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) [10] 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 [12, 6] 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 [11], 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 [5]. 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 [1] and [8] 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 [2]. 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 [3]. The first algorithm (*AST* in Sec. 3) simply transfers all the source samples. We also design two adaptive methods (*BAT* and *BTT* in Sec. 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 Sec. 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 proofs and additional experiments are available in [7].

2 Preliminaries

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

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

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 $[-V_{\max}, V_{\max}]$.

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

```
Input: Linear space $\mathcal{F} = \text{span}\{\varphi_i, 1 \le i \le d\}$, initial function $\widetilde{Q}^0 \in \mathcal{F}$
for $k = 1, 2, \dots$ 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 $\Phi = [\phi(X_1, A_1)^\top; \dots; \phi(X_L, A_L)^\top]$
    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.

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 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

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

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 [9], 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., $\overline{\mathcal{R}}_\lambda = \sum_{m=1}^{M} \lambda_m \mathcal{R}_m$, $\overline{\mathcal{P}}_\lambda = \sum_{m=1}^{M} \lambda_m \mathcal{P}_m$). We also denote by $\overline{\mathcal{T}}_\lambda$ its optimal Bellman operator. 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 $0 < \delta \le 1$, $\widehat{Q}^k$ (Eq. 1) satisfies*

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

with probability $1 - \delta$ (w.r.t. samples), where $||\varphi_i||_\infty \le C$ 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 $M = 1$ [7]. 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 does not depend on 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 $\widehat{Q}^k_s$ 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 $\widehat{Q}^k_s$ can be written as (up to constants and logarithmic factors)

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

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

with $N_1 = \lambda_1 L$ (on average). Both bounds have the same approximation error. The main difference is that $\widehat{Q}^k_s$ 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. 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 Sec. 4 we 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 Sec. 5 we consider a different setting where the number of samples in each source is limited.

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. [1]

Assumption 1. *[9] 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 $C_{\mu,\nu} = (1-\gamma^2)^2\sum_p p\gamma^{p-1}c(p) < \infty$.*

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., $\omega > 0$).*

Theorem 2. *Let Assumptions 1 and 2 hold and the setting be as in Thm. 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 (see [7] for further discussion). 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

[1] We refer to [9] 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.

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 [8] 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 $R_{s,m} = \mathcal{R}_m(X_s, A_s)$. 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

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

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 $(M$-$2)$-dimensional simplex, and then runs an iteration of AST with samples generated according to the proportions $\widehat{\lambda}^k$. We denote by $\lambda_*^k = \arg\min_{\lambda \in \Lambda} \mathcal{E}_\lambda(\widetilde{Q}^{k-1})$ the best combination at iteration k.

