Human-Centric Reward Design

Yu Qing Du

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-231
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-231.html

November 8, 2023

Human-Centric Reward Design

By

Yu Qing Du

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Pieter Abbeel, Chair
Professor Anca Dragan
Professor Alison Gopnik
Professor Stuart Russell

Fall 2023

Human-Centric Reward Design

Abstract

Human-Centric Reward Design

by

Yu Qing Du

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Pieter Abbeel, Chair

How can we elicit the behaviors we want from artificial agents? One way of guiding behaviors of intelligent systems is through reward design. By specifying reward functions to optimize, we can use reinforcement learning (RL) to enable agents to learn from their own experience and interactions. Thus, RL has seen great success in settings where it is feasible to hand-specify reward functions that are well-aligned with the intended behaviors (*e.g.*, using scores as rewards for games). However, as we progress to developing intelligent systems that have to learn more complex behaviors in the rich, diverse real world, reward design becomes increasingly difficult—and crucial. To address this challenge, *we posit that improving reward signals will require new ways of incorporating human input.*

This thesis comprises two main parts: reward design directly using human input or indirectly using general knowledge we have about people. In the first part, we propose a framework for building robust reward models from direct human feedback. We present a reward modeling formulation that is amenable to large-scale pretrained vision-language models, leading to more generalizable multimodal reward functions under visual and language distribution shifts. In the second part, we use broad knowledge about humans as novel forms of input for reward design. In the human assistance setting, we propose using human empowerment as a task-agnostic reward input. This enables us to train assistive agents that circumvent limitations of existing goal inference based methods, while also aiming to preserve human autonomy. Finally, we study the case of eliciting exploratory behaviors in artificial agents. Unlike prior work that indiscriminately optimizes for diversity in order to encourage exploration, we propose leveraging human priors and general world knowledge to design intrinsic reward functions that lead to more human-like exploration. To better understand how intrinsic objectives guiding human behavior can inform agent design, we also compare how well human and agent behaviors in an open-ended exploration setting align with commonly-proposed information theoretic objectives used as intrinsic rewards. We conclude with some reflections on reward design challenges and directions for future work.

To my parents.

Contents

List of Figures

List of Tables

Acknowledgments

I have been incredibly fortunate to have worked with an outstanding set of individuals during my PhD journey. First, I would like to thank my advisor Pieter Abbeel for supporting and guiding me throughout the past 4 years. It was with his initial encouragement that I even dove into the field of AI to begin with—I will forever be grateful for his guidance. I would also like to thank the rest of my committee, Anca Dragan, Stuart Russell, and Alison Gopnik, for their feedback and input in shaping this dissertation.

I have learned so much from the range of mentors and collaborators I have been able to work with over the past few years—without whom, the work in this thesis would not have been possible. Thank you Stas Tiomkin, Emre Kiciman, Daniel Polani, Eugene Vinitsky, Kanaad Parvate, Kathy Jang, Trevor Darrell, Deepak Pathak, Daniel Ho, Alexander Alemi, Eric Jang, Mohi Khansari, Aditya Grover, Zihan Wang, Cédric Colas, Abhishek Gupta, Jacob Andreas, Ying Fan, Kimin Lee, Shixiang Shane Gu, Hao Liu, Moonkyung Ryu, Craig Boutilier, Mohammad Ghavamzadeh, Kangwook Lee, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, Serkan Cabi, Jessy Lin, Danijar Hafner, Dan Klein, Eliza Kosoy, Li Dayan, Alexander Havrilla, Roberta Raileanu, and Sainbayar Sukhbaatar. An especially special thank you to Olivia Watkins for being my partner in crime throughout the PhD, the source of many random and insightful desk-side chats.

I would also like to thank the other members of our lab for shaping much of my research perspective and for providing a welcoming environment to grow in—Lerrel Pinto, Misha Laskin, Aditi Raghunathan, Stephen James, Coline Devin, Ignasi Clavera, Thanard Kurutach, Aravind Srinivas, Roshan Rao, Jason Peng, Ajay Jain, Fangchen Liu, Wilson Yan, Philipp Wu, Kevin Zakka, Sherry Yang, Alejandro Escontrela, Ademi Adeniji, Amy Lu, Kuba Grudzien, and Xinyue Chen. I feel fortunate to have been able to be a part of this lab at the same time as you all. More broadly, the Berkeley AI Research community has been a wonderful place to learn and grow. Thank you to Angie, Roxana, and Ami, for always keeping things running smoothly. I've been lucky to have made many dear friendships spanning the PhD—thank you Alok, Orr, Kathy, Dhruv, Eric, and Food Things! Cozy days and escapades with the NEoCWiML group (Amy, Judy, Deb, Paula) are unforgettable. I was also fortunate to have made many great friends in London—thank you Judy, Jesse, John, Eric, Sanjana, Sahra, Patrick, Lucio, and Kayo for being a family away from home and for Wednesday Cry(s). Thank you, Brent, for never failing to put a smile on my face.

My initial interest in research started with working on human-robot interaction at the CARIS Lab during my undergradudate degree at UBC. I am deeply grateful for the guidance of Elizabeth Croft, Machiel Van der Loos, and others at CARIS—all of whom shaped my initial research experiences and sparked intrigue in the premise of real-world systems that interact with humans. I am delighted and fortunate to say that this initial spark has remained as a core component of the research I have done since then and going forward.

Finally, thank you to my dear friends and family back home, for always being an indomitable source of support, for making Vancouver always feel like home, and for keeping me well-fed with my favourite foods.

CHAPTER 1 INTRODUCTION

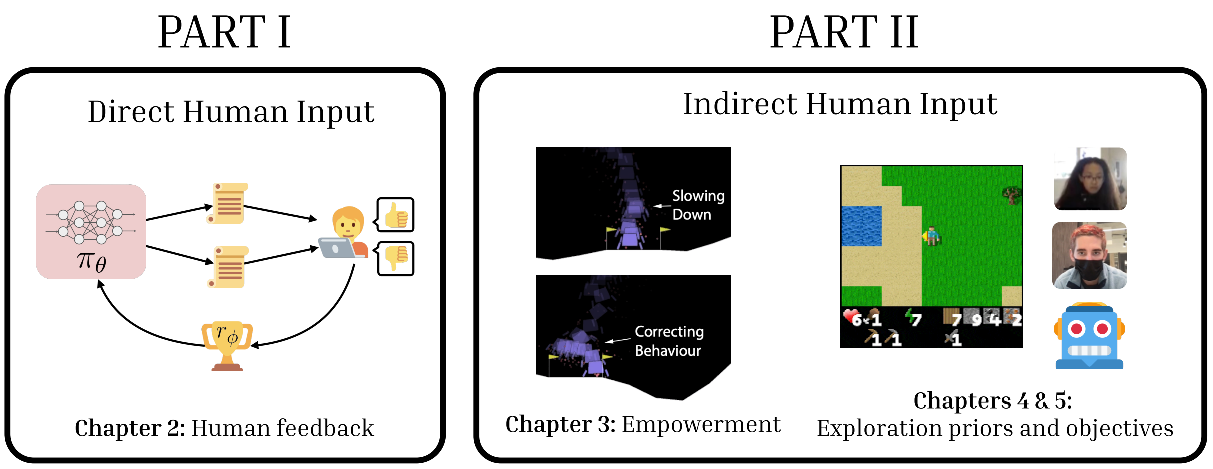


Figure 1.1: Thesis overview. We consider two main forms of human input for guiding reward design: direct and indirect. Part I discusses learning reward models directly from human feedback, where we focus on robustness against naturalistic distribution shifts. Part II discusses forming rewards based on broader knowledge we have about people—that they prefer to be empowered, have priors about the world, and employ various exploration strategies.

Eliciting desired behaviors from machines is a fundamental challenge in artificial intelligence (AI) research. While we have seen increasingly remarkable abilities in deep neural models—ranging from generating text (Brown et al., 2020a; Touvron et al., 2023) and images

(Ho et al., 2020; Rombach et al., 2022) to controlling robotic agents (Agarwal et al., 2023; Brohan et al., 2023)—ensuring that these systems consistently generate behaviors that are aligned with human objectives (*i.e.*, value alignment (Shapiro & Shachter, 2002; Hadfield-Menell et al., 2016) or agent alignment (Leike et al., 2018)) remains an open problem. At the same time, as these systems are deployed in the real world, both implementing and evaluating for alignment become increasingly crucial.

One method for tackling this problem is to train agents through imitation learning. That is, if we humans—ranging from everyday users interacting with intelligent systems to practitioners designing such systems—*know what desired behaviors likely look like*, one natural solution is to create demonstrations of such behaviors and train models to produce the same behaviors via a maximum likelihood objective. This approach of imitation learning can be effective in regimes where it is possible to collect vast amounts of high-quality demonstration data (*e.g.*, leveraging existing internet-scale text and image data in the case of training large language models (Brown et al., 2020a) or vision-language models (Alayrac et al., 2022), or having humans generate robotic task demonstrations (Sammut et al., 1992; Jang et al., 2022)). However, more often than not, the process of acquiring demonstrations that are high quality and have sufficient coverage over the distribution of desirable behaviors is arduous and expensive. Furthermore, it can be difficult to learn solely from demonstrations—the process can lead to brittle policies that fail if the agent diverges too far from given demonstrations (Camacho & Michie, 1995; Ross & Bagnell, 2010; Wang et al., 2017).

Reinforcement learning (RL) (Sutton & Barto, 2018) provides an alternative framework for training intelligent agents. Rather than imitating demonstrations of 'desirable behaviors', RL enables agents to learn from their own experience by optimizing for behaviors that achieve high reward. When a given reward function is well-aligned with desirable behaviors, this process reinforces learning policies that exhibit said desired behaviors. Thus, in this framework we defer the challenge of specifying desired behaviors to *reward design*. While we have seen deep RL techniques be highly successful in regimes where it is easy to hand-specify a reward function that is aligned with desired behaviors (*e.g.*, using scores in games (Samuel, 2000; Mnih et al., 2013)), reward design steadily grows more challenging as we try to train policies for complex behaviors in the real world. This simultaneous increase in reward design importance and difficulty is captured in the Reward Engineering Principle (Dewey, 2014). In fact, even in cases where reward design may seem straightforward, slight mispecifications or loopholes can lead to undesirable behaviors through reward hacking or gaming (Amodei et al., 2016; Leike et al., 2017). As we develop more generalist agents for increasingly complex task settings, how can we impose more human supervision in the reward design process?

In this thesis, *we posit that reward design for increasingly general and capable agents will require novel ways of incorporating human input.* We note that the idea of using human input to guide reward design in and of itself is not novel—prior work has extensively studied a range of methods for guiding agent objectives with human inputs. For example, inverse reinforcement learning extracts reward functions from human demonstrations (Russell, 1998; Ng et al., 2000; Ziebart et al., 2008), active learning or teaching can be used to inform reward estimation (Lopes et al., 2009; Cakmak & Lopes, 2012; Hadfield-Menell et al., 2016;

Sadigh et al., 2017b), or rewards can be modelled from human feedback using deep neural networks (Leike et al., 2018). Here, feedback can vary from preferences over generated behaviors (Christiano et al., 2017; Lee et al., 2021, 2023) to direct reward sketches (Cabi et al., 2020). Jeon et al. (2020) propose reward-rational implicit choice, a unifying formalism for the different forms of information provided by humans for reward learning. Shah et al. (2020) further unifies the paradigms of reward learning and assistance by demonstrating that reward learning problems can be recast as a special case of assistance.

The goal of this thesis is to push the frontier of reward design in two main ways: by using either direct or indirect forms of human input. In the former, we focus on building robustness and increasing human feedback efficiency in the setting of learning reward models from direct human feedback. In the latter, we propose novel ways of using general knowledge we have about humans to design rewards for more complex behaviors—namely, human assistance and exploration.

Concretely, Chapter 2 first introduces the general setting of learning reward models from different forms of aggregated human feedback. Within this domain, we present our work on developing more generalizable reward models for high-dimensional inputs by reformulating the reward modeling problem to be amenable to large-scale, pretrained vision-language models. Here, we demonstrate increased robustness to both language and visual distribution shifts. This enables us to extend multimodal reward models to rich, real world settings, where there can be many diverse ways of accomplishing the same underlying task or goal.

Next, we ask: can we design rewards based on general knowledge we have about humans? Chapter 3 presents our work on reward design for assistive agents. Assistance presents unique challenges where the reward function for the same task can vary depending on each person's unique preferences, and is acutely susceptible to failure modes when the agent incorrectly infers the individual person's goals. How can we design rewards that balance assistance with preserving individual human autonomy and preferences? We propose reframing the objective of assistance to explicitly increase human control by using human empowerment as a reward input, which significantly improves objective human performance under assistance as well as subjective assessments of assistant usefulness. Chapters 4 and 5 explore the more challenging domain of reward design for eliciting exploratory behaviors. While prior works have proposed intrinsic reward functions for exploration in RL agents, these intrinsic rewards often aim to maximize diversity indiscriminately. As such, they are unable to scale well to the complexities of real world exploration, where there can be infinitely many diverse and interesting states. On the other hand, humans are able to explore new settings in a way that is guided by rich prior knowledge as well as some set of intrinsic objectives. Chapter 4 proposes guiding exploration with rich human priors distilled into large language models. This enables us to explore open-ended environments by leveraging priors over meaningful behaviors that are common-sense and context sensitive. Chapter 5 presents our work contrasting human and agent exploratory behaviors in open-ended settings. Finally, in Chapter 6 we reflect on lessons learned and discuss exciting future directions.

CHAPTER 2 ROBUSTIFYING REWARD MODELS

This chapter is based on the paper "Vision-Language Models as Success Detectors" (Du et al., 2023a), written with Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi.

We begin this chapter with a brief introduction to the general setting of learning reward models from human input. In cases where it is challenging to hand-prescribe a reward function, prior methods have used human input in order to extract or model reward functions. The key idea behind this approach is to learn a reward model from human intentions, expressed in different forms of human input (Leike et al., 2018). The resulting reward model can be used for training or evaluating agents without humans having to provide direct in-the-loop feedback for all experiences an agent acquires in an environment.

Various forms of human input have been proposed in this setting, ranging from demonstrations (Ng et al., 2000; Finn et al., 2016; Ho & Ermon, 2016), to corrections over behaviors (Lopes et al., 2009; Sadigh et al., 2017b), to preferences over trajectories (Fürnkranz et al., 2012; Christiano et al., 2017; Ibarz et al., 2018; Lee et al., 2021), or even direct scalar rewards (Cabi et al., 2020). See Section 2.2 for more detailed related works. More recently, reinforcement learning using learned reward models from human feedback (RLHF) has seen prominent progress in the domain of aligning large-scale models (Stiennon et al., 2020; Ouyang et al., 2022); see Casper et al. (2023) for a survey of open challenges in RLHF. Many open questions remain, such as: investigating which forms of human feedback are useful and necessary in different domains, evaluating the intentions captured by reward models, managing reward hacking and gaming, and accommodating distribution shifts. We focus on the last challenge in this chapter.

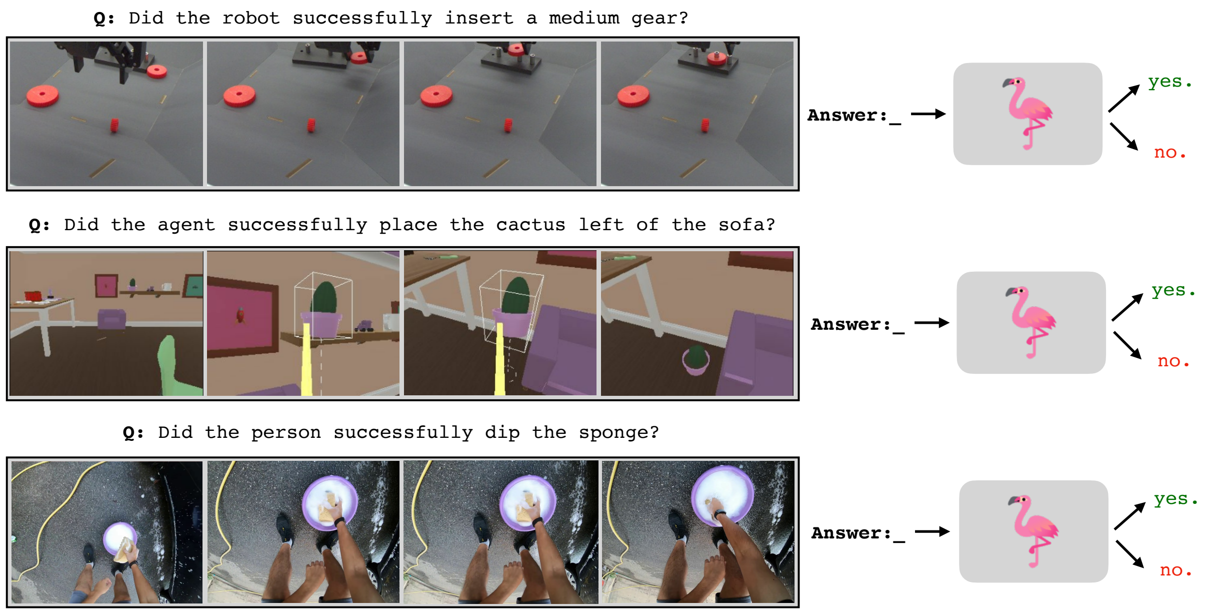

Q: Did the robot successfully insert a medium gear?
Answer:_
yes.
no.
Q: Did the agent successfully place the cactus left of the sofa?
Answer:_
yes.
no.
Q: Did the person successfully dip the sponge?
Answer:_
yes.
no.

Figure 2.1: **SuccessVQA**: Success detection tasks can be formulated as visual question answering (VQA) problems. Large multimodal language models, such as Flamingo, offer the opportunity to learn a generalizable success detector, which can act either as a reward model or agent evaluator across a broad range of domains.

2.1 Introduction

Being able to detect successful (*i.e.,* preferred) behavior is a crucial prerequisite for training intelligent agents. For example, a signal of successful behavior is necessary as a reward for policy learning or as an evaluation metric for identifying performant policies. As such, in this chapter we are concerned with developing accurate and generalizable *success detectors*, which *classify if a behavior is successful or not.* While it is possible to engineer success detectors in specific domains, such as games (Mnih et al., 2013) or control tasks (Tunyasuvunakool et al., 2020), they are often challenging to specify for real-world settings. Success detection in realistic settings can be difficult not only due to challenges with identifying the environment state (*e.g.,* detecting a particular object configuration from pixels), but also due to ambiguities about what a successful state is (*e.g.,* subjective goals, "generate an entertaining story"). One possible approach for developing success detectors is through reward modelling with human preference annotations (Christiano et al., 2017; Ouyang et al., 2022; Cabi et al., 2020; Abbeel & Ng, 2004; Ng et al., 2000). However, the trained reward models are often accurate only for the fixed set of tasks and narrow environment conditions observed in the preference-annotated training data, and thus they require extensive further labour-intensive annotations for increased coverage. This presents a significant bottleneck, as we would like success detectors to be able to generalize broadly – for instance, once a success detector learns what "successfully picking up a block" looks like, it should be able to detect this behavior regardless of background or agent morphology changes thanks to a semantic understanding of what "picking up a block" means.

Consider success detection in robotic manipulation, where tasks are specified with language instructions and observations consist of images. We posit that generalizable success detection is useful for learning generalizable policies in this domain. Here, effective success detectors should generalize to task variations along two axes. Firstly, they should generalize to *language variations* in the task specification. A model that is trained on detecting success for the instruction "lift a rubber duck" should also accurately measure success for "lift a toy duck object". Secondly, success detectors should generalize to *visual variations*. If a camera moves or additional objects are introduced in the scene, the model should still reliably detect success on accomplishing a known task. Standard reward models are typically trained for fixed conditions and tasks, and are thus unable to generalize well to such variations. As such, adapting success detectors to new conditions under distribution shifts typically requires collecting new annotations and re-training.

In this chapter, we aim to train success detectors that are robust with respect to variations in both language specifications and perceptual conditions. To this end, we leverage large pretrained vision-language models (VLMs), such as Flamingo (Alayrac et al., 2022), as a foundation for learning success detectors. We hypothesize that Flamingo's pretraining on vast amounts of diverse language and visual data will enable learning more robust success detectors. In particular, we show that the same simple approach of finetuning Flamingo with human annotations leads to generalizable success detection across vastly different domains. This simple approach allows us to use a unified architecture and training scheme, where we require only 1) videos describing the world state, and 2) text describing the desired behavior or task. We reframe the problem of success detection as a visual question answering (VQA) task and refer to this formulation as *SuccessVQA* (Figure 2.1).

