Reinforcement Learning from Demonstration through Shaping

Tim Brys and **Anna Harutyunyan**
Vrije Universiteit Brussel
{timbrys, aharutyu}@vub.ac.be

Halit Bener Suay and **Sonia Chernova**
Worcester Polytechnic Institute
{benersuay, soniac}@wpi.edu

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

Ann Nowé
Vrije Universiteit Brussel
anowe@vub.ac.be

Abstract

Reinforcement learning describes how a learning agent can achieve optimal behaviour based on interactions with its environment and reward feedback. A limiting factor in reinforcement learning as employed in artificial intelligence is the need for an often prohibitively large number of environment samples before the agent reaches a desirable level of performance. Learning from demonstration is an approach that provides the agent with demonstrations by a supposed expert, from which it should derive suitable behaviour. Yet, one of the challenges of learning from demonstration is that no guarantees can be provided for the quality of the demonstrations, and thus the learned behavior. In this paper, we investigate the intersection of these two approaches, leveraging the theoretical guarantees provided by reinforcement learning, and using expert demonstrations to speed up this learning by biasing exploration through a process called reward shaping. This approach allows us to leverage human input without making an erroneous assumption regarding demonstration optimality. We show experimentally that this approach requires significantly fewer demonstrations, is more robust against suboptimality of demonstrations, and achieves much faster learning than the recently developed HAT algorithm.

1 Introduction

Reinforcement learning (RL) has long been recognized as an approach animals use to learn and adapt to situations, by associating positive (and negative) stimuli with situations and responses [Thorndike, 1911; Skinner, 1938]. As RL was introduced in artificial intelligence [Sutton and Barto, 1998], simple algorithms were developed that operate with little prior knowledge and have strong theoretical guarantees for convergence to optimal behaviour. Yet, exactly because of this lack of prior knowledge, these algorithms often require (too) many experiences for learning, preventing them from being practical in most real world situations. A lot of progress is being made towards mitigating this sample complexity problem by incorporating prior knowledge of various forms into the learning process [Ng *et al.*, 1999; Taylor and Stone, 2009].

Meanwhile, in real world domains (e.g., robotics) previous work focused on using prior knowledge from expert demonstrators to render the learning problem tractable and speed up learning [Argall *et al.*, 2009]. A (human) demonstrator provides examples of how to achieve a task, and the agent uses these demonstrations to generate behaviour that mimicks and generalizes them. Even though this learning from demonstration (LfD) approach has had successes, physical and computational differences between demonstrator and learning agent, and limitations of the demonstrator, typically result in suboptimal demonstrations, compromising the quality of behaviour the agent derives from such demonstrations [Atkeson and Schaal, 1997; Taylor *et al.*, 2011b].

In this work, we look at the intersection of these two ways of agent learning. We want to leverage the theoretical guarantees for optimality and convergence available in RL, where the agent is provided with the ground truth, i.e. a reward signal that defines the task. To overcome the large sample complexity of RL algorithms, we integrate demonstrations, not to directly derive behaviour, but as a bias for the RL process. Treating the demonstrations as a heuristic allows us to relax the often faulty assumption of expert or demonstration optimality.

We first review some preliminaries on reinforcement learning and learning from demonstration, and describe how these two techniques can be elegantly integrated through a process called reward shaping. Experiments demonstrate the practical usefulness of the approach on a pole balancing task and the Mario domain [Karakovskiy and Togelius, 2012].

2 Preliminaries

2.1 Reinforcement Learning

Reinforcement learning (RL) [Sutton and Barto, 1998] is a paradigm that describes how an agent can improve its behaviour based on reward and punishment received from interactions with its environment. Maximizing the rewards accumulated over time equals solving the task by definition. We define an RL problem as a Markov Decision Process (MDP) $\langle S, A, T, \gamma, R \rangle$. S defines all environment states that can be observed, and A defines the actions an agent can take to affect this environment. T describes the dynamics of the system,

defining the probability of the environment transitioning from state s to state s', given that action a was taken: $T(s'|s,a)$. R is the reward function that yields a numerical reward for every state transition, defining the task. Finally, γ is called the discounting factor, and defines how important long-term rewards are, i.e. how short- or far-sighted the agent is.

