Transfer of Value Functions via Variational Methods

Andrea Tirinzoni* Politecnico di Milano andrea.tirinzoni@polimi.it

Rafael Rodriguez Sanchez* Politecnico di Milano rafaelalberto.rodriguez@polimi.it

Marcello Restelli Politecnico di Milano marcello.restelli@polimi.it

Abstract

We consider the problem of transferring value functions in reinforcement learning. We propose an approach that uses the given source tasks to learn a prior distribution over optimal value functions and provide an efficient variational approximation of the corresponding posterior in a new target task. We show our approach to be general, in the sense that it can be combined with complex parametric function approximators and distribution models, while providing two practical algorithms based on Gaussians and Gaussian mixtures. We theoretically analyze them by deriving a finite-sample analysis and provide a comprehensive empirical evaluation in four different domains.

1 Introduction

Recent advancements have allowed reinforcement learning (RL) [32] to achieve impressive results in a wide variety of complex tasks, ranging from Atari [25] through the game of Go [31] to the control of sophisticated robotics systems [16, 23, 22]. The main limitation is that these RL algorithms still require an enormous amount of experience samples before successfully learning such complicated tasks. One of the most promising solutions to alleviate this problem is transfer learning, which focuses on reusing past knowledge available to the agent in order to reduce the sample-complexity for learning new tasks. In the typical settings of transfer in RL [34], the agent is assumed to have already solved a set of *source tasks* generated from some unknown distribution. Then, given a *target task* (which is drawn from the same distribution, or a slightly different one), the agent can rely on knowledge from the source tasks to speed up the learning process. This reuse of knowledge constitutes a significant advantage over plain RL, where the agent learns each new task from scratch independently of any previous learning experience. Several algorithms have been proposed in the literature to transfer different elements involved in the learning process: experience samples [21, 33, 35], policies/options [10, 18], rewards [17], features [5], parameters [9, 15, 11]. We refer the reader to [34, 19] for a thorough survey on transfer in RL.

Under the assumption that tasks follow a specific distribution, an intuitive choice for designing a transfer algorithm is to attempt to characterize the uncertainty over the target task. Then, an ideal algorithm would leverage prior knowledge from the source tasks to interact with the target task to reduce the uncertainty as quickly as possible. This simple intuition makes Bayesian methods appealing approaches for transfer in RL, and many previous works have been proposed in this direction. In [37], the authors assume tasks share similarities in their dynamics and rewards and propose a hierarchical Bayesian model for the distribution of these two elements. Similarly, in [20], the authors assume that tasks are similar in their value functions and design a different hierarchical Bayesian model for transferring such information. More recently, [9], and its extension [15], consider tasks whose dynamics are governed by some hidden parameters, and propose efficient Bayesian models for quickly learning such parameters in new tasks. However, most of these algorithms require specific, and sometimes restrictive, assumptions (e.g., on the distributions involved or the function approximators adopted), which might limit their practical applicability. The importance of having

*Equal contribution

transfer algorithms that alleviate the need for strong assumptions and that easily adapt to different contexts motivates us to take a more general approach.

Similarly to [20], we assume tasks to share similarities in their value functions and use the given source tasks to learn a distribution over such functions. Then, we use this distribution as a prior for learning the target task and we propose a variational approximation of the corresponding posterior that is computationally efficient. Leveraging recent ideas from randomized value functions [26, 3], we design a Thompson Sampling-based algorithm which efficiently explores the target task by repeatedly sampling from the posterior and acting greedily w.r.t. (with respect to) the sampled value function. We show that our approach is very general, in the sense that it can work with any parametric function approximator and with any prior/posterior distribution models (in this paper we focus on the Gaussian and Gaussian mixture models). In addition to the algorithmic contribution, we also give a theoretical contribution by providing a finite-sample analysis of our approach and an experimental contribution showing its empirical performance on four domains with increasing level of difficulty.

2 Preliminaries

We consider a distribution $\mathcal{D}$ over tasks, where each task $\mathcal{M}_\tau$ is modeled as a discounted Markov Decision Process (MDP). We define an MDP as a tuple $\mathcal{M}_\tau = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}_\tau, \mathcal{R}_\tau, p_0, \gamma \rangle$, where $\mathcal{S}$ is the state-space, $\mathcal{A}$ is a finite set of actions, $\mathcal{P}_\tau(\cdot|s,a)$ is the distribution of the next state s' given that action a is taken in state s, $\mathcal{R}_\tau : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, p_0 is the initial-state distribution, and $\gamma \in [0, 1)$ is the discount factor. We assume the reward function to be uniformly bounded by a constant $R_{max} > 0$. A deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ is a mapping from states to actions. At the beginning of each episode of interaction, the initial state s_0 is drawn from p_0. Then, the agent takes the action $a_0 = \pi(s_0)$, receives a reward $\mathcal{R}_\tau(s_0, a_0)$, transitions to the next state $s_1 \sim \mathcal{P}_\tau(\cdot|s_0, a_0)$, and the process is repeated. The goal is to find the policy maximizing the long-term return over a possibly infinite horizon: $\max_\pi J(\pi) \triangleq \mathbb{E}_{\mathcal{M}_\tau,\pi}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_\tau(s_t, a_t)]$. To this end, we define the optimal value function of task $\mathcal{M}_\tau$, $Q^*_\tau(s, a)$, as the expected return obtained by taking action a in state s and following an optimal policy thereafter. Then, an optimal policy π^*_τ is a policy that is greedy with respect to the optimal value function, i.e., $\pi^*_\tau(s) = \mathrm{argmax}_a Q^*_\tau(s, a)$ for all states s. It can be shown (e.g., [27]) that Q^*_τ is the unique fixed-point of the optimal Bellman operator T_τ defined by $T_\tau Q(s, a) = \mathcal{R}_\tau(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}_\tau}[\max_{a'} Q(s', a')]$ for any value function Q. From now on, we adopt the term Q-function to denote any plausible value function, i.e., any function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ uniformly bounded by $\frac{R_{max}}{1-\gamma}$. In the following, to avoid cluttering the notation, we will drop the subscript τ whenever there is no ambiguity.

We consider a parametric family of Q-functions, $\mathcal{Q} = \left\{Q_{\boldsymbol{w}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \mid \boldsymbol{w} \in \mathbb{R}^d\right\}$, and we assume each function in $\mathcal{Q}$ to be uniformly bounded by $\frac{R_{max}}{1-\gamma}$. When learning the optimal value function, a quantity of interest is how close a given function $Q_{\boldsymbol{w}}$ is to the fixed-point of the Bellman operator. A possible measure is its Bellman error (or Bellman residual), defined by $B_{\boldsymbol{w}} \triangleq TQ_{\boldsymbol{w}} - Q_{\boldsymbol{w}}$. Notice that $Q_{\boldsymbol{w}}$ is optimal if and only if $B_{\boldsymbol{w}}(s, a) = 0$ for all s, a. If we assume the existence of a distribution ν over $\mathcal{S} \times \mathcal{A}$, a sound objective is to directly minimize the squared Bellman error of $Q_{\boldsymbol{w}}$ under ν, denoted by $\|B_{\boldsymbol{w}}\|^2_\nu$. Unfortunately, it is well-known that an unbiased estimator of this quantity requires two independent samples of the next state s' for each s, a (e.g., [24]). In practice, the Bellman error is typically replaced by the TD error $b(\boldsymbol{w})$, which approximates the former using a single transition sample $\langle s, a, s', r \rangle$, $b(\boldsymbol{w}) = r + \gamma \max_{a'} Q_{\boldsymbol{w}}(s', a') - Q_{\boldsymbol{w}}(s, a)$. Finally, given a dataset $D = \langle s_i, a_i, r_i, s'_i \rangle_{i=1}^N$ of N samples, the squared TD error is computed as $\|B_{\boldsymbol{w}}\|^2_D = \frac{1}{N} \sum_{i=1}^N (r_i + \gamma \max_{a'} Q_{\boldsymbol{w}}(s'_i, a') - Q_{\boldsymbol{w}}(s_i, a_i))^2 = \frac{1}{N} \sum_{i=1}^N b_i(\boldsymbol{w})^2$. Whenever the distinction is clear from the context, with a slight abuse of terminology, we refer to the squared Bellman error and squared TD error as Bellman error and TD error, respectively.

3 Variational Transfer Learning

In this section, we describe our variational approach to transfer in RL. In Section 3.1, we start by introducing our algorithm from a high-level perspective, in such a way that any choice of prior and posterior distributions is possible. Then, in Sections 3.2 and 3.3, we propose practical implementations based on Gaussians and mixtures of Gaussians, respectively. We conclude with some considerations on how to optimize the proposed objective in Section 3.4.

3.1 Algorithm

Let us observe that the distribution $\mathcal{D}$ over tasks induces a distribution over optimal Q-functions. Furthermore, for any MDP, learning its optimal Q-function is sufficient for solving the problem. Thus, we can safely replace the distribution over tasks with the distribution over their optimal value functions. In our parametric settings, we reduce the latter to a distribution $p(\boldsymbol{w})$ over weights.
Assume, for the moment, that we know the distribution $p(\boldsymbol{w})$ and consider a dataset $D = \langle s_i, a_i, r_i, s_i' \rangle_{i=1}^N$ of samples from some task $\mathcal{M}_\tau \sim \mathcal{D}$ that we want to solve. Then, we can compute the posterior distribution over weights given such dataset by applying Bayes theorem as $p(\boldsymbol{w}|D) \propto p(D|\boldsymbol{w})p(\boldsymbol{w})$. Unfortunately, this cannot be directly used in practice since we do not have a model of the likelihood $p(D|\boldsymbol{w})$. In such case, it is very common to make strong assumptions on the MDPs or the Q-functions to get tractable posteriors. However, in our transfer settings, all distributions involved depend on the family of tasks under consideration and making such assumptions is likely to limit the applicability to specific problems. Thus, we take a different approach to derive a more general, but still well-grounded, solution. Notice that our final goal is to move the total probability mass over the weights minimizing some empirical loss measure, which in our case is the TD error $\|B_{\boldsymbol{w}}\|_D^2$. Then, given a prior $p(\boldsymbol{w})$, we know from PAC-Bayesian theory that the optimal Gibbs posterior q minimizing an oracle upper bound on the expected loss takes the form (e.g., [8]):

$$q(\boldsymbol{w}) = \frac{e^{-\Lambda \|B_{\boldsymbol{w}}\|_D^2} p(\boldsymbol{w})}{\int e^{-\Lambda \|B_{\boldsymbol{w}'}\|_D^2} p(d\boldsymbol{w}')}, \tag{1}$$

for some parameter $\Lambda > 0$. Since Λ is typically chosen to increase with the number of samples N, in the remaining, we set it to $\lambda^{-1}N$, for some constant $\lambda > 0$. Notice that, whenever the term $e^{-\Lambda \|B_{\boldsymbol{w}}\|_D^2}$ can be interpreted as the actual likelihood of D, q becomes a classic Bayesian posterior. Although we now have an appealing distribution, the integral at the denominator of (1) is intractable to compute even for simple Q-function models. Thus, we propose a variational approximation $q_{\boldsymbol{\xi}}$ by considering a simpler family of distributions parameterized by $\boldsymbol{\xi} \in \Xi$. Then, our problem reduces to finding the variational parameters $\boldsymbol{\xi}$ such that $q_{\boldsymbol{\xi}}$ minimizes the Kullback-Leibler (KL) divergence w.r.t. the Gibbs posterior q. From the theory of variational inference (e.g., [6]), this can be shown to be equivalent to minimizing the well-known (negative) *evidence lower bound* (ELBO):

$$\min_{\boldsymbol{\xi} \in \Xi} \mathcal{L}(\boldsymbol{\xi}) = \mathbb{E}_{\boldsymbol{w} \sim q_{\boldsymbol{\xi}}} \left[\|B_{\boldsymbol{w}}\|_D^2 \right] + \frac{\lambda}{N} KL\left(q_{\boldsymbol{\xi}}(\boldsymbol{w}) \mid\mid p(\boldsymbol{w}) \right). \tag{2}$$

Intuitively, the approximate posterior balances between placing probability mass over those weights $\boldsymbol{w}$ that have low expected TD error (first term), and staying close to the prior distribution (second term). Assuming that we can compute the gradients of (2) w.r.t. the variational parameters $\boldsymbol{\xi}$, our objective can be optimized using any stochastic optimization algorithm, as shown in the next subsections.

We now highlight our general transfer procedure in Algorithm 1, while deferring a description of specific choices for the involved distributions to the next two subsections. Although the distribution $p(\boldsymbol{w})$ is not known in practice, we assume that the agent has solved a *finite* number of source tasks $\mathcal{M}_{\tau_1}, \mathcal{M}_{\tau_2}, \dots, \mathcal{M}_{\tau_M}$ and that we are given the set of their approximate solutions: $\mathcal{W}_s = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_M\}$ such that $Q_{\boldsymbol{w}_j} \simeq Q^*_{\tau_j}$. Using these weights, we start by estimating the prior distribution (line 1), and we initialize the variational parameters by minimizing the KL divergence w.r.t. such distribution (line 2).[2] Then, at each time step of interaction, we re-sample the weights from the current approximate posterior and act greedily w.r.t. the corresponding Q-function (lines 7,8). After collecting and storing the new experience (lines 9-10), we estimate the objective function gradient using a mini-batch of samples from the current dataset (line 11), and update the variational parameters (line 12).

