Enabling Robots to Communicate Reward Functions

Sandy H. Huang, David Held, Pieter Abbeel, Anca Dragan

Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, CA 94709

Abstract

Understanding a robot's reward function is key to anticipating how the robot will act in a new situation. Our goal is to generate a set of robot behaviors that best illustrates a robot's reward function. We build on prior work modeling inference of the reward function from example behavior via Inverse Reinforcement Learning (IRL). Prior work using IRL has focused on people teaching machines and assumes exact inference. Our insight is that when teaching people, they will not perform exact inference. We show that while leveraging models of noisy inference can be beneficial, it is also important to achieve coverage in the space of possible strategies the robot can use. We introduce a hybrid algorithm that targets informative examples via both a noisy inference model *and* coverage.

1 Introduction

Despite its importance in human-robot interaction, a robot's behavior is often difficult for a user to predict, since this behavior can vary significantly depending on the robot's objective. Formally, in general a robot's objective is to maximize its cumulative discounted reward, in expectation over some horizon [12]. The reward function may be manually specified, or it may be learned through reinforcement or interaction with a human teacher [2, 14]. Regardless of where this reward function comes from, it is what drives the robot's behavior.

Our thesis is that a user's understanding of this reward function is key to enabling him or her to predict what the robot would do in a novel situation – thus improving collaboration, safety, and trust. Making robots' reward functions more *transparent* also enables users to provide more useful *feedback* to the robot regarding their own preferences about how the robot should perform tasks [15]. Teaching users the robot's reward function is natural because humans implicitly assume agents choose actions to maximize expected reward, and infer the desires of others based on observing their behavior [8]. An alternative to communicating the robot's reward function is communicating its policy, or value function. However, policies do not extrapolate. The policy for a driver on the highway does not generalize to intersections, but knowing that the driver is aggressive (captured by the reward function) enables the user to predict what it will do at an intersection as well.

Unfortunately, a robot cannot just write down its reward function in a way that is interpretable to the average user: they would not be able to use it to predict the robot's behavior in new situations. However, what the robot can do is *showcase* its reward function with *informative* behavior. Imagine seeing an autonomous car drive. You immediately draw inferences about how aggressive it is, how close it is willing to get to other cars, how much it values a smooth ride. But that only applies if you happen to see the car in an interesting situation – when driving forward on an empty highway, all cars drive similarly. Likewise with other tasks, some instances of behavior are more informative.

For robots to generate informative behaviors, they need to have a model of how a human user might interpret that behavior and relate it to the robot's reward function. In robotics, we already have such a model: Inverse Reinforcement Learning (IRL) [10]. IRL uses demonstrations of behavior to

learn a reward function that explains these behaviors. Work in algorithmic teaching has successfully used exact-inference IRL as the learner model [6] to generate informative behaviors for teaching *machines*. The challenge in applying this method to teach *people* is that while machines can perform exact inference, people cannot: *Our insight is that the classical, or exact-inference IRL model, is inadequate for teaching people, as the robot teacher needs to account for noise in human inference.*

Our contribution is two-fold: an analysis of noisy human inference models, and a coverage-augmented noise-based teaching method. First, we investigate several ways to model noise in the learner's inference procedure. We differentiate between *distance metrics* (axes for noise), and between noise models with a *deterministic* versus *probabilistic* effect on the elimination of possible reward functions. We run a user study to analyze these noise factors in the autonomous driving domain, and find that while most models perform in the same range as the exact-inference IRL model, one particular model significantly improves performance.

Although this noise model outperformed showing the user random examples of optimal robot behavior, the difference was not statistically significant. We discover that users do well in test environments in which the car's optimal strategy was observed in one of the training examples. We thus introduce a coverage-augmented algorithm, which leverages the user model while covering the space of possible strategies. Results from a user study show that this method now significantly outperforms the random baseline, whereas just augmenting that baseline with the same coverage principle does not.

Overall, our results suggest that when it comes to communicating reward functions to people, we cannot rely on an exact-inference model: people are noisy in particular ways, and the training examples that they see bias their understanding of what kinds of strategies a robot can adopt.