The goal of an RL agent is to find behaviour, i.e. a policy $\pi : S \rightarrow A$ that maximizes the expected, discounted accumulation of rewards. Many approaches do not directly search the policy space, but instead estimate the quality of actions in each state, and derive a policy from these estimates. The quality of states and actions is defined by the Q-function $Q : S \times A \rightarrow \mathbb{R}$. Temporal difference algorithms, such as Q-learning [Watkins, 1989], approximate the true Q-function by iteratively updating their estimates:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\delta$$

where α is the learning rate, and δ is the temporal-difference error, the difference between the previous estimate and the target being tracked:

$$\delta = R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)$$

Q-learning is guaranteed to converge to the true Q-values, and thus an optimal policy, given certain assumptions such as a decreasing a learning rate [Tsitsiklis, 1994].

2.2 Reward shaping

One of the reasons that most value-based RL algorithms are slow, is that they do not employ prior knowledge to kickstart learning. Therefore, the best an RL algorithm can initially do is to explore state-action pairs uniformly at random. Only after enough state transitions and their associated rewards have been observed can the agent start to exploit this knowledge by biasing its action selection towards what its estimates indicate are good actions.

Reward shaping, derived from behavioural psychology [Skinner, 1938], is a popular way of including prior knowledge into the learning process in order to alleviate this problem. It provides a learning agent with extra intermediate rewards, much like a dog trainer would reward a dog for completing part of a task. This extra reward can enrich a sparse base reward signal (for example a signal that only gives a non-zero feedback when the agent reaches the goal), providing the agent with useful gradient information. This shaping reward F is added to the environment's reward R to create a new composite reward signal that the agent uses for learning:

$$R_F(s,a,s') = R(s,a,s') + F(s,a,s')$$

Of course, since the reward function defines the task, modifying the reward function may modify the total order over policies, and make the agent converge to suboptimal policies (with respect to the environment's reward alone).

If we define a potential function $\Phi : S \rightarrow \mathbb{R}$ over the state space, and take F as the difference between the new and old states' potential, Ng *et al.* [1999] proved that the total order over policies remains unchanged, and convergence guarantees are preserved:

$$F(s,a,s') = \gamma\Phi(s') - \Phi(s)$$

Prior knowledge can be incorporated by defining the potential function Φ accordingly. The shaping's effect is that the agent's exploration will no longer be uniformly random initially, but instead, it will be biased towards states with high potential. For example, using height as a potential function in Mountain Car [Singh and Sutton, 1996] biases the agent to selecting actions that will increase height. Since the goal location in Mountain Car lies on top of a hill, shaping using this heuristic helps an agent solve that task faster.

The definition of F and Φ was extended by [Wiewiora *et al.*, 2003] to include actions, allowing for the incorporation of behavioural knowledge, reflecting the quality of actions as well as states:

$$F(s,a,s',a') = \gamma\Phi(s',a') - \Phi(s,a)$$

Wiewiora et al. extended Ng's proof and show that this formulation also preserves the total order over policies.

2.3 Learning from Demonstration

As in RL, an agent operating in a learning from demonstration (LfD) setting looks for a policy[1] that allows it to execute a task [Argall *et al.*, 2009]. While LfD methods vary greatly, there is typically no ground truth, no reward signal that allows the agent to evaluate its behaviour. Instead, the agent is provided with a number of teacher demonstrations of the task, from which it needs to derive a policy that reproduces and generalizes the demonstrations. These demonstrations typically consist of sequences of state-action pairs $\{(s_0,a_0),\ldots,(s_n,a_n)\}$.

There are a number of factors that limit the quality of the policy derived from demonstrations by LfD techniques. One is that demonstrations often do not cover the whole state space, and that generalizations to these states may be far from correct [van Lent and Laird, 2001; Nicolescu and Mataric, 2003]. Furthermore, the quality of the demonstrations themselves can be suboptimal, limiting the quality of policies derived from them [Atkeson and Schaal, 1997]. These problems are recognized in the community, and one of the proposed directions forward is to use demonstrations to kickstart an RL process [Argall *et al.*, 2009]. We propose our contribution towards that goal in the next section.

