UNIVERSITY OF CALIFORNIA, MERCED

Personalizing Autonomous Driving with Rich Human Guidance

A dissertation submitted in partial satisfaction of the requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Chandrayee Basu

Committee in charge:

Professor Mukesh Singhal, Chair
Professor Marcelo Kallmann
Professor Paul P. Maglio
Asst. Professor Anca D. Dragan

2019

Copyright Notice

The Dissertation of Chandrayee Basu is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Anca D. Dragan

Marcelo Kallmann

Paul P. Maglio

Mukesh Singhal, Chair

University of California, Merced

2019

Contents

List of Figures

List of Tables

Acknowledgment

First of all, my biggest thank you to my advisor Mukesh Singhal. I could not have finished this thesis while still balancing my pregnancy and new motherhood without your support and persistent belief in my ability. Your confidence in my ability and your appreciation really boosted my confidence. Coming from an non-CS background didn't seem like a handicap at all, because of your support. I am utmost thankful to the flexibility that you gave me, that helped me forge wonderful collaborations with some of the best researchers in Human-Robot Interaction. Thank you for all your academic as well as non-academic advice.

One person who was really instrumental in this thesis was Anca Dragan. Anca, without your willingness to work with me, without your valuable advice and support, I would not be working on what I love the most now: Algorithms for Human Robot Interaction. It was so easy to collaborate with you remotely. I learned how to discuss equations over emails. My first HRI paper from this thesis was a fruit of your careful hand-holding. Thank you for having the confidence in me and for guiding me amidst your busy schedule. Finally, thank you for being on my thesis committee. I hope we can collaborate later on the more pressing problems in human-robot interaction and AI.

My sincere thank you to my other committee members Paul Maglio and Marcelo Kallmann. Paul, you helped me with the biggest challenge in any human-robot interaction work: IRB approval for user studies. Thank you for making it so easy for me. We had some great initial conversations that offered me a non-algorithmic perspective to practical problems in human-autonomous car interaction. I would also like to thank your student Umesh Krishnamurthy for his support and help with setting up my very first user study. Thank you Marcelo for your valuable advice on path planning algorithms and for being on my committee.

My last work in this thesis was in collaboration with Dorsa Sadigh and Erdem Bıyık. Dorsa, we had been talking about this project for a long time. I had a pregnancy and a baby meanwhile, and finally our project saw the light of the day. Thank you for your advice and patience. Thank you for introducing me to Erdem. Erdem, without your insight on mixture models, this thesis would not be complete. I look forward to more research interactions with you and Dorsa.

Throughout my thesis I have been surrounded by superb collaborators, fellow students and friends who supported me and gave me valuable advice. Thank you Santosh Chandrasekhar for helping me set up my lab. A special thanks to my friend and collaborator Qian Yang. I know I can ping you anytime if I have questions about user interaction and your advice will always be useful. Thanks to David Hungermann, my first undergraduate intern at UC Merced. You put in a lot of hard work. I wish you all success in all your future pursuits in CS. I was very lucky to have access to some extremely talented and budding researchers in the field of AI, thanks to Anca. Dylan Hadfield Mennel and Sandy Huang, thank you for casual advice from time to time. While, we did not get a chance to explicitly collaborate during my thesis, some of my work would not have been possible without the our conversations. I look forward to many more research conversations and collaborations with you.

I would not have the guts to pursue a Ph.D. in CS and for that matter in Engineering, if not for Alice Agogino. Alice, thank you for being the best mentor I have ever had. You gave me the first opportunity to work on project that involved wireless sensor network and machine learning, when I did not know what wireless network meant and what gradient descent was all about. You have been a constant source of inspiration since then. In your lab, I met some very successful women in engineering, an experience that has left me changed forever. You gave me the opportunity to mentor some really hard working students of UC Berkeley, Aparna Dhinakaran, Elizabeth Chen, Derek Tat, Julien Caubel and Ryan Paulson. I am thankful to Dr. Aaron Steinfeld for giving me the final push when I decided to apply for a Ph.D. in CS. Aaron, thank you for being a true mentor and thank you for introducing me to the field of human-robot interaction. I finally found my call.

Now a special shoutout to my dearest friend Sohyeong Kim. Soh, you were always there for me through thick and thin. Thank you for being my working buddy. Mariam, thank you for always being my first user. It is hard to be the first user of a pilot study. I will never forget your comment about the simulator. Thanks Shruti Patil for all the valuable career advice. Urmi, thank you for telling me about the opportunity to transition from architecture to engineering in India. You started it all.

Finally, many many thank you to Narayanan, my husband, for being a sport even when I was asking you questions about my paper on our first anniversary trip! Thank you for always having answers to my questions, be it on machine learning, optimization, paper organization or questions as simple as “does this graph look legible?”. You know very well that this thesis would not have been possible without your continuous support at every step. I hope to give you some rest now, may be for a week. I dedicate this thesis to my parents who had the patience and faith as I was pursuing a major career transition. My little baby Trinaabh, you are lucky that you

have your name on a Ph.D. thesis just a year after you were born. Thank you for giving me the most beautiful and wonderful moments of joy since you have arrived. You made me smile even on my most difficult workdays.

Abstract

Personalizing Autonomous Driving from Rich Human Guidance
A Ph.D. dissertation by: **Chandrayee Basu**
Electrical Engineering and Computer Science
University of California, Merced. 2019.
Committee chair: Professor Mukesh Singhal

With progress in enabling autonomous cars to drive safely on the road, it is time to ask *how* should they be driving. This dissertation focuses on learning the desired objective function for autonomous cars with the goal of *personalizing* autonomous driving: *drive following the passenger's preferences across diverse environments*. Traditionally autonomous cars have been trained using expert demonstrations, with an implicit assumption that the demonstrations are truly representative of optimal driving. Personalizing autonomous driving under this assumption would mean using Inverse Reinforcement Learning (IRL) to learn the objective function latent in the user's own demonstration and then adopt the user's own driving style. In this thesis, we question this assumption and propose algorithmic solutions for personalizing driving styles without demonstration data. Through user studies in a simulated driving environment, we first show that people do not want their autonomous cars to drive like them: they want a significantly more defensive car. Next we formalize driving preference as reward functions and propose several algorithms to learn them *interactively* from an alternative form of human guidance: *Preference-based Learning*. In Preference-based reward learning we show users several trajectory pairs sequentially and ask them to indicate their preference in each pair. This has been shown to be effective for learning reward functions in absence of demonstrations. Simple preference is, however, far less informative than all the demonstration data. The key contribution of this thesis is an algorithmic framework that leverages computational models of human behavior to enable learning from *richer* preference queries where response to each query contains more information than just a comparison. We propose different forms of rich preference queries. We ask people not only *what* they prefer, but also *why* they prefer. We design new queries to learn more complex reward functions that can potentially represent preferences in non-stationary environments. We introduce *reward dynamics* as a mixture of reward functions and parameters that govern *how* preferences change in response to the dynamics of the environment. We develop a unified formalism for treating all forms of human guidance as observations about the true preferences and use this formalism to derive objective functions for *actively* generating *rich* queries. We show empirically through simulations and also with user studies that richer preference queries can learn driving preference more accurately than comparison-alone queries. We also discover that richer queries not only speed up preference learning in practice but also offer more transparency into the decision-making algorithms of the autonomous car, thus enhancing people's trust in the system. Although the human-robot system of choice in this thesis is autonomous car, our algorithmic solutions

apply to personalizing other human-robot systems where the robot is a dynamical system that should match human preference and demonstrations are unavailable due to complexity of robot operation or disparity between preferences and demonstrations.

Chapter 1

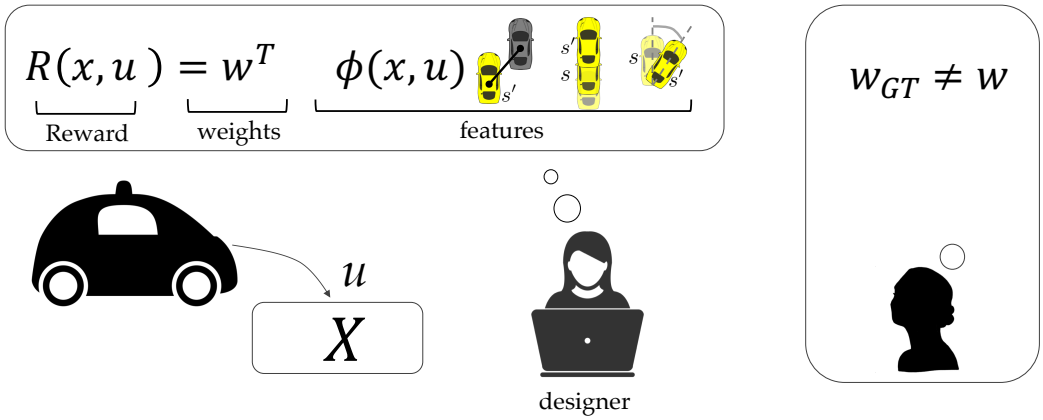
Introduction

1.1 Motivation

Today we are rapidly approaching towards a future where many of our daily tasks will be shared by the autonomous systems that interact and collaborate with humans. Autonomous car is one such powerful system that can relieve humans of the boring and dangerous task of driving. Humans will play a significant role in the operation of these systems: as part of the external traffic that the cars will interact with and as passengers who the cars will serve. Researchers and companies are getting ever-so-close to enabling autonomous cars to generate safe driving behavior that includes reaching the destination while satisfying safety constraints, like not colliding with other cars or pedestrians. Once these cars attain that level of capability, initially, they might be able to generate, for each driving situation, only one solution trajectory (or behavior) that satisfies these safety and feasibility constraints. But really, many solutions exist – there are many ways to drive. This depend on the individual trade-offs that each driver makes. We have an existence proof for that. Some of us are more *aggressive* drivers, valuing efficiency and being comfortable getting close to other cars on the road. Others are more *defensive*, a bit more conservative when it comes to safety, leaving a large distance to the next car for example, or quickly braking when someone attempts to merge in front. Individual driving preferences and disapproval thereof can also manifest in the form of backseat driving. *And these trade-offs may not always match the design* (see Fig.1.1). The single optimal behavior generated by the autonomous car may not be the preferable trajectory of its passenger. Overtly conservative behavior can engender impatience whereas aggressive behavior can lead to discomfort and distrust of risk-averse passengers.

The purpose of this thesis is to enable autonomous cars to drive following its user preferences.

For instance, consider a lane change situation where a car is approaching from behind in the target lane. For a user who weighs safety a lot more than efficiency the car will learn to slow down and merge after the car in the target lane and for the user who prefers high speed and efficiency the car will be trained to attempt a merge in



a) The robots take actions a that affect the state X of the world. These actions are optimal with respect to some reward function, expressed as a weighted combination of features. Currently these weights or reward parameters come from the robot designer.

b) The reward parameters of the designed robot may not align with the ground truth reward desired by the users of the technology

Figure 1.1: Right now the autonomous cars are able to generate one safe trajectory that can reach the destination without colliding with other objects. This trajectory is optimal with respect to some reward function that is provided by the designer or learned from expert demonstrations. But this reward function may not match the how the users want the autonomous cars to drive leading to discomfort and distrust. In this thesis, we develop algorithms that enable the cars to learn to drive following user preferences.

front of the other car.

1.2 Thesis Approach

Autonomous cars that are trained with expert driving demonstrations and learn to imitate expert driving in known traffic conditions have a generic safe and conservative driving behavior. However there is no guarantee that the optimal driving behavior of one expert will match the optimal driving preference of another human. Here we address the problem of *personalizing autonomous driving style* : matching the driving behavior of the autonomous car with the preferred style of the passenger, referred to as the *user* in the rest of the thesis. A common approach to following user preference is to learn or imitate the driving style of the user (79; 114; 81). Learning driving preference from user data have an underlying assumption that people want their cars to drive like them, that aggressive drivers prefer aggressive style and defensive drivers prefer defensive style. Our approach to behavior planning that match user preferences starts out by putting to test the assumption that user's own driving style is the same as their driving preferences. *We learn that user preferences are different from their*

own behavior.

We model the autonomous car as a dynamical system that should match human preferences and present the task of personalizing driving style as a human-robot interaction problem, where the human is the user and the autonomous car is the robot. We encode human preferences as reward functions governing trade-offs between features of the trajectory and learn these functions in a human-robot 2-player game setting similar to Cooperative Inverse Reinforcement learning (60; 49). In this partial information game, the human is a noisy rational agent who fully but noisily observes the true reward function, but the robot does not. The robot's objective is to interact with the human in a way to learn the most about this reward function during any single interaction. The robot has an initial belief over the true reward functions, which it updates through interactions. The robot's final payoff is the human's exact preference. We draw from recent work in *active Preference-based reward learning* (110) and design interactions where the robot learns by asking choice questions to the human.

- We explore comparison queries as the mode of interaction between the robot and the human
- We introduce the concept of *rich human guidance*: different ways to augment comparison queries into a hierarchical structure for richer information gain per query

Our approach broadly relates to the value alignment problem in AI where the goal is for an AI agent to attempt to act according to human values. Unlike prior approaches to learning policies or reward functions for autonomous cars, *passively* from human demonstrations, our goal in this thesis, is to learn preferences *interactively* in the absence of demonstrations.

1.3 Contributions

This thesis makes the following contribution (see Fig.1.2):

The goal of my thesis is to enable autonomous cars to drive according to user preference. Since human preferences do not match human driving demonstrations we develop algorithms that allow the cars to learn human preferences by seeking rich end-user guidance.

Should Autonomous Cars Learn from User Demonstrations?

We first attempt to learn whether user demonstrations are a way to go for personalizing autonomous driving. We hypothesize that *users want a driving style that is different from their own.* We design and conduct a user study to start analyzing

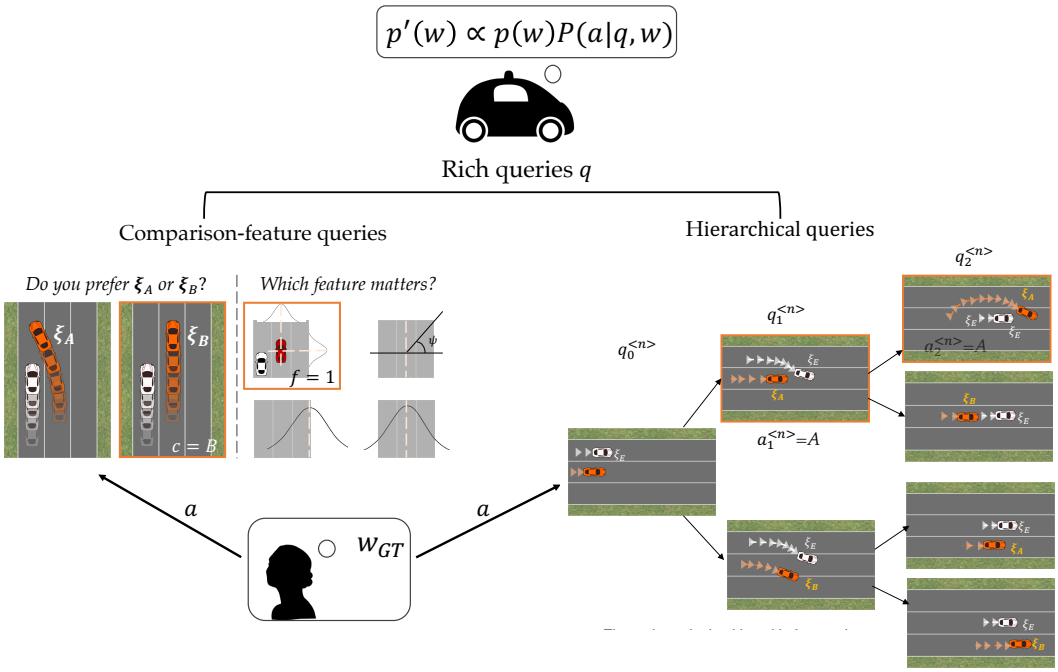


Figure 1.2: We introduce several forms of rich non-verbal guidance that robots can seek from humans, to be able to learn their desired reward functions. We first augment comparison queries with a follow-up feature query. We then design more complex hierarchical queries as series of connected comparison sub-queries for learning more complex dynamic preferences. The autonomous car (robot) has an initial belief over the reward functions $p(w)$. The belief updates to $p'(w)$ using the answers a to the queries.

the potential differences between how users drive and how they want to be driven. Our study, conducted in a driving simulator, has two parts: first, the users come in and demonstrate their driving in different environments; second, at a later date, the same users come in and test four driving styles: their own (though they do not know it is their own), an aggressive style, a defensive style, and another user's style. We measure their preference for these styles, as well as the perceived similarity to their own style.

Our results suggest that there is truth to both sides:

> Users do not actually want the car to drive like they drive. Instead, they want the car to drive like they *think* they drive.

We find a significant difference in user's own style and preferred style, with users typically preferring more defensive driving when they are passengers. Overall, our work does not contradict the need for customization, but suggests that it might not be sufficient to learn how the user drives. Instead, we need to learn how the user actually wants to be driven. This raises challenges for learning, because we can no longer rely on demonstrations – users can easily demonstrate how they drive, but they might not be able to demonstrate the driving style they want. Instead, we need to rely on different kinds of input and guidance from users in the learning process [Chapter.3] (21).

Personalization as a Human-Robot Interaction Game

Our goal is to integrate human driving preference into the behavior planning of an autonomous car, i.e. given a starting state of the car, the world and the surrounding traffic the planner should generate a trajectory for the next few time steps optimal with respect to the preference of the passenger. Traditional behavior planning modules of the autonomous cars treat humans like pedestrians and bicyclists as bounded disturbances and other drivers as having a fixed velocity profile or they follow a set of known trajectories. Other approaches have integrated human driving style into motion planning indirectly by learning from human demonstrations, formally called Learning From Demonstration (LfD). LfD is used to generalize a human demonstration. The two forms of LfD: learning a latent intent behind driving behavior as a reward function and learning a direct mapping from states to actions as a policy *passively* learn from human demonstrations. In this thesis, we treat personalizing driving behavior as a *human-robot interaction* problem. We formalize this problem as a game where the robot is *interactively* learning human preference by making queries. The human and the robot have the same reward and the robot employs active learning to generate these queries. We encode human preference as a tuple of reward functions parameterized by the preference mode and parameters governing the transitions between the mode-specific reward functions [Chapter.4].

Learning from Rich Queries

Inverse Reinforcement Learning (IRL) is a natural way for robots to *learn* the desired reward function. IRL collects demonstrations from a person of the desired *behavior*, rather than of the desired reward, and finds parameters for the reward function that explain the demonstrated behavior. In our first work we found that people cannot demonstrate their true preference when it comes to autonomous driving.

In this thesis, we focus on situations where we do not have access to demonstrations. Comparison-based learning has emerged as a promising alternative for learning reward in such cases. There, the robot iteratively shows users two possible trajectories (often in the same environment, for the same starting state), and asks which they prefer. It then uses the answer to update its understanding of the reward parameters. Comparisons are, however, slower form of learning than demonstrations because instead of giving the optimal of our reward functions we keep testing if one policy is better than another. We do this over several iterations, sometimes thousands with deep representation of the policies.

> *We propose that robots can extract richer guidance from people when learning reward functions.*

The robot takes a *hierarchical* approach to learning and seeks richer guidance in the form of augmented comparison query: a sequence of comparison and other choice type sub-queries. The key contribution of this thesis is a unified framework for treating answers to all kinds of queries as observations about the true preference. Further, we propose that *richer* guidance can lead to learning *richer* preference model. To this end we learn a more complex model of human preference representative of both static and dynamic environments, i.e. when the preference mode changes in response to the dynamics of the environment, for example sudden traffic congestion, change in courtesy level of other cars to name a few [Chapter.5].

Comparison and Feature Queries

We first learn a static preference model by augmenting comparison-based learning with a follow-up *why*:which feature in the reward model is the most responsible for the comparison answer.

Rich Queries combining Comparisons and Features We build on prior work that leveraged feature queries in the context of learning skills from demonstration to introduce *combined comparison-feature queries* for reward learning.

Learning from Rich Queries We generalize comparison-based learning to these richer queries by treating feature answers as observations about the true reward parameters. We introduce a unifying formalism whereby the person's answers are all treated as nosily-optimal responses conditioned on the true reward, and perform Bayesian inference to estimate the reward parameters.

Active Query Selection To speed up learning, we derive a rich query selection method that optimizes for gathering as much information as possible from each query.

Analysis of Rich Queries We conduct thorough experiments in simulation showing that *rich queries* learn faster than *comparison-only* queries, and follow-up with an in-lab study on learning driving style. We find that rich queries learn a reward that is significantly closer to the users' internal preference. This is evidenced by them preferring the robot that optimizes the reward function learned through rich queries over the robot that optimizes the reward learned through comparison-only queries. People also report a better experience with training the robot in case of rich queries [Chapter.6] (20).

Learning Reward Dynamics from Human Guidance

Enabling robots to act according to human preferences across diverse environments is a crucial task, extensively studied by both roboticists and machine learning researchers. To achieve it, human preferences are often encoded by a reward function which the robot optimizes for. This reward function is generally static in the sense that it does not vary with time or the interactions. Unfortunately, such static reward functions do not always adequately capture human preferences, especially, in non-stationary environments: Human preferences change in response to the emergent behaviors of the other agents in the environment. In this work, we propose learning *reward dynamics* that can adapt in non-stationary environments with several interacting agents. We formalize *reward dynamics* as a mixture of static reward functions representing different preference moods and a set of parameters for the transitions between the moods. We apply richer guidance to learn a distribution over *reward dynamics*. We allow the users to change their preferences within the same query.

Hierarchical Comparison Queries We extend our previous comparison query setting to *hierarchical* queries, where each query is a sequence of sub-queries. Each sub-query is a continuation of the trajectories from the previous sub-query. Prior works have learned static reward functions by asking people to compare between two different trajectories of robots. There, each query is a pair of short videos that demonstrate two trajectories of the system. Such short trajectories do not capture the nuances of interaction in a multi-agent system. The connected sub-queries are meant to capture the temporal aspect of the longer term interaction between the agents.

Learning a mixture of rewards Estimating reward parameters from a mixture is challenging due to identifiability issues related to label switching: invariance of likelihood to ordering of the parameters. We overcome this challenge by learning transitions between the reward functions and by introducing ordering within the transition parameters. This enables us to learn a bimodal preference model using pair-wise comparison unlike prior research (139).

Active Query Selection To speed up learning, we derive a *hierarchical* query selection method that optimizes for gathering as much information as possible from each query about changing human preferences. Our query selection algorithm selects feasible trajectories for each sub-query subject to the constraint that these sub-queries have connected trajectories.

Analysis of Reward Dynamics We analyze the performance of active preference

learning from richer guidance in simulation using random data and driving data in a bimodal setting i.e. people have two preference modes: cooperative and competitive. We show empirically that our algorithm learns accurate representation of the true preference function in both the modes [Chapter.7] (19).

Chapter 2

Related Work

We build upon a long history of research on robot cars (102; 46; 132; 34; 31; 131; 63; 12; 137; 32; 100) and borrow concepts like *human internal state inference* from human-robot interaction (17; 78; 50) and *preference modeling* from Economics and Machine Learning to design autonomous cars that can *weave in human preference into their behavioral decision making.*

2.1 Humans and Autonomous cars

The autonomous cars of today make decisions in a hierarchical fashion starting from route planning followed by behavioral decision making and then local motion planning and feedback control (100). The behavioral layer reasons about the environment including dynamic obstacles like human driven cars and pedestrians and generates a motion specification on a known route. The behavioral layer decides whether the car should change lane, cruise-in-lane or make a right turn. The motion planning module translates this behavioral choice into a path or trajectory. Our goal is to integrate human driving preference into *behavioral decision making* and *local motion planning* of an autonomous car, i.e. given a starting state of the car, the world and the surrounding traffic the planner should generate a trajectory for the next few time steps optimal with respect to the preference of the passenger. In this thesis, *we combine the tasks of behavioral decision making and motion planning into a single optimization problem.*

Some of the early cars were treated as finite state machines that could follow a behavior from a finite set of behaviors at any point of time with heuristics governing transitions like changing positions with respect to other traffic participants (34). These prototypes DID NOT HAVE AN EXPLICIT MODEL OF HUMAN BEHAVIOR.

HUMAN INTENTION AWARE MOTION PLANNING was introduced considering the uncertain behavior of other traffic participants. In fact, several autonomous car prototypes have used *intention prediction* of other traffic agents (31; 131; 63; 12;

137). Other approaches treated humans as bounded disturbances (59; 105), drivers as having a fixed velocity profile or they follow a set of known trajectories (90; 111). (117) used real-time intent prediction of the pedestrians for safer trajectory planning of autonomous cars. When it comes to actual motion planning, given the current perceptions and future predictions of the environment, these vehicles stick to entirely FUNCTIONAL solutions. But, human driving is far richer than a single functional solution. In contrast, in this thesis, we GO BEYOND PURELY FUNCTIONAL TRAJECTORIES and incorporate *the richness of human behavior* into the planned trajectories of the autonomous cars.

2.2 Driving Style from Human Demonstrations

End-to-end planning and control algorithms INTEGRATE HUMAN DRIVING STYLE INTO MOTION PLANNING INDIRECTLY by learning from human demonstrations (LfD) (102; 3; 124; 79; 96). LfD is used to generalize a human demonstration. Suppose a human shows a robot how to go from point a to b. With LfD a robot learns how to use the knowledge from this demonstration to go from point c to d perhaps with a new obstacle on the way. This approach falls into two categories: learning a mapping directly from states to actions (policy) using some supervised approach and learning the hidden intent behind driving style or the objective function which the car can later on optimize to generate driving policies.

2.2.1 Supervised Learning

Most of the early work in learning to drive by imitating an expert *imitation learning and apprenticeship learning* falls under the category of Behavioral Cloning: a way to find a direct mapping from states to actions $\pi : x \rightarrow u$, where π is called the policy (107; 65; 126; 125). The simplest approach is to perform a supervised learning using demonstration data in the form of state-action pairs $\mathcal{D} = (\mathbf{x}, \mathbf{u})$. Any regressor can be used to learn π. The training data consists of video camera recordings within and outside the car and target variable are the driver control inputs like steering, acceleration and braking. ALVINN (Autonomous Land Vehicle) (102) in a Neural Network) was the first attempt to use neural network mapping from images to navigation directions of the vehicle. More recently end-to-end vehicle control learning learn deep neural network policies using supervised regression. For example, Bojarski *et al.* used Convolutional Neural Network to map from front-view camera images to steering control (28; 27). Xu *et al.* proposed an end-to-end motion prediction approach that takes raw pixels from images and prior vehicle states signals as input and predict several sequences of discretized actions (135) . Kim and Canny used Attention models with CNN to explain deep driving policies as post-processed attention maps (77). While policy learning has proven to be useful for autonomous cars, it fails in learning long range and goal-directed behavior. In our case the lack of reliable demonstrations renders policy learning useless.