Algorithm 1 Variational Transfer

```
Require: Target task $\mathcal{M}_\tau$, source weights $\mathcal{W}_s$
 1: Estimate prior $p(\boldsymbol{w})$ from $\mathcal{W}_s$
 2: Initialize parameters: $\boldsymbol{\xi} \leftarrow \operatorname{argmin}_{\boldsymbol{\xi}} KL(q_{\boldsymbol{\xi}}||p)$
 3: Initialize dataset: $D = \emptyset$
 4: repeat
 5:     Sample initial state: $s_0 \sim p_0$
 6:     while $s_h$ is not terminal do
 7:         Sample weights: $\boldsymbol{w} \sim q_{\boldsymbol{\xi}}(\boldsymbol{w})$
 8:         Take action $a_h = \operatorname{argmax}_a Q_{\boldsymbol{w}}(s_h, a)$
 9:         $s_{h+1} \sim \mathcal{P}_\tau(\cdot|s_h, a_h)$, $r_{h+1} = \mathcal{R}_\tau(s_h, a_h)$
10:         $D \leftarrow D \cup \langle s_h, a_h, r_{h+1}, s_{h+1} \rangle$
11:         Estimate gradient $\nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi})$ using $D' \subseteq D$
12:         Update $\boldsymbol{\xi}$ from $\nabla_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{\xi})$ using any optimizer
13:     end while
14: until forever
```

[2]If the prior and approximate posterior were in the same family of distributions we could simply set $\boldsymbol{\xi}$ to the prior parameters. However, we are not making this assumption at this point.

The key property of our approach is the weight resampling at line 7, which resembles the well-known Thompson sampling approach adopted in multi-armed bandits [7] and closely relates to the recent value function randomization [26, 3]. At each time we guess what is the task we are trying to solve based on our current belief and we act as if such guess were true. This mechanism allows an efficient adaptive exploration of the target task. Intuitively, during the first steps of interaction, the agent is very uncertain about the current task, and such uncertainty induces stochasticity in the chosen actions, allowing a rather informed exploration to take place. Consider, for instance, that actions that are bad on average for all tasks are improbable to be sampled, while this cannot happen in uninformed exploration strategies, like ϵ-greedy, before learning takes place. As the learning process goes on, the algorithm will quickly figure out which task it is solving, thus moving all the probability mass over the weights minimizing the TD error. From that point, sampling from the posterior is approximately equivalent to deterministically taking such weights, and no more exploration will be performed. Finally, notice the generality of the proposed approach: as far as the objective $\mathcal{L}$ is differentiable in the variational parameters $\boldsymbol{\xi}$, and its gradients can be efficiently computed, any approximator for the Q-function and any prior/posterior distributions can be adopted. For the latter, we describe two practical choices in the next two sections.

3.2 Gaussian Variational Transfer

We now restrict to a specific choice of the prior and posterior families that makes our algorithm very efficient and easy to implement. We assume that optimal Q-functions (or better, their weights) follow a multivariate Gaussian distribution. That is, we model the prior as $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ and we learn its parameters from the set of source weights using maximum likelihood estimation (with small regularization to make sure the covariance is positive definite). Then, our variational family is the set of all well-defined Gaussian distributions, i.e., the variational parameters are $\Xi = \left\{(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \mid \boldsymbol{\mu} \in \mathbb{R}^d, \boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}, \boldsymbol{\Sigma} \succ 0\right\}$. To prevent the covariance from becoming not positive definite, we consider its Cholesky decomposition $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$ and we learn the lower-triangular Cholesky factor $\boldsymbol{L}$ instead. In this case, deriving the gradient of the objective is very simple. Both the KL between two multivariate Gaussians and its gradients have a simple closed-form expression. The expected log-likelihood, on the other hand, can be easily differentiated by adopting the reparameterization trick (e.g., [14, 28]). We report these results in Appendix B.1.

3.3 Mixture of Gaussian Variational Transfer

Although the Gaussian assumption of the previous section is very appealing as it allows for a simple and efficient way of computing the variational objective and its gradients, in practice it rarely allows us to describe the prior distribution accurately. In fact, even for families of tasks in which the reward and transition models are Gaussian, the Q-values might be far from being normally distributed. Depending on the family of tasks under consideration and, since we are learning a distribution over weights, on the chosen function approximator, the prior might have arbitrarily complex shapes. When the information loss due to the Gaussian approximation becomes too severe, the algorithm is likely to fail at capturing any similarities between the tasks. We now propose a variant to successfully solve this problem, while keeping the algorithm efficient and simple enough to be applied in practice.

Given the source tasks' weights $\mathcal{W}_s$, we model our estimated prior as a mixture with equally weighted isotropic Gaussians centered at each weight: $p(\boldsymbol{w}) = \frac{1}{|\mathcal{W}_s|}\sum_{\boldsymbol{w}_s \in \mathcal{W}_s} \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_s, \sigma_p^2 \boldsymbol{I})$. This model resembles a kernel density estimator [30] with bandwidth σ_p^2 and, due to its nonparametric nature, it allows capturing arbitrarily complex distributions. Consistently with the prior, we model our approximate posterior as a mixture of Gaussians. Using C components, our posterior is $q_{\boldsymbol{\xi}}(\boldsymbol{w}) = \frac{1}{C}\sum_{i=1}^{C} \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, with variational parameters $\boldsymbol{\xi} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_C, \boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_C)$. Once again, we learn Cholesky factors instead of full covariances. Finally, since the KL divergence between two mixtures of Gaussians has no closed-form expression, we rely on an upper bound to such quantity, so that the negative ELBO still upper bounds the KL between the approximate and the exact posterior. Among the many upper bounds available, we adopt the one proposed in [13] (see Appendix B.2).

3.4 Minimizing the TD Error

From Sections 3.2 and 3.3, we know that differentiating the negative ELBO $\mathcal{L}$ w.r.t. $\boldsymbol{\xi}$ requires differentiating $\|B_{\boldsymbol{w}}\|_D^2$ w.r.t. $\boldsymbol{w}$. Unfortunately, the TD error is well-known to be non-differentiable due to the presence of the max operator. This issue is rarely a problem since typical value-based algorithms are semi-gradient methods, i.e., they do not differentiate the targets (see, e.g., Chapter 11 of [32]). However, our transfer settings are quite different from common RL. In fact, our algorithm is likely to start from Q-functions that are very close to an optimum and aims only to adapt the weights in some direction of lower error so as to quickly converge to the solution of the target task. Unfortunately, this property does not hold for most semi-gradient algorithms. Even worse, many online RL algorithms combined with complex function approximators (e.g., DQNs) are well-known to be unstable, especially when approaching an optimum, and require many tricks and tuning to work well [29, 36]. This property is clearly undesirable in our case, as we only aim at adapting already good solutions. Thus, we consider using a residual gradient algorithm [4]. To differentiate the targets, we replace the optimal Bellman operator with the mellow Bellman operator introduced in [2], which adopts a softened version of max called *mellowmax*:

$$\operatorname*{mm}_{a} Q_{\boldsymbol{w}}(s,a) = \frac{1}{\kappa} \log \frac{1}{|\mathcal{A}|} \sum_{a} e^{\kappa Q_{\boldsymbol{w}}(s,a)} \tag{3}$$

where κ is a hyperparameter and $|\mathcal{A}|$ is the number of actions. The mellow Bellman operator, which we denote as $\widetilde{T}$, has several appealing properties: (i) it converges to the maximum as $\kappa \to \infty$, (ii) it has a unique fixed-point, and (iii) it is *differentiable*. Denoting by $\widetilde{B}_{\boldsymbol{w}} = \widetilde{T} Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}$ the Bellman residual w.r.t. the mellow Bellman operator $\widetilde{T}$, we have that the corresponding TD error, $||\widetilde{B}_{\boldsymbol{w}}||_D^2$, is now differentiable w.r.t. $\boldsymbol{w}$.

Although residual algorithms have guaranteed convergence, they are typically much slower than their semi-gradient counterpart. [4] proposed to project the gradient in a direction that achieves higher learning speed, while preserving convergence. This projection is obtained by including a parameter $\psi \in [0,1]$ in the TD error gradient:

$$\nabla_{\boldsymbol{w}} \left\| \widetilde{B}_{\boldsymbol{w}} \right\|_D^2 = \frac{2}{N} \sum_{i=1}^{N} \widetilde{b}_i(\boldsymbol{w}) \Big(\gamma \psi \nabla_{\boldsymbol{w}} \operatorname*{mm}_{a'} Q_{\boldsymbol{w}}(s_i', a') - \nabla_{\boldsymbol{w}} Q_{\boldsymbol{w}}(s_i, a_i) \Big),$$

where $\widetilde{b}_i(\boldsymbol{w}) = r_i + \gamma \operatorname{mm}_{a'} Q_{\boldsymbol{w}}(s_i', a') - Q_{\boldsymbol{w}}(s_i, a_i)$. Notice that ψ trades-off between the semi-gradient ($\psi = 0$) and the full residual gradient ($\psi = 1$). A good criterion for choosing such parameter is to start with values close to zero (to have faster learning) and move to higher values when approaching the optimum (to guarantee convergence).

4 Theoretical Analysis

A first important question that we need to answer is whether replacing max with mellow-max in the Bellman operator constitutes a strong approximation or not. It has been proven [2] that the mellow Bellman operator is a non-expansion under the L_∞-norm and, thus, has a unique fixed-point. However, how such fixed-point differs from the one of the optimal Bellman operator remains an open question. Since mellow-max monotonically converges to max as $\kappa \to \infty$, it would be desirable if the fixed point of the corresponding operator also monotonically converged to the fixed point of the optimal one. We confirm that this property actually holds in the following theorem.

Theorem 1. *Let Q^* be the fixed-point of the optimal Bellman operator T. Define the action-gap function $g(s)$ as the difference between the value of the best action and the second best action at each state s. Let $\widetilde{Q}$ be the fixed-point of the mellow Bellman operator $\widetilde{T}$ with parameter $\kappa > 0$ and denote by $\beta_\kappa > 0$ the inverse temperature of the induced Boltzmann distribution (as in [2]). Then:*

$$\left\| Q^* - \widetilde{Q} \right\|_\infty \leq \frac{2\gamma R_{max}}{(1-\gamma)^2} \left\| \frac{1}{1 + \frac{1}{|\mathcal{A}|} e^{\beta_\kappa g}} \right\|_\infty . \tag{4}$$

The proof is provided in Appendix A.1. Notice that $\widetilde{Q}$ converges to Q^* exponentially fast as κ (equivalently, β_κ) increases and the action gaps are all larger than zero. Notice that this result is of interest even outside our specific settings.

The second question that we need to answer is whether we can provide any guarantee on our algorithm's performance when given limited data. To address this point, we consider the two variants of Algorithm 1 from Section 3.2 and 3.3 with linear approximators. Specifically, we consider the family of linearly parameterized value functions $Q_{\boldsymbol{w}}(s,a) = \boldsymbol{w}^T \boldsymbol{\phi}(s,a)$ with bounded weights $\|\boldsymbol{w}\|_2 \leq w_{max}$ and uniformly bounded features $\|\boldsymbol{\phi}(s,a)\|_2 \leq \phi_{\max}$. We assume only a finite dataset is available and provide a finite-sample analysis bounding the expected (mellow) Bellman error under the variational distribution minimizing the objective (2) for any fixed target task $\mathcal{M}_\tau$.

Theorem 2. *Let $\widehat{\boldsymbol{\xi}}$ be the variational parameters minimizing the objective of Eq.* (2) *on a dataset D of N i.i.d. samples distributed according to $\mathcal{M}_\tau$ and ν. Let $\boldsymbol{w}^* = \operatorname{arginf}_{\boldsymbol{w}} \|\widetilde{B}_{\boldsymbol{w}}\|_\nu^2$ and define $\upsilon(\boldsymbol{w}^*) \triangleq \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*, \frac{1}{N}\boldsymbol{I})}\left[v(\boldsymbol{w})\right]$, with $v(\boldsymbol{w}) \triangleq \mathbb{E}_\nu\left[Var_{\mathcal{P}_\tau}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Then, there exist constants c_1, c_2, c_3 such that, with probability at least $1-\delta$ over the choice of the dataset D:*

$$\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \leq 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + \upsilon(\boldsymbol{w}^*) + c_1\sqrt{\frac{\log\frac{2}{\delta}}{N}} + \frac{c_2 + \lambda d \log N + \lambda\varphi\left(\mathcal{W}_s\right)}{N} + \frac{c_3}{N^2},$$

where $\varphi(\mathcal{W}_s) = \|\boldsymbol{w}^* - \boldsymbol{\mu}_p\|_{\boldsymbol{\Sigma}_p^{-1}}$ when the Gaussian version of Algorithm 1 is used with prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ estimated from $\mathcal{W}_s$, while:

$$\varphi(\mathcal{W}_s) = \frac{1}{\sigma_p^2}\sum_{\boldsymbol{w}\in\mathcal{W}_s}\frac{e^{-\beta\|\boldsymbol{w}^*-\boldsymbol{w}\|}}{\sum_{\boldsymbol{w}'\in\mathcal{W}_s} e^{-\beta\|\boldsymbol{w}^*-\boldsymbol{w}'\|}}\|\boldsymbol{w}^*-\boldsymbol{w}\| \tag{5}$$

is the softmin distance between the optimal and source weights when the mixture version of Algorithm 1 is used with C components and bandwidth σ_p^2 for the prior. Here $\beta = \frac{1}{2\sigma_p^2}$.

