Inter-Task Action Correlation for Reinforcement Learning Tasks

Matthew E. Taylor and Peter Stone
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188
{mtaylor, pstone}@cs.utexas.edu

Introduction

Reinforcement learning (RL) problems (Sutton & Barto 1998) are characterized by agents making decisions attempting to maximize total reward, which may be time delayed. RL problems contrast with classical planning problems in that agents do not know *a priori* how their actions will affect the world. RL differs from supervised learning because agents are never given training examples with the correct action labeled.

An RL agent operating successfully in an environment could have its available actions disabled, degraded, or even augmented. Instead of learning from scratch with its new action set, we would like the agent to identify which of the actions in the new set, A', are most similar to the actions in the original set, A, and then leverage its existing knowledge.

In the general problem of *transfer learning*, a learner trains in a *source task* and then utilizes its learned knowledge to speed up learning or increase performance in a *target task*. In this paper, we consider pairs of tasks with different action sets. We present *Inter-Task Action Correlation* (I-TAC), a method for identifying actions in the target task which are most similar to actions in the source task. This mapping can then be used to automate knowledge transfer by identifying similarities between pairs of RL tasks. We evaluate our algorithm by fully implementing and testing it in the RoboCup-soccer Keepaway domain.

Value Function Transfer

In our previous transfer learning work (Taylor, Stone, & Liu 2005), we use *temporal difference* (TD) methods (Sutton & Barto 1998), one popular way of solving RL problems. TD methods learn a *value function* that estimates the expected long-term discounted reward for taking a particular action in a state. *Value function transfer* speeds up learning by transferring a learned value function from agents in a source task to agents in a target task. The states and actions in the two tasks may be different and thus we must utilize a functional, $\rho(Q(S, A)) \mapsto Q'(S', A'))$, so that the learned value function is applicable in the target task.

The efficacy of value function transfer is largely determined by ρ which must currently be constructed by a human to take advantage of similarities between pairs of tasks. This work is a step towards automatically constructing ρ and performing value function transfer.

Inter-Task Action Correlation

Consider an RL agent that has learned in a source task with action set A and states S. We would like to use its value function Q to speed up learning in a target task that has actions A' and states S'. Further, suppose that we are given a function γ that maps every state in the target task to some state in the source task: $\gamma(S') \mapsto S$. We would like to discover a function, β, such that $\beta(A') \mapsto A$, where every action in the target task is mapped to an action in the source task. β can then be used to identify similarities between source and target actions, allowing us to use value function transfer.

To learn β, we train a classifier on data from the source task. The classifier is given pairs of states from the source task, s_1 and s_2, and the action a that the agent took to transition between the two states. The classifier trains on many of these tuples to learn to label pairs of states with actions.

We then use the trained classifier to label actions in the target task. Again, pairs of states will be given to the classifier but they must first be transformed from S' into S via γ so that they are valid inputs to the classifier. The classifier reads $\gamma(s'_1)$ and $\gamma(s'_2)$ and predicts the action in A that causes the most similar state transition. Thus, if the agent in the target task took action a' to transition from state s'_1 to state s'_2 and the classifier labeled the action as a, this would be one data point suggesting that action a' is most similar to action a. After collecting sufficient data, we can determine which actions are most similar and then use this mapping, along with the provided γ, to automatically construct a ρ and perform value function transfer. As we will argue in the Results section, learning β takes very little time when compared to the required training time and thus I-TAC has the potential to greatly outperform learning from scratch.

The Keepaway Domain

To evaluate I-TAC, we use the Keepaway domain, a standard RL benchmark task (Stone *et al.* 2006). This multiagent domain has noisy sensors and actuators as well as hidden state so that agents have only a partial view of the world. Details of the Keepaway task are presented elsewhere (Stone, Sutton, & Kuhlmann 2005). All our experiments are run on a code base derived from version 0.6 of the benchmark Keepaway implementation[1] (Stone *et al.* 2006).

An agent's state is composed of distances and angles from itself to its teammates, other *keepers*, and its opponents, the *takers*. In 3 vs. 2 Keepaway, three keepers attempt to maintain control of the ball from two takers. The state of 3 vs. 2 is described by 14 state features: those of the standard Keepaway task (Stone, Sutton, & Kuhlmann 2005) and distance to the ball. There are three possible actions in 3 vs. 2: hold ball, pass to closest teammate, and pass to 2^{nd} closest teammate. 4 vs. 3 has more players and is thus described by 20 state features. The extra keeper also adds a fourth possible action, pass to 3^{rd} closest teammate. γ, the mapping between states in 3 vs. 2 and 4 vs. 3, is defined as the state mapping in ρ from previous work (Taylor, Stone, & Liu 2005).

[1] http://www.cs.utexas.edu/~AustinVilla/sim/Keepaway/

To record s_1, s_2, a tuples, each keeper saves the state of the world when it begins an action. When the action ends, i.e. another keeper touches the ball, the keeper that took the action again records its state and saves the tuple. If the action is not completed because the ball is intercepted or the ball goes out of bounds, the saved state is discarded[2]. The I-TAC process in Keepaway is summarized in Figure 1.

