Reinforcement Learning via Practice and Critique Advice

Kshitij Judah and **Saikat Roy** and **Alan Fern** and **Thomas G. Dietterich**
School of Electrical Engineering and Computer Science,
Oregon State University,
1148 Kelley Engineering Center,
Corvallis, OR 97331-5501
{judahk,roy,afern,tgd}@eecs.oregonstate.edu

Abstract

We consider the problem of incorporating end-user advice into reinforcement learning (RL). In our setting, the learner alternates between practicing, where learning is based on actual world experience, and end-user critique sessions where advice is gathered. During each critique session the end-user is allowed to analyze a trajectory of the current policy and then label an arbitrary subset of the available actions as good or bad. Our main contribution is an approach for integrating all of the information gathered during practice and critiques in order to effectively optimize a parametric policy. The approach optimizes a loss function that linearly combines losses measured against the world experience and the critique data. We evaluate our approach using a prototype system for teaching tactical battle behavior in a real-time strategy game engine. Results are given for a significant evaluation involving ten end-users showing the promise of this approach and also highlighting challenges involved in inserting end-users into the RL loop.

1 Introduction

Humans often learn through iterations of practice followed by feedback from a teacher. The goal of this paper is to make progress toward computer agents that can learn via a similar process where feedback is provided by end-users with no knowledge of the agents' internal workings. With this motivation we consider the problem of integrating an end-user into a reinforcement learning (RL) process where the RL agent will alternate between two stages: 1) The *critique stage* where the end-user is able to observe a trajectory of the agent's current performance and then critique any of the possible actions as good or bad, and 2) The *practice stage* where the RL agent learns from autonomous practice in the environment while taking the user's critique into account.

We develop a novel approach for combining the critique and practice data for selecting a policy. The approach is based on optimizing an objective function over policy parameters that linearly combines objectives related to each type of data. One of our key contributions is to propose an interpretation of the critique data that leads to a well motivated objective for the critique data. In particular, the critique data is viewed as defining a distribution over a type of supervised learning problem we call *any-label learning*. The objective is then defined in terms of the expected loss of the policy parameters with respect to this distribution.

We present results of a user study in the domain of tactical battles in a real-time strategy game. The overall results show that our approach is able to significantly improve over pure RL and is a promising mechanism for leveraging end-users in the RL loop. Important usability issues also emerge.

2 Related Work

Prior work uses high-level rules as advice for RL. For example, rules in the form of programming-language constructs (Maclin and Shavlik 1996), logical rules derived from a constrained natural language (Kuhlmann et al. 2004), and rules about action utilities and preferences (Maclin et al. 2005b; 2005a). While experimental results show promise, there are no experiments that combine more than 2 or 3 pieces of advice at once and no end-user studies. Also, it is likely that end-users will have difficulty providing such advice due to the need to learn an advice language.

In the *learning by demonstration (LBD)* framework the user provides full demonstrations of a task that the agent can learn from (Billard et al. 2008). Recent work (Coates, Abbeel, and Ng 2008) incorporates model learning to improve on the demonstrations, but does not allow users to provide feedback on the agent's observed behavior. Other work combines LBD and human critiques on behavior (Argall, Browning, and Veloso 2007; 2008) similar to those used in our work, but there is no autonomous practice and no experiments involving end-users.

Another type of approach has been to allow users to provide real-time feedback to the agent during the learning process in the form of an end-user driven reward signal. Knox and Stone (Knox and Stone 2009) use supervised learning to predict the user reward signal and select actions in order to maximize the predicted reward. This approach has shown some promise in user studies, but does not integrate autonomous practice (until recently; see (Knox and Stone 2010)) and is hence limited by the abilities of the user. Thomaz and Breazeal (Thomaz and Breazeal 2008) rather combine the end-user reward signal with the environmental reward and use a Q-learning variant for RL. This method has also shown promise in user studies, but the semantics of the combination of reward signals is unclear in general.