3 Shaping Reinforcement Learning using Demonstrations

Simply stated, the setting we are interested in is one where both the ground truth (reward) and demonstrations are available. From here on, we refer to this intersection of RL and LfD as RLfD.[2] We recognize that defining an informative reward signal for a task can be non-trivial in complex domains, but we argue that one can often construct a very sparse signal that only gives non-zero feedback when the goal is reached or when the task can no longer be completed. Of course, such sparse reward signals typically make learning slow because it takes many experiences for the sparse information to propagate throughout the agent's representation of the state-space. We propose to make the demonstrations available to

[1] We consider plans to be policies, and trajectories partial ones.

[2] Not to be confused with Robot Learning from Demonstration

the learning agent for use as a bias for its exploration, helping the agent find and propagate the sparse rewards much faster.

To achieve this bias, we incorporate the demonstrations in the learning process as a potential-based reward shaping function. Therefore, we need to encode the demonstrated state-action pairs as a meaningful potential function. Intuitively, we want the potential $\Phi^D(s,a)$ of a state-action pair (s,a) to be high when action a was demonstrated in a state s^d similar to s, and we want the potential to be low when this action was not demonstrated in the neighbourhood of s. To achieve this, we need a similarity metric for state-action pairs. In this work, we use a non-normalized multi-variate Gaussian to calculate similarity between state-action pairs. More precisely, assuming discrete actions, if two state-action pairs differ in the action, their similarity is zero, otherwise, we calculate:

$$g(s, s^d, \Sigma) = e^{\left(-\frac{1}{2}(s-s^d)^T\Sigma^{-1}(s-s^d)\right)}$$

Similarity is 1 when $s = s^d$, and trails to zero as s and s^d get further apart. The covariance matrix Σ is crucial in defining the sphere of influence of demonstrated state-action pairs, and needs to be tailored for each domain. This could be automated by including metric learning techniques [Taylor *et al.*, 2011a]. In this work, we always normalize the state space (map state variables to $[0,1]$), and use Σ of the form $\Sigma = \sigma I$, i.e. the identity matrix times a constant σ.

To calculate the potential of a given state-action pair, we look through the set of demonstrations, and find the sample with the same action that yields the highest similarity:

$$\Phi^D(s,a) = \max_{(s^d,a)} g(s, s^d, \Sigma)$$

In other words, the potential function is piecewise Gaussian, a landscape with mountain ranges along the demonstrated trajectories, and flat plains where no demonstrations were provided.

This potential function can then be integrated in two ways into the learning process: first, by creating a shaping function F^D and adding it to the base reward:

$$F^D(s,a,s',a') = \gamma\Phi^D(s',a') - \Phi^D(s,a)$$

This rewards the agent for taking actions that were demonstrated in a similar state.

Second, by initializing the Q-function with Φ^D [Wiewiora *et al.*, 2003]:

$$Q_0(s,a) = \Phi^D(s,a)$$

This initialization allows the agent to immediately use the bias in action selection.

Shaping after initialization helps to maintain this bias, as the initial bias can be quickly lost. Note that initialization and potential-based shaping have only been shown to be equivalent *given the same experiences* [Wiewiora *et al.*, 2003]. In the case of complex function approximators, where initialization is non-trivial or impossible, the Q-value reuse technique [Taylor *et al.*, 2007] can be used to achieve the same effect as initialization.

This approach to learning, i.e. relying on the ground truth for learning and using demonstrations as a heuristic bias, has several advantages. Because we use RL on the true reward signal, we leverage the available theoretical guarantees for convergence and optimality. The potential sub-optimality of demonstrations will furthermore not compromise optimality, thanks to the theoretical guarantees of reward shaping. Finally, the problem of high sample complexity in RL without prior knowledge is mitigated thanks to the exploratory bias introduced by the demonstrations.

4 Related Work

In essence, RLfD techniques have two ways to bias the RL process based on demonstrations: either by affecting values (Q or R; *a priori* or during learning), or by taking over action selection (or both).

The prior work most closely related to ours is HAT, an algorithm that leverages transfer learning principles to combine demonstrations and RL [Taylor *et al.*, 2011b]. The algorithm derives a policy from the demonstrated samples using a simple classifier, much like some LfD techniques would do. This policy is then 'transferred' to the learning process that learns on the environment's reward. Transfer is achieved by either initializing Q-values, or probabilistically reusing the classifier's policy. We compare our technique with value-based HAT in the experimental section.

Smart and Kaelbling [2002] are also interested in the same RLfD setting. The approach they propose splits learning in two phases. In the first phase, the demonstrator is in control, choosing actions, and the RL agent passively learns from the demonstrations. In the second phase, the RL agent is put in control of the system and continues learning.

