajectories that focus on the bottom right of the room, while CASCADE agents manage to explore more of the room collectively.

E.2 Implementation Details

DreamerV2 consists of an image encoder that uses a Convolutional Neural Network (CNN, (LeCun et al., 1989)), a Recurrent State-Space Model (RSSM (Hafner et al., 2019)) that learns the dynamics, and predictors for the image, reward, and discount factor. The RSSM uses a sequence of deterministic recurrent states h_t . At each step, it computes a posterior state z_t conditioned on the current image x_t , as well as a prior state $\hat{z}_t$ without the current image. During world model training, the concatenation of h_t and z_t is used to reconstruct the current image x_t , and predict the reward r_t and discount factor γ_t . Once the world model is trained, it can be used to roll out “imaginary” trajectories where the model state is instead the concatenation of the deterministic state h_t and prior stochastic state $\hat{z}_t$.

$$\text{RSSM} \begin{cases} \text{Recurrent model:} & h_t = f_\psi(h_{t-1}, z_{t-1}, a_{t-1}) \\ \text{Representation model:} & z_t \sim q_\psi(z_t | h_t, x_t) \\ \text{Transition predictor:} & \hat{z}_t \sim p_\psi(\hat{z}_t | h_t) \end{cases} \quad (\text{E.1})$$

$$\begin{aligned} \text{Image predictor:} & \quad \hat{x}_t \sim p_\psi(\hat{x}_t | h_t, z_t) \\ \text{Reward predictor:} & \quad \hat{r}_t \sim p_\psi(\hat{r}_t | h_t, z_t) \\ \text{Discount predictor:} & \quad \hat{\gamma}_t \sim p_\psi(\hat{\gamma}_t | h_t, z_t). \end{aligned} \quad (\text{E.2})$$

Our implementation was built on top of the official DreamerV2 repository (<https://github.com/danijar/dreamerv2>). We used the default hyperparameter values in the DreamerV2 repository. Table E.1 lists the additional hyperparameters used in our experiments.

We also incorporate a latent disagreement ensemble, following Plan2Explore (Sekar et al., 2020). This involves training, alongside the RSSM model, an MLP ensemble with 10 members (each having 4 hidden layers, 400 units per layer, and different parameter initializations). These one-step models take action a_t and latents z_t , h_t as inputs, and try to predict the next stochastic latent z_{t+1} . The variance across these outputs during imagined rollouts is then used as the reward signal that forms the Information Gain (InfoGain) objective. Note that this ensemble is otherwise unused, and is solely trained for the purposes of the exploration objective.

Table E.1: CASCADE hyperparameters

Environment	Parameter	Value
MiniGrid-FourRooms	CASCADE weight (λ)	0.1
	batch size	200k
	deployments	5
	explorer train steps	10k
	offline model train steps	20k
MiniGrid-MultiRoom-N4S5	CASCADE weight (λ)	0.3
	batch size	200k
	deployments	5
	explorer train steps	10k
	offline model train steps	20k
Montezuma's Revenge	CASCADE weight (λ)	0.8
	batch size	200k
	deployments	15
	explorer train steps	10k
	offline model train steps	5k
Walker	CASCADE weight(λ)	0.1 (Random) 0.3 (Medium/Expert)
	batch size	200k
	deployments	2
	explorer train steps	250
	offline model train steps	5k

E.3 Deriving the Objective

E.3.1 An Information Theoretic Perspective

Inspired by [Sekar et al. \(2020\)](#), we derive a data acquisition objective based on the mutual information between the collected data and parameters of the world model $\mathcal{M}_\psi$. $\mathcal{M}_\psi$ can be either stochastic or deterministic, and can also be an ensemble of empirical models. Crucially however, we eschew the reliance on one-step information gain (as performed in [Sekar et al. \(2020\)](#)), and instead aim to maximize diversity directly in the space of trajectories. To do this, let $\Phi : \Gamma \rightarrow \Omega$ be an summary function mapping trajectories into an embedding ‘summary’ space. For any model $\mathcal{M}_\psi$ we use the notation $\mathbb{P}_\pi[\mathcal{M}_\psi]$ to denote the distribution over trajectories generated by policy π in $\mathcal{M}_\psi$ and use $\mathcal{M}_\psi \sim \mathbb{P}_{\mathcal{M}_\psi}$ to denote the posterior distribution over models. The later depends on the data collected so far.

In this work we consider different embedding functions Φ ,

1. **Final State Embedding.** In this setting we define $\Phi(\tau) = h_H$ where h_H corresponds to the H -th (and last) state embedding in the trajectory τ . Since we use an RNN world model, the use of this summary embedding is analogous to the final encoder embedding in seq2seq language models [Sutskever et al. \(2014\)](#).
2. **Visitation Frequencies.** In the tabular setting if we define $\Phi(\tau)$ as a discounted count statistic for states in τ , we recover $\mathbb{P}_\pi^\Phi[\mathcal{M}_\psi] = d_{\mathcal{M}_\psi}^\pi$ where $d_{\mathcal{M}_\psi}^\pi$ corresponds to the discounted visitation distribution of policy π in model $\mathcal{M}_\psi$.

The authors of [Sekar et al. \(2020\)](#) study a per state-action mutual information objective that informs the construction of a greedy mutual information maximizing policy. Implicitly, this per-step objective assumes the model to factor in *independent* transition operators pertaining to each state-action pair. This is certainly not the case when using powerful function approximators such as neural networks. In this case, the dynamics model corresponding to the state transitions of a specific

state action pair is correlated with the state transition model of *other* state-actions. This is not captured accurately by the objective in [Sekar et al. \(2020\)](#). Instead, we consider the following choice for an exploratory policy in the single policy setting studied in [Sekar et al. \(2020\)](#):

$$\pi_{\text{EXP}} = \max_{\pi} \mathcal{I} \left(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]; \mathcal{M}_{\psi} \right) = \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]) - \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi}). \quad (\text{E.3})$$

π_{EXP} is a policy whose distribution over summaries $\Phi(\tau)$ with $\tau \sim \mathbb{P}_{\pi}[\mathcal{M}_{\psi}]$ and $\mathcal{M}_{\psi} \sim \mathbb{P}_{\mathcal{M}_{\psi}}$ has large entropy, but such that the average entropy of the summaries in every individual model is small. The term $\mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}])$ captures the total uncertainty (epistemic + aleatoric) while subtracting the conditional entropy removes the aleatoric uncertainty resulting from noise within the model and the policy.