We refer the reader to Appendix A.2 for the proof and a specific definition of the constants. Four main terms constitute our bound: the approximation error due to the limited hypothesis space (first term), the variance (second and third terms), the distance to the prior (fourth term), and a constant term decaying as $\mathcal{O}(N^2)$. As we might have expected, the only difference between the bounds for the two versions of Algorithm 1 is in the term $\varphi(\mathcal{W}_s)$, i.e., the distance between the optimal weights $\boldsymbol{w}^*$ and the source weights $\mathcal{W}_s$. Specifically, for the mixture version we have the (smoothened) minimum distance to the source tasks' weights (Equation (5)), while for the Gaussian one we have the distance to the mean of such weights. This property shows a clear advantage of using the mixture version of Algorithm 1 rather than the Gaussian one: in order to tighten the bound, it is enough to have at least one source task that is close to the optimal solution of the target task. In fact, the Gaussian version requires the source tasks to be, on average, similar to the target task in order to perform well, while the mixture version only requires this property for one of them. In both cases, when the term $\varphi(\mathcal{W}_s)$ is reduced, the dominating error is due to the variance of the estimates, and, thus, the algorithm is expected to achieve good performance rather quickly, as new data is collected. Furthermore, as $N \to \infty$ the only error terms remaining are the irreducible approximation error due to the limited functional space and the variance term $\upsilon(\boldsymbol{w}^*)$. The latter is due to the fact that we minimize a biased estimate of the Bellman error and can be removed in cases where double sampling of the next state is possible (e.g., in simulation). We empirically verify these considerations in Section 6.

5 Related Works

Our approach is mostly related to [20]. Although we both assume the tasks to share similarities in their value functions, [20] consider only linear approximators and adopt a hierarchical Bayesian model of the corresponding weights' distribution, which is assumed Gaussian. On the other hand, our variational approximation allows for more general distribution families and can be combined with non-linear approximators. Furthermore, [20] propose a Dirichlet process model for the case where weights cluster into different classes, which relates to our mixture formulation and proves the importance of capturing more complicated task distributions. Finally, [20] considers the problem of jointly learning all given tasks, while we focus on transferring information from a set of source tasks to the target task. In [37], the authors propose a hierarchical Bayesian model for the distribution over MDPs. Unlike our approach and [20], they consider a distribution over transition probabilities and rewards, rather than value functions. In the same spirit of our method, they consider a Thompson sampling-based procedure which, at each iteration, samples a new task from the posterior and solves it. However, [37] consider only finite MDPs, which poses a severe limitation on the algorithm's applicability. On the contrary, our approach can handle high-dimensional tasks. In [9], the authors consider a family of tasks whose dynamics are governed by some hidden parameters and use Gaussian processes (GPs) to model such dynamics across tasks. Recently, [15] extended this approach by replacing GPs with Bayesian neural networks to obtain a more scalable approach. Both approaches result in a model-based algorithm that quickly adapts to new tasks by estimating their hidden parameters, while we propose a model-free method which does not require such assumptions.

Finally, our approach relates to recent algorithms for meta-learning/fast-adaptation of weights in neural networks [11, 12, 1]. Such approaches typically assume to have full access to the task distribution $\mathcal{D}$ (i.e., samples from $\mathcal{D}$ can be obtained on-demand) and build meta-models that quickly adapt to new tasks drawn from the same distribution. On the other hand, we assume only a fixed and limited set of source tasks, together with their approximate solutions, is available. Then, our goal is to speed-up the learning process of a new target task from $\mathcal{D}$ by transferring only these data, without requiring additional source tasks or experience samples from them.

6 Experiments

In this section, we provide an experimental evaluation of our approach in four different domains with increasing level of difficulty. In all experiments, we compare our Gaussian variational transfer algorithm (GVT) and the version using a c-component mixture of Gaussians (c-MGVT) to plain no-transfer RL (NT) with ϵ-greedy exploration and to a simple transfer baseline in which we randomly pick one source Q-function and fine-tune from its weights (FT). Finally, in Section 6.4 we empirically demonstrate the differences between our approach and the previously discussed fast-adaptation algorithms. We report the detailed parameters, together with additional results, in Appendix C.

6.1 The Rooms Problem

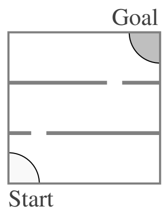
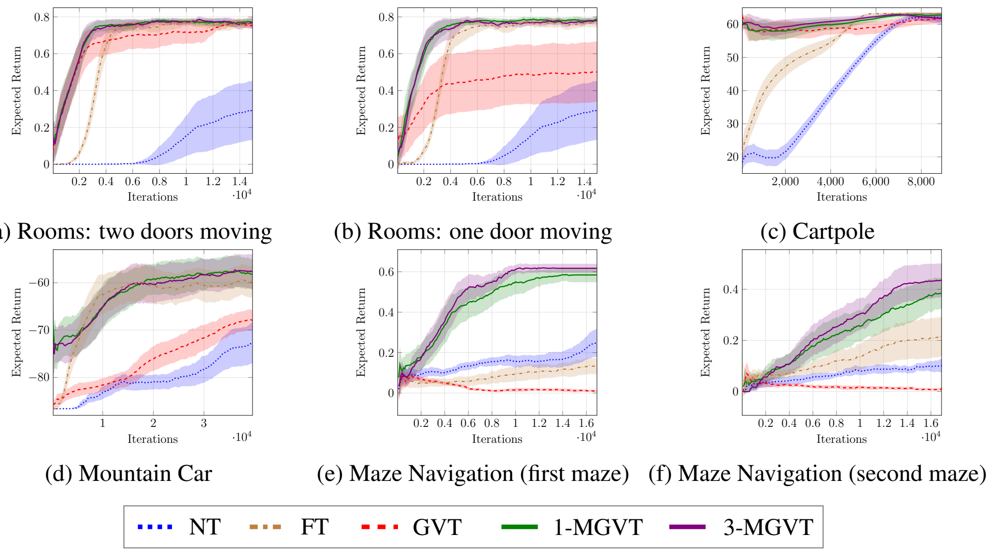


Figure 1: Rooms problem.

We consider an agent navigating in the environment depicted in Figure 1. The agent starts in the bottom-left corner and must move from one room to another to reach the goal position in the top-right corner. The rooms are connected by small doors whose locations are unknown to the agent. The state-space is modeled as a 10×10 continuous grid, while the action-space is the set of 4 movement directions (up, right, down, left). After each action, the agent moves by 1 in the chosen direction and the final position is corrupted by Gaussian noise $\mathcal{N}(0, 0.2)$. In case the agent hits a wall, its position remains unchanged. The reward is 1 when reaching the goal (after which the process terminates) and 0 otherwise, while the discount factor is $\gamma = 0.99$. In this experiment, we consider linearly parameterized Q-functions with 121 equally-spaced radial basis features.

We generate a set of 50 source tasks for the three-room environment of Figure 1 by sampling both door locations uniformly in the allowed space, and solve all of them by directly minimizing the TD error as presented in Section 3.4. Then, we use our algorithms to transfer from 10 source tasks sampled from the previously generated set. The average return over the last 50 learning episodes as a function of the number of iterations is shown in Figure 2a. Each curve is the result of 20 independent runs, each one resampling the target and source tasks, with 95% confidence intervals. Further details on the parameters adopted in this experiment are given in Appendix C.1. As expected, the no-transfer (NT) algorithm fails at learning the task in so few iterations due to the limited exploration provided by an ϵ-greedy policy. On the other hand, all our algorithms achieve a significant speed-up and converge to the optimal performance in few iterations, with GVT being slightly slower. FT achieves good performance as well, but it takes more time to adapt a random source Q-function. Interestingly, we notice that there is no advantage in adopting more than 1 component for the posterior in MGVT. This result is intuitive since, as soon as the algorithm figures out which is the target task, all the components move towards the same region.

To better understand the differences between GVT and MGVT, we now consider transferring from a slightly different distribution than the one from which target tasks are drawn. We generate 50 source tasks again but this time with the bottom door fixed at the center and the other one moving. Then, we repeat the previous experiment, allowing both doors to move when sampling target tasks. The results are shown in Figure 2b. Interestingly, MGVT seems almost unaffected by this change, proving that it has sufficient representation power to generalize to slightly different task distributions. The same



(a) Rooms: two doors moving (b) Rooms: one door moving (c) Cartpole

(d) Mountain Car (e) Maze Navigation (first maze) (f) Maze Navigation (second maze)

Figure 2: The online expected return achieved by the algorithm as a function of the number of iterations. Each curve is the average of 20 independent runs. 95% confidence intervals are shown.

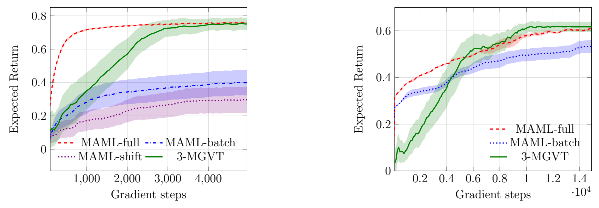
does not hold for GVT, which now is not able to solve many of the sampled target tasks, as can be noticed from the higher variance. Furthermore, the good performance of FT proves that GVT is, indeed, subject to a loss of information due to averaging the source weights. This result proves again that assuming Gaussian distributions can pose severe limitations in our transfer settings.

6.2 Classic Control

We now consider two well-known classic control environments: Cartpole and Mountain Car [32]. For both, we generate 20 source tasks by uniformly sampling their physical parameters (cart mass, pole mass, pole length for Cartpole and car speed for Mountain Car) and solve them by directly minimizing the TD error as in the previous experiment. We parameterize Q-functions using neural networks with one layer of 32 hidden units for Cartpole and 64 for Mountain Car. A better description of these two environments and their parameters is given in Appendix C.2. In this experiment, we use a Double Deep Q-Network (DDQN) [36] to provide a stronger no-transfer baseline for comparison. The results (same settings of Section 6.1) are shown in Figures 2c and 2d. For Cartpole (Figure 2c), all variational transfer algorithms are almost zero-shot. This result is expected since, although we vary the system parameters in a wide range, the optimal Q-values of states near the balanced position are similar for all tasks. On the contrary, in Mountain Car (Figure 2d) the optimal Q-functions become very different when changing the car speed. This phenomenon hinders the learning of GVT in the target task, while MGVT achieves a good jump-start and converges in fewer iterations. Similarly to the Rooms domain, the naive weight adaptation of FT makes it slower than MGVT in both domains.

6.3 Maze Navigation

Finally, we consider a robotic agent navigating mazes. At the beginning of each episode, the agent is dropped to a random position in a $10m^2$ maze and must reach a goal area in the smallest time possible. The robot is equipped with sensors detecting its absolute position, its orientation, the distance to any obstacle within $2m$ in 9 equally-spaced directions, and whether the goal is present in the same range. The only actions available are *move forward* with speed $0.5m/s$ or *rotate* (in either direction) with speed of $\pi/8\ rad/s$. Each time step corresponds to $1s$ of simulation. The reward is 1 for reaching the goal and 0 otherwise, while the discount factor is $\gamma = 0.99$. For this experiment, we design a set of 20 different mazes and solve them using a DDQN with two layers of 32 neurons and ReLU activations. Then, we fix a target maze and transfer from 5 source mazes uniformly sampled from such set (excluding the chosen target). To further assess the robustness of our method, we now consider transferring from the Q-functions learned by DDQNs instead of those obtained by minimizing the TD error as in the previous domains. From our considerations of Sections 3.4 and 4,



(a) The Rooms Problem (b) Maze Navigation

Figure 3: MAML vs 3-MGVT in our navigation problems.

the fixed-points of the two algorithms are different, which creates a further challenge for our method. We show the results for two different target mazes in Figure 2e, and Figure 2f, while referring the reader to Appendix C.3 for their illustration and additional results. Once again, MGVT achieves a remarkable speed-up over (no-transfer) DDQN. This time, using 3 components achieves slightly better performance than using only 1, which is likely due to the fact that the task distribution is much more complicated than in the previous domains. For the same reason, GVT shows negative transfer and performs even worse than DDQN. Similarly, FT performs much worse than in the previous domains and negatively transfer in the more complicated target maze of Figure 2e.

