Meta-Reinforcement Learning Based on Self-Supervised Task Representation Learning

Mingyang Wang[1], Zhenshan Bing[1], Xiangtong Yao[1], Shuai Wang[2], Huang Kai[3, 4], Hang Su[5], Chenguang Yang[6], *Alois Knoll[1]

[1]Department of Informatics, Technical University Munich,
[2]Tencent Robotics X Lab,
[3]School of Computer Science and Engineering, Sun Yat-Sen University,
[4]Shenzhen Institute, Sun Yat-Sen University
[5]Dipartimento di Elettronica, Politecnico di Milano,
[6]Bristol Robotics Laboratory, University of the West of England
mingyang.wang@tum.de, bing@in.tum.de, xiangtong.yao@tum.de, shuaiwanghit@gmail.com,
hang.su@polimi.it, cyang@ieee.org, huangk36@mail.sysu.edu.cn, knoll@in.tum.de

Abstract

Meta-reinforcement learning enables artificial agents to learn from related training tasks and adapt to new tasks efficiently with minimal interaction data. However, most existing research is still limited to narrow task distributions that are parametric and stationary, and does not consider out-of-distribution tasks during the evaluation, thus, restricting its application. In this paper, we propose MoSS, a context-based **M**eta-reinforcement learning algorithm based **o**n **S**elf-**S**upervised task representation learning to address this challenge.We extend meta-RL to broad non-parametric task distributions which have never been explored before, and also achieve state-of-the-art results in non-stationary and out-of-distribution tasks. Specifically, MoSS consists of a task inference module and a policy module. We utilize the Gaussian mixture model for task representation to imitate the parametric and non-parametric task variations. Additionally, our online adaptation strategy enables the agent to react at the first sight of a task change, thus being applicable in non-stationary tasks. MoSS also exhibits strong generalization robustness in out-of-distributions tasks which benefits from the reliable and robust task representation. The policy is built on top of an off-policy RL algorithm and the entire network is trained completely off-policy to ensure high sample efficiency. On MuJoCo and Meta-World benchmarks, MoSS outperforms prior works in terms of asymptotic performance, sample efficiency (3-50x faster), adaptation efficiency, and generalization robustness on broad and diverse task distributions.

Introduction

Modern deep reinforcement learning (RL) has made significant progress in learning complex behavior (Mnih et al. 2015; Silver et al. 2016, 2017; Bing et al. 2021a). However, they typically do not transfer learned skills to other tasks, require to re-train the policy for new tasks. In contrast, humans can learn new skills efficiently using prior knowledge and experience. Motivated by this, meta-reinforcement learning (meta-RL) was developed to mimic the human learning process by learning a prior model from a set of related training tasks and quickly adapt to unseen tasks during testing.

However, most existing meta-RL studies are severely confined to narrow task distributions that are parametric and stationary, let alone taking into consideration out-of-distribution (OOD) test tasks. In prior works (Finn, Abbeel, and Levine 2017; Stadie et al. 2018; Nichol and Schulman 2018; Rothfuss et al. 2019; Duan et al. 2016; Gupta et al. 2018; Rakelly et al. 2019; Fakoor et al. 2019; Humplik et al. 2019; Li, Yang, and Luo 2020; Zintgraf et al. 2020, 2021), task distributions only involve parametric variations, e.g., simulated robots can adapt to reach a new goal velocity after being trained on a set of different goal velocities previously. It is therefore unreasonable to expect generalization to a completely new control task, e.g., reaching a specified goal position. To extend the meta-RL adaptation to qualitatively distinct tasks, training tasks should also include such non-parametric variability in which task differences cannot be expressed only using continuous parameters. Also, humans can quickly adapt their behavior to unexpected changes and perturbations, we expect the meta-RL algorithm to be broadly useful in non-stationary scenarios where the task may vary at any time step and exhibit strong generalization robustness for out-of-distribution tasks.

As a promising approach to tackling the meta-RL problem, context-based meta-RL (Duan et al. 2016; Rakelly et al. 2019; Fakoor et al. 2019; Humplik et al. 2019; Zintgraf et al. 2020; Guo, Wu, and Lee 2022) extracts salient information from past experience and generates latent representations on which the policy is then conditioned. However, representing tasks on broad task distributions poses new challenges to the model as more complex relationships and dependencies between tasks must be captured. Existing studies (Rakelly et al. 2019; Zintgraf et al. 2020; Fakoor et al. 2019; Li, Yang, and Luo 2020) often uses a single-component Gaussian for task representation which is inadequate in such complex cases.

Furthermore, current meta-RL algorithms suffer from sample inefficiency due to on-policy optimization (Finn, Abbeel, and Levine 2017; Stadie et al. 2018; Nichol and Schulman 2018; Rothfuss et al. 2019; Duan et al. 2016;

*Corresponding Author

Gupta et al. 2018; Zintgraf et al. 2020, 2021), or adaptation inefficiency due to the few-shot adaptation strategy (Rakelly et al. 2019; Fakoor et al. 2019; Li, Yang, and Luo 2020). Although on-policy RL approaches are easy to incorporate into meta-RL, they require a large quantity of data for training, making them sample inefficient. Recent works (Rakelly et al. 2019; Fakoor et al. 2019; Li, Yang, and Luo 2020) integrated the off-policy RL algorithm into the meta-RL framework, resulting in a significant increase in sample efficiency. However, they still suffer from adaptation inefficiency as task representation is updated at the trajectory level, thus enable to accomplish non-stationary tasks.

With these considerations in mind, we propose MoSS, a context-based meta-RL algorithm based on self-supervised task representation learning. The method aims to be (1) applicable to diverse task distributions with parametric and non-parametric, stationary and non-stationary, in-distribution, and out-of-distribution tasks. The general performance of meta-RL algorithms on such broad task distributions has never been explored before, and (2) sample efficient by using as few data samples as possible for training, and adaptation efficient by taking as few environmental steps as possible to adapt to a new task during meta-testing.