To gain intuition about this objective, consider the case where Φ equals the *final state embedding*. For simplicity, we consider the scenario when all models $\mathcal{M}_{\psi} \sim \mathbb{P}$ and all policies $\pi \sim \Pi$ are such that for any realization of $\mathcal{M}_{\psi} = m_{\psi}$, the conditional entropy $\mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi} = m_{\psi}) = \sigma(m_{\psi})$ is a function of the world m_{ψ} and not of the policy. In this case, the policy π_{EXP} is the entropy maximizing policy:

$$\pi_{\text{EXP}} = \max_{\pi} \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}]).$$

The scenario above is realized for example when all models $\mathcal{M}_{\psi} \sim \mathbb{P}_{\mathcal{M}_{\psi}}$ are deterministic and all policies in Π are deterministic. In this case, for every realization of $\mathcal{M}_{\psi} = m_{\psi}$, the conditional entropy $\mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi} = m_{\psi}) = 0$. The assumption $\mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}] | \mathcal{M}_{\psi} = m_{\psi}) = \sigma(w)$ is also realized when the distribution $\Phi(\tau) \sim \mathbb{P}_{\pi}^{\Phi}[m_{\psi}]$ is approximated as a Gaussian distribution $\mathbb{P}_{\pi}^{\Phi}[m_{\psi}] \approx \mathcal{N}(\mu(m_{\psi}, \pi), \Sigma(m_{\psi}))$ with policy dependent mean $\mu(m_{\psi}, \pi)$ and policy independent covariance $\Sigma(m_{\psi})$.

In these cases π_{EXP} is the policy that produces the most diverse distribution over final states across the posterior distribution over models $\mathbb{P}_{\mathcal{M}_{\psi}}$. Since access to $\mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}])$ may not be possible, in our experiments we make further approximations inspired by [Sekar et al. \(2020\)](#). Under the same gaussian approximation $\mathbb{P}_{\pi}^{\Phi}[m_{\psi}] \approx \mathcal{N}(\mu(m_{\psi}, \pi), \Sigma(m_{\psi}))$, optimizing $\max_{\pi} \mathcal{H}(\mathbb{P}_{\pi}^{\Phi}[\mathcal{M}_{\psi}])$ is achieved by finding the policy

π that makes the ensemble means $\mu(m_\psi, \pi)$ as far apart as possible. A suitable surrogate for this objective is to maximize the empirical variance over means,

$$\text{Var}(\pi) = \frac{1}{|\mathcal{M}_\psi| - 1} \sum_{m_\psi} \|\mu(m_\psi, \pi) - \mu'(\pi)\|^2$$

where $|\mathcal{M}_\psi|$ denotes the number of samples m_ψ , and $\mu'(\pi) = \frac{1}{|\mathcal{M}_\psi|} \sum_{m_\psi} \mu(m_\psi, \pi)$.

We now consider a “batch” version of Eq. E.3 involving a population of B agents:

$$\begin{aligned} \{\pi_{\text{EXP}}^{(i)}\}_{i=1}^B &= \max_{\pi^{(1)}, \dots, \pi^{(B)} \in \Pi^B} \mathcal{I} \left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi]; \mathcal{M}_\psi \right) = \\ &\quad \mathcal{H} \left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \right) - \mathcal{H} \left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi \right) \end{aligned} \quad (\text{E.4})$$

where $\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi]$ is the product measure of the candidate policies embedding distributions over the model $\mathcal{M}_\psi$. By definition the conditional entropy factors as,

$$\mathcal{H} \left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi \right) = \sum_{i=1}^B \mathcal{H} \left(\mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi \right). \quad (\text{E.5})$$

The objective in Eq. E.4 has a similar interpretation as in the single policy case. We are hoping to find a set of policies whose average conditional entropies are small (Eq. 5.8), but whose total entropies are large. Intuitively, we see this objective is more amenable to our population of policies. Concretely, by considering distributions directly over the space of trajectories, we ensure that each agent does not ‘double count’ the explored states under Eq. E.4, whereas applying the same principal to a one-step information gain objective would simply ensure diversity *conditioned* on that state and action, and doesn’t explicitly ensure diversity in the visited states by the population.

Proof of Lemma 1

In the proof of Lemma 1 we will make use of the following supporting result,

Lemma 3. *Let $p \in (0, 1]$ and $p_1, p_2 > 0$ satisfying $p_1 + p_2 = p$ then,*

$$p \log(1/p) < p_1 \log(1/p_1) + p_2 \log(1/p_2).$$

Proof. Let $p_1 = \alpha p$ with $\alpha \neq 0, 1$, then,

$$\begin{aligned} p_1 \log(1/p_1) + p_2 \log(1/p_2) &= \alpha p \log(1/(\alpha p)) + (1 - \alpha)p \log(1/((1 - \alpha)p)) \\ &= p \log(1/p) + p \underbrace{\left(\alpha \log(1/\alpha) + (1 - \alpha) \log(1/(1 - \alpha)) \right)}_{>0} \\ &> p \log(1/p). \end{aligned}$$

□

Lemma 3 implies the following result,

Lemma 4. Let $\mathbf{p} \in [0, 1]^K$ be a vector satisfying $\sum_{i=1}^K \mathbf{p}_i = 1$ and let $\mathcal{C}$ be a partition of $[K]$ such that $|\mathcal{C}| \leq K - 1$. For all $C \in \mathcal{C}$ denote $\mathbf{p}(C) = \sum_{i \in C} \mathbf{p}_i$. The following inequality holds,

$$\sum_{C \in \mathcal{C}} \mathbf{p}(C) \log(1/\mathbf{p}(C)) \leq \sum_{i=1}^K \mathbf{p}_i \log(1/\mathbf{p}_i).$$

If $\mathbf{p}_i > 0$ the inequality is strict,

$$\sum_{C \in \mathcal{C}} \mathbf{p}(C) \log(1/\mathbf{p}(C)) < \sum_{i=1}^K \mathbf{p}_i \log(1/\mathbf{p}_i).$$

Proof. W.l.o.g we call C_1 the partition set containing 1 and assume $\mathbf{p}_1 > 0$. If $\mathbf{p}_i > 0$ for only one element of C_1 , it must be the case that $\mathbf{p}_1 \log(1/\mathbf{p}_1) = \mathbf{p}_{C_1} \log(1/\mathbf{p}_{C_1})$.