2 Related Work

Algorithmic Teaching. Our work is closely related to *algorithmic teaching*, or *machine teaching*. Algorithmic teaching focuses on selecting the optimal sequence (or set) of training examples to teach a learner a particular concept, given knowledge of the learner's learning algorithm [17]. This is an open problem, but there are approximate algorithms for broad classes of learners [16]. *Cooperative*, or *interactive*, teaching combines algorithmic teaching with feedback about the learner's current hypothesis after each training example [3, 19]. Cakmak and Lopes [6] extended algorithmic teaching to work for sequential decision tasks. Their approach teaches reward functions to a machine learner, for tasks with discrete state and action spaces. In contrast, our goal is to teach humans, and our domain has continuous state and action spaces.

Teaching Humans. Prior work has also applied algorithmic teaching to teaching humans. Singla et al. propose modeling the human as taking a random walk in the space of possible hypotheses (e.g., classification boundaries), staying at the current hypothesis until it contradicts a training example [13]. Patil et al. show that algorithmic teaching with a limited-capacity model for the human improves the effectiveness of the selected teaching examples, as opposed to using an unlimited-capacity model [11].

Prior work has also explored heuristic approaches to teach humans. Curriculum design, in which the training examples become increasingly difficult, is well-known as a heuristic that improves human performance [4, 5], and is also a strategy humans naturally use while teaching others [9]. In contrast to previous work on teaching humans, our training examples are robot trajectories rather than images. Additionally, our task of teaching reward functions is regression rather than classification.

Familiarization. Dragan and Srinivasa explored using familiarization to improve the predictability of robot arm reaching motions. The examples they show uniformly cover the space of possible reaching goals. They found that familiarization increases comfort with the robot and improves predictability of the robot's motions [7]. In contrast, our approach *optimizes* for which environments are most useful to show robot trajectories in, since the space of possible examples to show is larger.

3 Modeling Human Learning

3.1 Exact-Inference Model

Inverse Reinforcement Learning. Inverse Reinforcement Learning (IRL) [10] is the process of learning a distribution over reward functions from demonstrations. The motivation behind performing

IRL, rather than directly learning a policy from the demonstrations, is that learning the reward function should lead to better generalization to different versions of the task [1, 10]. Typically the learner (or agent) is a robot, and the demonstrations are provided by a human. But IRL can also be used as a model for how a human might infer the reward function from demonstrations by a robot.

We assume the reward function is a linear combination of features [1]:

$$R_\theta(s_t, a_t, s_{t+1}) = \theta^\top \phi(s_t, a_t, s_{t+1}), \tag{1}$$

where s_t is the state and a_t is the action taken at time t. There are no restrictions on how complex these features can be. Given a trajectory ξ composed of a sequence of state-action pairs $((s_0, a_0), (s_1, a_1), \ldots, (s_T, a_T))$, its cumulative discounted reward is $R_\theta(\xi) = \theta^\top \phi(\xi)$, where $\phi(\xi) = \sum_{t=0}^{T-1} \gamma^t \phi(s_t, a_t, s_{t+1})$. γ is a discount factor between 0 and 1 that favors earlier rewards.

Algorithmic Teaching for IRL. Let ξ_E^θ be the optimal trajectory with respect to R_θ in environment E: $\xi_E^\theta = \arg\max_{\xi_E} \theta^T \phi(\xi_E)$. Cakmak and Lopes [6] propose an algorithmic teaching approach to IRL, where the goal is to communicate reward weights θ^* to an IRL agent with as few optimal demonstration trajectories $\xi_{E_{1:n}}^{\theta*}$ as possible. The approach selects demonstrations to minimize the space of valid reward parameters: reward parameters under which any demonstrated trajectory is not optimal are considered invalid. This approach is efficient given that the learner has perfect perception and computation. We show it does not translate well to teaching noisy human learners.

