Importance Weighted Transfer of Samples in Reinforcement Learning

Andrea Tirinzoni [1] **Andrea Sessa** [1] **Matteo Pirotta** [2] **Marcello Restelli** [1]

Abstract

We consider the transfer of experience samples (i.e., tuples $\langle s, a, s', r \rangle$) in reinforcement learning (RL), collected from a set of source tasks to improve the learning process in a given target task. Most of the related approaches focus on selecting the most relevant source samples for solving the target task, but then all the transferred samples are used without considering anymore the discrepancies between the task models. In this paper, we propose a model-based technique that automatically estimates the relevance (importance weight) of each source sample for solving the target task. In the proposed approach, all the samples are transferred and used by a batch RL algorithm to solve the target task, but their contribution to the learning process is proportional to their importance weight. By extending the results for importance weighting provided in supervised learning literature, we develop a finite-sample analysis of the proposed batch RL algorithm. Furthermore, we empirically compare the proposed algorithm to state-of-the-art approaches, showing that it achieves better learning performance and is very robust to negative transfer, even when some source tasks are significantly different from the target task.

1. Introduction

The goal of transfer in Reinforcement Learning (RL) (Sutton & Barto, 1998) is to speed-up RL algorithms by reusing knowledge obtained from a set of previously learned tasks. The intuition is that the experience made by learning *source tasks* might be useful for solving a related, but different, *target task*. Transfer across multiple tasks may be achieved in different ways. The available approaches differ in the type of information transferred (e.g., samples, value functions, parameters, policies, etc.) and in the criteria used to establish whether such knowledge could be beneficial for solving the target or not.

This work focuses on the problem of transferring samples from a set of source MDPs to augment the dataset used to learn the target MDP. To motivate our approach, consider a typical learning scenario where samples are costly to obtain. This is often the case in robotics applications, where the interaction with the real environment could be extremely time-consuming, thus reducing the number of samples available. The typical remedy of adopting a simulator often leads to sub-optimal solutions due to the differences with respect to the real environment. A more effective approach is to transfer the simulated samples to speed-up learning in the target task.

The transfer of samples has been widely studied in the supervised learning community. In particular, Crammer et al. (2008) formalized the problem from a theoretical perspective and provided generalization bounds for the transfer scenario. An interesting result is a trade-off between the number of tasks from which to transfer and the total number of samples. In RL, Taylor et al. (2008) and Lazaric et al. (2008) proposed almost simultaneously methods to transfer single samples. While the former method focused on a model-based approach, the latter one proposed a selective approach to transfer samples into a batch RL algorithm (e.g., Fitted Q-Iteration (Ernst et al., 2005)). Furthermore, Lazaric et al. (2008) considered a model-free approach to compute a similarity measure between tasks, which was used to decide which samples to transfer. More recently, Lazaric & Restelli (2011) analyzed the transfer of samples in batch RL from a theoretical perspective, demonstrating again the trade-off between the total number of samples and the number of tasks from which to transfer. Finally, Laroche & Barlier (2017) proposed a way to transfer all the samples to augment the dataset used by Fitted Q-Iteration. The limitation of this approach resides in the restrictive assumption that all the tasks are assumed to share the same transition dynamics and differ only in the reward function. For a survey on transfer in RL, we refer the reader to (Taylor & Stone, 2009; Lazaric, 2012).

One of the main drawbacks of many previous works is that, even after a detailed selection, transferred samples are used

[1]Politecnico di Milano, Milan, Italy [2]SequeL Team, INRIA Lille, France. Correspondence to: Andrea Tirinzoni <andrea.tirinzoni@polimi.it>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

in the target task *without* accounting for the differences between the original (source) MDP and the target one, thus introducing a bias even in the asymptotic case. In this paper, we present a novel approach to transfer samples into a batch RL algorithm. Unlike other works, we do not assume any particular similarity between tasks besides a shared state-action space, and we develop a new model-based methodology to automatically select the relevance (importance weight) of each sample. Existing algorithms for transferring across different state-action spaces (e.g., Taylor et al., 2007) can be straightforwardly combined to our method. Our approach transfers all the samples, but their impact in solving the target task is proportional to their importance weight. To compute the importance weight of each sample, we rely on a non-parametric estimate of the MDP structure. In particular, we adopt Gaussian processes (Rasmussen & Williams, 2006) to estimate the reward and state transition models of the source and target tasks from samples. Then, we propose a robust way to compute two sets of importance weights, one for the reward model and one for the transition model. We introduce an approximate value iteration algorithm based on Fitted Q-iteration that uses such weights to account for the distribution shift introduced by the different MDPs, thus implicitly selecting which samples have higher priority based on their likelihood to be generated from the target MDP. We provide a theoretical analysis showing the asymptotic correctness of our approach and an empirical evaluation on two classical RL domains and a real-world task.

2. Preliminaries

In this section, we start by introducing our mathematical notation. Then, we recall concepts of Markov decision processes and approximate value iteration. Finally, we formalize the transfer settings considered in this work.

Notation. For a measurable space $\langle \Omega, \sigma_\Omega \rangle$, we denote by $\Delta(\Omega)$ the set of probability measures over σ_Ω and by $\mathcal{B}(\Omega, L)$ the space of measurable functions over Ω bounded by $0 < L < \infty$, i.e., $\forall f \in \mathcal{B}(\Omega, L), \forall x,\ |f(x)| \leq L$. Given a probability measure μ, we define the ℓ_p-norm of a measurable function f as $\|f\|_{p,\mu} = \left(\int |f|^p d\mu\right)^{1/p}$. Let $z_{1:N}$ be a $\mathcal{Z}$-valued sequence $(z_1, \dots, z_N)$ for some space $\mathcal{Z}$. For $\mathcal{D}_N = z_{1:N}$, the empirical norm of a function $f : \mathcal{Z} \to \mathbb{R}$ is $\|f\|^p_{p,\mathcal{D}_N} \coloneqq \frac{1}{N}\sum_{i=1}^N |f(z_i)|^p$. Note that when $Z_i \sim \mu$, we have that $\mathbb{E}[\|f\|^p_{p,\mathcal{D}_N}] = \|f\|^p_{p,\mu}$. Whenever the subscript p is dropped, we implicitly consider the ℓ_2-norm.