2.2.2 Reward Learning

Alternative to Behavioral Cloning is Inverse Reinforcement Learning (IRL), where instead of directly learning a policy, we assume that expert demonstrations are manifestation of optimal behavior and we recover the objective function or the *reward function* behind this behavior, denoted by $r(x^t, u^t)$. (x^t, u^t) is a state-action pair at time t. The reward function can naturally represent a style of behavior (latent variable or hidden state characterizing a specific behavior) and have been used extensively for that purpose in robotics (78; 17), animation (83) and autonomous driving (3; 79). Originally IRL is used to recover the objective function that potentially generated the human demonstration data. The probabilistic formulation of the algorithm is as follows:

$$\mathbb{E}[\sum_{t=0}^{\infty} r(x^t, u^t)|\pi^*] \geq \mathbb{E}[\sum_{t=0}^{\infty} r(x^t, u^t)|\pi] \quad \forall \pi \tag{2.1}$$

i.e. the expected sum of reward functions for demonstrated trajectory is greater than the reward functions that optimize all other trajectories. The IRL formulation assumes that humans are experts and demonstrations are always optimal with respect to a true reward function. One big challenge of IRL is that $r = 0$ is a valid solution. In fact, the above problem can have many solutions including degenerate ones(97). Algorithms for learning reward function from state action sequences as in Markov Decision Processes and those addressing degenerate solutions are recent developments (97; 4; 106; 140). Most of these algorithms use an approach called Feature Expectation Matching which models the reward as a weighted linear combination of features of the states x as:

$$r(x^t, u^t) = w^\top \phi(x^t, u^t) \tag{2.2}$$

where $\phi : X \to \mathbb{R}^d$ is an d dimensional feature function.

The problem in (2.1) then reduces to solving for a w^*, such that ${w^*}^\top \mu(\pi^*) \geq {w^*}^\top \mu(\pi)$ $\forall \pi$ where μ is the expected feature value also called feature expectation. Abbeel and Ng (2004) demonstrated that it is guaranteed to recover a policy as well as the expert policy if we can match the feature expectation $\mu(\pi^*)$ of the expert policy (4). Mathematically this can be written as:

$$\mu(\pi) - \mu(\pi^*) \leq \epsilon \tag{2.3}$$

implies that for all w with $\|w\| \infty \leq 1$

$$\left\| {w^*}^\top \mu(\pi) - {w^*}^\top \mu(\pi^*) \right\| \leq \epsilon \tag{2.4}$$

IRL formulation assumes that the expert behaves optimally. But when the real expert behavior is sub-optimal, i.e. no single policy maximizes the reward function Feature Expectation Matching breaks. Maximum Entropy IRL (140) extends IRL framework to sub-optimal expert behavior and attempts to find the maximum entropy

distribution over demonstrations ξ (sequences of states and actions of variable length) with the feature matching constraint as:

$$\begin{aligned} & max_P - \sum_{\xi} P(\xi) log P(\xi) \\ & s.t. \sum_{\xi} P(\xi)\mu(\xi) = \mu(\pi^*) \end{aligned} \tag{2.5}$$

Using the principle of maximum entropy, the distribution of human demonstration becomes:

$$P(\xi) = \frac{1}{Z(w)} exp(w^\top \mu(\xi)) \tag{2.6}$$

We use Maximum Entropy IRL to recover a reward function that represents human preference WHEN THE PREFERENCES DO NOT MATCH INDIVIDUAL DEMONSTRATIONS.

2.3 Learning Human Preferences

Broadly in robotics, generating trajectories that are aware of human's physical comfort and preferences is relatively new. For example, (50) considers robotic arm motions that human's can *read* and interpret, (74) learned how humans would prefer robot arms to perform manipulation tasks for dangerous objects. While researchers have largely used Learning from Demonstration to enable cars that can drive following a few different driving styles, making cars drive following individual preferences is only a very recent achievement (48). We build upon this work and borrow further from existing work on *robot teaching* and *models of human behavior* to develop algorithms that allow weaving in RICHER HUMAN PREFERENCES into the car's behavior.

2.3.1 Robot Learning from Queries

In this thesis, we enable robots to learn about human preferences THROUGH ACTIVE QUERIES. While preference learning for autonomous cars is a novel concept, the methods that we use for preference learning are certainly not new. Our query based approach to reward learning is inspired from several other works in robotics and machine learning. We build upon preference learning from human feedback in machine learning (30; 73; 5; 108; 7; 76; 35) and demonstration-free robot learning from human guidance (37; 36; 6; 38; 39; 66; 48; 45; 15) in robotics.

Preference learning is a sub-field in machine learning in which the goal is to learn a predictive preference model from observed preference information, like some form of *ranking* (5). Alternatively Fürnkranz (58) defined preference learning as learning

from observations that reveal information about the preferences of an individual or a class of individuals. In machine learning, preference learning can be categorized into label ranking, instance ranking, and object ranking. Typically in preference learning the queries are actively selected from a discrete set (51; 120; 76). Some of the successful applications include recommender systems and information retrieval (5; 7; 76). Fürnkranz *et al.* first applied preference learning to the problem of reinforcement learning (58). Jain *et al.* used a active ranking-based approach to learn user preferences for manipulation tasks (74). Recently comparison-based approach has emerged as a more intuitive form of learning in AI (134; 48; 43). Sadigh *et al.* learned driving preferences from active comparisons which involves showing people a pair of driving trajectories at every time step and asking for their preference over the two. While pair-wise comparisons are more intuitive and easier than ranking-based preference learning, they are much slower than learning from demonstration. In fact, learning deep reward representations can take more than 1000 such comparisons. We build on comparison-based learning because of its success in a demonstration-free setting, but, augment it with richer information to learn more efficiently. *We enable the robot to ask new kinds of questions to the users, answers to which can be treated as additional observations about their preference.*

In particular we draw from prior research on feature queries (103) in active classification tasks in machine learning and mixed queries (37) in active skill learning. The latter explored label queries (requesting for labels for unlabeled data), instance queries (requesting for example of a certain class) and feature queries (whether a state variable is relevant or important for a skill). They also found that the majority of the questions people ask during task learning fall in the category of feature queries (38). Therefore, feature queries are likely to be more intuitive for people to answer. WE INTRODUCE FEATURE QUERIES WITHIN THE PREFERENCE LEARNING FRAMEWORK.

2.3.2 Models of Human Behavior

Most of the early prototypes of autonomous cars treat humans as dynamic obstacles. The intent prediction algorithms used as part of behavioral decision making system so far are blackbox models that use some form of regression to map directly from sensor measurements to future positions (132; 86; 63; 12). A more recent genre of work in robotics is based on the idea that the robots of tomorrow should use theory of mind (49) to predict and infer human internal states like intent. This body of work uses Bayesian Theory of Mind to model people's action choices as function of some implicit reward (16). Inverse reinforcement learning already assumes human demonstrations as noisy evidences of their implicit reward functions (97; 104). The maximum entropy formulation of IRL: *humans are exponentially more likely to select actions that have higher rewards*, includes a plausible model of noisy rational behavior called Luce's Choice Axiom (89). The robot, however, in this case, is a passive

observer of demonstrations, inferring reward functions using this model. More recent work treats robot as an active observer that maintains models of people's responses to comparison queries in the domain of autonomous driving (48), models of how they will demonstrate actions in teaching mode (122; 57; 93), how people provide corrections while teaching robots (15), how they behave in collaboration (23; 60), how people may react to robot actions (116) and what people actually mean when they specify a reward function for the robot (62). We call these *observation models*. Like most of these works, we treat humans as noisily rational agents optimizing some reward function.

We extend the probabilistic observation model for comparison responses (48) and develop a UNIFIED FRAMEWORK FOR TREATING ALL FORMS OF ANSWERS AS OBSERVATIONS ABOUT TRUE HUMAN PREFERENCE.

2.4 Chapter Summary

Autonomous cars of today are great at *functional driving* that includes perception and navigation tasks like obstacle avoidance, lane-keeping and active steering and braking (54; 55). To do so, these cars maintain meticulously detailed models of the environment and continuously predict the intent and future trajectories of the dynamic traffic participants including the pedestrians. Moreover, intent prediction is based on simple regression or more sophisticated deep learning approaches. But, when it comes to behavioral decision making, we are able to produce only one solution trajectory for each scenario that meets all the functional constraints. Even, researchers who attempt to learn driving style assume that people want to drive their cars like they do and restrict to learning a few different driving styles from expert demonstrations. Our KEY INSIGHT in the thesis is that there is a lot of more richness in people's driving preferences. We can draw from the latest developments in the field of human-robot interaction and robot learning to incorporate this richness into the car's local trajectories.

Chapter 3

Should Autonomous Cars Drive Like Users?

The goal of this thesis is *personalization of autonomous driving*: to enable an autonomous car to follow trajectories according to user preferences. In all of the prior work, learning driving style entailed imitating an expert driver Chapter.2, which made an implicit assumption that humans want their cars to drive like they do, that sporty adventurous drivers was sporty and adventurous cars. Here we start out with an user study in a driving simulator to TEST THE ABOVE ASSUMPTION.

3.1 Definition of Driving Style

In order to find out whether users' preferred autonomous driving style match with their own driving style we first need a formal definition for *driving style*. We borrow from literature in traffic, transportation and robotics and define *driving style* as the typical behavioral patterns of a driver. This includes the choice of driving speed, headway, overtaking of other vehicles, or the tendency to commit traffic violations (133).

Defensiveness-aggressiveness is the most commonly used metric for defining driving style. Prior work refers to drivers as aggressive/assertive versus defensive (138); or mild versus moderate versus aggressive (136). In the Multidimensional Driving Style Inventory (MDSI), Taubman-Ben-Ari *et al.* identified four broad driving styles: (1) reckless and careless driving, characterized by, for example, higher speed; (2) anxious driving; (3) angry and hostile driving, characterized by more use of the horn and flash functionality; and (4) patient and careful driving (129). Similarly, Huysduynen categorized driving style as angry driving, anxious driving, dissociative driving, distress-reduction driving and careful driving style (133). Horswill *et al.* provided a valuable distinction between skill and style in the context of driving behaviors (70). Hong *et al.* (69) differentiated styles in terms of defensiveness, as well as by propensity for violation of rules. Scherer defined driving style in terms of comfort (119). Lee *et al.* (84) analyzed lane changes as a function of its severity (degree to which the

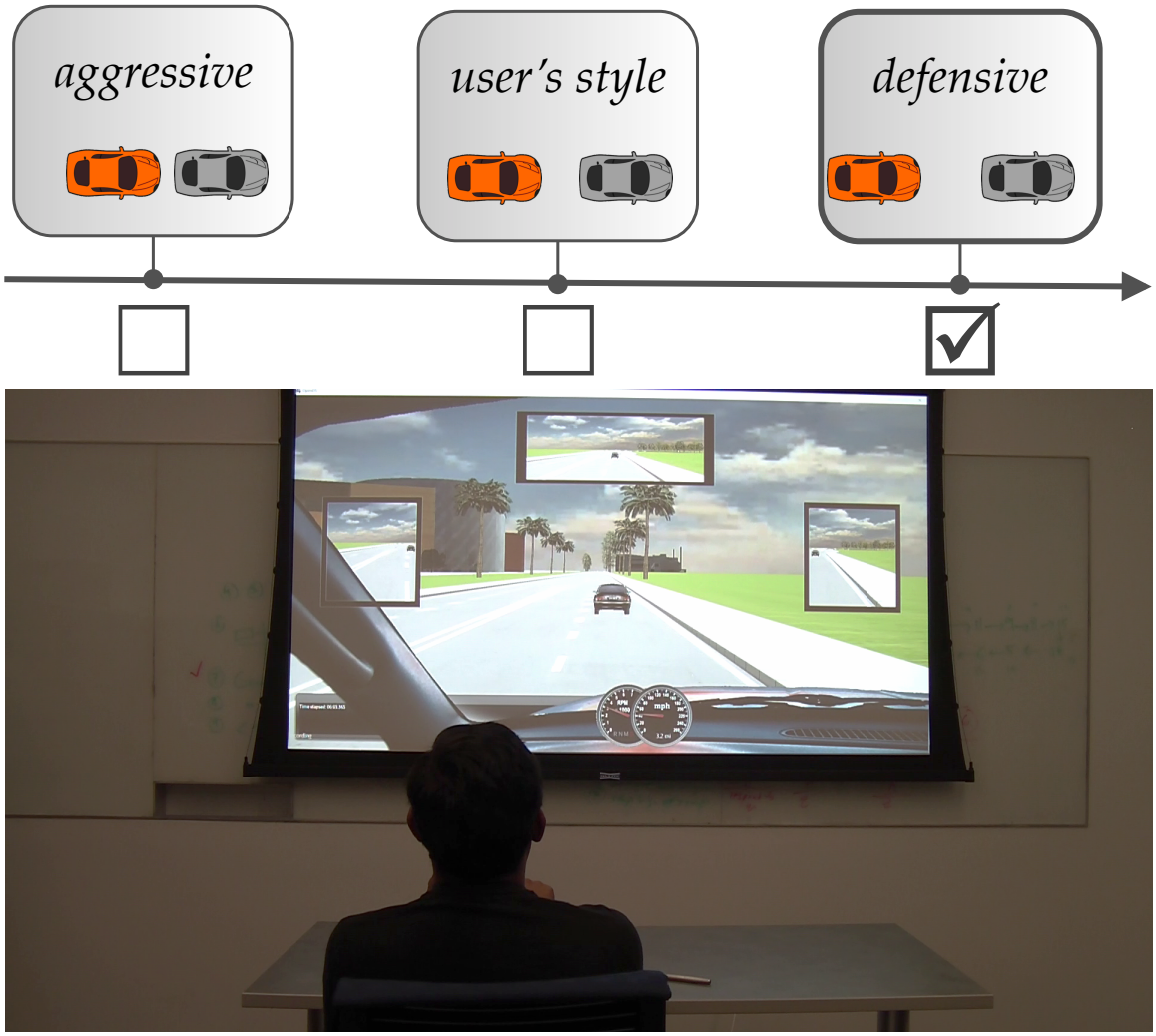


Figure 3.1: We first get data from user driving in different scenarios, and in a second session ask them to compare their own style (without knowing it is theirs), a more defensive style, and a more aggressive style. Participants tended to prefer a more defensive style than their own, but mistakenly thought they were actually picking their own.

vehicle in the destination lane was cut off), urgency (how soon the lane change was needed), and type classification for the full population of 8,667 lane changes.

We focus on driving style based on degree of defensiveness.

Driving style is a "humanized driving" quality (64). Hence, most of the driving style literature relates to understanding and modeling human driver behavior, in very specific traffic situations or contexts, like lane changing (84; 118; 92), intersection crossing (69; 18; 52), car following (29), and in terms of driving actions specific to those contexts (e.g., throttle and braking level, turning) and features thereof (e.g. rate of acceleration, rate of deceleration, maximum speed in a time window). *We*

define driving defensiveness in our work as an aggregate of driving features in various driving scenarios. Therefore, in our study, we present a combination of all of the aforementioned traffic conditions and scenarios to our participants.

Research on driving styles has been extended to autonomous cars in two forms. One body of work includes exploratory studies on understanding how explicitly-defined driving styles relate to comfort (119). The second body of work encompasses research on ways to teach an autonomous car how to drive from human demonstrations (4; 141; 79; 124). Both these groups assume that an autonomous car should learn their own user's driving style or driving behavior. But this assumption may not be true. In fact, research shows that people prefer slower speeds when not in control of the driving (70). Since autonomous car is an instantiation of not being in control of driving, we design the following two-stage user study to test this assumption.

3.2 Methods

3.2.1 Hypothesis

Because being a passenger is a different experience than being a driver, we hypothesize that:
H. *Users of autonomous cars prefer a driving style that is significantly different than their own.*

3.2.2 Study Design

In order to test our hypothesis, we leverage a driving simulator, and let users experience and evaluate autonomous cars with different driving styles, including their own style (without their knowledge).

We conducted a study in two parts. In the first part we collected driving data of participants in a simulation environment, so that we could let them experience their own style in the second part of the study.

3.2.3 Manipulated Variables

We manipulated the driving styles of autonomous cars at four levels of defensiveness: ***aggressive***, ***defensive***, ***own style***, and a ***distractor style*** (a different participant's style). Users did not know if any of the styles were their own. Likewise, we also consciously avoided the use of the phrase "driving style" anytime during the studies, as well as, in the pre-study screening.

We define the *defensiveness* of the style objectively, as a function of several driving features (e.g., distance to other cars – the larger the distance, the more defensive the driving). We use features informed by existing literature. We describe them in Sec. 3.2.6.

We created the aggressive and the defensive styles of driving by demonstration, and then validated these styles using our driving features (see our Manipulation Check Sec. 3.2.7).

3.2.4 Simulator and Driving Tasks

We conducted both parts of the study in a simulation environment. Our simulation environment consisted of a standard classroom projection screen and table in front of the screen fitted with Logitech G920 steering wheel, brake, and gas pedal. We used the OpenDS driving simulation software (99) for running each of the driving simulations. The simulation platform was set up on a standard PC augmented with NVIDIA GeForce GTX 1070 and was hidden from the participants' view.

In the first part, the participants drove on a 9.6 mile long test track that consisted of 14 different driving tasks designed using the City Engine software (Fig.3.2).

We define a *driving task* as a sequence of driving maneuvers in response to specific traffic conditions. For each task there are two to three simulated traffic conditions that resemble everyday traffic, so as to elicit natural driving behavior from the participant.

In the second part of the study, the participants experienced 6 of these 14 tasks, each performed by autonomous cars of four different styles.

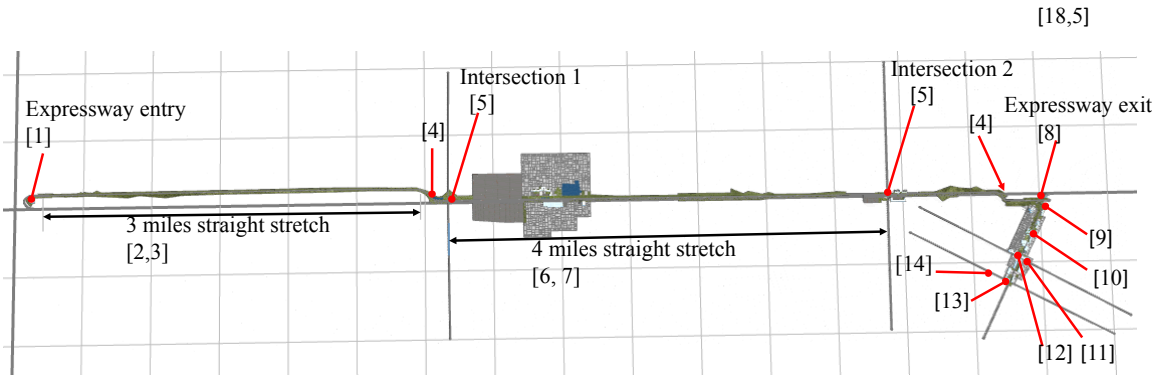
3.2.5 Procedure

Before the driving session in part one of the study, we familiarized participants to the driving simulator. We asked each participant to practice on two different test tracks until they felt that they were driving as they would in their everyday driving. The first track had several traffic signals and turns, and second one was on congested city roads with several traffic cars. Their driving was assisted by a voice navigation. There were road signs for speed change zone, speed limit, sharp turns, entry to expressway and exit from expressway. We instructed the participants to drive as they would on actual roads and to treat the speed limits the way they would in their usual driving. This practice session lasted 5-10 minutes for each participant.

Participants then began the first part of the study, which consisted of 15-20 minutes of driving along the 14 tasks-test track, followed by a 10 minute interview.

In the second part of the study, the autonomous cars performed six tasks (combined into four test tasks) from this list with the participant as a passenger, shown in bold letters on the list in Fig.3.2. To simplify, we combined the second and the third tasks in the list, i.e., lead car slows down forcing lane change and merge back to right lane into a single test task, which we refer to as Task 1. Likewise, we combined the sixth and the seventh tasks into a single test task, called Task 2 in the rest of the chapter. Thus, each autonomous car performed four test tasks in total. Two of the test tasks were on the expressway and lasted approximately 4 minutes for each style and the other two tasks on the inner city roads were shorter than 2 minutes.

After the participants had driven in an autonomous car of each driving style for each of the test tasks, we conducted a short interview-based survey with each



1. Merge onto expressway between two cars
2. **Lead car slow down forcing lane change**
3. **Merge back to right lane**
4. < 90 degrees Turn negotiation on expressway
5. Stop light at intersection on expressway
6. **Slow lead car in right lane forcing lane change when another car is approaching fast in the destination lane**
7. **Merge back into right lane with a continuous traffic moving at constant gap and constant speed**
8. Expressway exit
9. Sharp right turn no signal
10. Left turn on green light
11. Right turn at stop light
12. **Left turn at stoplight and yield at green**
13. **Right turn on green light**
14. Park along curb

Figure 3.2: Designed track: Tasks (shown in the list below the figure) are indicated in square brackets. Total road stretch is 9.6 miles.

participant.

3.2.6 Dependent Measures

Perceived similarity to real driving. In the first part of the study we conducted a post-driving open-ended interview with the participants to understand whether the manual driving in the simulation environment resembled their everyday driving. We asked three questions in this interview, each followed by a request for more elaboration. We asked the following questions in the interview:

1. Did you enjoy the drive?
2. Are there any positive or negative aspects of the simulation environment, the driving controls and the traffic conditions that you would like to mention?
3. On a scale of +3 to -3 (11), please rate how similar or different is this experience from your daily driving?

Open-ended responses. In the second part we asked each participant to think aloud about their emotions and feelings as they were experiencing autonomous driving.

Main subjective measures: Preference and perceived similarity to own style. After a participant had experienced each autonomous style for a given task, we conducted an interview-based survey. We asked the participants to rate each style of driving for *comfort*, *safety*, *preference for everyday use*, and *similarity with their own driving* on 7 point Likert scale.

Main objective measures: Driving style features and overall defensiveness. We measured the user's style quantitatively using task specific driving features, derived from existing literature. We carefully considered the contexts and subject demographics of each of these existing studies to ensure as much similarity in the context as possible with our study.

For car following, lane changing, and return to preferred lane, we selected the features described by Lee *et al.* in "A Comprehensive Examination of Naturalistic Lane-Changes" (84). This study analyzed the largest naturalistic lane change dataset and specifically labelled lane change data resulting from the slowing down of the leading car. The speed range of 45 mph to 55 mph matches our driving conditions. Their dataset consisted 8667 lane changes over 23,949 miles of driving from 16 commuters of age group 20 to 60. They studied car following, lane changing, and return of preferred lane in terms of distance, time to collision, and relative speed classified by severity and urgency of lane change.

The features for tasks like turning at the intersection with a green light or stop light were derived from our preliminary interview with the participants and from Hong *et al.* (69) and Banovic *et al.* (18).

Table 3.1 summarizes all the features for the four driving test tasks. We used *mean distance to lead car*, *mean time headway*, *time headway during lane change*, and *distance headway during lane change* as features for Task 1 and Task 2. Task 1 had an extra feature *distance headway merge back* for scoring the merge back behavior to the right lane.

Task 3 consisted of two sub-tasks (approaching intersection at a stop light and then making a left turn at green ball). We characterized this task with 5 features: *Braking Distance from the intersection*, *Average speed for 20 meters before intersection*, *Time To Stop*, *Speed at the intersection*, and *Maximum turn speed*.

Task 4 constituted approaching intersection at green ball and then turning right without stopping. The features for this tasks are *Speed at the intersection* and *Maximum Turn Speed*.

We objectively measured a participant's overall driving style in terms of a ***Defensiveness Score***. We first normalized the feature values across participants for each feature irrespective of the task. We calculated a *Defensiveness Score* for each participant and for each task as the average over all the normalized feature values for that participant and task. We then computed an *Aggregate Defensiveness Score* for each participant by averaging their scores across the four test tasks.

3.2.7 Manipulation Check

We performed a manipulation check on our aggressive and defensive driving styles. We measured the aggregate defensiveness score for each style, plotted on the bottom right of Fig.3.3. We found that indeed, the aggressive style was less defensive than the defensive style (lower defensiveness score). We found that 86.67 % of the users' styles scored higher than the aggressive style, and lower than the defensive style. This

Features	Definitions
Mean Distance to Lead car	During car following (with 200 meters distance) the average distance between middle of the driver car and the lead car.
Mean Time Headway	During car following (with 200 meters distance) average time headway, defined as ratio of Distance headway and speed of the driver car.
Time Headway during Lane change	Distance headway divided by the speed of the driver car during lane change.
Distance Headway during Lane change	Distance between the middle of the driver car and the lead car during lane change
Distance Headway Merge Back	This is the same as Distance Headway during lane change except measured in between driver car and the following car in the destination lane.
Braking Distance from the Intersection	The distance from the intersection at which a person starts applying brakes.
Time To Stop	Braking distance divided by the speed of the car right before brake is applied.
Maximum Turn Speed	Maximum speed of the driver car over a time window during a left turn or a right turn.
Speed at the Intersection	Instantaneous speed at the intersection.
Average Speed for 20 meters before Intersection	This is the speed of the driver car averaged over a distance range of 20 meters from the intersection.

Table 3.1: Features for style classification

suggests that the two reference driving styles created by demonstrations resulted in meaningful representations of aggressive and defensive driving.

3.2.8 Participants

Subject Allocation. We opted for a within-subjects allocation because the participants needed to choose a preferred style out of the set of available ones. We randomized the order of the conditions.

Demographics. We recruited 15 participants consisting of a mix of graduate students and undergraduate students. Before the study we sent out a screening form to each participant in order to ensure a wide distribution of demographics, driving experience and perceived driving behaviors of the participants. We also checked for a valid driving license. 3 of our participants were 30 to 31 years old, the rest of the participants were 18-24 years old.