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 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. When compared to single-task learning, BAT has a better performance whenever the best combination of source tasks has a small transfer error and the additional auxiliary estimation error 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

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 $[-11,-9]\cup[9,11]$

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 $\mathcal{F}$. 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 (in this case Λ).

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 assumption that L samples can always be generated from any source task [2] 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. 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

$$\hat{\beta}^k = \arg\min_{\beta\in[0,1]^M} \Big(\widehat{\mathcal{E}}_\beta(\widetilde{Q}^{k-1}) + \tau\sqrt{\frac{d}{\sum_{m=1}^M \beta_m N_m}}\Big), \tag{3}$$

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 bound in Thm. 1 and we do not provide any theoretical guarantee for it. The main technical difficulty 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 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 as AST and BAT could be derived.

6 Experiments

In this section, we report preliminary experimental results of the transfer algorithms. The main objective is to illustrate the functioning of the algorithms and compare their results with the theoretical findings. We consider a continuous extension of the chain walk problem proposed in [4]. The state is described by a continuous variable x and two actions are available: one that moves toward *left* and the other toward *right*. With probability p each action makes a step of length l, affected by a noise η,

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

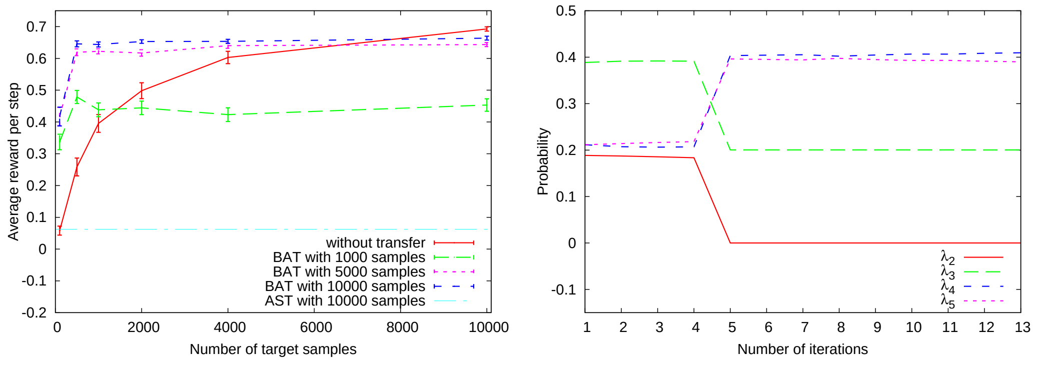

Figure 3: Transfer from $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$, $\mathcal{M}_5$. *Left:* Comparison between single-task learning, AST with $L = 10000$, BAT with $L = 1000, 5000, 10000$. *Right:* Source task probabilities estimated by BAT algorithm as a function of FQI iterations.

in the intended direction, while with probability $1-p$ it moves in the opposite direction. In 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 starting from the state $x_0 = 0$ with actions chosen uniformly at random. All the results are averaged 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. 3 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. 3. At the first iteration, FQI produces an approximation of the reward function. Given the first four source tasks, 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. 4. 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.

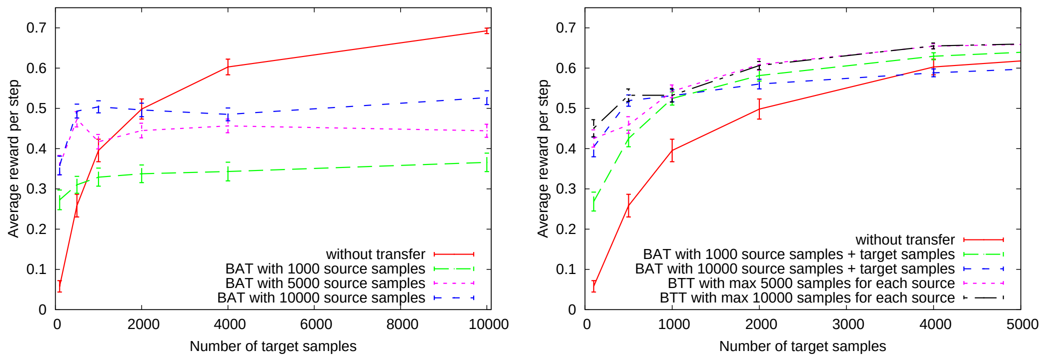

Figure 4: Transfer from $\mathcal{M}_6$, $\mathcal{M}_7$, $\mathcal{M}_8$, $\mathcal{M}_9$. *Left:* Comparison between single-task learning and BAT with $L = 1000, 5000, 10000$. *Right:* Comparison between single-task learning, BAT with $L = 1000, 10000$ in addition to the target samples, and BTT ($\tau = 0.75$) with 5000 and 10000 samples for each source task. To improve readability, the plot is truncated at 5000 target samples.

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. 4 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 Sec. 5). BTT algorithm provides a principled way to address such tradeoff, and, as shown by the right plot in Fig. 4, 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. 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.

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] 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.

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

[3] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

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

[5] 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.

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

[7] Alessandro Lazaric and Marcello Restelli. Transfer from Multiple MDPs. Technical Report 00618037, INRIA, 2011.

[8] 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.

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

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

[11] 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.

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