Concretely, we finetune Flamingo for success detection on three diverse domains: a simulated household (Abramson et al., 2021), real-world robotic manipulation, and in-the-wild egocentric human videos (Grauman et al., 2022). The universality of the SuccessVQA task formulation is instrumental in enabling use of the same training architecture in a wide range of tasks and environments. We demonstrate that the resulting success detectors are capable of zero-shot generalisation to unseen conditions (both in language and vision) where bespoke learned reward models fail.

2.2 Related Work

Vision-Language Models (VLMs). Multimodal vision-language models (VLMs) have shown remarkable success in recent years, where VLMs can serve as a foundation for various tasks using language, vision, or arbitrary combinations of modalities. VLMs can be trained with contrastive objectives (Jia et al., 2021; Radford et al., 2021) and/or generative objectives (Dai et al., 2022; Hu et al., 2022; Luo et al., 2020; Alayrac et al., 2022). In this work we rely on the Flamingo model (Alayrac et al., 2022), which leverages a contrastive objective for pretraining the vision encoder on text-and-image pairs. However, unlike other applications of VLMs in single-image VQA tasks (Tiong et al., 2022), we rely on videos to specify the world state, making our work more similar to video QA tasks (Xu et al., 2016). Variants of our approach (*e.g.,* by reducing the video input to a single frame) can also be applied with other image-based VLMs built on large language models (Li et al., 2023; Koh et al., 2023).

Reward Modelling. Reward modelling is often necessary when it is challenging to hard-code a reward function for an agent to learn from. To this end, many works have studied learning reward models from human data. When demonstrations of desirable behavior are available, one can leverage inverse reinforcement learning (IRL), where the key idea is to recover a reward function that best explains expert behavior (Ng et al., 2000; Finn et al., 2016; Ho & Ermon, 2016; Li et al., 2017; Fu et al., 2018; Merel et al., 2017; Zhu et al., 2018; Baram et al., 2017). However, IRL relies on expert demonstrations, makes assumptions about the relationship between the expert actions and the true reward, and can be difficult to train in practice. When demonstrations are difficult to acquire, a more natural way of providing human feedback is through comparative preferences that indicate the degree to which certain agent behavior is desirable. This can be done with comparisons of whole episodes (Akrour et al., 2012; Schoenauer et al., 2014; Brown et al., 2019; Sadigh et al., 2017a), trajectory segments (Christiano et al., 2017; Ibarz et al., 2018; Lee et al., 2021; Abramson et al., 2022a), or even synthesized hypothetical trajectories (Reddy et al., 2020). These methods then fit a reward function as a preference-predictor, *e.g.,* using a Bradley-Terry model (Bradley & Terry, 1952). Nevertheless, preferences are not always the most natural form of feedback from humans, and sometimes it can be easier for a person to provide direct success labels or scalar rewards with respect to a given goal. This can be done online in response to observed agent actions and state transitions (Knox & Stone, 2008; MacGlashan et al., 2017; Arumugam et al., 2019). In robotics, proposed methods vary from sparse, single frame annotations (Singh et al., 2019) to dense, whole trajectory annotations (Cabi et al., 2020). In this work we learn from reward annotations, focusing on training success detectors which can be viewed as binary reward functions. Since collecting human annotations for each new task and environment can be expensive, we aim to study whether pretrained, large VLMs can enable learning more generalizable success detectors from human annotations.

Large-Scale Pretraining for Success Detectors. Our work falls under the general category of using foundation models as reward models. In language modelling, reward models are typically trained by finetuning a pretrained large language model (LLM) with human preferences over LLM generations. This reward model can then be used to finetune an LLM with filtered supervised finetuning or reinforcement learning from human feedback (RLHF) (Stiennon et al., 2020; Nakano et al., 2022; Menick et al., 2022; Glaese et al., 2022;

Askell et al., 2021; Bai et al., 2022). For embodied agents, large-scale datasets of in-the-wild human videos have been used to train reward models (Ma et al., 2022; Chen et al., 2021a). Rather than using human reward annotations of agent behavior s, these methods rely on task-annotated human videos of successful behavior s. Most similar to our work are prior approaches that propose using contrastive VLMs as reward models. In simulated robot domains, Mahmoudieh et al. (2022); Cui et al. (2022) propose using CLIP (Radford et al., 2021) to generate task rewards from a text-based goal description and pixel observations. Fan et al. (2022) leverage large-scale Minecraft data to finetune a Minecraft-specific video CLIP model for detecting alignment (*i.e.,* reward) with text task descriptions. Beyond experimenting in different domains, we also leverage a generative VLM built on a frozen large language model, which we hypothesize enables better language generalisation.

2.3 SuccessVQA: Success Detection as a VQA Task

Our primary contribution is SuccessVQA, a framework that allows us to train multi-task success detectors by directly leveraging powerful pretrained VLMs, such as Flamingo. In SuccessVQA, the VLM is given a visual input representing the state of the world (*e.g.,* a single image or a short video clip) and a question asking if the specified task is successfully accomplished. This problem formulation has several advantages:

- It allows us to unify success detection across domains, using the same architecture and training scheme. We consider three domains: a simulated 3D playroom used in prior research on language-conditioned interactive agents (IA Playroom) (Abramson et al., 2021, 2020), real robotic manipulation, and "in-the-wild" human videos from Ego4D (Grauman et al., 2022).

- Relying on a pretrained vision-language model enables us to harness advantages of pretraining on a large multimodal dataset. We hypothesize this enables better generalization to language and visual distribution shifts.

- The task and state specification allows us to unify treatment of success detection across tasks defined by singular successful states or target behavior s (*i.e.,* detecting success requires reasoning across multiple frames).

SuccessVQA Datasets. To create the SuccessVQA datasets, we use behavior trajectories annotated by humans to indicate *whether* a task is completed successfully, and if so, *when* success occurs. There may be multiple annotations per trajectory from different human raters. In the cases where raters disagree, success or failure is determined by a majority vote, and the median (across the raters who annotated success) of the first annotated success frame is used as the 'point of success'. All subsequent frames are also successful, unless the task is reversed (*e.g.* removing a gear after inserting it for the robotics domain). To generate SuccessVQA examples, a trajectory is split into non-overlapping subsequences (Figure 2.2).

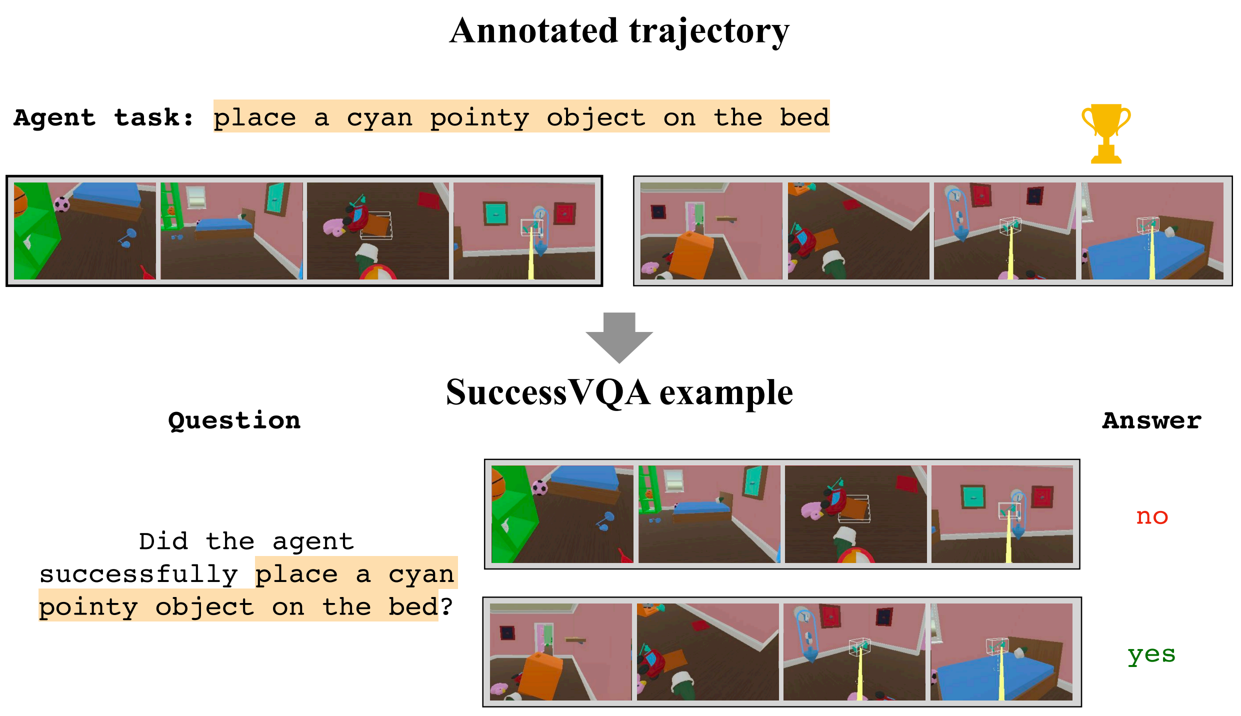

Annotated trajectory
Agent task: place a cyan pointy object on the bed
SuccessVQA example
Question
Answer
Did the agent successfully place a cyan pointy object on the bed?
no
yes

Figure 2.2: **SuccessVQA dataset creation:** A trajectory is annotated by human raters with a point of success (denoted by the trophy). Then the trajectory is split into subsequences and converted to multiple SuccessVQA datapoints with corresponding questions and answers.

For simplicity, we make the clip lengths the same as the pretraining clip lengths used for Flamingo: by first creating subsequences of length 211 frames, then downsampling from 30 FPS to 1 FPS to create 8-frame subsequences. We then generate the VQA question using one of two methods. When trajectories correspond to some known task, we use the template: `Did the robot/agent/person successfully {task}?`, for example, `Did the agent successfully place the cactus left of the sofa?` (see Figure 2.1, first and second rows). When no task is provided but there is a narration corresponding the actions in the clip, as in Ego4D, we use a frozen Flamingo model to rephrase the narrations into questions. For example, given a narration `The person is scooping the ice cream`, we convert it to the question `Did the person successfully scoop the ice cream?` (see Figure 2.1, last row). Finally, the answer is generated: `yes` if the given subsequence ends in one or more success frames, and `no` otherwise.

Training and Evaluation. We finetune the Flamingo (3B) vision-language model on the SuccessVQA dataset for each domain. Specifically, we finetune all the vision layers (vision encoder, perceiver, and cross attention layers) and keep the language layers frozen. In the experiments we refer to this model as FT Flamingo 3B. For evaluation we compute clip-

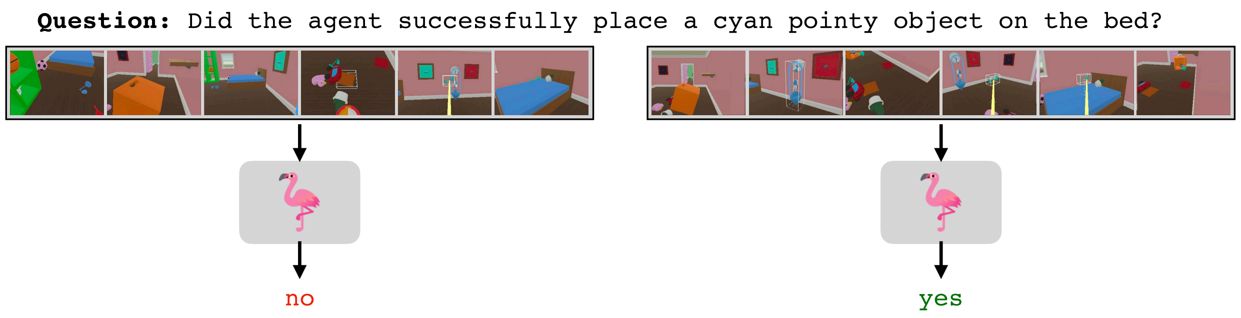

Question: Did the agent successfully place a cyan pointy object on the bed?
no
yes

Figure 2.3: We compute episode-level success detection accuracy during evaluation in order to compare against bespoke success detection models for each domain. To do this, we create subsequences and predict success on each clip individually, then consolidate the predictions at an episode level.

level success detection accuracy against the ground truth human annotations on held-out trajectories. In the simulated household and robotics domains (Sections 2.4 and 2.5) we also compute episode-level accuracy to directly compare against baseline bespoke success detection models, denoted bespoke SD. Note that these baselines were hand-designed independently and tuned specifically for each domain. While these models differ from Flamingo in both pretraining schemes and architecture, they represent a best attempt at designing an accurate reward model for in-distribution evaluations. Episode-level success detection is computed as follows: first, we generate subsequences from the test trajectories in the same way as during training. Next, the success detection model classifies each clip individually for success, as illustrated in Figure 2.3. We consolidate classifications in one of two ways. 1) When the success is completely defined by the observed environment state (as in the robotics tasks), we only look at the first and the last clip of an episode. Then, the entire episode as successful if the first clip is in a failure state and the last clip is in a success state. 2) When the success is defined by a particular behavior (as in the simulated household domain), if any subsequence in an episode is classified as success we classify the episode as successful. We report *balanced* accuracy on the test episodes, as there can be a large imbalance between the number of successful and failure episodes in the dataset. A random model would achieve 50% balanced accuracy.

Experiments Overview. We use the SuccessVQA problem formulation to train success detectors across a diverse range of tasks in vastly different domains: simulated household or IA Playroom (Section 2.4), robotics (Section 2.5), and Ego4D videos (Section 2.6). We investigate whether Flamingo as a success detector model backbone enables generalization across the following axes:

- **language generalization** (Section 2.4). Can we accurately detect success for novel tasks specified with language? To answer this question, we evaluate generalization to unseen tasks specified with language. For example, if we train on detecting success for the task `arrange objects in a row`, can we accurately detect success for the task `arrange objects in a circle`? For these experiments, we use simulated tasks in the IA Playroom where the dataset contains a large and diverse set of language-specified tasks.

- **visual robustness** (Section 2.5). Can we detect success in the presence of unseen visual variations? To answer this question, we evaluate success detection accuracy for a known semantic task, but in the presence of naturalistic visual perturbations. In these experiments, we use real-world robotic manipulation tasks where we introduce visual variations at test-time using different camera viewpoints and distractor objects.

We compare our model against bespoke evaluation models designed and trained specifically for each domain. Note that we do not expect the Flamingo-based models to outperform the bespoke models in a given in-distribution scenario, as we use bespoke baselines designed and tuned for their respective particular domain. Rather, we aim to investigate whether the Flamingo-based models have better robustness to both aforementioned language and visual changes, while also not requiring any domain-specific architectural or training changes. We emphasize that the benefit of SuccessVQA is the simple task formulation that can be applied across a wide range of domains and is directly amenable for use with large pretrained VLMs. Finally, in Section 2.6 we show an example of an in-the-wild SuccessVQA dataset derived from Ego4D (Grauman et al., 2022). Initial results for success detection in this domain are promising, and we hope to encourage further work on accurate reward modelling in unstructured real-world settings.

2.4 Language Robustness with Interactive Agents

In this section we train and evaluate success detectors in the simulated IA Playroom environment, a diverse 3D house environment designed for training language-conditioned interactive agents (Abramson et al., 2021, 2020). The environment consists of a "randomised set of rooms, with children's toys and domestic objects, as well as containers, shelves, furniture, windows, and doors" (see Figure 1 in Abramson et al. (2020)). The tasks are generated from human-human interactions in the IA Playroom, where a *setter* is instructed to provide a task via language for a *solver*, *e.g.,* `bring me the book from the living room`. Success detectors in this environment can serve as automated evaluators for trained policies.

There are two properties in this environment that are particularly challenging for automated success detection: large language variety and the environment's multi-task nature. Large language variations are present because the tasks were originally generated from human interactions, and people are likely to use diverse language to specify even semantically similar tasks. For example, the task of bringing an object to the setter can be phrased in many ways: `bring a fruit from the pantry`, `bring me the banana which is in the pantry`, `bring the yellow coloured object near me`. Moreover, success detection in this environment is intrinsically *multi-task* in its nature because: (1) there is a vast set of possible tasks that can be specified with different utterances, and (2) the behavior of different people and trained agents can vary greatly for the same task. For automated evaluation, it is not scalable to train a new model for each language and task variation.

Training Dataset. We use tasks and trajectories from the Standardized Test Suite (STS), designed specifically for evaluating learned Interactive Agents (Abramson et al., 2021, 2020). We focus on the movement-based tasks: tasks that require the solver agent to move around and interact with the environment. The STS consists of a set of "scenarios that typify the behavior [the Interactive Agents team] wishes to evaluate" (Abramson et al., 2022b), and various trained agent policies are tasked with accomplishing the given scenarios. These test episodes are then annotated by human raters to indicate if a task is successfully completed and if so, at which frame success occurred. We use these annotations to create a SuccessVQA dataset for FT Flamingo 3B finetuning and to train a baseline bespoke SD model for comparison. The training set consists of STS and human interaction data collected between September 2021 to April 2022, 546,887 trajectories in total (1,421,111 clips).

Baseline Success Detectors. For the bespoke SD baseline, we use a success detection model specifically and independently designed for the STS with the best practices we are aware of. We consider two types of baseline models: whole episode evaluation and autoregressive evaluation. As the whole episode model consistently outperformed the autoregressive model, in this section we only report the results from that baseline (see Appendix A.1 for additional results). This model creates a downsampled set of 32 frames from the entire evaluation episode and embeds the images with a ResNet-101. The agent input and output text are also embedded using a learned text embedding. All embeddings are then concatenated together and fed to a transformer with an MLP head that predicts the likelihood the episode was successful. An auxiliary instruction-matching contrastive loss is also applied.

Evaluation. To select the best success detection model, we use the model and checkpoint with the highest balanced accuracy on a held-out validation split from the same distribution as the training data. We then evaluate the chosen success detector model across three different test sets:

- **Test 1: unseen episodes (in distribution)** – a randomly held-out 10% of training dataset trajectories, which includes rephrasings of training tasks. This dataset contains 175,952 clips.
- **Test 2: unseen behavior (out of distribution agents)** – trajectories generated by *new agents* on tasks seen in the training dataset, including rephrasings of training tasks. These agents potentially demonstrate novel behavior , allowing us to assess success detector robustness to unseen behavior s on known tasks. This determines if we can reuse the same models as agent behavior evolves over time (*i.e.* the success

Property	Test 3 Examples
Unseen descriptor	`arrange 4 pointy objects in a square shape in the bed room`, where `square` is not mentioned in the training set. Instead, at train time we have tasks arranging objects in an `arc` or `triangle`.
Unseen objects	`push the train engine with water bird`, where neither `train engine` nor `bird` are mentioned in the training set.
Unseen actions	`hit the candle using the pillow which is left of airplane in the living room`, where the action `hit` is not mentioned in the training set.

Table 2.1: Examples of unseen task variants used in Test 3 (rightmost column in Table 2.2).

Model	Test 1: unseen episodes	Test 2: unseen behaviors	Test 3: unseen tasks
bespoke SD	80.6%	**85.4%**	49.9%
FT Flamingo 3B	**83.4%**	85.0%	**59.3%**

Table 2.2: Zero-shot episode-level balanced accuracies for IA Playroom STS evaluation models in new conditions. For reference, human level balanced accuracy is around 88% due to inter-rater disagreement.

detector is accurate even when the agent solves a known task in a novel way). This dataset contains 462,061 clips.

- **Test 3: unseen tasks (out of distribution agents and tasks)** – the most challenging setting: trajectories generated by new agents on *new tasks* not seen during training. For examples of such new tasks, see Table 2.1. Note that this set comprises *completely new tasks* as well as rephrasings of said tasks. As the tasks are new, the success detector models need to master a semantic understanding of language to properly generalise to success detection in this set. This dataset contains 272,031 clips.

