Theory-Based Learning in Humans and Machines

by

Pedro A. Tsividis

Submitted to the Department of Brain and Cognitive Sciences
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2019

Signature redacted

Author .....
Department of Brain and Cognitive Sciences
December, 2018

Signature redacted

Certified by.....
Joshua B. Tenenbaum
Professor of Computational Cognitive Science
Thesis Supervisor

Signature redacted

Accepted by ..
Matthew A. Wilson
Chairman, Department Committee on Graduate Theses

Theory-Based Learning in Humans and Machines

by

Pedro A. Tsividis

Submitted to the Department of Brain and Cognitive Sciences
on December, 2018, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Humans are remarkable in their ability to rapidly learn complex tasks from little experience. Recent successes in AI have produced algorithms that can perform complex tasks well in environments whose simple dynamics are known in advance, as well as models that can learn to perform expertly in unknown environments after a great amount of experience. Despite this, no current AI models are able to learn sufficiently rich and general representations so as to support rapid, human-level learning on new, complex, tasks.

This thesis examines some of the epistemic practices, representations, and algorithms that we believe underlie humans' ability to quickly learn about their world and to deploy that understanding to achieve their aims. In particular, the thesis examines humans' ability to effectively query their environment for information that helps distinguish between competing hypotheses (Chapter 2); children's ability to use higher-level amodal features of data to match causes and effects (Chapter 3); and adult human rapid-learning abilities in artificial video-game environments (Chapter 4). The thesis culminates by presenting and testing a model, inspired by human inductive biases and epistemic practices, that learns to perform complex video-game tasks at human levels with human-level amounts of experience (Chapter 5). The model is an instantiation of a more general approach, Theory-Based Reinforcement Learning, which we believe can underlie the development of human-level agents that may eventually learn and act adaptively in the real world.

Thesis Supervisor: Joshua B. Tenenbaum
Title: Professor of Computational Cognitive Science

Acknowledgments

This work would not have been possible without the direct and indirect support of a great many people; I take this opportunity to thank them.

First, I'd like to thank my advisor, Josh Tenenbaum. Your creativity, energy, and dedication to your students are an inspiration. Thanks for all the meetings over coffee or food, phone calls, video chats, and texts :). Having continual contact with you over the course of this project was really fun, and a real honor.

Thank you to Laura Schulz, for advising me in my early years at MIT, and for being patient with me as I figured out what I wanted to work on. Our early conversations deeply influenced how I think about intelligence; I dream to one day actually show you a satisfying explanation of error maps.

Thank you to Leslie Kaelbling and Michael Littman, for several fun and inspiring conversations over the last couple years, and for serving on my committee.

Thank you to Sam Gershman, for being both a friend and a mentor, and for seamlessly switching between the two.

Thank you to everyone who worked on or influenced the EMPA project: Mario Belledonne, Aritro Biswas, Jake Burga, Nate Foss, Mark Goldstein, Michael Janner, João Loula, Tim Menke, Eshaan Nichani, Thomas Pouncy, Eric Wu, Kevin Wu, Jackie Xu, Zhenglong Zhou.

Thanks to Max Siegel, for formative conversations early on in grad school; thanks to Ilker Yildirim and Kevin Smith for their last-minute pre-defense feedback; cheers to Owen Lewis and Eyal Dechter, and our several half-completed projects.

Thanks to COCOSCI and ECCL for good conversations, feedback, and fun.

Thank you to the many friends that made my time here so special: Max Siegel, João Loula, Thomas Pouncy, Julia Leonard, Hilary Richardson, Sam Norman-Haignere, Jorie Koster-Hale, Will Whitney, Max Kleiman-Weiner, Tejas Kulkarni, Rishi Rajalingham, Sophia Sanborn, Alex Kell.

Finally, I would like to thank my parents, Felicia and Yannis, for being the best parents I can imagine.

Contents

1 Introduction **15**

2 Information Selection in Noisy Environments with Large Action Spaces **21**

2.1 Introduction 21

2.2 An Information-search Task 22

2.3 Model 24

2.4 Experiment 1 27

2.4.1 Method 27

2.4.2 Results 28

2.5 Experiment 2 30

2.5.1 Method 30

2.5.2 Results 31

2.5.3 Model comparisons 33

2.6 Discussion 35

3 Hypothesis-space Constraints in Causal Learning **37**

3.1 Introduction 37

3.2 Distributional Properties 39

3.2.1 Methods 39

3.2.2 Results 41

3.3 Dynamic Properties 42

3.3.1 Methods 43

3.3.2 Results 46
3.4 Discussion 47

4 Human Learning in Atari **51**
4.1 Introduction 51
4.2 Gameplay 52
4.2.1 Methods 52
4.2.2 Learning Curves 53
4.3 Experimental Manipulations 54
4.3.1 Obscuring Object Identity 55
4.3.2 Reading the Instruction Manual 56
4.3.3 Learning from Observation 58
4.3.4 Results 58
4.4 Discussion 60

5 Theory-Based Reinforcement Learning in Games **61**
5.1 Introduction 61
5.2 Related Work 62
5.3 Explore, Model, Plan: Overview 64
5.4 Model 68
5.4.1 Representation 68
5.4.2 Bayesian Learning 71
5.5 Theory-Based Exploration 73
5.6 Planning 75
5.6.1 Overview 75
5.6.2 Planning modes 75
5.6.3 Intrinsic Rewards 76
5.6.4 State Pruning 77
5.6.5 Re-planning 78
5.7 Towards Learning Theories Without Fully Observable Events 78
5.7.1 Main Loop 79

5.7.2 Likelihood Function and Distance Metric 79
5.7.3 Error Signal and Theory Proposal 80
5.7.4 Error Signal 81
5.7.5 Factorized Proposals 82
5.8 Human-Level Theory-Based Reinforcement Learning 83
5.8.1 Results 85
5.8.2 Role of theory-based curiosity 87
5.8.3 Role of planning heuristics 89
5.9 Discussion 92

6 Conclusion **95**

A Figures **99**

B Tables **105**

C Algorithms **109**

D Game Descriptions **113**

List of Figures

2-1 A screenshot of the task. The four green rectangles are the scanners. The player has scanned three times and received one positive and two negative answers, as shown by the blue and red scanner outlines on the number line. 23

2-2 Analysis of one participant's four sequential actions in the information-search game. All the scanners were of reliability=0.87 and varied only in length. **A**: The participant uses the length=50 scanner and receives a response of 'yes'. The arcs correspond to the scaled information gain of each candidate action; note that the participant selects one of the highest-rated actions. This fact is also indicated in the insert, which shows the posterior-predictive entropies of all candidate actions (lower is better); the red dot indicates the posterior-predictive entropy of the action chosen by the participant. **B**: The participant employs the 'split-half' heuristic to test half the remaining space. Note that all of the highest-ranked model recommendations involve testing exactly half of the highest-posterior-probability region. **C**: The participant continues to use 'split-half'. **D**: The participant re-tests the entire region of highest posterior probability. This is highly rated, but not as much as it would be had they tested half the remaining space. 26

2-3 **A**: The 'mixed' condition reveals that participants are able to select the best scanner across a wide variety of reliability/length trade-offs. **B**: The proportion of times participants chose the most informative scanner (x axis) versus predictions of the *EIG* model (y axis), for each of the 8 sub-conditions in the 'mixed' condition. 32

2-4 In their overall query choices (combining scanner scope, reliability, and location), participants are highly sensitive to expected information gain; note also the tight fit of the *EIG* model ($\rho = 0.994$). 33

2-5 Log likelihoods assigned by each model to participants' joint choices of scanner type and placement. 34

3-1 Schematic of seeds and actual flower fields used. Left: 1:1. right: 7:1. 40

3-2 Experiment 1 results (N=16): Mean trials correct = 1.62. Probability of full success (2/2 trials) is .625; 95% confidence interval: $[.354, .848]$ (chance: .25). $p < .01$ by two-tailed binomial test. 42

3-3 'Lights' used for experiment 2. Top: periodic. Bottom: monotonic. . 44

3-4 Periodic (green) and monotonic (purple) bugs from stimulus set 2. The green bugs move at constant speed but increase and decrease in number of spots; the purple bugs increase speed monotonically, as indicated by the increasing length of the vectors. 45

3-5 Experiment 1 results (N=32): Mean trials correct = 1.375. Probability of full success (2/2 trials): .594, 95% confidence interval: $[.406, .763]$ (chance: .25). $p < .001$ by two-tailed binomial test. 47

4-1 Human learning curves for twelve Atari games. Black horizontal line: random play. Green horizontal line: 'expert' play. Red horizontal dashed line: DDQN after 46 hours of game-play experience. Red dotted line: DDQN after 920 hours. 53

4-2 Learning rate comparison. X-axis: DDQN experience, in millions of frames. Y-axis: Rate of improvement in log (points per minute), estimated using finite differences. Human rate of improvement is taken from score-matched points (shown as numbers annotating the curves). DDQN estimates are made from Figure 7 in Schaul et al. (2015). . . 55

4-3 Performance on the game of Frostbite shows the striking degree to which humans outperform existing AI models in learning how to play video games. 55

4-4 Screenshot of a blurred version of Frostbite. The light blue items are birds; the green items are fish. 56

4-5 Screenshot of the original Atari 2600 instructions to Frostbite, displayed to participants. 57

4-6 First-episode mean scores and 95% confidence intervals for normal, 'blurred', 'instructions', and 'observation' conditions. If semantics contributed significantly to human performance, blurring would have produced a decrement in first-episode performance; instead, this intervention makes no significant difference. By contrast, reading the instructions and observing another player each allow humans to capture approximately 1000 points in their first episode – this corresponds to human performance after about 5 minutes of play under normal conditions. 59

5-1 Schematic representation of the EMPA architecture. EMPA takes as input a symbolic description of its environment, specified in terms of objects and their locations as well as interactions that occur between objects. Bayesian inference scores theories on their ability to explain observed state sequences. Theory-based curiosity generates exploratory goals. The planner decomposes exploratory and win-related goals into subgoals and goal gradients, and uses this hierarchical decomposition to effectively find high-value actions for EMPA to take in the game environment. 65

5-2 A: Early-stage game screen showing contributions of the EMPA architecture. Explore (white): agent's epistemic goals. Model (pink): an early hypothesis about predicted paths of objects. Plan (yellow): The planner finds high-value trajectories according to the agent's current theory and its epistemic goals. B: A mid-stage game screen. The agent has learned that cars are dangerous, but is still curious about logs, water, and the flag. The planner produces paths that reach mid-range epistemic goals involving contact with plank and water objects. C: A late-stage game screen, at which point the agent has interacted with all objects except for the flag object. In yellow is the planner's suggested path. Upon interacting with the object the agent will win the level and learn a termination condition for the game. D: The theory of a game is invariant to permutations in object positions, and therefore automatically generalizes to new levels. Here the planner generates a path (orange) toward the known goal on a new level, using the same theory from the previous level. The agent is also automatically curious about the newly-occurring object in the game, and the planner suggests a path to that (yellow), as well. 66

5-3 Screenshots from Zelda, Frogs, and Bait. Learning how to win these (and many other) games requires reaching objects that are difficult to reach using a naive exploration policy. In order to achieve the first win, the agent must pick up the key and then reach the door while avoiding spiders (Zelda, left); reach the goal flag by first crossing a dangerous road filled with cars and then crossing a river by stepping on moving logs (Frogs, middle); solve a complex puzzle by pushing boxes into holes to clear space in order to reach the key (Bait, right). EMPA is able to learn about all of these win-related objects rapidly precisely because it specifies reaching them as goals, rather than waiting to reach them by chance in order to realize that the objects are, in fact, important to winning the games. 74

5-4 'Normal' and 'no-color' versions of the GVGAI game, "Frogs". All human participants and DDQN played 'no-color' versions; EMPA received a state description containing object positions and interactions occurring between objects. 84

5-5 Screenshots and learning curves for six games, for humans (green), EMPA (blue), and DDQN (grey). Humans and EMPA are able to learn new games in a matter of minutes (corresponding to a few hundred steps). 86

5-6 Distribution of EMPA and DDQN learning performance relative to humans across 90 games. 87

5-7 EMPA learning performance relative to humans and DDQN across 90 games. 88

5-8 Distribution of performance scores for DDQN, EMPA, and four variant exploration lesions of EMPA. Certain games pose significant exploration problems that neither DDQN nor the lesioned variants can solve. 90

5-9 Means and 95% CIs (computed in log space) of performance scores across all 90 games. 90

5-10 Means and 95% CIs (computed in log space) of performance scores across the 31 games for which all variants were able to solve at least one level. For games without significant exploration challenges, ϵ-greedy exploration contributes only slightly to a decrement in EMPA's performance. DDQN continues to significantly underperform humans and EMPA. 91

5-11 The combination of EMPA's intrinsic reward structure and node-pruning allows it to perform up to 1000 times better than a lesioned variant that lacks these components. 91

5-12 The combination of EMPA's intrinsic reward structure and node-pruning allow it to plan roughly twice as efficiently than a lesioned variant that lacks these components. Note that this metric only considers planner nodes per step taken, and not the quality of the actions. 92

A-1 Mean and 95% confidence intervals of each game's difficulty as rated by human subjects, grouped by source game. 100

A-2 Mean and 95% confidence intervals of human subjects' interestingness judgments for each game, grouped by source game. 101

A-3 Humans (green), EMPA (blue), and DDQN learning curves over the initial $10,000$ steps of play. 102

A-4 Humans (green), EMPA (blue), and DDQN learning curves over the initial 1 million steps of play. 103

B-1 Planning-mode selection policy. "repeated-deaths" examines the state history and returns true if the same events occurred during successive lost episodes. "no-new-objects(N)" returns TRUE if no new object has emerged on the screen for N game-steps. "no-scorechange" returns TRUE if the score has not changed in the last game step. 105

B-2 Parameters that constitute EMPA's distinct planning modes. 106

B-3 DDQN hyperparameters. 107

Chapter 1

Introduction

Humans are remarkable in their ability to rapidly learn complex tasks from little experience. Recent successes in AI have produced algorithms that can perform complex tasks well in environments whose simple dynamics are known in advance (Silver et al., 2016; Brown and Sandholm, 2018), as well as models that can learn to perform expertly in unknown environments after a great amount of experience (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017). Despite this, no current AI models are able to learn sufficiently rich and general representations so as to support rapid, human-level learning on new, complex, tasks.

This thesis examines some of the epistemic practices, representations, and algorithms that we believe underlie humans' ability to quickly learn about their world and to deploy that understanding to achieve their aims. In particular, it examines humans' ability to effectively query their environment for information that helps distinguish between competing hypotheses; children's ability to use higher-level amodal features of data to match cause and effects; and adult human rapid-learning abilities in artificial video-game environments. The thesis culminates by presenting and testing a model, inspired by human-like inductive biases and epistemic practices, that learns to perform complex video-game tasks at human-level performance and with human-level amounts of experience. The model is an instantiation of a more general approach, Theory-Based Reinforcement Learning, which we believe can underlie the

development of human-level agents that may eventually learn and act adaptively in the real world.

Many decades of research in cognitive science have shown that humans have early-arising "start-up" software – rich representations about objects and physics (Spelke, 1990; Baillargeon, 2004; Baillargeon et al., 2009; Rips and Hespos, 2015) and about agents (Johnson et al., 1998; Tremoulet and Feldman, 2000; Csibra et al., 2003; Schlottmann et al., 2006; Spelke and Kinzler, 2007; Csibra, 2008; Kiley Hamlin et al., 2013), and that humans have the capacity to rapidly acquire new concepts (Carey, 1978; Landau et al., 1988; Markman, 1989; Bloom, 2000; Xu and Tenenbaum, 2007; Lake et al., 2015) and build intuitive theories (Murphy and Medin, 1985; Carey, 1985; Gopnik and Meltzoff, 1997) — abstract, coherent, causal, ontologically-committed frameworks which they can use to explain (Lombrozo, 2009; Williams and Lombrozo, 2010), predict Rips (1975); Murphy and Ross (1994), and imagine (Ward, 1994; Jern and Kemp, 2013).

In addition to being born with the right inductive biases, humans, and in particular, children, learn rapidly because they employ powerful epistemic practices (see Gopnik and Wellman (2012); Schulz (2012); Tenenbaum et al. (2011) for reviews). Children rationally infer causal relationships from statistical evidence (e.g., Gopnik et al. (2004)), selectively explore when evidence is confounded or surprising (Schulz and Bonawitz, 2007; Bonawitz et al., 2012), evaluate the relationship between samples and populations (Denison and Xu, 2010; Gweon et al., 2010; Xu and Denison, 2009), infer the existence of unobserved variables to explain anomalous data (Schulz et al., 2008), isolate candidate causes in order to distinguish between competing hypotheses (Cook et al., 2011; van Schijndel et al., 2015), and effectively search through hypothesis spaces for information (Nelson et al., 2014). Adults, of course, can also effectively intervene on causal systems to learn their structure (Lagnado and Sloman, 2004; Sloman and Lagnado, 2005; Waldmann et al., 2006) and can use use optimal information-gain strategies when doing so and when asking questions, more generally (Nelson, 2005; Steyvers et al., 2003; Rothe et al., 2017). As humans explore the world and its affordances, they must ask questions that have scope, in that they rule

out large numbers of hypotheses at once, and they must also minimize query reliable sources in order to obtain useful information. **Chapter 2** addresses whether humans continue to follow information-gain imperatives in when confronted with confronted with large and noisy information spaces.

Chapter 3 addresses the question of internal search. When humans do seek data, they do so in order to address a hypothesis space that is already limited to "reasonable" hypotheses; that is, the hypothesis space of all possible answers has already been constrained by an internal search. We suspect that human learners do this by using certain types of abstract heuristics, and here we examine whether children — our best learners — can use such heuristics to select between different hypotheses in the absence of informative evidence.

Abstract heuristics may be especially important in the setting of causal learning. One may have the goal of identifying a causal mechanism responsible for something that blinks on and off. Given a choice between a mechanism like a doorbell or a mechanism like a pulley, one might favor the former; given an effect whose outcome space is discrete, a candidate cause with discrete outcomes may seem preferable to one that has continuous outcomes. Of course, nothing guarantees that this inference is correct, but in the absence of other information, it is a reasonable strategy for narrowing down the hypothesis space. An initial test of this general idea showed that four- and five-year-olds were sensitive to abstract properties relating the form of candidate causes and effects. In a series of experiments, children successfully mapped discrete causes to discrete effects and continuous causes to continuous effects (Magid et al., 2015).

If priors about causal processes enable learners to select good hypotheses in the absence of covariation data or content-specific prior knowledge, what form should these priors take? If they are to be effective across a variety of problems, these priors cannot be about lower-level cognitive features such as color, height, pitch, etc., since causal relations do not often preserve these. That is, color changes are not usually caused by other colors; pitch changes are not caused by other pitch changes; and so on. More importantly, the space of possible mappings between specific low-level

features is too large for an approach reliant on specific priors about such mappings to be efficient. However, certain higher-level amodal features such as extent, rate, arity, distributional properties, and dynamics, are invariant to the lower-level features, allowing for more relevant and more efficient comparisons. We propose that children are sensitive to these higher-level features and can use them to match effects with their causes. Our experiments show that children are indeed sensitive to at least two of these: distributional properties and dynamics.

Chapters 4 and **5** broaden the scope of our inquiries to the problem of learning to behave adaptively in unknown environments. Recently there has been great interest in the idea of building artificial agents that learn to play video games, as these serve as readily available microcosms for the development of algorithms that may eventually learn to function in the real world.

This line of research has been hugely successful: reinforcement learning algorithms using deep neural networks have surpassed human-level performance on games from the classic Atari 2600 platform (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017). The original Deep Q-learning Network (Mnih et al., 2015) used a a deep convolutional neural net trained through stochastic gradient descent to approximate the environment's optimal action-value function, $Q(s, a)$, and variants of this algorithm have achieved increasing asymptotic performance as well as increasing data efficiency by varying each of several relevant components: the Double DQN (Van Hasselt et al., 2016) addresses an overestimation bias caused by DQN's loss function, and Prioritized Experience Replay (Schaul et al., 2015) samples transitions for replay as a function of their last-encountered TD error, thereby more frequently sampling transitions from which there is more to learn (but also sampling more from stochastic transitions where there may no longer be anything to learn). The A3C algorithm (Mnih et al., 2016) executes multiple agents in parallel, thereby decorrelating their experience, and estimates both a value function, $V(s)$ and a policy, $\pi(s)$. These and other methods turn out to be largely complementary; Rainbow (Hessel et al., 2017) combines them to produce state-of-the-art performance on the Atari

benchmark, eventually reaching 210% of human performance across the 57 benchmark games. However, even a cursory examination of the behavior of these artificial agents reveals that they learn incredibly slowly (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017; Kansky et al., 2017) and that they generalize poorly to trivial variations of a given environment (see the experiments reported by Kansky et al. (2017)).

Our experiments show that humans are able to learn these tasks in a matter of minutes, suggesting that humans and leading AI algorithms employ different representations and learning mechanisms. In addition to bringing early-arising representations to bear on Atari tasks, humans come equipped with rich prior knowledge about the world – for example, knowledge about keys, doors, ice, birds, and so on. While such knowledge could give humans an edge over AI algorithms, we believe that it plays a minimal role in humans' ability to rapidly master these tasks. In **Chapter 4** we conduct a systematic analysis of human performance on these tasks and show data suggesting that it is not semantic knowledge but rather abstract theory-like knowledge that enables rapid learning.

In **Chapter 5** we present a novel approach, Theory-Based Reinforcement learning, that enables rapid learning in novel environments through the use of human-like "intuitive theories". We instantiate the approach in our Exploring, Modeling, Planning agent (EMPA). This agent uses priors about objects as spatiotemporally contiguous entities whose properties can be learned from experience; an imperative to explore these objects and to observe available evidence to rapidly build theory-like models of the game worlds it encounters; and an ability to use these models to simulate possible future worlds and generate effective plans. We compare our model's performance to human performance on a set of 90 challenging video games and show that our model is able to learn on human timescales and far outperforms a representative deep-RL model (DDQN).

Chapter 2

Information Selection in Noisy Environments with Large Action Spaces

2.1 Introduction

As scientists, we often encounter (or conduct) experimental work that is stunning in its breadth but disappointing in its rigor, or work that is categorically decisive but disappointingly narrow in scope. As child or adult intuitive scientists searching for information, we often deal with these two dimensions — scope and rigor — as well; we can make general queries that drastically narrow the hypothesis space of answers or make narrower ones, and we can seek information from reliable sources that are more likely to give us correct answers, or from sources that are less so. Our success, whether as professional or intuitive scientists, hinges on our ability to balance these two dimensions in order to produce queries whose answers will be informative.

Early work on information search seemed to show that people fail to make rational decisions when it comes to information acquisition; in the Wason (1968) selection task, only 4% of subjects made the normative selection in a task with a fairly small (12) action set. However, as Oaksford and Chater (1994) pointed out, the selection

made most often by participants in the original Wason task was normative when environmental statistics were taken into account. Specifically, participants' decisions were best explained as maximizing expected information gain, in the service of helping them to decide between competing hypotheses.

More recent work on information search has shown that children have strong intuitions about questions' usefulness and search adaptively (Nelson et al., 2014) and that adults can value information over explicit reward when the two are put in opposition (Markant and Gureckis, 2012).

Our interest is in whether these trends persist when people are confronted with large, noisy information spaces. The real world is just that — large and noisy — and so humans' ability to learn successful intuitive theories hinges on their ability to confront size and to deal with noisy information. As we explore the world and its affordances, we must select queries that have scope, in that they rule out large numbers of hypotheses at once. And, inasmuch as we can help it, we should minimize noise by querying reliable sources.

In the real world, these are often in opposition. So, we ask: how do people trade off scope and reliability when exploring large, noisy information spaces?

2.2 An Information-search Task

To answer the question of how people trade off scope and reliability when searching for information in noisy environments, we designed a novel information-search task. Our goal was for it to be as simple as possible, while having as many features approaching natural exploration as possible. Thus we paired a very simple game — identifying a hidden number on a number line — with a relatively complex search procedure. In playing the game, participants make queries that vary in the abstract features of scope and reliability. Additionally, at each point in the game, participants are faced with a very large number of potential specific queries to choose from. Our task is similar to the Markant and Gureckis (2012) task in that it involves exploring a geometric space to test particular hypotheses, but we wanted to make explicit the abstract features of

questions, rather than have these be implicit as a function of the hypotheses at hand.

Participants play by asking 'questions' about the hidden number's location, using 'scanners' that turn blue if the number is under the scanned region and red if the number is not. In each trial, a participant is given four scanners (Figure 2-1). In some conditions, the scanners vary in size. Larger scanners can cover larger regions of the number line, ruling out (or in) a larger set of hypotheses than a smaller scanner. Thus, in the context of this study, the 'scope' is directly related to the length of the scanner. However, the scanners are not deterministic; they also vary in their reliability, which is the probability of providing an accurate signal about the presence of the hidden number (false positives and false negatives are equally likely).

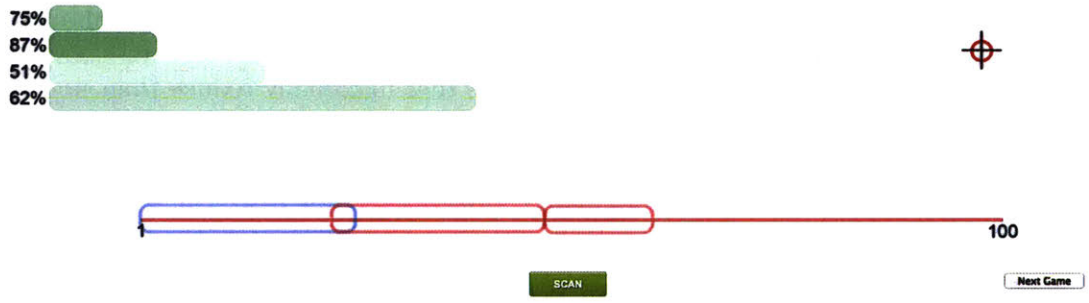


Figure 2-1: A screenshot of the task. The four green rectangles are the scanners. The player has scanned three times and received one positive and two negative answers, as shown by the blue and red scanner outlines on the number line.