We now assume that $|C_1| > 1$ and that C_1 has at least two indices $i \neq j$ such that $\mathbf{p}_i, \mathbf{p}_j > 0$. Let l denote the size of C_1 and w.l.o.g. let's say $C_1 = \{1, 2, \dots, l\}$.

Lemma 3 implies the following inequalities,

$$\begin{aligned} \mathbf{p}(C_1) \log(1/\mathbf{p}(C_1)) &< \mathbf{p}_1 \log(1/\mathbf{p}_1) + \mathbf{p}(C_1 \setminus \{1\}) \log(1/\mathbf{p}(C_1 \setminus \{1\})) \\ &< \dots \\ &< \sum_{i \in C_1} \mathbf{p}_i \log(1/\mathbf{p}_i). \end{aligned}$$

Applying this reasoning to each element in the partition $\mathcal{C}$ yields the desired result.

□

Lemma 1. *When all models $\mathcal{M}_\psi$ in the support of the model posterior are deterministic and tabular, and the space of policies Π consists only of deterministic policies, there always exists a solution $\{\pi_{\text{EXP}}^{(i)}\}_{i=1}^B$ satisfying $\pi_{\text{EXP}}^{(i)} \neq \pi_{\text{EXP}}^{(j)} \forall i \neq j$. Moreover, there exists a family of tabular MDP models, such that the maximum cannot be achieved by setting $\pi_{\text{EXP}}(i) = \pi$ for a fixed π .*

Proof. First let's observe that when the models and policies are all deterministic, the conditional entropies satisfy,

$$\mathcal{H} \left(\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi \right) = \sum_{i=1}^B \mathcal{H} \left(\mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi \right) = 0.$$

We assume the set of policies Π is of size at least B . Otherwise, the result cannot be true. The first result follows immediately by Lemma 4. For any fixed π , the product distribution $\prod_{i=1}^B \mathbb{P}_\pi^\Phi[\mathcal{M}_\psi]$ is a distribution over ‘diagonal’ tuples of the form $\underbrace{(b, \dots, b)}_{\text{size } B}$ where b is an embedding.

Consider an arbitrary set of policies $\pi^{(2)}, \dots, \pi^{(B)}$ satisfying $\pi^{(i)} \neq \pi^{(j)} \neq \pi$ for all $i, j \in \{2, \dots, K\}$. Call $\pi^{(1)} = \pi$. The distribution over embeddings induced by the product distribution $\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi]$ is made of tuples of the form $(b_1, \dots, b_B)$. Call $\mathbf{p}(b_1, \dots, b_B)$ the probability under the product measure $\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi]$ of the tuple $(b_1, \dots, b_B)$. Notice the ‘projection measure’ onto the first coordinate satisfies

$$\prod_{i=1}^B \mathbb{P}_{\pi^{(i)}}^\Phi[\mathcal{M}_\psi](b_1) = \prod_{i=1}^B \mathbb{P}_\pi^\Phi[\mathcal{M}_\psi](b_1, \dots, b_1) := \mathbf{p}_1(b_1)$$

and that $\sum_{b_2, \dots, b_K} \mathbf{p}(b_1, b_2, \dots, b_K) = \mathbf{p}_1(b_1)$. This induces a partition over the outcome space $(b_1, \dots, b_K)$ corresponding to $C(b_1) = \{(b_1, b_2, \dots, b_K)\}_{b_2, \dots, b_K}$.

A direct use of Lemma 4 implies the entropy of the product distribution over finer grained tuples is larger than the entropy over the diagonal of the product distribution having all coordinates equal to each other. In fact this result also informs when the inequality will be strict. This happens when (for example the measure over $\mathcal{M}_\psi$ is the counting measure) there exists two worlds $\mathcal{M}_\psi^{(1)}$ and $\mathcal{M}_\psi^{(2)}$ such that the embedding tuples $(b_1^{(1)}, \dots, b_B^{(1)})$ and $(b_1^{(2)}, \dots, b_B^{(2)})$ satisfy $b_1^{(1)} = b_1^{(2)}$ and $(b_2^{(1)}, \dots, b_B^{(1)}) \neq (b_2^{(2)}, \dots, b_B^{(2)})$.

We will use this observation to prove the second claim. Consider a family of MDPs formed of depth L binary trees. In this family, the paths leading to the $L - 1$ layer are the same, but the connections from layer $L - 1$ to layer L are unknown. The ‘posterior’ distribution is assumed to be uniform over all plausible trees. Layer $L - 1$ has size 2^{L-1} and layer L has size 2^L . W.l.o.g. we assume the set of nodes in layer L is labeled as $[1, 2, \dots, 2^L]$. The distribution over models is supported over the set of partitions of size 2^{L-1} of $[1, \dots, 2^L]$ where each partition set has size 2. We assume $2^L \geq B$ so that there are at least B distinct policies.

