Autonomous Robots (2019) 43:309–326
https://doi.org/10.1007/s10514-018-9771-0

Enabling robots to communicate their objectives

Sandy H. Huang[1] · **David Held**[2] · **Pieter Abbeel**[1] · **Anca D. Dragan**[1]

Received: 3 December 2017 / Accepted: 22 May 2018 / Published online: 18 June 2018

Abstract
The overarching goal of this work is to efficiently enable end-users to correctly anticipate a robot's behavior in novel situations. And since a robot's behavior is often a direct result of its underlying objective function, our insight is that end-users need to have an accurate mental model of this objective function in order to understand and predict what the robot will do. While people naturally develop such a mental model over time through observing the robot act, this familiarization process may be lengthy. Our approach reduces this time by having the robot model how people infer objectives from observed behavior, in order to then show those behaviors that are maximally informative. We introduce two factors to define candidate models of human inference, and show that certain models indeed produce example robot behaviors that better enable users to anticipate what it will do in novel situations. Our results also reveal that choosing the appropriate model is key, and suggest that our candidate models do not fully capture how humans extrapolate from examples of robot behavior. We leverage these findings to propose a stronger model of human learning in this setting, and conclude by analyzing the impact of different ways in which the assumed model of human learning may be incorrect.

Keywords Transparency · Explainable artificial intelligence · Human-robot interaction · Inverse reinforcement learning

1 Introduction

Imagine riding in a self-driving car that needs to quickly change lanes to make a right turn. The car suddenly brakes in order to merge safely behind another car, because it deems it unsafe to speed up and merge in front of the other car. A passenger who knows this self-driving car is defensive and that it values safety much more than efficiency would be able to anticipate this behavior. But passengers less familiar with the car would not anticipate this sudden braking, so they may be surprised and possibly frightened.

This is one of several papers published in *Autonomous Robots* comprising the "Special Issue on Robotics Science and Systems".

✉ Sandy H. Huang
shhuang@cs.berkeley.edu

David Held
dheld@andrew.cmu.edu

Pieter Abbeel
pabbeel@cs.berkeley.edu

Anca D. Dragan
anca@cs.berkeley.edu

[1] University of California, Berkeley, Berkeley CA,, USA

[2] Carnegie Mellon University, Pittsburgh, PA, USA

There are many reasons why it is beneficial for humans to be able to anticipate a robot's movements, from subjective comfort to ease of coordination when working with and around the robot (Sebanz et al. 2006; Dragan et al. 2015). Our goal is to enable end-users to accurately anticipate how a robot will act, even in *novel situations that they have not seen the robot act in before*—like a new traffic scenario, or a new placement of objects on a table that the robot needs to clear.

A robot's behavior in any situation is a direct consequence of the objective (or reward) function the robot is optimizing: (most) robots are rational agents, acting to maximize expected cumulative reward (Russell and Norvig 2009). Whether the robot's objective function is hard-coded or learned, it captures the trade-offs the robot makes between features relevant to the task. For instance, a car might trade off between features related to collision avoidance and efficiency (Levine and Koltun 2012), with more "aggressive" cars prioritizing efficiency at the detriment of, say, distance to obstacles (Sadigh et al. 2016a).

The insight underlying our approach is the following:

The key to end-users being able to anticipate what a robot will do in novel situations is having a good understanding of the robot's objective function.

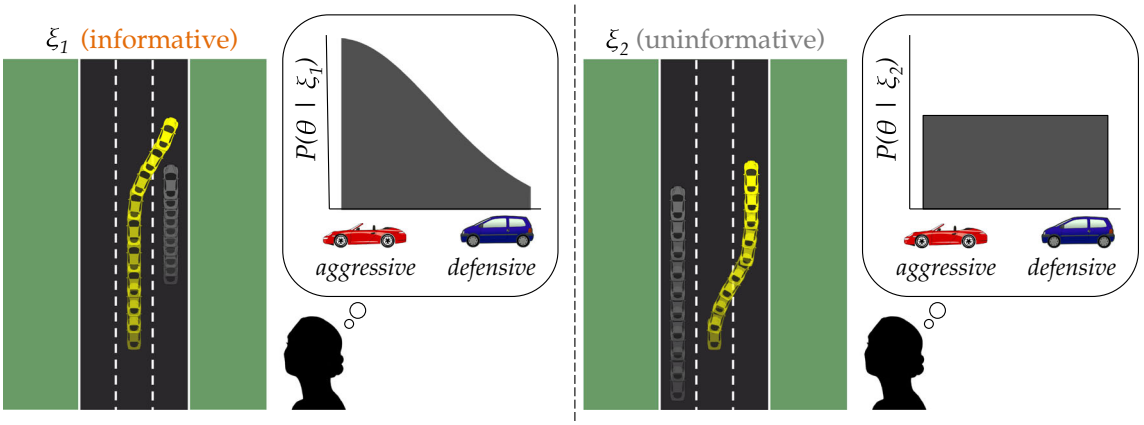

Fig. 1 We show examples ξ of the yellow autonomous car's behavior that are maximally informative in guiding the human toward understanding the robot's objective function (e.g., aggressive versus defensive). For instance, in environments where the car needs to merge into the right lane, its behavior is more informative when there is another car present (left) than when the lane is empty (right). We consider the case where the robot's objective function is represented by a linear combination of features, weighted by θ (Color figure online)

Note that understanding the objective function does not mean users must be able to explicate it—to write down the equation, or even to assign the correct reward to a behavior or a state-action pair. Rather, users only need to have an implicit representation of what drives the robot's behavior, i.e., a qualitative understanding of the trade-offs the robot makes.

Fortunately, users will naturally improve their mental model of how a robot acts, given examples of the robot behaving optimally (Dragan and Srinivasa 2014). Further, evidence suggests that people will use this behavior to make inferences about the robot's underlying objective function (Ullman et al. 2009; Baker et al. 2011; Jara-Ettinger et al. 2016), which will enable them to anticipate its behavior in *novel* situations.

A fundamental challenge with this prior work, including our own, is that it is *passive*: people get exposed to robot behavior in different environments as the robot encounters them. The difference in this work is that we explicitly account for the fact that not all environments are equally informative. In many environments, a robot's optimal behavior does not fully describe the trade-offs that the robot would make in other environments, i.e., parts of the robot's objective will remain under-determined. For example, an autonomous car driving down a highway with no cars nearby will drive at the speed limit and stay in its lane, regardless of its trade-off between efficiency versus staying far away from other cars. Another example is when an autonomous car changes lanes without interacting with any other cars (Fig. 1, right). An end-user mainly exposed to these types of behavior will have difficulty forming an accurate mental model of the robot's objective function and anticipating how the robot will behave in more complex scenarios. On the other hand, suppose an autonomous car speeds up to merge in front of another car, cutting it off (Fig. 1, left). This scenario more clearly illustrates the trade-offs this car makes regarding safety versus efficiency.

We focus on enabling robots to purposefully choose such *informative* behaviors that actively communicate the robot's objective function. We envision a training phase for interaction, where the robot showcases informative behavior in order to quickly *teach* the end-user what it is optimizing for. This training phase may need to happen in a simulator, when it is impractical to construct these informative environments in the real world.

In order to choose the most informative example behaviors for communicating a robot's objective function to humans, we take an *algorithmic teaching* approach (Goldman and Kearns 1995; Balbach and Zeugmann 2009; Rafferty et al. 2011; Zilles et al. 2011; Zhu 2015; Koedinger et al. 2013): we model how humans make inferences about the robot's objective function from examples of its optimal behavior, and use this model to generate examples that increase the probability of humans inferring the correct objective function.

The reverse problem, machines inferring objective functions from observed human behavior, can be solved using inverse reinforcement learning (IRL) (Ng and Russell 2000). Prior work has investigated how to teach an objective function through example behavior to *machine* learners running IRL (Cakmak and Lopes 2012). But the challenge in teaching *people* instead of machines is that while machines can perform *exact* inference, people are likely to be approximate in their inference. People do not have direct access to configuration-space trajectories and the exact environment state, whereas robots do, at least in kinesthetic teaching [and in Cakmak and Lopes (2012)]. People also cannot necessarily distinguish between a perfectly optimal trajectory for one objective and an ever-so-slightly suboptimal one (Vul et al. 2014).

Our main contribution is to introduce a systematic collection of approximate-inference models and, in a user study, compare their performance relative to the exact inference model. We focus on the autonomous driving domain, where a car chooses example behaviors that are informative about the trade-offs it makes in its objective function. We measure teaching performance—how useful the generated examples are in enabling users to anticipate the car's behavior in test situations—and find that one particular approximate-inference model significantly outperforms exact inference (while the others perform on par). This supports our central hypothesis that accounting for approximations in user inference is indeed helpful, but suggests that we need to be careful about *how* we model this approximate inference.

Further analysis shows teaching performance correlates with covering the full space of strategies that the robot is capable of adopting. For instance, the teaching algorithm cannot just show the car cutting others off; it also needs to show an example where it is optimal to brake and merge behind. We show the best results are obtained by a coverage-augmented algorithm that both leverages an approximate-inference user model and encourages full coverage of all possible driving strategies.

The importance of coverage implies that users are not only learning the robot's objective function and generalizing to novel scenarios based on that; they may also be relying on a memory-based, nearest-neighbor-like approach: directly comparing each novel scenario with the ones they have seen the robot act in, and making inferences based on that. Based on this observation, we study a combined approximate-inference and nearest-neighbor model that better captures how users learn in this domain.

We conclude with a discussion of the implications and limitations of our work. At the forefront of these limitations is our model's assumption of the robot and the human sharing a common understanding of the features that the robot's objective function depends on. We present an analysis of how the results change when this is not true.

Our work takes a stab at an important yet under-explored problem of communicating robot objective functions to people. This is important in the short term for human-robot interaction, because it enables transparency and makes robot behavior easier to anticipate. But it is also important in the long term. As AI systems become more capable of optimizing their objective functions, ensuring that these objective functions are aligned with what system designers and end users actually want will become key to ensuring that these systems behave in the intended way. We are hopeful that our work on clarifying these objectives through illustrative examples can help verify objective functions, and contribute towards aligning them with human values. Our results are encouraging, but also leave room for better models of how people extrapolate from observed robot behavior.