Markov Decision Process. We define a discounted Markov Decision Process (MDP) as a tuple $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is a measurable state space, $\mathcal{A}$ is a finite set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability kernel, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ is the reward probability kernel, and $\gamma \in [0, 1)$ is the discount factor. We suppose $R(s, a) = \mathbb{E}\left[\mathcal{R}(\cdot|s, a)\right]$ is uniformly bounded by $r_{\max}$. A Markov randomized policy maps states to distributions over actions as $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. As a consequence of taking an action a_t in s_t, the agent receives a reward $r_t \sim \mathcal{R}(\cdot|s_t, a_t)$ and the state evolves accordingly to $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. We define the action-value function of a policy π as $Q^\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t r_t \mid M, \pi, s_0 = s, a_0 = a\right]$ and the optimal action-value function as $Q^*(s, a) = \sup_\pi Q^\pi(s, a)$ for all (s, a). Notice that Q is bounded by $Q_{\max} \coloneqq \frac{r_{\max}}{1-\gamma}$. Then, the optimal policy π^* is a policy that is greedy with respect to Q^*, i.e., for all $s \in \mathcal{S}$, $\pi^*(s) \in \arg\max_{a \in \mathcal{A}}\{Q^*(s, a)\}$. The optimal action-value function is also the unique fixed-point of the *optimal Bellman operator* $L^* : \mathcal{B}(\mathcal{S} \times \mathcal{A}, Q_{\max}) \to \mathcal{B}(\mathcal{S} \times \mathcal{A}, Q_{\max})$, which is defined by $(L^*Q)(s, a) \coloneqq R(s, a) + \gamma \int_{\mathcal{S}} \mathcal{P}(ds'|s, a) \max_{a'} Q(s', a')$ (e.g., Puterman, 1994).

Approximate solution. Fitted Q-Iteration (FQI) (Ernst et al., 2005) is a batch RL algorithm that belongs to the family of Approximate Value Iteration (AVI). AVI is a value-based approach that represents Q-functions by a hypothesis space $\mathcal{H} \subset \mathcal{B}(\mathcal{S} \times \mathcal{A}, Q_{\max})$ of limited capacity. Starting from an initial action-value function $Q_0 \in \mathcal{H}$, at each iteration $k \geq 0$, AVI approximates the application of the optimal Bellman operator in $\mathcal{H}$ such that $Q_{k+1} \approx L^*Q_k$. Formally, let $\mathcal{D}_N = \{\langle s_i, a_i, s'_i, r_i \rangle\}_{i=1}^N$ be a set of transitions such that $(s_i, a_i) \sim \mu$ and define the *empirical* optimal Bellman operator as $(\widehat{L}^*Q)(s_i, a_i) \coloneqq r_i + \gamma \max_{a'} Q(s'_i, a')$. Then, at each iteration k, FQI computes

$$Q_{k+1} = \arg\min_{h \in \mathcal{H}} \left\| h - \widehat{L}^* Q_k \right\|^2_{\mathcal{D}_N}. \tag{1}$$

Transfer settings. We consider a set of tasks, i.e., MDPs, $\{M_j = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}_j, \mathcal{R}_j \rangle, j = 0, \dots, m\}$, where M_0 denotes the target and $M_1, \dots, M_m$ the sources. We suppose all tasks share the same state-action space and have potentially different dynamics and reward. Suppose that, for $j = 0, \dots, m$, we have access to a dataset of N_j samples from the j-th MDP, $\mathcal{D}_j = \{\langle s_i, a_i, s'_i, r_i \rangle\}_{i=1}^{N_j}$, where state-action pairs are drawn from a common distribution $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$.[1] The goal of transfer learning is to use the samples in $\mathcal{D}_1, \dots, \mathcal{D}_m$ to speed up the learning process in the target task M_0.

3. Importance Weights for Transfer

In this section, we introduce our approach to the transfer of samples. Recall that our goal is to exploit at best samples in

[1] This assumption can be relaxed at the price of a much more complex theoretical analysis.

Algorithm 1 Importance Weighted Fitted Q-Iteration

Input: The number of iterations K, a dataset $\widetilde{\mathcal{D}}^+ = \bigcup_{j=0}^{m} \bigcup_{i=0}^{N_j} \left\{ s_i^{(j)}, a_i^{(j)}, s'^{(j)}_i, r_i^{(j)}, \widetilde{w}_{r,i}^{(j)}, \widetilde{w}_{p,i}^{(j)} \right\}$, a hypothesis space $\mathcal{H}$

Output: Greedy policy π_K

$\widehat{R} \leftarrow \arg\inf_{h \in \mathcal{H}} \frac{1}{Z_r} \sum_{j,i} \widetilde{w}_{r,i}^{(j)} \left| h(s_i^{(j)}, a_i^{(j)}) - r_i^{(j)} \right|^2$

$Q_0 \leftarrow \widehat{R}$

for $k = 0, \dots, K-1$ **do**

$Y_i^{(j)} \leftarrow \widetilde{L}^* Q_k(s_i^{(j)}, a_i^{(j)}), \quad \forall i, j$

$Q_{k+1} \leftarrow \arg\inf_{h \in \mathcal{H}} \frac{1}{Z_p} \sum_{j,i} \widetilde{w}_{i,p}^{(j)} \left| h(s_i^{(j)}, a_i^{(j)}) - Y_i^{(j)} \right|^2$

end for

$\pi_K(s) \leftarrow \arg\max_{a \in \mathcal{A}} \{Q_K(s,a)\}, \ \forall s \in \mathcal{S}$

$\{\mathcal{D}_1, \dots, \mathcal{D}_M\}$ to *augment* the dataset $\mathcal{D}_0$ used by FQI to solve the target task, thus speeding up the learning process. In the rest of the paper we exploit the fact that FQI decomposes the RL problem into a sequence of supervised learning problems. It is easy to notice that the optimization problem (1) is an instance of empirical risk minimization, where $X_i = (s_i, a_i)$ are the input data, $Y_i = (\widehat{L}^* Q_k)(s_i, a_i)$ are the targets, and $\mathcal{L}(f(X_i), Y_i) = |f(X_i) - Y_i|^2$ is a squared loss.

