Learning Objective Functions from Many Diverse Signals

Smitha Milli

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2022-190
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-190.html

August 10, 2022

Learning Objective Functions from Many Diverse Signals

by

Smitha Milli

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Associate Professor Anca D. Dragan, Co-chair
Associate Professor Moritz Hardt, Co-chair
Professor Stuart J. Russell
Professor Thomas L. Griffiths

Summer 2022

Learning Objective Functions from Many Diverse Signals

Abstract

Learning Objective Functions from Many Diverse Signals

by

Smitha Milli

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Berkeley

Associate Professor Anca D. Dragan, Co-chair

Associate Professor Moritz Hardt, Co-chair

Specifying the correct objective function for a machine learning system is often difficult and error-prone. This thesis focuses on *learning* objective functions from many kinds of human inputs. It is comprised of three parts. First, we present our formalism, *reward-rational choice* (RRC), that unifies reward learning from many diverse signals. The key insight is that human behavior can often be modeled as a reward-rational implicit choice – a choice from an implicit set of options, that is approximately rational for the intended reward. We show how much of the prior work, despite using many diverse modalities of feedback, can be seen as an instantiation of RRC.

In the second part, we discuss implications of the RRC formalism. In particular, it allows us to learn from multiple feedback types *at once*. Through case studies and experiments, we show how RRC can be used to combine and actively select from feedback types. Furthermore, once a person has access to multiple types of feedback, even their choice of feedback type itself provides information to learn the reward function from. We use RRC to formalize and learn from a person's *meta-choice*, the choice of feedback type itself.

Finally, in the third part, we study settings in which the human may violate the reward-rational assumption. First, we consider the case where the human may be pedagogic, i.e., optimizing for teaching the reward function. We show that the reward-rational assumption provides robust reward inference even when the human is pedagogic. Second, we consider the case where the human may face temptation and act in ways that systematically deviate from their target preferences. We theoretically analyze such a setting and show that, with the right feedback type, one can still efficiently recover the individual's preferences. Lastly, we consider the recommender system setting. There, it is difficult to model all user behaviors as rational, but by leveraging one strong, explicit signal (e.g. "don't show me this"), we are still able to operationalize and optimize for a notion of "value" on these systems.

For Charlie.

We miss you.

Contents

Acknowledgments

First, I would like to thank my PhD advisors, Anca Dragan and Moritz Hardt. Your guidance over the past several years has made me a much better researcher. You have helped me to not only solve problems, but more importantly, develop a taste for choosing them. There were many bumps along the way, and I appreciate your patience and continued belief in me during this meandering journey.

Next, I would like to thank the members of my committee, Stuart Russell and Tom Griffiths, both of whom I had the privilege to work with during my undergraduate years at UC Berkeley. I first became interested in the theme of this thesis – learning objective functions – in 2016 because of Stuart frankly pointing out how misguided the "standard model" of AI, which takes for granted a correctly specified objective function, is. From Tom, I learned many fascinating ideas and frameworks from computational cognitive science that I continue to take inspiration from, both personally and in research.

And although I will not list them all, I would also like to thank all my other collaborators. I have learned from each and every one of them; it has been an honor to get to work with so many talented people.

Thank you to my friends for making the good times great and the bad times bearable: Emily Chao, Ishaan Gulrajani, Shiying He, Gail Hernandez, Miranda Li, Gloria Liu, Meg Majumder, Anna Papitto, Albert Pham, Joshua Price, Sophia Wang, Qiming Weng, Guy Wilson, Yifan Wu, Thomson Yeh, and Michael Zhang. I am so lucky and grateful to have you in my life.

Finally, I would like to thank my family: my cousin Naveen, brother Sachin, and parents, Suma and Laxman. Absolutely none of this would be possible without your endless support.

Berkeley is a weird, beautiful place that I love.

Chapter 1

Introduction

The field of machine learning has made significant progress in optimizing a known objective or reward function. For example, in game playing, where the objective is simple — win the game — machine learning systems have made huge strides, achieving human or superhuman performance in Go [Silver et al., 2016, 2017], Dota [Berner et al., 2019], and Atari [Mnih et al., 2013]. However, for most complex tasks, figuring out what the correct objective is remains a challenging bottleneck [Krakovna, 2018]. For example, it is difficult to directly specify how a self-driving car should trade off all the possible, relevant features of driving such as distance to other cars, following the speed limit, arriving at the destination, etc.

If specification is so hard, what is the alternative? Well, we want machine learning systems to do what *we* want them to do. Thus, the objective function, and information about it, must ultimately originate from people. Specification is just one way that people can give information about the objective function, but there are many others. For example, even if it is too difficult for someone to write down the objective function for a self-driving car, they can still provide demonstrations of driving, and we can use those demonstrations to infer the latent objective function [Ng and Russell, 2000, Abbeel and Ng, 2004]. And luckily, demonstrations are not the only source of information. We can also learn about the intended objective function by, say, asking people for comparisons between trajectories [Wirth et al., 2017, Sadigh et al., 2017, Christiano et al., 2017] or by grounding people's natural language instructions [MacGlashan et al., 2015, Fu et al., 2019].

Perhaps even more fortunate is that, even without being explicitly asked for it, people seem to reveal information left and right about the intended reward. For instance, if we push a robot away, this shouldn't just modify the robot's *current* behavior – it should also inform the robot about our preferences more *generally* [Jain et al., 2015, Bajcsy et al., 2017]. In particular, if someone turns a system off in a state of panic to avert a disaster, this shouldn't just stop the system right now. It should also inform the system about the intended reward function so that the system avoid the same disaster in the future: the robot should infer that whatever it was about to do has a tragically low reward. Even the *current state* of the world ought to inform the system about our preferences – it is a direct result of us having been acting in the world according to these preferences [Shah et al., 2019]! For instance, those

shoes didn't magically align themselves at the entrance: someone put effort into arranging them that way, so their state alone should tell the system something about what we want.

Overall, there are many, many signals that can be used to learn about the objective function. The immediate goal of this dissertation is to provide a *cohesive* and *unified* way to learn objective functions from many, diverse types of information at once. Learning a more accurate objective function will, in turn, make the system produce behavior that is more desirable — the ultimate goal. The rest of the dissertation is organized into three parts (see Figure 1.1 for an overview).

To begin, in Part I, we discuss our unifying formalism for learning reward functions. A reward function is an objective function that is additive over time and is typically used for autonomous agents such as a robot or a self-driving car. Researchers in reward learning[1] have proposed countless kinds of human behaviors, such as the ones already described, that can be used for learning reward functions, and surely, there are many more that have not yet been discovered. Is there a way we can view all these methods through one cohesive lens instead of re-inventing the wheel with each new type of information?

We contribute a formalism, *reward-rational choice*, which offers both a unifying lens with which to view past work, as well as a recipe for interpreting new sources of information that are yet to be uncovered. The key insight is that human behavior is a reward-rational implicit choice – a choice from an implicit set of options, which is approximately rational for the intended reward. We show that despite their diversity, many sources of information about reward functions can be characterized as instantiating this formalism, varying only in two components: 1) the set of options the person (implicitly) chose from, and 2) a grounding function that maps these options to robot behaviors.

In Part II, we focus on implications and applications of the formalism. In particular, RRC easily allows for the consideration of multiple feedback types at once. In Chapter 3, we show how the formalism can be used to both combine and actively select from feedback types. In a case study on a simulated robot, we learn from three different feedback types at once: proxy reward functions, physical corrections, and trajectory comparisons. To explore the benefit of actively selecting feedback types, we also run an experiment actively selecting between demonstrations and comparisons. Overall, we observe that demonstrations are optimal early on, when little is known about the reward function, while comparisons become optimal later, as a way to fine-tune the learned reward function.

[1]In reward learning, learning the reward function and acting based upon it are treated as two distinct stages. First, the system infers the reward function (or a distribution over reward functions), and then it takes actions while optimizing the expected reward under the inferred reward function (or distribution over reward functions). An assistance game is an alternative formulation that does not split up inference and action into distinct stages [Shah et al., 2020]. Our framework, reward-rational choice, is still just as relevant for assistance games. In an assistance game, the human is modeled as part of the environment and the reward function is modeled as a latent variable in the environment. The system learns about the latent reward function through the human's actions. To learn about the latent reward function in an assistance game, one still needs a model of how the human's actions relate to the reward function. The reward-rational choice framework provides a recipe for designing such a human model.

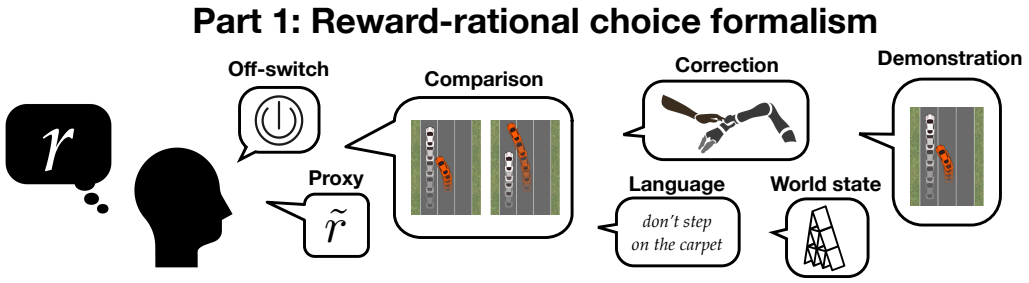

Figure 1.1: This dissertation is comprised of three parts. In the first part, we introduce our formalism, reward-rational choice which unifies reward learning from many diverse sources of information. In part two, we focus on applications of RRC to learn from multiple feedback types at once. In part three, we study settings in which the reward-rational assumption does not hold.

Once a human is no longer limited to only one feedback type, then even their choice of feedback type can convey information. For example, if a robot is about to do something disastrous, then a human may choose the sure-fire safeguard: turning it off immediately. On the other hand, if the robot is doing something wrong but not critically disastrous, then they may try to correct the robot, either physically or through language. Thus, the choice

of feedback type may itself be a useful signal for the robot to learn about the true reward function from. In Chapter 4, we instantiate the RRC framework to formalize *meta-choice*, the choice of feedback type, as a signal that can be learned from. Through experiments, we demonstrate the potential that meta-choice has for improving reward inference.

Finally, in Part III, we relax the reward-rational assumption, i.e., the assumption that the human's choices are rational with respect to the target reward function. We study three different settings in which the human may deviate in different ways from reward-rationality.

First, in Chapter 5, we study the setting where the human may be pedagogic, i.e., optimizing for teaching the reward function, rather than optimizing for reward itself. Here, we show that the reward-rational assumption still leads to robust inference even with a pedagogic human. We cast objective learning into the more general form of a common-payoff game between the robot and human, and prove that in any such game assuming the human is reward-rational is more robust than assuming that the human is pedagogic. Experiments with human data support our theoretical results and point to the sensitivity of the pedagogic assumption.

Next, in Chapter 6, we consider a setting in which the human may suffer from *temptation*, meaning that they may act in ways that conflict with what they would want upon reflection. In particular, we study a notable economic theory by Gul and Pesendorfer [2001] of individual choice under temptation. In this model, a decision-maker acts in two stages. In the first stage, the individual can anticipate temptation and restrict their choices for the second stage to a limited menu of items. In the second stage, the individual chooses from this menu, subject to temptation. A rational human who suffers no temptation would never need to restrict their choices at the first stage. However, a human that faces temptation may want to restrict their options so that they do not succumb to temptation in the second stage.

We focus on learning the individual's preferences before temptation: the first-stage preferences. We show that, if one models the human as rational while ignoring their capacity for temptation, it is impossible to recover their first-stage preferences. Instead, we model the human as being *sophisticated* [O'Donoghue and Rabin, 1999], i.e., rational in the first stage *given* that they face temptation in the second stage. We give an efficient algorithm to learn first-stage menu preferences over a set of n possible items using $O(n^2 \log n)$ menu comparisons from a sophisticated human. Furthermore, we show how to reduce the number of comparisons to $O(n \log n)$ for approximate recovery.

In Chapter 7, we consider the recommender system setting. Here, the user typically has many ways to interact with a given item, e.g., likes, clicks, shares, replies, etc. While the standard paradigm for algorithmic recommendation, *engagement optimization*, treats these behavioral signals as the objective for recommendation, we suggest that these signals are only *evidence* of the true, underlying objective function: to provide content that is of "value" to the user. However, such signals are often very noisy indicators of what a user values and there may be many unknown ways that people's behaviors. If we cannot model each behavior as a reward-rational choice, what is an alternative that still allows us to learn from all signals at once?

Building on the framework of measurement theory, we treat "value" as a theoretical con-

struct that must be operationalized through a measurement model [Causey, 1971, Hand, 2004, Jackman, 2009, Jacobs and Wallach, 2019]. We create a general latent-variable modeling approach for recommender systems that can be used to operationalize the target construct "value" and optimize for it. Our approach relies on using one sparse signal that provides strong evidence that the user either values or doesn't value an item (e.g. "don't show me this") to learn the relationship between all the other user behaviors and latent "value". Essentially, we make a rationality assumption on just one strong signal which allows us to learn how to interpret all other signals. We implement our approach on the Twitter platform on millions of users. In line with established approaches to assessing the validity of measurements, we perform a qualitative evaluation of how well our model captures a desired notion of "value".

Finally, in Chapter 8, we end by discussing avenues for future work.

Part I

The reward-rational choice formalism

Chapter 2

The reward-rational choice formalism

In recent years, researchers have proposed ways to learn reward functions from a plethora of sources, e.g., demonstrations [Ng and Russell, 2000, Abbeel and Ng, 2004, Ziebart et al., 2008], comparisons [Wirth et al., 2016, Sadigh et al., 2017, Christiano et al., 2017], physical corrections [Losey and O'Malley, 2017, Bajcsy et al., 2017], natural language instructions [Matuszek et al., 2012, Tellex et al., 2011, Fried et al., 2018b,a], proxy rewards [Hadfield-Menell et al., 2017b, Ratner et al., 2018], switching a robot off [Hadfield-Menell et al., 2017a], and even the initial state of the world [Shah et al., 2019].

Overall, there is much information out there. Some types of information are purposefully communicated, while others are implicitly "leaked" by the human. While existing papers are instructing us how to tap into some of it, one can only imagine that there is much more that is yet untapped. There are probably new yet-to-be-invented ways for people to purposefully provide feedback to robots – e.g., guiding them on which part of a trajectory was particularly good or bad. And there will probably be new realizations about ways in which human behavior already naturally leaks information, beyond the state of the world or turning the robot off. How will robots make sense of all these diverse sources of information?

There is a way to interpret all this information in a single unifying formalism[1]. The critical observation is that human behavior can be modeled as a *reward-rational implicit choice* (RRC) – a choice from an implicit set of options, which is approximately rational for the intended reward. This observation leads to a *recipe* for making sense of human behavior, from language to switching the robot off. The recipe has two ingredients: 1) the set of *options* the person (implicitly) chose from, and 2) a *grounding* function that maps these options to robot behaviors[2]. This is admittedly obvious for traditional feedback. In comparison feedback, for instance, the set of options is just the two robot behaviors presented

[1] In other frameworks, e.g., an assistive game [Hadfield-Menell et al., 2016, Shah et al., 2020] or Bayesian inverse reinforcement learning [Ramachandran and Amir, 2007], one can also theoretically accommodate various kinds of human behavior if given a model of how that behavior relates to the latent reward function. Critically, our formalism, reward rational choice, give us a recipe for designing such a model.

[2] The contents of this chapter were originally published with Hong Jun Jeon and Anca Dragan as "Reward-rational (implicit) choice: A unifying formalism for reward learning" in NeurIPS 2020.

to the human to compare, and the grounding is identity. In other types of behavior though, it is much less obvious. Take switching the robot off. The set of options is implicit: you can turn it off, or you can do nothing. The formalism says that when you turn it off, it should know that you could have done nothing, but (implicitly) chose not to. That, in turn, should propagate to the robot's reward function. For this to happen, the robot needs to ground these options to robot behaviors: identity is no longer enough, because it cannot directly evaluate the reward of an utterance or of getting turned off, but it can evaluate the reward of robot actions or trajectories. Turning the robot off corresponds to a trajectory – whatever the robot did until the off-button was pushed, followed by doing nothing for the rest of the time horizon. Doing nothing corresponds to the trajectory the robot was going to execute. Now, the robot knows you prefer the former to the latter. We have taken a high-level human behavior, and turned it into a direct comparison on robot trajectories with respect to the intended reward, thereby gaining reward information.

We use this perspective to survey prior work on reward learning. We show that despite their diversity, many sources of information about rewards proposed thus far can be characterized as instantiating this formalism (some very directly, others with some modifications). This offers a unifying lens for the area of reward learning, helping better understand and contrast prior methods. We end with discussion on how the formalism can help combine and actively decide among feedback types, and also how it can be a potentially helpful recipe for interpreting new types of feedback or sources of leaked information.

2.1 A formalism for reward learning

Reward-rational implicit choice

In reward learning, the robot's goal is to learn a reward function $r : \Xi \rightarrow \mathbb{R}$ from human behavior that maps trajectories[3] $\xi \in \Xi$ to scalar rewards.

(Implicit/explicit) set of options $\mathcal{C}$. We interpret human behavior as choosing an option c^* from a set of options $\mathcal{C}$. Different behavior types will correspond to different explicit or implicit sets $\mathcal{C}$. For example, when a person is asked for a *trajectory comparison*, they are explicitly shown two trajectories and they pick one. However, when the person gives a *demonstration*, we think of the possible options $\mathcal{C}$ as implicitly being all possible trajectories the person could have demonstrated. The implicit/explicit distinction brings out a general tradeoff in reward learning. The cleverness of implicit choice sets is that even when we cannot enumerate and show all options to the human, e.g. in demonstrations, we still rely on the human to optimize over the set. On the other hand, an implicit set is also risky – since it is not explicitly observed, we may get it wrong, potentially resulting in worse reward inference.

The grounding function ψ. We link the human's choice to the reward by thinking of the choice as (approximately) maximizing the reward. However, it is not immediately

[3]We consider finite fixed horizon T trajectories.

clear what it means for the human to maximize reward when choosing feedback because the feedback may not be a (robot) trajectory, and the reward is only defined over trajectories. For example, in *language feedback*, the human describes what they want in words. What is the reward of the sentence, "Do not go over the water"?

To overcome this syntax mismatch, we map options in $\mathcal{C}$ to (distributions over) trajectories with a grounding function $\psi : \mathcal{C} \to f_\Xi$ where f_Ξ is the set of distributions over trajectories for the robot Ξ. Different types of feedback will correspond to different groundings. In some instances, such as kinesthetic demonstrations or trajectory comparisons, the mapping is simply the identity. In others, like corrections, language, or proxy rewards, the grounding is more complex (see Section 2.3).

Human policy. Given the set of choices $\mathcal{C}$ and the grounding function ψ, the human's approximately rational choice $c^* \in \mathcal{C}$ can now be modeled via a *Boltzmann-rational* policy, a policy in which the probability of choosing an option is exponentially higher based on its reward:

$$\mathbb{P}(c^* \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot \mathbb{E}_{\xi \sim \psi(c^*)}[r(\xi)])}{\sum_{c \in \mathcal{C}} \exp(\beta \cdot \mathbb{E}_{\xi \sim \psi(c)}[r(\xi)])}, \tag{2.1}$$

where the parameter β is a coefficient that models how rational the human is. Often, we simplify Equation 2.1 to the case where ψ is a deterministic mapping from choices in $\mathcal{C}$ to trajectories in Ξ, instead of distributions over trajectories. Then, the probability of choosing c^* can be written as:[4]

$$\mathbb{P}(c^* \mid r, \mathcal{C}) \propto \exp(\beta \cdot r(\psi(c^*))) \tag{2.2}$$

Boltzmann-rational policies are widespread in psychology [Baker et al., 2009, Goodman et al., 2009, Goodman and Stuhlmüller, 2013], economics [Bradley and Terry, 1952, Luce, 1959, Plackett, 1975, Luce, 1959], and AI [Ziebart et al., 2008, Ramachandran and Amir, 2007, Finn et al., 2016, Bloem and Bambos, 2014, Dragan et al., 2013a] as models of human choices, actions, or inferences. In the next section, we provide a maximum-entropy motivation for the Boltzmann distribution.

Definition 2.1.1 (Reward-rational choice)**.** Finally, putting it all together, we call a type of feedback a *reward-rational choice* if, given a grounding function ψ, it can be modeled as a choice from an (explicit or implicit set) $\mathcal{C}$ that (approximately) maximizes reward, i.e., as in Equation 2.1.

[4]One can also consider a variant of Equation 2.1 in which we use a Boltzmann distribution over actions instead of trajectories. The human's choices are grounded to actions and are evaluated via a Q-value function, rather than the reward function. That is, $\psi : \mathcal{C} \to \mathcal{A}$, $\mathbb{P}(c^* \mid r, \mathcal{C}) \propto \exp(\beta \cdot \mathbb{E}_{a \sim \psi(c^*)}[Q^*(s, a)])$. Another possible extension is to model the human's choices as grounding to policies rather than trajectories. Then, one can use a Boltzmann distribution over policies as was recently proposed by Laidlaw and Dragan [2022].

Robot inference

Each feedback is an observation about the reward, which means the robot can run Bayesian inference to update its belief over the rewards. For a deterministic grounding,

$$\mathbb{P}(r \mid c^*) = \frac{1}{Z} \cdot \frac{\exp(\beta \cdot r(\psi(c^*)))}{\sum_{c \in \mathcal{C}} \exp(\beta \cdot r(\psi(c)))} \cdot \mathbb{P}(r)\,, \tag{2.3}$$

where $\mathbb{P}(r)$ is the prior over rewards and Z is the normalization over possible reward functions. The inference above is often intractable, and so reward learning work leverages approximations [Blei et al., 2017], or computes only the MLE for a parametrization of rewards (more recently as weights in a neural network on raw input [Christiano et al., 2017, Ibarz et al., 2018]).

Finally, when the human is highly rational ($\beta \to \infty$), the only choices in $\mathcal{C}$ with a non-neglible probability of being picked are the choices that exactly maximize reward. Thus, the human's choice c^* can be interpreted as *constraints* on the reward function (e.g. [Ratliff et al., 2006]):

$$\text{Find } r \text{ such that } r(\psi(c^*)) \geq r(\psi(c)) \quad \forall c \in \mathcal{C}\,. \tag{2.4}$$

2.2 Bounded rationality, maximum entropy, and Boltzmann-rational policies

A perspective on reward learning that makes use at its core the Boltzmann model from Equation 2.1 would not be complete without a formal justification for it within our context. In this section, we derive it as the maximum-entropy distribution for the choices made by a bounded, *satisficing* human. Our explanation is complementary to that of Ortega and Braun [2013] who derive an axiomatic, thermodynamic framework to modeling bounded-rational decision making. Their framework leads to much the same interpretation of the Boltzmann-rational distribution, but is significantly more complex than needed for our purposes.

A perfectly rational human choosing from the set $\mathcal{C}$ would always pick the choice with optimal reward, $\max_{c \in \mathcal{C}} r(\psi(c))$. However, since humans are bounded, we do not expect them to perform optimally. Herbert Simon proposed the influential idea that humans are *bounded rational* and merely *satisfice* [Simon, 1956], rather than maximize, i.e., they pick an option above some satisfactory threshold, rather than picking the best possible option.

We can construct a simplified model of a satisficing human by modeling their expected reward as equal to a satisficing threshold, $\max_{c \in \mathcal{C}} r(\psi(c)) - \epsilon$ where $\epsilon \in (0, \epsilon_{\max})$ is the amount of expected error. The maximum possible error, $\epsilon_{max} = \min_{c \in \mathcal{C}} r(\psi(c)) - \max_{c \in \mathcal{C}} r(\psi(c))$, corresponds to *anti-rationality*, i.e., always picking the worst option.

Given the constraint that the human's expected reward is satisfactory, how should we pick a distribution to model the human's choices? The principle of maximum entropy [Jaynes, 1957] gives us a guide. If we want to encode no extra information in the distribution, then

we ought to pick the distribution that maximizes entropy subject to the constraint on the satisficing threshold.

Definition 2.2.1 (Satisficing MaxEnt problem)**.** Let be be a distribution P over choice set $\mathcal{C}$ and let p be a density for P with respect to a base measure F. The Shannon entropy of P is defined as $H(P) = -\int_{\mathcal{C}} p(f) \log p(f) dF(f)$. The *satisficing maximum entropy problem* is to find a distribution P that maximizes entropy subject to the satisficing constraint (2.5):

$$\begin{gathered} \max_P H(P) \\ \text{subject to} \\ \mathbb{E}_{c\sim P}[r(\psi(c))] = \max_{c\in\mathcal{C}} r(\psi(c)) - \epsilon\,. \end{gathered} \tag{2.5}$$

It is well-known that the maximum-entropy distribution subject to linear constraints (such as a constraint on the mean as in (2.5)) is the unique exponential distribution that satisfies the constraints. Thus, for our special case, the maximum-entropy distrbution is the Boltzmann distribution with rationality coefficient β satisfying the satisficing constraint.

Theorem 2.2.1 ([Jaynes, 1957])**.** *The solution to the satisficing maximum entropy problem is the Boltzmann distribution $\mathbb{P}_\beta(f) \propto \exp(\beta \cdot r(\psi(c)))$ where β is the unique value satisfying the satisficing constraint (2.5).*

Since the expected reward $\mathbb{E}_\beta[r(\psi(c)]$ is monotonically increasing in the rationality parameter β, the satisficing error ϵ and rationality coefficient β have a one-to-one relationship, as summarized in the following corollary.

Corollary 2.2.1. *The solution to the satisficing maximum entropy problem is a Boltzmann-rational policy where the rationality coefficient β is monotonically decreasing in the satisficing error ϵ. In particular, we have the following:*

Human type	***Error*** ϵ	***Rationality*** β
Perfectly rational	$\epsilon \to 0$	$\beta \to +\infty$
Random	$\epsilon = \max_{c\in\mathcal{C}} r(\psi(c)) - \mathbb{E}_{c\sim Unif(\mathcal{C})}[r(\psi(c))]$	$\beta = 0$
Anti-rational	$\epsilon \to \epsilon_{max}$	$\beta \to -\infty$

Thus, we see that a Boltzmann-rational policy is the maximum entropy distribution for an ϵ-suboptimal human. By following the principle of maximum entropy, Boltzmann-rationality provides a way to model a suboptimal human without implicitly adding in any extra assumptions about the human's choice.

Table 2.1: The choice set $\mathcal{C}$ and grounding function ψ for different types of feedback described in Section 2.3, unless otherwise noted.

Feedback	**Choices $\mathcal{C}$**	**Grounding ψ**
Comparisons [Wirth et al., 2017]	$\xi_i \in \{\xi_1, \xi_2\}$	$\psi(\xi_i) = \xi_i$
Demonstrations [Ng and Russell, 2000]	$\xi_d \in \Xi$	$\psi(\xi) = \xi$
Corrections [Bajcsy et al., 2017]	$\Delta q \in Q - Q$	$\psi(\Delta q) = \xi_R + A^{-1}\Delta q$
Improvement [Jain et al., 2015]	$\xi \in \{\xi_{\text{improved}}, \xi_R\}$	$\psi(\xi) = \xi$
Off [Hadfield-Menell et al., 2017a]	$c \in \{\text{off}, -\}$	$\psi(c) = \begin{cases} \xi_R & c = - \\ \xi_R^{0:t}\xi_R^t \dots \xi_R^t & c = \text{off} \end{cases}$
Language [Matuszek et al., 2012]	$\lambda \in \Lambda$	$\psi(\lambda) = \text{Unif}(G(\lambda))$
Proxy Rewards [Hadfield-Menell et al., 2017b]	$\tilde{r} \in \tilde{\mathcal{R}}$	$\psi(\tilde{r}) = \pi(\xi \mid \tilde{r})$
Reward and Punishment [Griffith et al., 2013]	$c \in \{+1, -1\}$	$\psi(c) = \begin{cases} \xi_R & c = +1 \\ \xi_{\text{expected}} & c = -1 \end{cases}$
Initial state [Shah et al., 2019]	$s \in \mathcal{S}$	$\psi(s) = \text{Unif}(\{\xi_H^{-T:0} \mid \xi_H^0 = s\})$
Credit assignment (Discussion)	$\xi \in \{\xi_R^{i:i+k}, 0 \le i \le T\}$	$\psi(\xi) = \xi$

2.3 Prior work from the perspective of the formalism

We now instantiate the formalism above with different behavior types from prior work, constructing their choice sets $\mathcal{C}$ and groundings ψ. Some are obvious – comparisons, demonstrations especially. Others – initial state, off, reward/punish – are more subtle and it takes slightly modifying their original methods to achieve unification, speaking to the nontrivial nuances of identifying a common formalism.

Table 2.1 lists $\mathcal{C}$ and ψ for each feedback, while Table 2.2 shows the deterministic constraint on rewards each behavior imposes, along with the probabilistic observation model – highlighting, despite the differences in feedback, the pattern of the (exponentiated) choice reward in the numerator, and the normalization over $\mathcal{C}$ in the denominator. Fig. 2.1 will serve as the illustration for these types, looking at a grid world navigation task around a rug. The space of rewards we use for illustration is three-dimensional weight vectors for avoiding the rug, not getting dirty, and reaching the goal.

Trajectory comparisons. In trajectory comparisons [Wirth et al., 2016], the human is typically shown two trajectories $\xi_1 \in \Xi$ and $\xi_2 \in \Xi$, and then asked to select the one that they prefer. They are perhaps the most obvious exemplar of reward-rational choice: the set of choices $\mathcal{C} = \{\xi_1, \xi_2\}$ is explicit, and the grounding ψ is simply the identity. As Fig. 2.1 shows, for linear reward functions, a comparison corresponds to a hyperplane that cuts the space of feasible reward functions in half. For all the reward functions left, the chosen trajectory has higher reward than the alternative. Most work on comparisons is

Table 2.2: The probabilistic model (Equation 2.1) and the simplification to the constraint-based model (Equation 2.4).