Let π be a fixed policy. It is enough to show there exist two worlds $\mathcal{M}_\psi^{(1)}$ and $\mathcal{M}_\psi^{(2)}$ such that policy π ends in the same state for both $\mathcal{M}_\psi^{(1)}$ and $\mathcal{M}_\psi^{(2)}$ but that there exist distinct policies $\pi^{(2)}, \dots, \pi^{(B)}$ such that their endpoints $(b_2^{(1)}, \dots, b_B^{(1)})$ and $(b_2^{(2)}, \dots, b_B^{(2)})$ in worlds $\mathcal{M}_\psi^{(1)}$ and $\mathcal{M}_\psi^{(2)}$ satisfy $(b_2^{(1)}, \dots, b_B^{(1)}) \neq (b_2^{(2)}, \dots, b_B^{(2)})$. The latter always holds because among the set of models that maintain the same endpoint for π , there is a pair of models that has different endpoints for $\pi^{(2)}$ for any arbitrary $\pi^{(2)} \neq \pi$. This suffices to show the entropy of the ensemble of distinct policies is strictly larger than the entropy of the ‘diagonal choice’ $(\underbrace{\pi, \dots, \pi}_{\text{size } B})$. $\square$

InfoCascade

Since the mutual information objective in Eq. 5.7 is submodular, a simple greedy algorithm yields a $(1 - \frac{1}{e})$ approximation of the optimum [Nemhauser et al. \(1978\)](#). This is the same observation that gives rise to the greedy algorithm underlying other batch exploration objectives, such as in BatchBALD [Kirsch et al. \(2019\)](#).

Let’s start by assuming we have candidate policies $\pi^{(1)}, \dots, \pi^{(i-1)}$. InfoCascade selects $\pi^{(i)}$ based on the following greedy objective,

$$\begin{aligned} \pi^{(i)} &= \arg \max_{\tilde{\pi}^{(i)} \in \Pi} \mathcal{I} \left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi]; \mathcal{M}_\psi \middle| \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1 \right) \\ &= \mathcal{H} \left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi] \middle| \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1 \right) - \mathcal{H} \left(\prod_{j=1}^i \mathbb{P}_{\tilde{\pi}^{(j)}}^\Phi[\mathcal{M}_\psi] \middle| \mathcal{M}_\psi, \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1 \right). \end{aligned}$$

Equation 5.8 implies

$$\mathcal{H} \left(\prod_{j=1}^i \mathbb{P}_{\pi^{(j)}}^{\Phi}[\mathcal{M}_{\psi}] \middle| \mathcal{M}_{\psi}, \tilde{\pi}^{(j)} = \pi^{(j)} \quad \forall j \leq i-1 \right) = \sum_{j=1}^i \mathcal{H} \left(\mathbb{P}_{\pi^{(i)}}^{\Phi}[\mathcal{M}_{\psi}] \middle| \mathcal{M}_{\psi} \right) \quad (\text{E.6})$$

and therefore if we approximate $\mathbb{P}_{\pi}^{\Phi}[m_{\psi}]$ by a Gaussian $\mathbb{P}_{\pi}^{\Phi}[m_{\psi}] \approx \mathcal{N}(\mu(m_{\psi}, \pi), \Sigma(m_{\psi}))$, the conditional entropy becomes a policy independent term. In this case finding $\pi^{(i)}$ boils down to solving for the policy that maximizes $\mathcal{H} \left(\prod_{j=1}^i \mathbb{P}_{\pi^{(j)}}^{\Phi}[\mathcal{M}_{\psi}] \middle| \pi^{(1)}, \dots, \pi^{(i-1)} \right)$. Using the same approximations described for the single policy objective, a suitable surrogate for this objective is to maximize $\max_{\pi} \text{Var}(\pi | \pi^{(1)}, \dots, \pi^{(i-1)})$, the empirical variance over means with respect to the policies found so far $\{\pi^{(j)}\}_{j=1}^{i-1}$,

$$\text{Var}(\pi | \pi^{(1)}, \dots, \pi^{(i-1)}) = \frac{1}{|\mathcal{M}_{\psi}| |i| - 1} \sum_{m_{\psi}} \sum_{\tilde{\pi} \in \{\pi^{(1)}, \dots, \pi^{(i-1)}, \pi\}} \|\mu(m_{\psi}, \tilde{\pi}) - \mu'(\pi, \pi^{(1)}, \dots, \pi^{(i-1)})\|^2$$

$$\text{Where } \mu'(\pi, \pi^{(1)}, \dots, \pi^{(i-1)}) = \frac{1}{|\mathcal{M}_{\psi}| |i|} \sum_{m_{\psi}} \sum_{\tilde{\pi} \in \{\pi^{(1)}, \dots, \pi^{(i-1)}, \pi\}} \mu(m_{\psi}, \tilde{\pi}).$$

E.4 Theoretical RL Intuition

The problem we study in this work can be thought of as the “batch” version of the Reward Free Exploration formalism (Jin et al., 2020). In this setting, the learner interacts with an MDP in two phases: 1) a training phase, where it is allowed to collect data from the environment; 2) a test phase, where the learner is presented with an arbitrary task (parametrized by a reward function unseen during training) and it must produce a near optimal policy. When faced with this problem, the learner is required to design a careful exploration strategy that permits them to build an accurate model in all areas of the state-space that can be reached with sufficient probability. There have been multiple follow up works that have also studied this problem in the context of Linear MDPs (Wang et al., 2020) and neural function approximation (Qiu et al., 2021). Nonetheless, there has been limited focus on the batch setting, where the learner is required to collect data via parallel data gathering policies in each training phase.

Due to the adaptive nature of the algorithm (every single environment query uses all information collected so far), there is an inevitable drop in performance when moving from the sequential (fully online) setting to the batch (deployment efficient) setting, as measured by the total number of environment interactions (in our case, $T \times B$ where T is the number of training rounds and B the number of parallel collection policies). To illustrate how CASCADE mitigates this loss in sample-efficiency, we outline the pseudo-code of CASCADE-TS, a greedy Thompson Sampling algorithm (see Agrawal & Goyal (2012)) that produces the i -th batch exploration policy in a tabular environment by incorporating fake count data from rolling out policies $\pi^{(1)}, \dots, \pi^{(i-1)}$ in the model. In the algorithms we present in the main body, we incorporate data gathered in the model by previous $i - 1$ selected policies into the batch when building the i -th exploration policy; this is directly related to the approach behind CASCADE-TS. Concretely, encouraging policy $\pi^{(i)}$ to induce a high variance among the embeddings produced by $\{\pi^{(1)}, \dots, \pi^{(i-1)}\}$ can be thought as adding a bonus to encourage $\pi^{(i)}$ to visit regions of the space that have a low embedding visitation count by the previous policies in $\{\pi^{(1)}, \dots, \pi^{(i-1)}\}$.

