Integrating Reinforcement Learning with Human Demonstrations of Varying Ability

Matthew E. Taylor, Halit Bener Suay, and Sonia Chernova
Lafayette College, taylorm@lafayette.edu
Worcester Polytechnic Institute, {benersuay, soniac}@wpi.edu

ABSTRACT

This work introduces *Human-Agent Transfer* (HAT), an algorithm that combines transfer learning, learning from demonstration and reinforcement learning to achieve rapid learning and high performance in complex domains. Using experiments in a simulated robot soccer domain, we show that human demonstrations transferred into a baseline policy for an agent and refined using reinforcement learning significantly improve both learning time and policy performance. Our evaluation compares three algorithmic approaches to incorporating demonstration rule summaries into transfer learning, and studies the impact of demonstration quality and quantity, as well as the effect of combining demonstrations from multiple teachers. Our results show that all three transfer methods lead to statistically significant improvement in performance over learning without demonstration. The best performance was achieved by combining the best demonstrations from two teachers.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Miscellaneous

General Terms

Algorithms, Performance

Keywords

Reinforcement Learning, Learning from Demonstration, Human/Agent Interaction, Transfer Learning

1. INTRODUCTION

Agent technologies for virtual agents and physical robots are rapidly expanding in industrial and research fields, enabling greater automation, increased levels of efficiency, and new applications. However, existing systems are designed to provide niche solutions to very specific problems and each system may require significant effort to develop. The ability to acquire new behaviors through learning is fundamentally important for the development of general-purpose agent platforms that can be used for a variety of tasks.

Existing approaches to agent learning generally fall into two categories: independent learning through exploration and learning from labeled training data. Agents often learn independently from exploration via *Reinforcement learning* (RL) [25]. While such techniques have had great success in offline learning and software applications, the large amount of data and high exploration times they require make them intractable for most real-world domains.

On the other end of the spectrum are *learning from demonstration* (LfD) algorithms [1]. These approaches leverage the vast experience and task knowledge of a person to enable fast learning, which is critical in real-world applications. However, human teachers provide particularly noisy and suboptimal data due to differences in embodiment (e.g., degrees of freedom, action speed) and limitations of human ability. As a result, final policy performance achieved by these methods is limited by the quality of the dataset and the performance of the teacher.

This paper proposes a novel approach: use RL *transfer learning* methods [28] to combine LfD and RL and achieve both fast learning and high performance in complex domains. In transfer learning, knowledge from a *source task* is used in a *target task* to speed up learning. Equivalently, knowledge from a source agent is used to speed up learning in a target agent. For instance, knowledge has been successfully transferred between agents that balance different length poles [19], that solve a series of mazes [5, 34], or that play different soccer tasks [29, 31, 32]. The key insight of transfer learning is that previous knowledge can be effectively reused, even if the source task and target task are not identical. This results in substantially improved learning times because the agent no longer relies on an uninformed (arbitrary) prior.

In this work, we show that we can effectively transfer knowledge from a human to an agent, even when they have different perceptions of state. Our method, *Human-Agent Transfer* (HAT): 1) allows a human teacher to perform a series of demonstrations in a task, 2) uses an existing transfer learning algorithm, *Rule Transfer* [27], to learn rule-based summaries of the demonstration, and 3) integrates the rule summaries into RL, biasing learning while also allowing improvement over the transferred policy.

We perform empirical evaluation of HAT in a simulated robot soccer domain. We compare three algorithms for incorporating rule summaries into reinforcement learning, and compare learning performance for multiple demonstration source, quantity, and quality conditions. Our findings show statistically significant improvement in performance for all variants of HAT over learning with no prior. Additionally, we find that exposure even to suboptimal demonstration training data results in significant improvements over random exploration, and combining demonstrations from multiple teachers leads to the best performance.

2. BACKGROUND

This section provides background on the three key techniques discussed in this paper: reinforcement learning, learning from demonstrations, and transfer learning.

Cite as: Title, Author(s), *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. XXX-XXX.

2.1 Reinforcement Learning

Reinforcement learning is a common approach to agent learning from experience. We define reinforcement learning using the standard notation of Markov decision processes (MDPs) [16]. At every time step the agent observes its state $s \in S$ as a vector of k state variables such that $s = \langle x_1, x_2, \ldots, x_k \rangle$. The agent selects an action from the set of available actions A at every time step. An MDP's reward function $R : S \times A \mapsto \mathbb{R}$ and (stochastic) transition function $T : S \times A \mapsto S$ fully describe the system's dynamics. The agent will attempt to maximize the long-term reward determined by the (initially unknown) reward and transition functions.

A learner chooses which action to take in a state via a policy, $\pi : S \mapsto A$. Policy π is modified by the learner over time to improve performance, which is defined as the expected total reward. Instead of learning π directly, many RL algorithms instead approximate the action-value function, $Q : S \times A \mapsto \mathbb{R}$, which maps state-action pairs to the expected real-valued return. In this paper, agents learn using Sarsa [17, 20], a well known but relatively simple temporal difference RL algorithm, which learns to estimate $Q(s, a)$. While some RL algorithms are more sample efficient than Sarsa, this paper will focus on Sarsa for the sake of clarity.

Although RL approaches have enjoyed multiple past successes (e.g., TDGammon [30], inverted Helicopter control [12], and agent locomotion [18]), they frequently take substantial amounts of data to learn a reasonable control policy. In many domains, collecting such data may be slow, expensive, or infeasible, motivating the need for ways of making RL algorithms more sample-efficient.