1. In the source task, record multiple s_1, s_2, a tuples. s_1 is the state of the world before action a and s_2 is the state of the world after action a has finished.
2. Train a classifier via rule learning to label pairs of states with actions from the source task.
3. In the target task, record multiple s'_1, s'_2, a' tuples.
4. Use the state mapping γ and the trained classifier to classify each pair of states from the target task as an action from the source task: $\gamma(s'_1), \gamma(s'_2) \mapsto A$.
5. For each action a' in the target task, define $\beta(a') = a$ such that a was the action most often selected by the classifier when presented a pair of states from tuples containing a'.

Figure 1: Summary of the I-TAC methodology in Keepaway

Results

We utilize four pairings of Keepaway tasks to test I-TAC. Table 1 shows that actions in 3 vs. 2 are correctly identified from 1,066 tuples after training on 1,010 tuples of 3 vs. 2. Actions in 4 vs. 3 are also correctly identified from 1,051 tuples after training on 1,010 3 vs. 2 tuples. "Hold ball" and "pass to closest teammate" were correctly mapped between tasks. "Pass to 3^{rd} closest teammate" was mapped to "pass to 2^{nd} closest teammate" in 3 vs. 2 as both actions pass to the furthest teammate. Finally, "pass to 2^{nd} closest teammate" in the target task was mapped to both passes in the source task, and either are reasonable. Results (not shown) also hold for {source, target} pairs of {4 vs. 3, 4 vs. 3} and {4 vs. 3, 3 vs. 2}.

	3 vs. 2 Hold	3 vs. 2 Pass 1	3 vs. 2 Pass 2
3 vs. 2 target action: Hold	**382**	0	4
3 vs. 2 target action: Pass 1	0	**330**	26
3 vs. 2 target action: Pass 2	2	25	**297**
4 vs. 3 target action: Hold	**227**	0	71
4 vs. 3 target action: Pass 1	1	**174**	97
4 vs. 3 target action: Pass 2	0	**133**	**133**
4 vs. 3 target action: Pass 3	1	51	**163**

Table 1: Confusion matrix from two different learning trials that both use a source task (columns) of 3 vs. 2 Keepaway. The target tasks are 3 vs. 2 Keepaway and 4 vs. 3 Keepaway.

In informal experiments we found that increasing the number of training instances in the source task to 25,000 from 1,000 provided only a slight improvement in accuracy. Collecting 1,000 s_1, a, s_2 tuples takes approximately 10.7 minutes of simulation time, which is negligible when typical learning experiments in Keepaway take 10-50 simulated hours. We also tried reducing the number of training instances in the 3 vs. 2 source task to 25 and found that all actions in 4 vs. 3 were still identified correctly, although the number of misclassified instances was increased. Reducing the number of 3 vs. 2 training instances to 12 resulted in 4 vs. 3 actions being misclassified. This suggests I-TAC requires very little data to succeed. We expect that the speedup of learning via value function transfer will therefore still provide significant speedup over learning from scratch, even when some training time is used to learn β.

To our knowledge, this work is the first successful application of machine learning to identify similarities between actions in pairs of RL tasks. We demonstrate how I-TAC leverages γ to learn β, a mapping between actions in two tasks. We next plan to focus on learning γ when given β. A final step will be to attempt to learn both γ and β simultaneously, thus enabling fully autonomous transfer between tasks.

Related Work

I-TAC is inspired by our previous work on value function transfer (Taylor, Stone, & Liu 2005). There are other methods for transfer, such as using advice (Torrey *et al.* 2005), but this and other existing work relies on a human to specify the relationship between tasks. Morales (2003) suggests ways of speeding up learning between two *relational reinforcement learning* (RRL) tasks automatically. However, it is impractical to force some RL problems into the RRL framework.

Acknowledgments

We would like to thank the anonymous reviewers for helpful comments and suggestions. This research was supported in part by DARPA grant HR0011-04-1-0035, NSF CAREER award IIS-0237699, and NSF award EIA-0303609.

References

Morales, E. F. 2003. Scaling up reinforcement learning with a relational representation. In *Proc. of the Workshop on Adaptability in Multi-agent Systems*.

Stone, P.; Kuhlmann, G.; Taylor, M. E.; and Liu, Y. 2006. Keepaway soccer: From machine learning testbed to benchmark. In Noda, I.; Jacoff, A.; Bredenfeld, A.; and Takahashi, Y., eds., *RoboCup-2005: Robot Soccer World Cup IX*. Berlin: Springer Verlag. To appear.

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

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

Taylor, M. E.; Stone, P.; and Liu, Y. 2005. Value functions for RL-based behavior transfer: A comparative study. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*.

Torrey, L.; Walker, T.; Shavlik, J.; and Maclin, R. 2005. Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *Proceedings of the Sixteenth European Conference on Machine Learning*.

[2]We plan to relax this restriction in the future as failed actions may also contain useful information.