

GENERALISED LINEAR MODELS IN DEEP BAYESIAN RL WITH LEARNABLE BASIS FUNCTIONS

Anonymous authors

Paper under double-blind review

ABSTRACT

Bayesian Reinforcement Learning (BRL) provides a framework for generalisation of Reinforcement Learning (RL) problems from its use of Bayesian task parameters in the transition and reward models. However, classical BRL methods assume known forms of transition and reward models, reducing their applicability in real-world problems. As a result, recent deep BRL methods have started to incorporate model learning, though the use of neural networks directly on the joint data and task parameters requires optimising the Evidence Lower Bound (ELBO). ELBOs are difficult to optimise and may result in indistinctive task parameters, hence compromised BRL policies. To this end, we introduce a novel deep BRL method, **Generalised Linear Models in Deep Bayesian RL with Learnable Basis Functions (GLiBRL)**, that enables efficient and accurate learning of transition and reward models, with fully tractable marginal likelihood and Bayesian inference on task parameters and model noises. On challenging MetaWorld ML10 and ML45 benchmarks, GLiBRL improves the success rate of one of the state-of-the-art deep BRL methods, VariBAD, by up to $2.7\times$. [Comparing against representative or recent deep BRL / Meta-RL methods, such as MAML, RL², SDVT, TrMRL and ECET, GLiBRL also demonstrates its low-variance and decent performance consistently.](#)

1 INTRODUCTION

Reinforcement Learning (RL) algorithms have great potentials to enable robots to act intelligently without supervisions from humans. A well-known issue with RL algorithms is they generalise poorly to unseen tasks. Most standard RL algorithms by their designs do not consider possible variations in the transition and reward models, hence fail to adapt to new tasks whose models might be different from that of training environments.

Bayesian Reinforcement Learning (BRL) is an effective framework that can be used to improve the generalisation of RL. Instead of ignoring possible variations in transition and reward models, BRL methods explicitly take them into considerations by assuming parametric distributions of models and performing Bayesian inference on the parameters (Ghavamzadeh et al., 2015). Different parameters indicate different transition and reward models, hence implicitly encode various tasks. To solve BRL problems, many previous works use planners (Poupart et al., 2006; Guez et al., 2013) that search for Bayes-optimal policies. These methods are often limited in their scalability. Moreover, they require full information about the forms of transition and reward models, which restricts generalisation across different tasks.

Hence, recent deep BRL methods (Rakelly et al., 2019; Zintgraf et al., 2021) enable model learning by optimising the marginal likelihood of the data. However, most of the deep BRL methods do not support tractable Bayesian inference on the task parameters, because of the direct use of neural networks on the joint data and parameters. As a result, the exact marginal likelihood of the data is also not tractable and cannot be optimised directly. To this end, deep BRL methods adopt variational inference to optimise the evidence lower bound (ELBO) instead. However, the optimisation of ELBO is not an easy task as it may face issues such as high-variance Monte Carlo estimates, amortisation gaps (Cremer et al., 2018) and posterior collapse (Bowman et al., 2016; Dai et al., 2020). Such issues can preclude BRL methods from obtaining meaningful and distinctive distributions of task parameters, which are crucial to smooth Bayesian learning.

To alleviate the above issues, we propose **Generalised Linear Models in Deep Bayesian RL with Learnable Basis Functions (GLiBRL)**. GLiBRL assumes a generalised linear relation between task parameters and features of the data, computed via learnable basis functions. Such modelling allows exact posterior inference and marginal likelihood to be computed, avoiding optimising the challenging ELBO objective. Furthermore, GLiBRL generalises previous works such as (Harrison et al., 2018b) in that GLiBRL also performs inference on the noise, reducing the error of predictions in unseen tasks.

The performance of GLiBRL is evaluated with the challenging MetaWorld (Yu et al., 2021; McLean et al., 2025) benchmark and compared with standard baselines and recent methods, including deep BRL methods, VariBAD (Zintgraf et al., 2021) and SDVT (Lee et al., 2023), and Meta-Reinforcement Learning (Beck et al., 2023b) methods, including RL² (Wang et al., 2016; Duan et al., 2016), MAML (Finn et al., 2017), TrMRL (Melo, 2022) and ECET (Shala et al., 2025). GLiBRL has improved the success rate of one of the state-of-the-art deep BRL methods, VariBAD, by up to 2.7 \times , and has consistently outperformed all listed methods and demonstrated low-variance behaviours in the most complex subset of MetaWorld, ML45.

2 BACKGROUND

2.1 MARKOV DECISION PROCESSES AND REINFORCEMENT LEARNING

Markov Decision Processes (MDPs) are 5-tuples defined as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, R, T, \gamma)$, where \mathcal{S} is the set of *states*; \mathcal{A} , the set of *actions*; $R(s_t, a_t, s_{t+1}, r_{t+1}) = p(r_{t+1}|s_t, a_t, s_{t+1})$, the *reward* function; $T(s_t, a_t, s_{t+1}) = p(s_{t+1}|s_t, a_t)$, the *transition* function; and $\gamma \in (0, 1]$ a discount factor. In the above definition, $s_t \in \mathcal{S}, s_{t+1} \in \mathcal{S}, a_t \in \mathcal{A}$ and $r_{t+1} \in \mathbb{R}$. The goal of solving an MDP is to find a policy $\pi(a_t|s_t) : \mathcal{S} \rightarrow \mathcal{A}$, such that the expected return for a finite horizon $H > 0$, $\mathcal{J}(\pi) = \mathbb{E}_{T, \pi, R}[\sum_{t=0}^H \gamma^t R(s_t, a_t, s_{t+1}, r_{t+1})]$, is maximised.

While MDPs assume fully known transition and reward functions T and R , standard Reinforcement Learning (RL) problems optimise the same objective, however, with at least one of the T and R unknown but can be learnt from data samples.

2.2 BAYES-ADAPTIVE MDPs AND BAYESIAN REINFORCEMENT LEARNING

Bayes-Adaptive MDPs (BAMDPs) (Duff, 2002; Ghavamzadeh et al., 2015) is a Bayesian framework for solving RL. Compared to standard MDPs, BAMDPs assume *known* forms of functions of T and R , but parameterised by *unknown* parameters $\theta_T \in \Theta_T$ and $\theta_R \in \Theta_R$.

In BAMDPs, distributions (or, *beliefs*) $b_t = p(\theta_{T,t}, \theta_{R,t}) \in \mathcal{B}_T \times \mathcal{B}_R$ are placed on the unknown parameters, and updated to posteriors $b_{t+1} = p(\theta_{T,t+1}, \theta_{R,t+1})$ with Bayesian inference.

To efficiently use existing MDP frameworks, the beliefs can be absorbed into the original state space to form hyper-states $\mathcal{S}^+ = \mathcal{S} \times \mathcal{B}_T \times \mathcal{B}_R$. Hence, BAMDPs can be defined as 5-tuple $(\mathcal{S}^+, \mathcal{A}, R^+, T^+, \gamma)$ MDPs, where

$$\begin{aligned} T^+(s_t^+, a_t, s_{t+1}^+) &= p(s_{t+1}, b_{t+1}|s_t, b_t, a_t) \\ &= \mathbb{E}_{\theta_{T,t} \sim b_t} [p(s_{t+1}|s_t, a_t, \theta_{T,t})] \cdot \delta(b_{t+1} = p(\theta_{T,t+1}, \theta_{R,t+1})) \end{aligned} \quad (1)$$

$$\begin{aligned} R^+(s_t^+, a_t, s_{t+1}^+, r_{t+1}) &= p(r_{t+1}|s_t, b_t, a_t, s_{t+1}, b_{t+1}) \\ &= \mathbb{E}_{\theta_{R,t+1} \sim b_{t+1}} [p(r_{t+1}|s_t, a_t, s_{t+1}, \theta_{R,t+1})] \end{aligned} \quad (2)$$

The hyper-transition function (Equation 1) consists of the θ_T -parameterised expected regular MDP transition and a deterministic posterior update specified by the Dirac delta function $\delta(b_{t+1} = p(\theta_{T,t+1}, \theta_{R,t+1}))$. The hyper-reward function consists of the θ_R -parameterised expected regular MDP reward function. Accordingly, the expected return to maximise becomes $J^+(\pi^+) = \mathbb{E}_{T^+, \pi^+, R^+}[\sum_{t=0}^{H^+} \gamma^t R^+(s_t^+, a_t, s_{t+1}^+, r_{t+1})]$, where $H^+ > 0$ is the BAMDP horizon, and $\pi^+ : \mathcal{S}^+ \rightarrow \mathcal{A}$ is the policy of BAMDPs. Traditionally, problems that require solving BAMDPs are named Bayesian Reinforcement Learning (BRL). Aside from their generalisability, BRL methods are also recognised for offering principled approaches to the exploration–exploitation problem in RL (Ghavamzadeh et al., 2015).

108 However, classical BRL methods (Poupart et al., 2006; Guez et al., 2013; Tziortziotis et al., 2013)
 109 assume that the forms of transition T^+ and reward R^+ models are fully known, despite being pa-
 110 rameterised by unknown parameters. These methods are not sufficiently flexible in scenarios where
 111 the forms of T^+ and R^+ are not known a priori. A rough guess of the forms, however, may lead to
 112 significant underfit (e.g., assuming linear transitions while the ground truth is quadratic).

113 To generalise classical BRL methods, Hidden-Parameter MDPs (HiP-MDPs) (Doshi-Velez &
 114 Konidaris, 2016; Killian et al., 2017; Yao et al., 2018) have started to learn the forms of models
 115 through performing Bayesian inference on the weights. Doshi-Velez & Konidaris (2016) proposed
 116 HiP-MDPs with Gaussian Processes (GPs) to learn the basis functions for approximating transition
 117 models. Afterwards, Killian et al. (2017) discovered the poor scalability from the use of GPs, and
 118 applied Bayesian Neural Networks (BNNs) in HiP-MDPs for larger scale problems. (Yao et al.,
 119 2018) proposed to fix the weights of BNNs during evaluation for improved efficiency, at the cost of
 120 losing most of the test-time Bayesian features. These works focus on performing Bayesian inference
 121 on the weights of BNNs, which does not scale well with the size of BNN and is empirically
 122 demonstrated by (Yang et al., 2019). Reward functions in the HiP-MDP setting are also assumed to
 123 be known, which is generally infeasible in real-world applications .

