Leveraging Machine Learning to Automatically Derive Robust Planning Strategies from Biased Models of the Environment

Anirudha Kemtur[1], Yash Raj Jain[1], Aashay Mehta
Max Planck Institute for Intelligent Systems, Tübingen, Germany

Frederick Callaway
Department of Psychology, Princeton University, Princeton, NJ, USA

Saksham Consul, Jugoslav Stojcheski, Falk Lieder
Max Planck Institute for Intelligent Systems, Tübingen, Germany

[1] These authors contributed equally.

Abstract

Teaching clever heuristics is a promising approach to improve decision-making. We can leverage machine learning to discover clever strategies automatically. Current methods require an accurate model of the decision problems people face in real life. But most models are misspecified because of limited information and cognitive biases. To address this problem we develop strategy discovery methods that are robust to model misspecification. Robustness is achieved by modeling model-misspecification and handling uncertainty about the real-world according to Bayesian inference. We translate our methods into an intelligent tutor that automatically discovers and teaches robust planning strategies. Our robust cognitive tutor significantly improved human decision-making when the model was so biased that conventional cognitive tutors were no longer effective. These findings highlight that our robust strategy discovery methods are a significant step towards leveraging artificial intelligence to improve human decision-making in the real world.

Keywords: automatic strategy discovery; cognitive tutors; robust reinforcement learning; decision-making

Introduction

A promising approach to help people avoid bad decisions is to teach them clever decisions strategies that are well suited to common decision problems in everyday life (Gigerenzer & Todd, 1999; Hertwig & Grüne-Yanoff, 2017). A bottleneck of this approach is discovering efficient strategies that reliably lead to good decisions. Building on a new normative and prescriptive theory of good decision-making (Lieder & Griffiths, 2020), recent work has began to address this bottleneck by leveraging machine learning to develop algorithms for automatic strategy discovery (Callaway, Lieder, et al., 2018; Callaway, Gul, Krueger, Griffiths, & Lieder, 2018; Lieder, Krueger, & Griffiths, 2017). Automatic strategy discovery methods require a model of the environment. Given such a model, methods from machine learning are used to compute the strategy that achieves the best possible trade-off between performance and computational cost in the simulated environment. When there is a mismatch between simulation and reality, the discovered strategy can perform arbitrarily poorly in the real world. Such problems arise because people tend to make errors when they describe the real world. Human memory and human judgment are known to be fallible and prone to systematic errors known as cognitive biases (Tversky & Kahneman, 1974). Another fundamental limit to people's models of the real-world is uncertainty (Hertwig, Pleskac, & Pachur, 2019); there may be rare events and combinations of circumstances that the person specifying the model has never experienced before. Optimizing with respect to the resulting biased models might lead to strategies that fail in the real world.

To overcome this fundamental problem, this article proposes a robust approach to automatic strategy discovery that takes into account that the model of the environment might be incorrect. The resulting uncertainty about what the world might be like is modeled using Bayesian inference. Our approach computes the heuristic that performs best in expectation over all possible worlds that might have given rise to the provided specification. We thereby provide a first proof-of-concept for leveraging machine learning to discover robust heuristics in the face of uncertainty about the structure of the environment.

Our findings suggest that our methods are robust to uncertainty and errors in the model of the environment. The discovered heuristics tend to work well in the true environment even when the model of the environment was derived from a biased description of limited experience. This will be important for future efforts to derive clever heuristics from people's descriptions of the decisions they face in the real world.

The plan for this article is as follows: We start by introducing the theoretical and computational background of our approach; we then define the problem of robust strategy discovery; next, we present our solution to this problem and evaluate it in simulations; lastly, we apply the approach to improving human decision-making and discuss future directions.

Background

Robust strategy discovery builds on the definition of optimal heuristics advanced by Lieder and Griffiths (2020) and machine learning methods for deriving them automatically (Lieder et al., 2017; Callaway, Gul, et al., 2018). We study strategy discovery in the Mouselab-MDP paradigm (Callaway, Lieder, Krueger, & Griffiths, 2017) and incorporate robust strategy discovery into intelligent tutors that teach

people how to make better decisions. We briefly introduce each of these foundations in turn.

Resource-rational heuristics

Lieder and Griffiths (2020) recently introduced a new theory of bounded rationality that provides a realistic normative standard for human judgment and decision-making. Unlike previous normative theories, such as expected utility theory (von Neumann & Morgenstern, 1944), it takes into account that people's time and cognitive resources are bounded. Its prescriptions for good decision-making (Lieder & Griffiths, 2020; Lieder et al., 2017) thus, at least sometimes, resemble simple fast-and-frugal heuristics (Gigerenzer & Todd, 1999).