Algorithm 8 works by sampling a model from a model posterior, solving for an optimal uncertainty seeking policy and updating the model with ‘fake’ data corresponding to trajectories collected in the model from this uncertainty seeking policy. After producing the i -th exploration policy in the batch, the recomputed uncertainty bonuses of the model are reduced in the areas of the state space most visited by the i -th policy. This will encourage subsequent uncertainty seeking policies (i.e., $\pi_{k+1}^{(i+1)}, \pi_{k+1}^{(i+2)}, \dots, \pi_{k+1}^{(B)}$) to visit parts of the space not yet explored by previous policies.

Why Thompson Sampling? The reader may wonder why Algorithm 8 samples a model $\tilde{\mathbb{P}}_{k+1}^{(i)}$ from a TS posterior instead of using the empirical model resulting from fake and true data. The answer is that the raw empirical model may ascribe a probability of zero to certain transitions, which means we may miss exploring parts of the true state-action space. Having a TS prior that ascribes non-zero probabilities to all possible transitions fixes this potential issue.

Algorithm 8 CASCADE-TS

- 1: **Input:** Exploration batch size B . Fake samples parameter M .
 - 2: Initialize model $\hat{\mathbb{P}}_0$ over $\mathcal{S} \times \mathcal{A}$ state actions.
 - 3: Initialize the batch collection policies as $\{\pi_0^{(i)}\}_{i=1}^B$ to uniform.
 - 4: **for** time in $k = 0, 1, 2, \dots$ **do**
 - 5: Collect trajectory data $\{\tau_k^{(i)}\}_{i=1}^B$ from $\pi_k^{(i)}$ for all $i = 1, \dots, B$. Update $\mathcal{D}^{k+1} \leftarrow \mathcal{D}^k \cup \{\tau_k^{(i)}\}_{i=1}^B$.
 - 6: Initialize the fake data buffer $\mathcal{D}_{\text{fake}}^{k+1} = \emptyset$.
 - 7: **for** $i = 1, \dots, B$ **do**
 - 8: Sample MDP model from posterior $\tilde{\mathbb{P}}_{k+1}^{(i)} \sim \mathbb{P}(\cdot | \mathcal{D}^{k+1} \cup \mathcal{D}_{\text{fake}}^{k+1})$.
 - 9: Compute fake counts $N_{k+1}^{(i)}(s, a) = \sum_{(\tilde{s}, \tilde{a}, \tilde{s}') \in \mathcal{D}^{k+1} \cup \mathcal{D}_{\text{fake}}^{k+1}} \mathbf{1}(\tilde{s} = s, \tilde{a} = a)$.
 - 10: Compute fake uncertainty bonuses $b_{k+1}^{(i)}(s, a) \propto \sqrt{\frac{1}{N_{k+1}^{(i)}(s, a)}}$
 - 11: Solve for $\pi_{k+1}^{(i)}$ by solving,

$$\pi_{k+1}^{(i)} = \max_{\pi} \mathbb{E}_{\tau \sim \tilde{\mathbb{P}}_{k+1}^{(i)}, \pi} \left[\sum_{h=1}^H b_{k+1}^{(i)}(s_h, a_h) \right]$$
 - 12: Collect fake trajectory data $\{\tau_{\ell, \text{fake}}^{(i)}\}_{\ell=1}^M$ from posterior sampled model $\tilde{\mathbb{P}}_{k+1}^{(i)}$.
 - 13: Update fake data buffer $\mathcal{D}_{\text{fake}}^{k+1} \leftarrow \mathcal{D}_{\text{fake}}^{k+1} \cup \{\tau_{\ell, \text{fake}}^{(i)}\}_{\ell=1}^M$.
 - 14: **end for**
 - 15: **end for**
-

E.4.1 Proof of Lemma 2

Let's consider the class of deterministic MDPs with $S = |\mathcal{S}|$ states and $A = |\mathcal{A}|$ actions. It is clear that playing the same deterministic policy multiple times does not provide us with any more information than playing it once. We will then show that running Thompson TS, the learner will end up with a nonzero probability of producing at least two distinct policies within a batch of size $B > 2$.

We will further assume the posterior is aware of the deterministic nature of the MDP family so that any sample MDP model $\tilde{\mathbb{P}}_{k+1}^{(i)}$ is deterministic. For $i > 1$, the model $\tilde{\mathbb{P}}_{k+1}^{(i)}$ is one whose transitions are consistent with the fake data generated by $\pi_{k+1}^{(1)}, \dots, \pi_{k+1}^{(i-1)}$ (and the true data in $\mathcal{D}^k$).

A deterministic MDPs can be encoded as a set of triplets (s, a, s') . We say a model $\{(s, a, s')\}_{s \in \mathcal{S}, a \in \mathcal{A}}$ is ϵ -accurate if at least a fraction of $1 - \epsilon$ of the triplets are correct.

We now illustrate how the CASCADE-TS algorithm evolves in this setting. To

simplify our task, we will further restrict our attention to the family of deterministic MDPs made of binary trees of height L . The number of nodes in any of such trees equals $2^{L+1} - 1$. We will define the counts to be a large value ($2L$) when no data has been collected of a given state action pair. When data has been collected, we define the bonus terms to be of order at most 1.

To figure out the connectivity of any such trees it is necessary to try out $2^L - 1$ distinct sequences of L actions (the nature and connectivity of the remaining leaf nodes can be inferred once all the other ones have been decoded). To build an ϵ -accurate model, it is enough to know the connectivity structure of $1 - \epsilon$ proportion of paths corresponding to $(1 - \epsilon)(2^L - 1)$ leaf nodes.