2.2 Learning from Demonstration

Learning from demonstration research explores techniques for learning a policy from examples, or demonstrations, provided by a human teacher. LfD can be seen as a subset of Supervised Learning, in that the agent is presented with labeled training data and learns an approximation to the function which produced the data.

Similar to reinforcement learning, learning from demonstration can be defined in terms of the agent's observed state $s \in S$ and executable actions $a \in A$. Demonstrations are recorded as temporal sequences of t state-action pairs $\{(s_0, a_0), ..., (s_t, a_t)\}$, and these sequences typically only cover a small subset of all possible states in a domain. The agent's goal is to generalize from the demonstrations and learn a policy $\pi : S \mapsto A$ covering all states that imitates the demonstrated behavior.

Many different algorithms for using demonstration data to learn π have been proposed. Approaches vary by how demonstrations are performed (e.g., teleoperation, teacher following, kinesthetic teaching, external observation), the type of policy learning method used (e.g., regression, classification, planning), and assumptions about degree of demonstration noise and teacher interactivity [1]. Across these differences, LfD techniques possess a number of key strengths. Most significantly, demonstration leverages the vast task knowledge of the human teacher to significantly speed up learning either by eliminating exploration entirely [6, 13], or by focusing learning on the most relevant areas of the state space [22]. Demonstration also provides an intuitive programming interface for humans, opening possibilities for policy development to non-agents-experts.

However, LfD algorithms are inherently limited by the quality of the information provided by the human teacher. Algorithms typically assume the dataset to contain high quality demonstrations performed by an expert. In reality, teacher demonstrations may be ambiguous, unsuccessful, or suboptimal in certain areas of the state space. A naïvely learned policy will likely perform poorly in such areas [2]. To enable the agent to improve beyond the performance of the teacher, learning from demonstration must be combined with learning from experience.

Most similar to our approach is the work of Smart and Kaelbling, which shows that human demonstration can be used to bootstrap reinforcement learning in domains with sparse rewards by initializing the action-value function using the observed states, actions and rewards [22]. In contrast to this approach, our work uses demonstration data to learn generalized rules, which are then used to bias the reinforcement learning process.

2.3 Transfer Learning

The insight behind *transfer learning* (TL) is that generalization may occur not only within tasks, but also *across tasks*, allowing an agent to begin learning with an informative prior instead of relying on random exploration.

Transfer learning methods for reinforcement learning can transfer a variety of information between agents. However, many transfer methods restrict what type of learning algorithm is used by both agents (for instance, some methods require temporal difference learning [29] or a particular function approximator [32] to be used in both agents). However, when transferring from a human, it is impossible to copy a human's "value function" — both because the human would likely be incapable of providing a complete and consistent value function, and because the human would quickly grow wary of evaluating a large number of state-action pairs.

This paper uses *Rule Transfer* [27], a particularly appropriate transfer method that is agnostic to the knowledge representation of the source learner. The ability to transfer knowledge between agents that have different state representations and/or actions is a critical ability when considering transfer of knowledge between a human and an agent. The following steps summarize Rule Transfer:

1a: Learn a policy ($\pi : S \mapsto A$) in the source task. Any type of reinforcement learning algorithm may be used.

1b: Generate samples from the learned policy After training has finished, or during the final training episodes, the agent records some number of interactions with the environment in the form of (S, A) pairs while following the learned policy.

2: Learn a decision list ($D_s : S \mapsto A$) that summarizes the source policy. After the data is collected, a propositional rule learner is used to summarize the collected data to approximate the learned policy by mapping states to actions.[1] This decision list is used as a type of inter-lingua, allowing the following step to be independent of the type of policy learned (step 1a).

3: Use D_t to bootstrap learning of an improved policy in the target task. For instance, previous work [27] provided three ways of leveraging this knowledge; two of these methods are discussed later in Sections 3.1 and 3.2.

2.4 Additional Related Work

This section briefly summarizes three additional lines of related work.

Within psychology, *behavioral shaping* [21] is a training procedure that uses reinforcement to condition the desired behavior in a human or animal. During training, the reward signal is initially

[1] Additionally, if the agents in the source and target task use different state representations or have different available actions, the decision list can be translated via inter-task mappings [27, 29] (as step 2b). For the current paper, this translation is not necessary, as the source and target agents operate in the same task.

used to reinforce any tendency towards the correct behavior, but is gradually changed to reward successively more difficult elements of the task. Shaping methods with human-controlled rewards have been successfully demonstrated in a variety of software agent applications [3, 7]. An alternate form of shaping is to change the task over time, or construct a task sequence for an agent to train on [26, 36]. In contrast to shaping, LfD allows a human to demonstrate complete behaviors, which may contain much more information than a sequence of rewards or suggested tasks.

Most similar to our approach is the recent work by Knox and Stone [9] which combines shaping with reinforcement learning. Their TAMER [8] system learns to predict and maximize a reward that is interactively provided by a human. The learned human reward is combined in various ways with Sarsa(λ), providing significant improvements. The primary difference between HAT and this method is that we focus on leveraging human demonstration, rather than estimating and integrating a human reinforcement signal.

The idea of transfer between a human and an agent is somewhat similar to *implicit imitation* [15], in that one agent teaches another how to act in a task, but HAT does not require the agents to have the same (or very similar) representations.

Allowing for such shifts in representation gives additional flexibility to an agent designer; past experience may be transferred rather than discarded if a new representation is desired. Representation transfer is similar in spirit to HAT in that both the teacher and the learner function in the same task, but very different techniques are used since the human's "value function" cannot be directly examined.