Lieder and Griffiths (2020) define the extent to which using the cognitive strategy h in an environment E constitutes effective use of the limited computational resources of the agent's brain B as the strategy's resource-rationality

$$\begin{aligned} \mathrm{RR}(h,E,B) = \mathbb{E}_{P(\mathrm{result}|s_0,h,E,B)}\left[u(\mathrm{result})\right] \\ - \mathbb{E}_{P(t_h,\rho,\lambda|h,s_0,B,E)}\left[\mathrm{cost}(t_h,\rho)\right], \end{aligned} \tag{1}$$

where $u(\mathrm{result})$ is the agent's subjective utility u of the outcomes (result) of the choices made by the heuristic h, $s_0 = (o, b_0)$ comprises the observed information about the initial state of the external world (o) and the agent's initial internal state b_0, and $\mathrm{cost}(t_h,\rho)$ denotes the total opportunity cost of investing the cognitive resources ρ used or blocked by the heuristic h for the duration t_h of its execution. Both the result of applying the heuristic and its execution time depend on the situation in which it is applied. The expected values ($\mathbb{E}$) weigh the utility and cost for each possible situation by their posterior probability given the environment E and the observed characteristics of the current situation (o).

How resource-rational people's strategies can be is constrained by the brain's computational limitations and uncertainty about the environment. That is, the set of cognitive strategies that the brain can execute (H_B) is limited and the extent to which people can adapt to their environment is constrained by the limited information i that they have about the environment (Lieder & Griffiths, 2020) . Under these constraints, the resource-rational heuristic is

$$h^\star = \arg\max_{h \in H_B} \mathbb{E}_{E|i}\left[\mathrm{RR(h,E,B)}\right]. \tag{2}$$

Automatic strategy discovery

Given a model of the environment, the resource-rational heuristic $h^\star$ for an agent with the computational resources B can be computed by reformulating the definition of the resource-rational heuristic as the solution to a metalevel Markov Decision Process (MDP) and applying methods from dynamic programming or reinforcement learning to compute its optimal policy. This approach models the decision process as series of computations that can be chosen one by one. Each computation updates the person's beliefs about the returns of alternative courses of action. Rules for selecting computations correspond to alternative decision strategies.

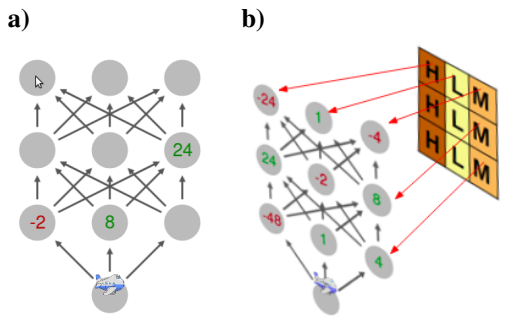

Figure 1: a) Screenshot of the Mouselab-MDP task as shown to participants. b) The environment can be represented as a grid of three types of nodes whose values are independently sampled from uniform distributions with high (H), medium (M), and low (L) variance respectively (i.e., $U(\{-48,-24,24,48\})$, $U(\{-8,-4,4,8\})$ and $U(\{-2,-1,1,2\})$.

Formally, a metalevel MDP is a four-tuple $M_{\mathrm{meta}} = (\mathcal{B}, \mathcal{C}, T_{\mathrm{meta}}, r_{\mathrm{meta}})$ comprising the set of possible beliefs $\mathcal{B}$ that the agent can have, the set of computational primitives $\mathcal{C}$, a probabilistic model $T_{\mathrm{meta}}(b,c,b')$ of how possible computations c might update the belief state (e.g., from b to b'), and the metalevel reward function r_{meta} which encodes the cost of computations $c \in \mathcal{C}$ and the utility of the action chosen when computation is terminated. In this formal framework, cognitive strategies correspond to metalevel policies ($\pi_{\mathrm{meta}} : \mathcal{B} \mapsto \mathcal{C}$) that specify which computation will be performed in a given belief state.

The Mouselab-MDP Paradigm

As it is not possible to observe human planning directly, the underlying cognitive processes must be inferred from people's behavior. This makes it difficult to study what strategies they discover, learn and use. Process tracing paradigms, such as the Mouselab paradigm (Payne, Bettman, & Johnson, 1988), present participants with tasks that make their behavior highly diagnostic of their unobservable cognitive strategies. The Mouselab-MDP paradigm (Callaway et al., 2017) is a process-tracing paradigm for measuring how people plan.

