Time-Variant Variational Transfer for Value Functions

Giuseppe Canonaco [*1] **Andrea Soprani** [*1] **Matteo Giuliani**[1] **Andrea Castelletti**[1] **Manuel Roveri**[1]
Marcello Restelli[1]

[1]Department of Electronics Information and Bioengineering, Politecnico di Milano, Milan, Italy

Abstract

In most of the transfer learning approaches to reinforcement learning (RL) the distribution over the tasks is assumed to be stationary. Therefore, the target and source tasks are i.i.d. samples of the same distribution. Unfortunately, this assumption rarely holds in real-world conditions, e.g., due to seasonality or periodicity, evolution in the environment or faults in the sensors/actuators. In the context of this work, we consider the problem of transferring value functions through a variational method when the distribution that generates the tasks is time-variant, proposing a solution that leverages this temporal structure inherent in the task generating process. Furthermore, by means of a finite-sample analysis, the previously mentioned solution is theoretically compared to its time-invariant version. Finally, the experimental evaluation of the proposed technique is carried out on the lake Como water system representing a real-world scenario and on three different RL environments with three distinct temporal dynamics.

1 INTRODUCTION

Reinforcement Learning (RL) literature [Sutton and Barto, 2011] usually assumes the task assigned to the agent to be stationary. This assumption is not likely to hold in real-world applications, where the system to be controlled may be subject to different variations over time. For instance, in the context of finance, applying RL under the assumption of stationary markets would impair the performance of our agent in the long run due to seasonality or market evolution usually intrinsic to this kind of scenario. Similarly, while controlling a water reservoir system, the agent must be able to take into account shifts to the system's dynamics induced through the decades by climate change [Giuliani et al., 2016]. Finally, also in the context of robotic systems, stationarity assumptions could impair the attainable performances because the agent is not prepared to deal with faults affecting sensors or actuators.

Being able to relax the stationarity assumption would highly increase the applicability of RL in real-world scenarios. For this reason, the research has recently increased its interest in the direction of RL for non-stationary environments. Chandak et al. [2020] propose a policy gradient algorithm which strives to optimize the future performance of the policy assuming that smooth changes in the environment imply smooth changes in a given policy performance. Cheung et al. [2020], instead, devise a sliding window approach to RL in non-stationary Markov Decision Processes (MDPs) [Puterman, 2014] together with a bandit over RL framework to remove the dependency of their algorithm on the variation budget. Domingues et al. [2020] propose an algorithm where time-dependent kernels are leveraged in order to recover a regret upper bound for continuous non-stationary environments. Finally, Canonaco et al. [2020] use an active-adaptive scheme to deal with non-stationary environments.

In addition to the stationarity assumption, RL algorithms require a huge amount of experience to achieve effective results [Vinyals et al., 2019, Silver et al., 2018, OpenAI et al., 2019], hence, in most cases, it is impractical to directly apply an RL algorithm onto a real system because the experience collection would be incredibly slow. This translates into the need for sample efficient RL algorithms, which could be built, among all other alternatives, through Transfer Learning (TL) [Taylor and Stone, 2009, Lazaric, 2012]. In a nutshell, TL enables an RL algorithm to reuse knowledge coming from a set of already solved tasks in order to speed up the learning phase on related new ones.

Depending on what kind of knowledge representation is being transferred, we have different TL algorithms in the related literature. Therefore, in order to perform the transfer, we may have algorithms leveraging policies or options [Fer-

[*]equal contribution

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

nández and Veloso, 2006, Konidaris and Barto, 2007], samples [Taylor et al., 2008, Lazaric et al., 2008, Tirinzoni et al., 2018b, 2019], features [Barreto et al., 2017, Lehnert and Littman, 2018], value-functions [Taylor et al., 2007, Tirinzoni et al., 2018a] or parameters [Killian et al., 2017, Nagabandi et al., 2018, Du and Narasimhan, 2019]. In the classical TL setting, the source and target tasks usually come from the same distribution, hence the Bayesian framework particularly fits because we can iteratively refine the prior knowledge coming from the source tasks as more evidence from the target is collected. Following this rationale, in Wilson et al. [2007], under the assumption that the tasks share similarities in their MDP representation, a hierarchical Bayesian solution is proposed, whose main drawback lies in the need to solve an auxiliary MDP in order to perform actions on the task currently faced. Another methodology, along this line of research, has been developed in Lazaric and Ghavamzadeh [2010], which still leverages hierarchical Bayesian models, but this time assuming the tasks share commonalities through their value functions. Furthermore, in Doshi-Velez and Konidaris [2016], a Bayesian framework able to adapt optimal policies to variations of the task dynamics is developed. They use a latent variable, which, together with the state-action couple, entirely describes the system dynamics. The uncertainty over the latent variable is modeled independently of the uncertainty over the state. This limitation is overcome in the extension to their framework proposed in Killian et al. [2017]. In Perez et al. [2020] another extension to Doshi-Velez and Konidaris [2016] is proposed, which accounts for multiple variation factors that potentially also come from the reward function. A more general and efficient approach is instead developed in Tirinzoni et al. [2018a], which iteratively refines the distribution over optimal value functions by means of a variational procedure as more experience from the target task is collected.

Applying RL techniques in a scenario where sample efficiency is of paramount importance and the available historical knowledge has an intrinsic time-variant nature is incredibly challenging. Therefore, inspired by the work of Tirinzoni et al. [2018a], we will propose, for the first time in literature, a TL algorithm for RL able to model time variations in the distribution inherent to the task generating process. In addition, we will provide a theoretical comparison between our solution and the time-invariant approach of Tirinzoni et al. [2018a] promising a performance improvement in our favor. Finally, we will provide an experimental comparison of the two approaches in three different RL environments with three distinct temporal dynamics and in a real-world scenario represented by a water reservoir system.

2 PRELIMINARIES

In this section, we extend the setting introduced in Tirinzoni et al. [2018a] by adding a time-variant distribution over the tasks. We introduce basic RL concepts and some notation in Section 2.1, and we describe the variational approach to transfer in Section 2.2.