3 Overall Approach

Problem Formulation. We consider reinforcement learning (RL) in the framework of *Markov decision processes (MDPs)*, where the goal is to maximize the expected total reward from an initial state. It is assumed that all policies are guaranteed to reach a terminal state in a finite number of steps. Extending to discounted infinite horizon settings is straightforward. In this work, we incorporate an end-user into the usual RL loop. The user-agent interaction alternates between two stages: 1) Get critique data C from the user regarding the current policy, and 2) Update policy via autonomous practice while factoring in the critique data.

In stage 1, we focus on a simple form of critique data that is reasonable for end-users to provide. The end-user is allowed to observe the execution of the agent's current policy and then scroll back and forth along the trajectory and mark any of the available actions in any state as good or bad. The result of the critiquing stages is a critique data set: $C = \{(s_1, c_1^+, c_1^-), ..., (s_M, c_M^+, c_M^-)\}$, where s_i is a state and c_i^+ and c_i^- are sets of actions that are labeled as good and bad for s_i. No constraints are placed on which states/actions are included in C, and it is expected that for many states on the trajectories one or both of c_i^+ and c_i^- will be empty. Intuitively, the actions in c_i^+ are those that the end-user believes are "equally optimal" while those in c_i^- are those that are considered to be sub-optimal. In many domains, it is possible for there to be multiple optimal/satisfactory actions, which is why we consider sets of actions. One contribution of this paper (Section 4) is to propose a semantics for C that supports a principled approach to policy optimization.

In stage 2 of our protocol, the agent practices by interacting with the MDP as in traditional RL, but while taking the current critique data into account. This generates a set of trajectories $T = \{T_1, ..., T_N\}$, where $T_i = \{(s_i^1, a_i^1, r_i^1), ..., (s_i^{L_i}, a_i^{L_i}, r_i^{L_i})\}$ and s_i^t, a_i^t, and r_i^t are the state, action and reward at time t in T_i. Thus, during the practice session and before the next critique session, the agent has two sets of data T and C available. The fundamental problem is thus to use T and C in order to select an agent policy that maximizes reward in the environment.

Solution Approach. We consider learning the parameters θ of a stochastic policy $\pi_\theta(a|s)$, which gives the probability of selecting action a in state s given θ. We use a standard log-linear policy representation $\pi_\theta(a|s) \propto \exp(f(s,a) \cdot \theta)$, where $f(s,a)$ is a feature vector of state-action pairs and θ weighs features relative to one another. Our problem is now to use the data sets T and C in order to select a value of θ that maximizes the expected total reward of π_θ.

Our high-level approach is to optimize θ with respect to a combined objective function:

$$J_\lambda(\theta, C, T) = \lambda U(\theta, T) - (1 - \lambda) L(\theta, C) \qquad (1)$$

where $\lambda \in [0, 1]$. $U(\theta, T)$ is an estimate of the expected utility of π_θ based on T (details below) and $L(\theta, C)$ is a loss function that measures the disagreement between θ and C (details in Section 4). λ specifies the relative influence of T versus C, where $\lambda = 1$ corresponds to pure RL, and $\lambda = 0$ corresponds to pure supervised learning from C. We will use differentiable functions for U and L, and thus, for a fixed λ, J_λ can be optimized via gradient descent. Supervised actor-critic learning (Rosenstein and Barto 2004) follows a similar approach where online policy updates are driven by a combined reward and supervisory signal. In that work, a schedule for adjusting the value of λ was proposed. However, we have found that in practice, fixing a schedule for λ is difficult since the best choice varies widely across users, domains, and data quality. Thus, instead we use an explicit validation approach, where part of the practice session is used to evaluate different values of λ.

