OCEAN: Online Task Inference for Compositional Tasks with Context Adaptation

Hongyu Ren
Stanford University

Yuke Zhu
The University of Texas at Austin, Nvidia

Jure Leskovec
Stanford University

Anima Anandkumar
California Institute of Technology, Nvidia

Animesh Garg
University of Toronto, Vector Institute, Nvidia

Abstract

Real-world tasks often exhibit a compositional structure that contains a sequence of simpler sub-tasks. For instance, opening a door requires reaching, grasping, rotating, and pulling the door knob. Such compositional tasks require an agent to reason about the sub-task *at hand* while orchestrating *global* behavior accordingly. This can be cast as an *online task inference* problem, where the current task identity, represented by a context variable, is estimated from the agent's past experiences with probabilistic inference. Previous approaches have employed simple latent distributions, e.g., Gaussian, to model a single context for the entire task. However, this formulation lacks the expressiveness to capture the composition and transition of the sub-tasks. We propose a variational inference framework OCEAN to perform online task inference for compositional tasks. OCEAN models global and local context variables in a joint latent space, where the global variables represent a mixture of sub-tasks required for the task, while the local variables capture the transitions between the sub-tasks. Our framework supports flexible latent distributions based on prior knowledge of the task structure and can be trained in an unsupervised manner. Experimental results show that OCEAN provides more effective task inference with sequential context adaptation and thus leads to a performance boost on complex, multi-stage tasks.

1 INTRODUCTION

Meta-reinforcement learning (meta-RL) algorithms aim to train a versatile agent that quickly adapts to unseen new tasks after trained on a domain of related tasks [1, 2, 3, 4, 5]. Meta-RL has shown promise in simple domains, where it requires much less data than training an RL agent from scratch for each task of interest. A key assumption that makes meta-RL desirable is that there exists rich latent structure and relationships among the tasks, implying that solving one task may reuse skills from others. For instance, the agent learning to walk backward may benefit from mastering how to run forward. However, leveraging this structure in task inference and adaptation [6] makes meta-RL much more challenging than RL, albeit offers an opportunity for improvement in sample efficiency of multi-task learning.

Recent context-based meta-RL models [5, 7] capture the current task with additional latent task variables. Specifically, the agent first stochastically explores the environment and keeps record of (potentially unordered) transition tuples, which are referred to as *contexts*. A trained context encoder uses the contexts to infer the latent task variables, which serve as the agent's belief over the current task. Given the latent task variable, the agent makes sequential decisions for the current task. These context-based meta-RL models provide an elegant framework that disentangles probabilistic task inference from decision making, and can be integrated with multiple off-policy learning algorithms [8, 9, 10], demonstrating improvement in both sample efficiency and asymptotic performance.

However, prior art fails in case of complex task structures in most real-world scenarios. Often real tasks require finishing a sequence of sub-tasks and the current sub-task may only be revealed to the agent after the previous sub-task is finished. For example, door opening requires one to reach, grasp, rotate and pull or push the door knob. Not only do these sub-tasks need to be carried out in an appropriate order, but the each subsequent sub-task is also conditioned on the success previous one. This compositional structure is of important value and needs to be accounted for during task infer-

Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI), PMLR volume 124, 2020.

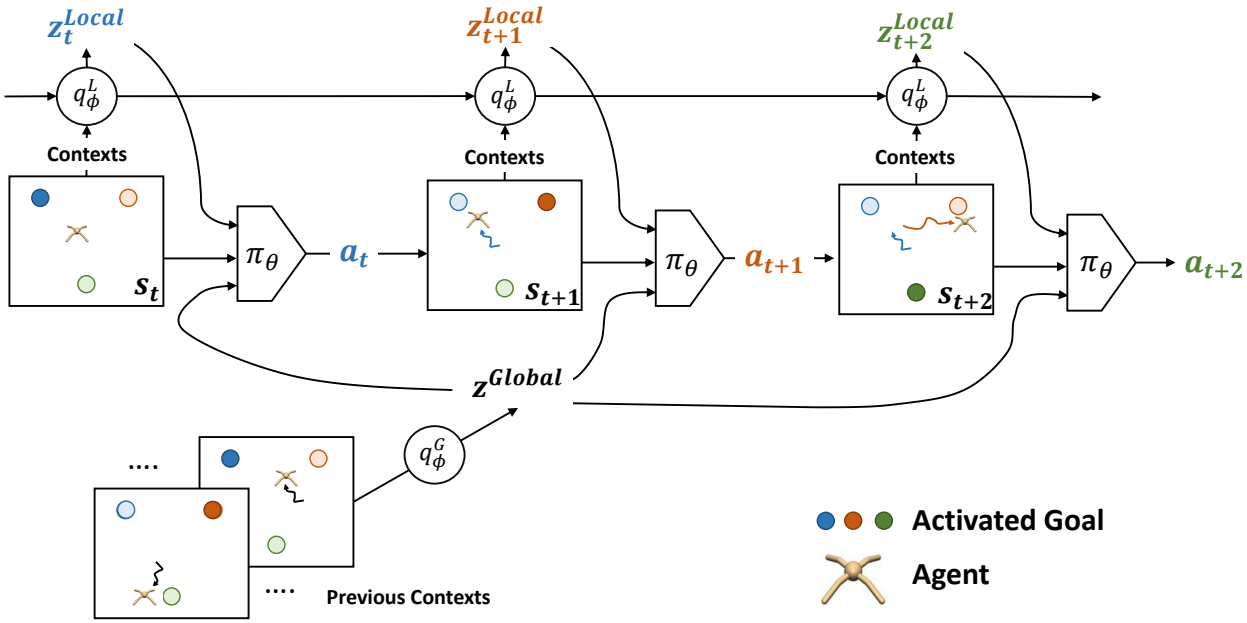

Figure 1: Our framework OCEAN performs task inference based on the contexts. OCEAN consists of two modules: global context encoder q_ϕ^G and local context encoder q_ϕ^L. q_ϕ^G leverages contexts from previous episodes to capture the global structure of the task, for example, goal locations, rewards. q_ϕ^L (with a recurrent architecture) uses the contexts from previous steps within an episode to do online sub-task inference, it aims to capture the current sub-task, in this case, currently which goal has been activated. The agent takes actions $\boldsymbol{a}_t$ based on the current state $\boldsymbol{s}_t$, the current local latent context variable $\boldsymbol{z}_t^{Local}$ and the global context variable $\boldsymbol{z}^{Global}$.

ence. However, current context-based meta-RL models do not leverage this sophisticated multi-stage structure and the dependencies between tasks. The reason is modelling the whole episode of a task with a fixed isotropic Gaussian context variables. As shown in Fig. 1, one task may consist of multiple stages where each stage may have a different goal, a single static latent context variable fails to model the sequential structure. Although posterior estimation may offer slight improvement through recurrence as more contexts accumulate, the following issues remain: (1) latent context variables that are held constant across an episode are not suitable to model the sequence of sub-tasks executed in an order; (2) isotropic Gaussian random variables are not flexible enough to model mixtures of tasks.