Feedback	**Constraint**	**Probabilistic**
Comparisons	$r(\xi_1) \geq r(\xi_2)$	$\mathbb{P}(\xi_1 \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot r(\xi_1))}{\exp(\beta \cdot r(\xi_1)) + \exp(\beta \cdot r(\xi_2))}$
Demonstrations	$r(\xi_D) \geq r(\xi) \quad \forall\, \xi \in \Xi$	$\mathbb{P}(\xi_D \mid r, \Xi) = \frac{\exp(\beta \cdot r(\xi_D))}{\sum_{\xi \in \Xi} \exp(\beta \cdot r(\xi))}$
Corrections	$r(\xi_R + A^{-1}\Delta q) \geq r(\xi_R + A^{-1}\Delta q') \; \forall \Delta q' \in Q - Q$	$\mathbb{P}(\Delta q' \mid r, Q-Q) = \frac{\exp(\beta \cdot r(\xi_R + A^{-1}\Delta q))}{\sum_{\Delta q \in Q-Q} \exp(\beta \cdot r(\xi_R + A^{-1}\Delta q))}$
Improvement	$r(\xi_{\text{improved}}) \geq r(\xi_R)$	$\mathbb{P}(\xi_{\text{improved}} \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot r(\xi_{\text{improved}}))}{\exp(\beta \cdot r(\xi_{\text{improved}})) + \exp(\beta \cdot r(\xi_R))}$
Off	$r(\xi_R^{0:t}\xi^t \dots \xi^t) \geq r(\xi_R)$	$\mathbb{P}(\text{off} \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot r(\xi_R^{0:t}\xi^t \dots \xi^t))}{\exp(\beta \cdot r(\xi_R^{0:t}\xi^t \dots \xi^t)) + \exp(\beta \cdot r(\xi_R))}$
Language	$\mathbb{E}_{\xi \sim \text{Unif}(G(\lambda^*))}\big[r(\xi)\big] \geq \mathbb{E}_{\xi \sim \text{Unif}(G(\lambda))}\big[r(\xi)\big] \; \forall \lambda \in \Lambda$	$\mathbb{P}(\lambda^* \mid r, \Lambda) = \frac{\exp(\beta \cdot \mathbb{E}_{\xi \sim \text{Unif}(G(\lambda^*))}\big[r(\xi)\big])}{\sum_{\lambda \in \Lambda} \exp(\beta \cdot \mathbb{E}_{\xi \sim \text{Unif}(G(\lambda))}\big[r(\xi)\big])}$
Proxy Rewards	$\mathbb{E}_{\tilde{\xi} \sim \pi(\tilde{\xi} \mid \tilde{r})}\big[r(\tilde{\xi})\big] \geq \mathbb{E}_{\tilde{\xi} \sim \pi(\tilde{\xi} \mid c)}\big[r(\tilde{\xi})\big] \quad \forall c \in \tilde{\mathcal{R}}$	$\mathbb{P}(\tilde{r} \mid r, \tilde{\mathcal{R}}) = \frac{\exp(\beta \cdot \mathbb{E}_{\tilde{\xi} \sim \pi(\tilde{\xi} \mid \tilde{r})}\big[r(\tilde{\xi})\big])}{\sum_{c \in \tilde{\mathcal{R}}} \exp(\beta \cdot \mathbb{E}_{\tilde{\xi} \sim \pi(\tilde{\xi} \mid c)}\big[r(\tilde{\xi})\big])}$
Reward/Punish	$r(\xi_R) \geq r(\xi_{\text{expected}})$	$\mathbb{P}(+1 \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot r(\xi_R))}{\exp(\beta \cdot r(\xi_R)) + \exp(\beta \cdot r(\xi_{\text{expected}}))}$
Initial state	$\mathbb{E}_{\xi \sim \psi(s^*)}[r(s^*)] \geq \mathbb{E}_{\xi \sim \psi(s)}[r(s)] \quad \forall s \in \mathcal{S}$	$\mathbb{P}(s^* \mid r, \mathcal{S}) = \frac{\exp(\beta \cdot \mathbb{E}_{\xi \sim \psi(s^*)}[r(\xi)])}{\sum_{s \in S} \exp(\beta \cdot \mathbb{E}_{\xi \sim \psi(s)}[r(\xi)])}$
Meta-choice	$\mathbb{E}_{\xi \sim \psi(\mathcal{C}_i)}[r(\xi)] \geq \mathbb{E}_{\xi \sim \psi(\mathcal{C}_j)}[r(\xi)] \quad \forall j \in [n]$	$\mathbb{P}(\mathcal{C}_i \mid r, \mathcal{C}_0) = \frac{\exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_i)}[r(\xi)]\right)}{\sum_{j \in [n]} \exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_j)}[r(\xi)]\right)}$
Credit assignment	$r(\xi^*) \geq r(\xi) \quad \forall\, \xi \in \mathcal{C}$	$\mathbb{P}(\xi^* \mid r, \mathcal{C}) = \frac{\exp(\beta \cdot r(\xi^*))}{\sum_{\xi \in \mathcal{C}} \exp(\beta \cdot r(\xi))}$

done in the preference-based RL domain in which the robot might compute a policy directly to agree with the comparisons, rather than explicitly recover the reward function [Wilson et al., 2012, Busa-Fekete et al., 2013]. Within methods that do recover rewards, most use the constraint version (left column of Table 2.2) using various losses [Akrour et al., 2011, Wirth and Fürnkranz, 2014]. Holladay et al. [2016] use the Boltzmann model (right column of Table 2.2) and proposes actively generating the queries, Sadigh et al. [2017] follow up with actively synthesizing the queries from scratch, and Christiano et al. [2017] introduce deep neural network reward functions.

Demonstrations. In demonstrations, the human is asked to demonstrate the optimal behavior. Reward learning from demonstrations is often called *inverse reinforcement learning* (IRL) and is one of the most established types of feedback for reward learning [Ng and Russell, 2000, Abbeel and Ng, 2004, Ziebart et al., 2008]. Unlike in comparisons, in demonstrations, the human is not explicitly given a set of choices. However, we assume that the human is *implicitly* optimizing over all possible trajectories (Fig. 2.1 (1st row, 2nd column) shows these choices in gray). Thus, demonstrations are a reward-rational choice in which the set of choices $\mathcal{C}$ is (implicitly) the set of trajectories Ξ. Again, the grounding ψ is the identity. In Fig. 2.1, fewer rewards are consistent with a demonstration than with a comparison. Early work used the constraint formulation with various losses to penalize violations [Ng and Russell, 2000,

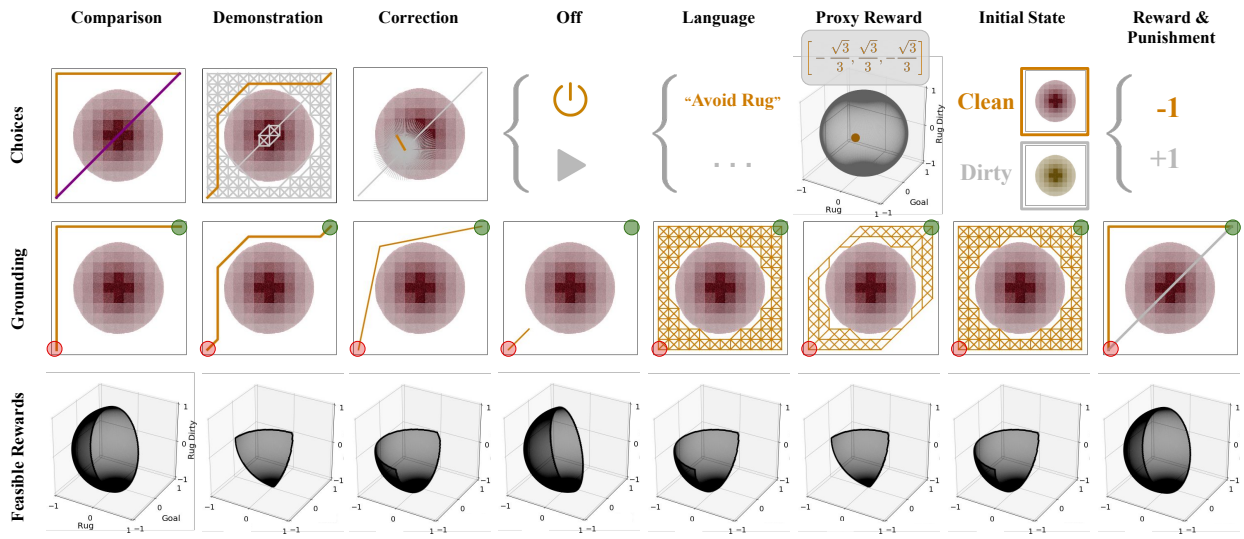

Figure 2.1: Different behavior types described in Sec. 2.3 in a gridworld with three features: avoiding/going on the rug, getting the rug dirty, and reaching the goal (green). For each, we display the choices, grounding, and feasible rewards under the constraint formulation of robot inference (2.4). Each trajectory is a finite horizon path that begins at the start (red). Orange is used to denote c^* and $\psi(c^*)$ while gray to denote other choices c in $\mathcal{C}$. For instance, the **comparison** affects the feasible reward space by removing the halfspace where going on the rug is good. It does not inform the robot about the goal, because both end at the goal. The **demonstration** removes the space where the rug is good, where the goal is bad (because alternates do not reach the goal), and where getting the rug dirty is good (because alternates slightly graze the rug). The **correction** is similar to the demonstration, but does not infer about the goal, since all corrections end at goal.

Ratliff et al., 2006]. Bayesian IRL [Ramachandran and Amir, 2007] exactly instantiates the formalism using the Boltzmann distribution by doing a full belief update as in Equation 2.3. Later work computes the MLE instead [Ziebart et al., 2008, Bloem and Bambos, 2014, Ho and Ermon, 2016] and approximates the partition function (the denominator) by a quadratic approximation about the demonstration [Levine and Koltun, 2012], a Laplace approximation [Dragan and Srinivasa, 2012], or importance sampling [Finn et al., 2016].

Corrections are the first type of feedback we consider that has both an implicit set of choices $\mathcal{C}$ and a non-trivial (not equal to identity) grounding. Corrections are most common in physical human–robot interaction (pHRI), in which a human physically corrects the motion of a robot. The robot executes a trajectory ξ_R, and the human intervenes by applying a correction $\Delta q \in Q$ that modifies the robot's current configuration. Therefore, the set of choices $\mathcal{C} = Q - Q = \{q_1 - q_2 \mid q_1, q_2 \in Q\}$ consists of all possible configuration differences Δq the person could have used (Fig. 2.1 1st row, 3rd column shows possible Δqs in gray and the selected one in orange). The way we can ground these choices is by finding a trajectory

that is closest to the original, but satisfies the constraint of matching a new point:

$$\min_{\xi} ||\xi - \xi_R||_A^2$$
$$s.t.\ \xi(0) = \xi_R(0),\ \xi(T) = \xi_R(T), \xi(t) = \xi_R(t) + \Delta q \tag{2.6}$$

where t is the time at which the correction was applied. Choosing a non-Euclidean inner-product, A (for instance $K^T K$, with K the finite differencing matrix), couples states along the trajectory in time and leads to a the resulting trajectory smoothly deforming – propagating the change Δq to the rest of the trajectory: $\psi(\Delta q) = \xi_R + A^{-1}[\lambda, 0, .., \Delta q, .., 0, \gamma]^T$ (with λ and γ making sure the end-points stay in place). This is the orange trajectory in the figure. Most work in corrections affects the robot's trajectory but not the reward function [Haddadin et al., 2008, Hogan, 1985], with [Losey and O'Malley, 2017] proposing the propagation via A^{-1} above. Bajcsy et al. [2017] propose that corrections are informative about the reward and use the propagation as their grounding, deriving an approximate MAP estimate for the reward. Losey and O'Malley [2018] introduce a way to maintain uncertainty.

Improvement. Prior work [Jain et al., 2015] has also modeled a variant of corrections in which the human provides an improved trajectory $\xi_{improved}$ which is treated as better than the robot's original ξ_R. Although [Jain et al., 2015] use the Euclidean inner product and implement reward learning as an online gradient method that treats the improved trajectory as a demonstration (but only takes a single gradient step towards the MLE), we can also naturally interpret improvement as a comparison that tells us the improved trajectory is better than the original: the set of options $\mathcal{C}$ consists of only ξ_R and $\xi_{improved}$ now, as opposed to all the trajectories obtainable by propagating local corrections; the grounding is identity, resulting in essentially a comparison between the robot's trajectory and the user provided one.

Off. In "off" feedback, the robot executes a trajectory, and at any point, the human may switch the robot off. "Off" appears to be a very sparse signal, and it is not spelled out in prior work how one might learn a reward from it. Reward-rational choice suggests that we first uncover the implicit set of options $\mathcal{C}$ the human was choosing from. In this case, the set of options consists of turning the robot off or not doing anything at all: $\mathcal{C} = \{\text{off}, -\}$. Next, we must ask how to evaluate the reward of the two options, i.e., what is the grounding? Hadfield-Menell et al. [2017a] introduced switching the robot off as a source of information to learn from and it as a choice in a one-shot game. There, not intervening means the robot takes its one possible action, and intervening means the robot takes the no-op action. This can be easily generalized to the sequential setting: not intervening means that the robot continues on its current trajectory, and intervening means that it stays at its current position for the remainder of the time horizon. Thus, the choices $\mathcal{C} = \{\text{off}, -\}$ map to the trajectories $\{\xi_R^{0:t} \xi_R^t \ldots \xi_R^t,\ \xi_R\}$.

Language. Humans might use rich language to instruct the robot, like "Avoid the rug." Let $G(\lambda)$ be the trajectories that are consistent with an utterance $\lambda \in \Lambda$ (e.g. all trajectories that do not enter the rug). Usually the human instruction is interpreted *literally*, i.e. any trajectory consistent with the instruction $\xi \in G(\lambda)$ is taken to be equally likely , although,

other distributions are also possible. For example, a problem with literal interpretation is that it does not take into account the other choices the human may have considered. The instruction "Do not go into the water" is consistent with the robot not moving at all, but we imagine that if the human wanted the robot to do nothing, they would have said that instead. Therefore, it would be incorrect for the robot to do nothing when given the instruction "Do not go into the water". This type of reasoning is called *pragmatic reasoning* [Grice, 1975], and indeed recent work shows that explicitly interpreting instructions pragmatically can lead to higher performance [Fried et al., 2018a,b]. The reward-rational choice formulation of language feedback naturally leads to pragmatic reasoning on the part of the robot, and is in fact equivalent to the rational speech acts model [Goodman and Stuhlmüller, 2013, Bergen, 2016], a model of pragmatic reasoning in language. The pragmatic reasoning arises because the human is explicitly modeled as choosing from a set of options.

The reward-rational choice formulation of language feedback naturally leads to pragmatic reasoning on the part of the robot, and is in fact equivalent to the rational speech acts model [Goodman and Stuhlmüller, 2013], a standard model of pragmatic reasoning in language. The pragmatic reasoning arises because the human is explicitly modeled as choosing from a set of options. Language is a reward-rational choice in which the set of options $\mathcal{C}$ is the set of instructions considered in domain Λ and the grounding ψ maps an utterance λ to the uniform distribution over consistent trajectories $\text{Unif}(G(\lambda))$. In language feedback, a key difficulty is learning which robot trajectories are consistent with a natural language instruction, the *language grounding problem* (and is where we borrow the term "grounding" from) [Matuszek et al., 2012, Tellex et al., 2011, Fu et al., 2019]. Fig. 2.1 shows the grounding for avoiding the rug in orange – all trajectories from start to goal that do not enter rug cells.

Proxy rewards are expert-specified rewards that do not necessarily lead to the desired behavior in all situations, but can be trusted on the training environments. They were introduced by Hadfield-Menell et al. [2017b], who argued that even when the expert attempts to fully specify the reward, it will still fail to generalize to some situations outside of the training environments. Therefore, rather than taking a specified reward at face value, we can interpret it as evidence about the true reward. Proxy reward feedback is a reward-rational choice in which the set of choices $\mathcal{C}$ is the set of proxy rewards the designer may have chosen, $\tilde{\mathcal{R}}$. The reward designer is assumed to be approximately optimal, i.e. they are more likely to pick a proxy reward $\tilde{r} \in \tilde{\mathcal{R}}$ if it leads to better trajectories *on the training environment(s)*. Thus, the grounding ψ maps a proxy reward r to the distribution over trajectories that the robot takes in the training environment given the proxy reward [Hadfield-Menell et al., 2017b, Mindermann et al., 2018, Ratner et al., 2018]. Fig. 2.1 shows the grounding for a proxy reward for reaching the goal, avoiding the rug, and not getting the rug dirty – many feasible rewards would produce similar behavior as the proxy. By taking the proxy as evidence about the underlying reward, the robot ends up with uncertainty over what the actual reward might be, and can better hedge its bets at test time.

Reward and punishment [Griffith et al., 2013, Loftin et al., 2014]. In this type of feedback, the human can either reward $(+1)$ or punish (-1) the robot for its trajectory ξ_R; the set of options is $\mathcal{C} = \{+1, -1\}$. A naive implementation would interpret reward

and punishment literally, i.e. as a scalar reward signal for a reinforcement learning agent, however empirical studies show that humans reward and punish based on how well the robot performs *relative to their expectations* [MacGlashan et al., 2017]. Thus, we can use our formalism to interpret that: reward (+1) grounds to the robot's trajectory ξ_R, while punish (−1) grounds to the trajectory the human expected ξ_{expected} (not necessarily observed).

Initial state. Shah et al. [2019] make the observation that when the robot is deployed in an environment that humans have acted in, the current state of the environment is already optimized for what humans want, and thus contains information about the reward. For example, suppose the environment has a goal state which the robot can reach through either a lawn or a carpet. If the lawn is pristine and untrodden, then humans must have intentionally avoided walking on it in the past (even though the robot hasn't observed this past behavior), and the robot can reasonably infer that it too should not go on the lawn.

The original paper inferred rewards from a single state s by marginalizing over possible pasts, i.e. trajectories $\xi_H^{-T:0}$ that end at s which the human could have taken, $P(s|r) = \sum_{\xi_H^{-T:0}|\xi_H(0)=s} P(\xi_H^{-T:0}|r)$. However, through the lens of our formalism, we see that initial states can also be interpreted more directly as reward-rational implicit choices. The set of choices $\mathcal{C}$ can be the set of possible initial states $\mathcal{S}$. The grounding function ψ maps a state $s \in \mathcal{S}$ to the uniform distribution over any human trajectory $\xi_H^{-T:0}$ that starts from a specified time before the robot was deployed ($t = -T$) and ends at state s at the time the robot was deployed ($t = 0$), i.e. $\xi_H^0 = s$. This leads to the $P(s|r)$ from Table 2.2, which is almost the same as the original, but sums over trajectories directly in the exponent, and normalizes over possible other states. The two interpretations would only become equivalent if we replaced the Boltzmann distribution with a linear one. Fig 2.1 shows the result of this (modified) inference, recovering as much information as with the correction or language.

Formalizing new feedback

The types of feedback or behavior we have discussed so far are by no means the only types possible. New ones will inevitably be invented. But when designing a new type of feedback, it is often difficult to understand what the relationship is between the reward r and the feedback c^*. Reward-rational choice suggests a recipe for uncovering this link – define what the implicit set of options the human is choosing from is, and how those options ground to trajectories. Then, Equation 2.1 provides a formal model for the human feedback.

For example, hypothetically, someone might propose a "credit assignment" type of feedback. Given a trajectory ξ_R of length T, the human is asked to pick a segment of length $k < T$ that has maximal reward. We doubt the set of choices in an implementation of credit assignment would be explicit, however the implicit set of choices $\mathcal{C}$ is then the set of all segments of length k. The grounding function ψ is simply the identity. With this choice of $\mathcal{C}$ and ψ in hand, the human can now be modeled according to Equation 2.1, as we show in the last rows of Tables 2.1 and 2.2.

Part II

Implications of reward-rational choice

Chapter 3

Combining and actively selecting feedback types

From demonstrations to reward/punishment to the initial state of the world, the robot can extract information from humans by modeling them as making approximate reward-rational choices. Often, the choices are implicit, like in turning the robot off or providing language instructions. Sometimes, the choices are not made in order to purposefully communicate about the reward, and rather end up leaking information about it, like in the initial state, or even in corrections or turning the robot off. Regardless, the RRC framework enables us to better understand how all these sources of information relate and compare.

So far we have talked about learning from individual types of behaviors. But we do not want our robots stuck with a single type: we want them to learn from all types of information. For example, the robot might receive demonstrations from a human during training, and then corrections during deployment, which are followed by the human prematurely switching the robot off. The observational model in (2.2) for a single type of behavior also provides a natural way to model combinations of behavior. If each observation is conditionally independent given the reward, then according to (2.2), the probability of observing a vector $\mathbf{c}$ of n behavioral signals (of possibly different types) is equal to

$$\mathbb{P}(\mathbf{c} \mid r) = \prod_{i=1}^{n} \frac{\exp(\beta_i \cdot r(\psi_i(\mathbf{c}_i)))}{\sum_{c \in \mathcal{C}_i} \exp(\beta_i \cdot r(\psi_i(c)))}\,. \tag{3.1}$$

Given this likelihood function for the human's behavior, the robot can infer the reward function using the approaches and approximations described in Sec. 2.1. Recent work has already built in this direction, combining trajectory comparisons and demonstrations [Ibarz et al., 2018, Palan et al., 2019]. We note that the formulation in Equation 3.1 is general and applies to *any* combination. In Section 3.1, we describe a case study on a novel combination of feedback types: proxy rewards, a physical improvement, and comparisons in which we use a constraint-based approximation (see Equation 2.4) to Equation 3.1.

Further, it also becomes natural to *actively decide* which feedback type to ask a human for. Rather than relying on a heuristic (or on the human to decide), the robot can maximize

expected information gain. Suppose we can select between n types of feedback with choice sets $\mathcal{C}_1, \ldots, \mathcal{C}_n$ to ask the user for. Let b_t be the robot's belief distribution over rewards at time t. The type of feedback i^* that (greedily) maximizes information gain for the next time step is

$$i^* = \arg\max_{i \in [n]} \mathbb{E}_{r_t, c_i^*} \left[\log \left(\frac{p(c_i^* \mid r_t)}{\int_{r_t \in \mathcal{R}} p(c_i^* \mid r_t) b_t(r_t)} \right) \right], \tag{3.2}$$

where $r_t \sim B_t$ is distributed according to the robot's current belief, $c_i^* \in \mathcal{C}_i$ is the random variable corresponding to the user's choice within feedback type i, and $p(c_i^* \mid r_t)$ is defined according to the human model in Equation 2.1. We also note that different feedback types may have different costs associated with them (e.g. of human time) and it is straightforward to integrate these costs into (3.2). In Section 3.2, we describe experiments with active selection of feedback types. In the environments we tested, we found that demonstrations are optimal early on, when little is known about the reward, while comparisons became optimal later, as a way to fine-tune the reward. The finding provides validation for the approach pursued by Palan et al. [2019] and Ibarz et al. [2018]. Both papers manually define the mixing procedure we found to be optimal: initially train the reward model using human demonstrations, and then fine-tune with comparisons.

3.1 A case study on combining feedback types

We now describe a case study in which we illustrate how the framework can be used to learn from multiple sources of information at once. Fig. 3.1 illustrates a case study for teaching a robot arm a reward for motion planning through a novel combination of feedback types. In each environment, the robot arm must plan a trajectory from a start configuration to a designated goal configuration. We want this trajectory to properly trade off efficiency against staying at an appropriate distances to the human and to the table. Hand-tuning a reward function that returns desirable trajectories in *all* possible environments is actually very challenging. You could imagine that as you increase the efficiency weight to produce a smoother trajectory in one environment, you break the behavior in another environment where the robot now gets too close to the human, etc. In fact, the first type of feedback in the case study illustrates this: we design a (proxy) reward function that works well in two (training) environments (top left), but there are many rewards that are consistent with that behavior, yet produce vastly different behaviors in the two test environments (right).

Therefore, we start by defining a proxy reward, but then follow it up with more feedback: an improvement, and a comparison between two trajectories. This narrows down the space of rewards such that the robot can now generalize what to do outside of the training environments, as shown by two testing environments (right).

Cost Function and Features.

$$\text{Efficiency} := \sum_{i=1}^{|\tau|-1} \|\tau[i] - \tau[i-1]\|_2^2,$$

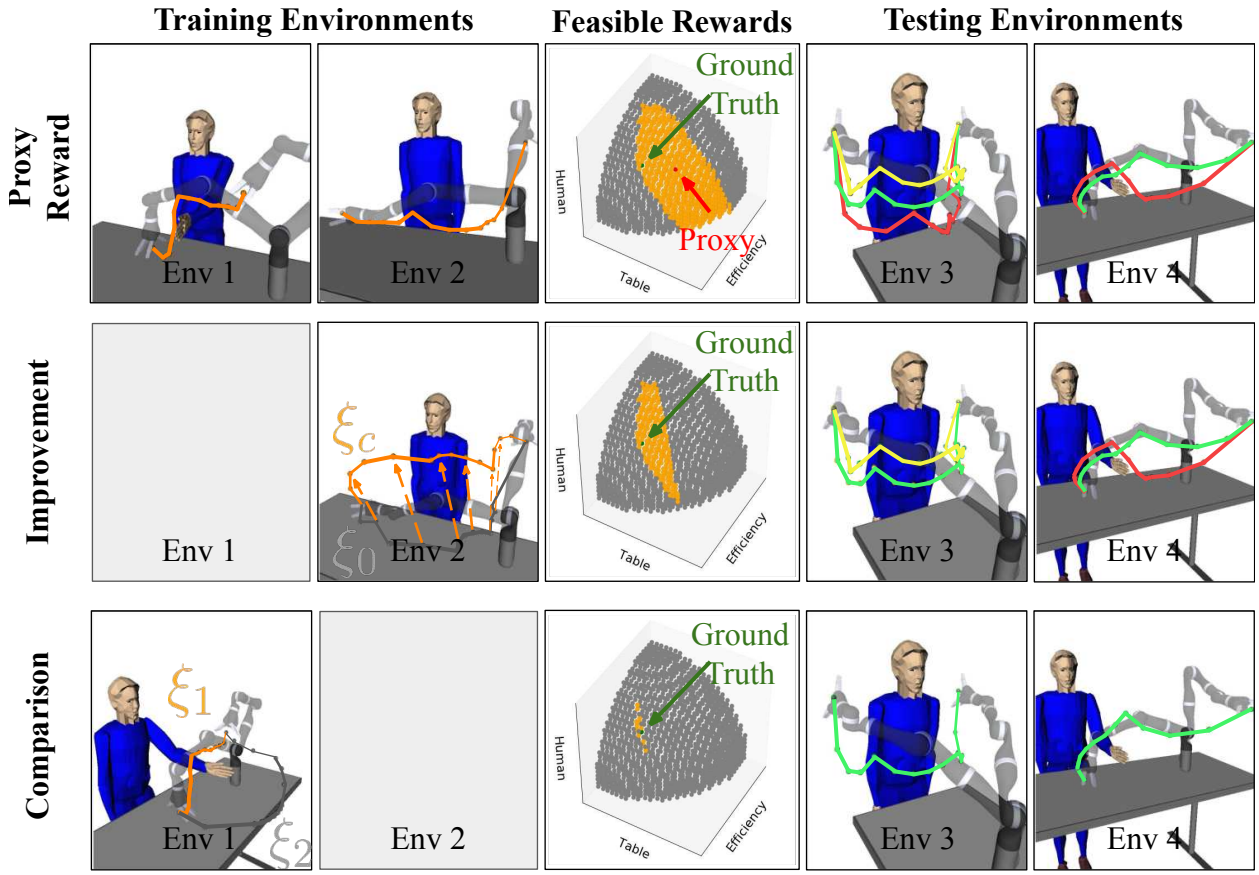

Figure 3.1: A case study for teaching a reward for robot arm motion using two training environments. The robot trades off efficiency, keeping distance away from the human, and also from the table. We use the constraints interpretation of feedback in this study. We start by defining a proxy reward that produces acceptable behavior (orange trajectories) in the training environments (1st row). This initial feedback significantly prunes the feasible space, but is not enough to guarantee good performance in other environments. On the right, we see trajectories still considered feasible in two test environments. The green one is correct, however, the other feasible trajectories are either too close to the human or too close to the table. After an improvement feedback and a comparison, the robot shrinks the space of feasible rewards, removing extraneous rewards that produce undesirable behavior at test time.

$$\text{Distance to Table} := \sum_{i=0}^{|\tau|-1} 1 - \exp(-\text{dist}_i)$$
$$\text{where dist}_i = |g_{st}(\tau[i])_z - \text{table}_z|\,,$$

$$\text{Distance to Human} := \sum_{i=0}^{|\tau|-1} 1 - \exp(-\text{dist}_i)$$
$$\text{where dist}_i = \|\text{proj}_x(g_{st}(\tau[i])) - \text{proj}_x(\text{human})\|_2^2\,.$$

Efficiency is the sum of squared configuration space distances between consecutive trajectory waypoints. The table and human features are expressed as 1 minus a radial basis function of a modified distance between the object and the robot's end effector positon (denoted by $g_{st}(\tau[i])$, where g_{st} is the forward kinematics that maps configuration $\xi[i] \in Q$ to its end effector location in $\mathbb{R}^3$). For the table, this modification is to only consider distance in the z-coordinate, effectively measuring the distance from the robot's end effector to the table *plane*. For the human, the modification is to treat the human as an axis x and consider distance in 2 dimensions after projecting onto the plane with normal x. In Figure 3.1, the main obstacle is either the human's body or his arm. When the body is the obstacle, $x = [0, 0, 1]$ and when the arm is the obstacle, $x = [0, 1, 0]$. This considers the human not as just a point, but rather a line along the body, or arm axis.