2.1 REINFORCEMENT LEARNING BACKGROUND

Let us consider a time-variant distribution $\mathcal{D}_t$ over tasks. We model each task $\mathcal{M}_t$ coming from $\mathcal{D}_t$ as a discounted MDP [Puterman, 2014], which is defined as a tuple $\mathcal{M}_t = \{\mathcal{S}, \mathcal{A}, \mathcal{P}_t, \mathcal{R}_t, p_0, \gamma\}$, where $\mathcal{S}$ and $\mathcal{A}$ represent the state space and the action space, respectively, $\mathcal{P}_t$ is the Markovian transition function with $\mathcal{P}_t(s'|s,a)$ being the transition density from state s to state s' given that the action a is executed on the environment. The reward function is defined as $\mathcal{R}_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, assumed to be uniformly bounded by a constant $R_{max} > 0$. Finally, p_0 and $\gamma \in [0, 1)$ are the initial state distribution and the discount factor, respectively. Therefore, for each task t our goal is to find a deterministic policy, $\pi_t : \mathcal{S} \to \mathcal{A}$, maximizing the long-term return over a possibly infinite horizon. In other words, this means being able to get $\pi_t^* \in \arg\max_\pi J_t(\pi)$, where $J_t(\pi) = \mathbb{E}_{\mathcal{M}_t,\pi}[\sum_{h=0}^{\infty} \gamma^h \mathcal{R}_t(s_h, a_h)]$. The optimal policy π_t^* is a greedy policy w.r.t. the optimal value function, i.e., $\pi_t^*(s) = \arg\max_a Q_t^*(s,a)$ for all s, where $Q_t^*(s,a)$ is defined as the expected return obtained by taking action a in state s and then following the optimal policy afterward. From now on, for the sake of readability, we will drop t whenever this does not imply ambiguity.

In this context, we focus on a set of parametrized value functions, $Q = \{Q_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R} | \theta \in \mathbb{R}^p\}$, also called Q-functions. We assume that each $Q_\theta \in Q$ is uniformly bounded by $\frac{R_{max}}{1-\gamma}$. An optimal Q-function is also the fixed point of the optimal Bellman operator [Puterman, 2014], which is defined as follows: $TQ_\theta(s,a) = \mathcal{R}(s,a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}}[\max_{a'} Q_\theta(s', a')]$. Therefore, a measure of optimality for a value function during learning is its Bellman error, defined as $B_\theta = TQ_\theta - Q_\theta$. Of course, if $B_\theta(s,a) = 0 \; \forall (s,a) \in \mathcal{S} \times \mathcal{A}$, then Q_θ is optimal, which implies that minimizing the squared Bellman error, $||B_\theta||_\nu^2$, is a good objective for learning (where ν is the distribution over $\mathcal{S} \times \mathcal{A}$, assumed to exist). In practice, the Bellman error is not used, since it requires two independent samples of the next state s' for each couple (s,a) [Maillard et al., 2010, Sutton and Barto, 2011]. For this reason, usually, the Bellman error is replaced by the Temporal Difference (TD) error $b(\theta)$, which corresponds to an approximation of the former using one sample $\langle s_h, a_h, r_h, s_{h+1} \rangle$, so $b_h(\theta) = r_h + \gamma \max_{a'} Q_\theta(s_{h+1}, a') - Q_\theta(s_h, a_h)$. Therefore, given a set $D = \langle s_h, a_h, r_h, s_{h+1} \rangle_{h=1}^N$, the squared TD error on D is $||B_\theta||_D^2 = \frac{1}{N} \sum_{h=1}^N b_h(\theta)^2$ (with a little abuse of notation w.r.t. the definition of the Bellman error).

2.2 VARIATIONAL TRANSFER OF VALUE FUNCTIONS

In the context described above, an optimal solution to an RL problem is a greedy policy w.r.t. an optimal value function that is parameterized by a vector of weights θ. Therefore, we can safely consider a distribution over optimal weights $p(\theta)$ instead of the distribution $\mathcal{D}$ over tasks since the latter induces a distribution over optimal Q-functions [Tirinzoni et al., 2018a]. Now, given a prior on the weights $p(\theta)$ and a dataset $D = \langle s_h, a_h, r_h, s_{h+1} \rangle_{h=1}^{N}$, the optimal Gibbs posterior that minimizes an oracle upper bound on the expected loss is defined as [Catoni, 2007]:

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

where $\Psi > 0$, which will be set to $\psi^{-1}N$, for some constant $\psi > 0$ as in Tirinzoni et al. [2018a]. It is worth noting that q becomes a Bayesian posterior every time $e^{-\Psi ||B_\theta||_D^2}$ can be interpreted as the likelihood of D. Since the integral at the denominator of Equation (1) is intractable, a variational approximation through a parametrized family of posteriors q_ξ, such that $\xi \in \Xi$, is proposed. In this way, it is sufficient to find ξ^* such that q_{ξ^*} minimizes the Kullback-Leibler (KL) divergence w.r.t. the Gibbs posterior q, which is equivalent to minimizing the (negative) evidence lower bound (ELBO) defined as [Blei et al., 2017]:

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

Therefore, the idea behind the variational transfer of value functions (as shown in Algorithm 1) is to alternate a sampling from the posterior on the optimal value function with the optimization of the posterior via $\nabla_\xi \mathcal{L}(\xi)$, assuming to have already solved a finite number of source tasks $\mathcal{M}_1 \dots \mathcal{M}_n$, which, in turn, implies having the set of their approximate solutions $\Theta_s = \{\theta_1, \dots, \theta_n\}$.[1] The weight resampling in line 8 can be interpreted as a guess on the task that we need to solve based on the current belief. After sampling, the algorithm acts on the RL problem as if such guess was correct (line 9) and then will adjust the belief based on the new experience through the optimization of the variational parameters ξ (lines 12 and 13). Notice that, as long as $\nabla_\xi \mathcal{L}(\xi)$ can be efficiently computed, any approximator for the Q-functions and any prior/posterior distributions can be used. To this end, since the $\max$ operator in the temporal difference error of Equation (2) is not differentiable, the *mellowmax* is used instead, which is differentiable and was proven to converge to the same fixed point of the optimal Bellman operator in Tirinzoni et al. [2018a]. From now on, we will denote the mellow Bellman error by $\tilde{B}_\theta$.