This paper proposes OCEAN, an Online ContExt AdaptatioN framework that addresses the aforementioned issues [1]. Our framework is composed of two parts that account for the local sub-task update and the global sub-task mixture respectively. We introduce a local context encoder with a recurrent architecture, which performs sub-task inference on-the-fly based on the history contexts within an episode as shown in top of Fig. 1. Vitally, the adaptive local context variables instruct the agent to make smooth transitions from one sub-task to another within a single episode. In order to model different sub-task structure, we also introduce a global context encoder facilitates a flexible latent space with rich prior distributions beyond Gaussian, categorical, Dirichlet and logistic normal distributions based on the domain knowledge of the task structure as shown in bottom of Fig. 1. With a joint latent space comprised of the local and global context variables, OCEAN gives rise to *a principled task inference framework for context-based meta-reinforcement learning* that is able to model complex, real-world tasks. We observe that previous context-based meta-RL models [5, 7] can be viewed as special cases of our general framework where global latent space is Gaussian and the local context variables are frozen.

We integrate OCEAN with soft actor-critic [8], an off-policy RL algorithm with high sample efficiency. The context encoders and the RL agent can be trained jointly to optimize for maximizing expected return across all the tasks in meta-training. Note that our framework does not require any labels or supervised signals of either sub-task mixture or sub-task transition when training both the global and local encoder. In meta-test phase, we sample the global context variables for a given new task,

[1]The implementation can be found in https://github.com/pairlab/ocean.

and stochastically explore the task with both the global context variables and the local context variables, which OCEAN estimate and update in an online fashion. Task inference becomes more and more accurate with more exploration, thus adapting to the new task in meta-test. We show through a 2D locomotion experiment that designing an appropriate global latent space matters for complex task structures. We further demonstrate that online context adaptation can greatly improve the performance in several continuous control tasks with sequential sub-task structure.

2 PRELIMINARIES

2.1 META-REINFORCEMENT LEARNING

We are interested in meta-reinforcement learning (meta-RL), where we are given a distribution of tasks $p(\mathcal{T})$. Each sample from $p(\mathcal{T})$ is a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma \rangle$, representing the state space, action space, transition probability distribution, reward function, the distribution of initial state and discount factor respectively. We assume that all tasks from $p(\mathcal{T})$ share the same known state and action space, but may differ in transition probability, reward function and initial state distribution, which are unknown but we can sample from them. We refer to the contexts $\boldsymbol{C}$ of a given task $\mathcal{T}$ as a collection of transition tuples sampled from $\mathcal{T}$. Each transition tuple $(\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}'_i)$ consists of the state $\boldsymbol{s}$, action $\boldsymbol{a}$, reward r, next state $\boldsymbol{s}'$ at a certain timestep. Meta-RL models aim to train a flexible agent $\pi_\theta(\boldsymbol{a}|\boldsymbol{s})$ (parameterized by θ) on a given set of training tasks sampled from $p(\mathcal{T})$ and the goal is that the agent can be adapted quickly to a

2.2 CONTEXT-BASED TASK INFERENCE

In order to make fast adaptation towards a given task, the key is to perform accurate task inference, which can be explicitly captured by latent task variables $p(\boldsymbol{z}|\mathcal{T})$. The RL agent $\pi_\theta(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{z})$ then takes actions based on the current observation (or state) $\boldsymbol{s}$ and the latent task variables $\boldsymbol{z} \sim p(\boldsymbol{z}|\mathcal{T})$ in order to make decisions in task $\mathcal{T}$. The latent task variables capture the uncertainty over the task and are crucial to achieve quick adaptation and high performance in meta-RL. However, the true posterior is unknown since we have no knowledge of the transition probability, reward function and initial state distribution of the task $\mathcal{T}$. Context-based meta-RL models approximate $p(\boldsymbol{z}|\mathcal{T})$ by first collecting a set of contexts $\boldsymbol{C}$ from the task $\mathcal{T}$ and calculating the posterior of latent *context* variable $p(\boldsymbol{z}|\boldsymbol{C})$, where essentially the contexts can be seen as the representative samples from the task $\mathcal{T}$. Although $p(\boldsymbol{z}|\boldsymbol{C})$ is still intractable, we can train an additional context encoder $q_\phi(\boldsymbol{z}|\boldsymbol{C})$ parameterized by ϕ to estimate the true posterior based on amortized variational inference. The corresponding evidence lower bound can be derived as

$$\mathbb{E}_{\mathcal{T}}[\mathbb{E}_{\boldsymbol{z}\sim q_\phi}[R(\mathcal{T}, \boldsymbol{z}) - \beta D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{C})||p(\boldsymbol{z}))]], \quad (1)$$

where $p(\boldsymbol{z})$ is the prior distribution and captures our prior knowledge of the task distribution, $R(\mathcal{T}, \boldsymbol{z})$ is implemented to recover the state-action value functions [5] or the transition functions [7], β is a trade-off hyperparameter that regularizes the capacity. One benefit of context-based meta-RL models is that it disentangles task inference from decision making and hence large amount of off-policy data can be used for policy update, which greatly improves the sample efficiency [5].

2.3 SOFT ACTOR-CRITIC

In this paper, we use the soft actor-critic algorithm (SAC) [8, 9] to perform policy update in order to achieve better sample efficiency. Based on the maximum entropy RL framework [11], SAC is an off-policy actor-critic algorithm that aims to maximize both the expected reward and casual entropy, explicitly regularizing the policy. Specifically, SAC aims to optimize the following objectives for the agent π_θ, the Q-function approximator Q_θ and the value function approximator V_θ using samples stored in a replay buffer $\mathcal{B}$:

$$\mathcal{L}_{critic} = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}',r)\sim\mathcal{B}}[Q_\theta(\boldsymbol{s}, \boldsymbol{a}) - (r + V_\theta(\boldsymbol{s}'))]^2, \quad (2)$$

$$\mathcal{L}_{actor} = \mathbb{E}_{\boldsymbol{s}\sim\mathcal{B}}\left[D_{\mathrm{KL}}\left(\pi_\theta(\cdot|\boldsymbol{s}) \middle\| \frac{\exp\left(Q_\theta(\boldsymbol{s}, \cdot)\right)}{\mathcal{Z}_\theta(\boldsymbol{s})}\right)\right]. \quad (3)$$

3 OCEAN: TASK INFERENCE WITH LATENT CONTEXT VARIABLES

Latent context variables are crucial in context-based meta RL models since they represent the belief over the current tasks. Thus, these context variables should be carefully tailored during task inference to take into account the complex compositional structure of real-world tasks in order to achieve the most accurate context variables. In Sec. 3.1, we introduce *local* latent context variables and perform online context adaptation and sub-task inference based on the contexts at past steps within an episode. We implement a local context encoder with a recurrent neural network for sequential probabilistic inference. In Sec. 3.2, we use *global* context variables to capture the global information of a task and further propose OCEAN, an Online ContExt AdaptatioN framework for meta-RL with joint latent space that consists of global

and local context variables. In Sec. 3.3, we demonstrate that our framework allows flexible latent distributions to suit different prior knowledge of the task/sub-task structure. Finally, in Sec. 3.4, we integrate our task inference framework with SAC for efficient policy update, and perform end-to-end training for the global and local context encoders as well as the agent.

