Transfer from Multiple MDPs

Alessandro Lazaric, Marcello Restelli

Transfer from Multiple MDPs

Alessandro Lazaric
INRIA Lille - Nord Europe, Team SequeL, France
alessandro.lazaric@inria.fr

Marcello Restelli
Department of Electronics and Informatics, Politecnico di Milano, Italy
restelli@elet.polimi.it

Abstract

Transfer reinforcement learning (RL) methods leverage on the experience collected on a set of source tasks to speed-up RL algorithms. A simple and effective approach is to transfer samples from source tasks and include them in the training set used to solve a target task. In this paper, we investigate the theoretical properties of this transfer method and we introduce novel algorithms adapting the transfer process on the basis of the similarity between source and target tasks. Finally, we report illustrative experimental results in a continuous chain problem.

1 Introduction

The objective of transfer in reinforcement learning (RL) [12] is to speed-up RL algorithms by reusing knowledge (e.g., samples, value function, features, parameters) obtained from a set of source tasks. The underlying assumption of transfer methods is that the source tasks (or a suitable combination of these) are somehow similar to the target task, so that the transferred knowledge can be useful in learning its solution. A wide range of scenarios and methods for transfer in RL have been studied in the last decade (see [14, 9] for a thorough survey). In this paper, we focus on the simple transfer approach where trajectory samples are transferred from source MDPs to increase the size of the training set used to solve the target MDP. This approach is particularly suited in problems (e.g., robotics, applications involving human interaction) where it is not possible to interact with the environment long enough to collect samples to solve the task at hand. If samples are available from other sources (e.g., simulators in case of robotic applications), the solution of the target task can benefit from a larger training set that includes also some source samples. This approach has been already investigated in the case of transfer between tasks with different state-action spaces in [13], where the source samples are used to build a model of the target task whenever the number of target samples is not large enough. A more sophisticated sample-transfer method is proposed in [8]. The authors introduce an algorithm which estimates the similarity between source and target tasks and selectively transfers from the source tasks which are more likely to provide samples similar to those generated by the target MDP. Although the empirical results are encouraging, the proposed method is based on heuristic measures and no theoretical analysis of its performance is provided. On the other hand, in supervised learning a number of theoretical works investigated the effectiveness of transfer in reducing the sample complexity of the learning process. In domain adaptation, a solution learned on a source task is transferred to a target task and its performance depends on how *similar* the two tasks are. In [2] and [10] different distance measures are proposed and are shown to be connected to the performance of the transferred solution. The case of transfer of samples from multiple source tasks is studied in [3]. The most interesting finding is that the transfer performance benefits from using a larger training set at the cost of an additional error due to the average distance between source and target tasks. This implies the existence of a *transfer tradeoff* between transferring as many samples as possible and limiting the transfer to sources which are similar to the target task. As a result, the

transfer of samples is expected to outperform single-task learning whenever *negative* transfer (i.e., transfer from source tasks far from the target task) is limited w.r.t. to the advantage of increasing the size of the training set. This also opens the question whether it is possible to design methods able to automatically detect the similarity between tasks and adapt the transfer process accordingly.

In this paper, we investigate the transfer of samples in RL from a more theoretical perspective w.r.t. previous works. The main contributions of this paper can be summarized as follows:

- *Algorithmic contribution.* We introduce three sample-transfer algorithms based on fitted Q-iteration [4]. The first algorithm (*AST* in Section 3) simply transfers all the source samples. We also design two adaptive methods (*BAT* and *BTT* in Section 4 and 5) whose objective is to solve the transfer tradeoff by identifying the best combination of source tasks.
- *Theoretical contribution.* We formalize the setting of transfer of samples and we derive a finite-sample analysis of AST which highlights the importance of the *average* MDP obtained by the combination of the source tasks. We also report the analysis for BAT which shows both the advantage of identifying the best combination of source tasks and the additional cost in terms of auxiliary samples needed to compute the similarity between tasks.
- *Empirical contribution.* We report results (in Section 6) on a simple chain problem which confirm the main theoretical findings and support the idea that sample transfer can significantly speed-up the learning process and that adaptive methods are able to solve the transfer tradeoff and avoid negative transfer effects.

The rest of the paper is organized as follows. In Section 2 we introduce the notation and we define the transfer problem. Section 3 reports the theoretical analysis of AST. BAT is described in Section 4 along with its theoretical analysis. A more challenging setting is introduced in Section 5 together with BTT. Section 6 reports the experimental results and Section 7 concludes the paper. Finally, in the appendix we report the proofs and some additional experimental analysis.

2 Preliminaries

In this section we introduce the notation and the transfer problem considered in the rest of the paper.

Notation for MDPs. We define a discounted Markov decision process (MDP) as a tuple $\mathcal{M} = \langle \mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$ where the state space $\mathcal{X}$ is a bounded closed subset of the Euclidean space, $\mathcal{A}$ is a finite ($|\mathcal{A}| < \infty$) action space, the deterministic[1] reward function $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is uniformly bounded by $R_{\max}$, the transition kernel $\mathcal{P}$ is such that for all $x \in \mathcal{X}$ and $a \in \mathcal{A}$, $\mathcal{P}(\cdot|x,a)$ is a distribution over $\mathcal{X}$, and $\gamma \in (0,1)$ is a discount factor. We denote by $\mathcal{S}(\mathcal{X} \times \mathcal{A})$ the set of probability measures over $\mathcal{X} \times \mathcal{A}$ and by $\mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max} = \frac{R_{\max}}{1-\gamma})$ the space of bounded measurable functions with domain $\mathcal{X} \times \mathcal{A}$ and bounded in $[-V_{\max}, V_{\max}]$. We define the optimal action-value function Q^* as the unique fixed-point of the optimal Bellman operator $\mathcal{T} : \mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max}) \to \mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max})$ defined by

$$(\mathcal{T}Q)(x,a) = \mathcal{R}(x,a) + \gamma \int_{\mathcal{X}} \max_{a' \in \mathcal{A}} Q(y,a') \mathcal{P}(dy|x,a).$$

Notation for function spaces. For any measure $\mu \in \mathcal{S}(\mathcal{X} \times \mathcal{A})$ obtained from the combination of a distribution $\rho \in \mathcal{S}(\mathcal{X})$ and a uniform distribution over the discrete set $\mathcal{A}$, and a measurable function $f : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$, we define the $L_2(\mu)$-norm of f as $||f||_\mu^2 = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \int_{\mathcal{X}} f(x,a)^2 \rho(dx)$. The supremum norm of f is defined as $||f||_\infty = \sup_{x \in \mathcal{X}} |f(x)|$. Finally, we define the standard L_2-norm for a vector $\alpha \in \mathbb{R}^d$ as $||\alpha||^2 = \sum_{i=1}^d \alpha_i^2$. We denote by $\phi(\cdot,\cdot) = \big(\varphi_1(\cdot,\cdot), \ldots, \varphi_d(\cdot,\cdot)\big)^\top$ a feature vector with features $\varphi_i : \mathcal{X} \times \mathcal{A} \to [-C, C]$, and by $\mathcal{F} = \{f_\alpha(\cdot,\cdot) = \phi(\cdot,\cdot)^\top \alpha\}$ the linear space of action-value functions spanned by the basis functions in ϕ. Given a set of state-action pairs $\{(X_l, A_l)\}_{l=1}^L$, let $\Phi = [\phi(X_1, A_1)^\top; \ldots; \phi(X_L, A_L)^\top]$ be the corresponding feature matrix. We define the orthogonal projection operator $\Pi : \mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max}) \to \mathcal{F}$ as $\Pi Q = \arg\min_{f \in \mathcal{F}} ||Q - f||_\mu$. Finally, by $T(Q)$ we denote the truncation of a function Q in the range $[-V_{\max}, V_{\max}]$.

[1] The extension to stochastic reward functions is straightforward.

Input: Linear space $\mathcal{F} = \text{span}\{\varphi_i, 1 \le i \le d\}$, initial function $\widetilde{Q}^0 \in \mathcal{F}$

for $k = 1, 2, \ldots$ **do**
  Build the training set $\{(X_l, A_l, Y_l, R_l)\}_{l=1}^L$ [according to *random* tasks design]
  Build the feature matrix $\Phi = [\phi(X_1, A_1)^\top; \ldots; \phi(X_L, A_L)^\top]$
  Compute the vector $p \in \mathbb{R}^L$ with $p_l = R_l + \gamma \max_{a' \in \mathcal{A}} \widetilde{Q}^{k-1}(Y_l, a')$
  Compute the projection $\hat{\alpha}^k = (\Phi^\top \Phi)^{-1} \Phi^\top p$ and the function $\widehat{Q}^k = f_{\hat{\alpha}^k}$
  Return the truncated function $\widetilde{Q}^k = T(\widehat{Q}^k)$
end for

Figure 1: A pseudo-code for All-Sample Transfer (AST) Fitted Q-iteration.

Problem setup. We consider the transfer problem in which M tasks $\{\mathcal{M}_m\}_{m=1}^M$ are available and the objective is to learn the solution for the target task $\mathcal{M}_1$ transferring samples from the source tasks $\{\mathcal{M}_m\}_{m=2}^M$. We define an assumption on how the training sets are generated.

Definition 1. ***(Random Tasks Design)*** *An input set $\{(X_l, A_l)\}_{l=1}^L$ is built with samples drawn from an arbitrary sampling distribution $\mu \in \mathcal{S}(\mathcal{X} \times \mathcal{A})$, i.e. $(X_l, A_l) \sim \mu$. For each task m, one transition and reward sample is generated in each of the state-action pairs in the input set, i.e. $Y_l^m \sim \mathcal{P}(\cdot|X_l, A_l)$, and $R_l^m = \mathcal{R}(X_l, A_l)$. Finally, we define the random sequence $\{M_l\}_{l=1}^L$ where the indexes M_l are drawn i.i.d. from a multinomial distribution with parameters $(\lambda_1, \ldots, \lambda_M)$. The training set available to the learner is $\{(X_l, A_l, Y_l, R_l)\}_{l=1}^L$ where $Y_l = Y_{l,M_l}$ and $R_l = R_{l,M_l}$.*

This is an assumption on how the samples are generated but in practice, a single realization of samples and task indexes M_l is available. We consider the case in which $\lambda_1 \ll \lambda_m$ $(m = 2, \ldots, M)$. This condition implies that (on average) the number of target samples is much less than the source samples and it is usually not enough to learn an accurate solution for the target task. We will also consider the *pure transfer* case in which $\lambda_1 = 0$ (i.e., no target sample is available). Finally, we notice that Def. 1 implies the existence of a generative model for all the MDPs, since the state-action pairs are generated according to an arbitrary sampling distribution μ.