3.2 Noisy Inference Models

Probabilistic IRL Teaching. Algorithmic teaching for IRL assumes exact inference: that the learner will exactly eliminate any possible θ whose optimal trajectory does not *perfectly* match the demonstration. Instead, we generalize [6] to a *probabilistic* approach – we model the learner as maintaining a probabilistic distribution over θ. We optimize for the sequence of optimal trajectories that will lead to the highest probability of the robot's actual θ, namely θ^*:

$$\arg\max_{\xi_{E_{1:n}}^{\theta*}} P(\theta^* | \xi_{E_{1:n}}^{\theta*}), \tag{2}$$

where $P(\theta^* | \xi_{E_{1:n}}^{\theta*})$ is computed via Bayesian inference as

$$P(\theta^* | \xi_{E_{1:n}}^{\theta*}) = \frac{P(\xi_{E_{1:n}}^{\theta*} | \theta^*) P(\theta^*)}{\sum_\theta P(\xi_{E_{1:n}}^{\theta^*} | \theta) P(\theta)} = \frac{P(\theta^*) \prod_{i=1}^n P(\xi_{E_i}^{\theta^*} | \theta^*)}{\sum_\theta P(\theta) \prod_{i=1}^n P(\xi_{E_i}^{\theta^*} | \theta)} \tag{3}$$

Deterministic vs. Probabilistic. We consider both deterministic θ elimination as well as each example continuously adjusting the probability of a θ. Given any distance metric d between trajectories,

- For deterministic effect, $P(\xi_E^{\theta^*} | \theta) \propto 0$ if $d(\xi_E^\theta, \xi_E^{\theta^*}) > \tau$, or 1 otherwise.
- For probabilistic effect, $P(\xi_E^{\theta^*} | \theta) \propto e^{-\lambda \cdot d(\xi_E^\theta, \xi_E^{\theta^*})}$.[1]

The deterministic effect models users who will either completely eliminate a θ or not, and will not eliminate θs for which the optimal trajectory is close to the example. In contrast, the probabilistic effect decreases the probability of θs that have distant optimal trajectories, never fully eliminating any. Exact-inference IRL [6] is a special case of deterministic reward-based with $\tau = 0$ (i.e., no noise).

Distance Metrics. We consider three possible distance metrics d, each of which makes a different assumption about user noise:

- reward-based[2]: $d(\xi_E^\theta, \xi_E^{\theta^*}) = \theta^\top (\phi(\xi_E^\theta) - \phi(\xi_E^{\theta^*}))$. This assumes users cannot perfectly distinguish between rewards of trajectories for a given setting of the reward parameters, so if $R_\theta(\xi_E^{\theta^*}) \approx R_\theta(\xi_E^\theta)$, then $P(\xi_E^{\theta^*} | \theta)$ will be high.
- Euclidean-based: $d(\xi_E^\theta, \xi_E^{\theta^*}) = \frac{1}{T} \sum_{t=1}^T ||s_{E,t}^\theta - s_{E,t}^{\theta^*}||_2$, where $s_{E,t}^\theta$ is the state at time t. This assumes users cannot perfectly distinguish between perceptually-similar trajectories.

[1] We noticed normalizing this distribution produced very similar results to leaving it unnormalized, so that is what we do in our experiments, similar to other algorithmic teaching work not based on reward functions [13].

[2] Note that this is always positive because ξ_E^θ has the highest reward w.r.t. θ