3.1 ONLINE CONTEXT ADAPTATION

Since real-world tasks often require an agent to finish a sequence of sub-tasks, a single static latent context variable is not sufficient to inform the agent of the transitions in the sequence. To capture the ongoing sub-task and the transitions between them, we use local latent context variables $\boldsymbol{z}^{Local}$ that are estimated online throughout the episode. To perform online probabilistic inference, we train an additional local context encoder q_ϕ^L parameterized by π that takes as input the contexts at previous steps and updates the posterior of the local context variable for future steps. Since the local latent contexts at different steps are not independent from each other, we design a recurrent architecture for q_ϕ^L. To better model the variability of dependencies between the local context variables across different timesteps, we adopt variational recurrent neural network [12], where the hidden state is conditioned on stochastic samples from the previous posterior. Specifically, the local context encoder q_ϕ^L consists of three modules: q_ϕ^{enc}, q_ϕ^{tran} and q_ϕ^{prior}, which represent the inference function, the transition function, the conditional prior respectively. Given the context $\boldsymbol{c}_t$ at timestep t, the latent context variable $\boldsymbol{z}_{t+1}^{Local}$ at timestep $t+1$ is sampled from the posterior calculated as follows:

$$\boldsymbol{z}_{t+1}^{Local} \sim q_\phi^{\text{enc}}(\boldsymbol{z}|\boldsymbol{c}_t, \boldsymbol{h}_t), \tag{4}$$

where $\boldsymbol{h}_t$ denotes the hidden state at timestep t. The hidden state is updated according to the following recurrence:

$$\boldsymbol{h}_t = q_\phi^{\text{tran}}(\boldsymbol{c}_{t-1}, \boldsymbol{z}_t^{Local}, \boldsymbol{h}_{t-1}). \tag{5}$$

Then the agent takes actions by sampling from $\pi_\theta(\boldsymbol{a}|\boldsymbol{s}_t, \boldsymbol{z}_t^{Local})$, which is conditioned on the observation as well as the updated local context variable at timestep t. Note that we use zero-valued vectors as initialization for $\boldsymbol{h}_0$, and we can directly sample $\boldsymbol{z}_0^{Local}$ from a predefined uninformative prior, such as isotropic Gaussian or uniform distribution.

Since we aim to capture the dependency between the current and past timesteps, the corresponding prior distribution of $\boldsymbol{z}_t^{Local}$ is conditioned on the previous hidden state $p(\boldsymbol{z}_t^{Local}) = q_\phi^{\text{prior}}(\boldsymbol{h}_{t-1})$ rather than a given fixed uninformative prior, which neglects the temporal structure of posterior at different steps.

Finally the loss of the local context encoder q_ϕ^L is defined by replacing the KL loss term in Eq. 1 as follows:

$$D_{\text{KL}}^{Local} = \sum_t D_{\text{KL}}(q_\phi^{\text{enc}}(\boldsymbol{z}|\boldsymbol{c}_t, \boldsymbol{h}_t)||q_\phi^{\text{prior}}(\boldsymbol{h}_t)), \tag{6}$$

which takes the sum of the KL loss at all timesteps. and is optimized in meta-training. Note that OCEAN does not assume prior knowledge of either the labels of the specific transition steps between each sub-task or the labels of the sub-tasks. Using the variational inference framework, our model is able to discover sub-task structure in an unsupervised manner. At meta-test time, we fix the local context encoder q_ϕ^L and the agent first samples from the prior distribution and then can execute the policy, collect the context and infer the posterior of the local context variable in a recurrent manner. Given a previously unseen test task, the agent explores at the first few steps and then with more steps collected, quickly converges to the optimal policy as the tasks inference becomes more and more accurate, thus efficiently adapted to the test task at hand.

Computation Efficiency. Note that the local context encoder requires stochastic sampling at each step, which is a sequential process. Given a limited computation budget, it can be costly. Although real-world tasks often contain sequence of sub-tasks, we do not need to update the posterior at each step. We assume that the sub-task in timestep t is very similar to the sub-task in timestep $t+1$, so one local context variable that represents the current sub-task at timestep t is also likely to be accurate enough for decision-making at timestep $t+1$. To reduce the computation overhead, one strategy is to perform posterior estimation on a subset of the timesteps $\{tr, 2tr, \dots\}$ with certain temporal resolution tr. Specifically for Eq. 4, $\boldsymbol{z}_{tr}^{Local}$ is sampled from $q_\phi^{\text{enc}}(\boldsymbol{z}|\boldsymbol{c}_{0:tr-1}, \boldsymbol{h}_0)$, where $\boldsymbol{c}_{0:tr-1}$ is the concatenated contexts from timestep 0 to $tr-1$. Then the cost of each posterior sampling can be amortized over tr steps of decision making.

3.2 JOINT LATENT SPACE FOR TASK INFERENCE

As introduced in Sec. 3.1, given a task, the agent first samples from uninformative prior and explores the environment at the beginning of the episode in order to infer the local sub-task with q_ϕ^L, however, these exploration steps may be sub-optimal since the agent has no knowledge of the task and explores with context variables drawn from uninformative prior. One limitation of the local context variables is that they fail to make use of past contexts from different trajectories. In this section, we introduce global context variables which leverages the contexts from past trajectories to give the agent a global overview of the task. If we have access to a pool of past

Algorithm 1: Meta-Training in OCEAN

Input: $\mathcal{T}_{1...T}$: Training tasks sampled from $p(\mathcal{T})$.
Initialize: $\mathcal{B}^i$: Replay buffers for each task;
$\theta_\pi, \theta_Q, \theta_V, \phi$: parameters in OCEAN; $\alpha_1, \alpha_2, \alpha_3$: learning rate;
K: number of trajectories collected in each iteration;
B: number of trajectories sampled in each iteration.