2 Related work

A key challenge in our work is modeling how humans make inferences about a robot's objective function from observing its behavior. Humans naturally infer plans, intentions, and goals from the behavior of other agents (Schmidt et al. 1978; Cohen et al. 1981). Recent work proposed a Bayesian framework for human reasoning, which predicts that humans score candidate hypotheses of an agent's beliefs and desires by the probability that they would have led to the observed behavior, weighted by a prior (Baker et al. 2009, 2011; Ullman et al. 2009; Jara-Ettinger et al. 2016). This probability may depend, for example, on the utility the desire would assign to the observed behavior (Jara-Ettinger et al. 2016). This Bayesian framework accurately predicts human reasoning in goal-oriented tasks. We also model humans as performing Bayesian inference, but we are interested in a trajectory-oriented setting. Even when the goal remains the same (e.g., drive to a particular destination), the trajectory a robot takes depends on the trade-offs it makes in its objective function. Thus, human end-users need to understand not only the robot's goal, but also its objective function. In our work, we introduce approximate-inference models of how humans may score candidate hypotheses for objective functions.

Prior work on enabling humans to better anticipate robot motion has relied on modifying the robot's motions, either by making them more human-like and thus easier to anticipate (Matsui et al. 2005; Gielniak et al. 2013), or by adding anticipatory motion to prepare users for the robot's upcoming action (Gielniak and Thomaz 2011; Szafir et al. 2014). Our work is complementary to these; it does not require modifying the robot's motion and would improve human anticipation even when the robot moves in a non-human-like way.

Related work explored communicating the payoff matrix in a discrete-action state-invariant game, in which the human observes the reward associated with an action once the robot executes it (Nikolaidis et al. 2017). What is challenging about our problem statement is that (1) there are multiple states, and the human needs to generalize the reward to novel state-action pairs, and (2) the human does not get to observe reward directly, but merely sees the robot act, thus all the information they get is that the action was optimal (with respect to discounted cumulative future reward).

Other methods for making robot behavior more transparent include explaining failure modes (Raman and Kress-Gazit 2013; Tellex et al. 2014), verbalizing experiences (Perera et al. 2016; Rosenthal et al. 2016), and explaining policies (Hayes and Shah 2017).

We leverage algorithmic teaching to choose the most informative examples of optimal robot behavior to show end-users. Prior work on using algorithmic teaching to teach humans primarily focuses on teaching binary classification of

images (Khan et al. 2011; Basu and Christensen 2013; Singla et al. 2014; Patil et al. 2014). In line with our work, Patil et al. (2014) show accounting for human limitations (in their case limiting the number of recalled examples) improves teaching performance.

In our algorithmic teaching framework, we model human learners as running Bayesian IRL (Chajewska et al. 2001; Ramachandran and Amir 2007): they start with a prior over objective functions that the robot may be optimizing, and update their belief with the likelihood of the observed robot trajectories given the objective function. Prior approaches for Bayesian IRL either rely on exact inference (Chajewska et al. 2001) or an action-based likelihood distribution (Ramachandran and Amir 2007). Action-based distributions have the undesirable effect of favoring trajectories with a smaller action branching factor, whereas trajectory-based distributions do not (Ziebart et al. 2008). Our work considers several trajectory-based distributions, motivated by how humans may perform approximate inference in this domain (Sect. 3.4). One of these is equivalent to that in MaxEnt IRL (Ziebart et al. 2008).

It is worth noting that many (non-Bayesian) IRL approaches are equivalent to finding the maximum a posteriori estimate in Bayesian IRL, with respect to a particular prior and likelihood function (Choi and Kim 2011). Modeling learners directly as Bayesian IRL allows us to choose informative example robot behaviors that maximize the posterior probability of the correct objective function while minimizing the probability of others.

3 Algorithmic teaching of objective functions

We model how people infer a robot's objective function from its behavior, and leverage this model to generate informative examples of behavior.

3.1 Preliminaries

Let S be the (continuous) set of states and A be the (continuous) set of actions available to the robot. We assume the robot's objective (or reward) function is represented as a linear combination of features[1] weighted by some θ^* (Abbeel and Ng 2004):

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

where s_t is the state at time t, a_t is the action taken at time t, and E is the environment (or world) description. In the case of driving, E contains information about the lanes, the trajectories of other cars, and the starting state of the robot.

A robot's trajectory ξ is defined by a sequence of states s_t and actions a_t for $t = 1, \ldots, T$. Given an environment E, the parameters θ of the objective function determine the robot's (optimal) trajectory ξ_E^θ:

$$\xi_E^\theta = \arg\max_{\xi_E \in \Xi_E} \theta^T \mu(\xi_E), \tag{2}$$

where $\mu(\xi_E) = \sum_{t=0}^{T-1} \gamma^t \phi(s_t, a_t, s_{t+1}; E)$ denotes the discounted accumulated feature vector of the trajectory, and γ is a discount factor between 0 and 1 that favors obtaining rewards earlier. Ξ_E refers to all possible trajectories in environment E.

3.2 Algorithmic teaching framework

We model the human observer as starting with a prior $P(\theta)$ over what θ^* might be, and updating their belief as they observe the robot act. We assume the human knows the features $\phi(\cdot)$ relevant to the task. (This is just our learner model for algorithmic teaching—we put this to the test with real users who do not necessarily have this understanding in Sect. 6. Further, in Sect. 8, we explore what happens when this assumption does not hold.) The robot behaves optimally with respect to the objective induced by θ^*, but as Fig. 1 shows, the details of the environment (e.g., locations of nearby cars and the robot's goal) influence its behavior, and therefore influence what effect this behavior has on the person's belief.

To best leverage this effect, we search for a sequence of environments $E_{1:n}$ such that when the person observes the optimal trajectories in those environments, their updated belief places maximum probability on the correct θ, i.e., θ^*:

$$\arg\max_{E_{1:n}} P(\theta^* | \xi^{\theta^*}_{E_{1:n}}) \tag{3}$$

To solve this optimization problem, the robot needs to model how examples update $P(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, the person's belief for the robot's true reward parameters. We propose to model $P(\theta^*|\xi^{\theta^*}_{E_{1:n}})$ via Bayesian inference:

$$P(\theta|\xi^{\theta^*}_{E_{1:n}}) \propto P(\xi^{\theta^*}_{E_{1:n}}|\theta)\, P(\theta) = P(\theta) \prod_{i=1}^{n} P(\xi^{\theta^*}_{E_i}|\theta). \tag{4}$$

Computing the normalization factor $\int_{\theta'} P(\xi^{\theta^*}_{E_{1:n}}|\theta')\, P(\theta')$ is intractable, so we approximate this by uniformly sampling candidate θs. Conditional independence can be assumed, since θ contains all the information needed to compute the probability of a trajectory.

[1] We can assume this without loss of generality, as there are no restrictions on how complex these features can be.

With this assumption, modeling how people infer the objective function parameters reduces to modeling $P(\xi|\theta)$: how probable they would find trajectory ξ if they assumed the robot optimizes the objective function induced by θ. We explore different models of this, starting with exact-inference IRL as a special case. We then introduce models that account for the inexactness that is inevitable when real people make this inference.

3.3 Exact-inference IRL as a special case

Inverse reinforcement learning (IRL) (Ng and Russell 2000) extracts an objective function from observed behavior by assuming that the observed behavior is optimizing some objective from a set of candidates. When that assumption is correct, IRL finds an objective function that assigns maximum reward (or minimum cost) to the observed behavior.

Algorithmic teaching has been applied to exact-inference IRL learners (Cakmak and Lopes 2012): the learner eliminates all objective functions which would *not* assign maximum reward to the observed behavior. This can be expressed by the model in (4) via a particular distribution for $P(\xi_E^{\theta^*}|\theta)$:

$$P(\xi_E^{\theta^*}|\theta) = \begin{cases} 1, & \text{if } \forall \xi_E, \theta^\top \mu(\xi_E^{\theta^*}) - \theta^\top \mu(\xi_E) \geq 0. \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

This assumes people assign probability 0 to trajectories that are not perfectly optimal with respect to θ, so those candidate θs receive a probability of zero. Thus, each trajectory that the person observes completely eliminates from their belief any objective function that would not have produced exactly this trajectory when optimized. Assuming learners start with a uniform prior over objective functions, the resulting belief is a uniform distribution across the remaining candidate objective functions—θs for which all observed trajectories are optimal.

While this is a natural starting point, it relies on people being able to perfectly evaluate whether a trajectory is *the* (or one of the) *global* optima of any candidate objective function. We relax this requirement in our approximate-inference models.

3.4 Approximate-inference models

We introduce a space of approximate-inference models, obtained by manipulating two factors in a 2-by-3 factorial design.

Deterministic versus probabilistic effect. In the exact-inference model, a candidate θ is either out or still in: the trajectories observed so far have either shown that θ is impossible (because they were not global optima for the objective induced by that θ), or have left it in the mix, assigning it equal probability as the other remaining θs.

We envision two ways to relax this assumption that a person can identify whether a trajectory is optimal given a θ:

One way is for observed trajectories to still either eliminate the θ or keep it in the running, but to be more conservative about which θs get eliminated. That is, even if the observed trajectory is not a global optimum for a θ, the person will not eliminate that θ if the trajectory is *close enough* (under some distance metric) to the global optimum. We call this the *deterministic* effect.

A second way is for observed trajectories to have a *probabilistic* effect on θs: rather than eliminating them completely, trajectories can make a θ less likely, depending on how far away its optimal trajectory is from the observed trajectory.

In both cases, $P(\xi_E^{\theta^*}|\theta)$ no longer depends on the example trajectory being optimal with respect to θ. Instead, it depends on the *distance* $d(\cdot,\cdot)$ between ξ_E^{θ}, the optimal trajectory for θ, and $\xi_E^{\theta^*}$, the observed trajectory which is optimal given θ^*.

Given some distance metric d along with hyperparameters $\tau, \lambda > 0$,

- For deterministic effect,
  $P(\xi_E^{\theta^*}|\theta) \propto 1$ if $d(\xi_E^{\theta}, \xi_E^{\theta^*}) \leq \tau$, or 0 otherwise.
- For probabilistic effect,
  $P(\xi_E^{\theta^*}|\theta) \propto e^{-\lambda \cdot d(\xi_E^{\theta}, \xi_E^{\theta^*})}$.[2]

The deterministic effect results in conservative hypothesis elimination: it models a user who will either completely eliminate a θ or not, but who will not eliminate θs with optimal trajectories close to the observed trajectory. In contrast, the probabilistic effect decreases the probability of θs with far away optimal trajectories, never fully eliminating any (Fig. 2).

The exact-inference IRL model (Sect. 3.3) is a special case with deterministic effect and a reward-based distance metric with $\tau = 0$; it assumes there is no approximate inference.