Algorithm 1 Main loop of our approach.

```
 1: loop
 2:     // Begin critique stage.
 3:     $C := C \cup \textsc{GetCritiqueData}(\theta)$
 4:     // Begin practice stage.
 5:     $T := T \cup \textsc{GetPracticeTrajectories}(\theta)$
 6:     for i = 1:n do
 7:         $\lambda = \lambda_i$    //choose a value of $\lambda$
 8:         $\theta_i = \mathrm{argmax}_\theta\, J_\lambda(\theta, C, T)$ // see Equation 1
 9:         $U_i = \textsc{EstimateUtility}(\theta_i)$ // by executing $\pi_{\theta_i}$
10:     end for
11:     $i^* = \mathrm{argmax}_i\, U_i$
12:     $\theta = \theta_{i^*}$
13: end loop
```

Algorithm 1 gives the main loop of our approach, which starts by collecting critique data using the current policy (line 3). Next the practice session begins, and additional trajectories (line 5) are collected via exploitation of the current policy and exploration[1]. Next (lines 6-10) the algorithm considers different values of λ, for each one optimizing J_λ and then estimating the expected utility of the resulting policy by averaging the results of a number of policy executions. In our experiments, we used 11 values of λ equally distributed in $[0, 1]$. Finally, θ is set to the parameter value that yielded the best utility estimate. The ability to easily optimize for different values of λ is one of the advantages of our batch-style learning approach. It is unclear how a more traditional online approach would support such validation.

Trajectory Data Objective. To compute $U(\theta, T)$ it is straightforward to use likelihood weighting in order to estimate the utility of π_θ using off-policy trajectories T, an approach that has been previously considered for policy gradient RL (Peshkin and Shelton 2002). Let $p_\theta(T_j)$ be the probability of π_θ generating trajectory T_j and let θ_j be the parameters of the policy that generated T_j. An unbiased utility estimate is given by: $U(\theta, T) = \frac{1}{W(T)} \sum_{T_j \in T} w(T_j) R_j$, $w(T_j) = \frac{p_\theta(T_j)}{p_{\theta_j}(T_j)}$ where $W(T) = \sum_{T_j \in T} w(T_j)$ and R_j denotes cumulative reward of trajectory T_j. Note that the unknown MDP dynamics cancel out of the weighting factor $w(T_j)$,

[1]In our experiments, each practice stage generated 10 trajectories in the following way: with 0.8 probability the trajectory was generated by the current policy; otherwise, the parameter vector was divided by a randomly drawn constant to increase the temperature of the policy to facilitate exploration.

leaving only factors related to the policies. The gradient of $U(\theta, T)$ has a compact closed form.

4 Critique Data Objective

We now give our formulation for $L(\theta, C)$. We first introduce a supervised learning problem we call *any-label learning (ALL)* and then extend it to incorporate critique data.

Any-Label Learning. We consider a view of the end-user, where for any state s, there is a set of actions $O(s)$ that the user considers to be "equally optimal" in the sense that actions in $O(s)$ are considered to be equally good, but strictly better than actions outside of $O(s)$. The set of actions will vary across users and even for the same user across time. Also, in many cases, these sets will not correspond to the true set of optimal actions in the MDP. However, if the end-user is reasonably competent in the application domain, it is reasonable to assume that, if available, learning from $O(s)$ would likely provide useful guidance.

Given an ideal end-user, we could elicit the full action sets $O(s)$ for states along the observed trajectories, resulting in a supervised training set of states labeled by action sets: $\{(s_1, O(s_1)), ..., (s_M, O(s_M))\}$. A natural learning goal for this data is to find a probabilistic policy, or classifier, that has a high probability of returning an action in $O(s_i)$ when applied to s_i. That is, it is not important which particular action is selected as long as the probability of selecting an action in $O(s)$ is high. This leads to the definition of the ALL supervised learning problem where the input is a set of data of the above form assumed to be drawn i.i.d. and the goal is to learn policy parameters that maximize the *ALL likelihood*: $\prod_{i=1}^{M} \pi_\theta(O(s_i)|s_i)$ where $\pi_\theta(O(s_i)|s_i) = \sum_{a \in O(s_i)} \pi_\theta(a|s_i)$.