MoSS is made up of two modules: a task inference module that takes the task trajectory as input and encodes it as latent variables, and a policy module that uses the task representation to explore the environment and learn optimal actions. First, in the task inference module, we realize representative and robust task inference using the Gaussian mixture latent space for task representation and the contrastive learning strategy. By extending the VAE-based inference network with a Gaussian mixture latent space, MoSS accommodates non-parametric variations using different Gaussian clusters and parametric variations with variability within each Gaussian component. Additionally, contrastive learning enhances the ability of MoSS to differentiate different tasks while clustering similar ones. Reliable task representation is a prerequisite for effective downstream RL. Second, we use the gated recurrent unit (GRU) (Chung et al. 2014) as the encoder and optimize it with the next-step reconstruction loss to achieve online adaptation. Unlike prior works (Rakelly et al. 2019; Zintgraf et al. 2020) that generate task variables based on trajectory-level context, our encoder updates latent variables based on local context, so that the agent can quickly adapt to new tasks and solve non-stationary tasks. Third, we build our policy on top of the soft-actor-critic(SAC) algorithm and condition the policy on the agent state and the latent task variable, to account for task uncertainty in its decision-making. We use a shared data buffer for task inference and policy training to realize a fully off-policy optimization that ensures high sample efficiency. We evaluate MoSS on MuJoCo (Todorov, Erez, and Tassa 2012) and Meta-World (Yu et al. 2020) benchmarks, including various robotic control and manipulation tasks. MoSS shows state-of-the-art results in asymptotic performance, sample and adaptation efficiency, and generalization robustness[1].

[1]Implementation and videos available at https://sites.google.com/view/metarl-moss

Background

From RL to Meta-RL

In standard RL, a task is formulated as a Markov Decision Process (MDP) (Bellman 1966) denoted as $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ where $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, $P(\boldsymbol{s}^{'}|\boldsymbol{s}, \boldsymbol{a})$ denoting the transition function, $R(r|\boldsymbol{s}, \boldsymbol{a})$ the reward function and γ the discount factor. The objective is to find an optimal policy π that maximizes the expected return in this MDP. Original RL algorithms are mostly trained on one task at a time. While meta-RL leverages a set of related tasks to learn a policy that can quickly adapt to unseen test tasks. Specifically, during meta-training, the algorithm has access to N_{train} tasks drawn from the task distribution $p(M)$. At meta-test time, new tasks are also sampled from $p(M)$. The meta-trained policy should quickly adapt to new tasks to achieve maximum return, i.e., $\boldsymbol{\theta}_* = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{M\sim p(M)}\Big[\mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau}|\pi_{\boldsymbol{\theta}})}\big[\sum_{t\geq 0}\gamma^t r_t\big]\Big]$. Here $\boldsymbol{\tau}_{:t} = \{\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1}\}_{0:t}$ represents the task trajectory that collects the agent's transitions $(\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1})$ up to the current time step.

Meta-RL Environments

Parametric and non-parametric variability The variability of task distributions is a key property in the meta-RL context. The formal definition of two properties of the task distribution is given in (Yu et al. 2020) based on the kind of structure tasks have in common. **Parametric variability** describes tasks that qualitatively share the same property, but the task parameterization varies, while **non-parametric variability** describes tasks that are qualitatively distinct. For example, in the *Cheetah-Multi-Task* benchmark, the *Cheetah-Velocity* task aims to control the robot to run at different goal velocities. All tasks essentially share the same qualitative task descriptions and are parameterized by the goal velocity. While *Cheetah-Velocity* and *Cheetah-Goal* tasks are qualitatively distinct. Although they both control the robot to exhibit certain behaviors, the difference between them cannot be described by parametric variations. Thus, they belong to different task families (or base tasks).

Adaptation in meta-RL environments When being exposed to new tasks, the meta-RL agent is usually allowed to collect context data for a few episodes to get an increasingly better task understanding. We use the term k-shot to represent the number of exploration episodes. The agent should identify tasks and adapt the policy within k episodes, so k is an important metric to evaluate the adaptation efficiency of meta-RL algorithms. Note that we use $k = 0$ for MoSS, i.e., reported results are from the first evaluation rollout without any previous data collection, which is a harder experiment setting but also more realistic as the agent should solve the task at first sight without any failed trials.

Related Work

Recent studies in meta-RL can be separated into two categories: optimization-based and context-based meta-RL. Optimization-based meta-RL (Finn, Abbeel, and Levine

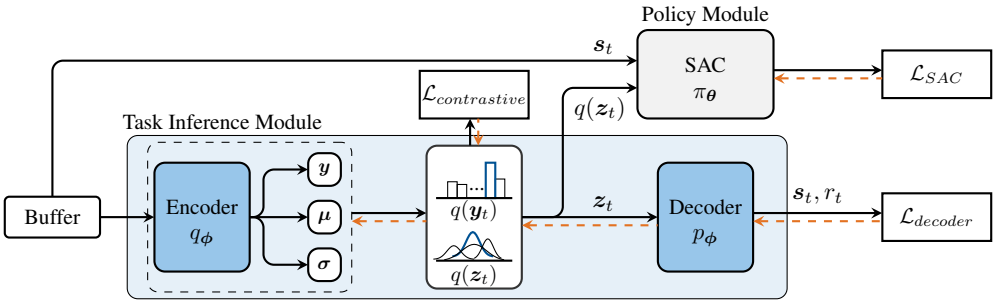


Figure 1: MoSS Architecture: MoSS consists of a task inference module and a policy module. The inference module encodes the task context $\boldsymbol{\tau}$ as latent task distributions $q_{\boldsymbol{\phi}}(\boldsymbol{z})$, then the policy module conditions on the agent state $\boldsymbol{s}$ and the task representation $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ to act in the environment.