[1] Notice that, in the context of this work, $\mathcal{M}_1 \dots \mathcal{M}_n$ are samples coming from a time-variant distribution, hence independent but not identically distributed.

Algorithm 1 Variational Transfer

1: **Input:** Target task $\mathcal{M}_t$, source weights Θ_s
2: Estimate prior $p(\theta)$ from Θ_s
3: Initialize parameters: $\xi \leftarrow \arg\min_{\xi \in \Xi} D_{KL}(q_\xi || p)$
4: Initialize dataset $D = \emptyset$
5: **while** $True$ **do**
6: Sample initial state $s_0 \sim p_0$
7: **while** s_h is not terminal **do**
8: Sample weights $\theta \sim q_\xi(\theta)$
9: Take action $a_h = \arg\max_a Q_\theta(s_h, a)$
10: $s_{h+1} \sim \mathcal{P}_t(\cdot | s_h, a_h)$, $r_{h+1} = \mathcal{R}_t(s_h, a_h)$
11: $D \leftarrow D \cup \langle s_h, a_h, r_{h+1}, s_{h+1} \rangle$
12: Estimate $\nabla_\xi \mathcal{L}(\xi)$ using $D' \subseteq D$
13: Update ξ with $\nabla_\xi \mathcal{L}(\xi)$ using any optimizer (e.g., Kingma and Ba [2014])
14: **end while**
15: **end while**

3 TIME-VARIANT KERNEL DENSITY ESTIMATION FOR VARIATIONAL TRANSFER

In the context of this work, we will model the evolution of time over a discrete grid of asymptotically dense time instants. Let $\{\theta_{ij}\}_{j=1}^{M_i}$ be a set of independent solutions for the i^{th} family of tasks, observed at time $t_i = \frac{i}{n}$, $1 \leq i \leq n$, with $\theta_{ij} \in \mathbb{R}^p$ and $\theta_{ij} \sim P(\cdot, t_i)$. Notice that, for the sake of generality, at time t_i, we allow to tackle M_i times the i^{th} family of tasks represented by the distribution $P(\cdot, t_i)$ with associated probability density function $p(\theta, t_i)$. Furthemore, let M_i be a discrete random variable for each i. Finally, let us introduce a Time-Variant Kernel Density Estimator defined as follows:

$$\hat{p}(\theta, t) = \frac{1}{a_0(-\rho) \bar{N} \lambda |H|^{\frac{1}{2}}} \sum_{i=1}^{n} K_T \left(\frac{t - t_i}{\lambda} \right) \sum_{j=1}^{M_i} K_S(H^{-\frac{1}{2}}(\theta - \theta_{ij})), \tag{3}$$

which is based on Hall et al. [2006] and will be used as a prior to model a time-variant distribution on the solved tasks. The factor $a_0(-\rho) = \int_{-\rho}^{1} K_T(t) dt$ is used to recover consistency at the boundaries [Jones, 1993], therefore also in $t = 1$, which represents the time instant that will be used in Algorithm 1 to produce a prior for the current family of tasks. K_T is the temporal kernel, whereas K_S is the multivariate non-negative spatial kernel. Furthermore, H is the spatial kernel bandwidth matrix, $\lambda \in [0, 1]$ is the temporal kernel bandwidth, and $\bar{N} = \sum_{i=1}^{n} M_i$.

Given the following assumptions, also stated in Hall et al. [2006]:

Assumption 3.1 (Task independence)**.** *For* $1 \leq i \neq i' \leq n$,$1 \leq j \leq M_i$, *and* $1 \leq j' \leq M_{i'}$, θ_{ij} *and* $\theta_{i'j'}$ *are independent;*

Assumption 3.2 (Differentiable density function)**.** $p(\theta, t) : \mathbb{R}^p \times (0, 1] \to \mathbb{R}$ *is twice differentiable for every* t, θ*;*

Assumption 3.3 (Bounded derivatives)**.** $p(\theta, t) : \mathbb{R}^p \times (0, 1] \to \mathbb{R}$ *has two bounded derivatives;*

Assumption 3.4 (On the spatial kernel)**.** *Let* $\alpha = (\alpha_1, \ldots, \alpha_p)$ *be a multi-index, with* $\alpha_i \geq 0$ *for* $i = 1, \ldots, p$, $\theta^\alpha = \prod_{i=1}^p \theta_i^{\alpha_i}$ *for each* $\theta \in \mathbb{R}^p$, *and* N_0 *is an index set where all* p *components of each member are either* 0 *or even integers.*

$$\int_{\mathbb{R}^p} K_S(\theta)d\theta = 1, \lim_{||\theta|| \to \infty} ||\theta||^p K_S(\theta) = 0,$$
$$\int_{\mathbb{R}^p} \theta^\alpha K_S(\theta)d\theta = \mu_\alpha \leq \infty, \alpha \in N_0,$$
$$\int_{\mathbb{R}^p} \theta^\alpha K_S(\theta)d\theta = 0, \alpha \notin N_0;$$

Assumption 3.5 (On the temporal kernel)**.**

$$\int_{-c}^{c} K_T(t)dt = 1, \int_{-c}^{c} tK_T(t)dt = 0,$$
$$\int_{-c}^{c} t^2 K_T(t)dt = \sigma_T \leq \infty;$$

the following theorem holds:

Theorem 3.6 (Uniform consistency of the density estimator)**.** *Assume 3.1 - 3.5. Moreover, assume that* K_S *is spherically symmetric, with a bounded, Hölder-continuous derivative, that* K_T *is a compactly supported kernel on a subset of* $\mathbb{R}$, *that all the* M_i*s are independent and identically distributed random variables with mean* $m > 0$ *and all moments finite, independent of the* θ_{ij}*s. Take* H *and* λ *such that* $|H|^{\frac{1}{2}}(n) \to 0$, $\lambda(n) \to 0$ *and* $n^{1-\epsilon}|H|^{\frac{1}{2}}\lambda \to \infty$ *for some* $\epsilon > 0$ *as* $n \to \infty$, *then*

$$\hat{p}(\theta, t) = p(\theta, t) + O\big[(\bar{N}|H|^{\frac{1}{2}}\lambda)^{-\frac{1}{2}}(\log n)^{\frac{1}{2}} + tr(H) + \lambda\big]$$

uniformly in $(\theta, t) \in \mathcal{K} \times \mathcal{I}$, *with probability 1, where* $\mathcal{K}$ *is a compact subset of* $\mathbb{R}^p$ *and* $\mathcal{I}$ *is a compact subset of* $(0, 1]$.