The mean driving experience of the participants was 5.46 years with a standard deviation of 4.5 years. Participants had driven an average 214 miles with a standard deviation of 188 miles on the week before they filled out the screening form.

We asked the participants to give us some information about their perceived driving behavior using the following questions: 1. Please rate if you consider yourself a conservative or an adventurous driver on a 7-point scale, 1 being *conservative* and 7 being *adventurous*. 2. Please rate on a 7-point scale what you like about driving, 1 being *joy of motion* (like feeling the force as you accelerate) and 7 being *comfort of steadiness*. You may like some of both. 3. Rate on a 7-point scale if you think you vary your driving by road conditions, traffic and time availability, 1 being *vary always* and 7 being *I don't vary at all*. 4. Please rate your driving experience from somewhat experienced to very skillful. The purpose of these questions was to acquire some information about the participants' driving styles without explicitly using the term style or in other words give away the original goal of the study.

Approximately 46 % of the participants considered themselves well experienced in driving, and 20 % considered themselves experienced. The rest were equally distributed between somewhat experienced to very skillful. The mean score for perceived conservative-adventurous driving behavior was 3.6. Most of the participants considered themselves to be in the middle of the spectrum. Only one participant considered himself to be conservative. More participants preferred comfort and steadiness over joy of driving, the average rating being 4.46. The mean rating for variation of driving style in response to environment and traffic was 3, which means most participants believed that they alter their driving behavior according to traffic.

3.2.9 External Validity and Controlling for Confounds

Driving environment. We used a simulator and not real autonomous cars. However, we designed a simulation track and traffic conditions so as to elicit natural driving responses. We also collected participant feedback in the first part of the study on the simulation environment and how their driving behavior in the simulated track related to their actual driving behavior.

Masking own style. One of the major challenges of this work was to ensure that a participant could not recognize his or her driving style from simulation peculiarities like scenes, traffic and controls. We wanted the participants to only recognize their driving style based on their traffic maneuvers and actions. We took several steps to camouflage the driving data of a participant in the second part of the study:

- We retained the traffic conditions and route from the first part of the study while changing the surrounding scenes and traffic cars, such that we can replicate the user's driving while removing the bias of familiar environment.
- We let the participant perform approximately 14 driving actions in the first part of the study and picked only some of these tasks for the second part of the study.
- During the second part of the study we presented the tasks in an order different from how they occurred in the manual mode. For example: In the first part, the participants first entered the expressway and performed some driving actions on the expressway and then exited the expressway and performed some more driving maneuvers on the city roads. During the second part, we presented one city road task and one expressway task in an alternate order.
- We presented the four styles for each of the test driving tasks in a randomized order, which made it more difficult to consistently recognize one style.
- We post-processed the users' driving to remove peculiarities, which we explain below.

During our pilot studies we found that due to some peculiarities of simulation environment (over-sensitive steering, less sensitive braking) and the resultant jitter in the driving data, some participants were able to recognize their own driving. For example, a participant mentioned: *"This looks like how I was driving. I had to stop at the intersection because I pressed the brake too early. The brake was tight."* Idiosyncrasies of the simulator led pilot participants to identify their driving behavior in the second part of the study. In order to eliminate these peculiarities of the simulation environment, we changed the brake stiffness and steering sensitivity and presented participants with smoothed version of their data in the second part of the study.

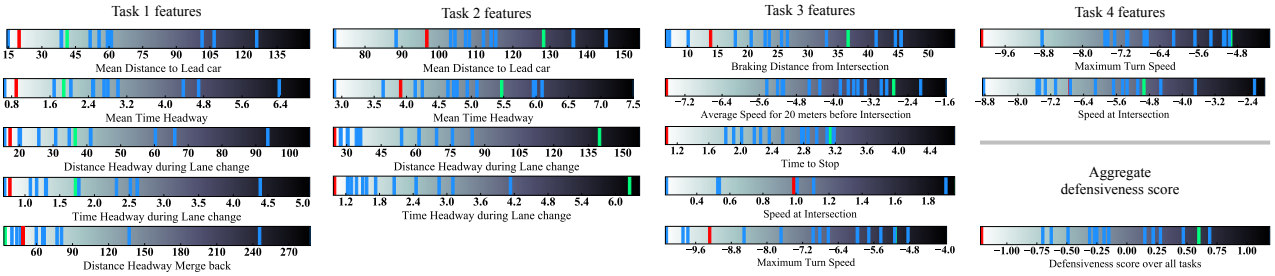


Figure 3.3: Participants' feature distribution

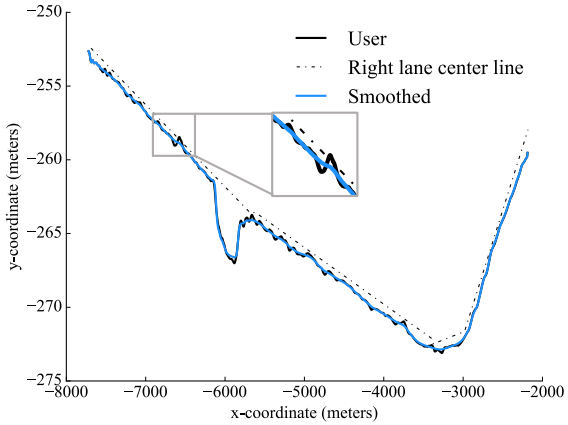


Figure 3.4: Smoothed trajectory compared to original trajectory of task 1 of one participant at 15 % smoothing

3.2.10 Trajectory Smoothing

We filtered the driving trajectories to eliminate idiosyncrasies that make the trajectory instantly recognizable.

We applied a Bilateral Filter (130) to reduce the lateral variance (or equivalently, the variance of the lateral displacements from the center of the lane) of the trajectories. By affecting only the lateral components of the trajectory, this filtering preserves distance between the cars. We applied filtering only to the stretches of the trajectory on the expressway.

Fig.3.4 shows a smoothed trajectory for one participant. It has 15 % lower lateral variance than the original trajectory.

3.3 Results

3.3.1 Simulation Realism

In the first part of the study, in addition to collecting user driving data, we also wanted to ensure that this driving data corresponded to participants' everyday driving

as much as possible. We conducted a post-driving interview, as described in the Dependent Measures subsection (Sec. 3.2.6). Here we present the results of the interview.

The rating mode for similarity between driving on road and driving in our study simulator was +1 on -3 to +3 scale. Four participants gave a rating of +2. Some of their positive comments were: "Not considering the room environment and just looking at the simulation graphics and the car it was pretty much the same environment as real. I would give +3 for surrounding traffic conditions". Other participants said that they felt relaxed in the simulator environment and that they could drive cautiously as they would in real traffic.

One participant who rated the driving experience similarity -2 complained about the lack of motion feedback in the system. This is the same participant who gave high rating for *joy of motion* in the screening question. However, no other participant had the same concern and got well-adjusted to the simulation environment.

Most of the participants who rated +1 to -1 found steering re-centering or brake insensitivity difficult. We also received quite opposite feedback from two participants when they compared their everyday driving to the simulator driving. For example, one participant mentioned "It felt real. It was something I could get used to after driving a while. The gas and brakes were more sensitive than my car". Another participant felt that the brakes were excellent, different from regular car.

One participant reported that she was so immersed after driving for a while, that she caught herself turning her head back to check for oncoming traffic in the destination lane. We found that participants with one or less years of driving experience could not use the simulation environment properly. Overall, the ratings and the comments supported that the simulator conditions are not *too* far from real conditions.

3.3.2 Feature Distribution for Participant Styles

We define driving style in terms of features mentioned in the Sec. 3.2.

Fig.3.3 shows, for each task, feature, and participant, what the participant's feature value was for that task (blue marks). The figure also shows the aggressive style values in red and defensive style values in green.

Higher negative values correspond to more aggressive behavior. All the feature values are arranged from aggressive on the left to defensive on the right. However, for features like speed where lower values mean more defensive we show and use the negation of these features.

The bottommost plot to the right shows the aggregate defensiveness score. This score is derived from the normalized feature values. 60 % of the participants are within 0.75 standard deviation aggressive and 40 % within 0.75 standard deviation defensive. Only two of the participants were more defensive than the autonomous defensive car, one of them being very close to the defensive car in the score.

> *When looking at the aggregate defensiveness, most participants lie between the aggressive and defensive styles.*

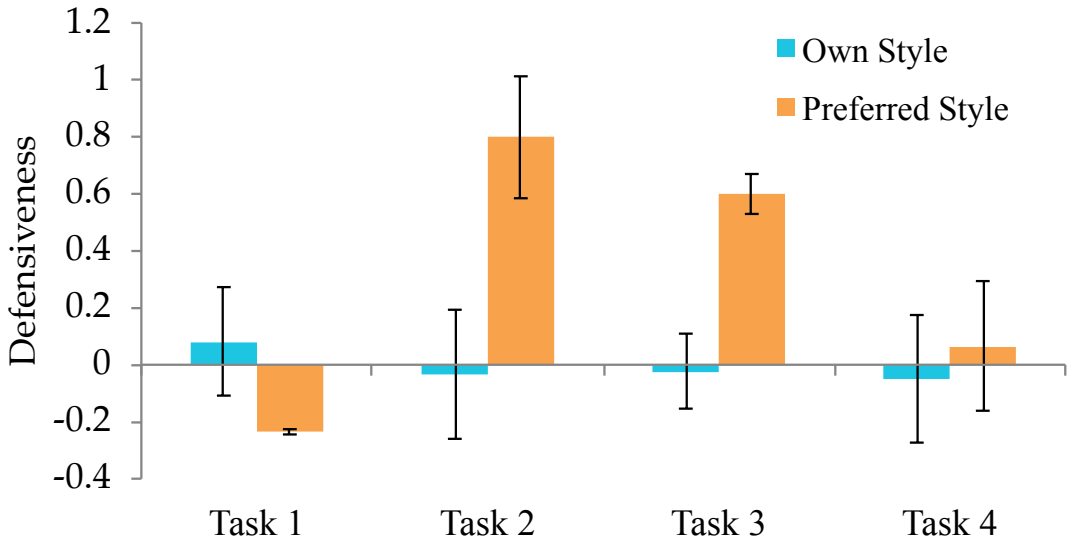


Figure 3.5: Mean Defensiveness Score Across Participants. The corresponding scores of aggressive and defensive autonomous cars are Task 1: (-0.768, -0.222) Task 2: (-0.885, 1.325), Task 3: (-1.82, 0.766) and Task4: (-1.49,0.72).

There are, however, exceptions, but for particular features in particular tasks. For task 1, several participants were more defensive than the defensive autonomous car. For the last feature of task 1, Distance Headway Merge Back, the aggressive car was not as aggressive as several participants and even our defensive car. In task 2, the aggressive and the defensive autonomous cars enclosed a middle section of the spectrum for Mean Distance Headway and Mean Time Headway. In other words, several participants were more aggressive and more defensive than the aggressive and defensive autonomous cars respectively. This is because these features were measured during car following over a long time span and are expected to have wider distributions than features characterizing instantaneous actions.

3.3.3 Preferred Style in Relation to Own Style

We asked participants to rate how much they would prefer driving with each style, for each task. We refer to the highest rated style(s) as the participant's preferred style(s).

Our main finding is that overall, users preferred a different style than their own. A total of 9 out of 15 participants preferred a different style than their own on at least one of the tasks. A matched pairs t-test comparing actual and preferred defensiveness score showed a significant difference ($t(1, 60) = -2.58$, $p = .0121$), supporting our hypothesis. Here, whenever a user's highest rating was for multiple styles as opposed to a single one, we included each preferred style as a data point.

Overall, people prefer a significantly more defensive style than their own.

We also investigated how this breaks down by task, and only found significant effects on the 2^{nd} and 3^{rd} tasks. See Fig.3.5 for comparison between average preferred style and own style of our participants for each of the four tasks. For task 1 we note that several participants were more defensive than other autonomous styles presented to them. However, they still preferred our defensive style, which explains that the average choice was more aggressive than the participants' own style.

Interestingly, some participants did not perceive the extra defensive nature of their own style in task 1 positively. One participant mentioned about their own style that "In this one I felt like we gave a lot of room, more than I would have probably." (ironically, since they did *exactly* that). Two other participants made similar comments about their own lane changing behavior. Besides, a few participants also considered driving features beyond the ones we accounted for.

For task 2 and task 3 the defensive autonomous car was more aggressive than only none to three participants across all features and it was more defensive than the rest of the population by a major margin, in features like Distance Headway and Time Headway During Lane Change.

The task had a significant effect on the difference ($F(3, 58) = 4.13$, $p = .0101$), suggesting that people's preferences for a driving style are not consistent, but rather *change based on the context.* This motivates future research on predicting the desired driving style not just based on the individual, but also based on the current driving context.

3.3.4 Perceived Own Style in Relation to Actual Own Style

We also asked participants to rate each style in terms of similarity to their own. From this, we learned what participants *perceived* their own style to be.

We found that even though participants did not pick their *actual* style as their preferred (Sec. 3.3.3), participants did tend to prefer their *perceived* style. On each task, between 80 and 93% of participants opted for the same style as the one they *thought* was the closest to their own (and sometimes rated other styles as well as equally good). We found a significant correlation between the perceived own and preferred styles, $r(58) = .86$, $p < .0001$. Fig.3.6 shows a scatter plot of preferred style by perceived style, with many points on the diagonal representing users who preferred driving in the style they thought was (closest to) their own.

However, even though the majority participants thought that they were picking their own style, they really were not. A total of 46 to 67% participants on each task did not correctly identify their actual own style, and the correlation between perceived and actual defensiveness score was only $r(56) = .40$ across tasks. Fig.3.7 paints a different picture from Fig.3.6: it plots the perceived style against the *actual* own style, showing many off-diagonal points, representing users who did not correctly identify their style.

In task 1 we see that several participants perceived themselves to be slightly more

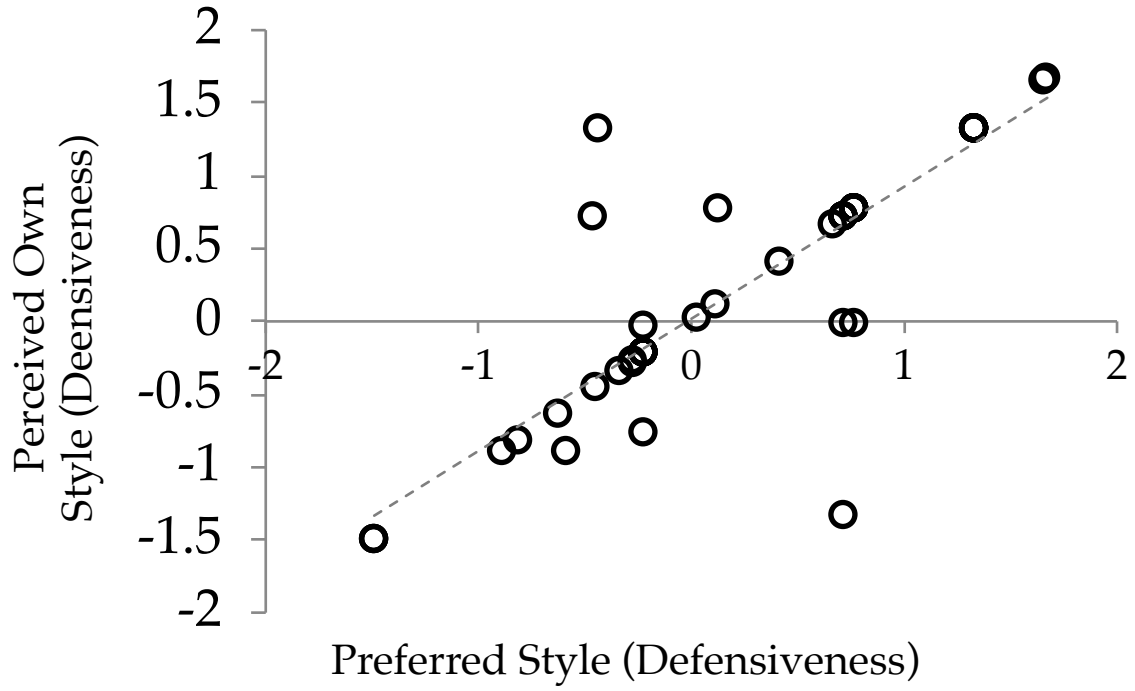


Figure 3.6: Scatter plot showing correlation between the style that users *thought* was their own and the style that they chose as their preferred.

aggressive irrespective of their actual style. Likewise, both for task 2 and task 3 several participants perceived themselves to be more defensive irrespective of their actual style.

> *Participants tended to prefer the style that they thought was their own, but in fact that style had little correlation to their actual own style.*

3.4 Chapter Summary

In this chapter, we hypothesized that users of future autonomous cars would prefer a driving style that is significantly different than their own. We conducted a user study in a driving simulator to test our hypothesis. We found that users preferred a more defensive style than their own. This echoes the finding from prior work (70) that when people are not in control of the driving they prefer lower speeds – autonomous cars are one instantiation of not being in control of the driving. Interestingly, over 80% of users preferred the style that they *thought* was their own, but many times

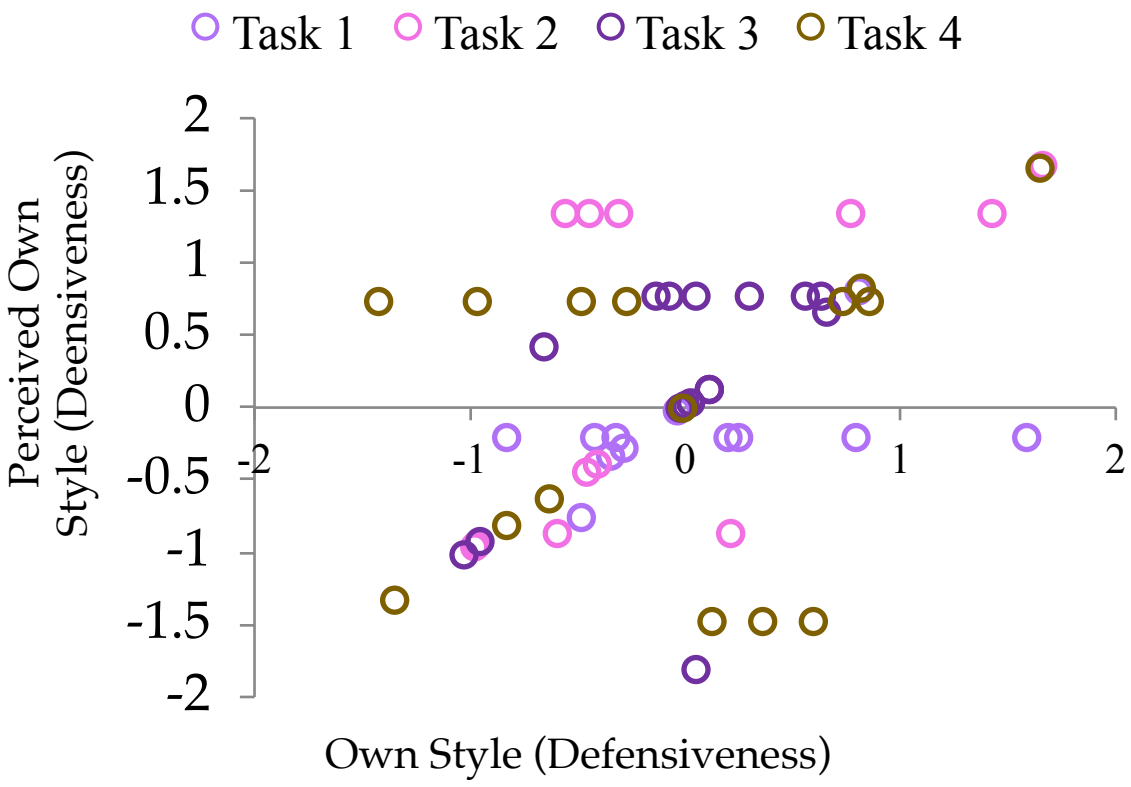


Figure 3.7: Scatter plot showing little correlation between own style and perceived own style: users did not tend to identify their own style correctly. as evidenced by the off-diagonal points.

they were incorrect in identifying their own style. These results open the door for learning what the user's preferred style will be, but caution against getting driving demonstration from the user, since people can drive like they do, not like they want to be driven. Our work is limited in the following ways:

- **Limited driving style features**. Following the most common conventions, we have only characterized style in terms of defensiveness. We also inherited from previous studies the feature choices in defining driving styles.
- **Limited driving style choices**. We presented participants with limited options along the spectrum of defensiveness and found that they preferred a style more defensive than their own. However, we did not learn the style they actually desired, only the best out of our few options. An natural extension of this work would be to test more diverse feature choices and driving style representations in a higher fidelity setting. It is also worthwhile exploring what features users' consider when they evaluate autonomous driving styles. These experiments will provide a comprehensive evaluation of the study presented in this chapter.

- **Limited fidelity of simulation environment**. Our simulation environment does not provide motion feedback, which may limit the users' perception of speed. Although the interview results validated that participants' perception of the driving styles are sufficient, experiment results in a higher fidelity simulation environment might be more accurate.

Despite the above limitations, this work serves as a preliminary proof of concept that driving preferences, when it comes to autonomous cars, is not the same a driving demonstrations. While prior research have shown that it is possible to learn driving preferences without demonstration data (110), *this is the first study with real users which empirically supports demonstration-free approaches for personalizing autonomous driving.* Going forward in this thesis, we explore learning autonomous driving from alternative forms of human guidance.

Chapter 4

Preference Model

Learning from Demonstrations have been extensively used in learning driving styles. Both Behavioral Cloning and Maximum Entropy IRL *passively* that learn a driving policy and a reward function representing driver intent respectively have an implicit assumption - that autonomous cars should drive like individual users (see Chapter.2). However, we learned in the previous chapter that it may not be sufficient to *personalize autonomous driving* from individual user demonstrations, because *end-user preferences do not match individual demonstrations.*

> *We propose that autonomous cars can learn driving preferences from richer forms of human guidance in absence of demonstrations.*

To validate this proposal we implement *interactive* algorithms that enable the robot to learn user preferences by making queries in a two-player game setting. Here we first formally introduce the problem domain. Similar to Chapter.3 and prior work on learning driving style with IRL (see Chapter.2), we define preferred driving style as a linear combination of features parameterized by a MIXTURE OF REWARD FUNCTIONS. We present a generic preference model that can apply to static preferences as well as dynamic preferences in non-stationary environment. We close this chapter with an introduction to the simulation environment that we used to learn this preference model.

4.1 Problem Domain

We tackle the following problem domain in this thesis. We denote the ego car that should match individual human preference $\wp$ as $\mathcal{H}$. The environment of the ego-car consists of other agents or traffic participants like manual cars, autonomous cars, bicycles and pedestrians. We denote these agents by $\mathcal{E} = \{\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_P\}$. These agents can act differently at different times. For example, in case of driving, some cars aggressively swerve through the traffic and others may follow a more cooperative strategy allowing other cars to merge smoothly.

We model the environment as a fully-observable dynamical system. For driving, the continuous state of the system $x \in X$ includes the positions and the velocities of $\mathcal{H}$ and $\mathcal{E}$. The state of the system changes based on actions of all the agents through a function f.

$$x^{t+1} = f(x^t, u_H^t, u_E^t) \tag{4.1}$$

where u_E are the actions of $\mathcal{E}$, which may affect $\wp$ and in turn the actions u_H of $\mathcal{H}$. We define a finite trajectory $\xi \in \Xi$ as a sequence of continuous state-action pairs $\xi = (x^0, u_H^0, u_E^0, \ldots, x^T, u_H^T, u_E^T)$ over a finite horizon T, and Ξ is the set of all feasible trajectories that satisfy the dynamics of the system. We assume a features function $\phi : \Xi \to \mathbb{R}^d$ that maps every trajectory to a d-dimensional feature space which includes features for lane keeping, distance to other cars, weaving for instance.

4.2 The Human-Robot Interaction Formulation

For the robots to act around and according to human preferences it is important for them to know human preferences and predict what people think and how people respond to their actions. Autonomous cars that will drive for human convenience and comfort should have models of human intent, anticipation, reaction, *preference* etc., which it can then blend into its control policies. A recent approach is to adopt methods from human-robot interaction to autonomous driving (115; 116; 72; 110; 56; 25; 82; 128; 127). We follow a similar approach and treat autonomous car as a robot that learns about human preference by interacting with people.

We present the task of personalizing driving style as a human-robot interaction problem, where the human $\mathcal{H}'$ is the user and the autonomous car $\mathcal{H}$ is the robot that learns to match $\mathcal{H}'$'s preferences for driving style *interactively*. We design the interaction between $\mathcal{H}'$ and $\mathcal{H}$ as a *two-player game* in a partially observable setting similar to Partially Observable Markov Decision Processes (POMDP).

4.2.1 Interaction as a two-player game

We can define human-robot interaction as a 2-player game between a human $\mathcal{H}'$ and a robot $\mathcal{H}$. There is a state space $\mathcal{S}$ with states s. s contains the physical states x which originally contains the robot's states x_H and the environment agents' states x_E. For the purpose of this game we add one more component to x: x'_H, the human's physical state in the game. s also contains the hidden internal states of the human like *intent, reward function, emotions* etc. which are integral components of interaction with a human. Each agent can take actions. They do so by optimizing some reward function, potentially different for each agent.

$$r'_H(x, q_H, a'_H; w'_H) \tag{4.2}$$

for the human and

$$r_H(x, q_H, a'_H; w_H) \tag{4.3}$$

for the robot, each with parameter w. $\mathcal{H}'$ and $\mathcal{H}$ may not know each other's w. T' is the time horizon of the game and $\mathbf{q}_H$ and $\mathbf{a}'_H$ are action sequences of $\mathcal{H}$ and $\mathcal{R}$ over T'. To not confuse with the previous notations, in the previous section robot action refers to control inputs and in the context of the game robot action mean making a query to the human. Likewise the time horizons refer to trajectory length and number of iterations of the game respectively. The action sequences in the game form the policy pairs. The cumulative reward is the sum of individual rewards over T'. One way to model this problem is consider humans as rationally solving the game. The problem is however computationally intractable and if $r_H \neq r'_H$ then there are several equilibria. This is also not a good model of human behavior (116). Humans do not solve games in everyday decision making. Several approximations have been introduced for this problem in the human-robot interaction domain (49) that reduce it to a tractable problem. Fortunately, in our case the interaction game has a single equilibrium, as $r_H = r'_H$, as we explain next.