2017; Stadie et al. 2018; Nichol and Schulman 2018; Rothfuss et al. 2019) learns a prior model based on training tasks and then performs task-specific fine-tuning on unseen tasks to quickly adapt to new tasks with a few gradient descent steps. Although the idea is conceptually elegant, the gradient descent update is uninformative for RL algorithms, making it hard to explore a new, unknown environment. In contrast, context-based meta-RL (Duan et al. 2016; Rakelly et al. 2019; Fakoor et al. 2019; Humplik et al. 2019; Zintgraf et al. 2020; Guo, Wu, and Lee 2022; Bing et al. 2022b, 2021b, 2022a; Yao et al. 2022) adapts to new tasks by aggregating past experience into latent representations on which the policy is then conditioned. Such methods realize a better exploration-exploitation trade-off, as they maintain belief over possible MDPs that enable reasonable exploration by acting optimally according to these MDPs. Even without any explicit fine-tuning, the meta-RL model can capture some task characteristics and adapt to new test tasks.

As a context-based meta-RL method, RL^2 (Duan et al. 2016) first proposed to condition the policy on hidden states encoded from task trajectory. By structuring the agent as a recurrent neural network, RL^2 enables online policy learning based on RNN dynamics. Another popular training mechanism is to disentangle the task inference and action selection process to enable explicit task representations. In PEARL (Rakelly et al. 2019), task context is encoded as probabilistic latent variables via variational inference. Then using posterior sampling, latent task encodings are integrated with RL policy and guide the agent to act in the environment. In the following work (Humplik et al. 2019), the task inference network is trained in a supervised way and the policy directly conditions the posterior distributions to reason for task uncertainty. Closely related to our approach is VariBAD (Zintgraf et al. 2020), which adopts VAE for task inference. The encoder processes the task context and generates latent task distributions, based on which the decoder predicts past and future states and rewards. The network is trained based on the prediction loss rather than using RL-loss (Rakelly et al. 2019) or using privileged task information(Humplik et al. 2019). Inspired by VariBAD, MoSS also uses VAE for task inference. However, it uses a different encoder-decoder training strategy. MoSS only reconstructs the next-step state and reward instead of modeling all past and future steps, i.e., we do not assume the task within one episode is always consistent. Therefore, MoSS can adjust its task belief to potential task changes within a single episode and is applicable to non-stationary environments.

All the studies mentioned above only focus on the performance of meta-RL algorithms on commonly used parametric task distributions and do not explore the applicability of these algorithms on broad task distributions with non-parametric task variability, non-stationary environments, and out-of-distribution evaluation tasks. Some other works discuss the meta-RL performance in non-stationary or OOD tasks (Nagabandi et al. 2018; Fakoor et al. 2019; Mendonca et al. 2020; Lee and Chung 2021). However, non-parametric task distributions have never been explored specifically, and no studies discuss these scenarios together to investigate the algorithms' general performance. For the first time, MoSS explores the meta-RL algorithm applicable to broad and diverse task distributions.

Methodology

In the following, we first give an overview of our MoSS algorithm. Then we explain the strategy of MoSS in the task inference module and policy module to make it applicable to the aforementioned task distributions with good sample and adaptation efficiency. We also summarize the meta-training procedure of MoSS as pseudo-code in Algorithm 1.

Algorithm Overview

MoSS disentangles task inference from policy control as in (Rakelly et al. 2019; Zintgraf et al. 2020; Zhao et al. 2020). The task inference module $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{\tau}_{:t}))$, parameterized by $\boldsymbol{\phi}$, encodes the task trajectory $\boldsymbol{\tau}_{:t}$ as latent task distributions $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t))$, and it is trained using a decoder $p_{\boldsymbol{\phi}}(\boldsymbol{s}_{t+1}, r_t))$ that predicts next-step states and rewards from current states and actions. The policy module $\pi_\theta(a_t|s_t, q_{\boldsymbol{\phi}}(\boldsymbol{z}_t))$, parameteried by $\boldsymbol{\theta}$, conditions on the agent state $\boldsymbol{s}_t$ with the task belief $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t)$ to act in the environment.

The model architecture of MoSS is shown in Figure 1. To accommodate both the parametric and non-parametric task variations in the task inference module, we extend the VAE architecture with Gaussian mixture latent space for task representation. To further improve the model's ability to capture complex relationships between tasks, inspired by (Guo, Wu,

Algorithm 1: MoSS Meta-training

Input: Task distribution $p(M)$, encoder $q_{\boldsymbol{\phi}}$, decoder $p_{\boldsymbol{\phi}}$, actor $\pi_{\boldsymbol{\theta}}$, critic $Q_{\boldsymbol{\omega}}$, replay buffer $\mathcal{B}$

while not done **do**
  Sample tasks $\boldsymbol{M} = \{M_i\}_{i=1}^{N}$ from $p(M)$
  Collect trajectories with π_θ and add to buffer $\mathcal{B}$ ▷ *Data collection*
  for step in training steps **do** ▷ *Training step*
    Sample training tasks $\boldsymbol{M}_{train}$ from $p(M)$
    for $M_i \in \boldsymbol{M}_{train}$ **do**
      Sample $\boldsymbol{\tau}_{:T} \sim \mathcal{B}_i$ with trajectory length T
      Infer task beliefs $\{q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{\tau}_{:t})\}_{0:T-1}$
      Calculate $ELBO_i = \sum_{t=0}^{T-1} ELBO_{i,\,t}(\boldsymbol{\phi})$ using Eq. 1
    end for
    Calculate contrastive loss $\mathcal{J}_{cont}$ using Eq. 2
    Update $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \alpha_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\phi}} (\sum_i ELBO_i + \mathcal{J}_{cont})$ ▷ *Inference network update*
    Update $(\boldsymbol{\theta}, \boldsymbol{\omega})$ with SAC algorithm ▷ *SAC update*
  end for
end while

and Lee 2022; Dorfman, Shenfeld, and Tamar 2021),we introduce contrastive loss as an auxiliary objective and jointly train the task inference network in a self-supervised manner. Moreover, to enable online adaptation, we use GRU (Chung et al. 2014) as our encoder and update task representations at the transition level. In the policy module, following the Bayes-Adaptive MDP setting (Duff 2002; Ghavamzadeh et al. 2015; Zintgraf et al. 2020), we condition the policy on the agent state augmented with the inferred task distribution to incorporate task uncertainty in its action selection process. Finally, we use a shared replay buffer for both modules and train the network in a fully off-policy manner.

