Investigating different exploration strategies for model-based reinforcement learning

Aseem Saxena, Joe Nguyen, Skand

Oregon State University

Abstract

Agents need to explore the world intelligently so as to discover new *skills* that are useful to perform downstream tasks. To perform exploration, there have been several methods that have been introduced in literature – however they lack a one-on-one comparison under the same policy setting. There is a discrepancy in terms of whether a model-based or a model-free policy is used to perform exploration and the choice of policy can effect the sample-efficiency of the agent significantly. In this project, we focus on implementing three exploration methods in model-based reinforcement learning setting and thoroughly investigate their qualitative and quantitative performance on the continuous control problem of *Point Maze*. Our experiments show that while ensemble based Plan2Explore (Sekar et al. 2020) performs the best, a naive and simple method such as Monte Carlo Dropout can perform on par with other exploration based methods.

Introduction

Reinforcement learning (RL) (Sutton and Barto 2018) agents learn by trial and error where they execute an action and obtain feedback from the environment in the form of a scalar reward. However, in sparse-reward settings, where immediate reward is unavailable and is only given to the agent at the end of episode on accomplishing the task successfully, a random exploration rarely leads the agent to successful states. Hence, it is important to explore *intelligently,* in a way that guides the agent cover the state space that would eventually lead to a successful learning of task.

In this work, we elucidate how different exploration algorithms measure uncertainty and quantify it as an intrinsic (or exploration) reward for a model-based RL setting. The particular choice of model-based agent is due to its sample-efficiency (Hafner et al. 2019a, 2021, 2023).

Concretely, our contributions can be summarized as follows:

- We investigate three *knowledge* based exploration methods in the context of continuous control point maze task.
- We implement and perform qualitative and quantitative experiments on Plan2Explore (Sekar et al. 2020), Curiosity (Pathak et al. 2017a), and Monte Carlo Dropout (Gal and Ghahramani 2016).
- In our ablation study, we look at the best performing methods (Plan2Explore) and ablate the number of ensembles to see its effect on exploration and downstream tasks.

Related Works

Exploration (or) Intrinsic Motivation: Learning only via an ϵ-greedy (Sutton and Barto 2018) method often tends the agent to learn a sub-optimal policy and does not elicit interesting behaviors that would achieve a higher reward. One common regime is that of *knowledge based* exploration that aims to explore based on the model's prediction. For example, curiosity based methods (Pathak et al. 2017a,b; Schmidhuber 1991) aim to train a dynamics model that explores by using the model prediction error. Similarly, Plan2Explore (Sekar et al. 2020) aims to quantify the intrinsic reward using the variance of the prediction from ensembles that are initialized with different weights. On similar lines, from a theoretical lens, Monte Carlo dropout (Gal and Ghahramani 2016) was proposed as a simple approximation to Bayesian inference where the weights of network were dropped out randomly to create an ensemble of predictions to compute uncertainty.

Another common line of work includes *competence* based methods in which an explicit *skill vector* is learnt by maximizing the mutual information between the encoded observation (input state) and skill (Eysenbach et al. 2018; Achiam et al. 2018).

Model Based RL: Model-Based RL from raw pixels approximates a representation space by minimizing the observation reconstruction loss. World Models (Ha and Schmidhuber 2018) learn the dynamics of the environment in a two-stage process to evolve linear controllers in imagination. Dreamer (Hafner et al. 2019b) utilizes RSSM (recurrent state-space model) to learn the dynamics model via long term imagination. Dreamer involves three major steps of learning dynamics from collected experience, learning a policy via imagination, and collecting more data based on the learned policy. We plan to use Dreamer as our framework since it is the first model-based RL algorithm that is competitive with the model-free algorithms on robotics and Deepmind control tasks (Tassa et al. 2018).

Preliminaries

Problem Formulation

We pose our problem as an infinite-horizon Markov Decision Process (MDP) defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, \rho_0)$. $\mathcal{S}$ represents the state space. $\mathcal{A} \in \mathbb{R}^m$ is an m-dimensional continuous action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition function, $\mathcal{R} : \mathcal{S} \to \mathbb{R}$ is the reward function, $\gamma \in [0, 1)$ is the discount factor and ρ_0 denotes the initial state distribution. The goal of the agent is to learn a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected sum of discounted rewards; $\max_\pi \mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^t \mathcal{R}(s_t)]$.