Distance metrics. Both deterministic and probabilistic effects rely on the person's notion of how close the optimal trajectory with respect to a candidate θ is from the observed trajectory. We envision that closeness can be measured either in terms of the reward of the trajectories with respect to θ, or in terms of the trajectories themselves.

We explore three options for d. The first depends on the reward. This distance metric models people with difficulty comparing the cumulative discounted rewards of two trajectories, with respect to a given setting of the reward parameters. So, if in environment E the observed trajectory

[2] We noticed normalizing this distribution produced very similar results to leaving it unnormalized, so we do the latter in our experiments, analogous to other algorithmic teaching work not based on reward functions (Singla et al. 2014).

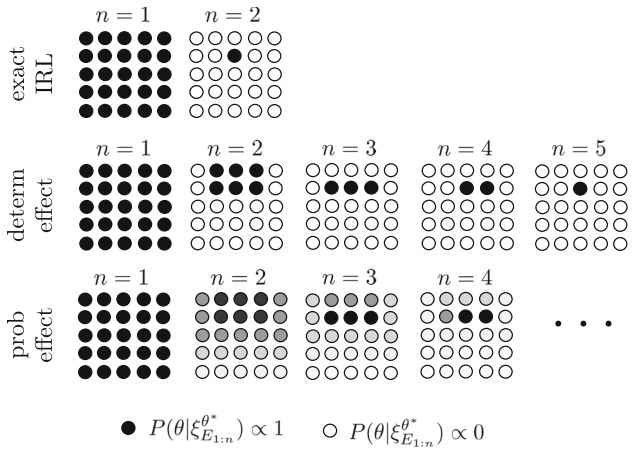

Fig. 2 A visual comparison of how probabilities of candidate reward parameters θs may be updated for exact-inference and approximate-inference learners, where each dot corresponds to a particular candidate θ. This assumes the models share the same distance metric and the same sequence of examples is shown to each model, until only the true parameters θ^* remain

$\xi_E^{\theta^*}$ has almost the same reward as ξ_E^{θ}, the optimal trajectory with respect to θ, then $P(\xi_E^{\theta^*}|\theta)$ will be high.

– Reward-based:[3] $d_r(\xi_E^{\theta}, \xi_E^{\theta^*}) = \theta^\top \mu(\xi_E^{\theta}) - \theta^\top \mu(\xi_E^{\theta^*})$.

The second option depends not on reward, but on the physical trajectories. It assumes it is not high reward that confuses people about whether the observed trajectory is optimal with respect to θ, but rather physical proximity to the true optimal trajectory: this models people who cannot perfectly distinguish between perceptually-similar trajectories. We measure physical proximity in terms of Euclidean distance, which approximately captures how humans judge perceptual similarity when stimulus dimensions are not easily separable (as is the case for object locations) (Shepard 1964; Ashby 1992).

– Euclidean-based: $d_e(\xi_E^{\theta}, \xi_E^{\theta^*}) = \frac{1}{T}\sum_{t=1}^{T} ||s_{E,t}^{\theta} - s_{E,t}^{\theta^*}||_2$. $s_{E,t}^{\theta}$ is the state at time t for trajectory ξ_E^{θ}.[4]

Finally, a more conservative version of the Euclidean distance metric is the strategy-based metric. The idea here is that for any environment E, trajectories generated by candidate θs can be clustered into types, or strategies. The strategy-based metric assumes people do not distinguish among trajectories that follow the same strategy. For instance, people will consider all trajectories in which the robot speeds up and merges in front of another car to be equivalent, and all trajectories in which the robot merges behind the car to be equivalent. So, if in environment E the observed trajectory and the optimal trajectory with respect to θ have the same strategy, then $P(\xi_E^{\theta^*}|\theta) \propto 1$.

– Strategy-based: $d_s(\xi_E^{\theta}, \xi_E^{\theta^*}) = 0$ if ξ_E^{θ} and $\xi_E^{\theta^*}$ are in the same trajectory strategy cluster, ∞ otherwise.

Note that the type of effect (deterministic versus probabilistic) does not matter for the strategy-based distance metric, since distances are either 0 or ∞. So, there are a total of five unique approximate-inference models.

Relation to MaxEnt IRL. MaxEnt IRL (Ziebart et al. 2008) is an IRL algorithm that assumes demonstrations are noisy (i.e., not necessarily optimal). In our setting, we instead assume demonstrations are optimal but the learner is approximate. These two sources of noise result in the same model: the MaxEnt distribution is equivalent to our *probabilistic reward-based* model:

$$P(\xi_E^{\theta^*}|\theta) \propto e^{\lambda\theta^T \mu(\xi_E^{\theta^*})} \tag{6}$$

$$\propto e^{\lambda(\theta^T \mu(\xi_E^{\theta^*}) - \theta^T \mu(\xi_E^{\theta}))} = e^{-\lambda \cdot d_r(\xi_E^{\theta}, \xi_E^{\theta^*})}. \tag{7}$$

3.5 (Submodular) example selection

Given a learner model $\mathcal{M}$ that predicts $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, our approach greedily selects the environment E_t that maximizes $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:t}})$. We allow the model to select up to ten examples; it stops early if no additional example improves this probability. Algorithm 1 outlines this approach.

This greedy approach is near-optimal for deterministic effect with a uniform prior, since this makes maximizing $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:t}})$ equivalent to maximizing a non-decreasing monotonic submodular function (Nemhauser et al. 1978): the total number of candidate θs that are eliminated after observing $\xi^{\theta^*}_{E_{1:t}}$. These two are equivalent because all non-eliminated θs are equally likely (Fig. 2). Recall that when deterministic-effect learners observe an example trajectory, they eliminate all candidate θs for which the observed trajectory is not close enough to the optimal trajectory under that θ. The total number of eliminated candidate θs is non-decreasing because adding example trajectories $\xi_E^{\theta^*}$ can only eliminate candidate θs, not add them, and we assume the set of candidate θs considered by the human does not change over time. Additionally, a particular observed trajectory $\xi_E^{\theta^*}$ eliminates the same set of θs no matter when it is added to the sequence. Thus, showing that example later on in the

[3] Note that this is always positive because ξ_E^{θ} has maximal reward with respect to θ.

[4] This assumes the two trajectories are the same length, as is the case in our example domain. When trajectories are not the same length, they must first be aligned temporally, for instance via dynamic time warping (Sakoe and Chiba 1978). In addition, this requires an appropriate representation of the state space, e.g., if the dimensions of $s_{E,t}^{\theta}$ have different ranges, normalization may be necessary. In our example domain, the state is the robot's two-dimensional location.

sequence cannot eliminate more θs than adding it earlier, which makes this function submodular.

Algorithm 1 Informative example selection

Require: Set of possible environments, $\mathcal{E}$
Require: Robot's reward parameters, θ^*
Require: Set of candidate reward parameters, Θ
Require: Prior over reward parameters, $P_{\mathcal{M}}(\theta)$
Require: Number of examples to select, n
$\{\hat{P}(\theta)$ keeps track of the learner's belief over reward parameters$\}$
Initialize $\hat{P}(\theta) \leftarrow P_{\mathcal{M}}(\theta)$
Initialize $X = []$
for $i = 1$ **to** n **do**
  for all $E \in \mathcal{E}$ **do**
    $\xi_E^{\theta^*} \leftarrow \arg\max_{\xi_E \in \Xi_E} \theta^{*T} \mu(\xi_E)$
    $Z \leftarrow \sum_\theta P_{\mathcal{M}}(\xi_E^{\theta^*}|\theta)\, \hat{P}(\theta)$
    $P_{\mathcal{M}}(\theta^*|\xi_{X+[E]}^{\theta^*}) = \frac{1}{Z} P_{\mathcal{M}}(\xi_E^{\theta^*}|\theta^*)\, \hat{P}(\theta^*)$
  end for
  $E_i \leftarrow \arg\max_E P_{\mathcal{M}}(\theta^*|\xi_{X+[E]}^{\theta^*})$
  $X \leftarrow X + [E_i]$
  $\hat{P}(\theta) \leftarrow \hat{P}(\theta)\, P_{\mathcal{M}}(\xi_E^{\theta^*}|\theta)$
end for
return Sequence of informative examples X

3.6 Hyperparameter selection

We would like to select values for hyperparameters τ and λ (for deterministic and probabilistic effect, respectively) that accurately model human learning in this domain. τ and λ affect the informativeness of examples. If τ is too large, then most environments will be uninformative, since the observed trajectory will be within τ distance away from optimal trajectories of many θs, so those θs will not be eliminated. Thus, $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$ will be low no matter which examples $\xi_{E_{1:n}}^{\theta^*}$ are selected. On the other hand, if τ is too small, then some environments will be extremely informative, so only one or a few examples will be selected before no further improvement in $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$ can be achieved. Analogous reasoning holds for λ.

We expect humans to be teachable (i.e., τ cannot be too large) and to have approximate rather than exact inference (i.e., τ cannot be too small), so they would benefit from observing several examples rather than just one or two. Based on this, we select τ and λ for each approximate-inference model by choosing the value in $\{10^{-5}, 10^{-4}, \ldots, 10^4, 10^5\}$ that results in an increase from $P_{\mathcal{M}}(\theta^*|\xi_{E_1}^{\theta^*})$ to $P_{\mathcal{M}}(\theta^*|\xi_{E_{1:n}}^{\theta^*})$ of at least 0.1, and selects the largest number of unique examples to show.

4 Example domain

We evaluate how our proposed approximate-inference models perform for teaching the driving style of a simulated autonomous car. In this domain, participants witness examples (in simulation) of how the car drives, with the goal of being able to anticipate how it will drive when they ride in it.

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	[20, 25, …, 80]
Acceleration time, non-auton.	[0, 0.5, 1, 1.5, 2]
Final velocity, non-auton. car (if acceleration time $\neq 0$)	[20, 30, 70, 80]

4.1 Driving simulator

We model the dynamics of the car with the bicycle vehicle model (Taheri and Law 1990). Let the state of the 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. Let L be the distance between the front and rear axles of the car. Then the dynamics model of the vehicle is

$$[\dot{x}\ \dot{y}\ \dot{\theta}\ \dot{v}\ \dot{\alpha}] = [v * \cos(\theta)\ \ v * \sin(\theta)\ \ \frac{v}{L}\tan(\alpha)\ \ v * u_1\ \ u_2] \quad (8)$$

4.2 Environments

We consider a total of 21,216 environments of highway driving configurations (Table 1). Each environment has three lanes and a single non-autonomous car. The autonomous car always starts in the middle lane with the same initial velocity, whereas the initial location and velocity of the non-autonomous car varies.