MoSS works as follows: during meta-training, we sample trajectory data from training tasks and feed them into the task inference module, which learns to infer latent variables $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t)$ via self-supervised representation learning (details see Section). Also, the policy network $\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t, q_{\boldsymbol{\phi}}(\boldsymbol{z}_t))$ takes in the agent state $\boldsymbol{s}_t$ with the current task belief $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t)$ and learns to select optimal actions. While meta-testing, the encoder generates and updates task representations $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{\tau}_{:t})$ at each time step based on the collected task experience $\boldsymbol{\tau}_{:t}$. Then the policy $\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t, q_{\boldsymbol{\phi}}(\boldsymbol{z}_t))$ conditions its actions on the agent state $\boldsymbol{s}_t$ augmented with the task belief $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t)$ to interact with the environment.

Task Inference Module

VAE with Gaussian Mixture Latent Space Previous context-based meta-RL works (Rakelly et al. 2019; Zhao et al. 2020; Zintgraf et al. 2020; Humplik et al. 2019) use single-component Gaussian distributions for task representation. However, representing tasks from complex task distributions poses new challenges to the inference network and single-component Gaussian-based representation is therefore insufficient. To address this problem, we construct a Gaussian mixture latent space that accommodates non-parametric variability with different Gaussian clusters and parametric variability using variations within each Gaussian component. We use a categorical variable $\boldsymbol{y} \sim Cat(\pi)$ to indicate the base task probability and latent variables $\boldsymbol{z} \sim N(\mu(\boldsymbol{y}), \sigma(\boldsymbol{y}))$ as base task-specific Gaussian components. Together with the input $\boldsymbol{x}$, the joint probability is factorized as $p(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = p(\boldsymbol{y})p(\boldsymbol{z}|\boldsymbol{y})p(\boldsymbol{x}|\boldsymbol{z})$ (Jiang et al. 2016; Rao et al. 2019). The posterior inference of $p(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$ is intractable, we employ the learned approximate posterior $q(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) = q(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})q(\boldsymbol{y}|\boldsymbol{x})$. The posterior inference and data generation process run as follows: first, given the input $\boldsymbol{x}$, the cluster inference model $q(\boldsymbol{y}|\boldsymbol{x})$ produces the categorical distribution $q(\boldsymbol{y})$, and task encoding models $q(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y}^{(k)})$ generate K independent Gaussians for each cluster. Then, the decoder $p(\boldsymbol{x}|\boldsymbol{z})$ reconstructs data $\boldsymbol{x}$ from the latent variable $\boldsymbol{z}$. Following the variational inference approach and Monte-Carlo approximation, the evidence lower bound objective (ELBO) is formulated as (derivation see Appendix):

$$
\begin{aligned}
ELBO \approx \sum_{k=1}^{K} & \overbrace{q(\boldsymbol{y}^{(k)}|\boldsymbol{x})}^{\text{Component posterior}} \Big[\overbrace{\log p(\boldsymbol{x}|\boldsymbol{z}^{(k)})}^{\text{Component-wise reconstruction loss}} \\
& - \alpha \overbrace{\mathbb{KL}(q(\boldsymbol{z}^{(k)}|\boldsymbol{x}, \boldsymbol{y}^{(k)}) \| p(\boldsymbol{z}|\boldsymbol{y}^{(k)}))}^{\text{Component-wise regularizer}} \Big] \\
& - \beta \overbrace{\mathbb{KL}(q(\boldsymbol{y}|\boldsymbol{x}) \| p(\boldsymbol{y}))}^{\text{Categorical regularizer}}
\end{aligned}
\tag{1}
$$

Here α and β represent the regularization weight for the KL divergence term of the component-wise Gaussian distribution and the categorical distribution, respectively. Intuitively, the model can either have high entropy over $q(\boldsymbol{y}|\boldsymbol{x})$, where all component-wise losses should be low, or assign a high $q(\boldsymbol{y}^{(k)}|\boldsymbol{x})$ for some k and use that component to model the data well. Here we introduce hyperparameters α and β to weight the component-wise and categorical regularization terms. We parameterize the decoder as two independent networks: state decoder $p_{\boldsymbol{\phi}}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{z}_t)$ and reward decoder $p_{\boldsymbol{\phi}}(\boldsymbol{r}_t|\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{z}_t)$. Both networks are modeled as regression networks, using the MSE loss between predictions ($\hat{\boldsymbol{s}}_{t+1}$, $\hat{\boldsymbol{r}}_t$) and true targets ($\boldsymbol{s}_{t+1}$, r_t) from buffer as supervision, therefore the networks are trained in a self-supervised manner. Unlike prior works (Duan et al. 2016; Rakelly et al. 2019), we do not back-propagate the RL loss into the encoder, which completely decouples task inference from policy learning. This modular structure facilitates the encoder to identify tasks only based on task characteristics. The decoder is not used at meta-test time. During meta-test, given the trajectory up to the current time step $\boldsymbol{\tau}_{:t}$, the encoder predicts the $p(\boldsymbol{y})$ distribution and Gaussian parameters $\{\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k\}_{k=1\cdots K}$. We pick the Gaussian component $q(\boldsymbol{z}_t) = N(\boldsymbol{\mu}_{\boldsymbol{z}}(\boldsymbol{y}_t(k^*)), \boldsymbol{\sigma}_{\boldsymbol{z}}(\boldsymbol{y}_t(k^*)))$ corresponding to the most likely base task $k^* = \arg\max_k \{q(\boldsymbol{y}_t^{(k)}|\boldsymbol{\tau}_{:t})\}_{k=1\cdots K}$ and get the best matching Gaussian at each time step.