A proof of the above theorem is shown in Appendix A and leverages the same approach as in Hall et al. [2006] being a weaker version, in terms of convergence rate, of their Theorem 1. This weakening was necessary to obtain an upper bound in closed-form expression of the KL-divergence between the prior and the posterior in Equation (2). [2] Indeed, if we choose $q_\xi(\theta) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{N}(\theta|\mu_k, \Sigma_k)$, with variational parameter $\xi = (\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K)$, and we choose K_S as a Gaussian kernel, then for a fixed time instant t our prior is a mixture of Gaussians with non-uniform weights. Therefore, through the upper bound on the KL-divergence shown in Appendix B which leverages Hershey and Olsen [2007], we have that the ELBO upper bounds the KL-divergence between the approximate and the exact posterior. Since the covariance matrices of the posterior must be positive definite, we will learn the factor L of their Cholesky decomposition as in Tirinzoni et al. [2018a].

Let us comment on the previous assumptions and their limiting effects on applications. For what concerns Assumptions 3.4 and 3.5, they do not pose any limit, since, as we know from kernel density estimation theory [Wand and Jones, 1994], the kernel type is not relevant for a good estimate of the density. Assumptions 3.2 and 3.3, instead, are necessary to have some regularity allowing the time-variant distribution to be learned (without those assumptions the kernel density estimator would not be consistent). The range of time-variant distributions where our approach will be theoretically effective is reduced due to Assumptions 3.2 and 3.3, but remains still relevant from an application perspective since it allows to solve real problems such as controlling the lake Como water system as shown in Section 6.6.

4 FINITE-SAMPLE ANALYSIS

In order to provide a finite sample analysis of Algorithm 1 based on the prior of Section 3, we extend Theorem 2 of Tirinzoni et al. [2018a] to deal with time-variant contexts, enabling also a theoretical comparison between the two respective versions of Algorithm 1. Therefore, considering the family of linearly parametrized value functions, $Q_\theta(s, a) = \theta^T \phi(s, a)$, having bounded weights $||\theta||_2 \leq \theta_{max}$ and uniformly bounded features $||\phi(s, a)||_2 \leq \phi_{max}$, and assuming that only finite data are available, we can bound the expected mellow Bellman error under the variational distribution minimizing Equation (2) for any fixed target task $\mathcal{M}_t$ through the following theorem.

Theorem 4.1 (Bound on the expected mellow Bellman error)**.** *Let* $\hat{\xi}$ *be the variational parameter minimizing Equation* (2) *on a dataset* D *of* N *i.i.d. samples distributed according to* $\mathcal{M}_t$ *and* ν. *Moreover, let* $\theta^* = \arg\inf_\theta ||\tilde{B}_\theta||_\nu^2$ *and define* $v(\theta^*) = \mathbb{E}_{\mathcal{N}(\theta^*, \frac{1}{N}I)}[v(\theta)]$, *with* $v(\theta) = \mathbb{E}_\nu[\mathbb{V}ar_{\mathcal{P}_t}[\tilde{b}(\theta)]]$, *where* $\tilde{b}(\theta) = r + \gamma mellow\text{-}max_{a'} Q_\theta(s', a') - Q_\theta(s, a)$. *Then, there exist constants* c_1, c_2, c_3 *such that with probability at least* $1 - \delta$ *over the choice of* D*:*

$$\mathbb{E}_{q_{\hat{\xi}}}\left[\left|\left|\tilde{B}_\theta\right|\right|_\nu^2\right] \leq 2\left|\left|\tilde{B}_{\theta^*}\right|\right|_\nu^2 + v(\theta^*) + c_1\sqrt{\frac{\log\frac{2}{\delta}}{N}} + \frac{\boldsymbol{c_2} + \psi p \log N + \psi\boldsymbol{\varphi(\Theta_s)}}{N} + \frac{c_3}{N^2},$$

[2] This upper bound cannot be obtained by directly using the estimator proposed in Hall et al. [2006] because of the negative weights associated with the spatial kernel.

where

$$\varphi(\Theta_s) = \frac{1}{\sigma^2} \sum_{j:\theta_j \in \Theta_s} \zeta(j) \text{ with} \quad (4)$$

$$\zeta(j) = \frac{c_j^{\hat{p}} e^{-\beta||\theta^* - \theta_j||}}{\sum_{j':\theta_{j'} \in \Theta_s} c_{j'}^{\hat{p}} e^{-\beta||\theta^* - \theta_{j'}||}} ||\theta^* - \theta_j||,$$

assuming the matrix H of Equation (3) *to be an isotropic covariance matrix with variance σ^2, $\beta = \frac{1}{2\sigma^2}$ and $c_j^{\hat{p}}$ the weight assigned to the j^{th} prior component. Furthermore, we are assuming $M_i = 1$ for each i in our estimator.*

The above theorem shows the difference between the plain mixture version of Algorithm 1 [Tirinzoni et al., 2018a] and our solution which lies in the constant c_2 and in the term $\varphi(\Theta_s)$. Looking at $\varphi(\Theta_s)$, we can shed some light on the different theoretical properties of the two versions. More specifically, in the plain mixture version, the factor $c_j^{\hat{p}}$ does not appear, which implies uniform importance of the source solutions Θ_s w.r.t. the target task. On the other hand, in our version of the algorithm, we can give different importance to each source solution through $c_j^{\hat{p}}$. Increasing the weight of sources similar to the target will reduce $\varphi(\Theta_s)$. In our time-variant scenario, this weight will be greater on more recent solutions than older ones, potentially enabling a reduction of the term $\varphi(\Theta_s)$ w.r.t. the time-invariant version. For what concern c_2, the main difference is due to a different expression of the KL-divergence upper bound, and the usage of non-uniform weights. A proof for the above theorem together with the definition of all the constants is provided in Appendix C.