These environments naturally fall into four classes, with two trajectory strategies per class (Fig. 3):

– *Merging*: When the non-autonomous car starts in the right lane, and the goal in this environment is to merge into the right lane. The two trajectory strategies are to either speed

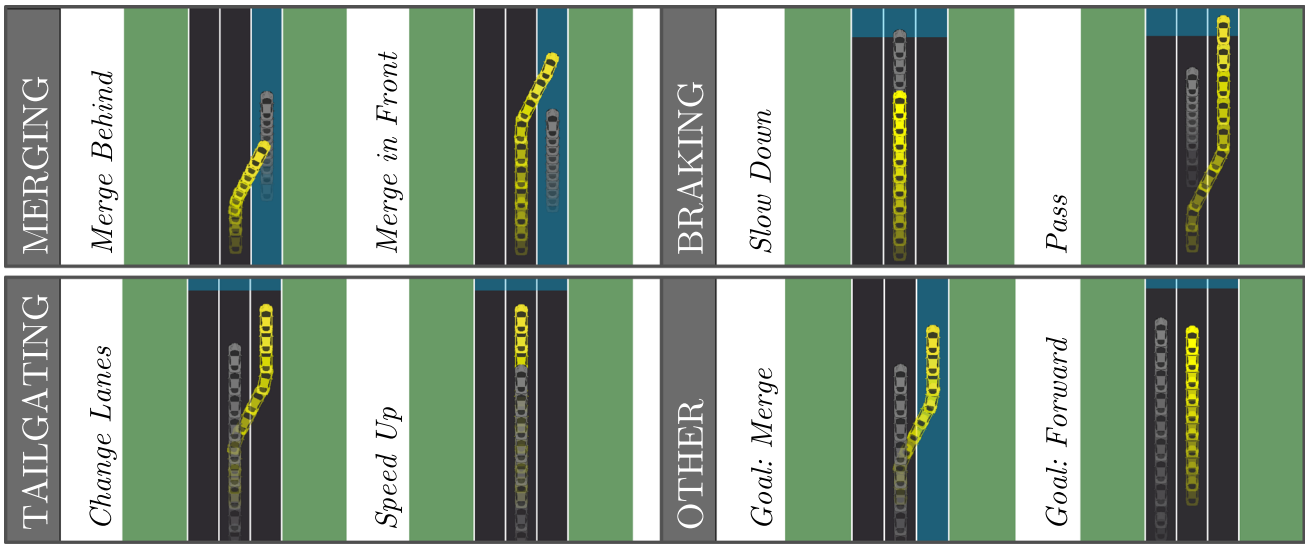

Fig. 3 The possible driving environments cluster naturally into four classes, with two trajectory strategies per class. Each image shows the trajectories of the autonomous car (yellow) and non-autonomous car (gray) in a particular environment. Positions later in the trajectory are more opaque. The goal of the autonomous car in each environment is highlighted in blue: merge into the right lane or drive forward (Color figure online)

up and merge ahead of the non-autonomous car, or slow down and merge behind the non-autonomous car.
- *Braking*: When the non-autonomous car starts in the center lane in front of the autonomous car, and the goal is to drive forward. The two trajectory strategies are to either keep driving in the center lane behind the non-autonomous car, or merge into another lane to pass it.
- *Tailgating*: When the non-autonomous car starts in the center lane behind the autonomous car, and the goal is to drive forward. The two trajectory strategies are to either change lanes to avoid the tailgater, or speed up to maintain a safe distance from the tailgater.
- *Other*: All environments not included in one of the first three. The autonomous car is able to reach its goal without any interaction with the other car.

The majority of environments (14,144) are in the *other* class. Of the remaining environments, 3536 are in the *merging* class, 1768 are in *tailgating*, and 1768 are in *braking*.

4.3 Reward features

We use the following reward features $\phi(\cdot)$:

- *Distance to other car*: $\sum_{t=0}^{T} \gamma^t e^{-\frac{1}{2}(p_t - p'_t)^\top \Sigma_t^{-1}(p_t - p'_t)}$. Each term corresponds to a multivariate Gaussian kernel. $p_t = [x_t, y_t]^\top$, p'_t is the position of the non-autonomous car, and Σ_t is chosen so that the major axis is along the non-autonomous car's heading.
- *Acceleration*, squared: $\sum_{t=0}^{T-1} \gamma^t (v_{t+1} - v_t)^2$
- *Deviation from initial speed*, squared: $\sum_{t=0}^{T} \gamma^t (v_t - v_1)^2$
- *Turning*: $\sum_{t=0}^{T} \gamma^t |\theta_t - \theta_0|$
- *Distance from goal*: $\sum_{t=0}^{T} \gamma^t \max(0, (x_1 + w) - x_t)^2$ if the goal is to merge into the right lane, and y_T 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 each feature across all trajectories is 1 and the minimum is 0. We use $\gamma = 1$.

4.4 Optimal θ

We select $\theta^* = [-64 \ -0.1 \ -1 \ -0.1 \ -0.5]^\top$, a reward function that is not overly cautious about staying away from other cars. We uniformly sample 1000 candidate θs, and assume learners start with a uniform prior $P(\theta)$.

5 Analysis of approximate inference models with ideal users

In Sect. 3.4, we proposed five possible approximate-inference user models $\mathcal{M}$. They all model people as judging candidate θs based on the distance between the trajectory they observed and the optimal trajectory with respect to θ, but they differ in what the distance metric is, and whether they completely eliminate candidate θs (deterministic effect) or smoothly reweight them (probabilistic effect).

Here, we investigate how well algorithmic teaching with these models performs for teaching θ^* to ideal users. First, we

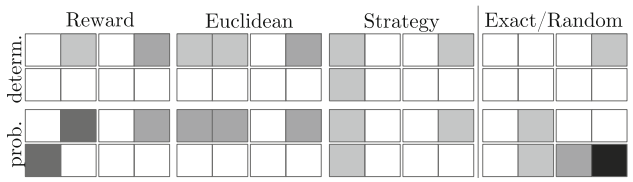

Fig. 4 The number of examples shown in each of the eight trajectory classes for the approximate-inference models, exact inference model, and random baseline. White = 0 examples shown, and black = 4. The environment classes are arranged in the 2×4 grid as in Fig. 3

generate a sequence of examples for each of our approximate-inference models $\mathcal{M}$, by greedily maximizing $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, as described in Sect. 3.5. We also generate the sequence for the exact-inference model and include a random sequence, for a total of seven sequences.

Run-time. For each of the approximate-inference models, it takes less than a minute to generate this sequence of examples. This includes hyperparameter selection of τ and λ (Sect. 3.6). Since we have a fixed set of environments $\mathcal{E}$ and candidate reward parameters Θ, we pre-computed the optimal trajectories and distances between pairs of optimal trajectories $\xi^{\theta^*}_E$ and ξ^{θ}_E for all $E \in \mathcal{E}$ and $\theta \in \Theta$, to speed up example selection.

Types of examples selected. Figure 4 summarizes the types of examples that each algorithm selected for its optimal sequence. The exact-inference model selects a single example, because that is enough to completely eliminate all other θs. This works well for an ideal user running exact inference, but our hypothesis is that it does not work as well for real users.

Evaluation with ideal users. We evaluate algorithmic teaching on six ideal users, whose learning is precisely exact-inference IRL or one of our five approximate-inference models. We measure, for each "user" $\mathcal{M}$, the probability they assign to the correct objective function parameters, $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, given $\xi^{\theta^*}_{E_{1:n}}$ from each of the seven generated sequences.

Figure 5 shows the results. First, we see for any ideal user $\mathcal{M}$, the sequence generated by assuming a learner model $\mathcal{M}$ performs best at teaching that user. This is by design—that sequence of examples is optimized to teach $\mathcal{M}$.

Looking across the columns of Fig. 5, we see all seven sequences perform equally well for teaching an exact IRL learner (column 1)—even random, because the examples it provides are enough to perfectly eliminate all incorrect θs. This suggests exact IRL does not accurately model real users, whose performance likely varies based on which examples they see. Looking across the rows, we notice assuming a Euclidean distance approximation when generating examples (rows 3 and 5) leads to robust performance across different user models. In the following section, Sect. 6, we evaluate these generated sequences on real users.

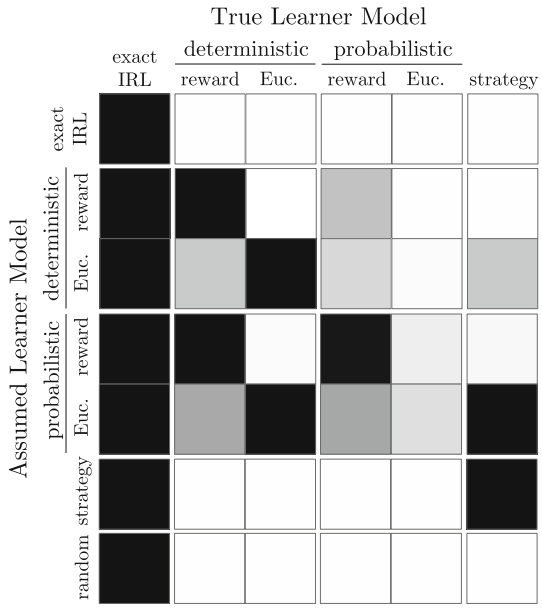

Fig. 5 The performance of each user model as evaluated by all other models. White indicates $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}}) \approx 0$, where $\mathcal{M}$ is the true learner model and environments $E_{1:n}$ are chosen based on the assumed learner model. Black indicates $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}}) = 1$

Finally, the random sequence is very uninformative for all ideal users except exact IRL, showing the utility of algorithmic teaching. We explore this utility with real users in Sect. 7.

6 User study

We now evaluate whether approximate-inference models are useful with real, as opposed to ideal, users.

6.1 Experiment design

Manipulated variables. We manipulate whether algorithmic teaching assumes exact- or approximate-inference. For the approximate-inference case, we manipulated two variables: the effect of approximate inference (either *deterministic* or *probabilistic*) and the distance metric (*reward-based*, *Euclidean-based*, or *strategy-based*), in a 2-by-3 factorial design, for a total of six approximate inference models. Recall that since the type of effect does not matter for the strategy-based distance metric, there are five unique approximate-inference models.

We show the participant one training environment at a time, in the order that the examples were selected by each algorithm.