World Model

We base our world model on the Recurrent State-Space Model (RSSM) framework (Hafner et al.) which learns a recurrent world model with a d-dimensional latent variable z. The RSSM model is derived from an evidence lower bound on the likelihood of an observation sequence $o_{1:T}$ given actions $a_{1:T}$. This results in a loss composed of two components: a reconstruction term measuring how well observations (and rewards) can be predicted from the latent representation and a KL divergence term keeping predicted latent states near their corresponding real observation encoding.

More concretely, the world model is parameterized by ϕ and consists of the following components:

$$\begin{aligned}
&\text{Representation:} && z_t \sim q_\phi(h_t, o_t) \\
&\text{Recurrent Model:} && h_t = f_\phi(z_{t-1}, h_{t-1}, a_{t-1}) \\
&\text{Dynamics:} && \hat{z}_t \sim p_\phi(\hat{z}_t | h_t) \\
&\text{Reward:} && \hat{r}_t \sim p_\phi(\hat{r}_t | h_t, z_t) \\
&\text{Continuation Predictor:} && \hat{c}_t \sim p_\phi(\hat{c}_t | h_t, z_t) \\
&\text{Decoder:} && \hat{x}_t \sim p_\phi(\hat{x}_t | h_t, z_t)
\end{aligned} \tag{1}$$

where we use $\sim$ to denote the sampling operation. The continuation flag $c_t \in \{0, 1\}$ indicates whether the episode has ended. Except for the input encoder network within q_ϕ, we retain architectural choices from (Hafner et al. 2023) for the other components.

Policy Learning

For the policy, we adopt the Actor-Critic framework (Konda and Tsitsiklis 1999) similar to DreamerV3 (Hafner et al. 2023), which consists of a *Critic* network that predicts the value at a given state and an *Actor* that predicts the action distribution given a state.

$$\begin{aligned}
&\text{Actor network:} && a_t \sim \pi_\psi(a_t | z_t) \\
&\text{Critic network:} && v_\psi(z_t) \approx \mathbb{E}_{q(.|z_t)}[\textstyle\sum_{\tau=t}^{t+H} (\gamma^{\tau-t} r_\tau)]
\end{aligned} \tag{2}$$

The critic is learned by discrete regression (Hafner et al. 2023) using generalized λ-targets (Schulman et al. 2018). We train the actor network to maximize the value function via dynamics back-propagation (Hafner et al. 2019a), updating actor parameters using the gradients computed through the world model. Further, we use **Symlog Predictions** (Hafner et al. 2023) for the reward predictor and the critic. Symlog is helpful in dealing with environments with varying reward scales across different tasks. The overall framework alternates between the world model training, policy training, and data collection using the most recent policy.

Exploration Methods

We consider three intrinsic motivation methods namely, (a) Intrinsic Curiosity Module (Pathak et al. 2017a), (b) Plan2Explore (Sekar et al. 2020), and (c) Monte Carlo Dropout (Gal and Ghahramani 2016). In this section we provide the details of each of the exploration models.

Curiosity Based exploration (Pathak et al. 2017a; Schmidhuber 1991): One of the earliest exploration models that operates on continuous controls was that of curiosity based exploration. These set of methods aim to discover new states of the world by trying to maximize their dynamics model prediction error. Specifically, given a state s_t and action a_t, the learned dynamics model predicts the next state $\hat{s_{t+1}}$. The intrinsic reward is formulated as the error between the ground truth and the prediction, $|s_{t+1} - \hat{s_{t+1}}|$. The higher this prediction error, the more uncertain the agent is about the world.

Plan2Explore (Sekar et al. 2020): Plan2Explore is a disagreement-based method to encourage the agent to learn about uncertain state regions. Plan2Explore measures the disagreement between an ensemble of dynamics models and uses the variance of predictions as reward to learn an exploration policy. The idea being if the variance is large, then the agent is uncertain about the region and we would want it to explore more in that region. It maintains an ensemble of N dynamic models and use disagreement (variance) among them as an intrinsic reward for exploration. Concretely, all the ensemble models take in the same state-action pair (s, a) as the input and predict the next state's representation $\{s_{t+1}^n\}_{n=1}^N$. The idea being if all the models predict the next state accurately, the disagreement would be 0 and hence the reward would be 0. This formulation encourages the agent to explore regions that have more uncertainty i.e maximizing the variance of the predictions.

MC Dropout

Dropout was introduced as a method to prevent overfitting in deep neural networks(Srivastava et al. 2014). It works by blocking out random units and their connections during training to prevent co-adaptation of units. Dropout is traditionally applied only during training. Gal and Ghahramani proposed that using dropout during evaluation can be a viable alternative to ensemble methods for constructing confidence intervals for uncertainty estimation by dropping different units during inference and getting different predictions as a consequence.