While not done **do**
  For $i = 1, \dots, T$ **do**
    TrajCollect($\mathcal{T}_i, \mathcal{B}^i, K$)
    Contexts $\boldsymbol{C} \sim \mathcal{S}_c(\mathcal{B}^i)$ and trajectories $\boldsymbol{B} \sim \mathcal{B}^i$
    $\boldsymbol{z}^{Global} \sim q_\phi^G(\boldsymbol{z}|\boldsymbol{C})$
    Initialize $\boldsymbol{z}^{Local} = \{\}$
    $\mathcal{L}_{KL}^i = \beta D_{\text{KL}}(q(\boldsymbol{z}|\boldsymbol{C})||p(\boldsymbol{z}))$
    For $b = 1, \dots, B$ **do**
      For $t = 1, 2, \dots$ **do**
        $\boldsymbol{z}_t^{Local} \sim q_\phi^L(\boldsymbol{z}|\boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1})$ and add to $\boldsymbol{z}^{Local}$
        Update $\boldsymbol{c}_t = (\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}'_t, r_t)$
        $\mathcal{L}_{KL}^i \mathrel{+}= \beta D_{\text{KL}}(q_\phi^L(\boldsymbol{z}|\boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1})||p(\boldsymbol{z}_t^{Local}))$
    $\mathcal{L}_{actor}^i = \mathcal{L}_{actor}(\boldsymbol{B}, \boldsymbol{z}^{Global}, \boldsymbol{z}^{Local})$
    $\mathcal{L}_{critic}^i = \mathcal{L}_{critic}(\boldsymbol{B}, \boldsymbol{z}^{Global}, \boldsymbol{z}^{Local})$
  $\phi \leftarrow \phi - \alpha_1 \nabla_\phi \sum_i \left(\mathcal{L}_{critic}^i + \mathcal{L}_{KL}^i\right)$
  $\theta_\pi \leftarrow \theta_\pi - \alpha_2 \nabla_{\theta_\pi} \sum_i \mathcal{L}_{actor}^i$
  $\theta_Q \leftarrow \theta_Q - \alpha_3 \nabla_{\theta_Q} \sum_i \mathcal{L}_{critic}^i$
  $\theta_V \leftarrow \theta_V - \alpha_3 \nabla_{\theta_V} \sum_i \mathcal{L}_{critic}^i$

Algorithm 2: TrajCollect

Input: $\mathcal{T}$: Task; $\mathcal{B}$: Replay Buffer; K: Number of trajectories.
Initialize: $\boldsymbol{C}$: Context set.

For $k = 1, \dots, K$ **do**
  $\boldsymbol{z}^{Global} \sim q_\phi^G(\boldsymbol{z}|\boldsymbol{C})$
  For $t = 1, 2, \dots$ **do**
    $\boldsymbol{z}_t^{Local} \sim q_\phi^L(\boldsymbol{z}|\boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1})$
    Roll out policy $\pi_\theta(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{z}^{Global}, \boldsymbol{z}_t^{Local})$
    Update $\boldsymbol{c}_t = (\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}'_t, r_t)$
    Accumulate context $\boldsymbol{C} = \boldsymbol{C} \cup \{\boldsymbol{c}_t\}$
Add $\boldsymbol{C}$ to $\mathcal{B}$

contexts $\boldsymbol{C}$ of size n collected from the same task (but not necessarily from the same episode), we leverage this task information to infer our belief over the global task. Following PEARL [5], we introduce a global context encoder $q_\phi^G(\boldsymbol{z}|\boldsymbol{c}_i)$ that infers the posterior based on each single past context $\boldsymbol{c}_i \in \boldsymbol{C}$. The posterior of the global context is calculated as a function of each independent posterior $q_\phi^G(\boldsymbol{z}|\boldsymbol{C}) = f(q_\phi^G(\boldsymbol{z}|\boldsymbol{c}_1), \dots, q_\phi^G(\boldsymbol{z}|\boldsymbol{c}_n))$ (detailed in Sec. 3.3). With larger set of the past contexts $\boldsymbol{C}$, we can achieve more accurate global task estimation.

Combined with the local context variables, we introduce OCEAN with a joint latent space for online task inference in meta-RL. The joint latent space consists of local and global context variables where the local context variables reason about the sub-task and are updated online, while the global context variables captures the big picture of the task. The global and local context encoder can be trained jointly with the agent with detailed description in Sec. 3.4. For simplicity, we assume the local and global context variables are independent, thus the KL term in the objective in Eq. 1 can be decomposed into sum of two separate terms:

$$D_{\text{KL}}(q_\phi^G(\boldsymbol{z}|\boldsymbol{C})||p(\boldsymbol{z})) + D_{\text{KL}}^{Local}, \tag{7}$$

where $p(\boldsymbol{z})$ is the prior distribution for global context variables and D_{KL}^{Local} is defined in Eq. 6.

3.3 FLEXIBLE PARAMETERIZATION OF THE LATENT SPACE

A suitable latent space is critical in accurate task inference. Based on the prior knowledge of the task, OCEAN supports flexible parameterization of the latent space for both the global and the local context variables. Besides the Gaussian prior used in PEARL [5], we also design latent space with categorical distribution to model tasks controlled by discrete factors, Dirichlet distribution and logistic normal (logit-normal) distribution to model multi-modal or proportional tasks. While it is straight forward for local context encoder to infer the posterior for the next step, estimating the global posterior is non-trivial since we need to consider all the past contexts from the replay buffer and design a suitable function f as introduced in Sec. 3.2.

When estimating the posterior of global latent context variables, we model each context as an independent factor in order to capture the minimal sufficient information [5]. Specifically, assume we have n contexts, we first estimate the posterior $q_\phi^G(\boldsymbol{z}|\boldsymbol{c}_i)$ for each single context $\boldsymbol{c}_i$ using the global context encoder q_ϕ^G. Then we model the global latent context variable $q_\phi^G(\boldsymbol{z}|\boldsymbol{C})$ as the weighted product of the independent posteriors: $q_\phi^G(\boldsymbol{z}|\boldsymbol{C}) \propto \prod_{i=1}^n q_\phi^G(\boldsymbol{z}|\boldsymbol{c}_i)^{\frac{1}{n}}$. Our framework allows the global latent space with Gaussian distribution, categorical distribution, Dirichlet distribution and logit-normal distribution. For all four distributions, the operation of weighted product of the probability density functions (PDFs) is closed, and thus the we can easily calculate the global posterior as shown below.

Gaussian / Logit-normal Distribution. Assume the posterior of context $\boldsymbol{c}_i$ has parameters μ_i and σ_i^2, then the global posterior has parameters $\mu = \frac{1}{\sum_i \frac{1}{\sigma_i^2}} \sum_i \frac{\mu_i^2}{\sigma_i^2}$ and $\sigma^2 = \frac{n}{\sum_i \frac{1}{\sigma_i^2}}$.