To efficiently find the hidden number, participants have to select scanners that provide a good trade-off between length and reliability, and then place them in informative regions on the number line. The optimal trade-off between these factors (length, reliability, location) can be captured by expected information gain, as described in the next section. Actions that are optimal in the sense of expected information gain minimize uncertainty about the location of the hidden number.

2.3 Model

Let I denote the number line (in our case, integers ranging from 0 to 100), and let $h \in I$ denote a hypothesis about the hidden number. On each trial, participants choose an action a (placing a particular scanner over a portion of the number line) and observe a binary outcome d (1 if the number was detected by the scanner, 0 if the number was not detected). There are almost 500 possible actions, since each of the four scanners can be placed anywhere as long as some part overlaps with the number line. The posterior distribution over h is updated on each trial according to Bayes' rule:

$$P(h|d, a, D) \propto P(d|h, a)P(h|D), \tag{2.1}$$

where D denotes the history of actions and outcomes prior to the current trial.

Intuitively, participants should choose actions that maximally reduce their uncertainty about the location of the hidden number; this corresponds to taking actions that maximally reduce posterior uncertainty, which can be quantified by the entropy:

$$H[P(h|d, a, D)] = -\sum_h P(h|d, a, D) \log P(h|d, a, D) \tag{2.2}$$

Minimizing posterior entropy is equivalent to maximizing information gain (the reduction of entropy after taking action a and observing d):

$$\mathrm{IG}(a, d) = H[P(h|D)] - H[P(h|d, a, D)]. \tag{2.3}$$

Because the outcome d is not available at the time of choosing a, the best that a participant can do is maximize *expected* information gain:

$$\mathrm{EIG}(a) = \sum_d P(d|a, D)\mathrm{IG}(a, d), \tag{2.4}$$

where the posterior predictive distribution is given by $P(d|a, D) = \sum_h P(d|a, h)P(h|a, D)$.

Let α denote a scanner's error probability and I_a denote the range covered by the scanner. The game generates signals according to:

True positive: $\Pr(d = 1 \mid \{I_a \cap I\} \neq \emptyset) = 1 - \alpha$.

True negative: $\Pr(d = 0 \mid \{I_a \cap I\} = \emptyset) = 1 - \alpha$.

False positive: $\Pr(d = 1 \mid \{I_a \cap I\} = \emptyset) = \alpha$.

False negative: $\Pr(d = 0 \mid \{I_a \cap I\} \neq \emptyset) = \alpha$.

Roughly speaking, conditioned on a scanner choice, participants should bisect the interval that has the highest posterior probability of containing the scanner. This corresponds to the *split-half* heuristic discussed by Navarro and Perfors (2011) and Nelson et al. (2013). However, because in this game signals are stochastic, there may not be a single contiguous interval of highest posterior probability, and thus no reasonable interval on which to precisely perform the split-half heuristic. Nevertheless, actions can still be ranked according to EIG.

To help hone our own intuitions about the task and about participants' actions, we created a display (Figure 2-2) that in this case shows a vignette of four sequential actions from one particular user's trial. The normative posterior distribution over the location of the hidden number (in grey at the bottom) is displayed for each trial, along with the scaled expected information gain[1] of each of the 500 available actions (the colored arcs; each dot represents the center of the range at which the scanner could be placed); these are the evaluations of the normative model. The same recommendations are shown as posterior-predictive entropies in the insert (the red dot in the insert shows the posterior-predictive entropy of the action actually chosen by the participant), along with the participant's action at that point (the flat green or red bar at the bottom), and the result of that action (in this display, green indicates 'yes' and red indicates 'no').

[1]Scaled expected information gain is obtained by dividing the expected information gain of each action by the highest expected information gain available at that decision point.

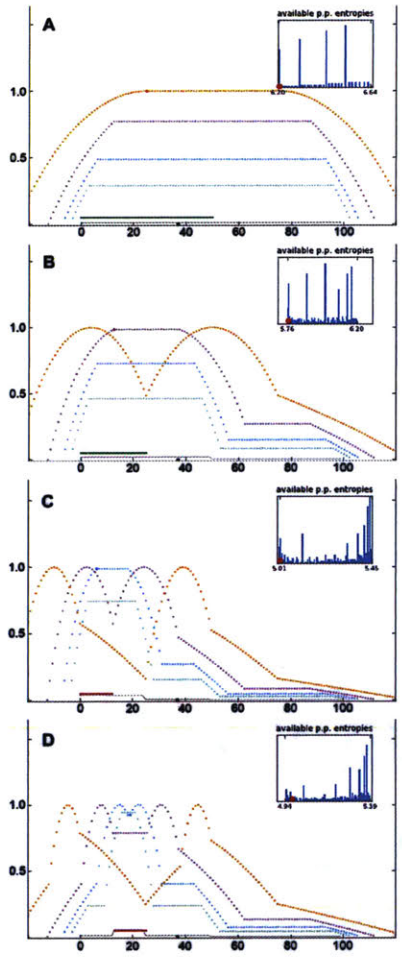


Figure 2-2: Analysis of one participant's four sequential actions in the information-search game. All the scanners were of reliability=0.87 and varied only in length. **A**: The participant uses the length=50 scanner and receives a response of 'yes'. The arcs correspond to the scaled information gain of each candidate action; note that the participant selects one of the highest-rated actions. This fact is also indicated in the insert, which shows the posterior-predictive entropies of all candidate actions (lower is better); the red dot indicates the posterior-predictive entropy of the action chosen by the participant. **B**: The participant employs the 'split-half' heuristic to test half the remaining space. Note that all of the highest-ranked model recommendations involve testing exactly half of the highest-posterior-probability region. **C**: The participant continues to use 'split-half'. **D**: The participant re-tests the entire region of highest posterior probability. This is highly rated, but not as much as it would be had they tested half the remaining space.

2.4 Experiment 1

2.4.1 Method

Participants. 26 participants completed the experiment for pay on Amazon Mechanical Turk.

Materials. We used a base set of scanner reliabilities in the set, $\{0.51, 0.62, 0.75, 0.87\}$, and of lengths in the set, $\{0.0625, 0.125, 0.25, 0.5\}$.[2] From these we created scanners as follows for the following conditions:

Reliabilities: One scanner in each of the four reliabilities above; all are of length 0.25.

Lengths: One scanner in each of the lengths above; all are of reliability 0.87.

Crossed: We wanted to see whether people could choose well when reliability and length were put in opposition, so we crossed them to create the following [length, reliability] pairings:

$$[0.5, 0.51], [0.25, 0.62], [0.125, 0.75], [0.0625, 0.87].$$

Each participant performed 6 trials: 4 'crossed' trials, 1 'reliabilities' trial, 1 'lengths' trial; the order of these was randomized.

Procedure. Each participant read a short sequence of sequential, interactive, instructions. They were told that with each round of the game a number would be hidden at a random location on the number line, told that they could use scanners to find the number, introduced to the scanners, shown that scanners glowed blue to indicate that the number was in that region and red to indicate that it was not, and given experience with the fact that the scanners could produce both false positives and false negatives. Finally, participants were told that once they thought they knew where the hidden number was located, they should use a provided set of crosshairs to indicate its location.

[2]The indicated length is as a proportion of the number line.

Following the end of the 'instructions' session, participants played six rounds of the game. Each game (trial) presented them with four scanners whose lengths and reliabilities were determined by the condition of the trial. As they played each round, the history of their queries was visible to them as the red or blue glow left by each scanner remained in its place after the scanner was removed; this was to ensure that memory load did not differentially affect participants' decisions. After a scanner was used, it returned to its original position. Using multiple scanners simultaneously was not allowed by the interface. Participants were allowed to make as many queries as they wanted as they went through a trial; trials ended only when participants indicated their believed location of the hidden number by using the provided crosshairs.

Our main interest was to see how participants would confront a trade-off between and reliability. Before examining this, however, we tested each separately as proof of concept.

2.4.2 Results

Lengths condition

First we tested whether participants were sensitive to the fact that different queries had more or less coverage of the hypothesis space. We gave them four scanners which varied only in their lengths; the scanners all had reliabilities of 0.87, and lengths of $[0.5, 0.25, 0.125, 0.0625]$. For each decision point reached by a participant (that is, before each scanning action), we calculated the expected information gain afforded by centering each available scanner at the center of the region actually queried by the participant. We then ranked those scanners according to this expected information gain. Participants picked the best scanner most often — 45% of the time — showing that they were sensitive to the imperative to cover as much ground of the hypothesis space as possible. A repeated-measures one-way ANOVA showed that choice proportions differed significantly across scanners $[F(3, 25) = 16.55, p < 0.0001]$. A post-hoc t-test showed that the difference between the first and second scanner ranks was significant $[t(25) = 5.80, p < 0.0001]$.

Reliability condition

Next we tested whether participants were sensitive to the imperative to reduce noise; that is — did participants make reliable queries when they were given a chance to do so?

Participants were given four scanners which varied only in reliability; the provided reliabilities were $[0.87, 0.75, 0.62, 0.51]$, and all scanners were of length 0.25.

Across all subjects and all trials of this condition, participants were strikingly sensitive to reliability, as they selected the best scanner 89% of the time. A repeated-measures one-way ANOVA showed that choice proportions differed significantly across scanners $[F(3, 25) = 52.88, p < 0.0001]$. A post-hoc t-test showed that the difference between the first and second scanner ranks was significant $[t(25) = 8.88, p < 0.0001]$.

Crossed condition

When confronted with a choice of questions that might either greatly reduce the size of the hypothesis space or provide reliable answers, how did participants choose?

We constructed a condition in which these two dimensions were in direct opposition: we offered an array of scanners such that the more coverage of the hypothesis space a scanner provided, the less reliable it would be. We used four scanners whose [lengths, reliabilities] were $[0.0625, 0.87], [0.125, 0.75], [0.25, 0.62], [0.5, 0.51]$. Participants were clearly sensitive to length/reliability tradeoffs, as they selected the best scanner 57% of the time. A repeated-measures one-way ANOVA showed that choice proportions differed significantly across scanners $[F(3, 25) = 56.91, p < 0.0001]$. A post-hoc t-test showed that the difference between the first and second scanner ranks was significant $[t(25) = 5.82, p < 0.0001]$.

While we found these results encouraging, we were concerned that the particular length/reliability trade-offs we provided were such that one dimension might have contributed significantly more to expected information gain than the other. If this were the case, participants might have made what looked like normative decisions driven by both length and reliability, even though in reality they were driven only by

length (or by reliability).

To examine this possibility, we ran a second experiment in which we provided participants with a large array of scanners of different [length, reliability] pairings. Length and reliability would trade off in terms of their relative contributions to a scanner's expected information gain, and therefore good performance in this condition would indicate an actual sensitivity to both dimensions.

2.5 Experiment 2

2.5.1 Method

Participants. 26 participants completed the experiment for pay on Amazon Mechanical Turk.

Materials. The same base set of lengths and reliabilities described in Experiment 1 was used to construct a new 'mixed' condition: We selected 8 of the 16 possible pairings of the 4 lengths and 4 reliabilities so as to cover a reasonable range of possibilities for available scanners. Specifically, participants were provided with scanners whose lengths and reliabilities were semi-randomly paired using the original $[0.5, 0.25, 0.125, 0.0625]$ lengths and $[0.87, 0.75, 0.62, 0.51]$ reliabilities. They then played the game in the same way as in Experiment 1.

We also ran a 'deterministic' condition in which scanners varied in length but were all of reliability $= 1$, randomly interleaved with the 'mixed' conditions. The results were similar to those reported in Experiments 1 and 2, and are left out for the sake of brevity.

Participants performed 6 trials (4 'mixed', 2 'deterministic'); the order of these was randomized for each subject.

Procedure. The procedure was identical to Experiment 1, except that in the instruction phase, participants were additionally provided an opportunity to use a scanner of each reliability as many times as they wanted, on a number line in which the target number was not hidden, in order to become fully familiar with the scanners they

would later use. The intent was to have participants learn about the reliabilities, so for this familiarization stage, all scanners were of equal length.

2.5.2 Results

Scanner choice: length + reliability

First we examine whether participants are able to select questions with the best abstract features — that is, do they select questions whose length and reliability combine to provide the highest expected information gain?

As in Experiment 1, we computed the expected information gain for each of the four scanners, conditioned on the placement actually chosen by each participant; we then ranked the scanners according to this EIG and examined where each participant's scanner choice fell in these rankings. A repeated-measures one-way ANOVA showed that choice proportions differed significantly across scanners [$F(3, 25) = 61.66, p < 0.0001$]. A post-hoc t-test showed that the difference between the first and second scanner ranks was significant [$t(25) = 7.94, p < 0.0001$].

Figure 2-3blueA shows the distribution over all scanner choices for the 'mixed' condition, together with the predictions of a softmax version of the EIG model, to be explained later. Participants frequently chose the best scanner, roughly 60% of the time across a wide range of length/reliability trade-offs, which suggests that they are sensitive to the expected information gain of both the scope and reliability of queries rather than to either of these factors alone. Across the 8 sub-conditions that comprised the 'mixed' condition, the probability that participants chose the best scanner also correlated with the EIG model's choice probabilities (Figure 2-3blueB, $\rho = 0.606$).

Joint choices of scanner and placement

Having determined that participants correctly weigh the trade-off between the abstract features of scope and reliability to select the best-available scanners, we can examine how well they select the specific queries they make.

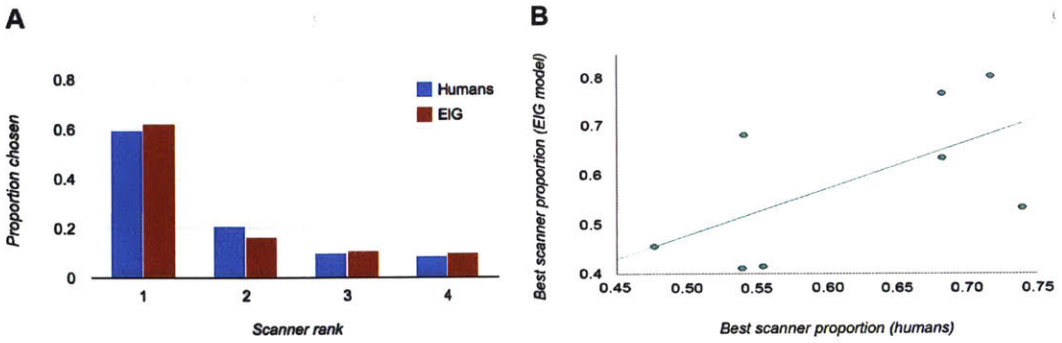


Figure 2-3: **A**: The 'mixed' condition reveals that participants are able to select the best scanner across a wide variety of reliability/length trade-offs. **B**: The proportion of times participants chose the most informative scanner (x axis) versus predictions of the *EIG* model (y axis), for each of the 8 sub-conditions in the 'mixed' condition.

Given a choice of scanner, how sensitive are participants to the expected information gain of where they place the scanner along the number line? For each decision, we ranked the expected information gain of each possible placement of the scanner chosen by participants. Participants chose a placement in the top 10th percentile 38% of the time, and in the top 20% almost 60% of the time. A repeated-measures one-way ANOVA showed that choice proportions differed significantly across percentile bins $[F(9, 25) = 55.43, p < 0.0001]$. A post-hoc t-test showed that the difference between the first and second bins was significant $[t(25) = 7.11, p < 0.0001]$.

We can also ask about the overall quality of the queries made, over all possible choices of query scope, reliability, and location. Participants are highly sensitive to expected information gain in this space, selecting queries in the top 10% of the available set more than 50% of the time (Figure 2-4). A repeated-measures one-way ANOVA showed that choice proportions differed significantly across percentile bins $[F(9, 25) = 71.4, p < 0.0001]$. A post-hoc t-test showed that the difference between the first and second bins was significant $[t(25) = 7.24, p < 0.0001]$. They are also decreasingly likely to select an action in each of the subsequent percentiles, with a profile very well fit by the *EIG* model ($\rho = 0.994$).

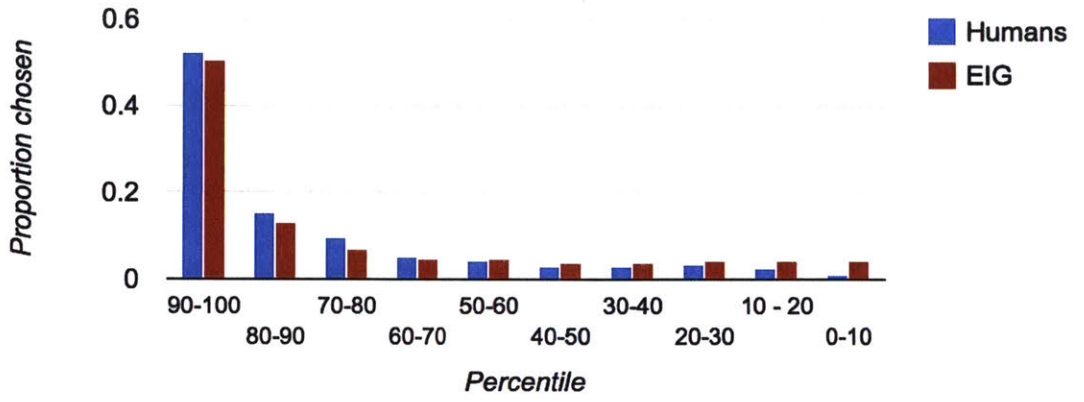


Figure 2-4: In their overall query choices (combining scanner scope, reliability, and location), participants are highly sensitive to expected information gain; note also the tight fit of the *EIG* model ($\rho = 0.994$).

2.5.3 Model comparisons

The above results suggest that participants are sensitive to expected information gain (EIG), rather than making selections on the basis of reliability or length alone. To test this claim more rigorously, we fit five alternative models in addition to *EIG*, which make decisions according to the following criteria: *length*, *reliability*, *length + reliability*, *EIG-length*, *EIG-reliability*. Each model computes a 'value' of an action, $V(a)$, as a linear function of the stated parameters: $V(a) = \sum_i \beta_i f_i$, where f_i is some feature of the current trial/action (i.e., EIG, length, or reliability) and β_i is a coefficient fit to each participant by maximum likelihood (when all features share the same coefficient, β is often referred to as the *inverse temperature*). This value is then transformed to a choice probability according to the softmax function, $P(a) \propto \exp\{V(a)\}$.

The *reliability*, *length*, and *reliability+length* models test the hypothesis that participants are sensitive to these features of scanners without using them to compute expected information gain. The *EIG-reliability* and *EIG-length* models test the hypothesis that participants are sensitive to expected information gain but insensitive to one or another feature of queries. Specifically, in the *EIG-reliability* model, decisions

are made according to the expected information gain that arises from considering the reliabilities of the scanners, but ignoring their lengths.[3] The *EIG-length* model, on the other hand, takes the correct scanner lengths into account but assumes a reliability of 1.0 for each scanner. The original *EIG* model tests the hypothesis that participants are sensitive to expected information gain with no qualifications.

For each model, we computed the (participant-specific) Bayesian Information Criterion approximation to the marginal likelihood, and then submitted the models to the Bayesian model selection algorithm of Stephan et al. (2009), which estimates the group-level exceedance probability for each model (the probability that a given model is more likely than all other models considered). The exceedance probabilities for the models are as follows: *EIG*: 0.9982, *EIG-reliability*: 0.0000, *EIG-length*: 0.0004, *reliability*: 0.0068, *reliability+length*: 0.0000, *length*: 0.0000.

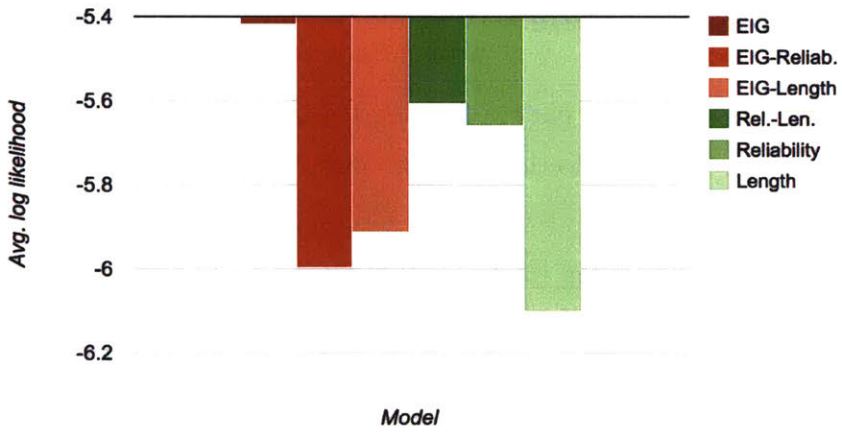


Figure 2-5: Log likelihoods assigned by each model to participants' joint choices of scanner type and placement.

How do the models predict participants' actions, in general? We computed the average log likelihood for participants' joint choices of scanner (length and reliability) and placement (the specific location on the number line queried) under each model (Figure 2-5) and find that the full *EIG* model, sensitive to all three factors, performs

[3]There is no way to ignore length in these calculations while still computing EIG, so to to model ignorance in this case we set the length of each scanner equal to the average length of the available scanners.

best. The *EIG-length* and *EIG-reliability* models do not fare well, as they operate on incorrect assumptions (that all scanners were deterministic, or that all scanners were of the same average length, respectively) which distort the EIG calculation. The *reliability* and *reliability + length* models do fairly well here, but looking at scanner placements conditioned on scanner choices, they predict no better than chance; this is the main reason they fare worse than the full *EIG* model in predicting overall actions choice.

2.6 Discussion

The ability to search efficiently for information in large and noisy environments is critical for real-world learning and discovery. Constructing scientific theories or their intuitive analogues hinges on being able to successfully test competing hypotheses, which often requires balancing the epistemic virtues of scope — to deal with the world's enormity — and reliability — to deal with the world's noise. In this paper we have shown that adults can appropriately balance these trade-offs, effectively conducting the most informative tests when conducting information search with large action spaces. This result is consistent with previous work on information search (cf. Nelson et al. (2013)), but more closely resembles natural exploration because of the large action spaces used and the fact that participants had to confront the scope-reliability trade-off.

In our attempt to understand how humans deal with this trade-off, we compared several alternative computational models. Three were sensitive to the abstractions of scope and reliability without any regard for the way in which these factored into a query's expected information gain; three others calculated expected information gain and used scope and reliability either explicitly or implicitly. We found that human actions were best explained by a model that selects actions according to their expected information gain, using both scope and reliability along with the precise location of a proposed test in the EIG calculation.

While this study was designed to capture abstract features of information search

in many natural environments, one might be concerned that our results are specific to visuospatial or physical domains in which the target exists at some location and participants might be able to deploy geometric intuitions about how to search for objects that may not hold for more abstract forms of information search. To some extent our study is less prone to this objection than other recent work using spatially organized tasks Markant and Gureckis (2012), because of the added complexity introduced by varying reliability which does not have an immediate spatial component. Future work should extend both the modeling and experimental paradigms to entirely nonspatial domains, where scope and location have more abstract interpretations.

One salient feature of our computational model is that it evaluates actions only on their one-step information gain; it is 'myopic', in AI terms. Perhaps human information search can effectively look further ahead than one step, considering EIG over the many possible outcomes of multiple choices. It is also possible that myopic EIG describes human search best. Future work should investigate these possibilities.

Chapter 3

Hypothesis-space Constraints in Causal Learning

3.1 Introduction

The last fifteen years have produced a spate of research highlighting the kinds of epistemic practices that allow children to effectively navigate their complex world. Children have been shown to rationally infer causal relationships from statistical evidence (e.g., Gopnik et al. (2004)), to selectively explore when evidence is confounded or surprising (Bonawitz et al., 2012), to constrain their generalizations depending on how evidence is sampled (Gweon et al., 2010; Denison and Xu, 2010; Xu and Denison, 2009), to infer the existence of unobserved variables to explain anomalous data (Schulz et al., 2008), and to isolate candidate causes in order to distinguish between competing hypotheses (Cook et al., 2011). Older children (ages 8-9) have also been shown to effectively confront hypothesis spaces as they search for information (Nelson et al., 2013). These practices combine to enable the formation of *intuitive theories* — abstract, coherent, causal, ontologically-committed frameworks that guide prediction, explanation, and action (Gopnik and Meltzoff, 1997; Carey, 1985; Murphy and Medin, 1985).

Powerful as these practices may be, the reality is that for even a simple problem — for example, *what causes day and night*? — the space of hypotheses is infinite.

The fact that children are able to prodigiously map causes and effects all around them suggests the existence of methods for constraining hypothesis spaces *before* engaging in (often costly) hypothesis-testing.

Schulz (2012) suggests that learners have information about the gaps between each candidate hypothesis and the data they seek to explain; these 'gaps' can be used to identify promising hypotheses. Problems contain information about the abstract properties of their solution — solutions to navigation problems are likely to involve 2-dimensional maps, explanations are likely to generate causal chains or trees, and so on. A progenitor of this idea can be found in the work of Langley et al. (1987), who framed the process of scientific discovery as one of means-ends problem-solving, and who focus on heuristics (inspired by research on human problem-solving) for reaching a goal situation without exploring all possible alternative intermediate states. An initial test of this general idea, by Magid et al. (2015), asked if children were sensitive to such information. In a series of experiments, children were shown two causes — one discrete (two-setting) button and one continuous slider on the same controller box — and two effects — one discrete and one continuous. Across a variety of conditions children successfully mapped the discrete cause to the discrete effect, and the continuous cause to the continuous effect.

Here we suggest that certain priors about causal processes may enable children and adults to prioritize certain regions of the hypothesis space when seeking to build good theories. But what form should these priors take? If they are to be effective across a variety of problems, lower-level cognitive features such as color, height, pitch, malleability, won't do, since causal relations do not often preserve these. That is, color changes are not usually caused by other colors; pitch changes are not caused by other pitch changes. However, certain higher-level amodal features such as extent, rate, arity[1], distributional properties, and dynamics, are invariant to the lower-level features, allowing for perhaps more relevant comparisons. We propose that children are sensitive to these higher-level features and can use them to match effects with their causes. In this paper we focus on two: distributional properties and dynamics.

[1]The number of states that a variable can take.