A lot of other research considering the intersection of LfD and RL focusses on learning a model of the environment (and possibly a reward function), and using RL on simulated experiences in that model to build a good policy [Abbeel and Ng, 2005; Argall *et al.*, 2009]. Learning a model is non-trivial, and the quality of the policy derived from this model of course depends on the quality of the model itself. We are more interested in work that avoids this model learning phase, and learns in the actual environment or a simulation.

Other authors do not consider demonstrations, but allow a human to interactively provide reward or advice during agent learning [Thomaz and Breazeal, 2006; Knox and Stone, 2010]. In contrast, we use demonstrations to shape the *real* reward function defined by the environment.

5 Experiments

To demonstrate the potential usefulness of the approach proposed in this work, we perform experiments in two domains: Cart Pole and Mario. We compare several approaches, to cover the whole spectrum between RL and LfD:

- RL ($Q(\lambda)$-learning)
- RLfD ($Q(\lambda)$-learning+shaping)
- RLfD ($Q(\lambda)$-learning+HAT)
- LfD (C4.5 decision tree classifier [Quinlan, 1993])

We will use a range of different demonstrations, varying in length and demonstrator (human, RL expert, etc.). For each type, we generated 10 distinct demonstrations, each of which

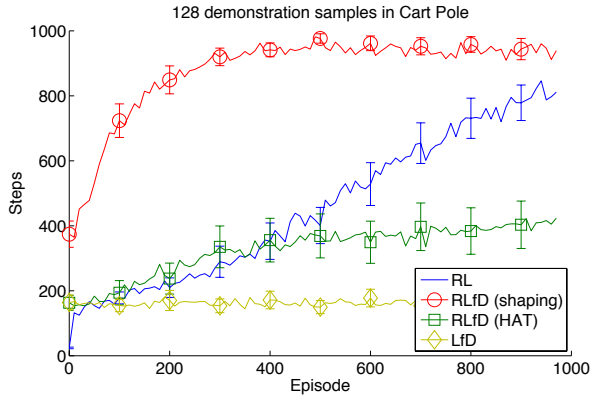


Figure 1: Comparison of RL and RLfD learning provided with a demonstration of 128 steps in Cart Pole (LfD performance is provided for comparison). RLfD (shaping) manages to greatly improve learning compared to RL.

is used in 10 distinct trials. All experiments are therefore averaged over 100 trials for statistical significance. In every figure, error bars indicate the 95% confidence interval. Algorithm parameters were hand selected after a set of preliminary experiments.

5.1 Cart Pole

Cart Pole [Michie and Chambers, 1968] is a task in which the agent controls a cart with a pole on top. The goal is to keep the pole balanced for as long as possible. The agent can move the cart either left or right within a given interval in a single dimension. The state space consists of the position of the cart, its velocity, the angle of the pole and its angular velocity $(x, \dot{x}, \theta, \dot{\theta})$. To learn the task with RL, we use $Q(\lambda)$-learning with tile-coding function approximation. Parameters are $\alpha = \frac{0.25}{16}$, $\gamma = 1$, $\epsilon = 0.05$, $\lambda = 0.25$, with 16 $10 \times 10 \times 10 \times 10$ tilings. For the shaping component, $\sigma = 0.2$, and we both initialize and shape with the potential function. With HAT, parameters are $B = 1$, meaning that the Q-values of the actions suggested by the LfD policy are initialized to 1 (and the others to 0), and $C = 0$, i.e. the LfD policy is not exclusively executed during the initial phases of learning.

Figure 1 shows learning curves for RL and RLfD (shaping and HAT), provided with an expert demonstration of 128 steps.[3] Given this demonstration, RLfD (shaping) manages to learn much faster compared to basic RL that only learns from the sparse reward signal. RLfD (HAT) performs worse, as the demonstration is not sufficient to derive a meaningful classifier that summarizes the demonstrator's policy, as shown by the LfD performance in the graph.