High-level advice and suggestions have also been used to bias agent learning. Such advice can provide a powerful learning tool that speeds up learning by biasing the behavior of an agent and reducing the policy search space. However, existing methods typically require either a significant user sophistication (e.g., the human must use a specific programming language to provide advice [11]) or significant effort is needed to design a human interface (e.g., the learning agent must have natural language processing abilities [10]). Allowing a teacher to demonstrate behaviors is preferable in domains where demonstrating a policy is a more natural interaction than providing such high-level advice.

3. METHODOLOGY

In this section we present HAT, our approach to combining LfD and RL. HAT consists of three steps, motivated by those used in Rule Transfer:

Demonstration The agent performs the task under the teleoperated control by a human teacher, or by executing an existing suboptimal controller. During execution, the agent records all state-action transitions. Multiple task executions may be performed.

Policy Summarization HAT uses the state-action transition data recorded during the Demonstration phase to derive rules summarizing the policy. These rules are used to bootstrap autonomous learning.

Independent Learning The agent learns independently in the task via reinforcement learning, using the policy summary to bias its learning. In this step, the agent must balance exploiting the transferred rules with attempting to learn a policy that outperforms the transferred rules.

In contrast to transfer learning, HAT assumes that either 1) the demonstrations are executed on the same agent, in the same task, as will be learned in the Independent Learning phase, or that 2) any differences between the agent or task in the demonstration phase are small enough that they can be ignored in the independent learning phase. Instead of transferring between different tasks, HAT focuses on transferring between different agents with different internal representations. For instance, it is not possible to directly use a human's "value function" inside an agent because 1) the human's knowledge is not directly accessible and 2) the human has a different state abstraction than the agent.

We next present three different ways that HAT can use a decision list to improve independent learning.

3.1 Value Bonus

The intuition behind the *Value Bonus* method [27] is similar to that of shaping in that the summarized policy is used to add a reward bonus to certain human-favored actions. When the agent reaches a state and calculates $Q(s, a)$, the Q-value of the action suggested by the summarized policy is given a constant bonus (B). For the first C episodes, the learner is forced to execute the action suggested by the rule set. This is effectively changing the initialization of the Q-value function, or, equivalently [33], providing a shaping reward to the state-action pairs that are selected by the rules.

We use $B = 10$ and $C = 100$ to be consistent with past work [27]; the Q-value for the action chosen by the summarized policy will be given a bonus of +10 and agents must execute the action chosen by the summarized policy for the first 100 episodes.

3.2 Extra Action

The *Extra Action* method [27] augments the agent so that it can select a *pseudo-action*. When the agent selected this pseudo-action, it executed the action suggested by the decision list. The agent may either execute the action suggested by the transferred rules, or it can execute one of the "base" MDP actions. Through exploration, the RL agent can decide when it should 1) follow the transferred rules by executing the pseudo-action or 2) execute a base MDP action (e.g., the transferred rules are sub-optimal). Were the agent to always execute the pseudo-action, the agent would never learn but would simply mimic the demonstrated policy.

As with the Value Bonus algorithm, the agent initially executes the action suggested by the decision list, allowing it to estimate the value of the decision list policy. We again set this period to be 100 episodes ($C = 100$).

3.3 Probabilistic Policy Reuse

The third method used is *Probabilistic Policy Reuse*, based on the π-reuse Exploration Strategy [4, 5]. In Probabilistic Policy Reuse, the agent will reuse a policy with probability ψ, explore with probability ϵ, and exploit the current value function with probability $1 - \psi - \epsilon$. By decaying ψ over time, the agent can initially leverage the decision list, but then learn to improve on it if possible. Note that Probabilistic Policy Reuse is similar to the recent TAMER+RL method #7 [9], where the agent tries to execute the action suggested by the learned human shaping reward, rather than follow a transferred policy.

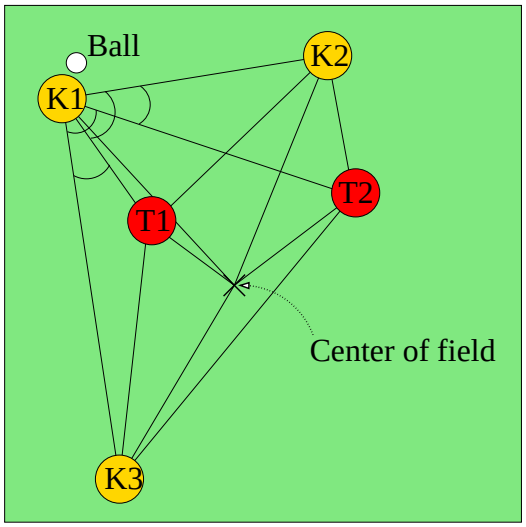

Figure 1: This diagram shows the distances and angles used to construct the 13 state variables used for learning with 3 keepers and 2 takers. Relevant objects are the 3 keepers (K) and the two takers (T), both ordered by distance from the ball, and the center of the field.

Figure 2: This figure shows a screenshot of the visualizer used for the human to demonstrate a policy in 3 vs. 2 Keepaway. The human controls the keeper with the ball (shown as a hollow white circle) by telling the agent when, and to whom, to pass. When no input is received, the keeper with the ball executes the `Hold` action, attempting to maintain possession of the ball.