Dependent measures. In the end, we are interested in how well human participants learn a specific setting of reward parameters θ^* from the training examples. Since we cannot

ask them to write down a θ, or to drive according to how they think the car will drive (people can drive like themselves, but not so easily like others), we evaluate this by testing each participant's ability to identify the trajectory produced by θ^* in a few test environments. For each test environment, we show the participant four trajectories and ask them to select the one that most closely matches the autonomous car's driving style, and report their confidence (from 1 to 7) for how closely each of the four trajectories matches the driving style.

We have two dependent variables: whether participants correctly identify $\xi_{E_{\text{test}}}^{\theta^*}$ for each test environment E_{test}, and their confidence in selecting that trajectory. We combine the two in a confidence score: the confidence if they are correct, negative of the confidence if they are not—this score captures that if one is incorrect, it is better to be not confident about it.

We use rejection sampling to select test environments in which there are a wider variety of possible robot trajectories. To make sure the four trajectories do not look too similar, we ensure the rewards of alternate trajectories under θ^* are below a certain threshold. In order to not bias the measure, we select one test environment for each of the two trajectory strategy clusters in each of the three informative environment classes, for a total of six test environments. For each test environment, we show two trajectory options in each strategy cluster.

Hypothesis. Accounting for approximate inference significantly improves performance (as measured by the confidence score). We leave open which approximate-inference models work well and which do not, since the goal is to identify which captures users' inferences the best.

Subject allocation. We used a between-subjects design, since examples of the same reward function interfere with each other. We ran this experiment on a total of 191 participants across the six conditions, recruited via Amazon Mechanical Turk. At the end of the experiment, we ask participants what the two possible goals were, to filter out those who were not paying attention. 30 out of 191 (15.7%) answered incorrectly. The average age of the 161 non-filtered participants was 37.0 ($SD = 11.0$). The gender ratio was 0.46 female.

6.2 Analysis

Number of examples. The number of examples shown depends on which user model is assumed. Exact-inference IRL may produce as few as one example (and does in our case). Approximate-inference models produce more, and random can produce an almost unlimited amount if allowed. Thus, a possible confound in our experiment is the number of examples.

We checked whether this is indeed a confound: do more examples help? Surprisingly, we found no correlation between the number of examples and performance: the sample Pearson correlation coefficient is $r = 0.03$ (Fig. 7, left).

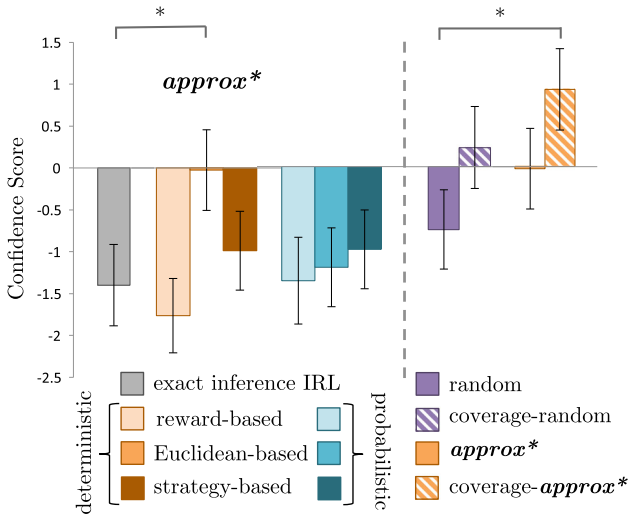

Fig. 6 Performance of human participants on identifying the autonomous car's trajectory in test environments, after seeing the example trajectories selected by the approximate-inference models (left) and after adding coverage (right) to the sequence of environments selected by the best-performing approximate-inference model, *approx**. Participants in the coverage-*approx** condition performed significantly better than those in the random condition. In contrast, enforcing coverage but selecting random sequences (coverage-random) does not lead to statistically significantly better performance

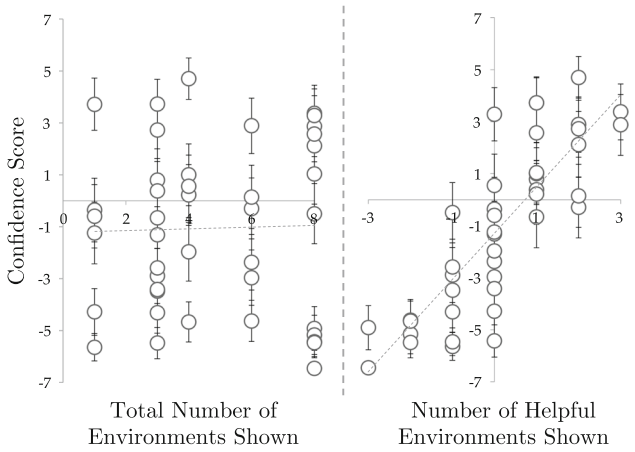

Fig. 7 Left: lack of correlation between total number of environments shown in a condition, and average participant performance on each test example in that condition. Right: correlation exists between the number of helpful training examples shown for an environment class, and average participant performance on the test example corresponding to that environment class

This suggests that example quantity matters less than example quality.

Approximate-inference models. We begin by comparing the different approximate-inference models. We ran a factorial ANOVA on *confidence score* with distance measure and determinism as factors (Fig. 6, left). We found a marginal effect for *distance* ($F(2, 163) = 2.69$, $p = .07$), with Euclidean-based distance performing the best (as suggested

by our experiment in Sect. 5, where Euclidean was the most robust across different ideal users), and reward-based distance performing the worst.

Euclidean-based might be better than reward-based because people decide to keep imperfect θs not when the trajectory they see obtains high reward under that θ, but when it is visually similar to what optimizing for θ would have produced. Euclidean-based might be better than strategy-based because people differentiate between trajectories even when they follow the same strategy. For example, a trajectory that gets very close to another car would be in the same strategy class as a trajectory that stays farther away, as long as they both merge behind the other car, but these two trajectories may give people very different impressions of the car's driving style.

There was no effect for determinism. On average, probabilistic models performed ever-so-slightly worse than deterministic ones, and the difference was largest for Euclidean distance. This might be because keeping track of what is possible is easier than maintaining an entire probability distribution.

The best-performing approximate-inference model used the Euclidean-based distance with deterministic effects. We refer to this as the *approx** model (Fig. 6, left).

Hypothesis: utility of approximate inference. We found that despite showing more examples, most approximate-inference models did not perform much better than exact-inference IRL. This shows that not just any approximate-inference model is useful. However, it does not imply that *no* approximate-inference model is useful. To test the utility of accounting for approximate inference, we compared the best model, *approx**, with exact-inference IRL, and found a significant improvement (Welch's t-test $p = 0.025$).

> *This supports our central hypothesis, that accounting for approximate inference in our model of human inferences about objective functions helps, with the caveat that not just any approximation will work.*

7 Utility of algorithmic teaching

So far, we have tested our central hypothesis, that accounting for approximate IRL inferences can indeed improve the performance of algorithmic teaching of humans in this domain. While this is promising, we also want to test the utility of algorithmic teaching itself: whether our approach is preferable not just to algorithmic teaching with exact inference, but to the robot not actively teaching. Instead, the person must learn from the robot's behavior in environments that it happens to encounter.

7.1 Baselining performance

Baseline condition. We ran a follow-up study comparing algorithmic teaching with a sequence of optimal trajectories in *random* environments—simulating that the robot does not choose these, but instead happens to encounter them. We recruited 33 users for this condition. The average age of the 28 non-filtered participants was 33.3 ($SD = 9.6$). The gender ratio was 0.46 female.

Controlling for confounds. There are several variables that could confound this study. First, when generating the random sequence, we might get very lucky or unlucky and generate a particularly informative or uninformative one. To avoid this, we randomly sample 1000 random sequences with the desired number of examples, and sort them based on $P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}})$, where $\mathcal{M}$ is the exact-inference IRL model. Then we choose the median sequence in that ranking as our random sequence, which will have median informativeness.

Second, different algorithms produce different numbers of examples. For instance, exact-inference IRL only selects one example, which eliminates all θs other than θ^*—because in that environment the optimal trajectories for all θs are at least slightly different than that for θ^*. To give the random baseline the best chance, we choose to select eight environments for it, which is the maximum number of examples shown by any of the other conditions. Since the majority of environments are uninformative (i.e., not in the merging, braking, or tailgating classes), providing the random condition with eight environments is needed to not put it at a serious disadvantage.

Analysis. Algorithmic teaching with our approximate inference model did outperform the random baseline, albeit not significantly (Welch's t-test $p = 0.23$ when comparing participants' confidence scores). Algorithmic teaching *without* accounting for approximate inference actually seems to perform poorly compared to random (Fig. 6, right).

Coverage. Digging deeper, we realized users tended to perform well on test cases for strategies in which they had seen a training example. In addition, for each pair of environment strategies A and B (e.g., merge-in-front and merge-behind), if users did not see an example from strategy A but saw one from strategy B, their performance was worse than if they did not see any examples from either A or B! In other words, if users 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.

Based on this observation, for a particular trajectory strategy A and training environments $E_{1:n}$, we define the number of *helpful (training) environments* for A as:

$$\begin{cases} \sum_{i=1}^{n} \mathbb{1}[h(\xi^{\theta^*}_{E_i}) = A], & \text{if } \sum_{i=1}^{n} \mathbb{1}[h(\xi^{\theta^*}_{E_i}) = A] > 0. \\ -\sum_{i=1}^{n} \mathbb{1}[h(\xi^{\theta^*}_{E_i}) = B], & \text{otherwise.} \end{cases} \tag{9}$$

The function h maps a trajectory to the strategy it belongs to, and B is the opposite strategy of A. For instance, if A is the speed-up strategy, then B is the other strategy for the *tailgating* environment, change-lanes. We found a strong correlation between the number of helpful environments shown and users' confidence scores, with a Pearson correlation coefficient of $r = 0.83$ ($p = 1.4 \times 10^{-11}$) (Fig. 7, right).

Motivated by this result, we introduce augmented algorithms that ensure coverage of strategies.

7.2 Coverage-augmented algorithmic teaching

Since coverage correlates with better user performance, we add a coverage term to our optimization over trajectories $\xi^{\theta^*}_{E_{1:n}}$:

$$\arg\max_{\xi^{\theta^*}_{E_{1:n}}} P_{\mathcal{M}}(\theta^*|\xi^{\theta^*}_{E_{1:n}}) + \nu \sum_{c} \mathbb{1}[\exists i, h(\xi^{\theta^*}_{E_i}) = c], \tag{10}$$

where the sum is over trajectory strategy clusters c.