6.4 A Comparison to Fast-Adaptation Algorithms

In order to provide a better understanding of the differences between our settings and the ones typically considered in fast-adaptation algorithms, we now show a comparison to the recently proposed meta-learner MAML [11]. We repeat the previous experiments, focusing on the navigation tasks, using two different versions of MAML. In the first one (MAML-full), we meta-train using the full distribution over tasks for a number of iterations that allows the meta-policy to converge. In the second one (MAML-batch), we meta-train only on the same number of fixed source tasks as the one used for our algorithm, allowing again the meta-policy to reach convergence. In both cases, we meta-test on random tasks sampled from the full distribution. The results are shown in Figure 3 in comparison to our best algorithm (3-MGVT), where each curve is obtained by averaging 5 meta-testing runs for each of 4 different meta-policies. Additional details are given in Appendix C.4. In both cases, the full version of MAML achieves a much better jumpstart and adapts much faster than our approach. However, this is no more the case when limiting the number of source tasks. In fact, this situation reduces to the case where the task distribution at meta-training is a discrete uniform over the fixed source tasks, while at meta-testing the algorithm is required to generalize to a different distribution. This is a case that arises quite frequently in practice for which MAML was not specifically designed. Things get even worse when we explicitly add a shift to the meta-training distribution as we did in Figure 2b for the rooms problem (MAML-shift in Figure 3a). Although we meta-trained on the full distribution, the final performance was even worse than the one using the fixed source tasks. Finally, notice that we compare the algorithms w.r.t. the number of gradient steps, even though our approach collects only one new sample at each iteration while MAML collects a full batch of trajectories.

7 Conclusion

We presented a variational method for transferring value functions in RL. We showed our approach to be general, in the sense that it can be combined with several distributions and function approximators, while providing two practical algorithms based on Gaussians and mixtures of Gaussians, respectively. We analyzed both from a theoretical and empirical perspective, proving that the Gaussian version has severe limitations, while the mixture one is much better for our transfer settings. We evaluated the proposed algorithms in different domains, showing that both achieve excellent performance in simple tasks, while only the mixture version is able to handle complex environments.

Since our algorithm effectively models the uncertainty over tasks, a relevant future work is to design an algorithm that explicitly explores the target task to reduce such uncertainty. Furthermore, our variational approach could be extended to model a distribution over optimal policies instead of value functions, which might allow better transferred behavior.

References

[1] Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended PAC-Bayes theory. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[2] Kavosh Asadi and Michael L Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pages 243–252, 2017.

[3] Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. Efficient exploration through bayesian deep q-networks. *arXiv preprint arXiv:1802.04412*, 2018.

[4] Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[5] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pages 4055–4065, 2017.

[6] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[7] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[8] Olivier Catoni. Pac-bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*, 2007.

[9] Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, page 1432, 2016.

[10] Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 720–727. ACM, 2006.

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[12] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[13] John R Hershey and Peder A Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, 2007.

[14] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[15] Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. In *Advances in Neural Information Processing Systems*, pages 6250–6261, 2017.

[16] Jens Kober and Jan R Peters. Policy search for motor primitives in robotics. In *Advances in neural information processing systems*, pages 849–856, 2009.

[17] George Konidaris and Andrew Barto. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 489–496. ACM, 2006.

[18] George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. 2007.

[19] Alessandro Lazaric. Transfer in reinforcement learning: a framework and a survey. In *Reinforcement Learning*. 2012.

[20] Alessandro Lazaric and Mohammad Ghavamzadeh. Bayesian multi-task reinforcement learning. In *ICML-27th International Conference on Machine Learning*, pages 599–606. Omnipress, 2010.

[21] Alessandro Lazaric, Marcello Restelli, and Andrea Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, 2008.

[22] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[23] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[24] Odalric-Ambrym Maillard, Rémi Munos, Alessandro Lazaric, and Mohammad Ghavamzadeh. Finite-sample analysis of bellman residual minimization. In *Proceedings of 2nd Asian Conference on Machine Learning*, pages 299–314, 2010.

[25] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

[26] Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions. *arXiv preprint arXiv:1402.0635*, 2014.

[27] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1994.

[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[29] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[30] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[31] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[32] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[33] Matthew E Taylor, Nicholas K Jong, and Peter Stone. Transferring instances for model-based reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 488–505. Springer, 2008.

[34] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

[35] Andrea Tirinzoni, Andrea Sessa, Matteo Pirotta, and Marcello Restelli. Importance weighted transfer of samples in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4936–4945. PMLR, 2018.

[36] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. 2016.

[37] Aaron Wilson, Alan Fern, Soumya Ray, and Prasad Tadepalli. Multi-task reinforcement learning: a hierarchical bayesian approach. In *Proceedings of the 24th international conference on Machine learning*, pages 1015–1022. ACM, 2007.

A Proofs

A.1 Proof of Theorem 1

Theorem 1. *Let Q^* be the fixed-point of the optimal Bellman operator T. Define the action-gap function $g(s)$ as the difference between the value of the best action and the second best action at each state s. Let $\widetilde{Q}$ be the fixed-point of the mellow Bellman operator $\widetilde{T}$ with parameter $\kappa > 0$ and denote by $\beta_\kappa > 0$ the inverse temperature of the induced Boltzmann distribution (as in [2]). Then:*

$$\left\|Q^* - \widetilde{Q}\right\|_\infty \leq \frac{2\gamma R_{max}}{(1-\gamma)^2} \left\| \frac{1}{1 + \frac{1}{|\mathcal{A}|} e^{\beta_\kappa g}} \right\|_\infty . \tag{4}$$

Proof. We begin by noticing that:

$$\begin{aligned}
\left\|Q^* - \widetilde{Q}\right\|_\infty &= \left\|TQ^* - \widetilde{T}\widetilde{Q}\right\|_\infty \\
&= \left\|TQ^* - \widetilde{T}Q^* + \widetilde{T}Q^* - \widetilde{T}\widetilde{Q}\right\|_\infty \\
&\leq \left\|TQ^* - \widetilde{T}Q^*\right\|_\infty + \left\|\widetilde{T}Q^* - \widetilde{T}\widetilde{Q}\right\|_\infty \\
&\leq \left\|TQ^* - \widetilde{T}Q^*\right\|_\infty + \gamma \left\|Q^* - \widetilde{Q}\right\|_\infty ,
\end{aligned}$$

where the first inequality follows from Minkowsky's inequality and the second one from the contraction property of the mellow Bellman operator. This implies that:

$$\left\|Q^* - \widetilde{Q}\right\|_\infty \leq \frac{1}{1-\gamma} \left\|TQ^* - \widetilde{T}Q^*\right\|_\infty . \tag{6}$$

Let us bound the norm on the right-hand side separately. In order to do that, we will bound the function $\left|TQ^*(s,a) - \widetilde{T}Q^*(s,a)\right|$ point-wisely for any pair $\langle s, a\rangle$. By applying the definition of the optimal and mellow Bellman operators, we obtain:

$$\begin{aligned}
\left|TQ^*(s,a) - \widetilde{T}Q^*(s,a)\right| &= \left|R(s,a) + \gamma\mathbb{E}\left[\max_{a'} Q^*(s',a')\right] - R(s,a) - \gamma\mathbb{E}\left[\operatorname*{mm}_{a'} Q^*(s',a')\right]\right| \\
&= \gamma \left|\mathbb{E}\left[\max_{a'} Q^*(s',a')\right] - \mathbb{E}\left[\operatorname*{mm}_{a'} Q^*(s',a')\right]\right| \\
&\leq \gamma\mathbb{E}\left[\left|\max_{a'} Q^*(s',a') - \operatorname*{mm}_{a'} Q^*(s',a')\right|\right] .
\end{aligned} \tag{7}$$

Thus, bounding this quantity reduces to bounding $|\max_a Q^*(s,a) - \mathrm{mm}_a Q^*(s,a)|$ point-wisely for any s. Recall that applying the mellow Bellman operator is equivalent to computing an expectation under a Boltzmann distribution with inverse temperature β_κ induced by κ [2]. Thus, we can write:

$$\begin{aligned}
\left|\max_a Q^*(s,a) - \operatorname*{mm}_a Q^*(s,a)\right| &= \left|\sum_a \pi^*(a|s)Q^*(s,a) - \sum_a \pi_{\beta_\kappa}(a|s)Q^*(s,a)\right| \\
&= \left|\sum_a Q^*(s,a)\left(\pi^*(a|s) - \pi_{\beta_\kappa}(a|s)\right)\right| \\
&\leq \sum_a |Q^*(s,a)| \left|\pi^*(a|s) - \pi_{\beta_\kappa}(a|s)\right| \\
&\leq \frac{R_{max}}{1-\gamma} \sum_a \left|\pi^*(a|s) - \pi_{\beta_\kappa}(a|s)\right| ,
\end{aligned} \tag{8}$$

where π^* is the optimal (deterministic) policy w.r.t. Q^* and π_{β_κ} is the Boltzmann distribution induced by Q^* with inverse temperature β_κ:

$$\pi_{\beta_\kappa}(a|s) = \frac{e^{\beta_\kappa Q^*(s,a)}}{\sum_{a'} e^{\beta_\kappa Q^*(s,a')}} .$$

Denote by $a_1(s)$ the optimal action for state s under Q^*. We can then write:

$$\begin{aligned}\sum_a |\pi^*(a|s) - \pi_{\beta_\kappa}(a|s)| &= |\pi^*(a_1(s)|s) - \pi_{\beta_\kappa}(a_1(s)|s)| + \sum_{a \neq a_1(s)} |\pi^*(a|s) - \pi_{\beta_\kappa}(a|s)| \\ &= |1 - \pi_{\beta_\kappa}(a_1(s)|s)| + \sum_{a \neq a_1(s)} |\pi_{\beta_\kappa}(a|s)| \\ &= 2\,|1 - \pi_{\beta_\kappa}(a_1(s)|s)|\,. \end{aligned} \tag{9}$$

Finally, denoting with $a_2(s)$ the second-best action in state s, let us bound this last term:

$$\begin{aligned} |1 - \pi_{\beta_\kappa}(a_1(s)|s)| &= \left|1 - \frac{e^{\beta_\kappa Q^*(s,a_1(s))}}{\sum_{a'} e^{\beta_\kappa Q^*(s,a')}}\right| \\ &= \left|1 - \frac{e^{\beta_\kappa (Q^*(s,a_1(s)) - Q^*(s,a_2(s)))}}{\sum_{a'} e^{\beta_\kappa (Q^*(s,a') - Q^*(s,a_2(s)))}}\right| \\ &= \left|1 - \frac{e^{\beta_\kappa g(s)}}{\sum_{a'} e^{\beta_\kappa (Q^*(s,a') - Q^*(s,a_2(s)))}}\right| \\ &= \left|1 - \frac{e^{\beta_\kappa g(s)}}{e^{\beta_\kappa g(s)} + \sum_{a' \neq a_1(s)} e^{\beta_\kappa (Q^*(s,a') - Q^*(s,a_2(s)))}}\right| \\ &\leq \left|1 - \frac{e^{\beta_\kappa g(s)}}{e^{\beta_\kappa g(s)} + |\mathcal{A}|}\right| \\ &= \left|\frac{1}{1 + \frac{1}{|\mathcal{A}|} e^{\beta_\kappa g(s)}}\right|. \end{aligned} \tag{10}$$

Combining Eq. (8), (9), and (10), we obtain:

$$\left|\max_a Q^*(s,a) - \operatorname*{mm}_a Q^*(s,a)\right| \leq \frac{2R_{max}}{1-\gamma}\left|\frac{1}{1 + \frac{1}{|\mathcal{A}|} e^{\beta_\kappa g(s)}}\right|.$$

Finally, using Eq. (7) we get:

$$\left|TQ^*(s,a) - \widetilde{T}Q^*(s,a)\right| \leq \frac{2\gamma R_{max}}{1-\gamma}\mathbb{E}\left[\left|\frac{1}{1 + \frac{1}{|\mathcal{A}|} e^{\beta_\kappa g(s')}}\right|\right].$$

Taking the norm and plugging this into Eq. (6) concludes the proof. □

A.2 Proof of Theorem 2

We begin by proving some important lemmas. Then, we use them to derive a finite-sample analysis of Algorithm 1 with linearly parameterized value functions for both Gaussian distributions (Theorem 3) and Gaussian mixture models (Theorem 4). Theorem 2 follows by combining these two results.

Finite-sample Analysis of the Variational Transfer Algorithm We start by proving some important properties of the variational approximation introduced in Section 3.1. Our results generalize those of existing works that consider variational approximations of intractable Gibbs posteriors [**?**]. From now on, we consider only Q-functions parameterized by weights $\boldsymbol{w}$ and assume them to be uniformly bounded by $\frac{R_{max}}{1-\gamma}$.