124 On the other hand, recent deep BRL methods (Harrison et al., 2018a; Rakelly et al., 2019; Zint-
 125 graf et al., 2021) adopt scalable regular deep neural networks, as Bayesian features remain by
 126 performing (approximate) Bayesian inference on task parameters θ_T, θ_R directly. GLiBRL fol-
 127 lows this line of works for scalability and also the more general assumption of unknown reward
 128 functions. We briefly introduce how the learning of the forms of transition and reward models is
 129 done in the deep BRL setting. The deep BRL agent is provided with MDPs with unknown tran-
 130 sitions T and / or rewards R , and simulators from which the agent can obtain samples of tuples,
 131 known as *contexts* (Rakelly et al., 2019; Zintgraf et al., 2021). The context at step t is defined as
 132 $c_t := (s_t \in \mathbb{R}^{1 \times D_S}, a_t \in \mathbb{R}^{1 \times D_A}, s_{t+1} \in \mathbb{R}^{1 \times D_S}, r_{t+1} \in \mathbb{R}^{1 \times 1})$. Denote those MDPs as \mathcal{M} , con-
 133 texts obtained from \mathcal{M} as $\mathcal{C}^{\mathcal{M}}$ where

$$134 \mathcal{C}^{\mathcal{M}} := \{c_t\}_{t=1}^N = \{s_t \in \mathbb{R}^{1 \times D_S}, a_t \in \mathbb{R}^{1 \times D_A}, s_{t+1} \in \mathbb{R}^{1 \times D_S}, r_{t+1} \in \mathbb{R}^{1 \times 1}\}_{t=1}^N$$

135 During deep BRL training, assume we have access to MDPs $\{\mathcal{M}_i\}_{i=1}^M$ and the joint contexts $\mathcal{C} =$
 136 $\bigcup_{i=1}^M \mathcal{C}^{\mathcal{M}_i}$. The objective of most recent deep BRL methods (Rakelly et al., 2019; Perez et al., 2020;
 137 Zintgraf et al., 2021) is to maximise the marginal log-likelihood of the joint context¹

$$139 \log p_{\zeta, \phi_T, \phi_R}(\mathcal{C}) = \sum_{i=1}^M \log p_{\zeta, \phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i})$$

$$141 = \sum_{i=1}^M \log \int \int p_{\zeta}(\theta_T, \theta_R) p_{\phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i} | \theta_T, \theta_R) d\theta_T d\theta_R$$
(3)

142 where ζ is a neural network to learn the parameters of the prior distribution $p(\theta_T, \theta_R)$,
 143 $p_{\phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i} | \theta_T, \theta_R) \propto \prod_{t=1}^N p_{\phi_T}(s_{t+1} | s_t, a_t, \theta_T) p_{\phi_R}(r_{t+1} | s_t, a_t, s_{t+1}, \theta_R)$, and ϕ_T, ϕ_R are neu-
 144 ral networks to learn the forms of transition and reward functions.

145 For the ease of learning, p_{ϕ_T} and p_{ϕ_R} are generally assumed to be Gaussian with mean and diagonal
 146 covariance determined by the output of neural networks ϕ_T, ϕ_R , and the prior $p_{\zeta}(\theta_T, \theta_R)$ is also
 147 assumed to be a Gaussian. However, note that even with these simplifications, Equation 3 is still
 148 not tractable as θ_T, θ_R are not linear with respect to contexts, because of the use of neural networks.
 149 Fortunately, variational inference provides a lower bound to Equation 3, named Evidence Lower
 150 Bound (ELBO) that can be used as an approximate objective function (proof see Appendix A.1):

$$151 \log p_{\zeta, \phi_T, \phi_R}(\mathcal{C}) \geq \sum_{i=1}^M \mathbb{E}_q [\log p_{\phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i} | \theta_T, \theta_R)] - D_{KL}(q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i}) || p_{\zeta}(\theta_T, \theta_R))$$
(4)

152 where $D_{KL}(\cdot || \cdot)$ is the KL-divergence, and $q(\cdot)$ is an approximate Gaussian posterior of θ_T, θ_R . An
 153 optimised $\log p_{\zeta, \phi_T, \phi_R}(\mathcal{C})$ will bring in models for transitions and rewards, with which both model-
 154 free (Rakelly et al., 2019; Zintgraf et al., 2021) and model-based (Guez et al., 2013; Harrison et al.,
 155 2018a) methods can be applied for learning the BRL policy.

156 ¹Henceforth, we drop the dependence on time step t of θ_T, θ_R for brevity.

ELBO-like objectives enable the learning of transition and reward models. However, ELBOs are challenging to optimise, for known issues such as high-variance Monte Carlo estimates, amortisation gaps (Cremer et al., 2018) and posterior collapse (Bowman et al., 2016; Dai et al., 2020). Unoptimised ELBOs may result in scenarios where learnt latent representations (e.g., θ_T, θ_R in BRL) are not meaningful and distinctive. Different from other tasks where meaningful latent representations are less important, BRL policies determine the next action to perform heavily dependent on continually updated distributions of latent representations. Indistinctive latent representations, hence meaningless posterior updates will substantially harm the performance of BRL policies.

Aside from issues in ELBO, it is also concerning how previous methods compute the posterior $q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i})$. As $\mathcal{C}^{\mathcal{M}_i}$ contains variable and large number of contexts, it is inefficient to directly use it as a conditional variable. Instead, Rakelly et al. (2019) applied factored approximation so that $q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i}) \approx \prod_{t=1}^N \mathcal{N}(\theta_T, \theta_R | g([\mathcal{C}^{\mathcal{M}_i}]_t))$, where $g(\cdot)$ is a neural network that takes the t -th context in $\mathcal{C}^{\mathcal{M}_i}$ as the input and returns the mean and covariance of a Gaussian as the output. From Bayes rule, $q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i}) = p(\theta_T, \theta_R)^{1-N} \prod_{t=1}^N q(\theta_T, \theta_R | [\mathcal{C}^{\mathcal{M}_i}]_t)$. This is to say, for the approximation to be accurate, $\mathcal{N}(\theta_T, \theta_R | g([\mathcal{C}^{\mathcal{M}_i}]_t)) = p(\theta_T, \theta_R)^{1/N-1} q(\theta_T, \theta_R | [\mathcal{C}^{\mathcal{M}_i}]_t)$. From the left-hand-side, $g(\cdot)$ tries to predict the mean and the covariance regardless of the prior, while the right-hand-side has the prior involved, meaning the same $\mathcal{N}(\theta_T, \theta_R | g([\mathcal{C}^{\mathcal{M}_i}]_t))$ gets implicitly assigned with different targets as N increases. This may result in inaccuracies of the approximation and unstable training. On the other hand, Zintgraf et al. (2021) summarise $\mathcal{C}^{\mathcal{M}_i}$ with RNNs to get hidden variables h , and compute $q(\theta_T, \theta_R | h)$. Despite its simplicity, it has been shown in Rakelly et al. (2019) that permutation-variant structures like RNNs may lead to worse performance.

3 GENERALISED LINEAR MODELS IN BAYESIAN RL WITH LEARNABLE BASIS FUNCTIONS

Previous deep BRL methods perform inaccurately approximated posterior updates and optimise challenging ELBOs. Both issues may lead to incorrect distributions of task parameters, compromising the performance of BRL policies. To this end, we introduce our method, GLiBRL. GLiBRL features generalised linear models that enable fully tractable and permutation-invariant posterior update, hence closed-form marginal log-likelihood, without the need to evaluate and optimise the ELBO. The linear assumption seems strong, but basis functions still enable linear models to learn non-linear transitions and rewards. The basis functions that maps from the raw data $\mathcal{C}^{\mathcal{M}_i}$ to the feature space are made *learnable* from the marginal log-likelihood, instead of being chosen arbitrarily, allowing for efficient learning under low-dimensional feature space. We elaborate on the learning of the forms of transition and reward functions in Section 3.1 and discuss efficient online policy learning in Section 3.2. The full GLiBRL algorithm is demonstrated in Algorithm 1.

3.1 LEARNING THE FORMS OF TRANSITIONS AND REWARDS

We rewrite $\mathcal{C}^{\mathcal{M}_i} = (\mathbf{S}_i \in \mathbb{R}^{N \times D_S}, \mathbf{A}_i \in \mathbb{R}^{N \times D_A}, \mathbf{S}'_i \in \mathbb{R}^{N \times D_S}, \mathbf{r}_i \in \mathbb{R}^{N \times 1})$ for compactness, where N is the number of context and D_S, D_A are the dimensions of the state and action space. We further let $\theta_T = (T_\mu \in \mathbb{R}^{D_T \times D_S}, T_\sigma \in \mathbb{R}^{D_S \times D_S})$, $\theta_R = (R_\mu \in \mathbb{R}^{D_R \times 1}, R_\sigma \in \mathbb{R}^{1 \times 1})$, where D_T, D_R are task dimensions. Note we explicitly perform Bayesian inference on T_σ, R_σ , instead of assuming known model noises. We make the following approximation:

$$p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i, \theta_T) = \mathcal{MN}(\mathbf{S}'_i | \mathbf{C}_T T_\mu, \mathbf{I}_N, T_\sigma) \quad (5)$$

$$p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \theta_R) = \mathcal{MN}(\mathbf{r}_i | \mathbf{C}_R R_\mu, \mathbf{I}_N, R_\sigma) \quad (6)$$

where $\mathbf{C}_T, \mathbf{C}_R$ ² are features of contexts $\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i$ computed through neural networks which act as learnable basis functions

$$\mathbf{C}_T \in \mathbb{R}^{N \times D_T} = \phi_T(\mathbf{S}_i, \mathbf{A}_i) \quad \mathbf{C}_R \in \mathbb{R}^{N \times D_R} = \phi_R(\mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) \quad (7)$$

and $\mathcal{MN}(\mathbf{W} | \mathbf{X}, \mathbf{Y}, \mathbf{Z})$ defines a matrix normal distribution with random matrix \mathbf{W} , mean \mathbf{X} , row covariance \mathbf{Y} and column covariance \mathbf{Z} . Different from other deep BRL methods such as PEARL (Rakelly et al., 2019) and VariBAD (Zintgraf et al., 2021), GLiBRL does not place neural networks on the joint contexts and task parameters (e.g., $\phi_T(\mathbf{S}_i, \mathbf{A}_i, \theta_T)$), in order for tractable inference.

²Dependence on i of $\mathbf{C}_T, \mathbf{C}_R$ is omitted for clarity. This also applies to $\mathbf{M}'_T, \mathbf{\Xi}'_T, \mathbf{\Omega}'_T, \mathbf{M}'_R, \mathbf{\Xi}'_R, \mathbf{\Omega}'_R$.

Algorithm 1: GLiBRL

```

216
217
218 Initialise: policy  $\pi_\psi^\dagger$ , horizon  $H$ ,  $\phi_T, \phi_R$ ;
219 while Training do
220   Sample  $K$  MDPs  $\{\mathcal{M}_i\}_{i=1}^K$ ;
221   Initialise: contexts  $\mathcal{C} = \{\}$ ;
222   // collecting contexts
223   for  $i \in \{1, 2, \dots, K\}$  do
224     Initialise: contexts in  $\mathcal{M}_i$ ,  $\mathcal{C}^{\mathcal{M}_i} = \{\}$ , state  $s$ ;
225     for  $t < H$  do
226        $b \leftarrow p_{\phi_T, \phi_R}(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i})$ ;
227        $a \sim \pi_\psi^\dagger(a | s, b)$ ;
228       Execute  $a$  from  $s$  in  $\mathcal{M}_i$  to get  $s', r$ ;
229        $s \leftarrow s'$ ;
230        $\mathcal{C}^{\mathcal{M}_i} \leftarrow \mathcal{C}^{\mathcal{M}_i} \cup \{s, a, s', r\}$ ;
231    $\mathcal{C} = \mathcal{C} \cup \mathcal{C}^{\mathcal{M}_i}$ 
232   // learning policy, transition and reward models
233   while Learning do
234     Sample  $\mathcal{D} \subseteq \mathcal{C}$ ;
235      $\psi \leftarrow \psi - \nabla_\psi \mathcal{L}_{\text{Policy}}(\mathcal{D})$ ;
236      $\phi_T \leftarrow \phi_T - \nabla_{\phi_T} \mathcal{L}_{\text{model}}(\mathcal{D})$ ;
237      $\phi_R \leftarrow \phi_R - \nabla_{\phi_R} \mathcal{L}_{\text{model}}(\mathcal{D})$ ;

