Accelerating Model-Based Reinforcement Learning with Skill Abstraction from Unlabeled Data

Aaron Trinh Shitij Govil Zenghui Sun Shaunak Halbe

Abstract

Model-Based Reinforcement Learning (MBRL) faces challenges with scalability and sample efficiency, especially in multi-task and spare reward settings. Using unlabeled trajectory data as pretraining is a promising method for skill abstraction, enabling better exploration. This paper introduces Model-based Skill Abstraction using Unlabeled Data (MoSAUD), a hierarchical reinforcement learning framework that combines model-based planning and skill pretraining. In the offline phase, low-level skills are extracted from unlabeled segments of the trajectory using a variational autoencoder, while a skill dynamics model is trained for the state space at the same time. In the online phase, a high-level policy learns to put together these reusable, pretrained skills while being guided by optimistic reward estimates and model predictive control (MPC). Evaluations on the FrankaKitchen benchmark show MoSAUD's ability to adapt to sparse-reward and long-horizon environments. Overall, this framework showcases the potential of skill abstraction and model-based Reinforcement Learning. Our code is available here: https://github.com/aarontrinh02/MoSAUD. Our video is available here: https://tinyurl.com/drlmosaud.

1. Introduction

Reinforcement Learning (RL) has been proven successful in solving complex tasks, however, its scalability and efficiency remain constrained when applied to diverse environments and sparse reward scenarios. Inspired by unsupervised pretraining across domains like language and vision (Devlin et al., 2018; He et al., 2022), we are trying to explore how pretraining and model-based planning can be combined to create more efficient RL. Unlike supervised learning, RL requires agents to iteratively improve through exploration. Therefore, methods that can both learn transferable representations and facilitate effective exploration are crucial for such techniques.

Unlabeled trajectory data captures diverse behavioral patterns across environments and thus provides a valuable resource for pretraining in RL. However, utilizing these data imposes two major challenges: (1) disentangling general knowledge of the environment from task-specific behaviors and (2) transforming this knowledge into actionable insights for solving new tasks. Previous approaches have addressed these challenges by learning low-level skills from trajectory data, but those approaches have mostly ignored the utility of the same data for high-level policy learning during online interactions. To bridge this gap, we introduce a hierarchical approach that fully leverages unlabeled trajectory data for both skill pretraining and online exploration. In this work, we present MoSAUD, an algorithm that aims to utilize such trajectories efficiently.

Our method integrates *Skills from Unlabeled Prior Experience (SUPE)* (Wilcoxson et al., 2024) with a learned dynamics model to create a framework to apply RL in long horizon and sparse-reward settings. In the offline phase, we employ SUPE to extract task-agnostic low-level skills from trajectory segments using a variational autoencoder (VAE) (Kingma and Welling, 2014). These skills will be used in the online phase, where Model Predictive Control (MPC) is used to select and compose these pretrained skills efficiently. By planning in the latent skill space, our method abstracts long action sequences, reduces the complexity of decision-making, and enables effective exploration via optimistic reward estimates.

Our framework offers two key benefits: (1) the low-level skills enable robust execution of meaningful behaviors, and (2) the model-based planning over skills accelerates learning by focusing on high-reward trajectories. By evaluating our method in the Kitchen environment, we demonstrate its performance over existing baseline RL approaches in sparse-reward and multi-task environments.

To summarize, the key contributions of our work are as follows:

- We introduce MoSAUD, a new framework that combines unsupervised skill pretraining and model-based planning to address the challenges of reinforcement learning in diverse and sparse-reward environments.

- Through MoSAUD, we develop a hierarchical approach that leverages pretrained skills for both efficient exploration and high-level policy learning during online interactions.

- We demonstrate some performance improvements over baseline methods in challenging sparse-reward and multi-task environments, highlighting the scalability and robustness of our approach.