Results. Table 2.2 presents the episode-level balanced accuracy on each test set. We find that without finetuning, the accuracy of the Flamingo model is close to random chance (see Appendix A.1 for details). This is unsurprising, as the IA domain differs greatly from Flamingo's pretraining data. When finetuning on the same data, FT Flamingo 3B matches the performance of the bespoke models in Test 1 (unseen episodes) and Test 2 (unseen behavior). More importantly, in Test 3 (unseen tasks), the performance of a bespoke model drops to a random chance, while FT Flamingo 3B outperforms it by a significant margin (10%). As the instructions in Test 3 are for novel tasks, not just rephrasings of tasks seen during training, this experiment demonstrates that the success detector exhibits some amount of semantic understanding of the scenes. We hypothesize that this is possible due to Flamingo's large language model backbone and web-scale pretraining. That said, there is still a large margin for improvement on the most challenging test set. For future work, it would be interesting to investigate how different model scales, dataset sizes, or cross-finetuning with different datasets can affect generalisation.

2.5 Visual Robustness with Robotic Manipulation

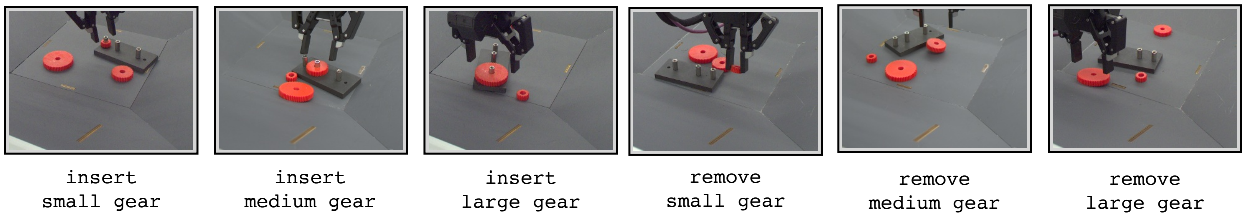


Figure 2.4: Successful frames for the 6 robotics gear manipulation tasks.

In this section we train and evaluate success detectors on a family of real-life robotic gear manipulation tasks with a Panda robot arm. There are six tasks corresponding to inserting or removing a small, medium, or large gear within a basket (Figure 2.4). We consider visual observations from a basket camera. Ideally, a success detector should remain accurate under naturalistic visual changes, such as different camera view angles, lighting conditions, or background changes. Furthermore, as the performance of learned policies improves, we may want to introduce new objects or tasks to the environment. It quickly becomes impractical to re-annotate and re-train success detectors from previous tasks in new conditions, thus making it important to train visually robust success detectors. For example, a model that has learned to detect successful gear insertion should still be able to robustly detect success even if the basket has additional task-irrelevant distractor objects or camera angle changes. To investigate this, we design our experiments to carry out zero-shot evaluations on test episodes with such visual changes.

Training Dataset. Human operators provide $101,789$ demonstrations for 6 tasks using a 6DoF control device. Each episode is then annotated by humans with rewards for each task (*e.g.,* every episode has 6 reward annotations, one for each task). Human annotators label positive rewards for all frames with a success state (*i.e.,* if the task is solved), and zero rewards otherwise. Note that it is possible for a task to be accidentally undone in the same episode, at which point the reward annotation would revert to zero. The reward annotations

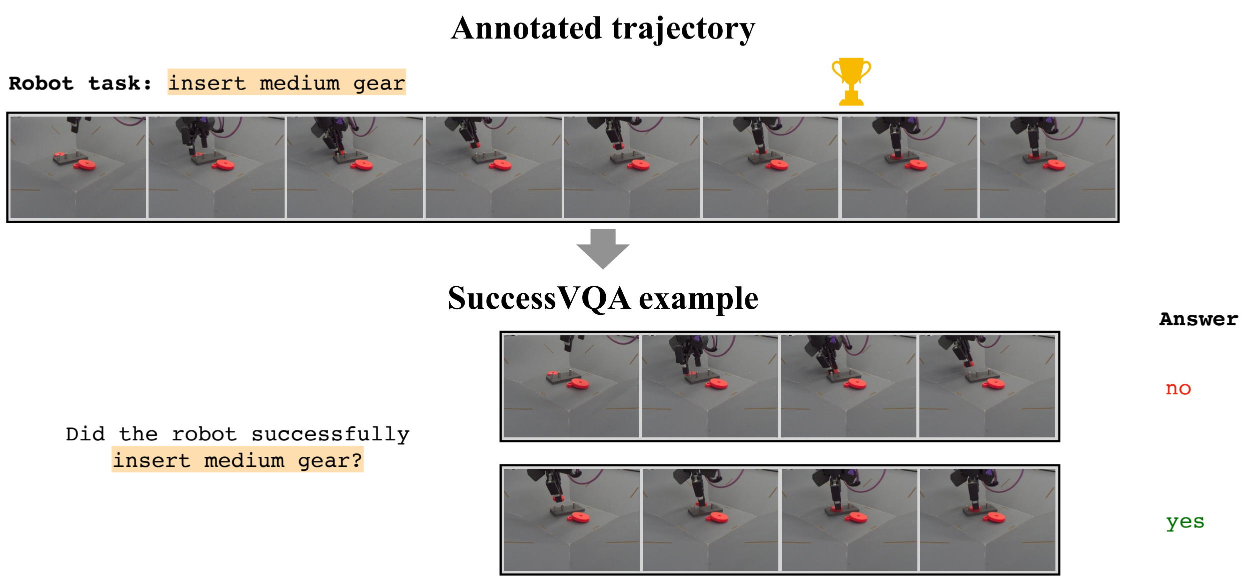

Annotated trajectory
Robot task: insert medium gear
SuccessVQA example
Answer
Did the robot successfully insert medium gear?
no
yes

Figure 2.5: Sample SuccessVQA example created from an annotated subsequence of a gear manipulation episode. Success annotation is shown with the trophy.

and corresponding episode frames are then converted into SuccessVQA examples (see Figure 2.5). The ground truth VQA answer is obtained from the human annotations: clip answers are labelled successful if they contain only a single transition from zero to positive reward or only have positive rewards throughout, otherwise they are labelled as unsuccessful. We train a single FT Flamingo 3B success detector model for all 6 tasks.

Baseline Success Detector. As a baseline, we consider a ResNet-based (He et al., 2016) per-frame success classification model, trained and tuned specifically for this domain independently of this work. The ResNet-18 is pretrained on ImageNet, and the classification layer is swapped out for a binary classification layer. We finetune a *separate* success classification model for each of the 6 gear tasks, applying image augmentations during finetuning. This is distinct from our method where we train a single multi-task model across all 6 tasks. We consider an episode successful if the first and last frames[1] of the episode are classified as a failure (output < 0.5) and success (output > 0.5) correspondingly. We will further refer to the baseline model simply as bespoke success detector (bespoke SD).

Evaluation. To compare against the bespoke SD, we look at episode-level balanced accuracy. Given an evaluation episode, we consider the episode successful under FT Flamingo 3B if the first clip is classified as unsuccessful and the last clip is classified as successful (see Figure A.3 in the Appendix). This matches the episode-level classification scheme of bespoke

[1]We find that incorporating more frames does not improve episode-level accuracy.

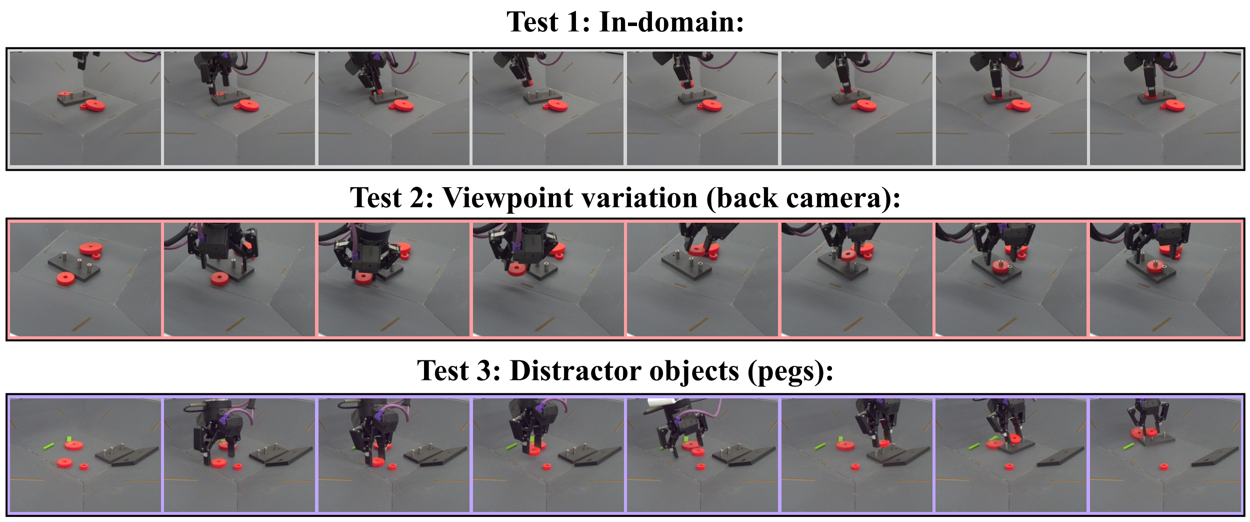

Test 1: In-domain:
Test 2: Viewpoint variation (back camera):
Test 3: Distractor objects (pegs):

Figure 2.6: Examples of three evaluation datasets: in-domain episodes similar to the training dataset, episodes with a different camera viewing angle and episodes with distractor objects in the basket.

SD. We conduct the evaluation on three test sets (see Figure 2.6): Test 1: In-domain episodes (first row), Test 2: Episodes with a **viewpoint variation**, using a different (back) camera (second row), and Test 3: Episodes with **distractor objects** in the basket, but the original camera (last row). The last two settings are designed to test the robustness of the models to naturalistic visual perturbations in the environment. The trained success detectors can then either be used as automated evaluators or reward models for agent training.

In-Domain Performance. In Test 1, we conduct an in-domain evaluation where the test set comes from the same visual conditions as the training set (see Figure 2.6, top row). The test set includes all the training episodes and an additional held out 2076 episodes. The results in Table 2.3 show that while the bespoke SD consistently outperforms the FT Flamingo 3B, the performance of the FT Flamingo 3B model is still comparable for the insertion task. Note that the accuracy of the Flamingo model on the remove tasks is lower, which we hypothesize is likely due to a data balancing issue. We have 5 times more training data available for insertion than removal, and training a single model across all tasks likely led to a tradeoff in accuracy between the insertion and removal tasks, which are temporal opposites of each other. We also conduct experiments into the efficacy of policies trained with these success detectors in Appendix A.2, with initial proof-of-concept evaluations suggesting that policy performance under either FT Flamingo 3B or bespoke SD are similar.

Visual Robustness. Next, we measure zero-shot accuracy on two natural visual variations described above: Test 2 and Test 3. In Test 2, we look at zero-shot robustness to

	Insert			Remove		
	Small	Medium	Large	Small	Medium	Large
bespoke SD	**98.0%**	**98.4%**	**99.1%**	**97.3%**	**98.7%**	**98.4%**
FT Flamingo 3B	96.0%	94.4%	95.0%	82.1%	83.4%	87.2%

Table 2.3: **In-Domain Episode-level Accuracy for Gear Manipulation.** Balanced accuracy evaluated on 50000-60000 episodes per task.

different viewpoints (Figure 2.6, middle row). Given that the success detectors were only trained on frames from the *front* basket camera, we evaluate robustness by measuring success detector accuracy on episodes recorded with the *back* basket camera. As we can see in Table 2.4, changing the camera angle drastically hurts the quality of bespoke SD (accuracy decreases of 10-50 absolute percentage points) while the performance of FT Flamingo 3B is more stable (accuracy decreases by less than 10%). Note that in some tasks the performance of the bespoke model drops to the level of random guess, essentially rendering the model useless for success detection. With this, **FT Flamingo 3B** becomes the best performing model in 5 out of 6 tasks.

	Insert			Remove		
	Small	Medium	Large	Small	Medium	Large
bespoke SD	78.0%	53.1%	50.9%	**85.8%**	53.8%	72.8%
	-19.9%	-45.4%	-48.3%	-11.5%	-44.9%	-25.5%
FT Flamingo 3B	**91.0%**	**89.8%**	**89.7%**	76.7%	**75.9%**	**79.4%**
	-4.0%	-4.6%	-5.3%	-5.5%	-7.5%	-7.8%

Table 2.4: **Viewpoint variation.** Zero-shot success detection balanced accuracy when training on front camera views and evaluating on back camera views. We show the absolute balanced accuracy and the percentage point change compared to Test 1 from Table 2.3.

In Test 3 we look at zero-shot robustness in the setting where some *distractor objects* (two pegs and a board, see Figure 2.6, last row) are introduced. Table 2.5 shows that detecting success on known tasks across this novel visual setting causes a 4-30% (absolute percentage points) drop in balanced accuracy for the bespoke model, while the accuracy mostly stays stable for the Flamingo-based models, with a 4.5% drop in accuracy at most.

These two experiments demonstrate that Flamingo-based success detection models are robust to natural visual variations. We hypothesize that the pretrained Flamingo-based success detection model is better suited to zero-shot visual generalisation than the bespoke baseline reward model, as Flamingo is pretrained on a diverse set of visual data with corresponding language grounding. While the baseline model was also pretrained and used image

	Insert			Remove		
	Small	Medium	Large	Small	Medium	Large
bespoke SD	88.8% **-9.2%**	85.0% **-13.4%**	71.8% **-27.4%**	**93.6%** **-3.8%**	**93.8%** **-4.9%**	**92.4%** **-6.0%**
FT Flamingo 3B	**96.1%** +0.1%	**95.6%** +1.2%	**90.6%** -4.5%	82.4% +0.3%	83.6% +0.1%	84.7% -2.5%

Table 2.5: **Distractor Objects.** Zero-shot success detection balanced accuracy on scenes with distractor objects. We show the absolute balanced accuracy and the percentage point change compared to Test 1 from Table 2.3.

augmentations during task finetuning, it was not exposed to such a diverse set of visual data or language. Large-scale diverse pretraining might contribute to better semantic tasks recognition under naturalistic visual changes. These encouraging results suggest that pre-trained VLM-based success detectors are likely better suited to the real-world tasks involving unstructured, open, and evolving settings.

2.6 Real World Success Detection with Ego4D

In this section we describe creating a SuccessVQA dataset using “in-the-wild" egocentric videos of humans performing tasks. This present a much more diverse setting than the prior two domains, in both visuals and language. We construct this dataset using annotations from the Ego4D dataset (Grauman et al., 2022), where unlike prior benchmarks in action recognition, the focus is on detecting a temporal point of success for a given action. It is an example of a realistic, unstructured setting where the ground-truth success labels can be obtained only from human annotations. While the FT Flamingo 3B success detector model shows initial promising results, our experiments show that the benchmark is nonetheless very challenging with much room for future progress.

Ego4D is a publicly available dataset of egocentric human-in-the-wild videos. The videos show people executing common tasks (*e.g.,* washing dishes, cleaning cars, gardening). To generate ‘successful’ and ‘unsuccessful’ action sequences, we make use of annotations from the Ego4D Forecasting + Hands & Objects (FHO) dataset, where corresponding narrations describe the actions of the camera wearer in the videos. Additionally, critical state changes are annotated: “how the camera wearer changes the state of an object by using or manipulating it–which we call an object state change” (Grauman et al., 2022). Each narration is centered on an 8-second clip, which is further annotated with the critical frames PRE, Point of No Return (PNR), and POST for indicating when the narrated state change has occurred. The PNR frame annotates the start of the state change, the PRE frame indicates a point before the state change, and the POST frame is a point after the state change is

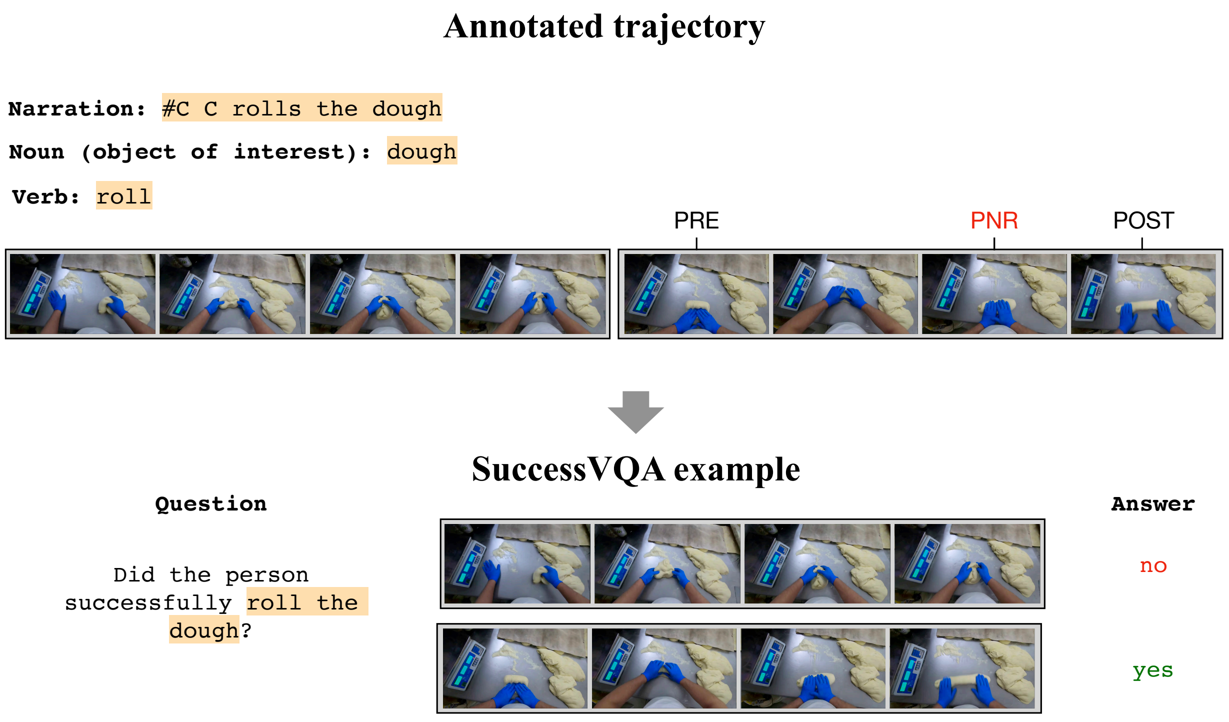

Annotated trajectory
Narration: #C C rolls the dough
Noun (object of interest): dough
Verb: roll
PRE
PNR
POST
SuccessVQA example
Question
Did the person successfully roll the dough?
Answer
no
yes

Figure 2.7: Sample Ego4D clip converted to SuccessVQA Examples. Ego4D annotations include PRE, POST and PNR (point of no return) annotations which are then used to generate answers in the SuccessVQA examples.

completed. We propose using the critical frame annotations as annotations of ‘success’ for the behavior described in the narration. Specifically, we treat PNR frame as a point at which ‘success’ occurs. To generate a negative example for a clip, we use the frames in the 8-second clip prior to the PRE frame. These frames do not contain the point of success, but they often demonstrate the beginning of the relevant action. We then generate the questions for SuccessVQA by rephrasing the narrations into questions using Flamingo (Figure 2.7).

Unlike the prior domains where there is only one relevant task per episode, a single Ego4D video can have multiple narrations corresponding to different actions. Thus, instead of episode-level accuracy we evaluate success detection accuracy on clips taken from completely held out videos. In our experiments, FT Flamingo 3B finetuned on the SuccessVQA dataset attains 99% training accuracy and 62% test set accuracy. For reference zero shot and 4-shot Flamingo only achieves 50% and 52% accuracy. That is, without finetuning, the Flamingo model is not capable of detecting success. Providing a few examples with few-shot prompting improves performance, but very slightly. Finetuning on the in-domain Ego4D SuccessVQA examples leads to a significant improvement. That said, there is still a large gap between train and test performance. We find that it is currently difficult to generalise fully accurately to unseen real world videos, so this domain provides an exciting avenue for future work.

2.7 Conclusion

In this chapter we propose SuccessVQA—a reformulation of reward modeling or success detection that is amenable to pretrained VLMs such as Flamingo. We investigate success detection across a wide range of domains: simulated language-conditioned interactive agents, real-world robotic manipulation, and "in-the-wild" human videos. We find that the pretrained VLM has comparable performance on most in-distribution tasks, and increased robustness across language and visual changes compared to task-specific reward models, and emphasize that our contribution is a more universal success detection task formulation that can be applied easily across vastly different domains.