```

Assuming the independence of θ_T and θ_R , dropping the neural network ζ of the prior, Equation 3 can be written as

$$\begin{aligned} \log p_{\phi_T, \phi_R}(\mathcal{C}) &= \sum_{i=1}^M \log \int p(\theta_T) p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i, \theta_T) d\theta_T \\ &\quad + \sum_{i=1}^M \log \int p(\theta_R) p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \theta_R) d\theta_R + \text{const.} \end{aligned} \quad (8)$$

Because of the linear relationship between θ_T, θ_R and features of the contexts, we can place Normal-Wishart priors conjugate to matrix normals on θ_T, θ_R for tractable inference

$$\begin{aligned} p(\theta_T) &= p(T_\mu, T_\sigma) = \mathcal{MN}(T_\mu | \mathbf{M}_T \in \mathbb{R}^{D_T \times D_S}, \mathbf{\Xi}'_T^{-1} \in \mathbb{R}^{D_T \times D_T}, T_\sigma) \\ &\quad \cdot \mathcal{W}(T_\sigma^{-1} | \mathbf{\Omega}'_T^{-1} \in \mathbb{R}^{D_S \times D_S}, \nu_T \in \mathbb{R}_{++}) \end{aligned} \quad (9)$$

$$\begin{aligned} p(\theta_R) &= p(R_\mu, R_\sigma) = \mathcal{MN}(R_\mu | \mathbf{M}_R \in \mathbb{R}^{D_R \times 1}, \mathbf{\Xi}'_R^{-1} \in \mathbb{R}^{D_R \times D_R}, R_\sigma) \\ &\quad \cdot \mathcal{W}(R_\sigma^{-1} | \mathbf{\Omega}'_R^{-1} \in \mathbb{R}^{1 \times 1}, \nu_R \in \mathbb{R}_{++}) \end{aligned} \quad (10)$$

where $\mathcal{W}(\mathbf{W} | \mathbf{X}, \nu)$ defines a Wishart distribution on positive definite random matrix \mathbf{W} with scale \mathbf{X} and degrees of freedom ν . It has been shown in Appendix A.2 that the posteriors are also Normal-Wishart distributions

$$p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \mathcal{MN}(T_\mu | \mathbf{M}'_T, \mathbf{\Xi}'_T{}^{-1}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1} | \mathbf{\Omega}'_T{}^{-1}, \nu'_T) \quad (11)$$

$$p_{\phi_R}(\theta_R | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \mathbf{r}) = \mathcal{MN}(R_\mu | \mathbf{M}'_R, \mathbf{\Xi}'_R{}^{-1}, R_\sigma) \cdot \mathcal{W}(R_\sigma^{-1} | \mathbf{\Omega}'_R{}^{-1}, \nu'_R) \quad (12)$$

where

$$\begin{aligned} \mathbf{M}'_T &= \mathbf{\Xi}'_T{}^{-1} [\mathbf{C}_T^\top \mathbf{S}' + \mathbf{\Xi}_T \mathbf{M}_T] & \mathbf{M}'_R &= \mathbf{\Xi}'_R{}^{-1} [\mathbf{C}_R^\top \mathbf{r} + \mathbf{\Xi}_R \mathbf{M}_R] \\ \mathbf{\Xi}'_T &= \mathbf{C}_T^\top \mathbf{C}_T + \mathbf{\Xi}_T & \mathbf{\Xi}'_R &= \mathbf{C}_R^\top \mathbf{C}_R + \mathbf{\Xi}_R \\ \mathbf{\Omega}'_T &= \mathbf{\Omega}_T + \mathbf{S}'^\top \mathbf{S}' & \mathbf{\Omega}'_R &= \mathbf{\Omega}_R + \mathbf{r}^\top \mathbf{r} \\ &+ \mathbf{M}_T^\top \mathbf{\Xi}_T \mathbf{M}_T - \mathbf{M}'_T{}^\top \mathbf{\Xi}'_T \mathbf{M}'_T & &+ \mathbf{M}_R^\top \mathbf{\Xi}_R \mathbf{M}_R - \mathbf{M}'_R{}^\top \mathbf{\Xi}'_R \mathbf{M}'_R \\ \nu'_T &= \nu_T + N & \nu'_R &= \nu_R + N \end{aligned} \quad (13)$$

Thus, we can find a closed-form marginal log-likelihood (proof in Appendix A.3):

$$\log p_{\phi_T, \phi_R}(\mathcal{C}) = -\frac{1}{2} \sum_{i=1}^M D_S \log |\Xi'_T| + \nu'_T \log \left| \frac{1}{2} \Omega'_T \right| + \log |\Xi'_R| + \nu'_R \log \left| \frac{1}{2} \Omega'_R \right| + \text{const.} \quad (14)$$

Equation 14 is to be maximised with related to \mathbf{C}_T and \mathbf{C}_R , hence ϕ_T and ϕ_R . We add squared Frobenius norms $\|\mathbf{C}_T\|_F^2$ and $\|\mathbf{C}_R\|_F^2$ to Equation 14 as regularisations, the effect of which being discussed in Appendix A.8. The regularised loss function is defined as

$$\mathcal{L}_{\text{model}} := -\log p_{\phi_T, \phi_R}(\mathcal{C}) + \lambda_T \|\mathbf{C}_T\|_F^2 + \lambda_R \|\mathbf{C}_R\|_F^2 \quad (15)$$

where $\lambda_T > 0$ and $\lambda_R > 0$ are hyperparameters. We note that $\mathcal{L}_{\text{model}}$ can be directly minimised with gradient descent, without the need to evaluate and optimise the ELBO.

3.2 LEARNING THE POLICY

A natural follow-up question is how to collect the contexts \mathcal{C} . We adopt a learnable BAMDP policy $\pi_\psi^+(a_t|s_t, b_t)$ parameterised by the neural network ψ . \mathcal{C} can be collected by rolling out π_ψ^+ in training MDPs $\{\mathcal{M}_i\}_{i=1}^M$, and ψ can be updated using \mathcal{C} with model-free or model-based RL methods.

$\pi_\psi^+(a_t|s_t, b_t)$ takes beliefs b_t as its input. To roll out π_ψ^+ online, the prior b_t needs to be continually updated to the posterior b_{t+1} from the new context $c = \{s_t, a_t, s'_t, r_{t+1}\}$, using the learnt transition and reward models. Therefore, fast posterior update is crucial for efficient context collections. One of the most time-consuming part in Equation 13 is the inversion of Ξ'_T and Ξ'_R , which is of time complexity $O(D_T^3)$ and $O(D_R^3)$, respectively. Fortunately, with the matrix inversion lemma

$$\Xi_T^{-1} = (\mathbf{C}_T^T \mathbf{C}_T + \Xi_T)^{-1} = \Xi_T^{-1} - \Xi_T^{-1} \mathbf{C}_T^T (\mathbf{I}_N + \mathbf{C}_T \Xi_T^{-1} \mathbf{C}_T^T)^{-1} \mathbf{C}_T \quad (16)$$

$$\Xi_R^{-1} = (\mathbf{C}_R^T \mathbf{C}_R + \Xi_R)^{-1} = \Xi_R^{-1} - \Xi_R^{-1} \mathbf{C}_R^T (\mathbf{I}_N + \mathbf{C}_R \Xi_R^{-1} \mathbf{C}_R^T)^{-1} \mathbf{C}_R \quad (17)$$

When updating the belief online with the new context, $\mathbf{C}_T \in \mathbb{R}^{1 \times D_T}$ and $\mathbf{C}_R \in \mathbb{R}^{1 \times D_R}$. Hence, $(\mathbf{I}_N + \mathbf{C}_T \Xi_T^{-1} \mathbf{C}_T^T)^{-1}$ and $(\mathbf{I}_N + \mathbf{C}_R \Xi_R^{-1} \mathbf{C}_R^T)^{-1}$ are reduced to reciprocals of scalars. Keeping track of the inverse of priors and posteriors, the inversion only takes $O(D_T^2)$ and $O(D_R^2)$. The full online update of all parameters takes $O(\max\{D_T^2 D_S, D_S^2 D_T\})$ and $O(D_R^2)$.

4 RELATED WORK

ALPaCA. First, we discuss the most relevant work, ALPaCA (Harrison et al., 2018b). ALPaCA is an efficient and flexible online Bayesian linear regression framework, which also involves Bayesian linear models with learnable basis functions. ALPaCA initially was not proposed as an BRL method, though follow-up work such as CAMELiD (Harrison et al., 2018a) uses controllers to compute the policy assuming known reward functions. Our method, GLiBRL, generalises ALPaCA and CAMELiD in (1) ALPaCA and CAMELiD assume a *known* noise in the likelihood function, instead of performing Bayesian inference, (2) ALPaCA and CAMELiD are not evaluated in online BRL settings. They only investigated scenarios where offline contexts are available with unknown transitions and relatively simple known rewards. In Section 5, we will argue empirically that the assumption of known noises incurs error in both predictions of transitions and rewards.

Reinforcement Learning. RL methods can be categorised as model-free and model-based. We use the former in this paper to learn BRL policies as a large proportion of RL work is model-free, such as Trust-Region Policy Optimisation (TRPO) (Schulman et al., 2015), Proximal Policy Optimisation (PPO) (Schulman et al., 2017) and Soft Actor-Critic (SAC) (Haarnoja et al., 2018). As GLiBRL learns the models, model-based methods can also be used for improved sample efficiency.

Hidden-Parameter MDPs. Hidden-Parameter MDP (HiP-MDP), proposed by Doshi-Velez & Konidaris (2016), is a framework for parametric Bayesian Reinforcement Learning. HiP-MDP was initially modelled using Gaussian Processes (GPs). Killian et al. (2017) improved the scalability by replacing GPs with Bayesian Neural Networks (BNNs). The weights of BNNs are updated with new data during evaluation, which has been empirically shown inefficient (Yang et al., 2019). Yao

et al. (2018) mitigated the inefficiency by fixing the test-time weights of BNNs and optimising task parameters. Despite the improved speed, we have observed that this would divert the agent from following Bayes-optimal policies. The shared objectives in (Killian et al., 2017) and (Yao et al., 2018) correspond to approximate Bayesian inference on BNN weights, but not on task parameters. Optimising the objective on task parameters with fixed BNN weights is equivalently performing Maximum Likelihood Estimations (MLEs) on task parameters, immediately removing the Bayesian features (which is also mentioned in (Zintgraf et al., 2021)). Most recent parametric deep BRL (Rakelly et al., 2019; Zintgraf et al., 2021; Lee et al., 2023), including GLiBRL, are considered orthogonal to this line of works as they perform (approximate) Bayesian inference on task parameters directly, rather than on the weights of the neural networks. Furthermore, HiP-MDPs also assume known reward functions, while GLiBRL and methods such as (Rakelly et al., 2019; Zintgraf et al., 2021; Lee et al., 2023) do not have this limiting assumption.

Classical Bayesian Reinforcement Learning. As mentioned in Section 2, classical BRL methods assume known forms of transitions and rewards. Poupart et al. (2006) presented a Partially Observable MDP (POMDP) formulation of BRL and a sampling-based offline solver. Guez et al. (2013) proposed an online tree-based solver, applying posterior sampling (Strens, 2000; Osband et al., 2013) for efficiency. Both methods use solver for the (approximately) optimal policy using planners, which is orthogonal to GLiBRL. GLiBRL shares the idea of using generalised linear models with Tziortziotis et al. (2013), while differs in that Tziortziotis et al. (2013) *chooses* the basis function, instead of learning them. Even a simple non-linear basis function, such as quadratic function, may result in $O(d^2)$ dimensional feature space, where d is the dimension of the raw input³. As demonstrated in Section 3, performing online Bayesian inference at least requires quadratic complexity with related to the feature dimension, meaning prohibitive $O(d^4)$ complexity is required for just a single inference step. In contrast, with learnt basis functions in GLiBRL, low-dimensional feature space are usually sufficient to capture the non-linearity, providing sufficient scalability.