3 All-Sample Transfer Algorithm

We first consider the case when the source samples are generated beforehand according to Def. 1 and the designer has no access to the source tasks. We study the algorithm called *All-Sample Transfer* (AST) (Fig. 1) which simply runs FQI with a linear space $\mathcal{F}$ on the whole training set $\{(X_l, A_l, Y_l, R_l)\}_{l=1}^L$. At each iteration k, given the result of the previous iteration $\widetilde{Q}^{k-1} = T(\widehat{Q}^{k-1})$, the algorithm returns

$$\widehat{Q}^k = \arg\min_{f \in \mathcal{F}} \frac{1}{L} \sum_{l=1}^{L} \left(f(X_l, A_l) - (R_l + \gamma \max_{a' \in \mathcal{A}} \widetilde{Q}^{k-1}(Y_l, a')) \right)^2. \tag{1}$$

In the case of linear spaces, the minimization problem is solved in closed form as in Fig. 1. In the following we report a finite-sample analysis of the performance of AST. Similar to [11], we first study the prediction error in each iteration and we then propagate it through iterations.

3.1 Single Iteration Finite-Sample Analysis

We define the *average* MDP $\overline{\mathcal{M}}_\lambda$ as the average of the M MDPs at hand. We define its reward function $\overline{\mathcal{R}}_\lambda$ and its transition kernel $\overline{\mathcal{P}}_\lambda$ as the weighted average of reward functions and transition kernels of the basic MDPs with weights determined by the proportions λ of the multinomial distribution in the definition of the random tasks design (i.e., $\overline{\mathcal{R}}_\lambda = \sum_{m=1}^M \lambda_m \mathcal{R}_m$, $\overline{\mathcal{P}}_\lambda = \sum_{m=1}^M \lambda_m \mathcal{P}_m$). The resulting average Bellman operator is

$$(\overline{\mathcal{T}}_\lambda Q)(x, a) = \Big(\sum_{m=1}^{M} \lambda_m \mathcal{T}^m Q \Big)(x, a) = \overline{\mathcal{R}}(x, a) + \gamma \int_{\mathcal{X}} \max_{a'} Q(y, a') \overline{\mathcal{P}}(dy|x, a). \tag{2}$$

In the random tasks design, the average MDP plays a crucial role since the implicit target function of the minimization of the empirical loss in Eq. 1 is indeed $\overline{\mathcal{T}}_\lambda \widetilde{Q}_{k-1}$. At each iteration k, we prove the following performance bound for AST.

Theorem 1. *Let M be the number of tasks $\{\mathcal{M}_m\}_{m=1}^M$, with $\mathcal{M}_1$ the target task. Let the training set $\{(X_l, A_l, Y_l, R_l)\}_{l=1}^L$ be generated as in Def. 1, with a proportion vector $\lambda = (\lambda_1, \dots, \lambda_M)$. Let $f_{\alpha_*^k} = \Pi \mathcal{T}_1 \widetilde{Q}^{k-1} = \arg\inf_{f \in \mathcal{F}} ||f - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu$, then for any $0 < \delta \leq 1$, $\widehat{Q}^k$ (Eq. 1) satisfies*

$$||T(\widehat{Q}^k) - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu \leq 4||f_{\alpha_*^k} - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu + 5\sqrt{\mathcal{E}_\lambda(\widetilde{Q}^{k-1})} + 24(V_{\max} + C||\alpha_*^k||)\sqrt{\frac{2}{L}\log\frac{9}{\delta}} + 32 V_{\max}\sqrt{\frac{2}{L}\log\left(\frac{27(12Le^2)^{2(d+1)}}{\delta}\right)}.$$

with probability $1 - \delta$ (w.r.t. samples), where $||\varphi_i||_\infty \leq C$ and $\mathcal{E}_\lambda(\widetilde{Q}^{k-1}) = \|(\mathcal{T}_1 - \overline{\mathcal{T}}_\lambda)\widetilde{Q}^{k-1}\|_\mu^2$.

Remark 1 (Analysis of the bound). We first notice that the previous bound reduces (up to constants) to the standard bound for FQI when $M = 1$ (see Section B). The bound is composed by three main terms: *(i)* approximation error, *(ii)* estimation error, and *(iii)* transfer error. The approximation error $||f_{\alpha_*^k} - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu$ is the smallest error of functions in $\mathcal{F}$ in approximating the target function $\mathcal{T}_1 \widetilde{Q}^{k-1}$ and it is independent from the transfer algorithm. The estimation error (third and fourth terms in the bound) is due to the finite random samples used to learn $\widehat{Q}^k$ and it depends on the dimensionality d of the function space and it decreases with the total number of samples L with the fast rate of linear spaces ($O(d/L)$ instead of $O(\sqrt{d/L})$). Finally, the transfer error $\mathcal{E}_\lambda$ accounts for the difference between source and target tasks. In fact, samples from source tasks different from the target might bias $\widehat{Q}^k$ towards a wrong solution, thus resulting in a poor approximation of the target function $\mathcal{T}_1 \widetilde{Q}^{k-1}$. It is interesting to notice that the transfer error depends on the difference between the target task and the average MDP $\overline{\mathcal{M}}_\lambda$ obtained by taking a linear combination of the source tasks weighted by the parameters λ. This means that even when each of the source tasks is very different from the target, if there exists a suitable combination which is similar to the target task, then the transfer process is still likely to be effective. Furthermore, $\mathcal{E}_\lambda$ considers the difference in the result of the application of the two Bellman operators to a given function $\widetilde{Q}^{k-1}$. As a result, when the two operators $\mathcal{T}_1$ and $\overline{\mathcal{T}}_\lambda$ have the same reward functions, even if the transition distributions are different (e.g., the total variation $||\mathcal{P}_1(\cdot|x,a) - \overline{\mathcal{P}}_\lambda(\cdot|x,a)||_{\text{TV}}$ is large), their corresponding averages of $\widetilde{Q}^{k-1}$ might still be similar (i.e., $\int \max_{a'} \widetilde{Q}(y,a')\mathcal{P}_1(dy|x,a)$ similar to $\int \max_{a'} \widetilde{Q}(y,a')\overline{\mathcal{P}}_\lambda(dy|x,a)$).

Remark 2 (Comparison to single-task learning). Let $\widehat{Q}_s^k$ be the solution obtained by solving one iteration of FQI with only samples from the source task, the performance bounds of $\widehat{Q}^k$ and $\widehat{Q}_s^k$ can be written as (up to constants and logarithmic factors)

$$\|T(\widehat{Q}^k) - \mathcal{T}_1 \widetilde{Q}^{k-1}\|_\mu \leq ||f_{\alpha_*^k} - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu + (V_{\max} + C||\alpha_*^k||)\sqrt{\frac{1}{L}} + V_{\max}\sqrt{\frac{d}{L}} + \sqrt{\mathcal{E}_\lambda}\,,$$

$$\|T(\widehat{Q}_s^k) - \mathcal{T}_1 \widetilde{Q}^{k-1}\|_\mu \leq ||f_{\alpha_*^k} - \mathcal{T}_1 \widetilde{Q}^{k-1}||_\mu + (V_{\max} + C||\alpha_*^k||)\sqrt{\frac{1}{N_1}} + V_{\max}\sqrt{\frac{d}{N_1}},$$

with $N_1 = \lambda_1 L$ (on average). Both bounds share exactly the same approximation error. The main difference is that $\widehat{Q}_s^k$ uses only N_1 samples and, as a result, has a much bigger estimation error than $\widehat{Q}^k$, which takes advantage of all the L samples transferred from the source tasks. At the same time, $\widehat{Q}^k$ suffers from an additional transfer error which does not exist in the case of $\widehat{Q}_s^k$. Thus, we can conclude that AST is expected to perform better than single-task learning whenever the advantage of using more samples is greater than the bias due to samples coming from tasks different from the target task. This introduces a *transfer tradeoff* between including many source samples, so as to reduce the estimation error, and finding source tasks whose combination leads to a small transfer error. In Section 4 we show how it is possible to define an adaptive transfer algorithm which selects proportions λ so as to keep the transfer error $\mathcal{E}_\lambda$ as small as possible. Finally, in Section 5 we consider a different setting where the maximum number of samples in each source is fixed.

3.2 Propagation Finite-Sample Analysis

We now study how the previous error is propagated through iterations. Let ν be the evaluation norm (i.e., in general different from the sampling distribution μ). We first report two assumptions. [2]

Assumption 1. *[11] Given μ, ν, $p \geq 1$, and an arbitrary sequence of policies $\{\pi_p\}_{p\geq 1}$, we assume that the future-state distribution $\mu\mathcal{P}^1_{\pi_1}\cdots\mathcal{P}^1_{\pi_p}$ is absolutely continuous w.r.t. ν. We assume that $c(p) = \sup_{\pi_1\cdots\pi_p} ||d(\mu\mathcal{P}^1_{\pi_1}\cdots\mathcal{P}^1_{\pi_p})/\nu||_\infty$ satisfies $C_{\mu,\nu} = (1-\gamma^2)^2\sum_p p\gamma^{p-1}c(p) < \infty$.*

We also need the features φ_i to be linearly independent w.r.t. μ.

Assumption 2. *Let $G \in \mathbb{R}^{d\times d}$ be the Gram matrix with $[G]_{ij} = \int \varphi_i(x,a)\varphi_j(x,a)\mu(dx,a)$. We assume that its smallest eigenvalue ω is strictly positive (i.e., $\omega > 0$).*

Using the two previous assumptions we derive the following performance bound for AST.