We summarize the intrinsic reward formulation of the three methods in Table 1.

Experiments

In this section we answer the following question: 1) How well does the exploration cover the state space qualitatively

Table 1: Knowledge based exploration algorithms and their reward formulation.

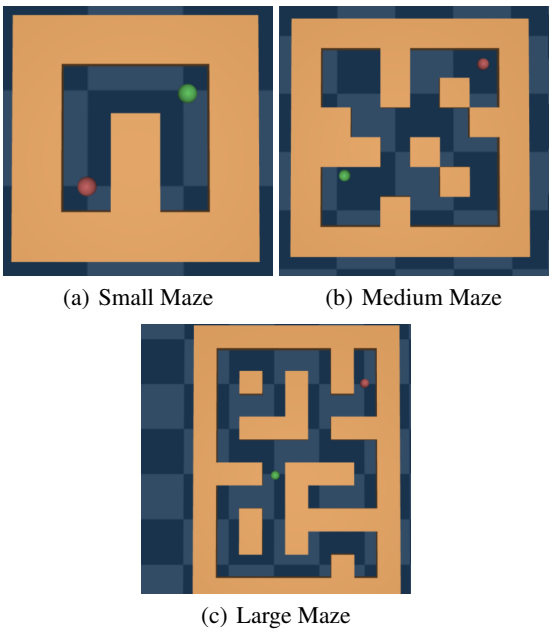
Name	Intrinsic Reward
Curiosity (Pathak et al. 2017a)	$\|f_{\text{dyn}}(\hat{s_{t+1}}\|s_t, a_t) - s_{t+1}\|^2$
Disagreement (Sekar et al. 2020)	$\text{Var}\{f^i_{\text{dyn}}((\hat{s_{t+1}}\|s_t, a_t) - s_{t+1})\} \quad i = 1, \ldots, N$
MC-Dropout (Gal and Ghahramani 2016)	$\text{Var}\{\text{Dropout}(f_{\text{dyn}})^i((\hat{s_{t+1}}\|s_t, a_t) - s_{t+1})\} \quad i = 1, \ldots, N$

for each of the three models? 2) Which exploration strategy learns general skills that can be deployed to the downstream task? 3) Does having higher number of ensembles for the disagreement reward help? We first look into the environment used and then delve into the results.

Point Maze

We run our experiments on Point Maze (Fu et al. 2021). The task in the environment is designed for a 2-Degree-of-Freedom green ball that is force-actuated in the Cartesian directions x and y, to reach a target red goal in a closed maze. We test our models in three different settings with increasing levels of complexity: small, medium and large maze, which are illustrated in Figure 1. Observation space consists of 4 keys (x, y, v_x, v_y): coordinates (x, y) of the ball and linear velocities in according to x, y: (v_x, v_y). Action space is the linear force $(motor_x, motor_y)$ exerted on the ball in the x and y direction. The rewards of this environment can be initialized as sparse or dense. In case of *dense* reward, the returned reward is the negative Euclidean distance between the achieved goal position and the desired goal. Meanwhile, the *sparse* reward is 0 if the ball hasn't reached the final destination, and 1 if it is within 0.5m to the final goal in terms of Euclidean distance. To test the exploration, we opt for the sparse reward setting in the first $2e5$ steps and use the dense reward for the rest of the task.

We run all the above methods namely Curiosity, Plan2Explore (P2E), and MC-Dropout within the codebase of DreamerV3 (https://github.com/NM512/dreamerv3 torch 2023).



(a) Small Maze (b) Medium Maze

(c) Large Maze

Figure 1: Maze Environments of varying complexities (small, medium, large) for studying exploration strategies in deep model-based reinforcement learning.

Results & Discussion

In this section we present first the qualitative evaluation and then present the quantitative evaluation of the three methods on a downstream task. Further we perform ablation study by varying the number of ensembles.

Qualitative Evaluation

First, we qualitatively evaluate the results of the three methods for different maze size (small, medium, and large). As shown in Figure 2, we observe that in terms of state space coverage, Plan2Explore performs the best. This is potentially due to the disagreement objective coming from different ensembles. It is interesting that these ensembles do not collapse into a single mode and that is because they are initialized with different weights.

More interestingly, we find that an incredibly simple technique such as Monte Carlo Dropout performs similar to curiosity based exploration.