4.2.2 Approximations

In this game the actions amount to the robot asking choice questions to $\mathcal{H}'$ intelligently and the human providing answer to these questions. r'_H represents the human preference for how $\mathcal{H}$ should act in a given situation (a combination of initial world states x_0 including the robot and a trajectories of other agents in the world.) $\mathcal{H}'$ is noisily aware of w'_H. The robot doesn't observe know r'_H. It has a model of human as being noisily rational, which serves as a map from r'_H to actions. It, therefore treats human actions, in this case, human answers as observations about the hidden internal state r'_H. We describe the model of human behavior in the Chapter.5. $\mathcal{H}$ starts out with an initial belief $p(w'_H)$ over w'_H. At every t' it asks a choice question q'_t to $\mathcal{H}'$, who in turn noisily picks the choice optimal with respect to true preference w'_H. Both the agents perform a local optimization for action selection. $\mathcal{H}$ uses this answer to update its belief as in Bayesian Inverse Reinforcement Learning (104; 110). Moreover, at every t', w_H is equal to the best estimate of w'_H. $\mathcal{H}$ leverages its current knowledge of user's ability to answer the questions and w_H to intelligently ask the next choice question. In our formulation, the human is unaware of this mechanism i.e. $\mathcal{H}'$ doesn't observe r_H. Alternative formulations explicitly consider humans as being aware that the robot is learning (60; 93). The optimal solutions to this game maximize the human reward and since the ultimate pay-off of the robot is equal to the true human preference, the problem ends up having a single Nash Equilibrium. The above *interaction* problem can also solved as a single actor POMDP as shown by (60). The game continues over a fixed number of iterations T' which is selected based on how many questions $\mathcal{H}$ can answer reliably without cognitive overload. Since $w'_H = w_H$, for the rest of this thesis we use just w to denote reward function representing the user's preference.

4.3 Preference model

We propose a generic model of human preference that can be applied to both static and dynamic preferences, i.e. when the preference mode changes in response to the dynamics of the environment, for example sudden traffic congestion, change in courtesy level of other cars to name a few. We assume that human preference is stable over a period of time defined by a small segment of the trajectory of the ego car and the environment agents and the corresponding states of the world and is characterized by a preference mode. The preference is represented by the sum of rewards over the small segment of trajectory. We encode human preference as a tuple of reward functions parameterized by the preference mode and parameters governing the transitions between the mode-specific reward functions.

Reward functions under known modes. We assume each human has a finite set of modes M and we enumerate each mode such that $M = \{1, 2, \ldots, k\}$ where k is the total number of modes. M_j is the j^{th} element of the set of modes M. We define a user specific reward function parameterized by the mode of the user: $R_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$ where Ξ is the set of all feasible trajectories of the ego car.

Mode transitions. The user changes their mode based on their immediate previous experience i.e. the previous stable period of preference. We consider a stochastic transition and model it in terms of an underlying mode-utility function. This mode-utility function is parameterized by the mode of the user: $V_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$ where Ξ is the set of all feasible trajectories of the ego car. The human quantifies the utility of the current trajectory of the ego car with respect to all the mode-specific utility functions. If the user thinks she would have higher utility with mode M_j, then she transitions to M_j for the next time step. As an example, imagine you are driving in a very calm mood. If someone suddenly cuts in front of you, you would think "if I were aggressive, I could keep a shorter headway with the car in front and the other car would not have been able to cut in front of me", and you also switch to an aggressive mood. It is of course also possible that you keep calm. Therefore, the transition should be stochastic.

Preference Model. Our human-specific preference model is a tuple as:

$$\wp = (R_{M_j}, U_{M_j}) \forall j \in [k] \tag{4.4}$$

The Learning Problem. Throughout the work we assume that R_{M_j} and U_{M_j} are linear functions of features derived from the trajectory of the ego-car defined earlier as ϕ. Therefore, $R_{M_j}(\xi) = w_{M_j}^\top \phi(\xi)$ where $\xi \in \Xi$ and $w \in \mathbb{R}^{d \times k}$ is a user-specific weight matrix, and w_j is the j^{th} column of w, with each column corresponding to a particular mode for a user. Our problem is to learn the parameters w_j and γ_j respectively of the linear functions and $U_{M_j}(\xi) = \gamma_{M_j}^\top \phi(\xi)$ where $\gamma \in \mathbb{R}^{d \times k}$ is another user-specific weight matrix and γ_j is the j^{th} column of γ. Our goal is to learn the tuple (w, γ). Assuming static preference reduces to learning a single reward function w.

Features The feature function ϕ is carefully engineered and has seven dimensions corresponding to seven features derived from the states of the ego-car $\mathcal{H}$. Human

preference $\wp$ governs the trade-off between these features. (see Fig. 6.1).

- $\phi_1 \propto c_1\ exp(-c_2 d_c^2)$ exponentially decreases as the distance to the center of the lane increases where c_1 and c_2 are the appropriate scaling factors
- ϕ_2 penalizes distances closer to the road edges with the penalty being maximum at and outside the road edges and is of the form $\phi_2 = \frac{g.exp(g)+s.exp(s)}{exp(g)+exp(s)}$, where g is a Gaussian function of the distance between $\mathcal{H}$ and the road edges similar to ϕ_1, controlling the penalty within the road boundaries, and s is a sigmoid function controlling the penalty outside the road. The above Softmax function picks the maximum of g and s outside the road as shown in Fig.
- $\phi_3 \propto \psi \cdot \mathbf{n}$ computes alignment between the car's heading and the road where ψ is the heading of $\mathcal{H}$ and $\mathbf{n}$ is a vector normal to the direction of the road
- ϕ_4 computes distance to other cars and is for collision-avoidance. It computes the non-spherical Gaussian over the distance between $\mathcal{H}$ and $\mathcal{E}$ with major axis along the heading of $\mathcal{E}$
- $\phi_5 \propto exp(-c_3.(v - v_{max})^2)$ gently increases with the speed of the car and favors high speed (read efficiency) where c_3 is the appropriate scaling factor
- $\phi_6 \propto c_4\ exp(-c_2 d_c^2)$ incentivizes a preference to be in the right lane
- ϕ_7 disincentivizes reverse motion

4.4 Simulation Environment

We test of algorithms in both simulations and with user study. In both of these we present the simulated users and the real users with queries with comparative trajectory snapshots. The snapshots are generated in a driving simulator that uses a simple point-mass model of the ego-car's dynamics. We define the physical state of the system $\mathbf{x} = [x, y, \psi, v]^T$, where x and y are the coordinates of the vehicle, ψ is the heading and v is the speed. We let $\mathbf{u} = [u_1, u_2]^T$ as the control input, where u_1 is the steering input and u_2 is the acceleration. We denote the friction coefficient as μ. We can write the dynamics model of the ego-car as:

$$[\dot{x}, \dot{y}, \dot{\psi}, \dot{v}] = [v \cdot cos(\psi), v \cdot sin(\psi), v \cdot u_1, u_2 - \mu \cdot v] \tag{4.5}$$

The simulator provides as top-down view of the environment.

Now given this general preference model, we developed a unified algorithmic framework that can learn both static and dynamic preferences using *rich human guidance* that does not involve demonstrations. Now what do we mean by rich human guidance? Here we showed how we treat learning preferences from rich guidance as a human-robot interaction game. In the following chapter we will formally define rich guidance and present our algorithmic framework.

Chapter 5

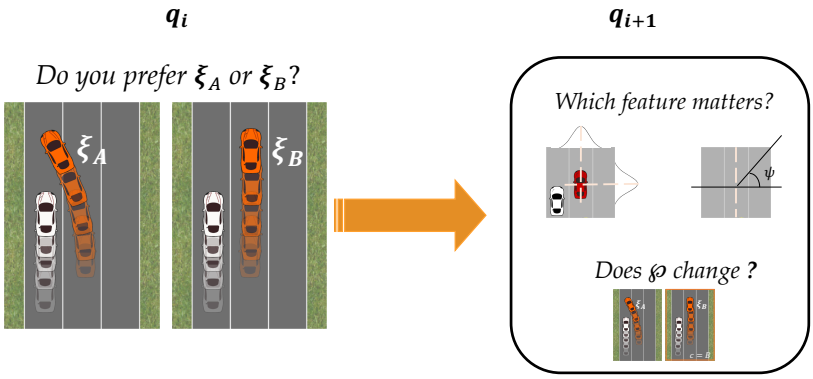
Learning Preference from Richer Queries

In Chapter.4 we presented a rich preference model that can represent both static and dynamic preferences for autonomous driving behavior. Our learning problem, in this thesis, involves learning a distribution over parameters of this model - (w, γ), where w is the weight vector for reward function representing a specific preference mode and γ is a parameter vector governing mode transitions as function of prior driving experience. We proposed that the robot can learn these parameters by seeking richer guidance from human. Now since comparison-based learning has emerged as a useful form of human guidance in learning preferred driving style (see Chapter.2) (5; 76; 73; 7; 30; 45; 74; 66; 43; 48)., we build on this approach and DEVELOP RICHER COMPARISON QUERIES.

5.1 Learning Preference from Rich Human Guidance

Comaprison Queries. In comparison queries, the robot iteratively shows users two possible trajectories (often in the same environment, for the same starting state), and asks which they prefer. For example, in Fig.5.1b. the autonomous car in orange is cutting in front of the white traffic car in option A, whereas in option B, it goes on straight without interacting with the white car. The car asks the user to pick between these two trajectories. It then uses the answer to update its understanding of the reward parameters. In comparison-based learning each answer tells you that one trajectory is better than another. This binary feedback is very little information compared to a demonstration where we directly get the optimal trajectory. In this thesis, we propose a middle ground between comparisons and demonstrations. We leverage several variants of the comparison-queries to collate richer information per query than just one trajectory is better than the other. We refer to these queries as *Rich Queries* as opposed to *Comparison-only* queries.

a. Comparisons are much less informative than demonstration.



b. The middle ground is augmented comparison queries.

Figure 5.1: a. We argue that comparisons are far less informative than demonstrations and can lead to slower learning of a continuous high dimensional reward function. Instead we look for a middle ground between comparisons and demonstrations. b. Our key insight in this thesis is that we can ask people for richer information in the form of *rich queries*: a series of hierarchical sub-queries at least one of which is a comparison query. In this example, q_i is a usual comparison query where user is asked to pick between ξ_A and ξ_B and q_{i+1} is a follow-up query like *which feature of the reward function was the most responsible for your choice?*

Rich Queries. Our insight is that we can extract much richer information from a single comparison query by asking some follow-up queries like emphwhy did you select A? or *would this choice change had the white car behaved differently?* (see q_{i+1} Fig.5.1). We propose *rich query* q as a sequence of connected sub-queries q_i, where the current sub-query q_i forms the context for the future sub-query q_{i+1}. A *rich query* has at least one *rich query* comparison-only query of the form $q_i = (x^0, \xi_A, \xi_B)$, where we ask the user "Which of the two trajectories do you prefer?". Our framework is designed to accommodate several sub-queries of different forms, for example, q_{i+1} could be a follow-up question seeking to understand the user choice in q_i or q_{i+1} could be another comparison-only query that uses q_i as a context. The user's response to q_i is denoted by a_i.

5.2 Unified Algorithmic Framework

We propose a unified framework for treating answers to all forms of sub-queries as observations about true human preference, and its parameters (w, γ) thereof. Our framework is probabilistic. The goal of the learning algorithm is to learn a distribution over (w, γ). Our robot $\mathcal{H}$ starts out with a belief over the parameter space. After receiving all the answers to a query q, $(a_1, a_2, \ldots, a_s)$, we perform a Bayesian update:

$$\begin{aligned} &p(w, \gamma | a_s, a_{s-1}, \ldots, a_1, q_s, q_{s-1}, \ldots, q_0) \\ &\propto p(a_s, a_{s-1}, \ldots, a_1 | w, \gamma, q_s, q_{s-1}, \ldots, q_0) p(w, \gamma) \end{aligned} \tag{5.1}$$

The two key components of the above equation needed to compute the posterior distribution over (w, γ) are the prior $p(w, \gamma)$ and the likelihood function, which is the joint distribution over the answers to all the components of a query conditioned on the query and the parameters: $p(a_s, a_{s-1}, \ldots, a_1 | w, \gamma, q_s, q_{s-1}, \ldots, q_0)$. We draw from the Theory of Mind to develop a probabilistic observation model of how a human would answer any robot query within our framework: the likelihood function. More specifically, we restrict all the subqueries to some form of choice question between several alternatives and use Luce's choice Axiom to model human decision making. We employ a common case of the Luce Choice Axiom, the Boltzmann (or *soft-max*) model of noisy rationality, in which the probability of choice decays exponentially as it's utility decreases in comparison to the competing options.

$$P(a = f) \propto \exp(\beta U_f) \tag{5.2}$$

where $\beta > 0$ is termed as the *rationality coefficient* and quantifies the concentration of choice around the optimum, as $\beta \to \inf$ means a perfectly rational or oracle human who always makes the choice based on the perfect knowledge of the utility of the items. As as $\beta \to 0$ the human responses get closer to random and U_f is the true utility of the f^{th} item. This model represents people's behavior reasonably well and has been adopted in several recent work in human-robot interaction and psychology as we discussed in Chapter.2. That said, there are other models of rational behavior that account for limitations within our rationality (75) or specific biases or errors inherent in our behavior (53; 91) and may also be suitable for modeling preferences when it comes to autonomous driving. We leave the investigation on applicability of these other models to future researchers. As we will see later in this thesis the hyperparameter β plays a significant role in getting the model right.

5.3 Chapter Summary

We want to personalize driving style without using the user's own driving demonstrations. Comparisons have proven to be a useful form of guidance in such scenarios. We

note that comparisons are, however, far less informative than demonstrations. The key insight in this work is that humans can provide richer guidance than just comparisons and robots can actively seek this guidance by making richer queries. Here we defined richer queries as *augmented comparison queries*. The main contribution of this thesis is a unified algorithmic framework that uses models of noisy rationality to treat user answers to all kinds of queries as observations about the true preference. While, our preference model supports represents constant preferences and people's behavior in changing environments, we first design rich queries to learn a static reward function w.

Chapter 6

Comparison-Feature Queries

We introduced the concept of rich queries in Chapter.5, which are augmented comparison queries and argue that we can learn much faster from rich queries than from comparisons alone. In this thesis we explore two different forms of rich queries: one for learning a static preference model or reward function, that is when people's preference for autonomous driving do not change with time or environment, and second type for learning a mixture of reward functions representing preferences that change with non-stationary environment. Here we start out by introducing the former: FEATURE-AUGMENTED COMPARISON QUERIES, where the robot also asks *why*: which *feature* in the reward function was responsible for the preference between the two options (see Fig.6.1). We generalize comparison-based learning to these richer queries by treating feature answers as observations about the true reward parameters. To speed up learning, we derive a rich query selection method that optimizes for gathering as much information as possible from each query. Feature queries have already been used in the context of learning a classifier from labels (103), and here we show their equivalent for learning a reward function from comparisons. We build on prior work that leveraged feature queries in the context of learning skills from demonstration (37) to introduce *combined comparison-feature queries* for reward learning.

6.1 Static Preference

The goal is here to learn a simple model of human preference $\wp$ that doesn't change over time i.e. $k = 1$ in (4.4) . Since each mode in our model is represented by a single reward parameter, our learning problem here reduces to learning a single reward function $w \in \mathbb{R}^d$ which parameterizes the reward model as $R = w^\top \phi$ and ϕ is the d-dimensional feature function.

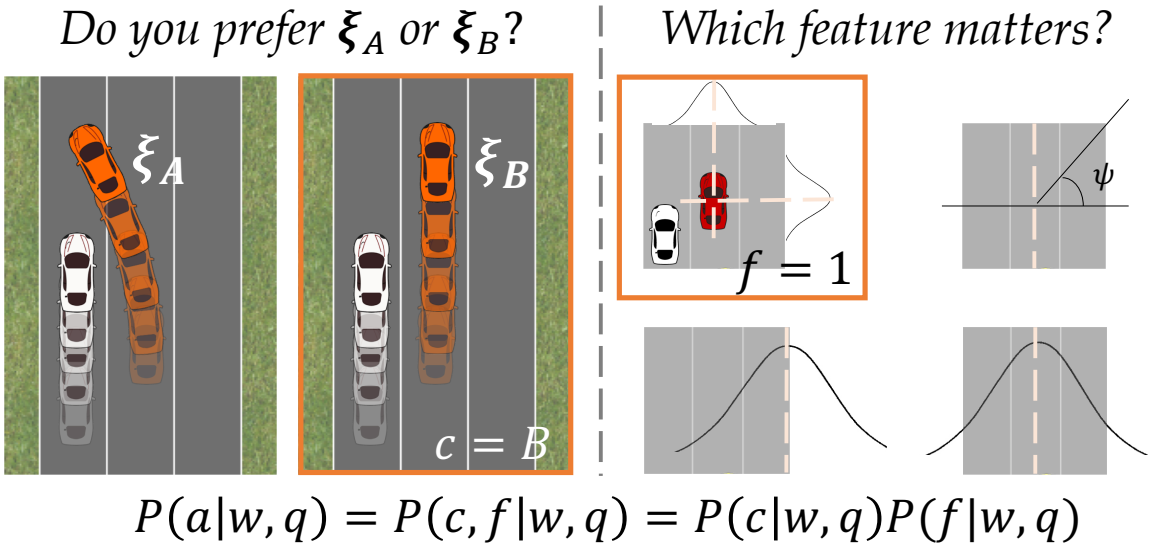


Figure 6.1: We augment comparisons between trajectories q_1 with feature queries q_2 that are responsible for these preferences. We assume a static preference model $\wp$ parameterized by a single reward function w and introduce a probabilistic model $P(c, f|w, q_1, q_2)$ of how a human might answer these queries given the desired objective function for the robot w, and use this model to actively generate rich queries that learn w.

6.2 Comparison-Feature Queries

Following the definition of queries q in Chapter.5 as a sequence of several sub-queries, we introduce comparison-feature query as a sequence of two sub-queries: the first one is a comparison query of the form $q_1 = (x^0, \xi_A, \xi_B)$ and the second sub-query q_2 is a follow-up *why*. We present *why* as choice question, where we ask the users to pick one of the feature $\phi_f \in \phi$ in the reward function the most responsible for their choice in q_1. "Which feature is most responsible for the difference in your preference between these two trajectories?". For ease of understanding we rename answer to the comparison part of the query q_1 as c and answer to the feature question q_2 as f in this chapter, where $c \in A, B$: A if ξ_A is preferred, B otherwise and f is a feature ID.

6.3 Active Preference Learning from Comparison-Feature Queries

6.3.1 The Learning Problem

The goal here is to learn a distribution over a single reward function w. Our robot $\mathcal{H}$ starts out with a belief over the parameter space. After receiving all the answers to a query q: (c, f), we perform a Bayesian update:

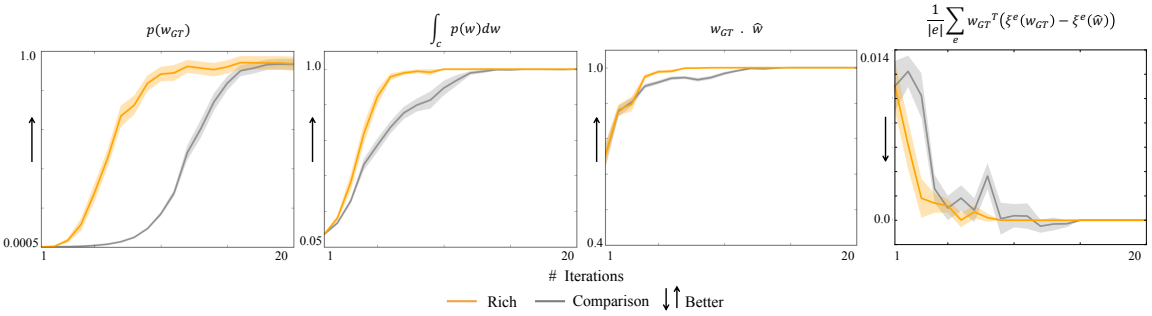


Figure 6.2: Four dependent measures:probability of the ground truth reward function, integral over rewards similar to the ground truth, dot product between the ground truth and the learned weights, and regret, averaged across 20 true weights for each algorithm, show that compared to comparison-only queries, rich queries learn the true reward much faster, especially, when people are fully aware of their true reward.

$$p(w|c, f, q_2, q_1) \propto p(c, f|w, q_2, q_1)p(w) \tag{6.1}$$

6.3.2 Observation Models

We propose a probabilistic observation model for the likelihood function $p(c, f|w, q_2, q_1)$. First of all we assume that c and f are conditionally independent given the reward. Therefore:

$$p(c, f|w, q_2, q_1) = P(c|w, q_1)P(f|w, q_2) \tag{6.2}$$

This is a design choice, and it means that people don't have to get the comparison correct to tell us the main feature that matters in the comparison. Alternately, we could condition the feature on the comparison answer, and model a feature as only probable when the sign of the weighted feature difference, not just the absolute value, is consistent with the weight vector. For each part we assume that people are noisily rational and probability of picking an answer follows the Boltzmann model of noisy rationality as in Chapter.5 in (5.2), i.e. we replace the utility term U_f with the true reward of the user associated with the options ξ_A, ξ_B. Applying Boltzmann model to the comparison answers we get:

$$P(c = A|w, q_1) = \frac{exp(\beta^c R(x^0, \xi_A))}{exp(\beta^c R(x^0, \xi_A)) + exp(\beta^c R(x^0, \xi_B))} \tag{6.3}$$

$$P(c = B|w, q_1) = \frac{exp(\beta^c R(x^0, \xi_B))}{exp(\beta^c R(x^0, \xi_A)) + exp(\beta^c R(x^0, \xi_B))} \tag{6.4}$$

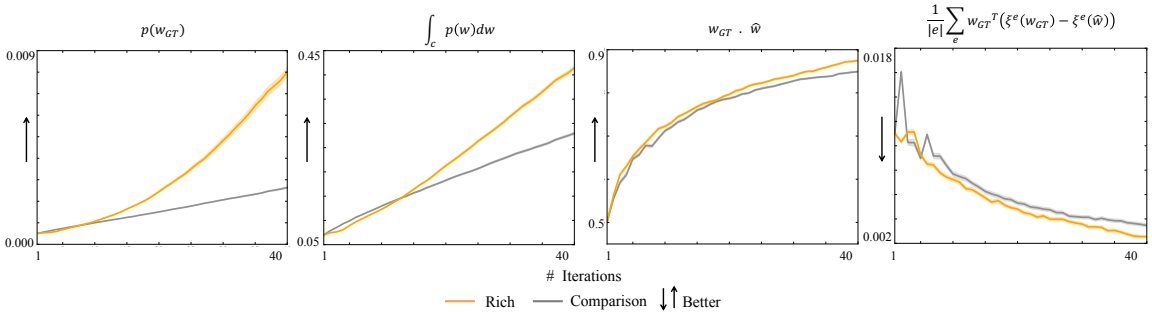


Figure 6.3: Four dependent measures: probability of the ground truth reward function, integral over rewards similar to the ground truth, dot product between the ground truth and the learned weights, and regret, averaged across 20 ground truth weights. The results show that even when people provide noisy responses to feature queries, the robot still learns better compared to comparison-only queries.

where β^c is the rationality coefficient for the specific case of comparison.

The Boltzmann model for feature ID choice is based on the assumption that people will noisily identify the feature f in a set of d features, which, when combined with how important it is (its weight in w), best accounts for the difference in reward between the two trajectories ξ_A and ξ_B:

$$P(f|w, q_2) = \frac{exp(\beta^f w_f \cdot |\Phi_f(x^0, \xi_A) - \Phi_f(x^0, \xi_B)|)}{\sum_i exp(\beta^f w_i \cdot |\Phi_i(x^0, \xi_A) - \Phi_i(x^0, \xi_B)|)} \tag{6.5}$$

with β^f a rationality coefficient for these types of queries, Φ_f the value of f^{th} feature summed over the trajectory, and w_f its weight in w.

6.3.3 Active Query Selection

So far we have defined our queries, and how to update the robot's belief given the answer to a query. Now, we turn to which queries to make, i.e. how the robot should select queries.

The simplest option for query selection is to draw queries at random from a set. However, we can speed up learning by *actively* selecting queries.

At every step, assuming a discrete space of ws we would ideally select the query that will remove as many ws from the hypothesis space as possible so that we converge faster to a single w^* closest to the true reward function w_{GT}. There are two challenges with this.

The first challenge is that we do not eliminate hypotheses, rather update a belief over them. We thus want to maximize some measure of the change in probability distribution over w. For the purpose of this sub-section we use $a = (c, f)$ as answer to query $q = (q_1, q_2)$. Following (48), we use volume removed: a query q with an answer

a removes the following volume:

$$V(q,a) = \sum_w P(w) - P(w)P(a|w,q) = \mathbb{E}_w[1 - P(a|w,q)] \tag{6.6}$$

where $P(a|w,q) = p(c,f|w,q_2,q_1)$ from (6.2).

The second challenge is that we don't know what answer we will get. However, our current belief induces a probability distribution over answers, so we can choose the query that *in expectation* removes the most volume:

$$\max_q \mathbb{E}_w \sum_a P(a|w,q)V(q,a) \tag{6.7}$$

where the sum over a is over all tuples (c,f).

Note that other measures are also possible, including information gain (reduction in Shannon entropy) (120).

6.3.4 Allowing "I don't know"

Since feature queries are more complex than their comparison counterpart, we also experiment with allowing users to say "I don't know". In that case, we skip the update based on the feature query, and only use the answer to the comparison query for update, i.e. $P(c|w,q_1)$.

More interestingly, we inform the query selection criterion about this option, so that it does not choose queries that will lead to no update from the feature response.

To achieve this, we model *when* a user might say that they don't know as their answer to the feature query. We model this "I don't know answer" as occurring whenever the probability distribution over their answers is too close to uniform, i.e.

$$P(skip|w,q) = \begin{cases} 1 & \frac{1}{d} - \epsilon \leq p(\mathbf{f}|\mathbf{w},\varphi) \leq \frac{1}{d} + \epsilon \ \forall f \in \{1..d\} \\ 0 & otherwise \end{cases} \tag{6.8}$$

with d the number of features and ϵ capturing how similar the contributions of the features need to be with respect to each other for the user to decide to skip the answer.