3.2 Distributional Properties

As long as objects in two (or more) sets can be grouped into types, the relative proportions of those types across sets can be evaluated. This holds regardless of the features that serve to establish object identity, making such operations widely applicable. Even young infants have been shown to understand proportion, as they map proportion to probability of outcomes and can use proportion to guide their actions (Xu and Garcia, 2008; Xu and Denison, 2009; Denison and Xu, 2010). Here we ask whether young children believe that causal processes preserve proportion and if they can use this information to select between candidate causal hypotheses that cannot be distinguished by other means (e.g., covariation data, surface features, or domain-specific prior knowledge).

3.2.1 Methods

Participants

Sixteen preschoolers (mean[2]: 5 years, 1 month; range: 4 years, 3 months — 5 years, 7 months.) were recruited from a local children's museum.

Materials

We used Paint Tool SAI to create four flowers, two for each stimulus set. Within each stimulus set, the flowers differed in shape but matched in color (yellow for one stimulus set; blue for the other). For each stimulus set, there was a warm-up picture displaying only the two kinds of flowers and two test pictures: one test picture had 16 flowers of each kind (1:1 proportions); the other flower had 28 flowers of one kind and 4 of the other (7:1 proportions). We also used four different kinds of seeds, two for each stimulus set. Within each stimulus set, the seeds were near-identical to each other in size and texture, but different in color both from each other and the flowers

[2]All children were 4 or 5 years old. Due to a data storage error, the ages of 8 of the children were only recorded accurately to the year; these children have been excluded from the estimated mean and range.

(black and red seeds paired with yellow flowers in the first stimulus set, and brown and orange seeds paired with blue flowers in the second). The seeds were combined either in 1:1 or 7:1 proportions and were presented in containers, each containing approximately 100 seeds. See Figure 3-1.

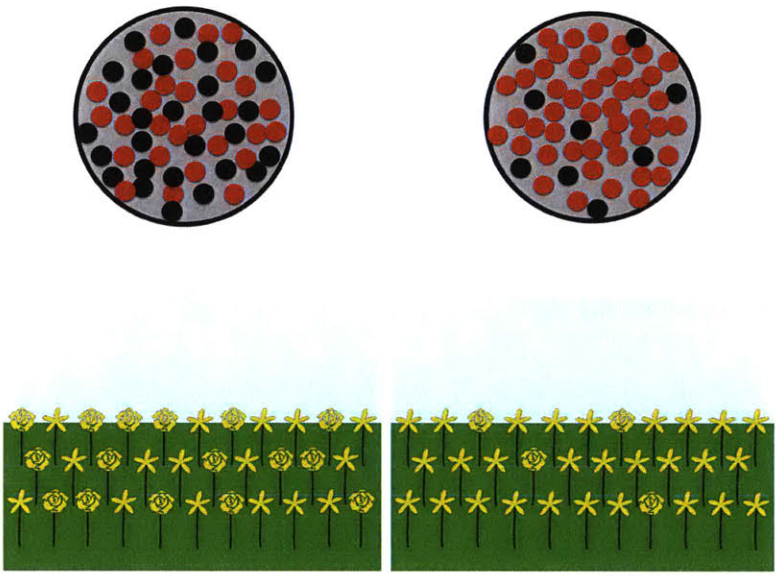


Figure 3-1: Schematic of seeds and actual flower fields used. Left: 1:1. right: 7:1.

Procedure

Children were tested individually in a private room off the museum floor. The experimenter started the experiment by placing the warm-up picture of two flowers on the table in front of the child. He pointed to the two flowers and said, "Look, we have two flowers. This is a daisy and this is a lily. Now, you know how flowers are grown, right? With seeds! You put seeds into the ground and you water them and give them sun, and then flowers bloom! But seeds and flowers are funny, because flowers end up not looking at all like the seeds they came from; the seeds change in all kinds of ways: they change in color, and size, and shape." Two seeds were placed on the table in front of the children, and introduced as the seeds that were used to make the flowers.

Children were told, "These are the seeds we used to make the flowers. And just like we just talked about, they look totally different from the flowers, so we can't tell which seeds made which flowers just by looking at them." Then children were then told, "Now, we actually have whole fields of flowers, but before I show you the fields, let me tell you about how they were made." The two capfuls were brought out and placed on the table, next to each other (left-right randomized). "We had these two capfuls. And what we did was we took a bunch of seeds from this capful and threw them on one field, and we took a bunch of seeds from this capful and threw them onto the other field." The experimenter made a grabbing and throwing motion from each capful to the floor using alternating hands to illustrate. The experimenter then said, "Now I'll show you what the two fields ended up looking like," and brought out the two pictures of the fields of flowers (one with the 1:1 proportions and the other with the 7:1 proportions), placing them one above the other on the table (top-bottom randomized). He pointed to each field in turn and said, "Which capful do you think was used to make this field?".

After the child pointed to match each field with a capful, the experimenter removed all the stimuli and then repeated the procedure for a second trial with the second stimulus set, transitioning by saying, "Now, let me show you some more flowers." Presentation order of the fields and of the capfuls within stimulus set was randomized across and within participants, as was stimulus-set order.

3.2.2 Results

There is no strict sense in which we can say that the children responded 'correctly' or 'incorrectly' given that there is no fact of the matter here. However, we can say whether children, as predicted, used the abstract property of proportionality to select one hypothesis over the other. Children were counted as having succeeded on a trial if they matched both capfuls in the trial to the correct fields. (The two questions — one for each field — within a trial were treated as non-independent because in introducing the capfuls, the experimenter had said that one capful was used for one field and one for the other. Thus the most conservative measure was to require a

correct response to both questions). The probability of succeeding on both trials by chance is .25. Ten out of sixteen children responded at ceiling, answering correctly on both trials. The probability of success is .625; bootstrap-estimated 95% confidence interval: $[.354, .848]$. $p < .01$ by two-tailed binomial test. Figure 3-2 shows the number of children who were correct on 0, 1, or both trials.

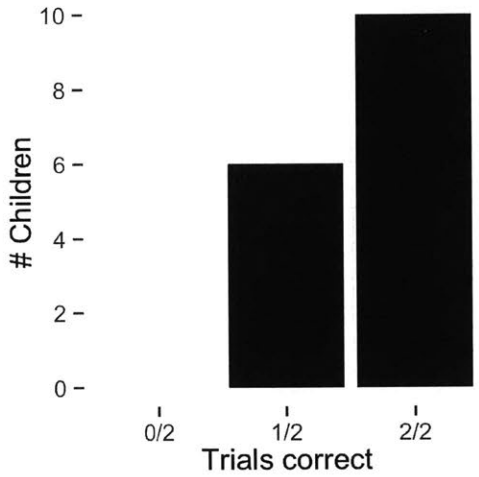


Figure 3-2: Experiment 1 results (N=16): Mean trials correct = 1.62. Probability of full success (2/2 trials) is .625; 95% confidence interval: $[.354, .848]$ (chance: .25). $p < .01$ by two-tailed binomial test.

Thus, children showed a clear preference for the proportion-preserving causal process, supporting our hypothesis.

3.3 Dynamic Properties

Experiment 1 suggests that children are sensitive to proportionality in mapping causes to effects. However, if children have a general ability to identify plausible hypotheses using abstract amodal features, they should be sensitive to other kinds of relationships, as well. In Experiment 2, we look at children's sensitivity to dynamic properties. Specifically, we expect that, given data that saliently vary over time on some dimension, children will infer the latent structure of the variation and expect the cause of the data to possess similar latent structure. Children could be sensitive to a variety

of dynamics. As a first pass, we investigate two: monotonicity and periodicity.

3.3.1 Methods

Participants

Thirty-two preschoolers (mean: 4 years, 9 months; range: 4 years, 0 months — 5 years, 9 months.) were recruited from the local children's museum.

Materials

We used sixteen 2"x2" pieces of white cardboard to make two sets of eight cards. For one set, four cards had a large red dot in the middle and four had a yellow one; for the other set, the dots on each of eight cards varied continuously from red through orange to yellow. These were presented in groups of eight, and represented the lights in two special rooms (See Figure 3-3). We also created four separate two-minute videos (two per stimulus set) using Adobe Flash and displayed them on the experimenter's laptop computer. Each video had ten identical 'alien bugs' moving around randomly. In two of these videos, the bugs changed in their speed over the course of the video. In the other two, the bugs grew spots on their back; the number of these spots changed throughout the course of the video. Each of these two features could change in two ways — either periodically or monotonically. We generated videos manifesting each of the four possible feature $\times$ dynamics combinations and split them into two stimulus sets as follows:

Stimulus Set 1 In video 1, the bugs' speed was governed by a periodic function — the bugs accelerated to a noticeably high speed (4 inches/second) over the course of 5 seconds, then decelerated to their original speed (.5 inches/second) over the course of 5 seconds, then re-accelerated to the high speed, and so on. This oscillation persisted throughout the two-minute video. In video 2, the bugs maintained a constant speed (.5 inches/second), but they grew black spots on their backs; the number of these spots increased from 0-20 over the course of the two-minute movie.

Stimulus Set 2 In video 1, the periodically-governed bugs moved at constant speed but varied in their number of spots, which oscillated between 0-10 spots (each half-period lasted 5 seconds, so that the fewest and most spots appeared matched the time-points at which the bugs were going slowest and fastest in stimulus set 1). In video 2, The monotonically-varying bugs changed in speed, starting out slowly and rising constantly throughout the video. See Figure 3-4 for a schematic depiction of these videos.

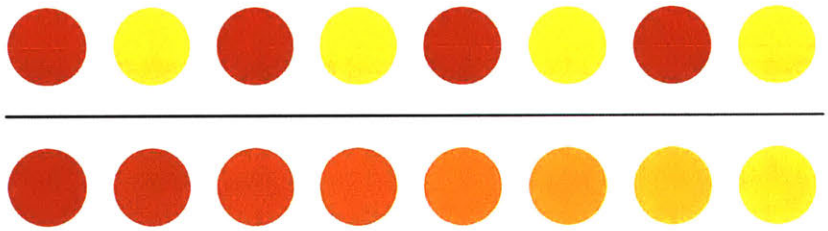


Figure 3-3: 'Lights' used for experiment 2. Top: periodic. Bottom: monotonic.

Procedure

Children were told that they would be shown some alien bugs, but that before seeing the bugs, they would learn about the rooms the bugs were in. These rooms were described as identical except that they differed in their 'special lights'. The experimenter said, "The lights in the first room start out looking like this... then after a while they look like this... then after a while they look like this...", placing a light on the table each time he said 'this'. The lights were placed one by one on the table, left-to-right facing the child. Once the eight lights from the first set were placed, the experimenter said, "In the other room, the lights start out looking like this... then after a while they look like this..." and placed the lights for that room in the same manner as for the first. For one of the rooms the experimenter used the red and yellow (periodic) lights in alternation; for the other the experimenter used the continuously-varying red-to-yellow (monotonic) lights. Room type (periodic or monotonic) was randomized, as was whether the first light in each room was red or yellow. The lan-

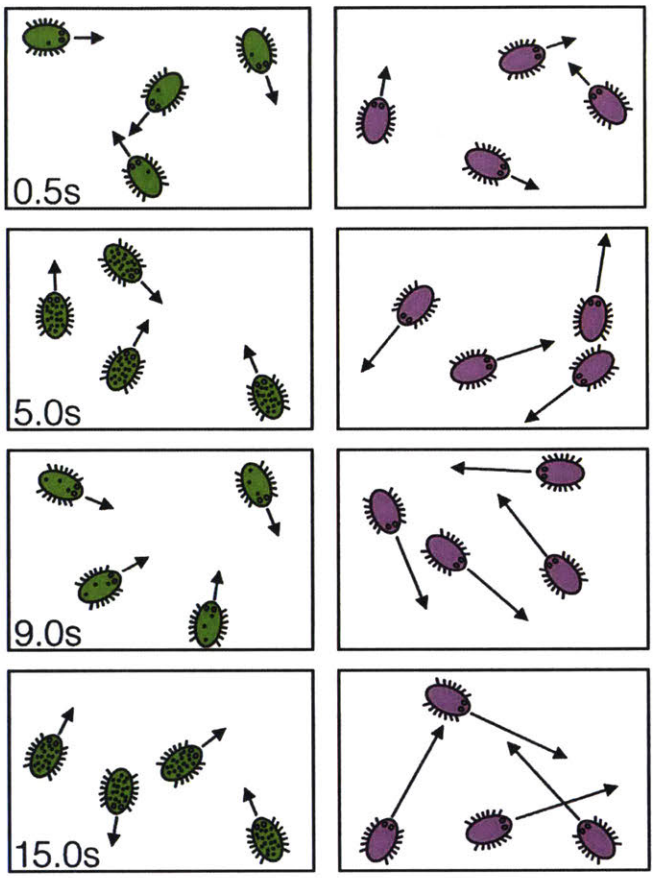


Figure 3-4: Periodic (green) and monotonic (purple) bugs from stimulus set 2. The green bugs move at constant speed but increase and decrease in number of spots; the purple bugs increase speed monotonically, as indicated by the increasing length of the vectors.

guage used to describe both sets of lights was identical. The first four lights in each set were placed on the table approximately every 3 seconds; to keep the description conversational, each of the last four lights was placed approximately every 1 second. This also ensured that there was no way to map the rate of presentation of the cards to the rate of change of either speed or spots in either display.

Following this, the children were invited to look at the bugs. In the first stimulus set, for the periodic bugs, the children were invited to attend to their speed: as the video played, the researcher pointed out when the bugs sped up ("See, now they're getting faster") and when they slowed down ("...and now they're getting slower"). For the monotonic bugs, the video was played twice. The first time, the experimenter

allowed the children to observe the changing number of spots on their own. After there were approximately 15 spots on each bug, the experimenter restarted the movie, and this time counted the number of spots on the bugs as these increased in number, summarizing the change after there were 6 spots ("They're getting more and more spots.").

For the second stimulus set, the procedure was identical, except that the language was modified appropriately to describe the different changes in the bugs. The experimenter pointed out when the spots on the periodic bugs were increasing or decreasing in number ("Now they're getting more/fewer spots"), and on the monotonic bugs, pointed out the increasing speed ("Now they're going faster... and now the're going faster... and now they're going even faster...", etc.).

Each child only saw one stimulus set — that is, one set of periodic bugs and one set of monotonic bugs. Stimulus-set assignment was counterbalanced.

After the children were familiarized with the stimuli, they were shown the first bugs they had seen, asked to remember how they changed, and then were told the following: "So, we know that these guys are in one of these two rooms that we talked about before. They can see the lights in the room but we can't. And what's causing the speed to change is the lights of the room they're in. They could be in this room or in this room. Do you know what room they're in?" Children were asked to point to the room they thought the bugs were in. After this, they were shown the other bugs and the above description and question were repeated verbatim, changing only 'speed' to 'spots' (or vice-versa if the first-seen bugs had varied in spots). It was emphasized that each of the bugs could be in each of the two rooms.

3.3.2 Results

Due to the fact that each group of bugs was independently described as being potentially in each room, the two questions are independent; the probability of answering both questions correctly by chance is .25. Children were sensitive to the fact that bugs could have been in the same room, as evidenced by the fact that six children placed both types of bugs in the same room. Nineteen out of thirty-two children were

at ceiling, answering correctly on both trials. The probability of success is: .594; bootstrap-estimated 95% confidence interval: $[.406, .763]$. $p < .001$ by two-tailed binomial test. Figure 3-5 shows the number of children who were correct on 0, 1, or 2 trials. Performance across the two stimulus sets was comparable: 10/16 children were at ceiling for the first and 9/16 children were at ceiling for the second.

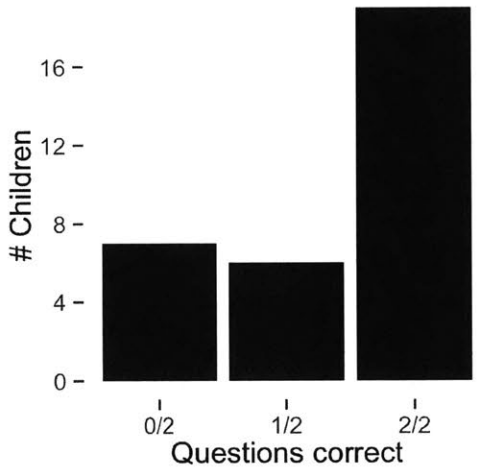


Figure 3-5: Experiment 1 results (N=32): Mean trials correct = 1.375. Probability of full success (2/2 trials): .594, 95% confidence interval: $[.406, .763]$ (chance: .25). $p < .001$ by two-tailed binomial test.

3.4 Discussion

In two experiments, we investigated whether children use higher-level features of data to match causes to effects. In Experiment 1, these features were static and distributional: children showed a preference for a causal process that preserved distributional identity, matching 1:1 seeds to 1:1 flowers and 7:1 seeds to 7:1 flowers irrespective of surface properties of the seeds and flowers (size, color, texture, etc.). In Experiment 2, the higher-level features were dynamic: children showed a preference for a causal process that preserved dynamic form — periodic or monotonic — irrespective of the lower-level features in which these dynamics were manifested; periodically-varying lights were seen as the cause of the spots on bugs *or* the speed of the bugs, depending

on which one varied periodically, and monotonically-varying lights were seen as the cause of the monotonically-varying feature of bugs. In both experiments, children received no information about how causes and effects covaried; inferences were made based only on abstract properties of the stimuli.

The idea of using high-level features to match percepts has been presented previously, in the literature on cross-modal matching (see, e.g., Lewkowicz and Turkewitz (1980); Spence (2011)). Cross-modal matching is often presented as a partial solution to the binding problem and the focus is on mappings between stimuli belonging to different perceptual modalities (i.e., the sight and feel of a stimulus). We believe that it is possible that the inferences children drew here and those shown in studies on cross-modal matching may rely on the same representational machinery; specifically, we believe that both rely on the use of particularly powerful higher-level features of the stimuli. However, the mechanisms we suggest are applicable to a far wider array of problems than mere cross-modal matching. The higher-level features we have examined, namely distributional properties and dynamics, are calculable both within and across modes; in our case we have examined their application within a perceptual modality and have found them to apply both to naturalistic stimuli such as seeds and flowers and to arbitrary stimuli, such as randomly-moving animated alien bugs. In principle, the same kinds of inferences could be used for problems entirely abstract in nature (e.g., using the dynamics of the interest rate to map it onto changes in the Gross Domestic Product).

Research on analogy Gentner and Markman (1997); Gentner (1977); Gick and Holyoak (1980) has shown that children and adults are able to bring distinct mental representations into structural alignment and to use the relations that obtain within one domain to reason about the other. We believe that analogical reasoning is an elegant example of the more general ability to use high-level features to constrain hypothesis spaces. Note that here, however, we did not set up a situation where children could go from a known problem and solution to a new problem and a new solution by setting up a relational mapping between arguments (e.g., Christie and Gentner (2010)). Rather, children had to infer the representation that might connect

the form of the effect to the form of the candidate causes and use this representation to guide their responses. We emphasize this not to minimize the importance of analogical reasoning, but because the general ability to represent these abstract high-level predicates may allow learners to narrow the hypothesis space even when problems do not present as analogies.

We have shown that children are sensitive to high-level features like proportionality and periodic or monotonic dynamics, and can use these to infer causal relationships. A variety of empirical questions remain: Do children use these features not just to select hypotheses but also to generate them? When such features conflict with lower-level features, such as color, texture, or size, how do children resolve the conflict? Perhaps most interestingly, how do children learn these high-level predicates, and which predicates are available to them at different times throughout development? Given the prevalence of phenomena in the world for which distributional properties and dynamics are coherent and relevant, it may not be surprising that four- and five-year-olds can use these; what about younger children and what about other kinds of properties (e.g,. extent, other kinds of dynamics, richer distributional information, or combinations of these)? Much remains to be understood about how children identify abstract features, and about what other kinds of high-level features they can recognize; much also remains to be understood about how we might computationally characterize the ability to represent, learn, and use these higher-level features. We hope to see future work address these questions.

Chapter 4

Human Learning in Atari

4.1 Introduction

In this chapter we broaden the scope of our inquiries to the problem of learning to behave adaptively in unknown environments. Video games offer a range of tasks that differ widely in their visual representation, dynamics, and goals; this makes games perfect microcosms for the development of algorithms that may eventually learn to function in the real world. Recent research on this front has been hugely successful: Deep reinforcement learning algorithms using deep neural networks have surpassed human-level performance on games from the classic Atari 2600 platform (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017). However, even a cursory examination of the behavior of these artificial agents reveals that they learn incredibly slowly (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017; Kansky et al., 2017) and that they generalize poorly to trivial variations of a given environment (see the experiments reported by Kansky et al. (2017)).

Our experiments in this chapter show that humans are able to learn these tasks in a matter of minutes, suggesting that humans and leading AI algorithms employ different representations and learning mechanisms. In addition to bringing early-arising representations to bear on Atari tasks, humans come equipped with rich prior

knowledge about the world – for example, knowledge about keys, doors, ice, birds, and so on. While such knowledge could give humans an edge over AI algorithms, we believe that it plays a minimal role in humans' ability to rapidly master these tasks. Here we conduct a systematic analysis of human performance on these tasks, experimentally manipulating (1) prior knowledge about specific objects, (2) prior knowledge about the game environment and rules, and (3) observational learning experience.

4.2 Gameplay

4.2.1 Methods

We selected twelve Atari 2600 games that spanned the range of DQN performance relative to human performance, as reported in Mnih et al. (2015). Of these, the asymptotic DQN outperformed humans in five games (Beamrider, Breakout, Kangaroo, Qbert, and Stargunner), and underperformed humans in seven (Amidar, Asteroids, Frostbite, Montezuma, Riverraid, Seaquest, and Venture).[1]

Participants found through Amazon Mechanical Turk were assigned to a single game that they had not previously played, and were told that they should play for a minimum of 15 minutes. All participants were paid $2.00 and were promised bonus pay of up to $2.00 extra depending on their cumulative score performance. Prior to playing the game, participants were told only that they could use the arrow keys and the spacebar, and that, beyond that, they should try to figure out how the game worked in order to play well. Participant numbers are as follows: (Amidar: 19, Asteroids: 20, Beamrider: 19, Breakout: 25, Frostbite: 71, Kangaroo: 18, Montezuma: 32, Qbert: 24, Riverraid: 24, Seaquest: 18, Stargunner: 19, Venture: 18).

[1]'Human performance' refers to the human game tester employed by DeepMind and used as the human benchmark in the original DQN paper. After the tester trained on each game for two hours, their scores over the 20 subsequent game episodes that lasted over 5 minutes were averaged and reported.

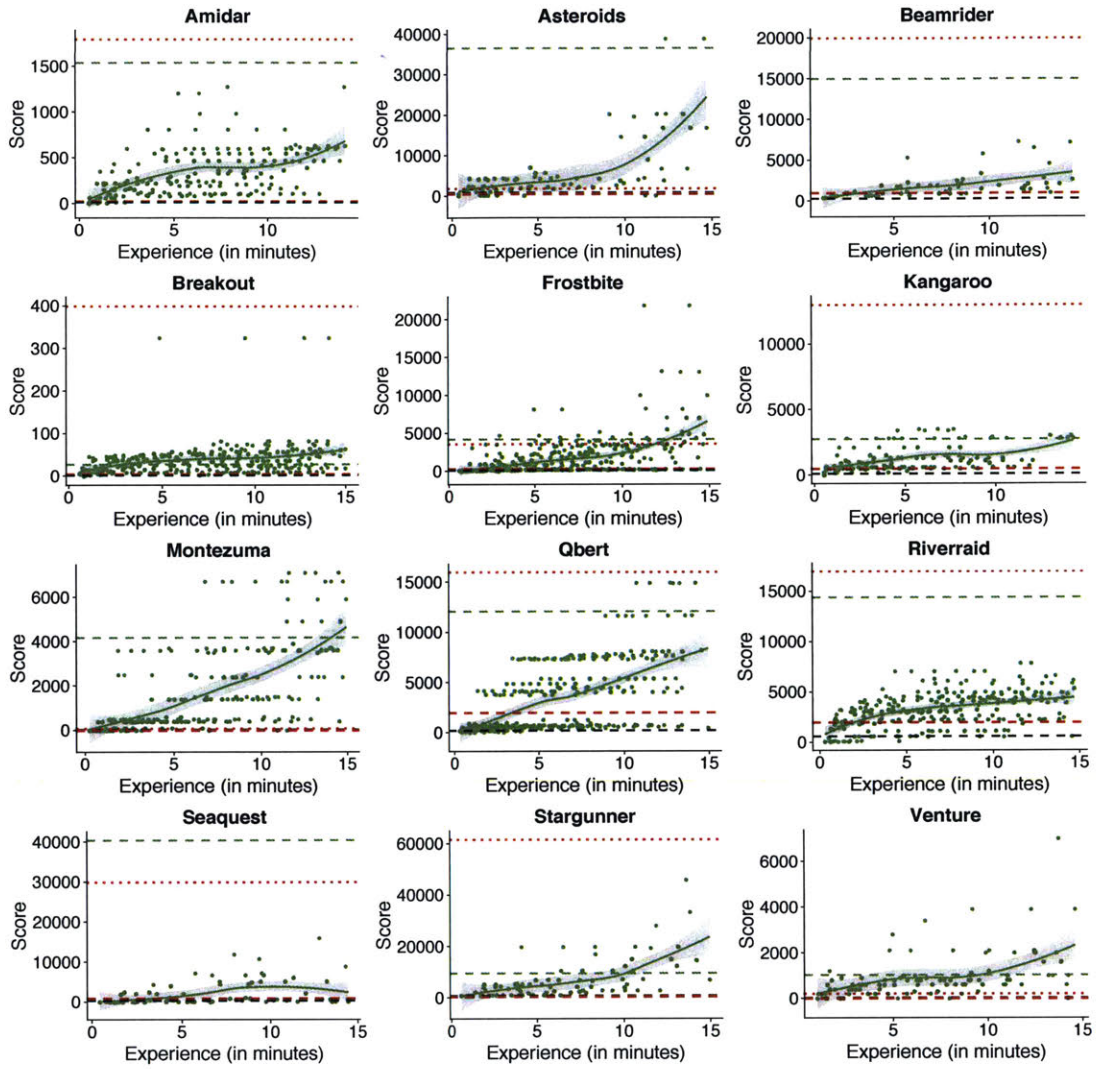


Figure 4-1: Human learning curves for twelve Atari games. Black horizontal line: random play. Green horizontal line: 'expert' play. Red horizontal dashed line: DDQN after 46 hours of game-play experience. Red dotted line: DDQN after 920 hours.