As mentioned in the introduction, we aim to exploit all the available samples to solve the target task. Suppose we adopt a naive approach where we concatenate all the samples, i.e., $\widetilde{\mathcal{D}} = \bigcup_{j=0}^{m} \mathcal{D}_j = \bigcup_{j=0}^{m} \bigcup_{i=0}^{N_j} \langle s_i^j, a_i^j, s'^j_i, r_i^j \rangle$, to solve (1). This approach suffers from sample selection bias (Cortes et al., 2008), i.e., samples are collected from different distributions or domains. In fact, although we assumed state-action pairs to be sampled from a fixed task-independent distribution, the target variables Y are distributed according to the MDP they come from.

A standard technique used to correct the bias or discrepancy induced by the distribution shift is *importance weighting*. This technique consists in weighting the loss function to emphasize the error on some samples and decrease it on others, to correct the mismatch between distributions (Cortes et al., 2008). The definition of the importance weight for the point X is $w(X) = P(X)/Q(X)$ where P is the distribution of the target, and Q is the distribution according to which sample X is collected. In our specific case, given an arbitrary sample (X, Y), its joint distribution under MDP M_j is $\mathbb{P}((X,Y)|M_j) = \mathbb{P}(Y|X, M_j)\mu(X)$. Denote by $(X_i^{(j)}, Y_i^{(j)})$ the i-th sample drawn from MDP M_j, then its importance weight is given by $w(X_i^{(j)}, Y_i^{(j)}) = \frac{\mathbb{P}(Y_i^{(j)}|X_i^{(j)}, M_0)}{\mathbb{P}(Y_i^{(j)}|X_i^{(j)}, M_j)}$.

By feeding FQI on the full dataset $\widetilde{\mathcal{D}}$ with samples weighted by $w_i^{(j)}$ (for short), we get an algorithm that automatically selects which samples to exploit, i.e., those that, based on the importance weights, are more likely to be generated from the target MDP. This approach looks appealing but presents several issues. First, the distribution $\mathbb{P}(Y|X, M_j)$ is, even in the case where the MDPs are known, very hard to characterize. Second, consider a simple case where we have a source MDP with the same transition dynamics as the target, but with entirely different reward. Then, the importance weights defined above are likely to be very close to zero for any source sample, thus making transfer useless. However, we would like a method able to leverage the fact that transition dynamics do not change, thus transferring only that part of the sample.

To overcome the second limitation, we consider the following variation of the FQI algorithm. At the first iteration of FQI, we use all the samples to fit a model $\widehat{R} \approx R$ of the target reward function:

$$\widehat{R} = \arg\inf_{h \in \mathcal{H}} \frac{1}{Z_r} \sum_{j=0}^{m} \sum_{i=0}^{N_j} w_{r,i}^{(j)} \left| h(X_i^{(j)}) - r_i^{(j)} \right|^2, \quad (2)$$

where $\mathcal{H} \subset \mathcal{B}(\mathcal{S} \times \mathcal{A}, Q_{max})$ is the hypothesis space we consider to represent action-value functions[2] and

$$w_{r,i}^{(j)} = \frac{\mathcal{R}_0(r_i^{(j)}|X_i^{(j)})}{\mathcal{R}_j(r_i^{(j)}|X_i^{(j)})}. \quad (3)$$

Problem (2) is unbiased if $Z_r = \sum_{j=0}^{m} N_j$, though $Z_r = \sum_{i,j} w_{r,i}^{(j)}$ is frequently used since it provides lower variance. The theoretical analysis is not affected by the choice of Z_r, while in the experiments we will use $Z_r = \sum_{i,j} w_{r,i}^{(j)}$. Then, at each iteration $k \geq 0$, FQI updates the Q-function as:

$$Q_{k+1} = \arg\inf_{h \in \mathcal{H}} \frac{1}{Z_p} \sum_{j=0}^{m} \sum_{i=0}^{N_j} w_{p,i}^{(j)} \left| h(X_i^{(j)}) - \widetilde{Y}_i^{(j)} \right|^2 \quad (4)$$

where $\widetilde{Y}_i^{(j)} = \widetilde{L}^* Q_k(X_i^{(j)}) := \widehat{R}(s_i^{(j)}, a_i^{(j)}) + \gamma \max_{a'} Q_k(s'^{(j)}_i, a')$ and $Q_0 = \widehat{R}$. Intuitively, instead of considering the reward $r_i^{(j)}$ in the dataset $\widetilde{\mathcal{D}}$, we use $\widehat{R}(s_i^{(j)}, a_i^{(j)})$. Since the stochasticity due to the reward samples is now removed, only the transition kernel plays a role, and the importance weights are given by:

$$w_{p,i}^{(j)} = \frac{\mathcal{P}_0(s'^{(j)}_i|X_i^{(j)})}{\mathcal{P}_j(s'^{(j)}_i|X_i^{(j)})}. \quad (5)$$

[2] Differently from other works (e.g., Farahmand & Precup, 2012; Tosatto et al., 2017), we suppose, for the sake of simplicity, the hypothesis space to be bounded by $Q_{\max}$. Although this is a strong assumption, it can be relaxed by considering truncated functions. We refer the reader to (Györfi et al., 2006) for the theoretical consequences of such relaxation.

The resulting algorithm, named Importance Weighted Fitted Q-Iteration (IWFQI), is shown in Algorithm 1. In practice, we have to compute an estimate of $w_{r,i}^{(j)}$ and $w_{p,i}^{(j)}$ since $\mathcal{P}_j$ and $\mathcal{R}_j$ are unknown quantities. We postpone this topic to Section 5 since several approaches can be exploited. Instead, in the following section, we present a theoretical analysis that is independent of the way the importance weights are estimated.