If the person does skip, then the volume removed is

$$V^{skip}(q,c) = \mathbb{E}_w[1 - P(c|w,q_1)] \tag{6.9}$$

Then the robot selects the next query by optimizing

$$\begin{aligned} \max_q \ \mathbb{E}_w P(skip|w,q) \cdot \sum_c P(c|w,q)V^{skip}(q,c) + \\ (1 - P(skip|w,q)) \cdot \sum_a P(a|w,q)V(q,a) \end{aligned} \tag{6.10}$$

6.4 Hypothesis

We hypothesize that using rich queries (i.e. our queries that seek feature clarification) leads to higher accuracy of the learned reward, given the same budget of queries. We test this hypothesis in simulation, and in a user study.

6.5 Experiments in Simulation

We start with experiments that simulate user responses based on some known true weights w_{GT}.

6.5.1 Experiment Design

Manipulated Factors. We manipulated the type of queries we used for learning, with two levels: *comparison* only queries vs. *rich* queries that ask for a comparison, along with the feature that is important in the difference.

We also manipulated the number of queries that each resulting algorithm gets to make, from 0 to 40.

Finally, we repeated the experiment for 20 possible ground truth weights.

Dependent Measures. We evaluated the accuracy of the learned probability distribution over reward parameters w in four ways:

- The learned probability of the true weights:

  $$P(w_{GT})$$

  The higher the probability assigned to the ground truth, the more accurate the learned distribution is.

- The integral of the probability of all "close" weights, i.e. all weights that have high dot product with the ground truth weight:

  $$\int_C P(w)dw$$

  with $C = \{w|w \cdot w_{GT} > 0.9\}$. This is a more robust version of the first measure, where we capture not only the probability of the ground truth, but also what the probability distribution does around the ground truth.

- The dot product between the ground truth weights and the learned weights:

  $$w_{GT} \cdot \hat{w}$$

  with $\hat{w} = \arg\max_w P(w)$. Rather than focusing on just the ground truth, this measure gets at how good the weights that we learned, $\hat{w}$, actually are.

- The regret resulting from converging to a reward function different from the true reward function measured across different test environments:

  $$\frac{1}{|e|} \sum_e w_{GT}^T(\xi^e(\hat{w}) - \xi^e(w_{GT}))$$

  with $\xi^e(\hat{w})$ denoting the optimal trajectory in environment e for the reward function $\hat{w}$ and $\xi^e(w_{GT})$ denoting the optimal trajectory for the true reward

function. In other words, we measure the difference in true reward between the *actually* optimal trajectory, and the *learned* optimal trajectory. This captures what happens when we go out into the world and optimize the reward that we learned, i.e. to what extent do we match the user's actual preference.

6.5.2 Problem Domain

We focus on learning driving preferences for a single ego-car $\mathcal{H}$. The dynamics and the environment are as described in the Chapter.5. Our environment consists of a single environment agent $\mathcal{E}$ that interacts with $\mathcal{H}$. We assume that the trajectory of $\mathcal{E}$ is predefined and fully observable by $\mathcal{H}$. $\mathcal{H}$ optimizes for the learned w including collision-avoidance with $\mathcal{E}$. w governs the trade-off between the seven driving features defined in Chapter.5, except for the penalty feature for proximity to the edges of the road ϕ_2. For this feature we just use a Gaussian function similar to ϕ_1 that increases closer to edges. However, note that for the rest of the thesis we use the version of ϕ_2 defined originally in Chapter.5 as the Gaussian function equally incentivizes staying away from the edges whether it is on the road outside the road. This is because the Gaussian function decreases symmetrically on either side of road edges.

Queries. Queries consist of an environment and two trajectories for that environment shown in a simulator (see Fig. 6.1). The environment consists of an initial state for the user's car, along with a trajectory for another traffic car with which the user's car shares the road. Our query trajectories for the user's car are always optimal with respect to *some* reward, such that users are essentially comparing reward functions. We pre-computed a query pool consisting of 7000 queries generated from a combination of 19 "plausible" reward functions and 40 environments. We rejection-sampled plausible reward functions by eliminating rewards that, for instance, incentivise the car to crash.

6.5.3 Oracle Users

The Users We Simulate. In our first experiment, we tested our hypothesis for the case that users are perfect, meaning their answers are noise-free. The user becomes an oracle, who is fully aware of the ground truth reward w_{GT} and uses it to answer queries, returning $I^* = argmax P(I|w, \varphi, \beta^c)$ for the comparison and $i^* = argmax P(i|w, \varphi, \beta^f)$ for the most influencing feature. Equivalently, this oracle user has $\beta^c = \beta^f = \infty$.

We thus *simulated* oracle answers based on w_{GT}, setting the simulated noise $\beta^c_s = \beta^f_s = \infty$. We refer to the simulation noise parameters as β^c_s and β^f_s, where s stands for *simulation*. We distinguish these parameters from model assumption of user noise, which we denote by β^c_m and β^f_m (m stands for *model*).

Our Model of the Users. First we used an accurate model of the oracle behavior: the learning algorithm models users as perfect as well, with the modeled noise parameters $\beta^c_m = \beta^f_m = \infty$.

Analysis. For each ground truth reward function and the number of queries allowed, a query in each learning method produces a probability distribution over reward parameters, which we evaluate according to our dependent measures. We analyzed the effect of the query type on each measure using a repeated measures ANOVA. Here we included the number of queries to help explain the difference between both algorithms' performance with very few queries and with many queries. We included an identifier for the ground truth weight as a random effect.

The results support our hypothesis. Rich queries lead to significantly higher probability being assigned to the ground truth reward function ($F(1, 778) = 374.02$, $p < .0001$), significantly higher integral over rewards similar to the ground truth ($F(1, 778) = 33.60$, $p < .0001$)), significantly higher dot product between the ground truth and the learned weights ($F(1, 778) = 4.89$, $p = .02$), and a significantly lower regret when used to optimize behavior ($F(1, 778) = 10.14$, $p < .01$).

Fig.6.2 summarizes these results.

6.5.4 Realistic, Noisy Users

The Users We Simulate. Real people are imperfect. They will not follow the $\beta^c = \beta^f = \infty$ assumption. We thus ran a pilot user study with a few real users to estimate the realistic values for β^c and β^f. We estimated the users' true reward parameters, then generated queries in different environments and asked them to provide responses to the queries. We ran a maximum likelihood estimation for the βs given their data:

$$\beta^{c*} = argmax \sum_{i=1}^{N} \log \frac{1}{1 + exp(-I_i \beta w^{*T} \Phi_i^*)} \tag{6.11}$$

$$\beta^{f*} = argmax \sum_{i=1}^{N} \log \frac{exp(|\beta^f w_f^* \phi_{if})|}{(1 + exp(-I_i \beta^c w^{*T} \Phi_i)) \sum_j exp(|\beta^f w_j^* \phi_{ij}|)} \tag{6.12}$$

Here $|N|$ denotes total number of queries made and $w_f^* \phi_{if}$ is the true reward contribution of the most influencing feature for that query. We decoupled the comparison query function and the feature query function in equation 6.12 and performed a search for β^c over several thousand values of β and then used this β^c to run a similar search for β^f. We also computed β^{c*} separately for rich queries and comparison queries. To our surprise, we found that β^c is much higher for our algorithm than for the comparison-only method: users were more accurate in comparison responses when the comparison queries are accompanied by feature queries. We found the average β^c to be 1.6 and 5.65 for the baseline and for our algorithm respectively. Objectively, this might be due to the difference in the nature of the queries selected by the two algorithms. Our algorithm selects queries with low norms that differ along only very few features, in order to make the feature answer more useful. This might lead to comparisons that are easier to make. Subjectively, too, most users seemed to think they were better at answering comparisons when they had feature queries too (the

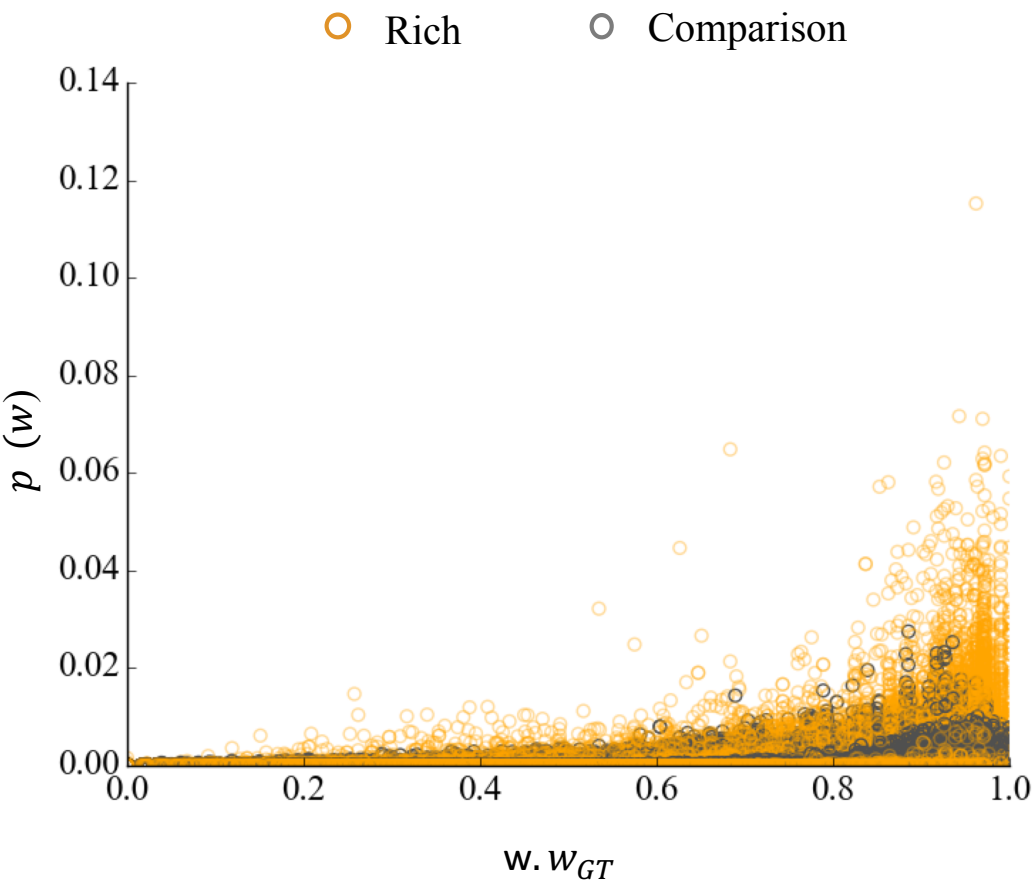


Figure 6.4: A scatter plot of the probability of all the weights in the data set by their dot product with a true reward shows that our algorithm converges much faster than comparison-only queries. Here $\beta_s^f = \beta_m^f = 2.5$, $\beta_s^{c1} = \beta_m^{c1} = 5.$ and $\beta_s^{c2} = \beta_m^{c2} = 2$.

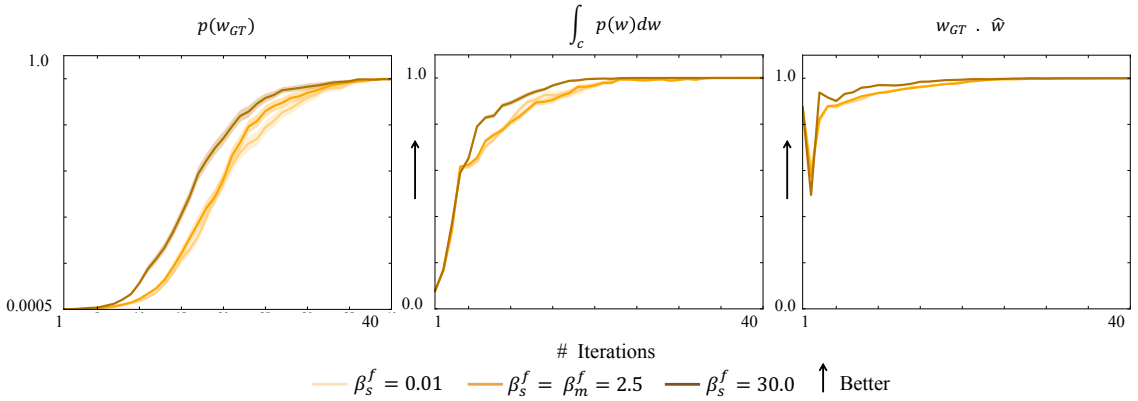


Figure 6.5: As users become more or less noisy in answering the feature query than we model them to be (captured by β_s^f), our algorithm's performance ranges between the performance of learning from comparisons-only queries, and that of learning from oracle users.

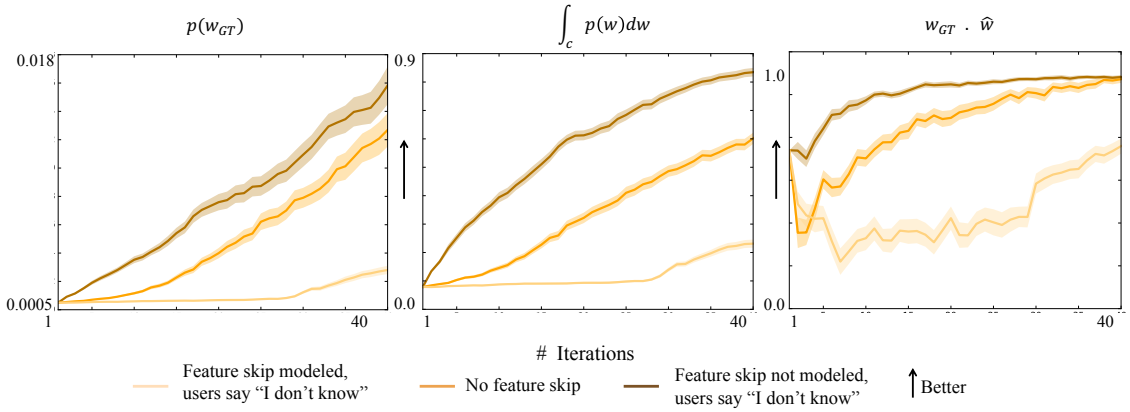


Figure 6.6: Graphs showing three dependent measures for the case when users are allowed to skip feature queries and this user behavior is incorporated in the model. The dependent measures are for a single true reward function averaged over 100 repetitions. Allowing users to skip the feature queries they are unsure of improves the performance of our algorithm. But interestingly, incorporating this behavior into the model hurts the performance.

average on a 7-point Likert scale for the question "I thought answering feature queries improved my ability to compare between the two trajectories" was 6.0).

We simulated imperfect real users with these estimated values of β_s^c and β_s^f. Following the results of the pilot we considered two different noise parameters for comparison answers: β_s^{c1} for the comparison-only method and β_s^{c2} for rich queries. Note that $\beta^f = 0$ is equivalent to the baseline, where we get no additional information because here the users are totally random in answering the feature queries and $\beta^f =$

∞ is equivalent to oracle feature picking.

Our Model of the Users. In this experiment, we first assumed that the models have accurate knowledge of the user behavior and set $\beta_m^{c1} = \beta_s^{c1}$ for the comparison-only algorithm and $\beta_m^{c2} = \beta_s^{c2}$ and $\beta_m^f = \beta_s^f$ for our algorithm. This scenario replicates what would happen when the learning algorithm has a good model of what noise level to expect from users.

Analysis. This time, since we simulate users to be noisy, we ran each simulated condition, which is a combination of ground truth weight and query type, 100 different times with different random seeds for each of the 20 ground truth weights. We ran these simulations in parallel using PyWren (1).

We repeated the analysis from before, and found support for our hypothesis despite the fact that we no longer learn from perfect user answers. Rich queries lead to significantly higher probability assigned to the ground truth reward, significantly higher integral over similar rewards to the ground truth, significantly higher dot product of the learned reward with the true reward, and significantly lower regret, all with $F(1, 1e + 5) > 335$ and $p < .0001$ throughout. The scatter plot between the probability distribution over the reward space and their dot product with the true reward, in fig. 6.4 shows a comparison between the convergence of the two algorithms. Our algorithm with feature queries converges faster to a reward function closer to the true reward.

This is not too surprising: even with noisy users, that the added information about features should still help. Fig.6.3 shows the results.

6.5.5 Users with Different Noise Level from the Model

Simulated Users and Model. In some situations, it will indeed be possible to get a good estimate of the user noise level. Here, we explore what happens when this is not the case, and the users are either much worse or much better at answering feature queries than we expect. We set β_m^f as before, but vary β_s^f, namely the simulated user noise. Throughout these experiments, we keep $\beta^c = \infty$ to avoid additional noise introduced through noisy comparison response.

Analysis. Not surprisingly, as β_s^f ranges from 0 to ∞, our algorithm's performance improves, interpolating between the comparison-only performance and the oracle performance. Note that $\beta_m^f = 2.5$, used in the noisy model, is less than the estimated optimal value of 3.8 from pilot studies. Fig. 6.5 shows that this is quite a conservative value. The algorithm performance does not decrease significantly with lower values of β^f. On the other hand, if people are more accurate about their true reward functions, feature queries can lead to a much faster learning.

6.5.6 Users that Say "I don't know."

The Users we Simulate. We now turn to studying the utility of allowing users to say "I don't know", i.e. to skip the feature part of the rich queries. We recover from a pilot study analogous to the one on the noise parameters a value for ϵ at 0.066.

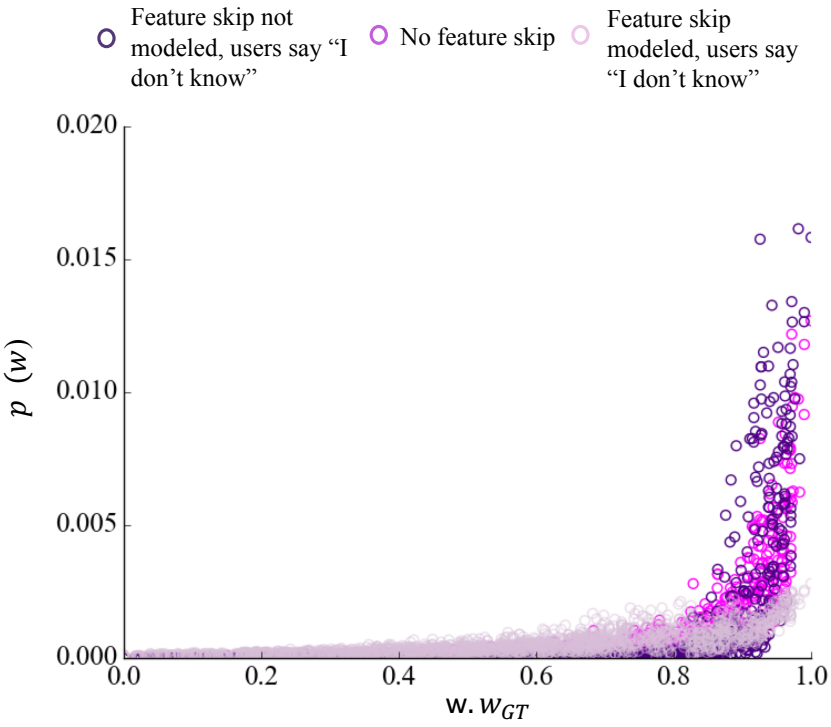


Figure 6.7: Scatter plot of the probability of all reward functions by their dot product with the true reward function at convergence. This graph suggests that allowing users to skip the feature queries they are unsure of leads to faster convergence. But incorporating this human behavior into the model hurts convergence.

We used this value of ϵ_s to simulate no response to the feature query. We ran these simulations assuming noisy responses to comparison and feature queries as before.

Our Model of the Users. Here we switch to the learning algorithm that models users as saying "I don't know" within some ϵ_m (see eq. 6.8), and set $\epsilon_m = \epsilon_s$.

Analysis. The results show that it is wise to avoid responding to feature queries when highly unsure, under the current assumption of noise ($\beta_m^f = 2.5$), matching our expectations. Our algorithm performs better when we allow people to say "I don't know" compared to when we do not allow them to. For a given true weight, we observed that after 40 iterations, the probability of the true weight improves from 0.012 to 0.016 as we allow people to skip feature queries. Likewise, allowing skipping leads to learning weights that are much closer to the true weight (probability integral of close ws $= 0.8$) than when people respond to every query (probability integral of close ws $= 0.6$). Fig. 6.6 shows these results. This result is also apparent from the scatter plot in Fig. 6.7 that shows a faster convergence when people are allowed to say "I don't know".

However, to our surprise, if we include the assumption that people might say "I don't know" in our model, the performance of the algorithm suffers. For the

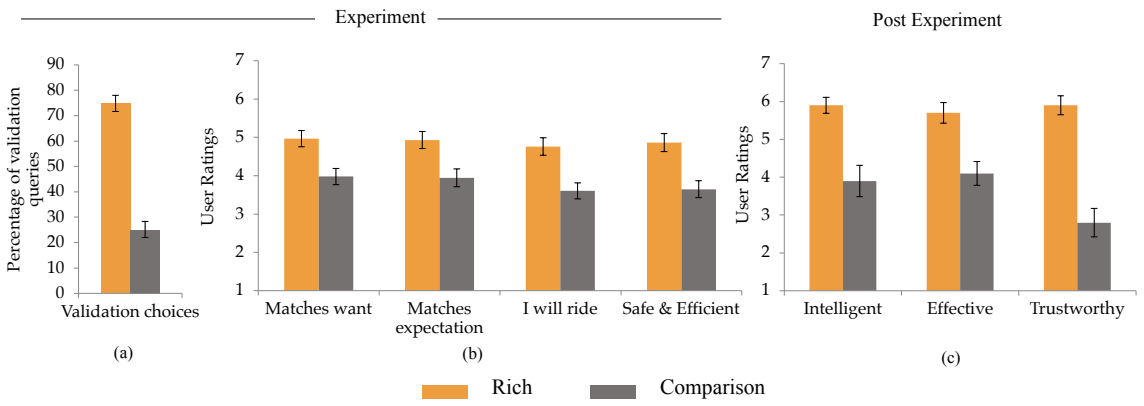


Figure 6.8: (a) During the validation phase most participants preferred the trajectories optimal for w^*_{rich}. (b) Overall trajectories learned with feature queries matched what participants' desired and expected driving behavior and also appeared to be safer and efficient. (c) Post experiment, most participants agreed that the robot with feature queries was more intelligent, effective and trustworthy than the one that just made preference query.

same true weight as above, we found the probability of this true weight and the probability integral of close ws to be 0.003 and 0.23 respectively, compared to 0.016 and 0.9 when feature skipping is not modeled. Furthermore, the scatter plot between probability distribution over weight space and dot product with true weights hardly shows any sign of convergence. This performance is comparable with the case when no feature query is made. Intuitively, having a better user model should lead to higher performance, but it seems like at least the volume removed metric (which optimizes purely for information) interacts poorly with this user model and makes active query selection less useful.

6.6 User Study

6.6.1 Experiment Design

Next, we tested our hypothesis with actual users.

Study Protocol Overview. Each user answered 20 queries of each type: comparison-only and rich. For each method, we used the 20 answers to learn a reward function: two reward functions per participant, that we then compared in order to test which method learned a more accurate reward. We then used 10 test environments for validation. We asked the participants, for each test environment, to choose between a pair of trajectories (one optimal with respect to the reward learned through comparisons only, one optimal with respect to the reward learned through rich queries). We also asked them to rate each of the two trajectories in each environment on Likert

scales for the statements in Table 6.1. Finally, at the end of the study, the participants answered questions about the experience of interacting with rich queries (Table 6.2).

Manipulated Factors. We manipulated the query type. Informed by the results of our simulation, we also allowed participants to skip queries if they were unsure of the answers.

Participants and Allocation Method. We recruited 10 participants consisting of a mix of undergraduate and graduate students, and used a within-subjects allocation so that participants could actually compare the outcome of the two learning methods. All the participants had 1+ years experience in driving and were comfortable with feature queries even with limited technical background.

Dependent Measures. Since with real people we no longer have access to the ground truth reward (it is internal to them), we had to settle on a different way to measure the quality of the learned reward. We thus had participants compare the learned rewards from each method by exposing them to two trajectories that optimize each reward in 10 test environments.

For each environment and trajectory, we asked participants 4 Likert-scale questions, capturing their preference for one or the other (see Table 6.1). We also asked them, for each environment, to choose the trajectory they prefer. These questions helped us understand if the users considered any of the two methods to be particularly effective or both methods to be ineffective.

Table 6.1: Measures for each trajectory in each test scenario.

Validation Likert Questions
Q1: Trajectory X matches what I want the car to do.
Q2: Trajectory X is what I would expect the car to do.
Q3: I would like to ride in the car using trajectory X.
Q4: I think trajectory A is the right combination of safety and efficiency.

6.6.2 Analysis

We analyzed participants' ratings of the trajectories produced by each learning method using a repeated-measures ANOVA for each item. We found significant effects of the query type across the board: rich queries led to significantly higher evaluation of whether the car matched what the user wanted ($F(1, 175) = 11.52$, $p =< .001$), whether it matched what they expected ($F(1, 175) = 10.00$, $p =< .01$), whether they would like to ride it ($F(1, 175) = 14.41$, $p =< .001$), and whether it was the right

combination of safety and efficiency ($F(1, 175) = 15.05$, $p =< .001$). The trajectory corresponding to the weight learned by the rich queries was chosen 74% of the time.

Participants found the robot that makes rich queries more intelligent, effective and trustworthy. Fig. 6.8 shows the results of the user study experiment and their responses to the post-study Likert Scale questions.

We asked participants to rate their experience with the feature queries on a 7-point Likert scale in the following form:

Table 6.2: Feedback on experience with feature queries.

Post-study Likert Questions
Q1: I thought answering feature queries was useful.
Q2: The method where the car uses feature queries was annoying.
Q3: I thought answering feature queries gave more transparency into the working of the car.
Q4: I thought answering feature queries improved my ability to compare between the two trajectories.
Q5: I liked when I did not have to answer feature queries.