4.2.2 Learning Curves

Figure 4-1 shows learning curves for the four games. Each points represents a (human, time, score) tuple; the score reported at each point is the highest score obtained by that particular subject after that amount of cumulative gameplay experience. For comparison, we have also plotted random performance (black horizontal), human

'expert' play[2] (green horizontal line), and performance of the more recent Double Deep Q-Network (DDQN) Van Hasselt et al. (2016) after 10 and 200 million frames of game-play experience (46 and 920 hours, respectively), in red (bottom to top)[3]. We highlight a few qualitative observations: human performance is above random performance within the first minute of play; in half of the games, humans reach 'expert' performance within the allotted 15 minutes; in all of the games, humans exceed DDQN's 46-hour score within just a few minutes; in Asteroids, Frostbite, Montezuma, and Venture, humans exceed even DDQN's asymptotic performance.

Of course, these time comparisons are unfair – after all, in addition to having potential cognitive advantages, humans come to these tasks with a working visual system, while DDQN has to learn a visual system from scratch. With this in mind, we can also look at DDQN's rate of improvement and compare it to human rates of improvement at score-matched points – including points at which DDQN is doing relatively well. Figure 4-2 shows such a comparison for three games. Note that the rate of improvement is measured in log units. Humans still learn dramatically faster than DDQN.

4.3 Experimental Manipulations

The game of Frostbite is a game worth analyzing closely, as humans exhibit particularly impressive performance relative to DQN, DDQN, and the more recent "Learning to Play in a Day" algorithm He et al. (2016) (Figure 4-3). In what follows we present several experimental manipulations. Our hope is that by understanding the representations that facilitate human performance in this game (and eventually in other games), we can help pave the way toward designing more human-like AI agents.

[2]This refers to the averaged scores of the DeepMind games tester

[3]Data taken from Schaul (2015). The model significantly outperforms the original DQN, as well as several subsequent variants.

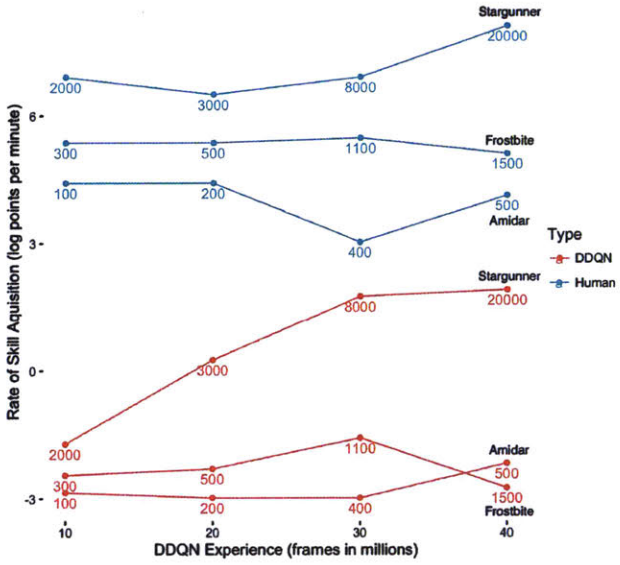


Figure 4-2: Learning rate comparison. X-axis: DDQN experience, in millions of frames. Y-axis: Rate of improvement in log (points per minute), estimated using finite differences. Human rate of improvement is taken from score-matched points (shown as numbers annotating the curves). DDQN estimates are made from Figure 7 in Schaul et al. (2015).

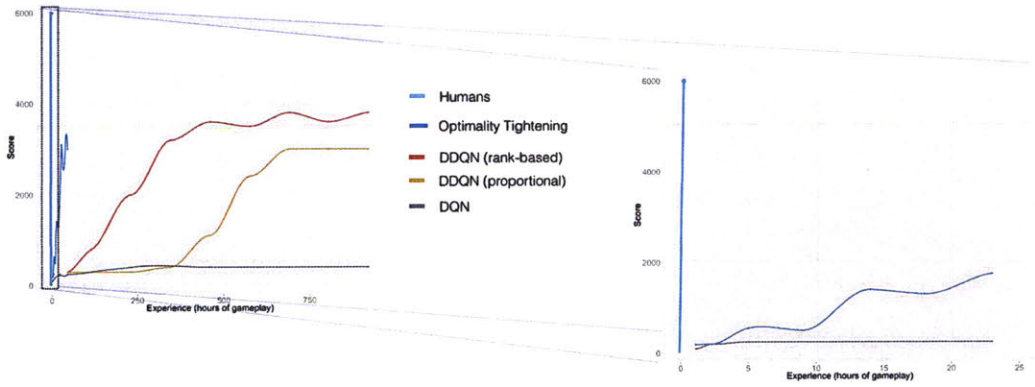


Figure 4-3: Performance on the game of Frostbite shows the striking degree to which humans outperform existing AI models in learning how to play video games.

4.3.1 Obscuring Object Identity

One specific hypothesis as to why humans rapidly learn to perform well at Frostbite is that they come equipped with strong priors about the objects they encounter. Parsing the game screen into platforms, igloos, birds, and fish must clearly make

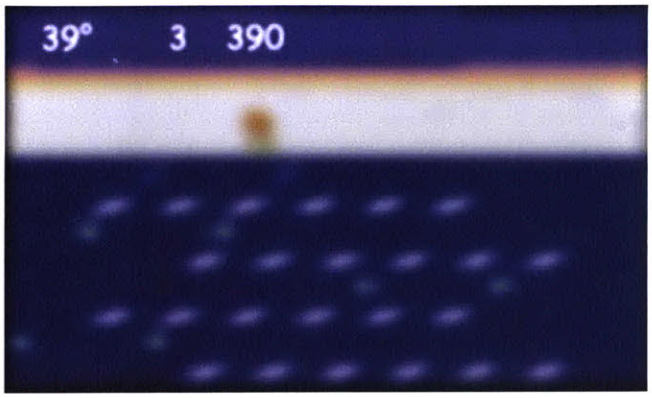


Figure 4-4: Screenshot of a blurred version of Frostbite. The light blue items are birds; the green items are fish.

the task easier for humans, as humans know things about the properties of each of these objects: platforms provide support; igloos provide shelter; fish and birds can be eaten. To test whether such knowledge is in fact a benefit to humans, we created a blurred version of the game, in which objects could be only be identified as generic objects – that is, we masked the semantic identity of the objects.[4] Figure 4-4 shows an example game screen. We present data from this condition after describing the other experimental manipulations.

4.3.2 Reading the Instruction Manual

If people's rapid learning is due to the formation of a model-like representation, then anything that would enable them to learn this representation should result in an increase in early performance. To test this hypothesis, we provided participants with the opportunity to read the game's original instruction manual prior to playing. The intent was to provide players with a short description of critical aspects of gameplay, which was mostly informative about objects and their roles in the game, as well as the goal of the game. Subjects read the manual, answered a short questionnaire intended to check that they understood the rules, and then played for 15 minutes.

[4]This resulted in blurred birds, fish, crabs, and clams, but clearly-identifiable water, ice floes, and igloos.

FROSTBITE BASICS

The object of the game is to help Frostbite Bailey build igloos by jumping on floating blocks of ice. Be careful to avoid these deadly hazards: killer clams, snow geese, Alaskan king crab, grizzly polar bears and the rapidly dropping temperature.

To move Frostbite Bailey up, down, left or right, use the arrow keys. To reverse the direction of the ice floe you are standing on, press the spacebar. But remember, each time you do, your igloo will lose a block, unless it is completely built.

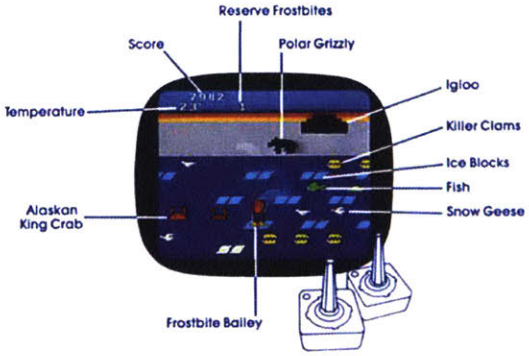
You begin the game with one active Frostbite Bailey and three on reserve. With each increase of 5,000 points, a bonus Frostbite is added to your reserves (up to a maximum of nine).

Frostbite gets lost each time he falls into the Arctic Sea, gets chased away by a Polar Grizzly or gets caught outside when the temperature drops to zero.

The game ends when your reserves have been exhausted and Frostbite is 'retired' from the construction business.

IGLOO CONSTRUCTION

Building codes. Each time Frostbite Bailey jumps onto a white ice floe, a "block" is added to the igloo. Once jumped upon, the white ice turns blue. It can still be jumped on, but won't add points to your score or blocks to your igloo. When all four rows are blue, they will turn white again. The igloo is complete when a door appears. Frostbite may then jump into it.



Work hazards. Avoid contact with Alaskan King Crabs, snow geese, and killer clams, as they will push Frostbite Bailey into the fatal Arctic Sea. The Polar Grizzlies come out of hibernation at level 4 and, upon contact, will chase Frostbite right off-screen.

No Overtime Allowed. Frostbite always starts working when it's 45 degrees outside. You'll notice this steadily falling temperature at the upper left corner of the screen. Frostbite must build and enter the igloo before the temperature drops to 0 degrees, or else he'll turn into blue ice!

SPECIAL FEATURES OF FROSTBITE

Fresh Fish swim by regularly. They are Frostbite Bailey's only food and, as such, are also additives to your score. Catch' em if you can.

Figure 4-5: Screenshot of the original Atari 2600 instructions to Frostbite, displayed to participants.

4.3.3 Learning from Observation

Humans' ability to form a theory-like representation of the game should also be aided by observing others play. We randomly selected an episode in the 75-85th percentiles of episodes from the first round of experiments, and had all participants in this condition watch a video of that episode prior to playing. The episode corresponded to the 79th percentile of all Frostbite episodes, and lasted 1 minute, 26 seconds.

4.3.4 Results

Figure 4-6 shows means and 95% CIs of first-episode scores for normal, 'blur', 'instructions', and 'observation' conditions (participant Ns are 71, 63, 72, 72, respectively). The difference in first-episode performance between normal and blurred conditions is not significant (p=0.663. Mean, CIs: Normal: 356, [167, 545]. Blur: 417, [216, 618]). This is unsurprising: a bird, a priori, could be useful (it can be hunted and eaten) or it could be harmful (it can attack you). In the actual game the interaction between the agent and the bird is, in fact, quite implausible a priori – real-life birds are much lighter than humans and are generally not capable of pushing them, and yet, in the game they do so. While priors about object identity and the resulting behaviors of objects may be minimally useful to a novice player, most of the important properties of objects in these games come from their role in the particular game.[5] It is after observing objects' movements and interactions that humans rapidly form theories of the game dynamics.

As we expected, reading the instruction manual and observing competent players provided participants with a significant first-episode advantage over normal play (Instructions vs. Normal: p=.0001. Observation vs. Normal: p=.002. Mean, CIs: Normal: 356, [167, 545]. Instructions: 1848, [1144, 2552]. Observation: 1144, [683, 1605]). Had players only learned from observation, one might posit that they simply copied a successful policy. However, the fact that the instruction manual was helpful suggests that this is not the case, and hints at the possibility that humans used this

[5] Additional experiments conducted after ours (see Dubey et al. (2018)) recapitulate these findings.

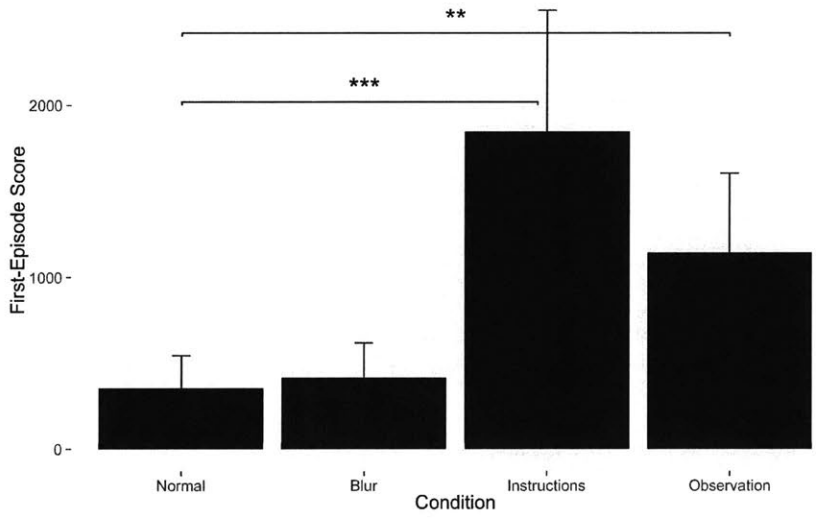


Figure 4-6: First-episode mean scores and 95% confidence intervals for normal, 'blurred', 'instructions', and 'observation' conditions. If semantics contributed significantly to human performance, blurring would have produced a decrement in first-episode performance; instead, this intervention makes no significant difference. By contrast, reading the instructions and observing another player each allow humans to capture approximately 1000 points in their first episode – this corresponds to human performance after about 5 minutes of play under normal conditions.

information to build a model of the game dynamics, which they then used to play successfully.

4.4 Discussion

The real power of human intuitive theories is that they enable humans to explain the world, generalize from few examples, think counterfactually, and generate effective plans. The experiments above show that humans are capable of learning complex Atari tasks from just a few minutes of gameplay, and that their behavior is aided by information that would be helpful in theory-building. In the next section we propose an algorithm that incorporates exploration, theory-building, and planning to produce human-level performance on a suite of visually-simple, Atari-inspired, video games.

Chapter 5

Theory-Based Reinforcement Learning in Games

5.1 Introduction

Humans are remarkable in their ability to rapidly learn complex tasks from little experience.[1] Recent successes in AI have produced algorithms that can perform complex tasks well in environments whose simple dynamics are known in advance (Silver et al., 2016; Brown and Sandholm, 2018), as well as models that can learn to perform expertly in unknown environments after a great amount of experience (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017). Despite this, no current AI models are able to learn sufficiently rich and general representations so as to support rapid, human-level learning on new, complex, tasks.

Games, and in particular, video games, serve as readily available microcosms through which to study and develop such algorithms. The model-free deep reinforcement learning algorithms mentioned above have surpassed human performance on games from the classic Atari 2600 platform, but even a cursory examination of their

[1]Some of the introductory material in this section was presented in Chapters 1 and 4, but we reprise it here in the interest of giving the reader a clear stand-alone sense of the modeling goals and experimental setting of this chapter.

behavior reveals that they learn incredibly slowly (Guo et al., 2014; Mnih et al., 2015; Van Hasselt et al., 2016; Schaul et al., 2015; Stadie et al., 2015; Mnih et al., 2016; He et al., 2016; Hessel et al., 2017; Kansky et al., 2017) and that they generalize poorly to trivial variations of a given environment (see the experiments reported by Kansky et al. (2017)). Humans, on the other hand, learn these behaviors nearly instantly (Figure 5-5).

We propose a novel approach, Theory-Based Reinforcement Learning, that enables rapid learning in novel environments through the use of human-like "intuitive theories" — rich causal models that support explanation, prediction, and action (Murphy and Medin, 1985; Carey, 1985; Gopnik and Meltzoff, 1997). The approach learns probabilistic generative models that carve the world into a natural ontology of of objects, physics, agents, and goals, inspired by early-arising human representations (Spelke, 1990; Baillargeon, 2004; Spelke and Kinzler, 2007; Csibra, 2008). These theories are expressed as abstract descriptions of game rules, and are connected to a simulator that supports prediction and planning.

We instantiate the approach concretely in our Exploring, Modeling, Planning agent (EMPA). Our agent models the world by constructing theories that best explain the observations under a Bayesian criterion. The process of acquiring data is itself driven by curiosity about the correct explanation of the dynamics of a particular game — the agent actively builds a theory by looking to bring about observations that are informative about its hypothesis space. Goals, whether they are curiosity- or performance-oriented, are achieved through the use of a planning algorithm that takes advantage of the interpretability of the theory to automatically generate game-specific heuristics that enable efficient search.

5.2 Related Work

The original Deep Q-learning Network (Mnih et al., 2015) used a deep convolutional neural net trained through stochastic gradient descent to approximate the environment's optimal action-value function, $Q(s, a)$, and variants of this algorithm have

achieved increasing asymptotic performance as well as increasing data efficiency by varying each of several relevant components: the Double DQN (Van Hasselt et al., 2016) addresses an overestimation bias caused by DQN's loss function, and Prioritized Experience Replay (Schaul et al., 2015) samples transitions for replay as a function of their last-encountered TD error, thereby more frequently sampling transitions from which there is more to learn (but also sampling more from stochastic transitions where there may no longer be anything to learn). The A3C algorithm (Mnih et al., 2016) executes multiple agents in parallel, thereby decorrelating their experience, and estimates both a value function, $V(s)$ and a policy, $\pi(s)$. These and other methods turn out to be largely complementary; Rainbow (Hessel et al., 2017) combines them to produce state-of-the-art performance on the Atari benchmark, eventually reaching 210% of human performance across the 57 benchmark games.

Our approach tries to learn explicit, object-oriented, relational dynamics models of the environment; in this sense it shares common threads with many existing model-based algorithms. LIVE (Shen and Simon, 1989) learned rules that explained domains such as Towers of Hanoi and a simplified Mendelian pea-breeding prediction task. EXPO (Gil, 1994) refined an initial partial domain model and used a planner to select actions in a robot planning domain and a complex process planning domain. However, neither algorithm was able to handle nondeterministic environments. More recently, Guestrin et al. (2003) presented Relational MDPs, which used a state-space that represented objects as instances of distinct classes. This class-oriented representation enabled them to achieve strong generalization of learned value functions across environments. Diuk et al. (2008) introduced the idea of Object-Oriented Markov Decision Processes, where transitions over object attributes were learned from experience. Their algorithm learned with low sample complexity because of the object- and attribute-oriented representation, and performed adaptively in their "Taxi" domain as well as in the first screen of the video game, Pitfall.

Pasula et al. (2004) learned STRIPS-like operators for planning in three planning domains, including a simplified "slippery gripper" physics domain. Given a user-defined representation language, their algorithm was able to learn precondition-effect

rules that compactly described the domain. Pasula et al. (2007) learned richer probabilistic representations that enabled them to plan effectively in the complex, noisy environment of a simulated blocks world. More recently, Xia et al. (2018) used deictic references to learn sparse transition models using feed-forward neural networks. Their algorithm was able to learn probabilistic state transition models for a simulated blocks domain with just a thousand training examples and was robust to the complexity induced by increasing numbers of objects in the domain.

Scholz et al. (2014) presented Physics-Based Reinforcement Learning, which performed adaptively in two simulated physics domains after learning correct parameterizations of a physics engine with just a few steps of experience. More recently, Kansky et al. (2017) presented Schema Networks, which learn causal generative models over object attributes. After an initial training period on the Atari game of Breakout, these networks were able to generalize quickly to new, small, variants of the game.

In contrast with the above approaches, our goal is to build a model that explains how humans learn so many tasks, so quickly. In order to do this, we build an agent that has rich prior knowledge about the possible worlds it might encounter. This prior knowledge corresponds to knowledge that adult humans have, and that young children and infants have in some form, as well. Combined with the right experience in a given environment, this knowledge allows for the sample-efficient construction of human-like intuitive theories that explain each environment's dynamics. A full understanding of each environment involves observing transitions that are unlikely to be generated by chance; EMPA explores its environment specifically in order to generate such transitions. Finally, learned dynamics are exploited by use of a planning algorithm that uses the explicit nature of the learned theories to generate game-specific heuristics that enable efficient search.

5.3 Explore, Model, Plan: Overview

EMPA learns simple latent descriptions of game dynamics, independently learning about entities, relations between entities, and goals. A complete theory of a game

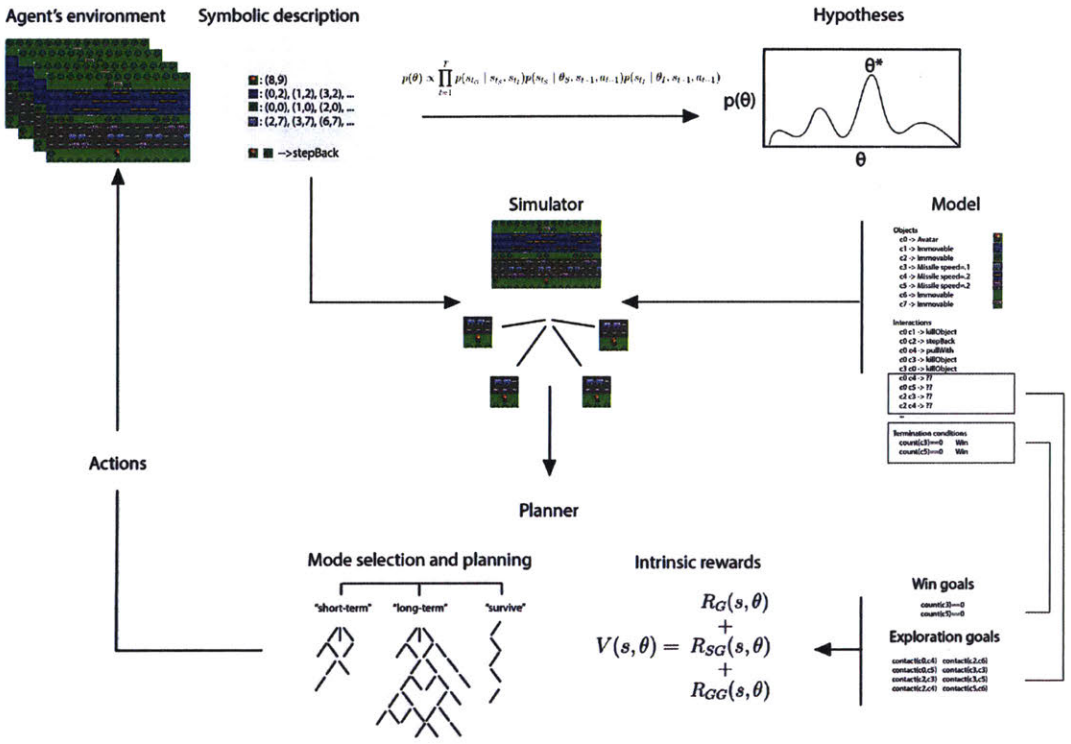


Figure 5-1: Schematic representation of the EMPA architecture. EMPA takes as input a symbolic description of its environment, specified in terms of objects and their locations as well as interactions that occur between objects. Bayesian inference scores theories on their ability to explain observed state sequences. Theory-based curiosity generates exploratory goals. The planner decomposes exploratory and win-related goals into subgoals and goal gradients, and uses this hierarchical decomposition to effectively find high-value actions for EMPA to take in the game environment.

supports the evolution of simulated future states conditioned on a real or hypothesized game state. *Explore* uses curiosity about the correct theory to generate epistemic goals. *Model* uses a theory-based representation and scores theories on the fidelity of their corresponding simulations with respect to observed states. *Plan* decomposes goals into reachable subgoals and generates a goal gradient that enables the agent to solve games with extremely sparse rewards. The agent interacts with the environment in a continuous cycle of exploring, modeling, planning, and acting; see Figure 5-1 for a schematic of the EMPA architecture and Figure 5-2 for an illustration of the step-by-step contributions of the EMPA components.

The theories learned by EMPA are object-oriented, relational, and compositional,

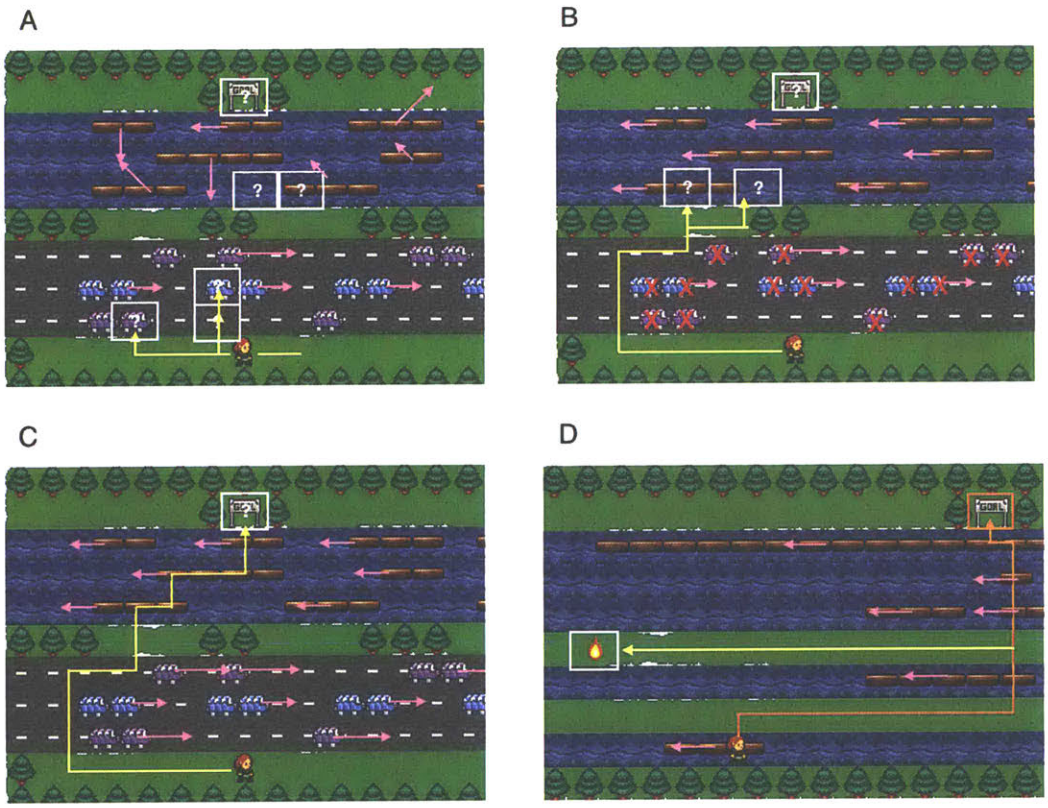


Figure 5-2: A: Early-stage game screen showing contributions of the EMPA architecture. Explore (white): agent's epistemic goals. Model (pink): an early hypothesis about predicted paths of objects. Plan (yellow): The planner finds high-value trajectories according to the agent's current theory and its epistemic goals. B: A mid-stage game screen. The agent has learned that cars are dangerous, but is still curious about logs, water, and the flag. The planner produces paths that reach mid-range epistemic goals involving contact with plank and water objects. C: A late-stage game screen, at which point the agent has interacted with all objects except for the flag object. In yellow is the planner's suggested path. Upon interacting with the object the agent will win the level and learn a termination condition for the game. D: The theory of a game is invariant to permutations in object positions, and therefore automatically generalizes to new levels. Here the planner generates a path (orange) toward the known goal on a new level, using the same theory from the previous level. The agent is also automatically curious about the newly-occurring object in the game, and the planner suggests a path to that (yellow), as well.

and are specified in terms of object properties, agents, physics, events, and goals. This ontology is a deliberate embodiment of the kinds of priors that are critical to babies' abilities to rapidly learn about their world (Spelke, 1990; Baillargeon, 2004; Spelke and Kinzler, 2007; Csibra, 2008).