4. Theoretical Analysis

We now study the theoretical properties of our IWFQI algorithm. We analyze the case where we have samples from one source task, but no samples from the target task are available, i.e., $m = 1$, $N_0 = 0$, and $N_1 = N$. A generalization to the case where target samples or samples from more sources are available is straightforward, and it only complicates our derivation. To ease our notation, we adopt the subscript "T" and "S" to denote the target and the source. Furthermore, we want to emphasize that the results provided in this section are *independent from the way the importance weights are estimated.*

Consider the sequence of action-value functions $Q_0, Q_1, \ldots, Q_K$ computed by IWFQI. At each iteration k, we incur in an error $\epsilon_k = L^*Q_k - Q_{k+1}$ in approximating the optimal Bellman operator. Our goal is to bound, in terms of such errors, $\|Q^* - Q^{\pi_k}\|_{1,\rho}$, i.e., the expected error under distribution ρ between the performance of the optimal policy and that of the policy π_k greedy w.r.t. Q_k. Here ρ is an arbitrary evaluation distribution over $\mathcal{S} \times \mathcal{A}$ that the user can freely choose. In practice, it might coincide with the sampling distribution μ. Since IWFQI belongs to the family of AVI algorithms, we can resort to Theorem 3.4 in (Farahmand, 2011). We report here the version with ℓ_1-norm for the sake of completeness.

Theorem 1. *(Theorem 3.4 of (Farahmand, 2011)) Let K be a positive integer and $Q_{\max} \leq \frac{r_{\max}}{1-\gamma}$. Then, for any sequence $(Q_k)_{k=0}^K \subset B(\mathcal{S} \times \mathcal{A}, Q_{\max})$ and the corresponding sequence $(\epsilon_k)_{k=0}^K$, where $\epsilon_k = L^*Q_k - Q_{k+1}$, we have:*

$$\|Q^* - Q^{\pi_K}\|_{1,\rho} \leq \frac{2\gamma}{(1-\gamma)^2}\left[2\gamma^K Q_{\max} + \inf_{b \in [0,1]} \left\{ C_{\mathrm{VI},\rho,\mu}^{\frac{1}{2}}(K;b)\mathcal{E}^{\frac{1}{2}}(\epsilon_0, \ldots, \epsilon_{K-1}; b) \right\}\right],$$

where:

$$\mathcal{E}(\epsilon_0, \ldots, \epsilon_{K-1}; b) = \sum_{k=0}^{K-1} \alpha_k^{2b} \|\epsilon_k\|_\mu^2.$$

We refer the reader to Chapter 3 of (Farahmand, 2011) for the definitions of the coefficients $C_{\mathrm{VI},\rho,\mu}$ and α_k.

Intuitively, the bound given in Theorem 1 depends on the errors made by IWFQI in approximating the optimal Bellman operator at each iteration. Thus, our problem reduces to bounding such errors. Cortes et al. (2010) already provided a theoretical analysis of importance weighted regression. However, their results are not immediately applicable to our case since they only consider a regression problem where the target variable Y is a deterministic function of the input X. On the other hand, we have the more general regression estimation problem where Y is a random variable, and we want to learn its conditional expectation given X. Thus, we extend Theorem 4 of (Cortes et al., 2010) to provide a bound on the expected ℓ_2-error $\|\widehat{h} - h^*\|_\mu$ between the hypothesis $\widehat{h}$ returned by a weighted regressor (with estimated weights $\widetilde{w}$) and the regression function h^*. Following (Cortes et al., 2010), we denote by $Pdim(U)$ the pseudo-dimension of a real-valued function class U. The proof is in the appendix.

Theorem 2. *Let $\mathcal{H} \subset \mathcal{B}(X, F_{\max})$ be a functional space. Suppose we have a dataset of N i.i.d. samples $\mathcal{D} = \{(x_i, y_i)\}$ distributed according to $Q(X,Y) = q(Y|X)\mu(X)$, while $P(X,Y) = p(Y|X)\mu(X)$ is the target distribution. Assume $|Y| \leq F_{\max}$ almost surely. Let $w(x,y) = \frac{p(y|x)}{q(y|x)}$, $\widetilde{w}(x,y)$ be any positive function, $\widehat{h}(x) = \arg\min_{f \in \mathcal{H}} \widehat{\mathbb{E}}_{\mathcal{D}}\left[\widetilde{w}(X,Y)|f(X) - Y|^2\right]$, $h^*(x) = \mathbb{E}_p[Y|x]$, $g(x) = \mathbb{E}_q[\widetilde{w}(x,Y)|x] - 1$, and $M(\widetilde{w}) = \sqrt{\mathbb{E}_Q[\widetilde{w}(X,Y)^2]} + \sqrt{\widehat{\mathbb{E}}_{\mathcal{D}}[\widetilde{w}(X,Y)^2]}$, where $\widehat{\mathbb{E}}_{\mathcal{D}}$ denotes the empirical expectation on $\mathcal{D}$. Furthermore, assume $d = Pdim(\{|f(x) - y|^2 : f \in \mathcal{H}\}) < \infty$ and $\mathbb{E}_Q[\widetilde{w}(X,Y)^2] < \infty$. Then, for any $\delta > 0$, the following holds with probability at least $1 - 2\delta$:*

$$\|\widehat{h} - h^*\|_\mu \leq \inf_{f \in \mathcal{H}} \|f - h^*\|_\mu + F_{\max}\sqrt{\|g\|_{1,\mu}} + 2^{13/8} F_{\max}\sqrt{M(\widetilde{w})}\left(\frac{d\log\frac{2Ne}{d} + \log\frac{4}{\delta}}{N}\right)^{\frac{3}{16}} + 2F_{\max}\|\widetilde{w} - w\|_Q$$

Notice that this result is of practical interest outside of the reinforcement learning field. Here it is used to bound the errors $\|\epsilon_k\|_\mu$ in order to state the following result.