ALL is distinct from other supervised learning problems where instances/states are associated with sets of labels/actions. The *multi-label learning problem* (Tsoumakas and Katakis 2007) differs in that the goal is to learn a classifier that outputs all of the labels in sets and no others. This objective is not appropriate for our problem, where we only care about outputting one of the labels and do not care about the relative probabilities assigned to the actions in $O(s_i)$. The *multiple-label learning problem* (Jin and Ghahramani 2003) interprets the label sets as possible labels only one of which is the true label. Thus, in multiple-label learning, the label sets are used to encode label ambiguity, which is in sharp contrast to ALL.

Expected Any-Label Learning. It is unlikely that end-users will actually provide the full action sets $O(s)$. For example, they will often indicate that the action selected by the agent is bad and possibly mark one other action as good, even when there are multiple good alternatives. Thus, in reality we must make due with only the good and bad action sets c^+ and c^- for a state, which only provide partial evidence about $O(s)$. The naive approach of treating c^+ as the true set $O(s)$ can lead to difficulty. For example, when there are actions outside of c^+ that are equally good compared to those in c^+, attempting to learn a policy that strictly prefers the actions in c^+ results in difficult learning problems. Furthermore, we need a principled approach that can handle the situation where either c^+ or c^- is empty.

The key idea behind our loss function is to define a user model that induces a distribution over ALL problems given the critique data. The loss is then related to the *expected ALL likelihood* with respect to this distribution. We define a user model to be a distribution $Q(O(s)|c^+, c^-)$ over sets $O(s) \in 2^A$ (where A is the action set) conditioned on the critique data for s and assume independence among different states. We analyze a simple model involving two noise parameters ϵ^+ and ϵ^- and a bias parameter δ. The parameters ϵ^+ (ϵ^-) represents the rate of noise for good (bad) labels and δ is the bias probability that an unlabeled action is in $O(s)$. The user model Q further assumes that the actions in $O(s)$ are conditionally independent given the critique data, giving the following:

$$Q(O(s)|c^+, c^-) = \prod_{a \in O(s)} q(a \in O(s)|c^+, c^-) \prod_{a \notin O(s)} q(a \notin O(s)|c^+, c^-) \quad (2)$$

$$q(a \in O(s)|c^+, c^-) = \begin{cases} 1 - \epsilon^+, & \text{if } a \in c^+ \\ \epsilon^-, & \text{if } a \in c^- \\ \delta, & \text{if } a \notin c^+ \cup c^- \end{cases}$$

While simple, this model can capture useful expectations about both a domain (e.g. typical number of good actions) and the users (e.g. mislabeling rate).

Given Q, the distribution over ALL training sets $D = \{(s_1, O(s_1)), \ldots, (s_M, O(s_M))\}$ conditioned on critique data $C = \{(s_1, c_1^+, c_1^-), ..., (s_M, c_M^+, c_M^-)\}$ is: $P(D|C) = \prod_{i=1}^{M} Q(O(s_i)|c_i^+, c_i^-)$. Our loss function $L(\theta, C)$ is now defined as the negative log of the expected ALL likelihood,

$$L(\theta, C) = -\sum_{i=1}^{M} \log(E\left[\pi_\theta(O(s_i)|s_i)|c_i^+, c_i^-\right]) \quad (3)$$

where $O(s_i)$ is a random variable distributed as $Q(O(s_i)|c_i^+, c_i^-)$. Computing $L(\theta, C)$ by expanding each expectation is impractical, since there are $|2^A|$ terms in each. However, the loss function has a compact closed form for the above user model.

Theorem 1. *If the user model is given by Equation 2 then*

$$L(\theta, C) = -\sum_{i=1}^{M} \log\left(\delta + (1 - \epsilon^+ - \delta)\pi_\theta(c_i^+) + (\epsilon^- - \delta)\pi_\theta(c_i^-)\right)$$

.