Optimization We approximate the space of reward parameters Θ by uniform discretization at the surface of the non-negative octant of the 3 dimensional sphere (1371 points). Robotic motion planners cannot, in general, compute the globally optimal trajectory for a given $\theta \in \Theta$ so we resort to computing a set $\hat{\mathcal{T}}$ of locally optimal trajectories for each θ via TrajOpt [Schulman et al., 2014]. The optimal trajectory for a given θ is then defined as

$$\xi(\theta) := \arg\min_{\xi \in \hat{\mathcal{T}}} \; \theta^T \phi(\xi)$$

Proxy Reward. For this case study, the robot begins by asking the human designer for a proxy reward (cost) function. It is difficult for humans to provide proxies that work across all environments [Ratner et al., 2018], so the robot asks for a proxy that produces the desired behavior in the two training environments. The human can provide the proxy weights: $[0.55, 0.55, 0.55]$ and produce trajectories that match those of ξ_{θ^*} (Figure 3.1 depicted in orange). Providing a proxy applies constraints that shrink our feasible set from Θ to $\mathcal{F}_{\text{proxy}}$:

$$\mathcal{F}_{\text{proxy}} = \{\tilde{\theta} \;\; : \theta^{*T}\phi(\xi_{\tilde{\theta}}^{(1)}) \geq \theta^{*T}\phi(\xi_{\theta}^{(1)})\,,\, \theta^{*T}\phi(\xi_{\tilde{\theta}}^{(2)}) \geq \theta^{*T}\phi(\xi_{\theta}^{(2)}) \quad \forall\, \theta \in \Theta\}\,,$$

where $\xi_\theta^{(i)}$ denotes the optimal trajectory[1] w.r.t. cost parameter θ in environment i. The new feasible set $\mathcal{F}_{\text{proxy}}$ contains only the parameters θ that produce optimal trajectories

[1] In our case study, the optimal trajectory is unique.

with respect to the true weights θ^* in environments 1 and 2. Although it is a subset of the original feasible set Θ, the new feasible set $\mathcal{F}_{\text{proxy}}$ is still a reasonably large set (Figure 3.1, top, middle, orange area). Furthermore, although the proxy produces optimal trajectories in environments 1 and 2, it does not necessarily for environments 3 and 4. Figure 3.1 (top, right) illustrates the different trajectories that result from optimizing different $\theta \in \mathcal{F}_{\text{proxy}}$. To further narrow our feasible set, we will ask for another form of feedback: Improvement.

Improvement. The robot will now (actively) provide a nominal trajectory, and ask the human to improve it, i.e. alter the trajectory to better suit their preferences. Suppose the robot presents the human with the nominal trajectory shown in gray (Figure 3.1, middle, left). This nominal trajectory is inefficient, staying too close to the table. Based on θ^*, the human could provide the improved orange trajectory (Figure 3.1, middle, left) that is more efficient and doesn't emphasize closeness to the table as much. This improvement reduces our feasible set from $\mathcal{F}_{\text{proxy}}$ to $\mathcal{F}_{\text{improvement}}$:

$$\mathcal{F}_{\text{improvement}} = \{\theta \ : \theta^T \phi(\xi_{\text{improved}}) \geq \theta^T \phi(\xi_R) \quad \theta \in \mathcal{F}_{\text{proxy}}\}.$$

Figure 3.1 (middle, middle) shows the effect of applying this constraint, shrinking the orange feasible set. The feasible set has shrunk, but not enough to guarantee optimal behavior in all environments. The improvement establishes that closeness to the table should not come at the cost of efficiency. As a result, it removes the red trajectory in environment 3, which greatly traded off efficiency for proximity to the table (Figure 3.1, middle, right). To further fine tune, we will ask the human to answer a trajectory comparison.

Trajectory Comparison. The robot presents the human with two trajectories (Figure 3.1 bottom, left, orange and gray) and asks which incurs less cost. The human answers "orange", the trajectory that prioritizes efficiency over distance to the table. This comparison feedback shrinks our feasible set from $\mathcal{F}_{\text{improvement}}$ to $\mathcal{F}_{\text{comparison}}$:

$$\mathcal{F}_{\text{comparison}} = \{\theta \in \mathcal{F}_{\text{improvement}} \ : \theta^T \phi(\xi_{\text{orange}}) \geq \theta^T \phi(\xi_{\text{gray}})\}.$$

We finally see a very small orange feasible set (Figure 3.1, bottom, middle). Appropriately, in all four environments now, every $\theta \in \mathcal{F}_{\text{comparison}}$ produces a trajectory ξ_θ s.t. $\phi(\xi_\theta) = \phi(\xi_{\theta^*})$. This is illustrated in Figure 3.1 (bottom, right) as only the optimal green trajectory remains in each environment.

Our case study showcases the usefulness of combining types of feedback. A designer might start with their best guess at a reward function, the robot might misbehave in new environments, the designer or even end-user might observe this and intervene to correct or stop the robot, etc. – over time, the robot should narrow in on what people actually want it to do.

3.2 Actively selecting which type of feedback to use

Given that we can mix and match types of feedback, we may also wonder what is the best *type* to ask for at each point in time. To showcase the benefit of actively selecting feedback

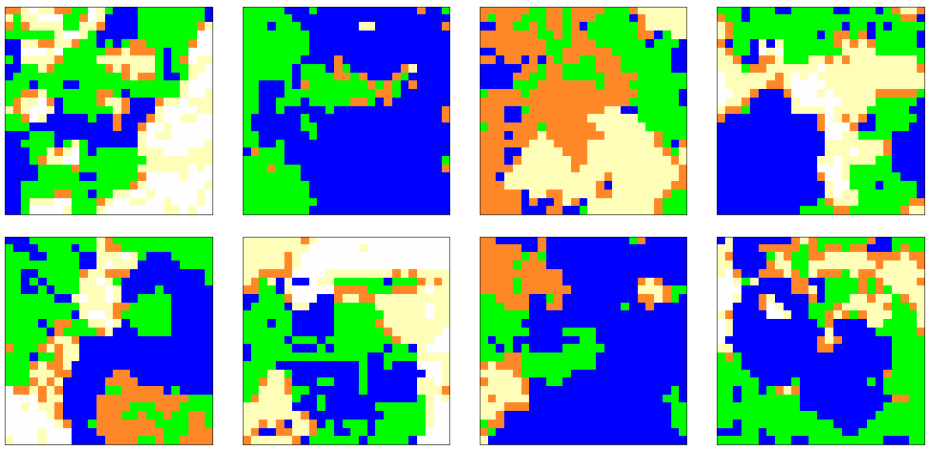

Figure 3.2: Environments used for experiments on active selection of feedback. (Top) These four environments were used during "training". (Bottom) These four environments were held as a test set to measure maximum and average regret.

types, we run an experiment actively selecting between demonstrations and comparisons. We measure regret (maximum and expected difference, on holdout environments, in ground truth reward between 1) optimizing with ground truth vs. 2) optimizing with the learned reward). We manipulate whether we have access to demonstrations only, comparisons only, or both, as well as the number of feedback instances queried.

One may initially wonder whether comparisons are necessary, given that demonstrations seem to provide so much information early on. Overall, we observe that demonstrations are optimal early on, when little is known about the reward, while comparisons become optimal later, as a way to fine-tune the reward (Fig. 3.3 shows our results)[2]. The observation also serves to validate the approach contributed by Palan et al. [2019], Ibarz et al. [2018] in the applications of motion planning and Atari game-playing, respectively. Both papers manually define the mixing procedure we found to be optimal: initially train the reward model using human demonstrations, and then fine-tune with comparisons.

Experiment Details We tested 3 different active learning methods: active querying of demonstrations, active querying of comparisons, and active querying of demonstrations and comparisons, across 8 different gridworld environments depicted in Figure 3.2. The top 4 environments were used in training while the bottom 4 were held for testing. Each environment e is a 25x25 gridworld MDP with a linear reward function in 3 features: RGB color values of each pixel. We assign each e with 10 different start goal pairs (s, g) from which the algorithms can ask queries. The goal of each algorithm is to efficiently recover a ground truth reward r^* through querying.

[2] More recent theoretical work by Skalse et al. [2022] on the *ambiguity* of different information sources may provide insight into why different sources encode complementary information about the reward function.

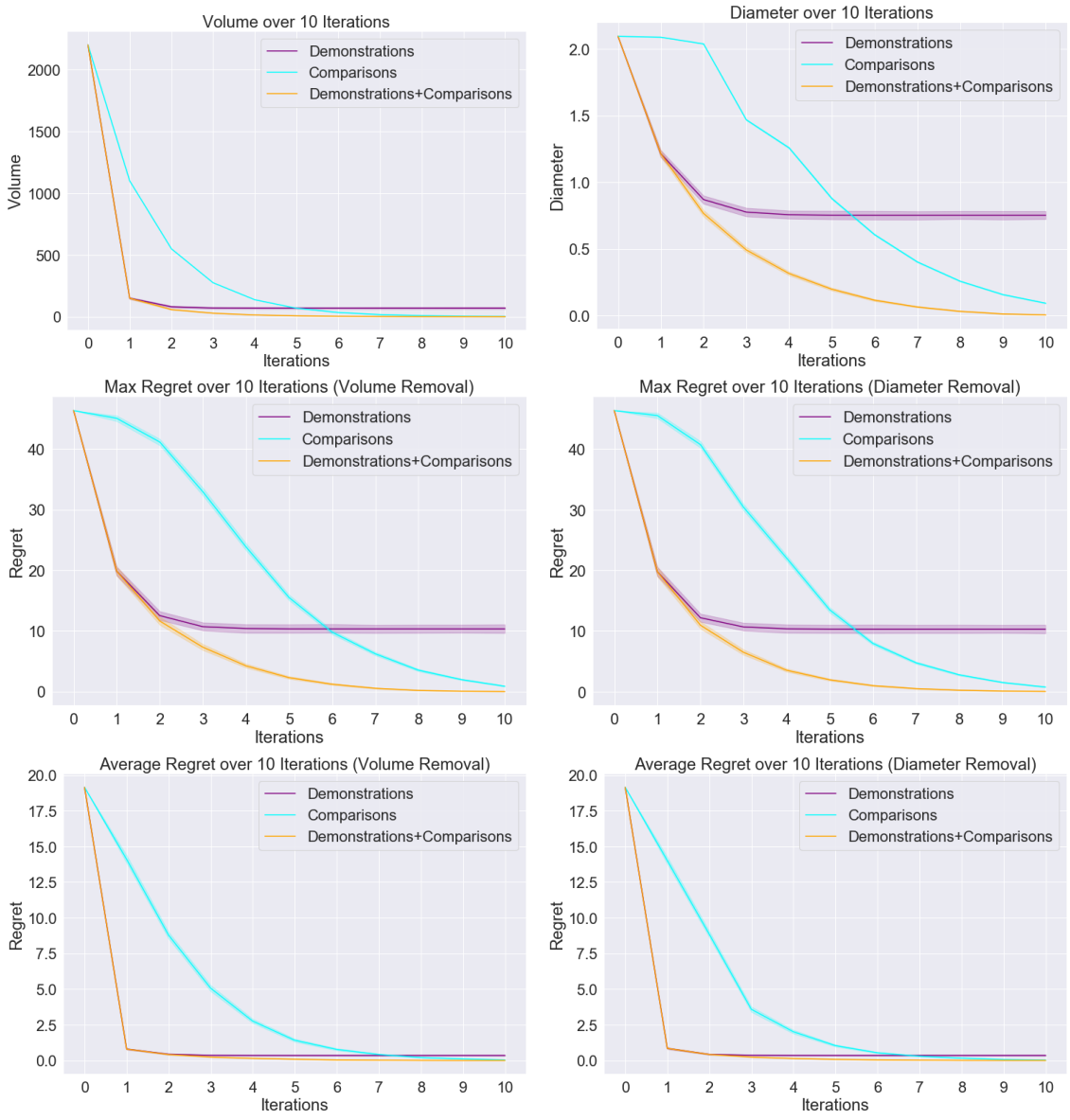

Figure 3.3: Statistics computed over 10 iterations of our greedy maximum information gain algorithm. We notice that demonstrations (purple) are initially very information-dense but quickly flatten out, whereas comparisons (cyan) obtain more information but less efficiently. We notice that combining the two methods (orange) inherits the positive aspects of both, the efficiency of demonstrations with the precision of comparisons.

Since our rewards are linear in RGB, the feasible reward set $\mathcal{R}$ consists of 3D parameters that weight the value of each feature in the reward function. $\mathcal{R}$ can be constrained to the surface of the 3D unit sphere since reward functions in MDPs are scale invariant. We uniformly discretize points at the surface of the 3D sphere to approximate $\mathcal{R}$ via $\hat{\mathcal{R}}$. To approximate Ξ, we first compute the optimal trajectory under each $r \in \hat{\mathcal{R}}$ to make $\{\arg\max_\xi \ r(\xi);\ r \in \hat{\mathcal{R}}\}$. We include trajectories that are not the result of optimizing reward functions by inserting noise into the value function when computing optimal trajectories as above.

Demonstrations and Comparisons as Hard Constraints The algorithms recover r^* by narrowing a set of feasible rewards with active queries. We use $\mathcal{R}_i$ to denote the set of feasible rewards at iteration i of querying. Demonstrations and comparisons shrink the feasible set in the following way:

$$\mathcal{R}_{i+1}^{\text{demo}}(\xi_d) = \{r : r(\xi_d) \geq r(\xi) \quad r \in \mathcal{R}_i; \quad \forall \xi \in \Xi\}$$

$$\mathcal{R}_{i+1}^{\text{comp}}(\xi_1, \xi_2) = \begin{cases} \mathcal{R}_{i+1}^{\text{comp}}(\underline{\xi_1}, \xi_2) = \{r : r(\xi_1) \geq r(\xi_2) \quad r \in \mathcal{R}_i\} & \xi_1 > \xi_2 \\ \mathcal{R}_{i+1}^{\text{comp}}(\xi_1, \underline{\xi_2}) = \{r : r(\xi_2) \geq r(\xi_1) \quad r \in \mathcal{R}_i\} & \xi_2 > \xi_1 \end{cases}$$

For our experiments, we performed the following greedy volume removal over possible (s, g) pairs that we specified in each environment.

$$\mathcal{R}_{i+1} = \begin{cases} \mathcal{R}_{i+1}^{\text{demo}}(\xi_d^*) & V_{\text{demo}} < V_{\text{comp}} \\ \mathcal{R}_{i+1}^{\text{comp}}(\xi_1, \xi_2) & V_{\text{comp}} < V_{\text{demo}} \end{cases}$$

$$V_{\text{comp}} = \min_{(s,g)} \max \ \left\{\mathcal{R}_{i+1}^{\text{comp}}(\underline{\xi_1}, \xi_2),\ \mathcal{R}_{i+1}^{\text{comp}}(\xi_1, \underline{\xi_2})\right\}$$

$$V_{\text{demo}} = \min_{(s,g)} \ \mathbb{E}_{r^* \in \mathcal{R}_i}\left[|\mathcal{R}_{i+1}(\xi_{r^*}^{\text{demo}})|\right]$$

For demonstrations, we look for the (s, g) pair that in expectation produces a demonstration that leave the smallest feasible set (size of feasible set is volume or diameter described below). For comparisons, we look for the pair of trajectories (ξ_1, ξ_2) that produces the minimum worst-case feasible region remaining. For the method with demonstrations and comparisons, we computed the above two metrics and select the feedback type with the smaller feasible region. We run this algorithm for 10 iterations and average our results across 50 different ground truth r^*. We plot several statistics for each iteration in Figure 3.3 including

$$\begin{cases} |\mathcal{R}_i| & \text{Volume at iteration i} \\ \sup_{r_1, r_2 \in \mathcal{R}_i} \|r_1 - r_2\|_2 & \text{Diameter at iteration i} \\ \max_{e;(s,g);\ r \in \mathcal{R}_i} \ r^*(\xi_{r^*}^{(e,s,g)}) - r^*(\xi_r^{(e,s,g)}) & \text{Max regret at iteration i} \\ \mathbb{E}_{e;(s,g);\ r \in \mathcal{R}_i} \left[r^*(\xi_{r^*}^{(e,s,g)}) - r^*(\xi_r^{(e,s,g)})\right] & \text{Avg regret at iteration i} \end{cases}$$

where e is a holdout environment and (s, g) is a start-goal pair in the MDP. Each metric is a proxy for how accurate our estimate of r^* is. We notice that the combination of demonstrations and comparisons achieves lower volume, diameter, max regret, and average regret than demonstrations alone and that it achieves this in fewer iterations than comparisons alone.

Chapter 4

Learning from the choice of feedback type

While of course the reward-rational choice formalism won't apply to *all* types of feedback, we believe that it applies to *many*, even to types that initially seem to have a more obvious, literal interpretation (e.g. reward and punishment, Section 2.3). Most immediately, we are excited about using it to formalize a particular new source of (leaked) information we uncovered while developing the formalism itself: the moment we enable robots to learn from multiple types of feedback, users will have the *choice* of which feedback to provide. Interpreted literally, each feedback gives the robot evidence about the reward. However, this leaves information on the table: if the person decided to, say, turn the robot off, they *implicitly* decided to *not* provide a correction, or use language. Intuitively, this means that turning off the robot was a more appropriate intervention with respect to the true reward. Interpreting the feedback type itself as reward-rational implicit choice has the potential to enable robots to extract more information about the reward from the same data. We call the choice of feedback type "meta-choice". In this chapter, we formalize meta-choice and conduct experiments that showcase its potential importance.

In Chapter 3, we described a straight-forward way of combining feedback types: treat each individual feedback received as an independent reward rational choice, and update the robot's belief (Equation 3.1). However, the moment we open it up to multiple types of feedback, the person is not stuck with a single type and is actually choosing which type to use. *We propose that this itself is a reward-rational implicit choice, and therefore leaks information about the reward.* We call the choice of feedback "meta-choice", and in this section, we formalize it and empirically showcase its potential importance.

4.1 Formalizing meta-choice

The assumption of conditional independence that the formulation in (3.1) uses is natural and makes sense in many settings. For example, during training time, we might control what

feedback type we ask the human for. We might start by asking the human for demonstrations, but then move on to other types of feedback, like corrections or comparisons, to get more fine-grained information about the reward function. Since the human is only ever considering one type of feedback at a time, the conditional independence assumption makes sense.

But the assumption breaks when the human has access to multiple types of feedback at once because *the types of feedback the robot can interpret influence what the human does in the first place.*[1] If the human intervenes and turns the robot off, that means one thing if this were the only feedback type available, and a whole different thing if, say, corrections were available too. In the latter case, we have more information - we know that the user chose to turn the robot off rather than provide a correction.

The type of feedback itself leaks information about the reward, and the RRC framework gives us a recipe for formalizing this new source: we need to uncover the set of options the human is choosing from. The human has two stages of choice: the first is the choice between feedback types, i.e corrections, language, turn-off, etc. and the second is the choice within the chosen feedback type, i.e the specific correction that the human gave. Our formalism can leverage both sources of information by defining a *hierarchy* of reward-rational choice.

Suppose the user has access to n types of feedback with associated choice sets $\mathcal{C}_1, \ldots, \mathcal{C}_n$, groundings $\psi_1, \ldots \psi_n$, and Boltzmann rationalities $\beta_1, \ldots, \beta_n$. For simplicity, we assume deterministic groundings. The set of choice sets $\mathscr{C}_0$ for the first-stage choice is $\{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ The grounding $\psi_0 : \mathcal{C} \to f_\Xi$ for the first stage choice maps a feedback type $\mathcal{C}_i$ to the distribution of trajectories defined by the human's behavior and grounding in the second stage:

$$\psi_0(C_i) \;=\; \mathbb{P}(\xi \mid r, \mathcal{C}_i) = \sum_{c_i \in \mathcal{C}_i : \psi_i(c_i) = \xi} \mathbb{P}(c_i \mid r, \mathcal{C}_i)\,, \tag{4.1}$$

where, as usual, $\mathbb{P}(c_i \mid r, \mathcal{C}_i)$ is given by Equation 2.1. Instantiating Equation 2.1 to model the first-stage decision as well, results in the following model for the human picking feedback type $\mathcal{C}_i$:

$$\mathbb{P}(\mathcal{C}_i \mid r) = \frac{\exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_i)}[r(\xi)]\right)}{\sum_{j \in [n]} \exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_j)}[r(\xi)]\right)}\,, \tag{4.2}$$

The human is more likely to pick a feedback type $\mathcal{C}_i$ in the first stage, if the distribution of choices $\psi_0(\mathcal{C}_i)$ that they would choose in the second stage achieves higher reward: $\mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_i)}[r(\xi)]$. To get the probability that the human gives feedback c^*, we combine the

[1] We note that this adaptation by the human only applies to types of behavior that the human uses to purposefully communicate with the robot, as opposed to sources of information like initial state.

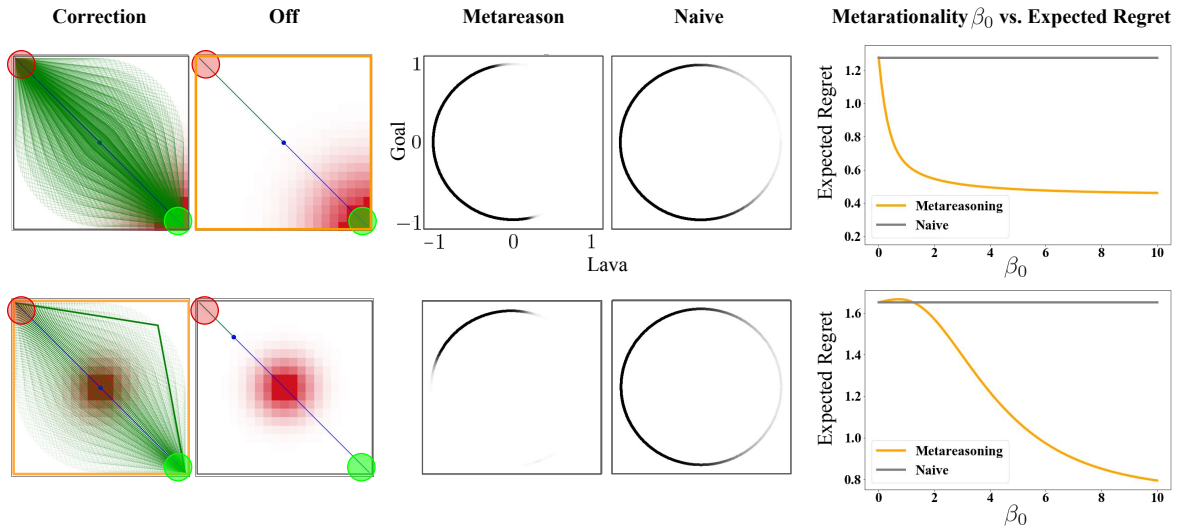

Figure 4.1: (Left) Environment with designated start (red circle), goal (green circle) and lava area (red tiles). The human can provide a correction (one of the green trajectories) or turn off the robot, forcing the robot to stop at the marked dot. (Middle) Belief distribution over rewards after the human provides feedback ($\beta_0 = 10.0$). Darker indicates higher probability. The metareasoning model is able to rule out more reward functions than the naive model. (Right) When the human's metareasoning has no signal ($\beta_0 = 0$), then the metareasoning (orange) and naive model (gray) perform equally well. As β_0 increases, the advantage of the metareasoning model also increases.

policies at the first and second stage:

$$\mathbb{P}(c^* \mid r) = \sum_i \mathbb{P}(c^* \mid r, \mathcal{C}_i) \cdot \mathbb{P}(\mathcal{C}_i \mid r) \tag{4.3}$$

$$= \sum_i \left(\frac{\exp\left(\beta_i \cdot r(\psi_i(c^*))\right)}{\sum_{c \in \mathcal{C}_i} \exp\left(\beta_i \cdot r(\psi_i(c))\right)} \cdot \frac{\exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_i)}[r(\xi)]\right)}{\sum_{j \in [n]} \exp\left(\beta_0 \cdot \mathbb{E}_{\xi \sim \psi_0(\mathcal{C}_j)}[r(\xi)]\right)} \right). \tag{4.4}$$

The first-stage decision can be interpreted as the human *metareasoning* over the best type of feedback. The benefit of modeling the hierarchy is that we can cleanly separate and consider noise at both the level of metareasoning (β_0) and the level of execution of feedback ($\beta_1, \ldots, \beta_n$). Noise at the metareasoning level models the human's imperfection in picking the optimal type of feedback. Noise at the execution level might model the fact that the human has difficulty in physically correcting a heavy and unintuitive robot.[2]

[2]Although we modeled rationality with respect to the reward r that the robot should optimize, we can easily extend our formalism to capture that the person might trade-off between that and their own effort – this is especially interesting at this meta-choice level, where one type of feedback might be much more difficult and thus people might want to avoid it unless it is particularly informative.

4.2 Comparing the literal interpretation to meta-choice

We showcase the potential importance of accounting for the meta-choice in an experiment in a gridworld setting, in which an agent navigates to a goal state while avoiding lava (Figure 4.1, left). The reward function is a linear combination of 2 features that encode the goal and lava. The human has access to two channels of feedback: "off" and corrections. We simulate the human feedback as choosing between *feedback types* according to Equation 4.4. We manipulate three factors: 1) whether the robot is *naive*, i.e. only accounts for the information *within* the feedback type, or *metareasons*, i.e. accounts for the other feedback types that were available but not chosen; 2) the meta-rationality parameter β_0 modeling human imperfection in selecting the optimal type of feedback; and 3) the location of the lava, so that the rational meta-choice changes from off to corrections. We measure regret over holdout environments.

Figure 4.1 (left) depicts the possible grounded trajectories for corrections and for off. For the top, off is optimal because all corrections go through lava. For the bottom, the rational meta-choice is to correct. In both cases, we find that meta-reasoning gains the learner more information, as seen in the belief (center). For the top, where the person turns it off, the robot can be more confident that lava is bad. For the bottom, the fact that the person had the off option and did not use it informs the robot about the importance of reaching the goal. This translates into lower regret (right), especially as β_0 increases and there is more signal in the feedback type choice.

4.3 What happens when metarationality is misspecified?

In our main metareasoning experiments, we assumed that the simulated human metareasoned with β_0 and that our algorithm somehow knew this quantity. However, in practice, we will not have access to β_0. This brings about an interesting question: What are the effects of inference under a misspecified β_0? What are the effects of overestimating or underestimating the human's rationality?

To test this, we designed an experiment in which our simulated human provided supervision with a fixed ground truth r^* and β_0^* while our algorithm performs belief updates with various β_0 above and below β_0^*. The first way to measure the extent of misspecification is to measure the KL divergence between the belief induced by β_0^* and that induced by β_0.

$$D_{KL}(P(r \mid c^*) \| Q(r \mid c^*))$$

$$P(r \mid c^*) \propto P(c^* \mid r, \beta_0^*) \cdot P(r)$$

$$Q(r \mid c^*) \propto P(c^* \mid r, \beta_0) \cdot Q(r)$$

Additionally, we wanted to measure the expected regret given a human that provides supervision with rationality β_0^* and the algorithm that performs belief updates with rationality β_0.

$$\mathbb{E}[\text{Regret} \mid c^*, C_i, r^*, \beta_0] = \sum_{r \in \mathcal{R}} (r^*(\phi(r^*)) - r^*(\phi(r))) \cdot \mathbb{P}(r \mid c^*, C_i, \beta_0)$$

$$\mathbb{E}[\text{Regret} \mid \beta_0] = \sum_{r^* \in \mathcal{R}} \sum_{i \in \mathcal{C}_0} \sum_{c^* \in C_i} \mathbb{E}[\text{Regret} \mid c^*, C_i, r^*, \beta_0] \cdot \mathbb{P}(C_i \mid r^*, \beta_0) \cdot \mathbb{P}(c^* \mid C_i, r^*, \beta_0)$$

We plot the results in Figure 4.2 averaged over 50 randomly sampled reward functions and $\beta_0 \in [0.0, 10.0]$. We notice that when the human does not metareason ($\beta_0^* = 0.0$), the KL divergence in the belief distribution update is large. In comparison, with any moderate level of metareasoning ($\beta_0^* = 2.5, 5.0, 7.5$), the KL divergence is very low. We notice this too in the expected regret. Note that the minimum expected regret is not achieved by $\beta_0 = \beta_0^*$. This is because β_0^* is used to compute the frequency at which the human provides each type of feedback as an answer. Simply matching β_0 with β_0^* doesn't guarantee minimum expected regret (the optimal β_0 for minimizing expected regret is a function of β_0^*). These experiments suggest that if we detect that the human is poor at metareasoning (low β_0^*), it is safer to drop the metareasoning assumption. However, if the human is displaying metareasoning, we can leverage this to improve learning.

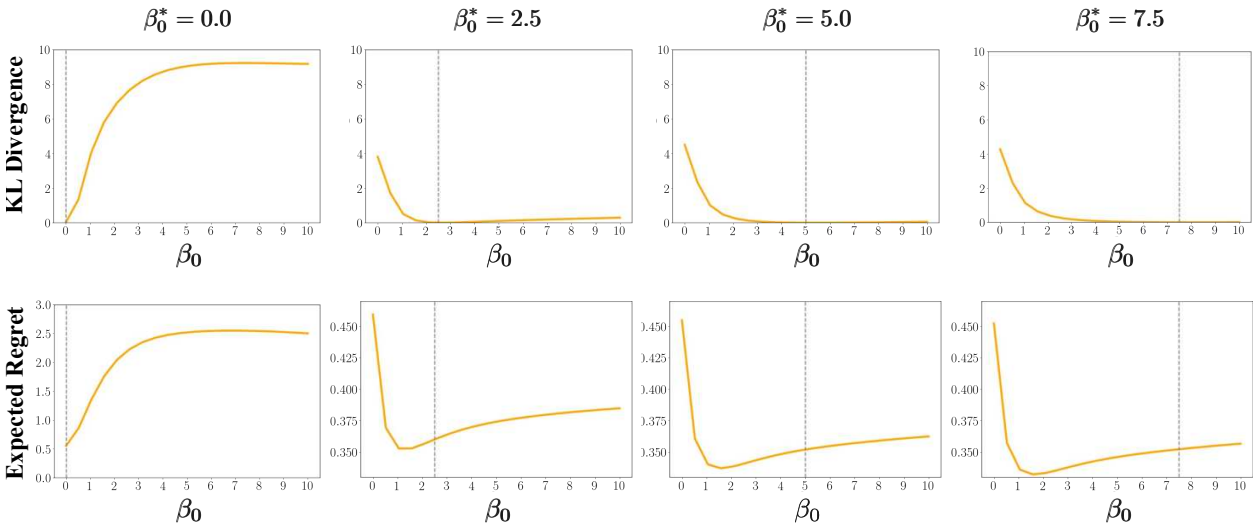

Figure 4.2: In each plot, the human operates under a true metarationality denoted by β_0^*. We measure performance drop from misspecification by computing the KL divergence and expected regret of the belief distribution over rewards for robots with misspecified metarationalities $\beta_0 \in [0.0, 10.0]$. (Top) The plots display the KL divergence between the true belief with β_0^* and various misspecified beliefs. We notice that assuming metareasoning when the human does not metareason (left $\beta_0^* = 0$) results in significant divergence in the belief distribution. (Bottom) The plots show the expected regret for robots that learn, with misspecified β_0's, from a human who gives feedback with β_0^*. As with *KL divergence*, the *expected regret* incurred by assuming metareasoning when the human does not metareason is high. Additionally, we note that a robot learning with $\beta_0 = \beta_0^*$ does not necessarily incur minimum expected regret (as mentioned in the main text).