2. Related Work

Model based reinforcement learning. Traditional model-based reinforcement learning (MBRL) is a subfield of RL where agents create internal models of their environment's dynamics $p(s_{t+1}|s_t, a_t)$ to predict future states and rewards. By simulating interactions with the dynamics model, agents are able to plan without relying on exploring (and potentially failing) in the environment. An early example was the Dyna architecture (Sutton, 1991), which combined direct RL with model learning. The framework allowed agents to improve sample efficiency by updating policies with both real and simulated experiences. Subsequent advances like PILCO (Deisenroth and Rasmussen, 2011) were developed, employing probabilistic models. In recent years, MBRL has seen significant advances with methods like Temporal Difference Model Predictive Control (TD-MPC) (Hansen et al., 2022) and Dreamer (Hafner et al., 2019). TD-MPC integrates temporal difference with model predictive control (MPC), using a learned latent dynamics model for local trajectory optimization in addition to a terminal value function for long-term return estimation. Another notable MBRL algorithm, Dreamer, learns a "world model" to create a compact representation of the agent's environment in a latent space. It then learns behavior by imaging trajectories in its latent world model.

Unsupervised skill discovery in RL focuses on enabling agents to autonomously learn a diverse set of behaviors without reward signals via structured, reusable skills that can be applied to various tasks. Early methods in this domain such as Variational Intrinsic Control (Gregor et al., 2016) introduced the idea of maximizing mutual information between skills and their resulting trajectories to encourage distinct behaviors. Building on this, the Option-Critic architecture (Bacon et al., 2016) provided a framework for learning "options" (essentially temporal abstractions) directly from the interaction data, facilitating hierarchical learning. The field then moved towards emphasizing learning a diverse set of distinguishable skills with Diversity is All You Need (Eysenbach et al., 2018) and Contrastive Intrinsic Control (Laskin et al., 2022) where contrastive learning was used to maximize mutual information between states and latent skills, creating more distinct behaviors. Additionally, newer methods like Disentangled Unsupervised Skill Discovery (Hu et al., 2024) focus on learning disentangled skills, where each skill component only affects specific factors in a state. Another notable contribution is Skills from Unlabeled Prior data for Exploration (SUPE) (Wilcoxson et al., 2024), which combines offline unsupervised skill pretraining with online

Offline to Online RL is a paradigm that uses pre-collected offline data to initialize a policy, which is then iteratively improved via online interactions. The most fundamental version of this approach involves pre-training a policy with offline RL on the offline dataset, and then fine-tuning it with an online RL method on new interactions. However, this approach often fails because the offline RL objectives constrain the policy too much to prior data, significantly limiting the agent's online exploration capabilities. To address this, multiple methods have been proposed. An example is Ensemble-based Offline-to-Online RL (Zhao et al., 2023) which uses multiple Q-networks to transition from offline pretraining to online finetuning with minimal performance loss. This is done by adjusting the amount of pessimism in Q-value estimation in addition to exploration methods utilizing ensemble techniques. Another notable method is Adaptive Policy Learning (Zheng et al., 2023) which applies pessimistic updates for offline dataset, but optimistic updates for online dataset. Ultimately, these algorithms provide frameworks to balance exploiting offline data while also exploring sufficiently in an online setting.

3. Problem Formulation

We consider a Markov Decision Process (MDP) characterized by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, p_0)$, where $\mathcal{S} \in \mathbb{R}^n$ and $\mathcal{A} \in \mathbb{R}^m$ are continuous state and action spaces. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}_+$ is the transition (dynamics) function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and p_0 is the initial state distribution.

We assume access to a dataset of transitions $\mathcal{D} = \{(s_i, a_i, s'_i)\}_{i=1}^N$, where transitions are collected from the same MDP without reward labels. During the offline phase, our goal is to extract reusable skills (defined as a sequence of actions) from this unlabeled dataset using a trajectory Variational Autoencoder (VAE) (Kingma and Welling, 2014). In the online phase, the agent interacts with the environment and learns a high-level off-policy policy $\pi_{\text{high}}(g \mid s)$ that selects which skill $g \in \mathcal{G}$ to execute every H steps. The aim is to accelerate task learning by leveraging pre-extracted skills for efficient exploration and reward collection in the environment.