Meta-Reinforcement Learning. Meta-Reinforcement Learning (Meta-RL) aims to learn policies from *seen* tasks that are capable of adapting to *unseen* tasks following similar task distributions (Beck et al., 2023b). According to Beck et al. (2023b), Meta-RL methods can be categorised as **(1) parameterised policy gradient (PPG)** methods (e.g., Finn et al. (2017); Yoon et al. (2018); Finn et al. (2018)) that learn by performing meta-policy gradients on the meta-parameterised policy; **(2) black-box methods** (e.g., Duan et al. (2016); Wang et al. (2016); Melo (2022); Shala et al. (2025)) that learn from summaries of histories and **(3) task-inference methods** (e.g., Rakelly et al. (2019); Zintgraf et al. (2021); Lee et al. (2023)) that learn from the belief of the task parameters (i.e., θ_T, θ_R in GLiBRL). Most of the deep BRL methods can be viewed as task inference methods, hence a subset of Meta-RL. BRL differs from other Meta-RL methods in that it performs inference on task parameters, hence enjoying nice properties such as uncertainty quantification. As a result, BRL methods have natural integrations with model-based algorithms, hence significant to research fields such as control and planning under uncertainty.

5 EXPERIMENTS

In this section, we investigate the performance of GLiBRL using MetaWorld (Yu et al., 2021; McLean et al., 2025), one of the most famous and challenging meta-RL benchmark. The most recent and standardised version, MetaWorld-V3 (McLean et al., 2025), is used for fairness of comparisons. We focus on the most challenging subset of the MetaWorld benchmark, Meta-Learning 10 (ML10) and Meta-Learning 45 (ML45). ML10 / ML45 consists of 10 / 45 training tasks and 5 testing tasks. The testing tasks remain unseen during training. In all tasks, $D_S = 39$ and $D_A = 4$.

We follow the majority of the experiment settings in McLean et al. (2025). ML10 and ML45 experiments are run for $2e7$ steps and $9e7$ steps, respectively. We do not allow any adaptive steps during test time, hence evaluating the zero-shot performance. The zero-shot performance is critical to Bayesian RL and Meta-RL, as the goal of Meta-RL is to adapt with as little data as possible (Beck et al., 2023b). We compare the Inter-Quartile Mean (IQM) of the results, as suggested by Agarwal et al. (2021). Each experiment is run with a single A100 GPU⁴ for 10 times.

³In GLiBRL, $d = D_S + D_A$ for transition models and $d = 2 \cdot D_S + D_A$ for reward models.

⁴A100 is not mandatory. GLiBRL is runnable with ≤ 8 GB GPU memory, see Appendix A.10.

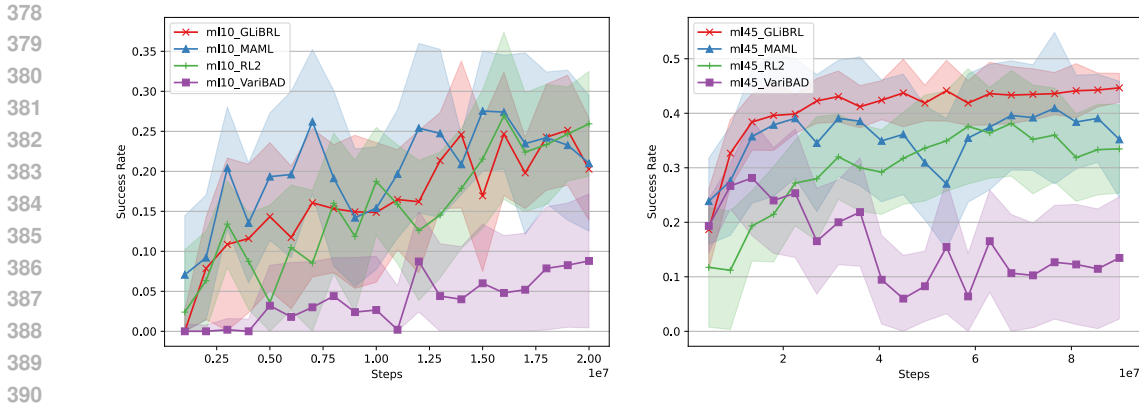


Figure 1: IQM and 95% CI of testing success rate of GLiBRL, VariBAD, MAML and RL², with related to the number of training steps. Left: the ML10 benchmark; Right: the ML45 benchmark.

We list the comparators as follows. First, we compare GLiBRL to standard deep BRL and Meta-RL baselines, including deep BRL method VariBAD (Zintgraf et al., 2021), PPG-based Meta-RL methods MAML, and black-box Meta-RL method RL². Following McLean et al. (2025); Beck et al. (2023a), MAML learns its policy with TRPO, and VariBAD, RL² and GLiBRL use PPO. Our PPO implementation uses standard linear feature baseline (Duan et al., 2016), as suggested by McLean et al. (2025). Afterwards, we compare GLiBRL to recent deep BRL and Meta-RL baselines, including deep BRL method SDVT (Lee et al., 2023) and two Transformer-based (Vaswani et al., 2017) black-box Meta-RL methods, TrMRL (Melo, 2022) and ECET (Shala et al., 2025). Finally, we perform ablation studies on whether it is useful to place a Wishart distribution on model noises (i.e., comparing with ALPaCA). It is also worth mentioning why we do not compare with PEARL – it has been demonstrated empirically in Yu et al. (2021) that PEARL performs much worse than other methods.⁵

5.1 IMPLEMENTATION DETAILS

We report the details of our implementations of GLiBRL and other methods. For GLiBRL, we set task dimensions $D_T = 16$ and $D_R = 256$ in both ML10 / ML45. We demonstrate an analysis on the sensitivity of GLiBRL with related to task dimensions D_T and D_R in Appendix A.9. The model networks ϕ_T, ϕ_R are Multi-Layer Perceptrons (MLPs) consisting of *feature* and *mixture* networks. Feature networks convert raw states and actions to features and are shared in ϕ_T and ϕ_R . Mixture networks mix the state and action features, further improving the representativeness. The training of $\pi_\psi^+(a|s, b)$ requires representing the beliefs b as parameters. Empirically, we find representing b using flattened mean matrices $\mathbf{M}_T, \mathbf{M}_R$ have the best performance. As flattening $\mathbf{M}_T \in \mathbb{R}^{D_T \times D_S}$ directly results in a large number of parameters, we consider instead flattening the lower triangle of $\mathbf{M}_T \mathbf{M}_T^T \in \mathbb{R}^{D_T \times D_T}$. The policy network then takes the flattened and normalised parameters as representations of the belief b . For MAML and RL², we use the implementation provided by Beck et al. (2023a). For VariBAD, we implement our own, following official implementations in Zintgraf et al. (2021); Beck et al. (2023a). The reason we re-implement VariBAD is to use the standardised framework provided by Beck et al. (2023a), written in JAX (Bradbury et al., 2018). Our VariBAD implementation has experiment results matching that of Beck et al. (2023a). We use the tuned hyperparameters from official implementations in MAML, RL² and VariBAD. The table of all related hyperparameters of GLiBRL is shown in Appendix A.10.

5.2 COMPARISONS AMONG GLiBRL, VARIBAD, MAML AND RL²

We demonstrate the results in Figure 1, which shows the testing success rates with related to training steps. The success rates are averaged across 5 testing tasks (with 10 seeds per experiment), and per-task success rates are shown in Appendix A.5 and Appendix A.6. Overall, GLiBRL substantially

⁵For example, PEARL barely succeeded ($< 3\%$) in ML10 with $1e8$ steps, see Figure 17 in Yu et al. (2021).

Table 1: Maximal testing success rate of GLiBRL, SDVT, TrMRL and ECET. Results from SDVT, TrMRL and ECET are taken as reported. Results of TrMRL are from ECET, as they are not reported in (Melo, 2022). †: results with $5e7$ training steps - corresponding GLiBRL results with $5e7$ steps are bracketed. Best results are in **bold**.

ML10				ML45			
GLiBRL	SDVT	TrMRL	ECET	GLiBRL	SDVT	TrMRL†	ECET†
0.25	0.19	0.14	0.18	0.45	0.20	0.23 (0.44)	0.38 (0.44)

outperforms one of the state-of-the-art deep BRL methods, VariBAD, and shows the lowest variance in both ML10 and ML45 benchmarks.

GLiBRL and VariBAD: The main comparison is between GLiBRL and VariBAD as they are both BRL methods. In both benchmarks, we can see substantial improvement, by up to $2.7\times$ in ML10, using GLiBRL. We also noticed an interesting decreasing trend of VariBAD in the ML45 benchmark. We suspect that the more number of training tasks leads to increased difficulty of learning meaningful and distinctive latent representations, partly because of the ELBO objective in VariBAD. By design, GLiBRL avoids the use of ELBO, hence achieving better and more stable performance. In Appendix A.7, we show that VariBAD suffers from posterior collapse even in the simpler benchmark ML10, while GLiBRL learns meaningful task representations.

GLiBRL, MAML and RL²: GLiBRL achieves consistently higher success rates than MAML and RL² in the more complex ML45 benchmark. Notably, GLiBRL also admits low variance which can be inferred from the tightest CI. In Appendix A.9, we show that GLiBRL reveals even **higher** success rates (**29%**) in ML10 when setting $D_T = 8$, at the cost of slightly higher predictive error.

5.3 COMPARISONS AMONG GLiBRL, SDVT, TrMRL AND ECET

Aside from comparisons against standard baselines in deep BRL and Meta-RL, we compare the success rate of GLiBRL in both ML10 / ML45 to more recent work, namely SDVT (Lee et al., 2023), TrMRL (Melo, 2022) and ECET (Shala et al., 2025).

Table 1 demonstrates the detailed results. We can observe that GLiBRL outperforms all of these recent deep BRL / Meta-RL methods. Comparing against computationally heavy Transformer-based models TrMRL and ECET, GLiBRL is more performant and lightweight ($\sim 200K$ parameters) hence applicable to, for example, mobile robots with lower-end computing resources.

Having outperformed most of the recent or state-of-the-art methods already, GLiBRL learns policies from PPO that is not fully revealing its potentials. Model-based policy learners that cannot be applied to black-box / PPG-based Meta-RL methods are expected to further improve the sample efficiency and performance of GLiBRL.