Limitations and Future Work. There still exist some gaps between the Flamingo-based reward models and the bespoke reward models in our experiments. Furthermore, inference with a larger VLM is expensive, making online success detection challenging. Lastly, we find that finetuning on a sufficient amount of in-domain data is necessary for robust success detection, as zero-shot or few-shot performance in our chosen domains is not yet sufficient. Nonetheless, we are optimistic that further progress on broadly improving VLMs will result in more accurate few-shot success detection. VLMs are often used as policies, see *e.g.,* Reed et al. (2022), but in this work we have demonstrated that there is also great value in using them as reward models—in other words, VLMs as rewards focuses on the 'what to do' and not on 'how to do it'. We therefore expect such models to transfer more easily than policies when the same task can be accomplished in many ways, and where fine visual details are not necessary (*e.g.,* grasp angle for fine motor control).

To address the limitations of the current approach, improving inference speed or distillation to a smaller model can help with efficient online success detection. Before deployment as a reward model for learning policies,further investigations into model accuracy to avoid reward hacking and more thorough understanding of the impacts of reward model false positives and false negatives are necessary. So far we have experimented with a Flamingo 3B, but larger models might bring further improvements in robustness and generalisation. The shared SuccessVQA format can also enable shared finetuning across different datasets (*e.g.,* combining Ego4D SuccessVQA and VQAv2 (Goyal et al., 2017)) to study the impact of cross-task transfer. Lastly, the flexibility in multimodal models allows us to consider success detection tasks where the task is specified visually (*e.g.,* with a goal image) or the state is described in language (*e.g.,* a dialogue agent) in the same framework as the current work.

CHAPTER 3 HUMAN EMPOWERMENT FOR ASSISTANCE

This chapter is based on the paper "AvE: Assistance via Empowerment" (Du et al., 2020), written with Stas Tiomkin, Emre Kiciman, Daniel Polani, Pieter Abbeel, and Anca Dragan.

3.1 Introduction

We aim to enable artificial agents, whether physical or virtual, to assist humans in a broad array of tasks. However, designing rewards to train assistive agents is challenging when the human's goal is unknown, because that makes it unclear what the agent should learn to do. Assistance games (Hadfield-Menell et al., 2016) formally capture this as the problem of working together with a human to maximize a common reward function whose parameters are only known to the human and not to the agent. Naturally, approaches to assistance in both shared workspace (Pellegrinelli et al., 2016; Fisac et al., 2018, 2020; Pérez-D'Arpino & Shah, 2015; Nikolaidis & Shah, 2013; Macindoe et al., 2012) and shared autonomy (Javdani et al., 2015, 2018; Dragan & Srinivasa, 2013; Gopinath et al., 2016; Pellegrinelli et al., 2016) settings have focused on inferring the human's goal (or, more broadly, the hidden reward parameters) from their ongoing actions, building on tools from Bayesian inference (Baker et al., 2006) and inverse reinforcement learning (Ng et al., 2000; Russell, 1998; Abbeel & Ng, 2004; Argall et al., 2009; Ziebart et al., 2008; Ho & Ermon, 2016; Ramachandran & Amir, 2007; Duan et al., 2017; Mainprice & Berenson, 2013). However, goal inference can fail when the human model is misspecified, e.g. because people are not acting noisy-rationally (Majumdar et al., 2017; Reddy et al., 2018b), or because the set of candidate goals the agent

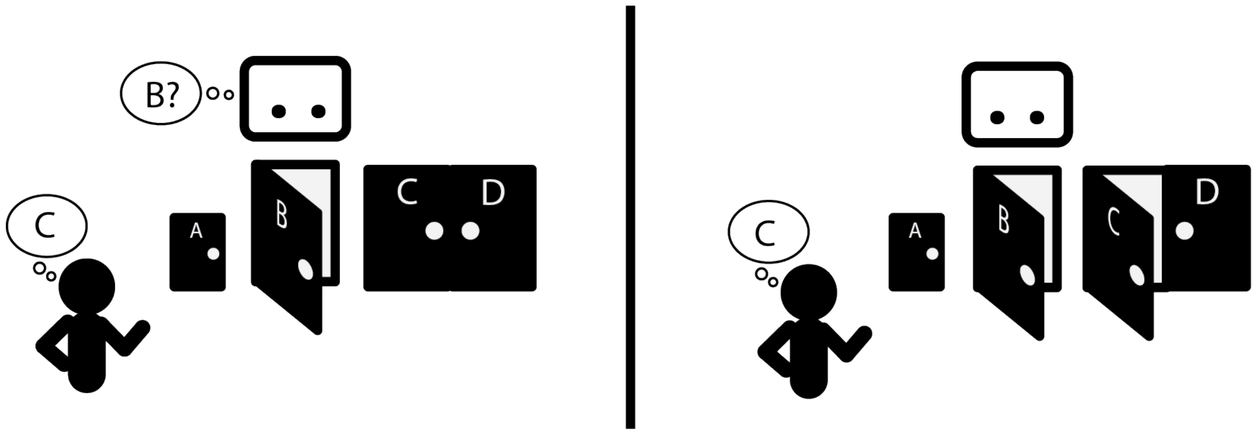


Figure 3.1: Toy scenario where a robot operates multiple doors. On the left, the robot attempts to infer the human's intended goal, C, but mistakenly infers B. On the right, the robot assesses that the human's empowerment would increase with doors B and C open. Naively opening all doors will not increase empowerment as A is too small for the person and D leads to the same location as C.

is considering is incorrect (Bobu et al., 2018). In such cases, the agent can infer an incorrect goal, causing its assistance (along with the human's success) to suffer, as in Figure 3.1.

Even in scenarios where the agent correctly infers the human's goal, we encounter further questions about the nature of collaborations between humans and assistive agents: what roles do each of them play in achieving the shared goal? One can imagine a scenario where a human is attempting to traverse down a hallway blocked by heavy objects. Here, there are a range of assistive behaviors: a robot could move the objects and create a path so the human is still the main actor, or a robot could physically carry the person down the hallway, making the human passive. Depending on the context, either solution may be more or less appropriate. How do we design rewards that respect and capture individual human preferences throughout assistance? Specifically, the boundary between *assisting* humans in tackling challenging tasks and *solving* these tasks in the place of humans is not clearly defined. As AI improves at 'human' jobs, it is crucial to consider how the technology can complement and amplify human abilities, rather than replace them (Wilson & Daugherty, 2018a,b).

Our key insight is that agents can assist humans without inferring their goals or limiting their autonomy by instead increasing the human's *controllability* of their environment – in other words, their *ability to affect the environment through actions*. We capture this via *empowerment*, an information-theoretic quantity that is a measure of the controllability of a state through calculating the logarithm of the number of possible distinguishable future states that are reachable from the initial state (Salge et al., 2014b). In our method, Assistance via Empowerment (AvE), we formalize the learning of assistive agents as an augmentation of reinforcement learning with a reward input based on human empowerment.

The intuition behind our method is that by prioritizing agent actions that increase the human's empowerment, we are enabling the human to more easily reach whichever goal they want. Thus, we are assisting the human without information about their goal – the agent does not carry the human to the goal, but instead clears a path so they can get there on their own. Without any information or prior assumptions about the human's goals or intentions, our agents can still learn to assist humans.

We test our insight across different environments by investigating whether having the agent's behavior take into account human empowerment during learning will lead to agents that are able to assist humans in reaching their goal, despite having no information about what the goal truly is. Our proposed method is the first one, to our knowledge, that successfully uses the concept behind empowerment with real human users in a human-agent assistance scenario. Our experiments suggest that while goal inference is preferable when the goal set is correctly specified and small, empowerment can significantly increase the human's success rate when the goal set is large or misspecified. This does come at some penalty in terms of how quickly the human reaches their goal when successful, pointing to an interesting future work direction in hybrid methods that try to get the best of both worlds. As existing methods for computing empowerment are computationally intensive, we also propose an efficient empowerment-inspired proxy metric that avoids the challenges of computing empowerment while preserving the intuition behind its usefulness in assistance. We demonstrate the success of this algorithm and our method in a user study on shared autonomy for controlling a simulated dynamical system. We find that the strategy of stabilizing the system naturally emerges out of our method, which in turn leads to higher user success rate. While our method cannot outperform an oracle that has knowledge of the human's goal, we find that increasing human empowerment provides a novel step towards generalized assistance, including in situations where the human's goals cannot be easily inferred.

Concretely, in this chapter we formalize learning for human-agent assistance via using human empowerment as a component of the agent reward function. To show the advantage of this formulation, we directly compare our method against a goal inference approach and confirm where our method is able to overcome potential pitfalls of inaccurate goal inference in assistance. In order to enact this method in a practical user study in a continuous domain, we propose a computationally efficient proxy for empowerment, thus enabling human-in-the-loop experiments of learned assistance in a challenging simulated teleoperation task.

3.2 Empowerment Preliminary

To estimate the effectiveness of actions on a given environment, Klyubin et al. (2005a) propose computing the *channel capacity* between an action sequence, $\vec{a}_T \doteq (a_1, a_2, \ldots, a_T)$, and the final state, s_T, where the channel is represented by the environment. Formally, empowerment of a state s is:

$$\mathcal{E}(s) = \underset{p(\vec{a}_T \mid s)}{\operatorname{maximum}} \, \mathcal{I}[\vec{a}_T; s_T \mid s], \tag{3.1}$$

where $\mathcal{I}[\vec{a}_T; s_T \mid s] \doteq \mathcal{H}(s_T \mid s) - \mathcal{H}(s_T \mid \vec{a}_T, s)$ is the mutual information between $\vec{a}_T$ and the s_T, $\mathcal{H}(\cdot)$ is entropy, and T is the time horizon, which is a hyperparameter. The *probing probability distribution* $p(\vec{a}_T|s)$ is only used for the computation of the empowerment/channel capacity $\mathcal{E}(s)$, and never generates the behavior directly. Note that the actions are applied in an open-loop fashion, without feedback.

In the context of learning, empowerment as an information-theoretic quantity has mainly been seen as a method for producing intrinsic motivation (Salge et al., 2014b; Mohamed & Rezende, 2015; Klyubin et al., 2005b; de Abril & Kanai, 2018). Guckelsberger et al. (2016) showed that a composition of human empowerment, agent empowerment, and agent-to-human empowerment are useful for games where a main player is supported by an artificial agent. An alternative view of this compositional approach was proposed in Salge & Polani (2017), where the agent-to-human empowerment was replaced by human-to-agent empowerment. The latter is a human-centric approach, where only the humans' actions affect the agents' states in a way which is beneficial for the human. These approaches have only conceptually discussed the applicability of empowerment to human-agent collaborative settings. Our work concretely proposes that assistance in shared workspaces and shared autonomy tasks can be cast as an empowerment problem, and evaluates this approach with real users.

Given the challenge of computing empowerment, some existing approximations are:

- **Tabular case approximations.** In tabular cases with a given channel probability, $p(s_T \mid \vec{a}_T, s)$, the problem in (3.1) is solved by the Blahut-Arimoto algorithm (Blahut, 1972). This method does not scale to: high dimensional state and/or action space or long time horizons. An approximation can be done by Monte Carlo simulation (Jung et al., 2011), however, the computational complexity precludes user studies of real time empowerment-based methods for assistance in an arbitrary state or action space.

- **Variational approximations.** Previous work has proposed a method for using variational approximation to provide a lower bound on empowerment (Mohamed & Rezende, 2015). This was extended later to closed-loop empowerment (Gregor et al., 2016). Both of these methods estimate empowerment in a model-free setting. Recent work has also proposed estimating empowerment by the latent space water-filling algorithm (Zhao et al., 2019), or by applying bi-orthogonal decomposition (Tiomkin et al., 2017), which assumes known system dynamics. However, these estimation methods are computationally hard, which precludes their use in reinforcement learning, especially, when a system involves learning with humans, as in our work.

3.3 Assistance via Empowerment

Problem Setting. We formulate the human-assistance problem as a reinforcement learning problem, where we model assistance as a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}_a, \mathcal{T}, \gamma, \mathcal{R})$. $\mathcal{S}$ consists of the human and agent states, (S_h, S_a), $\mathcal{A}_a$ is the set of agent actions, $\mathcal{T}$ is an unknown transition function, γ is a discount factor, and $\mathcal{R}$ is a reward function, . We assume the human has an internal policy π_h to try to achieve a goal $g^* \in S_h$ that is unknown to the agent. Note that the human may not behave completely rationally and π_h may not enable them to achieve g^*, requiring assistance. As the agent does not know the human policy, we capture state changes from human actions in the environment transition function $(s_h^{t+1}, s_a^{t+1}) = \mathcal{T}((s_h^t, s_a^t), a_a)$.

Our method uses reinforcement learning to learn an assistive policy π_a that maximizes the expected sum of discounted rewards $\mathbb{E}[\sum_t \gamma^t \mathcal{R}(s_a^t, s_h^t)]$. For assistance, we propose the augmented reward function

$$\mathcal{R}(s_a, s_h) = \mathcal{R}_{\text{original}}(s_a, s_h) + c_{emp} \cdot \mathcal{E}_{\text{human}}(s_h) \tag{3.2}$$

where $\mathcal{R}_{\text{original}}$ captures any general parts of the reward that are non-specific to assisting with the human's goal, such as a robot's need to avoid collisions, and where $\mathcal{E}_{\text{human}}(s_h)$ is the agent's estimation of the human's empowerment in their current state, s_h. In simple tabular environments we can directly compute empowerment, however, this does not extend to more realistic assistance scenarios. Thus for our user study, we propose an empowerment-inspired proxy as detailed in Section 4.3. We access the human's action and state space either through observation in the shared workspace case or directly in shared autonomy. The coefficient $c_{emp} \geq 0$ is included to balance the weighting between the original reward function and the empowerment term.

Approach for Human-in-the-Loop Training. As our work is intended for real-time human assistance applications, we strongly prioritize computational efficiency over the numerical accuracy of the empowerment estimation in our user study. To compute true empowerment, one considers potential forward-simulations of agent actions of duration T starting in s_{init} and their resulting effect. However, since existing methods for approximating empowerment in continuous domains from empirical data are computationally quite hard and do not scale well, here we instead draw on the intuition of empowerment to propose a proxy metric using a measure of diversity of final states as a surrogate for the channel capacity.

Namely, we use a fast and simple analogy of the sparse sampling approximation from (Salge et al., 2014a) for the empowerment method. In that work, the number of discrete states visited by the forward-simulations was counted. A large number of different final states corresponded to an initial state with high empowerment. In the continuum, counting distinct states becomes meaningless. Therefore, here, we measure diversity of the final state by computing the variance of the flattened sample vectors of S_f, as summarized in Algorithm 1, and use this pragmatic approach directly in place of $\mathcal{E}$ in Eq. 3.1. While this proxy relies on an assumption of homogeneous noise and high SNR, it can be computed much more efficiently than empowerment, lending it to be directly applicable to human-in-the-loop training of assistive agents. Even as a coarse proxy for empowerment, we find that it is sufficient to significantly increase human success in our user study in Section 3.4. To empirically motivate our proxy, we compare the known empowerment landscape of a non-

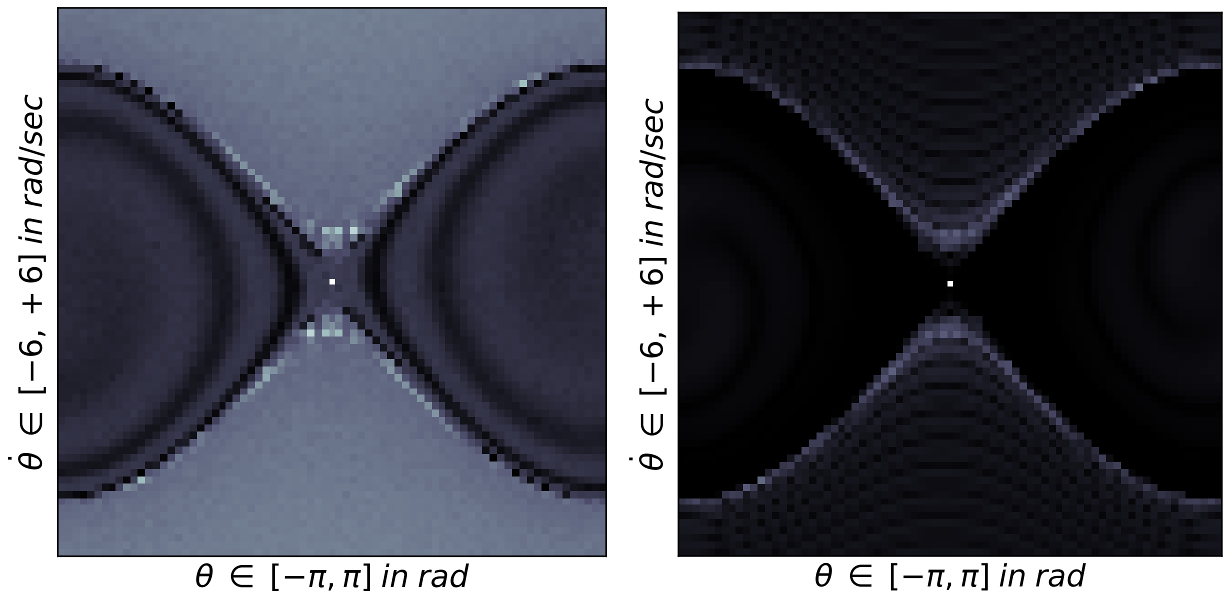


Figure 3.2: Comparison between our proxy for empowerment (left) and the known landscape (right) of the non-linear pendulum over a long time horizon. The proxy landscape captures the essential properties of the empowerment landscape: maximum at the upright position and comparatively low values for states with energy below the separatrix energy.

linear pendulum with our proxy result in Figure 3.2. A rigourous study of the properties of the proxy is deterred to the future work.

Algorithm 1 Empowerment-inspired Diversity Bonus

```
Initialize environment at state $s_{init}$
Initialize empty list of final states $s_f$
Rollout $N$ trajectories of horizon $T$
for n = 1 ... N do $s \leftarrow s_{init}$
    for t = 1 ... T do
        # randomly generate actions and update state
        $a$ = sample action
        $s \leftarrow \mathcal{T}(s, a)$
    end for
    $s_f \leftarrow s_f + [s]$
end for
# To compute scalar reward bonus
return Var(flatten($s_f$))
```

3.4 Experiments

In this section we evaluate our method in two distinct assistive tasks where a human attempts to achieve a goal: firstly, a shared workspace task where the human and assistant are independent agents, and secondly, a shared autonomy control task. The first task is used to directly compare the empowerment method in a tabular case against a goal inference baseline to motivate our method, and the second task is used to test both our proxy metric in Algorithm 1 and evaluate our overall method with real users in a challenging simulated teleoperation task. For accompanying code, see `https://github.com/yuqingd/ave`.

Shared Workspace: Gridworld

Experiment Setup. To motivate our method as an alternative when goal inference may fail, we constructed a gridworld simulation as follows: a simulated proxy human attempts to move greedily towards a goal space. The grid contains blocks that are too heavy for them to move, however, an agent can assist by moving these blocks to any adjacent open space. The agent observes the location of the blocks and the humans but not the location of the goal. As a metric for successful assistance, we measure the success rate of the human reaching the goal and the average number of steps taken.

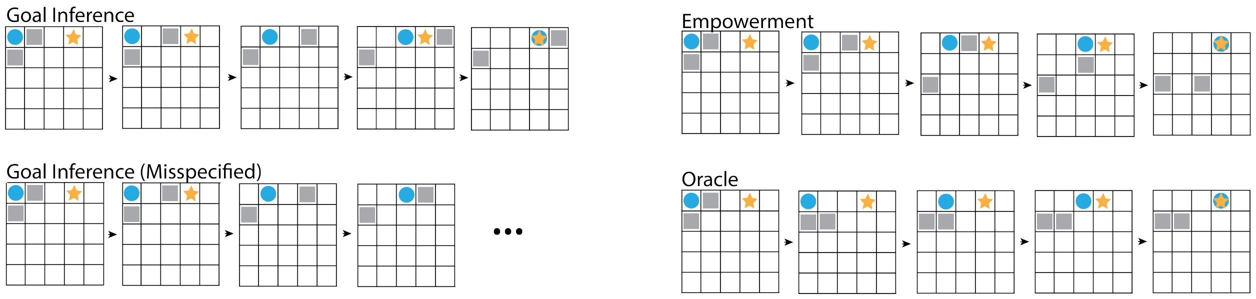


Figure 3.3: Sample rollouts with proxy human ●, immovable blocks ■, and goal at the star. Each frame consists of an action taken by the agent. Left column shows two cases of goal inference: ideal case (top) and failure mode with misspecified case (bottom) where the goal is blocked.