An example Mouselab-MDP environment used in this study is shown in Figure 1a. Participants are tasked to select one of several possible paths through a spatial environment, where each location harbors a reward. The participant's goal is to maximize the sum of the rewards along the chosen path. All of the rewards are initially concealed, but the participant can uncover them by clicking on the locations. Critically, each click has a cost of \$1. Thus, the participant has to trade the cost of collecting information off against the value of the collected information for making a better decision. Figure 1b) illustrates the statistical structure of one of the Mouselab-MDP task environments we used in this study.This environment is motivated to capture sequential nature of decision-making in real-life.

Cognitive tutors

Building on automatic strategy discovery, Lieder et al. (2019) developed an intelligent tutor that teaches people optimal decisions strategies in the Mouselab-MDP paradigm. Participants learned to use the automatically discovered strategies, remembered them, and used them in novel environments with a similar structure. These findings suggest that automatic strategy discovery can be used to improve human decision-making if the discovered strategies are well-adapted to the situations where people might use them.

The problem of robust strategy discovery

The challenge of robust strategy discovery is to derive strategies that work well in the real-world environments (e) from potentially incorrect information about what those environments are like (i). This information takes the form of a potentially misspecified model provided by a person who has experienced the real-world environment e but might misremember parts of their experience. We propose that this problem can be solved by developing probabilistic models of model misspecification, $P(i|e)$, invert them using Bayesian inference, and then train machine learning methods for automatic strategy discovery on the posterior distribution over possible true environments given the provided information, $P(e|i)$.

Modeling model misspecification

One approach to modeling model misspecification is to build on the previous literature on the systematic errors of human memory and human judgment known as cognitive biases (Tversky & Kahneman, 1974). For instance, people tend to remember more recent events better than earlier ones (Deese & Kaufman, 1957) and tend to underestimate the frequency of rare events in decisions from experience (Hertwig, Barron, Weber, & Erev, 2004).

As a first proof of concept, we worked with two admittedly simplistic models of how the recency effect and the underestimation of rare events affect how a person would describe a 3-step Mouselab-MDP environment they experienced by

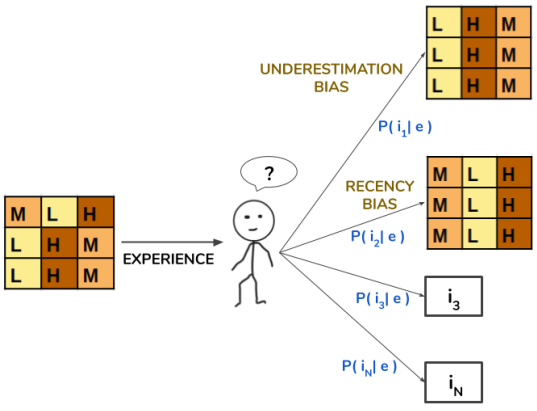

Figure 2: Illustration of how cognitive biases might give rise to misspecified models of the environment shown on the left.

$P(i|e,m_{\text{recency}})$ and $P(i|e,m_{\text{underestimation}})$. Figure 2 illustrates these two models. In brief, our model of the recency bias always misremembers the rewards in the first two steps (i.e., the two rows at the bottom) as having been identical to the reward in the last step (top row). Our model of the underestimation of the frequency of rare events always remembers the rare event in each column as the more frequent one.

For simplicity, we assume that half of the time model misspecification arises from the recency bias and other half of the time it arises from the underestimation of the frequency of rare events. We therefore model the probability that a person with experience in environment e will describe this environment by the specification i as

$$P(i|e) = \frac{1}{2}\cdot P(i|e,m_{\text{recency effect}}) + \frac{1}{2}\cdot P(i|e,m_{\text{underestimation}}) \tag{3}$$

These assumptions merely serve as a placeholder for a more realistic model of model-misspecification to be developed in future work. The contribution of this article is the general approach that combines the Bayesian inversion of such a model with automatic strategy discovery methods.