Theorem 3. *Let $\mathcal{H} \subset B(\mathcal{S} \times \mathcal{A}, Q_{\max})$ be a hypothesis space, μ a distribution over $\mathcal{S} \times \mathcal{A}$, $(Q_i)_{i=0}^{k+1}$ a sequence of Q-functions as defined in Equation (4), and L^* the optimal Bellman operator of the target task. Suppose to have a dataset of N i.i.d. samples $\mathcal{D}$ drawn from the source task M_S according to a joint distribution ϕ_S. Let w_p, w_r denote the ideal importance weights defined in (5) and (3), and $\widetilde{w}_r(r|s,a)$, $\widetilde{w}_p(s'|s,a)$ denote arbitrary positive functions with bounded second moments. Define $g_r(s,a) = \mathbb{E}_{\mathcal{R}_S}[\widetilde{w}_r(r|s,a)|s,a] - 1$, $M(\widetilde{w}_r) = \sqrt{\mathbb{E}_{\phi_S^R}[\widetilde{w}_r(r|s,a)^2]} + \sqrt{\widehat{\mathbb{E}}_{\mathcal{D}}[\widetilde{w}_r(r|s,a)^2]}$, where*

$\phi_S^R(r|s,a) = \mu(s,a)\mathcal{R}_S(r|s,a)$. *Similarly, define* g_p, $M(\widetilde{w}_p)$, *and* $\phi_S^P(s'|s,a)$ *for the transition model. Then, for any* $\delta > 0$, *with probability at least* $1 - 4\delta$:

$$\|L^*Q_k - Q_{k+1}\|_\mu \leq Q_{\max}\sqrt{\|g_p\|_{1,\mu}} + 2r_{\max}\sqrt{\|g_r\|_{1,\mu}}$$
$$+ 2Q_{\max}\|\widetilde{w}_p - w_p\|_{\phi_S^P} + 4r_{\max}\|\widetilde{w}_r - w_r\|_{\phi_S^R}$$
$$+ \inf_{f\in\mathcal{H}}\|f - (L^*)^{k+1}Q_0\|_\mu + 2\inf_{f\in\mathcal{H}}\|f - R\|_\mu$$
$$+ \frac{Q_{\max}}{2^{-\frac{13}{8}}}\left(\sqrt{M(\widetilde{w}_p)} + 2\sqrt{M(\widetilde{w}_r)}\right)\left(\frac{d\log\frac{2Ne}{d} + \log\frac{4}{\delta}}{N}\right)^{\frac{3}{16}}$$
$$+ \sum_{i=0}^{k-1}(\gamma C_{\mathrm{AE}}(\mu))^{i+1}\|\epsilon_{k-i-1}\|_\mu,$$

where C_{AE} *is the concentrability coefficient of one-step transitions as defined in (Farahmand, 2011, Definition 5.2).*

As expected, four primary sources of error contribute to our bound: (i) the bias due to estimated weights (first four terms), (ii) the approximation error (fifth and sixth term), (iii) the estimation error (seventh term), (iv) the propagation error (eighth term). Notice that, assuming to have a consistent estimator for the importance weights (an example is given in Section 5), the bias term vanishes as the number of samples N tends to infinity. Furthermore, the estimation error decreases with N, thus vanishing as the number of samples increases. Thus, in the asymptotic case our bound shows that the only source of error is due to the limited capacity of the functional space $\mathcal{H}$ under consideration, as in most AVI algorithms. Furthermore, we notice that fitting the reward function and using it instead of the available samples propagates an error term through iterations, i.e., the approximation error $\inf_{f\in\mathcal{H}}\|f - R\|_\mu$. If we were able to estimate the importance weights for the typical case where both reward and transition samples are used, we could get rid of such error. However, since the resulting weights somehow depend on the joint densities between $\mathcal{P}$ and $\mathcal{R}$, we expect their variance, as measured by $M(\widetilde{w})$, to be much bigger, thus making the resulting bound even larger. Furthermore, we argue that, when the reward function is simple enough and only a limited number of samples is available, a separate fit might be beneficial even for plain FQI. In fact, the variance of the empirical optimal Bellman operator can be reduced by removing the source of stochasticity due to the reward samples at the cost of propagating a small approximation error through iterations. The bounds for AVI, (e.g., Munos & Szepesvári, 2008; Farahmand, 2011; Farahmand & Precup, 2012), can be straightforwardly extended to such case by adopting a procedure similar to the one described in the proof of Theorem 3. Finally, in most practical applications the reward function is actually known and, thus, does not need to be fitted. In such cases, it is possible to get rid of the corresponding terms in Theorem 3, allowing transfer to occur without errors even when rewards are completely different between tasks.

5. Estimation of Importance Weights

In this section, we specify how to compute the importance weights. Since $\mathcal{P}$ and $\mathcal{R}$ are unknown, we only have access to an estimation of $w_{r,i}^{(j)}$ and $w_{p,i}^{(j)}$ used in (3) and (5), respectively. To obtain an approximation of the unknown densities, we consider Gaussian Processes (GPs) although any distribution matching technique and/or probabilistic model can be used.

Gaussian Processes. We use the available samples to fit two Gaussian processes (GPs) (Rasmussen & Williams, 2006) for each task M_j: one for the transition model $\mathcal{P}_j$ and one for the reward model $\mathcal{R}_j$. To motivate our choice, GPs have been successfully adopted to model stochastic dynamical systems with high-dimensional and continuous state-action spaces in many existing works (e.g., Kuss & Rasmussen, 2004; Deisenroth & Rasmussen, 2011; Doshi-Velez & Konidaris, 2016; Berkenkamp et al., 2017). For the sake of simplicity, we only show how to compute the importance weights for the reward model. Our procedure straightforwardly generalizes to the transition model.

Given a sample $\langle s,a,r\rangle$ from the j-th task, the j-th GP returns a Gaussian distribution over the reward's mean, i.e., $\overline{r}(s,a) \sim \mathcal{N}(\mu_{GP_j}(s,a), \sigma^2_{GP_j}(s,a))$, which, together with the target GP's prediction, induces a distribution over the importance weights. In practice, the choice of a single importance weight can rely on some statistics of such distribution (e.g., its mean or mode). Perhaps not surprisingly, this is made non-trivial by the fact that explicitly characterizing such distribution is very complicated, and computing empirical statistics requires an expensive repeated sampling from the GPs' posteriors. Interestingly, the following theorem shows that this is not necessary when the reward model follows a Gaussian law, as the expected weights under their unknown distribution can be computed in closed-form.