We compare against a goal inference method where the agent maintains a probability distribution over a candidate goal set and updates the likelihood of each goal based on the human's action at each step. We test variations of goal inference that highlight potential causes of failure: 1) when the goal set is too large, and 2) when the true goal is not in the goal set. Here, we define failure as the human's inability to achieve the goal before the experiment times out at 1000 steps. In this discrete scenario, we can compute empowerment directly by sampling 1000 trajectories and computing the logarithm of the number of distinct states the blocks and the human end up in. For comparison, we also provide results where we approximate empowerment with our proxy. The agent assumes that the human can move into any adjacent free space but does not know the human's policy or goal. We simulate a variety of initializations of the human's position, the number of blocks and their positions, and the goal position. Here we highlight the results of two main scenarios in Table 3.1: where the blocks trap the human in a corner, and where the blocks trap the human in the center of the grid. This is because the center of the grid has highest empowerment and the corner of the grid has lowest empowerment, and situations where the human is trapped are where assistance is most crucial. We ran 100 trials of randomized goal initializations and computed the number of steps the human takes to get to their goal (if successful) under each of the reward formulations.

Human in Corner | Success, (Mean Steps)	LG Set	SG Set	NG
GI (known)	90%, (4.35)	**100%**, (4.35)	
GI (unknown)	90%, (4.35)	92%, (4.51)	
Empowerment			**100%**, (5.24)
Empowerment Proxy			**99%**, (8.79)
Oracle			100%, (4.1)

Human in Center | Success, (Mean Steps)	LG Set	SG Set	NG
GI (known)	50%, (2.27)	**100%**, (2.57)	
GI (unknown)	50%, (2.27)	55%, (3.3)	
Empowerment			**100%**, (4.8)
Empowerment Proxy			**100%**, (6.71)
Oracle			100%, (1.91)

Table 3.1: Success rates and mean steps to goal (after removing failed trials) for human in corner (top) and human in center (bottom) scenarios. For human in corner, the human is randomly initialized in one of the four corners of the grid with two blocks trapping them. For human in center, the human is initialized at the center of the grid with four blocks surrounding them, one on each side. GI – goal inference with either the goal in the goal set (known) goal or the goal missing from the goal set (unknown). Large Goal (LG) Set considers every possible space as a goal, Small Goal (SG) Set only considered two possible goals, and No Goal (NG) is our method.

Analysis. Our results suggest that in the case where we have a small and correctly specified goal set, goal inference is the best strategy to use as it consistently succeeds and takes fewer steps to reach the goal on average. However, this represents an ideal case. When we have a misspecified goal set and/or a larger goal set, the success rate drops significantly. The failure modes occur when the agent is mistaken or uncertain about the goal location, and chooses to move a block on top of the true goal. As the human can no longer access the goal, they wait in an adjacent block and cannot act further to inform the agent about the true goal location, leading to an infinite loop. Since our empowerment method does not maintain a goal set, it does not encounter this issue and is able to succeed consistently, albeit with a tradeoff in higher mean steps performance. Interestingly, we note that the failure cases that the goal inference method encounters could easily be resolved by switching to the empowerment method – that is, since the human is next to a block that is on top of the goal, increasing human empowerment would move the block away and allow the human to access it. This highlights a potential for future work in hybrid assistance methods. In assistance scenarios where accurate goal inference may be too challenging to complete or the risks of assisting with an incorrect goal are too high, our goal-agnostic method provides assistance while circumventing potential issues with explicitly inferring human goals or rewards. As motivation for our proxy, we note that the proxy increases the mean steps to goal and can lead to a 1% decrease in success rate. This is because the inaccurate measure of empowerment can occasionally lead to blocking the goal, as in the non-ideal goal inferences cases. However, the proxy success rate is still much higher than that of goal inference.

Shared Autonomy: Lunar Lander

Our previous experiments motivate the benefits of a goal-agnostic human assistance method using human empowerment. To evaluate our method with real users, we ran a user study in the shared autonomy domain. The purpose of this experiment is two-fold: we demonstrate that a simplified empowerment-inspired proxy metric is sufficient for assisting the human while avoiding the computational demands of computing empowerment, and we demonstrate the efficacy of our method by assisting real humans in a challenging goal-oriented task without inference. For clarity, in this section we use ‘empowerment’ to denote the proxy defined in Section 4.3.

We build on recent work in this area by Reddy et al. (2018a) which proposed a human-in-the-loop model-free deep reinforcement learning (RL) method for tackling shared autonomy without assumptions of known dynamics, observation model, or set of user goals. Their method trains a policy (via Q-learning) that maps state and the user control input to an action. To maintain the user’s ability to control the system, the policy only changes the user input when it falls below the optimal Q-value action by some margin α. In this case, RL will implicitly attempt to infer the goal from the human input in order to perform better at the task, but this is incredibly challenging. With our method, empowerment is a way to explicitly encourage the shared autonomy system to be more controllable, allowing the user to more easily control the shared system. As a result, we hypothesize that optimizing for human empowerment will lead to a more useful assistant faster than vanilla RL.

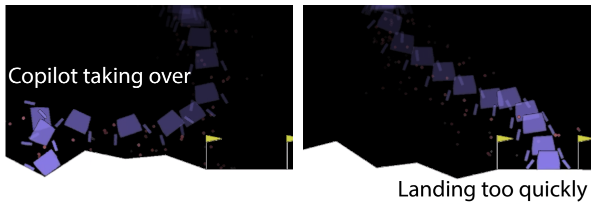
Experiment Setup. The main simulation used is Lunar Lander from OpenAI Gym (Brockman et al., 2016), as it emulates challenges of teleoperating a control system. Generally human players find the game incredibly difficult due to the dynamics, fast reaction time required, and limited control interface. The goal of the game is to control a lander using a main thruster and two lateral thrusters in order to land at a randomly generated site indicated by two flags, giving the player a positive reward. The game ends when the lander crashes on the surface, exceeds the time limit, or flies off the edges of the screen, giving the player a negative reward. The action space consists of six discrete actions that are combinations of the main thruster settings (on, off) and the lateral thruster settings (left, right, off). To assist players, we use DQN to train a copilot agent whose reward depends on landing successfully at the goal, but cannot observe where that goal is and instead keeps a memory of the past 20 actions input by the user. The user input and copilot action space are identical and the state-space contains the lander's position, velocity, tilt, angular velocity, and ground contact sensors. We compute the empowerment-based reward bonus by randomly rolling out 10 trajectories at each state and computing the variance in the final positions of the lander, as described in Section 4.3. Since both the human and agent control the same lander, the empowerment of the shared autonomy system is coupled. We hypothesize that including the empowerment term would be particularly useful for a challenging control task such as Lunar Lander, since states where the lander is more stable are easier for the player to control.

Simulation Experiments. We first test our method using simulated 'human' pilots as designed in the original study (Reddy et al., 2018a). First, an optimal pilot is trained with the goal as a part of its state space using DQN. Each of the 'human' pilots are imperfect augmentations of this optimal pilot as follows: the Noop pilot takes no actions, the Laggy pilot repeats its previously taken action with probability $p = 0.85$, the Noisy pilot takes a wrong action with probability $p = 0.3$ (e.g. go down instead of up), and the Sensor pilot only moves left or right depending on its position relative to the goal, but does not use the main thruster. We use these pilots as simulated proxies for humans and train two copilots with each of them: one with empowerment added to the reward function and one without. For the empowered copilots, we do a hyperparameter sweep over c_{emp} from 0.00001 to 10.0 and found the best performance with $c_{emp} = 0.001$. The copilots were trained on 500 episodes of max length 1000 steps on AWS EC2. We then conduct a cross evaluation for each copilot and pilot pair by counting successful landings across 100 evaluation episodes, averaged across 10 seeds.

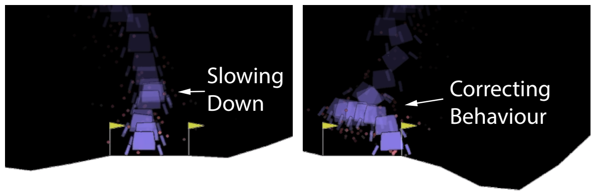
	Full	Noop	Laggy	Noisy	Sensor
No Copilot	**58.2**	0	11.9	11.6	0
No Empowerment (Baseline)		5.9	**38.4**	8.5	2.8
Empowerment		**9.7**	37.3	**30.7**	**10.8**

Table 3.2: Best successful landing percentages for each simulated pilot in 100 episodes, averaged across 10 seeds. Our method improves upon the baseline for all pilots except Laggy, where the performance is on par with the baseline.

We find that for almost all simulated pilots, our method leads to increased successful landing rate (shown in Table 3.2) when paired with the best copilot. In the only case where we do not outperform the baseline, with the Laggy pilot, the success rates between the baseline and our method only differ by 1.1%. In all cases, empowerment perform better than the simulated single pilot with no copilot. From our simulation results we see that our method can increase controllability for these simulated pilots, leading to higher successful landing rates in Lunar Lander.



(a) Without empowerment.



(b) With empowerment.

Figure 3.4: Sample trajectories from the user study. The leftmost image shows the copilot take over and swing the lander to the left even though the user is trying to move right. On the second image from the left, the copilot does not slow the motion down sufficiently so the lander has too much momentum and fails to land on the legs, leading to a crash. In the two right images, we see the frames overlap more than in the no empowerment case due to the slower, more stable motion. The rightmost image shows corrective stabilization behavior when approaching the goal from an inconvenient angle. See https://youtu.be/oQ1TvWG-Jns for a video summary of user study results.

User Study. To test our method with real pilots, we conducted an IRB-approved user study to compare human player performance on Lunar Lander with human-in-the-loop training. As with the simulated experiments, we manipulate the objective the copilot is trained with: our method with the empowerment bonus in the reward function ($c_{emp} = 0.001$) and the baseline with no empowerment. We found that the quality of assistance depends on this coefficient as follows: increasing c_{emp} generally makes the copilot more inclined to focus on stabilization, but if c_{emp} is too high, the copilot tends to override the pilot and focus only on hovering in the air. The objective measure of this study is the successful landing rate of the human-copilot team, and the subjective measure is based on a 7-point Likert scale survey about each participant's experience with either copilot. We designed the survey to capture whether the copilots improved task performance, increased/decreased the user's autonomy and control, and directly compare the user's personal preference between the two copilots (refer to Table 3.4). We recruited 20 (11 male, 9 female) participants aged 21-49 (mean 25). Each participant was given the rules of the game and an initial practice period of 25 episodes to familiarize themselves with the controls and scoring, practice the game, and alleviate future learning effects.

To speed up copilot learning, each copilot was first pretrained for 500 episodes with the simulated Laggy pilot, then fine-tuned as each human participant played for 50 episodes with each of the two copilot conditions without being informed how the copilots differ. To alleviate the confounding factor of improving at the game over time, we counterbalanced the order of the two conditions.

The objective success rates from the user study are summarized in Table 3.3. We ran a paired two-sample t-Test on the success rate when the copilot was trained with and without empowerment and found that the copilot with empowerment helped the human succeed at a significantly higher rate ($p < 0.0001$), supporting our hypothesis.

	No Empowerment	Empowerment
Success Rate	17 ± 9	$\mathbf{33 \pm 7}$

Table 3.3: Objective User Study Results: Average success rates with standard deviation.

The results of our survey are summarized in Table 3.4 where we report the mean response to each question for each copilot and the p-value of a paired two-sample t-Test. To account for multiple comparisons, we also report the Bonferroni-corrected p-values. We find that the participants perceived the empowerment copilot's actions made the game significantly easier ($p = 0.007$) as compared to the no-empowerment copilot. Furthermore, the two comparison questions have a significant level of internal consistency, with Chronbach's $\alpha = 0.96$, and the most frequent response is a strong preference (7) for the empowerment copilot. Although on average the perception of control and assistance is higher with the empowerment copilot, this difference was not found to be significant. Comments from the participants suggest that the empowerment copilot generally provided more stabilization at the expense of decreased human left/right thruster control, which the most users were able to leverage and collaborate with – the copilot increased stability and reduced the lander speed, allowing the user to better navigate to the goal (see Figure 3.4b). On the other hand, the no-empowerment copilot did not consistently provide stability or would take over user control, moving away from the goal (see Figure 3.4a). These results are exciting as prior work in shared autonomy has proposed heuristics for assistance, such as virtual fixtures (Marayong et al., 2002; Aarno et al., 2005) or potential fields (Aigner & McCarragher, 1997; Crandall & Goodrich, 2002), but we find that one benefit of empowerment is the natural emergence of stabilization without relying on heuristics. We also note that our empowerment-inspired reward bonus allowed us to leverage the intuitive benefits of controllability without the large computational costs of the empowerment method, allowing the empowerment-based copilot to learn alongside each human player in real time. While our method empirically demonstrated the merits of empowerment-based assistance, the user study also highlighted limitations of assistance through pure empowerment. Predominantly optimizing for empowerment can lead to failure modes where the assistant prevents landing altogether due to prioritizing stability excessively.

	Question	Emp	No-Emp	p-value	p-value*
Autonomy	I had sufficient control over the lander's movement.	4.7	3.6	**0.025**	0.175
	The copilot often overrode useful controls from me.	4.5	5.35	0.053	0.371
	The copilot did not provide enough assistance.	3	3.45	0.14	0.98
	The copilot provided too much assistance.	3.15	3.95	0.12	0.84
	The quality of the copilot improved over time.	6.25	5.75	0.096	0.672
Perform	The copilot's actions made parts of the task easier.	6.4	5.55	**0.001**	**0.007**
	The copilot's assistance increased my performance at the task.	6.4	5.6	**0.032**	0.21

Comparison Questions	Chronbach's α	Mean	Mode
I preferred the assistance from the empowerment copilot to the no empowerment copilot. I was more successful at completing the task with the empowerment copilot than the no empowerment copilot.	0.96	4.9 ± 2	7

Table 3.4: Subjective User Study Survey Results. 1 = strongly disagree, 7 = strongly agree. **p-value*** are Bonferonni-corrected p-values.

3.5 Conclusion

In this chapter, we introduce a novel formalization of assistance: rather than attempting to infer a human's goal(s) and explicitly helping them to achieve what we think they want, we instead propose empowering the human to increase their controllability over the environment. This serves as proof-of-concept for a new direction in human-agent collaboration. In particular, our work circumvents typical challenges in goal inference and shows the advantage of this method in different simulations where the agents are generally able to assist the human without assumptions about the human's intentions. We also propose an efficient algorithm inspired by empowerment for real time assistance-based use cases, allowing us to conduct human-in-the-loop training with our method and demonstrate success in assisting humans in a user study.

Limitations and Future Work. Our experiments find that in cases where the human's goals can be inferred accurately, general empowerment is not the best method to use. As there exist situations where optimizing for human empowerment will not be the best way to provide assistance, in future work we seek to formalize how goal-agnostic and goal-oriented assistance can be combined. For example, a natural continuation of this work we will explore a hybrid approach, combining local and global planning. Another area of future work is to explore the use of human empowerment for assisting humans with general reward functions. In this work, we primarily focus on assisting with goal states as a way to compare with existing goal inference assistance methods, but the human empowerment objective can potentially apply to more general reward formulations. Furthermore, we proposed a proxy to empowerment in order to make a user study feasible, which came at a cost of not being an accurate measure of true empowerment. Although we found that our simplified method led to significant improvement in our user study, the current proxy assumes homogeneous noise and is sensitive to scenarios with noise varying between different states, and the sample-based method naturally requires more computational power as action space grows – that being said, there are many meaningful assistance applications with small action spaces (e.g. navigation with mobility devices, utensil stabilization). Future work can analyze the trade-offs between computational tractability and numerical empowerment accuracy in the human assistance domain.

CHAPTER 4 HUMAN PRIORS FOR EXPLORATION

This chapter is based on the paper "Guiding Pretraining in Reinforcement Learning with Large Language Models" (Du et al., 2023c), written with Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas.

4.1 Introduction

Reinforcement learning algorithms work well when learners receive frequent rewards that incentivize progress toward target behaviors. But hand-defining such reward functions requires significant engineering efforts in all but the simplest cases (Amodei et al., 2016; Lehman et al., 2020). To master complex tasks in practice, RL agents may therefore need to learn some behaviors in the absence of externally-defined rewards. What should they learn? How can we design intrinsic rewards to elicit such behaviors?

Prior intrinsically motivated RL methods answer this question by augmenting rewards with auxiliary objectives based on novelty, surprise, uncertainty, or prediction errors (Bellemare et al., 2016; Pathak et al., 2017; Burda et al., 2019; Zhang et al., 2021; Liu & Abbeel, 2021; Yarats et al., 2021). But not everything novel or unpredictable is useful: noisy TVs and the movements of leaves on a tree may provide an infinite amount of novelty, but do not lead to meaningful behaviors (Burda et al., 2019). More recent approaches compute novelty with higher-level representations like language (Tam et al., 2022; Mu et al., 2022), but can continue driving the agent to explore behaviors that are unlikely to correspond to any human-meaningful goal—like enumerating unique configurations of furniture in a household. It is not sufficient for extrinsic-reward-free RL agents to optimize for novelty alone: learned behaviors must also be useful.

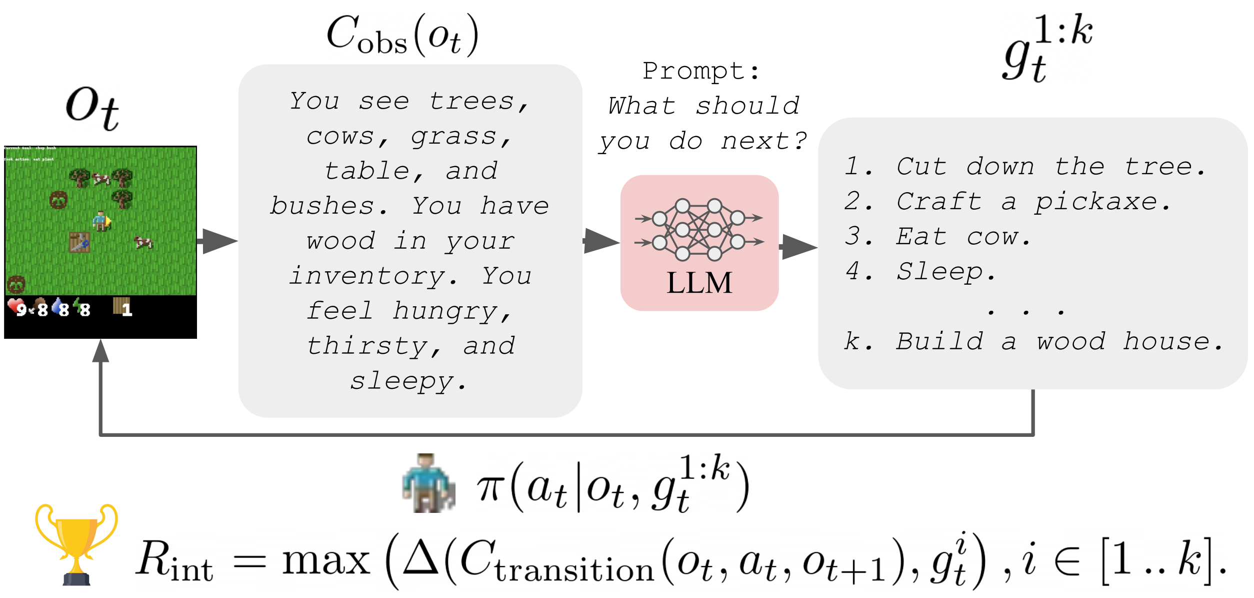

$C_{\text{obs}}(o_t)$
o_t
You see trees, cows, grass, table, and bushes. You have wood in your inventory. You feel hungry, thirsty, and sleepy.
Prompt: What should you do next?
LLM
$g_t^{1:k}$
1. Cut down the tree.
2. Craft a pickaxe.
3. Eat cow.
4. Sleep.
. . .
k. Build a wood house.
$\pi(a_t|o_t, g_t^{1:k})$
$R_{\text{int}} = \max\left(\Delta(C_{\text{transition}}(o_t, a_t, o_{t+1}), g_t^i\right), i \in [1 \,..\, k].$

Figure 4.1: **ELLM** uses a pretrained large language model (LLM) to suggest plausibly useful goals in a task-agnostic way. Building on LLM capabilities such as context-sensitivity and common-sense, ELLM trains RL agents to pursue goals that are likely meaningful without requiring direct human intervention. Prompt is illustrative; see full prompt and goal format in Appendix B.3.