Benchmarks

To create a first set of benchmarks for robust strategy discovery, we build a data set $\mathcal{D}$ comprising 66 (e,i,p)-tuples where the environment e is a version of the task illustrated in Figure 1, the description i is generated according to the model of model-misspecification described above, and the probability $p = P(E=e)\cdot P(i|e)$ specifies the relative frequency of this pair in the set of benchmarks. There are 36 equally-probable true environments ($P(E=e) = 1/36$). They all share the property that each row contains one high (H), one medium (M), and one low (L) variance node. Furthermore, the arrangement of node types is the same in the bottom two rows. For each true environment e, we obtain 2 possible descriptions i - one generated according to the recency effect and the other from the underweighting of rare events. For six such environments, the resulting descriptions from the two biases turned out to be the same.

What makes these benchmark problems difficult is that many different true environments can give rise to the same specification. As a result, each description i could have plausibly been generated from 11 different true environments.

Measuring robustness

We define the robustness ρ of a strategy discovery method m as the expected performance of the algorithm $m(i)$ that it discovers based on the provided information i on the true environment e in expectation across all n benchmarks $\mathcal{D} = \{(e_1,i_1,p_1),\cdots,(e_n,i_n,p_n)\}$, that is

$$\rho(m) = \sum_{k=1}^{n} p_k \cdot \mathbb{E}\left[R|E=e_k,\pi=m(i_k))\right], \tag{4}$$

where the random variable R denotes the return that the strategy $m(i_k)$ will achieve when applied to a randomly-generated instance of the environment e_k.

To make the robustness score more interpretable, we normalize the expected benefit of the discovered strategy over choosing moves randomly without any planning ($\pi_{\text{no planning}}$) by the corresponding benefit of the strategy that is optimal for the true environment ($\pi^{\star}_{\text{meta},E}$), that is

$$\rho_{\text{rel}}(m) = \frac{\rho(m) - \mathbb{E}_E\left[\mathbb{E}_R\left[R|E,\pi_{\text{no planning}}\right]\right]}{\mathbb{E}_E\left[\mathbb{E}_R\left[R|E,\pi^{\star}_{\text{meta},E}\right] - \mathbb{E}_R\left[R|E,\pi_{\text{no planning}}\right]\right]}, \tag{5}$$

where the expectation $\mathbb{E}_E$ is taken with respect to the prior over environments. The best possible score is 1 (optimal performance in the true environment) and when the discovered heuristics perform at chance level in the true environment, then the method's relative robustness is 0.

Standard methods are not robust enough

Dynamic programming (DP) and standard RL methods like the Deep Q-Network (DQN) are not robust to model misspecification. For instance, if a person has recency bias and remembers only his last observations, they might erroneously claim that all of the high-variance nodes are located in the rightmost column because this is true in the top row (last step; see Figure 2). Applying DP or a DQN directly to this biased model would produce a strategy that only considers the right column (see Figure 3). But, this strategy is clearly suboptimal for the true environment where the structure of the top row is different.

In the next section, we develop strategy discovery methods that are robust to errors in the environment model. In this example, a robust method should produce a strategy that continues searching the top row for a high variance node when, contrary to the model, the value on the top-right is small.

The solution: ML methods for robust strategy discovery

Here we evaluate three approaches to achieving robustness: Handling uncertainty about the true environment according to Bayesian inference, meta-reinforcement-learning (Wang et al., 2016), and building adaptive inductive biases into the space of possible policies (Callaway, Gul, et al., 2018).

The Bayesian approach to robustness

According to Equation 2 the optimal heuristic given a specification i achieves the best possible cost-benefit trade-off in expectation across all possible environments. In this expectation, the heuristic's performance in each possible environment e is weighted by the posterior probability of that environment given the specification i. This suggests that strategy discovery algorithms can be made robust by applying them to samples from the posterior distribution $P(E|i)$ instead of applying them to the specification i itself. Therefore, our solution proceeds in two steps:

1. Estimate $\mathbf{P(E|i)}$. We use Bayesian inference to get a distribution over the possible true environments. The posterior distribution over possible environments is

$$P(E = e|i) = \frac{P(E = e) \cdot P(I = i|e)}{P(I = i)}, \tag{6}$$

where $P(E = e)$ is the prior over possible environments and the likelihood function $P(I = i|e)$ is a probabilistic model of model-misspecification (e.g., Equation 3). A good prior distribution should reflect the statistical structure of the environment so that each situation's prior probability corresponds to how often the situation tends to occur in the modeled environment. Since each possible situation occurs equally often in the set of benchmarks defined above, we chose a uniform prior. To accommodate complex likelihood functions that do not admit a closed-form solution, we approximate the posterior by training a neural network on simulated data where the ground truth is known.