To prove the result of Lemma 2, we observe that any batch strategy can be simulated by a fully sequential learner, so it immediately follows that $T(\epsilon, \text{Sequential}) \leq T(\epsilon, \text{AnyBatchStrategy})$. To show that CASCADE-TS will be better than SinglePolicyBatch in this tree example, as long as B is smaller than the remaining number of paths the learner has not tried out, it will produce a set of B policies different from any policy played so far and also different from each other.

To see why the second inequality is true, it is enough to see that complete paths (real or imagined) have a total reward score of at most L , since the counts are always at most one for each edge that is present. Nonetheless, the counts over any path (sequence of L actions) that has not been visited nor has been imagined to be visited will be at least $2L$. Thus, every step in the sequential batch construction mechanism will produce a new policy (sequence of L actions) not existent neither in the true nor the fake data so far. This finalizes the proof. Notice that the SinglePolicyBatch strategy will have tried only K unique paths after K batches have been collected, whereas the CASCADE-TS strategy will have tried BK , resulting in a potentially substantial improvement in sample efficiency.

Chapter Bibliography

Shipra Agrawal and Navin Goyal. Analysis of Thompson Sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pp. 39–1. JMLR Workshop and Conference Proceedings, 2012.

- Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565, 2019.
- Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In *International Conference on Machine Learning*, pp. 4870–4879. PMLR, 2020.
- Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. BatchBALD: Efficient and diverse batch acquisition for deep Bayesian active learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.
- George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming*, 14(1):265–294, 1978.
- Shuang Qiu, Jieping Ye, Zhaoran Wang, and Zhuoran Yang. On reward-free RL with kernel and neural function approximations: Single-agent MDP and Markov game. In *International Conference on Machine Learning*, pp. 8737–8747. PMLR, 2021.
- Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pp. 8583–8592. PMLR, 2020.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. 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. URL <https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf>.
- Ruosong Wang, Simon S Du, Lin Yang, and Russ R Salakhutdinov. On reward-free reinforcement learning with linear function approximation. *Advances in Neural Information Processing Systems*, 33:17816–17826, 2020.

F

Efficient Online Reinforcement Learning with Offline Data: Appendix

F.1 Detailed Experiments

F.1.1 Sparse Adroit

Full Results Here we present the full results for Sparse Adroit, including LayerNorm and environment-specific design choice ablations.

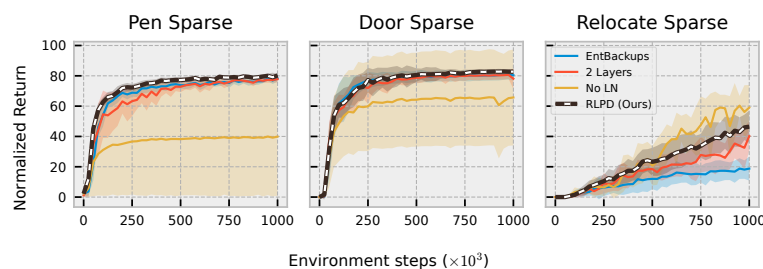


Figure F.1: Full Adroit Results.

We see that LayerNorm greatly reduces variance on the Pen and Door environments, however slightly harms mean performance on Relocate. Taken together however, LayerNorm is still a vital ingredient for reliable performance on this domain.

F.1.2 AntMaze

Full Results Here we present the full results for AntMaze, including LayerNorm and environment-specific design choice ablations.

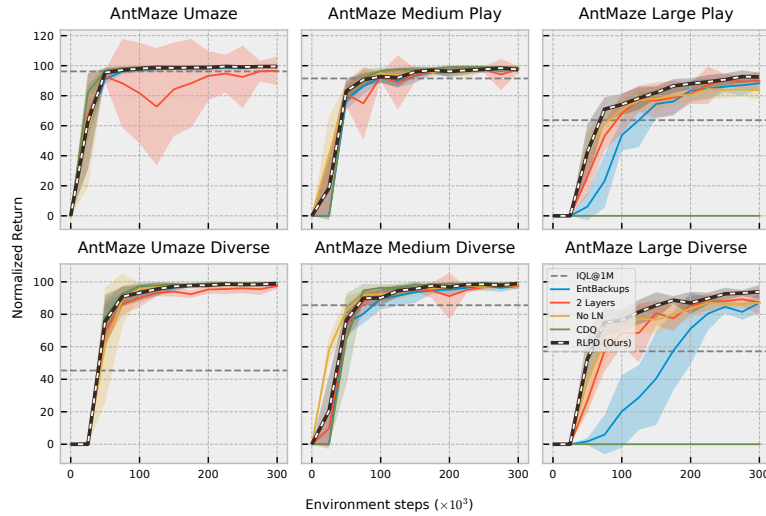


Figure F.2: Full AntMaze results

We see that our recommended design choices are vital for strong performance on the harder Large tasks, and furthermore see that using deeper 3-layer networks seems to help stability across all tasks. As we see, the best design choices on the Large environments are also optimal for the Umaze and Medium environments.

Gradient Steps per Environment Step Ablations We see LayerNorm is incredibly effective in a setting whereby we perform a single gradient update per time-step in Figure F.3. This may be required for learning on real robots, where computational efficiency is important.

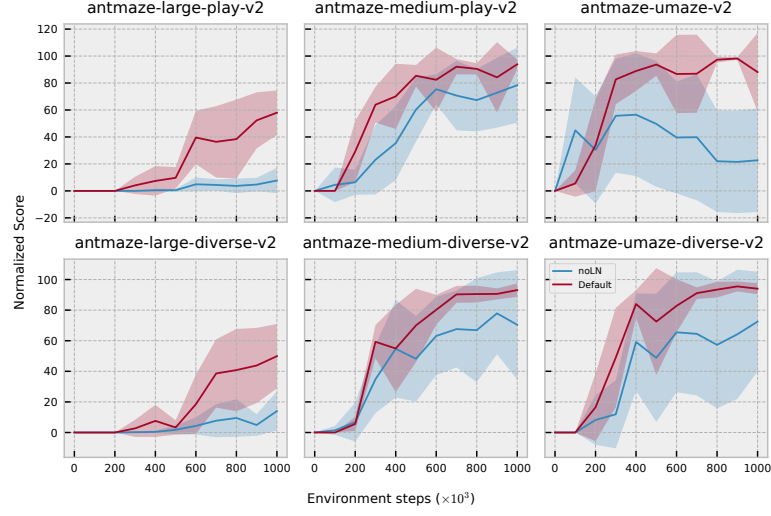


Figure F.3: SAC with and without Layer Normalization.