Proof. (Sketch) The expected ALL likelihood inside each logarithm term of $L(\theta, C)$ is given by

$$E\left[\pi_\theta(O(s_i)|s_i)|c_i^+, c_i^-\right] = \sum_{O \in 2^A} Q(O|c_i^+, c_i^-)\pi_\theta(O|s_i)$$

$$= \sum_{O \in 2^A} Q(O|c_i^+, c_i^-) \sum_{a \in O} \pi_\theta(a|s_i)$$

$$= \sum_{a \in A} \pi_\theta(a|s_i) \sum_{O \in 2^A : a \in O} Q(O|c_i^+, c_i^-)$$

$$= \delta + (1 - \epsilon^+ - \delta)\pi_\theta(c_i^+) + (\epsilon^- - \delta)\pi_\theta(c_i^-)$$

The final equality follows from the following relation.

$$\sum_{O \in 2^A : a \in O} Q(O|c_i^+, c_i^-) = q(a \in O(s_i)|c_i^+, c_i^-)$$

□

In our experiments, we assume a noise free ($\epsilon^+ = \epsilon^- = 0$) and unbiased ($\delta = 0.5$) user model which yields: $L(\theta, C) = \sum_{i=1}^{M} \log(1 + \pi_\theta(c_i^+) - \pi_\theta(c_i^-)) + k$ where k is a constant. Optimizing this loss function results in maximizing the probability on c_i^+ while minimizing the probability on c_i^-, which agrees with intuition for this model. The gradient of this objective is straightforward to calculate for our log-linear policy representation.

5 Experimental Setup

RTS Tactical Micro-Management. We evaluate our approach on the problem of micro-management of units in tactical battles in the real-time strategy (RTS) game of Wargus, which runs on the open-source RTS engine Stratagus. We consider controlling a group of 5 friendly close-range military units against a group of 5 enemy units. Such micro-management is very difficult for human players due to the fast pace, and thus it is typical for players to rely on the default game behavior for micro-management, which can often be considerably sub-optimal. In our experiments we asked end-users to help teach the learning agent to improve tactical micro-management against the in-built Wargus AI. This problem is a natural match for learning from critiques and practice rather than other types of human interfaces. For example, given the fast pace and multiple units acting in parallel, it is difficult to provide real-time feedback as in the TAMER system (Knox and Stone 2009). It is also difficult to provide demonstrations due to the difficulty of the control.

We instrumented Stratagus with an API for an RL agent and with an interface that allows an end-user to watch a battle involving the agent and pause at any moment. The user can then scroll back and forth within the episode and mark any possible action of any agent as good or bad. The RL agent makes decisions every 20 game cycles where the available actions for each military unit are to attack any of the units on the map (enemy or friendly) giving a total of 10 actions per unit. To control all friendly units the RL agent uses a simple strategy of looping through each unit at each decision cycle and selecting an action for it using a policy that is shared among the units. The experiences of all the units are pooled together for learning the shared policy. The reward function provides positive and negative rewards for inflicting and receiving damage respectively, with a positive and negative bonus for killing an enemy or being killed. Our log-linear policy used 27 hand-coded features that measure properties such as "the number of friendly units attacking the targeted unit", "the health of the target unit", etc.

Evaluated Systems. We evaluate three learning systems: 1) *Pure RL*, which corresponds to removing the critique stage from Algorithm 1 and setting $\lambda = 1$, 2) *Pure Supervised*, corresponds to removing the practice stage from Algorithm 1 and setting $\lambda = 0$, 3) *Combined System*, which learns from both types of data according to Algorithm 1.