Theorem 4 (Reward Weights in Gaussian Models)**.** *Assume each task to have Gaussian reward distribution* $\mathcal{R}_j(\cdot|s,a) = \mathcal{N}\left(\mu_r^{(j)}(s,a), \sigma_j^2(s,a)\right)$ *with unknown mean. Given the available samples in* $\widetilde{\mathcal{D}}$, *we build an estimate of the reward distribution such that, for any MDP* M_j, $\overline{r}^{(j)}(s,a) \sim \mathcal{N}(\mu_{GP_j}(s,a), \sigma^2_{GP_j}(s,a))$. *Then, given a sample* $\langle s,a,r\rangle$ *from the* j*-th MDP, its importance weight* $w = \frac{\mathcal{N}\left(r|\overline{r}^{(0)}(s,a),\sigma_0^2(s,a)\right)}{\mathcal{N}\left(r|\overline{r}^{(j)}(s,a),\sigma_j^2(s,a)\right)} \sim \mathcal{G}$, *where* $\mathcal{G}$ *is the distribution induced by the GPs' predictions. Let* $C = \frac{\sigma_j^2(s,a)}{\sigma_j^2(s,a) - \sigma^2_{GP_j}(s,a)}$ *and suppose* $\sigma^2_{GP_j}(s,a) < \sigma_j^2(s,a)$, *then*

$$\mathbb{E}_{\mathcal{G}}[w] = C\,\frac{\mathcal{N}\left(r\middle|\mu_{GP_0}(s,a),\sigma_0^2(s,a)+\sigma^2_{GP_0}(s,a)\right)}{\mathcal{N}\left(r\middle|\mu_{GP_j}(s,a),\sigma_j^2(s,a)-\sigma^2_{GP_j}(s,a)\right)}. \quad (6)$$

The proof is in Appendix A. In practice, we estimate the importance weights by taking their expectation as in (6), i.e., $\widetilde{w} = \mathbb{E}_{\mathcal{G}}[w]$. Intuitively, using the expected weights is more robust than merely taking the ratio of the estimated densities. Furthermore, the estimated weights converge to the true ones when the GP predictions are perfect, i.e., when $\mu_{GP}(s,a) = \mu_r(s,a)$ and $\sigma^2_{GP}(s,a) \to 0$, both in the source and in the target. This is a significant advantage over the more common approach of density-ratio estimation (Sugiyama et al., 2012), where a parametric form for the weight function is typically assumed. One drawback is that the expectation diverges when $\sigma^2_{GP_j}(s,a) > \sigma^2_j$, that is, when the source GP has a prediction variance that is greater than the intrinsic noise of the model. Notice, however, that this happens very rarely since the source GP is never asked to predict samples it has not seen during training. Furthermore, since in practice the model noise is unknown and has to be estimated, an overestimation is beneficial as it introduces a regularization effect (Mohammadi et al., 2016), thus avoiding the problem mentioned above.

6. Related Work

In (Taylor et al., 2008), the authors propose a method to transfer samples for model-based RL. Although they assume tasks might have different state-action space, inter-task mappings are used to map source samples to the target. However, the proposed method does not account for differences in the transition or reward models, which could lead to significant negative transfer when the tasks are different. Our approach, on the other hand, can selectively discard samples based on the estimated difference between the MDPs. Lazaric et al. (2008) compute a compliance measure between the target and source tasks and use it to specify from which tasks the transfer is more likely to be beneficial. Furthermore, a relevance measure is computed within each task to determine what are the best samples to transfer. These two measures are then combined to transfer samples into FQI. Once again, our approach does not require any explicit condition to decide what to transfer, nor does it require any assumption of similarity between the tasks. Furthermore, the compliance and relevance measures computed in (Lazaric et al., 2008) jointly account for both the reward and transition models, thus discarding samples when either one of the models is very different between the tasks. On the other hand, our approach can retain at least the part of the sample that is similar, at the cost of introducing a small bias. In (Laroche & Barlier, 2017), the authors propose a technique for transferring samples into FQI under the assumption that the transition dynamics do not change between the tasks. Similarly to our method, they learn the reward function at the first iteration and substitute the predicted values to the reward samples in the dataset. This allows them to safely adopt the full set of target and source samples in the remaining FQI iterations, as all tasks share the same transition model and, thus, samples are unbiased. However, we argue that this assumption of shared dynamics indeed limits the applicability of the transfer method to most real-world tasks.

In the supervised learning literature, Crammer et al. (2008) analyzed the transfer problem from a theoretical perspective and extended the classical generalization bounds to the case where samples are directly transferred from a set of source tasks. The most relevant result in their bounds is a trade-off between the total number of samples transferred and the total number of tasks from which transfer occurs. Increasing the first term decreases the variance, while it is likely to increase the bias due to the differences between the tasks. On the other hand, decreasing the first term also decreases the bias, but it is likely to increase the variance due to the limited number of samples. We observe that such trade-off does not arise in our case. Our method transfers all samples while accounting for the differences between the tasks. The only bias term is due to the errors in the estimation of the task models, which is likely to decrease as the number of samples increases.

Another work from the supervised learning literature that is related to our approach is (Garcke & Vanck, 2014). The authors proposed a method to transfer samples from a different dataset and used importance weighting to correct the distribution shift. However, they leveraged ideas from density-ratio estimation (e.g., Sugiyama et al., 2012) and supposed the weight function to have a given parametric form, thus directly estimating it from the data. Conversely, we estimate the densities involved and try to characterize the weight distribution, taking its expectation as our final estimate.

7. Experiments

We evaluate IWFQI on three different domains with increasing level of complexity. In all experiments, we compare our method to two existing algorithms for transferring samples into FQI: the relevance-based transfer (RBT) algorithm of (Lazaric et al., 2008) and the shared-dynamics transfer (SDT) algorithm of (Laroche & Barlier, 2017).

7.1. Puddle World

Our first experimental domain is a modified version of the puddle world environment presented in (Sutton, 1996). Puddle world is a discrete-action, continuous-state (stochastic) navigation problem (see Appendix C.1 for a complete description). At each time-step, the agent receives a reward of -1 plus a penalization proportional to the distance from all puddles. Each action moves the agent by α in the corresponding direction. In particular, we con-

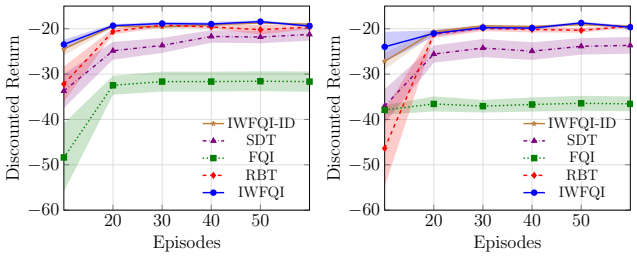

Figure 1. Puddle world with 20×3 episodes transferred from 3 source tasks in the case of shared dynamics *(left)* and puddle-based dynamics *(right)*.