When greedily selecting the next environment E_t that maximizes Eq. (10), we set

$$\nu = \mathbb{1}[P_{\mathcal{M}}(\theta^*|\xi^{\theta*}_{E_{1:t}}) - P_{\mathcal{M}}(\theta^*|\xi^{\theta*}_{E_{1:t-1}}) < \epsilon],$$

so that only after no examples will significantly increase the probability of θ^*, extra examples are selected to provide coverage across the strategies. We select these extra examples by choosing the best with respect to the approximate-inference model $\mathcal{M}$, to ensure they are informative. Using this approach, we augment our best approximate-inference model, *approx**, to achieve coverage of all possible strategies.

7.3 User study on coverage

We next run a study to test the benefit of coverage.

Manipulated variables. We manipulate two variables: whether we augmented the training examples with coverage, and whether we used a user model to generate the examples or sampled uniformly. We select our best model for the former, *approx**. From the previous experiment, we have obtained user performance data along the no-coverage dimension—for random and *approx**—so we run this experiment on only the two new conditions that incorporate coverage: coverage-random and coverage-*approx**. We generate random sequences with coverage by randomly selecting exactly one random environment from each of the eight trajectory strategy classes.

Dependent measures. We keep the same dependent measures as in our previous user study (Sect. 6.1).

Hypothesis. We hypothesize coverage augmentation improves user performance in both conditions, random and *approx**, compared to the respective conditions without coverage.

Subject allocation. We used a between-subjects design, with a total of 63 participants across the two conditions. The average age of the 53 non-filtered participants was 34.43 ($SD = 9.0$). The gender ratio was 0.53 female.

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.

Coverage-*approx** performed best out of the four conditions. The coverage augmentation enabled it to significantly outperform the random baseline (with a Welch's t-test $p = 0.049$), which suggests coverage is useful. Coverage augmentation did not enable the coverage-random condition to outperform the random baseline ($p = 0.159$), which suggests the approximate-inference model is useful (Fig. 6, right).

Overall, coverage alone helped, but was not sufficient to outperform the random baseline. From the previous experiment, we know that the approximate-inference user model helped, but was also not sufficient to outperform this baseline. The improvement is largest (and significant, modulo compensating for multiple hypotheses) when we have a coverage-augmented approximate-inference IRL model.

> *When leveraged together, coverage with the right model of approximate inference have a significant teaching advantage over random teaching, as well as over IRL models that assume exact-inference users.*

8 Analysis of alternative learner models

Algorithmic teaching relies on having a reasonably accurate model of how the learner learns. For instance, we found that when accounting for approximate inference in the learner model, most approximations did not perform significantly better than the exact-inference IRL baseline, although the best model did (Sect. 6). In this section, we will reconsider several key assumptions that our approach makes about human learning of objective functions, and analyze how teaching effectiveness is affected when they do not hold. These assumptions are that human end-users (1) pay attention to exactly the same set of features that define the robot's objective function, (2) adhere to a particular setting of hyperparameters for approximate inference, and (3) use only their understanding of the robot's objective function to anticipate its behavior.

8.1 Feature mismatch

Recall that we model human end-users as updating their belief over candidate θs as they observe the robot act (Sect. 3.2). Each candidate θ corresponds to a particular setting of trade-offs between the reward features $\phi(\cdot)$ considered by the robot. The set of candidates includes θ^*, the true parameters for the objective function optimized by the robot.

But what if end-users do not pay attention to exactly the same reward features $\phi(\cdot)$ as the robot does? In the context of our autonomous driving example domain, perhaps they do not notice sudden braking (the *acceleration* feature) or unnecessary lane changes (the *turning* feature). To determine how much of an impact feature mismatch has, we analyze how well algorithmic teaching performs for teaching θ^* to ideal end-users, that reason over either a subset or superset of the features that define the robot's objective function.

Note that when users reason over a different set of reward features $\hat{\phi}(\cdot)$, the space of candidate $\hat{\theta}$s no longer includes the true parameters θ^*. So, the best we can do is to teach users the true trade-offs between the reward features in common for the users' and robot's objective functions. Let $\hat{\Theta}^*$ denote the set of all correct candidates, that have the same tradeoffs as θ^* for these shared reward features. As in Sect. 5, we evaluate teaching performance by measuring, for each "user" $\mathcal{M}$, the probability they assign to the correct objective function parameters ($\max_{\hat{\theta}^* \in \hat{\Theta}^*} P_{\mathcal{M}}(\hat{\theta}^* | \xi^{\theta^*}_{E_{1:n}})$), given $\xi^{\theta^*}_{E_{1:n}}$ from each of the generated sequences of examples.[5] Unlike in Sect. 5, these ideal users no longer correspond to one of the models used to select examples.

Missing features. We first greedily maximize $P_{\mathcal{M}}(\theta^* | \xi^{\theta^*}_{E_{1:n}})$ to generate a sequence of teaching examples for each of our five approximate-inference models $\mathcal{M}$ and the exact-inference model, as well as generate a random sequence (same as those in Sect. 5). We then take our original set of five features (Sect. 4.3) and remove either the acceleration or turning feature, resulting in two subsets of features that our ideal users could reason over. We always keep the features for distance to other car, distance from goal, and deviation from initial speed (i.e., the speed limit) since these are fundamental to the driving task. There is a single true reward parameter $\hat{\theta}^*$ that matches the trade-offs of θ^* in the subset of features considered by the user.

We found that when users only pay attention to a subset of features, teaching is much less effective (Fig. 8a). This may be because our original set of five features does not contain any redundant ones. So, for many environments, the optimal trajectory for θ^* is different than that for $\hat{\theta}^*$.[6]

Consequently, we see that now almost no sequence of examples is able to teach the exact-inference IRL learner. Since for many environments E, the observed trajectory $\xi^{\theta^*}_E$ is not optimal for $\hat{\theta}^*$, this leads to $P(\xi^{\theta^*}_E | \hat{\theta}^*) = 0$ for exact-inference IRL learners and the true reward parameter $\hat{\theta}^*$ is then eliminated. For similar reasons, it is no longer the case that for any ideal user model $\mathcal{M}$, the sequence generated by assuming that learner model performs best at teaching that user.

Out of all the models, it seems deterministic Euclidean is slightly more robust for teaching users who reason over a subset of features. In particular, when (ideal) users expect that the robot does not consider the acceleration feature (e.g., gradually braking and slamming on the brakes are equally fine), the results are similar to those in our user study (Sect. 6): the sequence of examples generated by assuming deterministic, Euclidean-based approximate-inference is most effective at teaching (Fig. 8a, top, row 3). This suggests that the real users in our study may have overlooked acceleration of the autonomous car—this is reasonable, since they were viewing the car's trajectory as a video; acceleration would have been much more apparent if users sat in a realistic driving simulator with force feedback.

Additional features. To measure the effect of users reasoning over a superset of the features that determine the robot's objective function, we alter the robot's objective function to be a linear combination over four of the original five reward features $\phi(\cdot)$, ignoring either the acceleration or turning feature. For each of these two subsets of features, we recompute a sequence of teaching examples for each of the approximate-inference models $\mathcal{M}$ and the exact-inference model by greedily maximizing $P_{\mathcal{M}}(\theta^* | \xi^{\theta^*}_{E_{1:n}})$, as well as generate a random sequence.

Our ideal users reason over the full set of features. Since the robot's goal is to communicate the tradeoffs between only the features that it is paying attention to, there is no restriction on the reward weights corresponding to features that the robot is *not* paying attention to. So, there are multiple correct reward candidates $\hat{\theta}^*$ considered by the user, that match the trade-offs of θ^* in the subset of features considered by the robot.

We found that when users pay attention to a superset of features, teaching is as effective as when users pay attention to the exact set of features that the robot does (Figs. 8b, 5,

[5] We take the maximum probability assigned to a correct candidate because we would be happy with the user learning any of these.

[6] When $\hat{\theta}^*$ ignores the acceleration feature, the optimal trajectory differs in 50% of the environments selected as teaching examples, and the strategy of the optimal trajectory differs in 21%. When $\hat{\theta}^*$ ignores the turning feature, the optimal trajectory differs in 50% and the strategy differs in 14%. (Two trajectories are the same if they consist of the same sequence of states.)