4. EXPERIMENTAL VALIDATION

This section first discusses Keepaway [24], a simulated robot soccer domain and then explains the experimental methodology used to evaluate HAT.

4.1 Keepaway

Keepaway is a domain with a continuous state space and significant amounts of noise in the agent's actions and sensors. One team, the *keepers*, attempts to maintain possession of the ball within a 20m $\times$ 20m region while another team, the *takers*, attempts to steal the ball or force it out of bounds. The simulator places the players at their initial positions at the start of each episode and ends an episode when the ball leaves the play region or is taken away from the keepers.

The keeper with the ball has the option to either pass the ball to one of its two teammates or to hold the ball. In *3 vs. 2 Keepaway* (3 keepers and 2 takers), the state is defined by 13 hand-selected state variables (see Figure 1) as defined in [24]. The reward to the learning algorithm is the number of time steps the ball remains in play after an action is taken. The keepers learn in a constrained policy space: they have the freedom to decide which action to take only when in possession of the ball. Keepers not in possession of the ball are required to execute the `Receive` macro-action in which the player who can reach the ball the fastest goes to the ball and the remaining players follow a handcoded strategy to try to get open for a pass.

For policy learning, the Keepaway problem is mapped onto the discrete-time, episodic RL framework. As a way of incorporating domain knowledge, the learners choose not from the simulator's primitive actions but from a set of higher-level macro-actions implemented as part of the player [24]. These macro-actions can last more than one time step and the keepers have opportunities to make decisions only when an on-going macro-action terminates. Keepers can choose to `Hold` (maintain possession), $\texttt{Pass}_1$ (pass to the closest teammate), and $\texttt{Pass}_2$ (pass to the further teammate). Agents then make decisions at discrete time steps (when macro-actions are initiated and terminated).

To learn Keepaway with Sarsa, each keeper is controlled by a separate agent. Many kinds of function approximation have been successfully used to approximate an action-value function in Keepaway, but a Gaussian Radial Basis Function Approximation (RBF) has been one of the most successful [23]. All weights in the RBF function approximator are initially set to zero; every initial state-action value is zero and the action-value function is uniform. Experiments in this paper use the public versions 11.1.0 of the RoboCup Soccer Server [14], and 0.6 of UT-Austin's Keepaway players [23].

4.2 Experimental Setup

In the Demonstration phase of HAT, Keepaway players in the simulator are controlled by the teacher using the keyboard. This allows a human to watch the visualization and instruct the keeper with the ball to execute the `Hold`, $\texttt{Pass}_1$, or $\texttt{Pass}_2$ actions. During demonstration, we record all (s, a) pairs selected by the teacher. It is worth noting that the human has a very different representation of the state than the learning agent. Rather than observing a 13 dimensional state vector like the RL agent, the human uses a visualizer (Figure 2). It is therefore critical that whatever method used to glean information about the human's policy does not require the agent and the human to have identical representations of state.

To be consistent with past work [23], our Sarsa learners use $\alpha = 0.05$, $\epsilon = 0.10$, and RBF function approximation. After conducting initial experiments with five values of ψ, we found that $\psi = 0.999$ was at least as good as other possible settings. In the Policy Summarization Phase, we use a simple propositional rule learner to generate a decision list summarizing the policy (that is, it learns to generalize which action is selected in every state). For these experiments, we use JRip, as implemented in Weka [35].

Finally, when measuring speedup in RL tasks, there are many possible metrics. In this paper, we measure the success of HAT along three related dimensions. The initial performance of an agent in a target task may be improved by transfer. Such a *jumpstart* (relative to the initial performance of an agent learning without the benefit of any prior information), suggests that transferred information is immediately useful to the agent. In Keepaway, the jumpstart

is measured as the average episode reward (corresponding to the average episode length in seconds), averaged over 1,000 episodes without learning. The jumpstart is a particularly important metric when learning is slow and/or expensive.

The *final reward* acquired by the algorithm at the end of the learning process (at 30 simulator hours in this paper) indicates the best performance achieved by the learner. This value is computed by taking the average of the final 1,000 episodes to account for the high degree of noise in the Keepaway domain.

The *total reward* accumulated by an agent (i.e., the area under the learning curve) may also be improved. This metric measures the ability of the agent to continue to learn after transfer, but is heavily dependent on the length of the experiment. In Keepaway, the total reward is the sum of the average episode durations at every integral hour of training:

$$\sum_{t:0\rightarrow n} (\text{average episode reward at training hour } t)$$

where the experiment lasts n hours and each average reward is computed by using a sliding window over the past 1,000 episodes.[2]

5. EMPIRICAL EVALUATION

This section presents results showing that HAT can effectively use human demonstration to bootstrap RL in Keepaway agents.

To begin, we recorded a demonstration from a teacher (*Subject A*) which lasted for 20 episodes (less than 3 minutes). Next, we used JRip to summarize the policy with a decision list. The following rules were learned, where $state_k$ represents the k^{th} state variable, as defined in the keepaway task [23]:

if $(state_{11} \geq 74.84$ *and* $state_3 \leq 5.99$ *and* $state_{11} \leq 76.26)$ $\rightarrow$ *Action* $= 1$

elseif $(state_{11} \geq 53.97$ *and* $state_4 \leq 5.91$ *and* $state_0 \geq 8.45$ *and* $state_8 \leq 7.06)$ $\rightarrow$ *Action* $= 1$

elseif $(state_3 \leq 4.84$ *and* $state_0 \geq 7.33$ *and* $state_{12} \geq 43.66$ *and* $state_8 \leq 5.57)$ $\rightarrow$ *Action* $= 2$

else $\rightarrow$ *Action* $= 0$

While not the focus of this work, we found it interesting that the policy was able to be summarized with only four rules, obtaining over 87% accuracy on when using stratified cross-validation.