Categorical Distribution. Assume the posterior of context $\boldsymbol{c}_i$ has parameters $(p_{i1}, \dots, p_{iK})$, the global posterior has parameters $(\frac{\sqrt[n]{\prod_i p_{i1}}}{Z}, \dots, \frac{\sqrt[n]{\prod_i p_{iK}}}{Z})$, where $Z = \sum_j \sqrt[n]{\prod_i p_{ij}}$.

Dirichlet Distribution. Assume the posterior of context $\boldsymbol{c}_i$ has parameters $(\alpha_{i1}, \dots, \alpha_{iK})$, the global posterior has parameters $(\frac{\sum_i \alpha_{i1}}{n}, \dots, \frac{\sum_i \alpha_{iK}}{n})$.

After we achieve the posterior of global and local contexts, we can use the reparameterization trick to sample from these distributions in order to optimize the Eq. 7. Specifically, we use several reparameterization tricks for Gaussian [13], categorical distribution [14, 15] and Dirichlet distribution [16]. Note that the latent space can be composite and may consist of random variables from different distributions mentioned above based on the prior knowledge of the tasks.

3.4 TRAINING WITH OFF-POLICY UPDATE

Sample efficiency is one of the most critical issues in both RL and meta-RL. Following PEARL [5], since our framework disentangles task inference (both global and local) from decision making, the agent in our framework can safely be trained using off-policy RL algorithms. As discussed in Sec. 2.3, we use soft actor-critic algorithm [8] and further extend the definition of π_θ, Q_θ and V_θ to further take the (global and local) latent context variables as input. Following PEARL, we implement the R term in Eq. 1 to recover the value function and additionally design a sampler $\mathcal{S}_c$ for context sampling from the replay buffer $\mathcal{B}$ so that the contexts do not diverge too much from that achieved by the current parameters. We list the meta-training algorithm of OCEAN in Algorithm 1. In order to integrate our joint latent space with SAC, during meta-training time, we first collect trajectories for each task in each iteration as defined in Algorithm 2. Then we sample contexts and batches of trajectories from the replay buffer. We use q_ϕ^G to calculate the global latent context variable and then run q_ϕ^L to achieve the local context variable for each step in the trajectory batch before we optimize the loss. During meta-test, we directly roll out our policy while using q_ϕ^L to update the local context variable at each step, which is very similar to trajectory collection in Algorithm 2.

4 RELATED WORK

Our work is closely related to meta-learning or learning to learn in reinforcement learning settings.

Meta-Learning. Meta-learning tries to address the problem of learning to learn [17]. The goal is to leverage the existing knowledge and data to grant the model with more inductive bias so that when facing a set of new tasks, the model can quickly adapt to it [18, 19].

Gradient-based Meta-RL. In the context of reinforcement learning, one line of meta-RL models uses gradient-based updates for few-shot adaptation [1, 2, 3, 20]. The objective for meta-training is to find a set of parameters such that they are good initialization for a wide range of tasks. At meta-test time, a few gradient updates can potentially result in high performance, achieving adaptation towards unseen test tasks. The gradient-based meta-RL models do not explicitly infer tasks and hence cannot model the structure of sub-tasks. Another limitation is that this line of work is mostly optimized using on-policy data, thus making the learning process sample inefficient.

Meta-RL with Memory. Another line of meta-RL models uses recurrent network structure for the agent [4, 21]. The goal herein is to model previous steps implicitly with the hidden states in the recurrent network. While it is closely related to our work, however, it does not reason over the uncertainty about the task structure and nor does it perform task inference explicitly. This line of work also demonstrates limited sample efficiency due to the on-policy RL update.

Context Conditioned Meta-RL. The most related line of work is context based meta-RL models [5, 7, 22], where the goal is to explicitly perform task inference using contexts. These models represent tasks with latent context variables and the objective performs context inference as a separate module on top of Q-learning [10]. The task identification is formalized as a variational inference problem, and they naturally disentangle task inference from decision making by using context conditioned value functions or agents. Integrated with off-policy algorithms, these models achieve significant higher sample efficiency and asymptotic performance in meta-RL. Given a new task, these models make quick adaptation through exploration with posterior sampling [23]. However, in this line of work the latent context variables are fixed across episode and does not allow for compositional tasks with multi-step sub-tasks or subgoals.

Hierarchical RL. Our work is also related to hierarchical RL [24]. Hierarchical RL models generally learn policies with hierarchical structure, which often results in a high-level and low-level policy. The high-level policy reasons over the structure of the task and either selects a skill to execute [25], or assigns a goal for the low-level policy [26, 27]. Our work focuses on meta-RL and models the task distribution with a global and local context variables rather than directly learn a hierarchical policy [28].

OCEAN is a unified framework for meta-learning in sequential decision making. Our model combines the advantages of both world of recurrent memory and latent contexts. We leverage the contexts to infer the task on a global scale while design a local context encoder with recurrent structure to perform online context adaptation and infer the local sub-task. In addition to this we also enable a richer context modeling with a flexible parame-

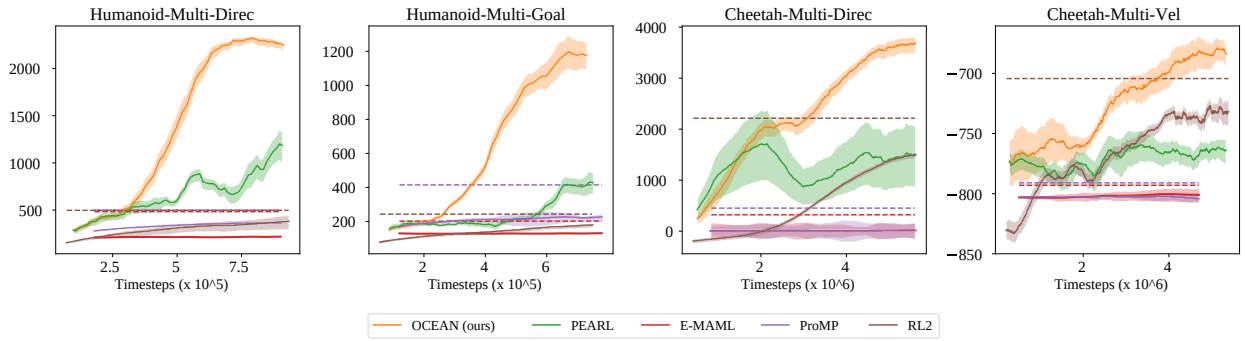

Figure 2: Our framework OCEAN v.s. several state-of-the-art baselines in multi-stage tasks. OCEAN achieves significantly better sample efficiency and performance since we are able to perform accurate online task inference that fits especially in multi-stage setting where each task requires finishing a sequence of sub-tasks.

terization of the latent space. These empower the model to learn in a complex setting with structured, compositional tasks.