Theorem 2. *Let Assumptions 1 and 2 hold and the setting be as in Theorem 1. After K iterations, AST returns an action-value function $\widetilde{Q}_K$, whose corresponding greedy policy π_K satisfies*

$$||Q^* - Q^{\pi_K}||_\nu \leq \frac{2\gamma}{(1-\gamma)^{3/2}}\sqrt{C_{\mu,\nu}}\Bigg[4\sup_{g\in\mathcal{F}}\inf_{f\in\mathcal{F}}||f-\mathcal{T}_1 g||_\mu + 5\sup_\alpha \|(\mathcal{T}_1 - \overline{\mathcal{T}}_\lambda)T(f_\alpha)\|_\mu$$
$$+ 56(V_{\max} + \frac{V_{\max}}{\sqrt{\omega}})\sqrt{\frac{2}{L}\log\frac{9K}{\delta}} + 32V_{\max}\sqrt{\frac{2}{L}\log\left(\frac{27K(12Le^2)^{2(d+1)}}{\delta}\right)} + \frac{2V_{\max}}{\sqrt{C_{\mu,\nu}}}\gamma^K\Bigg].$$

Remark (Analysis of the bound). The bound reported in the previous theorem displays few differences w.r.t. to the single-iteration bound. The additional term $O(\gamma^K)$ accounts for the error due to the finite number of iterations of FQI and it decreases exponentially with base γ. The approximation error is now $\sup_g \inf_f ||f - \mathcal{T}^1 g||_\mu$. This term is referred to as the *inherent Bellman error* [11] of the space $\mathcal{F}$ and it is related to how well the Bellman images of functions in $\mathcal{F}$ can be approximated by $\mathcal{F}$ itself. It is possible to show that for particular classes of MDPs (e.g., Lipschitz), if a large enough number of carefully designed features is available, then this term is small. In the estimation error, the norm $||\alpha^k_*||$ is bounded using the linear independency between features (Assumption 2) and the boundedness of the functions $\widetilde{Q}^k$ returned at each iteration. The resulting term has an inverse dependency on the smallest eigenvalue ω which tends to be small whenever the Gram matrix is not well-defined (i.e., the features are almost linearly dependent). The transfer error $\sup_\alpha \|(\mathcal{T}_1 - \overline{\mathcal{T}}_\lambda)T(f_\alpha)\|_\mu$ characterizes the difference between the target and average Bellman operators through the space $\mathcal{F}$. As a result, even MDPs with significantly different rewards and transitions might have a small transfer error because of the functions in $\mathcal{F}$. This introduces a tradeoff in the design of $\mathcal{F}$ between a "large" enough space containing functions able to approximate $\mathcal{T}_1 Q$ (i.e., small approximation error) and a small function space where the Q-functions induced by $\mathcal{T}_1$ and $\overline{\mathcal{T}}_\lambda$ can be closer (i.e., small transfer error). This term also displays interesting similarities with the notion of *discrepancy* introduced in [10] in domain adaptation.

4 Best Average Transfer Algorithm

As discussed in the previous section, the transfer error $\mathcal{E}_\lambda$ plays a crucial role in the comparison with single-task learning. In particular, $\mathcal{E}_\lambda$ is related to the proportions λ inducing the average Bellman operator $\overline{\mathcal{T}}_\lambda$ which defines the target function approximated at each iteration. We now consider the case where the designer has direct access to the source tasks (i.e., it is possible to choose how many samples to draw from each source) and can define an arbitrary proportion λ. In particular, we propose a method that adapts λ at each iteration so as to minimize the transfer error $\mathcal{E}_\lambda$.

We consider the case in which L is fixed as a parameter of the algorithm and $\lambda_1 = 0$ (i.e., no target samples are used in the learning training set). At each iteration k, we need to estimate the quantity $\mathcal{E}_\lambda(\widetilde{Q}^{k-1})$. We assume that for each task additional samples available. Let $\{(X_s, A_s, R_{s,1}, \ldots, R_{s,M})\}_{s=1}^S$ be an *auxiliary* training set where $(X_s, A_s) \sim \mu$ and $R_{s,m} =$

[2] We refer to [11] for a thorough explanation of the concentrability terms.

Input: Space $\mathcal{F} = \text{span}\{\varphi_i, 1 \le i \le d\}$, initial function $\widetilde{Q}^0 \in \mathcal{F}$, number of samples L

Build the auxiliary set $\{(X_s, A_s, R_{s,1}, \ldots, R_{s,M}\}_{s=1}^S$ and $\{Y_{s,1}^t, \ldots, Y_{s,M}^t\}_{t=1}^T$ for each s
for $k = 1, 2, \ldots$ **do**
  Compute $\widehat{\lambda}^k = \arg\min_{\lambda \in \Lambda} \widehat{\mathcal{E}}_\lambda(\widetilde{Q}^{k-1})$
  Run one iteration of AST (Fig. 1) using L samples generated according to $\widehat{\lambda}^k$
end for

Figure 2: A pseudo-code for the Best Average Transfer (BAT) algorithm.

$\mathcal{R}_m(X_s, A_s)$. In each state-action pair, we generate T next states for each task, that is $Y_{s,m}^t \sim \mathcal{P}_m(\cdot|X_s, A_s)$ with $t = 1, \ldots, T$. Thus, for any function Q we define the estimated transfer error as

$$\widehat{\mathcal{E}}_\lambda(Q) = \frac{1}{S}\sum_{s=1}^{S}\left[R_{s,1} - \sum_{m=2}^{M}\lambda_m R_{s,m} + \frac{\gamma}{T}\sum_{t=1}^{T}\Big(\max_{a'} Q(Y_{s,1}^t, a') - \sum_{m=2}^{M}\lambda_m \max_{a'} Q(Y_{s,m}^t, a')\Big)\right]^2. \quad (3)$$

At each iteration, the algorithm *Best Average Transfer* (BAT) (Fig. 2) first computes $\widehat{\lambda}^k = \arg\min_{\lambda \in \Lambda} \widehat{\mathcal{E}}_\lambda(\widetilde{Q}^{k-1})$, where Λ is the $(M\text{-}2)$-dimensional simplex, and then runs an iteration of AST with samples generated according to the proportions $\widehat{\lambda}^k$. We denote by λ_*^k the best combination at iteration k, that is

$$\lambda_*^k = \arg\min_{\lambda \in \Lambda} \mathcal{E}_\lambda(\widetilde{Q}^{k-1}) = \arg\min_{\lambda \in \Lambda} \mathbb{E}_\mu\left[\Big(\sum_{m=2}^{M}\lambda_m(\mathcal{T}^m\widetilde{Q}^{k-1})(x,a) - (\mathcal{T}^1\widetilde{Q}^{k-1})(x,a)\Big)^2\right]. \quad (4)$$

The following performance guarantee can be proved for BAT.

Lemma 1. *Let $\{(X_s, A_s, R_s^1, \ldots, R_s^M)\}_{s=1}^S$ be a training set where $(X_s, A_s) \overset{iid}{\sim} \mu$ and $R_s^m = \mathcal{R}^m(X_s, A_s)$ and for each state-action pair and for each task m, T next states $Y_{s,t}^m \sim \mathcal{P}^m(\cdot|X_s, A_s)$ with $t = 1, \ldots, T$ are available. For any fixed bounded function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max})$, the $\widehat{\lambda}$ returned by minimizing Eq. 3 is such that*

$$\mathcal{E}_{\widehat{\lambda}}(Q) - \mathcal{E}_{\lambda_*}(Q) \le 2V_{\max}\sqrt{\frac{(M-2)\log 4S/\delta}{S}} + 16V_{\max}^2\frac{\log 4SM/\delta}{T} \quad (5)$$

with probability $1 - \delta$.

From the previous lemma the approximation performance of BAT at each iteration follows.

Theorem 3. *Let $\widetilde{Q}^{k-1}$ be the function returned at the previous iteration and $\widehat{Q}_{BAT}^k$ the function returned by the BAT algorithm (Fig. 2). Then for any $0 < \delta \le 1$, $\widehat{Q}_{BAT}^k$ satisfies*

$$\begin{aligned}||T(\widehat{Q}_{BAT}^k) - \mathcal{T}_1\widetilde{Q}^{k-1}||_\mu &\le 4||f_{\alpha_*^k} - \mathcal{T}_1\widetilde{Q}^{k-1}||_\mu + 5\sqrt{\mathcal{E}_{\lambda_*^k}(\widetilde{Q}^{k-1})} \\ &+ 5\sqrt{2V_{\max}}\left(\frac{(M-2)\log 8S/\delta}{S}\right)^{1/4} + 20V_{\max}\sqrt{\frac{\log 8SM/\delta}{T}} \\ &+ 24(V_{\max} + C||\alpha_*^k||)\sqrt{\frac{2}{L}\log\frac{18}{\delta}} + 32V_{\max}\sqrt{\frac{2}{L}\log\left(\frac{54(12Le^2)^{2(d+1)}}{\delta}\right)}.\end{aligned}$$

with probability $1 - \delta$.

Remark 1 (Comparison with AST and single-task learning). The analysis of the bound shows that BAT outperforms AST whenever the advantage in achieving the smallest possible transfer error $\mathcal{E}_{\lambda_*^k}$ is larger than the additional estimation error due to the auxiliary training set. It is also interesting to compare BAT to single-task learning. In fact, BAT performs better than single-task learning

whenever the best possible combination of source tasks has a small transfer error and the additional estimation error related to the auxiliary training set is smaller than the estimation error in single-task learning. In particular, this means that $O((M/S)^{1/4}) + O((1/T)^{1/2})$ should be smaller than $O((d/N)^{1/2})$ (with N the number of target samples). The number of calls to the generative model for BAT is ST. In order to have a fair comparison with single-task learning we set $S = N^{2/3}$ and $T = N^{1/3}$, then we obtain the condition $M \leq d^2 N^{-4/3}$ that constrains the number of tasks to be smaller than the dimensionality of the function space $\mathcal{F}$. We remark that the dependency of the auxiliary estimation error on M is due to the fact that the λ vectors (over which the transfer error is optimized) belong to the simplex Λ of dimensionality M-2. Hence, the previous condition suggests that, in general, adaptive transfer methods may significantly improve the transfer performance (i.e., in this case a smaller transfer error) at the cost of additional sources of errors which depend on the dimensionality of the search space used to adapt the transfer process (i.e., in this case Λ).

Remark 2 (Iterations). BAT recomputes the proportions $\widehat{\lambda}^k$ at each iteration k. In fact a combination λ_1 approximating well the reward function $\mathcal{R}_1$ at the first iteration (i.e., $\mathcal{R}_1 \approx \overline{\mathcal{R}}_{\lambda^1}$) does not necessarily have a small transfer error $||(\mathcal{T}_1 - \mathcal{T}_{\lambda^1})\widetilde{Q}^1||_\mu$ at the second iteration. We further investigate how the best source combination changes through iterations in the experiments of Section 6.