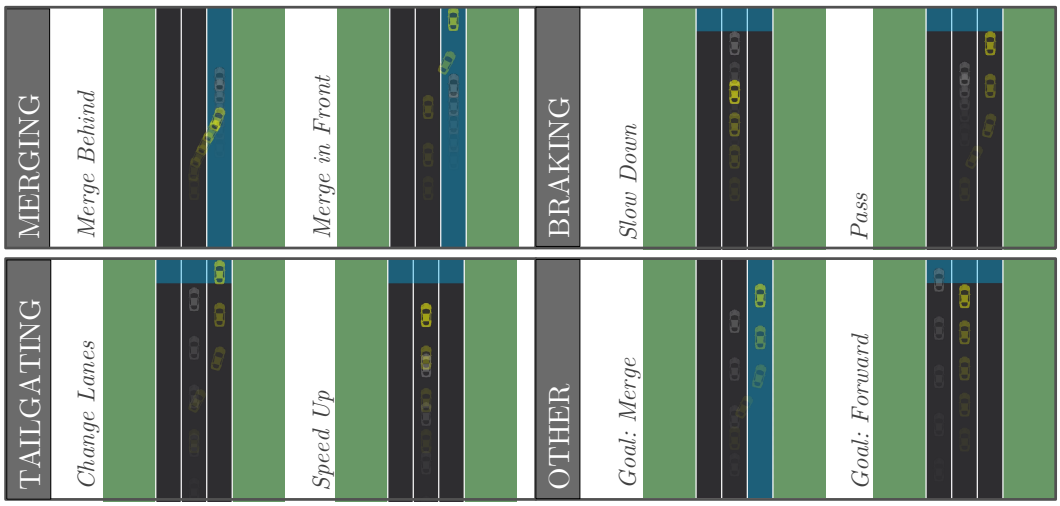

Figure 1: Our driving environments can be naturally divided into four classes, with two trajectory strategies per class. Each image shows the trajectories of the autonomous car (yellow) and non-autonomous car (gray) in a given environment. Positions later in the trajectory are more opaque. Blue indicates the goal of the robot: either merge to the right or drive forward.

- strategy-based: $d(\xi_E^\theta, \xi_E^{\theta^*}) = 0$ if ξ_E^θ and $\xi_E^{\theta^*}$ are in the same strategy cluster, ∞ otherwise. This assumes users perceive all trajectories in the same strategy cluster as the same.[3]

Relation to MaxEnt IRL. MaxEnt IRL [18] is an IRL algorithm that assumes demonstrations are noisy. In algorithmic teaching, this is similar to the learner doing a noisy optimization of the reward function. The MaxEnt distribution is the same as our *probabilistic reward-based* model:

$$P(\xi_E^{\theta^*}|\theta) \propto e^{\lambda\theta^T\phi(\xi_E^{\theta^*})} \propto e^{\lambda(\theta^T\phi(\xi_E^{\theta^*})-\theta^T\phi(\xi_E^{\theta}))} = e^{-d_{\text{reward}}(\xi_E^\theta,\xi_E^{\theta^*})} \tag{4}$$

Implementation Details. Given a noise model $\mathcal{M}$ that predicts $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$, our approach greedily selects environment E_t to maximize $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:t}}^{\theta^*})$. It stops either when no environment E will improve this probability, or when ten examples have been selected.

To select τ or λ for each noise model, we do a coarse search over $\{10^{-5}, 10^{-4}, \ldots, 10^4, 10^5\}$ and choose the value that results in selection of the maximum number of unique environments to show, and where the increase from $P_{\mathcal{M}}(\theta^*|\xi_{E_1}^{\theta^*})$ to $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$ is at least 0.1.

4 Experimental Domain

We evaluate the performance of our proposed noise models in the task of communicating the driving style of a simulated autonomous car. In this domain, the goal is for users to witness examples (in simulation) of how the car drives, so that they can anticipate how it will drive when they ride in it.

Driving Simulator. We model the dynamics of the car with the bicycle vehicle model. Let the state of the a car be $\mathbf{x} = [x \;\; y \;\; \theta \;\; v \;\; \alpha]^\top$, where (x, y) are the coordinates of the center of the car's rear axle, θ is the heading of the car, v is its velocity, and α is the steering angle. Let $\mathbf{u} = [u_1 \;\; u_2]^\top$ represent the control input, where u_1 is the change in steering angle and u_2 is the acceleration. Additionally, let L be the distance between the front and rear axles of the car. Then the dynamics model is

$$[\dot{x} \;\; \dot{y} \;\; \dot{\theta} \;\; \dot{v} \;\; \dot{\alpha}] = [v \cdot cos(\theta) \quad v \cdot sin(\theta) \quad \frac{v}{L}tan(\alpha) \quad v \cdot u_1 \quad u_2] \tag{5}$$

Environments. We search over 21,216 environments of highway driving configurations (Table 1). Each environment has three lanes, and a single non-autonomous car. These driving environments can

[3] Strategy-based distance is the only task-specific distance metric we consider. One could cluster trajectories to automatically obtain these different strategies. In our experimental domain, it was straightforward to categorize them for each environment type. In other applications, they could be homotopy classes.