Lemma 1. *Let p and q be arbitrary distributions over weights $\boldsymbol{w}$, and ν be a probability measure over $\mathcal{S} \times \mathcal{A}$. Consider a dataset D of N i.i.d. samples where state-action couples are distributed according to ν and define $v(\boldsymbol{w}) \triangleq \mathbb{E}_\nu\left[Var_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Then, for any $\lambda > 0$ and $\delta > 0$, with probability at least $1-\delta$, the following two inequalities hold simultaneously:*

$$\mathbb{E}_q\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \leq \mathbb{E}_q\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right] - \mathbb{E}_q\left[v(\boldsymbol{w})\right] + \frac{\lambda}{N}KL(q||p) + 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}} \tag{11}$$

$$\mathbb{E}_q\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right] \leq \mathbb{E}_q\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] + \mathbb{E}_q\left[v(\boldsymbol{w})\right] + \frac{\lambda}{N}KL(q||p) + 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}. \tag{12}$$

Proof. From Hoeffding's inequality we have:

$$P\left(\left|\mathbb{E}_{\nu,\mathcal{P}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right]-\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right|>\epsilon\right)\leq 2\exp\left(-\frac{2N\epsilon^2}{\left(2\frac{R_{max}}{1-\gamma}\right)^4}\right)$$

which implies that, for any $\delta>0$, with probability at least $1-\delta$:

$$\left|\mathbb{E}_{\nu,\mathcal{P}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right]-\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right|\leq 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}.$$

Under independence assumptions, the expected TD error can be re-written as:

$$\begin{aligned}\mathbb{E}_{\nu,\mathcal{P}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right]&=\mathbb{E}_{\nu,\mathcal{P}}\left[\frac{1}{N}\sum_{i=1}^{N}(r_i+\gamma\operatorname*{mm}_{a'}Q_{\boldsymbol{w}}(s_i',a')-Q_{\boldsymbol{w}}(s_i,a_i))^2\right]\\&=\mathbb{E}_{\nu,\mathcal{P}}\left[(R(s,a)+\gamma\operatorname*{mm}_{a'}Q_{\boldsymbol{w}}(s',a')-Q_{\boldsymbol{w}}(s,a))^2\right]\\&=\mathbb{E}_{\nu}\left[\mathbb{E}_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})^2\right]\right]\\&=\mathbb{E}_{\nu}\left[Var_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})\right]+\mathbb{E}_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})\right]^2\right]\\&=v(\boldsymbol{w})+\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2,\end{aligned}$$

where $v(\boldsymbol{w})\triangleq\mathbb{E}_{\nu}\left[Var_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Thus:

$$\left|\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2+v(\boldsymbol{w})-\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right|\leq 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}. \tag{13}$$

From the change of measure inequality [?], we have that, for any measurable function $f(\boldsymbol{w})$ and any two probability measures p and q:

$$\log\mathbb{E}_p\left[e^{f(\boldsymbol{w})}\right]\geq\mathbb{E}_q\left[f(\boldsymbol{w})\right]-KL(q||p).$$

Thus, multiplying both sides of (13) by $\lambda^{-1}N$ and applying the change of measure inequality with $f(\boldsymbol{w})=\lambda^{-1}N\left|\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2+v(\boldsymbol{w})-\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right|$, we obtain:

$$\mathbb{E}_q\left[f(\boldsymbol{w})\right]-KL(q||p)\leq\log\mathbb{E}_p\left[e^{f(\boldsymbol{w})}\right]\leq 4\frac{R_{max}^2\lambda^{-1}N}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}},$$

where the second inequality holds since the right-hand side of (13) does not depend on $\boldsymbol{w}$. Finally, we can explicitly write:

$$\mathbb{E}_q\left[\left|\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2+v(\boldsymbol{w})-\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_D^2\right|\right]\leq\frac{\lambda}{N}KL(q||p)+4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}$$

from which the lemma follows straightforwardly. □

From Lemma 1 we can straightforwardly prove the following result which will be of fundamental importance in the remaining.

Lemma 2. *Fix a task $\mathcal{M}_\tau$. Let p be a prior distribution over weights $\boldsymbol{w}$, and ν be a probability measure over $\mathcal{S}\times\mathcal{A}$. Assume $\widehat{\xi}$ is the minimizer of* (2) *for a dataset D of N i.i.d. samples where state-action couples are distributed according to ν. Define $v(\boldsymbol{w})\triangleq\mathbb{E}_{\nu}\left[Var_{\mathcal{P}_\tau}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Then, for any $\delta>0$, with probability at least $1-\delta$:*

$$\mathbb{E}_{q_{\widehat{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right]\leq\inf_{\xi\in\Xi}\left\{\mathbb{E}_{q_\xi}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right]+\mathbb{E}_{q_\xi}\left[v(\boldsymbol{w})\right]+2\frac{\lambda}{N}KL(q_\xi||p)\right\}+8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}.$$

Proof. Let us use Lemma 1 for the specific choice $q = q_{\widehat{\xi}}$. From Eq. (11), we have:

$$\begin{aligned}
\mathbb{E}_{q_{\widehat{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] &\leq \mathbb{E}_{q_{\widehat{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{D}^{2}\right] - \mathbb{E}_{q_{\widehat{\xi}}}\left[v(\boldsymbol{w})\right] + \frac{\lambda}{N} KL(q_{\widehat{\xi}}||p) + 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}} \\
&\leq \mathbb{E}_{q_{\widehat{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{D}^{2}\right] + \frac{\lambda}{N} KL(q_{\widehat{\xi}}||p) + 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}} \\
&= \inf_{\xi\in\Xi}\left\{\mathbb{E}_{q_{\xi}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{D}^{2}\right] + \frac{\lambda}{N} KL(q_{\xi}||p)\right\} + 4\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}},
\end{aligned}$$

where the second inequality holds since $v(\boldsymbol{w}) > 0$, while the equality holds from the definition of $\widehat{\xi}$. We can now use Eq. (12) to bound $\mathbb{E}_{q_{\xi}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{D}^{2}\right]$, thus obtaining:

$$\mathbb{E}_{q_{\widehat{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] \leq \inf_{\xi\in\Xi}\left\{\mathbb{E}_{q_{\xi}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] + \mathbb{E}_{q_{\xi}}\left[v(\boldsymbol{w})\right] + 2\frac{\lambda}{N} KL(q_{\xi}||p)\right\} + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}.$$

This concludes the proof. □

It is worth noting the generality of Lemma 2: in bounding the expected Bellman error we do not need to assume any particular distribution, nor we have to assume any particular function approximator.

Finite-sample Analysis of GVT and MGVT We are now ready to state our main results. We start from the Gaussian case and then straightforwardly extend the proof to the mixture one.

Theorem 3. *Fix a target task $\mathcal{M}_\tau$. Assume linearly parameterized value functions $Q_{\boldsymbol{w}}(s,a) = \boldsymbol{w}^T\boldsymbol{\phi}(s,a)$ with bounded weights $\|\boldsymbol{w}\|_2 \leq w_{max}$ and uniformly bounded features $\|\boldsymbol{\phi}(s,a)\|_2 \leq \phi_{\max}$. Consider the Gaussian version of Algorithm 1 with prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ and denote by $(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})$ the variational parameter minimizing the objective of Eq.* (2) *on a dataset D of N i.i.d. samples distributed according to τ and ν. Let $\boldsymbol{w}^* = \operatorname{arginf}_{\boldsymbol{w}} \left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}$ and define $v(\boldsymbol{w}^*) \triangleq \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*, \frac{1}{N}\boldsymbol{I})}\left[v(\boldsymbol{w})\right]$, with $v(\boldsymbol{w}) \triangleq \mathbb{E}_{\nu}\left[Var_{\mathcal{P}}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Then, there exist constants c_1, c_2, c_3 such that, with probability at least $1-\delta$ over the choice of the dataset D:*

$$\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] \leq 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_{\nu}^{2} + v(\boldsymbol{w}^*) + c_1\sqrt{\frac{\log\frac{2}{\delta}}{N}} + \frac{c_2 + \lambda d \log N + \lambda\left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}}}{N} + \frac{c_3}{N^2}. \tag{14}$$

Proof. Using Lemma 2 with variational parameters $\widehat{\boldsymbol{\xi}} = (\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})$, we have:

$$\begin{aligned}
\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] &\leq \inf_{\boldsymbol{\xi}\in\Xi}\left\{\mathbb{E}_{q_{\boldsymbol{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] + \mathbb{E}_{q_{\boldsymbol{\xi}}}\left[v(\boldsymbol{w})\right] + 2\frac{\lambda}{N} KL(q_{\boldsymbol{\xi}}||p)\right\} + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}} \\
&\leq \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*, c\boldsymbol{I})}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_{\nu}^{2}\right] + \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*, c\boldsymbol{I})}\left[v(\boldsymbol{w})\right] + 2\frac{\lambda}{N} KL\left(\mathcal{N}(\boldsymbol{w}^*, c\boldsymbol{I}) \mid\mid p\right) \\
&\quad + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}},
\end{aligned} \tag{15}$$

where the second inequality is due to the fact that, since Lemma 2 contains an infimum over the variational parameters, we can upper bound its right-hand side by choosing any specific $\boldsymbol{\xi}$ from Ξ. Here, we choose $\boldsymbol{\mu} = \boldsymbol{w}^*$ and $\boldsymbol{\Sigma} = c\boldsymbol{I}$, for some positive constant $c > 0$. Let us now bound these terms separately.

Bounding the expected Bellman error We have:

$$\begin{aligned}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] &= \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\mathbb{E}_\nu\left[(\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}})^2\right]\right]\\ &= \mathbb{E}_\nu\left[\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[(\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}})^2\right]\right]\\ &= \mathbb{E}_\nu\left[\mathbb{E}^2_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right]\right] + \mathbb{E}_\nu\left[Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right]\right].\end{aligned} \tag{16}$$

Let us bound these two terms point-wisely for each pair $\langle s, a\rangle$. For the first expectation, we have:

$$\begin{aligned}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right] &= \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[R(s,a) + \gamma\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\boldsymbol{\phi}(s',a')\right] - \boldsymbol{w}^T\boldsymbol{\phi}(s,a)\right]\\ &= R(s,a) + \gamma\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\boldsymbol{\phi}(s',a')\right]\right] - \boldsymbol{w}^{*T}\boldsymbol{\phi}(s,a).\end{aligned} \tag{17}$$

To bound the second term, we adopt Jensen's inequality:

$$\begin{aligned}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\boldsymbol{\phi}(s',a')\right]\right] &= \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'} e^{\kappa\boldsymbol{w}^T\boldsymbol{\phi}(s',a')}\right]\right]\\ &\le \mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[e^{\kappa\boldsymbol{w}^T\boldsymbol{\phi}(s',a')}\right]\right].\end{aligned} \tag{18}$$

Now, since we know that $\boldsymbol{w}^T\boldsymbol{\phi}(s',a') \sim \mathcal{N}(\boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a'), c\,\boldsymbol{\phi}(s',a')^T\boldsymbol{\phi}(s',a'))$, $e^{\kappa\boldsymbol{w}^T\boldsymbol{\phi}(s',a')}$ follows a log-normal distribution with mean $e^{\kappa\boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')+\frac{1}{2}\kappa^2 c\boldsymbol{\phi}(s',a')^T\boldsymbol{\phi}(s',a')}$. Thus:

$$\begin{aligned}\mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[e^{\kappa\boldsymbol{w}^T\boldsymbol{\phi}(s',a')}\right]\right] &= \mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'} e^{\kappa\boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')+\frac{1}{2}\kappa^2 c\boldsymbol{\phi}(s',a')^T\boldsymbol{\phi}(s',a')}\right]\\ &\le \mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'} e^{\kappa\boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')}e^{\frac{1}{2}\kappa^2 c\boldsymbol{\phi}^2_{max}}\right]\\ &= \mathbb{E}_{s'}\left[\frac{1}{\kappa}\log\frac{1}{|\mathcal{A}|}\sum_{a'} e^{\kappa\boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')}\right] + \frac{1}{2}\kappa c\boldsymbol{\phi}^2_{max}\\ &= \mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')\right] + \frac{1}{2}\kappa c\boldsymbol{\phi}^2_{max}.\end{aligned}$$

Plugging this into (18) and then into (17), we obtain:

$$\begin{aligned}\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right] &\le R(s,a) + \gamma\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^{*T}\boldsymbol{\phi}(s',a')\right] + \frac{1}{2}\gamma\kappa c\boldsymbol{\phi}^2_{max} - \boldsymbol{w}^{*T}\boldsymbol{\phi}(s,a)\\ &= \widetilde{B}_{\boldsymbol{w}^*} + \frac{1}{2}\gamma\kappa c\boldsymbol{\phi}^2_{max}.\end{aligned}$$

This implies:

$$\begin{aligned}\mathbb{E}^2_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right] &\le \left(\widetilde{B}_{\boldsymbol{w}^*} + \frac{1}{2}\gamma\kappa c\boldsymbol{\phi}^2_{max}\right)^2\\ &\le 2\widetilde{B}^2_{\boldsymbol{w}^*} + \frac{1}{2}\gamma^2\kappa^2c^2\boldsymbol{\phi}^4_{max},\end{aligned}$$

where the second inequality follows from Cauchy-Schwarz inequality. Going back to (16), the first term can now be upper bounded by:

$$\mathbb{E}_\nu\left[\mathbb{E}^2_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right]\right] \le 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + \frac{1}{2}\gamma^2\kappa^2c^2\boldsymbol{\phi}^4_{max}.$$