EMPA's theories are represented explicitly using the Video Game Description Language (VGDL) (Schaul, 2013). VGDL is a light-weight language for describing video games, accompanied by software that permits parsing and instant playing of a given game. It contains a description of the appearance and basic properties of the different objects in the game, which manifest as causal constraints on their dynamic properties: whether they move by default and in what directions, how quickly they move, what their goals are, and so forth. Relational rules specify the outcomes of contact events between objects; these rules encompass natural concepts such as pushing, destroying, picking up, and so forth. Finally, game goals specify the win/loss conditions for a particular game.

Together with a description of an initial state, these rules specify the transition dynamics of a given game world. EMPA's hypothesis space corresponds to a restricted but vast space of possible VGDL descriptions: for a game with 10 unique classes, there are over 10^{37} possible VGDL descriptions. The posterior probability of a theory is $p(\theta \mid s_{0:T}, a_{0:T-1}) \propto p(s_{0:T} \mid \theta, a_{0:T-1})\, p(\theta)$. The likelihood function, $p(s_{0:T} \mid \theta, a_{0:T-1})$, can be decomposed as

$$p(s_{0:T} \mid \theta, a_{0:T-1}) = p(s_0) \prod_{t=1}^{T} p(s_{t_G} \mid s_{t_S}, s_{t_I}) p(s_{t_S} \mid \theta_S, s_{t-1}, a_{t-1}) p(s_{t_I} \mid \theta_I, s_{t-1}, a_{t-1})$$

where each of the terms corresponds to a factorization of the state into goals, objects, and interaction events, respectively. Our agent maintains a resource-bounded approximation to the Bayesian posterior for these theories; see Chapter 5.4.2 for details.

EMPA's theory-based curiosity incentivizes it to observe game states that are informative about its hypothesis space. In particular, the object-oriented, relational nature of the hypothesis space means that EMPA wants to observe agent-object and object-object interactions (see epistemic goals in Figure 5-2). A crucial feature of this

curiosity is that it is not myopic; EMPA often generates long-ranging plans whose sole purpose is to generate informative interactions. This enables EMPA to solve games that are challenging or impossible for traditional exploration methods.

EMPA uses its simulator, parameterized by the highest-posterior theory, to imagine future states conditioned on current actions. Our planning algorithm ameliorates the exponential explosion of the state space by using Iterative Width (Geffner and Lipovetzky, 2012; Lipovetzky and Geffner, 2017). States that fail to fulfill the IW notion of 'novelty' are pruned from the search tree. Sparse rewards are often a problem for AI agents, but EMPA is able to circumvent this problem by generating intrinsic rewards on imagined states from the theory. These rewards involve goals (hypothesized or known in-game win conditions as well as epistemic goals supplied by the curiosity module), subgoals that come from the decomposition of hypothesized game goals, and 'goal gradients' that help the planner more quickly find goals and subgoals: $V(s,\theta) = R_G(s,\theta)+R_{SG}(s,\theta)+R_{GG}(s,\theta)$. Best-first search on the pruned space using this intrinsic reward function allows the agent to efficiently generate effective plans.

5.4 Model

5.4.1 Representation

The theories learned by EMPA are object-oriented, relational, and compositional, and are specified in terms of object properties, agents, physics, events, and goals. These theories are represented using the Video Game Description Language VGDL, (Schaul, 2013). VGDL is a light-weight language for describing video games, accompanied by software (PyGame (Shinners, 2011)) that permits parsing and instant playing of a given game. The main aspects of these descriptions carve the world into an ontology akin to that of the human "start-up" software:

Objects (SpriteSet): A description of the appearance and basic properties of the different object classes in the game. These properties all manifest as causal constraints on the dynamic properties of the objects: whether they move by default,

how often they do, whether they have agent-like goals and ways of reaching them, and so forth. We call these properties the *dynamic type* of each object. At every time-step, VGDL uses the dynamic type to calculate proposed positions for each object. If the proposed positions are empty, the objects move into those positions. If they do not, VGDL handles the resulting collisions by looking at the rules specified in the InteractionSet.

Relations (InteractionSet): A list of rules that specify how pairs of objects interact to produce events. These events encompass natural concepts such as pushing, destroying, picking up, and so on. All state changes beyond the movement patterns specified in the SpriteSet are caused by these events. In turn, events are caused only by collisions between objects.

Goals (TerminationSet): A specification of the win/loss conditions for a game. We restrict our scope to games whose terminations involve statements regarding the counts of particular objects appearing on the screen (e.g., `WIN IF count(BLUE)==0`).

Each VGDL description specifies an initial state, a transition function that specifies how the state evolves between successive time-steps (conditioned on the player's actions), and a reward function. The state at any time can be described by the object instances, their class, their location, the set of items 'carried' by the Avatar, as well as the events occurring between pairs of objects that are participating in collisions at any given time-step. It is useful to decompose this state s into s_I, which refers to the pairs of objects that are participating in events as well as the nature of those events, s_S, which refers to the remaining objects, and s_G, which refers to the `Win/Loss/Continue` status of the environment. For the purposes of learning, we assume that the state-space is fully observed.

Our concrete task is to learn a distribution over VGDL descriptions, $\theta_{1:N}$, that explain a sequence of frames of game-play. For clarity, we decompose a theory θ_i into the three components corresponding to the SpriteSet (θ_S), the InteractionSet (θ_I), and the terminationSet (θ_G).

The SpriteSet, θ_S consists of dynamic type definitions for each unique class in the game. For a given class c_i, its definition is a parameter vector $\theta_{S_{c_i}}$ which consists of

an assignment of values to parameters needed to fully describe a VGDL sprite. The `vgdlType` corresponds to a high-level constraint on the behavior of the object, for example, whether it is an agent with goals, a random agent, a projectile, and so on. The remaining parameters specify details of the object's behavior, such as its speed or orientation. The positions of sprites are updated at each step using simple programs that describe their behavior. For example:

- `nextPos(Missile, speed=2, orientation=Right, pos=(x,y)) = (x+2, y)`
- `nextPos(Random, speed=1, pos=(x,y)) = random.choice([(x,y), (x+1,y), (x-1,y), (x,y+1), (x,y-1)]`

These programs imply a distribution over next positions for any given parameterization; we use this distribution to calculate per-object likelihoods in Equation 5.4.

The InteractionSet, θ_I consists of *rules*, $r_{1:M}$, corresponding to lines in the VGDL InteractionSet. A rule, r_i, is a tuple (c_j, c_k, ϕ, p) that prescribes a corresponding *event*, $e_i = \epsilon(r_i) = (c_j, c_k, \phi)$, such that predicate ϕ is applied to an object of class c_j when that object collides with an object of class c_k and preconditions p have been met. In accord with the intuitive concept of "no action at a distance", all events are triggered only by contact between objects. The relationship between rules and events is many-to-one, as many preconditions may be met for a particular event. For convenience, we introduce the notation $pair(\cdot)$ to invoke the class pair implicated in a rule or event. Because a rule and its corresponding event share the same class pair, $pair(e_i) = pair(r_i)$, which is the ordered pair in the tuples. The predicates $\phi \in \Phi$ are observable state transformations on objects, such as `push`, `destroy`, `pickUp`, and so on. Preconditions in VGDL are restricted to statements of the form $f(\text{agentState}) < N$ or $f(\text{agentState}) > N$, and are meant to capture common contingencies such as, "The Avatar cannot go through doors unless it holds the corresponding key." We use $\pi(r_i)$ to refer to the precondition of rule r_i. When a rule's precondition is met, we write that $\pi(r_i) =$ `True`.

If an event e occurs between a given class pair (c_j, c_k), all the other rules in the theory that correspond to that class pair are expected to occur, provided their preconditions $\pi(r_i)$ are met. We call these expected events *implications*. Formally, the implications $\upsilon(e, \theta)$ of an event are: $\upsilon(e, \theta) = \{\epsilon(r_i) \mid \big(pair(r_i) = pair(e)\big) \wedge \big(r_{i_p} = \texttt{True}\big)\}$. Abusing notation, we will write that $\upsilon(e, \theta) = \texttt{True} \iff \upsilon_j = \texttt{True}, \forall \upsilon_j \in \upsilon(e, \theta)$.

Finally, we use $\eta(e, \theta) = \texttt{True}$ to denote that an event e is *explained* by a theory, θ. An event is explained in a theory if there is a rule in the theory that can account for the event. Formally, $\eta(e, \theta) = \texttt{True} \iff \exists r_i \in \theta_I \mid \big(\epsilon(r_i) = e\big) \wedge \big(r_{i_p} = \texttt{True}\big)$.

The TerminationSet, θ_G, consists of *termination rules* $g_{1:L}$ each of which specifies a function that maps states to $\{Win, Lose, Continue\}$. The VGDL termination rules posit that the episode ends in a win or a loss if some condition is met, but otherwise continues. The conditions in the termination rules are statements that the number of objects of some class must equal some number (usually zero). We use $\gamma_i(s_I \cup s_S)$ to refer to the state prescribed by rule g_i at state (s_I, s_S). If the condition is met, $\gamma_i(s_I \cup s_S) \in \{Win, Lose\}$. Otherwise $\gamma_i(s_I \cup s_S) = Continue$.

5.4.2 Bayesian Learning

The posterior probability of a theory is $p(\theta \mid s_{0:T}, a_{0:T-1}) \propto p(s_{0:T} \mid \theta, a_{0:T-1})\, p(\theta)$. For simplicity, we use a uniform prior over theories.

When an object is not involved in a collision, its movement patterns result only from its type and previous state. By contrast, when an object is involved in a collision, its subsequent state is a function only of the interaction rules that govern the classes of the objects involved in the collision. This enables us to decompose the likelihood function into two components corresponding to s_I (the colliding objects and the events occurring between them) and s_S (the freely moving objects).[2] Once the state of all objects for a time-step has been resolved, the termination conditions can be evaluated, resulting in whether the the environment is in `Win`, `Loss`, or `Continue` status. This

[2]Technically, the fact that two objects are participating in an interaction may be useful for inferring their type, but the simpler decomposition works just as well in practice.

enables us to additionally factorize out the s_G component of the state when we write the likelihood function, below.

The likelihood function can be decomposed as

$$p(s_{0:T} \mid \theta, a_{0:T-1}) = p(s_0) \prod_{t=1}^{T} p(s_{t_G} \mid s_{t_S}, s_{t_I}) p(s_{t_S} \mid \theta_S, s_{t-1}, a_{t-1}) p(s_{t_I} \mid \theta_I, s_{t-1}, a_{t-1}) \tag{5.1}$$

where

$$p(s_{t_G} \mid s_{t_S}, s_{t_I}) = \prod_{l=0}^{L} 1[s_{t_G} = \gamma_i(s_{t_S}, s_{t_I})], \tag{5.2}$$

$$p(s_{t_I} \mid \theta_I, s_{t-1}, a_{t-1}) = \begin{cases} 1 & \big(\eta(e_i, \theta) = \texttt{True}\big) \wedge \big(\upsilon(e_i, \theta) = \texttt{True}\big), \forall e_i \in s_{t_I} \\ 0 & \text{otherwise} \end{cases}, \tag{5.3}$$

and

$$p(s_{t_S} \mid \theta_S, s_{t-1}, a_{t-1}) = \prod_{k=0}^{K} p(o_k \mid \theta_S, o_{t-1}, a_{t-1}) \tag{5.4}$$

for the objects $o_{1:K}$ in the game that are not involved in collisions. The per-object likelihoods are a function of the particular object parameterizations specified by the theory.

The above allows us to maintain a factorized posterior that corresponds to the three things we aim to learn: the dynamic type for each class, the interaction rules that resolve collisions, and the termination rules that explain the win/loss criteria for a game. For the dynamic types we maintain an independent posterior over each parameterization, which we enumerate and update. For the interaction rules, we represent only the *maximum a posteriori* hypothesis, which is easy to update because the interactions are deterministic. That is, when we observe an interaction we assume its corresponding rule is in the MAP hypothesis. When we see a violation of a learned

rule, we assume the violation is explained by some conditional rule instead, so we propose the minimal conditional rule that would explain that. The conditional rules we propose are limited to existential and universal quantifiers over aspects of the agent's state. For the termination rules we maintain a superset of possible explanations, which the planner tries to simultaneously satisfy.

The current verison of EMPA is boostrapped with knowledge of the avatar type (for instance, whether it is a `Shooter` or a `MovingAvatar`), as well as the nature of the projectile the avatar emits, if there is one. Learning these directly would merely expand the size of the hypothesis space for the dynamic types, but would not require any additional inferential mechanisms.

5.5 Theory-Based Exploration

EMPA explores its environment in a way that enables it to quickly reduce uncertainty about its hypothesis space. The parameterizations of the game's objects, corresponding to θ_S in the theory, can be learned by mere observation, as moving objects reveal their type dynamic type simply by moving in their characteristic patterns. The interaction rules θ_I, on the other hand, can only be learned by observing contact events between objects. Some of these occur naturally — a ball may collide with a box and push it, or a predator might capture its prey and eat it. However, many other events, and in fact, most events critical to winning a game, do not happen naturally. The agent has to discover that it can pick up the key in Zelda; it has to learn that boxes can be pushed into holes in Sokoban; and so on.

An optimal exploration policy with the ability to take advantage of a simulator would search agent trajectories and select ones that, over some given horizon, maximized expected information gain (EIG) with respect to the agent's distribution over theories. Given that: 1) EMPA uses a uniform prior over interactions; 2) interactions in VGDL are deterministic, and 3) there is no dependency structure between interaction rules in VGDL, contact events between classes rapidly collapse uncertainty about pairwise interactions to precisely the interactions that have been observed. As

Figure 5-3: Screenshots from Zelda, Frogs, and Bait. Learning how to win these (and many other) games requires reaching objects that are difficult to reach using a naive exploration policy. In order to achieve the first win, the agent must pick up the key and then reach the door while avoiding spiders (Zelda, left); reach the goal flag by first crossing a dangerous road filled with cars and then crossing a river by stepping on moving logs (Frogs, middle); solve a complex puzzle by pushing boxes into holes to clear space in order to reach the key (Bait, right). EMPA is able to learn about all of these win-related objects rapidly precisely because it specifies reaching them as goals, rather than waiting to reach them by chance in order to realize that the objects are, in fact, important to winning the games.

a result, EIG with respect to interaction rules is maximized whenever new pairwise contact events are observed. This leads us to adopt a simple heuristic that is functionally equivalent to calculating EIG at each step: we simply specify the observation of contact events between pairs of objects as first-class goals to be achieved by the planner, and allow the planner to find plans that satisfy these goals without any need for explicit EIG calculations.

The consequences of having theory-based curiosity are vast, as there are many games in which the winning event is extremely unlikely to be brought about by chance, or by any naive exploration policy. Because EMPA's curiosity is based on its explicit theories, it is able to generate long-range plans whose purpose is to bring about specific, informative, events which it can use to quickly master games. (See Figure 5-3 for an illustration).

5.6 Planning

5.6.1 Overview

The EMPA planner takes as input a theory, θ and state, s, and searches action sequences using the algorithm detailed in Algorithm 1. Once a plan is found, it is executed by the agent until the plan ends. If, during execution of the plan, the world state differs sufficiently from EMPA's predicted state, the planner is re-run. We explain re-planning in Section 5.6.5

In order to plan efficiently in games with sparse rewards, EMPA generates intrinsic rewards that operate at three distinct levels: goals, subgoals, and goal gradients. Goals correspond to known and hypothesized termination conditions, as well as to contact goals specified by the curiosity module. Subgoals are decompositions of the known and hypothesized termination goals: since all termination goals specify conditions of the form, $|c_j| = N$, subgoals constitute any change in the right direction, of the number of instances of a class. Finally, goal gradients allow the agent to prefer level configurations that are closer to achieving a subgoal, by taking into account the pairwise distance between objects of classes that are relevant to goals. The full intrinsic reward function is a sum of goal reward, subgoal reward and goal gradient reward: $V(s,\theta) = R_G(s,\theta) + R_{SG}(s,\theta) + R_{GG}(s,\theta)$. We explain intrinsic rewards further in Section 5.6.3

5.6.2 Planning modes

The planner operates in three modes: 'long-term', 'short-term', and 'survive'. 'Long-term' mode returns the first plan that leads to a known or hypothesized win state, as well as any plan that satisfies the goals of the curiosity module. The default 'short-term' mode returns any plan that fulfills the above criteria, as well as any plan that fulfills any subgoal. That is, if the number of objects of a class moves closer to the count specified by any termination condition, a subgoal has been reached and 'short-term' mode returns a plan. 'Survive' mode simply returns short plans that do not

result in `Loss` states; this mode is often run when the other two modes have failed to return a satisfactory plan. Table B-1 shows EMPA's policy for determining which mode to plan in, and Table B-2 shows the parameter settings that constitute each planning mode.

5.6.3 Intrinsic Rewards

The full intrinsic reward function is a sum of goal reward, subgoal reward and goal gradient reward: $V(s,\theta) = R_G(s,\theta) + R_{SG}(s,\theta) + R_{GG}(s,\theta)$. All planner modes return a plan if a goal state is reached and stop searching states that stem from loss states, so $R_g(s,\theta)$ is effectively Inf for `Win` states, $-Inf$ for `Loss` states, and 0 otherwise.

By contrast, subgoal rewards only cause the 'short-term' planner to return a plan, but they drive all planners' intrinsic rewards. Taking advantage of the fact that all termination goals specify conditions of the form $|c_j| = N$, states are penalized proportional to their distance from object-count goals. Specifically, the subgoal reward for a state s is

$$R_{SG}(s,\theta) = \rho_1 \sum_{g_i \in \theta_T} \frac{N_{g_{i_c}} - |c(g_i)|}{|c(g_i)|^2} (-1)^{1[g_i = Win]}$$

where $N_{g_{i_c}}$ is the class count specified by g_i and $|c(g_i)|$ is the actual class count in the state, of the class specified in g_i. ρ_1 calibrates the relative value of subgoal reward to goal gradient reward, ρ_2; We use $\rho_1 = 100\rho_2$.

The numeric subgoal reward operates at the abstract level of object counts. Changes in this quantity are too sparse to guide search on their own, so the planner uses 'goal gradients' to score states at a lower level of abstraction by maximizing

$$R_{GG}(s,\theta) = \rho_2 \sum_{g_i \in \theta_T} \frac{d_{min}\big(c'(g_i), c(g_i)\big)}{|c(g_i)|^2} (-1)^{1[g_i = Win]}$$

where $c'(g_i)$ refers to the class (if one exists) that can destroy items of class $c(g_i)$, and $d_{min}(c_j, c_k)$ refers to the distance between the most proximal instances of classes c_j and c_k. This biases the agent to seek proximity between objects that need to be destroyed and the objects that can destroy those objects, and vice-versa for objects

that need to be preserved.

The specifics of the above intrinsic rewards (which particular objects to approach, avoid, ignore, and so on) are generated by the planner within each game by analyzing EMPA's highest-posterior theory. For example, if the highest-posterior theory has $|c_j| = 0$ as a `Win` condition for some class, c_j, the planner finds all classes c'_j that the theory believes can destroy c_j and generates goal gradients as specified above. We view the ability to generate such abstract heuristics that work for any game in the hypothesis space as a strength of our approach: just as a strong domain theory allows EMPA to make rapid inductive leaps about the agent's environment, a strong domain theory for planning, specified at the same level of abstraction of objects, relations, and goals, allows the planner to efficiently find high-value action sequences across a wide variety of games.

5.6.4 State Pruning

The planner ameliorates the exponential explosion of the state space by using Iterative Width (Geffner and Lipovetzky, 2012; Lipovetzky and Geffner, 2017), which prunes states that are not sufficiently different from states previously encountered in the course of search. This is achieved by defining *atoms*, logical propositions that can evaluate to `True` or `False`. We indicate the presence of each object in the state with the atom, (`object`, `True`), and we use (`object`, `False`) for a known object that is no longer in the game state. We encode the location of each object with the atom, (`object`, `posx`, `posy`). For the agent avatar, whose transition probabilities are additionally affected by its orientation, we use a (`object`, `posx`, `posy`, `orientation`) tuple. The algorithm maintains a table of all atoms that have been made true at some point in the search originating from a particular actual game state. During search, IW(1) prunes any state that fails to make some atom true for the first time in the search; IW(2) prunes any state that fails to make some pair of atoms true for the first time; and so on. For small k, IW prunes aggressively and quickly searches all states it can represent. By contrast, for large k, IW is able to make fine-grained distinctions between states (e.g., it being the first time in the search tree that all

three of (`object1`,2, 2) `AND` (`object2`, `Null`, `Null`) `AND` (`object3`, 1, 4) have each had that particular value), but as a result many more states that may be functionally equivalent are treated as distinct, often making search hopelessly slow.

In games with many moving objects, IW (k) can experience sufficient novelty over a great many states without the agent needing to make a move; this leads to very inefficient search. To incentivize the agent to move, we additionally use a penalty on repeated agent positions, $\alpha \cdot \texttt{count(agent, posx, posy, orientation)}^2$, with $\alpha \in \{-1, -10\}$. To further ameliorate IW's sensitivity to moving objects, we do not generate atoms for the locations of objects hypothesized to be random or whose presence in the game is predicted to last fewer than 5 seconds.

5.6.5 Re-planning

In all planning modes, the planner takes as input an (s_t, θ) pair and returns a list of high-value actions, $a_{t:N}$, together with their corresponding predicted states from the simulator, $\hat{s}_{t+1:N+1}$. When the agent takes an action a_t, the true environment returns $s_{t+1} \sim p_{a_t}(s_{t+1}, s_t)$. After each action, agent compares the true s_{t+1} to the predicted $\hat{s}_{t+1}$. If the agent is in sufficient danger or if its own position is not as predicted, the planner is run again; otherwise, the agent continues execution of the action sequence. Danger is determined by the theory and the state: being next to an object that kills the agent is not dangerous if the object is thought to be inert, but it is if the object is thought to move randomly. The last row of Table B-1 shows the specific conditions that trigger re-planning.

5.7 Towards Learning Theories Without Fully Observable Events

As described in Section 5.4.1 of this chapter, EMPA learns from a factorized state representation that consists of s_I, the pairs of objects that are participating in events as well as the nature of those events, s_S, which refers to the remaining objects and their

positions, and s_G, which refers to the `Win/Loss/Continue` status of the environment. Below we outline an approach that is able to learn theories without receiving event labels as part of s_I; that is, we outline an approach that is able to learn theories simply from object positions and the `Win/Loss/Continue` status of the environment. This approach would modify the "Model" component of EMPA and would leave the agent otherwise untouched.

5.7.1 Main Loop

The main loop of our algorithm performs a theory-induction step after each action, by comparing the state predicted by each theory in the hypothesis set to the actual observed state, proposing modifications, and filtering the new hypothesis space according to a pre-specified threshold τ. See Algorithm 2 for details.

Given that our proposals aim to make local, greedy, modifications to the theory, it is possible that updates to a theory that reduce its prediction error at time t might increase its error at some previous time-step, $t - k$. Proposals are therefore always evaluated on a replay of previously-encountered transitions.

The main loop runs by evaluating and expanding on a constantly-changing set of hypotheses. We initialize this set to be a single hypothesis: that all objects are solid and stationary. This has implications for the effective nature of our prior over theories; a discussion of this is beyond the scope of this synopsis.

5.7.2 Likelihood Function and Distance Metric

We define a distance metric d_{θ_i} over states such that

$$p(s_{t+1} \mid s_t, a_t, \theta_i) \propto 1 - d_{\theta_i}(s_{t+1}, s_{t+1}^{\theta_i})$$

where s_{t+1} is the true resulting state and $s_{t+1}^{\theta_i} \sim T_{\theta_i}(s_t, a_t)$.

We first match each object o_{t+1} to its nearest match, $o_{t+1}^{\theta_i}$ so that we can evaluate

$$d_{\theta_i} = \prod_{o \in Objects} d_{\theta_i}(o_{t+1}, o_{t+1}^{\theta_i})$$

We then define:

$$d_{\theta_i}(o_{t+1}, o_{t+1}^{\theta_i}) = \begin{cases} 1 - \epsilon & pos(o_{t+1}) = pos(o_{t+1}^{\theta_i}) \\ \epsilon & \text{otherwise} \end{cases}$$

to reflect the assumption that the correct theory will correctly predict the correct subsequent state for all deterministic transitions. For sprite types that our θ_i hypothesizes to be stochastic such as the the `RandomNPC` and the `Chaser`, we instead use a likelihood function derived from the VGDL update function for the type. That is, instead of getting a single prediction for $o_{t+1}^{\theta_i}$, we obtain a distribution over next positions from the program itself and use that as the likelihood. This is only feasible in a small set of cases and so does not constitute the bulk of our likelihood evaluations.