Finally, agents are trained in 3 vs. 2 Keepaway without using transfer rules (No Prior), using the Value Bonus, using the Extra Action, or using the Probabilistic Policy Reuse method. All learning algorithms were executed for 30 simulator hours (processor running time of roughly 2.5 hours) to ensure convergence.

Figure 3 compares the performance of the four methods, averaged over 10 independent trials. Using 20 episodes of transferred data from *Subject A* with HAT can improve the jumpstart, the final reward, and the cumulative reward. The horizontal line in the figure shows the average duration of the teacher's demonstration episodes; all four of the RL-based learning methods improve upon and outperform the human teacher. The performance of the different algorithms is measured quantitatively in Table 1, where significance is tested with a Student's t-test.

[2]Recall that the reward in Keepaway is +1 per time step, where a time step is a 10th of a simulator second. Thus, the reward for the first hour of training is always $60 \times 60 \times 10 = 36000$ — a metric for the total reward over time must account for the reward *per episode* and simply summing the total amount of reward accrued is not appropriate.

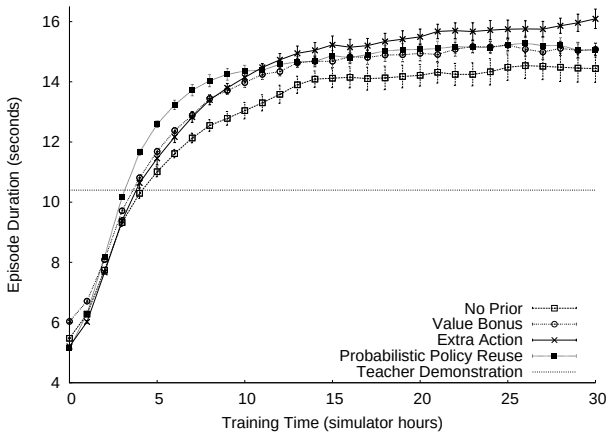

Figure 3: This graph summarizes performance of Sarsa learning in Keepaway using four different algorithms. One demonstration of 20 episodes was used for all three HAT learners. Error bars show the standard error in the performance.

Method	Jumpstart	Final	Total Reward
No Prior	N/A	14.3	380
Value Bonus	**0.57**	15.1	**401**
Extra Action	**-0.29**	**16.0**	**407**
Probabilistic Policy Reuse	**-0.30**	15.2	**411**

Table 1: This table shows the jumpstart, final reward and total reward metrics for Figure 3. Values in bold have statistically significant differences in comparison to the No Prior method ($p < 0.05$).

While the final reward performance of the all four methods is very similar (only Extra Action has a statistically significant[3] improvement over No Prior), the total reward accumulated by all three algorithms is significantly higher than with No Prior learning. This result is an indication that although the same final performance is achieved in the long term because the learning algorithm is able to learn the task in all cases, high performance is achieved *faster* by using a small number of demonstrations. This difference can be best observed by selecting an arbitrary threshold of episode duration and comparing the number of simulation hours each algorithm takes to achieve this performance. In the case of a threshold of 14 seconds, we see that No Prior learning takes 13.5 hours, compared to 10.1, 8.57 and 7.9 hours for Value Bonus, Extra Action and Probabilistic Policy Reuse respectively. These results show that transferring information via HAT from the human results in significant improvements over learning without prior knowledge.

Section 5.1 will explore how performance changes with different types or amounts of demonstration, while Section 5.2 discusses how teacher ability affects learning performance. In all further experiments we use the Probabilistic Policy Reuse method as it was not dominated by either of the other two methods. Additionally, in some trials with other methods we found that the learner could start with a high jumpstart but fail to improve as much as other trials. We posit this is due to becoming stuck in a local minimum. However, because ψ explicitly decays the effect from the rules, this phenomena was never observed when using Probabilistic Policy Reuse.

[3]Throughout this paper, t-tests are used to calculate significance, defined as $p < 0.05$.

5.1 Comparison of Different Teachers

Above, we used a single demonstration data set to evaluate and compare three algorithms for incorporating learned rules into reinforcement learning. In this section, we examine how demonstrations from different people impact learning performance of a single algorithm, Probabilistic Policy Reuse. Specifically, we compare three different teachers:

1. *Subject A*: This teacher has many years of research experience with the Keepaway task. (The same as Figure 3.)
2. *Subject B*: This teacher is new to Keepaway, but practiced for approximately 100 games before recording demonstrations.
3. *Subject C*: This teacher is an expert in LfD, but is new to Keepaway. The teacher practiced 10 games before recording demonstrations.

Each teacher recorded 20 demonstration episodes while trying to play Keepaway to the best of their ability. Figure 4 summarizes the results and compares performance of using these three demonstration sets against learning the Keepaway task without a prior. All reported results are averaged over 10 learning trials. Table 2 presents summary of the results, highlighting statistically significant changes in bold.

Method	Jumpstart	Final	Total Reward
No Prior	N/A	14.3	380
Subject A	**-0.30**	15.2	**411**
Subject B	**3.35**	**15.7**	**423**
Subject C	**0.15**	**16.2**	**424**

Table 2: This table shows the jumpstart, final reward and total reward metrics for Figure 4, where all HAT methods use Probabilistic Policy Reuse with 20 episodes of demonstrated play. Values in bold have statistically significant differences in comparison to the No Prior method.