Part III

Beyond the reward-rational assumption

Chapter 5

Learning from a human that may be pedagogic

The reward-rational choice framework assumes that the human's choice is near-optimal with respect to the reward function to be inferred. This in the natural assumption for many situations, e.g. an autonomous car learning a reward function for driving through observing natural, human demonstrations of driving [Levine and Koltun, 2012]. However, recent work shows that this assumption may not hold in certain collaborative settings [Ho et al., 2016] in which the human is *aware* that the robot needs to learn. In such settings, the person might optimize for *teaching* the robot about the objective, which is not the same as directly optimizing the objective itself [Dragan et al., 2013b, Hadfield-Menell et al., 2016, Ho et al., 2016].

Prior work has suggested that when the human is teaching, the robot should infer the reward function using a pedagogic model of the human, rather than a reward-rational model of the human [Fisac et al., 2018]. Here, we study impact of the human model (pedagogic vs reward-rational) on reward inference. We find that the reward-rational assumption is quite robust to humans who are teaching. First, we prove a theoretical result stating that if one does not know whether the human is pedagogic or reward-rational, it is more robust to use the reward-rational assumption. Then, in experiments with humans, we find that the reward-rational model performs better than the pedagogic model, even with humans are acting pedagogically! Given that any pedagogic model will also always deviate from actual human behavior, the reward-rational assumption may simply be a more robust assumption.

In this chapter, we refer to a reward-rational human as *literal* and a human who is teaching as *pedagogic*. We similarly define a literal and pedagogic robot based upon which model of the human (literal or pedagogic) the robot uses.

1. The **literal human** directly optimizes for the objective, i.e., the reward-rational assumption.

2. The **literal robot** infers the objective while assuming the human is literal (reward-rational).

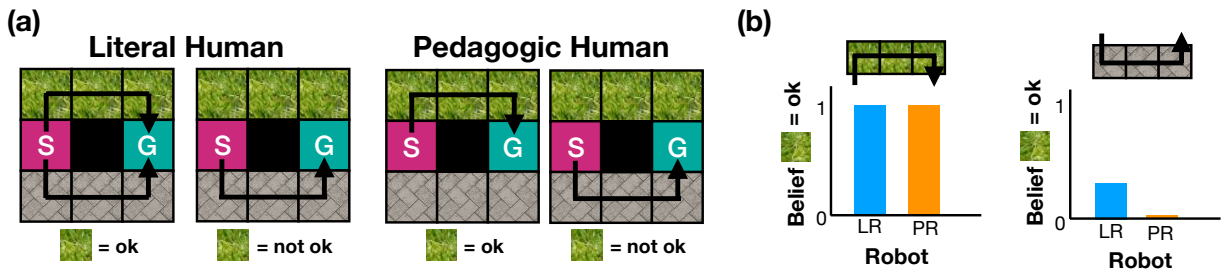

Figure 5.1: In this scenario, the robot infers whether it is ok to walk on the grass or not from a demonstration provided by the human (it already knows pavement is always ok). (a) The literal and pedagogic human's demonstrations. When grass is ok, the literal human is equally likely to walk on the grass or pavement. On the other hand, when grass is ok, the pedagogic human always walks on the grass in order to signal that grass is ok. (b) The beliefs of the literal robot **LR** and pedagogic robot **PR** after observing a demonstration. **LR** and **PR** assume human is literal and pedagogic, respectively. **LR** and **PR** differ in how strong their beliefs are after witnessing the human walk on pavement, leading to different problems when the human is misspecified. If the human is literal, but the robot is pedagogic, then it makes too strong an inference. On the other hand, if the human is pedagogic, but the robot is literal, then it makes too weak of an inference.

3. The **pedagogic human** optimizes for teaching the *literal* robot the objective.

4. The **pedagogic robot** (sometimes called "pragmatic" [Fisac et al., 2018]) infers the objective while assuming the human is *pedagogic*.

Figure 5.1 shows an example of the difference between a literal and pedagogic human. When the human is literal and optimizes for the objective, they take any optimal path to the goal. When the human is pedagogic and optimizes for teaching the objective, they take the path that best signals the objective.

Which is worse—interpreting a pedagogic human literally (literal robot + pedagogic human) or interpreting a literal human pedagogically (pedagogic robot + literal human)? In both cases, the model of the human is incorrect, and we might expect that which is better depends on the context and the task; surprisingly, we are able to prove that regardless of the task, the literal robot is more robust, suggesting that it may be safer to simply model the human as reward-rational. Our contributions are the following[1]:

- Section 5.1: *Theoretical Analysis.* We cast objective learning into the more general form of a common-payoff game between the human and robot. We then prove that a pedagogic robot and a literal human always do worse than a literal robot and a

[1]The contents of this chapter were originally published with Anca Dragan as "Literal or Pedagogic Human? Analyzing Human Model Misspecification in Objective Learning" in UAI 2019.

pedagogic human, showing that misspecification is worse in one direction than the other.

- Section 5.2: *Empirical Analysis.* We test the effects of misspecification on data from human teaching. The data confirms that assuming pedagogic behavior when people are literal is worse than assuming literal behavior when people are pedagogic. Surprisingly, we find that the literal robot does better than the pedagogic robot *even when people are trying to be pedagogic*, because of the discrepancy between the pedagogic *model* of humans and real human behavior.

- Section 5.3: *Theoretical vs Empirical.* Our empirical results are surprising, in that the pedagogic model is a state of the art cognitive science model that is fit to the human data and has relatively high predictive accuracy, yet the pedagogic robot does worse than the literal robot *with humans who are trying to be pedagogic*. We use our theory to derive a hypothesis for why this could be. The hypothesis is that in practice different humans vary in how literal or pedagogic they are, which our theory implies will degrade the performance of the pedagogic robot more than the literal robot. We find positive evidence for this hypothesis, indicating that robustness to a *population* of humans is an important consideration in choosing a pedagogic versus literal robot.

- Section 5.4: *Can we "fix" the pedagogic robot?* An intuitive idea for improving the pedagogic robot is to give it a model of the pedagogic human that has higher predictive accuracy. For example, what if instead of assuming all people are pedagogic, the robot estimated how pedagogic each person is? Unfortunately, we find that this makes no difference. And in fact, we show that better models can actually *worsen* performance due to a subtle, yet remarkable fact: a human model with higher *predictive accuracy* does not necessarily imply higher *inferential accuracy* for the robot.

In conclusion, we found that not only are pedagogic robots less robust to the human's recursion level, they can also perform worse when people are actually being pedagogic – even with a state of the art pedagogic model tuned to human data. This points to a surprising brittleness of the pedagogic assumption. Further, we point out that a more predictive human model will not necessarily solve the problem because improving the model's predictive accuracy does not necessarily lead to better robot inference.

The difficulty of *reward specification* was an important motivation for pursuing reward learning in the first place. But in our pursuit of reward learning, we ought to be careful to not simply trade the problem of reward specification for the equally, if not more, difficult problem of *human specification*. In practice, humans will deviate from our models of them, and thus, it is important to understand which assumptions are robust and which are not. The reward-rational choice framework provides a simple model of humans that is robust even in collaborative settings when the human may be pedagogic.

5.1 Theoretical analysis of robustness

In this section, we cast objective learning into the more general form of a common-payoff[2] game between the robot and human. We then formalize the literal/pedagogic robot/human, and prove that in *any* common-payoff game a literal robot and pedagogic human perform better than a pedagogic robot and literal human.

Generalizing objective learning to CI(RL)

Before proceeding to our proof, we give background on how common-payoff games generalize standard objective learning. In particular, objective learning can be modeled as a cooperative inverse reinforcement learning (CIRL) game [Hadfield-Menell et al., 2016], a common-payoff game in which only the human knows an objective/reward function r.

Formally, a CIRL game is defined as a tuple $\langle \mathcal{S}, \{\mathcal{A}^{\mathbf{H}}, \mathcal{A}^{\mathbf{R}}\}, \mathsf{T}, \{\mathcal{R}, r\}, P_0, \gamma\}$ where $\mathcal{S}$ is the set of states, $\mathcal{A}^{\mathbf{H}}$ and $\mathcal{A}^{\mathbf{R}}$ are the set of actions available to the human and robot, $\mathsf{T}(s' \mid s, a^{\mathbf{H}}, a^{\mathbf{R}})$ is the transition distribution specifying the probability of transitioning to a new state s' given the previous state s and the actions $a^{\mathbf{H}}$ and $a^{\mathbf{R}}$ of both agents, R is the space of reward functions, $r : \mathcal{S} \times \mathcal{A}^{\mathbf{H}} \times \mathcal{A}^{\mathbf{R}} \to \mathbb{R}$ is the shared reward function (known only to $\mathbf{H}$), P_0 is the initial distribution over states and reward functions, and γ is the discount factor.

The joint payoff in a CIRL game is traditionally *value*, expected sum of rewards. Since only the human in CIRL knows the shared reward function, this indirectly incentivizes the human to act in ways that signal the reward function and incentivizes the robot to learn about the reward function from the human's behavior. In fact, in Demonstration-CIRL (Example 5.1.1), the best that the robot can do is infer a posterior distribution over rewards from the human's demonstrations, and then act optimally with respect to the posterior mean. However, we can also consider a version that directly incentivizes reward inference by making the payoff the accuracy of the robot's inference.

To illustrate how objective learning settings are special cases of CIRL games, we give an example for the learning-from-demonstrations setting.

Example 5.1.1 (Demonstration-CI(RL))**.** Learning from demonstrations can be modeled as a common-payoff game with two phases. In the first phase, the human provides demonstrations. In the second phase,

(a) in CIRL, the robot acts in the environment. The game's joint payoff for the robot and human is the value (expected sum of rewards) attained by the robot.

(b) in CI, the robot outputs an estimate of the reward function. The game's joint payoff for the robot and human is a measure of the accuracy of the robot's inference.

The second formulation (b) is a *cooperative inference* (CI) problem [Yang et al., 2018, Wang et al., 2019]. In CI, there is a teacher (e.g, human) who is teaching a learner (e.g.,

[2] A game in which all agents have the same payoff.

robot) a hypothesis r (e.g, the reward) via data d (e.g., demonstrations). The human has a distribution over demonstrations $p^{\mathbf{H}}(d \mid r)$ and the robot has a distribution over rewards $p^{\mathbf{R}}(r \mid d)$. Given a starting distribution of human demonstrations, $p_0^{\mathbf{H}}(d \mid r)$, the optimal solution for these two distributions can be found via fixed-point iteration of the following recursive equations[3]:

$$p_k^{\mathbf{R}}(r \mid d) \propto p_k^{\mathbf{H}}(d \mid r)\,, \tag{5.1}$$

$$p_{k+1}^{\mathbf{H}}(d \mid r) \propto p_k^{\mathbf{R}}(r \mid d)\,. \tag{5.2}$$

Theoretical comparison: literal robots are more robust

We now proceed to formalize what we mean by literal and pedagogic and to prove that the literal robot is more robust to misspecification of whether the human is literal or pedagogic. Suppose that the human and robot are acting in a common-payoff game. Let $\mathcal{H}$ and $\mathcal{R}$ be the space of policies for the human and robot, respectively. The joint payoff for the human and robot is denoted by $U : \mathcal{H} \times \mathcal{R} \to \mathbb{R}$. The joint payoff function could be value, as assumed by CIRL, or accuracy of inference, as assumed by CI.

To define literal and pedagogic, we define a set of recursive policies for the human and robot. Let H_0 be a starting human policy. For $k \geq 0$, define the recursive policies

$$R_k = \mathbf{BR}(H_k)\,, \tag{5.3}$$

$$H_{k+1} = \mathbf{IR}(R_k)\,. \tag{5.4}$$

We assume that at each level of recursion the robot does a best response $\mathbf{BR} : \mathcal{H} \to \mathcal{R}$. However, for the human, we only assume that at each step she improves over her previous policy. This allows for arbitrary irrationalities, so long as the next policy is at least as good as the previous one. We call this an "improving" response $\mathbf{IR} : \mathcal{R} \to \mathcal{H}$, and define both types of responses below.

Definition 5.1.1. A *best response* for the robot is a function $\mathbf{BR} : \mathcal{H} \to \mathcal{R}$ such that for any human policy $H \in \mathcal{H}$ and robot policy $R \in \mathcal{R}$,

$$U(H, \mathbf{BR}(H)) \geq U(H, R)\,.$$

Definition 5.1.2. An *improving response* for the human is a function $\mathbf{IR} : \mathcal{R} \to \mathcal{H}$ such that $\forall\, k \geq 0$,

$$U(\mathbf{IR}(R_k), R_k) \geq U(H_k, R_k)\,.$$

[3]Wang et al. [2019] show that fixed-point iteration converges for all discrete distributions. For simplicity, we have written equations (5.1) and (5.2) assuming that the human and robot's prior over rewards and demonstrations is uniform, but in general, the equations can incorporate any prior [Yang et al., 2018].

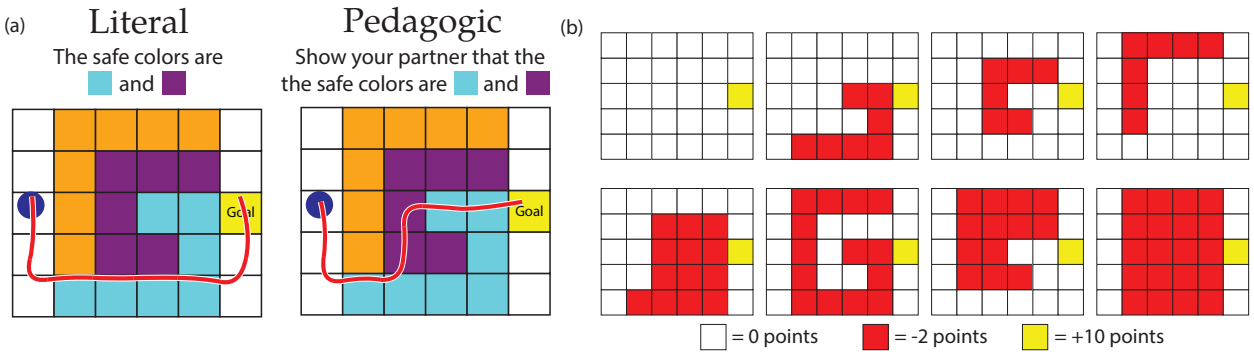

Figure 5.2: (a) The instructions given to participants in the literal and pedagogic condition, and a sample demonstration from both conditions. In the pedagogic case, participants were more likely to visit multiple safe colors, as well as to loop over safe tiles multiple times. (b) All possible reward functions. Each tile color can be either safe (0 points) or dangerous (-2 points). Figure modified from Ho et al. [2018].

We give special emphasis to what we call the *literal* human and robot, H_0 and R_0, and the *pedagogic* human and robot, H_1 and R_1.[4] We now provide an example of the literal and pedagogic policies for the Demonstration-CI setting (Example 5.1.1b). In this case, the human and robot policies can be modeled by the recursive CI equations (5.1) and (5.2).

1. The **literal human** H_0 is noisily-optimal with respect to the reward function r. The probability $p_0^{\mathbf{H}}(d \mid r)$ that she gives a demonstration d is exponentially proportional to the reward of the demonstration, denoted by $r(d)$:

$$p_0^{\mathbf{H}}(d \mid r) \propto \exp(r(d))\,.$$

2. The **literal robot** R_0 does a **BR** to the literal human, i.e. does Bayesian inference assuming the human is literal,

$$p_0^{\mathbf{R}}(r \mid d) \propto p_0^{\mathbf{H}}(d \mid r)\,,$$

   and then uses the posterior mode as its estimate for the reward r.

3. The **pedagogic human** H_1 picks demonstrations that are informative to the literal robot[5]:

$$p_1^{\mathbf{H}}(d \mid r) \propto p_0^{\mathbf{R}}(r \mid d)\,.$$

[4]There is nothing that requires the literal level to be at recursion level 0 and the pedagogic level to be at recursion level 1. Our proof of Claim 5.1.1 holds when the literal level is any $k \geq 0$ and the pedagogic level is $k+1$.

[5]Typically, the human is modeled as being exponentially more informative: $p_1^{\mathbf{H}}(d \mid r) \propto \exp(p_0^{\mathbf{R}}(r \mid d))$. This is the form we will use in our experiments.

Note this is not a best response, which would unrealistically require the human to choose $\arg\max_d p_0^{\mathbf{R}}(r \mid d)$.

4. The **pedagogic robot** R_1 does a **BR** to the pedagogic human, i.e. does Bayesian inference assuming the human is literal,

$$p_1^{\mathbf{R}}(r \mid d) \propto p_1^{\mathbf{H}}(d \mid r)\,,$$

and then uses the posterior mode as its estimate for the reward r.

We show that for *any* common-payoff game the payoffs for the literal/pedagogic human/robot pairs have the following ranking:

$$\begin{aligned} &\text{Pedagogic } R_1,\, H_1 \geq \text{Literal } R_0,\, \text{Pedagogic } H_1 \\ &\geq \text{Literal } R_0,\, H_0 \geq \text{Pedagogic } R_1,\, \text{Literal } H_0\,. \end{aligned} \tag{5.5}$$

In particular, a pedagogic robot R_1 and a literal human H_0 always do worse than a literal robot R_0 and a pedagogic human H_1, showing misspecification is worse one way than the other. The ranking has the following straight-forward proof.

Claim 5.1.1. In any common-payoff game, the ranking of payoffs between a literal/pedagogic human/robot is

$$U(H_1, R_1) \geq U(H_1, R_0) \geq U(H_0, R_0) \geq U(H_0, R_1)\,.$$

Proof. Since $R_1 = \mathbf{BR}(H_1)$, we have $U(H_1, R_1) \geq U(H_1, R_0)$. Since $H_1 = \mathbf{IR}(R_0)$, we have $U(H_1, R_0) \geq U(H_0, R_0)$. Since $R_0 = \mathbf{BR}(H_0)$, we have $U(H_0, R_0) \geq U(H_0, R_1)$. □

5.2 Empirical analysis of robustness

In this section, we test our results in practice using experimental data from actual humans, who provide demonstrations that the robot uses to infer the objective from. These experiments are a way of stress-testing our theoretical results, which assumed that at each level of recursion, the robot computes a best response and that the human never does worse than her previous level. In practice, there will be a difference between what humans actually do, and the model of the human, which could cause both the human and robot to break our theoretical assumptions.

Experimental design

We test the performance of literal/pedagogic robot/human pairs on data from an experiment run by Ho et al. [2016][6], which was subsequently followed up on by Ho et al. [2018]. In the

[6]The experiment is Experiment 2 in their paper.

experiment, humans are asked to act in different types of gridworlds. The gridworld has one goal state that is worth 10 points and three other types of tiles (orange, purple, cyan). Each type of tile can each be either “safe" (0 points) or “dangerous" (-2 points). Thus, there are $2^3 = 8$ possible reward functions, which are depicted in Figure 5.2b.

Sixty particpants were recruited from Mechanical Turk. The participants are told that they will get two cents of bonus for each point they get. Participants were split into two conditions, a literal and pedagogic condition, depicted in Figure 5.2a. In the literal condition, the participant only gets points for their own actions in the gridworld. In the pedagogic condition, the participant is told that their demonstration will be shown to another person, a learner, who will then apply what they learn from the demonstration to act in a separate gridworld. The participant still gets points based on their own actions, but is also told that the number of points the learner receives will be added as a bonus.

Human and Robot Models

We model the robot and human following that of Ho et al. [2018]. We use the same model parameters that Ho et al. [2018] found to be the best qualitative match to the human demonstrations.

Notation. Let $\mathcal{S}$ be the set of states and $\mathcal{A}$ be the set of actions. The gridworld has a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$. The optimal Q-value function for a reward function r is denoted by $Q^*_r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The robot has a uniform prior belief over the reward function, i.e, it puts uniform probability on all $2^3 = 8$ reward functions depicted in Figure 5.2b. The human and robot models are as follows.

1. **Literal H.** At each time step t, the probability $H_L(a_t|s_t, r)$ that the literal human takes action a_t given state s_t and the reward r is exponentially proportional to the optimal Q-value,

$$H_L(a_t \mid s_t, r) \propto \exp(Q^*_r(s_t, a_t)/\tau_L)\,. \tag{5.6}$$

   The temperature parameter τ_L controls how noisy the human is.

2. **Literal R.** The robot does Bayesian inference, while assuming that the human is literal. The robot's posterior belief at time $t + 1$ is

$$\begin{aligned} R^{t+1}_L(r) \propto\ & H_L(a_t \mid s_t, r) \\ & \cdot T(s_{t+1} \mid s_t, a_t) \cdot R^t_L(r)\,. \end{aligned} \tag{5.7}$$

3. **Pedagogic H.** The pedagogic human optimizes a reward function r' that trades off between optimizing the reward in the gridworld and teaching the literal robot the reward. At time step t, the reward is $r'(s_t, a_t, s_{t+1}) = r(s_t, a_t, s_{t+1}) + \kappa(R^{t+1}_L(r) - R^t_L(r))$. The parameter $\kappa \geq 0$ controls how “pedagogic” the human is. The probability $H_P(a_t \mid s_t, r)$ that the pedagogic human takes action a_t given state s_t and the reward r is

exponentially proportional to the optimal Q-value associated with the modified reward r',

$$H_P(a_t \mid s_t, r) \propto \exp(Q^*_{r'}(s_t, a_t)/\tau_P)\,. \tag{5.8}$$

The temperature parameter τ_P controls how noisy the human is.

4. **Pedagogic R.** The robot does Bayesian inference, while assuming that the human is pedagogic. The robot's posterior belief at time $t+1$ is

$$\begin{aligned} R_P^{t+1}(r) \propto\ & H_P(a_t \mid s_t, r) \\ & \cdot T(s_{t+1} \mid s_t, a_t) \cdot R_P^t(r)\,. \end{aligned} \tag{5.9}$$

Results

We evaluate each literal/pedagogic robot/human pair on the accuracy of the robot's inference, i.e, $\mathbb{P}(\hat{r} = r)$, where r is the true reward and $\hat{r}$ is the robot's guess. We take the robot's guess $\hat{r}$ to be the mode of its belief, as given by the robot models (5.7) and (5.9). We test each pair with both the demonstrations generated by actual humans and demonstrations generated by simulating humans according to the human models (5.6) and (5.7).

Figure 5.3a depicts our experimental results[7]. We refer to the actual human as **AH**, the human model as **H**, and the robot as **R**. Consistent with the theory, the performance of pedagogic **R** and literal **AH** is (significantly) worse than that of literal **R** and pedagogic **AH**, validating that misspecification is worse one way than the other. However, the overall ranking of robot/human pairs does not match the theoretical ranking (Equation 5.5) we expected. Surprisingly, even when **AH** is pedagogic, pedagogic **R** performs (insignificantly) worse than literal **R**. The empirical ranking is

$$\begin{aligned} &\text{Literal } \mathbf{R}, \text{ Pedagogic } \mathbf{AH} \geq \text{Pedagogic } \mathbf{R}, \mathbf{AH} \\ &\geq \text{Literal } \mathbf{R}, \mathbf{AH} \geq \text{Pedagogic } \mathbf{R}, \text{Literal } \mathbf{AH}\,. \end{aligned}$$

The empirical ranking carries a stronger implication than our theoretical ranking—it implies that regardless of whether the human is literal or pedagogic, the robot should be literal.

5.3 Explaining the gap between the empirical and theoretical results

In our empirical analysis, we found a perplexing result: the literal robot does better than the pedagogic robot *even when people are trying to be pedagogic*. How could this be? Figure 5.3b shows the accuracy of robot/human pairs when the human demonstrations are simulated

[7] All error bars and confidence bands in the paper depict bootstrapped 95% confidence intervals.

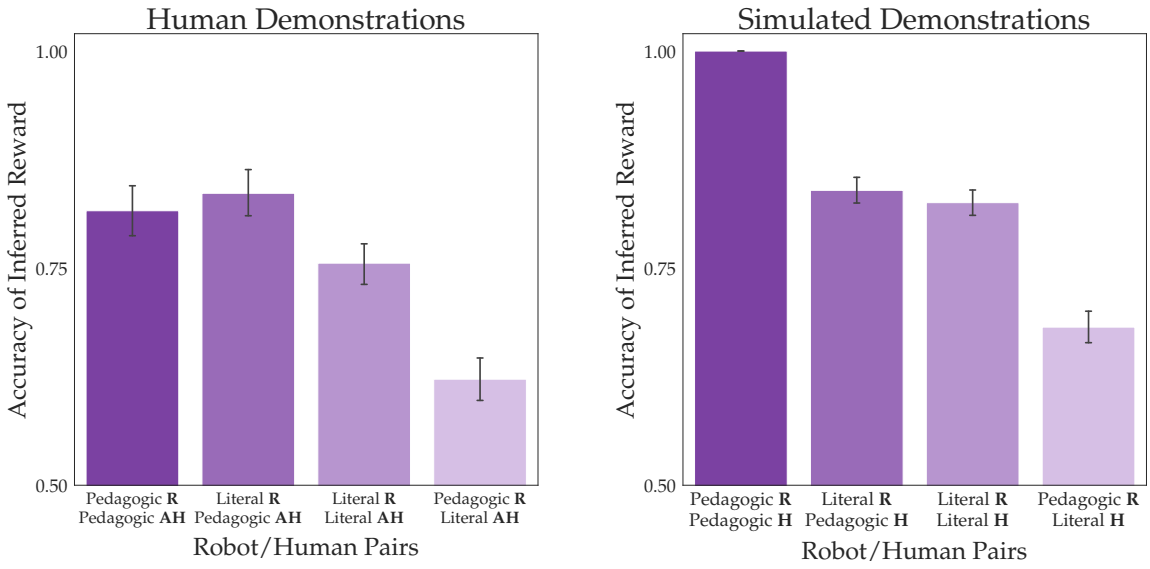

Figure 5.3: The accuracy of the robot's inferred reward for different pairs of human/robot pairs under the demonstrations provided by actual humans (left/a) and humans simulated according to the human models in Section 5.2. (right/b).

from the literal **H** and pedagogic **H** models described in equations (5.6) and (5.8). In the simulations, we find, as expected, that pedagogic **R** and pedagogic **H** do better than literal **R** and pedagogic **H**. So clearly, the reason pedagogic **R** does worse in the human experiments is that pedagogic **AH** (actual humans) is not the same as the pedagogic **H** (the human model).

But, in *what way* does pedagogic **AH** deviate from pedagogic **H**? Why is literal **R** more robust to the deviation than pedagogic **R**? We derive a hypothesis from our theory. In the theoretical ranking (Equation 5.5), the performance of literal **R** and literal/pedagogic **H** is sandwiched between the performance of pedagogic **R** + pedagogic **H** and pedagogic **R** + literal **H**. Thus, literal **R** is more robust to whether **H** is literal or pedagogic than pedagogic **R**. This suggests an explanation for why literal **R** does better, namely, that pedagogic **AH** is actually sometimes literal!

Hypothesis 5.3.1. Pedagogic **AH** is actually "in between" literal **H** and pedagogic **H**. If this is true, then our theory implies that literal **R** will be affected less than pedagogic **R**, potentially explaining why pedagogic **R** does worse with pedagogic **AH** than literal **R**.

To test Hypothesis 5.3.1, we create two models that are mixtures of the literal and pedagogic model. We then measure how much better the mixture models are at predicting

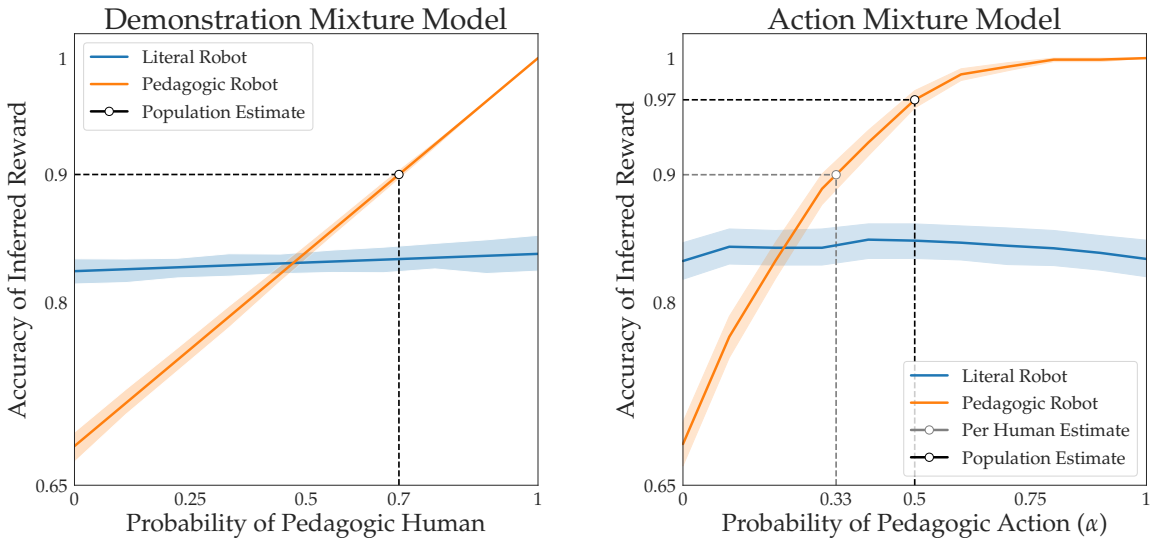

Figure 5.4: The performance of the literal and pedagogic robot when demonstrations are simulated according to the demonstration mixture model (left/a) or the action mixture model (right/b).

pedagogic **AH**, and how much the literal and pedagogic **R** are affected by demonstrations generated from the mixture models.