Table 1: Environment Parameters

Axis of Variation	**Acceptable Values**
Goal	[merge to right, drive forward]
Distance between autonomous and non-autonomous car	[-240, -220,. . . , -100], [100, 120, . . . , 240]
Lane of non-autonomous car	[Left, Center, Right]
Initial velocity, non-autonomous car	[20, 25, . . . , 80]
Acceleration time, non-autonomous car	[0, 0.5, 1, 1.5, 2]
Final velocity, non-autonomous car (if acceleration time $\neq 0$)	[20, 30, 70, 80]

Table 2: The performance of each user model, as evaluated by all other models. Row: model of user used for evaluation; column: how training examples were generated

	exact-inf. IRL	determ, reward	determ, Euclidean	determ, strategy	prob, reward	prob, Euclidean	prob, strategy	random
exact-inference IRL	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
deterministic, reward	0.002	**1.000**	0.333	0.004	**1.000**	0.500	0.004	0.002
deterministic, Euclidean	0.001	0.014	**1.000**	0.013	0.030	**1.000**	0.013	0.001
deterministic, strategy	0.007	0.002	0.333	**1.000**	0.050	**1.000**	**1.000**	0.001
probabilistic, reward	0.003	0.372	0.250	0.008	**0.975**	0.508	0.008	0.003
probabilistic, Euclidean	0.002	0.009	0.027	0.008	0.113	**0.208**	0.008	0.002
probabilistic, strategy	0.007	0.002	0.333	**1.000**	0.050	**1.000**	**1.000**	0.001

naturally be categorized into four classes (merging, braking, tailgating, and other), with two trajectory strategies per class (Figure 1). The "other" class contains all environments where the autonomous car can reach the goal without interacting with the non-autonomous car.

Reward Features. The reward of a trajectory ξ is a linear combination of these reward features:

- distance to other cars: we center a Gaussian on the non-autonomous car, with the major axis along that car's heading
- acceleration, squared: $\sum_{t=1}^{T}(s_{t+1}[3] - s_t[3])^2$
- deviation from initial speed, squared: $\sum_{t=1}^{T}(s_t[3] - s_1[3])^2$
- turning: $\sum_{t=1}^{T}|s_t[2] - s_1[2]|$
- distance from goal: $\sum_{t=1}^{T}\max(0, (s_0[0] + w) - s_t[0])^2$ if the goal is to merge into the right lane, and $s_T[1]$ if the goal is to drive forward. w is the width of one lane.

The last four features do not depend on the environment, so we normalize them such that the maximum value of that feature across all trajectories is 1 and the minimum is 0.

Optimal θ. We select $\theta^* = [-64 \quad -0.1 \quad -1 \quad -0.1 \quad -0.5]^\top$: a reward function that is not too cautious when it comes to staying away from other cars, which is atypical for autonomous cars. [4]

5 Analyzing the Noise Models with Simulated Users

In Section 3.2, we introduced 6 possible noisy inference user models $\mathcal{M}$ We generate the optimal sequence of examples for each model $\mathcal{M}$, that greedily maximizes $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$. Figure 2 plots the types of examples that each algorithm selected for its optimal sequence. We also generate a random sequence, and the optimal sequence for the exact-inference model.

Relative Evaluation. We evaluate, for each algorithm, how well its sequence performs with respect to the other user models. We measure performance based on $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, where higher values are better. Table 2 shows the results. First, we see that the sequence generated by an algorithm performs best for that algorithm, and this is by design – the sequence optimizes that algorithm's user model.

[4] In future work, we will explore how other values of θ^* affect teachability.

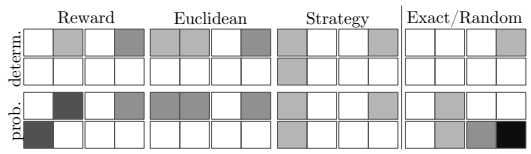

Figure 2: The number of examples shown in each of the 8 trajectory strategies, for each of the noisy inference models, exact-inference, and the random baseline. White = 0 shown, black = 4. Each 2x4 grid corresponds to strategies as in Figure 1.