In Figure F.4, we see the impact of increasing the number of gradient steps per time-step. As we see, both sample efficiency stability improves greatly.

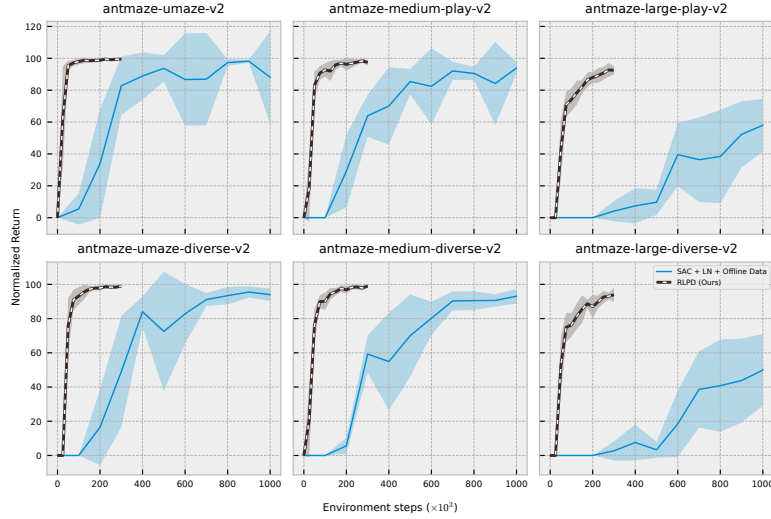


Figure F.4: RLPD v.s. SAC + LN + Offline Data.

Full IQL comparison Here we compare to IQL + Finetuning. We show pre-training gradient steps on IQL as negative indices in the x-axis. As we see in Figure F.5, despite IQL learning a strong initialization, it struggles to improve beyond this. In comparison, RLPD is able to quickly match, then greatly exceed IQL.

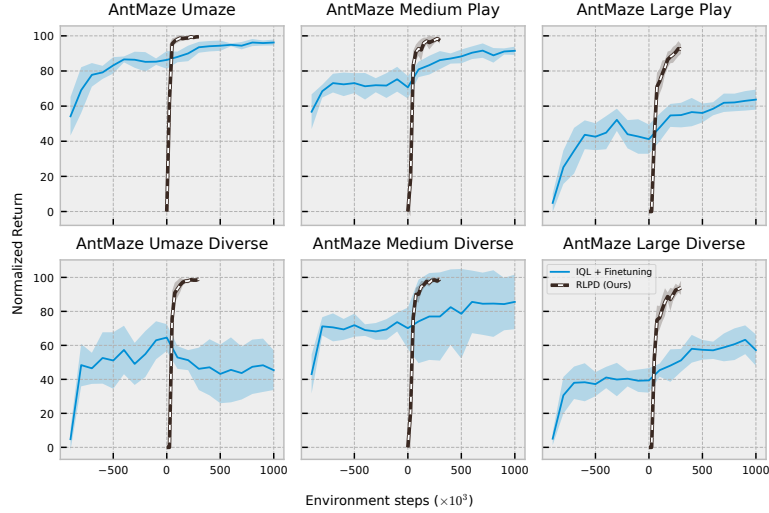


Figure F.5: RLPD v.s. IQL + Finetuning. We show offline pre-training steps for IQL using negative indices.

F.1.3 D4RL Locomotion

Full Results Here we present the full results for D4RL Locomotion, including LayerNorm and baselines. We emphasize that these tasks are not necessarily a good use case for online learning with offline datasets, as the tasks can be solved relatively quickly with purely online approaches, and there is no inherent difficulty regarding exploration.

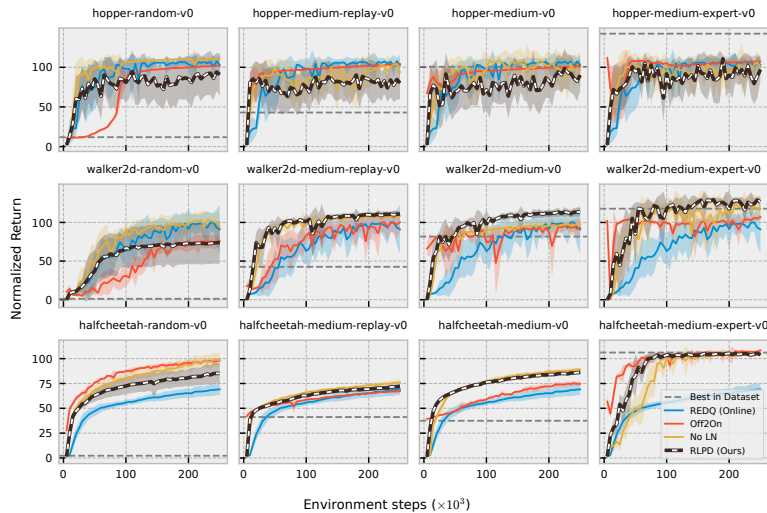


Figure F.6: D4RL Ablations.

The impact of LayerNorm is not so clear cut in Figure F.6; this is to be expected as online approaches already achieve strong results in this domain. Notably, we see

strong performance compared to baselines in the medium-expert domains.

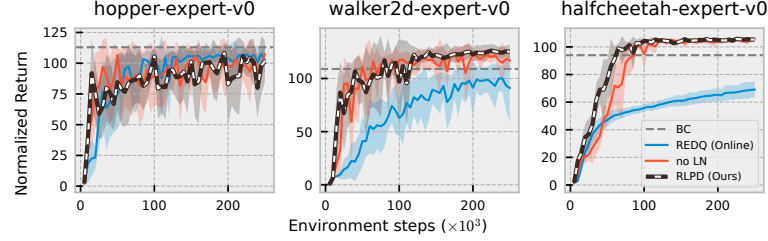


Figure F.7: D4RL Ablation on Expert Data.