5.7.3 Error Signal and Theory Proposal

When we observe a game state s_t that is not well-explained by θ_i, we propose modifications to θ_i that may help to explain the observation. We denote the proposal function by

$$g : \Theta \to \Theta^m$$
$$g(\theta_i) = \left(\tilde{\theta}_i^{(1)}, \ldots, \tilde{\theta}_i^{(m)}\right),$$

where $\left(\tilde{\theta}_i^{(1)}, \ldots, \tilde{\theta}_i^{(m)}\right)$ is a set of modified theories that aim to better match the observations. Since we are incrementally building a tree of theories, and since theories are expensive to evaluate, we want to maintain a space of theories that is as small as possible while ensuring that we do not fail to propose good theories. To this end, instead of making random modifications to an inadequate theory we use an **error signal** to guide our proposal function.

5.7.4 Error Signal

After observing the true state evolution $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t, \theta^*)$, we compare it to our predicted state $s_{t+1}^{\theta_i} \sim p(s_{t_1}^{\theta_i} \mid s_t, a_t, \theta_i)$ for each theory. In addition to computing a distance between predicted and observed states, we can analyze the nature of the difference to provide a diagnosis of the possible causes of the prediction error. For example, we can return simple diagnoses that are easily calculated from the predicted and actual states, such as:

- unexpectedDisappearance(c_1)
- unexpectedAppearance(c_2)
- unexpectedPosition(c_3)

In these examples, c_1, c_2, c_3 are termed the *target classes.* We can also calculate other information that will be helpful in diagnosing the error. For example, given that by design in VGDL, all effects are applied only when there is contact between objects, we can easily calculate the set of objects that may have made contact with the target class at the time of the prediction error. By calculating possible neighbors of a target sprite in the previous time step, we can generate a list of possible culprit classes. Here there is a tradeoff – a liberal calculation of the possible culprits will result in a larger set of proposed child theories; a less liberal calculation will result in fewer proposed child theories, but may result in missing the correct culprit class. For a particular state transition and theory, our error signal function returns a set of tuples (diagnosis, interactionPair, targetClass, culpritClasses). The proposal function takes a theory and one such tuple and returns a set of child theories that contain possible fixes to the error. There are two components to this function, corresponding to the two components of the VGDL theory that we are ultimately trying to learn: the the sprite types of the objects, and the interaction set.

5.7.5 Factorized Proposals

Sprite type proposals We remind the reader that in VGDL, sprite types (object properties) govern the motion properties of each object. As a result, only diagnoses that involve unexpected object positions can be explained by different sprite-type hypotheses. When we receive such a diagnosis, we simply enumerate all possible parameterizations of sprite hypotheses. That is, we generate the cross-product of all the possible values of all the features a sprite type can have: $\mathbf{F} = \{f_1\} \times \{f_2\} \times \cdots \times \{f_N\}$, where each row $\mathbf{f}_i$ of $\mathbf{F}$ is a particular feature combination, i.e., a particular parameterized sprite type. Example features and values are:

- Type: (Immovable, Missile, Random)
- Speed: $[0, 0.1, \cdots, 5]$
- Orientation: [up, down, left, right]
- Cooldown: $[0, 1, \cdots, 10]$

This set of hypotheses is generated in the same way every time we want to generate possibilities for a particular class. Given these hypotheses, we are immediately in a position to evaluate them. We have implemented a likelihood function $p(pos(o)_{t+1} \mid pos(o)_t, \mathbf{f})$, where $pos(o)_t, pos(o)_{t+1}$ are the locations of a particular sprite token, o, at successive time steps. This function takes into account the position-updating functionality of VGDL. For example, for a particular object we might have $\mathbf{f}$ = (Type=Missile, Speed=1.7, Orientation=left, Cooldown=0). For this particular object, the likelihood function yields

$$p(x_{t+1}, y_{t+1} \mid x_t, y_t, \mathbf{f}) = \begin{cases} 1 - \epsilon & (x_{t+1}, y_{t+1}) = (x_t - 1.7, y_t), \\ \epsilon & \text{otherwise} \end{cases}$$

In general, likelihood functions over positions for objects are very sparse, so the end result is that while we might initialize $20,000$ feature combinations for a particular object, only 20 or so features explain the observed transition with non-negligible

probability. With these surviving hypotheses, we can generate $|\text{rows}(\mathbf{F}_{surviving})|$ children of θ_i, where the feature set denoted by each row is taken to be the sprite type for the target class. All other aspects of θ_i are left unchanged.

Interaction rule proposals In addition to generating all sprite-type variants for θ_i for a particular target class, we also generate all InteractionSet variants. This is achieved in the procedure detailed in Algorithm 3.

The function `predicates(`δ`)` maps a given diagnosis, δ, to a list of predicates that can potentially fix the error. For example:

- `predicates`(unexpectedPosition) = [`push, stepBack, reverseDirection`]
- `predicates`(newObjectAppeared) = [`cloneSprite, transformTo`]

The most straightforward thing to do would be to map each diagnosis to all predicates (or to not use diagnoses at all), but given the combinatorial nature of this theory-generation process, it is in practice very helpful to keep this number small. The mapping we use has been specified by us, but it is possible for an agent to learn it from experience, over long time-scales, i.e., by optimizing the diagnoses-to-predicate sets mapping with respect to the number of proposed theories it takes to learn a high-fidelity simulator for a particular game.

5.8 Human-Level Theory-Based Reinforcement Learning

In addition to developing EMPA, we directly compared humans, EMPA, and other computational models on a set of 90 challenging video games, many of which require nontrivial exploration and long-range planning. Our core set of games consists of 10 original games and 17 games based on ones from the General Video-Game AI (GVGAI) competition (Perez-Liebana et al., 2016). For each of these 27 core games, we generated 1-4 variant games that roughly matched their originals in difficulty and subjective 'interestingness' (see Figures A-1 and A-2). The full set of 90 games

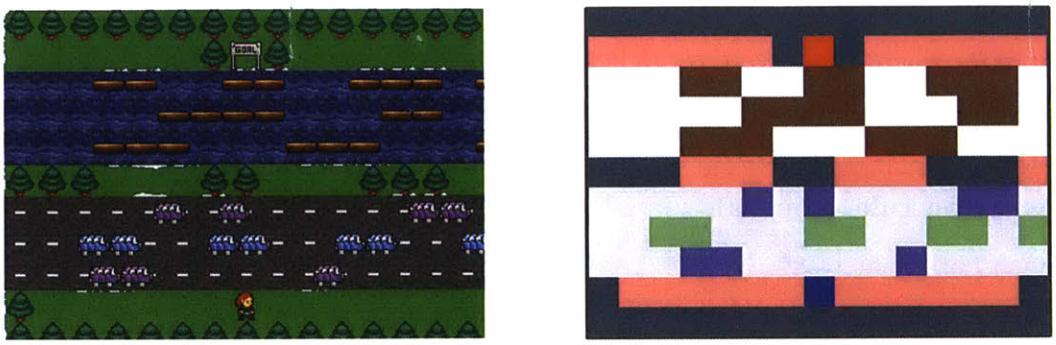


Figure 5-4: 'Normal' and 'no-color' versions of the GVGAI game, "Frogs". All human participants and DDQN played 'no-color' versions; EMPA received a state description containing object positions and interactions occurring between objects.

is described in Appendix D, and can be played on a Google Chrome browser at vgdl.herokuapp.com/games.

300 human participants were recruited through Amazon Mechanical Turk and paid $3.50 plus a bonus of up to $1.00 depending on their cumulative performance across all the games they played. Prior to playing the games, they were told only that they could use the arrow keys and the spacebar, and that, beyond that, they should try to figure out how each the game worked in order to play well. Each participant played 6 games in random order; 20 participants played each game. No participant played more than one version of a game.[3]

In order to test learning as it occurs independently of semantic content conveyed by object appearance, we displayed all games in 'color-only' mode, where objects differ from one another only in their color. Additionally, color assignments were randomized across subjects to ensure that no color-related semantic associations were conveyed to subjects. See Figure 5-4 for screenshots of "Frogs" in 'normal' and 'color-only' mode.

Humans were compared to a leading Deep-Reinforcement-Learning agent (DDQN) Van Hasselt et al. (2016), EMPA, and various lesioned versions of EMPA, designed to highlight the relative contributions of its modules. The DDQN implementation was borrowed from https://github.com/dxyang and was run with the hyperparameters

[3]That is, if a participant played "Frogs", they played no "Frogs" variants.

listed in Table B-3. EMPA received the game state as described in Chapter 5.4.1.

5.8.1 Results

As in Atari, people can learn to play these games and generalize across levels in just a few minutes (several hundred agent steps; see Figure 5-5), whereas DDQN often takes orders of magnitude more experience. Figure A-3 shows performance of humans, EMPA, and DDQN across all 90 games over the first $10,000$ steps, and Figure A-4 shows performance over the first 1 million steps.

The goals and dynamics in these games vary considerably: in Avoidgeorge (top left), the player has to give candy to annoyed citizens while an enemy chases citizens in order to annoy them, and wins as long as 500 game steps pass without all citizens becoming annoyed. Here DDQN wins one level by chance but fails to learn a good policy quickly. EMPA takes unnecessary actions in the game and therefore wins more slowly than people, but still learns the game within a few thousand steps. In Bait (top right), the player has to push boxes into holes in the ground in order to reach a key that unlocks the exit door. EMPA performs as well as a median human does, solving all five levels in fewer than $1,000$ steps, while DDQN fails to solve a single level of the game even in $1,000,000$ steps. In "Butterflies", the player has to chase all the butterflies before the butterflies reach all the cocoons. DDQN takes $200,000$ steps before winning the game, despite the simplicity of the goal and policy; EMPA performs as well as the best humans.

To more concisely evaluate the performance of these learning agents, we propose a performance metric,

$$\text{performance} = \frac{\text{levels completed}}{\text{levels in game}} \times \frac{\text{levels completed}}{\text{steps to completion}}$$

This notion incorporates two natural notions of concern — the degree to which the task was completed, and the efficiency with which it was completed. Figure 5-6 shows the distribution of human-normalized performance scores across all games, for EMPA and DDQN. EMPA and humans are most often within an order of magnitude of one

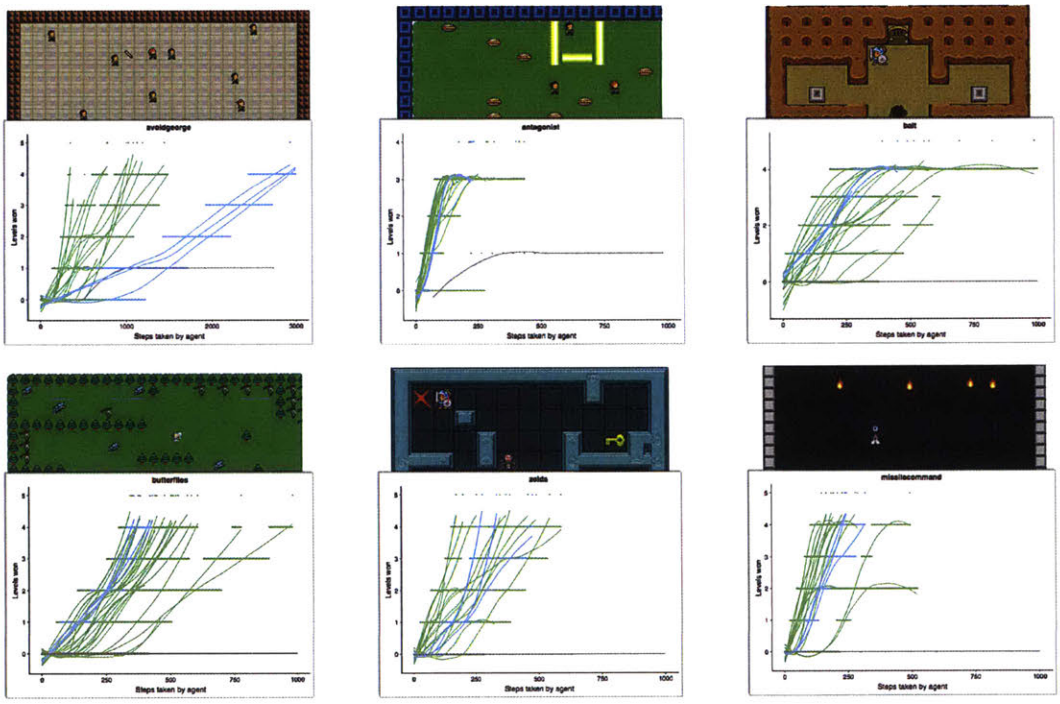


Figure 5-5: Screenshots and learning curves for six games, for humans (green), EMPA (blue), and DDQN (grey). Humans and EMPA are able to learn new games in a matter of minutes (corresponding to a few hundred steps).

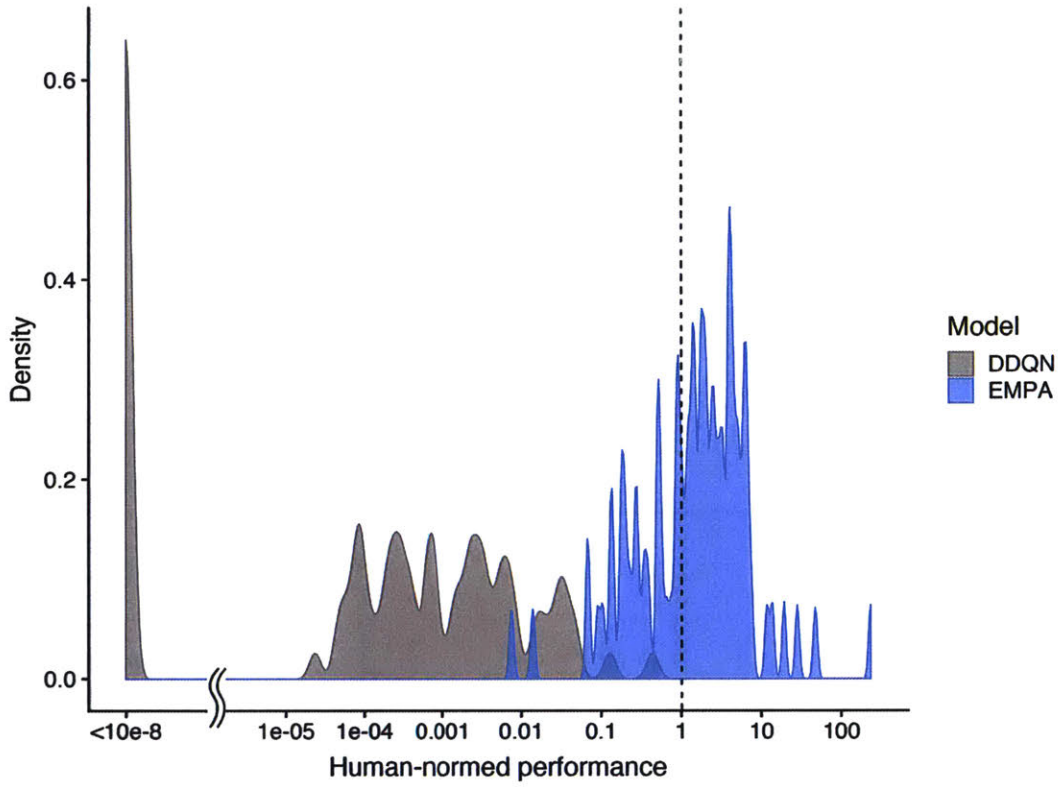


Figure 5-6: Distribution of EMPA and DDQN learning performance relative to humans across 90 games.

another, whereas DDQN falls orders of magnitude below humans on most games. Figure 5-7 shows model scores on this metric for all 90 games.

5.8.2 Role of theory-based curiosity

To examine the role of EMPA's theory-based curiosity we compared EMPA to a lesioned variant of EMPA that does not generate epistemic goals. This version still performs inference over VGDL theories, and employs the full planner to achieve goals. However, the goals provided to the planner are only win-related. As a result, the model only learns from object motion and events that occur independently of the agent, or agent-caused events that occur by chance as the agent moves through the game according to a standard ϵ-greedy exploration policy.

One difficulty raised by this lesion arises from the nature of goals that are gener-

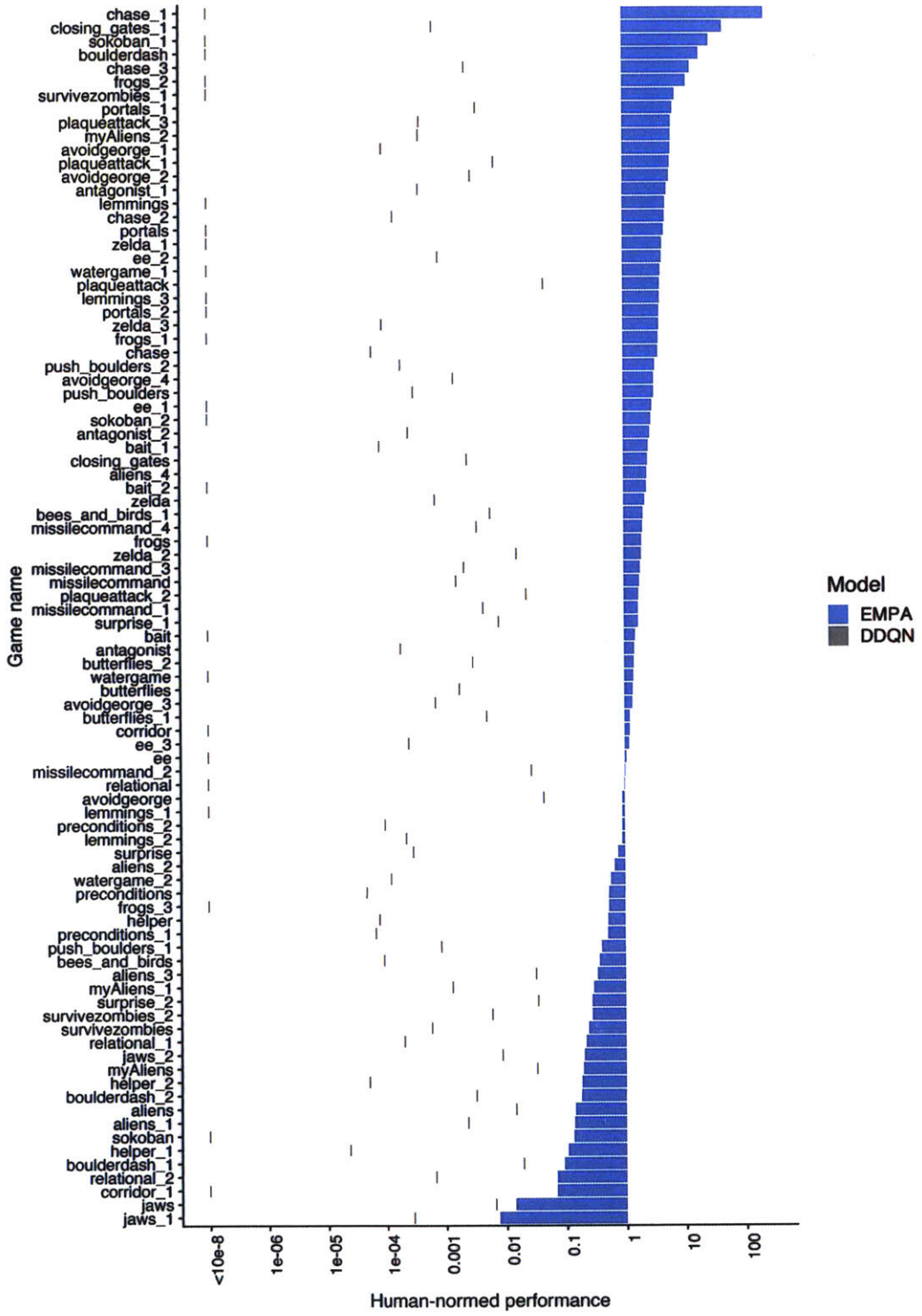


Figure 5-7: EMPA learning performance relative to humans and DDQN across 90 games.

ated and passed to the EMPA planner: EMPA only hypothesizes win conditions for classes it has learned are able to disappear in the game. In the absence of having witnessed such events, such goals do not exist and the planner is not given win- or curiosity-related goals. As a result, the planner is not able to return a win state. Given the metacontroller policy, this results in switching to long-term planning and then rapidly increasing the maximum allowable nodes searched by the planner, leading to a scenario where the planner spends a great many cycles only to return no plan. To ameliorate this problem, we ran an ϵ-gredy 'DS' variant that delays switching to long-term planning, and that re-sets the maximum number of nodes allowed under long-term planning to the initial value after any level forfeit.

For both 'vanilla' ϵ-greedy and DS-ϵ-greedy, we annealed the ϵ linearly from 1 to 0.05 over the course of $1,000$ and $2,000$ steps. Figure 5-8 shows the distribution of performance scores (computed in log space) across all games for DDQN, EMPA, and the four variants. Figure 5-9 shows means and 95% CIs for these scores. Figure 5-10 shows results for the 31 games for which all variants completed at least one level. Taken together, these results suggest that a nontrivial part of EMPA's success can be attributed to its ability to solve the significant exploration challenges posed by these games.

5.8.3 Role of planning heuristics

To examine the role of EMPA's planning module, we compared EMPA to lesioned variants that did not use 1) subgoals, 2) goal gradients, 3) IW, 4) subgoals or goal gradients, 5) subgoals, goal gradients, or IW.

Figure 5-11 shows results pooled results across the 90 games (means and 95% CIs computed in log space). Individual lesions perform at worst two orders of magnitude worse than EMPA and humans, but the combined lesions decrease the model's performance by three orders of magnitude, suggesting that EMPA benefits greatly from its intrinsic rewards and node pruning.

The contributions of the goal-related and planning heuristics do not only occur at the behavioral level; they also occur at the planning level. That is, not only do

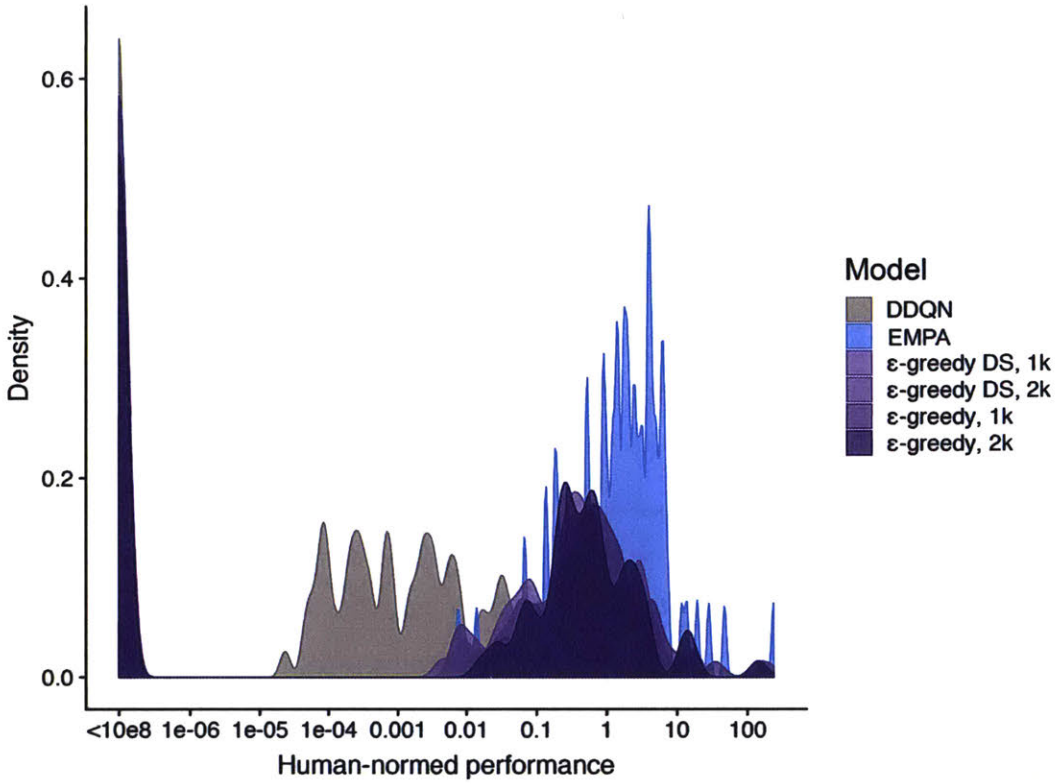


Figure 5-8: Distribution of performance scores for DDQN, EMPA, and four variant exploration lesions of EMPA. Certain games pose significant exploration problems that neither DDQN nor the lesioned variants can solve.

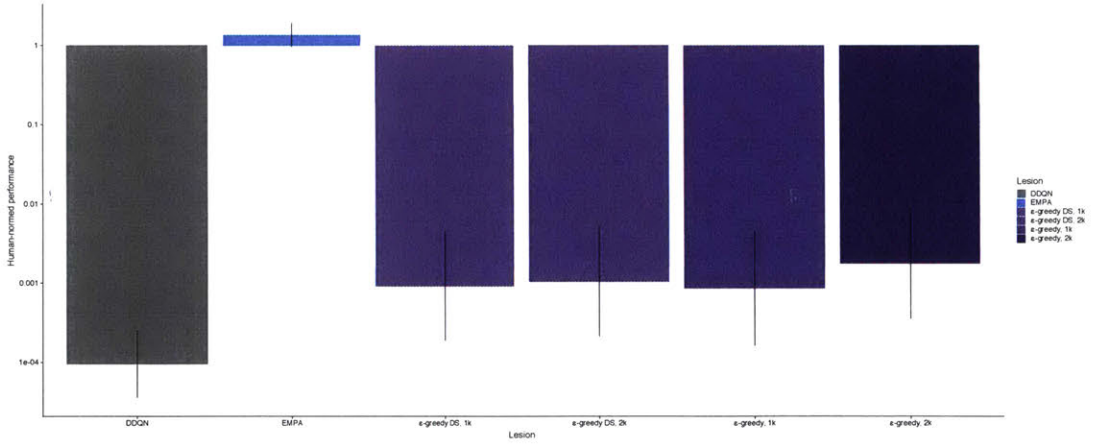


Figure 5-9: Means and 95% CIs (computed in log space) of performance scores across all 90 games.