Remark 3 (Best source combination). The previous theorem shows that BAT achieves the smallest transfer error $\mathcal{E}_{\lambda_*^k}(\widetilde{Q}^{k-1})$ at the cost of an additional estimation error which scales with the size of the auxiliary training set as $O((M/S)^{1/4}) + O((1/T)^{1/2})$. We notice that the first term of the estimation error depends on how well the μ is approximated by using a finite number S of state-action pairs and it has a slower rate w.r.t. the other terms. The second term depends on the number of next states T simulated at each state-action pair which are used to estimate the value of the Bellman operators. As a result, in order to reduce the estimation error we need to increase both S and the number of next states T in each state-action pair. It is interesting to notice that similar estimation errors appear in FVI [11] where the optimal Bellman operator is approximated by Monte-Carlo estimation.

Remark 4 (Training set). The implicit assumption in the definition of the auxiliary training set is that it is possible to generate a series of next states and rewards for all the tasks at the same state-action pairs. If the source training sets are fixed in advance and the designer has no access to the source tasks, then this assumption is not verified and it is not possible to test the similarity between the MDP $\overline{\mathcal{M}}$ and the target task. Nonetheless, if the generative model for the source tasks is available at learning time, the auxiliary training set could be generated before the learning phase actually begins. Furthermore, in the theoretical analysis, BAT does not use the samples in the auxiliary training set at learning time. A trivial improvement is to include the auxiliary samples to the training set.

Remark 5 (Comparison to other transfer methods). In [8] a method to compute the similarity between MDPs is proposed. In particular, the authors introduce the definition of *compliance* as the average probability of the target samples to be generated from an sample-based estimation of the source MDPs. The compliance is later used to determine the proportion of samples to be transferred from each of the source tasks. Although this algorithm shares a similar objective as BAT, they use different notions of similarity. In particular, the method in [8] tries to identify source tasks which are *individually* similar to the target task, while the transfer error minimized in BAT considers the *average* MDP obtained by the transfer process. Furthermore, the notion of compliance tries to measures the overall distance between two MDPs, while $\mathcal{E}_\lambda(Q)$ always measures the distance of the images of a function Q through two different Bellman operators.

Remark 6 (Computational complexity). Finally, we notice that the minimization of $\widehat{\mathcal{E}}_\lambda$ is a convex quadratic problem since the objective function is convex in λ and λ belongs to the $(M$-2)-dimensional simplex.

5 Best Transfer Trade-off Algorithm

The previous algorithm is proved to successfully estimate the combination of source tasks which better approximates the Bellman operator of the target task. Nonetheless, BAT relies on the implicit

Input: Linear space $\mathcal{F} = \text{span}\{\varphi_i, 1 \leq i \leq d\}$, initial function $\widetilde{Q}^0 \in \mathcal{F}$, maximum number of samples available for each task N_m, transfer parameter c

Build a training set $\{X_s, A_s, R_s^1, \ldots, R_s^M\}_{s=1}^S$ and the next states $\{Y_{s,t}^1, \ldots, Y_{s,t}^M\}_{t=1}^T$ for each state-action pair

for $k = 1, 2, \ldots$ **do**
  Compute $\hat{\beta} = \arg\min_{\beta \in [0,1]^M} \widehat{\mathcal{E}}_\beta + c\sqrt{\frac{d}{\sum_{m=1}^M \beta_m N_m}}$
  Run one iteration of AST (Fig. 1) using L samples generated according to $\hat{\beta}$
end for

Figure 3: A pseudo-code for Best Tradeoff Transfer (BTT).

assumption that L samples can always be generated from any source task [3] and it cannot be applied to the case where the number of source samples is limited. Here we consider the more challenging case where the designer has still access to the source tasks but only a limited number of samples is available in each of them. In this case, an adaptive transfer algorithm should solve a tradeoff between selecting as many samples as possible, so as to reduce the estimation error, and choosing the proportion of source samples properly, so as to control the transfer error. The solution of this tradeoff may return non-trivial results, where source tasks similar to the target task but with few samples are removed in favor of a pool of tasks whose average roughly approximate the target task but can provide a larger number of samples.

Here we introduce the *Best Tradeoff Transfer* (BTT) algorithm (see Figure 3). Similar to BAT, it relies on an auxiliary training set to solve the tradeoff. We denote by N_m the maximum number of samples available for source task m. Let $\beta \in [0,1]^M$ be a weight vector, where β_m is the fraction of samples from task m used in the transfer process. We denote by $\mathcal{E}_\beta$ ($\widehat{\mathcal{E}}_\beta$) the transfer error (the estimated transfer error) with proportions λ where $\lambda_m = (\beta_m N_m)/\sum_{m'}(\beta_{m'} N_{m'})$. At each iteration k, BTT returns the vector β which optimizes the tradeoff between estimation and transfer errors, that is

$$\hat{\beta}^k = \arg\min_{\beta \in [0,1]^M} \Big(\widehat{\mathcal{E}}_\beta(\widetilde{Q}^{k-1}) + \tau \sqrt{\frac{d}{\sum_{m=1}^M \beta_m N_m}} \Big), \tag{6}$$

where τ is a parameter. While the first term accounts for the transfer error induced by β, the second term is the estimation error due to the total amount of samples used by the algorithm.

Unlike AST and BAT, BTT is a heuristic algorithm motivated by the performance bound in Theorem 1 and we do not provide any theoretical guarantee about its performance. The main technical difficulty w.r.t. the previous algorithms is that the setting considered here does not match the random task design assumption (see Def. 1) since the number of source samples is constrained by N_m. As a result, given a proportion vector λ, we cannot assume samples to be drawn at random according to a multinomial of parameters λ. Without this assumption, it is an open question whether a similar bound to AST and BAT could be derived. Nonetheless, the experimental results reported in Section 6 show the effectiveness of BTT in solving the transfer tradeoff.

6 Experiments

In this section, we report and discuss preliminary experimental results of the transfer algorithms introduced in the previous sections. The main objective is to illustrate the functioning of the algorithms and compare their results with the theoretical findings. Thus, we focus on a simple problem and we leave more challenging problems for future work.

We consider a continuous extension of the 50-state variant of the chain walk problem proposed in [6]. The state space is described by a continuous state variable x and two actions are available: one that

[3] If $\lambda_m = 1$ for task m, then the algorithm would generate all the L training samples from task m.

Table 1: Parameters for the first set of tasks

tasks	p	l	η	Reward
$\mathcal{M}_1$	0.9	1	0.1	$+1$ in $[-11,-9] \cup [9,11]$
$\mathcal{M}_2$	0.9	2	0.1	-5 in $[-11,-9] \cup [9,11]$
$\mathcal{M}_3$	0.9	1	0.1	$+5$ in $[-11,-9] \cup [9,11]$
$\mathcal{M}_4$	0.9	1	0.1	$+1$ in $[-6,-4] \cup [4,6]$
$\mathcal{M}_5$	0.9	1	0.1	-1 in $[-6,-4] \cup [4,6]$

Table 2: Parameters for the second set of tasks

tasks	p	l	η	Reward
$\mathcal{M}_1$	0.9	1	0.1	$+1$ in $[-11,-9] \cup [9,11]$
$\mathcal{M}_6$	0.7	1	0.1	$+1$ in $[-11,-9] \cup [9,11]$
$\mathcal{M}_7$	0.1	1	0.1	$+1$ in $[-11,-9] \cup [9,11]$
$\mathcal{M}_8$	0.9	1	0.1	-5 in $[-11,-9] \cup [9,11]$
$\mathcal{M}_9$	0.7	1	0.5	$+5$ in $[-11,-9] \cup [9,11]$

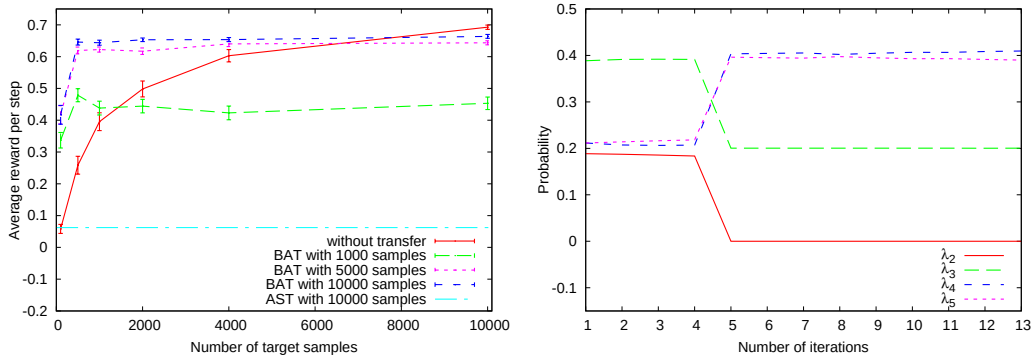

Figure 4: Transfer from $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$, $\mathcal{M}_5$. *Left:* Comparison between single-task learning, AST with $L = 10000$, BAT with $L = 1000, 5000, 10000$. *Right:* Source task probabilities estimated by BAT algorithm as a function of FQI iterations.

moves toward *left* and the other toward *right*. With probability p each action makes a step of length l, affected by a noise η, in the intended direction, while with probability $1-p$ it moves in the opposite direction. For the target task $\mathcal{M}_1$, the state–transition model is defined by the following parameters: $p = 0.9$, $l = 1$, and η is uniform in the interval $[-0.1, 0.1]$. The reward function provides $+1$ when the system state reaches the regions $[-11, -9]$ and $[9, 11]$ and 0 elsewhere. Furthermore, to evaluate the performance of the transfer algorithms previously described, we considered eight source tasks $\{\mathcal{M}_2, \ldots, \mathcal{M}_9\}$ whose state–transition model parameters and reward functions are reported in Tab. 1 and 2. To approximate the Q-functions, we use a linear combination of 20 radial basis functions. In particular, for each action, we consider 9 Gaussians with means uniformly spread in the interval $[-20, 20]$ and variance equal to 16, plus a constant feature. The number of iterations for the FQI algorithm has been empirically fixed to 13. Samples are collected through a sequence of episodes, each one starting from the state $x_0 = 0$ with actions chosen uniformly at random. For all the experiments, we average over 100 runs and we report standard deviation error bars.