To evaluate the effect of demonstration length on speed of learning and policy quality, we experimented with demonstrations ranging from just a single step, to demonstrations of 4096 steps. Figure 2 (a) shows the average performance of each of the on-line learning techniques (RL and RLfD) during learning (indicative of the speed of learning). RLfD (shaping) needs many fewer demonstrated samples to quickly learn good policies than RLfD (HAT), which requires more samples in order to derive a meaningful classifier. The final quality of the policies generated by all techniques (after 1000 learning episodes for RL and RLfD) is shown in the (b) part of that figure. Here we see the advantage of combining RL and LfD. LfD on its own manages to build near-optimal policies only with the longest of demonstrations. RLfD (HAT), biasing learning with the classifiers built by our LfD implementation, improves final policy quality over basic LfD through on-line experiences. Finally, RLfD (shaping) obtains near-optimal policies with demonstrations that are orders of magnitude smaller, and does so much faster than basic RL. This indicates better sample efficiency of RLfD (shaping) in certain domains. In Cart Pole, an optimal policy can be sufficiently well demonstrated with just two samples, and RLfD (shaping) manages to take advantage of this.

The fact that for the longest demonstrations, RLfD (shaping) is outperformed by RLfD (HAT) and LfD can be explained by C4.5 smoothing out the inconsistencies present in the demonstrations (5% random moves), while RLfD (shaping) uses each sample individually, and comparatively suffers from the accumulation of inconsistent data in longer demonstrations. This is the difference between using the demonstrations globally or locally. The same effect was observed by Sammut *et al.* [2002] when applying a C4.5 decision tree to flight trajectories supplied by human pilots. The many inconsistent and correcting actions taken by the pilots were "cleaned up" by the decision tree, resulting in much smoother control than that exhibited by the human demonstrators.

5.2 Mario

The Mario benchmark problem [Karakovskiy and Togelius, 2012] is a public reimplementation of the original Super Mario Bros® game. It involves an agent (Mario) that navigates a 2D level, collecting points for finishing the level, finding coins, getting hurt (negative points), etc. The goal is to collect as many points as possible. An episode is ended when time runs out, Mario dies, or when he finishes the level. The state space in Mario is fairly complex, as Mario observes the locations and types of enemies on the screen, he observes all information pertaining to himself, e.g. what mode he is in (small, big, fire), and furthermore he is surrounded by a grid-like receptive field in which each cell indicates what type of object is in it (a brick, a coin, a mushroom, an enemy, etc.). Mario can take 12 distinct composite actions, each being a combination of one action from these three sets: {left, right, no direction}, {jump, do not jump} and {run, do not run}.

In the experiments involving RL, we use tabular $Q(\lambda)$-learning. The state-space used by the agent consists of four boolean variables (telling whether Mario is able to jump, on the ground, which direction he is facing, and whether he is able to shoot fireballs), and two variables indicating the relative position (x and y) of the closest enemy within a 21×21 grid surrounding Mario. Learning parameters are $\alpha = 0.01$, $\gamma = 0.9$, $\epsilon = 0.05$, and $\lambda = 0.5$. For RLfD (shaping), $\sigma = 0.5$, and we initialize the Q-function with the potential function. For HAT, $B = 1$ and $C = 0$.

[3] The demonstrator is an RL agent that learned a near-optimal policy with the same setup as the RL baseline agent. The demonstrations include 5% random moves.