All three HAT experiments outperformed learning without a bias from demonstration, with statistically significant improvements in total reward. However, as in any game, different Keepaway players have different strategies. While some prefer to keep the ball in one location as long as possible, others pass frequently between keepers. As a result, demonstrations from three different teachers led to different learning curves. Demonstration data from *Subjects A* and *C* resulted in a low jumpstart, while *Subject B*'s demonstration gave the learner a significant jumpstart early in the learning process. The final reward also increased for all three HAT trials, with statistically significant results in the case of *Subjects B* and *C*. These results indicate that HAT is robust to demonstrations from different people with varying degrees of task expertise.

An important factor to consider with any algorithm that learns from human input, is whether combining demonstrations from two or more different teachers helps the agent to learn faster, or whether exposure to possibly conflicting demonstrations from different teachers slows the learning process. In the following evaluation we compared five demonstration types:

1. *Subject A (20)*: Set of the original 20 demonstrations by Subject A: average duration of 10.4 seconds/episode
2. *Subject A (10)*: Set of 10 randomly selected demonstrations by Subject A: average duration 7.5 seconds/episode
3. *Subject C (20)*: Set of the original 20 demonstrations by Subject C : average duration of 11.3 seconds/episode

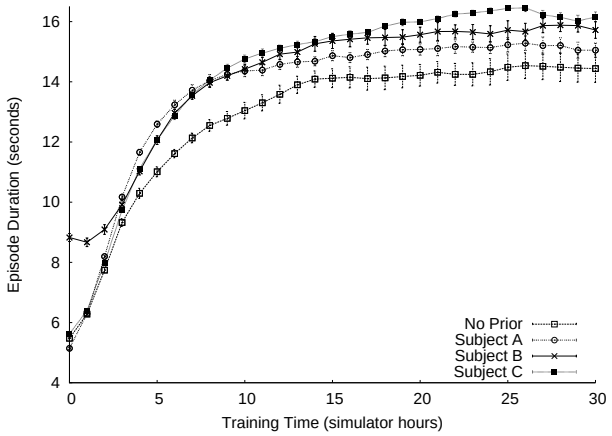

Figure 4: This graph summarizes performance of no prior learning and Probabilistic Policy Reuse learning using demonstrations from three different teachers. Each teacher performed demonstrations for 20 episodes. Error bars show the standard error in performance across 10 trials.

4. *Subjects A + C Best (20)*: The 10 best (longest) demonstration episodes each from Subjects A and C: average duration of 17.2 and 18.0 seconds/episode, respectively
5. *Subjects A + C Worst (20)*: The 10 worst (shortest) demonstration episodes each from Subjects A and C: average duration of 4.6 seconds/episode for both

This analysis provides insight about the impact of combining demonstrations from multiple teachers (conditions 1 and 3 vs. 4 and 5) and the impact of demonstration quantity (condition 1 vs. 2) and quality (condition 4 vs. 5). Figure 5 presents a comparison of the five learning conditions, and Table 3 summarizes the results.

Method	Jumpstart	Final	Total Reward
Subject A (20)	**-0.30**	15.2	**411**
Subject A (10)	**-2.23**	**15.8**	**407**
Subject C (20)	**0.15**	**16.2**	**424**
Subjects A + C Best	**2.15**	**15.7**	**431**
Subjects A + C Worst	**0.37**	**16.1**	**419**

Table 3: This table shows the jumpstart, final reward and total reward metrics for Figure 5, where all HAT methods use Probabilistic Policy Reuse with 20 demonstrated episodes. Values in bold have statistically significant differences in comparison to the No Prior method (not shown).

With respect to learning from multiple teachers, results show that combining data from different subjects leads to performance as good as or better than learning from a single teacher. Condition *Subjects A + C Best* performs better than either *Subject A* or *Subject C* alone, and significantly outperforms all other methods in the group, in large part due to the early lead it has due to its high jumpstart. Condition *Subjects A + C Worst* shows no statistically significant change in performance between it and learning from *Subject A* or *Subject C* alone.[4] This result is significant because it indicates that while quality is important, as shown by the

[4]Note that because we have few subjects, our claims of significance are limited to results from demonstrations with the three subjects tested. Future work will generalize our findings by considering many more subjects.

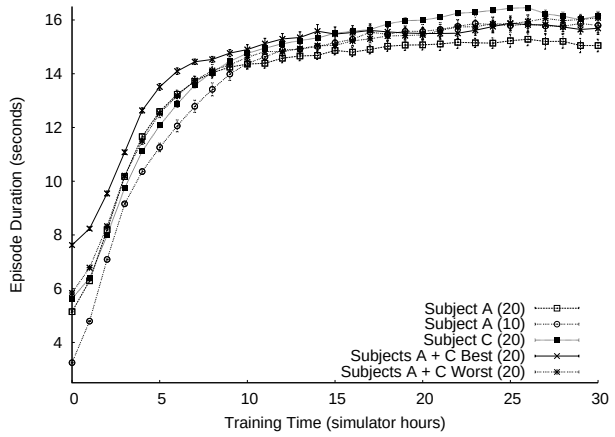

Figure 5: This graph summarizes performance of Probabilistic Policy Reuse learning using five different demonstration sets. Error bars show the standard error in performance across 10 trials.