5.4 ABLATION STUDIES

GLiBRL can be viewed as a generalised deep BRL version of ALPaCA, as GLiBRL performs Bayesian inference on model noises T_σ, R_σ , while ALPaCA simply assumes $T_\sigma = \Sigma_T, R_\sigma = \Sigma_R$ are fully known a priori. Under the assumption of ALPaCA, Equation 14 reduces to (see Appendix A.4)

$$\log_{\phi_T, \phi_R}(C) = -\frac{1}{2} \sum_{i=1}^M D_S \log |\Xi'_T| - \text{Tr}(\Sigma_T^{-1} \mathbf{M}_T^T \Xi'_T \mathbf{M}'_T) - \frac{1}{2} \sum_{i=1}^M \log |\Xi'_R| - \text{Tr}(\Sigma_R^{-1} \mathbf{M}_R^T \Xi'_R \mathbf{M}'_R) + \text{const.} \quad (18)$$

We studied on whether inferring on model noises is necessary for learning accurate transition and reward models. GLiBRL and its variant without noise inference (GLiBRL_wo_NI) are tested with

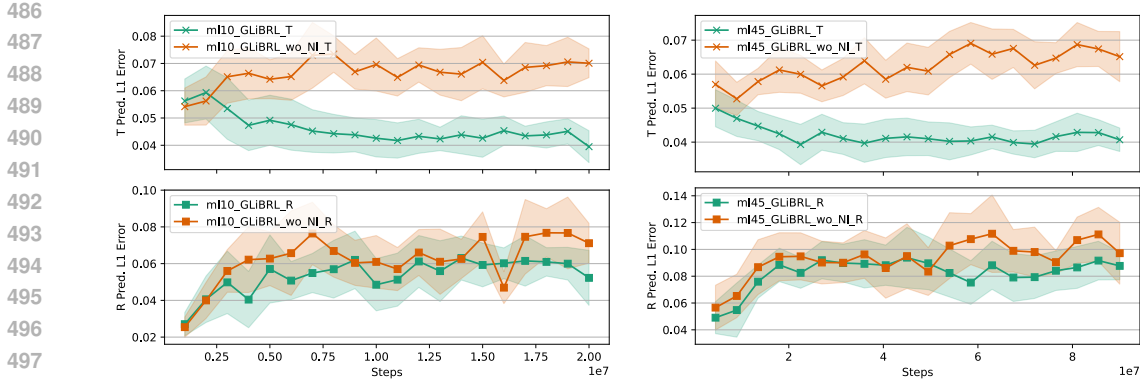


Figure 2: IQM and 95% CI of errors in transition and reward predictions, comparing GLiBRL and GLiBRL_wo_NI. Up: Transitions; Bottom: Rewards; Left: ML10; Right: ML45.

identical hyperparameters on both ML10 and ML45 benchmarks ⁶. The metric being evaluated are L_1 norms of prediction errors in both transitions (defined as $|\mathbf{S}' - \mathbf{C}_T T_\mu|_1$) and rewards (defined as $|\mathbf{r} - \mathbf{C}_R R_\mu|_1$). ⁷ The results are shown in Figure 2.

Overall, with noise inference, GLiBRL admits lower prediction errors in both transitions and rewards, compared to GLiBRL_wo_NI assuming known noises. Both methods become increasingly erroneous in reward predictions with training steps. This is expected, as more steps result in higher success rates, hence increased magnitude of rewards and errors. However, the increasing trend of transition errors of GLiBRL_wo_NI is abnormal, as magnitudes of states are rather bounded and less relevant with the success rate, compared to that of rewards. In Equation 18, the term governing the fit of the transition model $\text{Tr}(\Sigma_T^{-1} \mathbf{M}_T^T \Xi_T' \mathbf{M}_T)$ has a fixed learning rate from the fixed Σ_T , leading to continual unstable / underfit behaviours if the learning rate is too big / too small. On the contrary, in GLiBRL, the model fit term $\nu_T' \log |\frac{1}{2} \Omega_T'|$ has dynamic learning rates from dynamic ν_T' . This enables self-adaptive and effective model learning, hence the expected decreasing trend in Figure 2. The lower prediction error of GLiBRL allows better integrations with model-based methods using imaginary samples, the quality of which depending highly on the accuracy of the prediction.

6 CONCLUSION

We propose GLiBRL, a novel deep BRL method that enables fully tractable inference on the task parameters and efficient learning of basis functions with ELBO-free optimisation. Instead of assuming known noises of models, GLiBRL performs Bayesian inference, which has been shown empirically to reduce the error of prediction in both transition and reward models. The results on challenging MetaWorld ML10 and ML45 benchmarks demonstrate a substantial improvement compared to one of the state-of-the-art deep BRL methods, VariBAD. [Low-variance and decent performance of GLiBRL can also be inferred from its comparisons against representative or recent deep BRL / Meta-RL methods, including MAML, RL², SDVT, TrMRL and ECET.](#)

Multiple directions of future work arise naturally from the formulation of GLiBRL, with the most interesting one being model-based methods. As GLiBRL is capable of learning accurate transition and reward models, model-based methods can be applied easily for improved sample efficiency and performance. However, model-based methods usually require frequent sampling from the learnt models, revealing limitations in GLiBRL, as sampling from Wishart distributions can be slow. Another exciting direction is, if we prefer model-free methods, to seek a better way of utilising the task parameters in the policy network. In the paper, we simply feed the policy network with normalised means of task parameters. A naive normalisation of the parameters may confuse the policy network, and the use of means only loses uncertainty information from covariances.

⁶They also have the same initial noises. $(\nu_T \Omega_T)^{-1} = \Sigma_T = 0.025 \cdot \mathbf{I}$ and $(\nu_R \Omega_R)^{-1} = \Sigma_R = 0.5$.

⁷Comparisons of success rates are not included, as there is no obvious difference in IQMs or CIs.

540 ETHICS STATEMENT

541

542 This work does not raise any questions regarding the Code of Ethics.

543

544

545 REPRODUCIBILITY STATEMENT

546

547 All experiments can be reproduced with the source code provided in the supplementary materials.

548

549 REFERENCES

550

551 Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare.
552 Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural informa-*
553 *tion processing systems*, 34:29304–29320, 2021.

554

555 Jacob Beck, Matthew Thomas Jackson, Risto Vuorio, and Shimon Whiteson. Hypernetworks in
556 meta-reinforcement learning. In *Conference on Robot Learning*, pp. 1478–1487. PMLR, 2023a.

557

558 Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shi-
559 mon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*,
2023b.

560

561 Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio.
562 Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL conference*
563 *on computational natural language learning*, pp. 10–21, 2016.

564

565 James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal
566 Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao
567 Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL [http://](http://github.com/google/jax)
github.com/google/jax.

568

569 Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoen-
570 coders. In *International conference on machine learning*, pp. 1078–1086. PMLR, 2018.

571

572 Bin Dai, Ziyu Wang, and David Wipf. The usual suspects? reassessing blame for vae posterior
573 collapse. In *International conference on machine learning*, pp. 2313–2322. PMLR, 2020.

574

575 Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semi-
576 parametric regression approach for discovering latent task parametrizations. In *IJCAI: proceed-*
577 *ings of the conference*, volume 2016, pp. 1432, 2016.

578

579 Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. R^2 : Fast
580 reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

581

582 Michael O’Gordon Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov*
583 *decision processes*. University of Massachusetts Amherst, 2002.

584

585 Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation
586 of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

587

588 Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *Advances*
589 *in neural information processing systems*, 31, 2018.

590

591 Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement
592 learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

593

594 Arthur Guez, David Silver, and Peter Dayan. Scalable and efficient bayes-adaptive reinforcement
595 learning based on monte-carlo tree search. *Journal of Artificial Intelligence Research*, 48:841–
883, 2013.

596

597 Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy
598 maximum entropy deep reinforcement learning with a stochastic actor. In *International confer-*
599 *ence on machine learning*, pp. 1861–1870. Pmlr, 2018.

- 594 James Harrison, Apoorva Sharma, Roberto Calandra, and Marco Pavone. Control adaptation via
595 meta-learning dynamics. In *Workshop on Meta-Learning at NeurIPS*, volume 2018, 2018a.
596
- 597 James Harrison, Apoorva Sharma, and Marco Pavone. Meta-learning priors for efficient online
598 bayesian regression. In *International Workshop on the Algorithmic Foundations of Robotics*, pp.
599 318–337. Springer, 2018b.
- 600 Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and effi-
601 cient transfer learning with hidden parameter markov decision processes. *Advances in neural*
602 *information processing systems*, 30, 2017.
603
- 604 Suyoung Lee, Myungsik Cho, and Youngchul Sung. Parameterizing non-parametric meta-
605 reinforcement learning tasks via subtask decomposition. *Advances in Neural Information Pro-*
606 *cessing Systems*, 36:43356–43383, 2023.
- 607 Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng
608 He, KR Zentner, Ryan Julian, JK Terry, Isaac Woungang, et al. Meta-world+: An improved,
609 standardized, rl benchmark. *arXiv preprint arXiv:2505.11289*, 2025.
610
- 611 Luckeciano C Melo. Transformers are meta-reinforcement learners. In *international conference on*
612 *machine learning*, pp. 15340–15359. PMLR, 2022.
- 613 Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via
614 posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.
615
- 616 Christian Perez, Felipe Petroski Such, and Theofanis Karaletsos. Generalized hidden parameter
617 mdps: Transferable model-based rl in a handful of trials. In *Proceedings of the AAAI Conference*
618 *on Artificial Intelligence*, volume 34, pp. 5403–5411, 2020.
- 619 Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. An analytic solution to discrete
620 bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Ma-*
621 *chine learning*, pp. 697–704, 2006.
622
- 623 Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy
624 meta-reinforcement learning via probabilistic context variables. In *International conference on*
625 *machine learning*, pp. 5331–5340. PMLR, 2019.
- 626 John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region
627 policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR,
628 2015.
629
- 630 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy
631 optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- 632 Gresa Shala, André Biedenkapp, Pierre Krack, Florian Walter, and Josif Grabocka. Efficient cross-
633 episode meta-rl. In *The Thirteenth International Conference on Learning Representations*, 2025.
634
- 635 Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML*, volume 2000, pp.
636 943–950, 2000.
- 637 Nikolaos Tziortziotis, Christos Dimitrakakis, and Konstantinos Blekas. Linear bayesian reinforce-
638 ment learning. In *IJCAI*, pp. 1721–1728, 2013.
639
- 640 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez,
641 Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural informa-*
642 *tion processing systems*, 30, 2017.
- 643 Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos,
644 Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn.
645 *arXiv preprint arXiv:1611.05763*, 2016.
646
- 647 Jiachen Yang, Brenden Petersen, Hongyuan Zha, and Daniel Faissol. Single episode policy transfer
in reinforcement learning. *arXiv preprint arXiv:1910.07719*, 2019.

648 Jiayu Yao, Taylor Killian, George Konidaris, and Finale Doshi-Velez. Direct policy transfer via
649 hidden parameter markov decision processes. In *LLARLA Workshop, FAIM*, volume 2018, 2018.
650

651 Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn.
652 Bayesian model-agnostic meta-learning. *Advances in neural information processing systems*,
653 31, 2018.

654 Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya
655 Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and
656 evaluation for multi-task and meta reinforcement learning, 2021. URL <https://arxiv.org/abs/1910.10897>.
657

658 Luisa Zintgraf, Sebastian Schulze, Cong Lu, Leo Feng, Maximilian Igl, Kyriacos Shiarlis, Yarin
659 Gal, Katja Hofmann, and Shimon Whiteson. Varibad: Variational bayes-adaptive deep rl via
660 meta-learning. *Journal of Machine Learning Research*, 22(289):1–39, 2021.
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701

A APPENDIX

USE OF LLMs

This work does not use LLMs in any significant ways.