5 RELATED WORKS

Our work is inspired by Tirinzoni et al. [2018a]. Differently from them, we leverage a time-variant structure underlying the task generating process, which lets us cope with time-variant scenarios. A theoretical comparison between the two solutions is available in Section 4 through Theorem 4.1, whereas the experimental comparison is in Section 6. Furthermore, our work relates both to Wilson et al. [2007], which deals with finite MDPs, and to Lazaric and Ghavamzadeh [2010], which leverages the commonalities in the value function structure, but, in contrast to our work, they do not account for a time-variant distribution. The work done in Doshi-Velez and Konidaris [2016], Killian et al. [2017], Perez et al. [2020] leverage latent embeddings in order to model variations between tasks, which eventually are solved through a model-based RL algorithm, while we propose a model-free approach.

Another related work is Hall and Willett [2015], in which the authors develop a theoretical low-regret algorithm accounting for potential underlying dynamics. However, they use the online learning framework, whereas we are working in a transfer learning setting. Furthermore, in Du and Narasimhan [2019], videos are used to learn a prior (mainly to model the physical dynamics) which is incorporated into a model-based RL algorithm. In Yang et al. [2020], a single-episode policy-transfer methodology was developed leveraging variational inference, but for contexts in which the differences in dynamics can be identified in the early steps of an episode. In the context of supervised learning, our work relates also to Minku and Yao [2014], which proposes a transfer learning mechanism in the context of a possibly non-stationary environment through a weighting approach, and Du et al. [2019], which, instead, do transfer in non-stationary environments through ensembles. Finally, in the meta-learning framework, Khodak et al. [2019] is able to consider optimal initializations varying through time, Mendonca et al. [2020] provide robustness to distributional shifts during meta-testing through an experience relabeling mechanism, and Fu et al. [2020] develop a Context-based Meta-RL algorithm which leverages contrastive learning and an information-gain-based exploration strategy showing good performances in out-of-distribution tasks. These last three approaches are meta-learning based, while our work considers a transfer learning setting.

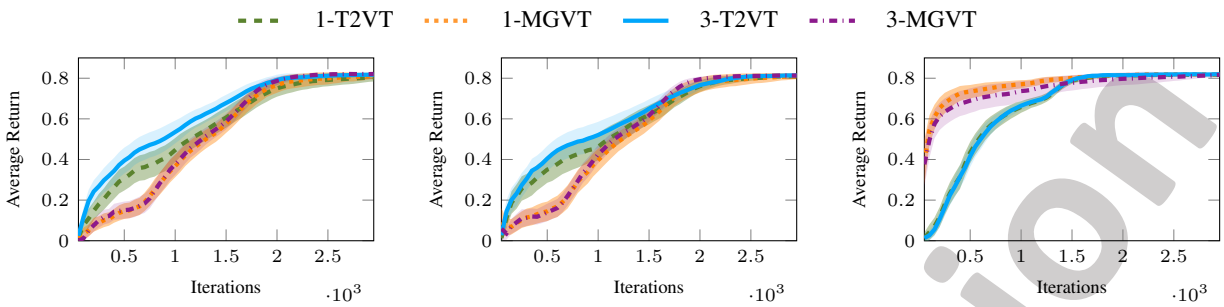
6 EXPERIMENTS

In this section, we compare our time-variant solution for transfer learning with the associated non-time-variant solution of Tirinzoni et al. [2018a] in three different domains with three different temporal dynamics and a real-world scenario.[3] The first three domains were chosen from Tirinzoni et al. [2018a] (adding the temporal dynamics) in order to enable a faithful comparison. The real-world problem consists in controlling a water reservoir system, where the temporal dynamic is due to the climate change across the decades. A detailed description of the used parameters together with the analytical expression of the employed temporal dynamics are provided in Appendix D.

6.1 TEMPORAL DYNAMICS

The distribution over the tasks is usually a given distribution over one or more parameters defining the task itself. Therefore, in order to obtain time variance in such distribution, we will change its mean over time according to a certain dynamic. These dynamics are linear, polynomial, and sinusoidal. In the context of these experiments, we will use a time-variant Gaussian distribution, clipping its realizations within the domain of the task-defining parameters (for further details see Appendix D). Instead, in the water reservoir system, the temporal dynamic is inherent to the data and, as already mentioned, due to climate change.

[3] Code at https://github.com/AndreaSoprani/T2VT-RL.



(a) 2-rooms polynomial dynamic. (b) 2-rooms linear dynamic. (c) 2-rooms sin dynamic.

Figure 1: Average return achived by the algorithms with 95% confidence intervals computed using 50 independent runs.

6.2 TWO-ROOMS ENVIRONMENT

In this setting, we have an agent navigating two rooms separated by a wall (see Figure 2). The agent starts from the bottom-left corner and must reach the opposite one. The only way to reach this goal is to pass through the door whose position is unknown to the agent. The actions available to the agent are *up*, *down*, *left*, and *right*, which let the agent to move in the respective directions by one position, unless he/she hits a wall (in this last case the position remains unchanged). Furthermore, the final position of the agent after a movement action is altered by a Gaussian noise $\mathcal{N}(0, 0.2)$. The state space is modeled through a 10×10 continuous grid. Finally, the reward function is 0 everywhere except in the goal state, where it is 1. The discount factor $\gamma = 0.99$. For this setting, we used linearly parametrized Q-functions with 121 evenly-spaced radial basis features.

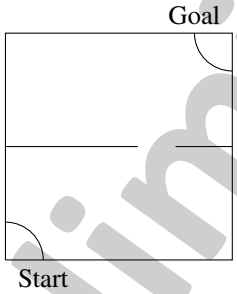


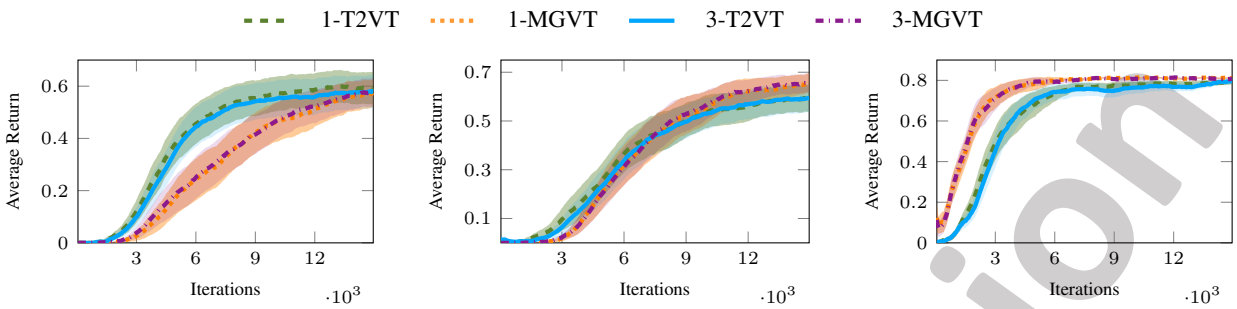
Figure 2: 2-Rooms Environment.