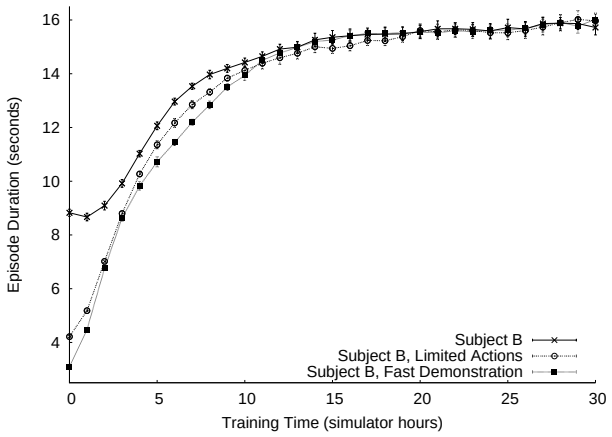

Figure 6: This graph summarizes performance of Probabilistic Policy Reuse learning using three sets of demonstrations from Subject B recorded under different simulator conditions: normal, fast and with limited actions. Each demonstration set consists of 20 episodes. Error bars show the standard error in performance across 10 trials.

difference between *Subjects A + C Best* and *Worst*, any demonstration is beneficial. The fact that the worst demonstrations still lead to performance well above *No Prior* learning is an indication that exposure to any training data is better than random exploration.

In fact, quantity of demonstration may matter more than quality, as shown by the comparison of conditions 1 and 2. Reducing the number of demonstrations by half resulted in a significant decrease in jumpstart. Although performance eventually recovered to achieve a final reward comparable to that of the other methods, achieving that result took longer and there is a statistically significant difference between the total reward of the two conditions.

Most significantly, we highlight that all demonstration-based methods, regardless of data source, quantity or quality, resulted in statistically significant performance improvements over *No Prior* learning. This critical result indicates that HAT learning can benefit from variable degrees of demonstration quality. The algorithm does not require the teacher to be a task expert and easily surpasses the performance of the teacher. In the following section, we further explore the effects of suboptimal demonstrations.

5.2 Impact of Teacher Ability on Learning

In the above experiments, all three teachers demonstrated the task to their best ability. In this evaluation, we alter the simulation environment to make the teacher's demonstrations inherently suboptimal. Specifically, we compare three types of demonstration:

1. *Subject B*: Same as above: average duration 10.5 sec./episode
2. *Subject B Fast*: Simulator speed during training was increased to approximately 5 times faster than real time: average duration 4.3 seconds/episode
3. *Subject B Limited Actions*: The teacher was limited to executing only two actions, `Hold` and $\texttt{Pass}_1$, disallowing passes to the further keeper: average duration 5.2 seconds/episode

The two test conditions are designed to handicap the teacher and reduce the quality of demonstrations, either by affecting reaction time (*Subject B Fast*) or by providing the learning agent with demonstrations of only a subset of the state/action space (*Subject B Limited Actions*). The handicapping effects were successful, reducing the average duration of the teacher's demonstration episodes by more than half.

Figure 6 presents a comparison of the three learning conditions and Table 4 summarizes the results. Importantly, we see again that poor teacher performance does not negatively impact the final performance of the agent. The data further supports our earlier findings that in the long-term, Probabilistic Policy Reuse can learn the task regardless of the initialization method, and there is no statistically significant difference in final reward values between conditions 1 and 2, and conditions 1 and 3. Statistically significant differences are observed, however, in the rate of learning, both with respect to jumpstart and total reward, indicating that suboptimal demonstrations slow the learning process. However, even with the added handicaps, learning from human data shows statistically significant improvements over *No Prior* learning.

Method	Jumpstart	Final	Total Reward
Subject B	**3.35**	**15.7**	**423**
Limited Actions	**-1.26**	**16.0**	**404**
Fast Demonstration	**-2.37**	**16.0**	**401**

Table 4: This table shows the jumpstart, final reward and total reward metrics for Figure 6, where all HAT methods use Probabilistic Policy Reuse. All demonstrations are 20 episodes, recorded by Subject B. Values in bold have statistically significant differences in comparison to the No Prior method (not shown).

6. FUTURE WORK AND CONCLUSION

This paper has introduced HAT, a novel method to combine learning from demonstration with reinforcement learning by leveraging an existing transfer learning algorithm. Using empirical evaluation in the Keepaway domain we showed that given training data from just a few minutes of human demonstration, HAT can increase the learning rate of the task by several simulation hours. We evaluated three different variants which used different methods to bias learning with the human's demonstration. All three methods performed statistically significantly better than learning without demonstration. Probabilistic Policy Reuse consistently performed at least as well as the other methods, likely because it explicitly balances ex-

ploiting the human's demonstration, exploring, and exploiting the learned policy. Additional evaluation using demonstrations from different teachers, combined demonstrations from multiple teachers, and suboptimal demonstrations all showed that HAT is robust to variations in data quality and quantity. The best learning performance was achieved by combining the best demonstrations from two teachers.

One of the key strengths of this approach is its robustness. It is able to take data of good or poor quality and use it well without negative effects. This is very important when learning from humans because it can naturally handle the noisy, suboptimal data that usually occurs with human demonstration. Its ability to deal with poor teachers opens up opportunities for non-expert users.