A.1 EVIDENCE LOWER BOUND

The evidence lower bound in Equation 4 is derived as follows

$$\begin{aligned}
 \log p_{\phi_T, \phi_R}(\mathcal{C}) &= \sum_{i=1}^M \log \iint p(\mathcal{C}^{\mathcal{M}_i}, \theta_T, \theta_R) \frac{q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i})}{q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i})} d\theta_T d\theta_R \\
 &\geq \sum_{i=1}^M \mathbb{E}_q \left[\log \frac{p(\theta_T, \theta_R)}{q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i})} + \log p(\mathcal{C}^{\mathcal{M}_i} | \theta_T, \theta_R) \right] \\
 &= \sum_{i=1}^M \mathbb{E}_q [\log p(\mathcal{C}^{\mathcal{M}_i} | \theta_T, \theta_R)] - D_{KL}(q(\theta_T, \theta_R | \mathcal{C}^{\mathcal{M}_i}) || p(\theta_T, \theta_R))
 \end{aligned}$$

A.2 NORMAL-WISHART-NORMAL CONJUGACY

Given Equation 9

$$p(\theta_T) = p(T_\mu, T_\sigma) = \mathcal{MN}(T_\mu | \mathbf{M}_T \in \mathbb{R}^{D_T \times D_S}, \mathbf{\Xi}_T^{-1} \in \mathbb{R}^{D_T \times D_T}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1} | \mathbf{\Omega}_T^{-1} \in \mathbb{R}^{D_S \times D_S}, \nu_T \in \mathbb{R}_{++}) \quad (9)$$

and Equation 5

$$p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i, \theta_T) = \mathcal{MN}(\mathbf{S}'_i | \mathbf{C}_T T_\mu, \mathbf{I}_N, T_\sigma) \quad (5)$$

We prove Equation 11

$$p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \mathcal{MN}(T_\mu | \mathbf{M}'_T, \mathbf{\Xi}'_T^{-1}, T_\sigma) \cdot \mathcal{W}(T_\sigma^{-1} | \mathbf{\Omega}'_T^{-1}, \nu'_T) \quad (11)$$

where

$$\begin{aligned} \mathbf{M}'_T &= \mathbf{\Xi}'_T^{-1} [\mathbf{C}_T^T \mathbf{S}' + \mathbf{\Xi}_T \mathbf{M}_T] \\ \mathbf{\Xi}'_T &= \mathbf{C}_T^T \mathbf{C}_T + \mathbf{\Xi}_T \\ \mathbf{\Omega}'_T &= \mathbf{\Omega}_T + \mathbf{S}'^T \mathbf{S}' \\ &\quad + \mathbf{M}_T^T \mathbf{\Xi}_T \mathbf{M}_T - \mathbf{M}'_T{}^T \mathbf{\Xi}'_T \mathbf{M}'_T \\ \nu'_T &= \nu_T + N \end{aligned} \quad (13)$$

PROOF:

The density of the prior distribution $p(\theta_T)$ is

$$\begin{aligned} p(\theta_T) &= \frac{|\mathbf{\Xi}_T|^{D_S/2}}{\sqrt{(2\pi)^{D_T D_S} |T_\sigma|^{D_T/2}}} \cdot \exp \left[-\frac{1}{2} \text{Tr} (T_\sigma^{-1} (T_\mu - \mathbf{M}_T)^T \mathbf{\Xi}_T (T_\mu - \mathbf{M}_T)) \right] \\ &\cdot \sqrt{\frac{|\mathbf{\Omega}_T|^{\nu_T}}{2^{\nu_T D_S}} \frac{|T_\sigma|^{(1-D_S-\nu_T)/2}}{\Gamma_{D_S}(\nu_T/2)}} \cdot \exp \left[-\frac{1}{2} \text{Tr} (\mathbf{\Omega}_T T_\sigma^{-1}) \right] \\ &\propto |T_\sigma|^{(1-D_S-\nu_T-D_T)/2} \cdot \exp \left\{ -\frac{1}{2} \text{Tr} \left[T_\sigma^{-1} \left[(T_\mu - \mathbf{M}_T)^T \mathbf{\Xi}_T (T_\mu - \mathbf{M}_T) + \mathbf{\Omega}_T \right] \right] \right\} \end{aligned} \quad (19)$$

where from Equation 19 to Equation 20 we treat multiplicative parameters irrelevant to θ_T as constants. The joint density of $p_{\phi_T}(\theta_T, \mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = p(\theta_T) \cdot p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i, \theta_T)$ is hence

$$\begin{aligned} &\frac{|\mathbf{\Xi}_T|^{D_S/2}}{\sqrt{(2\pi)^{D_T D_S} |T_\sigma|^{D_T/2}}} \cdot \exp \left[-\frac{1}{2} \text{Tr} (T_\sigma^{-1} (T_\mu - \mathbf{M}_T)^T \mathbf{\Xi}_T (T_\mu - \mathbf{M}_T)) \right] \\ &\cdot \sqrt{\frac{|\mathbf{\Omega}_T|^{\nu_T}}{2^{\nu_T D_S}} \frac{|T_\sigma|^{(1-D_S-\nu_T)/2}}{\Gamma_{D_S}(\nu_T/2)}} \cdot \exp \left[-\frac{1}{2} \text{Tr} (\mathbf{\Omega}_T T_\sigma^{-1}) \right] \\ &\cdot \frac{1}{\sqrt{(2\pi)^{N D_S} |T_\sigma|^{N/2}}} \cdot \exp \left[-\frac{1}{2} \text{Tr} (T_\sigma^{-1} (\mathbf{S}'_i - \mathbf{C}_T T_\mu)^T (\mathbf{S}'_i - \mathbf{C}_T T_\mu)) \right] \end{aligned} \quad (21)$$

$$\begin{aligned} &\propto |T_\sigma|^{(1-D_S-\nu_T-N-D_T)/2} \cdot \exp \left\{ -\frac{1}{2} \text{Tr} \left[T_\sigma^{-1} \left[(T_\mu - \mathbf{M}_T)^T \mathbf{\Xi}_T (T_\mu - \mathbf{M}_T) \right. \right. \right. \\ &\quad \left. \left. \left. + (\mathbf{S}'_i - \mathbf{C}_T T_\mu)^T (\mathbf{S}'_i - \mathbf{C}_T T_\mu) + \mathbf{\Omega}_T \right] \right] \right\} \end{aligned} \quad (22)$$

Matching the second-order then the first-order term with related to T_μ , we can rewrite

$$22 = |T_\sigma|^{(1-D_S-\nu'_T-D_T)/2} \cdot \exp \left\{ -\frac{1}{2} \text{Tr} \left[T_\sigma^{-1} \left[(T_\mu - \mathbf{M}'_T)^T \mathbf{\Xi}'_T (T_\mu - \mathbf{M}'_T) + \mathbf{\Omega}'_T \right] \right] \right\} \quad (23)$$

We can find Equation 20 and Equation 23 match exactly, indicating the Normal-Wishart-Normal conjugacy. Note, we just use the posterior update of $p(\theta_T)$ as an example. The exact same proof applies to the posterior update of $p(\theta_R)$ as well. Such conjugacy allows exact posterior update and marginal likelihood, enabling efficient learning.

810 A.3 MARGINAL LOG-LIKELIHOOD OF NORMAL-WISHART-NORMAL

811 We prove Equation 14 has the following closed form

$$812 \log p_{\phi_T, \phi_R}(\mathcal{C}) = -\frac{1}{2} \sum_{i=1}^M D_S \log |\Xi'_T| + \nu'_T \log \left| \frac{1}{2} \mathbf{\Omega}'_T \right| + \log |\Xi'_R| + \nu'_R \log \left| \frac{1}{2} \mathbf{\Omega}'_R \right| + \text{const.} \quad (14)$$

813 PROOF:

814 Consider

$$815 p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = \frac{p_{\phi_T}(\theta_T, \mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i)}{p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i)} \quad (24)$$

816 From Appendix A.2, we know the numerator is

$$817 \frac{|\Xi'_T|^{D_S/2} |\mathbf{\Omega}'_T|^{\nu_T/2}}{2^{\nu_T D_S/2} \cdot (2\pi)^{D_S(D_T+N)/2} \cdot \Gamma_{D_S}(\nu_T/2)} \cdot \text{Equation 23} \quad (25)$$

818 The denominator is

$$819 \frac{|\Xi'_T|^{D_S/2} |\mathbf{\Omega}'_T|^{\nu'_T/2}}{2^{\nu'_T D_S/2} \cdot (2\pi)^{D_S D_T/2} \cdot \Gamma_{D_S}(\nu'_T/2)} \cdot \text{Equation 23} \quad (26)$$

820 Hence,

$$821 p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = \frac{1}{(2\pi)^{D_S N/2}} \cdot \frac{|\Xi'_T|^{D_S/2} \left| \frac{1}{2} \mathbf{\Omega}'_T \right|^{\nu_T/2} \cdot \Gamma_{D_S}(\nu'_T/2)}{|\Xi'_T|^{D_S/2} \left| \frac{1}{2} \mathbf{\Omega}'_T \right|^{\nu'_T/2} \cdot \Gamma_{D_S}(\nu_T/2)} \quad (27)$$

822 Similarly,

$$823 p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \frac{1}{(2\pi)^{N/2}} \cdot \frac{|\Xi'_R|^{1/2} \left| \frac{1}{2} \mathbf{\Omega}'_R \right|^{\nu_R/2} \cdot \Gamma(\nu'_R/2)}{|\Xi'_R|^{1/2} \left| \frac{1}{2} \mathbf{\Omega}'_R \right|^{\nu_R/2} \cdot \Gamma(\nu_R/2)} \quad (28)$$

824 Note that

$$825 p_{\phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i}) = p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) p(\mathbf{S}_i, \mathbf{A}_i) \quad (29)$$

826 By taking the logarithm, and note the independence of $p(\mathbf{S}_i, \mathbf{A}_i)$ with related to ϕ_T, ϕ_R ,

$$827 \log p_{\phi_T, \phi_R}(\mathcal{C}^{\mathcal{M}_i}) = \log p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) + \log p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) + \text{const.} \quad (30)$$

828 Sum the above equation on both sides with related to i , then we have Equation 14.

864 A.4 MARGINAL LOG-LIKELIHOOD OF NORMAL-NORMAL

865 We prove Equation 18 has the following form

$$866 \log_{\phi_T, \phi_R}(\mathcal{C}) = -\frac{1}{2} \sum_{i=1}^M D_S \log |\Xi'_T| - \text{Tr}(\Sigma_T^{-1} \mathbf{M}'_T \Xi'_T \mathbf{M}'_T) \\ 867 \\ 868 - \frac{1}{2} \sum_{i=1}^M \log |\Xi'_R| - \text{Tr}(\Sigma_R^{-1} \mathbf{M}'_R \Xi'_R \mathbf{M}'_R) + \text{const.} \quad (18)$$

874 **PROOF:**

875 The distributions without inferring on the noise are listed as follows:

876 **Likelihood:**

$$877 p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i, \theta_T) = \mathcal{MN}(\mathbf{S}'_i | \mathbf{C}_T T_\mu, \mathbf{I}_N, \Sigma_T \in \mathbb{R}^{D_S \times D_S}) \quad (31)$$

$$880 p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \theta_R) = \mathcal{MN}(\mathbf{r}_i | \mathbf{C}_R R_\mu, \mathbf{I}_N, \Sigma_R \in \mathbb{R}^{1 \times 1}) \quad (32)$$

882 **Prior:**

$$883 p(\theta_T) = p(T_\mu) = \mathcal{MN}(T_\mu | \mathbf{M}_T \in \mathbb{R}^{D_T \times D_S}, \Xi_T^{-1} \in \mathbb{R}^{D_T \times D_T}, \Sigma_T) \quad (33)$$

$$884 p(\theta_R) = p(R_\mu) = \mathcal{MN}(R_\mu | \mathbf{M}_R \in \mathbb{R}^{D_R \times 1}, \Xi_R^{-1} \in \mathbb{R}^{D_R \times D_R}, \Sigma_R) \quad (34)$$