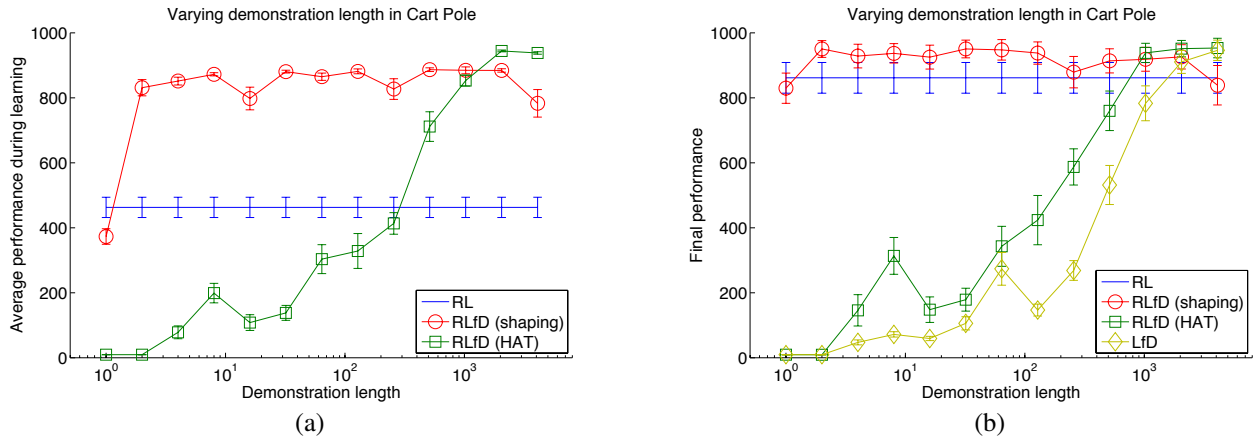


Figure 2: The effect of demonstration length on RLfD and LfD in Cart Pole (RL performance provided for comparison). Figure (a) shows average performance over 1000 learning episodes, an indication of the speed of learning (excluding LfD), (b) shows the final performance (after 1000 learning episodes) of the policies proposed by each technique. RLfD (shaping) needs many fewer demonstration samples to quickly learn good policies in this domain. Note the x log-scale.

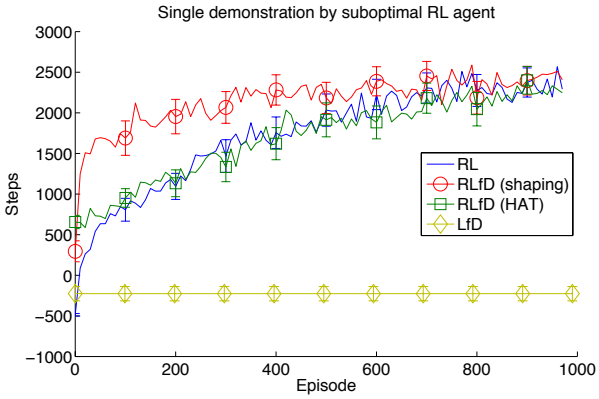


Figure 3: Comparison of RL and RLfD learning provided with a single run demonstration by a suboptimal RL agent in Mario (LfD performance is provided for comparison). RLfD (shaping) generates faster learning compared to RL and RLfD (HAT).

Figure 3 shows learning curves for RL and RLfD (shaping and HAT), provided with a demonstration by a suboptimal RL agent. The demonstration consists of a single run in Mario, either ending in death, time-out or completion of the level. This demonstrator achieves below 1000 points on average. RLfD (shaping) again yields much faster learning than RL alone, while RLfD (HAT) yields a small initial improvement, but otherwise shows statistically the same behaviour as RL. Notice the LfD line, showing that the quality of a classifier derived from this suboptimal demonstration is low.

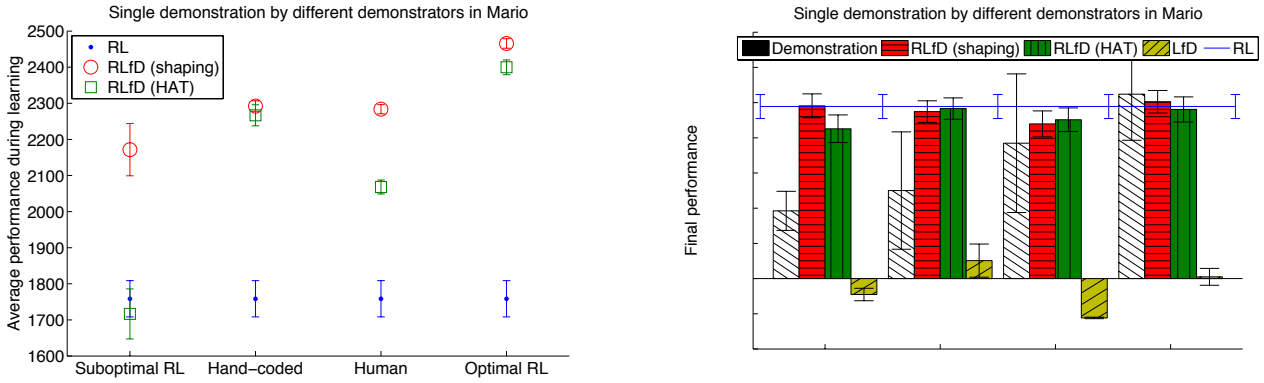
Besides potential suboptimality of demonstrations, another problem for LfD can be the differences in embodiment and capabilities of the demonstrator and the student. In the following experiment, we look at four types of demonstrators, ordered by the quality of their demonstrations (the average demonstrator performance is indicated between brackets):