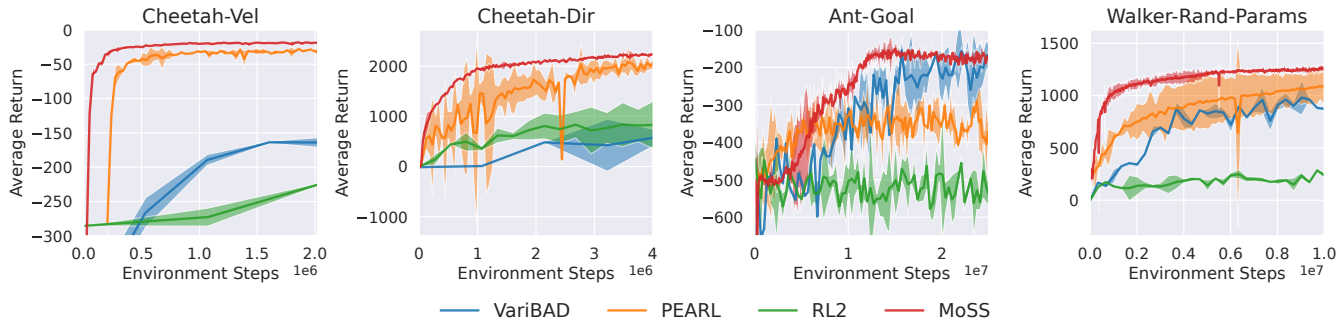


Figure 2: Meta-test performance in parametric MuJoCo environments: Average return (y-axis) against collected environment steps during meta-training (x-axis).

Contrastive Representation Learning Essentially, we expect the encoder to cluster similar tasks and distinguish different tasks by organizing their latent embeddings in a structured way. This coincides with the idea of contrastive learning, which learns a latent embedding space where similar sample pairs stay close while dissimilar ones are far apart. Motivated by (Guo, Wu, and Lee 2022; Dorfman, Shenfeld, and Tamar 2021), we introduce a contrastive learning objective for task inference training to realize better task identification. In contrastive learning, input data are usually organized into positive and negative pairs. We use the key-query definition in (Wu et al. 2018; Tian et al. 2020; Wang et al. 2021): a key-query pair is positive if they are representations belong to the same data instance and negative otherwise. Given a query $\boldsymbol{Q}$ and a set of keys $\mathbb{K} = \{\boldsymbol{k}_i\}_{i=0...N_K}$ consisting of 1 positive key and $N_K - 1$ negative keys, contrastive learning aims to ensure that the query $\boldsymbol{Q}$ matches with the positive key $\boldsymbol{k}_+$ more than any of the negative keys.

We use task embeddings $\boldsymbol{z}$ as queries and keys. Embeddings from the same task (but different time steps) build positive pairs, while embeddings from different tasks are negative pairs. Specifically, we sample two latent embeddings of different time steps from each task as the query and its corresponding positive key, and sample $N_K - 1$ embeddings from other tasks as negative keys. We use the InfoNCE score (Oord, Li, and Vinyals 2018) with the Euclidean distance as the similarity metric to calculate the contrastive objective:

$$\mathcal{J}_{cont} = \frac{1}{N_Q} \sum_{i=0}^{N_Q-1} \log \frac{\exp(\text{sim}(\boldsymbol{Q}_i, \boldsymbol{k}_+))}{\sum_{j=0}^{N_K-1} \exp(\text{sim}(\boldsymbol{Q}_i, \boldsymbol{k}_j))} \quad (2)$$

Here N_Q is the number of sampled queries, and N_K is the number of keys for each query. The objective can be interpreted as the log-likelihood of an N_K-way softmax classifier where $\boldsymbol{k}_+$ is the label of the corresponding query. Then we get the final objective $\mathcal{J}(\boldsymbol{\phi}) = ELBO + \mathcal{J}_{cont}$.

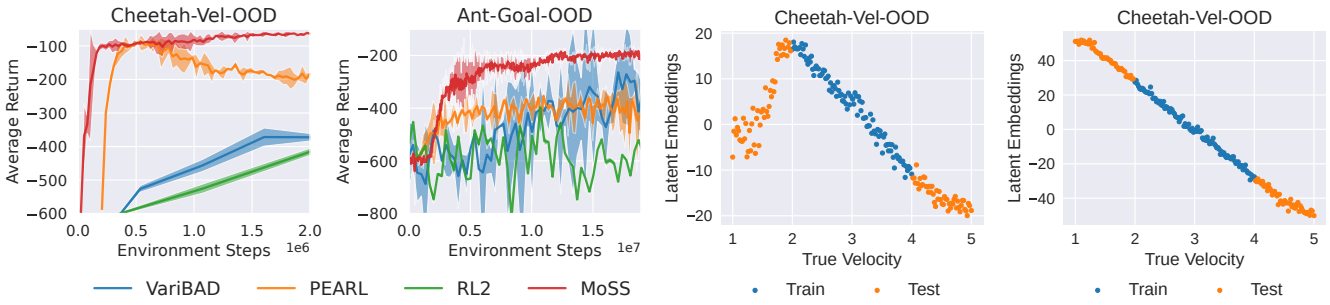
RNN-based Online Inference Moreover, we introduce a recurrent version of the VAE (Chung et al. 2015) to map the sequential input to time step-wise latent variables. Specifically, we model a VAE at each time step and explicitly study the dependencies between latent variables across consecutive time steps. Instead of sharing a global prior distribution $p_0(\boldsymbol{y})$ and $p_0(\boldsymbol{z})$ for $\boldsymbol{y}$ and $\boldsymbol{z}$ of all time steps (as in (Rakelly et al. 2019)), we use posterior distributions from the previous time step $q(\boldsymbol{z}|\boldsymbol{x}_{<t}, \boldsymbol{y})$ and $q(\boldsymbol{y}|\boldsymbol{x}_{<t})$ as current prior. The resulting time step-wise variational lower bound ELBO is given in Appendix.