We also test on the narrow Expert dataset, not tested by (Lee et al., 2021). In Figure F.7 we see LayerNorm can marginally help both sample efficiency, and asymptotic performance.

F.1.4 V-D4RL Locomotion

Full Results Here we present the full results for V-D4RL Locomotion, including LayerNorm and environment-specific design choice ablations.

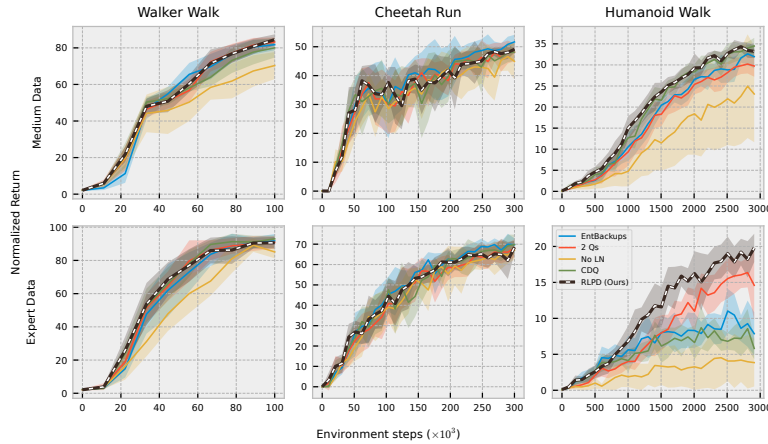


Figure F.8: VD4RL Ablations.

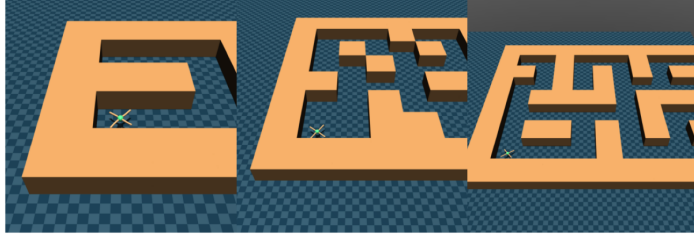
As we see, LayerNorm helps significantly in the Walker and Humanoid environments. We also see the positive impact of our recommended design choices in the complex Humanoid domain.

F.2 Experimental Details

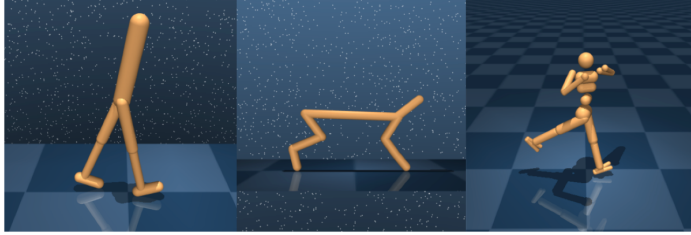
F.2.1 Further Environment Details



(a) The Sparse Adroit Domain. Pen, Door and Relocate tasks respectively.



(b) The AntMaze Domain. Umaze, Medium and Large tasks respectively.



(c) The V-D4RL Domain. Walker Walk, Cheetah Run and Humanoid Walk respectively.

Figure F.9: Visualizations of the environments we consider.

We provide further details about the key domains we evaluate on. In Figure F.9 we provide visualizations of the environments.

Sparse Adroit In these tasks, reward is a binary variable that indicates whether the task has been completed successfully or not. In prior work (Nair et al., 2020; Rudner et al., 2021), it is common to see success rate used as the metric, which simply determines if the task has been completed in any time step. However Kostrikov et al. (2022) use a more challenging metric that involves speed of completion. Concretely, return is calculated as the percentage of the total timesteps in which the task is considered solved (note that there are no early terminations). For example, in the ‘Pen’ task, where the horizon is 100 timesteps, if a policy achieves

a Normalized Score of 80, that means in 80 timesteps, the task is considered solved. This effectively means that the policy was able to solve the task in 20 timesteps. At each evaluation, we perform 100 trials.

D4RL: AntMaze In these tasks, reward is a binary variable that indicates whether the agent has reached the goal. Upon reaching the goal, the episode terminates. The normalized return is therefore the proportion of evaluation trials that were successful. We follow prior work, and perform 100 trials, and measure Normalized Return as the percentage of successful trials.

F.2.2 Hyperparameters

Here we list the hyperparameters for RLPD in Table F.1, and the environment-specific hyperparameters in Table F.2.

Table F.1: RLPD hyperparameters.

Parameter	Value
Online batch size	128
Offline batch size	128
Discount (γ)	0.99
Optimizer	Adam
Learning rate	3×10^{-4}
Ensemble size (E)	10
Critic EMA weight (ρ)	0.005
Gradient Steps (State Based) (G or UTD)	20
Network Width	256 Units
Initial Entropy Temperature (α)	1.0
Target Entropy	$-\dim(\mathcal{A})/2$
Pixel-Based Hyperparameters	
Action repeat	2
Observation size	[64, 64]
Image shift amount	4

Table F.2: Environment specific hyperparameters.

Environment	CDQ	Entropy Backups	MLP Architecture
Locomotion	True	True	2 Layer
AntMaze	False	False	3 Layer
Adroit	True	False	3 Layer
DMC (Pixels)	False	False	2 Layer

Chapter Bibliography

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=68n2s9ZJWF8>.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic Q-ensemble. In *5th Annual Conference on Robot Learning*, 2021. URL <https://openreview.net/forum?id=AlJXhEI6J5W>.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv*, June 2020.

Tim G. J. Rudner, Cong Lu, Michael Osborne, Yarin Gal, and Yee Whye Teh. On pathologies in KL-regularized reinforcement learning from expert demonstrations. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=sS8rRmgAatA>.