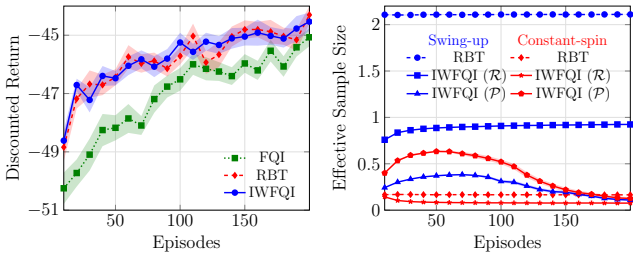

Figure 2. Acrobot swing-up with $(100 + 50)$ episodes transferred from 2 source tasks. *(left)* learning performance. *(right)* relative number of samples transferred from each source task.

sider two versions of the environment: (i) *shared dynamics*, where $\alpha = 1$ is fixed, and (ii) *puddle-based dynamics*, where α slows-down the agent proportionally to the distance from all puddles.

We consider three source tasks and one target task, where each task has different puddles in different locations (see Appendix C.1). For each source task, we generate a dataset of 20 episodes from a nearly-optimal policy. We run IWFQI with weights computed according to Equation (6), where we set the model noise to be ten times the true value. For evaluating our weight estimation procedure, we also run IWFQI with ideal importance weights (computed as the ratio of the true distributions). In each algorithm, FQI is run for 50 iterations with Extra-Trees (Ernst et al., 2005). An ϵ-greedy policy ($\epsilon = 0.3$) is used to collect data in the target task.

Shared dynamics. We start by showing the results for $\alpha = 1$ in Figure 1*(left)*. All results are averaged over 20 runs and are reported with 95% confidence intervals. As expected, FQI alone is not able to learn the target task in such a small number of episodes. On the other hand, IWFQI has a good jump-start and converges to an optimal policy in only 20 episodes. Interestingly, IWFQI with ideal weights has almost the same performance, thus showing the robustness of our weight estimation procedure. RBT also learns the optimal policy rather quickly. However, the limited number of target and source samples available in this experiment makes it perform significantly worse in the first episodes. Since in this version of the puddle world the dynamics do not change between tasks, SDT also achieves good performance, converging to a nearly-optimal policy.

Puddle-based dynamics. We also show the results for the more challenging version of the environment were puddles both penalize and slow-down the agent (see Figure 1*(right)*). Notice that, in this case, transition dynamics change between tasks, thus making the transfer more challenging. Similarly, as before, our approach quickly learns the optimal policy and is not affected by the estimated weights. Furthermore, the benefits of over-estimating the model noise can be observed from the small improvement over IWFQI-ID. RBT is also able to learn the optimal policy. However, the consequences of inaccurately computing compliance and relevance are more evident in this case, where the algorithm negatively transfers samples in the first episodes. Finally, SDT still shows an improvement over plain FQI, but it is not able to learn the optimal policy due to the bias introduced by the different dynamics.

7.2. Acrobot

Acrobot (Sutton & Barto, 1998) is a classic control problem where the goal is to swing-up a two-link pendulum by applying positive or negative torque to the joint between the two links. Due to its non-linear and complex dynamics, Acrobot represents a very challenging problem, requiring a considerable amount of samples to be solved. In this experiment, we consider a multi-task scenario where robots might have different link lengths (l_1, l_2) and masses (m_1, m_2). Our target task is the classic Acrobot *swing-up* problem, where the robot has lengths $(1.0, 1.0)$ and masses $(1.0, 1.0)$. Furthermore, we consider two source tasks. The first is another swing-up task where the robot has lengths $(1.1, 0.7)$ and masses $(0.9, 0.6)$. The second is a *constant-spin* task, where the goal is to make the first joint rotate at a fixed constant speed, with lengths $(0.95, 0.95)$ and masses $(0.95, 1.0)$. The exact definition of the tasks' dynamics and rewards is in Appendix C.2. Notice the intrinsic difficulty of transfer: the first source task has the same reward as the target but very different dynamics, and conversely for the second source task. Using nearly-optimal policies, we generate 100 episodes from the first source and 50 episodes from the second. We run all algorithms (except SDT since the problem violates the shared-dynamics assumption) for 200 episodes and average over 20 runs. Results are shown in Figure 2*(left)*. We notice that both our approach and RBT achieve a good jump-start and learn faster than plain FQI. However, to better investigate how samples are transferred, we show the transfer ratio from each source task in Figure 2*(right)*. Since RBT transfers rewards and transitions jointly, it decides to compensate the highly biased re-

ward samples from the constant-spin task by over-sampling the first source task. However, it inevitably introduces bias from the different dynamics. Our approach, on the other hand, correctly transfers almost all reward samples from the swing-up task, while discarding those from the constant-spin task. Due to transition noise over-estimation, IWFQI achieves an interesting adaptive behaviour: during the initial episodes, when few target samples are available, and the GPs are inaccurate, more samples are transferred. This causes a reduction of the variance in the first phases of learning that is much greater than the increase of bias. However, as more target samples are available, the transfer becomes useless, and our approach correctly decides to discard most transition samples, thus minimizing both bias and variance.

7.3. Water Reservoir Control

In this experiment, we consider a real-world problem where the goal is to learn how to optimally control a water reservoir system. More specifically, the objective is to learn a per-day water release policy that meets a given demand while keeping the water level below a flooding threshold. Castelletti et al. (2010) successfully addressed such problem by adopting batch RL techniques. However, the authors proved that, due to the highly non-linear and noisy environment, an enormous amount of historical data is needed to achieve good performance. Consider now the case where a new water reservoir, for which no historical data is available, needs to be controlled. Since each sample corresponds to one day of release, learning by direct interaction with the environment is not practical and leads to poor control policies during the initial years, when only a little experience has been collected. Although we do not know the new environment, it is reasonable to assume that we have access to operational data from existing reservoirs. Then, our solution is to transfer samples to immediately achieve good performance. However, such reservoirs might be located in very different environments and weight objectives differently, thus making transfer very challenging.