We use the GRU (Chung et al. 2014) as the encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{\tau}_{:t})$. By recursively reasoning with its hidden states $q_\phi(\boldsymbol{z}_t|\boldsymbol{\tau}_{:t}) = q_\phi(\boldsymbol{z}_t|\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{r}_t, \boldsymbol{h}_{t-1})$, MoSS combines historical information with the current input and adjusts actions step by step. On the one hand, it enables more efficient task inference and policy adaptation. On the other hand, MoSS is, therefore, applicable to non-stationary environments where the task can potentially change at any time step.

Policy Module

We adopt the same strategy with (Zintgraf et al. 2020) in the policy module by interpreting the meta-RL problem as a planning problem in Bayes-Adaptive MDPs (BAMDPs) (Duff 2002). A BAMDP is described by $(\mathcal{S}^+, \mathcal{A}, P, R, \gamma)$ where the hyper-state $\boldsymbol{s}_t^+ \in \mathcal{S}^+ = \mathcal{S} \times \mathcal{B}$ consists of the agent state $\boldsymbol{s}_t$ and the task belief $\boldsymbol{b}$. In BAMDPs, the RL policy conditions not only on the agent state but also on the agent's belief about the environment. A Bayes-optimal agent learns to maximize the expected return by systematically seeking out actions to reduce its environmental uncertainty (*exploration*) and then taking promising actions to maximize the expected return (*exploitation*) (Zintgraf et al. 2020). Thus, the policy automatically learns how to trade off exploration and exploitation under task uncertainty.

In MoSS, the hyper-state is denoted as $\boldsymbol{s}_t^+ = (\boldsymbol{s}_t, q_{\boldsymbol{\phi}}(\boldsymbol{z}_t))$ with $q_{\boldsymbol{\phi}}(\boldsymbol{z}_t)$ from the task inference module. It directly incorporates the task uncertainty, unlike the policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{z})$ in (Gupta et al. 2018; Rakelly et al. 2019; Zhao et al. 2020) uses $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})$ via posterior sampling. We build our policy on top of the SAC (Haarnoja et al. 2018) and use a shared data buffer for VAE and RL training to realize fully off-policy training, unlike (Rakelly et al. 2019) that still uses an on-policy VAE training buffer, which makes MoSS outperform prior algorithms with respect to sample efficiency.



(a) Training Curve (b) PEARL embeddings (c) MoSS embeddings

Figure 3: Meta-test performance in parametric MuJoCo environments with OOD tasks

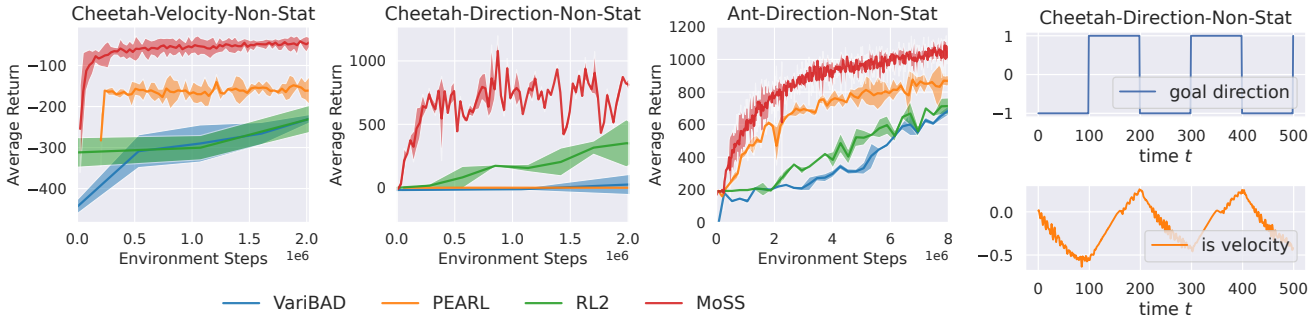


Figure 4: Meta-test performance in non-stationary MuJoCo environments: (Left) Training curves; (Right) Agent Response in the *Cheetah-Direction-Non-Stat* environment.

Experiments

We evaluate the performance of MoSS on Mujoco (Todorov, Erez, and Tassa 2012) and Meta-World (Yu et al. 2020) benchmarks. We compare MoSS with RL^2, PEARL and VariBAD (Duan et al. 2016; Rakelly et al. 2019; Zintgraf et al. 2020). First, we run experiments on commonly used parametric task distributions. Additionally, to verify the broad applicability of MoSS, we run experiments on diverse task distributions with out-of-distribution test tasks, non-stationary environments and non-parametric variations. We also evaluate several design choices of MoSS through ablation in Appendix. Note that we conduct the zero-shot meta-test on MoSS ($k = 0$). In contrast, for MuJoCo tasks, VariBAD and RL^2 use $k = 2$ and PEARL uses $k = 3$. For Meta-World tasks numbers are given in Table 1. All the choices of k are the same as their original papers. We truncate the $x-$axis at the number of time steps required for MoSS to converge; for the full timescale results see Appendix. Other hyperparameters can be found in Appendix.

Parametric task distributions To validate the performance of MoSS in parametric environments, we first compare MoSS with baseline methods on MuJoCo and Meta-World ML1 tasks. Results are given in Figure 2 and Table 1. MuJoCo locomotion tasks require the model to adapt across various reward functions or system dynamics (task details see Appendix). Meta-World ML1 benchmark consists of three robotic manipulation tasks where task variations are specified goal positions and use success rate as the evaluation metric. Training curves are given in Appendix.

Method	k-th episode	Reach	Push	Pick-Place
RL^2	10	**100**	96	98
PEARL	10	68	44	28
MoSS	**1**	86	**100**	**100**

Table 1: Meta-World V2 ML1 result comparison: Success rate results are given in percentage [2]

Parametric Task Distributions with Out-of-Distribution Test Tasks To further investigate the generalization robustness of our algorithm, we set up two OOD environments where we distinguish task ranges during meta-training and meta-test. For example, in *Cheetah-Vel-OOD*, we train the agent on the velocity range of $[2.0, 4.0]$ and test it on $[1.0, 2.0] \cup [4.0, 5.0]$. For the detailed setup see Appendix.