Demonstration mixture model

First, we test what we call the "demonstration mixture model". It is possible that the humans in the pedagogic condition from Ho et al. [2016] followed different strategies; some may have attempted to be pedagogic, but others may have simply been literal. If we look at which model, literal or pedagogic, is a better fit on a per-individual basis, we find that 89.7% of literal **AH** are better described by literal **H**, but in comparison, only 70.0% of pedagogic **AH** are better described by pedagogic **H**. If we simulate pedagogic humans as only following the model 70.0% of the time, then the accuracy of the pedagogic robot drops to 90% (Figure 5.4a). The literal robot is hardly impacted.

Action mixture model

We now test a more continuous version of the previous setting. We now model humans as acting according to a mixture policy H_m between the literal H_L and pedagogic H_P policies. In particular, at each time t the probability the human picks action a_t from state s in an

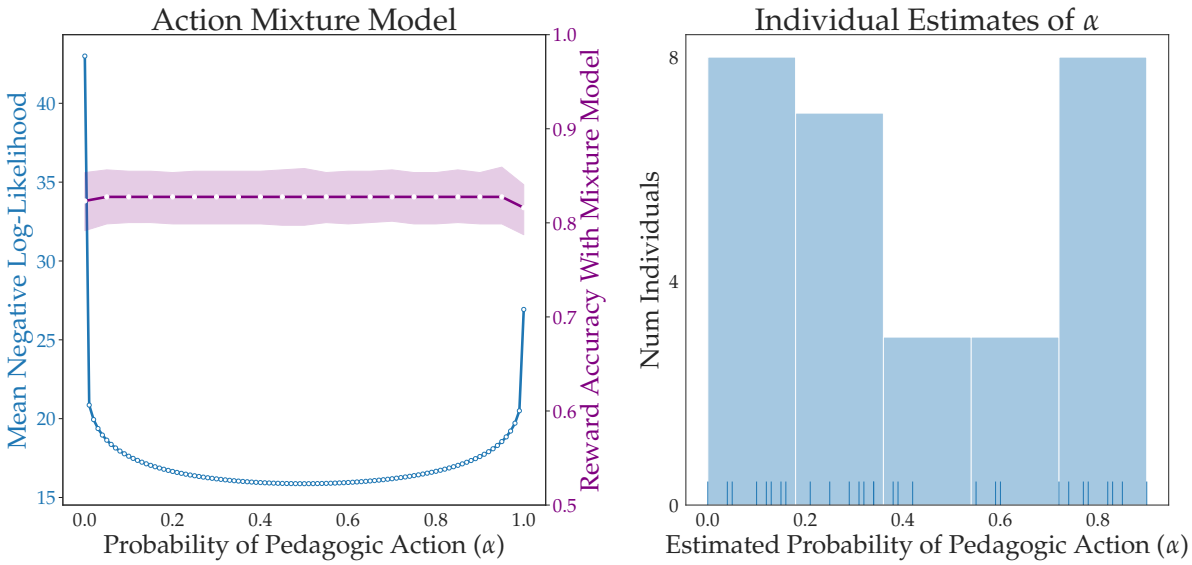

Figure 5.5: (left/a) In blue is the mean negative log-likelihood of the pedagogic human demonstrations under the action mixture model with probability α. All individuals are assumed to have the same value of α. Note that a mixture between the literal and pedagogic model is far better than either the literal model ($\alpha = 0$) or the pedagogic model ($\alpha = 1$). In purple is the accuracy of the robot's inferred reward, when given demonstrations from pedagogic **AH**, if it assumes the human acts according to the action mixture model. (right/b) The mixture probability α that maximized log-likelihood for each individual pedagogic **AH**.

environment with reward r is

$$H_M(a_t \mid s_t, r) = \\ \alpha H_P(a_t \mid s_t, r) + (1-\alpha) H_L(a_t \mid s_t, r)\,. \tag{5.10}$$

The parameter α is the probability of picking an action according to the pedagogic model. We plot the likelihood of the actual pedagogic human demonstrations as a function of α, (5.10). The best value of α over the whole population was $\alpha_P = 0.5$ (Figure 5.5a). But surprisingly, even a mixture with $\alpha = 0.01$ or $\alpha = 0.99$ is far better than either the literal ($\alpha = 0$) or pedagogic ($\alpha = 1$) model. In addition, the best estimates for α on an individual basis are somewhat bimodal (Figure 5.5b), indicating that there is high individual variation.

Figure 5.4b shows the performamce of the literal and pedagogic robot as α varies. At the best population level $\alpha_P = 0.5$, we find that the pedagogic robot gets 97% accuracy. However, if we simulate the pedagogic humans using the values of α estimated at an individual level, then the pedagogic robot's accuracy drops to 90%, highlighting the importance of individual variation. The literal robot again remains unaffected.

Discussion

In both the demonstration and action mixture models, we found that a mixture between pedagogic **H** and literal **H** was a much better fit to **AH**. In both cases, when pedagogic **H** is simulated according to the more accurate mixture model pedagogic **R**'s accuracy drops ten percentage points, but literal **R**'s accuracy hardly changes. Thus our results provide positive evidence for Hypothesis 5.3.1.

However, Hypothesis 5.3.1 does not explain the full story. The hypothesis can only account for a ten percentage drop in accuracy, but even with this drop, pedagogic **R** would still be better than literal **R**. Furthermore, with a cursory glance at Figure 5.4, one might be tempted to consider pedagogic **R** quite robust, as it remains high-performing for large ranges of α. However, as usual, the real problem is the *unknown unknowns*. Our empirical results imply that there are other ways that humans deviate from the model and that literal **R** is more robust than pedagogic **R** to these unknown deviations. Rather than robustness to α, the more compelling reason for choosing to use literal **R** is robustness to these unknown deviations.

5.4 Can we "fix" the pedagogic robot?

An intuitive idea for improving the performance of the pedagogic robot **R** is to give it a model of the pedagogic human **H** that is more predictive. Unfortunately, it is not so simple. For example, in Section 5.3, we showed that a mixture between the literal and pedagogic human model was a much better fit to actual pedagogic humans than either the literal or pedagogic human model. What if we had pedagogic **R** use the more accurate mixture model as its model of the pedagogic human? Unfortunately, as shown in Figure 5.4a (purple line), even if pedagogic **R** uses a better predictive model, i.e. the action mixture model, it does not perform any better.

In fact, it is possible for pedagogic **R** to do *worse* when given a *more* predictive model of the human. The reason is that a model that is better for *predicting* behavior (e.g. human demonstrations) is not necessarily better for *inferring* underlying, latent variables (e.g. the reward function), which is what is important for the robot. This may help explain why pedagogic **R** does worse than literal **R** with pedagogic **AH**, even though the pedagogic **H** model assumed by pedagogic **R** is more predictive of pedagogic **AH** than the literal **H** model assumed by literal **R**.

To illustrate, suppose there is a latent variable $\theta \in \Theta$ with prior distribution $p(\theta)$ and observed data $x \in \mathcal{X}$ generated by some distribution $p(x \mid \theta)$. In our setting, θ corresponds to the objective and x corresponds to the human input. For simplicity, we assume Θ and $\mathcal{X}$ are finite. We have access to a training dataset $\mathcal{D} = \{(\theta_i, x_i)\}_{i=1}^n$ of size n. A *predictive model* $m(x \mid \theta)$ models the conditional probability of the data x given latent variable θ for all $x \in \mathcal{X}, \theta \in \Theta$. In our case, the predictive model is the model of the human. The *predictive*

likelihood $\mathcal{L}_{\mathcal{X}}$ of a predictive model m is simply the likelihood of the data under the model:

$$\mathcal{L}_{\mathcal{X}}(m) = \prod_{i=1}^{n} m(x_i \mid \theta_i)\,. \tag{5.11}$$

The *inferential likelihood* is the likelihood of the latent variables after applying Bayes' rule:

$$\mathcal{L}_{\Theta}(m) = \prod_{i=1}^{n} \frac{m(x_i \mid \theta_i)p(\theta_i)}{\sum_{\theta} m(x_i \mid \theta)p(\theta)}\,. \tag{5.12}$$

Next, we show higher predictive likelihood does not necessarily imply higher inferential likelihood.

Claim 5.4.1 (Predictive vs inferential likelihood)**.** There exist settings in which there are two predictive models m_1, m_2 such that $\mathcal{L}_{\mathcal{X}}(m_1) > \mathcal{L}_{\mathcal{X}}(m_2)$, but $\mathcal{L}_{\Theta}(m_1) < \mathcal{L}_{\Theta}(m_2)$.

Proof. Suppose that $\Theta = \{\theta_1, \theta_2\}$ and $\mathcal{X} = \{x_1, x_2, x_3\}$, the prior $p(\theta)$ is uniform over Θ, and the dataset $\mathcal{D}$ contains the following $n = 9$ items:

$$\begin{aligned}\mathcal{D} = \{&(\theta_1, x_1), (\theta_1, x_1), (\theta_1, x_2), (\theta_2, x_2), (\theta_2, x_2),\\ &(\theta_2, x_3), (\theta_2, x_3), (\theta_2, x_3), (\theta_2, x_3)\}\,.\end{aligned}$$

Define the models $m_1(x \mid \theta)$ and $m_2(x \mid \theta)$ by the following conditional probabilities tables.

$m_1(x \mid \theta)$

	x_1	x_2	x_3
θ_1	2/3	1/3	0
θ_2	0	1/3	2/3

$m_2(x \mid \theta)$

	x_1	x_2	x_3
θ_1	2/3	1/3	0
θ_2	0	2/3	1/3

The model m_1 has predictive likelihood $\mathcal{L}_{\mathcal{X}}(m_1) = (2/3)^6(1/3)^3$ and inferential likelihood $\mathcal{L}_{\Theta}(m_1) = (1/2)^3$. The model m_2 has predictive likelihood $\mathcal{L}_{\mathcal{X}}(m_2) = (2/3)^4(1/3)^5$ and inferential likelihood $\mathcal{L}_{\Theta}(m_2) = (1/3)(2/3)^3$. Thus, $\mathcal{L}_{\mathcal{X}}(m_1) > \mathcal{L}_{\mathcal{X}}(m_2)$, but $\mathcal{L}_{\Theta}(m_1) < \mathcal{L}_{\Theta}(m_2)$. □

In our context, Claim 5.4.1 means that a human model that is better in terms of *prediction* may actually be worse for the robot to use for *inference*. An alternative approach could be to directly fit models that optimize for inferential likelihood. Unfortunately, this optimization becomes much trickier because of the normalization over the latent variable space Θ in the denominator of (5.12), see e.g., Dragan et al. [2013a].

Chapter 6

Learning from a human that may be tempted

In the real world, individuals often have to grapple with temptation and self-control. Imagine a dieter who hides their oreos in the back of their cabinet, but occasionally caves and eats them. If a robot assumes that the person is reward-rational, then it will interpret even their momentary lapses as being evidence of the target reward function. A particularly "helpful" robot might even move the oreos to the kitchen table to make it easier for the person.

Clearly, sometimes a system should satisfy a person's "reflective" preferences rather than their immediate, in-the-moment preferences. A weakness of the reward-rational choice formalism is that it does not distinguish between reflective and immediate preferences. In this chapter, we study decision-making when individuals have both kinds of preferences and investigate what kinds of feedback are sufficient to learn the individual's reflective preferences. Here, we find that it is important to account for the potential for temptation and to ask feedback at a time with human is reflective and can plan sophisticatedly [O'Donoghue and Rabin, 1999, Evans et al., 2016], i.e., anticipate their future temptation and account for it.

We focus on a notable economic theory from Gul and Pesendorfer [2001] that models individual decision-making under temptation and self-control. In the model, a decision-maker acts in two stages. In the first stage, the individual can anticipate temptation and restrict their choices for the second stage to a limited menu of items. In the second stage, the individual chooses from this menu, subject to temptation. Analogously, the social media user may block certain provocative accounts, so that she is not tempted to look at them later on.

To our knowledge, we are the first to study the Gul-Pesendorfer (GP) model from a learning-theoretic perspective. We focus on learning the individual's first-stage menu preferences—the individual's preferences before facing temptation. First, we show that the first-stage menu preferences cannot be identified from the items chosen by the individual in the second-stage. Given the infeasibility of learning from second-stage choices alone, we introduce a type of query at the first-stage: *menu comparisons*. In a menu comparison, we show the user two menus of options and ask which menu they prefer. We then give an

efficient algorithm to learn the first stage menu preferences over a set of n possible items using $O(n^2 \log n)$ comparisons between menus. Finally, we show how to reduce the number of comparisons to $O(n \log n)$ for approximate recovery.

6.1 Modeling individuals with temptation and commitment

In this section, we give an overview of the Gul-Pesendorfer model of decision-making under temptation.

A dieter facing temptation

First, we describe an illustrative example that will be helpful for intuition. Imagine a dieter who wishes to avoid eating dessert. The dieter is choosing a restaurant to go to for dinner. The two restaurants they are choosing between are part of a chain; the only difference between them is that one of them doesn't serve dessert. The dieter knows that if they choose the one with dessert, then unfortunately, once they are at the restaurant and can smell the mouth-watering chocolate lava cakes, then it will be difficult to resist ordering one. So, they opt to go to the restaurant without dessert.

The dieter's choice can be thought of as a two-stage decision problem. In the first stage, they pick the restaurant. In the second stage, they order a meal from the menu of the restaurant they picked in the first stage. Crucially, between the first and second stage, the dieter undergoes a shift in their ability to resist temptation. In the first stage, the dieter is at home, is thinking reflectively, and can act without temptation. In the second stage, the dieter is at the restaurant, is overwhelmed by the sights and smells of scrumptious pastries, and thus, has difficulty resisting temptation.

Our fictional dieter anticipates that they will face temptation in the second stage, so they chose to *commit* to the restaurant without dessert in the first stage. Commitment, i.e. up-front restriction of one's future choices, is a well-documented strategy for overcoming temptation. For example, studies have shown that commitment can be used to help people eat healthier [Schwartz et al., 2012, Wertenbroch, 1998], save more money [Ashraf et al., 2006, Beshears et al., 2015], avoid procrastination [Ariely and Wertenbroch, 2002], and reduce smoking [Giné et al., 2010].

The Gul-Pesendorfer model

Like our dieter, an individual in the GP model faces a two-stage decision problem. Let $\mathcal{Z} = \{z_1, \ldots, z_n\}$ be a set of items, Δ be the set of all lotteries (discrete probability distributions) over $\mathcal{Z}$, and $\mathcal{A}$ be the set of compact subsets of Δ.

1. In the first stage, the individual chooses a menu of lotteries $A \subseteq \mathcal{A}$.

2. In the second stage, the individual chooses a lottery $x \in A$.

Let's first consider how a "standard" decision-maker that does not face temptation would act. Assume that the decision-maker has preferences over the lotteries chosen in the second stage that can be represented with a utility function u, i.e. if $u(x) \geq u(y)$, then the decision-maker weakly prefers the lottery x to y. How would such a decision-maker choose menus at the first stage? If we assume that how good a menu is depends only upon how good the lottery picked from that menu is, then the decision-maker would simply rank menus according to the utility function

$$U' = \max_{x \in A} u(x)\,. \tag{6.1}$$

Any menu that contains the optimal lottery would be optimal. Notably, a menu that is always optimal for any utility function u is $A = \mathcal{A}$, the menu that contains all possible lotteries. So, the decision-maker never has a preference for commitment, i.e. they never have a preference to restrict their options at the first stage.

In contrast, Gul and Pesendorfer formulate an axiomatic definition of temptation (described later in this section) that is based on the existence of a preference for commitment. They show that an individual whose preferences satisfy these axioms have the following succinct utility representations. In the first stage, the individual's preference over menus can be represented with a ***menu utility*** U that has the form

$$U(A) = \max_{x \in A}[c(x) + t(x)] - \max_{x \in A} t(x)\,, \tag{6.2}$$

and the individual's preference over lotteries in the second stage can be represented by a ***consumption utility*** u that has the form

$$u(x) = c(x) + t(x)\,, \tag{6.3}$$

where c and t are von Neumann-Morgenstern utility functions, i.e. there exists u, v such that $c(x) = u^\intercal x$ and $t(x) = v^\intercal x$.

The menu utility U can be written as $U(A) = \max_{x \in A} u(x) - \max_{x \in A} t(x)$, which can be contrasted with the standard decision-maker with utility $U'(A) = \max_{x \in A} u(A)$. The difference between the two is the "penalty term" $\max_{x \in A} t(x)$. Since menus are penalized for containing lotteries with higher utility according to t, we refer to t as the ***temptation utility***. The presence of the penalty term means that, unlike the standard decision-maker, an individual in the GP model may have a preference for commitment.

The menu utility of a singleton set $\{x\}$ is equal to $c(x)$. Since c defines the utility of completely commiting to any item, we refer to c as the ***commitment utility***. The commitment utility can be thought of as the "true" utility of an item, absent temptation. In the second stage, the individual would ideally like to choose lotteries according to commitment utility c. But unfortunately, in the second stage, the individual faces temptation and the temptation utility t affects the individual's evaluation of lotteries; it acts like an additive noise term to the commitment utility c.

We now give more details on the axioms around temptation that these utility representations arise from. Let $\succeq^*$ be the individual's preference over the set of possible choices in both stages: $\mathcal{L} = \{(A, x) : x \in A\}$ where A is the menu chosen in the first stage and $x \in A$ is the lottery chosen in the second stage. The definition of temptation in the GP model is based upon whether the individual has a preference for commitment or not.

Definition 6.1.1. The item y is said to *tempt* the item x if the individual prefers to commit to x in the first stage, rather than have to choose x over y in the second stage: $(\{x\}, x) \succ^* (\{x, y\}, x)$. For example, we may have $(\{\texttt{salad}\}, \texttt{salad}) \succ^* (\{\texttt{salad}, \texttt{cake}\}, \texttt{salad})$.

Notably, under this definition, an item y can tempt an item x even if the individual can overcome temptation and choose x over y. In both of the outcomes we are comparing, i.e. $(\{x\}, x)$ and $(\{x, y\}, x)$, the individual ultimately chooses x in the second stage. We interpret a preference for $(\{x\}, x)$ over $(\{x, y\}, x)$ as an aversion to the costly self-control that would be necessary to choose x over y. For example, eating a salad is less enjoyable when one has to choose it over a cake.

In addition, the following three axioms about temptation are assumed:

(T1) if x tempts y, y does not tempt x;

(T2) unless an option y tempts every option in a menu A, adding y to A does not make A worse;

(T3) $(A, x) \succeq^* (B, x)$ if $A \subset B$, i.e. eliminating temptations cannot make the individual worse.

The extended preference $\succeq^*$ induces a preference over menus $\succeq$ where $\succeq$ is defined so that $A \succeq B$ if and only if there exists $x \in A$ such that $(A, x) \succeq (B, y)$ for all $y \in B$. Gul and Pesendorfer prove that the individual's menu preference $\succeq$ has a utility representation as in (6.2).

Theorem 6.1.1 (Gul and Pesendorfer [2001], Theorem 1 and 5)**.** *If the extended preference $\succeq^*$ satisfies the axioms T1-3 and is upper semi-continuous*[1]*, and the induced menu preference $\succeq$ is continuous and independent*[2] *then the menu preference $\succeq$ has a utility representation of the form*

$$U(A) = \max_{x \in A}[c(x) + t(x)] - \max_{x \in A} t(x)\,, \tag{6.4}$$

where c and t are von Neumann-Morgenstern utility functions.

[1]The preference $\succeq^*$ is upper semi-continuous if the upper-contour sets are closed, i.e. the sets $\{(A, x) : (A, x) \succeq^* (B, y)\}$ are closed for all (B, y).

[2]The preference $\succeq$ is continuous if its upper and lower contour sets are closed, i.e. the sets $\{B : B \succeq A\}$ and $\{B : A \succeq B\}$ are closed for all A. The preference $\succeq$ is independent if $A \succ B$ and $\alpha \in (0, 1)$ implies $\alpha A + (1 - \alpha)C \succ \alpha B + (1 - \alpha)C$.

Building on Theorem 6.1.1, we define a *Gul-Pesendorfer menu preference* as the following.

Definition 6.1.2 (GP menu preference)**.** If a preference $\succeq$ over menus has a utility representation of the form in (6.4), then we call the preference a Gul-Pesendorfer (GP) menu preference and say that it is represented by (c, t).

At the second stage, the individual is assumed to maximize the conditional preference, i.e. given a menu A, the individual chooses $x \in A$ such that $(A, x) \succeq (A, y)$ for all $y \in A$. Gul and Pesendorfer prove that, if the menu preference is represented by (c, t), then the individual chooses an option x^* from the menu A to maximize the *consumption utility* $u \coloneqq c + t$.

Theorem 6.1.2 ([Gul and Pesendorfer, 2001], Theorem 7)**.** *Suppose the extended preference $\succeq^*$ satisfies T1-3, is upper semi-continuous, and is minimally congruent*[3]*. Suppose also that the induced menu preference $\succeq$ is a GP menu preference represented by (c, t) and is regular*[4]*. Then, in the second stage, the individual picks a lottery x^* out of the available menu A such that*

$$x^* \in \arg\max_{x \in A} u(x) = \arg\max_{x \in A} [c(x) + t(x)] \,. \tag{6.5}$$

Notation for learning GP menu preferences

In the next two sections, we analyze learning of an individual's first-stage menu preference $\succeq$. We focus on learning the first-stage menu preference because it is the individual's preference before they are subject to temptation.

We are interested in a slight modification of the setting studied by Gul and Pesendorfer. In particular, we are interested in the case where the individual selects items in the second stage, rather than lotteries. In other words, we restrict the lotteries considered over $\mathcal{Z}$ to the degenerate lotteries. So for convenience, we write the utilities c, t, u as functions directly from $\mathcal{Z}$ to $\mathbb{R}$. Furthermore, we define the commitment preference $\succeq_c$, temptation preference $\succeq_t$, and consumption preference $\succeq_u$ as the preferences with utility representations c, t, u. The commitment, temptation, and consumption preferences all operate directly over the items $\mathcal{Z}$. Similarly, the menu preferences $\succeq$ then operate over the set $2^{\mathcal{Z}}/\emptyset$.

6.2 Consumption choices are insufficient for learning menu preferences

In accordance with the principle of revealed preference, consumption choices are typically taken to be ground-truth for what an individual prefers. A line of theoretical work already

[3]The extended preference $\succeq^*$ is minimally congruent if $A \succeq \{y\}$ for all A implies $(\{x, y\}, x) \succeq^* (\{x\}, x)$ for all $x \in \Delta$. Minimal congruence requires that a lottery that is ranked lowest by commitment utility is not tempting.

[4]A GP menu preference represented by (c, t) is regular if neither c nor t is constant and neither are affine transformations of each other.

analyzes the use of consumption choices, dubbed *revealed preference queries*, for learning the individual's preference over items, i.e. their consumption preference [Beigman and Vohra, 2006, Zadimoghaddam and Roth, 2012, Balcan et al., 2014, Kallus and Udell, 2016, Roth et al., 2016].

Definition 6.2.1 (Revealed preference query)**.** A revealed preference query is a query to an individual's choice function[5] $C(A) = \arg\max_{x \in A} u(x)$ which takes in a menu A and returns the item they choose to consume from the menu A.

From a practical standpoint, it is desirable to learn preferences from consumption choices because such choices are readily available, e.g. the deluges of data recommender platforms have on what content users consume. We show, however, that revealed preference queries are insufficient for learning GP menu preferences.

Insufficiency of standardized preferences

First, we analyze the limitations of a natural way of constructing menu preferences from consumption choices. There is already a body of prior work that uses revealed preference queries to learn the individual's preference over items, i.e. the individual's consumption preference $\succeq_u$ [Beigman and Vohra, 2006, Zadimoghaddam and Roth, 2012, Balcan et al., 2014, Kallus and Udell, 2016, Roth et al., 2016]. Once $\succeq_u$ is recovered, so is the *standardized preference*: the menu preference that a "standard" decision-maker that does not face temptation would have. In particular, the standardized preference $\succeq'$ of a GP menu preference $\succeq$ with representation (c, t) is the preference $\succeq'$ with utility representation $U'(A) = \max_{x \in A} u(x) = \max_{x \in A} c(x) + t(x)$ (see Equation 6.1).

Using the standardized preference $\succeq'$ is intuitively appealing and requires nothing beyond inferring the consumption preference $\succeq_c$. However, we prove that a GP menu preference equals its standardized preference only for a subset of GP menu preferences that have no *preference for commitment*, defined below. Furthermore, we show empirically that as the true preference exhibits a greater preference for commitment, the standardized preference becomes a worse approximation.

Definition 6.2.2 (Preference for commitment)**.** The menu preference $\succeq$ has a *preference for commitment at the menu* A if there exists $B \subset A$ such that $B \succ A$. The preference $\succeq$ has a *preference for commitment* if $\succeq$ has a preference for commitment at some menu. Furthermore, we say that the preference $\succeq_1$ has a *greater preference for commitment* than $\succeq_2$, if for every menu A where $\succeq_2$ has a preference for commitment, $\succeq_1$ also has a preference for commitment.

The following lemma from [Gul and Pesendorfer, 2001] describes the conditions under which a GP menu preference has no preference for commitment.

[5] We assume ties in the individual's choice function are broken arbitrarily.

Lemma 6.2.1 ([Gul and Pesendorfer, 2001])**.** If $\succeq$ is a GP menu preference represented by (c, t), then it has no preference for commitment if and only if either c is constant, t is constant, or c is a positive affine transformation of t [Gul and Pesendorfer, 2001].

With this lemma in hand, we prove that a GP menu preference $\succeq$ equals its standardized preference $\succeq'$ only if $\succeq$ lies within a subset of GP menu preferences that have no preference for commitment.

Theorem 6.2.1. *A GP menu preference $\succeq$ is equal to its standardized preference $\succeq'$ if and only $\succeq$ has a representation (c, t) such that either t is constant or c is a positive affine transformation of t.*

Proof. First, we show that if t is constant or c is a positive affine transformation of t, then the standardized and true preference are equal. If t is constant, i.e. $t(x) = \beta$ for all x and some constant $\beta \in \mathbb{R}$, then the preference $\succeq$ can be represented by $U(A) = \max_{x \in A} c(x)$, and thus, can also be represented by $U'(A) = \max_{x \in A} c(x) + t(x) = \max_{x \in A} c(x) + \beta$. If c is a positive affine transformation of t, i.e. $c = \alpha t + \beta$ for $\alpha > 0$ and some $\beta \in \mathbb{R}$, then the preference $\succeq$ can be represented by $U(A) = \max_{x \in A}(1 + \alpha)t(x) + \beta - \max_{x \in A} t(x) = \alpha \max_{x \in A} t(x) + \beta$, and thus, can also be represented by $U'(A) = \max_{x \in A} c(x) + t(x) = (1 + \alpha) \max_{x \in A} t(x)$.

Now we show that if $\succeq$ does not have a representation (c, t) such that t is constant or c is a positive affine transformation of t, then the true and standardized preference are not equal. By Lemma 6.2.1, $\succeq$ either has a representation such that c is constant or has a preference for commitment. If the true preference $\succeq$ has a preference for commitment, then clearly the standardized preference $\succeq'$ cannot be equal to it because the standardized preference cannot have a preference for commitment. If c is constant, then $U(A) = 0$ is a utility representation for $\succeq$ and $U'(A) = \max_{x \in A} t(x)$ is a utility representation for $\succeq'$. However, by assumption t is not constant, and therefore, $\succeq$ and $\succeq'$ cannot be equal. □

Next, we show that as the true preference exhibits a greater preference for commitment, the standardized preference becomes a worse approximation. We make use of a representation theorem from Gul and Pesendorfer. They prove that if $\succeq_1$ and $\succeq_2$ are two GP menu preferences with representation (c_1, t_1) and (c_2, t_2), respectively, and if c_1 and t_1 are convex combinations of c_2 and t_2, then $\succeq_1$ has a greater preference for commitment than $\succeq_2$.

Lemma 6.2.2 ([Gul and Pesendorfer, 2001], Theorem 8)**.** Let $\succeq_1, \succeq_2$ be two GP menu preferences that are regular[6] and let (c_1, t_t) be a representation of $\succeq_1$. Then $\succeq_1$ has a greater preference for commitment than $\succeq_2$ if and only if there exists c_2, t_2 such that $(c_2, \gamma t_2)$ represents $\succeq_2$ and

$$c_2 = \alpha c_1 + (1 - \alpha)t_1\,, \qquad\qquad t_2 = \beta c_1 + (1 - \beta)t_1$$

for some $\alpha, \beta \in [0, 1]$ and some $\gamma > 0$.

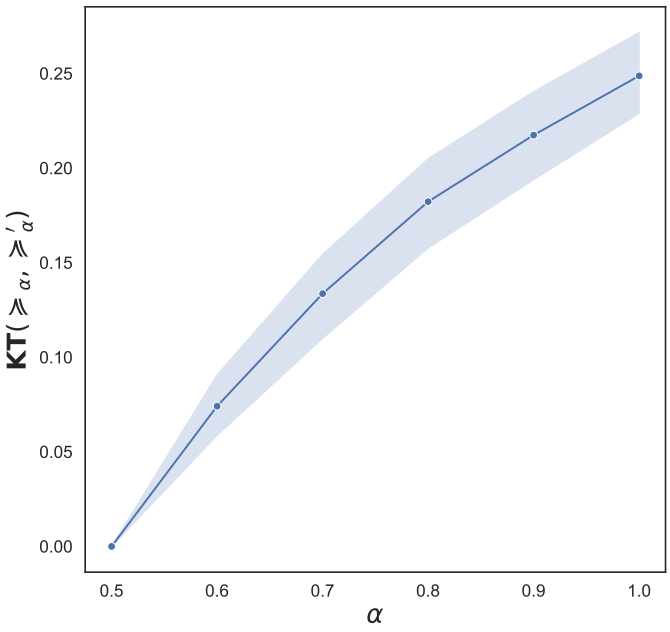

Figure 6.1: As the true preference exhibits a greater preference for commitment, the standardized preference becomes a worse approximation. The true preference $\succeq_\alpha$ is defined by the representation (c_α, t_α) where $c_\alpha = \alpha c + (1-\alpha)t$ and $t_\alpha = \alpha t + (1-\alpha)c$. We generate c and t as random Gaussian noise, i.e. $c(x_i), t(x_i) \sim \mathcal{N}(0,1)$ for all items $x_1, \ldots x_n$ (here $n = 10$). As α increases, $\succeq_\alpha$ has a greater preference for commitment, and the Kendall-Tau distance between $\succeq_\alpha$ its standardized preference $\succeq'_\alpha$ increases.