User Study. For the user study we constructed two battle maps, which differed only in the initial placement of the units. Both maps had winning strategies for the friendly team and are of roughly the same difficulty. The user study involved 10 end-users not familiar with the system, 6 with CS backgrounds and 4 without a CS background. For each user, the study consisted of teaching both the pure supervised and the combined systems, each on a different map, for a fixed amount of time. We attempted to maintain a balance between which map and learning system was used first by each user. We allowed 30 minutes for the fully supervised system and 60 minutes for the combined system, due to the additional time required by the combined system for the practice periods. This time difference is intended to give the users the chance of providing roughly the same amount of advice in both cases. Note that in a real training scenario, an end-user would not be confined to the system during the practice stages, as they are in the user study, meaning that the practice sessions will typically have lower cost to the user in reality than in the user study. It also means that we necessarily had to use shorter practice stages than might typically be used in a real application. In our studies, practice session times ranged from 3 to 7 minutes depending on the amount of data and convergence rate of gradient descent.

6 Experimental Results

6.1 Simulated Experiments with User Data

To provide a more controlled setting for evaluating the effectiveness of our approach, we first present simulated experiments using real critique data collected from our user studies. For each of our two maps, we selected the worst and best performing users when training the combined system, for a total of 2 users per map. The total amount of critique data (number of good and bad labels provided) for each user was: User1 - 36, User2 - 91, User3 - 115, User4 - 33. This shows that different users have very different practices in terms of the amount of advice. For each user, we divided their critique data into four equal sized segments, from which we created four data sets for each user containing 25%, 50%, 75%, and 100% of their critique data. We provided the combined system with each of these data sets and allowed it to practice for 100 episodes. All results are averaged over 5 runs and are given in Table 1. For each user and map, the rows correspond to the percentage of critique data used with the number of episodes $|T|$ increasing from 0 (pure supervised) to 100. The top row of each table corresponds to pure RL (0% critique data). Each cell in the table records the value estimate of the best policy (estimated via running the policy several times) found in the practice episode up to that point. For example, data points recorded for 50 episodes show the value of the best policy found after any of the first 50 episodes. This evaluation reflects that in practice a user would use the best policy found during the session, rather than simply use the final policy. However, our conclusions are unchanged if instead we employ the value of the actual policy resulting after each episode.

Benefit of Critiques. Pure RL is unable to learn a winning policy on either map.[2] We also see that in almost all cases, as the amount of critique data increases, the performance improves for a fixed number of practice episodes.

[2]We have found that, in this domain, our pure RL approach is generally more effective than several other alternatives including OLPOMDP and Q-learning.

Map 1 Results

	%C \ $\lvert T\rvert$	0	10	20	30	40	50	60	70	80	90	100
RL	0	-230.9	-188.6	-170.5	-166.9	-166.9	-132.5	-124.6	-108.7	-104.9	-81.8	-79.2
User 4	25	-166.0	-103.0	-103.0	-73.0	-68.4	-59.4	-59.4	-59.4	-47.8	-41.1	-34.8
	50	-33.0	-33.0	88.0	89.2	89.2	89.2	99.4	99.4	99.4	99.4	99.4
	75	126.0	126.0	126.0	129.6	135.2	135.2	135.2	135.2	135.2	135.2	135.2
	100	160.0	160.0	161.2	161.2	161.2	161.2	161.2	161.2	161.2	161.2	161.2
User 3	25	-119.0	-123.6	-105.4	-97.9	-76.8	-72.8	-72.8	-69.4	-69.4	-69.4	-49.0
	50	-86.0	-38.2	-38.2	-35.6	-27.6	-20.2	-9.8	-9.8	-9.8	-9.0	-9.0
	75	-130.0	-88.3	-28.4	34.0	45.6	47.0	47.0	47.0	76.8	76.8	76.8
	100	-96.0	-93.0	65.9	78.3	78.7	83.1	83.1	91.5	91.5	91.5	92.6