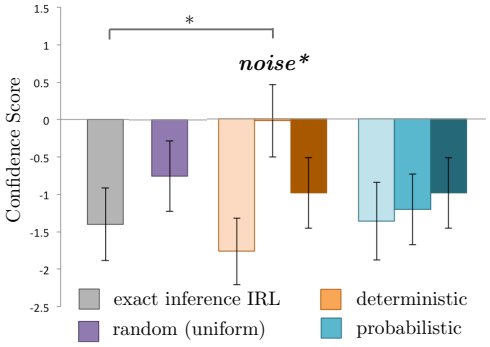

Figure 3: Performance of human participants on identifying the car's trajectory in test environments, after seeing the examples selected by the model. The deterministic and probabilistic conditions are our proposed models of human inference noise. For these conditions, the lightest shade (left) corresponds to the model with reward-based distance, the medium shade Euclidean-based distance, and the darkest shade (right) strategy-based distance.

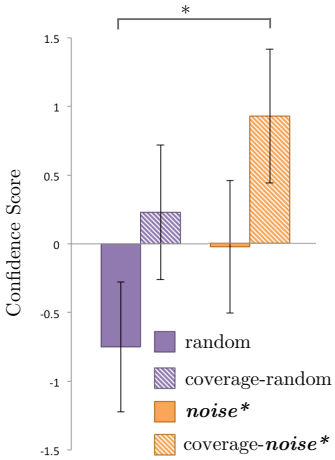

Figure 4: We explore adding coverage to the sequence of environments selected by our best-performing model from the previous experiment, deterministic effect with Euclidean-based distance. Participants in this condition performed significantly better than those in the original random condition. In contrast, enforcing coverage but selecting random sequences does not lead to statistically significantly better performance.

All algorithms perform just as well under the exact-inference IRL model assumption – even random, because it gives enough examples that it ends up perfectly eliminating any wrong reward parameters. This suggests that this metric is not discriminative enough. It is unlikely that real users will be able to perfectly learn the reward parameters from any of these sequences of training examples. In particular, the random sequence is very uninformative according to every metric except exact-inference IRL.

6 User Study on Noise Models

6.1 Experiment Design

Manipulated Variables. We manipulated two variables for noisy inference, the type of noise (either ***deterministic*** or ***probabilistic***) and the distance metric (***reward-based***, ***Euclidean-based***, or ***strategy-based***), in a 2 by 3 factorial, for a total of six noisy inference models. For the strategy-based distance metric, the probabilistic effect does not matter since distances are either 0 or ∞, so there are five unique noisy inference models. We compare these models against two baseline approaches, exact-inference IRL and uniform random selection of training environments. We show the participant one training environment at a time, in the order that the examples were selected.

Dependent Measures. We are interested in how well participants learn a specific setting of reward parameters θ^*. We evaluate this by asking them to identify the trajectory that most closely matches the autonomous car's driving style in a few test environments, each with four trajectory options. We would like to test user performance over a variety of environments, so we randomly select one test environment for each of six non-"other" trajectory strategy clusters. For each test environment, we show two trajectory options in each strategy cluster, and ensure the alternate trajectories under θ^* are above a reward-based distance threshold, to make sure they do not look too similar.

We have two dependent variables: the fraction of participants who correctly identify $\xi_{\theta^*}^{E_{\text{test}}}$ for each test environment E_{test}, and their confidence in their answer. We combine them in a confidence score: positive confidence if they are correct, negative confidence if not – this captures that it is best to be correct and confident, but better to be incorrect and not confident than both incorrect and confident.

Hypothesis. *Accounting for noisy inference significantly improves performance*. We leave open which noise models work well; the goal is to identify which best captures participants' inferences.

Subject Allocation. We ran a between-subjects experiment on a total of 224 participants across the seven conditions, recruited via Amazon Mechanical Turk. At the end of the experiment, we ask participants what the two possible driving goals are, to filter out participants who were not paying attention. 35 out of the 224 (15.6%) did not answer this control question correctly. After filtering, there were 25 remaining in the exact-inference IRL condition, 28 in uniform random, 29 in deterministic reward-based, 27 in deterministic Euclidean-based, 24 in probabilistic reward-based, 27 in probabilistic Euclidean-based, and 29 in strategy-based. The average age of the 189 non-filtered participants was 36.4, with a standard deviation of 10.9. The gender ratio was .46 female.