- A suboptimal RL agent. This demonstrator agent has the same internal representation as the student agent, but its policy and demonstrations are suboptimal. (962 points)
- A handcoded agent. This agent randomly takes one of three actions in any state: (1) go right, (2) go right and jump, and (3) go right, jump and run. This policy is easy to summarize, but also suboptimal. (1250.4 points)
- A human demonstrator. Humans are very different from the student agent in internal representation and capabilities. Humans are less responsive, and often take time to asses the situation in this domain, resulting in very inconsistent demonstrations. (1923.6 points)
- A (near-) optimal RL agent. This demonstrator has the same internal representation as the student agent, and provides near-optimal demonstrations. (2619.4 points)

Figures 4 (a) and (b) show respectively the average performance during learning and the final performance (after 1000 episodes of learning). Despite the suboptimality of demonstrations and differences in internals, RLfD (shaping) shows a higher sample-efficiency while converging on equally good policies as other techniques. Deriving a policy by using the C4.5 LfD technique alone is insufficient in this case. The consistency (or summarizability) of a demonstration clearly affects the LfD technique, with the simple hand-coded policy demonstrations yielding the best LfD performance, while the very inconsistent human demonstrations yield the worst LfD performance. Yet, using these low quality policies to bootstrap learning in RLfD (HAT) yields improved performance compared to just RL, while RLfD (shaping) remains more sample-efficient.

6 Discussion

The results presented in this paper confirm the intuition that combining RL and LfD can be a powerful way to build good policies. The naive LfD technique employed generated low



(a) (b)

Figure 4: The effect the type of demonstrator has on RLfD and LfD in Mario (RL performance provided for comparison). Figure (a) shows average performance over 1000 learning episodes, an indication of the speed of learning (excluding LfD), (b) shows the final performance (after 1000 learning episodes) of the policies proposed by each technique. RLfD (shaping) always outperforms or matches the performance of other techniques.

quality policies, in general, due to the small size or inconsistency of the demonstrations. But, integrated with RL, these demonstrations proved to be a useful bias for finding better policies quickly. RLfD (HAT) depends on the ability of the LfD technique to summarize a good policy, and therefore also suffers (to a lesser extent) from short, inconsistent demonstrations, while it performs well with more extensive demonstrations. In these experiments, RLfD (shaping) appeared to be better at exploiting the limited and inconsistent information provided by the demonstrations, by using each demonstrated sample locally. When enough demonstrations are available on the other hand, summarizing them in a global way can be more beneficial.

Before we conclude, we need to discuss some properties of our proposed approach. We formulated RLfD (shaping) in a mathematically succinct and readable way that is not necessarily the most computationally efficient way of calculating the potential function. Instead of calculating the Gaussian for each demonstrated state-action pair, one can store the demonstrations in a k-d tree and quickly find the closest demonstration. Alternatively, samples that are too similar can be discarded from the demonstrations to reduce computation times.

Furthermore, there are some design decisions that impact the effect of the technique. Σ and the scaling of the potential function (always 1 in our experiments) are free parameters that impact how much effect the technique has on learning. Also, the potential function can be used to shape, initialize or both. In Cart Pole, both initializing and shaping proved to be the best alternative, while in Mario, this proved less effective than only initializing (results not included).[4] These factors make that the technique is not yet an off-the-shelf solution, but rather one requiring some tuning. Further research should help mitigate these problems: ensemble techniques could prove to be a promising step towards avoiding parameter tuning, by allowing learning in parallel with multiple differently configured shapings and initializations [Brys *et al.*, 2014; Harutyunyan *et al.*, 2015].

[4]Note that our Mario setup is non-Markovian (partial representation of the state-space), which can cause abnormal behaviour.

7 Conclusions

Learning from demonstration (LfD) and reinforcement learning (RL) provide two approaches for an agent to learn how to achieve a task. The former relies on expert demonstrations, the latter relies on a reward signal. Both approaches have their advantages and limitations. RL often provides theoretical guarantees for convergence to the optimal policy, but suffers from a high sample complexity due to sparse, uninformative rewards. LfD techniques on the other hand can learn a policy off-line, but often lack guarantees for the quality of the policy, which can be compromised due to suboptimal demonstrations, or differences in demonstrator and student embodiment and capabilities. We have investigated the intersection between the two approaches (RLfD), a setting where both the ground truth (reward), as well as demonstrations are available. Using demonstrations as a bias for learning allows us to relax the assumption of demonstration optimality, while speeding up learning and preserving convergence guarantees. We have shown that our new approach based on reward shaping can be more sample-efficient and more robust against suboptimal and inconsistent demonstrations in two simulated domains.