We considered source tasks taken at ten different time instants to learn the target, corresponding to the eleventh instant of time. We sampled five tasks from the time-variant distribution for each $i = 1, \ldots, 11$. The parameter that defines the task is the door location, hence the time-variant distribution is over that parameter, as we mentioned above. We solve all the source tasks by directly minimizing the TD error, then we exploit the learned solutions to perform the transfer over the target. We compare our time-variant variational transfer algorithm leveraging a c-components posterior (c-T2VT) with the mixture of Gaussian variational transfer using still c-components (c-MGVT) [Tirinzoni et al., 2018a]. More specifically, our time-variant prior will consider the source task solutions as equally spaced samples in the time interval $[0, 1]$, moreover, in order to perform transfer to the eleventh task, we will use the distribution provided by our estimator for $t = 1$. Finally, the temporal kernel will be Epanechnikov [Epanechnikov, 1969, Wand and Jones, 1994] in the context of all the experiments.

The average return over the last 50 learning episodes as a function of the number of training iterations is shown in Figure 1, for the time dynamics mentioned in Section 6.1. Each learning curve is computed using 50 independent runs, each of which resamples both the source and target tasks, with 95% confidence intervals. For polynomial and linear dynamics, we can see an advantage of our technique in the early learning iterations. The sinusoidal dynamic is designed to disadvantage our technique w.r.t. c-MGVT, indeed, it makes the target task appear twice in the sources. This fact inevitably favors c-MGVT, which will give a higher weight to those source tasks being sampled from the same distribution of the target. Observe that c-MGVT gives uniform weights to all the source tasks, hence increasing the replicas importance within the sources, whereas c-T2VT gives increasing weights the more recent the source solution.

6.3 THREE-ROOMS ENVIRONMENT

This scenario is an extension of the previous one, hence the environmental settings remain the same, the agent has just an additional wall to traverse in order to reach his/her goal. Of course, the position of the door for this additional wall is still unknown to the agent. To increase the complexity of the dynamics, we let the two doors move in opposite directions starting at the two far ends of the room, each door with the same dynamic. In Figure 3, we compare c-T2VT with c-MGVT using still 95% confidence intervals. As for the polynomial dynamics, we observe a better performance of c-T2VT w.r.t. c-MGVT, whereas, for the sinusoidal dynamics, we have essentially the same behavior as in the two rooms



(a) 3-rooms polynomial dynamic. (b) 3-rooms linear dynamic. (c) 3-rooms sin dynamic.

Figure 3: Average return achived by the algorithms with 95% confidence intervals computed using 50 independent runs.

environment. Finally, in the linear dynamics, we observe that the difference in performance between the two algorithms is not statistically significant.

6.4 MOUNTAIN CAR

In this section, we consider a classic control problem known as Mountain Car [Sutton and Barto, 2011]. In Mountain Car, the agent is an underpowered car whose goal is to escape a valley. Due to the limitation to its engine, the car has to drive up along the two slopes of the valley in order to gain sufficient momentum to overcome gravity (further details in Appendix D.4). In Figure 4, we have a comparison between c-T2VT and c-MGVT on the three proposed dynamics. We observe a statistically significant improvement in the polynomial dynamics across the whole learning process for c-T2VT, which also extends to the sinusoidal dynamic case. We would like to highlight the differences between the sinusoidal dynamic in Mountain Car w.r.t. the previous two environments. Here our algorithm is able to perform better due to a bias-variance trade-off in its favor. More specifically, the value functions vary more rapidly in Mountain Car than in the room environments w.r.t. a change in the task-defining parameters. Therefore, our prior estimator has less variance, since it considers only the latest sources, at the cost of a bias increase, because it discards the first task, which has the same parametrization as the target (due to the periodicity of the sin function). c-MGVT considers all the source tasks with the same weight, hence it is able to consider the tasks that have an equivalent parametrization to the target, but are farther behind in the sources' history. This fact decreases the bias at the cost of accepting a greater variance in the prior estimation. In Mountain Car, the trade-off proposed by our algorithm is more advantageous than the one proposed by c-MGVT due to the more rapidly changing behavior of the value functions. As for the linear dynamics, we do not observe a statistically significant difference in performance between the two algorithms, even though the average of 1-T2VT is the best one.

6.5 CHOOSING λ THROUGH MAXIMUM-LIKELIHOOD

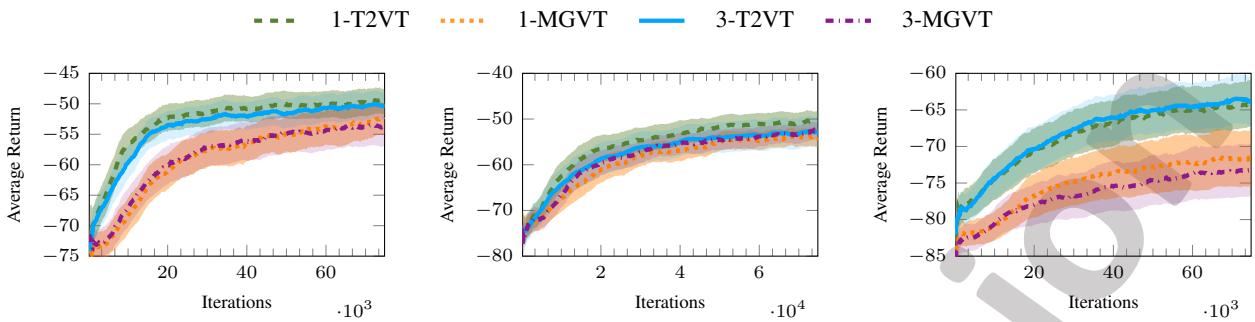
Up to now, we have kept λ and H at given constant values in order to provide a more faithful comparison between c-T2VT and c-MGVT (H was the same in the two algorithms whereas λ was set to 0.3333 leveraging the intuition that the more recent tasks were more important than the older ones). Of course, from the theory of Kernel Density Estimation [Wand and Jones, 1994], we know that appropriately setting these parameters is crucial to get a good estimate of the density. Therefore, an automatic data-driven approach would be desirable. In the context of this work, we propose a maximum likelihood scheme (assuming $M_i = 1 \; \forall \; i$):