5 EXPERIMENTS

In this section we aim to investigate the following questions: (1) Can OCEAN succeed in several multi-stage meta-RL tasks with the online context adaptation? (2) What's the impact of each module in our framework? (3) Do we need to update the posterior within an episode or can we only update the context variables by directly resampling from the posterior at each step? (4) Do different choices of global latent space matter given prior knowledge of the task distributions?

5.1 EXPERIMENTAL SETUP

We evaluate the performance of our framework OCEAN on a 2D point-robot navigation task and five simulated environments in Mujoco [29] with continuous control, four of which have a compositional sub-task structure. We provide the details regarding the tasks as well as the baselines below.

Point robot navigation. The agent in 2D plane aims to navigate to different goals on the edge of a half-circle.

Cheetah-Fwd-Back. 2D cheetah agent aims to run forward or backward, this environment is not multi-stage and only has 2 tasks.

Cheetah-Multi-Vel. 2D cheetah agent aims to run at goal velocity. One task may contain multiple goal velocities and the steps that the goal velocity shifts are randomly sampled.

Cheetah-Multi-Direc / Humanoid-Multi-Direc. 2D cheetah agent / 3D humanoid agent aims to run in goal direction. One task may contain multiple goal directions and the steps that the goal direction shifts are randomly sampled.

Humanoid-Multi-Goal. 3D humanoid agent aims to run to several goals. One task may contain multiple goals and the steps that the goal shifts are randomly sampled.

The environments in Mujoco are adapted from previous meta-RL works [1, 5], the main feature is that we added multi-stage sub-tasks in each environment. For all the environments, we adhere to the protocol that we sample a fixed number of training and test tasks before training, after the models are trained on the fixed set of training tasks, we evaluate whether the model is able to quickly adapt to the test tasks.

Baselines. We compare OCEAN with several state-of-the-art meta-RL methods, including gradient-based meta-RL models: E-MAML [2], ProMP [3]; models with recurrent policy: RL2 [4]; and context-based meta-RL model: PEARL [5] [2]. In all the experiments, we have the same number of latent variables as PEARL. We aim to show that our online task inference scheme can make the best use of the latent variables than using them all to model global contexts.

5.2 MAIN RESULTS

We first conduct sanity check on Cheetah-Fwd-Back, where each task is single-stage. We directly compare our model with PEARL. As shown in Fig. 3, our model could achieve comparable results with PEARL although the context variables do not need update in this setting. Our framework essentially makes learning harder as opposed to PEARL. In this setting, PEARL aligns well with the inductive bias that no multi-stage sub-tasks exist. However, our method is much more flexible in the design of latent space and PEARL can be viewed as a special case of our model. If we already have prior knowl-

[2]The implementation of the baselines can be found in https://github.com/jonasrothfuss/ProMP and https://github.com/katerakelly/oyster.

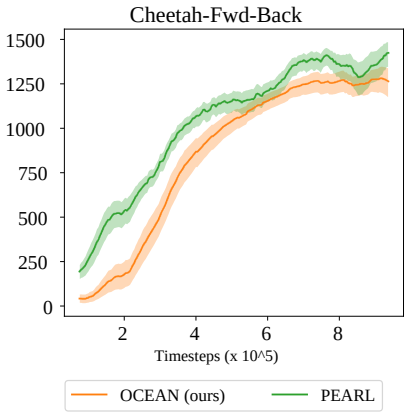

Figure 3: We compare OCEAN with PEARL in a task without multi-stage sub-tasks. Since this task does not require modeling the sub-task structure, we observe that OCEAN is at par with the task-inference baseline.

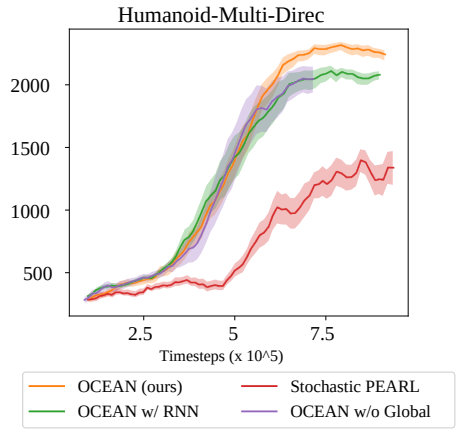

Figure 4: We compare OCEAN with several variants on Humanoid-Multi-Direc.

edge of the task structure, we can reduce our framework to PEARL.

We further evaluate all the baselines and OCEAN in several complex tasks that consist of sequence of sub-tasks, including Cheetah-Multi-Vel, Cheetah-Multi-Direc, Humanoid-Multi-Direc, Humanoid-Multi-Goal. As shown in Fig. 2, our framework OCEAN significantly outperforms all the other baselines in both sample efficiency and asymptotic performance. The final converged performance of the baselines are shown in dashed lines. We observe that RL2 achieves better performance than the gradient-based meta-RL models. The reason is that policy that is based on recurrent model can naturally takes the multi-stage into account, however RL2 cannot reason over the uncertainty of the tasks and also does not have global context variables, both of which make RL2 limited especially when the task structure is complex, for example, in Humanoid-Multi-Direc and Humanoid-Multi-Goal. Although in 2D cheetah environments, RL2 is able to achieve comparable and even better results than PEARL, in these 3D environments, RL2 fails miserably due to its incompetence of probabilistic task inference.

5.3 ABLATION STUDY

Here we conduct several ablative experiments to evaluate the importance of each component in our model.

Architecture. We first ablate the variational recurrent neural network architecture and investigate the benefit of the stochastic transition function as in Eq. 5 and the dependency between the prior distribution of neighbour steps. We replace the VRNN module with a normal LSTM [30] as the architecture of the local context encoder q_ϕ^L, resulting in **OCEAN w/ RNN**. With a LSTM architecture, the hidden vector $\boldsymbol{h}_t$ at timestep t no longer depends on stochastic latent code $\boldsymbol{z}_t^{Local}$ and the prior distribution of every $\boldsymbol{z}_t^{Local}$ will be uninformative prior as $\boldsymbol{z}_0^{Local}$. As shown in Fig. 4, in Humanoid-Multi-Direc, OCEAN w/ RNN can still achieve better performance than baseline PEARL because it is able to adaptively adjust the local context variables, but it is likely to get stuck into local minima and performs worse than OCEAN with VRNN architecture as it does not model the dependencies between steps.