Future work includes implementing and comparing these algorithms on a robotic manipulation setup, as robotics has a critical need for more efficient learning algorithms. Additionally, work combining multiple shapings through ensemble techniques could be used to combine demonstrations from multiple teachers.

Acknowledgments

Tim Brys is supported by the FWO, and Anna Harutyunyan by the IWT-SBO project MIRAD (grant nr. 120057). This work was furthermore supported in part by NSF IIS-1149917 and ONR N00014-14-1-0795.

References

[Abbeel and Ng, 2005] Pieter Abbeel and Andrew Y Ng. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 1–8. ACM, 2005.

[Argall *et al.*, 2009] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[Atkeson and Schaal, 1997] Christopher G. Atkeson and Stefan Schaal. Robot learning from demonstration. In *ICML*, volume 97, pages 12–20, 1997.

[Brys *et al.*, 2014] Tim Brys, Ann Nowé, Daniel Kudenko, and Matthew E. Taylor. Combining multiple correlated reward and shaping signals by measuring confidence. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1687–1693, 2014.

[Harutyunyan *et al.*, 2015] Anna Harutyunyan, Tim Brys, Peter Vrancx, and Ann Nowé. Multi-scale reward shaping via an off-policy ensemble. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015.

[Karakovskiy and Togelius, 2012] Sergey Karakovskiy and Julian Togelius. The Mario AI benchmark and competitions. *Computational Intelligence and AI in Games, IEEE Transactions on*, 4(1):55–67, 2012.

[Knox and Stone, 2010] W. Bradley Knox and Peter Stone. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 5–12, 2010.

[Michie and Chambers, 1968] Donald Michie and R.A. Chambers. Boxes: An experiment in adaptive control. *Machine intelligence*, 2(2):137–152, 1968.

[Ng *et al.*, 1999] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, volume 99, pages 278–287, 1999.

[Nicolescu and Mataric, 2003] Monica N Nicolescu and Maja J Mataric. Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 241–248. ACM, 2003.

[Quinlan, 1993] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[Sammut *et al.*, 2002] Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Learning to fly. *Imitation in animals and artifacts*, page 171, 2002.

[Singh and Sutton, 1996] Satinder P. Singh and Richard S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158, 1996.

[Skinner, 1938] Burrhus Frederic Skinner. The behavior of organisms: An experimental analysis. 1938.

[Smart and Kaelbling, 2002] William D. Smart and Leslie Pack Kaelbling. Effective reinforcement learning for mobile robots. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 4, pages 3404–3410. IEEE, 2002.

[Sutton and Barto, 1998] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

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

[Taylor *et al.*, 2007] Matthew E. Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(1):2125–2167, 2007.

[Taylor *et al.*, 2011a] Matthew E. Taylor, Brian Kulis, and Fei Sha. Metric learning for reinforcement learning agents. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2011. 22

[Taylor *et al.*, 2011b] Matthew E. Taylor, Halit Bener Suay, and Sonia Chernova. Integrating reinforcement learning with human demonstrations of varying ability. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 617–624, 2011.

[Thomaz and Breazeal, 2006] Andrea Lockerd Thomaz and Cynthia Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *AAAI*, volume 6, pages 1000–1005, 2006.

[Thorndike, 1911] Edward Lee Thorndike. *Animal intelligence: Experimental studies*. Macmillan, 1911.

[Tsitsiklis, 1994] John N. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16(3):185–202, 1994.

[van Lent and Laird, 2001] Michael van Lent and John E Laird. Learning procedural knowledge through observation. In *Proceedings of the 1st international conference on Knowledge capture*, pages 179–186. ACM, 2001.

[Watkins, 1989] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge, 1989.

[Wiewiora *et al.*, 2003] Eric Wiewiora, Garrison Cottrell, and Charles Elkan. Principled methods for advising reinforcement learning agents. In *ICML*, pages 792–799, 2003.