2. Apply strategy discovery algorithms to samples from the posterior To generate a training set that encourages robust solutions, we sample the training environments from the posterior, that is $e_{\text{training},1}, \cdots, e_{\text{training},N} \sim P(E|i)$. Given sufficiently many samples from the posterior, standard reinforcement learning methods can be used to compute a robust policy π_{robust} by maximizing the average return across the MDPs defined by $e_{\text{training},1}, \cdots, e_{\text{training},N}$. Here, we evaluate how much this Bayesian approach increases the robustness of a standard deep reinforcement learning algorithm, a meta-RL algorithm, and metalevel reinforcement learning algorithm.

Algorithms

We consider four algorithms ranging from standard reinforcement learning methods to a newly developed metalevel reinforcement learning method. The first algorithm we evaluate is a deep Q-learning method that uses a recurrent network (Hausknecht & Stone, 2015). The second algorithm we evaluate is the metalevel reinforcement learning method introduced by Wang et al. (2016). This method learns to learn how to perform well in an initially known environment. This capacity might allow this method to learn different strategies for different types of environments and to adaptively select between them by first exploring the environment.

The third method we evaluated is the Bayesian metalevel Policy Search (BMPS) algorithm (Callaway, Gul, et al., 2018). It learns to approximate the value of computation by a linear combination of information-theoretic features. The weights of these features are learned using Bayesian optimization. The features of BMPS rely on a model of the environment. For the basic version of the BMPS algorithm we set this model to the specification i.

The robust version of BMPS performs online inference on the environment based on the description i and the belief state b that the agent has formed by interacting with the environment, that is $\mathbb{E}_{E|i,b}\left[\hat{Q}^{(\text{BMPS})}_{\text{meta},E}(b,c)\right]$. Since this approximation becomes computationally expensive when there are

many possible environments, we approximate this expected value by the normalized weighted average across the smallest set of possible environments $\mathcal{E}$ whose combined posterior probability p_{total} is at least $p_{\text{thresh}} = 0.99$, that is

$$\mathbb{E}_{E|i,b}\left[\hat{Q}^{(\text{BMPS})}_{\text{meta},E}(b,c)\right] \approx \frac{1}{p_{\text{total}}} \cdot \sum_{e \in \mathcal{E}} P(E = e|i,b) \cdot \hat{Q}^{(\text{BMPS})}_{\text{meta},e}(b,c). \tag{7}$$

Simulation results

As shown in Figure 4, we found that the BMPS algorithm with Bayesian inference on the true environment achieved an almost perfect relative robustness score ($\rho_{\text{rel}} = 0.99$, absolute score = 53.25) and outperformed all of the other methods (all $p < .0001$). The second-best method was meta-RL with Bayesian inference on the true environment ($\rho_{\text{rel}} = 0.91$, absolute score = 49.16). The addition of Bayesian inference on the true environment significantly improved the robustness of all methods: It improved the performance of BMPS from 42.47 ± 0.41 to 53.25 ± 0.42 ($t(47914) = -34.78$, $p < .001$; effect size $d = .318$) and had similar effects on the performances of meta-RL (35.54 ± 0.43 vs. 49.16 ± 0.41; $t(47894) = -44.60$, $p < .001$, $d = .408$) and DRQN (35.75 ± 0.42 vs. 46.26 ± 0.40; $t(47894) = -35.38$, $p < .001$, $d = .323$).

Figure 3, illustrates the behavior of two strategies discovered by the most robust method versus the least robust strategy discovery algorithm in two different scenarios.

In Example A, where the true environment matches the model, both algorithms make similar clicks. But in Example B, when the true environment differs from the model, the non-robust algorithm fails to uncover the high-variance nodes because it inflexibly follows the strategy that would have been optimal if the model were correct. By contrast, the robust strategy quickly adapts to the discrepancy between the model and the true environment and collects all of the most valuable information.

Application to improving human planning

To explore whether teaching the heuristics discovered by our robust strategy discovery methods is a viable approach to improving human decision-making, we leveraged the robust strategy discovery methods introduced above to develop a robust version of the cognitive tutor introduced by Lieder et al. (2019). This tutor uses our most-robust and best-performing strategy discovery algorithm, the robust BMPS algorithm, to discover a robust planning strategy from a description of the environment and then teaches it to people by showing them video demonstrations of its planning behavior.