Global Latent Space. Next, we aim to show the importance of global context variables by designing a variant of our framework without global context variables **OCEAN w/o Global**. During both meta-training and meta-test, the agent only leverages local context variables to infer the sub-tasks and updates the local context variables online. However, the drawback is that the method cannot leverage the explored contexts (from other episodes) to achieve more information about the task, in other words, the agent does not have a memory buffer and may always start by exploring randomly in the environment and gradually adapt to the task, whereas, our model OCEAN can leverage the previous contexts to infer the sub-task structure with a global view, the global context variables also narrow down an agent's exploration area at the beginning of an episode. As shown in Fig. 4, we observe that OCEAN w/o Global has higher variance in training than OCEAN since it generally requires more exploration steps to gradually take optimal actions.

Update Posterior Locally. Here we investigate the benefit of adaptively updating the posterior online and design a variant of PEARL with fixed posterior. PEARL infers the posterior of the context variables, samples the context variables from the posterior at the beginning of an episode and holds them constant across the episode. Thus, PEARL can be viewed as freezing the posterior as

	PEARL	Stochastic PEARL	OCEAN
Posterior	Fixed	Fixed	Update
Context Variables	Fixed	Update	Update

Table 1: Comparison among three methods. PEARL has fixed posterior and context variables; Stochastic PEARL, a variant of PEARL, repeatedly samples the context variables from the fixed posterior at each step; OCEAN updates both the posterior and the context variables online.

well as the sample of global context variables. In order to adapt PEARL to the multi-stage setting, one variant of PEARL is to resample the context variables at each step from the same posterior, named **Stochastic PEARL**. In this sense, the context variables are still updated "locally". We list the difference in Table 1. The result on Humanoid-Multi-Direc is shown in Fig. 4, Stochastic PEARL behaves poorly in this multi-stage task, and barely improves compared with PEARL. The results demonstrate that naïvely sampling from a fixed posterior does not improve the accuracy of task inference, since it only introduces additional noise instead of effectively reasoning about the compositional structure.

5.4 INCORPORATE PRIOR KNOWLEDGE OF TASK STRUCTURE

Here we investigate how to leverage prior knowledge of the task structure by designing the appropriate latent space. We conduct experiments on 2D point robot navigation environments, where the objective of the agent is to navigate to the goal, distributed on the edge of a half circle. In order to create a multi-modal task distribution, we sample goal locations as interpolations between the two end points of the half circle, where the interpolation weight follows Dirichlet distribution $Dir(0.2, 0.2)$. Since we already know the goals (tasks) are Dirichlet distributed, we can design a latent space with a Dirichlet prior. Note that since this task is not multi-stage, we did not use the local context variables and only investigate the effect of different global latent space on the performance. For comparison, we also design two alternatives with Gaussian or Categorical distribution as the global latent space. As listed in Table. 2, OCEAN with Dirichlet prior displays the best performance as it aligns well with the task structure. The categorical distribution on the other hand is not expressive enough for this setting. This result demonstrates that it is beneficial to incorporate the prior knowledge of task structure as inductive bias to the framework for more accurate task inference.

Return	Gaussian	Categorical	Dirichlet
Point-Robot	10.0	8.5	11.7

Table 2: The result of OCEAN with different choices of global prior distribution in a point-robot navigation task whose goals follow the Dirichlet distribution.

6 CONCLUSION AND FUTURE WORK

In this paper we propose a generalized online task inference framework for meta-reinforcement learning based on online variational inference. The framework consists of global and local latent context variables estimated by two encoders respectively. The global variables capture the general structure of the tasks and can be tailored to the task if the structure is known to the user. The local variables are updated and estimated online across an episode that enables the agent to make smooth transitions from one sub-task to another. Extensive experiments on real-world continuous control tasks have shown superiority of our framework over the state-of-the-art meta-RL methods. One exciting future direction is to train the local context encoders with an auxiliary task to predict the time step of the transition. This auxiliary inference may help the agent in the exploration steps and can also improve the accuracy of sub-task inference. Another future direction is to utilize the contexts from previous episodes in the training of the local context encoder so that we do not need to always start with sampling local context variables from uninformative prior.

Acknowledgements

A.G. is a CIFAR AI chair and also acknowledges Vector Institute for computing support. J. L. is a Chan Zuckerberg Biohub investigator. We gratefully acknowledge the support of DARPA under Nos. FA865018C7880 (ASED), N660011924033 (MCS); ARO under Nos. W911NF-16-1-0342 (MURI), W911NF-16-1-0171 (DURIP); NSF under Nos. OAC-1835598 (CINES), OAC-1934578 (HDR), CCF-1918940 (Expeditions), IIS-2030477 (RAPID); Stanford Data Science Initiative, Wu Tsai Neurosciences Institute, Chan Zuckerberg Biohub, Amazon, Boeing, Chase, Docomo, Hitachi, Huawei, JD.com, NVIDIA, Dell. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ARO, or the U.S. Government.

References

[1] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning (ICML)*, 2017.

[2] B. C. Stadie, G. Yang, R. Houthooft, X. Chen, Y. Duan, Y. Wu, P. Abbeel, and I. Sutskever, "Some considerations on learning to explore via meta-reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[3] J. Rothfuss, D. Lee, I. Clavera, T. Asfour, and P. Abbeel, "Promp: Proximal meta-policy search," in *International Conference on Learning Representations (ICLR)*, 2019.

[4] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, "Rl2: Fast reinforcement learning via slow reinforcement learning," *arXiv preprint arXiv:1611.02779*, 2016.

[5] K. Rakelly, A. Zhou, D. Quillen, C. Finn, and S. Levine, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in *International Conference on Machine Learning (ICML)*, 2019.

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

[7] L. Zintgraf, M. Igl, K. Shiarlis, A. Mahajan, K. Hofmann, and S. Whiteson, "Variational task embeddings for fast adapta-tion in deep reinforcement learning," in *International Conference on Learning Representations Workshop (ICLRW)*, 2019.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning (ICML)*, 2018.

[9] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, 2015.

[11] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning.," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2008.

[12] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, 2014.

[14] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *International Conference on Learning Representations (ICLR)*, 2017.

[15] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *International Conference on Learning Representations (ICLR)*, 2017.

[16] M. Figurnov, S. Mohamed, and A. Mnih, "Implicit reparameterization gradients," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[17] J. Schmidhuber, *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[18] S. Thrun and L. Pratt, "Learning to learn: Introduction and overview," in *Learning to learn*, Springer, 1998.

[19] Y. Bengio, S. Bengio, and J. Cloutier, *Learning a synaptic learning rule*. Citeseer, 1990.

[20] T. Xu, Q. Liu, L. Zhao, and J. Peng, "Learning to explore via meta-policy gradient," in *International Conference on Machine Learning (ICML)*, 2018.