Unlike traditional reinforcement learning (RL) setups, the agent begins with zero knowledge of the reward function and must actively explore the environment to identify the task by receiving rewards through interactions. The dataset $\mathcal{D}$ is leveraged to guide exploration and efficiently collect reward information during the online phase. The objective is to output a well-performing policy $\pi(a \mid s)$ that maximizes the cumulative return:

$$\eta(\pi) = \mathbb{E}_{s_0 \sim p_0, a_t \sim \pi(a_t|s_t), s_{t+1} \sim \mathcal{T}(s_{t+1}|s_t, a_t)} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t) \right]$$

This formulation differs from zero-shot RL settings, where the reward function is specified only during evaluation. In our setting, the reward function is unknown, requiring the agent to actively interact with the environment to identify the task.

Model Predictive Control. In actor-critic reinforcement learning (RL) algorithms, the policy Π is typically represented by a neural network, which approximates $\Pi_\theta(\cdot|\mathbf{s}) \approx \arg\max_{\mathbf{a}} \mathbb{E}[Q_\theta(\mathbf{s}, \mathbf{a})]\ \forall \mathbf{s} \in \mathcal{S}$, aiming to approximate the globally optimal policy. In contrast, control theory traditionally implements Π as a trajectory optimization procedure. To ensure computational feasibility, this is often solved locally at each time step t, by finding an optimal sequence of actions $\mathbf{a}t : t + H$ over a finite horizon H, and then executing only the first action $\mathbf{a}t$. This approach is known as *Model Predictive Control* (MPC):

$$\Pi_\theta^{\text{MPC}}(\mathbf{s}_t) = \arg \max_{\mathbf{a}_{t:t+H}} \mathbb{E} \left[\sum_{i=t}^{H} \gamma^i \mathcal{R}(\mathbf{s}_i, \mathbf{a}_i) \right].$$

A solution involves iteratively fitting distribution parameters (e.g., μ, σ for a multivariate Gaussian with diagonal covariance) to the action space over a finite horizon using the derivative-free Cross-Entropy Method (CEM; Rubinstein (1997)) and model-generated trajectories.

4. MoSAUD

In this section, we describe MoSAUD, which utilizes a dataset of unlabeled trajectories. Our method requires two phases. First, in the offline pretraining phase, we extract reusable skills and learn the skill dynamics model. Then, in the online phase, we learn a high level policy from both the offline data and online experience, leveraging the skill dynamics model for model predictive control (MPC) in the skill space. Our method is described in Algorithm 1.

Unlabeled trajectories. First, we note that our setting is different from offline-RL because our dataset consists of N state-action trajectories $\mathcal{D} = \{\tau_1, \cdots, \tau_N\}$, where each trajectory has the form $\tau = \{s_0, a_0, \cdots, s_{E-1}, a_{E-1}, s_E\}$, letting E denote the length of an episode. Since our dataset lacks reward labels (unlabeled), we cannot use traditional offline-RL methods to learn the policy. Additionally, the dataset does not necessarily contain expert demonstrations, so learning naively with behavior cloning will not lead to an optimal solution.