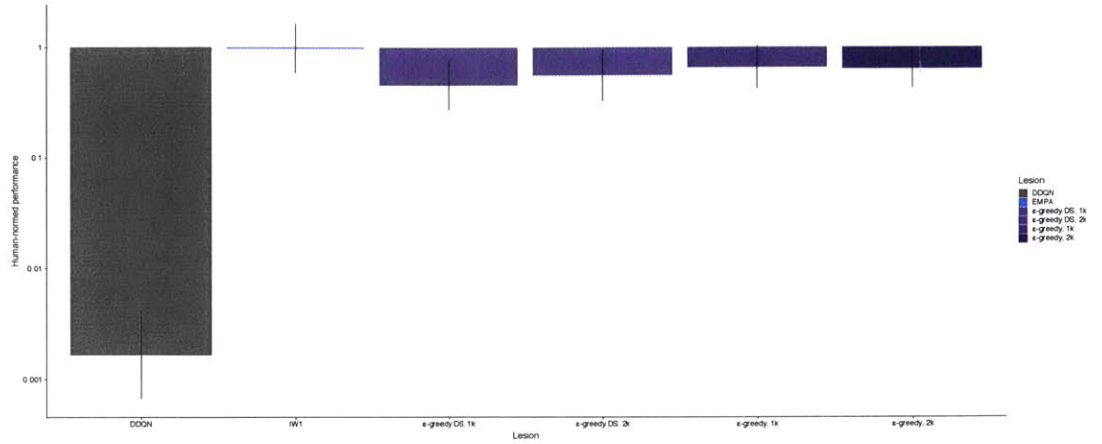


Figure 5-10: Means and 95% CIs (computed in log space) of performance scores across the 31 games for which all variants were able to solve at least one level. For games without significant exploration challenges, ϵ-greedy exploration contributes only slightly to a decrement in EMPA's performance. DDQN continues to significantly underperform humans and EMPA.

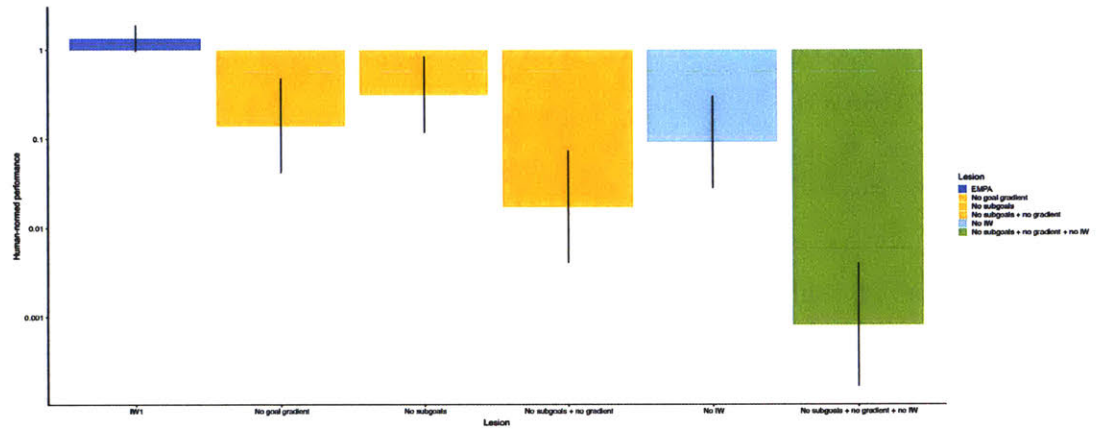


Figure 5-11: The combination of EMPA's intrinsic reward structure and node-pruning allows it to perform up to 1000 times better than a lesioned variant that lacks these components.

the final plans become worse; but the process of planning becomes less efficient. To examine this effect, we examine the planner efficiency of each model, m_j, which we define as

$$\text{planner efficiency} = \frac{\min(\text{planner-nodes-per-step}(m_i \in M))}{\text{planner-nodes-per-step}(m_j)}$$

Figure 5-12 shows results for each lesion pooled across the 90 games.

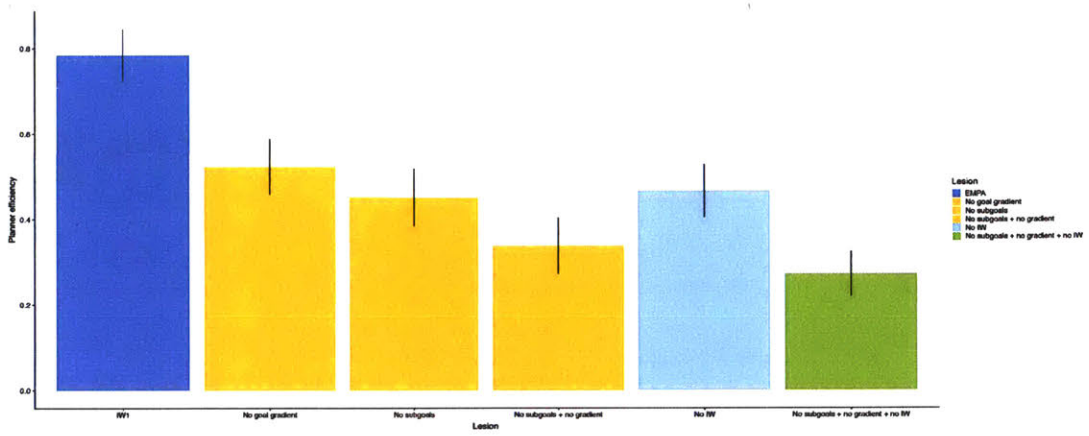


Figure 5-12: The combination of EMPA's intrinsic reward structure and node-pruning allow it to plan roughly twice as efficiently than a lesioned variant that lacks these components. Note that this metric only considers planner nodes per step taken, and not the quality of the actions.

5.9 Discussion

Our Exploring, Modeling, Planning algorithm (EMPA) is able to learn to play a range of video games with just a few minutes of experience, as humans do. It accomplishes this by using human-inspired inductive biases, specified at the level of objects, relations and goals. These inductive biases are coupled with an imperative to explore in a theory-oriented way, and adaptive behavior more generally is made possible by means of a planner that can generate subgoals and goal gradients from the hypothesized theory and use these to effectively guide search.

We believe that model-learning, curiosity, and planning are the main drivers of human performance on video-game tasks. To make it easier to focus on these, we bootstrapped EMPA with the full game state — including the ability to perceive events when they occur — rather than making it operate directly on pixel input. While it is trivial to pick out objects in VGDL, the use of events to learn theories simplifies the learning problem considerably, as they help to distinguish between theories that can produce otherwise-identical states. More importantly, video-game

domains outside of VGDL do not provide the agent with labeled events as they occur. While it is relatively straightforward to perform object detection in video-game domains, event detection may pose a more challenging problem. Chapter 5.7 outlined an approach that can learn theories directly from object locations, without the need to use events at all. Work is underway to produce a version of EMPA that combines a simple perception front-end with this approach in order to operate directly on pixels.

We designed EMPA using human-inspired inductive biases about objects, relations, and goals. But in fact the types of objects, relations, and goals that EMPA is able to consider perfectly match the types of objects, relations, and goals that can be composed to generate the environments EMPA was tested in; VGDL was both the representation used for EMPA's hypothesis space and the one we used to generate the set of 90 games. The learning problem is still substantial; as noted in Chapter 5.3, for a game with just 10 unique classes, there are over 10^{37} possible game descriptions. Still, the *a priori* significant overlap between the algorithm's hypothesis space and the distribution from which the environments were implicitly drawn raises the question of whether a theory-driven approach could learn to operate in environments that were produced using a different "world engine". Stated more clearly, what remains to be done before EMPA can play games of roughly equivalent difficulty to VGDL (for example, Atari games) that are generated using a different "world engine" than the one it has?

We believe that, in fact, not very much is missing. There are various real-world notions that can be incorporated into EMPA's representation language to make it significantly more capable. These include: more realistic and more flexible physics; the ability to define sets of objects with coordinated properties; slightly more sophisticated agent models; action-at-a-distance (such as when a light switch turns on a light that is not visibly connected to the switch); and more generally, a decoupled notion of cause and effect. An updated representation language, coupled with a stronger game engine to support simulation worlds with these properties, is well underway.

Chapter 6

Conclusion

This thesis examined some of the epistemic practices, representations, and algorithms that we believe underlie humans' ability to quickly learn about their world and to deploy that understanding to achieve their aims. In particular, it examined humans' ability to effectively query their environment for information that helps distinguish between competing hypotheses; children's ability to use higher-level amodal features of data to match cause and effects; and adult human rapid-learning abilities in artificial video-game environments. The thesis culminated by presenting and testing our Exploring, Modeling, Planning agent (EMPA) that learns to perform complex video-game tasks at human-level performance and with human-level amounts of experience. The model is an instantiation of a more general approach, Theory-Based Reinforcement Learning, which we believe can underlie the development of human-level agents that may eventually learn and act adaptively in the real world.

There are many frontiers on which we plan to continue this work. On the very near term, we aim to leverage two components of EMPA's ability to learn models of its environment. As mentioned in Chapter 4, humans benefit greatly from observing a few minutes of other players' gameplay; EMPA can likely do the same. More interestingly, humans can use what they have learned about a game to achieve completely novel goals that were not part of the original environment; EMPA can likely do this, as well, and we would like to showcase both of these abilities.

We would also like to examine the degree to which our human-level EMPA al-

gorithm is also human-like. That is, can EMPA pass a Turing test? Can humans distinguish between videos of EMPA gameplay and human gameplay? To what degree do its trajectories through novel environments resemble those of human learners? Do the planning problems that challenge EMPA also challenge humans, and vice-versa? We hope to use the insights from the above to reveal more about the representations and algorithms that humans use when engaged with these tasks, and to use what we learn from human experiments to develop more capable Theory-Based RL agents.

Concretely, we are planning several projects that will help advance the project of building agents that can operate in increasingly complex worlds. As mentioned in 5.9, a core ongoing project is to develop a richer space of theories that would be capable of representing games that were generated using different "world engines". The richer hypothesis space would in turn necessitate the use of more sophisticated inference procedures, and would raise a natural question of whether the current heuristic approximation to optimal theory-based curiosity would continue to be effective in these richer environments. It would also necessitate the development of more powerful planning algorithms, with a more flexible intrinsic reward structure. A promising avenue in this direction may be for EMPA to learn planning heuristics through imagined play, using a combination of bottom-up deep-learning methods and program synthesis techniques. We also plan to combine EMPA with Deep Reinforcement Learning methods in order to learn a fast, reactive policy that can be deployed in games that do not require long-range planning, where the EMPA planner is used to generate training data for the policy.

Perhaps the most ambitious extension of this work concerns the very nature of the approach. Theory-Based Reinforcement Learning is not intended as a model of how intelligence evolves from scratch, as Deep Reinforcement Learning researchers often set as their target. Rather it is a reflection of the end-result of human evolution: flexible but powerful inductive biases that can be deployed to learn new tasks quickly, even in environments very different from our experience at the level of pixels or raw observations. In the short term, we plan to develop hybrid approaches that combine theory-based inductive biases to quickly learn components of new tasks, with slower

but more flexible program-synthesis techniques that can learn new primitives, in order to capture learning both on fast and slower timescales. In the long term, we would like to develop program-synthesis approaches powerful enough to learn the entire representation language that is capable of supporting theory-based, human-level learning.

Appendix A

Figures

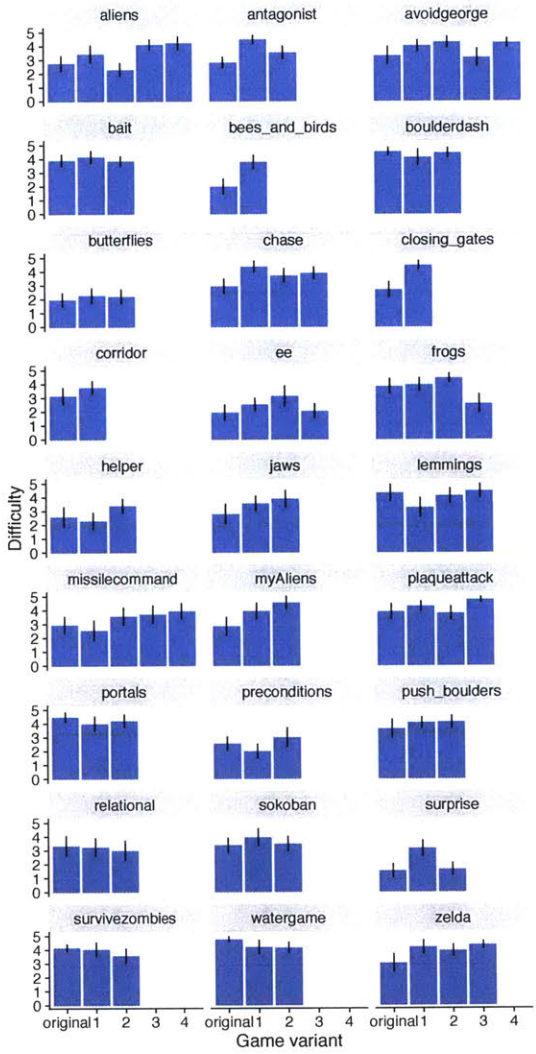


Figure A-1: Mean and 95% confidence intervals of each game's difficulty as rated by human subjects, grouped by source game.

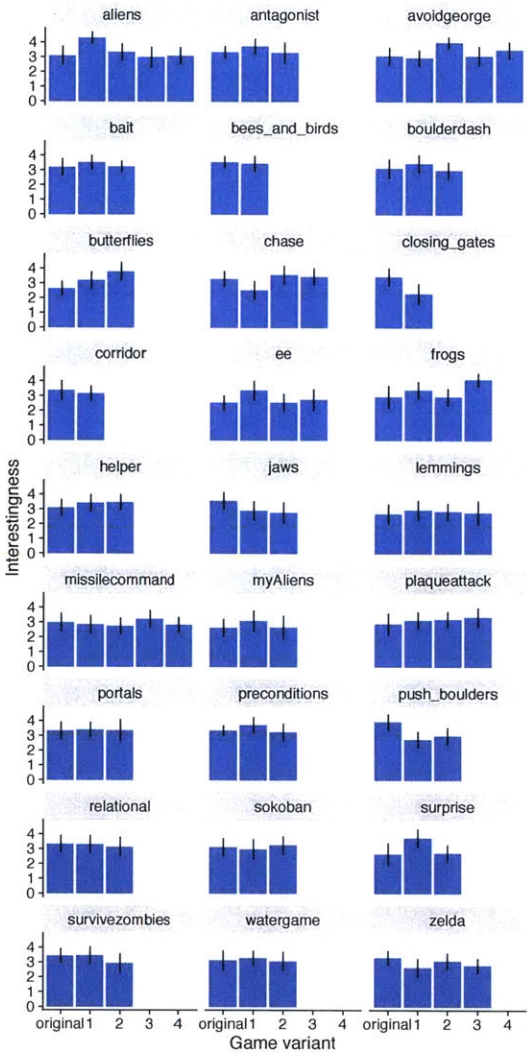


Figure A-2: Mean and 95% confidence intervals of human subjects' interestingness judgments for each game, grouped by source game.

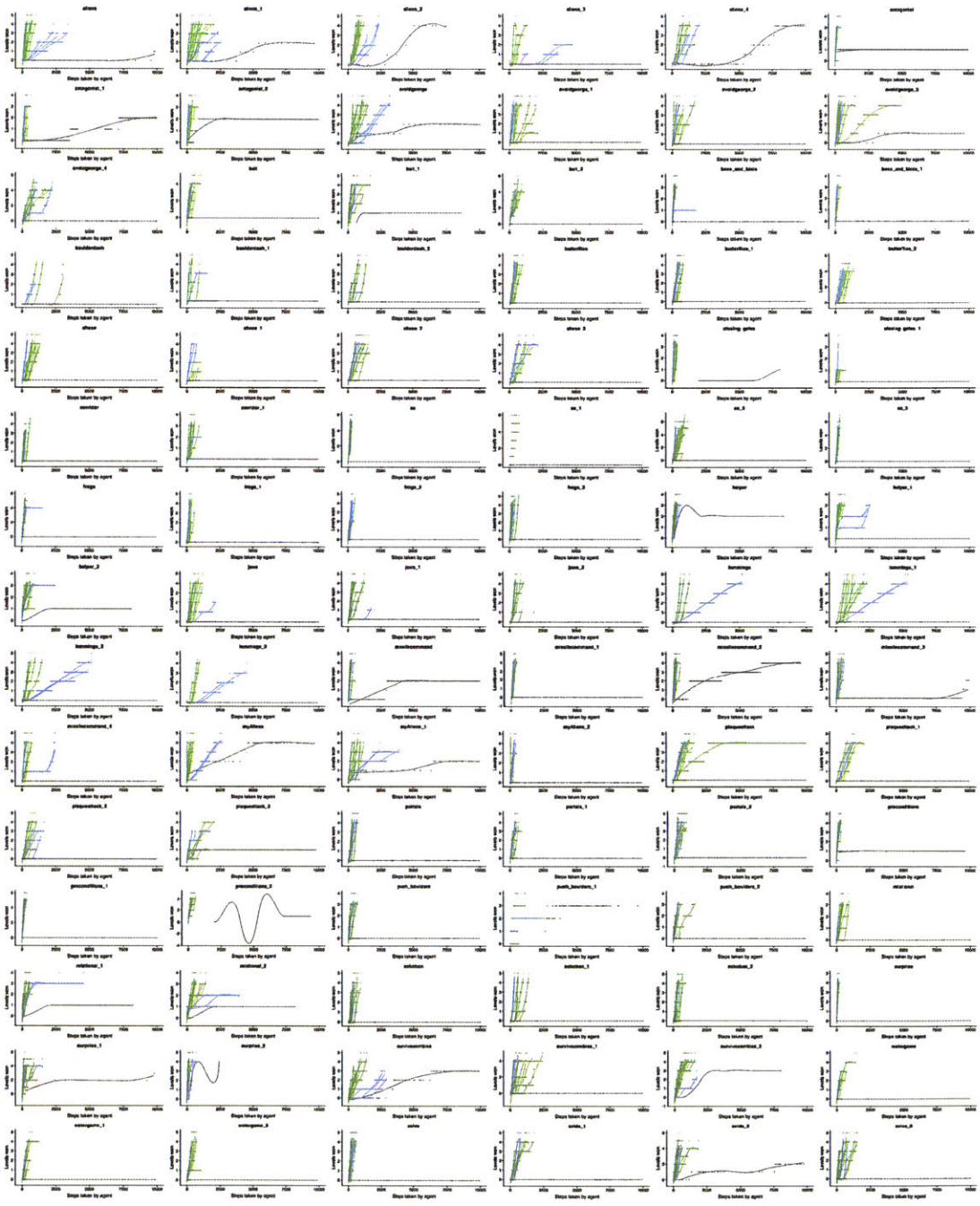


Figure A-3: Humans (green), EMPA (blue), and DDQN learning curves over the initial 10,000 steps of play.

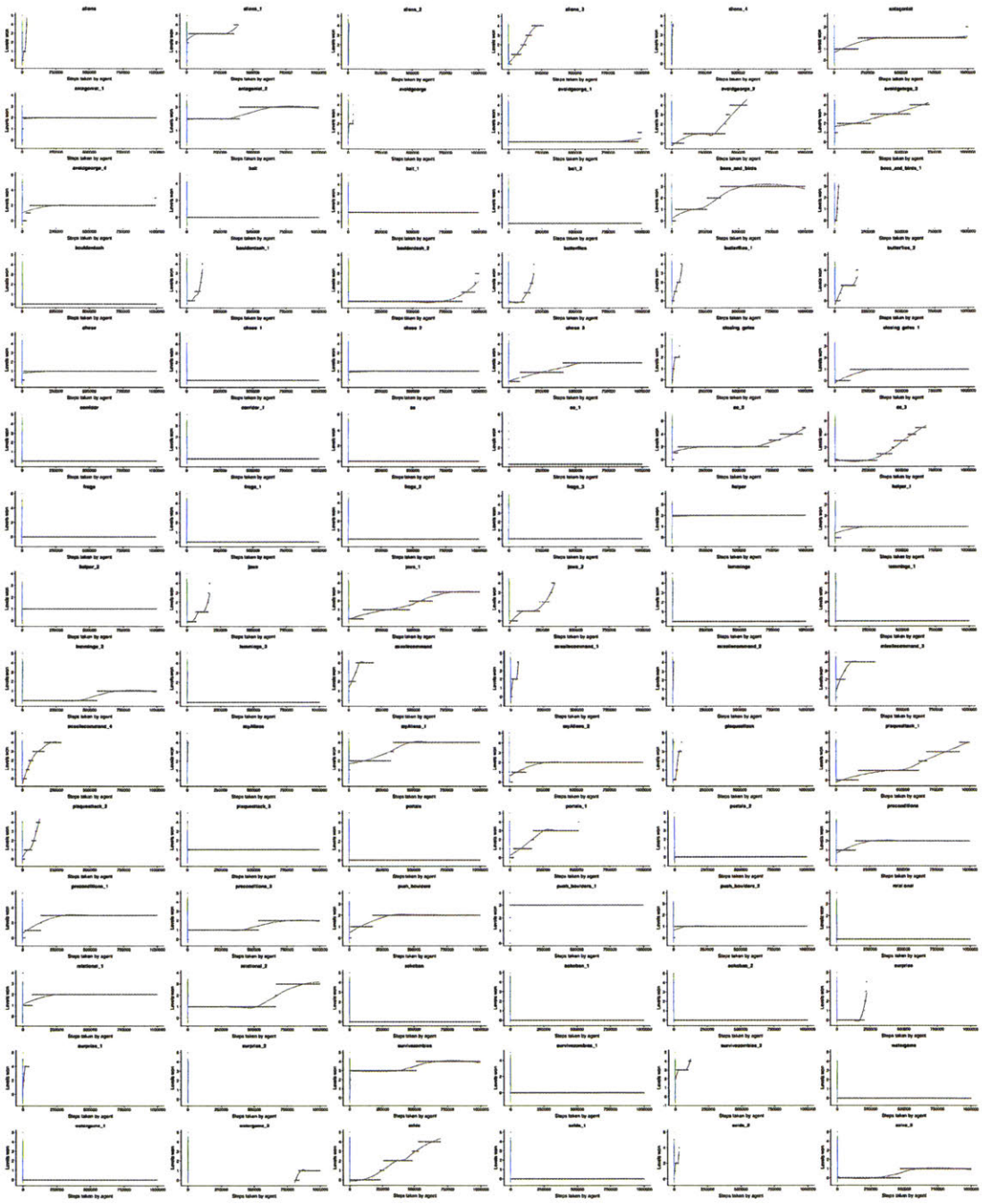


Figure A-4: Humans (green), EMPA (blue), and DDQN learning curves over the initial 1 million steps of play.

Appendix B

Tables

Planner mode	Conditions	Behavior
Short-range	Found plan	Execute plan
Short-range	Did not find plan AND (count(repeated-deaths)==2) OR (no-new-objects(55) AND (no-moving-types() OR no-scorechange()))	Switch to long-range
Short-range	Did not find plan AND NOT (count(repeated-deaths)==2) OR (no-new-objects(55) AND (no-moving-types() OR no-scorechange()))	Switch to survive
Long-range	Found plan	Execute plan
Long-range	Did not find plan AND long-range-plan-failures>2	max-nodes = τ•max-nodes long-range-plan-failures = 0 restart(level)
Long-range	Did not find plan AND long-range-plan-failures<2	long-range-plan-failures += 1 Take 5 random steps
Survive	Always	Execute plan Switch to short-range
All	Agent not where expected OR (dangerous-object not where expected) AND (distance(dangerous-object, Agent)<3)	Re-plan

Figure B-1: Planning-mode selection policy. "repeated-deaths" examines the state history and returns true if the same events occurred during successive lost episodes. "no-new-objects(N)" returns TRUE if no new object has emerged on the screen for N game-steps. "no-scorechange" returns TRUE if the score has not changed in the last game step.

	Short-term	**Long-term**	**Survive**
Starting max_nodes	~ {200, 500, 1000}	1000	50
Annealing factor (τ)	1	1	2
Return plan on reaching subgoal	TRUE	FALSE	TRUE

Figure B-2: Parameters that constitute EMPA's distinct planning modes.

Parameter	Value
α	0.00025
γ	0.999
τ	100
Annealing schedule for ε	Linearly from 1 to .1 over 200 steps
Experience-replay max(steps)	50,000
Batch size	32
Image input recrop size	64x64x3

Figure B-3: DDQN hyperparameters.

Appendix C

Algorithms

Algorithm 1

```
1: function PLAN(s_0, maxNodes, plannerMode)
2:     numNodes, frontier, nodes = 0, [s_0],[s_0]
3:     while numNodes < maxNodes do
4:         s = argmax([v(s) for s in frontier])
5:         frontier.remove(s)
6:         for a ∈ actions do
7:             s' ~ p(s' | s, a, θ)
8:             numNodes + = 1
9:             if novel(s', IW) then
10:                frontier.add(s')
11:            end if
12:            if winCriterion(s', plannerMode) then
13:                return actions(s'.sequence), s'.sequence
14:            end if
15:        end for
16:        s' = argmax([v(s) for s in nodes])
17:        return actions(s'.sequence), s'.sequence
```

Algorithm 2

function RUNEPISODE($s_1, a_{1:T-1}$)
  for $t = 1 : T - 1$ **do**
    for $\theta_i \in$ hypothesisSet **do**
      likelihood, error = errorSignal($\theta_i, (s_t, s_{t-1}, a_{t-1})$)
      $\theta_{new_1} : \theta_{new_M}$ = proposals(θ_i, error)
      hypothesisSet.extend($\theta_{new_1} : \theta_{new_M}$)
    end for
    hypothesisSet = filter(hypothesisSet, τ)
  end for

Algorithm 3

```
1: function EXPAND(θ, δ, targetClass, culpritClasses)        ▷ δ: A diagnosis
2:     proposals = []
3:     for pair ∈ targetClass × culpritClasses do
4:         for π ∈ predicates(δ) do
5:             θ_new = copy(θ)
6:             θ_new_Ip = π          ▷ Replace all predicates that involve pair with π
7:             proposals.append(θ_new)
8:         end for
9:     end for
```

Appendix D

Game Descriptions

Antagonist The player has to eat all the hot dogs before the antagonists eat the burger. The player can win by playing keepaway with the burger while trying to reach the hot dogs, but in the last level there are too many hot dogs to eat and only one burger — the only way to win is to push the burger through a forcefield that the antagonists cannot get to.