We first consider the *pure* transfer problem where no target samples are actually used in the learning training set (i.e., $\lambda_1 = 0$). The objective is to study the impact of the transfer error due to the use of source samples and the effectiveness of BAT in finding a suitable combination of source tasks. The left plot in Fig. 4 compares the performances of FQI with and without the transfer of samples from the first four tasks listed in Tab. 1. In case of single-task learning, the number of target samples refers to the samples used at learning time, while for BAT it represents the size S of the auxiliary training set used to estimate the transfer error. Thus, while in single-task learning the performance increases with the target samples, in BAT they just make estimation of $\mathcal{E}_\lambda$ more accurate. The number of source samples added to the auxiliary set for each target sample was empirically fixed to one ($T = 1$). We first run AST with $L = 10000$ and $\lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = 0.25$ (which on average corresponds to 2500 samples from each source). As it can be noticed by looking at the models in Tab. 1, this combination is very different from the target model and AST does not learn any good policy. On the other hand, even with a small set of auxiliary target samples, BAT is able to learn good policies. Such result is due to the existence of linear combinations of source tasks which closely approximate the target task $\mathcal{M}_1$ at each iteration of FQI. An example of the proportion coefficients computed at each iteration of BAT is shown in the right plot in Fig. 4. At the first iteration, FQI produces an approximation of the reward function. Given the first four source tasks,

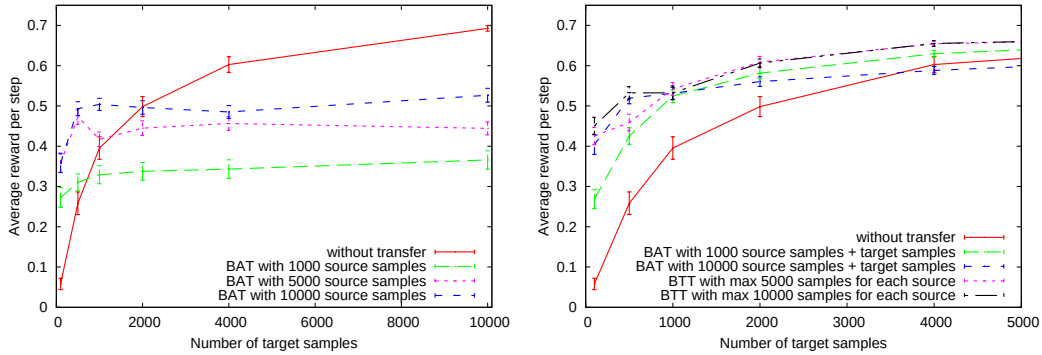

Figure 5: Transfer from $\mathcal{M}_6$, $\mathcal{M}_7$, $\mathcal{M}_8$, $\mathcal{M}_9$. *Left:* Comparison between single-task learning and BAT with $L = 1000, 5000, 10000$. *Right:* Comparison between single-task learning, BAT with $L = 1000, 10000$ in addition to the target samples, and BTT ($\tau = 0.75$) with 5000 and 10000 samples for each source task. To improve readability, the plot is truncated at 5000 target samples.

BAT finds a combination ($\lambda \simeq (0.2, 0.4, 0.2, 0.2)$) that produces the same reward function as $\mathcal{R}_1$. However, after a few FQI iterations, such combination is no more able to accurately approximate functions $\mathcal{T}_1\widetilde{Q}$. In fact, the state–transition model of task $\mathcal{M}_2$ is different from all the other ones (the step length is doubled). As a result, the coefficient λ_2 drops to zero, while a new combination among the other source tasks is found. Note that BAT significantly improves single-task learning, in particular when very few target samples are available.

In the general case, the target task cannot be obtained as any combination of the source tasks, as it happens by considering the second set of source tasks ($\mathcal{M}_6$, $\mathcal{M}_7$, $\mathcal{M}_8$, $\mathcal{M}_9$). The impact of such situation on the learning performance of BAT is shown in the left plot in Fig. 5. Note that, when a few target samples are available, the transfer of samples from a combination of the source tasks using the BAT algorithm is still beneficial. On the other hand, the performance attainable by BAT is bounded by the transfer error corresponding to the best source task combination (which in this case is large). As a result, single-task FQI quickly achieves a better performance.

Results presented so far for the BAT transfer algorithm assume that FQI is trained only with the samples obtained through combinations of source tasks. Since a number of target samples is already available in the auxiliary training set, a trivial improvement is to include them in the training set together with the source samples (selected according to the proportions computed by BAT). As shown in the plot in the right side of Fig. 5 this leads to a significant improvement. From the behavior of BAT it is clear that with a small set of target samples, it is better to transfer as many samples as possible from source tasks, while as the number of target samples increases, it is preferable to reduce the number of samples obtained from a combination of source tasks that actually does not match the target task. In fact, for $L = 10000$, BAT has a much better performance at the beginning but it is then outperformed by single-task learning. On the other hand, for $L = 1000$ the initial advantage is small but the performance remains close to single-task FQI for large number of target samples. This experiment highlights the tradeoff between the need of samples to reduce the estimation error and the resulting transfer error when the target task cannot be expressed as a combination of source tasks (see Section 5). BTT algorithm provides a principled way to address such tradeoff, and, as shown by the right plot in Fig. 5, it exploits the advantage of transferring source samples when a few target samples are available, and it reduces the weight of the source tasks (so as to avoid large transfer errors) when samples from the target task are enough. It is interesting to notice that increasing the number of samples available for each source task from 5000 to 10000 improves the performance in the first part of the graph, while keeping unchanged the final performance. This is due to the capability of the BTT algorithm to avoid the transfer of source samples when there is no need for them, thus avoiding *negative transfer* effects.

7 Conclusions

In this paper, we formalized and studied the sample-transfer problem. We first derived a finite-sample analysis of the performance of a simple transfer algorithm which includes all the source samples into the training set used to solve a given target task. At the best of our knowledge, this is the first theoretical result for a transfer algorithm in RL showing the potential benefit of transfer over single-task learning. Then, in the case when the designer has direct access to the source tasks, we introduced an adaptive algorithm which selects the proportion of source tasks so as to minimize the bias due to the use of source samples. Finally, we considered a more challenging setting where the number of samples available in each source task is limited and a tradeoff between the amount of transferred samples and the similarity between source and target tasks must be solved. For this setting, we proposed a principled adaptive algorithm. Finally, we report a detailed experimental analysis on a simple problem which confirms and supports the theoretical findings.

This work opens several directions for future work.

- *Transfer with transformations.* In many problems, there exist simple transformations to the source tasks dynamics and reward which would increase their similarity w.r.t. the target task, thus making the transfer process more effective. How affine transformations could be used in the adaptive transfer algorithms presented in this paper is an interesting direction for future work. In particular, it is an open question whether the cost (in terms of samples) of finding a suitable transformation would be effectively counter-balanced by transferring more similar samples.
- *Transfer between tasks with different state-action spaces.* In many real applications source and target tasks might have a different number of state variables and different actions. Thus, the current work should be extended to the more general case of tasks with different state-action spaces and it should be integrated with inter-task mapping transfer methods (see [14]).
- *Transfer with fixed tasks design.* Definition 1 prescribes the process used to generate the training set used in learning the target task. At each state-action pair, the sample is generated from a source task chosen at random according to a multinomial distribution. When the designer has no access to the source tasks and their samples are generated beforehand, this generative model is not reasonable. A different model (*fixed* tasks design) should be defined where each sample is coming from a specific source which is fixed in advance. An interesting direction for future work is to understand how this different generative model affects the performance of the transfer algorithm and whether it is possible to define effective adaptive algorithms for this case.

Acknowledgments This work was supported by French National Research Agency through the projects EXPLO-RA n° ANR-08-COSI-004, by Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council and FEDER through the "contrat de projets état region 2007–2013", and by PASCAL2 European Network of Excellence. The research leading to these results has also received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n 231495.

References

[1] Andras Antos, Rémi Munos, and Csaba Szepesvari. Fitted Q-iteration in continuous action-space MDPs. In *Neural Information Processing Systems*, Vancouver, Canada, 2007.

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 2010.

[3] Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9:1757–1774, 2008.

[4] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *J. Mach. Learn. Res.*, 6:503–556, December 2005.

[5] L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer-Verlag, New York, 2002.

[6] M.G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[7] A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of LSTD. Technical Report inria-00482189, INRIA, 2010.

[8] A. Lazaric, M. Restelli, and A. Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the Twenty-Fifth Annual International Conference on Machine Learning (ICML'08)*, pages 544–551, 2008.

[9] Alessandro Lazaric. *Knowledge Transfer in Reinforcement Learning*. PhD thesis, Poltecnico di Milano, 2008.

[10] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proceedings of the 22nd Conference on Learning Theory (COLT'09)*, 2009.

[11] R. Munos and Cs. Szepesvári. Finite time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9:815–857, 2008.

[12] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[13] Matthew E. Taylor, Nicholas K. Jong, and Peter Stone. Transferring instances for model-based reinforcement learning. In *Proceedings of the European Conference on Machine Learning (ECML'08)*, pages 488–505, 2008.

[14] Matthew E. Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(1):1633–1685, 2009.

Input: Linear space $\mathcal{F} = \text{span}\{\varphi_i, 1 \le i \le d\}$, initial function $\widetilde{Q}^0$

for $k = 1, 2, \ldots$ **do**
  Draw training samples $\{(X_n, A_n, Y_n, R_n)\}_{n=1}^N$
  Build the feature matrix $\Phi = [\phi(X_1, A_1)^\top; \ldots; \phi(X_n, A_n)^\top]$
  Compute the vector $p_n = R_n + \gamma \max_{a' \in \mathcal{A}} \widetilde{Q}^{k-1}(Y_n, a')$
  Compute the projection $\hat{\alpha}^k = (\Phi^\top \Phi)^{-1} \Phi^\top p$
  Return the truncated function $\widetilde{Q}^k = T(f_{\hat{\alpha}_k})$
end for

Figure 6: A pseudo-code for Fitted Q-iteration.