Skill extraction. To extract reusable skills, we define a skill as a fixed length H sequence of actions $(a_0, \cdots, a_{H-1})$. Following prior methods, we learn a VAE (Kingma and Welling, 2014) to encode sequences of actions into a latent skill z (Pertsch et al., 2020). The H length trajectory $\tau = \{s_0, a_0, \cdots, s_{H-1}\, a_{H-1}\}$ is embedded to the latent skill through the VAE encoder, $q_\theta(z|\tau)$, and the actions are reconstructed through the skill policy $\pi_\theta(a|s, z)$. Following (Pertsch et al., 2020), we also learn a state-dependent skill prior $p_\theta(z|s)$ to guide the agent in exploring plausible skills over the large skill distribution. We follow the implementation of SUPE (Wilcoxson et al., 2024), where the skill extraction loss is shown in Equation 1.

$$\mathcal{L}_\theta = \beta D_{KL}(q_\theta(z|\tau)||p_\theta(z|s_0)) - \mathbb{E}_{z \sim q_\theta(z|\tau)} \left[\sum_{h=0}^{H} \log \pi_\theta(a_h|s_h, z) \right] \quad (1)$$

Skill Dynamics Model. To facilitate efficient learning in the skill space, we also learn a dynamics model from the offline data. Different from most methods, however, we follow (Shi et al., 2022) by employing a model that plans in the skill space. Given the current state s_t and latent skill z_t, the skill dynamics model $D_\phi(s_t, z_t)$ predicts the state after the skill execution, s_{t+H}. By planning in the skill space, our framework can accurately plan over long horizons, reducing compounding errors that arise when using single-step dynamics models for long horizon planning. We also update the dynamics model online for more accurate predictions. Practically, we batch the offline and online data together for updates (see Algorithm 1 for more details). The skill dynamics loss is shown in Equation 2.

$$\mathcal{L}_\phi = \mathbb{E}_{s_t, z_t, s_{t+H} \sim \mathcal{D}} \left[\|D_\phi(s_t, z_t) - s_{t+H}\|_2^2 \right] \quad (2)$$

Online learning. To effectively use our unlabeled trajectories as off policy data, we follow (Li et al., 2023) to relabel the offline data with an optimistic reward estimate, $r_{UCB}(s, z)$. Additionally, following (Wilcoxson et al., 2024), we use a reward bonus for the online data, which encourages exploration. We use the implementation from (Li et al., 2023), where we first update a reward network with the loss:

$$\mathcal{L}_\nu = \|r_\nu(s_0, z) - r\|_2^2 \quad (3)$$

Then, Random Network Distillation (RND) (Burda et al., 2018) initializes two networks $f_\xi(s, z)$ and $\bar{f}_\xi(s, z)$, each outputting an L dimensional feature vector. $\bar{f}_\xi(s, z)$ is kept frozen while the other network is updated to minimize the

Algorithm 1 MoSAUD

```
Require: Unlabeled dataset of trajectories $\mathcal{D}$, trajectory segment length $H$ and batch size $B$
 1: for each pretraining step do
 2:     Sample a batch of trajectory segments of length $H$, $\{\tau_1, \cdots, \tau_B\}$ from $\mathcal{D}$
 3:     Optimize the skill policy $\pi_\theta(a|s,z)$, the trajectory encoder $q_\theta(z|\tau)$, along with the state-dependent prior $p_\theta(z|s)$ with
        the VAE loss $\frac{1}{B}\sum_{i=1}^{B}\mathcal{L}_\theta(\tau_i)$.
 4: end for
 5: $\mathcal{D}_{\text{replay}} \leftarrow \emptyset$
 6: Initialize the optimistic reward module $r_{\text{UCB}}(s,z)$
 7: for every $H$ online environment steps do
 8:     Sample the trajectory latent $z \sim \pi_\psi(z|s)$ from CEM
 9:     Run the skill policy $\pi_\theta(a|s,z)$ for $H$ steps in the environment: $\{s_0, a_0, r_0, \cdots, s_H\}$
10:     Add the high-level transition to buffer $\mathcal{D}_{\text{replay}} \leftarrow \mathcal{D}_{\text{replay}} \cup \{(s_0, z, s_H, \sum_{i=0}^{H-1}[\gamma^i r_i])\}$
11:     Sample a batch of trajectory segments of length $H$, $\{\tau_1, \cdots, \tau_B\}$ from $\mathcal{D}$
12:     Encode each trajectory segment using the trajectory encoder: $\hat{z}^i \sim q_\theta(z|\tau_i)$
13:     Use $q_\theta$ and $r_{\text{UCB}}$ to transform each unlabeled trajectory segment into a high-level transition with pseudo-labels:
        $\mathbf{B}_{\text{offline}} = \{(s_0^i, \hat{z}^i, \hat{r}^i, s_H^i)\}_{i=1}^{B}$
14:     Sample batch $\mathbf{B}_{\text{online}}$ from $\mathcal{D}_{\text{replay}}$
15:     Run off-policy RL update on $\mathbf{B}_{\text{online}} \cup \mathbf{B}_{\text{offline}}$ to train $\pi_\psi(z|s)$
16:     Update $\mathcal{L}_\phi$ on $\mathbf{B}_{\text{online}} \cup \mathbf{B}_{\text{offline}}$
17: end for
Ensure: A hierarchical policy consisting of a high-level $\pi_\psi(z|s)$ and low-level $\pi_\theta(a|s,z)$
```