Correcting for Confounds. To give the random baseline the best chance, we select eight examples, which is the maximum number of examples shown by any of the other conditions. Since we might generate a particularly informative or uninformative random sequence by chance, we sample 1000 random sequences, and sort them based on $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$ where $\mathcal{M}$ is the exact-inference IRL model. Then we choose the median sequence in that ranking, which will have median informativeness.

6.2 Analysis

Noise Models. We ran a factorial ANOVA on ***confidence score*** with distance measure and determinism as factors. We found no effect for determinism, but a marginal effect for ***distance*** ($F(2, 163) = 2.69$, $p = .07$), with the Euclidean distance performing the best, and the reward distance performing the worst. On average, probabilistic models performed slightly worse than deterministic ones. Results for ***accuracy*** were analogous. The best noise model used Euclidean distance with deterministic noise effects. We refer to this as the ***noise**** model (Figure 3).

Utility of Noise. Despite showing more examples, most noise models did not perform much better than exact-inference IRL. In fact, the deterministic reward-distance model performed worse, and the probabilistic one performed almost as poorly. This shows that not just any noise model is useful. To test the utility of incorporating noise, we compared the best noise model, ***noise****, with perfect IRL, and found a significant improvement (Welch's t-test p-value of 0.025). *This supports our hypothesis that incorporating noise helps, with the caveat that not just any noise will work.*

Utility of User Modeling. Algorithmic teaching is rooted in the assumption that it is useful to model the learner and use that model to compute the optimal sequence of teaching examples. Our results surprisingly challenge that assumption. The random condition outperformed exact-inference IRL (although not significantly). When comparing our best performer with a model, namely ***noise****, with showing the users random examples, we found that there was no significant improvement.

Coverage. Digging deeper, we realized participants tended to perform well on test cases in classes in which they saw a training example. In addition, if they see one trajectory strategy in the training examples and not the opposite strategy, they tend to think the autonomous car will always take the first strategy in that environment type. We found a strong correlation between the number of *helpful* environments shown and participants' confidence scores, with a Pearson correlation coefficient of $r = 0.83$. Given that x training examples are shown in trajectory strategy A and y from B, we define the number of helpful environments shown in A as equal to x if $x > 0$, and equal to $-y$ otherwise. We leverage this result to introduce augmented algorithms that ensure coverage.

7 User Study on Coverage

Coverage-Augmented Algorithmic Teaching. Since coverage correlates with better user performance, we add a coverage term to our optimization over which example trajectories $\xi_{E_{1:n}}^{\theta*}$ to show:

$$\arg\max_{\xi_{E_{1:n}}^{\theta*}} P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta*}) + \lambda \sum_s \mathbb{1}[\exists i, h(\xi_{E_i}^{\theta*}) = s], \tag{6}$$

where the sum is over trajectory strategies s and the function h maps a trajectory to its strategy cluster. If we set the trade-off parameter λ at 0 until $[P_{\mathcal{M}}(\theta^*|\xi^{\theta*}_{E_{1:t}}) - P_{\mathcal{M}}(\theta^*|\xi^{\theta*}_{E_{1:t-1}})] < \epsilon$, then first examples that maximize the probability of θ^* will be selected, and only after no further examples will significantly improve the probability of θ^*, extra examples are selected to provide coverage across strategies. We select these extra examples by choosing the best with respect to $\mathcal{M}$, to ensure they are informative. Using this approach, we augment our best noise model, ***noise****, to achieve coverage.

7.1 Experiment Design

Manipulated Variables. We manipulate two variables: whether we augment the training examples with coverage, and whether we use a user model to generate the examples or sample randomly. We select our best model for the former, ***noise****. From the previous experiment, we already have participant data along the no-coverage dimension – for random and ***noise**** – so we run this experiment on only the two new conditions that incorporate coverage, which we will refer to as coverage-random and coverage-***noise****. We generate random sequences with coverage by randomly selecting exactly one environment from each of the eight trajectory strategies. We keep the same dependent measures as in our previous experiment (Section 6.1).

Hypothesis. We hypothesize that augmenting with coverage will improve participants' performance on both models, random and ***noise****, compared to the respective models without coverage.

Subject Allocation. We ran this experiment between-subjects, on a total of 63 participants. 10 out of the 63 (15.9%) did not answer the control question correctly. After filtering, there were 27 remaining in the coverage-random condition, and 26 in coverage-***noise****. The average age of the 53 non-filtered participants was 34.43, with a standard deviation of 9.0. The gender ratio was 0.53 female.