In this chapter, we describe a method for intrinsic reward design using not just language-based representations but **pretrained language models** (LLMs) as a source of information about useful behavior. LLMs are probabilistic models of text trained on large text corpora; their predictions encode rich information about human common-sense knowledge and cultural conventions. Our method, **E**xploring with **LLM**s (ELLM), queries LMs for possible goals given an agent's current context and rewards agents for accomplishing those suggestions. As a result, exploration is biased towards completion of goals that are diverse, context-sensitive, and human-meaningful. ELLM-trained agents exhibit better coverage of useful behaviors during pretraining, and outperform or match baselines when fine-tuned on downstream tasks.

4.2 Related Work

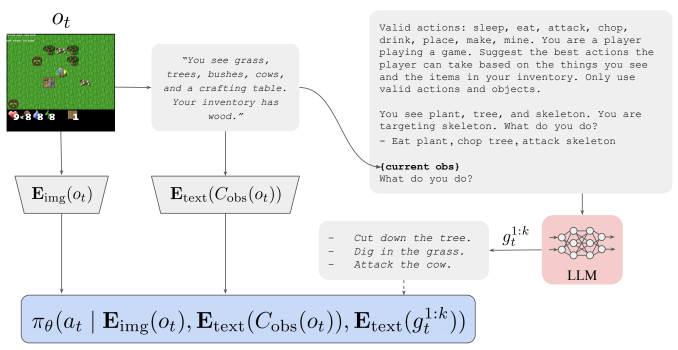
Intrinsically Motivated RL. When reward functions are sparse, agents often need to carry out a long, specific sequence of actions to achieve target tasks. As action spaces or target behaviors grow more complex, the space of alternative action sequences to explore grows combinatorially. As such, undirected exploration that randomly perturbs actions or policy parameters has little chance of succeeding (Ten et al., 2022; Ladosz et al., 2022).

Many distinct action sequences can lead to similar outcomes (Baranes & Oudeyer, 2013)—for example, most action sequences cause a humanoid agent to fall, while very few make it walk. Building on this observation, **intrinsically motivated** RL algorithms (IM-RL) choose to explore *outcomes* rather than actions (Oudeyer & Kaplan, 2007; Ten et al., 2022; Ladosz et al., 2022). **Knowledge-based IMs** (KB-IMs) focus on maximising the diversity of states (reviews in Aubret et al., 2019; Linke et al., 2020). **Competence-based IMs** (CB-IMs) maximise the diversity of *skills* mastered by the agent (review in Colas et al., 2022). Because most action sequences lead to a very restricted part of the outcome space (*e.g.* different ways of *falling on the floor* likely correspond to a single outcome), these methods lead to a greater diversity of outcomes than undirected exploration (Lehman et al., 2008; Colas et al., 2018).

However, maximizing diversity of outcomes may not always be enough. Complex environments can contain sources of infinite novelty. In such environments, seeking ever-more-novel states might drive learning towards behaviors that have little relevance to the true task reward. Humans do not explore outcome spaces uniformly, but instead rely on their physical and social common-sense to explore *plausibly-useful* behaviors first. In video games, they know that keys should be used to open doors, ladders should be climbed, and snakes might be enemies. If this semantic information is removed, their exploration becomes severely impacted (Dubey et al., 2018). The approach we introduce in this chapter, ELLM, may be interpreted as a CB-IM algorithm that seeks to explore the space of possible and plausibly-useful skills informed by human prior knowledge.

Linguistic Goals and Pretrained Language Models. One way of representing a diverse outcome space for exploration is through language. Training agents to achieve language goals brings several advantages: (1) goals are easy to express for non-expert users; (2) they can be more abstract than standard state-based goals (Colas et al., 2022); and (3) agents can generalize better thanks to the partial compositionality and recursivity of language (Hermann et al., 2017; Hill et al., 2019; Colas et al., 2020). Such linguistic goals can be used as instructions for language-conditioned imitation learning or RL. In RL, agents typically receive language instructions corresponding to the relevant reward functions (Luketina et al., 2019) and are only rarely intrinsically motivated (with the exception of Mu et al., 2022; Colas et al., 2020; Tam et al., 2022), where language is also used as a more general compact state abstraction for task-agnostic exploration.

Representing goals in language unlocks the possibility of using text representations and generative models of text (large language models, or LLMs) trained on large corpora. In imitation learning, text pretraining can help learners automatically recognize sub-goals and learn modular sub-policies from unlabelled demonstrations (Lynch & Sermanet, 2021; Sharma et al., 2021), or chain pre-trained goal-oriented policies together to accomplish high-level tasks (Yao et al., 2020; Huang et al., 2022a; Ahn et al., 2022; Huang et al., 2022b). In RL, LM-encoded goal descriptions greatly improve the generalization of instruction-following agents across instructions (Chan et al., 2019) and from synthetic to natural goals (Hill et al., 2020). LLMs have also been used as proxy reward functions when prompted with desired behaviors (Kwon et al., 2023). Unlike these approaches, ELLM uses pretrained LLMs to constrain exploration towards plausibly-useful goals in a task-agnostic manner. It does not



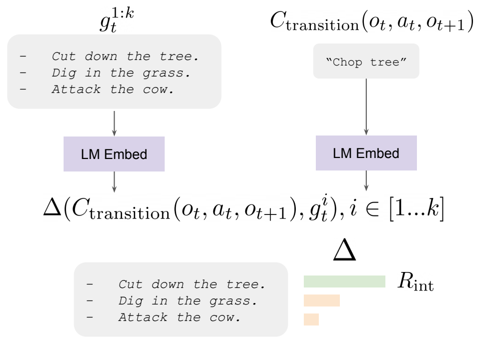
```
Valid actions: sleep, eat, attack, chop,
drink, place, make, mine. You are a player
playing a game. Suggest the best actions the
player can take based on the things you see
and the items in your inventory. Only use
valid actions and objects.

You see plant, tree, and skeleton. You are
targeting skeleton. What do you do?
- Eat plant, chop tree, attack skeleton

{current obs}
What do you do?
```

o_t
"You see grass, trees, bushes, cows, and a crafting table. Your inventory has wood."
$\mathbf{E}_{\text{img}}(o_t)$
$\mathbf{E}_{\text{text}}(C_{\text{obs}}(o_t))$
- Cut down the tree.
- Dig in the grass.
- Attack the cow.
$g_t^{1:k}$
LLM
$\pi_\theta(a_t \mid \mathbf{E}_{\text{img}}(o_t), \mathbf{E}_{\text{text}}(C_{\text{obs}}(o_t)), \mathbf{E}_{\text{text}}(g_t^{1:k}))$

(a) Policy parametrization for ELLM. We optionally condition on embeddings of the goals $E_{\text{text}}(g_t^{1:k})$ and state $E_{\text{text}}(C_{\text{obs}}(o_t))$.



(b) LLM reward scheme. We reward the agent for the similarity between the captioned transition and the goals.

Figure 4.2: ELLM uses GPT-3 to suggest adequate exploratory goals and SentenceBERT embeddings to compute the similarity between suggested goals and demonstrated behaviors as a form of intrinsically-motivated reward.

assume a pretrained low-level policy, demonstrations, or task-specific prompts. Most similar to our work, Choi et al. (2022) also prompt LLMs for priors. However, they use LM priors to classify safe and unsafe states to reward, which is a subset of common-sense exploratory behaviors ELLM should generate. Also similar to our work, Kant et al. (2022) query LLMs for zero-shot commonsense priors in the Housekeep environment, but they apply these to a planning task rather than as rewards for reinforcement learning.

4.3 Structuring Exploration with LLM Priors

Problem Description. We consider Partially Observed Markov Decision Processes (POMDP) defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \Omega, \mathcal{T}, \gamma, \mathcal{R})$, in which observations $o \in \Omega$ derive from environment states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$ via $\mathcal{O}(o \mid s, a)$. $\mathcal{T}(s' \mid s, a)$ describes the dynamics of the environment while $\mathcal{R}$ and γ are the environment's reward function and discount factor.

IM agents optimize for an intrinsic reward $\mathcal{R}_{\text{int}}$ alongside or in place of $\mathcal{R}$. CB-IM methods, in particular, define $\mathcal{R}_{\text{int}}$ via a family of goal-conditioned reward functions:

$$\mathcal{R}_{\text{int}}(o,\, a,\, o') = \mathbb{E}_{g \sim \mathcal{G}}[\mathcal{R}_{\text{int}}(o,\, a,\, o' \mid g)]. \tag{4.1}$$

A CB-IM agent is expected to perform well with respect to the original $\mathcal{R}$ when the intrinsic reward $\mathcal{R}_{\text{int}}$ is both easier to optimize and well aligned with $\mathcal{R}$, such that behaviors maximizing $\mathcal{R}_{\text{int}}$ also maximize $\mathcal{R}$. Every CB-IM algorithm must define two elements in Equation 4.1: (1) the distribution of goals to sample from, i.e. $\mathcal{G}$, and (2) the goal-conditioned reward functions $\mathcal{R}_{\text{int}}(o, a, o' \mid g)$. Given these, A CB-IM algorithm trains a goal-conditioned policy $\pi(a \mid o, g)$ to maximize R_{int}. For some intrinsic reward functions, agents may achieve high reward under the original reward function $\mathcal{R}$ immediately; for others, additional fine-tuning with $\mathcal{R}$ may be required. In eq. (4.1), the space of goals $\mathcal{G}$ is determined by the goal-conditioned reward function $R_{\text{int}}(\cdot \mid g)$: every choice of g induces a corresponding distribution over optimal behaviors.

Goal-based Exploration Desiderata. How should we choose $\mathcal{G}$ and $\mathcal{R}_{\text{int}}(\cdot \mid g)$ to help agents make progress toward general reward functions $\mathcal{R}$? Goals targeted during exploration should satisfy three properties:

- **Diverse**: targeting diverse goals increases the chance that the target behavior is similar to one of them.
- **Common-sense sensitive**: learning should focus on feasible goals (`chop a tree` $>$ `drink a tree`) which are likely under the distribution of goals humans care about (`drink water` $>$ `walk into lava`).
- **Context sensitive**: learning should focus on goals that are feasible in the current environment configuration (e.g. `chop a tree` only if a tree is in view).

Most CB-IM algorithms hand-define the reward functions R_{int} (2) and the support of the goal distribution (1) in alignment with the original task $\mathcal{R}$, but use various intrinsic motivations to guide goal sampling (1): e.g. novelty, learning progress, intermediate difficulty (see a review in Colas et al., 2022). In **E**xploring with **L**arge **L**anguage **M**odels (ELLM), we propose to leverage language-based goal representations and language-model-based goal generation to alleviate the need for environment-specific hand-coded definitions of (1) and (2). We hypothesize that world knowledge captured in LLMs will enable the automatic generation of goals that are diverse, human-meaningful and context sensitive.

Goal Generation with LLMs ($\mathcal{G}$). Pretrained large language models broadly fall into three categories: autoregressive, masked, or encoder-decoder models Min et al. (2021). Autoregressive models (e.g. GPT; Radford et al., 2018), are trained to maximize the log-likelihood of the next word given all previous words, and are thus capable of language generation. Encoder-only models (e.g. BERT; Devlin et al., 2018), are trained with a masked objective, enabling effective encoding of sentence semantics. Pretraining LMs on large text corpora yields impressive zero- or few-shot on diverse language understanding and generation tasks, including tasks requiring not just linguistic knowledge but world knowledge Brown et al. (2020b).

ELLM uses autoregressive LMs to generate goals and masked LMs to build vector representations of goals. When LLMs generate goals, the support of the goal distribution becomes as large as the space of natural language strings. While querying LLMs unconditionally for goals can offer diversity and common-sense sensitivity, context-sensitivity requires knowledge of agent state. Thus, at each timestep we acquire goals by prompting the LLM with a list of the agent's available actions and a text description of the current observation via a *state captioner* $C_{\text{obs}} : \Omega \to \Sigma^*$, where Σ^* is the set of all strings (see Figure 4.2).

We investigate two concrete strategies for extracting goals from LLMs: (1) open-ended generation, in which the LLM outputs text descriptions of suggested goals (e.g. `next you should...`), and (2) closed-form, in which a possible goal is given to the LLM as a QA task (e.g. `Should the agent do X? (Yes/No)`). Here the LLM goal suggestion is only accepted when the log-probability of `Yes` is greater than `No`. The former is more suited for open-ended exploration and the latter is more suited for environments with large but delimitable goal spaces. While the LLM does not have prior knowledge of all possible goals, we can provide some guidance towards desirable suggestions through few-shot prompting. See Appendix B.3 for the full prompt.

Rewarding LLM Goals ($\mathbf{R_{int}}$). Next we consider the goal-conditioned reward (2). We compute rewards for a given goal g ($\mathcal{R}_{\text{int}}$ in Eq. 4.1) by measuring the semantic similarity between the LLM-generated goal and the description of the agent's transition in the environment as computed by a *transition captioner* $C_{\text{transition}} : \Omega \times \mathcal{A} \times \Omega \to \Sigma$:

$$\mathcal{R}_{\text{int}}(o, a, o' \mid g) = \begin{cases} \Delta(C_{\text{transition}}(o, a, o'), g) & \text{if } > T \\ 0 & \text{otherwise.} \end{cases}$$

Here, the semantic similarity function $\Delta(\cdot\,,\,\cdot)$ is defined as the cosine similarity between *representations* from an LM encoder $E(\cdot)$ of captions and goals:

$$\Delta(C_{\text{transition}}(o, a, o'), g) = \frac{E(C_{\text{transition}}(o, a, o')) \cdot E(g)}{\|E(C_{\text{transition}}(o, a, o'))\| \|E(g)\|}.$$

In practice, we use a pretrained SentenceBERT model (Reimers & Gurevych, 2019) for $E(\cdot)$. We choose cosine similarity to measure alignment between atomic agent actions and freeform LLM generations, as done in prior work (Huang et al., 2022a). When the caption of a transition is sufficiently close to the goal description ($\Delta > T$), where T is a similarity threshold hyperparameter, the agent is rewarded proportionally to their similarity. Finally, since there can be multiple goals suggested, we reward the agent for achieving any of the k suggestions by taking the maximum of the goal-specific rewards:

$$\Delta^{\max} = \max_{i=1\ldots k} \Delta\left(C_{\text{transition}}(o_t, a_t, o_{t+1}), g_t^i\right).$$

As a result, the general reward function of CB-IM methods from Equation 4.1 can be rewritten:

$$\mathcal{R}_{\text{int}}(o, a, o') = \mathbb{E}_{\mathbf{LLM}(g^{1\,..\,k} \mid C_{\text{obs}}(o))}\left[\Delta^{\max}\right]. \tag{4.2}$$

Implementation Details. The full ELLM algorithm is summarized in Algorithm 2. See Figure 4.1 for the high-level pipeline. To impose a novelty bias, we also filter out LM suggestions that the agent has already achieved earlier in the same episode. This prevents the agent from exploring the same goal repeatedly. In Appendix B.12 we show this step is essential to the method.

We consider two forms of agent training: (1) a **goal-conditioned** setting where the agent is given a sentence embedding of the list of suggested goals, $\pi(a \mid o, E(g^{1:k}))$, and (2) a **goal-free** setting where the agent does not have access to the suggested goals, $\pi(a \mid o)$. While R_{int} remains the same in either case, training a goal-conditioned agent introduces both challenges and benefits: it can take time for the agent to learn the meaning of the different goals and connect it to the reward, but having a language-goal conditioned policy can be more amenable to downstream tasks than an agent just trained on an exploration reward. We also consider two types of policy inputs– (1) just the partially observed pixel observations, or (2) the pixel observations combined with the embedded language-state captions $E(C_{\text{obs}}(o))$. Since (2) performs better (see analysis in Appendix B.5), we use (2) for all experiments unless otherwise specified. All variants are trained with the DQN algorithm (Mnih et al., 2013), with implementation details in Appendix B.8.

This method focuses on the benefits of LLM priors for RL exploration and mostly assumes a pre-existing captioning function. In simulation, this can be acquired for free with the ground truth simulator state. For real world applications, one can use object-detection (Zaidi et al., 2022), captioning models (Stefanini et al., 2022), or action recognition models (Kong & Fu, 2022). Alternatively, one could use multi-modal vision-language models with a similar LM component (Alayrac et al., 2022). To test the robustness of our method under varying captioning quality, Section 4.4 studies a relaxation of these assumptions by looking at a variant of ELLM using a learned captioner trained on human descriptions.

4.4 Experiments

Our experiments test the following hypotheses:

- **(H1)** Prompted pretrained LLMs can generate plausibly-useful exploratory goals satisfying the desiderata described above: diversity, common-sense and context sensitivity.
- **(H2)** Training an ELLM agent on these exploratory goals improves performance on downstream tasks compared to methods that do not leverage LLM-priors.

We evaluate ELLM in two complex environments: (1) *Crafter*, an open-ended environment in which exploration is required to discover long-term survival strategies (Hafner, 2022), and (2) *Housekeep*, an embodied robotics environment that requires common-sense to restrict the exploration of possible rearrangements of household objects (Kant et al., 2022). Besides environment affordances, these environments also differ in viewpoint (3rd vs 1st person) and

Algorithm 2 ELLM Algorithm

```
Initialize untrained policy $\pi$
$t \leftarrow 0$
$o_t \leftarrow$ env.RESET()
while $t <$ max_env_steps do
    # Generate $k$ suggestions, filtering achieved ones
    $g_t^{1:k} \leftarrow$PREV_ACHIEVED(LLM($C_{\text{obs}}(o_t)$))
    # Interact with the environment
    $a_t \sim \pi(a_t | o_t, \mathrm{E}(C_{\text{obs}}(o_t))), \mathrm{E}(g_t^{1:k}))$
    $s_{t+1} \leftarrow$ env.STEP($a_t$)
    # Compute suggestion achievement reward
    $r_t \leftarrow 0$
    $\Delta^{max} \leftarrow \max_{i=1...k} \Delta(C_{\text{transition}}(o_t, a_t, o_{t+1}), g_t^i)$
    if $\Delta^{max} >$ threshold then
        $r_t = \Delta^{max}$
    end if
    # Update agent using any RL algorithm
    $\text{Buffer}_{t+1} \leftarrow \text{Buffer}_t \cup (o_t, a_t, g_t^{1:k}, r_t, o_{t+1})$
    $\pi \leftarrow$UPDATE($\pi$, $\text{Buffer}_{t+1}$)
end while
```

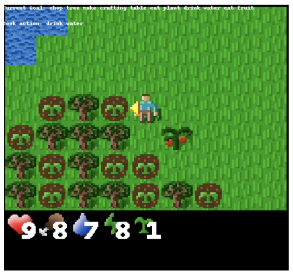
action space (large high-level vs low-level). In each environment, we compare ELLM with existing IM-RL methods (Liu & Abbeel, 2021; Burda et al., 2019), an oracle with ground-truth rewards, and ablations of ELLM; see Table 4.1.

Crafter

Environment description. We first test ELLM in the Crafter environment, a 2D version of Minecraft (Hafner, 2022). Like Minecraft, Crafter is a procedurally generated and partially observable world that enables collecting and creating a set of artifacts organized along an achievement tree which lists all possible achievements and their respective prerequisites (see Figure 4 in Hafner, 2022). Although Crafter does not come with a single main task to solve, we can track agent progress along the achievement tree.

We modify the original game in two ways. Crafter's original action space already incorporates a great deal of human domain knowledge: a single `do` action is interpreted in different ways based on the agent's context, each of which would correspond to a very different low-level action in a real environment ('`do`' means '`attack`' in front of a zombie but '`eat`' in front of a plant). We remove this assistance by augmenting the action space with more specific verb + noun pairs that are not guaranteed to be useful (e.g. '`eat zombie`'). This makes it possible in Crafter to attempt a wide range of irrelevant/nonsensical tasks, providing an opportunity for an LM narrow the goal space down to reasonable goals. See Appendix B.2 for details. Second, to make RL training easier across all conditions, we increase the damage

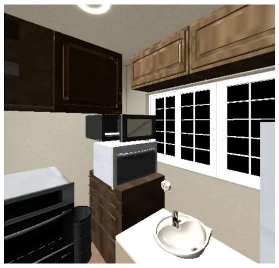
```
You see {observation}.
You have in your inventory {items}*.
You feel {health status}*.
*omitted if empty.
```



```
You see bush, grass, plant, tree, and
water. You have in your inventory
sapling.
```

🏆
- Plant sapling
- Chop tree
- Chop bush

```
Seen objects: {object, receptacle}.
Seen receptacles: {receptacles}.
You are holding {gripped_object}.
```



```
Seen objects: clock in kitchen sink.
Seen receptacles: kitchen bottom
cabinet, kitchen sink, living room
shelf, living room carpet …
You are holding a cereal box.
```

🏆
- Place cereal box in kitchen cabinet
- Pick clock

Figure 4.3: Sample templated captions and suggested goals.

the agent does against enemies and reduce the amount of wood required to craft a table from 2 to 1; see Appendix Figure B.2 for comparisons.