mean-square error loss between the predicted and frozen features on each new transition $(s_0^{\text{new}}, z^{\text{new}}, r^{\text{new}}, s_H^{\text{new}})$:

$$\mathcal{L}_\xi = \|f_\xi(s_0^{\text{new}}, z^{\text{new}}) - \bar{f}_\xi(s^{\text{new}}, z^{\text{new}})\|_2^2 \tag{4}$$

This forms an estimate about the uncertainty of the current state action pair, which can be used to form the optimistic reward estimate to guide exploration:

$$r_{UCB}(s,z) \leftarrow r_\nu(s_0,z) + \alpha\|f_\xi(s_0^{\text{new}}, z^{\text{new}}) - \bar{f}_\xi(s^{\text{new}}, z^{\text{new}})\|_2^2 \tag{5}$$

where α is a hyperparameter controlling the inclination toward exploration.

During the online phase, our framework learns a high-level agent that acts in the skill space, deciding which skills to use every H steps, using the skill policy $\pi_\theta(a|s,z)$ for low level actions between skills. We follow the implementation of SUPE (Wilcoxson et al., 2024), using an SAC agent $\pi_\psi(z|s)$ (Haarnoja et al., 2017) for high-level skill selection.

To aid in efficient policy learning, we leverage the learned dynamics model $D_\phi(s_t, z_t)$ for planning with imaginary rollouts, using the high level policy to guide trajectories, following TD-MPC (Hansen et al., 2022). We use model predictive control (MPC) for skill selection, replanning with the cross entropy method (CEM) (Rubinstein, 1997) at each skill step after performing the first skill in the plan. To evaluate the plan, we use the learned reward model $r_\nu(s_0, z)$ to estimate the rewards of the executed skills in the planning horizon and directly use the learned Q-value from the SAC critic for an estimate of the return beyond the plan. This procedure is described in Algorithm 2 (taken from (Shi et al., 2022)).

Algorithm 2 CEM Planning

```
Require: $\theta, \psi, \phi, \nu$ : learned parameters, $\mathbf{s}_t$: current state
 1: $\mu^0, \sigma^0 \leftarrow \mathbf{0}, \mathbf{1}$
 2: for $i = 1, ..., N_{\text{CEM}}$ do
 3:     Sample $N_{\text{sample}}$ trajectories of length $N$ from
        $\mathcal{N}(\mu^{i-1}, (\sigma^{i-1})^2)$
 4:     Sample $N_\pi$ trajectories of length $N$ using $\pi_\psi, D_\phi$
 5:     Estimate $N$-step returns of $N_{\text{sample}} + N_\pi$ trajectories
        using $r_\nu, Q_\psi$
 6:     Compute $\mu^i, \sigma^i$ with top-k return trajectories
 7: end for
 8: Sample a skill $\mathbf{z} \sim \mathcal{N}(\mu^{N_{\text{CEM}}}, (\sigma^{N_{\text{CEM}}})^2)$
 9: return $\mathbf{z}$
```

5. Experiments

Our experiments aim to evaluate whether our approach can efficiently learn an optimal strategy under long-horizon environments with sparse rewards by leveraging skill abstraction, offline data, and planning. We aim to address the following:

1. Does planning in the skill space with the skill dynamics model lead to faster convergence for long horizon, sparse reward tasks compared to prior methods?