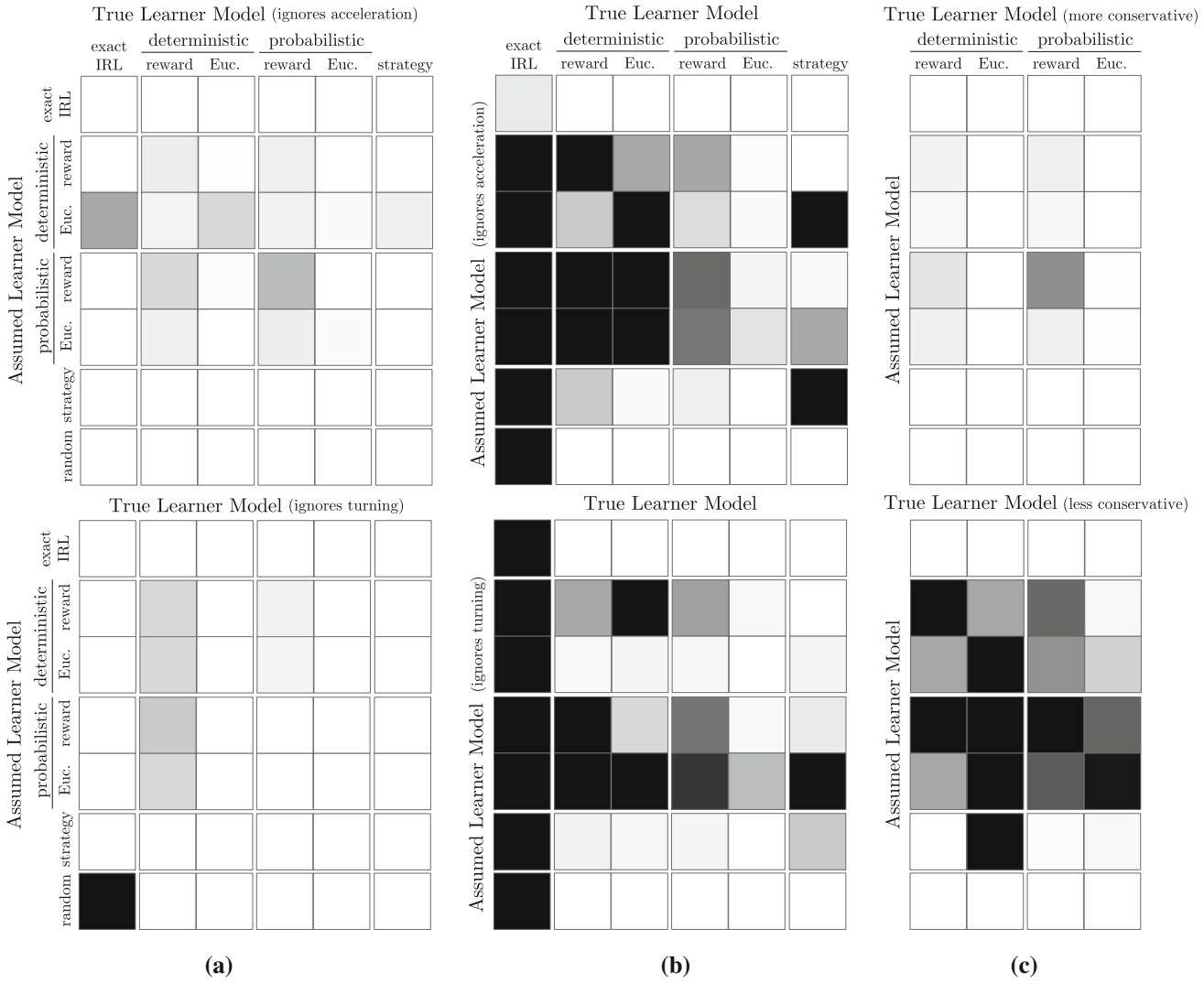

Fig. 8 The performance of each user model, as evaluated by all other models. The assumed and true learner models differ in terms of either features considered or hyperparameters. White indicates $P_{\mathcal{M}}(\hat{\theta}^*|\xi^{\theta^*}_{E_{1:n}}) \approx 0$, where $\mathcal{M}$ is the true learner model and environments $E_{1:n}$ are chosen based on the assumed learner model. Black indicates $P_{\mathcal{M}}(\hat{\theta}^*|\xi^{\theta^*}_{E_{1:n}}) = 1$. **a** The assumed learner model considers all (five) features $\phi(\cdot)$, but the true learner model ignores one of them. **b** The true learner model considers all features, but the assumed learner model ignores one of them. **c** Both the assumed and true learner models consider all features. However, the true learner model is either more or less conservative in the elimination or downweighting of candidate θs than the corresponding assumed learner model

respectively). This may be because in our problem setup, we are equally happy with the user learning any of the correct candidates in $\hat{\Theta}^*$, which may not be the case in practice.

8.2 Approximate-inference hyperparameter mismatch

Recall that our proposed approximate-inference models require setting a hyperparameter: τ for deterministic effect, λ for probabilistic effect. This hyperparameter controls how conservative the learner is in eliminating or downweighting candidate θs (Sect. 3.6). More conservative means that learners eliminate fewer θs or downweight them by less, which corresponds to a larger τ and a smaller λ. Less conservative means the opposite, and corresponds to a smaller τ or a larger λ. We originally chose the hyperparameter for each approximate-inference model $\mathcal{M}$ with a heuristic, to be similar to how humans might learn (Sect. 3.6).

We consider teaching ideal users that are either more or less conservative, by a factor of two for τ and λ, in how they eliminate or downweight candidate reward parameters. The teaching sequences are the same as in Sect. 5. As one might expect, when users are more conservative in eliminating or downweighting θs, they become less teachable: even after observing multiple example trajectories, they do not prune

their space of candidate θs by much, so $P(\theta^*|\xi^{\theta^*}_{E_{1:n}})$ is close to zero across all models and teaching sequences (Fig. 8c, top). On the other hand, when users are less conservative, they tend to perform well across all teaching sequences computed for approximate-inference models (with a reward- or Euclidean-based distance metric). However, these ideal users are still unable to learn from a random sequence or the single example computed to teach exact-inference IRL learners (Fig. 8c, bottom).

8.3 Hybrid models

A key assumption underlying our approach is that human end-users use their understanding of the robot's objective function to anticipate the robot's behavior. This understanding is what enables them to generalize and anticipate the robot's behavior in *novel* scenarios, that are different than those they have seen the robot act in previously (e.g., the test environments in our user studies).

However, the importance of coverage implies that users are unable to generalize perfectly, since they need to see all possible strategies of robot behavior. This suggests something else is going on: users may also be learning the robot's policy directly (analogous to robots using imitation learning to learn from human demonstrations), or they may be learning objective functions that are complex and environment-dependent, and therefore difficult to generalize across test cases. Alternatively, people may use hierarchical reasoning, in which they first determine which trajectory strategy the robot will take (for which they need examples of each possible strategy), and then select the most likely trajectory within that strategy cluster.

Along the lines of direct policy learning, we explore the possibility that users are learning a policy that maps from environment to the trajectory taken by the robot, by relying on a nearest neighbor approach: directly comparing each novel environment with the ones they have seen the robot act in, and making predictions based on that. This is motivated by the observation that humans seem to use such a nearest-neighbor, or *exemplar*-based, approach in other domains, in particular category learning (Medin and Schaffer 1978; Nosofsky 1986).

We hypothesize that a hybrid model, that combines approximate inference with exemplar-based reasoning, will better capture how humans learn in this domain. We will first introduce a framework for combining these two types of learning, and then evaluate how well a hybrid model correlates with responses from our user studies.

Instead of modeling people as solely performing inference over objective function parameters, we can model them as computing a distribution over possible trajectories for a new environment E_t:

$$P(\xi_{E_t}|\xi^{\theta^*}_{E_{1:n}}).$$

If humans compute this distribution by only reasoning over objective function parameters (as we have assumed up until now), then we have

$$P_{\mathcal{M}}(\xi_{E_t}|\xi^{\theta^*}_{E_{1:n}}) \propto \mathbb{E}_{\theta\sim P_{\mathcal{M}}(\theta|\xi^{\theta^*}_{E_{1:n}})} P(\xi_{E_t}|\theta) \tag{11}$$

In contrast, if humans compute this distribution through exemplar-based reasoning, then we have

$$P(\xi_{E_t}|\xi^{\theta^*}_{E_{1:n}}) \propto \sum_{i=1}^{n} g_E(E_t, E_i) g_T(\xi_{E_t}, \xi^{\theta^*}_{E_i}) \tag{12}$$

where g_E and g_T are similarity metrics between environments and robot trajectories, respectively.

One way to combine these two models into a hybrid model is to take the smaller of the two values:

$$P(\xi_{E_t}|\xi^{\theta^*}_{E_{1:n}}) \propto \min\left(\mathbb{E}_{\theta\sim P(\theta|\xi^{\theta^*}_{E_{1:n}})} P(\xi_{E_t}|\theta), \sum_{i=1}^{n} g_E(E_t, E_i) g_T(\xi_{E_t}, \xi^{\theta^*}_{E_i})\right). \tag{13}$$

This models end-users as being conservative: both lines of reasoning must indicate that a particular trajectory is likely, in order for people to anticipate that the robot will take that trajectory.

Evaluation. Recall that in our user studies, we showed participants a sequence of autonomous car trajectories $\xi^{\theta^*}_{E_{1:n}}$ (that varied across conditions) and then asked them to assign a confidence rating for each of four trajectories in a new environment E_t, based on whether they thought it was the same autonomous car driving for that new trajectory. These confidence ratings capture how likely a participant thinks that trajectory is in environment E_t, given the observed examples of robot behavior. So, we can evaluate models $\mathcal{M}$ of human learning based on how well their predicted values of $P_{\mathcal{M}}(\xi_{E_t}|\xi^{\theta^*}_{E_{1:n}})$ correlate with the confidence ratings from our user studies, averaged across all users in the same condition. The (five) approximate-inference models all have a weak correlation, with Pearson correlation coefficients r between 0.34 and 0.40 (p values $< 10^{-5}$).

For the exemplar-based model, we set $g_E(E_a, E_b)$ to 1 if E_a and E_b are in the same environment class (e.g., merging), and 0 if they are not. This captures that when faced with a novel environment, people might recall what they observed the robot doing in other environments with the same class, and use that to anticipate what the robot will do in the new environment. We set $g_T(\xi_a, \xi_b)$ to 0 if ξ_a and ξ_b are not in the same trajectory strategy cluster, and to $d_e(\xi_a, \xi_b)$ if they are. This captures that people consider a trajectory more likely if it

shares the same strategy and is similar (in terms of Euclidean distance) to one they have observed in the past. In particular, people only realize the robot is capable of executing a particular strategy if they have previously observed the robot doing it. This exemplar-based model also correlates weakly with the user data, with a Pearson correlation coefficient r of 0.40 (p value $< 10^{-5}$).

This particular exemplar-based model relies on having a reasonable specification of environment classes and trajectory strategies. In our experimental domain, these were defined based on hand-picked rules, but they could be automatically computed instead. Environment classes can be obtained automatically by clustering the environments. Trajectory strategies can be obtained automatically by clustering the trajectories in each environment class, e.g., based on trajectory feature vectors $\mu(\xi_E)$. Or, trajectory strategies could correspond to the homotopy classes in an environment.

Our proposed hybrid model [Eq. (13)] correlates more strongly with the user data than either the approximate-inference or exemplar-based models alone, with Pearson correlation coefficients r between 0.40 and 0.61 (p values $< 10^{-5}$). This suggests that participants rely on both modes of learning in this domain.

9 Discussion

Summary. We take a step towards communicating robot objective functions to people. We found that an approximate-inference model using a deterministic Euclidean-based update on the space of candidate objective function parameters performed best at teaching real users, and outperforms algorithmic teaching that assumes exact inference. After augmenting such a model with a coverage objective, it outperformed letting the user passively familiarize to the robot.

We additionally provide an empirical analysis of alternative learner models, in which we evaluate how mismatch between the assumed learner model and the true model impacts the effectiveness of algorithmic teaching, and propose a combined approximate-inference and exemplar-based model that better captures how users learn in this domain.

Implications. Making robot objective functions more transparent to people is a worthwhile goal to work towards. As robots become increasingly capable and are deployed in more situations, understanding what a robot is optimizing for (and thus being able to anticipate how it will move around in its environment) is key to safe, comfortable, and coordinated human-robot interaction.

As an example of a concrete future use case, imagine allowing users to tune a knob that controls the objective function a robot optimizes. This knob would likely not correspond to the true reward features; it may instead collapse them onto one or two interpretable axes. For instance, for an autonomous car, this knob may control how efficient versus defensive the car drives. Each point on the dial would correspond to a different setting of reward parameters. Our approach could be used to efficiently communicate the behavior corresponding to each point, so passengers can choose the one they are most comfortable with.