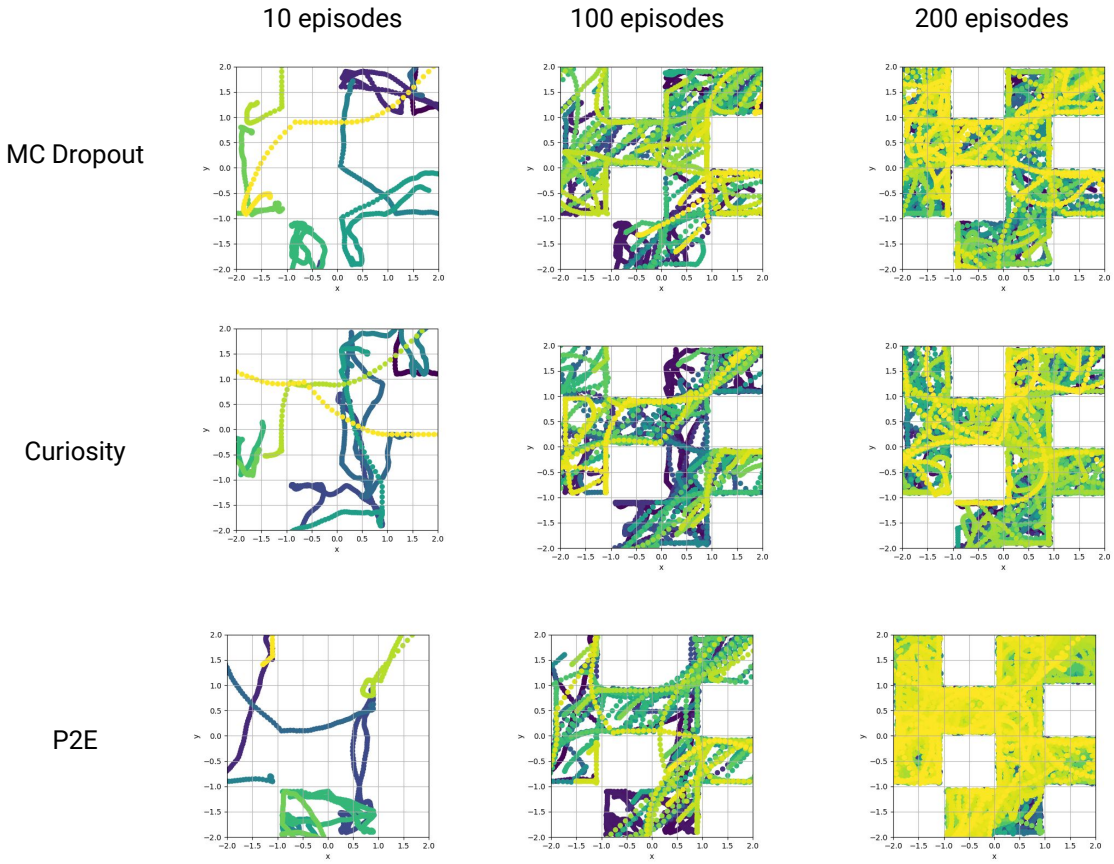


Figure 2: Visualization on Medium point maze depicting the state space ((x, y) coordinates) coverage of different exploration methods. Plan2Explore performs the best, but interestingly a simple method such as MC Dropout works equally well as Curiosity based methods. **Blue** dots means that the states were explored earlier in the training process and **Yellow** dots means that the states were explored later in the training process. Best seen in color.

Quantitative Evaluation

To further look at how each exploration method performs on Medium Maze, we quantitatively evaluate the three methods by first exploring for 2e5 steps and then adding in the task specific reward of reaching the red goal. Based on the plot in Figure 3, we observe that Plan2Explore performs the best – potentially because of higher state space coverage (as shown in Figure 2). Interesting, even in the quantitative assessment, we find that MC-Dropout (Gal and Ghahramani 2016) to perform on par with Curiosity based exploration (Pathak et al. 2017a).

Effect of the number of ensembles (N)

We also investigate the effect of the number of ensembles for Plan2Explore by experimenting with $N = \{2, 5\}$ and find that the higher ensembles lead to better downstream task performance as shown in Figure 4. The reason why a lesser number of ensembles perform poorer is that disagreement captures the uncertainty between the N dynamics models, and the lesser the number of models, the poorer the level of uncertainty in the variance. We did not find any significant increase in performance beyond $N = 5$.

Codebase and Implementation Details

In this work, we have used the existing codebase for Dreamer (https://github.com/NM512/dreamerv3 torch 2023). We have added the disagreement (Sekar et al. 2020), curiosity (Pathak et al. 2017a) and MC-Dropout (Gal and Ghahramani 2016) code in `exploration.py`. See the shared code for more details.

Limitations and Future work

Our work has the following limitations.