To evaluate how beneficial it is for people to be trained by the robust tutor, we conducted a behavioral experiment. For simplicity, we conducted this behavioral experiment in the same setting we simulated above. That is, for each participant we sampled one of the 66 benchmark problems defined above according to their respective probabilities; for instance, the

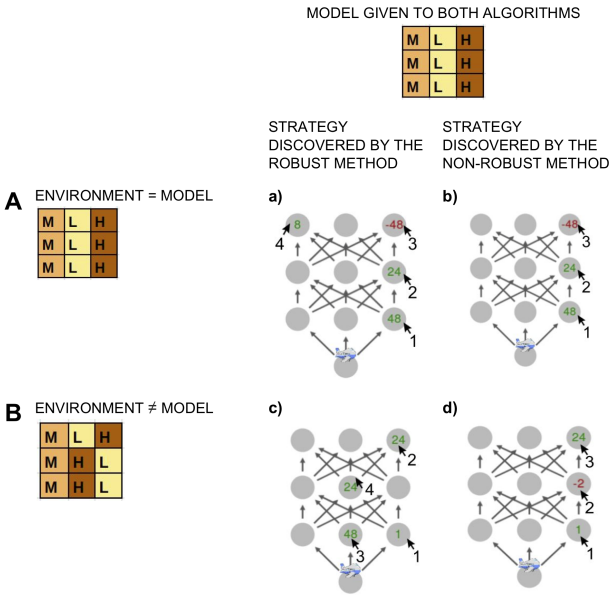

Figure 3: Comparison of the planning strategies discovered by a robust algorithm (BMPS with Bayesian inference) versus a non-robust algorithm (DRQN without Bayesian inference). Example A illustrates the strategies in the specified environment. Example B illustrates their behavior in a different environment that could have given rise to the same specification. Critically, the planning strategy discovered with the robust method performs well in both environments whereas the strategy discovered with the non-robust strategy fails when the true environment does not match the specification.

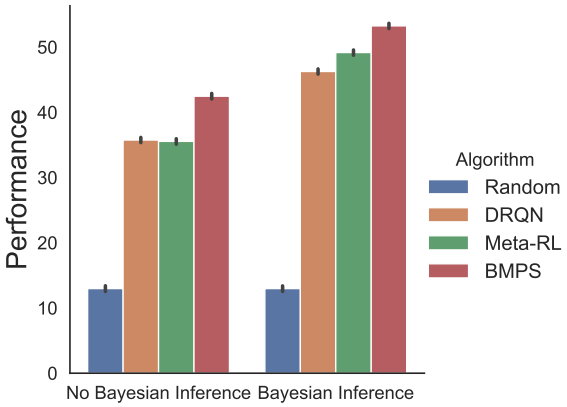

Figure 4: Performance of different strategy discovery algorithms with versus without Bayesian robustness.

benchmark problem (e_k, i_k, p_k) would be sampled with probability p_k. The participant was assigned the role of the novice who is being trained by the robust cognitive tutor and then tested on the true environment e_k. Critically, the cognitive tutor does not know the true environment e_k but only the usually

erroneous description i_k. The robust tutor infers what the true environment might be given the description ($P(E|i_k)$), derives the optimal strategy from this probabilistic knowledge, and then demonstrates it on environments sampled from its posterior distribution over possible environments ($P(E|i_k)$). We compared the robust tutor against having people perform the task without tutoring and a conventional cognitive tutor that assumes that i_k is the true environment (non-robust tutor).

Participants and procedure

We recruited 300 participants on Amazon Mechanical Turk (average age 38.9 years, range: 19–72 years; 152 female). Participants were paid $1.20 plus a performance-dependent bonus (average bonus $1.45). The average duration of the experiment was 9.8 min. Participants were randomly assigned to the control condition without tutoring (100 participants), the experimental condition with the non-robust cognitive tutor (99 participants), or the experimental condition with the robust cognitive tutor (101 participants). All three groups went through 5 practice trials to get used to the environment and 15 test trials. Additionally, in the two experimental conditions, participants were shown 10 tutor demonstrations between the practice trials and the test trials. To ensure high data quality, we applied two pre-determined exclusion criteria. We excluded the 3% of participants who affirmed that they had not paid attention to the instructions or had not tried to achieve a high score in the task. We excluded 15% of the remaining participants who did not make a single click on more than half of the test trials because not clicking is highly indicative of speeding through the experiment without engaging with the task.