2. How does the availability of relevant, task-specific

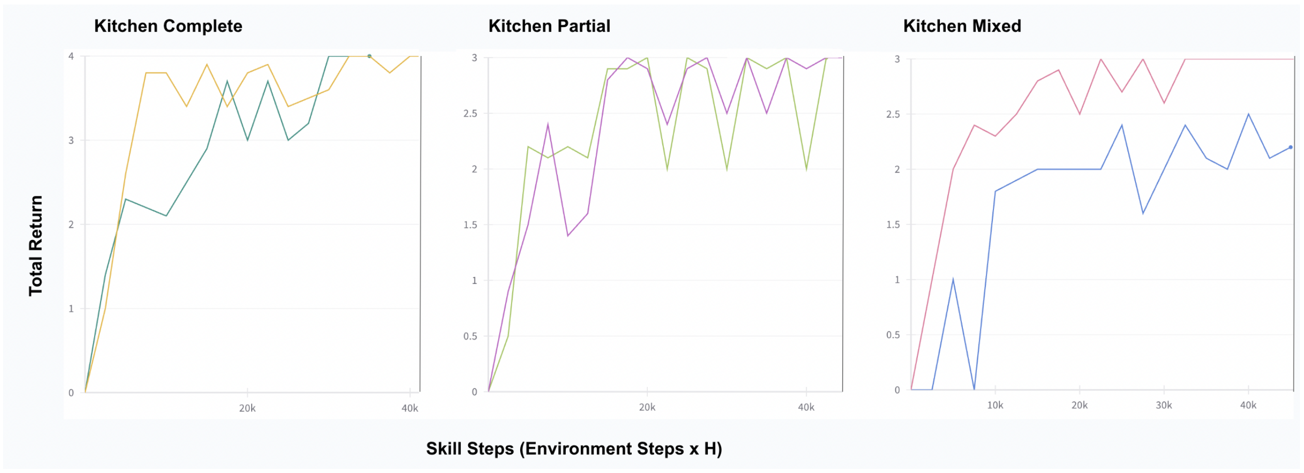

Figure 1. **Total return across the three Kitchen tasks versus Skill Steps (Environment Steps $\times H$. Ours** achieves earlier convergence on **kitchen-complete**; **ours** is in green, **SUPE** is in yellow. **Ours** and **SUPE** achieve approximately equal performance on **kitchen-partial**; **ours** is in green, **SUPE** is in pink. **SUPE** achieves better performance on **kitchen-mixed**; **ours** is in blue, **SUPE is in pink**.

offline data affect prediction accuracy of the dynamics model?

5.1. Tasks

Kitchen FrankaKitchen is a standard benchmark from D4RL (Fu et al., 2020) where the Franka robotic arm performs various tasks in a simulated kitchen. The tasks to be completed are *Microwave, Kettle, Light Switch, Sliding Cabinet Door* in sequence. This is a sparse reward task, with a reward of 1 for completing each sequential task and 0 otherwise. There are three associated datasets, **kitchen-complete**, **kitchen-partial**, and **kitchen-mixed**. **kitchen-complete** is the simplest and contains demonstrations of the four required tasks successfully completed sequentially. **kitchen-partial** only contains some demonstrations of the four required tasks successfully completed sequentially, but also contains additional tasks mixed in with these demonstrations. **kitchen-mixed** contains no successful demonstrations of the four tasks completed in sequence and only contains additional tasks mixed in. For our setting, we remove the reward labels in the offline data

5.2. Baselines

Our baseline is *SUPE* (Wilcoxson et al., 2024), the paper that our work is built off of. *SUPE* extracts reusable skills offline with the trajectory VAE and utilizes the offline data by assigning it an optimistic reward estimate. Different from our method, however, they do not learn a skill dynamics model to plan in the skill space.