As plotted in Figure 3 (left), MoSS outperforms baseline methods as it achieves higher average returns and better sta-

[2]Baseline results are taken from (Yu et al. 2020) as the performance is highly sensitive to hyperparameters and hard to reproduce. VariBAD (Zintgraf et al. 2020) only uses the original Meta-World V1, for a fair comparison we also include results comparison on Meta-World V1 in Appendix.

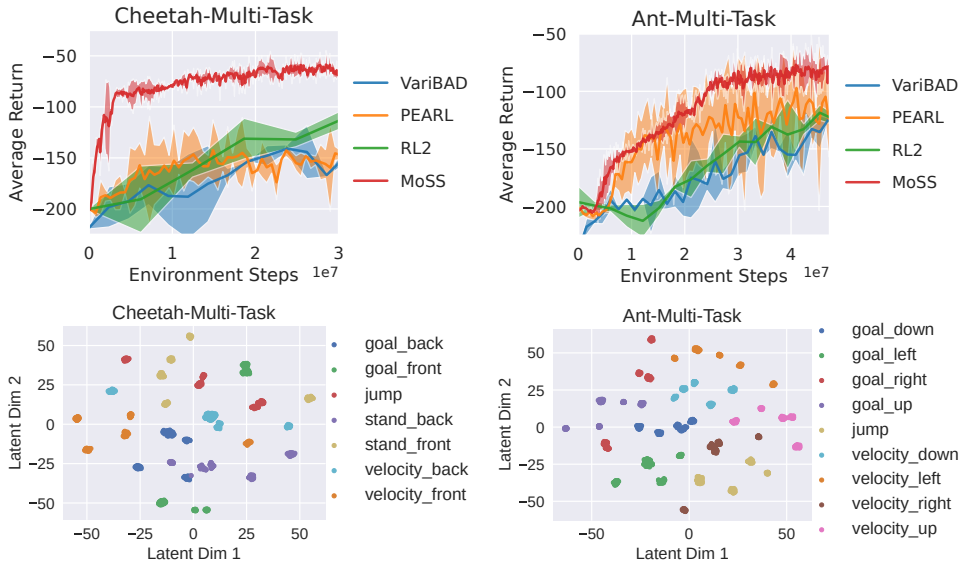


Figure 5: Meta-test performance in non-parametric MuJoCo environments

bility. We suppose this comes from the robust task representation of MoSS. Faced with new OOD test tasks, MoSS can still capture the task characteristics, which share similarity with previously seen tasks, and produce understandable representations for the downstream policy network. In Figure 3 (right), we visualize the latent space of PEARL and MoSS on *Cheetah-Vel-OOD* to demonstrate our assumption. The $x-$axis is the ground-truth target velocity and the $y-$axis is the t-SNE (Van der Maaten and Hinton 2008) visualization of latent embeddings (1D). Over the entire velocity range, MoSS shows a clear linear relationship between the latent embeddings and true targets. While PEARL struggles to figure out how to represent OOD test tasks, especially in the velocity range of $[1.0, 2.0]$, which makes it hard for the RL policy to learn optimal actions under unclear instruction.

Parametric, Non-stationary Task Distributions We also set up three non-stationary environments where the goal velocity or direction changes randomly every 100 time step. We choose the changing steps of 100 as the agent needs some time steps to adjust its behavior. For this reason, we increase the time horizon to 500 600 in non-stationary tasks. Additionally, at every 100 steps we hold a change probability $p = 0.5$, i.e., the task changes at every 100 time step at the probability of 0.5. Randomness is necessary for non-stationary tasks otherwise the agent would remember the changing patterns and cannot adapt well to dynamic changes in more general non-stationary environments. Given the changing randomness, the task is different in each episode so that the agent will not remember the fixed changing pattern. In Figure 4, we plot the experiment results and visualize the agent response to task changes in *Cheetah-Direction-Non-Stat*, more visualizations see Appendix.

Non-parametric Task Distributions Finally, we set up two non-parametric benchmarks *Cheetah-Multi-Task* and *Ant-Multi-Task*. Both of them consist of multiple base tasks that are qualitatively distinct and sub-tasks inside each base task. The detailed task description is given in Appendix.

As Figure 5 shows, MoSS outperforms prior works in both environments. We use t-SNE (Van der Maaten and Hinton 2008) to visualize the latent embeddings of MoSS in two dimensions in Figure 5. We use different colors to represent sub-tasks with different goal directions from the same base task as they usually exhibit dissimilar task characteristics, e.g., *goal_back* and *goal_front* are both in base task *Cheetah-Goal*, but we plot them in different colors. Each refined base task consists of 5 sub-tasks with different goal values corresponding to small clusters in the picture. MoSS can differentiate different tasks and cluster embeddings from the same task. Given explicit task representations as guidance, it is easier for the policy to learn optimal actions.

Conclusion

In this paper, we propose MoSS, an algorithm to address the meta-RL problem on diverse task distributions with superior sample and adaptation efficiency. We showed that reliable and robust task representations are crucial for meta-RL, especially on complex task distributions. Our approach represents tasks as a mixture of Gaussians in the latent space to accommodate parametric and non-parametric task variations. Together with contrastive learning, we realize effective task identification, which alleviates the difficulty of downstream policy learning. Furthermore, MoSS can rapidly adapt to new tasks in non-stationary environments via GRU-based online task inference and adaptation. We train the network completely off-policy to ensure high sample efficiency. MoSS outperforms existing methods in terms of asymptotic performance, sample and adaptation efficiency, and generalization robustness on various robot control and manipulation tasks with the single episode result.

Acknowledgements

This work was supported by the Shenzhen Basic Research Grants (JCYJ20180508152434975, JCYJ20180507182508857), the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539 (Human Brain Prohect SGA3).

References

Bellman, R. 1966. Dynamic programming. *Science*, 153(3731): 34–37.

Bing, Z.; Brucker, M.; Morin, F. O.; Li, R.; Su, X.; Huang, K.; and Knoll, A. 2021a. Complex Robotic Manipulation via Graph-Based Hindsight Goal Generation. *IEEE Transactions on Neural Networks and Learning Systems*, 1–14.