886 **Posterior:**

$$887 p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \mathcal{MN}(T_\mu | \mathbf{M}'_T, \Xi_T'^{-1}, \Sigma_T) \quad (35)$$

$$889 p_{\phi_R}(\theta_R | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i, \mathbf{r}) = \mathcal{MN}(R_\mu | \mathbf{M}'_R, \Xi_R'^{-1}, \Sigma_R) \quad (36)$$

890 where

$$891 \mathbf{M}'_T = \Xi_T'^{-1} [\mathbf{C}_T^T \mathbf{S}'_i + \Xi_T \mathbf{M}_T] \quad \mathbf{M}'_R = \Xi_R'^{-1} [\mathbf{C}_R^T \mathbf{r} + \Xi_R \mathbf{M}_R] \\ 892 \Xi_T' = \mathbf{C}_T^T \mathbf{C}_T + \Xi_T \quad \Xi_R' = \mathbf{C}_R^T \mathbf{C}_R + \Xi_R \quad (37)$$

893 Similar to Appendix A.3,

$$894 p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = \frac{p_{\phi_T}(\theta_T, \mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i)}{p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i)} \quad (38)$$

895 As

$$900 p_{\phi_T}(\theta_T, \mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = \frac{|\Xi_T|^{D_S/2} |\Sigma_T|^{-D_T/2}}{\sqrt{(2\pi)^{D_T D_S}}} \cdot \exp \left[-\frac{1}{2} \text{Tr}(\Sigma_T^{-1} (T_\mu - \mathbf{M}_T)^T \Xi_T (T_\mu - \mathbf{M}_T)) \right] \\ 901 \\ 902 \cdot \frac{|\Sigma_T|^{-N/2}}{\sqrt{(2\pi)^{N D_S}}} \cdot \exp \left[-\frac{1}{2} \text{Tr}(\Sigma_T^{-1} (\mathbf{S}'_i - \mathbf{C}_T T_\mu)^T (\mathbf{S}'_i - \mathbf{C}_T T_\mu)) \right] \\ 903 \\ 904 \\ 905 \\ 906 \\ 907 \quad (39)$$

908 And

$$909 p_{\phi_T}(\theta_T | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = \frac{|\Xi_T'|^{D_S/2} |\Sigma_T|^{-D_T/2}}{\sqrt{(2\pi)^{D_T D_S}}} \cdot \exp \left[-\frac{1}{2} \text{Tr}(\Sigma_T^{-1} (T_\mu - \mathbf{M}'_T)^T \Xi_T' (T_\mu - \mathbf{M}'_T)) \right] \\ 910 \\ 911 \\ 912 \quad (40)$$

913 Hence

$$914 p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) \propto |\Xi_T'|^{-D_S/2} \cdot \exp \left[-\frac{1}{2} \text{Tr}(-\mathbf{M}'_T \Xi_T' \mathbf{M}'_T) \right] \quad (41)$$

$$915 \log p_{\phi_T}(\mathbf{S}'_i | \mathbf{S}_i, \mathbf{A}_i) = -\frac{D_S}{2} \log |\Xi_T'| + \frac{1}{2} \text{Tr}(\mathbf{M}'_T \Xi_T' \mathbf{M}'_T) + \text{const.} \quad (42)$$

918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971

Similarly,

$$\log p_{\phi_R}(\mathbf{r}_i | \mathbf{S}_i, \mathbf{A}_i, \mathbf{S}'_i) = -\frac{1}{2} |\boldsymbol{\Xi}'_R| + \frac{1}{2} \text{Tr}(\mathbf{M}'_R \boldsymbol{\Xi}'_R \mathbf{M}'_R) + \text{const.} \quad (43)$$

Following Appendix A.3,

$$\begin{aligned} \log_{\phi_T, \phi_R}(\mathcal{C}) = & -\frac{1}{2} \sum_{i=1}^M D_S \log |\boldsymbol{\Xi}'_T| - \text{Tr}(\boldsymbol{\Sigma}_T^{-1} \mathbf{M}'_T \boldsymbol{\Xi}'_T \mathbf{M}'_T) \\ & - \frac{1}{2} \sum_{i=1}^M \log |\boldsymbol{\Xi}'_R| - \text{Tr}(\boldsymbol{\Sigma}_R^{-1} \mathbf{M}'_R \boldsymbol{\Xi}'_R \mathbf{M}'_R) + \text{const.} \quad (18) \end{aligned}$$

A.5 ML10 SUCCESS RATE COMPARISONS PER-TASK

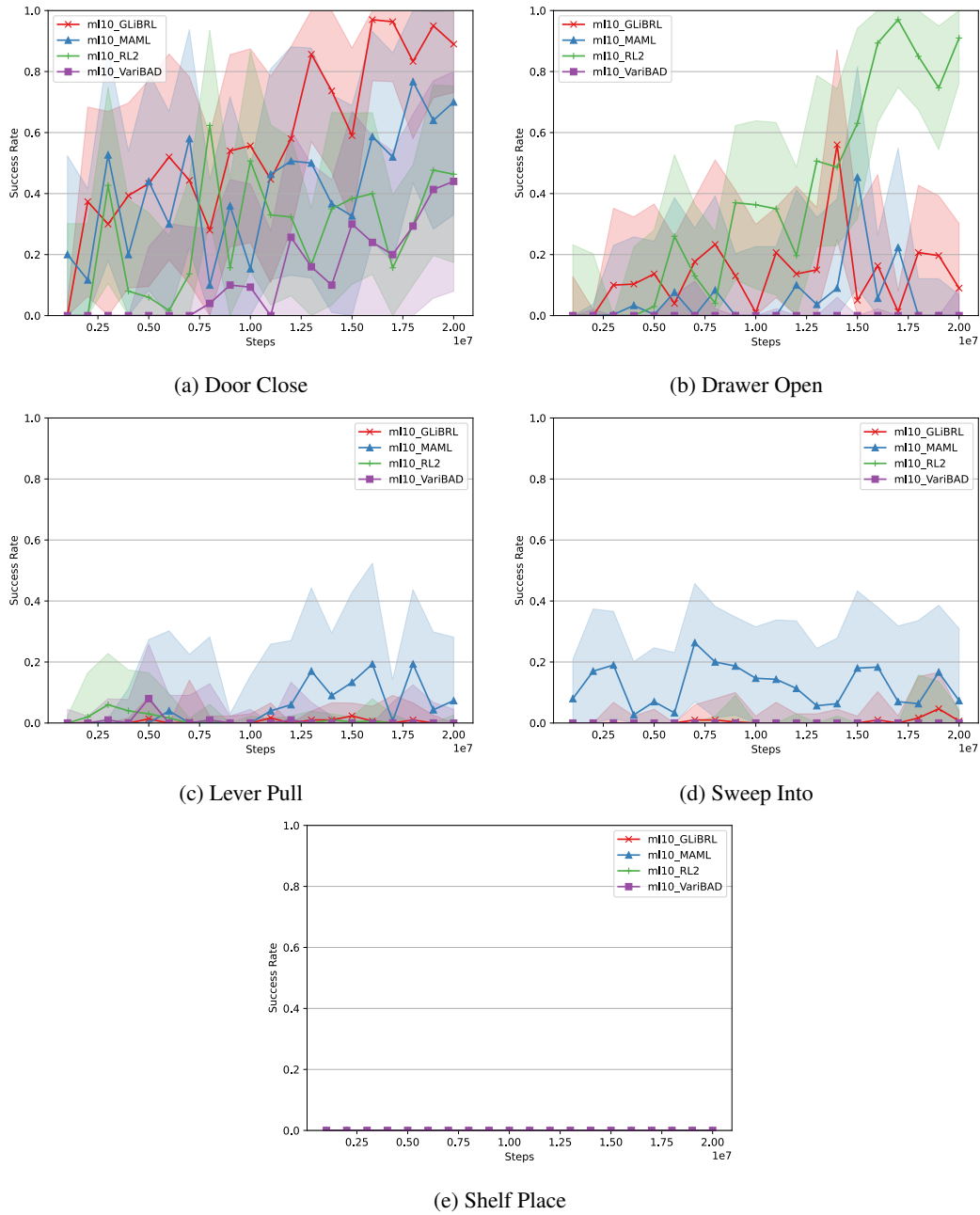
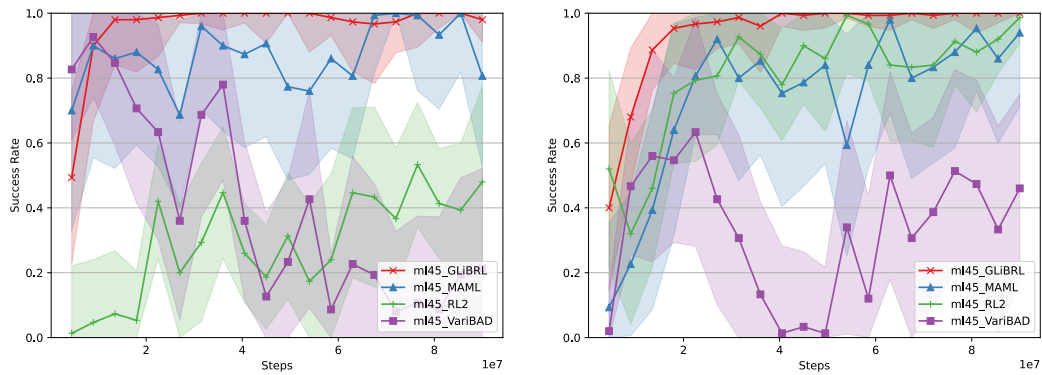


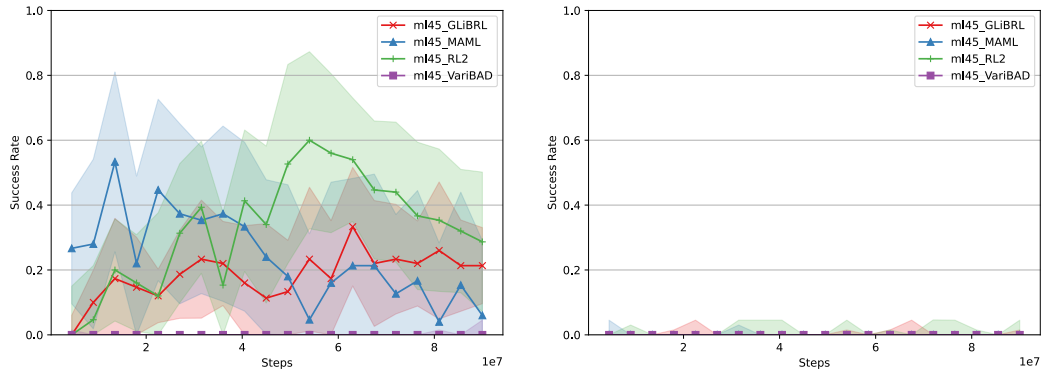
Figure 3: IQM and 95% CI of success rate for each testing scenario in ML10. The Shelf Place scenario is challenging as none of the method can achieve a single success.