[21] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick, "Learning to reinforcement learn," *arXiv preprint arXiv:1611.05763*, 2016.

[22] R. Fakoor, P. Chaudhari, S. Soatto, and A. J. Smola, "Meta-Q-Learning," in *International Conference on Learning Representations (ICLR)*, 2020.

[23] M. Strens, "A Bayesian framework for reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2000.

[24] T. G. Dietterich, "Hierarchical reinforcement learning with the MAXQ value function decomposition," *Journal of artificial intelligence research*, 2000.

[25] C. Florensa, Y. Duan, and P. Abbeel, "Stochastic neural networks for hierarchical reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2017.

[26] S. Li, R. Wang, M. Tang, and C. Zhang, "Hierarchical reinforcement learning with advantage-based auxiliary rewards," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[27] O. Nachum, S. S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[28] T. Haarnoja, K. Hartikainen, P. Abbeel, and S. Levine, "Latent space policies for hierarchical reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2018.

[29] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.

Appendix

A ADDITIONAL ABLATION

In this section, we perform some additional ablation experiments on a new set of environments to evaluate the performance of OCEAN that uses different latent space. Specifically, we tune the complexity of the new environments we proposed. For Cheetah-Multi-Vel (a, b), we tune the minimum goal velocity a and the maximum goal velocity b in order to control the difficulty of the tasks. For Cheetah-Multi-Direc, this task requires cheetah to run with a sequential backward/forward goal direction, e.g., [Backward, Forward, Backward]. We make the task even more challenging by creating a gap between the meta-train and meta-test tasks. Besides the transition steps, all the combination of goal directions in meta-test tasks are not covered in meta-train. For Humanoid-Multi-Goal-r, we tune the radius of the circle where we sample the tasks.

As shown in Fig. 5 and Fig. 6, OCEAN with local context variables generally works better than OCEAN without local context variables (PEARL), since local context variables can better model the transition of the sub-tasks. It often contributes to a boost in both asymptotic performance as well as sample efficiency, especially in Cheetah-Multi-Vel (-1.0, 3.0) and Cheetah-Multi-Direc-Complex, where the transition is clear and the agent cannot simply execute a same policy to approximately finish the complex task. It shows similar trends in Humanoid environments. In Cheetah-Multi-Vel (0.0, 5.0), since all the goal velocities are positive, an agent can learn to go forward with the averaged velocity in order to finish the task.

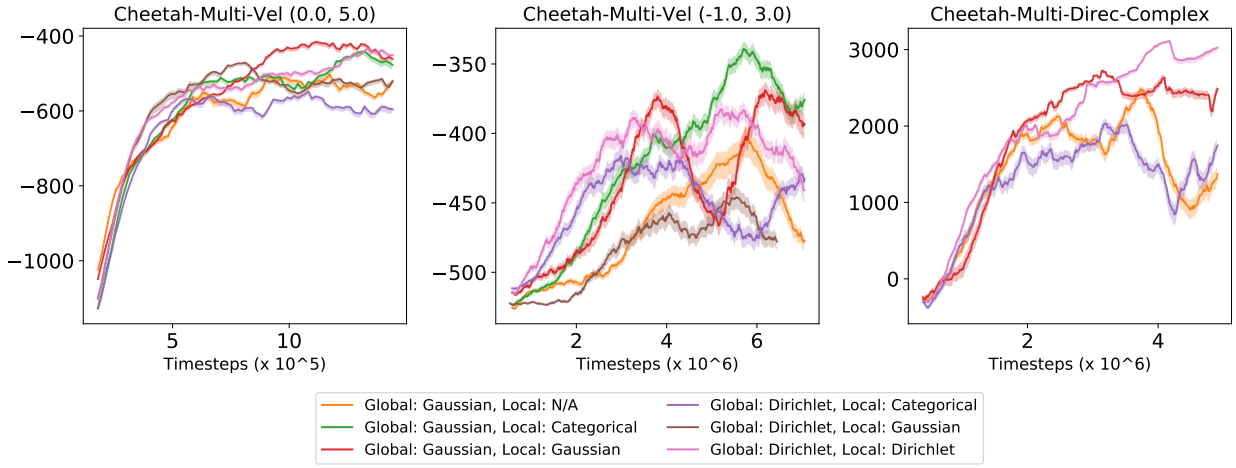

Figure 5: Additional experiments on Cheetah environments.

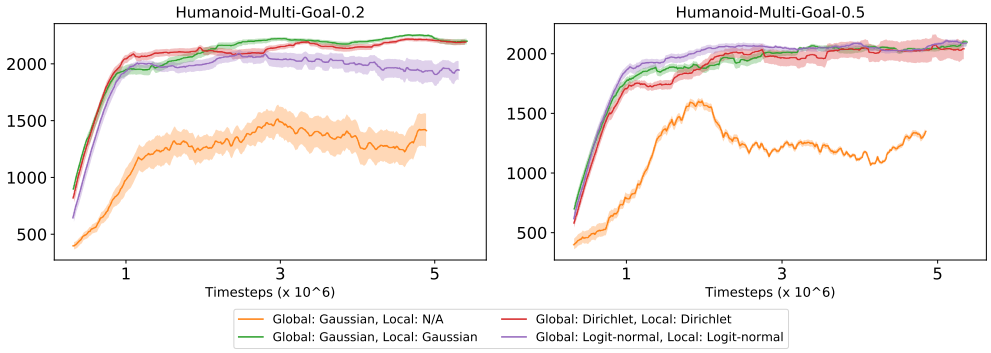

Figure 6: Additional experiments on Humanoid environments.

B EXPERIMENTAL DETAILS

We use Pytorch for OCEAN and can be found at https://github.com/pairlab/ocean. The latent dimension is 12 in total. We use the global latent space to be two Dirichlet distribution, each of dimension 3 to model the mixture of tasks. For local latent space, we use two categorical distribution, each of dimension 3 to model a single on-going sub-task. For fair comparison, the PEARL baseline also has 12 dimension of Gaussian random variables. We train our model for 300 epochs for all the continuous control environment. Each epoch contains 4000 iterations of SAC update. For the point-robot navigation experiment, the latent space we used are 6 dimension of Gaussian distribution versus 3 categorical distribution, each of 2 dimension, versus 3 Dirichlet distribution, each of 2 dimension. Hence, the number of random variables stay the same for fair comparison.

For the continuous control tasks, the number of training and test tasks on Cheetah is 50 and 10; the number of training and test tasks on Humanoid is 15 and 5. Given the small set of training tasks, our task is considered harder than the previous benchmarks, where they do random sampling at the beginning of each episode. The maximum episode length are both 500.