Map 2 Results

RL	0	-210.9	-130.6	-66.1	-61.6	-43.9	-43.9	-40.6	-40.6	-35.4	-33.7	-33.7
User 1	25	-147.0	-88.7	-70.4	-52.8	-44.0	-44.0	-29.6	-26.9	-18.3	-18.3	-5.7
	50	-123.0	-84.9	-20.0	-13.8	-13.8	-4.4	-4.4	-0.4	-0.4	-0.4	-0.4
	75	-35.0	-35.0	-29.8	-4.2	6.4	15.2	29.0	35.2	35.2	50.0	56.0
	100	21.0	21.0	33.0	51.2	51.4	60.8	77.9	86.3	87.6	87.6	87.6
User 2	25	-156.0	-154.8	-117.4	-116.4	-113.2	-112.4	-102.8	-101.4	-96.0	-84.4	-83.0
	50	-160.0	-116.9	-57.2	-39.4	14.8	18.2	18.2	39.4	39.4	53.6	53.6
	75	-48.0	-17.8	-14.0	37.2	52.4	58.8	58.8	76.2	76.2	82.2	95.0
	100	64.0	64.0	64.0	72.4	76.4	96.0	117.6	118.8	118.8	118.8	118.8

Table 1: Results of simulated experiments for four end-users. Rows correspond to using different percentages of the critique data, with the number of practice episodes $|T|$ increasing from 0 to 100. Negative values in a cell indicate a losing policy, and positive values indicate a winning policy. The magnitudes reflect the total hit points remaining for the winning team.

	Map 1		Map 2		both maps	
performance level	Supervised	Combined	Supervised	Combined	Supervised	Combined
-50	6/6	4/4	4/4	6/6	10/10	10/10
0	6/6	3/4	4/4	6/6	10/10	9/10
50	5/6	2/4	4/4	6/6	9/10	6/10
80	3/6	1/4	3/4	3/6	6/10	4/10
100	1/6	0/4	1/4	2/6	2/10	2/10

Table 2: Results of user studies on Map 1 and Map 2. Each row corresponds to a policy performance level, indicated by the first column entry. Each table entry lists the fraction of participants for the map/system pair that were able to achieve the corresponding performance level within the time constraints of the user study. For reference, using much more time, pure RL was unable to achieve level 0 for Map 1 and was only able to achieve a performance of 12 on map 2.

That is, the learning curves associated with less critique data are dominated by the learning curves associated with more critique data. The only exception is User 3, where moving from 50% to 75% critique data hurt performance for 0 and 10 RL episodes. However, beyond 10 RL episodes, the combined system is able to recover, and we see improvement in going from 50% to 75%. This is possibly due to the introduction of noisy critiques between 50% and 75% that could dominate when too few practice episodes are available. In all cases, a winning policy was found using no more than 75% of the critique data. We ran pure RL for up to 500 episodes and found that it achieved -7.4 on Map1 and 12.1 on Map 2, both significantly worse than the combined system. These results give strong evidence that our approach is effective at leveraging end-user critique data to improve the effectiveness of practice.

Benefit of Practice. Each column of data shows the performance trend of increasing the amount of critique data for a fixed amount of practice. First, we see that for all but one user, the pure supervised approach (column $|T| = 0$) was able to find a winning strategy when considering 100% of the critique data. This indicates that even with no practice, the critique data from most users was sufficient to outperform pure RL. Next, comparing columns from left to right (increasing practice) there is a relatively rapid improvement in performance for each fixed amount of critique data. Overall, this provides strong evidence that our approach is able to leverage practice episodes in order to improve the effectiveness of a given amount of end-user critique data.

6.2 Results of User Study

Table 2 summarizes our user study. Each row corresponds to a particular performance level and each column corresponds to a particular learning system (pure supervised or combined) either on Map 1, Map 2, or the combined map results. Each entry in the table lists the fraction of end-users, for the particular system/map combination, that were able to achieve the corresponding performance level during the study. For example, on Map 1, 5 of the 6 users that trained the supervised system achieved a performance of at least 50.

Comparing to Pure RL. Recall that the performance of the pure RL system after 500 episodes is -7.4 and 12.16 on Map 1 and 2 respectively, which requires significantly more time than our users were given. On Map 1 all users were able to achieve -50 using either system, and all but one user (for the combined system) was able to achieve 0. Half of the users were able to achieve 50 with the combined system, and all but one achieved 50 with the pure supervised system. For Map 2, all users were able to achieve at least 50 using either system. These results show that the users were able to significantly outperform pure RL using both systems.