Let us now consider the variance term of (16) and derive a bound that holds point-wisely for any s, a. We have:

$$\begin{aligned}
Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right] &= Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[R(s,a) + \gamma\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\boldsymbol{\phi}(s',a')\right] - \boldsymbol{w}^T\boldsymbol{\phi}(s,a)\right] \\
&= Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\gamma\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\boldsymbol{\phi}(s',a') - \frac{1}{\gamma}\boldsymbol{w}^T\boldsymbol{\phi}(s,a)\right]\right] \\
&= Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\gamma\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \boldsymbol{w}^T\left(\boldsymbol{\phi}(s',a') - \frac{1}{\gamma}\boldsymbol{\phi}(s,a)\right)\right]\right] \\
&= \gamma^2 Var_{\mathcal{N}(\boldsymbol{w}^*,\boldsymbol{I})}\left[\mathbb{E}_{s'}\left[\operatorname*{mm}_{a'} \sqrt{c}\boldsymbol{w}^T\left(\boldsymbol{\phi}(s',a') - \frac{1}{\gamma}\boldsymbol{\phi}(s,a)\right)\right]\right].
\end{aligned}$$

From Cauchy-Schwarz inequality:

$$\begin{aligned}
\sqrt{c}\left|\boldsymbol{w}^T\left(\boldsymbol{\phi}(s',a') - \frac{1}{\gamma}\boldsymbol{\phi}(s,a)\right)\right| &\le \sqrt{c}\left\|\boldsymbol{w}\right\|\left\|\boldsymbol{\phi}(s',a') - \frac{1}{\gamma}\boldsymbol{\phi}(s,a)\right\| \\
&\le \sqrt{c}\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}\frac{1+\gamma}{\gamma}.
\end{aligned}$$

Then, the random variable over which the variance is computed is limited in $[-\sqrt{c}\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}\frac{1+\gamma}{\gamma}, \sqrt{c}\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}\frac{1+\gamma}{\gamma}]$ and the variance can be straightforwardly bounded using Popoviciu's inequality:

$$Var_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\widetilde{T}Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\right] \le \gamma^2\frac{1}{4}\left(2\sqrt{c}\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}\frac{1+\gamma}{\gamma}\right)^2 = c\left(\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}(1+\gamma)\right)^2.$$

We can finally plug everything into (16), thus obtaining:

$$\mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2\right] \le 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + \frac{1}{2}\gamma^2\kappa^2c^2\boldsymbol{\phi}_{max}^4 + c\left(\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}(1+\gamma)\right)^2.$$

Bounding the KL divergence We have:

$$\begin{aligned}
KL\left(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I}) \,||\, p\right) &= KL\left(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I}) \,||\, \mathcal{N}(\boldsymbol{\mu}_p,\boldsymbol{\Sigma}_p)\right) \\
&= \frac{1}{2}\left(\log\frac{|\boldsymbol{\Sigma}_p|}{c^d} + c\mathrm{Tr}\left(\boldsymbol{\Sigma}_p^{-1}\right) + \left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}} - d\right) \\
&\le \frac{1}{2}d\log\frac{\sigma_{max}}{c} + \frac{1}{2}d\frac{c}{\sigma_{min}} + \frac{1}{2}\left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}}.
\end{aligned}$$

Now, putting all together into (15):

$$\begin{aligned}
\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \le{}& 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + \frac{1}{2}\gamma^2\kappa^2c^2\boldsymbol{\phi}_{max}^4 + c\left(\boldsymbol{w}_{max}\boldsymbol{\phi}_{max}(1+\gamma)\right)^2 + \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[v(\boldsymbol{w})\right] \\
&+ \frac{\lambda}{N}d\log\frac{\sigma_{max}}{c} + \frac{\lambda}{N}d\frac{c}{\sigma_{min}} + \frac{\lambda}{N}\left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}} + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}.
\end{aligned}$$

Since the bound holds for any $c > 0$, we can set it to $1/N$, thus obtaining:

$$\begin{aligned}
\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \le{}& 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + v(\boldsymbol{w}^*) + \frac{1}{N^2}\left(\frac{1}{2}\gamma^2\kappa^2\boldsymbol{\phi}_{max}^4 + \frac{\lambda d}{\sigma_{min}}\right) \\
&+ \frac{1}{N}\left(\boldsymbol{w}_{max}^2\boldsymbol{\phi}_{max}^2(1+\gamma)^2 + \lambda d(\log\sigma_{max} + \log N) + \lambda\left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}}\right) \\
&+ 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}
\end{aligned}$$

Finally, defining the constants $c_1 = \frac{8R_{max}^2}{\sqrt{2}(1-\gamma)^2}$, $c_2 = \boldsymbol{w}_{max}^2\boldsymbol{\phi}_{max}^2(1+\gamma)^2 + \lambda d\log\sigma_{max}$, and $c_3 = \frac{1}{2}\gamma^2\kappa^2\boldsymbol{\phi}_{max}^4 + \frac{\lambda d}{\sigma_{min}}$, we obtain:

$$\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \le 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + v(\boldsymbol{w}^*) + c_1\sqrt{\frac{\log\frac{2}{\delta}}{N}} + \frac{c_2 + \lambda d\log N + \lambda\left\|\boldsymbol{w}^* - \boldsymbol{\mu}_p\right\|_{\boldsymbol{\Sigma}_p^{-1}}}{N} + \frac{c_3}{N^2}.$$

□

Theorem 4. *Fix a target task $\mathcal{M}_\tau$. Assume linearly parameterized value functions $Q_{\boldsymbol{w}}(s,a) = \boldsymbol{w}^T\boldsymbol{\phi}(s,a)$ with bounded weights $\|\boldsymbol{w}\|_2 \le w_{max}$ and uniformly bounded features $\|\boldsymbol{\phi}(s,a)\|_2 \le \phi_{\max}$. Consider the mixture version of Algorithm 1 using C components, source task weights $\mathcal{W}_s$, and bandwidth σ_p^2 for the prior. Denote by $\widehat{\boldsymbol{\xi}} = (\widehat{\boldsymbol{\mu}}_1,\dots,\widehat{\boldsymbol{\mu}}_C,\widehat{\boldsymbol{\Sigma}}_1,\dots,\widehat{\boldsymbol{\Sigma}}_C)$ the variational parameters minimizing the objective of Eq. (2) on a dataset D of N i.i.d. samples distributed according to τ and ν. Let $\boldsymbol{w}^* = \operatorname{arginf}_{\boldsymbol{w}} ||\widetilde{B}_{\boldsymbol{w}}||_\nu^2$ and define $\upsilon(\boldsymbol{w}^*) \triangleq \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,\frac{1}{N}\boldsymbol{I})}[v(\boldsymbol{w})]$, with $v(\boldsymbol{w}) \triangleq \mathbb{E}_\nu\left[Var_{\mathcal{P}_\tau}\left[\widetilde{b}(\boldsymbol{w})\right]\right]$. Then, there exist constants c_1, c_2, c_3 such that, with probability at least $1-\delta$ over the choice of the dataset D:*

$$\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] \le 2\left\|\widetilde{B}_{\boldsymbol{w}^*}\right\|_\nu^2 + \upsilon(\boldsymbol{w}^*) + c_1\sqrt{\frac{\log\frac{2}{\delta}}{N}} + \frac{c_2 + \lambda d\log N + 2\lambda\varphi(\Delta)}{N} + \frac{c_3}{N^2},$$

where Δ is the vector of distances to the source tasks' weights, $\Delta_j = \frac{1}{2\sigma_p^2}\|\boldsymbol{w}^* - \boldsymbol{w}_j\|$, and, for a vector $\boldsymbol{x} = (x_1,\dots,x_d)$, $\varphi(\boldsymbol{x}) \triangleq \sum_i \frac{e^{-x_i}}{\sum_j e^{-x_j}} x_i$ is the softmin function.

Proof. Similarly to the previous proof, we can apply Lemma 2 with variational parameters $\widehat{\boldsymbol{\xi}} = (\widehat{\boldsymbol{\mu}}_1,\dots,\widehat{\boldsymbol{\mu}}_C,\widehat{\boldsymbol{\Sigma}}_1,\dots,\widehat{\boldsymbol{\Sigma}}_C)$, while choosing the same specific parameters for the right-hand side: $\boldsymbol{\mu}_i = \boldsymbol{w}^*$ and $\boldsymbol{\Sigma}_i = c\boldsymbol{I}$ for all $i = 1,\dots,C$. Then, we obtain:

$$\begin{aligned}\mathbb{E}_{q_{\widehat{\boldsymbol{\xi}}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] &\le \inf_{\boldsymbol{\xi}\in\Xi}\left\{\mathbb{E}_{q_{\boldsymbol{\xi}}}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] + \mathbb{E}_{q_{\boldsymbol{\xi}}}[v(\boldsymbol{w})] + 2\frac{\lambda}{N}KL(q_{\boldsymbol{\xi}}||p)\right\} + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}} \\ &\le \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}\left[\left\|\widetilde{B}_{\boldsymbol{w}}\right\|_\nu^2\right] + \mathbb{E}_{\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})}[v(\boldsymbol{w})] + 2\frac{\lambda}{N}KL\left(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,p\right) \\ &\quad + 8\frac{R_{max}^2}{(1-\gamma)^2}\sqrt{\frac{\log\frac{2}{\delta}}{2N}}. \end{aligned} \tag{19}$$

The only difference w.r.t. Eq. (15) of Theorem 3 is the KL divergence term, which now contains a mixture distribution. From Theorem 5 we have:

$$KL(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,p) \le KL(\chi^{(2)}||\chi^{(1)}) + \sum_j \chi_j^{(2)} KL(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,\mathcal{N}(\boldsymbol{w}_j,\sigma_p^2\boldsymbol{I})), \tag{20}$$

where the vectors $\chi^{(1)}$ and $\chi^{(2)}$ are the ones defined in Theorem 5. Notice that, since we reduced the posterior to one component, we can get rid of the index i. Using the definitions of these two vectors from Section 8 of [13], we have:

$$\chi_j^{(1)} = \frac{1}{|\mathcal{W}_s|}\ \forall j = 1,\dots,|\mathcal{W}_s|$$

$$\chi_j^{(2)} = \frac{e^{-KL(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,\mathcal{N}(\boldsymbol{w}_j,\sigma_p^2\boldsymbol{I}))}}{\sum_{j'} e^{-KL(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,\mathcal{N}(\boldsymbol{w}_{j'},\sigma_p^2\boldsymbol{I}))}}\ \forall j = 1,\dots,|\mathcal{W}_s|\,. \tag{21}$$

Since the KL divergence is:

$$KL(\mathcal{N}(\boldsymbol{w}^*,c\boldsymbol{I})\,||\,\mathcal{N}(\boldsymbol{w}_j,\sigma_p^2\boldsymbol{I})) = \frac{1}{2}\left(d\log\frac{\sigma_p^2}{c} + d\frac{c}{\sigma_p^2} + \frac{1}{\sigma_p^2}\|\boldsymbol{w}^* - \boldsymbol{w}_j\| - d\right),$$

Eq. (21) can be rewritten as:

$$\chi_j^{(2)} = \frac{e^{-\frac{1}{2\sigma_p^2}\|\boldsymbol{w}^*-\boldsymbol{w}_j\|}}{\sum_{j'} e^{-\frac{1}{2\sigma_p^2}\|\boldsymbol{w}^*-\boldsymbol{w}_{j'}\|}}\ \forall j = 1,\dots,|\mathcal{W}_s|\,.$$

Let us bound the two terms of (20) separately. For the first one, we have:

$$
\begin{aligned}
KL(\chi^{(2)}||\chi^{(1)}) &= \sum_j \chi_j^{(2)} \log \frac{\chi_j^{(2)}}{\chi_j^{(1)}} \\
&= \sum_j \chi_j^{(2)} \log \chi_j^{(2)} - \sum_j \chi_j^{(2)} \log \frac{1}{|\mathcal{W}_s|} \\
&\leq \log |\mathcal{W}_s| \,,
\end{aligned}
$$

where the inequality holds since the first term is negative. For the second term of (20):

$$
\begin{aligned}
\sum_j \chi_j^{(2)} KL(\mathcal{N}(\boldsymbol{w}^*, c\boldsymbol{I}) \,||\, \mathcal{N}(\boldsymbol{w}_j, \sigma_p^2 \boldsymbol{I})) &= \frac{1}{2} \sum_j \chi_j^{(2)} \left(d \log \frac{\sigma_p^2}{c} + d \frac{c}{\sigma_p^2} + \frac{1}{\sigma_p^2} \|\boldsymbol{w}^* - \boldsymbol{w}_j\| - d \right) \\
&\leq \frac{1}{2} d \log \frac{\sigma_p^2}{c} + \frac{1}{2} d \frac{c}{\sigma_p^2} + \sum_j \chi_j^{(2)} \frac{1}{2\sigma_p^2} \|\boldsymbol{w}^* - \boldsymbol{w}_j\| \\
&= \frac{1}{2} d \log \frac{\sigma_p^2}{c} + \frac{1}{2} d \frac{c}{\sigma_p^2} + \varphi(\Delta) \,.
\end{aligned}
$$

where we defined the vector Δ whose components are $\Delta_j = \frac{1}{2\sigma_p^2} \|\boldsymbol{w}^* - \boldsymbol{w}_j\|$. Putting the two terms together:

$$
KL(\mathcal{N}(\boldsymbol{w}^*, c\boldsymbol{I}) \,||\, p) \leq \log |\mathcal{W}_s| + \frac{1}{2} d \log \frac{\sigma_p^2}{c} + \frac{1}{2} d \frac{c}{\sigma_p^2} + \varphi(\Delta) \,.
$$