We use Codex (Chen et al., 2021b) as our LLM with the open-ended suggestion generation variant of ELLM, where we directly take the generated text from the LLM as the set of suggested goals to reward. Each query prompt consists of a list of possible verbs the agent can use (but not a list of all possible nouns), a description of the agent's current state, and the question '`What do you do?`'. We add two examples of similar queries to the start of the prompt in order to guide the language model to format suggestions in a consistent way; see the full prompt in Appendix B.3.

Goals suggested by the LLM. To answer **H1**, we study the goals suggested by the LLM in Table 4.2: are they diverse, context-sensitive and common-sensical? The majority of suggested goals (64.9%) are context-sensitive, sensible, and achievable in the game. Most of the 5% of goals not allowed by Crafter's physics (e.g. `build a house`) are context- and common-sensitive as well. The last third of the goals violate either context-sensitivity (13.6%) or common-sense (16.4%). See Appendix B.11 for details.

Pretraining exploration performance. A perfect exploration method would unlock all Crafter achievements in every episode, even without prior knowledge of the set of possible achievements. Thus, we measure exploration quality as the average number of unique achievements per episode across pretraining (Figure 4.4). Although it is not given access to Crafter's achievement tree, ELLM learns to unlock about 6 achievements every episode,

Method	Description
ELLM (ours)	Rewards the agent for achieving any goal suggested by the LLM using the similarity-based reward functions R_{int} defined in Eq. 4.2. It only rewards the agent for achieving a given goal once per episode (novelty bias).
Oracle (Crafter only)	The upper bound: it suggests all context-sensitive goals at any step, only common-sensical ones (from the list of valid goals) and uses the same novelty bias as ELLM. Rewards are computed exactly with a hard-coded R_{int}.
Novelty	This baseline removes the common-sense sensitivity assumption of the *Oracle* and rewards the agent for achieving any of the goals expressible in the environment including invalid ones (e.g. `drink tree`) as long as the agent performs the goal-reaching action in the right context (e.g. while facing a tree). Uses a hard-coded R_{int} and a novelty bias like the *Oracle*.
Uniform	This variant removes the novelty bias from *Novelty* and samples uniformly from the set of expressible goals.
APT (Liu & Abbeel, 2021)	State-of-the-art KB-IM algorithm that maximizes state entropy computed as the distance between the current state's embedding e_s and its K nearest neighbors $e_{s[1..K]}$ within a minibatch uniformly sampled from memory. There is no goal involved and $R_{\text{int}} = \log \lVert e_s - e_{s[1..K]} \rVert$.
RND (Burda et al., 2019)	State-of-the-art KB-IM algorithm that rewards the agent for maximizing a form of novelty estimated by the prediction error of a model h trained to predict the output of a random network $\tilde{h}$. $R_{\text{int}} = \lVert h(s,a) - \tilde{h}(s,a) \rVert$.

Table 4.1: Descriptions of the compared algorithms. (Additional comparisons in Appendix B.14).

against 9 for the ground-truth-reward Oracle (Figure 4.4). It outperforms all exploration methods that only focus on generating novel behaviors (APT, RND, Novelty) — all limited to less than 3 achievements in average. As shown in Table 4.2, ELLM does not only focus on novelty but also generates common-sensical goals. This boosts exploration in Crafter, supporting **H1**.

We also test variants of each method (with / without goal conditioning, with / without text observations) where applicable. We do not find goal conditioning to bring a significant advantage in performance during pretraining. The non-conditioned agent might infer the goals (and thus the rewarded behaviors) from context alone. Similarly to Mu et al. (2022) and Tam et al. (2022), we find that agents trained on visual + textual observations (as computed by $E(C_{\text{obs}}(o))$) outperform agents trained on visual observations only (opaque vs semi-transparent bars in Appendix Figure B.4). That said, optimizing for novelty alone, whether in visual or semantic spaces, seems insufficient for fully solving Crafter.

The naïve approach of finetuning a pretrained policy on the downstream task performs poorly across all pretraining algorithms. We hypothesize this is because relevant features and

	Suggested	Rewarded
Context-Insensitive	13.6%	1.1%
Common-Sense Insensitive	16.4%	32.4%
Good	64.9%	66.5%
Impossible	5.0%	0%

Table 4.2: Fractions of suggested and rewarded goals that fail to satisfy context-sensitivity or common-sense sensitivity; that satisfy these properties and are achievable in Crafter (Good); or that are not allowed by Crafter's physics. See Appendix B.11 for examples of each.

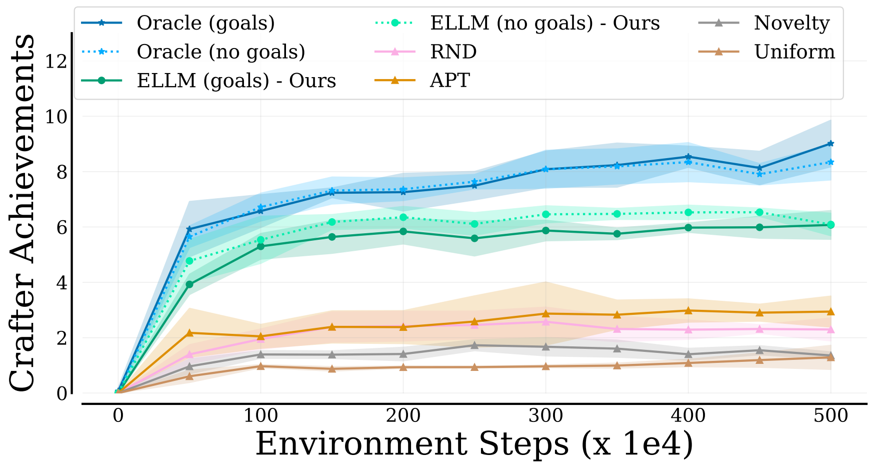


Figure 4.4: Ground truth achievements unlocked per episode across pretraining, mean±std across 5 seeds.

Q-values change significantly between pretraining and finetuning, especially when the density of rewards changes. Instead, we find it is more effective to use the pretrained policy for guided exploration. We initialize and train a new agent, but replace 50% of the algorithm's randomly-sampled ϵ-greedy exploration actions with actions sampled from the pretrained policy. In Appendix B.13 we include the poor finetuning results discuss why we think guided exploration does better.

Figure 4.5 compares the downstream performance of ELLM to the performance of the two strongest baselines RND and APT using both transfer methods. (full comparisons with all baselines shown in Appendix B.1). For the goal-conditioned version of ELLM, we provide the agent with the sequence of subgoals required to achieve the task. Even though not all subgoals were mastered during pretraining, we still observe that the goal-conditioned pretrained agents outperform the unconditioned ones.

Performance of the different methods varies widely task-to-task and even seed-to-seed since each task requires a different set of skills, and any given agent may or may not have learned a particular skill during pretraining. For instance, ELLM agents typically learn to place crafting tables and attack cows during pretraining, leading to low-variance learning curves. They typically do not learn to make wood swords, so we see a high-variance learning curve which depends on how quickly each agent stumbles across the goal during finetuning. Despite the variance, we see that goal-conditioned ELLM stands out as the best-performing method on average. Notably, ELLM (both goal-conditioned and goal-free) is the only method with nonzero performance across all tasks.

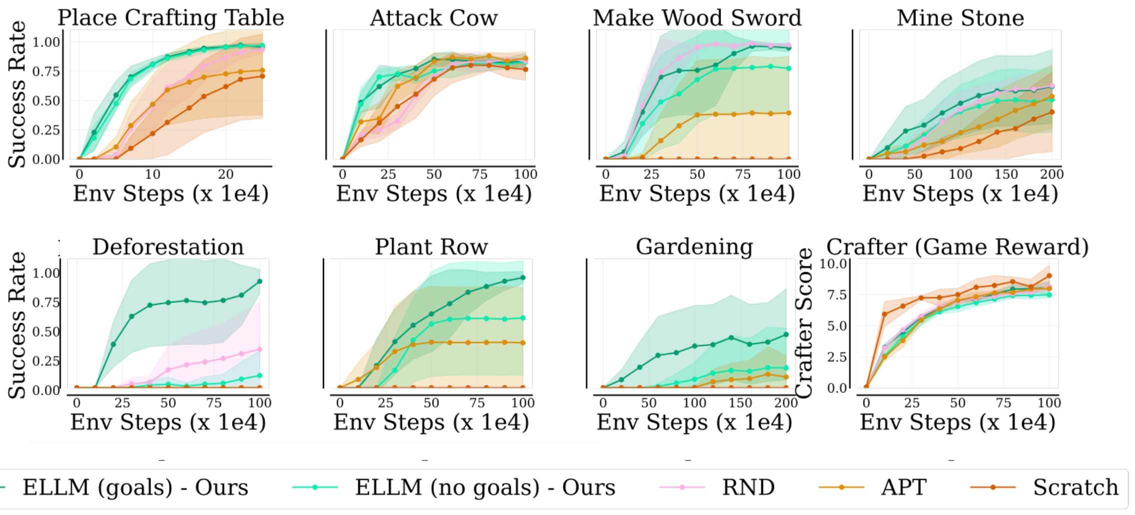


Figure 4.5: Success rates across training for each of the seven downstream tasks in Crafter. Each run trains an agent from scratch while leveraging a pretrained policy for exploration. Plots show mean ± std for 5 seeds. Some plots have multiple overlapping curves at 0.

ELLM with imperfect transition captioner. Perfect captioners might not be easy to obtain in some environments. However, trained captioners might generate more linguistic diversity and make mistakes. To test the robustness of ELLM to diverse and imperfect captions, we replace the oracle transition captioner $C_{\text{transition}}$ with a captioner trained on a mixture of human and synthetic data (847+900 labels) using the ClipCap algorithm (Mokady et al., 2021b). Synthetic data removes some of the human labor while still providing a diversity of captions for any single transition (3 to 8). Appendix B.10 presents implementation details and analyzes how the trained captioner might cause errors in generated rewards. Although its false negative rate is low (it detects goal achievements well), its false positive rate is rather high. This means it might generate rewards for achievements that were not unlocked due to a high similarity between the generated caption and goal description generated by the LLM. In ELLM pretraining, we use the learned captioner to caption transitions where an action is successful and use that caption to compute the reward via the similarity

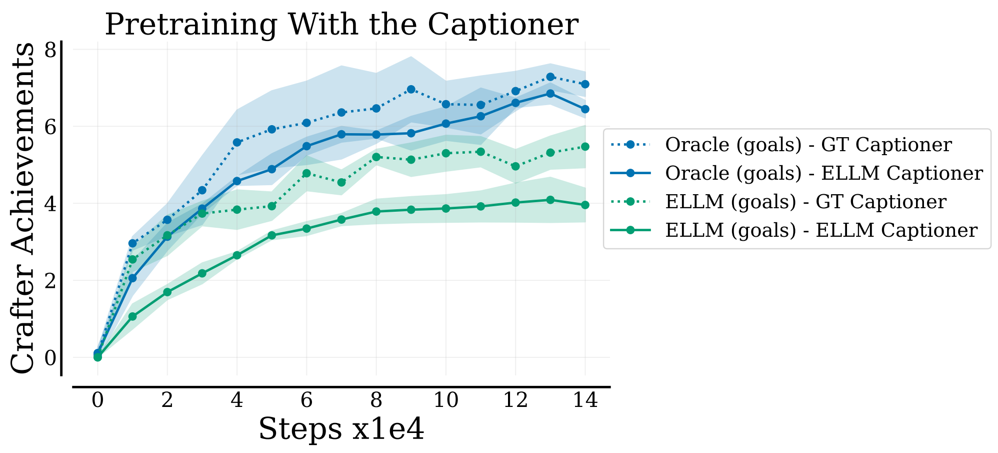
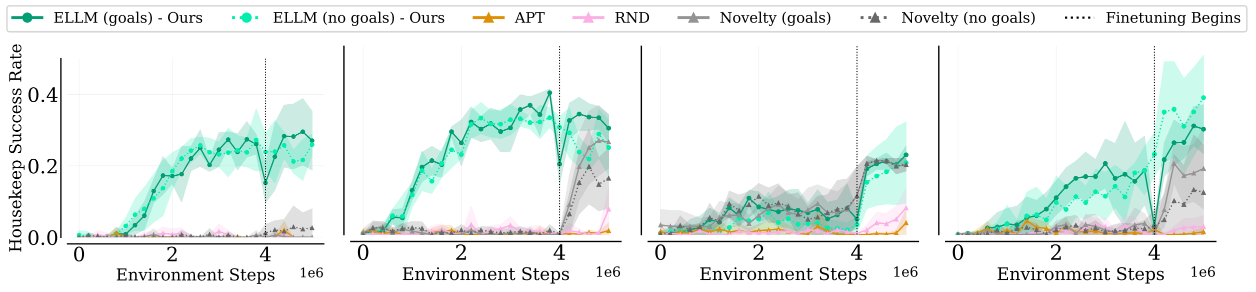
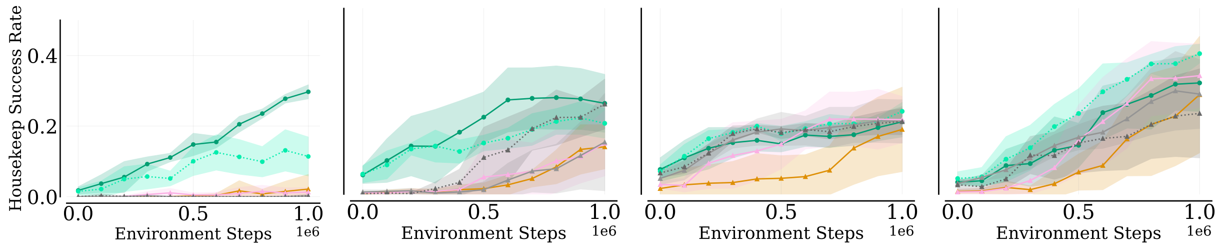


Figure 4.6: Pretraining with a learned captioner vs a ground truth captioner. We see performance drops, especially for ELLM, but still relatively good performance. (3 seeds, mean± std.)

metric (see Section 4.3). Figure 4.6 shows that ELLM performance is overall robust to this imperfect captioner.



(a) *Pretraining and finetuning:* pretraining for 4M steps then finetuning for 1M steps on the ground truth correct arrangement.



(b) *Downstream evaluation:* Using the frozen pretrained exploration policies only for ϵ-greedy-style action selection for 1M steps.

Figure 4.7: *Housekeep:* Correct arrangement success rates on 4 object-receptacle task sets. Mean $\pm$ std over 5 seeds.

	Task 1	**Task 2**	**Task 3**	**Task 4**
Match Acc.	85.7%	87.5%	50%	66.7%
Mismatch Acc.	93.8%	90.1%	94.0%	87.6%

Table 4.3: Classification accuracy of LLM for each Housekeep task (top row is true positives, bottom row is true negatives).

Housekeep

Environment description. Housekeep is an embodied robotics environment where the agent is tasked with cleaning up a house by rearranging misplaced objects (Kant et al., 2022). The agent must successfully match the environment's ground truth correct mapping of objects to receptacles without direct instructions specifying how objects need to be rearranged. This mapping was determined via crowd-sourcing common-sense object-receptacle combinations. An example layout of the task can be found in Figure 1 in Kant et al. (2022). Common-sense priors are necessary for learning to rearrange misplaced objects into reasonable configurations.

We focus on a simplified subset of Housekeep consisting of 4 different scenes with one room each, each with 5 different misplaced objects and a suite of different possible receptacles; see Appendix B.6 for details. Because the agent does not have access to the ground truth target locations, we use the game reward's rearrangement success rate as a measure of exploration quality: common-sensical exploration should perform better. A success rate of 100% means the agent has picked and placed all 5 misplaced objects in correct locations. Note that we intentionally focus on a domain where the downstream application benefits strongly from exploring reasonable goals during pretraining. Rather than designing reward functions that correspond to all correct rearrangements for all possible objects, we investigate whether ELLM can be a general purpose method that guides learning human-meaningful behaviors.

Unlike Crafter's combinatorial and high-level action space, Housekeep operates with low-level actions: moving forward, turning, looking up or down, and picking or placing an object. This allows us to investigate whether ELLM enables high-level exploration despite using lower-level control. We assume access to an egocentric instance segmentation sensor to generate captions of in-view objects and receptacles, and use the `text-davinci-002` InstructGPT model (Ouyang et al., 2022) as our LLM. Given a description of visible objects, the receptacles the objects are currently in, and all previously seen receptacles, we create a list of all possible object-receptacle mappings. We use the closed-form variant of ELLM and query the LLM for whether each object should be placed in each receptacle as a `yes/no` question. By querying for each object-receptacle combination individually, we are able to cache and efficiently reuse LLM queries. The agent can be given two types of goals: (1) picking an object if it is not already in a suggested receptacle, and (2) placing a gripped object in a suggested receptacle.

Goals suggested by LLM. In Housekeep, we assess LLM goals by looking at the classification accuracy of correct and incorrect arrangements (Table 4.3). We find that the LLM accuracy at identifying mismatches (e.g. `vase in kitchen sink`) are all above 87%, however, accuracy of identifying matches varies greatly depending on the available objects and receptacles (ranging from 50-90%). Since there are only a few correct positions, each false negative hurts accuracy greatly. Taking a closer look, we find that some LLM labels are reasonable despite disagreeing with the environment mapping: e.g. suggesting `vase in living room table`, and not suggesting `pan in living room cabinet`. This suggests that there are ambiguities in the ground truth mappings, likely due to human disagreement.

Pretraining and downstream performance. To investigate **H1**, we compare ELLM against the strongest baselines (RND, APT, Novelty) described in Table 4.1. In Housekeep the novelty baseline rewards the agent for novel instances of pick or place actions in an episode, allowing us to differentiate between success attributable solely to the captioner and the pick/place prior, and success attributable to any LLM common-sense priors. For brevity, we focus only on the pixel + text-observation variant of all methods. Sample efficiency curves measuring the ground truth rearrangement success during both pretraining and finetuning are shown in Figure 4.7a. In three of the four tasks, we find that the ELLM bias leads to higher success rates during pretraining, suggesting coverage better aligned with the downstream task compared to the baselines. We also find much higher pretraining success rates in the first two tasks. Since Table 4.3 shows higher LLM accuracy for these two tasks, this difference shows the impact of LLM inaccuracies on pretraining.

For **H2**, we test two different ways of using the pretrained models in the downstream rearrangement task. First, we directly finetune the pretrained model on the ground truth correct rearrangement; shown after the dashed vertical line in Figure 4.7a. Here, the success rates for finetuned ELLM matches or outperform the baselines, especially if pretraining has already led to high success rates. Interestingly, we also find that the goal-conditioned ELLM variant consistently suffers a drop in performance when finetuning starts. We hypothesize this is due to the treatment of all suggested goals as a single string, so if any single goal changes between pretraining and finetuning the agent must relearn the goal embedding changes. Second, in Figure 4.7b we present results for directly training a new agent on the downstream task, using the frozen pretrained model as an exploratory actor during ϵ-greedy exploration. Once again, we observe that ELLM consistently matches or outperforms all baselines. We also see here that the KB-IM baselines are more competitive, suggesting that this training scheme is better suited for pretrained exploration agents that are not well-aligned to the downstream task.

4.5 Conclusion

We present ELLM, an intrinsic motivation method that aims to bias exploration towards common-sense and plausibly useful behaviors via a pretrained LLM. We have shown that such priors are useful for pretraining agents in extrinsic-reward-free settings that require common-sense behaviors that other exploration methods fail to capture.