Comparing Combined and Pure Supervised. The end-users had slightly greater success with the pure supervised system versus the combined system. In particular, more users were able to achieve performance levels of 50 and 100 using the supervised system. Based on the questionnaire feedback and our observations of the users, we conjecture a combination of reasons for this difference.

Some users found the pure supervised system to be more user friendly, since there was no delay experienced while waiting for the practice stages to end. Because of this, the users were able to see the effects of their critiques almost immediately, and as a result they gave more and possibly better advice. For the majority of the users, the amount of advice given to the pure supervised system was nearly twice that given to the combined system. Note that in a real scenario the end-user would not need to wait around for the practice period to end, but could periodically check in on the system.

One could argue that the practice episodes of the combined system should compensate for the reduced amount of critique data. However, it was observed that, for some users, the policies returned after certain practice sessions were quite poor and even appeared to ignore the user's previous critiques. The users found this quite frustrating, and it likely impacted the quality of their critiques. We believe that such occurrences can be partly attributed to the fact that the amount of practice used in the user study between critique stages was not sufficient to overcome the reduction in the amount and perhaps quality of critique data. Indeed our simulated experiments showed that, with a reasonable amount of practice, there were significant performance gains over no practice at all. An important future investigation is to design a user study where the user is not captive during the practice sessions, but rather checks in periodically, which will better reflect how the combined system might be used in reality. Nevertheless, in general, it is likely that with our current combined system, users will experience occasional decreases in performance after practice and perceive that the system has not taken their critiques into account. A lesson from this study is that such behavior can be detrimental to the user experience and the overall performance achieved.

7 Future Work

Our results are promising. However, the user study suggests significant usability challenges when end-users enter the learning loop. As future work we plan to conduct further user studies in order to pursue the most relevant directions including: enriching the forms of advice, increasing the stability of autonomous practice, studying user models that better approximate end-users, and incorporating end-users into model-based RL systems.

Acknowledgments. The authors acknowledge support of the Defense Advanced Research Projects Agency (DARPA) under grant HR0011-07-C-0060. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA.

References

Argall, B.; Browning, B.; and Veloso, M. 2007. Learning by demonstration with critique from a human teacher. In *International Conference on Human-Robot Interaction*.

Argall, B.; Browning, B.; and Veloso, M. 2008. Learning robot motion control with demonstration and advice-operators. In *IROS*.

Billard, A.; Calinon, S.; Dillmann, R.; and Schaal, S. 2008. Robot programming by demonstration. In *Handbook of Robotics*.

Coates, A.; Abbeel, P.; and Ng, A. 2008. Learning for control from multiple demonstrations. In *ICML*.

Jin, R., and Ghahramani, Z. 2003. Learning with multiple labels. In *NIPS*.

Knox, W., and Stone, P. 2009. Interactively shaping agents via human reinforcement: the TAMER framework. In *International Conference on Knowledge Capture*.

Knox, W. B., and Stone, P. 2010. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*.

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

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

Maclin, R.; Shavlik, J.; Torrey, L.; Walker, T.; and Wild, E. 2005a. Giving advice about preferred actions to reinforcement learners via knowledge-based kernel regression. In *AAAI*.

Maclin, R.; Shavlik, J.; Walker, T.; and Torrey, L. 2005b. Knowledge-based support-vector regression for reinforcement learning. *Reasoning, Representation, and Learning in Computer Games*.

Peshkin, L., and Shelton, C. 2002. Learning from scarce experience. In *ICML*.

Rosenstein, M., and Barto, A. 2004. Supervised actor-critic reinforcement learning. In Si, J.; Barto, A.; Powell, W.; and Wunsch, D., eds., *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*. 359–380.

Thomaz, A., and Breazeal, C. 2008. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence* 172(6-7):716–737.

Tsoumakas, G., and Katakis, I. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3(3):1–13.