The experimental task is based on the Mouselab-MDP paradigm (Callaway et al., 2017). The experiment was structured into instructions that introduced the Mouselab-MDP paradigm, a quiz that tested people's understanding of the paradigm, a training block (only in the experimental conditions), and a test block. Participants were randomly assigned to one of three conditions (2 experimental conditions and 1 control condition). In the training block of the two experimental conditions, each participant was shown a series of 10 demonstrations of an automatically discovered strategy (see Figure 3a-c). Each demonstration started from a different fully occluded instance of the environment illustrated in Figure 1a. The demonstration then showed the participant the first click that the automatically discovered strategy would make and the reward that it revealed. After a 1.1 seconds delay the demonstration showed the second click that the strategy would make based on the outcome of the first click. This continued until the strategy decided to terminate planning. At this point, the participant was shown the sequence of moves that the strategy would choose and the rewards collected along the way. In the first experimental condition, the demonstrations showed the strategies that the DRQN method without Bayesian inference derived from the potentially misspecified models (Non-Robust Tutor). These strategies always click on the same three nodes that should have high variance according to the models regardless of what their values are (see Figure 3b and d). In the second experimental condition, the demonstrations showed the strategies discovered by BMPS with Bayesian inference (Robust Tutor). When a high variance node is not in its expected location, then those robust strategies continue to search for it until they find it (see Figure 3a and c). In the Non-Robust Tutor, all demonstrations were performed on the reward structure specified by the model, as illustrated in Figure 3b. By contrast, in the Robust Tutor the reward structures were sampled from the posterior distribution over the true environment given the model specification; thus, some demonstrations were performed on environments that differed from the model as illustrated in Figure 3c. To motivate participants to pay close attention to these demonstrations, they were told that their bonus would depend on correctly answering a quiz about the demonstrated strategy and were given the option to review the demonstrations before moving on to the quiz. In the control condition there was no training block and participants were *not* shown any demonstrations.

Results

A Kruskal-Wallis ANOVA showed that the three groups differed significantly in their performance on the test trials ($H = 15.3$, $p < .001$). Planned pair-wise comparisons confirmed that teaching people strategies discovered by the robust method significantly improved their performance (39.5 points/trial) compared to the control group (29.9 points/trial, $t(162) = 3.75$, $p = .001$, $d = .587$). By contrast, teaching strategies discovered by the non-robust method failed to improve people's performance (32.9 points/trial, $t(157) = 1.20$, $p = .265$, $d = 0.178$) and led to significantly lower performance than teaching strategies discovered by the robust method ($t(167) = 2.75$, $p = .007$, $d = .425$). Each person's score in the Mouselab-MDP task is the sum of the rewards they collected minus the cost of their clicks. We can therefore interpret it as a measure of how well their strategy trades off the quality of the resulting decisions with the cost of decision-making. To make the scores more interpretable, we compute each group's resource-rationality quotient ($^{\mathrm{RR}_{\mathrm{people}}}/_{\mathrm{RR}_{h^\star}}$ where $\mathrm{RR}_{\mathrm{people}}$ is the group's average score). As shown in Figure 5, teaching people strategies discovered by the robust method brought their performance closer to the performance of the resource-rational heuristic for the test environment. Concretely, people's resource-rationality quotient increased from 56.8% in the control group to 73.6% in the robust tutor group and to only 61.5% in the non-robust tutor group.

These differences in performance reflect differences in the underlying planning strategies. Inspecting the planning strategies that participants used in the test block showed that participants who had been taught by the robust tutor inspected the values of all of the most informative high-variance nodes on 62.0% of the trials whereas participants in the non-robust tutor condition or the control condition did so significantly less often (41.3% and 46.7%, respectively, $\chi^2(2) = 117.3$, $p < .001$).

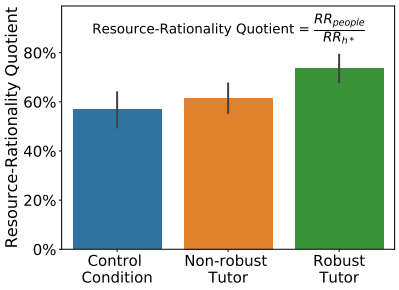

Figure 5: Resource-rationality quotient by condition.