Most participants considered feature queries to be extremely useful (average rating 5.8 and that answering feature queries helped them to understand how the car worked and thus improve their comparison capabilities (average ratings 5.9 and 6.3 respectively). They all agreed that they would help the car to learn their preferences better by answering feature queries, even if it was extra work (average rating for Q5 in Table 6.2). Overall the participants reported positive experience with feature queries and gave an average rating of 6.0 over all the experience-related statements.

6.7 Chapter Summary

The key idea of this chapter is that robots can extract richer information from people about their preferences if they make the right kinds of queries. We introduced feature queries as a way to augment comparison-only queries and get richer guidance from users when learning reward functions. We did an in-depth analysis in simulation, emulating perfect and noisy responses. We found that the richer queries consistently outperform comparison-only queries in being able to extract the correct reward faster. We then did an in-lab user study where participants interacted with each learning method, and again found that rich queries led to better outcomes within the same number of iterations.

Limitations and Future Work. While rich queries are really helpful, they assume that the features of the reward function are interpretable, and that they can be explained to end-users. This was the case in our application, but it will not always hold. There has been a lot of progress in learning features from sensorimotor data directly via deep learning, and even though there is much excitement about being able to interpret or visualize these features, this is still work in progress.

Next we extend the concept of *rich* queries into a hierarchical structure and learn more complex *reward dynamics* that represents changing preference in non-stationary environments.

Chapter 7

Multi-modal Preference from Hierarchical Queries

We already demonstrated that we can use *rich queries* along with an intelligent query selection algorithm to learn a static reward function that represents constant human preference for autonomous driving. This is the most common form of reward model used across all IRL (97; 106; 140) and robotics literature (3; 79). Despite the popularity, such static reward functions do not always adequately capture human preferences, especially, in non-stationary environments: Driving preferences change in response to the emergent behaviors of the other agents in the environment. In this chapter, we propose a second form of rich query HIERARCHICAL COMPARISON QUERIES to learn non-stationary preferences.

7.1 Multi-modal Preference

Real world is often non-stationary due to environment complexity or changes in objectives in the environment. Surrounding agents continuously change their behavior which in turn requires the robot to adapt to these changes. As an important class of non-stationary environments, human-robot and robot-robot adaptation have recently attracted much attention. However, unlike existing works that try to ensure robots adapt to their changing environments and other agents (98; 8), our goal is to learn the reward functions that dynamically change depending on the interactions between the agents and the environment.

Consider the example in Fig.7.1: On the left the white traffic car $\mathcal{E}$ is attempting a merge in front of our user's autonomous car. the users car $\mathcal{H}$ decides to be nice and slows down to facilitate the merge. $\mathcal{E}$ still cuts close in front of our car and continues slowly after merging. The user in this case showed a cooperative mode of preference indicated by M_1 in the figure. This can frustrate the user who then decides to speed up and overtake the white car, as we see in the right. He transitions to a more aggressive behavior. We consider this a change in preference mode to aggressive, indicated by

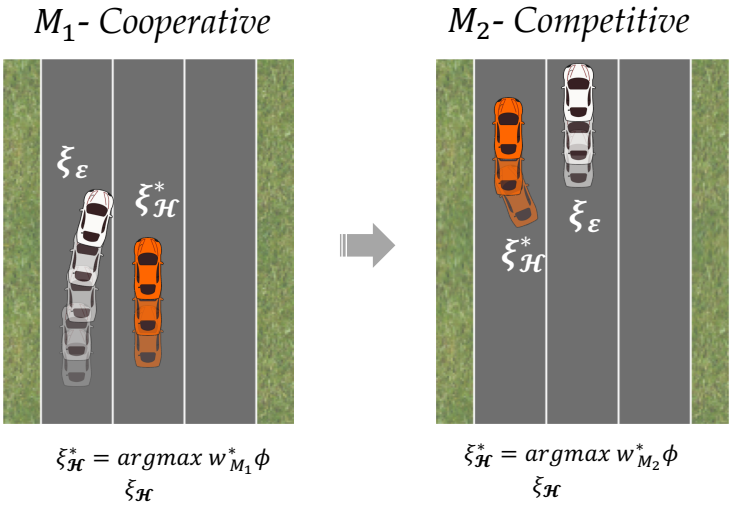


Figure 7.1: The orange car is the autonomous car and the white car is the environment agent. On the left the user prefers a trajectory of the autonomous car optimal with respect to a reward function that represents a more cooperative behavior (mode M_1): letting the environment agent merge. But the environment agent's behavior frustrates the user and he next prefers a more aggressive behavior: speed up and overtake the white car. This trajectory is optimal with respect to a different reward function that represents more aggressive behavior (mode M_2).

M_2. If we consider the user as a rational entity whose actions can be explained by a reward function, it is almost as though the user is optimizing two different reward functions w_{M_1} and w_{M_2}. Modeling driving behaviors in non-stationary environment is a well-studied problem especially for driving. For example, (47) characterizes driving styles based on sensor data using deep learning. In a more related work (96), the authors modeled the drivers with a latent state space which can affect their driving behavior. While they stated these latent states might change over time, both of these works made the assumption that latent states remain unchanged over the trajectories of interest, so they did not address changing behaviors. In (22), the authors modeled the latent states of the drivers using Hidden Markov Models (HMM) where they also allow adaptation. However, they did not specifically learn reward functions, and they focused on identifying the maneuvers the drivers will perform from a predefined database. With a similar objective, (80) employed HMMs for latent state estimation for human-robot interaction. We are interested in a particular kind of non-stationary environment: which changes in response to other agent $\mathcal{E}$'s behavior as in Fig.7.1.

We represent this behavior by a *mixture of reward functions* w_{M_j} for all $j \in [k]$, one for each j^{th} mode of preference, as we mentioned in Chapter.4. We call this REWARD DYNAMICS. Prior works have also theoretically investigated how to perform preference-based learning for multi-modal reward functions (88; 139; 87; 41). Specifically in (139), it was shown that pairwise comparisons can be used to learn only unimodal reward functions, or as we call, static reward functions. This is because most of these works consider that the two modes are independent of each other. We have this problem too. Unlike prior formulations of Bayesian update as in Chapter.6, where we knew which distribution w came from, in case of two modes we do not know the current mode of the user. Speaking in terms of mixture model, we do not know the parameters of the individual distributions and the parameter governing the component membership at the same time. Other researchers have made uniformity assumptions as a work around (41) i.e. any mode is equally likely. We do not make a uniformity assumption. Instead we propose a computational model for mode transitions, which takes care of the *identifiability* issues. We referred to this as *Mode utility Function* U_{M_j} for all $j \in [k]$ in Chapter.4. Now we know why we need a function governing mode transitions when we are trying to learn multi-modal preferences. So *reward dynamics* now encodes not only different human preferences but also how the preferences change.

7.2 Overview of Contributions

Our contribution is in the way we adapt preference-based learning into a *hierarchical* approach that aims at learning not only reward functions but also how they evolve based on interactions. We derive a probabilistic observation model of how people will respond to the hierarchical queries. Our algorithm leverages this model to actively select hierarchical queries that will maximize the volume removed from a continuous hypothesis space of reward dynamics. We empirically demonstrate reward dynamics

can match human preferences accurately.

In our work, as well as how to learn the reward functions, we are also interested in how they change in non-stationary environments.

In this thesis, we propose to learn an expressive representation of preferences in non-stationary scenarios, where interactions and adaptations better reflect the real-world conditions. We assume that the non-stationary scenarios arise from changing behaviors of other agents interacting with our system, which in turn affect human preferences. We formalize reward dynamics which encodes not only different human preferences but also *how* they change.

> *Our insight is that reward dynamics matches human preferences more accurately in a wide range of scenarios than a static reward function.*

We actively select comparison queries from a database, similar to (25), to learn a probability distribution over *reward dynamics*: a mixture of static reward functions representing different moods and a set of parameters for the transitions between the moods. We tailor comparison queries to capture longer term interactions between the robot and the surrounding agents, and develop a mathematical model of user responses for any number of static reward functions and the transitions between them.

In this work, we make the following contributions:

Reward Dynamics: User preferences may change based on the behaviors of other agents in the environment. We encode the momentary human preference by a static reward function and assume at any point of time the human has an internal *preference mode* (mood) which dictates what static reward function the human will optimize next. We introduce the notion of *reward dynamics* as a tuple of reward functions each representing a different preference, and mode-utility functions that are parameters governing transitions between different modes.

Hierarchical Queries: We formally define *hierarchical queries* which are a sequence of hierarchical pairwise comparison questions, each of which will be called a *sub-query*. These subqueries sequentially follow each other so that the user moods will be reflected into their behaviors. The hierarchical queries are essential in learning the user's reward dynamics.

Active query selection: We provide an algorithm that actively selects informative hierarchical queries in order to efficiently learn reward dynamics through interactions with the users. We evaluate our algorithm on an autonomous driving example in simulation. Our user study suggests that we can efficiently learn changes in humans' preferences based on interactions with different environments.

Problem Domain Prior works on autonomous driving(116; 115; 112; 113; 14; 121; 56; 127) assume that $\mathcal{H}$ should follow the same reward function over time in both of the above scenarios. We argue that user preferences may vary in response to the changing behaviors of the environment agents in both driving and potentially other multi-agent environments. Our goal is to learn an expressive reward function corresponding to these dynamic preferences.

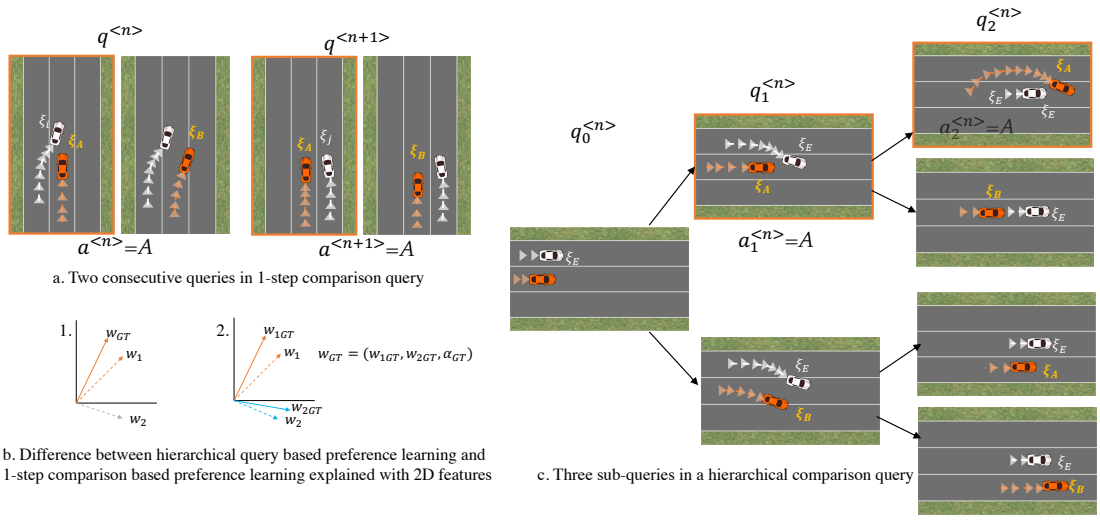


Figure 7.2: In a. we see in two consecutive 1-step comparison queries people may give two different preferences, preferring defensive behavior in the first one and aggressive in the second one. This registers as noise in 1-step comparison based learning as shown in b. In b. w_{GT} is a single 2-D ground truth reward function for 1-step comparison query (1.) and w_{GT} is a tuple of 2-D ground truth reward dynamics for hierarchical comparison query (2.). In (c) we show the hierarchical structure of our proposed queries, where we can learn that both w_1 and w_2 represent true user intent w_{GT}.

7.3 Hierarchical Comparison Queries

Prior works have learned static reward functions by asking people to compare between two different trajectories of robots. There, each query is a pair of short videos that demonstrate two trajectories of the system (110; 25; 43). Such short trajectories do not capture the nuances of interaction in a multi-agent system. As an example, let us consider a sequence of two comparison queries as shown in Fig. 7.2 (a). In the first query, an environment agent $\mathcal{E}_i$ (white car) aggressively merged in front of $\mathcal{H}$ (orange car). One option is for our user to slow down (optimize a cooperative reward function). This sudden slow down may have frustrated the user. So, in a different query, if one previous sub-query induces some change in the user's mode, then in a similar situation when another agent $\mathcal{E}_j$ (white car) tries to merge in front, our user prefers the trajectory where $\mathcal{H}$ (orange car) speeds up and does not let $\mathcal{E}_j$ merge. Here the user showed a more competitive behavior (optimized with respect to a competitive reward function). This change of preferences between the two comparison queries will manifest as noise in 1-step preference-based reward learning approaches. However, we would like to learn a composite reward function that not only captures both of these preferences but also captures *how* the humans' preferences have changed in such non-stationary environment.

To do so, we allow the users to change their preferences within the same query. We propose a second type of rich query *Hierarchical Comparison Queries*, where each query q as a sequence of several sub-queries. Each sub-query q_i in the sequence is a continuation from the final state of the trajectory of the previous sub-query q_{i-1}. This allows us to learn how the behavior of other interacting agents in one sub-query affects user preference in the next sub-query. We assume that the users' next immediate *preference mode* depends only on their current experience. We, therefore, reset their preference mode at the beginning of each query with a demonstration, the first sub-query, which we denote as q_0 (see the hierarchical query structure in Fig. 7.2 (b). After q_0, each sub-query presents a pair of trajectories $q^A, q^B \in \Xi$ for the user to indicate their preferences by selecting which trajectory they would prefer. q_1^A and q_1^B are both continuations of q_0. In general, for the rest of the sub-queries, q_i^A and q_i^B are continuations from $q_{i-1}^{a_{i-1}}$ where a_{i-1} is the option the user picked in the $(i-1)^{\text{th}}$ sub-query, as shown in Fig. 7.2 (c). Following the definition of *rich queries* in Chapter.5, therefore, all our sub-queries are pairwise comparison queries between trajectories.

7.4 Reward Dynamics Model

7.4.1 Preliminaries

Throughout the chapter, we will use $[n]$ to denote the integer set $\{1, 2, \ldots, n\}$ for $n \in \mathbb{Z}_{>0}$.

We denote the i^{th} sub-query as q_i for $i \in [s] \cup \{0\}$, where the number of sub-queries within one query is s.

We assume there is a finite set of modes M and we enumerate each mode such that $M = \{1, 2, \ldots, k\}$ where k is the total number of modes. M_j is the j^{th} element of the set of modes M. We also assume the mode of the user is stable during each time period, the duration of a sub-query. We denote the mode in the i^{th} sub-query as $m_i \in M$.

Each q_i, except q_0, consists of two trajectories $q_i^A, q_i^B \in \Xi$. The user selects A or B as his/her response to each of these sub-queries. The user's response to q_i is denoted as $a_i \in \{A, B\}$. $\bar{a}_i$ denotes the complement of a_i, i.e. $\{a_i\} \cup \{\bar{a}_i\} = \{A, B\}$.

In addition, we assume a features function $\phi : \Xi \to \mathbb{R}^d$ that maps every trajectory to a d-dimensional feature space. This function depends on both $\mathcal{H}$ and $\mathcal{E}$. We assume the d features of the environment $\mathcal{F}$ are known. For example, some representative features for driving in traffic can be distance to the closest environment car, distance to the road boundaries, the speed or heading angle of the vehicles, etc.

7.4.2 Human Preference Model

Reward functions under known modes. Reiterating from Chapter.4, we defined a user-specific reward function parameterized by the mode of the user: $R_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$. With linearity assumption as in (110; 25), it is defined as: $R_{M_j}(\xi) = w_{M_j}^\top \phi(\xi)$ where $\xi \in \Xi$ and $w \in \mathbb{R}^{d \times k}$ is a user-specific weight matrix, and w_j is the j^{th} column of w, with each column corresponding to a particular mode for a user. Then, the user response to a sub-query q_i is probabilistic based on Luce's Choice Axiom (89; 67) which is a widely used human decision model in cognitive science as it nicely captures the noise in humans' choices:

$$P(a_i | q_i, m_i, w) = \frac{\exp(R_{m_i}(q_i^{a_i}))}{\exp(R_{m_i}(q_i^{a_i})) + \exp(R_{m_i}(q_i^{\bar{a}_i}))}$$

This expression captures probability of the human making a choice given a sub-query, the humans' mode during that sub-query, and the user specific preferences and relates this probability to the human's preference reward function in that particular mode.

Prior on mode transitions. We also define a matrix $G \in \mathbb{R}^{k \times k}$ that defines the prior over the mode transitions. That is, on a completely neutral case (the meaning of this will become clear in a while), if the user is in mode M_{j_0}, then its probability of transitioning to mode M_{j_1} is given by $G_{j_0 j_1}$. This implies that G is constrained to be a proper Markov chain matrix. We note Markov chains are employed similarly for mood changes by psychiatrists, e.g. (68). We are going to assume G is provided by the model designer.

Mode transition model. The user changes its mode based on the previous time interval, i.e. previous sub-query. We model this stochastic transition as follows: The user has an underlying mode-utility function that quantifies the trajectory. This

mode-utility function is parameterized by the mode of the user: $U_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$. Again with linearity assumption, it is defined as: $U_{M_j}(\xi) = \gamma_{M_j}^\top \phi(\xi)$ where $\gamma \in \mathbb{R}^{d \times k}$ is another user-specific weight matrix and γ_j is the j^{th} column of γ. If the user thinks she would have higher utility with mode M_j, then she transitions to M_j. As an example, imagine you are driving in a very calm mood. If someone suddenly cuts in front of you, you would think "if I were aggressive, I could keep a shorter headway with the car in front and the other car would not have been able to cut in front of me", and you also switch to an aggressive mood. It is of course also possible that you keep calm. Therefore, the transition should be stochastic.

Moreover, some mode transitions are naturally more likely than the others. For example, if we have three modes that correspond to defensive, neutral and offensive moods, then it would be more likely for a defensive user to switch to the neutral mode than to the offensive mode. G captures this prior. We will give some examples after we define the probabilistic mode transition with a softmax model:

$$\begin{aligned} &P_{j_0 j_1}(q_{i-1}, a_{i-1}, \gamma) \\ &\quad := P(m_i = M_{j_1} | m_{i-1} = M_{j_0}, q_{i-1}, a_{i-1}, \gamma) \\ &\quad = \frac{1}{Z} \frac{\exp(U_{M_{j_1}}(q_{i-1}^{a_{i-1}}))}{\sum_{m \in M} \exp(U_m(q_{i-1}^{a_{i-1}}))} G_{j_0 j_1} \end{aligned}$$

where Z is the normalization constant. Now, it should be clear what we previously meant by "completely neutral case". That is, when the softmax gives equal values for each mode, the transition is solely defined by the prior G. Some examples of G are:

- $G_{j_0 j_1} = 1/k$ for $\forall (j_0, j_1) \in [k]^2$ means that the user may change from any mode to any other mode just based on the previous subquery with a uniform prior. This is suitable when the modes are categorical, not sequential.
- $G = I$ means the user will not ever change her mode and will remain in her initial mode. Note that the initial mode will also be modeled in a probabilistic way.
- If G is a band matrix, then the user can only change between the modes that are "close". This is suitable for sequential modes.

While our model will be valid for any feasible G, we will do simplifying assumptions to actively select the hierarchical queries for sample-efficient learning.

Definition 7.4.1. We define *reward dynamics* of a user as a tuple of (w, γ), which governs both the user preferences and how they change with the interactions the user is involved.

Therefore, our aim is to learn the *reward dynamics* rather than a static reward function.

Initial State. We do not know the initial mode m_0 of the user, which is the active mode during q_0. One simple way is to assume uniform distribution over all modes.

However, imagine G is such that transitioning to M_j is very unlikely from any mode. Then, the uniform assumption will not hold, because the user is unlikely to be in mode M_j. Then a better model is the following:

$$P_j := P(m_0 = M_j) = \pi_{M_j}(G)$$

where π_{M_j} denotes the probability of mode M_j in the stationary distribution of the Markov chain G. If there exist several stationary distributions, the designer should pick one of them using domain knowledge. For example, for $G = I$, one option is to assume P_j is uniformly distributed.

7.4.3 Learning Reward Dynamics

To make the learning of reward dynamics effective and efficient, we should restrict the continuous space of reward dynamics. For that, we make assumptions on the norms of the columns of w and γ similar to (110; 25).

There is also the problem of *label switching*. That is, all the probabilities will remain the same if we switch the order of modes both in w and γ. Since this can completely disable the learning, we enforce another constraint on the ordering, as mentioned by (139), such that $\gamma_{M_1,1} > \gamma_{M_2,1} > \cdots > \gamma_{M_k,1}$ where $\gamma_{M_j,1}$ is the first row of M_j^{th} column of γ.

Our goal is to learn a distribution over the *reward dynamics* by making informative queries. We start with a uniform prior over the space of all feasible (w, γ). After receiving all the answers to a query q, $(a_1, a_2, \ldots, a_s)$, we perform a Bayesian update:

$$\begin{aligned} &p(w, \gamma | a_s, a_{s-1}, \ldots, a_1, q_s, q_{s-1}, \ldots, q_0) \\ &\propto p(a_s, a_{s-1}, \ldots, a_1 | w, \gamma, q_s, q_{s-1}, \ldots, q_0) p(w, \gamma) \end{aligned}$$

Next we derive the expression for the update function $p(a_s, a_{s-1}, \ldots, a_1 | w, \gamma, q_s, q_{s-1}, \ldots, q_0)$ and present some simplifications that we adopted for our implementation.

7.4.4 Derivation and Simplifications

In this section, we present how we compute the update function for $p(w, \gamma)$. We note q_0 does not receive any response. For the simplicity of notation, we assume $q_0^{a_0}$ gives the associated trajectory in q_0, so that $P_{j_0 j_1}(q_0, a_0, \gamma)$ is well-defined for $\forall (j_0, j_1) \in [k]^2$. We then derive

$$\begin{aligned} &P(a_s, a_{s-1}, \ldots, a_1 | w, \gamma, q_s, q_{s-1}, q_0) \\ &= \sum_{(j_0, \ldots, j_s) \in [k]^{s+1}} P_{j_0} P_{j_0 j_1}(q_0, a_0, \gamma) \ldots P_{j_{s-1} j_s}(q_{s-1}, a_{s-1}, \gamma) \\ &\prod_{l \in [s]} P(a_l | w, q_l, m_l = M_{j_l}) \end{aligned}$$

In our implementation, we restrict ourselves to the cases where $s = 2$. Then, the above equation is simplified as

$$\begin{aligned}&P(a_2, a_1|w, \gamma, q_2, q_1, q_0)\\&= \sum_{j_0\in[k]}\sum_{j_1\in[k]}\sum_{j_2\in[k]} P_{j_0}P_{j_0j_1}(q_0, a_0, \gamma)P_{j_1j_2}(q_1, a_1, \gamma)\\&\qquad P(a_1|w, q_1, m_1 = M_{j_1})P(a_2|w, q_2, m_2 = M_{j_2})\end{aligned}$$

To eliminate the normalization Z from the equation, we assume $G_{j_0j_1} \in \{0, 1/c_{j_0}\}$ for $\forall(j_0, j_1) \in [k]^2$ where c_{j_0} is an appropriate constant. That is, we assume the model designer will just decide on whether or not it is possible to move between any two modes and will not assign specific prior probabilities. Then,

$$P_{j_0j_1}(q_0, a_0, \gamma) = \frac{\exp(\gamma_{M_{j_1}}^\top \phi(q_0^{a_0}))}{\sum_{j'\in[k]:G_{j_0j'}=1/c_{j_0}} \exp(\gamma_{M_{j'}}^\top \phi(q_0^{a_0}))}$$

If we further assume $k = 2$ and $G_{j_0j_1} = 1/2$ for $\forall(j_0, j_1) \in [k]^2$, such as the case of cooperative and competitive modes, we also have $P_{j_0} = \frac{1}{2}$, so we can write:

$$\begin{aligned}&P(a_2, a_1|w, \gamma, q_2, q_1, q_0)\\&= \sum_{(j_1,j_2)\in\{1,2\}^2}\prod_{i\in\{1,2\}} \frac{\exp(w_{M_{j_i}}^\top \phi(q_i^{a_i}))}{\exp(w_{M_{j_i}}^\top \phi(q_i^{a_i})) + \exp(w_{M_{j_i}}^\top \phi(q_i^{\bar{a}_i}))}\\&\qquad\qquad \frac{\exp(\gamma_{M_{j_i}}^\top \phi(q_{i-1}^{a_{i-1}}))}{\exp(\gamma_{M_1}^\top \phi(q_{i-1}^{a_{i-1}})) + \exp(\gamma_{M_2}^\top \phi(q_{i-1}^{a_{i-1}}))}\end{aligned}$$

This formulation is completely independent from m_0 thanks to the simplifying assumptions on G.

7.5 Active Query Selection

In this section, $\mathcal{D}$ denotes all the information about w and γ up to the current iteration of interest – we dropped the subscript for simplicity.