$$\arg\max_{\lambda} L_\lambda = \prod_{h=1}^{n} \frac{\hat{p}_{-h}(\theta_h, t_h)}{\hat{p}_{-h}(t_h)}, \text{ where} \tag{5}$$

$$\hat{p}_{-h}(\theta_h, t_h) = \frac{1}{a_0(-\rho)(\bar{N}-1)\lambda|H|^{\frac{1}{2}}} \sum_{i \neq h} K_T\left(\frac{t_h - t_i}{\lambda}\right) K_S(H^{-\frac{1}{2}}(\theta_h - \theta_i))$$

$$\hat{p}_{-h}(t_h) = \int \hat{p}_{-h}(\theta_h, t_h) d\theta_h = \frac{1}{a_0(-\rho)(\bar{N}-1)\lambda} \sum_{i \neq h} K_T\left(\frac{t_h - t_i}{\lambda}\right).$$

In Appendix D.3, we report the performance achievable with this approach together with a sensitivity analysis w.r.t. the parameter λ for every environment discussed so far. Furthermore, still in Appendix D.3, we include some implementation details related to the optimization of the likelihood function in Equation (5). Note that, in accordance with what has been done in Tirinzoni et al. [2018a], the spatial bandwidth is set to $10^{-5}I$ which would prevent us from successfully optimizing Equation (5) due to numerical issues, hence we set it to I in order to select the best lambda.



(a) Mountain Car polynomial dynamic. (b) Mountain Car linear dynamic. (c) Mountain Car sin dynamic.

Figure 4: Average return achived by the algorithms with 95% confidence intervals computed using 50 independent runs.

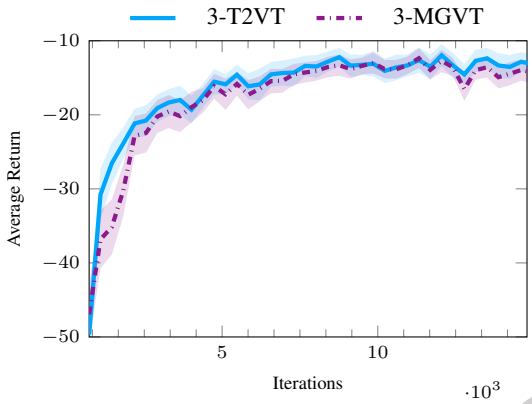


Figure 5: Average return achived by the algorithms with 95% confidence intervals computed using 100 independent runs on the lake Como environment.

6.6 REAL-WORLD SCENARIO: CONTROLLING THE LAKE COMO WATER SYSTEM

Lake Como is the third largest lake in Italy thanks to its surface area of 146 km^2 and the fifth deepest in Europe with its maximum depth at 425 meters. It is a lake of glacial origin which has been regulated since 1946 by a human operator to prevent flooding along the lake shores and supply water to the downstream users, which are composed of 4 irrigation districts (total irrigated area of 1400 km^2) and 9 run-of-river power plants (total capacity of 92 MW).

To design the optimal lake operation, we can leverage RL to find an optimal control policy π^* for the water reservoir system of lake Como [Castelletti et al., 2010]. For this setting, the state space includes the day of the year and lake storage volume. The first is encoded as sine and cosine functions of $2\pi\frac{t}{period}$, where t is the day and $period$ is the year's length, which enables accounting for the time-dependency and cyclostationarity of the system, and, consequently, of the operating policy. The second one is governed by the mass conservation equation $v_{t+1} = v_t + i_{t+1} - \varrho_{t+1}$, where i_{t+1} is the net inflow volume in the time interval $[t, t+1)$ and $\varrho_{t+1} = f_t(v_t, a_t, i_{t+1})$ is the actual release accomplished by the system. The release function $f_t(\cdot)$ accounts for physical and normative constraints on the storage and release [Soncini-Sessa et al., 2007]. Observe that, the actual release depends on the previous storage volume, the policy's decision (corresponding to the amount of water the agent would like the system to release), and the inflow i_{t+1}, which is influencing the system throughout the whole time period $[t, t+1)$. The reward function is composed of three main costs related to water demand, flooding events, and actions feasibility. It is noteworthy to point out the fact that, being the net inflow volume composed of historical data, this setting constitutes an environment incredibly close to the real-world system. Further details are in Appendix D.4.

In order to successfully apply RL onto the lake Como water system, we need to carefully take into account the time-variant nature of the net inflow volume, which has changed much since the mid $40s$ due to the climate change our planet is currently undergoing [Giuliani et al., 2016]. Furthermore, if we were to leverage an RL algorithm to control the water reservoir system, TL would be a must to reduce the amount of data needed to reach an optimal behavior and to mitigate the usage of sub-optimal policies onto the system. Since we do not have another time-variant transfer algorithm for RL in the literature, we will again compare T2VT with MGVT to analyze the benefit of accounting for time variance.

Historical data span from 1946 to 2006 and will be split into 12 years chunks, each one representing a task. Hence, the sources will consist of the tasks $[1946, 1957]$, $[1958, 1969]$, $[1970, 1981]$, $[1982, 1993]$, whereas the target is represented by the task $[1994, 2006]$. Results are reported in Figure 5, where we compare 3-T2VT coupled with the maximum-likelihood approach of Section 6.5 against 3-MGVT. As we can see, there is a beneficial effect on the optimization process by accounting for time-variance in the source solutions. Indeed, our algorithm performs better than 3-MGVT especially in the early iterations where the performance difference is statistically significant.