- We did not consider *competence* based methods such as DIAYN (Eysenbach et al. 2018) and other mutual information methods based on DIAYN formulation and leave that as a future work.
- Although challenging, we only considered maze like environments. It would be interesting to see a thorough investigation of these methods more complex locomotion

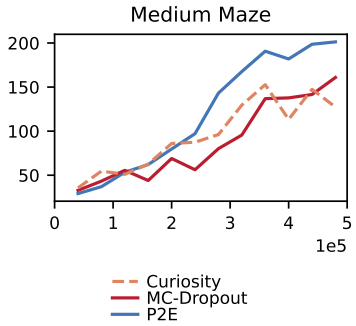


Figure 3: Quantitative analysis of exploration for 2e5 steps and addition of task reward in the RL objective from 2e5 - 5e5 steps. The plots show that Plan2Explore (Sekar et al. 2020) achieves better task performance due to its higher state space coverage.
. The x axis represents the number of steps executed and the y axis represents the average cumulative reward.

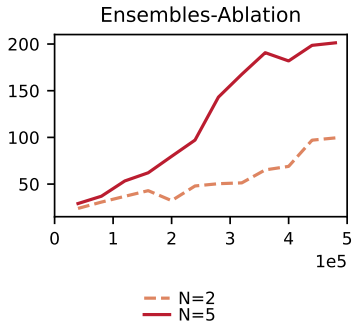


Figure 4: Quantitative analysis with different ensemble size on Plan2Explore for 2e5 steps and addition of task reward in the RL objective from 2e5 - 5e5 steps. The plot shows that higher number of ensembles is typically preferred. We did not find any significant increase in performance beyond N=5. The x axis represents the number of steps executed and the y axis represents the average cumulative reward.

tasks such as DM Control (Tassa et al. 2018) which have much larger state and action spaces.

Conclusion

In this project, we investigate three different exploration methods for model-based RL in the continuous control problem of Point Maze: 1) Plan2Explore (Sekar et al. 2020), 2) Curiosity Based exploration (Pathak et al. 2017b; Schmidhuber 1991) and 3) Monte-Carlo Dropout (Gal and Ghahramani 2016). Our experiments show that Plan2Explore performs the best in task performance. Additionally, we show that Monte Carlo Dropout serves as a strong baseline which is easy to implement and is relatively inexpensive computationally. We show that just 5 ensemble members can be used to obtain reliable uncertainty estimates for the downstream task of exploration in model-based RL.

References

Achiam, J.; Edwards, H.; Amodei, D.; and Abbeel, P. 2018. Variational Option Discovery Algorithms. *ArXiv*, abs/1807.10299.

Eysenbach, B.; Gupta, A.; Ibarz, J.; and Levine, S. 2018. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2021. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. ArXiv:2004.07219 [cs, stat].

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*.

Ha, D.; and Schmidhuber, J. 2018. World models. *arXiv preprint arXiv:1803.10122*.

Hafner, D.; Lillicrap, T.; Ba, J.; and Norouzi, M. 2019a. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*.

Hafner, D.; Lillicrap, T.; Ba, J.; and Norouzi, M. 2019b. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*.

Hafner, D.; Lillicrap, T.; Fischer, I.; Villegas, R.; Ha, D.; Lee, H.; and Davidson, J. ???? Learning Latent Dynamics for Planning from Pixels.

Hafner, D.; Lillicrap, T.; Norouzi, M.; and Ba, J. 2021. Mastering atari with discrete world models. *Internation Conference on Learning Representations*.

Hafner, D.; Pasukonis, J.; Ba, J.; and Lillicrap, T. 2023. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*.

https://github.com/NM512/dreamerv3 torch. 2023. Dreamerv3-torch. *Github*.

Konda, V.; and Tsitsiklis, J. 1999. Actor-Critic Algorithms. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017a. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, 2778–2787. PMLR.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017b. Curiosity-Driven Exploration by Self-Supervised Prediction. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 488–489.

Schmidhuber, J. 1991. A possibility for implementing curiosity and boredom in model-building neural controllers.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2018. High-Dimensional Continuous Control Using Generalized Advantage Estimation. ArXiv:1506.02438 [cs].

Sekar, R.; Rybkin, O.; Daniilidis, K.; Abbeel, P.; Hafner, D.; and Pathak, D. 2020. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, 8583–8592. PMLR.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tassa, Y.; Doron, Y.; Muldal, A.; Erez, T.; Li, Y.; Casas, D. d. L.; Budden, D.; Abdolmaleki, A.; Merel, J.; Lefrancq, A.; et al. 2018. Deepmind control suite. *arXiv preprint arXiv:1801.00690*.