Each answer tuple (a_1, a_2) removes some volume from the hypothesis space of (w, γ), where, volume removed is given as the difference between the unnormalized posterior distribution over (w, γ), and its prior distribution. We have a belief over what the user answers could be. We leverage this probabilistic model to *actively* select a query at each iteration that will maximize the expected volume removal. Formally, we solve the following optimization:

$$\begin{aligned}&(q_0^*, q_1^*, q_2^*)\\&= \arg\max_{q_0,q_1,q_2} \mathbb{E}_{a_1,a_2}\left[\mathbb{E}_{w,\gamma}\left[1 - p(a_2, a_1|w, \gamma, q_2, q_1, q_0)\right]\right]\end{aligned}$$

where both expectations are taken given $\mathcal{D}$, q_0, q_1 and q_2. To compute the inner expectation, we sample (w, γ) from $p(w, \gamma|\mathcal{D})$ using Markov Chain Monte Carlo methods. Unlike previous works in active reward learning (110; 25), our update function is not log-concave. We, therefore, resort to Metropolis-Hastings algorithm. Now, let's say we take M samples. We let $\bar{w}$ and $\bar{\gamma}$ represent those samples:

$$\begin{aligned}
&(q_0^*, q_1^*, q_2^*) \\
&\approxeq \arg\max_{q_0,q_1,q_2} \mathbb{E}_{a_1,a_2}\left[\frac{1}{M}\sum_{\bar{w},\bar{\gamma}}(1 - p(a_2, a_1|\bar{w}, \bar{\gamma}, q_2, q_1, q_0))\right] \\
&= \arg\min_{q_0,q_1,q_2} \mathbb{E}_{a_1,a_2}\left[\frac{1}{M}\sum_{\bar{w},\bar{\gamma}} p(a_2, a_1|\bar{w}, \bar{\gamma}, q_2, q_1, q_0)\right]
\end{aligned}$$

By formally writing the expectation, we obtain the objective as

$$\sum_{(a_1,a_2)} p(a_2, a_1|q_2, q_1, q_0, \mathcal{D})\frac{1}{M}\sum_{\bar{w},\bar{\gamma}} p(a_2, a_1|\bar{w}, \bar{\gamma}, q_2, q_1, q_0)$$

where the first sum is over $\{A, B\}^2$. By the law of large numbers, we also have:

$$p(a_2, a_1|q_2, q_1, q_0, \mathcal{D}) = \lim_{M\to\infty}\frac{1}{M}\sum_{\bar{w},\bar{\gamma}} p(a_2, a_1|\bar{w}, \bar{\gamma}, q_2, q_1, q_0)$$

which is because $\bar{w}$ and $\bar{\gamma}$ were drawn from $p(w, \gamma|\mathcal{D})$ and are independent from q_2, q_1, q_0; and (a_1, a_2) are conditionally independent from $\mathcal{D}$. Then, for large M, we can write the optimization as:

$$\begin{aligned}
&(q_0^*, q_1^*, q_2^*) \\
&= \arg\min_{q_0,q_1,q_2} \sum_{(a_1,a_2)\in\{A,B\}^2}\left(\sum_{\bar{w},\bar{\gamma}} p(a_2, a_1|\bar{w}, \bar{\gamma}, q_2, q_1, q_0)\right)^2
\end{aligned}$$

where the probability expression in the objective function is already derived in Section 7.4.

7.6 Simulation Experiments

7.6.1 Problem Domain

We focus on learning driving preferences. Each component of the learned *reward dynamics* weighs 5 features for driving: a feature for the distance to the road boundary, a feature for velocity, and three more features of proximity to lane borders, to other cars and alignment with the road, similar to (110). Each sub-query consists of the driving environment and a pair of trajectories of $\mathcal{E}$ and $\mathcal{H}$ whose preferred behavior we are learning. The environment is represented by the trajectory of an environment car and the initial states of $\mathcal{H}$. Our query database consists of 10000 randomly generated hierarchical comparison queries.

7.6.2 Dependent Measures

In our implementation we learned $\alpha_1 := \gamma_1 - \gamma_2$, instead of γ, as it has fewer parameters. The same approach generalizes to any k with $\alpha_j := \gamma_j - \gamma_k$ for $\forall j \in [k-1]$.

We measure the performance of hierarchical preference learning in terms of expected dot product between learned weights and true weight as in (110; 25), separately for each component of (w, α). r is a measure of convergence as value of r close to 1 indicates learned weights are close to the true weights.

$$r = \mathbb{E}\left[\frac{\hat{v}.v^*}{\|\hat{v}\|_2 \|v^*\|_2}\right] \tag{7.1}$$

where $v \in \{w_1, w_2, \alpha_1\}$, $\hat{v}$ and v^* are the estimated and true weights, respectively, and the expectation is taken over the sampled $\hat{v}$ values.

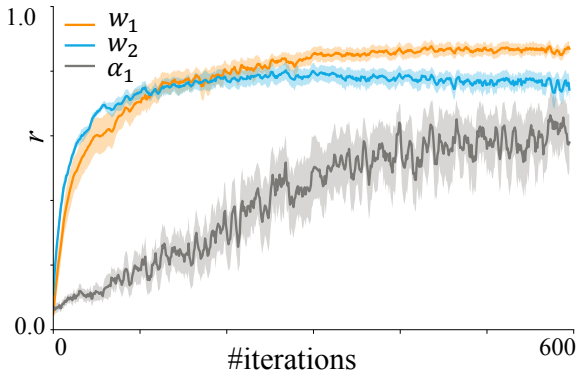


Figure 7.3: r value shows that our algorithm converges well for non-driving data with non-active query selection when the simulated user is *oracle*. Here we show an average m over 5 different ground truth *reward dynamics*.

7.6.3 Experiments with Random Data

We first conduct experiments with completely random and independent sub-queries without the driving environment. We assume we can generate queries in an unconstrained way such that any ϕ-vector is possible, i.e. there is no dynamics constraint in the generation of queries. Here, we simulate *oracle users*: users who are perfectly aware of their true reward dynamics. That is, they always behave (change mode and respond) with respect to the higher probability out of softmax models. In Fig. 7.3, the average results of 5 different simulated oracle users show convergence of (w_1, w_2, α_1) whose true values were independently drawn from standard normal distribution independently for each entry.

7.6.4 Experiments with Driving Data

Active versus non-active query selection. We compare the performance of our active query selection algorithm with a non-active baseline where we uniformly sample the queries from a discrete database of 10000 queries. Here our simulated users are always oracle. We test the following hypothesis:

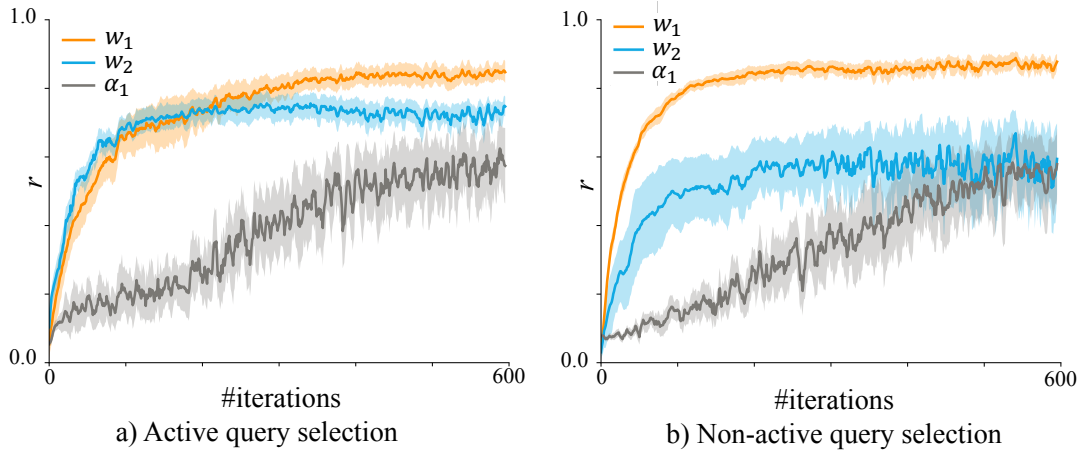


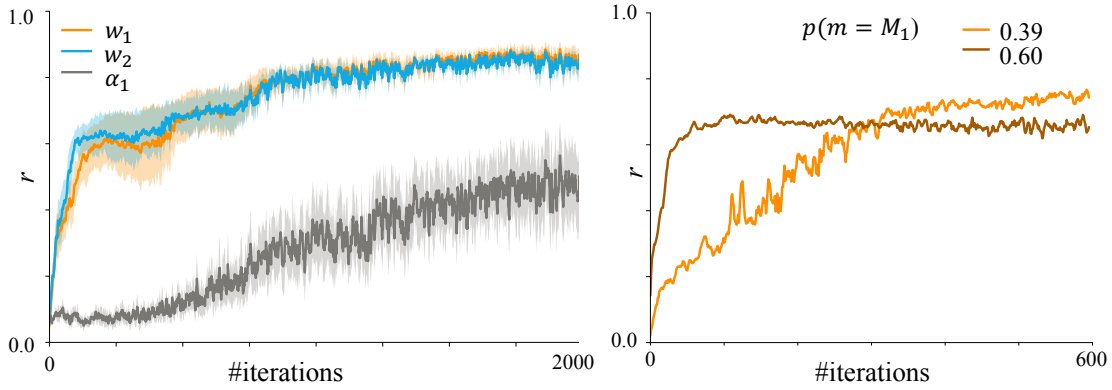
Figure 7.4: r values show that our algorithm with active query selection from dataset of 10000 discrete queries (left) can learn reward dynamics faster than non-active query selection (right) when the simulated user is *oracle*. Here we show an average r over 5 different ground truth reward dynamics.

H1. *The reward dynamics learned with our active query selection algorithm converge to their true weights faster compared to the non-active baseline.* Our results in Fig. 7.4 supports this hypothesis by demonstrating that active query selection accelerated the learning of one of the modes (the average r value at convergence for (w_1, w_2, α_1) are 0.84, 0.74 and 0.6 respectively) compared to the non-active baseline (the average r value at convergence for (w_1, w_2, α_1) are 0.85, 0.62 and 0.63 respectively). In fact, at the end of 600 queries, active selection led to significantly higher r_{w_2}.

Testing different mode preferences. Next we simulate 5 noisy users, who choose between options A and B with respect to $p(a|w, \gamma, q)$. Our algorithm actively selects queries from the same discrete dataset of size 10000 as in the case of oracle users. We first set the following hypothesis: **H2**. *Our algorithm learns the reward dynamics even when the users are noisy.*

We also test the performance of our algorithm for different mode likelihoods, i.e. probability of transitioning to a given mode, $p(m = M_j)$. We manipulated the ground truth reward dynamics to reflect different mode likelihoods. For example, one user might be in one mode 80% of the time while another user has equal chances of being in one of the two modes. Although this might actually affect the priors P_1 and P_2 as we explained in Section 7.4, we still adopted the derivations based on uniform prior to test

the robustness of our framework. Therefore, we test the following hypothesis: **H3**. *Our algorithm learns the weights w that correspond to both modes, and it converges faster for w_{M_j} if $p(m = M_j)$ is higher.*



a) r value for noisy users

b) r value for varying mode preference

Figure 7.5: m value shows even when the users are noisy our algorithm can learn the true *reward dynamics* (left) and that as $p(m_1)$ increases, w_1 converges faster (right).

We present the results in Fig. 7.5. The first plot shows that our algorithm was able to learn w_1, w_2 and α_1 converge even when the users are noisy. However, we note that throughout the simulated conditions α_1 converges slowly, as we were not able to fully avert the *label switching* problem. The second plot shows we are able to learn the reward weights w_j of a mode j regardless of its likelihood probability being high or low. The same plot also shows the algorithm converges faster for the modes that are visited more often. This is very intuitive as the algorithm is able to gather more information about those modes, even though it does not perfectly know the user is in the corresponding mood. Hence, **H3** has a strong empirical support.

7.7 User Study

7.7.1 Hypotheses

We test the following hypotheses with the user study: **H1**. *Our algorithm learns weights that can represent the driving behavior of the users.* **H2**. *Some people indeed change preferences depending on the driving behaviors of the interacting agents.*

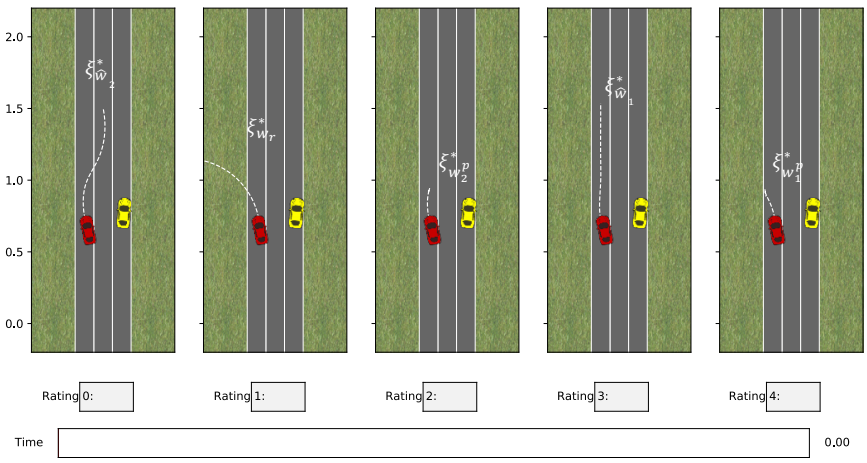
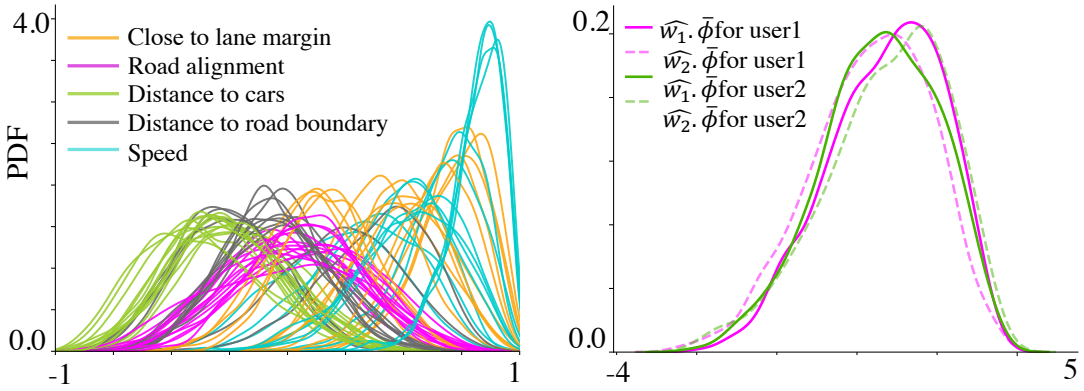


Figure 7.6: The validation interface showed each user 5 trajectories per query: two optimal with respect to the learned weights for the two modes, two optimal with respect to the perturbed versions of these weights and one optimal with respect to a random weight vector.



a) Distribution of learnt $w_1$and w_2 combined across all users.

b) Average rewards with respect to $\widehat{w_1}$ and $\widehat{w_2}$ for two users showing minor difference.

Figure 7.7: Distribution of $\hat{w}_1$ and $\hat{w}_2$ across all users for individual features in a) shows that user preferences vary widely for adherence to lane center and distance to road boundaries, but are very similar for efficiency (speed) and safe driving (collision avoidance). While we did not learn significantly different w_1 and w_2 for individual users, b) shows that the average reward w.r.t to $\hat{w}_1$ and $\hat{w}_2$ differ slightly for some of our study participants.

7.7.2 Study Design

In order to validate our hypotheses, we collected data from 10 real users in a within-subjects study. We first learn a general reward dynamics $(\hat{w}, \hat{\alpha})$ using 50 hierarchical queries. We then use the distributions over these parameters to jump-start the process for each subject with a reasonable prior that better represents legally correct driving. During validation, we ask users to provide ratings for trajectories locally optimized with respect to the learned reward dynamics. We compare the expressiveness of the learned weights $(\hat{w_1}, \hat{w_2})$ against their perturbed versions (w_1^p, w_2^p). We sampled these perturbed versions from Gaussian distributions centered around $(\hat{w_1}, \hat{w_2})$ and a standard deviation of $0.5 \times |\hat{w_1}|$ and $0.5 \times |\hat{w_2}|$. While creating perturbed versions of $\hat{w_1}$ and $\hat{w_2}$, we make sure that all weights generate legally correct driving, so that we only compare people's preferences for different modes. To do so, we constrained the weight components for staying within the road and collision with cars. Each rating question consists of two parts. The first part is similar to q_0 of the learning step, where we show user one trajectory demonstration of $\mathcal{H}$ as an attempt to set their initial mode. In the next part, we show users 5 trajectories continued from the first part, optimal with respect to 5 reward functions: $\hat{w_1}$, $\hat{w_2}$, their perturbed versions w_1^p and w_2^p, and a random weight w_{r}. For fair comparison, we also generated w_{r} from a Gaussian distribution centered on either $\hat{w_1}$ or $\hat{w_2}$ and a standard deviation of $2 \times |\hat{w_j}|$ with the j being the corresponding mode index. For each of the 5 trajectories, we ask users a 7-point Likert scale rating question: *Indicate your level of agreement with the following statement: I would like to ride this car* (see Fig.7.6).

In **H1** we claim 1) users will give the highest overall rating to the trajectories generated from $\hat{w_1}$ and/or $\hat{w_2}$ most of the time, and 2) if $p(M_j)$ is very high, we expect people to give the highest rating to trajectories generated from corresponding weight $\hat{w}_{M_j}$. To validate the first part, we repeat the same demonstration across several rating queries preserving $\mathcal{E}$'s trajectory alike and changing the trajectory of $\mathcal{H}$, varying between different local optimal with respect to w_1, w_2 and the other weights. We randomize demonstration trajectories across the rating questions. In **H2** we hypothesize that subject to different interactions in the environment, users will sometimes give higher rating to trajectory optimal for $\hat{w_1}$ and sometimes to those optimal for $\hat{w_2}$.

7.7.3 Results

Like previous work in this area (110; 25), we found that the users have somewhat similar preferences, proximity to cars have high negative weight and speed has high positive weight showing that people generally prefer safe and efficient driving (see Fig. 7.7). However, features like preferences to stay on the road and alignment with the road vary more. While the general direction of the feature weights is similar between $\hat{w_1}$ and $\hat{w_2}$ for each user, there is some difference in the magnitudes. We computed the percentage difference between average reward with respect to $\hat{w_1}$ and $\hat{w_2}$ as $\frac{\hat{w_1}.\overline{\phi} - \hat{w_2}.\overline{\phi}}{\hat{w_1}.\overline{\phi}}$, where $\overline{\phi}$ is the average feature values for our application. This gave

us the percentage difference in the average reward. We found that of all the users the maximum difference is 12% and the minimum difference is 6%. While we also learnt $\hat{alpha}_1$, it is relatively less important here as $\hat{w}_1$ and $\hat{w}_2$ are close.

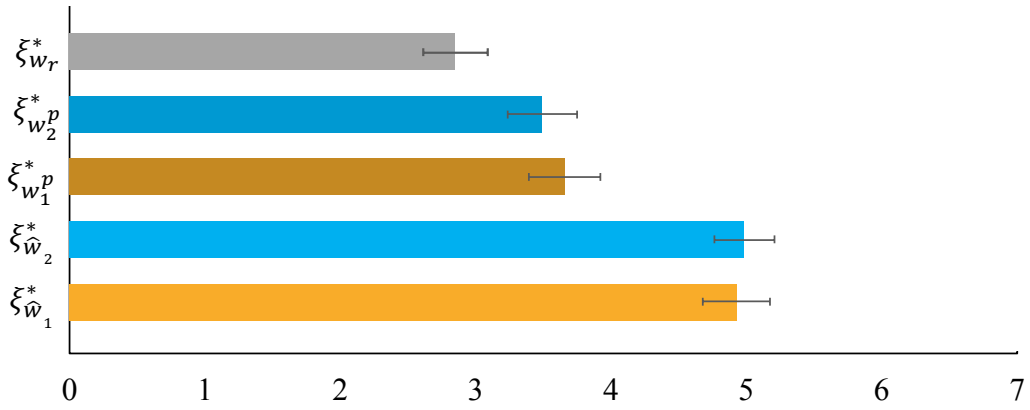


Figure 7.8: Most users gave high ratings to the trajectories optimal for $\hat{w}_1$ and $\hat{w}_2$ and low ratings to trajectories optimal for their perturbed versions w_1^p and w_2^p and the lowest rating to the trajectories that were optimal with respect to some random weight w_r.

As it can be seen in Fig. 7.8, the users gave the highest scores to the trajectories generated from $\hat{w}_1$ and $\hat{w}_2$ with statistical significance. This suggests an empirical evidence for **H1**. While we also observed that users sometimes gave high ratings to $\hat{w}_1$ and sometimes to $\hat{w}_2$, we have not observed a significant dependence on the modes. This is due to the fact that the learned weights were very close to each other as they represent the legal driving behavior, which is a very small subset of all the reward space.

7.8 Chapter Summary

In this chapter, we developed a model of how humans change their moods (latent states) based on the interactions they are involved in the environment, as well as how they respond to the given queries when they are in a particular mood. Using this model, we developed a maximum volume removal based active learning algorithm to efficiently learn the reward functions and mode transitions. We demonstrated through simulations and user studies that this framework can efficiently learn expressive reward dynamics.

Limitations. While the framework is very general, we tested it only for driving environment. In fact, because legal driving is a very small subset of the space of static reward functions, we observed in our experiments that queries focus mostly on learning this small subset, and we would need higher number of queries to recover different modes within this set. More extensive experiments on various environments

where all reward functions are sound but the personalization is more important could give more interesting results.

We also used a specific set of parameters $k = 2$, $s = 2$. While larger s values can ease learning, larger k values require further research, because there is currently no theoretical guarantee of identifiability (139) even with the relaxed problem of multimodal reward learning we presented.

In this thesis we attempted to find some solutions to the problem of *personalizing autonomous driving* from within the domain of *human-robot interaction*. We found that unlike popular belief in the research community, learning from user's own driving demonstrations is not an adequate solution. We turned to alternative means of human guidance for learning driving preferences. We developed the concept of *rich queries* as augmented comparison-queries and leveraged two forms of rich queries: a feature-augmented comparison query in the previous chapter and hierarchical comparison query in this chapter to learn complex dynamic user preferences. This chapter brings us to last contribution of this thesis. However, there are many more things to say: *the other possible directions* that this thesis could have taken and *future work* that can continue from this thesis, in the concluding remarks next.

Chapter 8

Concluding Remarks

8.1 Summary of Contributions

Autonomous cars are a reality of today. But only a handful of them can drive safely on an already charted route. Like several other systems of autonomy, they were designed for a safer and better future of mobility. There are, however, several challenges to mass adoption of the system which include initial software failures (85), lack of trust in automation (2) and the safety-critical nature of the system in itself (71). One thing to remember is that these systems do not exist in vacuum. Humans are in continuous interaction with autonomy and many of the above mentioned challenges can be overcome by a designing systems that humans can understand better, operate effortlessly and interrupt if necessary (61). It is, however, difficult to guarantee continuous supervision. It defeats the purpose of having the systems in the first place. A better model is to build into the autonomous systems, an ability to understand what humans are thinking and how humans want them to act. These systems should be able to reason about what humans need to know in order to provide efficient and effective supervision. For example, an autonomous car, on encountering an unknown scenario where it doesn't know how to act, should be able to make the supervisor aware of the situation. Likewise, it should know to avoid drastic unanticipated maneuvers that can surprise or scare an user. But for that the car should know what maneuver is unpredictable for its user. In other words, it needs to know the user's preferred driving behavior. The field of human-robot interaction addresses some of the fundamental challenges in realizing safe human robot co-existence. In this thesis, we borrow methods and concepts from human-robot interaction and interactive machine learning to enable an autonomous car to learn about it's user's preference. We formally define personalizing autonomous driving as a *behavior planning*, where the goal is for the car is to follow trajectories like lane change, car following, merging etc. that match the user's preferences. Chapter.1. We show empirically that the traditional approach to learning from expert driving demonstrations does not work in our case. This is because autonomous driving preferences do not match individual demonstrations Chapter.3. We propose to use alternative *richer* forms of human

guidance for personalization. We model humans as noisily rational agents whose preferences can be encoded by a mixture of reward functions Chapter.2. The most important contribution of this thesis is a general framework that expands the scope of Preference-based reward learning (48) to include other forms of human guidance, leading to more efficient learning. Within the limited scope of this thesis we have explored two different types of *richer* guidance: i) feature augmented comparison queries Chapter.6 and ii) hierarchical comparison queries Chapter.7. As I close this thesis, more and more researchers in AI and robotics are adopting Preference learning (33; 44). We hope that our proposed framework will foster further research in interactive preference learning: incorporating novel forms of human-robot learning interactions and will be applied to other dynamical systems with continuous state-space besides autonomous car.

8.2 Value Aligned AI

In this thesis, we model our autonomous car as a *rational agent* that perceives and acts to maximize a *Utility function*, also called *reward function*. When designing such an agent it is important to carefully imbue the *correct* objective function in the system, so as to avoid unindented consequences that can harm the society (9). The idea of *value alignment* suggests that for the AI agents to act in the best interest of the society, they should not just optimize any carefully designed reward function, but one that matches true human *values*. In other words, the agent's value function should be aligned with that of human (109). It is notoriously difficult to define such a reward function because we as humans do not always know what we desire (9; 62; 95). One approach to solve this problem is to learn a reward function by observing how people generally behave using *Inverse Reinforcement Learning*. But desires are not constant across people. Humans are diverse in both preferences and circumstances (109). Here we attempt to build an autonomous car whose *values* are aligned with the user's individual *preferences*. We learn from *Cooperative Inverse Reinforcement Learning* (CIRL), which extends the IRL framework to incorporate human robot collaboration in the process of value alignment (60). Similar to CIRL, we design preference learning as a two-player partial information game where the human is noisily aware of his reward function, but the robot is not. The robot's payoff is exactly same as the human's reward function. The robot queries the human and the human provides answer that reflects the true preferences. While original CIRL framework considers learning a static reward function, we design our queries with a *hierarchical* structure that can capture the variations within the individual preferences in the same game. That said, our approach to learning a mixture of reward functions had several challenges, both in theoretical and pragmatic senses. Theoretically, while our approximations mitigated some of the issues related to label-switching in mixture models, it slowed down the overall convergence of the algorithm. Practically, we were unable to capture the emotions that lead to preference changes in the real users. This brings us back to the problem of complexity of human desires, values and intents and

challenges associated with learning these hidden states thereof. This is even more true for safety-critical systems like autonomous cars where we have to rely on simulations. For better value aligned AI systems, research on algorithms for collaborative human-robot learning (72; 60; 62) should go hand in hand with development of high fidelity simulation environments that can elicit true human preferences.

8.3 Models of Human Behavior

We treat user-autonomous car as a human-robot system where successful robot operation is dependent on a learning an accurate mental model of human preference. The robot starts out with two model assumptions: i) humans are noisily rational and ii) the distribution over choices follow the principle of maximum entropy. As we see in Chapter.6, the efficiency of this learning is largely governed by the hyperparameter of the model: the *rationality coefficient*. In this work, this hyperparameter represented how accurate people are in their answers to comparison and feature queries. While, research shows model-based approaches to robot learning are more sample efficient (49; 42), it also comes with the limitations stemming from the model assumptions and the data quality. Data quality refers to expressiveness of the features that constitute the linear reward model and diversities of the scenarios for which the human model is learnt. Reward functions representing human preferences are as good as the expressiveness of the features that form it. In this thesis, the hand-coded features focused mainly on legal driving which is a small subset of the space of true preferences. We note that the performance of the same algorithm may vary in real life depending on the expressiveness of the features. We also found that for interactive learning the features should be human interpretable. In the current self-driving industry, it is impossible to avoid deep learning algorithms, where we can learn highly expressive features. But interpretability remains a major challenge with these algorithms. In future, further research can investigate the trade-off between expressiveness and interpretability of features encoding human preferences for autonomous driving.