Now, suppose we have a consumption utility c and a temptation utility t. For $\alpha \in [0.5, 1]$, define the preference $\succeq_\alpha$ as the GP menu preference with representation (c_α, t_α) where

$$c_\alpha = \alpha c + (1-\alpha)t\,,\; t_\alpha = \alpha t + (1-\alpha)c\,, \tag{6.6}$$

As α increases, the preference $\succeq_\alpha$ has a greater preference for commitment. Thus, by varying α, we can measure how well the standardized preference approximates the true preference as its preference for commitment increases or decreases. We measure how close the standardized preference $\succeq'_\alpha$ is to the true preference $\succeq_\alpha$ by the Kendall-Tau distance (Definition 6.2.3). Figure 6.1 shows that as α increases, the Kendall-Tau distance between $\succeq'_\alpha$ and $\succeq_\alpha$ increases. In other words, the standardized preference becomes a worse approximation as the true preference exhibits a greater preference for commitment.

[6] A GP menu preference represented by (c, t) is regular if neither c nor t is constant and neither are affine transformations of each other.

Definition 6.2.3 (Kendall-Tau distance)**.** Let $\succeq_1$ and $\succeq_2$ be preferences over k elements. The (normalized) Kendall-Tau distance between $\succeq_1$ and $\succeq_2$ is the portion of pairwise comparisons that are inconsistent between the two preferences:

$$\mathbf{KT}(\succeq_1, \succeq_2) = \binom{k}{2}^{-1} \left(\sum_{(x,y)\,:\,x \succ_1 y} 1\{y \succeq_2 x\} + \frac{1}{2} \sum_{(x,y)\,:\,x \sim_1 y} 1\{y \succ_2 x \text{ or } x \succ_2 y\} \right).$$

Insufficiency of any algorithm

We now prove that no algorithm that learns from revealed preference queries can learn the individual's true menu preferences. Below, we formally define the problem of learning GP menu preferences from revealed preference queries.

Definition 6.2.4 (Learning menu preferences from revealed preference queries)**.** An algorithm $\mathcal{A}$ is said to learn the GP menu preferences from $m = m(n)$ revealed preference queries, if for any possible GP menu preference $\succeq$ over the set of items $\mathcal{Z}$, if the learning algorithm can query the choice function C, then after at most m queries the algorithm outputs $\succeq$.

This is simply because second-stage behavior is defined entirely by the consumption preference $\succeq_u$, but the consumption preference does not identify the menu preference. In fact, our result is stronger than non-identification: after any set of revealed preference queries, there exist two menu preferences with exactly the opposite ranking of singleton sets, i.e. with exactly the opposite commitment preferences. What this means is revealed preference queries reveal essentially nothing about the menu preferences.

Theorem 6.2.2. *Let $\succeq$ be a GP menu preference represented by (c, t) and let C be its associated choice function. Let $D = \{(R_i, C(R_i))\}_{i=1}^m$ be a dataset of revealed preference queries. There exists a GP menu preference $\succeq'$ which is consistent with D and for which $\succeq$ and $\succeq'$ have exactly the opposite ranking of singleton sets. Therefore, no algorithm can learn GP menu preferences from revealed preference queries.*

Proof. Let $x_1^*, \ldots, x_n^*$ be the items as ranked by commitment utility c, i.e. x_1^* has the highest commitment utility and x_n^* has the lowest commitment utility. Construct the GP menu preference $\succeq'$ represented by (c', t') where c' and t' are defined as $c'(x_i^*) = -i$ and $t'(x_i^*) = c(x_i^*) + t(x_i^*) + i$. Clearly the consumption utility of $\succeq$ and $\succeq'$ are equal: $c' + t' = c + t$. Therefore, $\succeq'$ and $\succeq$ induce the same choice function, which means that $\succeq'$ is consistent with the dataset D. However, c' has the opposite ranking of items as c, and therefore $\succeq'$ has the opposite ranking of singleton sets as $\succeq$. No algorithm that learns from revealed preference queries can distinguish between $\succeq$ and $\succeq'$, and thus, no algorithm can learn menu preferences from revealed preference queries alone. □

6.3 Learning menu preferences from menu comparisons

In the previous section, we showed that second-stage consumption choices are insufficient for recovering menu preferences, which now motivates us to consider queries at the first-stage. In particular, we consider learning menu preferences from *menu comparisons*. A menu comparison is a query that asks which of two menus A or B is preferred and returns either $A \succ B$, $A \sim B$, or $A \prec B$.[7] We first consider exact recovery and show that the menu preferences can be recovered in $O(n^2 \log n)$ menu comparisons. Then, we show how to reduce the number of comparisons to $O(n \log n)$ for approximate recovery.

In both exact and approximate recovery, we restrict ourselves to considering *strict GP menu preferences*.

Definition 6.3.1. A GP menu preference is *strict* if its associated preferences over items; $\succeq_c$, $\succeq_t$, and $\succeq_u$; are strict preferences; i.e. for all items $x, y \in \mathcal{X}$, either $x \succ_z y$ or $y \succ_z x$ for $z \in \{c, t, u\}$.

Exact recovery from menu comparisons

Below, we define the problem of exactly recovering menu preferences from menu comparisons.

Definition 6.3.2 (Learning menu preferences exactly)**.** An algorithm $\mathcal{A}$ is said to exactly learn menu preferences from $m = m(n)$ menu comparisons, if for any strict GP menu preference $\succeq$ over the set of items $\mathcal{X}$, after at most m menu comparison queries, it outputs the menu preference $\succeq$.

Since there are $O(2^n)$ possible menus, recovering the menu preferences with a standard comparison-based sorting algorithm would require $O(2^n \log 2^n) = O(n2^n)$ menu comparisons. Instead, we give a simple algorithm that can learn menu preferences using only $O(n^2 \log n)$ menu comparisons. The algorithm has two steps. In the first step, we partition the menus into $O(n^2)$ equivalence classes. The individual is indifferent between all menus that lie within the same equivalence class. In the second step, we sort the $O(n^2)$ equivalence classes, which can be done in $O(n^2 \log n)$ queries through a standard comparison-based sorting algorithm. Thus, the overall complexity of our algorithm is $O(n^2 \log n)$. Pseudocode for the algorithm is given in Algorithm 2.

First, we show that for any GP menu preference, the menus can be partitioned $O(n^2)$ equivalence classes. For any item $x \in \mathcal{X}$, define its *consumption rank* $r_u(x)$ and *temptation rank* $r_t(x)$ as its rank according to consumption utility u and to temptation utility t, e.g. the item with highest temptation utility has temptation rank equal to one. Then, for a menu A, define its consumption rank $R_u(A)$ and temptation rank $R_t(A)$ as the highest

[7] We could also consider just returning $A \succeq B$ or $A \precsim B$ and none of our results are affected.

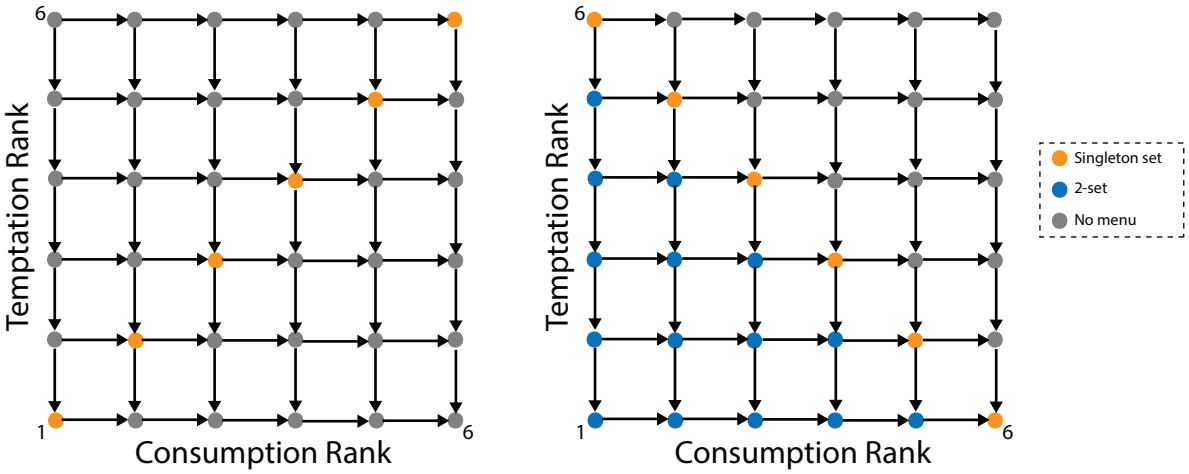

Figure 6.2: The possible rank representations under two different GP menu preferences for $n = 6$. A directed path from a vertex (a_1, b_1) to another vertex (a_2, b_2) indicates that $(a_1, b_1) \succ (a_2, b_2)$. The preference on the left achieves Lemma 6.3.2's lower bound of $n = 6$ rank representations, and the preference on the right achieves the upper bound of $\frac{n(n+1)}{2} = 21$ rank representations. The orange points represent equivalence classes that include a singleton set (a menu with only one item), the blue points are equivalence classes that include two-sets (a menu with two items) but not a singleton set, and the gray points are rank representations that no menu has. The rank representations of the singleton sets fix which of the other rank representations are possible. It is not possible to have any rank representations that are directly above or to the right of an orange point. For example, consider the orange point $(2, 4)$ in Figure 1(b). Let $\{y\}$ be the singleton set with rank representation $(2, 4)$. It is not possible to have $(x, 4)$ for $x > 2$ because any menu A with temptation rank equal to four must include y, which has consumption rank equal to two. Since the consumption rank of menu A is the minimum consumption rank of the items it contains, its consumption rank must be less than or equal to two. All rank representations that are not directly above or to the right of an orange point are possible and can be attained by menus with two items.

consumption rank and temptation rank of its items: $R_u(A) = \min_{x \in A} r_u(A)$ and $R_t(A) = \min_{x \in A} r_t(A)$. Together, we call the consumption rank and temptation rank of a menu its *rank representation*.

Definition 6.3.3 (Rank representation)**.** The *rank representation* of a menu A is the tuple of its consumption and temptation rank: $(R_u(A), R_t(A))$. If the rank representation of a menu A is (a, b), we write $A \equiv (a, b)$.

Recall that any GP menu preference has a utility representation of the form $U(A) = \max_{x \in A} u(x) - \max_{x \in A} t(x)$. The utility of a menu A is therefore defined by its consumption and temptation rank. Thus, The individual is indifferent between any two menus with the same rank representation:

Lemma 6.3.1 (Equivalence classes for menus)**.** Let $\succeq$ be a GP menu preference. If two menus A and B have the same rank representation under $\succeq$, then the individual is indifferent between the two menus: $A \sim B$.

Proof. The menu preference $\succeq$ has a utility representation $U(S) = \max_{x \in S} u(x) - \max_{x \in S} t(x)$. Since the menus A and B have the same consumption and temptation rank, $\max_{x \in A} u(x) = \max_{x \in B} u(x)$ and $\max_{x \in A} t(x) = \max_{x \in B} t(x)$, thus $U(A) = U(B)$ and $A \sim B$. □

Suppose two menus have the rank representations (a_1, b_1) and (a_2, b_2). If $a_1 \leq a_2$ and $b_1 \leq b_2$ where at least one of the inequalities is strict, then we must have $(a_1, b_1) \succ (a_2, b_2)$ because (a_1, b_1) has higher consumption utility and lower temptation utility than (a_2, b_2). Figure 6.2 visualizes this structure between rank representation as a lattice. The rank representations are represented as vertices where a directed path from a vertex (a_1, b_1) to (a_2, b_2) means that $(a_1, b_1) \succ (a_2, b_2)$.

The next lemma states that for any GP menu preference, the number of unique rank representations is between $\Omega(n)$ and $O(n^2)$. Figures 6.2a and 6.2b show examples of two GP menu preferences in which the lower bound and upper bound are achieved, respectively.

Lemma 6.3.2. For any strict GP menu preference $\succeq$, the number of unique rank representations $N(\succeq)$ is such that $n \leq N(\succeq) \leq \frac{n(n+1)}{2}$. The lower and upper bound are tight, i.e there exists strict GP menu preferences $\succeq$ and $\succeq'$ such that $N(\succeq) = n$ and $N(\succeq') = \frac{n(n+1)}{2}$.

Proof. First, we consider the upper bound. Since the utility of a menu A only depends on the most consumable item $x_u^*(A) = \arg\max_{x \in A} u(x)$ and the most tempting item $x_t^*(A) = \arg\max_{x \in A} t(x)$, it has the same utility as the menu $B = \{x_u^*(A), x_t^*(A)\}$. Therefore, for every menu A, there exists a menu B such that $A \sim B$ and $|B| = 1$ or $|B| = 2$ (we have $|B| = 2$ if $x_u^*(A) \neq x_t^*(A)$ and $|B| = 1$ otherwise). Thus, an upper bound on the number of unique rank representations is the number of unique menus of size one or two: $n + \binom{n}{2} = \frac{n(n+1)}{2}$. For the lower bound, since the GP menu preference is strict, each item has a unique consumption and temptation rank, and therefore each singleton set has a unique rank representation. There are n singleton sets, so we have $N(\succeq) \geq n$. Figure 6.2 has two examples that achieve the upper and lower bound. □

	$x \,?_c\, y$	$x \,?_u\, y$	$x \,?_t\, y$	$U(\{x,y\})$	$\{x\} \,?_U\, \{x,y\} \,?_U\, \{y\}$
1	$x \succ_c y$	$x \succ_u y$	$x \succ_t y$	$u(x) - t(x) = c(x)$	$\{x\} \sim \{x,y\} \succ \{y\}$
2	$x \prec_c y$	$x \succ_u y$	$x \succ_t y$	$u(x) - t(x) = c(x)$	$\{x\} \sim \{x,y\} \prec \{y\}$
3	$x \succ_c y$	$x \prec_u y$	$x \prec_t y$	$u(y) - t(y) = c(y)$	$\{x\} \succ \{x,y\} \sim \{y\}$
4	$x \prec_c y$	$x \prec_u y$	$x \prec_t y$	$u(y) - t(y) = c(y)$	$\{x\} \prec \{x,y\} \sim \{y\}$
5	$x \succ_c y$	$x \succ_u y$	$x \prec_t y$	$u(x) - t(y)$	$\{x\} \succ \{x,y\} \succ \{y\}$
6	$x \prec_c y$	$x \prec_u y$	$x \succ_t y$	$u(y) - t(x)$	$\{x\} \prec \{x,y\} \prec \{y\}$

Table 6.1: The ordering of $\{x\}, \{x,y\}$, and $\{y\}$ is uniquely determined by the item comparisons between x and y.

Determining the rank representation of each menu is equivalent to recovering the consumption and temptation preferences $\succ_u$ and $\succ_t$. If we had access to consumption comparisons $x \,?_u\, y$ (a revealed preference query of size two) and temptation comparisons $x \,?_t\, y$ ("which is more tempting?"), then to recover $\succ_u$ and $\succ_t$ we could simply apply a comparison-based sorting algorithm with consumption comparisons and temptation comparisons, respectively.

It turns out we can simulate consumption and temptation comparisons with menu comparisons. Since the utility of a menu depends only on the item with highest consumption rank and the item with highest temptation rank, menu comparisons give us information about the underlying consumption and temptation preferences. For example, if we observe that $A \not\sim A \cup \{x\}$, then we know the item x is more desirable for consumption than any $y \in A$ or x is more tempting than any $y \in A$. Through menu comparisons between $\{x\}$, $\{y\}$, $\{x,y\}$, we can determine the consumption comparison $x \,?_u\, y$ and the temptation comparison $x \,?_t\, y$. Then, using consumption and temptation comparisons, we can recover the rank representations. Algorithm 1 details the procedure. The following lemma proves that Algorithm 1 returns the rank representations in $O(n \log n)$ menu comparisons.

Lemma 6.3.3. Under any strict GP menu preference $\succeq$, Algorithm 1 recovers the rank representations of all menus in $O(n \log n)$ menu comparisons.

Proof. The function `consumption-temptation-comparison` takes in two items x and y and uses menu comparisons (denoted by $?_U$) to simulate a consumption comparison $x \,?_u\, y$ and a temptation comparison $x \,?_t\, y$. In particular, there is a 1-1 relationship between the individual's ordering of the menus $\{x\}, \{y\}$, and $\{x,y\}$ and the individual's ordering of the items x and y according to the preferences $\succ_c, \succ_u$, and $\succ_t$. The function uses menu comparisons to determine the ordering of the menus $\{x\}, \{y\}$, and $\{x,y\}$, and by doing so, also determines the answer to the consumption comparison $x \,?_u\, y$ and temptation comparison $x \,?_t\, y$.

Table 6.1 lists, for each possible value for $x \,?_c\, y$, $x \,?_u\, y$, $x \,?_t\, y$, the ordering of $\{x\}, \{y\}$, and $\{x,y\}$. There are six rows total, rather than the $2^3 = 8$ one might expect, because when

Algorithm 1: Recovering the equivalence classes of menus

Using queries to `consumption-temptation-comparison`, sort $x_1, \ldots x_n$ to get $\succ_u$ and $\succ_t$

Function `consumption-temptation-comparison`(x, y)**:**

- Query menu comparison between $\{x\}$ and $\{x, y\}$
- Query menu comparison between $\{y\}$ and $\{x, y\}$
- **if** $\{x, y\} \sim \{x\}$ **then**
  - **return** $x \succ_u y$ and $x \succ_t y$
- **else if** $\{x, y\} \sim \{y\}$ **then**
  - **return** $x \prec_u y$ and $x \prec_t y$
- **else if** $\{x\} \succ \{x, y\} \succ \{y\}$ **then**
  - **return** $x \succ_u y$ and $x \prec_t y$;
- **else if** $\{y\} \succ \{x, y\} \succ \{x\}$ **then**
  - **return** $x \prec_u y$ and $x \succ_t y$

Algorithm 2: Recovering menu preferences exactly

Run Algorithm 1 to get the rank representation of each menu.
Sort all unique rank representations to get $\succeq$

$x \succ_u y$ and $y \succ_t x$ or when $y \succ_u x$ and $x \succ_t y$, then the value of the commitment comparison is determined as $x \succ_c y$ or $y \succ_c x$, respectively.

We now prove, for each row, that the values of the comparisons determine the corresponding ordering of $\{x\}$, $\{x, y\}$, and $\{y\}$. In rows 1-2, the menu utility of $\{x, y\}$ (which is determined by $\succ_t$ and $\succ_u$) is equal to the menu utility of $\{x\}$, indicating that $\{x, y\} \sim \{x\}$. The commitment comparison $x \,?_c\, y$ is equivalent to the menu comparison $\{x\} \,?_U\, \{y\}$. Therefore, we have that $\{x\} \sim \{y\} \,?_U\, \{x, y\}$ where the value of $?_U$ is determined by the commitment comparison $x \,?_c\, y$. A similar analysis holds for rows 3-4.

For row 5, note that

$$\begin{aligned} U(\{y\}) &= \max_{z \in \{y\}} u(z) - \max_{z \in \{y\}} t(z) \\ &< \max_{z \in \{x,y\}} u(z) - \max_{z \in \{y\}} t(z) \\ &= \max_{z \in \{x,y\}} u(z) - \max_{z \in \{x,y\}} t(z) = U(\{x, y\}) \\ &< \max_{z \in \{x\}} u(z) - \max_{z \in \{x\}} t(z) = U(\{x\}), \end{aligned}$$

and so, $\{x\} \succ \{x, y\} \succ \{y\}$. An analogous analysis holds for row 6. Thus, there is a 1-1 relationship between the item comparisons and the ordering of the menus $\{x\}$, $\{x, y\}$, and $\{y\}$. The function `consumption-temptation-comparison` determines the ordering of the menus and then infers the item comparisons according to Table 6.1. The function uses only two menu comparisons to determine the temptation and consumption comparisons. Therefore,

a standard comparison-based sorting algorithm that uses temptation and consumption comparisons (as simulated by `consumption-temptation-comparison`) takes $O(n \log n)$ menu comparisons to determine the consumption and temptation preferences, which then determine the rank representation of each menu. □

After recovering the rank representations, in the second step, we simply sort them. As stated in Lemma 6.3.2, the number of unique rank representations ranges from n to $\frac{n(n+1)}{2}$. Thus in the best-case, the algorithm take $O(n \log n)$ comparisons, while in the worst-case, it takes $O(n^2 \log n)$ comparisons. We prove this in the following theorem.

Theorem 6.3.1. *Algorithm 2 learns menu preferences in $O(n^2 \log n)$ menu comparisons. In the best-case, Algorithm 2 takes $O(n \log n)$ menu comparisons.*

Proof. By Lemma 6.3.3, Algorithm 1 recovers the rank representations in $O(n \log n)$ menu comparisons. By Lemma 6.3.2, every strict GP menu preference will have between n and $\frac{n(n+1)}{2}$ rank representations. Therefore, a standard sorting algorithm will sort the rank representations in $O(n \log n)$ menu comparisons in the best-case and $O(n^2 \log n)$ menu comparisons in the worst-case. □

Approximate recovery from menu comparisons

Next, we turn to approximate recovery: finding preferences that are close to the true GP menu preferences. We measure "close" using the Kendall-Tau distance (Definition 6.2.3). Our approach reduces the problem of approximate recovery of GP menu preferences to the standard problem of learning a linear classifier. Thus, any black-box algorithm for learning a linear classifier can be used to approximately learn GP menu preferences in $O(n \log n)$ menu comparisons.

The algorithm is broken into the same two steps as the algorithm for exact recovery (Algorithm 2): (1) find the rank representation of each menu (2) sort the rank representations. The first step is exactly the same: we use Algorithm 1 to find the rank representations in $O(n \log n)$ menu comparisons. In the second step, rather than exactly sorting the rank representations like in Algorithm 2, we reduce the problem of sorting rank representations to the problem of learning a linear classifier, which allows us to approximately recover the ordering of rank representations in $O(n)$ comparisons. Thus, overall, our algorithm will take $O(n \log n)$ menu comparisons.

We now describe the reduction to linear classification. After finding the rank representation of a menu A, it can be represented as a binary vector $v(A) \in \{0,1\}^{2n}$ with ones at the positions of the consumption and temptation rank:

$$v(A)_i = \begin{cases} 1 & i = R_u(A) \text{ or } i = R_t(A), \\ 0 & \text{otherwise} \end{cases}.$$

The utility of each menu can now be written as $U(A) = v(A)^\intercal \beta$ where the vector $\beta \in \mathbb{R}^{2n}$ can be defined as

$$\beta_i = \begin{cases} u(x_i^u) & 1 \leq i \leq n \\ t(x_i^t) & n+1 \leq i \leq 2n \end{cases}$$

where x_i^u is the i-th ranked item according to consumption utility and x_i^t is the i-th ranked item according to temptation utility.

A menu comparison between two menus A and B with different rank representations can be represented as a vector $v(A) - v(B)$ with an associated label y where $y = 1$ if $A \succ B$ and $y = -1$ if $B \succ A$. Or equivalently, where $y = 1$ if $(v(A) - v(B))^\intercal \beta$ is positive. Thus, we can reduce the problem of determining the ordering of rank representations to the problem of determining a weight vector $\hat{\beta}$ that classifies all pairwise comparisons correctly.

We query comparisons by sampling pairs of menus uniformly at random from the $\binom{2^n-1}{2}$ possible pairs. Then, we use a black-box linear classification algorithm $\mathcal{A}$ to determine the approximate ranking. Any algorithm which *PAC-learns* the class of linear separators will suffice.

Definition 6.3.4 (PAC-learning linear separators)**.** An algorithm $\mathcal{A}$ PAC-learns the class of linear separators if there exists a function $m_{\mathcal{A}} : [0,1]^2 \times \mathbb{N} \to \mathbb{N}$ if for any $\epsilon, \delta \in (0,1)$, for every distribution $\mathcal{D}$ over $\mathcal{X} = \mathbb{R}^d$, and for every linear function $c : \mathcal{X} \to \{0,1\}$, if after receiving $m_{\mathcal{A}}(\epsilon, \delta; d)$ i.i.d examples generated by $\mathcal{D}$ and labeled by c, the algorithm returns a linear function h such that with probability of at least $1 - \delta$,

$$\mathbb{P}_{x \sim \mathcal{D}} \left[h(x) \neq c(x) \right] \leq \epsilon \,.$$

The function $m_{\mathcal{A}}$ is called the *sample complexity* of $\mathcal{A}$.

Our full algorithm is given in Algorithm 3. The following theorem proves a straightforward upper bound on the number of samples required to achieve a small Kendall-Tau distance with high probability. For simplicity of exposition, we assume that if two menus A and B have different rank representations, then $U(A) \neq U(B)$. However, our results can be modified appropriately for the case where they may have the same utility.

Theorem 6.3.2. *For every strict GP menu preference $\succeq$, Algorithm 3 recovers menu preferences $\succeq'$ such that with probability $1 - \delta$*

$$\mathbf{KT}(\succeq, \succeq') \leq \epsilon$$

with $O(n \log n + m_{\mathcal{A}}(\epsilon, \delta; 2n))$ menu comparisons where $m_{\mathcal{A}}$ is the sample complexity of the black-box linear classification algorithm $\mathcal{A}$.

Proof. By Lemma 6.3.3, the invocation to Algorithm 1 requires $O(n \log n)$ menu comparisons. Next, the Kendall-Tau distance between the true preferences $\succeq$ and the recovered preferences

Algorithm 3: Learning menu preferences approximately

Input: ϵ error rate
Run Algorithm 1 to get the rank representation of each menu
Select pairs of menus (A_i^1, A_i^2) uniformly at random for $i = 1, \ldots, m_{\mathcal{A}}(\epsilon, \delta; 2n)$
For all i, query comparison (A_i^1, A_i^2) and assign the result to y_i
Create dataset $D = \{(v(A_i^1) - v(A_i^2), y_i)\}_{i=1}^{m_{\mathcal{A}}(\epsilon,\delta;2n)}$
$\hat{\beta} \leftarrow$ weights from black-box linear classification algorithm $\mathcal{A}(D)$
Return the preference relation with utility representation $U(A) = \hat{\beta}^\intercal v(A)$

$\succeq'$ can be written as

$$\begin{aligned}\mathbf{KT}(\succeq, \succeq') &= \mathbb{P}_{(A,B)\sim\mathcal{U}}(A \succ B \text{ and } B \succ' A \text{ or } B \succ A \text{ and } A \succ' B) \\ &= \mathbb{P}_{(A,B)\sim\mathcal{U}}(\operatorname{sgn}(\beta^\intercal(v(A) - v(B))) \neq \operatorname{sgn}(\hat{\beta}^\intercal(v(A) - v(B)))),\end{aligned}$$

where $\mathcal{U}$ is the uniform distribution over pairs of menus. Thus, the Kendall-Tau distance $\mathbf{KT}(\succeq, \succeq')$ is equivalent to the generalization error of the linear classification algorithm, which must be less than ϵ with probability $1-\delta$ because we queried for $m_{\mathcal{A}}(\epsilon, \delta; 2n)$ samples. □

In the realizable setting, a linear classifier that achieves zero training error can easily be found through a linear program. A standard VC dimension bound shows that empirical risk minimization (ERM) achieves a sample complexity of $m(\epsilon, \delta; d) \leq O\left(\frac{d+\log\frac{1}{\delta}}{\epsilon}\right)$ of learning linear separators in the realizable setting [Bousquet et al., 2003]. Thus, as a corollary to Theorem 6.3.2, GP menu preferences can be learned approximately in $O(n \log n)$ menu comparisons.

Corollary 1. For every strict GP menu preference $\succeq$, if empirical risk minimization is used as the black-box linear classification algorithm, Algorithm 3 returns preferences $\succeq'$ in $O\left(\frac{n\log n+\log\frac{1}{\delta}}{\epsilon}\right)$ comparisons which satisfies $\mathbf{KT}(\succeq, \succeq') \leq \epsilon$ with probability $1-\delta$.

Chapter 7

Learning objective functions on recommender systems

So far, in Chapters 5 and Chapter 6, we explored specific ways that a human may deviate from reward-rationality. In Chapter 5, a pedagogic human did not act rationally with respect to the target reward function, but rather with respect to *teaching* the target reward function. In Chapter 6, the human was time-inconsistent and *sophisticated*, i.e. they act rationally in the first stage *given* that they may succumb to temptation in the second stage. In reality, there may be many unknown ways that the human deviates from reward-rationality.

The recommender system context is one that exemplifies this potential for unknown deviations from a reward-rational model. First, the objectives that people have on social media are complex and uncertain. Potential goals social media users may have include getting more followers, sharing information, making friends, making money. Secondly, the dynamics of how the user's actions end up affecting the goals of interest are also unknown. How does replying to a particular conversation change the number of followers you'll have down the road? The human will likely deviate in many unexpected ways from any reward-rational model that we explicitly write down.

How do we learn an objective despite such uncertainty over any kind of reward-rational model? Well, the defacto objective function on recommender systems today is to maximize user engagement, e.g. maximizing the probability that a user will click on an item or not. However, there is potentially a large gap between engagement signals and a desired notion of *value* that is worth optimizing for [Ekstrand and Willemsen, 2016]. Just because a user engages with an item doesn't mean they value it. A user might reply to an item because they are angry about it, or click an item in order to gain more information about it [Wen et al., 2019], or watch addictive videos out of temptation.