A Additional Notation

Besides the notation introduced in Section 2, here we introduce additional symbols used in the proofs. We define two empirical norms on functions and vectors. Given a set of N state-action pairs $\{(X_n, A_n)\}_{n=1}^N$ drawn i.i.d. from μ we define the empirical norm $||f||_{\hat{\mu}}$ as

$$||f||_{\hat{\mu}}^2 = \frac{1}{N} \sum_{n=1}^N f(X_n, A_n)^2.$$

Similarly, given a vector $y \in \mathbb{R}^N$ we define the empirical norm $||y||_N$ as

$$||y||_N^2 = \frac{1}{N} \sum_{n=1}^N y_n^2.$$

Given a set of N state-action pairs $\{(X_n, A_n)\}_{n=1}^N$, let $\Phi = [\phi(X_1, A_1)^\top; \ldots; \phi(X_N, A_N)^\top]$ be the feature matrix defined at the states $\{(X_n, A_n)\}_{n=1}^N$, and $\mathcal{F}_n = \{\Phi\alpha,\ \alpha \in \mathbb{R}^d\} \subset \mathbb{R}^N$ be the corresponding vector space. We denote by $\hat{\Pi} : \mathbb{R}^N \to \mathcal{F}_N$ the empirical orthogonal projection onto $\mathcal{F}_N$, defined by

$$\hat{\Pi} y = \arg\min_{z \in \mathcal{F}_N} ||y - z||_N. \tag{7}$$

Note that the orthogonal projection $\hat{\Pi} y$ of any $y \in \mathbb{R}^N$ always exists and is unique.

B Fitted Q-iteration with Linear Spaces

Although fitted iterative methods have been already analyzed in detail in [11] and [1], at the best of our knowledge no explicit finite-sample bounds for FQI with linear spaces is available. Since at each iteration, FQI solves an explicit regression problem, the derivation is mostly a straightforward application of regression bounds for linear spaces and quadratic loss. Here we just report the result and the proof of the single iteration error for the so-called fixed and random samples design settings.

In Algorithm 6 we report the structure of the algorithm.

B.1 Fixed Samples Design

Similar to the analysis of LSTD in [7] we first derive the fixed design bound (i.e., the performance is evaluated exactly on the states in the training set).

Theorem 4. *Let $\mathcal{F} = \{\phi(\cdot,\cdot)^\top \alpha, \alpha \in \mathbb{R}^d\}$ be a d-dimensional linear space. Let $\{(x_n, a_n, Y_n, R_n)\}_{n=1}^N$ be the training set where $\{(x_n, a_n)\}_{n=1}^N$ is an arbitrary sequence of state-action pairs, $Y_n \sim \mathcal{P}(\cdot|x_n, a_n)$, and $R_n = \mathcal{R}(x_n, a_n)$. Given a function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A}, V_{\max})$, let $q \in \mathbb{R}^N$ be the vector whose components are $q_n = (\mathcal{T}Q)(x_n, a_n)$ and $\hat{q}$ be the solution of a single iteration of fitted value iteration. Then with probability $1 - \delta$ (w.r.t. the random next states Y_n), $\hat{q}$*

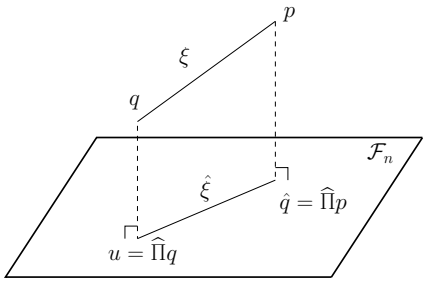

Figure 7: This figure shows that the vectors used in the proof of Theorem 4.

satisfies

$$||\hat{q} - q||_N \leq ||\hat{\Pi} q - q||_N + 4V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{3(9Ne^2)^{d+1}}{\delta}\right)}. \tag{8}$$

Proof. We denote by $u \in \mathbb{R}^N$ the orthogonal projection of the target vector q onto the vector space $\mathcal{F}_N$, that is $u = \hat{\Pi} q$. By the definition of orthogonal projection and the Pythagorean theorem we decompose the error $||\hat{q} - q||_N$ as

$$||\hat{q} - q||_N^2 = ||\hat{q} - u||_N^2 + ||u - q||_N^2, \tag{9}$$

where the first term represents the estimation error and the second term is the approximation error (see Fig. 7). We denote by $\xi_n = p_n - q_n$ the noise in the observations p w.r.t. q. It is easy to notice that

$$\mathbb{E}\left[\xi_n\right] = \mathbb{E}_{Y \sim \mathcal{P}(\cdot|x_n,a_n)}\left[\mathcal{R}(x_n, a_n, Y) + \gamma \max_{a' \in \mathcal{A}} Q(Y, a')\right] - (\mathcal{T}Q)(x_n, a_n) = 0, \tag{10}$$

and that $|\xi_n| \leq 2V_{\max}$. We also define the projected noise $\hat{\xi}_n = \hat{q}_n - u_n$, that is $\hat{\xi} = \hat{\Pi}\xi$. Thus, we can rewrite the estimation error as

$$||\hat{q} - u||_N^2 = ||\hat{\xi}||_N^2 = \langle \hat{\xi}, \hat{\xi} \rangle = \langle \xi, \hat{\xi} \rangle, \tag{11}$$

where the last equality follows from the fact that $\hat{\xi}$ is the orthogonal projection of ξ. Since $\hat{\xi} \in \mathcal{F}_N$, let $f_\beta \in \mathcal{F}$ be any function such that $f_\beta(x_n, a_n) = \hat{\xi}_n$, and by a straightforward application of a variation of Pollard's inequality [5] we obtain

$$\begin{aligned}\langle \xi, \hat{\xi} \rangle = \frac{1}{N}\sum_{n=1}^{N} \xi_n f_\beta(x_n, a_n) &\leq 4V_{\max}\left(\frac{1}{N}\sum_{n=1}^{N} f_\beta(x_n, a_n)^2\right)^{1/2} \sqrt{\frac{2}{N}\log\left(\frac{3(9Ne^2)^{d+1}}{\delta}\right)} \\ &= 4V_{\max}||\hat{\xi}||_N \sqrt{\frac{2}{N}\log\left(\frac{3(9Ne^2)^{d+1}}{\delta}\right)}\end{aligned} \tag{12}$$

with probability $1 - \delta$. Thus from equation 11 we bound the estimation error by

$$||\hat{q} - u||_N \leq 4V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{3(9Ne^2)^{d+1}}{\delta}\right)}. \tag{13}$$

Putting together the estimation error bound and the approximation error term, the statement of the theorem follows. □

B.2 Random Samples Design

While in the previous section we analyzed the performance of FQI on the very same state-action pairs in the training set, we now focus on the generalization (i.e., prediction) performance on the whole state-action space.

Let $\hat{Q}$ be any function $f_{\hat{\alpha}} \in \mathcal{F}$ satisfying $\Phi\hat{\alpha} = \hat{q}$, where $\hat{q}$ is the vector defined in the previous section. Then we derive the following theorem.

Theorem 5. *Let $\mathcal{F} = \{\phi(\cdot,\cdot)^\top \alpha, \alpha \in \mathbb{R}^d\}$ be a d-dimensional linear space. Let $\{(X_n, A_n, Y_n, R_n)\}_{n=1}^N$ be the training set where $(X_n, A_n) \overset{iid}{\sim} \mu$, $Y_n \sim \mathcal{P}(\cdot|X_n, A_n)$, and $R_n = \mathcal{R}(X_n, A_n)$. Given a function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A}, V_{\max})$, let $\widehat{Q}$ be the solution of a single iteration of fitted value iteration. Then with probability $1-\delta$ (w.r.t. the samples and the next states), $\widehat{Q}$ satisfies*

$$
\begin{aligned}
||T(\widehat{Q}) - \mathcal{T}Q||_\mu \leq 4 \inf_{f_\alpha \in \mathcal{F}} ||f_\alpha - \mathcal{T}Q||_\mu \\
&+ 24(V_{\max} + L||\alpha_*||)\sqrt{\frac{2}{N}\log\frac{9}{\delta}} \\
&+ 32 V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{27(12Ne^2)^{2(d+1)}}{\delta}\right)}.
\end{aligned} \tag{14}
$$

Proof. The proof mainly relies on the application of concentration of measures inequalities for linear spaces to the deterministic design bound in Theorem 4.

Let $f_{\hat{\alpha}_*} \in \mathcal{F}$ be any function such that $f_{\hat{\alpha}_*}(X_n) = (\hat{\Pi} q)_n$, thus the approximation error $||\hat{\Pi} q - q||_N$ can be rewritten as $||f_{\hat{\alpha}_*} - \mathcal{T}Q||_{\hat{\mu}}$. Furthermore we denote by $f_{\alpha_*} = \Pi(\mathcal{T}Q)$, that is the best approximation of the target function $\mathcal{T}Q$ onto $\mathcal{F}$ w.r.t. the distribution μ. Since $f_{\hat{\alpha}_*}$ is the minimizer of the empirical squared error, any function in $\mathcal{F}$ different from $f_{\hat{\alpha}_*}$ has a bigger empirical loss, thus we obtain

$$
||f_{\hat{\alpha}_*} - \mathcal{T}Q||_{\hat{\mu}} \leq ||f_{\alpha_*} - \mathcal{T}Q||_{\hat{\mu}} \leq 2||f_{\alpha_*} - \mathcal{T}Q||_\mu + 12(V_{\max} + L||\alpha_*||)\sqrt{\frac{2}{N}\log\frac{3}{\delta'}}, \tag{15}
$$

with probability $1-\delta'$, where the second inequality is an application of a variation of Theorem 11.2 in [5] with a bound $||f_{\alpha_*} - \mathcal{T}Q||_\infty \leq V_{\max} + L||\alpha_*||$. Similar, we notice that the left hand side of Eq. 8 is $||\hat{q} - q||_N = ||\widehat{Q} - \mathcal{T}^* Q||_{\hat{\mu}}$ and we obtain

$$
2||\widehat{Q} - \mathcal{T}Q||_{\hat{\mu}} \geq 2||T(\widehat{Q}) - \mathcal{T}Q||_{\hat{\mu}} \geq ||T(\widehat{Q}) - \mathcal{T}Q||_\mu - 24 V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{9(12eN)^{2(d+1)}}{\delta'}\right)} \tag{16}
$$

with probability $1-\delta'$, where the second inequality is an application of a variation of Theorem 11.2 in [5]. Putting together Eqs 8, 15, and 16 we obtain

$$
\begin{aligned}
||T(\widehat{Q}) - \mathcal{T}Q||_\mu \leq & 2\Bigg(2||f_{\alpha_*} - \mathcal{T}Q||_\mu + 12(V_{\max} + L||\alpha_*||)\sqrt{\frac{2}{N}\log\frac{3}{\delta'}} + \\
& + 4V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{3(9Ne^2)^{d+1}}{\delta'}\right)}\Bigg) + 24 V_{\max}\sqrt{\frac{2}{N}\log\left(\frac{9(12eN)^{2(d+1)}}{\delta'}\right)}
\end{aligned}
$$