Notice that, from now on, one can simply apply the proof of Theorem 3 with $\sigma_{max} = \sigma_{min} = \sigma_p^2$ and $\frac{1}{2} \|\boldsymbol{w}^* - \boldsymbol{\mu}_p\|_{\boldsymbol{\Sigma}_p^{-1}}$ replaced by $\varphi(\Delta)$. Thus, by redefining the three constants to $c_1 = \frac{8R_{max}^2}{\sqrt{2}(1-\gamma)^2}$, $c_2 = \boldsymbol{w}_{max}^2 \boldsymbol{\phi}_{max}^2 (1+\gamma)^2 + \lambda d \log \sigma_p^2 + 2\lambda \log |\mathcal{W}_s|$, and $c_3 = \frac{1}{2}\gamma^2 \kappa^2 \boldsymbol{\phi}_{max}^4 + \frac{\lambda d}{\sigma_p^2}$, we can write that, with probability at least $1 - \delta$:

$$
\mathbb{E}_{q_{\boldsymbol{\xi}}} \left[\left\| \widetilde{B}_{\boldsymbol{w}} \right\|_\nu^2 \right] \leq 2 \left\| \widetilde{B}_{\boldsymbol{w}^*} \right\|_\nu^2 + \upsilon(\boldsymbol{w}^*) + c_1 \sqrt{\frac{\log \frac{2}{\delta}}{N}} + \frac{c_2 + \lambda d \log N + 2\lambda \varphi(\Delta)}{N} + \frac{c_3}{N^2}.
$$

□

Proof of Theorem 2. The theorem follows straightforwardly by combining Theorem 3 and Theorem 4. □

B Additional Details on the Algorithms

B.1 Gaussian Variational Transfer

Under Gaussian distributions, all quantities of interest for using Algorithm 1 can be computed very easily. The KL divergence between the prior and approximate posterior can be computed in closed-form as:

$$
KL\left(q_{\boldsymbol{\xi}}(\boldsymbol{w}) \,||\, p(\boldsymbol{w})\right) = \frac{1}{2} \left(\log \frac{|\boldsymbol{\Sigma}_p|}{|\boldsymbol{\Sigma}|} + \mathrm{Tr}\left(\boldsymbol{\Sigma}_p^{-1} \boldsymbol{\Sigma}\right) + (\boldsymbol{\mu} - \boldsymbol{\mu}_p)^T \boldsymbol{\Sigma}_p^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_p) - d \right), \tag{22}
$$

for $\boldsymbol{\xi} = (\boldsymbol{\mu}, \boldsymbol{L})$ and $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$. Its gradients with respect to the variational parameters are:

$$
\nabla_{\boldsymbol{\mu}} KL\left(q_{\boldsymbol{\xi}}(\boldsymbol{w}) \,||\, p(\boldsymbol{w})\right) = \boldsymbol{\Sigma}_p^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_p) \tag{23}
$$

$$
\nabla_{\boldsymbol{L}} KL\left(q_{\boldsymbol{\xi}}(\boldsymbol{w}) \,||\, p(\boldsymbol{w})\right) = \boldsymbol{\Sigma}_p^{-1} \boldsymbol{L} - (\boldsymbol{L}^{-1})^T \tag{24}
$$

Finally, the gradients w.r.t. the expected likelihood term of the variational objective (2) can be computed using the reparameterization trick (e.g., [14, 28]):

$$
\nabla_{\boldsymbol{\mu}} \mathbb{E}_{\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{L}\boldsymbol{L}^T)} \left[||B_{\boldsymbol{w}}||_D^2 \right] = \mathbb{E}_{\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})} \left[\nabla_{\boldsymbol{w}} ||B_{\boldsymbol{w}}||_D^2 \right] \quad \text{for} \quad \boldsymbol{w} = \boldsymbol{L}\boldsymbol{v} + \boldsymbol{\mu} \tag{25}
$$

$$
\nabla_{\boldsymbol{L}} \mathbb{E}_{\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{L}\boldsymbol{L}^T)} \left[||B_{\boldsymbol{w}}||_D^2 \right] = \mathbb{E}_{\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})} \left[\nabla_{\boldsymbol{w}} ||B_{\boldsymbol{w}}||_D^2 \cdot \boldsymbol{v}^T \right] \quad \text{for} \quad \boldsymbol{w} = \boldsymbol{L}\boldsymbol{v} + \boldsymbol{\mu} \tag{26}
$$

B.2 Mixture of Gaussian Variational Transfer

As mentioned in the main paper, for the mixture version of Algorithm 1 we rely on the upper bound on the KL divergence between two mixture of Gaussians presented in [13]. We report it here for the sake of completeness.

Theorem 5 ([13]). *Let $p = \sum_i c_i^{(p)} f_i^{(p)}$ and $q = \sum_j c_j^{(q)} f_j^{(q)}$ be two mixture of Gaussian distributions, where $f_i^{(p)} = \mathcal{N}(\boldsymbol{\mu}_i^{(p)}, \boldsymbol{\Sigma}_i^{(p)})$ denotes the i-th component of p, $c_i^{(p)}$ denotes its weight, and similarly for q. Introduce two vectors $\chi^{(1)}$ and $\chi^{(2)}$ such that $c_i^{(p)} = \sum_j \chi_{j,i}^{(2)}$ and $c_j^{(q)} = \sum_i \chi_{i,j}^{(1)}$. Then:*

$$KL(p||q) \leq KL(\chi^{(2)}||\chi^{(1)}) + \sum_{i,j} \chi_{j,i}^{(2)} KL(f_i^{(p)}||f_j^{(q)}). \tag{27}$$

Our new algorithm replaces the KL with the above-mentioned upper bound. Each time we require its value, we have to recompute the parameters $\chi^{(1)}$ and $\chi^{(2)}$ that tighten the bound. As shown in [13], we can use a simple fixed-point procedure for this purpose, alternating the computation of the two parameters as:

$$\chi_{i,j}^{(2)} = \frac{c_j^{(q)} \chi_{j,i}^{(1)}}{\sum_{i'} \chi_{j,i'}^{(1)}}, \quad \chi_{j,i}^{(1)} = \frac{c_i^{(p)} \chi_{i,j}^{(2)} e^{-KL(f_i^{(p)}||f_j^{(q)})}}{\sum_{j'} \chi_{i,j'}^{(2)} e^{-KL(f_i^{(p)}||f_{j'}^{(q)})}}. \tag{28}$$

Finally, both terms in the objective are now linear combinations of functions of the variational parameters of different components, and their gradients easily derive from the ones of the Gaussian case. Consider a posterior with C components, $q_{\boldsymbol{\xi}}(\boldsymbol{w}) = \frac{1}{C}\sum_{i=1}^{C} \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, and a prior distribution, constructed from the set of weights $\mathcal{W}_s = \left\{\boldsymbol{w}_1, ..., \boldsymbol{w}_{|\mathcal{W}_s|}\right\}$ of the sources' optimal Q-functions, $p(\boldsymbol{w}) = \frac{1}{|\mathcal{W}_s|}\sum_{j=1}^{|\mathcal{W}_s|} \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_j, \sigma_p^2 \boldsymbol{I})$. Then, using Theorem 5:

$$KL\left(q_{\boldsymbol{\xi}}(\boldsymbol{w}) \,||\, p(\boldsymbol{w})\right) \leq KL(\chi^{(2)}||\chi^{(1)}) + \sum_{i=1}^{C}\sum_{j=1}^{|\mathcal{W}_s|} \chi_{j,i}^{(2)} KL(\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \,||\, \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_j, \sigma_p^2 \boldsymbol{I})). \tag{29}$$

Substituting (29) in the negative ELBO in (2), we get the following upper bound on the objective:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\xi}) \leq \widetilde{\mathcal{L}}(\boldsymbol{\xi}) = \mathbb{E}_{\boldsymbol{w}\sim q_{\boldsymbol{\xi}}}\left[\|B_{\boldsymbol{w}}\|_D^2\right] + \frac{\lambda}{N} KL(\chi^{(2)}||\chi^{(1)}) \\ + \frac{\lambda}{N}\sum_{i=1}^{C}\sum_{j=1}^{|\mathcal{W}_s|} \chi_{j,i}^{(2)} KL(\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \,||\, \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_j, \sigma_p^2 \boldsymbol{I})).\end{aligned} \tag{30}$$

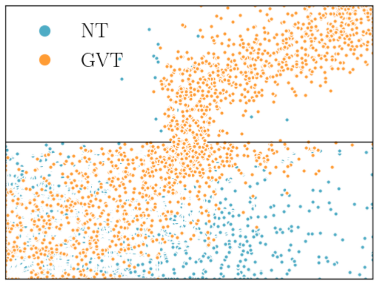
Finally, using this upper bound as objective of our optimization problem, we can then exploit the linearity of the expectation operator to obtain:

$$\begin{aligned}\widetilde{\mathcal{L}}(\boldsymbol{\xi}) = \frac{1}{C}\sum_{i=1}^{C} \mathbb{E}_{\boldsymbol{w}\sim\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}\left[\|B_{\boldsymbol{w}}\|_D^2\right] + \frac{\lambda}{N} KL(\chi^{(2)}||\chi^{(1)}) \\ + \frac{\lambda}{N}\sum_{i=1}^{C}\sum_{j=1}^{|\mathcal{W}_s|} \chi_{j,i}^{(2)} KL(\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \,||\, \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_j, \sigma_p^2 \boldsymbol{I})),\end{aligned} \tag{31}$$

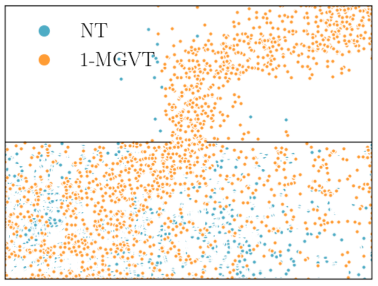
which is easily differentiable with respect to $\boldsymbol{\xi} = (\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_C, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_C)$ using the equations (23), (24), (25), (26) derived for the Gaussian case.

C Additional Details on the Experiments

In the present section, we provide details on the parameters adopted in all experiments and provide further empirical evaluation to complement the results reported in the main paper.



(a)



(b)

Figure 4: Two-Rooms Problem: (a) ϵ-greedy vs. GVT, and (b) ϵ-greedy vs. 1-MGVT

C.1 The Rooms Problem

Parameters We use ADAM as the stochastic optimizer for all algorithms. In order to train the source tasks, we directly minimize the TD error as described in Section 3.4. We use a *batch size* of 50, a *buffer size* of 50000, $\psi = 0.5$, and a learning rate $\alpha = 0.001$. Additionally, we use an ϵ-greedy policy for exploration, with ϵ linearly decaying from 1 to 0.02 in a fraction of 0.7 the maximum number of iterations.

For the transfer algorithm GVT, we set a *batch size* of 50 and a *buffer size* of 10000. We use $\psi = 0.5$, $\lambda = 10^{-4}$ and 10 weights to estimate the expected TD error. For the learning rates, $\alpha_\mu = 0.001$ for the mean of the posterior and $\alpha_L = 0.1$ to learn its Cholesky factor L. Furthermore, we restrict the minimum value reachable by the eigenvalues of these factors to be $\sigma^2_{min} = 0.0001$. In the case of MGVT we use, instead, $\lambda = 10^{-6}$, $\alpha_\mu = 0.001$, and $\alpha_L = 0.1$. Finally, we use a bandwidth $\sigma^2_p = 10^{-5}$ for the prior.

Additional Results We investigate the exploratory behavior induced by our transfer algorithms and compare it to simple ϵ-greedy exploration. In Figure 4, we show the positions visited by the agent when running 2000 iterations of the no-transfer (NT) algorithm, GVT, and 1-MGVT. Observing Figure 4a, it is possible to understand the difference between the ϵ-greedy exploration and the resulting behavior from GVT. It is noticeable that NT is not capable to lead the agent to the goal within the given iterations as most of the states visited are sparse within the first room, whereas GVT is able to concentrate more of its effort in looking for the door around the middle of the wall. After finding it, within the second room, the positions concentrate in the path leading to the goal. This is not surprising as the value function should be equal for all tasks after crossing the door. In the other case, Figure 4b shows a similar situation, but it is quite interesting to notice how sparser the exploration of 1-MGVT is with respect to GVT. Indeed, 1-MGVT is able to actually explore the right part of the first room within these iterations, which might be seen as the result of the prior model being able to capture more information than the Gaussian; hence, the higher speed-up in convergence and robustness to changes in the distribution from which target tasks are drawn. Indeed, as 1-MGVT is able to allow for more flexible exploration, it is capable to discover how to best solve the task much faster than GVT.