7.2 Analysis

We ran a factorial ANOVA on ***confidence score*** with coverage and model as factors. We found a marginal effect for coverage ($F(1, 107) = 1.82$, $p = .07$), suggesting that coverage improves performance. There was no interaction effect, suggesting that coverage helps regardless of using a user model for teaching or not.

The coverage-augmented ***noise**** was the best out of the four conditions. Coverage augmentation enabled it to significantly outperform the random baseline (with a Welch's t-test p-value of 0.049), suggesting that coverage is useful. In contrast, the coverage-random condition did not outperform the random baseline (p-value 0.159), suggesting the noisy model is useful (Figure 4).

Overall, coverage alone helped, but was not sufficient to outperform the baseline. From the previous experiment, we know that the noisy user model helped, but was also not sufficient to outperform the baseline. Overall, the improvement is largest (and significant, modulo compensating for multiple hypotheses) when we have a coverage-augmented noisy IRL model. *When leveraged together, coverage with the right noise model have a significant teaching advantages over random teaching, as well as over IRL models that assume perfect users.*

8 Discussion

Summary. We found that a noisy inference model using a deterministic Euclidean-distance update on the space of candidate reward functions performed best at teaching real users, and outperforms algorithmic teaching that assumes exact inference. We also found that only after augmenting this model with a coverage objective did it significantly outperform the random baseline. We were surprised by the performance of the random baseline, and by how difficult it is to get the noisy inference model right. We see this as a valuable area of future investigation.

Limitations and Future Work. The task of teaching reward functions is highly complex, and more work is needed to investigate what an appropriate model for user inferences would be, that addresses both noisy inference and coverage. Furthermore, we only explored a single medium of communication, namely optimal trajectories for the robot in different environments. There might be other, more effective channels. However, we do believe that people draw inferences based on seeing the robot behave, and think that better leveraging the behavior channel and augmenting it with others might be most effective.

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*. ACM Press, 2004.

[2] B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 67:469–483, 2009.

[3] F. J. Balbach and T. Zeugmann. *Recent Developments in Algorithmic Teaching*. 2009.

[4] S. Basu and J. Christensen. Teaching classification boundaries to humans. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, pages 109–115. AAAI Press, 2013.

[5] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, 2009.

[6] M. Cakmak and M. Lopes. Algorithmic and human teaching of sequential decision tasks. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, 2012.

[7] A. Dragan and S. Srinivasa. Familiarization to robot motion. In *Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 366–373, 2014.

[8] J. Jara-Ettinger, H. Gwen, L. E. Schulz, and J. B. Tenenbaum1. The Naïve Utility Calculus: Computational Principles Underlying Commonsense Psychology. *Trends in Cognitive Sciences*, 20(8):589–604.

[9] F. Khan, B. Mutlu, and X. Zhu. How do humans teach: On curriculum learning and teaching dimension. In *Advances in Neural Information Processing Systems 24*, pages 1449–1457. 2011.

[10] A. Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proc. 17th International Conf. on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

[11] K. R. Patil, X. Zhu, L. u. Kopeć, and B. C. Love. Optimal teaching for limited-capacity human learners. In *Advances in Neural Information Processing Systems 27*. 2014.

[12] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[13] A. Singla, I. Bogunovic, G. Bartok, A. Karbasi, and A. Krause. Near-optimally teaching the crowd to classify. In *In Proceedings of the Thirty-first International Conference on Machine Learning*, 2014.

[14] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[15] A. L. Thomaz. *Socially Guided Machine Learning*. PhD thesis, Cambridge, MA, USA, 2006. AAI0809139.

[16] X. Zhu. Machine teaching for bayesian learners in the exponential family. In *Advances in Neural Information Processing Systems 26*. 2013.

[17] X. Zhu. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 2015.

[18] B. D. Ziebart, A. Maas, J. A. D. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, July 2008.

[19] S. Zilles, S. Lange, R. Holte, and M. Zinkevich. Models of Cooperative Teaching and Learning. *Journal of Machine Learning Research*, 2011.