Finally, by setting $\delta = 3\delta'$ the statement follows. □

C Analysis of AST

C.1 Proof of Theorem 1

Proof. Since the proof follows similar steps as in the proof of Theorem 5, we discuss here only the fixed samples design bound. We define the vector $p \in \mathbb{R}^L$ such that for any $l = 1, \dots, L$, $p_l = \sum_{m=1}^M \mathbb{I}\{M_l = m\}(R_l^m + \gamma \max_{a'} Q(Y_l^m, a'))$. The target vector $q \in \mathbb{R}^L$ is the image of the function Q through the average optimal Bellman operator. In fact, by defining $q_l = (\overline{\mathcal{T}}_\lambda Q)(X_l, A_l)$ we obtain a zero-mean noise vector $\xi_l = p_l - q_l$ such that $\mathbb{E}[\xi_l] = 0$ and $|\xi_l| \leq 2V_{\max}$. [4]

[4] The expectation is taken w.r.t. both the random realization of the reward R_l^m and next state Y_l^m and task index M_l.

The statement of the theorem simply follows by decomposing the prediction error of $\widehat{Q}$ as

$$\|T(\widehat{Q}) - \mathcal{T}_1 Q\|_\mu \leq \|T(\widehat{Q}) - \overline{\mathcal{T}}_\lambda Q\|_\mu + \|\overline{\mathcal{T}}_\lambda Q - \mathcal{T}_1 Q\|_\mu. \tag{17}$$

By substituting $\|T(\widehat{Q}) - \overline{\mathcal{T}}_\lambda Q\|_\mu$ with a FQI bound w.r.t. the target function $\overline{\mathcal{T}}_\lambda Q$ we obtain

$$\begin{aligned}\|T(\widehat{Q}) - \mathcal{T}_1 Q\|_\mu \leq 4||f_\alpha - \overline{\mathcal{T}}_\lambda Q||_\mu + \|\overline{\mathcal{T}}_\lambda Q - \mathcal{T}_1 Q\|_\mu \qquad (18)\\ + 24(V_{\max} + C||\alpha||)\sqrt{\frac{2}{L}\log\frac{9}{\delta}}\\ + 32 V_{\max}\sqrt{\frac{2}{L}\log\left(\frac{27(12Le^2)^{2(d+1)}}{\delta}\right)}. \qquad (19)\end{aligned}$$

By rewriting the approximation error as $||f_\alpha - \overline{\mathcal{T}}_\lambda Q||_\mu \leq ||f_\alpha - \mathcal{T}^1 Q||_\mu + ||\mathcal{T}^1 Q - \overline{\mathcal{T}}_\lambda Q||_\mu$ and using $\alpha = \alpha_*$ the final bound follows.

□

C.2 Proof of Theorem 2

Proof. [Sketch] The main structure of the proof is exactly the same as in [11]. The main differences are due to the use of linear spaces and the transfer error. Following the passages in the proof of Theorem 2 in [11], we obtain

$$||Q^* - Q^{\pi_K}||_\nu \leq \frac{2\gamma}{(1-\gamma)^{3/2}}\left[\sqrt{C_{\mu,\nu}}\max_k ||T(\widehat{Q}^k) - \mathcal{T}^1\widetilde{Q}^k||_\mu + 2V_{\max}\gamma^K\right].$$

Thus, we need to study all the terms in the statement of Theorem 1 affected by the maximization over the iterations.

Approximation error. The approximation term becomes

$$\max_k \min_{f\in\mathcal{F}} ||f - \mathcal{T}^1\widetilde{Q}^k||_\mu \leq \sup_{g\in\mathcal{F}}\min_{f\in\mathcal{F}}||f - \mathcal{T}^1 g||_\mu.$$

This term is referred to as the inherent Bellman error of the space $\mathcal{F}$ and it is related to how well the Bellman images of functions in $\mathcal{F}$ can be approximated by $\mathcal{F}$ itself.

Estimation error. The second relevant term is the term $||\alpha_*^k||$ appearing in the estimation error. We recall that $f_{\alpha_*^k} = \Pi\mathcal{T}_1\widetilde{Q}^{k-1}$ is the projection on $\mathcal{F}$ of the Bellman image of the function returned at the previous iteration. The function $\widetilde{Q}^{k-1}$ is truncated in the interval $[-V_{\max}, V_{\max}]$ and its Bellman image $\mathcal{T}_1\widetilde{Q}^{k-1}$ is still bounded in the same interval. Since the projection operator Π is a non-expansion, we finally have that $||f_{\alpha_*^k}||_\infty \leq V_{\max}$. Using Assumption 2, for any $f_\alpha \in \mathcal{F}$, it is possible to relate the norm of the function to the norm of the vector α as

$$||f_\alpha||_\mu^2 = ||\phi^\top\alpha||_\mu^2 = \alpha^\top G\alpha \geq \omega\alpha^\top\alpha = \omega||\alpha||^2.$$

By combining the bound on α with the bound on f_α, we obtain that

$$\max_k ||\alpha_*^k|| \leq \max_k \frac{||f_{\alpha_*^k}||_\mu}{\sqrt{\omega}} \leq \frac{V_{\max}}{\sqrt{\omega}}$$

Transfer error. Since $\widetilde{Q}^k$ is the truncation of a function $f_{\hat{\alpha}^k} = \widehat{Q}^k$ belonging to $\mathcal{F}$, the transfer error is

$$\max_k ||(\mathcal{T}_1 - \overline{\mathcal{T}}_\lambda)\widetilde{Q}^k||_\mu = \sup_\alpha \|(\mathcal{T}_1 - \overline{\mathcal{T}}_\lambda)T(f_\alpha)\|_\mu.$$

Finally, the statement of the theorem follows by taking a union bound over K iterations.

□

D Analysis of BAT

D.1 Proof of Theorem 3

Lemma 2. *Let $\{(X_s, A_s, R_s^1, \dots, R_s^M)\}_{s=1}^S$ be a training set where $(X_s, A_s) \overset{iid}{\sim} \mu$ and $R_s^m = \mathcal{R}^m(X_s, A_s)$ and for each state-action pair and for each task m, T next states $Y_{s,t}^m \sim \mathcal{P}^m(\cdot|X_s, A_s)$ with $t = 1, \dots, T$ are available. For any fixed bounded function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A}; V_{\max})$, the $\widehat{\lambda}$ returned by minimizing Eq. 3 is such that*

$$\mathcal{E}_{\widehat{\lambda}}(Q) - \mathcal{E}_{\lambda_*}(Q) \leq 2V_{\max}\sqrt{\frac{(M-2)\log 4S/\delta}{S}} + 16V_{\max}^2\frac{\log 4SM/\delta}{T} \tag{20}$$

with probability $1-\delta$.

Proof. [Lemma 2]

The sketch of the proof is as follows. For any state-action pair X_s, A_s, we define

$$\widehat{\mathcal{E}}_\lambda(X_s, A_s) = R_s^1 - \sum_{m=2}^M \lambda_m R_s^m + \gamma\frac{1}{T}\sum_{t=1}^T \Big(\max_{a'}\widetilde{Q}^{k-1}(Y_{s,t}^1, a') - \sum_{m=2}^M \lambda_m \max_{a'}\widetilde{Q}^{k-1}(Y_{s,t}^m, a')\Big),$$

and

$$\mathcal{E}_\lambda(X_s, A_s) = (\mathcal{T}_1\widetilde{Q}^{k-1})(X_s, A_s) - \sum_{m=2}^M \lambda_m(\mathcal{T}^m\widetilde{Q}^{k-1})(X_s, A_s).$$

As a result, $\mathcal{E}_\lambda = \mathbb{E}_\mu\big[\mathcal{E}_\lambda(x,a)^2\big]$ and $\widehat{\mathcal{E}}_\lambda = \frac{1}{S}\sum_{s=1}^S \widehat{\mathcal{E}}_\lambda(X_s, A_s)^2$. By Pollard's inequality on the (M-2)-dimensional simplex Λ, we have for any $\lambda \in \Lambda$

$$|E_\mu\big[\mathcal{E}_\lambda(x,a)^2\big] - \frac{1}{S}\sum_{s=1}^S \mathcal{E}_\lambda(X_s, A_s)^2| \leq V_{\max}\sqrt{\frac{(M-2)\log S/\delta'}{S}} \tag{21}$$

with probability $1-\delta'$. Using Chernoff-Hoeffding inequality we now bound the distance between the true Bellman operators in $\mathcal{E}_\lambda(X_s, A_s)$ and their estimates in $\widehat{\mathcal{E}}_\lambda(X_s, A_s)$. By triangle inequality and the previous definitions, we obtain the following series of inequlities

$$\begin{aligned}
|\frac{1}{S}\sum_{s=1}^S &\mathcal{E}_\lambda(X_s, A_s)^2 - \frac{1}{S}\sum_{s=1}^S \widehat{\mathcal{E}}_\lambda(X_s, A_s)^2| \leq |\frac{1}{S}\sum_{s=1}^S (\mathcal{E}_\lambda(X_s, A_s) - \widehat{\mathcal{E}}_\lambda(X_s, A_s))^2| \\
&\leq \max_s \Big(\mathcal{E}_\lambda(X_s, A_s) - \frac{1}{S}\sum_{s=1}^S \widehat{\mathcal{E}}_\lambda(X_s, A_s)\Big)^2 \\
&\leq 2\max_s \max_m \Big((\mathcal{T}^m\widetilde{Q}^{k-1})(X_s, A_x) - R_s^m - \gamma\frac{1}{T}\sum_{t=1}^T \max_{a'} Q(Y_{s,t}^m, a')\Big)^2 \\
&\leq 2\left(2V_{\max}\sqrt{\frac{\log SM/\delta'}{T}}\right)^2
\end{aligned} \tag{22}$$