In Figure 5, we analyze the (online) expected return achieved by the transfer algorithms as a function of the number of source tasks used to estimate their prior. In particular, we show the resulting curves after 1000 iterations in Figure 6a and after 1950 iterations in Figure 6b. It is interesting to notice the difference between MGVT and GVT whenever there is a small number of source tasks. MGVT clearly provides faster adaptation in the presence of low prior knowledge as it can be seen from the two plots. This should be expected from the properties of the two algorithms discussed in Section 4.

Finally, we analyze the transfer performance as a function of how likely the target task is according to the prior. We consider a two-room version of the environment of Figure 2. Unlike before, we generate tasks by sampling the door position from a Gaussian with mean 5 and standard deviation 1.8, so that tasks with the door near the sides are very unlikely. Figure 6 shows the performance reached by GVT and 1-MGVT at fixed iterations as a function of how likely the target task is according to such distribution. As expected, GVT achieves poor performance on very unlikely

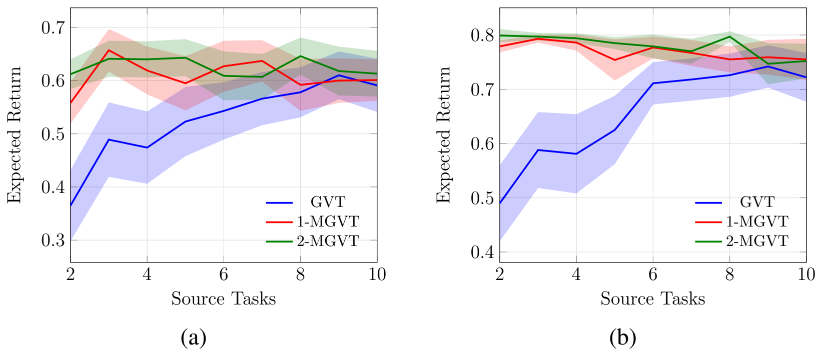


Figure 5: Expected return w.r.t. to the number of source tasks after (a) 1000 iterations, and (b) 1950 iterations.

tasks, even after many iterations. In fact, estimating a single Gaussian distribution definitely entails some information loss, especially about the unlikely tasks. On the other hand, MGVT keeps such information and, consequently, performs much better. Perhaps not surprisingly, MGVT reaches the optimal performance in $4k$ iterations no matter what task is being solved.

C.2 Classic Control

Cartpole For this environment, we generate tasks by uniformly sampling the cart mass in the range $[0.5, 1.5]$, the pole mass in $[0.1, 0.3]$, and the pole length in $[0.2, 1.0]$.

During the training of the source tasks, we use a *batch size* of 150 and a *buffer size* of 50000. Specifically, for DDQN we use a *target update frequency* of 500, *exploration fraction* of 0.35, and a learning rate $\alpha = 0.001$. We use a Multilayer Perceptron (MLP) with ReLU as activation function and a single hidden layer of 32 neurons.

For the transfer experiments, we set the *batch size* to 500, the number of *weights* sampled to approximate the expected TD error to 5, $\lambda = 0.001$, and $\psi = 0.5$. We use $\alpha_\mu = 0.001$ as the learning rate for the mean of the Gaussian posterior. For its the Cholesky factor L we use $\alpha_L = 0.0001$ and set the limit that the minimum eigenvalue may reach to $\sigma^2_{min} = 0.0001$. Additionally, for MGVT we set the variance of the prior components $\sigma^2_p = 10^{-5}$ and leave the learning rates of the posterior components' means and Cholesky factor the same as GVT.

C.2.1 Mountain Car

We generate tasks by sampling uniformly the base speed of the car in the range $[0.001, 0.0015]$.

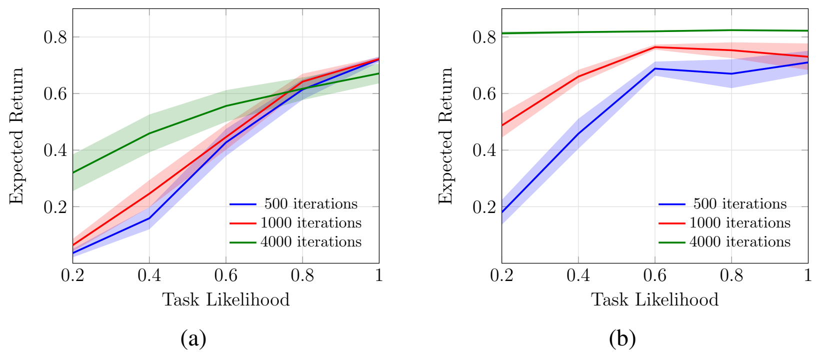
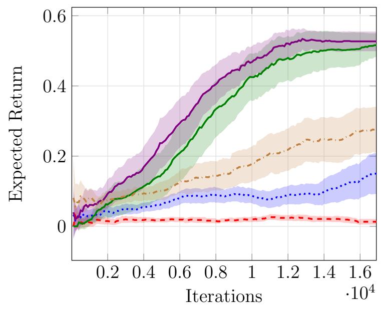
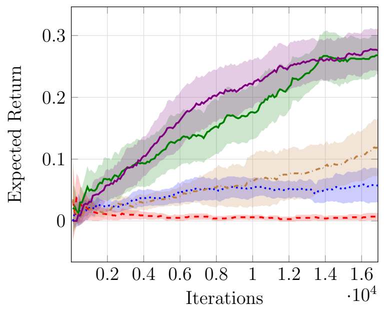


Figure 6: Expected return as a function of the (normalized) target task likelihood after a specified number of iterations.



(a)



(b)

Figure 7: Performance on (a) Maze 8i, and (b) Maze 8n.

For the sources, we train the tasks using DDQN with a *target update frequency* of 500, a *batch size* of 32, a *buffer size* of 50000 and learning rate $\alpha = 0.001$. Moreover, we set the *exploration fraction* to 0.15. We use an MLP with single hidden layer of 64 neurons with ReLU activation function.

For the transfer experiments, we set the *batch size* to 500, and use 10 *weights* to approximate the expected TD error, $\lambda = 10^{-5}$ and $\psi = 0.5$. For the learning rates, we use $\alpha_\mu = 0.001$ for the means of the Gaussians. In the case of the Cholesky factors L, we use $\alpha_L = 0.0001$ and allow the eigenvalues to reach a minimum value of $\sigma^2_{min} = 0.0001$. In the case of MGVT, additionally, we set the prior covariance to be $\sigma^2_p = 10^{-5}$.

C.3 Maze Navigation

Parameters The mazes adopted in the experiments of Section 6.3 are shown in Figure 8. Our 20 mazes have varying degree of difficulty and are designed to hold few similarities that would be useful for transferring. Moreover, we ensure 4 groups of mazes that are characterized by the same goal position.

For the experiments, we use as approximator an MLP with two hidden layers of 32 neurons with ReLU activation functions. For training the sources, we use a DDQN with a *batch size* of 70, a *buffer size* of 10000, and a *target update frequency* of 100, setting the *exploration fraction* to 0.1 and learning rate to $\alpha = 0.001$.

In the transfer experiments, we use $\psi = 0.5$, a *batch size* of 50, a *buffer size* of 50000 and use 10 sampled *weights* from the posterior to approximate the TD error. Moreover, we use $\lambda = 10^{-6}$. For GVT, in particular, we use $\alpha_\mu = 0.001$, $\alpha_L = 10^{-7}$, and set the minimum value reachable by its eigenvalues to be $\sigma_{min} = 0.0001$. In the case of MGVT, we set $\alpha_\mu = 0.001$ and $\alpha_L = 10^{-6}$. Finally, we use $\sigma^2_p = 10^{-5}$ as the prior bandwidth.

Additional Results The mazes used for the experiments in the main paper are Maze 8a (for Figure 2e) and Maze 8g (for Figure 2f). Figure 7 shows the performances achieved the algorithms on two more mazes (Maze 8i and 8n). The results are consistent with those presented in the main paper. In particular, we can appreciate that MGVT is able to provide significant speed-up in a consistent manner. On the other hand, GVT consistently fails at transferring, while FT has variable behavior in the different target mazes.

C.4 A Comparison to Fast-Adaptation Algorithms

The Rooms Problem For meta-training, we used a *meta-batch size* of 20 tasks, a *fast batch size* of 20, and a *fast learning rate* of 0.1. The meta-objective was optimized until convergence by TRPO, while the fast-adaptation step was vanilla policy gradient (REINFORCE) with baseline estimated by generalized advantage estimation. Policies were represented by neural networks using one layer of 32 neurons on top of our radial basis representation (see Section 6.1). We used the same batch size and fast learning rate at meta-testing.

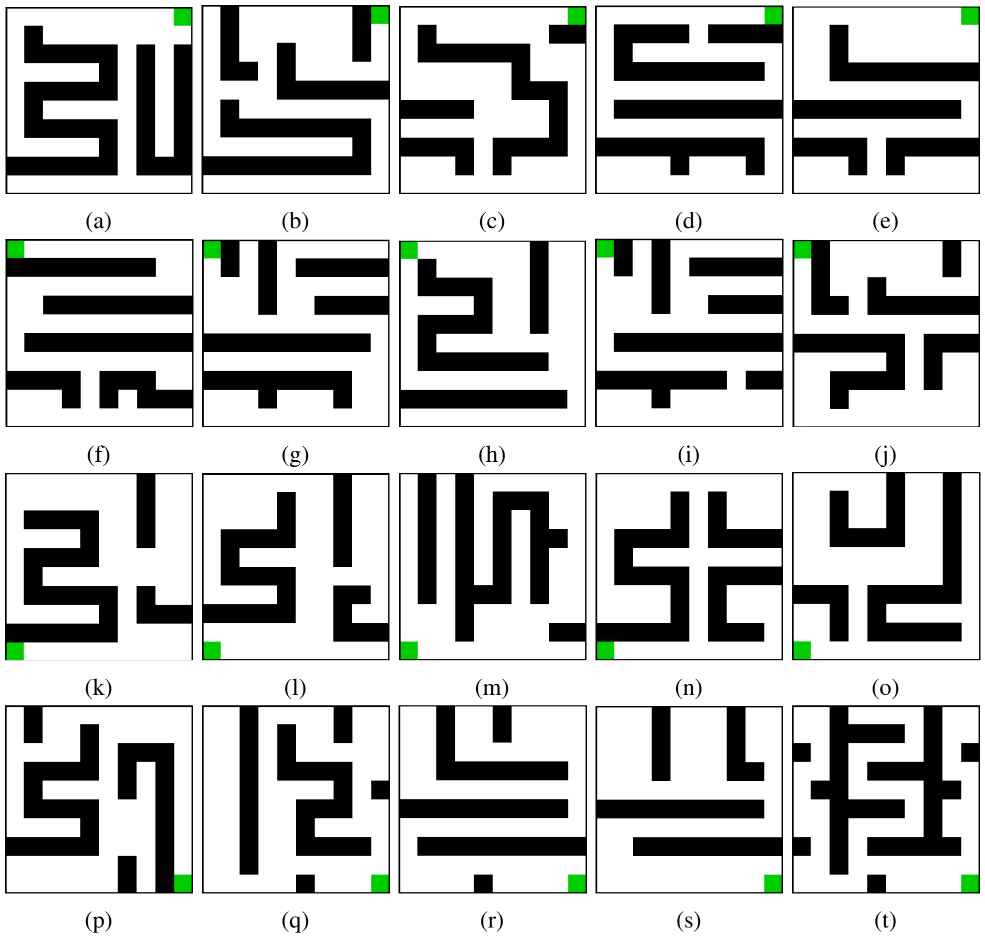


Figure 8: Set of mazes for the Maze Navigation task.

For the experiment using MAML-batch, each meta-training run sampled a fixed number of 10 source tasks before learning started. Each meta-batch was then re-sampled from a uniform distribution over these 10 fixed tasks. For the experiment using MAML-shift, we meta-trained on the full distribution by re-sampling only the position of the top door, while keeping the bottom one fixed in the middle as in our experiment of Figure 2b. We then meta-tested on the original distribution with both doors moving.

Maze Navigation For meta-training, we used a *meta-batch size* of 20 tasks, a *fast batch size* of 50, and a *fast learning rate* of 0.1. The objectives were optimized using the same algorithms as for the rooms problem, this time using a neural network with two layers of 32 neurons.

For the experiment using MAML-full, we meta-trained on the discrete distribution over all 20 mazes, while, for the experiment using MAML-batch, we used only 5 source mazes, making sure that they did not contain the target. We meta-tested only on the maze shown in Figure 8a (which is the one adopted in Figure 2e).