It is clear that engagements provide some signal for "value", but are not equivalent to it. Further, different types of engagement may provide differing levels of evidence for value. For example, if a user explicitly likes an item, we are more likely to believe that they value it, compared to if they had merely clicked on it. Ideally, we want the objective for our recommender system to take engagement signals into account, but only insofar as they

relate to a desired notion of "value". However, for the reasons just discussed, it is difficult to model each signal as being reward-rational. Instead, our approach relies on learning an objective through the use of one, strong signal (e.g. "don't show me this") for which we *can* assume a kind of rationality. This one strong signal will allow us to learn a model of how much evidence all other behaviors provide for "value".

Our contributions

We make three primary contributions[1].

1. We propose measurement theory as a principled approach to aggregating engagement signals into an objective function that captures a desired notion of "value". The resulting objective function can be optimized from data, serving as a plug-in replacement for the ad-hoc objectives typically used in engagement optimization frameworks.

2. Our approach is based on the creation of a latent variable model that relates value to various observed engagement signals. We devise a new identification strategy for the latent variable model tailored to the intended use case of online recommendation systems. Our identification strategy needs only a single robust engagement signal for which we know the conditional probability of value given the signal. Essentially, we invoke a reward-rational assumption on just one strong, explicit signal.

3. We implemented our approach on the Twitter platform on millions of users. In line with an established validity framework for measurement theory, we conduct a qualitative analysis of how well our model captures "value".

Measurement theory and latent variable models

The framework of *measurement theory* [Hand, 2004, Jackman, 2009, Causey, 1971] is widely used in the social sciences as a guide to measuring *unobservable theoretical constructs* like "quality of life", "political ideology", or "socio-economic status". Under the measurement approach, theoretical constructs are operationalized as latent variables, which are related to observable data through a latent variable model (LVM).

Similarly, we treat the "value" of a item[2] to a particular user as a theoretical construct, which we operationalize as a (binary) latent variable V. We represent the LVM as a a *Bayesian network* [Pearl, 2009] that contains V as well as each of the possible types of user engagements (clicks, shares, etc). The structure of the Bayesian network allows us to specify

[1]The contents of this chapter were originally published with Luca Belli and Moritz Hardt as "From Optimizing Engagement to Measuring Value" in FaccT 2021.

[2]We note that some items may only provide value to the user the first time they are interacted with. For example, if a user has just bought a particular dress, then it is probably not prudent to recommend that dress again, even though the user "values" the dress. Our measurement of value does not capture this dependence since it only depends on the user and item. Most real-world recommender systems have a candidate generation stage that comes before the ranking stage, and items that the user has interacted with before can simply be filtered out during the candidate generation stage [Thorburn et al., 2022].

conditional independences between variables, enabling us to capture dependencies like e.g. needing to click an item before replying to it.

Under the measurement approach, the ideal objective becomes clear: $\mathbb{P}(V = 1 \mid \texttt{Behaviors})$ - the probability the user values the item given their engagements with it. Such an objective uses all engagement signals, but only insofar provide evidence of Value V. If we can identify $\mathbb{P}(V = 1 \mid \texttt{Behaviors})$, then it can be used as a drop-in replacement for any objective that scores items based on engagement signals.

Our key insight is that we can identify $\mathbb{P}(V \mid \texttt{Behaviors})$ — the probability of Value given *all* behaviors — through the use of a single *anchor variable* A for which we know $\mathbb{P}(V = 1 \mid A = 1)$. The anchor variable, together with the structure of the Bayesian network, is what gives "value" its meaning. Through the choice of the anchor variable and the structure of the Bayesian network, the designer has the flexibility to give "value" subtly different meanings.

Recommendation engines have natural candidates for anchor variables: strong, explicit feedback from the user. For example, strong negative feedback could include downvoting or reporting a content item, or blocking another user. Strong positive feedback could be explicitly liking or upvoting an item. For negative feedback, we make the assumption that $\mathbb{P}(V = 1 \mid A = 1) = \epsilon$ for $\epsilon \approx 0$, while for positive feedback we make the assumption that $\mathbb{P}(V = 1 \mid A = 1) = 1 - \epsilon$.

A case study on the Twitter platform

We implemented our approach on the Twitter platform on millions of users. On Twitter, there are numerous user behaviors: clicks, favorites, retweets, replies, and many more. It would be difficult to directly specify an objective that properly trades off all these behaviors. Instead, we identify a natural anchor variable. On Twitter, users can give explicit feedback on tweets by clicking "See less often" (SLO) on them. We use SLO as our anchor and assume that the user does not value tweets they click "See less often" on. After specifying the anchor variable and the Bayesian network, we are able to learn $\mathbb{P}(V \mid \texttt{Behaviors})$ from data.

The model automatically learns a natural ordering of which behaviors should provide stronger evidence for Value V, e.g. $\mathbb{P}(V = 1 \mid \texttt{Retweet} = 1) > \mathbb{P}(V = 1 \mid \texttt{Reply} = 1) > \mathbb{P}(V = 1 \mid \texttt{Click} = 1)$. Furthermore, it learns complex inferences about the evidence provided by *combinations* of behavior. Such inferences would not be possible under the standard approach, which uses a linear combination of behaviors as the objective.

Unlike other work on recommender systems, we do not evaluate through engagement metrics. If we believe that engagement is not the same as the construct "value", then we cannot evaluate our approach merely by reporting engagement numbers. Instead, we must take a more holisitc approach. We discuss established approaches to assessing the *validity* [American Educational Research Association, American Psychological Association, National Council on Measurement in Education, Joint Committee on Standards for Educational and Psychological Testing, 2014, Messick, 1987, Reeves and Marbach-Ad, 2016] of a measurement,

and explain how they translate to the recommender system setting by using Twitter as an example.

7.1 Identification of the latent variable model

We now describe our general approach to operationalizing a target construct through a latent variable model (LVM) with an *anchor variable*. We operationalize the construct for value through a LVM in which the construct is represented through an unobserved, binary latent variable V that the other binary, observed behaviors provide evidence for. We assume there is one observed behavior, an *anchor variable* A, which we know $\mathbb{P}(V = 1 \mid A = 1)$ for. We represent all other observed behaviors in the binary random vector $\mathbf{B} = (B_1, \ldots, B_n)$. We refer to A as an anchor variable because it will provide the crucial link to identifying $\mathbb{P}(V \mid A, \mathbf{B})$. In other words, it will *anchor* the other observed behaviors $\mathbf{B}$ to Value V.

We represent the LVM as a Bayesian network. A Bayesian network is a directed acyclic graph (DAG) that graphically encodes a factorization of the joint distribution of the variables in the network. In particular, the DAG encodes all conditional independences among the nodes through the d-separation rule [Pearl, 2009]. This is important because in most real-world settings, the observed behaviors have complex dependencies among each other (e.g. one may need to click on an item before replying to it). Through our choice of the DAG we can model both the dependencies among the observed behaviors as well as the dependence of the unobserved variable V on the observed behaviors.

Our goal is to determine $\mathbb{P}(V \mid A, \mathbf{B})$ so that it can later be used downstream as a target for optimization. We now discuss sufficient conditions for identifying the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$. There are three assumptions on the anchor variable A that we will consider in turn.

Notation. We use $\mathrm{Pa}(X)$ to denote the parents of a node X and use $\mathrm{Pa}_{-V}(X) = \mathrm{Pa}(X) \backslash V$ to denote all parents of X except for V.

Assumption 1 (Value-sensitive)**.** For every realization b of the random vector $\mathbf{B}$, we have that $\mathbb{P}(A = 1 \mid \mathbf{B} = b, V = 1) \neq \mathbb{P}(A = 1 \mid \mathbf{B} = b, V = 0)$.

Assumption 1 simply means that the anchor A carries signal about Value V, regardless of what the other variables $\mathbf{B}$ are.[3]

Assumption 2 (No children)**.** The anchor variable A has no children.

Since the anchor A is chosen to be a strong type of explicit feedback, it is usually the last type of behavior the user engages in on a content item (e.g. a "report" button that removes the content from the user's timeline), and thus, it typically makes sense to model A as having no children.

[3]When combined with Assumption 2, Assumption 1 simplifies to the condition $\mathbb{P}(A = 1 \mid \mathrm{Pa}_{-V}(A) = z, V = 1) \neq \mathbb{P}(A = 1 \mid \mathrm{Pa}_{-V}(A) = z, V = 0)$ for every realization z of $\mathrm{Pa}_{-V}(A)$, the parents of A excluding V.

Assumption 3 (One-sided conditional independence)**.** Let $\mathrm{Pa}_{-V}(A)$ be all parents of A excluding V. Value V is independent from $\mathrm{Pa}_{-V}(A)$ given that $A = 1$:

$$\mathbb{P}(V = 1 \mid A = 1, \mathrm{Pa}_{-V}(A)) = \mathbb{P}(V = 1 \mid A = 1)\,.$$

Assumption 3 means that when the user has opted to give feedback ($A = 1$), the level of information that feedback contains about Value V does not depend on the other parents of A. The assumption rests on the fact that A is a strong type of feedback that the user only provides when they are confident of their assessment.

Conditions for identification

The next theorem establishes that under A1, the distribution of observable behaviors $\mathbb{P}(A, \mathbf{B})$ and the conditional distribution $\mathbb{P}(A \mid V, \mathbf{B})$ are sufficient for identifying the conditional distribution, $\mathbb{P}(V \mid A, \mathbf{B})$. The proof uses a *matrix adjustment method* [Rothman et al., 2008; pg. 360] and is very similar to that in Pearl [2010], Kuroki and Pearl [2014].

Theorem 7.1.1. *Let V and A be binary random variables and let $\mathbf{B} = (B_1, \ldots, B_n)$ be a binary random vector. If A1 holds, then the distributions $\mathbb{P}(A, \mathbf{B})$ and $\mathbb{P}(A \mid V, \mathbf{B})$ uniquely identify the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$.*

Proof. Since the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$ is equal to $\frac{\mathbb{P}(\mathbf{B},V)\cdot\mathbb{P}(A|\mathbf{B},V)}{\mathbb{P}(A,\mathbf{B})}$, we can reduce the problem to determining the distribution $\mathbb{P}(\mathbf{B}, V)$. We can relate $\mathbb{P}(\mathbf{B}, V)$ to the given distributions, $\mathbb{P}(A, \mathbf{B})$ and $\mathbb{P}(A \mid \mathbf{B}, V)$, via the law of total probability:

$$\mathbb{P}(A, \mathbf{B}) = \sum_{v \in \{0,1\}} \mathbb{P}(\mathbf{B}, V = v)\mathbb{P}(A \mid \mathbf{B}, V = v)\,. \tag{7.1}$$

For every realization b of the random vector $\mathbf{B}$, we can write Equation 7.1 as $z^b = \mathbf{P}^b \mu^b$ where the matrix $\mathbf{P}^b \in [0,1]^{2\times 2}$ and the vectors $\mu^b, z^b \in [0,1]^2$ are defined as

$$\begin{aligned}
&\mathbf{P}^b_{i,j} = \mathbb{P}(A = i \mid \mathbf{B} = b, V = j) \;\text{ for }\; i, j \in \{0, 1\}\,,\\
&\mu^b = [\mathbb{P}(\mathbf{B} = b, V = 0), \mathbb{P}(\mathbf{B} = b, V = 1)]^T\,,\\
&z^b = [\mathbb{P}(\mathbf{B} = b, A = 0), \mathbb{P}(\mathbf{B} = b, A = 1)]^T\,.
\end{aligned}$$

Determining the distribution $\mathbb{P}(\mathbf{B}, V)$ is equivalent to determining μ^b for all b. By Assumption 1, for all b we have $\mathbb{P}(A = 1 \mid B = b, V = 1) \neq \mathbb{P}(A = 1 \mid B = b, V = 0)$, which implies that the determinant of the matrix $\mathbf{P}^b$ is non-zero. Therefore, for all b, the vector μ^b is equal to $\mu^b = (\mathbf{P}^b)^{-1} z^b$. Thus, $\mathbb{P}(\mathbf{B}, V)$, and therefore the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$, is identified by the given distributions. □

If we add Assumption 2, i.e. the anchor A has no children, then the distributions $\mathbb{P}(A, \mathbf{B})$ and $\mathbb{P}(A \mid \mathrm{Pa}(A))$ are sufficient to identify $\mathbb{P}(V \mid A, \mathbf{B})$.

Corollary 2. If the joint distribution $\mathbb{P}(V, A, \mathbf{B})$ is Markov[4] with respect to a DAG G in which A1 and A2 hold, then the distributions $\mathbb{P}(A, \mathbf{B})$ and $\mathbb{P}(A \mid \text{Pa}(A))$ uniquely identify the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$.

Proof. In a Bayesian network, the *Markov blanket* for a variable X is the set of variables $\text{MB}(X) \subseteq \mathcal{Z}$ that shield X from all other variables $\mathcal{Z}$ in the DAG, i.e. $\mathbb{P}(X \mid \mathcal{Z}) = \mathbb{P}(X \mid \text{MB}(X))$ [Pearl, 2009]. The Markov blanket for a variable X consists of its parents, children, and parents of its children. Since the anchor A has no children, $\mathbb{P}(A \mid V, \mathbf{B}) = \mathbb{P}(A \mid \text{MB}(A)) = \mathbb{P}(A \mid \text{Pa}(A))$. Thus, by Theorem 7.1.1, $\mathbb{P}(A \mid \text{Pa}(A))$, and $\mathbb{P}(A, \mathbf{B})$ identify the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$ □

Finally, when we add Assumption 3, one-sided conditional independence, then the distributions $\mathbb{P}(V)$, $\mathbb{P}(A, \mathbf{B})$, $\mathbb{P}(V = 1 \mid A = 1)$, and $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$ are sufficient. The proof follows from Corollary 2 because, under Assumption 3, the distributions $\mathbb{P}(V = 1 \mid A = 1)$, $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$, and $\mathbb{P}(V)$ identify $\mathbb{P}(A \mid \text{Pa}(A))$.

Corollary 3. If the joint distribution $\mathbb{P}(V, A, \mathbf{B})$ is Markov with respect to a DAG G in which A1-3 hold, then $\mathbb{P}(V)$, $\mathbb{P}(A, \mathbf{B})$, $\mathbb{P}(V = 1 \mid A = 1)$, and $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$ uniquely identify the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$.

Proof. We will show that, under Assumption 3, the distributions $\mathbb{P}(V = 1 \mid A = 1)$, $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$, and $\mathbb{P}(V)$ identify $\mathbb{P}(A \mid \text{Pa}(A))$. The proof then follows from Corollary 2.

We show that we can identify $\mathbb{P}(A \mid \text{Pa}(A))$ by solving a set of linear equations. For shorthand let $p_{w,a,v} = \mathbb{P}(\text{Pa}_{-V}(A) = w, A = a, V = v)$. For any realization w, by marginalizing over A and V, we can derive the following four equations for the four unknown probabilities $p_{w,0,0}$, $p_{w,0,1}$, $p_{w,1,0}$, $p_{w,1,1}$:

$$\mathbb{P}(\text{Pa}_{-V}(A) = w, A = 0) = p_{w,0,0} + p_{w,0,1} \tag{7.2}$$

$$\mathbb{P}(\text{Pa}_{-V}(A) = w, A = 1) = p_{w,1,0} + p_{w,1,1} \tag{7.3}$$

$$\mathbb{P}(\text{Pa}_{-V}(A) = w, V = 0) = p_{w,0,0} + p_{w,1,0} \tag{7.4}$$

$$\mathbb{P}(\text{Pa}_{-V}(A) = w, V = 1) = p_{w,0,1} + p_{w,1,1} \tag{7.5}$$

Note that the LHS of Equations 7.2 and 7.3 are given by $\mathbb{P}(A, \mathbf{B})$ and the LHS of Equations 7.4 and 7.5 are given by the prior $\mathbb{P}(V)$ and $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$.

From Assumption 3, one-sided conditional independence, we know that $\mathbb{P}(V = 1 \mid A = 1, \text{Pa}_{-V}(A)) = \mathbb{P}(V = 1 \mid A = 1)$. Under one-sided conditional independence, the probability $p_{w,1,1}$ is determined by the given distributions:

$$\begin{aligned} p_{w,1,1} = \; & \mathbb{P}(A = 1) \cdot \mathbb{P}(\text{Pa}_{-V}(A) = w \mid A = 1) \\ & \cdot \mathbb{P}(V = 1 \mid A = 1, \text{Pa}_{-V}(A) = w) \\ = \; & \mathbb{P}(A = 1) \cdot \mathbb{P}(\text{Pa}_{-V}(A) = w \mid A = 1) \\ & \cdot \mathbb{P}(V = 1 \mid A = 1)\,. \end{aligned} \tag{7.6}$$

[4] A distribution $\mathbb{P}(X_1, \ldots, X_n)$ is said to be Markov with respect to a DAG G if it factorizes according to G, i.e. $\mathbb{P}(X_1, \ldots, X_n) = \prod_{i \in [n]} \mathbb{P}(X_i \mid \text{Pa}(X_i))$.

Since $p_{w,1,1}$ is determined by the given distributions, so are $p_{w,0,0}$, $p_{w,1,0}$, and $p_{w,0,1}$, which can be solved for through Equations 7.2-7.5. Since this holds for any realization w, the distribution $\mathbb{P}(A, V, \text{Pa}_{-V}(A)) = \mathbb{P}(A, \text{Pa}(A))$ is determined, which by Collorary 2 means that the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$ is determined. □

Specifying the distributions for identification

Corollary 3 establishes that, under Assumptions 1-3, the distributions $\mathbb{P}(V)$, $\mathbb{P}(A, \mathbf{B})$, $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$, and $\mathbb{P}(A = 1 \mid V = 1)$ are sufficient to determine the conditional distribution $\mathbb{P}(V \mid A, \mathbf{B})$ for the LVM. Where do we get these distributions?

1. The distribution of observable nodes $\mathbb{P}(A, \mathbf{B})$ is estimated by the empirical distribution of observed data.

2. The distribution $\mathbb{P}(V)$ over the latent variable for value V is a prior distribution that is specified by the modeler. Recall that our goal with the LVM is to use $\mathbb{P}(V = 1 \mid \mathbf{B}, A)$ as an objective to optimize. Since the prior $\mathbb{P}(V)$ only has a scaling effect on $\mathbb{P}(V = 1 \mid \mathbf{B}, A)$, it does not matter greatly. We set $\mathbb{P}(V)$ to be uniform, i.e. $\mathbb{P}(V = 1) = 0.5$.

3. The conditional probability $\mathbb{P}(V = 1 \mid A = 1)$ is specified by our assumption on the the anchor variable. The probability $\mathbb{P}(V = 1 \mid A = 1)$ is set to ϵ where $\epsilon \approx 0$ if A is explicit negative feedback or to $1 - \epsilon$ if A is explicit positive feedback.

4. That leaves the distribution $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$. We estimate $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$ heuristically using two sources of historical data that vary in their distribution of Value V. Suppose we have access to a dataset of historical recommendations $\mathcal{D}_R$ that were sent to users at random, as well as a dataset of historical recommendations that were algorithmically chosen, $\mathcal{D}_C$. Both kinds of datasets are commonly available on recommender systems due to the prevalence of A/B testing which typically tests new algorithmic changes against a randomized baseline. The randomized and algorithmic datasets will have different distributions of valuable content, $\mathbb{P}_R(V)$ and $\mathbb{P}_C(V)$, and different distributions of observed behavior, $\mathbb{P}_R(A, \mathbf{B})$ and $\mathbb{P}_C(A, \mathbf{B})$. However, we assume that $\mathbb{P}(A, \mathbf{B} \mid V)$, the probability of the observed behavior given Value V, is the same between the two datasets.[5] The following equations then hold:

$$\begin{aligned} \mathbb{P}_R(\text{Pa}_{-V}(A)) = \; & \mathbb{P}(\text{Pa}_{-V}(A) \mid V = 1)\mathbb{P}_R(V = 1) \\ & + \mathbb{P}(\text{Pa}_{-V}(A) \mid V = 0)\mathbb{P}_R(V = 0)\,, \qquad (7.7) \\ \mathbb{P}_C(\text{Pa}_{-V}(A)) = \; & \mathbb{P}(\text{Pa}_{-V}(A) \mid V = 1)\mathbb{P}_C(V = 1) \\ & + \mathbb{P}(\text{Pa}_{-V}(A) \mid V = 0)\mathbb{P}_C(V = 0)\,. \qquad (7.8) \end{aligned}$$

[5] If our DAG has Value V as a root node and can be interpreted as a causal Bayesian network [Pearl, 2009], then this is equivalent to assuming that the difference between the datasets corresponds to an intervention on V.

We specify $\mathbb{P}_R(V)$ and $\mathbb{P}_C(V)$ in an application-dependent way, but, generally, we assume the randomized dataset is lower value than the algorithmic one: $\mathbb{P}_R(V) < \mathbb{P}_C(V)$. Once we specify $\mathbb{P}_R(V)$ and $\mathbb{P}_C(V)$ and estimate $\mathbb{P}_R(A, \mathbf{B})$ and $\mathbb{P}_C(A, \mathbf{B})$ empirically, then we can solve Equations 7.7 and 7.8 to estimate $\mathbb{P}(\text{Pa}_{-V}(A) \mid V = 1)$. This is a heuristic approach that is appropriate for getting a rough estimate, but needs to be used with care. In practice, not all the differences between the randomized and algorithmic dataset can be explained by an intervention on Value V. For example, if the recommendation algorithm has historically been optimized for user clicks, then users in the algorithmic dataset may click on items more, but for reasons other than increased value.

Algorithm for identification

We now give more details on how we calculate the joint distribution $\mathbb{P}(V, A, \mathbf{B})$ given the distributions $\mathbb{P}(V)$, $\mathbb{P}(A, \mathbf{B})$, $\mathbb{P}(V = 1 \mid A = 1)$ and $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$. We use the structure of the Bayesian network to efficiently identify the joint distribution $\mathbb{P}(V, A, \mathbf{B})$ by fitting each factor $\mathbb{P}(X \mid \text{Pa}(X))$ for every variable X.

1. The factor for V is given by the prior $\mathbb{P}(V)$.[6]

2. The factor for A, i.e. $\mathbb{P}(A \mid \text{Pa}(A))$, can be identified from $\mathbb{P}(V)$, $\mathbb{P}(V = 1 \mid A = 1)$, and $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$ by solving a set of linear equations as in the proof of Corollary 3.

3. The factor for any behavior that does not have V as a parent is directly identified by the distribution of observable behaviors $\mathbb{P}(A, \mathbf{B})$.

4. The factors for the remaining behaviors which have V as a parent are fit through a *matrix adjustment method* [Rothman et al., 2008; pg. 360]. In particular, note that

$$\begin{aligned}&\mathbb{P}(X = 1, \text{Pa}_{-V}(X) = z, \text{Pa}_{-V}(A) = w, A = a) = \\ &\Big(\sum_{v \in \{0,1\}} \mathbb{P}(A = a \mid \text{Pa}_{-V}(A) = w, V = v) \\ &\cdot \mathbb{P}(X = 1, \text{Pa}_{-V}(X) = z, \text{Pa}_{-V}(A) = w, V = v)\Big)\end{aligned}$$

   We can also write the above equation in matrix form. Let $z_1, \ldots, z_m$ be all realizations

[6] Assuming that V is a root node, which is the case in any network we are interested in.

of $\mathrm{Pa}_{-V}(X)$, and define the matrices $Q^w \in [0,1]^{2\times m}$, $R^w \in [0,1]^{2\times 2}$, $S^w \in [0,1]^{2\times m}$ as[7]

$$Q^w_{a,i} = \mathbb{P}(X = 1, \mathrm{Pa}_{-V}(X) = z_i, \quad (7.9)$$
$$\mathrm{Pa}_{-V}(A) = w, A = a),$$
$$R^w_{av} = \mathbb{P}(A = a \mid \mathrm{Pa}_{-V}(A) = w, V = v), \quad (7.10)$$
$$S^w_{v,i} = \mathbb{P}(X = 1, \mathrm{Pa}_{-V}(X) = z_i, \quad (7.11)$$
$$\mathrm{Pa}_{-V}(A) = w, V = v).$$

Then, $Q^w = R^w S^w$ and $S^w = (R^w)^{-1} Q^w$.[8] Let S be the marginalization over w: $\sum_w S^w = (R^w)^{-1} Q^w$. Then $S_{v,i} = \mathbb{P}(X = 1, \mathrm{Pa}_{-V}(X) = z_i, V = v)$. Thus, the factor for X is equal to $\mathbb{P}(X \mid \mathrm{Pa}_{-V}(X) = z_i, V = v) = S_{v,i} / \mathbb{P}(\mathrm{Pa}_{-V}(X) = z_i, V = v)$. We fit nodes with V as a parent in topological order, so that we can always calculate the denominator from previously fit factors.

7.2 Application to Twitter

We implemented our approach on the Twitter platform on millions of users. On Twitter, there are many kinds of user behaviors: clicks, replies, favorites, retweets, etc. The typical approach to recommendations would involve optimizing an objective that trades-off these behaviors, usually with linear weights. However, designing an objective is a non-trivial problem. How exactly should we weigh favorites compared to clicks or replies or retweets or any of the numerous other behaviors? It is difficult to assess whether the weights we chose match the notion of "value" we intended.

Furthermore, even supposing that we could manually specify the "correct" weights through laborious trial-and-error, the correct weights change over time. For example, after videos shared on Twitter began to auto-play, the signal of whether or not a user watched a video presumably became less relevant. The reality is that the objective is never static—how users interact with the platform is constantly changing, and the objective must change accordingly.

Our approach provides a principled solution to objective specification. We directly operationalize our intended construct "value" as a latent variable V. The meaning of Value V is defined by the Bayesian network and the *anchor variable* A, a behavior that we believe provides strong evidence for value or the lack of it. On Twitter, the user can provide strong, explicit feedback by clicking "See less often" (SLO) on a tweet. We use SLO as our anchor A and assume that if a user clicks "See less often" on a tweet, they do not value it: $\mathbb{P}(V = 1 \mid \texttt{SLO} = 1) = 0$.

Under this approach, there is no need to manually specify how all the behaviors should factor into the objective. Having operationalized Value, the ideal objective to use is clear:

[7] If $\mathrm{Pa}_{-V}(X) \cap \mathrm{Pa}_{-V}(A) \neq \emptyset$ and $\mathrm{Pa}_{-V}(X) = z_i$ and $\mathrm{Pa}_{-V}(A) = w$ conflict, then simply set $Q^w_{0,i} = Q^w_{1,i} = 0$.

[8] R^w is invertible because of Assumption 1.

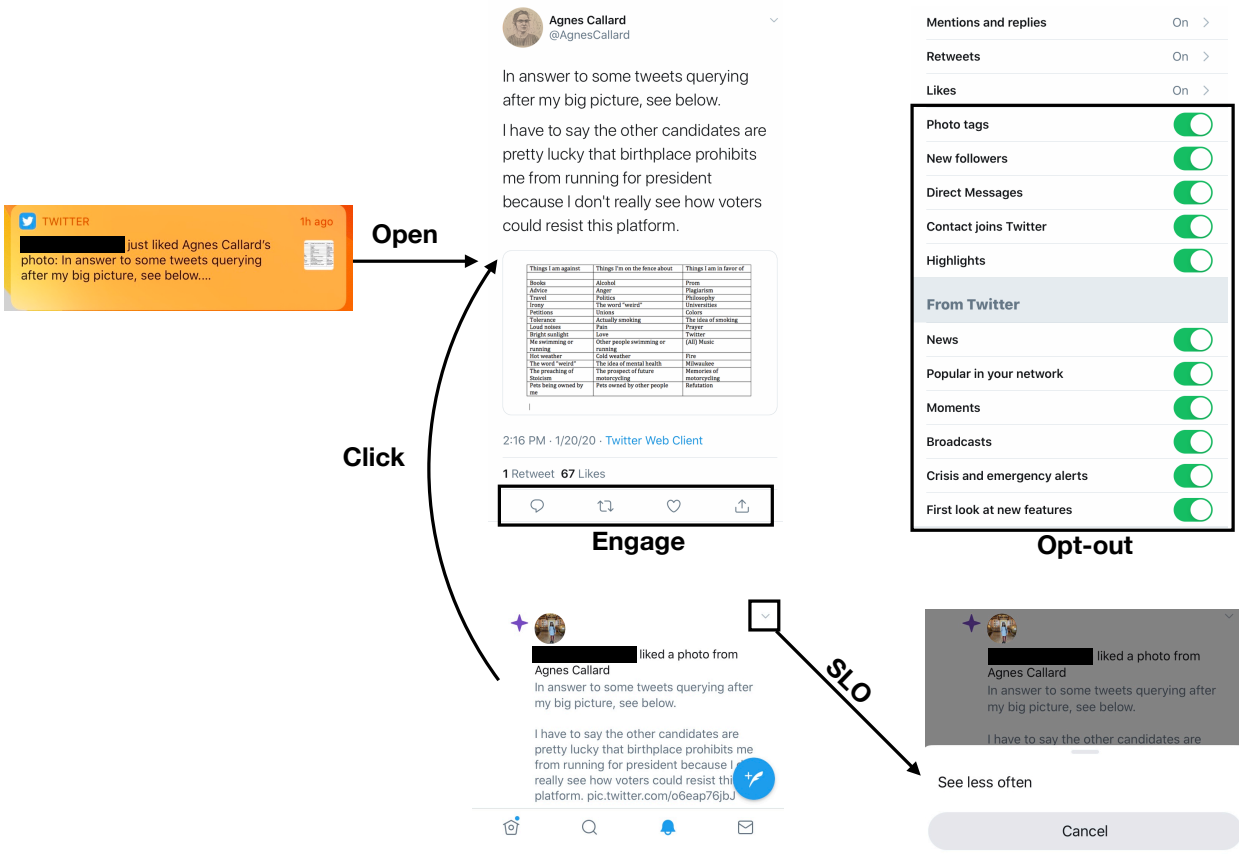

Figure 7.1: A workflow of how users can interact with ML-based notifications on Twitter. To view the tweet, the user can either "open" the notification from the home screen on their phone or "click" on it from the notifications tab within the app. If the user sees the tweet from their notifications tab, they can also click "See Less Often" on it. Once the user has opened or clicked on the notification, they can engage with the tweet in many ways, e.g. replying, retweeting, or favoriting. At any point, the user can opt-out of notifications all together.