By using Eqs 21 and 22, we have for any $\lambda \in \Lambda$

$$|\mathcal{E}_\lambda - \widehat{\mathcal{E}}_\lambda| \leq V_{\max}\sqrt{\frac{(M-2)\log S/\delta'}{S}} + 8V_{\max}^2\frac{\log SM/\delta'}{T},$$

with probability $1-2\delta'$. Finally, we can prove the following sequence of inequalities

$$\begin{aligned}
\mathcal{E}_{\widehat{\lambda}} - \mathcal{E}_{\lambda_*} &= \mathcal{E}_{\widehat{\lambda}} - \widehat{\mathcal{E}}_{\widehat{\lambda}} + \widehat{\mathcal{E}}_{\widehat{\lambda}} - \widehat{\mathcal{E}}_{\lambda_*} + \widehat{\mathcal{E}}_{\lambda_*} - \mathcal{E}_{\lambda_*} \\
&\leq 2\sup_{\lambda\in\Lambda}|\mathcal{E}_\lambda - \widehat{\mathcal{E}}_\lambda| \leq 2V_{\max}\sqrt{\frac{(M-2)\log S/\delta'}{S}} + 16V_{\max}^2\frac{\log SM/\delta'}{T},
\end{aligned}$$

with probability $1-4\delta'$. By setting $\delta = 4\delta'$ the statement follows. □

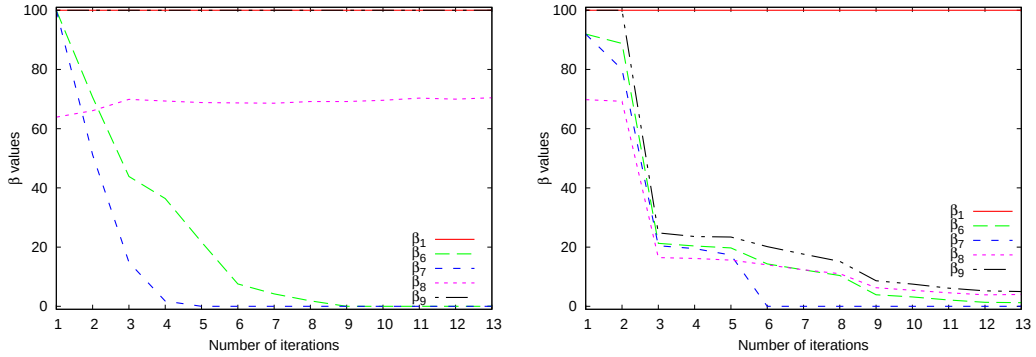

Figure 8: Percentage of samples for each task as selected by BTT as a function of FQI iterations when 5000 samples are available for each source task. *Left:* 100 target samples available. *Right:* 10000 target samples available.

E Additional Experimental Analysis

In this section, we provide additional experimental results related to the BTT algorithm.

E.1 Analysis of parameters β

In order to have a better understanding on how BTT trades off between the need for samples and the risk of introducing a large transfer error, in Figure 8 we show the values of parameters β (which represent the percentage of samples transferred from each task) as optimized by the BTT algorithm at each FQI iteration. The tasks considered are the target task $\mathcal{M}_1$ and the source tasks $\mathcal{M}_6, \mathcal{M}_7, \mathcal{M}_8, \mathcal{M}_9$, each one with 5000 samples available. Figure 8 compares the values of β in two scenarios: when the available target samples are 100 (left pane) and 10000 (right pane). Obviously, BTT always exploits all the target samples ($\beta_1 = 1$). When few target samples are available, BTT transfers high percentages of samples from the source tasks. In particular, it transfers all the samples available from task $\mathcal{M}_9$ in each iteration, and also the percentage of samples taken from task $\mathcal{M}_8$ is almost constant (about 0.7). The percentage of samples transferred from tasks $\mathcal{M}_6$ and $\mathcal{M}_7$ starts from 100% and decreases (with different rates) through iterations reaching zero after iteration 10. This behavior can be explained by the attempt to include as many samples as possible at the earlier iterations when it is still possible to find combinations of sources with a small transfer error. As the iterations continue, no suitable combination of sources is possible and the algorithm is forced to reduce the number of samples from the more different source tasks. On the other hand, when the number of target samples is large enough, we notice that the percentage of samples transferred from all the source tasks drop down after the first FQI iterations. In fact, in this case, BTT exploits a lot of source samples to produce a more accurate approximations only when a very small error is introduced. On the other hand, as the iterations progress, the samples from the source tasks (even when optimally combined) provide a poor approximation of the Q-functions and, as a result, BTT, given the large number of target samples (10000), prefers to reduce the number of samples transferred from the source tasks.

In Figure 9 we show the proportions λ induced by the weights β computed by BTT. When only 100 target samples are available, BTT tries to compensate the lack of target samples by transferring a large amount of samples from a suitable combination of source tasks, while, when many target samples are available, it considers source samples only when they can guarantee a good approximation of the target Q-functions, otherwise the proportions are changed in favor of the target samples.

Finally, in Figure 10, we consider the total number of samples used to train FQI at each iteration under the two scenarios. As expected, at the first iterations, due to the similarity between source tasks and target task, the number of samples provided to FQI by BTT is very large and then it decreases through iterations. It is interesting to notice that the total number of samples selected in the two scenarios are quite similar (in particular starting from the third iteration), which is an effect of the tradeoff realized by the BTT algorithm.

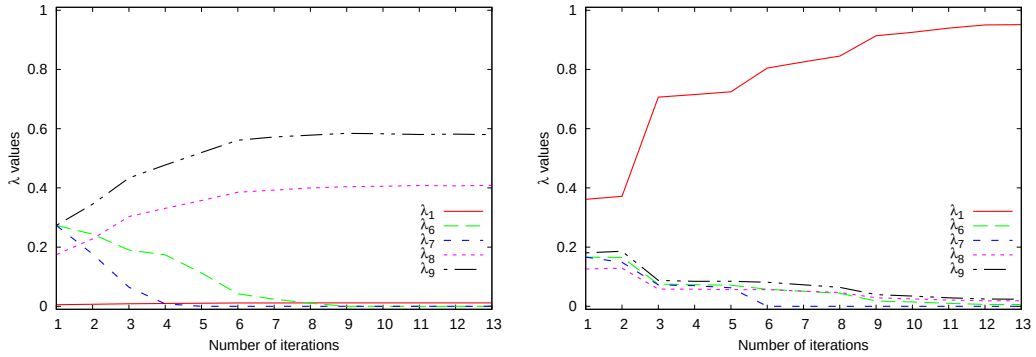

Figure 9: Proportions in the combination of task samples induced by BTT β parameters as a function of FQI iterations. *Left:* 100 target samples available. *Right:* 10000 target samples available.

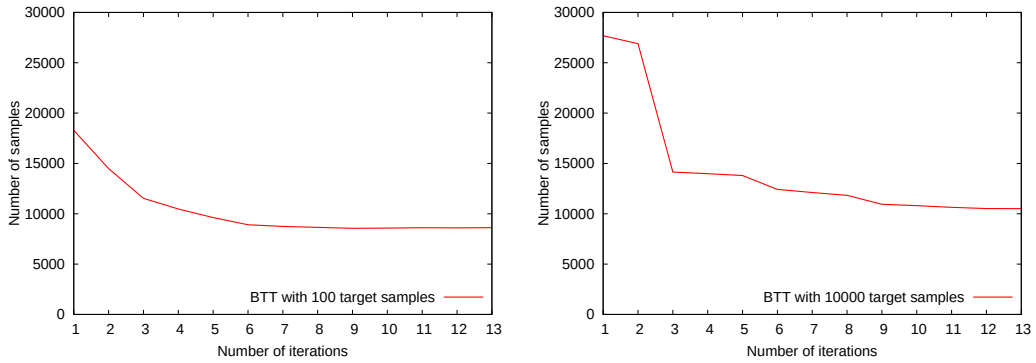

Figure 10: Number of samples actually used at each iteration (after the transfer) by FQI. *Left:* 100 target samples available. *Right:* 10000 target samples available.

E.2 Analysis of parameter τ

The tradeoff realized by the BTT algorithm is tuned by the parameter τ multiplying the estimation error. In Figure 11 we analyze the effect of τ on the learning performances. Different values of the tradeoff parameter have been tried ($\tau = 0.25, 0.50, 0.75, 1.0$) when both 5000 samples (left pane) and 10000 samples (right pane) are available for each source task. As we can notice, BTT is quite robust w.r.t. the choice of the tradeoff parameter. The main differences appear when a small number of target samples is available. In this case, low values of τ make BTT more concerned about the transfer error and, as a result, it tends to avoid transferring source samples, even if target samples are not enough. On the other hand, with high values of τ, BTT is pushed to use more source samples, and this may negatively affect the performance when several target tasks are available and no combination of source tasks provides a good target approximation.

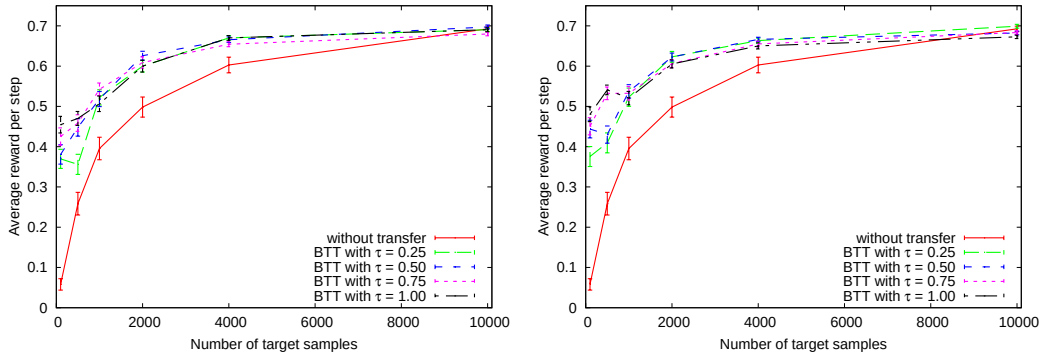

Figure 11: Comparison of the performance of FQI using BTT algorithm with different values of the tradeoff parameter τ. *Left:* 5000 samples available for each source. *Right:* 10000 samples available for each source.