5.3. Results

Our results are summarized in *Figure 1*.

kitchen-complete. We demonstrate significantly faster convergence to the optimal policy on this task compared to *SUPE*. Our method is able to successfully learn a skill dynamics model with low prediction error, allowing it to use planning for more sample efficient learning.

kitchen-partial. We demonstrate approximately equivalent performance in terms of sample efficiency and total reward to *SUPE* on this task. Our method likely suffers during training of the skill dynamics model due to the decreased availability of task-specific offline data. Thus, compounding prediction errors lead our method to perform similarly to model-free control.

kitchen-mixed. Our method is outperformed by *SUPE* in terms of both total reward and sample efficiency. Due to unavailability of task-specific, sequentially ordered data relevant to the tasks, this suggests that our skill dynamics model fails to learn accurate state predictions, leading to errors when planning under long horizons.

6. Discussion

In this work, we propose a novel method for efficient reinforcement learning, MoSAUD. We follow prior works to leverage offline data for (1) skill extraction, (2) learning a dynamics model, and (3) off policy data. While we find that our method outperforms current methods in some tasks in terms of sample efficiency, it underperforms in other domains. We believe this is due to the effectiveness of the dynamics model pretraining; **kitchen-complete** has the dataset with the most offline data available and useful for model pretraining while **kitchen-mixed** has much more data that can *corrupt* the dynamics model with transition prediction errors. Additionally, there is less relevant data for learning the dynamics model on the correct sequence of tasks, leading to higher prediction error on **kitchen-mixed**. The framework of skill learning is a promising avenue for sample efficiency, and we advance its applicability by furthering this direction on unlabeled prior data.

References

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *Arxiv*, 2018. URL https://arxiv.org/pdf/1810.12894.

Marc Peter Deisenroth and Carl Edward Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning*, 2011. URL https://mlg.eng.cam.ac.uk/pub/pdf/DeiRas11.pdf.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control, 2016.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. 2017.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *ICML*, 2022.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

Jiaheng Hu, Zizhao Wang, Peter Stone, and Roberto Martín-Martín. Disentangled unsupervised skill discovery for efficient hierarchical reinforcement learning, 2024.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *n International Conference on Learning Representations*, 2014. URL https://arxiv.org/pdf/1312.6114.

Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Cic: Contrastive intrinsic control for unsupervised skill discovery, 2022.

Qiyang Li, Jason Zhang, Dibya Ghosh, Amy Zhang, and Sergey Levine. Accelerating exploration with unlabeled prior data. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Itorzn4Kwf.

Karl Pertsch, Youngwoon Lee, and Joseph J. Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on Robot Learning (CoRL)*, 2020.

R. Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research, 99(1):89–112*, 1997.

Lucy Xiaoyang Shi, Joseph J. Lim, and Youngwoon Lee. Skill-based model-based reinforcement learning. In *Conference on Robot Learning*, 2022.

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

Max Wilcoxson, Qiyang Li, Kevin Frans, and Sergey Levine. Leveraging skills from unlabeled prior data for efficient online exploration. In *Arxiv*, 2024. URL https://arxiv.org/abs/2410.18076.

Kai Zhao, Jianye Hao, Yi Ma, Jinyi Liu, Yan Zheng, and Zhaopeng Meng. Enoto: Improving offline-to-online reinforcement learning with q-ensembles, 2023.

Han Zheng, Xufang Luo, Pengfei Wei, Xuan Song, Dongsheng Li, and Jing Jiang. Adaptive policy learning for offline-to-online reinforcement learning, 2023.