Limitations and future directions. Our results reveal the promise of algorithmic *teaching* of robot objective functions. Future work could apply this framework to teach more complex objective functions, for instance those that reason about the intentions and goals of other agents, or how other agents will respond to the robot's behavior [as in Sadigh et al. (2016b)].

However, the coverage results suggest that an IRL-only model is not sufficient for capturing how people extrapolate from observed robot behavior. We took a step towards investigating whether users may be directly learning policies in addition to reasoning about the robot's objective function. There is more work to be done on finding accurate models of human learning in this domain.

Our analysis also suggests that it is important for the robot and end-users to achieve common ground on which features are important. This analysis was limited to end-users considering either a subset or superset of the features that define the robot's objective function. Future work could study more complex differences in features considered, or investigate interactions for achieving common ground on which features to pay attention to.

Furthermore, in this work we focused on the robot's physical behavior as a communication channel because people naturally infer utility functions from it. Future work could augment this with other channels, such as visualizations of the objective function or language-based explanations.

Funding This study was funded in part by Intel Labs, Air Force #16RT0676, an NSF CAREER Award (#1351028), DARPA XAI, and the Berkeley Deep Drive consortium. Sandy Huang was supported by an NSF Fellowship.

Compliance with ethical standards

Conflict of interest Anca Dragan also serves as a research scientist at Waymo, Inc. Waymo did not contribute to or support this work. The remaining authors declare that they have no conflict of interest.

Ethical approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards.

References

Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st international conference on machine learning*

Ashby, F. G. (1992). *Multidimensional models of perception and cognition* (1st ed.). Hove: Psychology Press.

Baker, C. L., Saxe, R., & Tenenbaum, J. B. (2009). Action understanding as inverse planning. *Cognition*, *113*(3), 329–349.

Baker, C. L., Saxe, R. R., & Tenenbaum, J. B. (2011). Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of the 33rd annual conference of the cognitive science society* (pp. 2469–2474)

Balbach, F. J., Zeugmann, T. (2009). Recent developments in algorithmic teaching. In *Proceedings of the 3rd international conference on language and automata theory and applications*

Basu, S., & Christensen, J .(2013). Teaching classification boundaries to humans. In *Proceedings of the 27th AAAI conference on artificial intelligence*

Cakmak, M., & Lopes, M. (2012). Algorithmic and human teaching of sequential decision tasks. In *Proceedings of the 26th AAAI conference on artificial intelligence*

Chajewska, U., Koller, D., & Ormoneit, D .(2001). Learning an agent's utility function by observing behavior. In *Proceedings of the 18th international conference on machine learning* (pp. 35–42)

Choi, J., & Kim, K. (2011). MAP inference for bayesian inverse reinforcement learning. In *Proceedings of the 24th annual conference on neural information processing systems* (pp. 1989–1997)

Cohen, P., Perrault, C., & Allen, J. (1981). Beyond question answering. In *Strategies for natural language processing* (pp. 245–274)

Dragan, A., & Srinivasa, S. (2014). Familiarization to robot motion. In *Proceedings of the 9th international conference on human-robot interaction*

Dragan, A., Bauman, S., Forlizzi, & J., Srinivasa, S. (2015). Effects of robot motion on human-robot collaboration. In *Proceedings of the 10th international conference on human-robot interaction*

Gielniak, M. J., & Thomaz, A. L. (2011). Generating anticipation in robot motion. In *20th IEEE international symposium on robot and human interactive communication*

Gielniak, M. J., Liu, C. K., & Thomaz, A. L. (2013). Generating human-like motion for robots. *International Journal of Robotics Research*, *32*(11), 1275–1301.

Goldman, S. A., & Kearns, M. J. (1995). On the complexity of teaching. *Journal of Computer and System Sciences*, *50*(1), 20–31.

Hayes, B., & Shah, J. A. (2017). Improving robot controller transparency through autonomous policy explanation. In *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*

Jara-Ettinger, J., Gwen, H., Schulz, L. E., & Tenenbaum, J. B. (2016). The naïve utility calculus: Computational principles underlying commonsense psychology. *Trends in Cognitive Sciences*, *20*(8), 589–604.

Khan, F., Mutlu, B., & Zhu, X. (2011). How do humans teach: On curriculum learning and teaching dimension. In *Proceedings of the 24th annual conference on neural information processing systems*

Koedinger, K. R., Brunskill, E., de Baker, R. S. J., McLaughlin, E. A., & Stamper, J. C. (2013). New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, *34*(3), 27–41.

Levine, S., & Koltun, V. (2012). Continuous inverse optimal control with locally optimal examples. In *Proceedings of the 29th international conference on machine learning*

Matsui, D., Minato, T., MacDorman, K. F.,& Ishiguro, H. (2005). Generating natural motion in an android by mapping human motion. In *IEEE/RSJ international conference on intelligent robots and systems*

Medin, D. L., & Schaffer, M. M. (1978). Context theory of classification learning. *Psychological Review*, *85*(3), 207–238.

Nemhauser, G. L., Wolsey, L. A., & Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, *14*(1), 265–294.

Ng, A. Y., & Russell, S. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the 17th international conference on machine learning*

Nikolaidis, S., Nath, S., Procaccia, A.,& Srinivasa, S. (2017). Game-theoretic modeling of human adaptation in human-robot collaboration. In *Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction*

Nosofsky, R. M. (1986). Attention, similarity, and the identification-categorization relationship. *Journal of Experimental Psychology: General*, *115*(1), 39–57.

Patil, K. R., Zhu, X., Kopeć, Ł.,& Love, B. C. (2014). Optimal teaching for limited-capacity human learners. In *Proceedings of the 27th annual conference on neural information processing systems*

Perera, V., Selvaraj, S. P., Rosenthal, S.,& Veloso, M. M. (2016). Dynamic generation and refinement of robot verbalization. In *25th IEEE international symposium on robot and human interactive communication*

Rafferty, A. N., Brunskill, E., Griffiths, T. L., & Shafto, P. (2011). Faster teaching by POMDP planning. In *Proceedings of the 15th international conference on artificial intelligence in education* (pp. 280–287)

Ramachandran, D., & Amir, E. (2007). Bayesian inverse reinforcement learning. In *Proceedings of the 28th International Joint Conference on Artifical Intelligence* (pp. 2586–2591)

Raman, V., & Kress-Gazit, H. (2013). Explaining impossible high-level robot behaviors. *IEEE Transactions on Robotics*, *29*(1), 94–1104.

Rosenthal, S., Selvaraj, S. P., & Veloso, M. M. (2016). Verbalization: Narration of autonomous robot experience. In *Proceedings of the 25th international joint conference on artificial intelligence*

Russell, S. J., & Norvig, P. (2009). *Artificial intelligence: A modern approach* (3rd ed.). Upper Saddle River: Prentice Hall Press.

Sadigh, D., Sastry, S. S., Seshia, S. A.,& Dragan, A. (2016a). Information gathering actions over human internal state. In *IEEE/RSJ international conference on intelligent robots and systems*

Sadigh, D., Sastry, S. S., Seshia, S. A., & Dragan, A. D. (2016b). Planning for autonomous cars that leverage effects on human actions. In *Proceedings of robotics: Science and systems*

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *26*(1), 43–49.

Schmidt, C., Sridharan, N., & Goodson, J. (1978). The plan recognition problem: An intersection of psychology and artificial intelligence. *Artificial Intelligence*, *11*(1), 45–83.

Sebanz, N., Bekkering, H., & Knoblich, G. (2006). Joint action: Bodies and minds moving together. *Trends in Cognitive Sciences*, *10*, 70–76.

Shepard, R. N. (1964). Attention and the metric structure of the stimulus space. *Journal of Mathematical Psychology*, *1*(1), 54–87.

Singla, A., Bogunovic, I., Bartok, G., Karbasi, A., & Krause, A. (2014). Near-optimally teaching the crowd to classify. In *Proceedings of the 31st international conference on machine learning*

Szafir, D., Mutlu, B., & Fong, T. (2014). Communication of intent in assistive free flyers. In *Proceedings of the 2014 ACM/IEEE international conference on human-robot interaction*

Taheri, S., & Law, E. H. (1990). Investigation of a combined slip control braking and closed loop four wheel steering system for an automobile during combined hard braking and severe steering. In *Proceedings of the American control conference*

Tellex, S., Knepper, R., Li, A., Rus, D., & Roy, N. (2014). Asking for help using inverse semantics. In *Proceedings of robotics: Science and systems*

Ullman, T., Baker, C., Macindoe, O., Evans, O., Goodman, N., & Tenenbaum, J. B. (2009). Help or hinder: Bayesian models of social goal inference. In *Proceedings of the 22nd annual conference on neural information processing systems*

Vul, E., Goodman, N. D., Griffiths, T. L., & Tenenbaum, J. B. (2014). One and done? Optimal decisions from very few samples. *Cognitive Science*, *38*(4), 599–637.

Zhu, X. (2015). Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Proceedings of the 29th AAAI conference on artificial intelligence*

Ziebart, B. D., Maas, A., Bagnell, J. A., & Dey, A. (2008). Maximum entropy inverse reinforcement learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*

Zilles, S., Lange, S., Holte, R., & Zinkevich, M. (2011). Models of cooperative teaching and learning. *Journal of Machine Learning Research*, *12*, 349–384.

Sandy H. Huang is a Ph.D. candidate at UC Berkeley in Computer Science. Her research focuses on transparent and explainable artificial intelligence, with the goal of providing users with a deeper understanding of what a robot or agent has learned. Sandy received her B.Sc. in Computer Science from Stanford University in 2013.

David Held is an assistant professor at Carnegie Mellon University in the Robotics Institute. His research focuses on robotic perception for autonomous driving and object manipulation. Prior to coming to CMU, David was a post-doctoral researcher at UC Berkeley, and he completed his Ph.D. in Computer Science at Stanford University where he developed methods for perception for autonomous vehicles. He has also worked as an intern on Google's self-driving car team. He has a B.S. and M.S. in Mechanical Engineering at MIT. He is a recipient of the Google Faculty Research Award in 2017.

Pieter Abbeel is a professor at UC Berkeley in Computer Science. His research primarily focuses on studying deep learning for robotics, with autonomous manipulation, flight, locomotion, and driving as targeted application domains. Pieter received his Ph.D. from Stanford University in 2008.

Anca D. Dragan is an assistant professor at UC Berkeley in Computer Science. Her research focuses on algorithms for human-robot interaction, with the goal of enabling robots to work with, around, and in support of people. Her work draws from optimal control, planning, estimation, learning, and cognitive science. Anca received her Ph.D. from Carnegie Mellon University in 2015.