Discussion

Discovering and teaching clever heuristics is a promising approach to improving human decision-making. Previous work has suggested that it might be possible to leverage machine learning to automate the process of strategy discovery if the method is given a highly accurate model of the decision problems to be solved. But uncertainty and cognitive biases put a fundamental limit on how accurate we can expect people's descriptions of real-life scenarios to be. With previous methods, even small errors in the model of the decision environment can lead to strategies that perform very poorly in the real-world. In this article, we presented robust strategy discovery methods that overcome this problem by taking into account that the model might be a biased description of reality. Our simulations show that the developed methods are significantly more robust to model misspecification than previous methods and can discover strategies that tend to work very well in the true environment even when the provided model is compromised by cognitive biases. The findings of our behavioral experiment show that robust strategy discovery methods can allow us to improve human-decision-making in cases where non-robust methods fail. One limitation of the present work is that it assumed a perfect model of the biases in the generation of model specifications. The methods introduced in this article are an important step towards leveraging automatic strategy discovery to improve human decision-making in the real world. The findings of our first proof-of-concept case-study show that uncertainty about the real-world and the errors that experts may commit when describing their domain do not have to hold us back from developing intelligent cognitive tutors that automatically discover and teach clever strategies that enable people to make better decisions.

The progress reported in this article opens up several exciting avenues for future work. One line of future work is to apply robust strategy discovery methods to human-generated descriptions of decisions they face in the real world. To support this application, we will refine our method's model of people's cognitive biases based on empirical data. In a related line of work, we will apply robust strategy discovery to increasingly more realistic decision-problems, such as online shopping and investing, and investigate transfer to decision-making in the real world. As the decision problems and strategies become more complex, it might become challenging for people to grasp them from seeing demonstrations alone. To overcome this challenge, we will develop cognitive tutors that combine demonstrations with automatically discovered interpretable descriptions of the demonstrated strategies (Skirzynski, Becker, & Lieder, 2020) and give people feedback on their attempts to apply the taught strategy to practice problems.

References

Callaway, F., Gul, S., Krueger, P., Griffiths, T. L., & Lieder, F. (2018). Learning to select computations. In *Uncertainty in artificial intelligence: Proceedings of the thirty-fourth conference.*

Callaway, F., Lieder, F., Das, P., Gul, S., Krueger, P., & Griffiths, T. (2018). A resource-rational analysis of human planning. In C. Kalish, M. Rau, J. Zhu, & T. Rogers (Eds.), *CogSci 2018.*

Callaway, F., Lieder, F., Krueger, P. M., & Griffiths, T. L. (2017). Mouselab-MDP: A new paradigm for tracing how people plan. In *The 3rd Multidisciplinary Conference on Reinforcement Learning and Decision Making, Ann Arbor, MI.* Retrieved from `https://osf.io/vmkrq/`

Deese, J., & Kaufman, R. A. (1957). Serial effects in recall of unorganized and sequentially organized verbal material. *Journal of experimental psychology*, *54*(3), 180.

Gigerenzer, G., & Todd, P. M. (1999). *Simple heuristics that make us smart*. Oxford University Press.

Hausknecht, M., & Stone, P. (2015). Deep recurrent Q-learning for partially observable MDPs. In *2015 aaai fall symposium series.*

Hertwig, R., Barron, G., Weber, E. U., & Erev, I. (2004). Decisions from experience and the effect of rare events in risky choice. *Psychological science*, *15*(8), 534–539.

Hertwig, R., & Grüne-Yanoff, T. (2017). Nudging and boosting: Steering or empowering good decisions. *Perspectives on Psychological Science*, *12*(6), 973–986.

Hertwig, R., Pleskac, T. J., & Pachur, T. (2019). *Taming uncertainty*. Cambridge, MA: MIT Press.

Lieder, F., Callaway, F., Jain, Y., Krueger, P., Das, P., Gul, S., & Griffiths, T. (2019). A cognitive tutor for helping people overcome present bias. In *RLDM 2019.*

Lieder, F., & Griffiths, T. L. (2020). Resource-rational analysis: understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences*, *3*, 1–85.

Lieder, F., Krueger, P. M., & Griffiths, T. L. (2017). An automatic method for discovering rational heuristics for risky choice. In G. Gunzelmann, A. Howes, T. Tenbrink, & E. Davelaar (Eds.), *Cogsci 2017.*

Payne, J. W., Bettman, J. R., & Johnson, E. J. (1988). Adaptive strategy selection in decision making. *Journal of experimental psychology: Learning, Memory, and Cognition*, *14*(3), 534.

Skirzynski, J., Becker, F., & Lieder, F. (2020). Automatic discovery of interpretable planning strategies. *arXiv preprint arXiv:2005.11730.*

Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*, *185*(4157), 1124–1131.

von Neumann, J., & Morgenstern, O. (1944). *The theory of games and economic behavior*. Princeton, NJ: Princeton University Press.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., ... Botvinick, M. (2016). Learning to reinforcement learn. *CogSci 2016*.