In order to better understand HAT and possible variants, the following questions should be explored in future work:

- Is it possible to identify the characteristics that make one set of demonstrations lead to better learning performance than another? Can we identify what influences jumpstart (e.g., Subject B's high jumpstart in Figure 4).
- Rather than performing 1-shot transfer, could HAT be extended so that the learning agent and teacher could iterate between learning autonomously and providing additional demonstrations?
- In this work, the human teacher and the learning agent had different representations of state, and in one case had different action sets. Will HAT still be useful if the teacher and agent are performing different tasks? How similar does the demonstrated task need to be to the autonomous learning task for HAT to be effective?

Acknowledgements

The authors would like to thank Shivaram Kalyanakrishnan for sharing his code to allow a human to control the keepers via keyboard input. We also thank the anonymous reviewers and W. Bradley Knox for useful comments and suggestions.

7. REFERENCES

[1] B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469 – 483, 2009.

[2] C. G. Atkeson and S. Schaal. Robot learning from demonstration. In *ICML*, 1997.

[3] B. Blumberg, M. Downie, Y. Ivanov, M. Berlin, M. P. Johnson, and B. Tomlinson. Integrated learning for interactive synthetic characters. *SIGGRAPH*, 2002.

[4] F. Fernández, J. García, and M. Veloso. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems*, 58(7):866–871, 2010.

[5] F. Fernández and M. Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *AAMAS*, 2006.

[6] D. H. Grollman and O. C. Jenkins. Dogged learning for robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007.

[7] F. Kaplan, P.-Y. Oudeyer, E. Kubinyi, and A. Miklosi. Robotic clicker training. *Robotics and Autonomous Systems*, 38(3-4):197 – 206, 2002.

[8] W. B. Knox and P. Stone. Interactively shaping agents via human reinforcement: The TAMER framework. In *KCAP*, 2009.

[9] W. B. Knox and P. Stone. Combining manual feedback with subsequent MDP reward signals for reinforcment learning. In *AAMAS*, 2010.

[10] G. Kuhlmann, P. Stone, R. J. Mooney, and J. W. Shavlik. Guiding a reinforcement learner with natural language advice: Initial results in robocup soccer. In *Proceedings of the AAAI Workshop on Supervisory Control of Learning and Adaptive Systems*, 2004.

[11] R. Maclin and J. W. Shavlik. Creating advice-taking reinforcement learners. *Machine Learning*, 22(1-3):251–281, 1996.

[12] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, 2004.

[13] M. Nicolescu, O. Jenkins, A. Olenderski, and E. Fritzinger. Learning behavior fusion from demonstration. *Interaction Studies*, 9(2):319–352, 2008.

[14] I. Noda, H. Matsubara, K. Hiraki, and I. Frank. Soccer server: A tool for research on multiagent systems. *Applied Artificial Intelligence*, 12:233–250, 1998.

[15] B. Price and C. Boutilier. Accelerating reinforcement learning through implicit imitation. *Journal of Artificial Intelligence Research*, 19:569–629, 2003.

[16] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[17] G. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG-RT 116, Engineering Department, Cambridge University, 1994.

[18] M. Saggar, T. D'Silva, N. Kohl, and P. Stone. Autonomous learning of stable quadruped locomotion. In G. Lakemeyer, E. Sklar, D. Sorenti, and T. Takahashi, editors, *RoboCup-2006: Robot Soccer World Cup X*, volume 4434 of *Lecture Notes in Artificial Intelligence*, pages 98–109. Springer Verlag, Berlin, 2007.

[19] O. G. Selfridge, R. S. Sutton, and A. G. Barto. Training and tracking in robotics. In *IJCAI*, 1985.

[20] S. Singh and R. S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine Learning*, 22:123–158, 1996.

[21] B. F. Skinner. *Science and Human Behavior*. Colliler-Macmillian, 1953.

[22] W. D. Smart and L. P. Kaelbling. Effective reinforcement learning for mobile robots. In *ICRA*, 2002.

[23] P. Stone, G. Kuhlmann, M. E. Taylor, and Y. Liu. Keepaway soccer: From machine learning testbed to benchmark. In I. Noda, A. Jacoff, A. Bredenfeld, and Y. Takahashi, editors, *RoboCup-2005: Robot Soccer World Cup IX*, volume 4020, pages 93–105. 2006.

[24] P. Stone, R. S. Sutton, and G. Kuhlmann. Reinforcement learning for RoboCup-soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

[25] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.

[26] M. E. Taylor. Assisting transfer-enabled machine learning algorithms: Leveraging human knowledge for curriculum design. In *AAAI Symposium: Agents that Learn from Human Teachers*, 2009.

[27] M. E. Taylor and P. Stone. Cross-domain transfer for reinforcement learning. In *ICML*, 2007.

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

[29] M. E. Taylor, P. Stone, and Y. Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(1):2125–2167, 2007.

[30] G. Tesauro. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

[31] L. Torrey, J. W. Shavlik, T. Walker, and R. Maclin. Relational macros for transfer in reinforcement learning. In *ILP*, 2007.

[32] L. Torrey, T. Walker, J. W. Shavlik, and R. Maclin. Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *ECML*, 2005.

[33] E. Wiewiora. Potential-based shaping and Q-value initialization are equivalent. *Journal of Artifical Intelligence Research*, 19(1):205–208, 2003.

[34] A. Wilson, A. Fern, S. Ray, and P. Tadepalli. Multi-task reinforcement learning: a hierarchical Bayesian approach. In *ICML*, 2007.

[35] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[36] P. Zang, A. J. Irani, P. Zhou, C. L. I. Jr., and A. L. Thomaz. Learn via human-provided sequence of tasks using training regimens to teach expanding. In *AAMAS*, 2010.