7 DISCUSSION AND CONCLUSIONS

In this paper, we presented a time-variant approach for transferring value functions through a variational scheme. In order to deal with a time-variant distribution of the tasks, we have devised a suitable estimator for the prior to be used in the variational scheme providing its uniform consistency over a compact subset of $\mathbb{R}^p \times (0, 1]$. We have, then, provided a finite sample analysis on the performance of the variational transfer algorithm based on our estimator, enabling a theoretical comparison with the time-invariant version of Tirinzoni et al. [2018a]. Finally, we have experimentally proved our algorithm abilities to deal with time-variant distributions even in a real-world scenario represented by the lake Como water system.

Notice that discriminating the source tasks w.r.t. time is an additional step that brings transfer learning approaches and learning in non-stationary environments a bit closer together [Minku, 2019]. It is also important to highlight the fact that, instead of considering time, we could switch to any other variable (e.g., the task-defining parameter) as long as it is available together with each source solution and we can properly remap it into $(0, 1]$. This could enable us to leverage completely different structures in order to perform transfer to the target task. Finally, we would also like to highlight the possibility of using this time-variant transfer paradigm in lifelong learning scenarios [Chen and Liu, 2018] as a potential future direction.

References

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.

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.

Giuseppe Canonaco, Marcello Restelli, and Manuel Roveri. Model-free non-stationarity detection and adaptation in reinforcement learning. In *European Conference on Artificial Intelligence*, pages 1047–1054. IOS Press, 2020.

A Castelletti, Stefano Galelli, Marcello Restelli, and Rodolfo Soncini-Sessa. Tree-based reinforcement learning for optimal water reservoir operation. *Water Resources Research*, 46(9), 2010.

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

Yash Chandak, Georgios Theocharous, Shiv Shankar, Sridhar Mahadevan, Martha White, and Philip S Thomas. Optimizing for the future in non-stationary mdps. *arXiv preprint arXiv:2005.08158*, 2020.

Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. Reinforcement learning for non-stationary markov decision processes: The blessing of (more) optimism. In *International Conference on Machine Learning*, pages 1843–1854. PMLR, 2020.

Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Emilie Kaufmann, and Michal Valko. A kernel-based approach to non-stationary reinforcement learning in metric spaces. *arXiv preprint arXiv:2007.05078*, 2020.

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. NIH Public Access, 2016.

Honghui Du, Leandro L Minku, and Huiyu Zhou. Multi-source transfer learning for non-stationary environments. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

Yilun Du and Karthic Narasimhan. Task-agnostic dynamics priors for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1696–1705, 2019.

Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

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, 2006.

Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. *arXiv preprint arXiv:2009.13891*, 2020.

Matteo Giuliani, Yu Li, Andrea Castelletti, and C Gandolfi. A coupled human-natural systems analysis of irrigated agriculture under changing climate. *Water Resources Research*, 52(9):6928–6947, 2016.

Eric C Hall and Rebecca M Willett. Online convex optimization in dynamic environments. *IEEE Journal of Selected Topics in Signal Processing*, 9(4):647–662, 2015.

Peter Hall, Hans-Georg Müller, and Ping-Shi Wu. Real-time density and mode estimation with application to time-dynamic mode tracking. *Journal of Computational and Graphical Statistics*, 15(1):82–100, 2006.

John R Hershey and Peder A Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–317. IEEE, 2007.

M Chris Jones. Simple boundary correction for kernel density estimation. *Statistics and computing*, 3(3):135–146, 1993.

Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, pages 5915–5926, 2019.

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.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, volume 7, pages 895–900, 2007.

Alessandro Lazaric. Transfer in reinforcement learning: a framework and a survey. In *Reinforcement Learning*, pages 143–173. Springer, 2012.

Alessandro Lazaric and Mohammad Ghavamzadeh. Bayesian multi-task reinforcement learning. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 599–606, 2010.

Alessandro Lazaric, Marcello Restelli, and Andrea Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 544–551, 2008.

Lucas Lehnert and Michael L Littman. Transfer with model features in reinforcement learning. *arXiv preprint arXiv:1807.01736*, 2018.

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.

Russell Mendonca, Xinyang Geng, Chelsea Finn, and Sergey Levine. Meta-reinforcement learning robust to distributional shift via model identification and experience relabeling. *arXiv preprint arXiv:2006.07178*, 2020.

Leandro L Minku. Transfer learning in non-stationary environments. In *Learning from Data Streams in Evolving Environments*, pages 13–37. Springer, 2019.

Leandro L Minku and Xin Yao. How to make best use of cross-company data in software effort estimation? In *Proceedings of the 36th International Conference on Software Engineering*, pages 446–456, 2014.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. 2019. URL `https://arxiv.org/abs/1912.06680`.

Christian F Perez, Felipe Petroski Such, and Theofanis Karaletsos. Generalized hidden parameter mdps transferable model-based rl in a handful of trials. *arXiv preprint arXiv:2002.03072*, 2020.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362 (6419):1140–1144, 2018.

R. Soncini-Sessa, A. Castelletti, and E. Weber. *Integrated and participatory water resources management: Theory*. Elsevier, Amsterdam, NL, 2007.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 2011.

Matthew E. Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(56):1633–1685, 2009. URL `http://jmlr.org/papers/v10/taylor09a.html`.

Matthew E Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(Sep): 2125–2167, 2007.

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.

Andrea Tirinzoni, Rafael Rodriguez Sanchez, and Marcello Restelli. Transfer of value functions via variational methods. *Advances in Neural Information Processing Systems*, 31:6179–6189, 2018a.

Andrea Tirinzoni, Andrea Sessa, Matteo Pirotta, and Marcello Restelli. Importance weighted transfer of samples in reinforcement learning. In *International Conference on Machine Learning*, pages 4936–4945. PMLR, 2018b.

Andrea Tirinzoni, Mattia Salvini, and Marcello Restelli. Transfer of samples in policy search via multiple importance sampling. In *International Conference on Machine Learning*, pages 6264–6274, 2019.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, 2019.

Matt P Wand and M Chris Jones. *Kernel smoothing*. CRC press, 1994.

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, 2007.

Jiachen Yang, Brenden Petersen, Hongyuan Zha, and Daniel Faissol. Single episode policy transfer in reinforcement learning. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rJeQoCNYDS`.