Bing, Z.; Knak, L.; Robin, F. O.; Huang, K.; and Knoll, A. 2021b. Meta-Reinforcement Learning in Broad and Non-Parametric Environments. *arXiv preprint arXiv:2108.03718*.

Bing, Z.; Koch, A.; Yao, X.; Morin, F. O.; Huang, K.; and Knoll, A. 2022a. Meta-Reinforcement Learning via Language Instructions. *arXiv preprint arXiv:2209.04924*.

Bing, Z.; Lerch, D.; Huang, K.; and Knoll, A. 2022b. Meta-Reinforcement Learning in Non-Stationary and Dynamic Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–17.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28.

Dorfman, R.; Shenfeld, I.; and Tamar, A. 2021. Offline Meta Reinforcement Learning–Identifiability Challenges and Effective Data Collection Strategies. *Advances in Neural Information Processing Systems*, 34: 4607–4618.

Duan, Y.; Schulman, J.; Chen, X.; Bartlett, P. L.; Sutskever, I.; and Abbeel, P. 2016. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*.

Duff, M. O. 2002. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst.

Fakoor, R.; Chaudhari, P.; Soatto, S.; and Smola, A. J. 2019. Meta-q-learning. *arXiv preprint arXiv:1910.00125*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *34th International Conference on Machine Learning, ICML 2017*, 3: 1856–1868.

Ghavamzadeh, M.; Mannor, S.; Pineau, J.; Tamar, A.; et al. 2015. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6): 359–483.

Guo, Y.; Wu, Q.; and Lee, H. 2022. Learning Action Translator for Meta Reinforcement Learning on Sparse-Reward Tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 6792–6800.

Gupta, A.; Mendonca, R.; Liu, Y.; Abbeel, P.; and Levine, S. 2018. Meta-reinforcement learning of structured exploration strategies. *Advances in neural information processing systems*, 31.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. PMLR.

Humplik, J.; Galashov, A.; Hasenclever, L.; Ortega, P. A.; Teh, Y. W.; and Heess, N. 2019. Meta reinforcement learning as task inference. *arXiv preprint arXiv:1905.06424*.

Jiang, Z.; Zheng, Y.; Tan, H.; Tang, B.; and Zhou, H. 2016. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*.

Lee, S.; and Chung, S.-Y. 2021. Improving Generalization in Meta-RL with Imaginary Tasks from Latent Dynamics Mixture. *arXiv preprint arXiv:2105.13524*.

Li, L.; Yang, R.; and Luo, D. 2020. Focal: Efficient fully-offline meta-reinforcement learning via distance metric learning and behavior regularization. *arXiv preprint arXiv:2010.01112*.

Mendonca, R.; Geng, X.; Finn, C.; and Levine, S. 2020. Meta-reinforcement learning robust to distributional shift via model identification and experience relabeling. *arXiv preprint arXiv:2006.07178*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M. A.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518: 529–533.

Nagabandi, A.; Clavera, I.; Liu, S.; Fearing, R. S.; Abbeel, P.; Levine, S.; and Finn, C. 2018. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*.

Nichol, A.; and Schulman, J. 2018. Reptile: a scalable meta-learning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3): 4.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Rakelly, K.; Zhou, A.; Finn, C.; Levine, S.; and Quillen, D. 2019. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, 5331–5340. PMLR.

Rao, D.; Visin, F.; Rusu, A. A.; Teh, Y. W.; Pascanu, R.; and Hadsell, R. 2019. Continual unsupervised representation learning. *arXiv preprint arXiv:1910.14481*.

Rothfuss, J.; Asfour, T.; Lee, D.; Clavera, I.; and Abbeel, P. 2019. PrOMP: Proximal meta-policy search. *7th International Conference on Learning Representations, ICLR 2019*, 1–25.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587): 484–489.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; Chen, Y.; Lillicrap, T.; Hui, F.; Sifre, L.; van den Driessche, G.; Graepel, T.; and Hassabis, D. 2017. Mastering the game of Go without human knowledge. *Nature*, 550: 354–.

Stadie, B. C.; Yang, G.; Houthooft, R.; Chen, X.; Duan, Y.; Wu, Y.; Abbeel, P.; and Sutskever, I. 2018. Some considerations on learning to explore via meta-reinforcement learning. *arXiv preprint arXiv:1803.01118*.

Tian, Y.; Sun, C.; Poole, B.; Krishnan, D.; Schmid, C.; and Isola, P. 2020. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems*, 33: 6827–6839.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 5026–5033. IEEE.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Wang, B.; Xu, S.; Keutzer, K.; Gao, Y.; and Wu, B. 2021. Improving Context-Based Meta-Reinforcement Learning with Self-Supervised Trajectory Contrastive Learning. *arXiv preprint arXiv:2103.06386*.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3733–3742.

Yao, X.; Bing, Z.; Zhuang, G.; Chen, K.; Zhou, H.; Huang, K.; and Knoll, A. 2022. Learning from Symmetry: Meta-Reinforcement Learning with Symmetric Data and Language Instructions. *arXiv preprint arXiv:2209.10656*.

Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; and Levine, S. 2020. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, 1094–1100. PMLR.

Zhao, T. Z.; Nagabandi, A.; Rakelly, K.; Finn, C.; and Levine, S. 2020. MELD: Meta-Reinforcement Learning from Images via Latent State Models. *arXiv preprint arXiv:2010.13957*.

Zintgraf, L.; Shiarlis, K.; Igl, M.; Schulze, S.; Gal, Y.; Hofmann, K.; and Whiteson, S. 2020. VariBAD: a very good method for Bayes-adaptive deep RL via meta-learning. *Proceedings of ICLR 2020*.

Zintgraf, L. M.; Feng, L.; Lu, C.; Igl, M.; Hartikainen, K.; Hofmann, K.; and Whiteson, S. 2021. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, 12991–13001. PMLR.