1026 A.6 ML45 SUCCESS RATE COMPARISONS PER-TASK
 1027
 1028



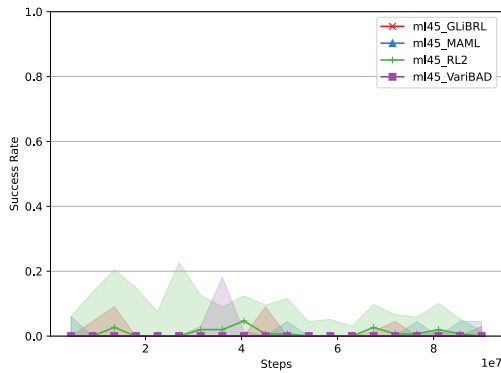
(a) Door Lock

(b) Door Unlock



(c) Hand Insert

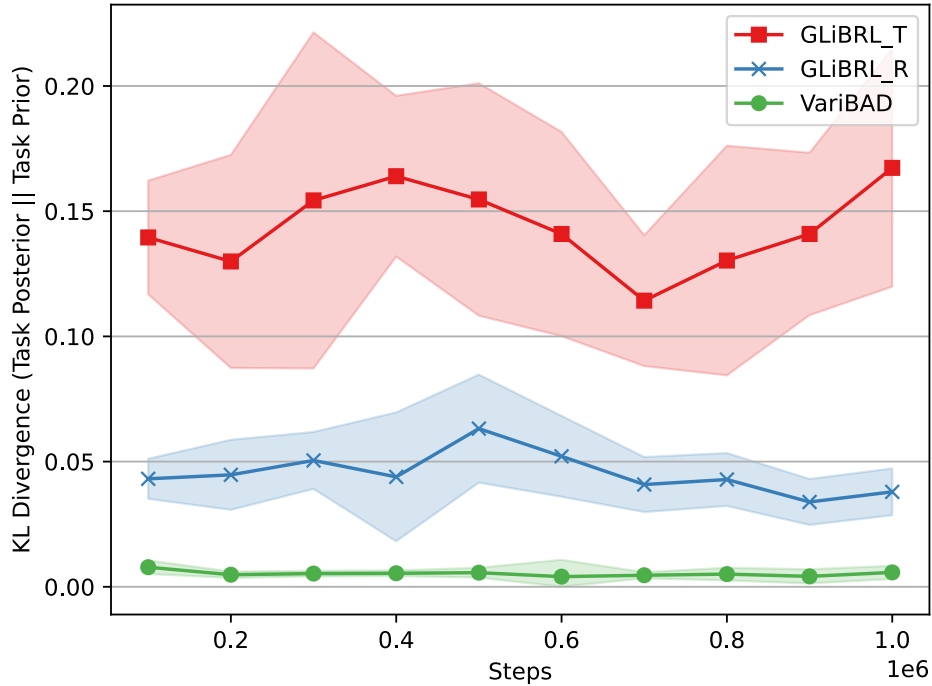
(d) Bin Picking



(e) Box Close

1070 Figure 4: IQM and 95% CI of success rate for each testing scenario in ML45. GLiBRL achieves
 1071 nearly 100% testing success rates in both Door Lock and Door Unlock scenarios.
 1072
 1073
 1074
 1075
 1076
 1077
 1078
 1079

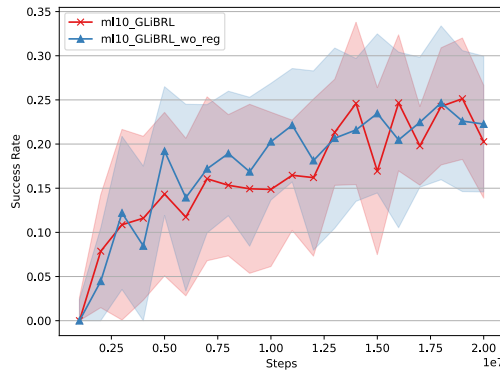
A.7 POSTERIOR COLLAPSE IN VARIBAD



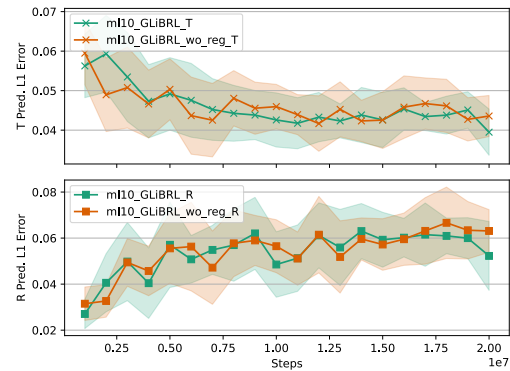
To verify if VariBAD can learn meaningful task representations, we check the IQM and 95% CI of the expected KL-divergence between task posteriors and task priors $\mathbb{E}[D_{KL}(q(\theta_T, \theta_R | \mathcal{C}_{t+1}) || q(\theta_T, \theta_R | \mathcal{C}_t))]$, out of 10 runs in the ML10 benchmark, where $\mathcal{C}_{t+1} = \mathcal{C}_t \cup c_{t+1}$ updates the set of contexts \mathcal{C}_t with the context c_{t+1} at time step $t + 1$. VariBAD uses a single latent variable to model both transitions and rewards, hence contributing to only one line in the above figure.

Intuitively, if the expected divergence is close to 0, the majority of posterior updates has collapsed to priors, meaning barely any meaningful task representation has been learnt. Clearly from the above figure, VariBAD fails to learn meaningful representations, while GLiBRL demonstrates obvious divergence between posteriors and priors.

A.8 REGULARISATION IN THE MARGINAL LOG-LIKELIHOOD



(a) Success Rate

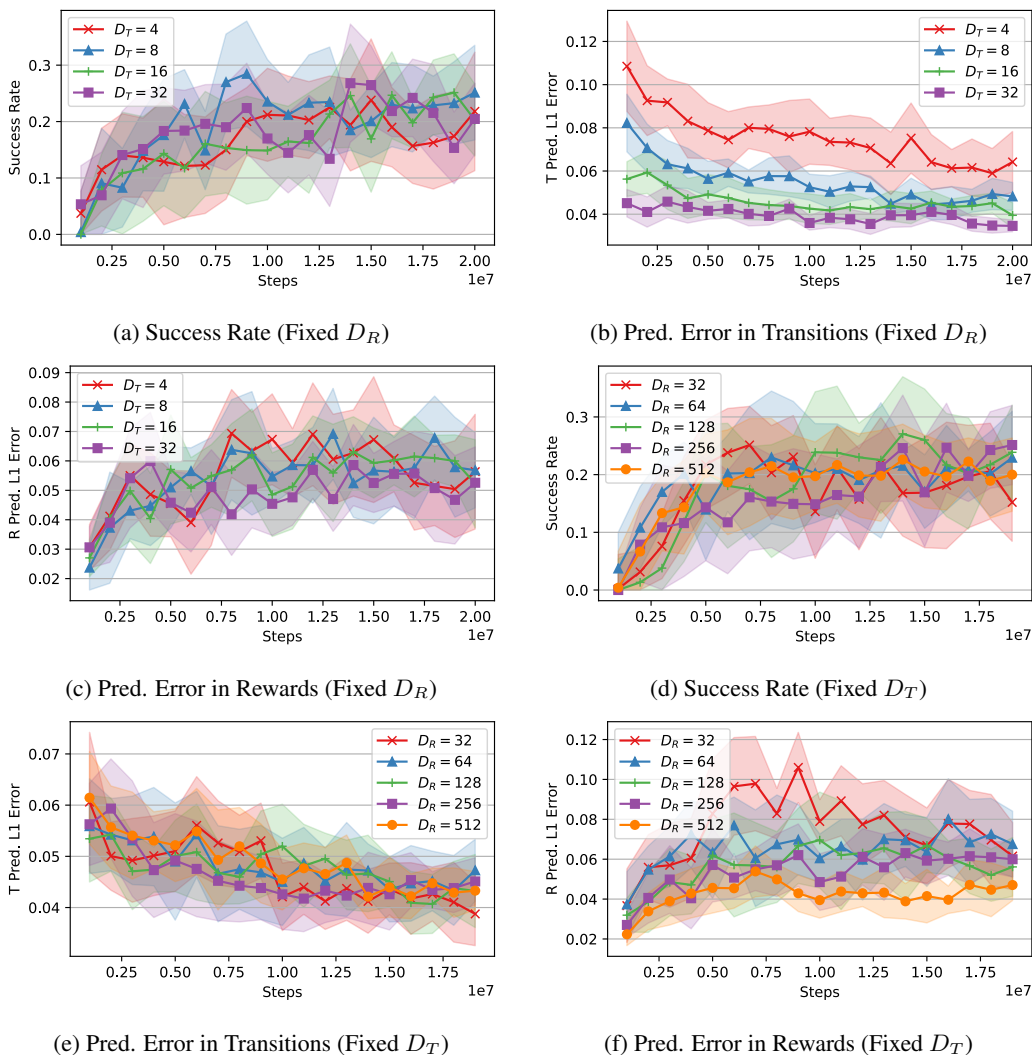


(b) Prediction Error

Running GLiBRL and its variant without regularisation in the ML10 benchmark, we can tell the regularisations do not change the overall trend in both success rate and prediction error. However, as the number of training step increases, GLiBRL begins to show in (a): higher success rates with narrower CI; in (b): lower prediction error in both transitions and rewards.

A.9 SENSITIVITY OF LATENT TASK DIMENSIONS

To demonstrate the sensitivity of GLiBRL with related to latent task dimensions D_T and D_R , we perform hyperparameter search in ML10 on (1) $D_T = \{4, 8, 16, 32\}$ while fixing $D_R = 256$ and (2) $D_R = \{32, 64, 128, 256, 512\}$ while fixing $D_T = 16$. We compare on (1) the success rate, (2) the predictive error in transitions, and (3) the predictive error in rewards.



In general, GLiBRL shows stable success rates with related to latent task dimensions D_T , D_R , which can be inferred from Figure (a) and (d). We have observed that GLiBRL achieves state-of-the-art performance on ML10 (29% success rate) with $D_T = 8$, with the cost of predictive errors. Figure (b) and (f) indicate that as D_T, D_R grows, the corresponding error reduces, offering an efficiency-accuracy trade-off. Figure (c) and (e) are sanity checks that have confirmed transitions do not affect rewards, and vice versa. Overall, D_R are set to be much larger than D_T , as reward functions are generally much harder to learn, compared to transition function.

A.10 HYPERPARAMETERS, RUNTIME AND MEMORY

We list all hyperparameters of GLiBRL in the following table. We use the same hyperparameters for both ML10 and ML45.

Name	Value	Name	Value
policy_learner	PPO	feat_out_activation	True
policy_layers	[256, 256]	t_mix_layers	[64, 32]
a_feat_out_activation	True	t_mix_layernorm	True
policy_activation	Tanh	t_mix_out_activation	False
policy_optimiser	Adam	r_mix_layers	[128, 64]
policy_lr	5e-4	r_mix_layernorm	True
policy_opt_max_norm	1	r_mix_out_activation	False
policy_weight_init	Xavier	t_reg_coef	5e-3
policy_bias_init	0	r_reg_coef	1e-3
policy_log_std_min	1e-6	model_activation	ReLU
policy_log_std_max	2	model_optimiser	Adam
policy_grad_epochs	10	model_lr	2e-4
policy_grad_steps	20	model_opt_max_norm	None
ppo_clip_eps	0.5	model_grad_epochs	1
ppo_gamma	0.99	model_grad_steps	20
ppo_gae_lambda	0.95	init_mt	zeros
ppo_entropy_coef	5e-3	init_mr	zeros
s_feat_layers	[64, 32]	init_xit	ones
s_feat_outdim	32	init_xir	ones
s_feat_layernorm	False	init_omegat	ones
a_feat_layers	[32, 16]	init_omegar	ones
a_feat_outdim	16	init_nut	40
a_feat_layernorm	False	init_nur	2

GLiBRL is rather efficient in both time and memory. Although all of our experiments are run using A100, we have tested that GLiBRL can run fast on much lower-end GPUs with 8GB memory, such as RTX 3070, with each run costing less than 2 hours. The runtime does not vary too much with changes in D_T and D_R , due to the quadratic online inference complexity.