$\mathbb{P}(V = 1 \mid \mathbf{B}, A)$ - the probability of Value V given the observed behaviors. As discussed in Section 7.2, we can directly estimate $\mathbb{P}(V = 1 \mid \mathbf{B}, A)$ from data. Furthermore, presuming that the anchor and structure of Bayesian network remain stable, we can regularly re-estimate the model with new data at any point, allowing us to account for change in user behavior on the platform.

The Bayesian network. We applied our approach to ML-driven notifications on Twitter. These notifications have various forms, e.g. "Users A, B, C just liked User Z's tweet", "User A just tweeted after a long time", or "Users A, B, C followed User Z". Figure 7.1 shows an example notification and how a user can interact with it. The Bayesian network in

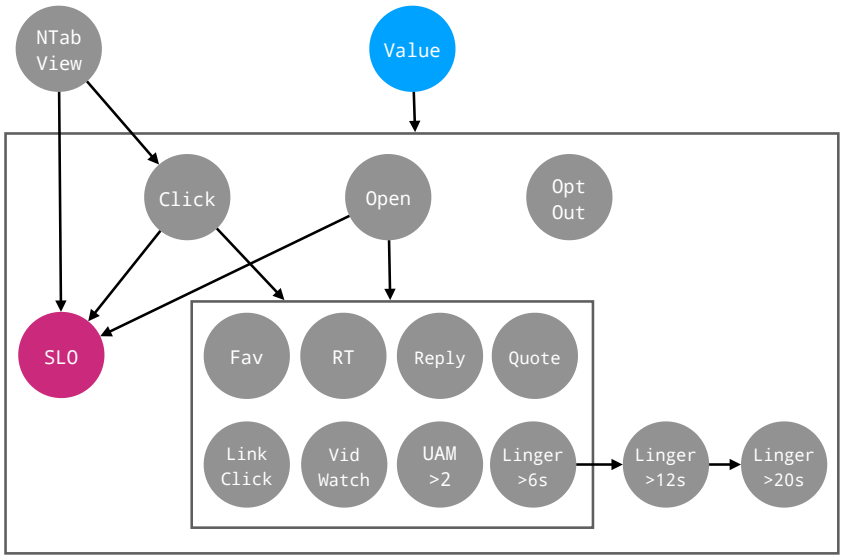

Figure 7.2: Bayesian network for Twitter notifications. An arrow from a node X to a box means that the node X is a parent of all the nodes in the box, e.g. `Click` and `Open` are parents of `Fav`, `RT`, ..., `Linger` $>$ `6s`. The latent variable `Value` is a parent of everything except `NTabView`. The anchor node `SLO` is highlighted in pink.

Figure 7.2 succinctly encodes the dependencies between different types of interactions users can have with notifications.[9]

Notifications are sent both to the user's home screen on their mobile phone, as well as to the notifications tab within the Twitter app. The user can start their interaction either by seeing the notification in their notification tab (`NTabView`), and then clicking on it (`Click`), or by seeing it as a the notification on their phone home screen and opening it from there directly (`Open`). After clicking or opening the notification, the user can engage in many more interactions: they can favorite (`Fav`), retweet (`RT`), quote retweet (`Quote`), or reply (`Reply`) to the tweet; if the tweet has a link, they can click on it (`LinkClick`); if it has a video, they can watch it (`VidWatch`). In addition, other implicit signals are logged: whether the amount the user lingered on the tweet exceeds certain thresholds (`Linger` $>$ `6s`, `Linger` $>$ `12s`, `Linger` $>$ `20s`) and whether the number of user active minutes (`UAM`) spent in the app after clicking/opening the notification exceeds a threshold.

Furthermore, when the user is in the notification tab, the user can provide explicit feedback on a particular notification by clicking "See Less Often" (`SLO`) on it. Notably, unlike other types of behavior, the user does not need to actually click or open the notification before clicking SLO. However, we found empirically that users are more likely to click SLO after clicking or opening the notification, probably because they need to gain more informa-

[9]The network can be interpreted as a *causal* Bayesian network [Pearl, 2009], although for our purposes, we do not strictly need the causal interpretation.

tion before making an assessment. Thus, in addition to `NTabView`, we also model `Click` and `Open` as parents of `SLO`.

Finally, at any time the user can opt-out of notifications to their phone home screen (`OptOut`). If the user saw the tweet on the same day they opted out of notifications, then `OptOut` = 1. Since ML-based notifications are relatively rare on Twitter (users usually get less than one a day), there are usually at most one or two notifications attributed to an opt-out event.

We model the latent variable V as being a parent of all behaviors except `NTabView` (whether or not the user saw the notification in their notifications tab or not). Since users may check their notifications tab for many other notifications, it is difficult to attribute `NTabView` to a particular notification, and so we consider it to be an exogenous, random event.

Identifying the joint distribution We fit our model on three days of data that contained of all user interactions with ML-based push notifications on Twitter. In Section 7.1, we proved that the target objective - the conditional distribution $\mathbb{P}(V = 1 \mid \mathbf{B}, A)$ - is uniquely identified from $\mathbb{P}(V = 1 \mid A = 1)$, $\mathbb{P}(V)$, $\mathbb{P}(\mathbf{B}, A)$, and $\mathbb{P}(\text{Pa}_{-V}(A) \mid A)$ (see Corollary 3). We set the four distributions as follows. We used `SLO` as our anchor variable A and assumed that $\mathbb{P}(V = 1 \mid A = 1) = 0$, i.e. a user never says "See less often" if they value the notification. The prior distribution of value $\mathbb{P}(V)$ was set to be uniform. The distribution of observed behaviors $\mathbb{P}(\mathbf{B}, A)$ was set to the empirical distribution. The distribution $\mathbb{P}(\text{Pa}_{-V}(A) \mid V)$ was estimated as described in Section 7.1 by using two sources of historical data, one in which notifications were sent at random and the other in which notifications were sent according to a recommendation algorithm.[10]

Evaluation of internal structure. Assessing our measure of "value" for validity will necessarily be an on-going and multi-faceted process. We do not, unlike other papers on recommendation, report engagement metrics. The reason is that if we expect our measure of "value" to differ from engagement, we cannot evaluate it by simply reporting engagement metrics. The evaluation of a measurement necessitates a more holistic approach. In Section 7.3, we describe the five categories of evidence for validity described by the *Standards for Educational and Psychological Testing*, the handbook considered the gold standard on approaches to testing [American Educational Research Association, American Psychological Association, National Council on Measurement in Education, Joint Committee on Standards for Educational and Psychological Testing, 2014].

Here, we focus on evaluating what is known as *evidence based on internal structure*, i.e whether expected theoretical relationships between the variables in the model hold. To justify why the structure of our Bayesian network is necessary, we compare our full model from Figure 7.2 to two other models: a naive Bayes model and the full model but without arrows from `Open` and `Click` to `SLO`. In Table 7.1, we show $\mathbb{P}(V = 1 \mid \texttt{Behavior} = 1)$ for all behaviors and models. As noted by prior work [Pearl, 2009, Halpern et al., 2016],

[10] We assume that the dataset of randomized notifications has a prior probability $\mathbb{P}_R(V = 1) = 0$ and the dataset of algorithmically chosen notifications has a prior probability $\mathbb{P}_C(V = 1) = 0.5$.

$\mathbb{P}(V = 1 \mid \texttt{Behavior} = 1)$

Behavior	Naive Bayes	Click, Open $\nrightarrow$ SLO	Full Model
OptOut	0	0	0
Click	0	0.316	0.652
Open	0	0.442	0.685
UAM	0	0.157	0.719
VidWatch	0	0.254	0.772
Linger > 6s	0	0.264	0.802
LinkClick	0	0.320	0.836
Reply	0.358	0.570	0.932
Linger > 12s	0	0.245	0.948
Fav	0.579	0.672	0.949
RT	0.680	0.720	0.956
Linger > 20s	0.019	0.296	0.991
Quote	1.0	1.0	1.0

Table 7.1: The inferences made by LVMs with different DAGs. For each model and for each behavior, we list $\mathbb{P}(V = 1 \mid \texttt{Behavior} = 1)$ – how much evidence the model learns that a behavior provides for Value V (when all other behaviors are marginalized over).

matrix adjustment methods can result in negative values when conditional independence assumptions are not satisfied. To address this, we clamp all inferences to the interval $[0, 1]$.

The first, simple theoretical relationship we expect to hold is that compared to observing no user interaction, observing any user behavior besides opt-out should increase the probability that the user values the tweet, i.e. $\mathbb{P}(V = 1 \mid \texttt{Behavior} = 1) < \mathbb{P}(V = 1) = 0.5$ for all $\texttt{Behavior} \neq \texttt{OptOut}$. Furthermore, we also expect some behaviors to provide stronger signals of value than others, e.g. that $\mathbb{P}(V = 1 \mid \texttt{Fav} = 1) > \mathbb{P}(V = 1 \mid \texttt{Click} = 1)$.

The first model is the naive Bayes model, which simply assumes that all behaviors are conditionally independent given Value V. It does extremely poorly - almost all inferences have negative values and are clamped to zero, indicating that the conditional independence assumptions are unrealistic.

The second model is the full model except without arrows from Click and Open to SLO. It models all pre-requisite relationships between behaviors, i.e. if a behavior X is required before another behavior Y, then there is an arrow from X to Y. Compared to the naive Bayes model, the second model does not make mainly negative-valued inferences, indicating

that its conditional independence assumptions are more realistic. However, relative to the prior, most behaviors actually reduce the probability of `Value`, rather than increase it!

After investigation, we realized that although users were not technically *required* to click or open the notification before clicking SLO, in practice, they were more likely to do so, probably because they needed to gain information before making an assessment. We found that explicitly modeling the connection, i.e. adding arrows from `Click` and `Open` to `SLO` was critical for making reasonable inferences. We believe this takeaway will apply across recommender systems. The user never has perfect information and may need to engage with an item before providing explicit feedback [Wen et al., 2019]. It is important to model the relationship between information-gaining behavior and explicit feedback in the Bayesian network.

Our full model satisfies the theoretical relationships we expect. All the behaviors that we expect to increase the probability of Value V do indeed do so. Furthermore, the relative strength of different types of behavior seems reasonable as well, e.g. $\mathbb{P}(V = 1 \mid \texttt{Fav} = 1)$ and $\mathbb{P}(V = 1 \mid \texttt{RT} = 1)$ are higher than $\mathbb{P}(V = 1 \mid \texttt{VidWatch} = 1)$ and $\mathbb{P}(V = 1 \mid \texttt{LinkClick} = 1)$.

The full model also makes more nuanced theoretical inferences. Recall that `UAM` is whether or not the user had high user active minutes after either clicking the notification from notifications tab or by opening the notification from their phone home screen. The model learns that `UAM` is a highly indicative signal after `Open`, but not after `Click`: $\mathbb{P}(V = 1 \mid \texttt{Open} = 1, \texttt{UAM} = 1) = 0.906$ and $\mathbb{P}(V = 1 \mid \texttt{Click} = 1, \texttt{UAM} = 1) = 0.641$. This makes sense because if the user clicks from notifications tab, it means they were already in the app, and it is difficult to attribute their high UAM to the notification in particular. On the other hand, if the user enters the app because of the notification, it is a much more direct attribution.

It is clear that manually specifying the parameters our model infers would be very difficult. The advantage of our approach is that after specifying (a) the anchor variable and (b) the Bayesian network, we can automatically learn these parameters from data. Further, the model ends up learning complex inferences (e.g. that `UAM` is more reliable after `Open` than `Click`) that would be impossible to specify under the typical linear weighting of behaviors.

7.3 Assessing validity

Thus far, we have described our framework for designing a measure of "value", which can be used as a principled replacement for the ad-hoc objectives ordinarily used in engagement optimization. How do we evaluate such a measure? Notably, we do not advocate evaluating the measure purely through engagement metrics. If we expect our measure of "value" to differ from engagement, then we cannot evaluate it by simply reporting engagement metrics. Instead, the assessment of any measure is necessarily an ongoing, multi-faceted, and interdisciplinary process.

To complete the presentation of our framework, we now discuss approaches to assess the *validity* [Messick, 1987, American Educational Research Association, American Psycho-

logical Association, National Council on Measurement in Education, Joint Committee on Standards for Educational and Psychological Testing, 2014, Reeves and Marbach-Ad, 2016] of a measurement. In the most recent (2014) edition of the *Standards for Educational and Psychological Testing*, the handbook considered the gold standard on approaches to testing, there are five categories of evidence for validity. We visit each in turn, and describe how they translate to the recommender system setting, using Twitter as an example.

Evidence based on content refers to whether the content of a measurement is sufficient to fully capture the target construct. For example, we may question whether a measure of "socio-economic status" that includes income, but does not account for wealth, accurately captures the content of the construct [Jacobs and Wallach, 2019]. In the recommender engine setting, content-based evidence asks us to reflect on whether the behaviors available on the platform are sufficient to capture a worthy notion of the construct "value". For example, if the only behavior observed on the platform were clicks by the user, then we may be skeptical of any measurement of "value" derived from user behavior. What content-based evidence makes clear is that to measure any worthy notion of "value", it is essential to design platforms in which users are empowered with richer channels of feedback. Otherwise, no measurement derived from user behavior will accurately capture the construct.

Evidence based on cognitive processes. Measurements derived from human behavior are often based on implicit assumptions about the cognitive processes subjects engage in. Cognitive process evidence refers to evidence about such assumptions, often derived from explicit studies with subjects. For example, consider a reading comprehension test. We assume that high-scoring students succeed by using critical reading skills, rather than a superficial heuristic like picking the answers with the longest length. To gain evidence about whether this assumption holds, we might, for instance, ask students to take the test while verbalizing what they are thinking.

Similarly, in the recommender engine setting, we want to verify whether user behaviors occur for the reasons we think they do. On Twitter, one might think to use `Favorite` as an anchor for Value V, assuming that $\mathbb{P}(V = 1 \mid \texttt{Favorite} = 1) \approx 1$. However, users actually choose to favorite items for reasons that may not reflect value – such as to bookmark a tweet or to stop a conversation. Cognitive process evidence highlights the importance of user research in assessing the validity of any measure of "value".

Evidence based on internal structure refers to whether the observations the measurement is derived from conform to expected, theoretical relationships. For example, for a test with questions which we expect to be of increasing difficulty, we would assess whether students actually perform worse on later questions, compared to earlier ones. In the recommender system context, we may have expectations on which types of user behaviors should provide stronger signal for value. In Section 7.2, we evaluated internal structure by comparing $\mathbb{P}(V = 1 \mid \texttt{Behavior} = 1)$ for all behaviors.

Evidence based on relations with other variables is concerned with the relationships between the measurement and other variables that are external to the measurement. The external variables could be variables which the measurement is expected to be similar to or predict, as well as variables which the measurement is expected to differ from. For

example, a new measure of depression should correlate with other, existing measures of depression, but correlate less with measures of other disorders. In the recommender system context, we might look at whether our derived measurement of "value" is predictive of answers that users give in explicit surveys about content they value. We could also verify that our measure of "value" does not differ based on protected attributes, such as the sex or race of the author of the content.

Evidence based on consequences. Finally, the consequences of a measurement cannot be separated from its validity. Consider a test to measure student mathematical ability. The test is used to sort students into beginner or advanced classes with the hypothesis that all students will do better after being sorted into their appropriate class. If it turns out that students sorted by the test do *not* perform better, that may give us reason to reassess the original test. In the recommender system context, if we find that after using our measurement of value to optimize recommendations, more users complain or quit the platform, then we would have reason to revise our measurement.

Chapter 8

Conclusion

The goal of this thesis was to provide a *cohesive* and *unified* way to learn objective functions from many sources of information. We began by introducing our formalism, *reward-rational choice*, that unifies reward learning from many sources of information. The RRC framework provides a common lens from which to understand previously proposed feedback types as well as a recipe for formalizing new sources of information. We also showed how RRC supports learning from and actively selecting from multiple feedback types at once. Finally, when someone can give multiple types of feedback, we showed how RRC can be used to formalize and learn from a person's choice of feedback type itself.

After exploring the uses and implications of the RRC formalism, we turned towards relaxing the core assumption: that the human is rational with respect to the reward function to be inferred. We considered three different settings. First, the case where the human may be pedagogic, i.e. rational with respect to *teaching* the target reward function rather than rational with respect to the target reward function itself. Second, the case where the human is time-inconsistent and sophisticated, i.e. they rationally restrict their options earlier on in order to avoid succumbing to temptation later. Third, we looked at recommender systems, a setting where it is difficult the model the human as rational with respect to *any* objective function, let alone the target reward function.

There are many avenues for future work. First, we would like to further expand upon the applications of RRC. In the thesis, we gave examples, case studies, and experiments to showcase the ways that RRC can be used to combine, actively select, and formalize new feedback types. But we would like to illustrate these benefits in more complex domains, e.g. in more complex simulated environments like Atari or on a physical robot.

We investigated the robustness of RRC when the human may be acting pedagogically (Chapter 5) or the human's meta-rationality may be misspecified (Section 4.3), but there are many other ways an RRC model may be misspecified and it would be informative to understand how different kinds of misspecification affect reward inference. For example, recent work investigates the effects of choice set misspecification [Jonnavittula and Losey, 2021, Freedman et al., 2021] and shows that under-estimating a choice set can actually be desirable as it leads to the robot to be more risk-averse in its reward inference.

In the recommender system setting, we did not explicitly model the human as optimizing an objective function. Instead, we built upon the framework of measurement theory to learn a measurement model for what the user "values". We used strong explicit feedback (the "anchor" variable) to learn how much all other behavioral signals (e.g. clicks, replies, likes) should provide evidence for what the user "values". Is there a way to combine the benefits of the anchor variable approach with the RRC approach? For example, can we use anchor variables to learn an explicit human model? Can we apply measurement style evaluations of validity to the reward learning setting?

There are surely many more sources of information that have not yet been formalized but can be leveraged to learn about human preferences. We hope that the frameworks and ideas presented in this thesis can provide insight into how to harness all these signals to create systems that are aligned with human values.

Bibliography

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.

Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 12–27. Springer, 2011.

American Educational Research Association, American Psychological Association, National Council on Measurement in Education, Joint Committee on Standards for Educational and Psychological Testing. *Standards for educational and psychological testing*. AERA, 2014.

Dan Ariely and Klaus Wertenbroch. Procrastination, deadlines, and performance: Self-control by precommitment. *Psychological science*, 13(3):219–224, 2002.

Nava Ashraf, Dean S Karlan, and Wesley Yin. Household decision making and savings impacts: further evidence from a commitment savings product in the philippines. *Yale University Economic Growth Center Discussion Paper*, (939), 2006.

Andrea Bajcsy, Dylan P Losey, Marcia K O'Malley, and Anca D Dragan. Learning robot objectives from physical human interaction. *Conference on Robot Learning (CoRL)*, 2017.

Chris L Baker, Rebecca Saxe, and Joshua B Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.

Maria-Florina Balcan, Amit Daniely, Ruta Mehta, Ruth Urner, and Vijay V Vazirani. Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics (WINE)*, 2014.

Eyal Beigman and Rakesh Vohra. Learning from revealed preference. In *ACM Conference on Economics and Computation (EC)*, 2006.

Leon Bergen. *Joint inference in pragmatic reasoning*. PhD thesis, Massachusetts Institute of Technology, 2016.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

John Beshears, James J Choi, Christopher Harris, David Laibson, Brigitte C Madrian, and Jung Sakong. Self control and commitment: can decreasing the liquidity of a savings account increase deposits? Technical report, National Bureau of Economic Research, 2015.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Michael Bloem and Nicholas Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. In *53rd IEEE Conference on Decision and Control*, pages 4911–4916. IEEE, 2014.

Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Summer School on Machine Learning*, pages 169–207. Springer, 2003.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, Weiwei Cheng, and Eyke Hüllermeier. Preference-based evolutionary direct policy search. In *ICRA Workshop on Autonomous Learning*, 2013.

Robert L Causey. Patrick suppes and joseph l. zinnes. basic measurement theory. handbook of mathematical psychology, volume i, edited by r. duncan luce, robert r. bush, and eugene galanter, john wiley and sons, inc., new york and london1963, pp. 1–76. *The Journal of Symbolic Logic*, 36(2):322–323, 1971.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pages 4299–4307, 2017.

Anca D Dragan and Siddhartha S Srinivasa. *Formalizing assistive teleoperation*. MIT Press, July, 2012.

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, 2013a.

Anca D. Dragan, Siddhartha S. Srinivasa, and Kenton C. T. Lee. Teleoperation with intelligent and customizable interfaces. *Journal of Human-Robot Interaction*, 2013b.

Michael D Ekstrand and Martijn C Willemsen. Behaviorism is not enough: better recommendations through listening to users. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 221–224, 2016.

Owain Evans, Andreas Stuhlmüller, and Noah Goodman. Learning the preferences of ignorant, inconsistent agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*, pages 49–58, 2016.

Jaime Fisac, Monica A Gates, Jessica B Hamrick, Chang Liu, Dylan Hadfield-Mennell, Malayandi Palaniappan, Dhruv Malik, S Shankar Sastry, Thomas L Griffiths, and Anca D Dragan. Pragmatic-pedagogic value alignment. In *International Symposium on Robotics Research (ISRR)*, 2018.

Rachel Freedman, Rohin Shah, and Anca Dragan. Choice set misspecification in reward inference. *arXiv preprint arXiv:2101.07691*, 2021.

Daniel Fried, Jacob Andreas, and Dan Klein. Unified pragmatic models for generating and following instructions. *NAACL*, 2018a.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pages 3314–3325, 2018b.

Justin Fu, Anoop Korattikara, Sergey Levine, and Sergio Guadarrama. From language to goals: Inverse reinforcement learning for vision-based instruction following. *arXiv preprint arXiv:1902.07742*, 2019.

Xavier Giné, Dean Karlan, and Jonathan Zinman. Put your money where your butt is: a commitment contract for smoking cessation. *American Economic Journal: Applied Economics*, 2(4):213–35, 2010.

Noah D Goodman and Andreas Stuhlmüller. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science*, 5(1):173–184, 2013.

Noah D Goodman, Chris L Baker, and Joshua B Tenenbaum. Cause and intent: Social reasoning in causal learning. In *Proceedings of the 31st annual conference of the cognitive science society*, pages 2759–2764. Citeseer, 2009.

Herbert P Grice. Logic and conversation. In *Speech acts*, pages 41–58. Brill, 1975.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in neural information processing systems*, pages 2625–2633, 2013.

Faruk Gul and Wolfgang Pesendorfer. Temptation and self-control. *Econometrica*, 69(6): 1403–1435, 2001.

Sami Haddadin, Alin Albu-Schaffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3356–3363. IEEE, 2008.

Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, pages 3909–3917, 2016.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. The off-switch game. In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017a.

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan. Inverse reward design. In *Advances in neural information processing systems*, pages 6765–6774, 2017b.

Yoni Halpern, Steven Horng, and David Sontag. Clinical tagging with joint probabilistic models. In *Conference on Machine Learning for Health Care*, 2016.

David J Hand. *Measurement theory and practice: The world through quantification*. Arnold London, 2004.

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

Mark K Ho, Michael Littman, James MacGlashan, Fiery Cushman, and Joseph L Austerweil. Showing versus doing: Teaching by demonstration. In *Neural Information Processing Systems (NeurIPS)*, 2016.

Mark K Ho, Michael L Littman, Fiery Cushman, and Joseph L Austerweil. Effectively learning from pedagogical demonstrations. In *Annual Conference of the Cognitive Science Society (CogSci)*, 2018.

Neville Hogan. Impedance control: An approach to manipulation: Part ii—implementation. 1985.

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.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in Atari. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Simon Jackman. Measurement. In *The Oxford Handbook of Political Methodology*, chapter 9. Oxford University Press, 09 2009. ISBN 9780199286546.

Abigail Z Jacobs and Hanna Wallach. Measurement and fairness. *arXiv preprint arXiv:1912.05511*, 2019.

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.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Ananth Jonnavittula and Dylan P Losey. I know what you meant: learning human objectives by (under) estimating their choice set. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2747–2753. IEEE, 2021.

Nathan Kallus and Madeleine Udell. Revealed preference at scale: Learning personalized preferences from assortment choices. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 821–837. ACM, 2016.

Victoria Krakovna. Specification gaming examples in AI, Jun 2018. URL `https://vkrakovna.wordpress.com/2018/04/02/specification-gaming-examples-in-ai/`.

Manabu Kuroki and Judea Pearl. Measurement bias and effect restoration in causal inference. *Biometrika*, 101(2):423–437, 2014.

Cassidy Laidlaw and Anca Dragan. The boltzmann policy distribution: Accounting for systematic suboptimality in human models. *arXiv preprint arXiv:2204.10759*, 2022.

Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. *arXiv preprint arXiv:1206.4617*, 2012.

Robert Tyler Loftin, James MacGlashan, Bei Peng, Matthew E Taylor, Michael L Littman, Jeff Huang, and David L Roberts. A strategy-aware technique for learning behaviors from discrete human feedback. In *AAAI Conference on Artificial Intelligence*, 2014.

Dylan P Losey and Marcia K O'Malley. Including uncertainty when learning from human corrections. *arXiv preprint arXiv:1806.02454*, 2018.

Dylan P Losey and Marcia K O'Malley. Trajectory deformations from physical human–robot interaction. *IEEE Transactions on Robotics*, 34(1):126–138, 2017.

R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Wiley, 1959.

James MacGlashan, Monica Babes-Vroman, Marie desJardins, Michael L Littman, Smaranda Muresan, Shawn Squire, Stefanie Tellex, Dilip Arumugam, and Lei Yang. Grounding english commands to reward functions. In *Robotics: Science and Systems*, 2015.

James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2285–2294. JMLR.org, 2017.

Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In *International Conference on Machine Learning (ICML)*, 2012.

Samuel Messick. Validity. *ETS Research Report Series*, 1987(2):i–208, 1987.

Sören Mindermann, Rohin Shah, Adam Gleave, and Dylan Hadfield-Menell. Active inverse reward design. In *Proceedings of the 1st Workshop on Goal Specifications for Reinforcement Learning*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

Ted O'Donoghue and Matthew Rabin. Doing it now or later. *American economic review*, 89 (1):103–124, 1999.

Pedro A Ortega and Daniel A Braun. Thermodynamics as a theory of decision-making with information-processing costs. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 469(2153):20120683, 2013.

Malayandi Palan, Nicholas C Landolfi, Gleb Shevchuk, and Dorsa Sadigh. Learning reward functions by integrating human demonstrations and preferences. In *RSS*, 2019.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Judea Pearl. On measurement bias in causal inference. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 425–432. AUAI Press, 2010.

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *IJCAI*, volume 7, pages 2586–2591, 2007.

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.

Ellis Ratner, Dylan Hadfield-Menell, and Anca D Dragan. Simplifying reward design through divide-and-conquer. *arXiv preprint arXiv:1806.02501*, 2018.

Todd D Reeves and Gili Marbach-Ad. Contemporary test validity in theory and practice: A primer for discipline-based education researchers. *CBE—Life Sciences Education*, 15(1): rm1, 2016.

Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. Watch and learn: Optimizing from revealed preferences feedback. In *ACM Symposium on Theory of Computing*, 2016.

Kenneth J Rothman, Sander Greenland, and Timothy L Lash. *Modern epidemiology*, volume 3. Wolters Kluwer Health/Lippincott Williams & Wilkins Philadelphia, 2008.

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.

John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *International Journal of Robotics Research (IJRR)*, 2014.

Janet Schwartz, Jason Riis, Brian Elbel, and Dan Ariely. Inviting consumers to downsize fast-food portions significantly reduces calorie consumption. *Health Affairs*, 31(2):399–407, 2012.

Rohin Shah, Dmitrii Krasheninnikov, Jordan Alexander, Pieter Abbeel, and Anca Dragan. Preferences implicit in the state of the world. In *ICLR*, 2019.

Rohin Shah, Pedro Freire, Neel Alex, Rachel Freedman, Dmitrii Krasheninnikov, Lawrence Chan, Michael D Dennis, Pieter Abbeel, Anca Dragan, and Stuart Russell. Benefits of assistance over reward learning. *Workshop on Cooperative AI, NeurIPS)*, 2020.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Herbert A Simon. Rational choice and the structure of the environment. *Psychological review*, 63(2):129, 1956.

Joar Skalse, Matthew Farrugia-Roberts, Stuart Russell, Alessandro Abate, and Adam Gleave. Invariance in policy optimisation and partial identifiability in reward learning. *arXiv preprint arXiv:2203.07475*, 2022.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

Luke Thorburn, Priyanjana Bengani, and Jonathan Stray. How platform recommenders work, Feb 2022. URL `https://medium.com/understanding-recommenders/how-platform-recommenders-work-15e260d9a`

Pei Wang, Pushpi Paranamana, and Patrick Shafto. Generalizing the theory of cooperative inference. *AISTATS*, 2019.

Hongyi Wen, Longqi Yang, and Deborah Estrin. Leveraging post-click feedback for content recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 278–286, 2019.

Klaus Wertenbroch. Consumption self-control by rationing purchase quantities of virtue and vice. *Marketing science*, 17(4):317–337, 1998.

Aaron Wilson, Alan Fern, and Prasad Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In *Advances in neural information processing systems*, pages 1133–1141, 2012.

Christian Wirth and Johannes Fürnkranz. On learning from game annotations. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(3):304–316, 2014.

Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. Model-free preference-based reinforcement learning. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *The Journal of Machine Learning Research*, 18(1):4945–4990, 2017.

Scott Cheng-Hsin Yang, Yue Yu, Arash Givchi, Pei Wang, Wai Keen Vong, and Patrick Shafto. Optimal cooperative inference. *AISTATS*, 2018.

Morteza Zadimoghaddam and Aaron Roth. Efficiently learning from revealed preference. In *International Workshop on Internet and Network Economics (WINE)*, 2012.

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.