Bounded Rationality. The whole thesis is based on the same assumption about human behavior as most modern human-robot interaction algorithms (140; 141; 49), i.e. humans are *imperfect rational agents*. While many of our day to day driving decisions may be rational, in extreme events like accidents, human decisions cannot be explained as outcomes of rational decision making. Learning true preferences for such edge cases in itself is a difficult and controversial task (13). In this work, we do not attempt to learn how people would prefer their cars to drive in case of an imminent accident. However, in future we can borrow from theories in behavioral economics (75) to design autonomous systems like cars that account for limitations of humans' rationality, also called *bounded rationality*. Such models can also enable estimating probability of loss in the events of accidents.

Biased Rationality. We also know that human rationality is not uniformly noisy. It has *systematic* biases like risk aversion, myopia etc. A proof of concept in this thesis is that people can answer certain types of queries more easily and accurately

than others even though they carry the same amount of information for both types of answers (see Chapter.6). Other forms of biases could arise in our interaction, if we made people aware that they were in fact teaching the robot (57). For example, instead of actually revealing the true preferences that can include both altruistic and selfish behavior (128; 24), the teacher may only pick options that show socially nice behavior. In a human robot interaction scenario as ours or CIRL where the goal for the robot is to exploit the model of human rationality to learn faster, it may not be sufficient to learn or rely on just one model (of perfect rationality or Boltzmann rationality). There are IRL algorithms that account for *systematic* biases (53; 91). Choosing biased models may have worse risks than choosing models with uniform noise assumption (94). But one of the reasons that we stick to the model of noisy rationality is that it is extremely challenging to learn reward functions for humans with unknown rationality models (10). A recent solution to learn the demonstrator's planning algorithm as a differentiable planner (123) can overcome this limitation. Future work can incorporate concepts from *meta-learning* within the CIRL framework for learning models of biased preferences.

8.4 Collaborative Human-Robot Learning and Trust

In this thesis, we enable the robot (autonomous car) to learn about human preferences or reward dynamics interactively. The human is, however, not aware of the ultimate goal of the robot, which is to match the human reward function or more generally act according to human preference. Alternative formulations of human-robot collaborative learning game (60; 62; 57) enable humans to explicitly act as a teacher. Research shows that in such a game the objective function that human optimizes for is different from that when they are just indicating preference over choices. As discussed in the previous paragraph, making the teacher aware of his teaching role may lead to biased preference learning. However, there is also a possibility that the teaching role can make the user think harder about his own preferences. Other formulations allow users to even discover their own preferences collaboratively with a robot within the multi-armed bandit framework (40). In future, it would be worth exploring different versions of the human-robot interaction game within our richer preference learning framework and investigate how these interactions affect user trust. Our work shows that *richer* queries, especially human interpretable feature queries can enhance trust in the robot making such queries, by exposing the internal workings of the system. This opens up a possibility of further investigations into how interactive machine learning algorithms can improve human trust by the virtue of their design without having to deliberately communicate the internal workings (72; 71).

8.5 Final Words

Our work in this thesis is a first step towards enabling autonomous cars to drive following human preferences. We treated the user-autonomous car system as a human-robot system and designed algorithms to efficiently and interactively learn human preferences. We showed empirically and through real user studies that our learned model represents true preferences accurately. That said, there are still existing challenges that need to be addressed for improving the generalizability of the learned model across diverse environments. We need to better account for variations in individual behavior in AI. Our approach is just one way to speed up comparison-based learning. More recent works combined IRL with preference-based learning (101) or used batch active learning to speed up the query selection part (25; 26). In order to ensure reliability of performance of such safety-critical systems as autonomous cars, we must test our algorithms in realistic simulation environments. Autonomous cars of today know to drive safely on a known route, treating humans as obstacles or bounded disturbances, leading to conservative and uninterpretable driving behaviors. When it comes to functional aspects of driving like perception, obstacle avoidance and path planning, these systems are approaching near perfection and almost ready for real world deployment. A bigger question is what happens when not just one in hundred other cars is autonomous, but when more than 50 % of the traffic is autonomous. We believe that at that point functional driving must be augmented to account for individual sense of safety and risk aversion. Even in ride sharing scenario, the cars should learn to drive following some population preference model.

Bibliography

[1] pywren. `http://pywren.io/`, 2017. Accessed: 2017-12-24.

[2] AAA. Three in four americans remain afraid of fully self-driving vehicles. `https://newsroom.aaa.com/2019/03/americans-fear-self-driving-cars-survey/`, 2019.

[3] Pieter Abbeel, Dmitri Dolgov, Andrew Y Ng, and Sebastian Thrun. Apprenticeship learning for motion planning with application to parking lot navigation. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1083–1090. IEEE, 2008.

[4] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

[5] Nir Ailon and Mehryar Mohri. Preference-based learning to rank. *Machine Learning*, 80(2):189–211, 2010.

[6] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[7] Riad Akrour, Marc Schoenauer, and Michèle Sebag. April: Active preference learning-based reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 116–131. Springer, 2012.

[8] Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

[9] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016.

[10] Stuart Armstrong and Sören Mindermann. Impossibility of deducing preferences and rationality from human policy. *CoRR*, abs/1712.05812, 2017.

[11] ASHRAE. Standard 55 - thermal environmental conditions for human occupancy, 2010.

[12] The Atlantic. The trick that makes google's self-driving cars work. `https://www.theatlantic.com/technology/archive/2014/05/all-the-world-a-track-the-trick-that-makes-googles-self-driving-cars-work/370871/`, 2014.

[13] Edmond Awad, Sohan Dsouza, Richard Kim, Jonathan Schulz, Joseph Henrich, Azim Shariff, Jean-François Bonnefon, and Iyad Rahwan. The moral machine experiment. *Nature*, 563(7729):59, 2018.

[14] Haoyu Bai, Shaojun Cai, Nan Ye, David Hsu, and Wee Sun Lee. Intention-aware online pomdp planning for autonomous driving in a crowd. In *2015 ieee international conference on robotics and automation (icra)*, pages 454–460. IEEE, 2015.

[15] Andrea Bajcsy, Dylan P Losey, Marcia K O?Malley, and Anca D Dragan. Learning robot objectives from physical human interaction. In *Conference on Robot Learning*, pages 217–226, 2017.

[16] Chris Baker and Joshua B. Tenenbaum. Modeling human plan recognition using bayesian theory of mind. *Plan, Activity, and Intent Recognition: Theory and Practice*, pages 177–204, 03 2014.

[17] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Algorithmic foundations of robotics X*, pages 475–491. Springer, 2013.

[18] Nikola Banovic, Tofi Buzali, Fanny Chevalier, Jennifer Mankoff, and Anind K Dey. Modeling and understanding human routine behavior. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 248–260. ACM, 2016.

[19] Chandrayee Basu, Erdem Bıyık, Zhixun He, Mukesh Singhal, and Dorsa Sadigh. Active learning of reward dynamics from hierarchical queries. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[20] Chandrayee Basu, Mukesh Singhal, and Anca D Dragan. Learning from richer human guidance: Augmenting comparison-based learning with feature queries. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 132–140. ACM, 2018.

[21] Chandrayee Basu, Qian Yang, David Hungerman, Mukesh Singhal, and Anca D Dragan. Do you want your autonomous car to drive like you? In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 417–425. ACM, 2017.

[22] Holger Berndt, Jorg Emmert, and Klaus Dietmayer. Continuous driver intention recognition with hidden markov models. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*, pages 1189–1194. IEEE, 2008.

[23] Aaron Bestick, Ruzena Bajcsy, and Anca D Dragan. Implicitly assisting humans to choose good grasps in robot to human handovers. In *International Symposium on Experimental Robotics*, pages 341–354. Springer, 2016.

[24] Erdem Bıyık, Daniel Lazar, Ramtin Pedarsani, and Dorsa Sadigh. Altruistic autonomy: Beating congestion on shared roads. *arXiv preprint arXiv:1810.11978*, 2018.

[25] Erdem Bıyık and Dorsa Sadigh. Batch active preference-based learning of reward functions. In *Conference on Robot Learning*, pages 519–528, 2018.

[26] Erdem Bıyık, Kenneth Wang, Nima Anari, and Dorsa Sadigh. Batch active learning using determinantal point processes. *arXiv preprint arXiv:1906.07975*, 2019.

[27] Mariusz Bojarski, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Larry D. Jackel, Urs Muller, and Karol Zieba. Visualbackprop: visualizing cnns for autonomous driving. *CoRR*, abs/1611.05418, 2016.

[28] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[29] Mark Brackstone and Mike McDonald. Car-following: a historical review. *Transportation Research Part F: Traffic Psychology and Behaviour*, 2(4):181–196, 1999.

[30] Darius Braziunas. Computational approaches to preference elicitation. *Department of Computer Science, University of Toronto, Tech. Rep*, 2006.

[31] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic mdp-behavior planning for cars. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1537–1542. IEEE, 2011.

[32] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 392–399. IEEE, 2014.

[33] Daniel S. Brown and Scott Niekum. Machine teaching for inverse reinforcement learning: Algorithms and applications. *CoRR*, abs/1805.07687, 2018.

[34] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[35] Róbert Busa-Fekete, B Szörenyi, P Weng, W Cheng, and E Hüllermeier. Preference-based reinforcement learning. In *European Workshop on Reinforcement Learning, Dagstuhl Seminar*, 2013.

[36] Maya Cakmak and Andrea L Thomaz. Optimality of human teachers for robot learners. In *Development and Learning (ICDL), 2010 IEEE 9th International Conference on*, pages 64–69. IEEE, 2010.

[37] Maya Cakmak and Andrea L Thomaz. Active learning with mixed query types in learning from demonstration. In *in Proc. of the ICML Workshop on New Developments in Imitation Learning*. Citeseer, 2011.

[38] Maya Cakmak and Andrea L. Thomaz. Designing robot learners that ask good questions. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '12, pages 17–24, New York, NY, USA, 2012. ACM.

[39] Maya Cakmak and Andrea L Thomaz. Eliciting good teaching from humans for machine learners. *Artificial Intelligence*, 217:198–215, 2014.

[40] Lawrence Chan, Dylan Hadfield-Menell, Siddhartha S. Srinivasa, and Anca D. Dragan. The assistive multi-armed bandit. *CoRR*, abs/1901.08654, 2019.

[41] Flavio Chierichetti, Ravi Kumar, and Andrew Tomkins. Learning a mixture of two multinomial logits. In *International Conference on Machine Learning*, pages 960–968, 2018.

[42] Rohan Choudhury, Gokul Swamy, Dylan Hadfield-Menell, and Anca D Dragan. On the utility of model learning in hri. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 317–325. IEEE, 2019.

[43] Paul Christiano, Jan Leike, Tom B Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *arXiv preprint arXiv:1706.03741*, 2017.

[44] Yuchen Cui and Scott Niekum. Active reward learning from critiques. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6907–6914, 2018.

[45] Christian Daniel, Malte Viering, Jan Metz, Oliver Kroemer, and Jan Peters. Active reward learning. In *Robotics: Science and Systems*, 2014.

[46] Ernst Dieter Dickmanns and Volker Graefe. Dynamic monocular machine vision. *Machine vision and applications*, 1(4):223–240, 1988.

[47] Weishan Dong, Jian Li, Renjie Yao, Changsheng Li, Ting Yuan, and Lanjun Wang. Characterizing driving styles with deep learning. *arXiv preprint arXiv:1607.03611*, 2016.

[48] Anca D Dragan Dorsa Sadigh, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems (RSS)*, 2017.

[49] Anca D Dragan. Robot planning with mathematical models of human state and action. *arXiv preprint arXiv:1705.04226*, 2017.

[50] Anca D Dragan, Kenton CT Lee, and Siddhartha S Srinivasa. Legibility and predictability of robot motion. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pages 301–308. IEEE Press, 2013.

[51] Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 81–90. Association for Computational Linguistics, 2009.

[52] Mohammed Elhenawy, Arash Jahangiri, Hesham A Rakha, and Ihab El-Shawarby. Modeling driver stop/run behavior at the onset of a yellow indication considering driver run tendency and roadway surface conditions. *Accident Analysis & Prevention*, 83:90–100, 2015.

[53] Owain Evans, Andreas Stuhlmüller, and Noah Goodman. Learning the preferences of ignorant, inconsistent agents. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[54] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on control systems technology*, 15(3):566–580, 2007.

[55] Paolo Falcone, H Eric Tseng, Jahan Asgari, Francesco Borrelli, and Davor Hrovat. Integrated braking and steering model predictive control approach in autonomous vehicles. *IFAC Proceedings Volumes*, 40(10):273–278, 2007.

[56] Jaime F Fisac, Eli Bronstein, Elis Stefansson, Dorsa Sadigh, S Shankar Sastry, and Anca D Dragan. Hierarchical game-theoretic planning for autonomous vehicles. *arXiv preprint arXiv:1810.05766*, 2018.

[57] Jaime F. Fisac, Monica A. Gates, Jessica B. Hamrick, Chang Liu, Dylan Hadfield-Menell, Malayandi Palaniappan, Dhruv Malik, S. Shankar Sastry, Thomas L. Griffiths, and Anca D. Dragan. Pragmatic-pedagogic value alignment. *CoRR*, abs/1707.06354, 2017.

[58] Johannes Frnkranz and Eyke Hllermeier. *Preference Learning*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[59] Andrew Gray, Yiqi Gao, J Karl Hedrick, and Francesco Borrelli. Robust predictive control for semi-autonomous vehicles with an uncertain driver model. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 208–213. IEEE, 2013.

[60] Dylan Hadfield-Menell, Anca D. Dragan, Pieter Abbeel, and Stuart J. Russell. Cooperative inverse reinforcement learning. *CoRR*, abs/1606.03137, 2016.

[61] Dylan Hadfield-Menell, Anca D. Dragan, Pieter Abbeel, and Stuart J. Russell. The off-switch game. *CoRR*, abs/1611.08219, 2016.

[62] Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J. Russell, and Anca D. Dragan. Inverse reward design. *CoRR*, abs/1711.02827, 2017.

[63] Frank Havlak and Mark Campbell. Discrete and continuous, probabilistic anticipation for autonomous robots in urban environments. *IEEE Transactions on Robotics*, 30(2):461–474, 2013.

[64] HERE360. How to humanize the autonomous car. `http://360.here.com/2015/04/23/humanized-driving/`, 2015.

[65] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pages 4565–4573, 2016.

[66] Rachel Holladay, Shervin Javdani, Anca Dragan, and Siddhartha Srinivasa. Active comparison based learning incorporating user uncertainty and noise. In *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.

[67] Rachel Holladay, Shervin Javdani, Anca Dragan, and Siddhartha Srinivasa. Active comparison based learning incorporating user uncertainty and noise. In *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.

[68] EA Holmes, MB Bonsall, SA Hales, H Mitchell, F Renner, SE Blackwell, P Watson, GM Goodwin, and M Di Simplicio. Applications of time-series analysis to mood fluctuations in bipolar disorder to promote treatment innovation: a case series. *Translational Psychiatry*, 6(1):e720, 2016.

[69] Jin-Hyuk Hong, Ben Margines, and Anind K Dey. A smartphone-based sensing platform to model aggressive driving behaviors. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 4047–4056. ACM, 2014.

[70] Mark S Horswill and Frank P McKenna. The effect of perceived control on risk taking1. *Journal of Applied Social Psychology*, 29(2):377–391, 1999.

[71] Sandy H Huang, Kush Bhatia, Pieter Abbeel, and Anca D Dragan. Establishing appropriate trust via critical states. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3929–3936. IEEE, 2018.

[72] Sandy H Huang, David Held, Pieter Abbeel, and Anca D Dragan. Enabling robots to communicate their objectives. *Autonomous Robots*, 43(2):309–326, 2019.

[73] Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(16):1897–1916, 2008.

[74] Ashesh Jain, Shikhar Sharma, Thorsten Joachims, and Ashutosh Saxena. Learning preferences for manipulation tasks from online coactive feedback. *The International Journal of Robotics Research*, 34(10):1296–1313, 2015.

[75] Daniel Kahneman. A perspective on judgment and choice: mapping bounded rationality. *American psychologist*, 58(9):697, 2003.

[76] Amin Karbasi, Stratis Ioannidis, et al. Comparison-based learning with rank nets. *arXiv preprint arXiv:1206.4674*, 2012.

[77] Jinkyu Kim and John F. Canny. Interpretable learning for self-driving cars by visualizing causal attention. *CoRR*, abs/1703.10631, 2017.

[78] Henrik Kretzschmar, Markus Spies, Christoph Sprunk, and Wolfram Burgard. Socially compliant mobile robot navigation via inverse reinforcement learning. *The International Journal of Robotics Research*, 35(11):1289–1307, 2016.

[79] Markus Kuderer, Shilpa Gulati, and Wolfram Burgard. Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2641–2646. IEEE, 2015.

[80] Dana Kulic and Elizabeth A Croft. Affective state estimation for human–robot interaction. *IEEE Transactions on Robotics*, 23(5):991–1000, 2007.

[81] Chi-Pang Lam, Allen Y Yang, and S Shankar Sastry. An efficient algorithm for discrete-time hidden mode stochastic hybrid systems. In *Control Conference (ECC), 2015 European*, pages 1212–1218. IEEE, 2015.

[82] Nicholas C Landolfi and Anca D Dragan. Social cohesion in autonomous driving. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8118–8125. IEEE, 2018.

[83] Seong Jae Lee and Zoran Popović. Learning behavior styles with inverse reinforcement learning. *ACM transactions on graphics (TOG)*, 29(4):122, 2010.

[84] Suzanne E Lee, Erik CB Olsen, and Walter W Wierwille. A comprehensive examination of naturalistic lane-changes. Technical report, Virginia Tech Transportation Institute, 2004.

[85] Timothy B. Lee. Software bug led to death in uber's self-driving crash. `https://arstechnica.com/tech-policy/2018/05/report-software-bug-led-to-death-in-ubers-self-driving-crash/`, 2018.

[86] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168. IEEE, 2011.

[87] Allen Liu and Ankur Moitra. Efficiently learning mixtures of mallows models. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 627–638. IEEE, 2018.

[88] Tyler Lu and Craig Boutilier. Effective sampling and learning for mallows models with pairwise-preference data. *The Journal of Machine Learning Research*, 15(1):3783–3829, 2014.

[89] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

[90] Brandon D Luders, Mangal Kothariyand, and Jonathan P How. Chance constrained rrt for probabilistic robustness to environmental uncertainty. 2010.

[91] Anirudha Majumdar, Sumeet Singh, Ajay Mandlekar, and Marco Pavone. Risk-sensitive inverse reinforcement learning via coherent risk models. In *Robotics: Science and Systems*, 2017.

[92] Hiren M Mandalia and Mandalia Dario D Salvucci. Using support vector machines for lane-change detection. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 49, pages 1965–1969. SAGE Publications, 2005.

[93] Smitha Milli, Pieter Abbeel, and Igor Mordatch. Interpretable and pedagogical examples. *CoRR*, abs/1711.00694, 2017.

[94] Smitha Milli and Anca D. Dragan. Literal or pedagogic human? analyzing human model misspecification in objective learning. *CoRR*, abs/1903.03877, 2019.

[95] Sören Mindermann, Rohin Shah, Adam Gleave, and Dylan Hadfield-Menell. Active inverse reward design. *arXiv preprint arXiv:1809.03060*, 2018.

[96] Jeremy Morton and Mykel J Kochenderfer. Simultaneous policy learning and latent state inference for imitating driver behavior. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2017.

[97] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, pages 663–670, 2000.

[98] Stefanos Nikolaidis, David Hsu, and Siddhartha Srinivasa. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36(5-7):618–634, 2017.

[99] OpenDS. Welcome to opends 4.0! `https://www.opends.eu/home`, 2016.

[100] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[101] Malayandi Palan, Nicholas C. Landolfi, Gleb Shevchuk, and Dorsa Sadigh. Learning reward functions by integrating human demonstrations and preferences. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2019.

[102] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems*, pages 305–313, 1989.

[103] Hema Raghavan, Omid Madani, and Rosie Jones. Active learning with feedback on features and instances. *Journal of Machine Learning Research*, 7(Aug):1655–1686, 2006.

[104] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2586–2591, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[105] Vasumathi Raman, Alexandre Donzé, Mehdi Maasoumy, Richard M Murray, Alberto Sangiovanni-Vincentelli, and Sanjit A Seshia. Model predictive control with signal temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 81–87. IEEE, 2014.

[106] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736. ACM, 2006.

[107] Stéphane Ross, Narek Melik-Barkhudarov, Kumar Shaurya Shankar, Andreas Wendel, Debadeepta Dey, J Andrew Bagnell, and Martial Hebert. Learning monocular reactive uav control in cluttered natural environments. In *2013 IEEE international conference on robotics and automation*, pages 1765–1772. IEEE, 2013.

[108] Constantin A Rothkopf and Christos Dimitrakakis. Preference elicitation and inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 34–48. Springer, 2011.

[109] Stuart J Russell. Provably beneficial artificial intelligence. 2017.

[110] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems (RSS)*, 2017.

[111] Dorsa Sadigh and Ashish Kapoor. Safe control under uncertainty. *CoRR*, abs/1510.07313, 2015.

[112] Dorsa Sadigh, Nick Landolfi, Shankar S Sastry, Sanjit A Seshia, and Anca D Dragan. Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state. *Autonomous Robots*, 42(7):1405–1426, 2018.

[113] Dorsa Sadigh, S Shankar Sastry, and Sanjit A Seshia. Verifying robustness of human-aware autonomous cars. *IFAC-PapersOnLine*, 51(34):131–138, 2019.

[114] Dorsa Sadigh, S Shankar Sastry, Sanjit A Seshia, and Anca Dragan. Information gathering actions over human internal state. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 66–73. IEEE, 2016.

[115] Dorsa Sadigh, S Shankar Sastry, Sanjit A Seshia, and Anca Dragan. Information gathering actions over human internal state. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 66–73. IEEE, 2016.

[116] Dorsa Sadigh, Shankar Sastry, Sanjit A Seshia, and Anca D Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and Systems*, volume 2. Ann Arbor, MI, USA, 2016.

[117] Khaled Saleh, Mohammed Hossny, and Saeid Nahavandi. Real-time intent prediction of pedestrians for autonomous ground vehicles via spatio-temporal densenet. *arXiv preprint arXiv:1904.09862*, 2019.

[118] Dario D Salvucci. Inferring driver intent: A case study in lane-change detection. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 48, pages 2228–2231. SAGE Publications, 2004.

[119] Svenja Scherer, André Dettmann, Franziska Hartwich, Timo Pech, Angelika C Bullinger, Josef F Krems, and Gerd Wanielik. How the driver wants to be driven-modelling driving styles in highly automated driving. *Tagungsband*, 7:2015–26, 2015.

[120] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.

[121] Volkan Sezer, Tirthankar Bandyopadhyay, Daniela Rus, Emilio Frazzoli, and David Hsu. Towards autonomous navigation of unsignalized intersections under uncertainty of human driver intent. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3578–3585. IEEE, 2015.

[122] Patrick Shafto, Noah D Goodman, and Thomas L Griffiths. A rational account of pedagogical reasoning: Teaching by, and learning from, examples. *Cognitive psychology*, 71:55–89, 2014.

[123] Rohin Shah, Noah Gundotra, Pieter Abbeel, and Anca Dragan. On the feasibility of learning, rather than assuming, human biases for reward inference. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5670–5679, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[124] David Silver, J Andrew Bagnell, and Anthony Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *The International Journal of Robotics Research*, 2010.

[125] Jiaming Song, Hongyu Ren, Dorsa Sadigh, and Stefano Ermon. Multi-agent generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pages 7472–7483, 2018.

[126] Bradly C Stadie, Pieter Abbeel, and Ilya Sutskever. Third-person imitation learning. *arXiv preprint arXiv:1703.01703*, 2017.

[127] Elis Stefansson, Jaime Fisac, Dorsa Sadigh, Shankar Sastry, and Karl H. Johansson. Human-robot interaction for truck platooning using hierarchical dynamic games. In *European Control Conference (ECC)*, June 2019.

[128] Liting Sun, Wei Zhan, Masayoshi Tomizuka, and Anca D Dragan. Courteous autonomous cars. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 663–670. IEEE, 2018.

[129] Orit Taubman-Ben-Ari, Mario Mikulincer, and Omri Gillath. The multidimensional driving style inventory–scale construct and validation. *Accident Analysis & Prevention*, 36(3):323–332, 2004.

[130] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846. IEEE, 1998.

[131] Simon Ulbrich and Markus Maurer. Probabilistic online pomdp decision making for lane changes in fully automated driving. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 2063–2067. IEEE, 2013.

[132] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[133] Hanneke Hooft van Huysduynen, Jacques Terken, Jean-Bernard Martens, and Berry Eggen. Measuring driving styles: a validation of the multidimensional driving style inventory. In *Proceedings of the 7th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 257–264. ACM, 2015.

[134] Aaron Wilson, Alan Fern, and Prasad Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1133–1141. Curran Associates, Inc., 2012.

[135] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. *CoRR*, abs/1612.01079, 2016.

[136] Li Xu, Jie Hu, Hong Jiang, and Wuqiang Meng. Establishing style-oriented driver models by imitating human driving behaviors. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2522–2530, 2015.

[137] Sze Zheng Yong, Minghui Zhu, and Emilio Frazzoli. Generalized innovation and inference algorithms for hidden mode switched linear stochastic systems with unknown inputs. In *53rd IEEE Conference on Decision and Control*, pages 3388–3394. IEEE, 2014.

[138] Nidzamuddin Md. Yusof and Juffrizal Karjanto. Comfort determination in autonomous driving style. In *3rd Workshop on User Experience of Autonomous Vehicles at AutoUI'15*, pages 257–264. ACM, 2015.

[139] Zhibing Zhao, Peter Piech, and Lirong Xia. Learning mixtures of plackett-luce models. In *International Conference on Machine Learning*, pages 2906–2914, 2016.

[140] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[141] Brian D Ziebart, Andrew L Maas, Anind K Dey, and J Andrew Bagnell. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 322–331. ACM, 2008.