Antagonist 1 The player can no longer play keepaway with the burger, but can push boxes into the antagonists' path to block them.

Antagonist 2 The hot dogs are harder to get to. To make things a bit easier, the player can put frosting on the burgers, which makes the antagonists not recognize their food. There's a smarter antagonist that isn't hungry but takes the frosting off the burgers, making them recognizable again. The player has to balance frosting burgers and eating hotdogs in order to win.

Aliens (GVGAI) This is the VGDL version of the classic "Space Invaders". The player moves a spaceship left and right and can shoot surface-to-air missiles. Alien bombers move side-to-side and rain bombs at the agent, who can hide under a base. The base slowly gets destroyed, so the player must often dodge bombs while shooting all the aliens.

Aliens — Variant 1 The bombs shot by the aliens move randomly.

Aliens — Variant 2 The way to win now is to destroy the protective base.

Aliens — Variant 3 The player's surface-to-air missiles move three times as

slowly, making aiming more challenging.

Aliens — Variant 4 Aliens move three times as fast, shoot bombs six times as often, and the player's base no longer destroys incoming bombs. To compensate, the player can now shoot surface-to-air missiles continuously.

Avoidgeorge (GVGAI) Evil George chases citizens. If he touches them, they become annoyed, and if there are no calm citizens in the game, the player loses. To avoid this, the player can feed the annoyed citizens candy, which makes them calm down. If George touches the player, the player dies. Keeping citizens calm for 500 game steps results in a win.

Avoidgeorge — Variant 1 The player can now use the candy to make annoyed citizens disappear, and if all annoyed citizens disappear, the player wins.

Avoidgeorge — Variant 2 The player can still make annoyed citizens disappear, and if all annoyed citizens disappear, the player wins. George moves faster, but the player can now throw the candy.

Avoidgeorge — Variant 3 Same as Variant 2, but the player begins the game stuck in a room and needs to tunnel out of it before doing anything else.

Avoidgeorge — Variant 4 Same as Variant 3, but George moves even faster, and the candy no longer makes citizens disappear.

Bait (GVGAI) A puzzle game. The player has to pick up a key to go through a door to win. Falling into water results in death, but pushing boxes into water vaporizes the water and clears the path.

Bait — Variant 1 Same as the original, but there are a few laser cannons lying around that vaporize any water in the laser's path.

Bait — Variant 2 The levels are a little easier in this variant, but the player has to make the key by pushing metal into a mold. Only then can one go to the door to win.

Bees and Birds The player has to get the goal while avoiding swarming bees. In level 2, the goal is surrounded by an electric fence, but the player can release a bear that can eat through the fence. In level 3, the player can either go through a dangerous swarm of bees or take a longer path that involves releasing a bear that eats

the fence. In level 4, the player can get to the goal simply by going around the fence. Releasing the bears will actually cause the bees to be released, making winning more challenging.

Bees and Birds — Variant 1 The player has to get the goal while avoiding swarming bees. In all subsequent levels, there are lots of swarming bees, so the best stategy is to release a mockingbird that will eat the bees, clearing the path to the goal.

Boulderdash (GVGAI) This is the VGDL version of the classic "Dig Dug". The player has to pick up 9 diamonds and then exit the level. Boulders in the game are supported by dirt, and if a boulder falls on the player, the player dies. The player has to avoid two types of subterranean enemies to avoid dying.

Boulderdash — Variant 1 Same as the original, except that one of the subterranean enemies converts dirt to diamonds instead of being trapped by it.

Boulderdash — Variant 2 Same as the original, except the player only has to pick up three diamonds to win.

Butterflies The player has to get all the butterflies before the butterflies activate all the cocoons. The butterflies move randomly, and if they touch a cocoon, the cocoon turns into a butterfly.

Butterflies — Variant 1 Same as the original, but butterflies are faster.

Butterflies — Variant 2 There are more cocoons, meaning there will soon be many more butterflies. The player can destroy cocoons, but still dies if all cocoons are gone.

Chase (GVGAI) The player has to chase all the goats, which run away. Touching a goat turns it into a carcass. If a scared goat touches a carcass, the carcass turns into a zombie goat which chases and can kill the player.

Chase — Variant 1 Same as the original, except that there are a few gates that release goats sporadically.

Chase — Variant 2 Touching a goat kills it, rather than turning it into a carcass. The game now contains a wolfgate, which if the player shoots, releases a wolf that quickly chases and kills the player.

Chase — Variant 3 In addition to the goats, there are sheep that rapidly pace in a predetermined path. If these touch a carcass, they turn into zombies.

Closing Gates The player has to get to the exit before large sliding gates close.

Closing Gates — Variant 1 The gates close more quickly. The only way for the player to escape is to shoot a bullet onto the gates' path; this blocks the gates and lets the player squeeze through.

Corridor The player has to get to the exit at the end of a long corridor while avoiding boulders of different speeds that roll toward the player.

Corridor — Variant 1 The player can now shoot bullets at the boulders. Depending on the speed of a boulder, the bullet will either slow it down or speed it up (in fact, the speed of any boulder can be toggled in a cycle by hitting it with more bullets).

Explore/Exploit The board is full of colorful gems, and the player wins by picking up all gems of any given color. Different board configurations incentivize exploring new ways of winning versus exploiting known ways.

Explore/Exploit — Variant 1 Same as the original game, except evil gnomes chase the player as the player collects the gems.

Explore/Exploit — Variant 2 There are no evil gnomes, but now the gems are being carried by gnomes that flee the player.

Explore/Exploit — Variant 3 Same as the original, except one of the gems the player encounters early on is poisonous.

Helper The player has to help minions get to their food. Sometimes the food is blocked by a fence the player can destroy, and sometimes the minions are boxed-in by a fence that the player can push food through.

Helper — Variant 1 The player wins by eating all the food, but can only eat it after taking it to the minions for processing.

Helper — Variant 2 The player's goal is once again to feed minions. In this variant the player can shoot a path through red fences to free minions. In the last level, the player has manually clear one path, then shoot through some fences, and then push food through a third fence in order to feed the minion.

Frogs (GVGAI) This is the classic "Frogger". The player has to cross a road while avoiding dangerous trucks and then step carefully on moving logs to cross a river, to get to an exit. The level layouts of this game are slightly modified from the original GVGAI layouts, in order to make the game playable in the original version of VGDL this project was built on.

Frogs — Variant 1 The trucks move differently now; they move at different speeds, and when they hit the edge of the screen they rapidly turn around and drive in the other direction.

Frogs — Variant 2 Now there aren't any logs to help the player cross the water, but the player can throw mud at the water to build bridges.

Frogs — Variant 3 Back to the original layout, but things are made much simpler by the presence of a teleporter.

Jaws (GVGAI) The player dies if touched by a chasing shark. Cannons on the side of the screen shoot cannonballs at the player. The player can shoot bullets at the cannonballs, to convert them into a new kind of metal. Picking this metal up gives points. The player wins by surviving for 500 game steps.

Jaws — Variant 1 Now there is a fence that prevents the player from getting too close to the cannons.

Jaws — Variant 2 Each piece of metal grants 5 health points; having any health points protects the player from the shark; the shark takes away one health point each time there's contact. The shark can be destroyed if the player has 15 health points. The player wins after destryoing the shark or surviving for 500 game steps.

Lemmings (GVGAI) In this classic game, the player has to help a group of lemmings get to their own exit. To do so, the player must shovel a tunnel, and incur a loss of points with every shovel action.

Lemmings — Variant 1 Now the player gets points for shoveling, rather than losing points.

Lemmings — Variant 2 The player can release a mole, which loves to shovel and will clear all the dirt. But if the mole happens to touch the player, the player dies.

Lemmings — Variant 3 The mole is still there and will shovel dirt when released. But if the mole touches a lemming, the mole turns into a snake that chases and kills the player.

Missile Command (GVGAI) Spaceships want to destroy the player's bases. The player defends the bases by shooting short-range lasers, and wins upon destroying all the spaceships, or loses after all the bases are destroyed.

Missile Command — Variant 1 The bases don't stay in place; they float around randomly.

Missile Command — Variant 2 The player can now shoot long-range lasers. However, these lasers bounce off the edges of the game and kill the player upon contact.

Missile Command — Variant 3 A newer, faster spaceship, tries to destroy the player's base, and the player is only equipped with a short-range laser.

Missile Command — Variant 4 There are more fast spaceships, making it very difficult to destroy all of them before they get to the bases. However, the player can shoot the short-range laser at the ozone layer, which transforms into a shield that the enemies cannot pass through.

MyAliens (GVGAI) The player can only move sideways. Fast-moving bombs drop from the top. One type of bomb kills the player; the other gives points. Surviving the onslaught for 500 game steps results in a win.

MyAliens — Variant 1 The player can now collect the safe bombs, and can go to an exit once five of those have been collected.

MyAliens — Variant 1 The player can now move in all directions, but bombs come from all directions, too. The player wins by either picking up five safe bombs and getting to the exit, or by surviving for 500 game steps.

Plaqueattack (GVGAI) 40 Burgers and hotdogs emerge from their bases and attack cavities; if they get all the cavities the player dies. The player wins by destroying all the hotdogs and burgers with a laser.

Plaqueattack — Variant 1 The burgers and hotdogs move faster, but there are only 24 of them. The player has to destroy them while avoiding a roving drill.

Plaqueattack — Variant 2 There is no drill, but now there are bouncing projectiles that kill the player on contact.

Plaqueattack — Variant 3 Now the player has to face 40 burgers/hotdogs again, as well as the bouncing projectiles, but can win by destroying all the gold fillings, instead.

Portals (GVGAI) The player wins by picking up the green gem. Certain rooms, including the room with the gem, are accessible only by going to portals that teleport the player to their portal-exits. Most portals have multiple portal-exits, and the player's final location is randomly chosen from those. While figuring this out, the player has to avoid bouncing missiles, roving space-monsters, and inert traps.

Portals — Variant 1 Same dynamics as above, but with a new portal type and more complicated level layouts.

Portals — Variant 2 Uses the complex layouts as in Variant 1. The player now has to contend with roving monsters that can pass through walls.

Preconditions The player wins by picking up a diamond. Getting the differently-colored fake diamond does nothing. Usually the diamond is surrounded by poison, which the player can pass through only after drinking an antidote. In the last level the player has to drink an antidote to pass through the poison that blocks two antidotes, in order to pass through the double-poison in order to get to the triple antidote, in order to get through the triple poison that guards the goal.

Preconditions — Variant 1 The player can also pick up a more powerful antidote that allows passage through two poisons. Different level layouts encourage using either the single or double antidote.

Preconditions — Variant 2 Same rules as the original, except only one antidote can be ingested at a time, forcing the player to shuttle back and forth.

Push Boulders The goal is to get to the exit. Touching orange or yellow fire results in a loss. Green boulders can be pushed by the player; these destroys orange fire and push yellow fire. Light blue clouds can be destroyed by the player but otherwise have no effect on the game. The game often involves solving maze-like challenges and necessitates using boulders to clear dangerous obstacles.

Push Boulders — Variant 1 Purple boulders destroy yellow fire (but they don't destroy or push orange fire). The mazes are more difficult and involve multiple uses of boulders to clear fire.

Push Boulders — Variant 2 Now the purple boulders are too heavy to move, but one-time-use-only cannons shoot cannon balls that destroy any boulders or fire in their path. The player has to fire up the right cannons to clear paths. Light yellow clouds do nothing.

Relational The goal is to make all the blue gems disappear, which happens when they are pushed into a vat of yellow potion. In the second level, there is no yellow potion, but touching the green potion converts it to yellow. In the third level there is no green or yellow potion, but pushing an orange box into a purple converts the two into yellow potion. In the fourth level, a pink box can be converted into an orange one that can be pushed into purple to make the yellow potion.

Relational — Variant 1 Same relational rules as above, but the blue gem is now carried by a gnome that chases the player. While nothing happens if the gnome touches the player, the player has to move around in such a way as to get the gnome to run into yellow gems.

Relational — Variant 1 Now the gnome is fleeing the player. While nothing happens if the player touches the gnome, the player has to move around in such a way as to get the gnome to run into yellow gems.

Sokoban (GVGAI) In this classic puzzle game, the player wins by pushing boxes into holes.

Sokoban — Variant 1 The player can make use of portals, which teleport either the player or boxes to a particular portal-exit location on the game screen.

Sokoban — Variant 2 This variant does not have portals. Instead, there is dirt lying around that can fill holes. Filling certain holes makes the levels unwinnable.

Surprise The player has to pick up all of the red apples. Touching a cage releases a randomly-moving Tasmanian devil. If the Tasmanian devil touches a green gem, the Tasmanian devil gets cloned and the gem disappears. The player can pass through gems and Tasmanian devils

Surprise — Variant 1 The player can no longer pass through the Tasmanian devils; releasing the devils makes winning much more challenging.

Surprise — Variant 2 The goal is to destroy all the gems, which can only be done by releasing the Tasmanian devil.

Survive Zombies (GVGAI) The player has to survive for 500 game steps while avoiding zombies that emerge from fiery gates. The fiery gates are dangerous, too. Fortunately, bees come out of flowers, and when bees touch zombies, the two collide and turn into honey. Eating honey gives the player temporary immunity from zombies.

Survive Zombies — Variant 1 The way to win is to collect all the honey.

Survive Zombies — Variant 2 The way to win is now to kill all the zombies, by first eating some honey for immunity.

Watergame (GVGAI) A puzzle game. The player has to reach an exit, which is usually surrounded by water that drowns the player. Pushing dirt onto water clears the water.

Watergame — Variant 1 Red boulders can be pushed onto the dirt, thereby destroying it. Destroying some dirt is necessary in order to clear space, but destroying the wrong pieces of dirt makes levels unwinnable.

Watergame — Variant 2 More dirt can be made by mixing light-green and yellow potions.

Zelda (GVGAI) The player has to pick up a key and get to a door, while using a sword to destroy randomly-moving monsters.

Zelda — Variant 1 The player has to pick up three keys before getting to the door.

Zelda — Variant 2 The player only needs one key, now, but both the key and the door move randomly throughout the level.

Zelda — Variant 3 Now the key and door are carried by elves that flee the player.

Bibliography

Baillargeon, R. (2004). Infants' physical world. *Current directions in psychological science*, 13(3):89–94.

Baillargeon, R., Li, J., Ng, W., and Yuan, S. (2009). An account of infants' physical reasoning. *Learning and the infant mind*, pages 66–116.

Bloom, P. (2000). *How children learn the meanings of words*. MIT Press, Cambridge, MA.

Bonawitz, E. B., van Schijndel, T. J., Friel, D., and Schulz, L. (2012). Children balance theories and evidence in exploration, explanation, and learning. *Cognitive psychology*, 64(4):215–234.

Brown, N. and Sandholm, T. (2018). Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424.

Carey, S. (1978). The child as word learner.

Carey, S. (1985). *Conceptual Change in Childhood*. MIT press.

Christie, S. and Gentner, D. (2010). Where hypotheses come from: Learning new relations by structural alignment. *Journal of Cognition and Development*, 11(3):356–373.

Cook, C., Goodman, N. D., and Schulz, L. (2011). Where science starts: Spontaneous experiments in preschoolers' exploratory play. *Cognition*, 120(3):341–349.

Csibra, G. (2008). Goal attribution to inanimate agents by 6.5-month-old infants. *Cognition*, 107(2):705–717.

Csibra, G., Bíró, S., Koós, O., and Gergely, G. (2003). One-year-old infants use teleological representations of actions productively. *Cognitive science*, 27(1):111–133.

Denison, S. and Xu, F. (2010). Twelve-to 14-month-old infants can predict single-event probability with large set sizes. *Developmental science*, 13(5):798–803.

Diuk, C., Cohen, A., and Littman, M. L. (2008). An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 240–247. ACM.

Dubey, R., Agrawal, P., Pathak, D., Griffiths, T. L., and Efros, A. A. (2018). Investigating human priors for playing video games. *arXiv preprint arXiv:1802.10217*.

Geffner, H. and Lipovetzky, N. (2012). Width and serialization of classical planning problems.

Gentner, D. (1977). Children's performance on a spatial analogies task. *Child development*, pages 1034–1039.

Gentner, D. and Markman, A. B. (1997). Structure mapping in analogy and similarity. *American psychologist*, 52(1):45.

Gick, M. L. and Holyoak, K. J. (1980). Analogical problem solving. *Cognitive psychology*, 12(3):306–355.

Gil, Y. (1994). Learning by experimentation: Incremental refinement of incomplete planning domains. In *Machine Learning Proceedings 1994*, pages 87–95. Elsevier.

Gopnik, A., Glymour, C., Sobel, D. M., Schulz, L., Kushnir, T., and Danks, D. (2004). A theory of causal learning in children: causal maps and bayes nets. *Psychological review*, 111(1):3.

Gopnik, A. and Meltzoff, A. N. (1997). *Words, thoughts, and theories.* MIT Press.

Gopnik, A. and Wellman, H. M. (2012). Reconstructing constructivism: Causal models, bayesian learning mechanisms, and the theory theory. *Psychological bulletin*, 138(6):1085.

Guestrin, C., Koller, D., Gearhart, C., and Kanodia, N. (2003). Generalizing plans to new environments in relational mdps. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pages 1003–1010. Morgan Kaufmann Publishers Inc.

Guo, X., Singh, S., Lee, H., Lewis, R. L., and Wang, X. (2014). Deep learning for real-time atari game play using offline monte-carlo tree search planning. In *Advances in neural information processing systems*, pages 3338–3346.

Gweon, H., Tenenbaum, J. B., and Schulz, L. (2010). Infants consider both the sample and the sampling process in inductive generalization. *Proceedings of the National Academy of Sciences*, 107(20):9066–9071.

He, F. S., Liu, Y., Schwing, A. G., and Peng, J. (2016). Learning to play in a day: Faster deep reinforcement learning by optimality tightening. *arXiv preprint arXiv:1611.01606.*

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. (2017). Rainbow: Combining improvements in deep reinforcement learning. *arXiv preprint arXiv:1710.02298.*

Jern, A. and Kemp, C. (2013). A probabilistic account of exemplar and category generation. *Cognitive psychology*, 66(1):85–125.

Johnson, S., Slaughter, V., and Carey, S. (1998). Whose gaze will infants follow? the elicitation of gaze-following in 12-month-olds. *Developmental science*, 1(2):233–238.

Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. (2017). Schema networks: Zero-

shot transfer with a generative causal model of intuitive physics. *arXiv preprint arXiv:1706.04317*.

Kiley Hamlin, J., Ullman, T., Tenenbaum, J., Goodman, N., and Baker, C. (2013). The mentalistic basis of core social cognition: experiments in preverbal infants and a computational model. *Developmental science*, 16(2):209–226.

Lagnado, D. A. and Sloman, S. (2004). The advantage of timely intervention. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 30(4):856.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

Landau, B., Smith, L. B., and Jones, S. S. (1988). The importance of shape in early lexical learning. *Cognitive development*, 3(3):299–321.

Langley, P., Simon, H., Bradshaw, G., and Zyikow, J. (1987). *Scientific discovery: Computational explorations of the creative processes*. MIT press.

Lewkowicz, D. J. and Turkewitz, G. (1980). Cross-modal equivalence in early infancy: Auditory–visual intensity matching. *Developmental psychology*, 16(6):597.

Lipovetzky, N. and Geffner, H. (2017). Best-first width search: Exploration and exploitation in classical planning. In *AAAI*, pages 3590–3596.

Lombrozo, T. (2009). Explanation and categorization: How "why?" informs "what?". *Cognition*, 110(2):248–253.

Magid, R. W., Sheskin, M., and Schulz, L. E. (2015). Imagination and the generation of new ideas. *Cognitive Development*, 34:99–110.

Markant, D. and Gureckis, T. (2012). Does the utility of information influence sampling behavior? In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 34.

Markman, E. (1989). *Categorization and Naming in Children: Problems of Induction*. Bradford Books. MIT Press.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Murphy, G. L. and Medin, D. L. (1985). The role of theories in conceptual coherence. *Psychological review*, 92:289–316.

Murphy, G. L. and Ross, B. H. (1994). Predictions from uncertain categorizations. *Cognitive psychology*, 27(2):148–193.

Navarro, D. J. and Perfors, A. F. (2011). Hypothesis generation, sparse categories, and the positive test strategy. *Psychological review*, 118(1):120.

Nelson, J. D. (2005). Finding useful questions: on bayesian diagnosticity, probability, impact, and information gain. *Psychological review*, 112(4):979.

Nelson, J. D., Divjak, B., Gudmunsdottir, G., Martignon, L. F., and Meder, B. (2013). Children's sequential information search is sensitive to environmental probabilities. *Cognition*, 130:74–80.

Nelson, J. D., Divjak, B., Gudmunsdottir, G., Martignon, L. F., and Meder, B. (2014). Children's sequential information search is sensitive to environmental probabilities. *Cognition*, 130:74–80.

Oaksford, M. and Chater, N. (1994). A rational analysis of the selection task as optimal data selection. *Psychological Review*, 101(4):608.

Pasula, H., Zettlemoyer, L. S., and Kaelbling, L. P. (2004). Learning probabilistic relational planning rules. In *ICAPS*, pages 73–82.

Pasula, H. M., Zettlemoyer, L. S., and Kaelbling, L. P. (2007). Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research*, 29:309–352.

Perez-Liebana, D., Samothrakis, S., Togelius, J., Schaul, T., Lucas, S. M., Couëtoux, A., Lee, J., Lim, C.-U., and Thompson, T. (2016). The 2014 general video game playing competition. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(3):229–243.

Rips, L. J. (1975). Inductive judgments about natural categories. *Journal of verbal learning and verbal behavior*, 14(6):665–681.

Rips, L. J. and Hespos, S. J. (2015). Divisions of the physical world: Concepts of objects and substances. *Psychological bulletin*, 141(4):786.

Rothe, A., Lake, B. M., and Gureckis, T. (2017). Question asking as program generation. In *Advances in Neural Information Processing Systems*, pages 1046–1055.

Schaul, T. (2013). A video game description language for model-based or interactive learning. In *Computational Intelligence in Games (CIG), 2013 IEEE Conference on Games*, pages 1–8. IEEE.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. (2015). Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Schlottmann, A., Ray, E. D., Mitchell, A., and Demetriou, N. (2006). Perceived physical and social causality in animated motions: Spontaneous reports and ratings. *Acta psychologica*, 123(1):112–143.

Scholz, J., Levihn, M., Isbell, C., and Wingate, D. (2014). A physics-based model prior for object-oriented mdps. In *International Conference on Machine Learning*, pages 1089–1097.

Schulz, L. (2012). The origins of inquiry: Inductive inference and exploration in early childhood. *Trends in cognitive sciences*, 16(7):382–389.

Schulz, L. and Bonawitz, E. B. (2007). Serious fun: preschoolers engage in more exploratory play when evidence is confounded. *Developmental psychology*, 43(4):1045.

Schulz, L., Goodman, N. D., Tenenbaum, J. B., and Jenkins, A. C. (2008). Going beyond the evidence: Abstract laws and preschoolers' responses to anomalous data. *Cognition*, 109(2):211–223.

Shen, W.-M. and Simon, H. A. (1989). Rule creation and rule learning through environmental exploration. In *IJCAI*, pages 675–680.

Shinners, P. (2011). Pygame. `http://pygame.org/`.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484.

Sloman, S. A. and Lagnado, D. (2005). The problem of induction. *The Cambridge handbook of thinking and reasoning*, pages 95–116.

Spelke, E. S. (1990). Principles of object perception. *Cognitive science*, 14(1):29–56.

Spelke, E. S. and Kinzler, K. D. (2007). Core knowledge. *Developmental science*, 10(1):89–96.

Spence, C. (2011). Crossmodal correspondences: A tutorial review. *Attention, Perception, & Psychophysics*, 73(4):971–995.

Stadie, B. C., Levine, S., and Abbeel, P. (2015). Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*.

Stephan, K., Penny, W., Daunizeau, J., Moran, R. J., and Friston, K. J. (2009). Bayesian model selection for group studies. *Neuroimage*, 46:1004–1017.

Steyvers, M., Tenenbaum, J. B., Wagenmakers, E.-J., and Blum, B. (2003). Inferring causal networks from observations and interventions. *Cognitive science*, 27(3):453–489.

Tenenbaum, J. B., Kemp, C., Griffiths, T. L., and Goodman, N. D. (2011). How to grow a mind: Statistics, structure, and abstraction. *Science*, 331(6022):1279–1285.

Tremoulet, P. D. and Feldman, J. (2000). Perception of animacy from the motion of a single object. *Perception*, 29(8):943–951.

Van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double q-learning. In *AAAI*, volume 16, pages 2094–2100.

van Schijndel, T. J., Visser, I., van Bers, B. M., and Raijmakers, M. E. (2015). Preschoolers perform more informative experiments after observing theory-violating evidence. *Journal of experimental child psychology*, 131:104–119.

Waldmann, M. R., Hagmayer, Y., and Blaisdell, A. P. (2006). Beyond the information given: Causal models in learning and reasoning. *Current directions in psychological science*, 15(6):307–311.

Ward, T. B. (1994). Structured imagination: The role of category structure in exemplar generation. *Cognitive psychology*, 27(1):1–40.

Wason, P. (1968). Reasoning about a rule. *Quarterly Journal of Experimental Psychology*, 20:273–281.

Williams, J. J. and Lombrozo, T. (2010). The role of explanation in discovery and generalization: Evidence from category learning. *Cognitive science*, 34(5):776–806.

Xia, V., Wang, Z., and Kaelbling, L. P. (2018). Learning sparse relational transition models. *arXiv preprint arXiv:1810.11177*.

Xu, F. and Denison, S. (2009). Statistical inference and sensitivity to sampling in 11-month-old infants. *Cognition*, 112(1):97–104.

Xu, F. and Garcia, V. (2008). Intuitive statistics by 8-month-old infants. *Proceedings of the National Academy of Sciences*, 105(13):5012–5015.

Xu, F. and Tenenbaum, J. B. (2007). Word learning as bayesian inference. *Psychological review*, 114(2):245.