We adopt a system model similar to the one proposed in (Castelletti et al., 2010). The state variables are the current water storage s_t and day $t \in [1, 365]$, while there are 8 discrete actions, each corresponding to a particular release decision. The system evolves according to the simple mass balance equation $s_{t+1} = s_t + i_t - a_t$, where i_t is the net inflow at day t and is modeled as periodic function, with period of one year, plus Gaussian noise. Given the demand d and the flooding threshold f, the reward function is a convex combination of the two objectives, $R(s_t, a_t) = -\alpha\ max\{0, s_t - f\} - \beta(\max\{0, d - a_t\})^2$, where $\alpha, \beta \geq 0$. Different tasks have different inflow functions and different reward weights, which model different geographic regions and objectives, respectively.

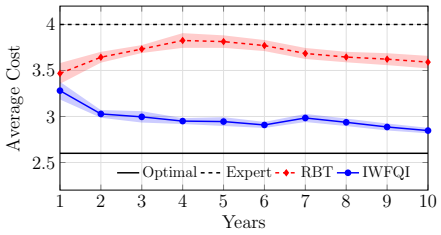

Figure 3. Water reservoir control. Average *cost* per day during the first 10 years of learning. IWFQI outperforms the expert and quickly achieves near-optimal performance.

We collected 10800 samples, corresponding to 30 years of historical data, from each of 6 source water reservoirs under a hand-coded expert policy. Further details about the tasks are given in Appendix C.3. We compared our approach to FQI and RBT over the first 10 years of learning. An ϵ-greedy policy ($\epsilon = 0.3$) was used to collect batches of 1 year of samples, except for the first batch, for which an expert's policy was used. Results, averaged over 20 runs, are shown in Figure 3. We notice that IWFQI immediately outperforms the expert's policy and quickly achieves near-optimal performance. RBT, on the other hand, has a good jump-start but then seems to worsen its performance. Once again, this is because each source task has at least few samples that can be transferred. However, selecting such samples is very complicated and leads to negative transfer in case of failure. Finally, FQI performs significantly worse than all alternatives and is, thus, not reported.

8. Conclusions

In this paper, we presented Importance Weighted Fitted Q-Iteration, a novel AVI algorithm for transferring samples in batch RL that uses importance weighting to automatically account for the difference in the source and target distributions. IWFQI exploits Gaussian processes to learn transition and reward models that are used to compute the importance weights. The use of two different processes for reward and transition models allows maximizing the information transferred. We theoretically investigated IWFQI showing (i) its asymptotic correctness in general settings, and (ii) how to compute a robust statistical estimate of the weights for Gaussian models. Finally, we empirically proved its effectiveness in common benchmarks and on a real-world water control problem.

One of the drawbacks of our method is that it does not fully exploit possible similarities between tasks. Recent approaches (e.g., Doshi-Velez & Konidaris, 2016; Killian et al., 2017) learn models relating a family of tasks to ease the transfer of knowledge. Exploring how such relations can benefit our approach (e.g., to improve the weight estimates) is an interesting line for future developments.

References

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. Safe model-based reinforcement learning with stability guarantees. In *Advances in Neural Information Processing Systems 30*, pp. 908–919. Curran Associates, Inc., 2017.

Bromiley, P. Products and convolutions of gaussian probability density functions. 2003.

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

Cortes, C., Mohri, M., Riley, M., and Rostamizadeh, A. Sample selection bias correction theory. In *International Conference on Algorithmic Learning Theory*, pp. 38–53. Springer, 2008.

Cortes, C., Mansour, Y., and Mohri, M. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pp. 442–450, 2010.

Crammer, K., Kearns, M., and Wortman, J. Learning from multiple sources. *Journal of Machine Learning Research*, 9(Aug):1757–1774, 2008.

Deisenroth, M. and Rasmussen, C. E. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

Doshi-Velez, F. and Konidaris, G. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, pp. 1432. NIH Public Access, 2016.

Ernst, D., Geurts, P., and Wehenkel, L. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

Farahmand, A.-m. *Regularization in reinforcement learning*. PhD thesis, University of Alberta, 2011.

Farahmand, A. M. and Precup, D. Value pursuit iteration. In *Advances in Neural Information Processing Systems*, pp. 1340–1348, 2012.

Garcke, J. and Vanck, T. Importance weighted inductive transfer learning for regression. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 466–481. Springer, 2014.

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Killian, T. W., Daulton, S., Doshi-Velez, F., and Konidaris, G. Robust and efficient transfer learning with hidden parameter markov decision processes. In *Advances in Neural Information Processing Systems 30*, pp. 6251–6262, 2017.

Kuss, M. and Rasmussen, C. E. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems 16*, pp. 751–758. MIT Press, 2004.

Laroche, R. and Barlier, M. Transfer reinforcement learning with shared dynamics. In *AAAI*, pp. 2147–2153. AAAI Press, 2017.

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

Lazaric, A. and Restelli, M. Transfer from multiple mdps. In *Advances in Neural Information Processing Systems*, pp. 1746–1754, 2011.

Lazaric, A., Restelli, M., and Bonarini, A. Transfer of samples in batch reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 544–551. ACM, 2008.

Mohammadi, H., Riche, R. L., Durrande, N., Touboul, E., and Bay, X. An analytic comparison of regularization methods for gaussian processes. *arXiv preprint arXiv:1602.00853*, 2016.

Munos, R. and Szepesvári, C. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1994. ISBN 0471619779.

Rasmussen, C. E. and Williams, C. K. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Sugiyama, M., Suzuki, T., and Kanamori, T. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

Sutton, R. S. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in neural information processing systems*, pp. 1038–1044, 1996.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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

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

Taylor, M. E., Jong, N. K., and Stone, P. Transferring instances for model-based reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 488–505. Springer, 2008.

Tosatto, S., Pirotta, M., D'Eramo, C., and Restelli, M. Boosted fitted q-iteration. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3434–3443. PMLR, 2017.