ELLM goes beyond standard novelty search approaches by concentrating exploration on common-sensical goals. This is helpful in environments offering a wide array of possible behaviors among which very few can said to be *plausibly useful*. It is less helpful in environments with little room for goal-based exploration, when human common-sense is irrelevant or cannot be expressed in language (e.g. fine-grained manipulation), or where state information is not naturally encoded as a natural language string.

Limitations and Future Work. LLM performance is sensitive to prompt choice. Even with a well-chosen prompt, LLMs sometimes make errors, often due to missing domain-specific knowledge. False negatives can permanently prevent the agent from learning a key skill: in Crafter, for example, the LLM never suggests creating wood pickaxes. There are multiple avenues to address this limitation: (1) combining ELLM rewards with other KB-IM rewards like RND, (2) prompting LLMs with descriptions of past achievements (or other feedback about environment dynamics) so that LLMs can learn about the space of achievable goals, (3) injecting domain knowledge into LLM prompts, or (4) fine-tuning LLMs on task-specific data. While ELLM does not rely on this domain knowledge, when this information exists it is easy to incorporate.

ELLM requires states and transition captions. Our learned captioner experiments Figure 4.6 suggest we can learn these from human-labeled samples, but in some environments training this captioner might be less efficient than collecting demonstrations or hard-coding a reward function. Still, we are optimistic that as progress in general-purpose captioning models continues, off-the-shelf captioners will become feasible for more tasks. Lastly, suggestion quality improves considerably with model size. Querying massive LLMs regularly may be time- and cost-prohibitive in some RL environments.

As general-purpose generative models become available in domains other than text, ELLM-like approaches might also be used to suggest plausible visual goals, or goals in other state representations. ELLM may thus serve as a platform for future work that develops even more general and flexible strategies for incorporating human background knowledge into reinforcement learning.

CHAPTER 5 HUMAN INTRINSIC OBJECTIVES

This chapter is based on the paper "What can AI Learn from Human Exploration? Intrinsically-Motivated Humans and Agents in Open-World Exploration" (Du et al., 2023b), written with Eliza Kosoy, Li Dayan, Maria Rufova, Pieter Abbeel, and Alison Gopnik.

5.1 Introduction

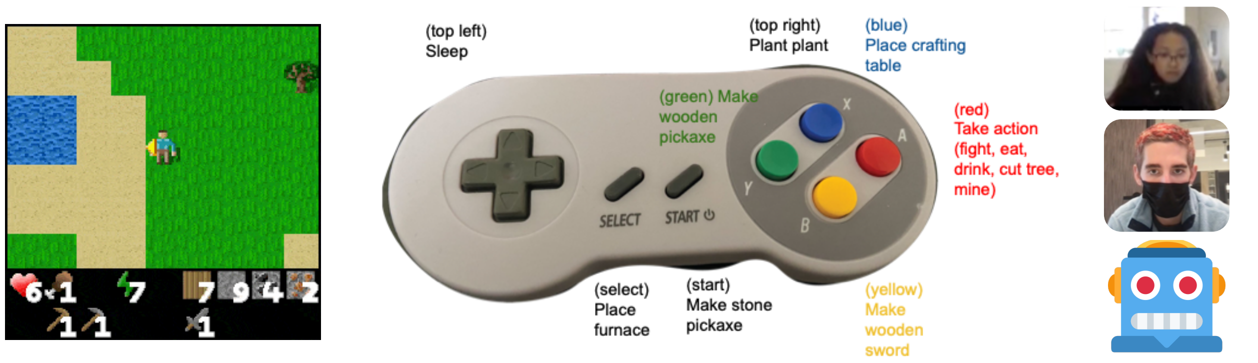


Figure 5.1: Left: Example screen from Crafter Hafner (2022). The player is at the center of the screen; the yellow arrow shows which direction they are facing. Their health, food, water and energy status are at the bottom left, the raw materials they have collected are at the bottom right, and the tools built so far are in the bottom row. Middle: Actions available to the human participants and RL agents. Right: We compare behaviors of children, adults, and RL agents.

Humans often explore new environments remarkably effectively, even in the total absence of external rewards (Matusch et al., 2020). There have been many attempts to formalize the natural curiosity of humans, but evidence from empirical studies about what is actually motivating human exploration remains inconclusive (Poli et al., 2022). Furthermore, these studies are limited in that they tend to put humans in very simple and unrealistic environments and look at at limited range of exploratory behaviors. Tracking the complexities of more open-ended and spontaneous exploration has proven challenging. Kosoy et. al have begun designing more complex unified online environments which allow spontaneous exploration (Kosoy et al., 2020, 2022), but have still primarily focused on simple tasks. There is not yet a realistic environment that allows for more than one type of activity and many of the environments are task specific. Other work such as (Pelz, 2020) are limited by what can be clicked within the realm of a specific game. Our goal is to study human exploration in a more complex setting that can also help us develop more effective intrinsic rewards for artificial agents. Reinforcement learning (RL) agents must actively collect meaningful experience in order to find optimal behaviors in initially unknown environments. To facilitate this, existing works have proposed various objectives that guide exploration by approximating some notion of novelty or curiosity, with some commonly-used concepts being count-based state-visitation (Ostrovski et al., 2017; Machado et al., 2018), prediction error (Pathak et al., 2017; Schmidhuber, 1991), state novelty (Burda et al., 2018; Zhang et al., 2021), skill learning (Eysenbach et al., 2018; Lee et al., 2019), or information gain (Houthooft et al., 2016; Pathak et al., 2019); see (Aubret et al., 2019; Portelas et al., 2020) for surveys. However, general intrinsically motivated RL agents still don't come close to human-level sample efficiency, these intrinsic rewards sometimes produce counterproductive behavior (Burda et al., 2018), and it remains unclear if engineered intrinsic rewards are truly aligned with human exploration. Few studies use human exploration as a basis for agent exploration—some examples include illuminating key differences between human and agent priors (Dubey et al., 2018), or developing objectives loosely inspired by curiosity and novelty seeking (Pathak et al., 2017).

Motivated by the gap between human and agent exploration, we study the behavior of humans and agents in the same environment—Crafter (Hafner, 2022), a Minecraft-like complex, open-ended environment. We collect play data from both children and adults, emphasizing that the child behavioral data is important for gaining insights into fundamental untrained exploration capacities of humans. We propose five ways of scoring exploration in the game and analyze how well the exploration performance of humans and agents correlate with commonly-used information theoretic objectives that have also been used to explain exploration motivations: Entropy, Information Gain, and Empowerment. Interestingly, we find that only human exploration performance is consistently positively correlated with the information theoretic objectives, despite the agents being explicitly trained to optimize approximations of the given objectives and the overall exploration scores of humans and agents spanning a similar range. We also record and transcribe human utterances during play, and in a preliminary analysis find a significant positive correlation between children's frequency of verbalizing goals and their Empowerment, supporting previous work in psychology which has suggested that self-talk could play an important role in children's creative problem-solving

(Lee, 2011) and in AI which suggests goal-generation aids exploration in agents (Du et al., 2023c; Hu et al., 2023).

5.2 Related Work

Exploration in AI. Exploration strategies in AI range in complexity from occasionally taking random actions (*e.g.*, ϵ-greedy) to optimizing complex intrinsic reward functions. Intrinsic rewards are often motivated by objectives such as: increasing entropy, information gain, and/or empowerment. State entropy maximization objectives use the intuition that exploration is motivated by visiting diverse states. Information gain measures the amount of information gained about the environment (Lindley, 1956), where exploration is motivated by reducing surprise, developing a better understanding of environment dynamics. Empowerment measures of the number of available options (Klyubin et al., 2005b,a), such that maximizing empowerment encourages exploration that increases the agent's control.

However, outside of very simple environments, actual implementation of these objectives is limited. Approximations have been proposed in prior works: count-based exploration bonuses (Tang et al., 2017; Bellemare et al., 2016), entropy-maximization (Hazan et al., 2019; Liu & Abbeel, 2021; Yarats et al., 2021), curiosity-based approaches that encourage agents to take actions that are maximally informative about the environment; for example, by rewarding states or transitions that the agent can not yet predict well (Schmidhuber, 1991; Pathak et al., 2017; Burda et al., 2018; Zhang et al., 2021), leveraging Bayesian networks (Houthooft et al., 2016), or network ensembles (Pathak et al., 2019). Empowerment-based objectives can learn behaviors that have measurable influence over the environment (Gregor et al., 2016; Eysenbach et al., 2018; Klyubin et al., 2005a). However, even with sophisticated exploration objectives, RL agents often lag far behind human sample efficiency (Matusch et al., 2020; Hafner et al., 2023). In the rare cases that human-level exploration is achieved, this is done by a painstaking amount of hard-coded structure (Tsividis et al., 2021).

One explanation for the gap between human and agent behavior is the vast prior knowledge that humans have due to their life experience. (Dubey et al., 2018) find that removing visual priors greatly reduces human abilities, while agents are unimpacted. (Du et al., 2023c) propose using large language models as a fuzzy repository of human knowledge as a way of incorporating human priors in RL exploration. That said, prior knowledge by itself is useless without an exploration objective. Our work aims to understand which *objectives* motivate human exploration, and how that can inform intrinsic rewards for agents.

Exploration in Cognitive Science. Dating back to Piaget (1933), developmental researchers have conceived of children as active and curious learners who are intrinsically motivated to explore the world in systematic and rational ways (Schulz & Bonawitz, 2007; Cook et al., 2011; Legare, 2012; Schulz, 2012); see Schulz (2012) for a review. As in the AI literature, there is just as much variety in proposed objectives underlying human curiosity and exploration (Ten et al., 2022; Pelz, 2020), including perceived novelty (Taffoni et al., 2014; Berlyne, 1950; Smock & Holt, 1962; Poli et al., 2022) which simply suggests that humans are drawn to stimuli that appear more novel, expected learning progress (Baldassarre et al., 2014; Ten et al., 2021; Oudeyer & Kaplan, 2007) which is the idea that people find it intrinsically rewarding to improve their performance, information gain (Liquin et al., 2021; Ruggeri et al., 2021; Addyman & Mareschal, 2013) where the driver of exploration is to gather maximal information about the environment, (or even more specifically the possibility of learning causal relations (Taffoni et al., 2014)) maximizing empowerment (Brändle et al., 2022, 2023) and totally random exploration more common in young children (Meder et al., 2021).

Although humans have been found to be sensitive to many of the above exploration objectives, evidence on what exactly people base their exploration on is inconsistent (Poli et al., 2022). Many papers propose that humans are driven by a combination of objectives, such as a desire for both knowledge of task space and competence across that space (Baranes et al., 2014), or that humans find it rewarding to perform above a certain level while simultaneously making substantial learning progress (Ten et al., 2021). This mirrors how RL agents also commonly maximize the weighted sum of multiple objective functions, the weighting of which is also often changed during the course of the agent's lifetime of environment interactions (Pitis et al., 2020; Tsividis et al., 2021). Studies thus far have been limited to highly simplistic and unrealistic environments, typically stateless or with only a couple different states (Gershman, 2018) or where participants are asked to choose from a limited set of options (Baranes et al., 2014; Ten et al., 2021). At the more naturalistic end are 3D maze environments where participants can move around (Kosoy et al., 2020), but these are still quite limited with navigation being the only available task. Our hope is that studying human exploration in a richer environment can help shed light onto which exploration objectives people actually use and why they are so effective.

Language and Exploration. We also partially use utterances to understand human exploration in this work; while we are not aware of existing work on verbalizations and intrinsic motivation, there are some works on verbalization and problem solving. It has been found across a wide range of studies that verbalization or private speech can be helpful for understanding situations and surmounting difficulties, for instance by focusing attention on important features and discarding irrelevant ones (Schunk, 1986; Granato et al., 2020), or assisting with coding and retention of information (Gidley Larson & Suchy, 2015). This suggests that participants with more verbalizations might explore better because they can better process the flow of information from their environment. Furthermore, overt verbalization (i.e. thinking aloud) is especially common for younger children (ages 6-7), and particularly when encountering obstacles (Vygotsky, 1962). This could suggest a stronger correlation between success in exploration and frequency of utterances for children, but not adults.

5.3 Environment and Data Collection

Open-Ended Environment: Crafter. Motivated by the lack of human exploration studies in rich and open-ended environments, we conduct our comparisons in Crafter Hafner (2022). Similar to Minecraft, Crafter comprises of exploration challenges in both the breadth and depth of activities to explore. The player controls a character in a procedurally generated world containing various resources that can be collected and used to replenish health or build tools (Figure 5.1). Players are able to explore a breadth of skills: collecting food and water, sleeping, and avoiding or killing enemies, as well as depth of skills: crafting increasingly complex tools. This gives rise to the achievement tree in Figure 5.2.

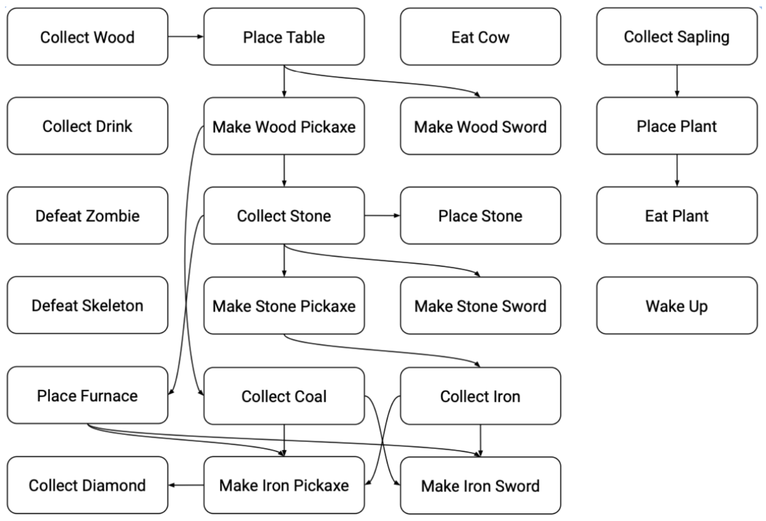


Figure 5.2: Dependency tree of all the achievements that can be unlocked in Crafter (from Figure 4, Hafner (2022)). Due to our action pruning, `Collect Diamond, Make Iron Pickaxe, Make Iron Sword, Make Stone Sword, Place Stone` are not achievable in our setting.

The available actions a either move the player or enable interactions with the environment. Interactions only affect the cell that the player is directly facing, with the "do" action being the most versatile (used to eat, drink, cut tree, mine, fight). Seven additional actions execute a unique action. Note that we remove three of the most complex actions from the original game as they were not feasible to fit on a conventional game controller (see Figure 5.1, centre). Interaction actions have no effect if the player lacks sufficient prerequisites (*e.g.* `place crafting table` only works if the player has sufficient wood in their inventory). While the original Crafter work contained expert human data, we focus on collecting play data from adults and children who are fully unfamiliar with the game in a reward-free setting so we can observe how they explore in an unknown environment.

We also modify the game to make it easier for human play. First, we slightly lengthen the fraction of an in-game day that is spent in daylight by changing the daylight function from $1 - |\cos(\pi x)|^3$ to $1 - |\cos(\pi x)|^{12}$. We also add an explicit 'Game Over' screen when an episode ends so participants are aware of episode transitions. Lastly, we slightly prune the action space to fit the available actions onto the handheld controller (Figure 5.2).

Participants. In this pre-registered, IRB approved study (AsPredicted reference: 92521). We recruited 51 children between the ages of 6-10 years (Mean age: 8.6 years, Female: 19, Male: 32) from the Bay Area Discovery Museum (BADM), as well as 24 adults from the University of California, Berkeley campus ages 18-25 years (Mean age 24.8, Female: 10, Male: 14). No direction was given about the game in order to encourage open-ended play, and participants were allowed to play for up to 20 minutes. Participants who were not able to complete at least one full game round were excluded. We found that 80% of children had video game experience with 64.7% having Minecraft-specific experience, and 79.1% of adults had video game experience with 54.1% having played Minecraft previously.

Data Collection Procedure. In this study, we introduced children and adults to the novel "Crafter" game. Participants were first shown a short tutorial video explaining what each controller button did (Figure 5.1) and then allowed to play for up to 20 minutes, with the option to quit early. During this time period, the game automatically restarts a new episode any time the player died to a Game Over screen. Participants were not shown any score or given any objective–which was reflected in the wide variation in responses to the question about the point of the game, ranging from "just have fun" or "try not to rage quit" to "killing the skeletons" or "don't die". All actions taken and the complete world state was recorded for every time-step while playing, along with audio from the participant which was later transcribed manually with timestamps. Due to a lack of consent for audio recording for all participants, this resulted in transcripts from 35 children and 22 adults.

Agent Training Procedure. We train three RL agents and use one random agent as baselines. The random agent samples noop 47.5% of the time in order to match the average reaction time of the human players, and uniformly samples all available actions otherwise. For trained agents, we compare against state-of-the-art intrinsic RL objectives: NovelD (Zhang et al., 2021) and APT (Liu & Abbeel, 2021). NovelD incentivizes information gain by providing a large intrinsic reward at the boundary between explored and unexplored regions, using RND (Burda et al., 2018) as a measure of state novelty. APT uses a particle-based entropy estimator (Singh et al., 2003) to reward the agent for maximizing state entropy in an abstract representation space. As a measure of best-case performance we also train an agent with the game extrinsic reward function, which reveals the possible set of achievements. The game reward function provides a sparse reward of 1 every time a new achievement is unlocked alongside a small health-based reward every time the agent is hurt or healed. The agent policy input is a simplified semantic representation of the game: the material in the cell the agent is facing, the status, and the inventory. All RL agents are trained with Rainbow DQN (Hessel et al., 2018) for one million timesteps, using the same hyperparameters for Crafter from Table B.3. We report 12 seeds for each agent.

5.4 Experiments

Overview of Exploration Scores

As there is no single objective measure for “good exploration", we construct five exploration scores for Crafter (Table 5.1). As the difficulty and number of achievements unlocked is a simple measure of how well a player explores semantically meaningful state changes in Crafter, four measures are achievement-based. The last one, map coverage, is based on task-agnostic physical exploration.

Exploration Score	Definition
Achievement Score	Number of unique achievements (cells of Figure 5.2) unlocked throughout gameplay.
Weighted Achievement Score	Same as score, but accounts for task complexity by weighting each achieved task by its level in the skill tree (*i.e.* deeper tasks contribute more to the score).
Breadth Score	A measure of how broadly the player has explored the task space by calculating how much progress has been made in a breadth-first traversal of the skill tree (*i.e.* only count tasks up to and including the first incomplete level of the tree).
Depth Score	A measure of how deeply the player has explored the task space by returning the depth of the deepest task achieved.
Map Coverage Score	Percentage of the game map covered.

Table 5.1: Description of exploration scores proposed for our study.

Summary Statistics

We present summary statistics across all human and agent data. First, Figure 5.3 shows summary histograms for each measure, showing the normalized density of people and agents on the proposed exploration scores. We find that adults generally score better than children, and there is a wider diversity of performance among both children and adults than any individual agent condition. However, we note that the overall spread of human and agent performances are similar–*i.e.*, it is not the case that humans greatly outperform the agents or vice versa.

Next, we look at how exploration progression over time differs between humans and agents. Figure 5.4 looks at a random subset of participants and trained agents, plotting the total number of unique achievements they have ever unlocked over time. Again, adults on average score higher than children. There is also larger variation in children gameplay, with many children who quit playing early, including those who were making rapid progress.

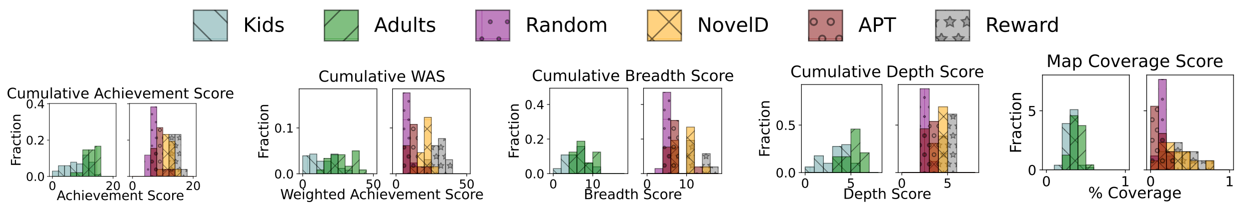


Figure 5.3: Summary density histograms for each exploration score. Cumulative measures are cumulative across all episodes, while Map Coverage averages across episodes. Left plot for each measure shows human performance, right plot shows agent performance.

Agents are much less sample efficient, reaching similar performance only after over 100× the number of environment interactions used by humans.

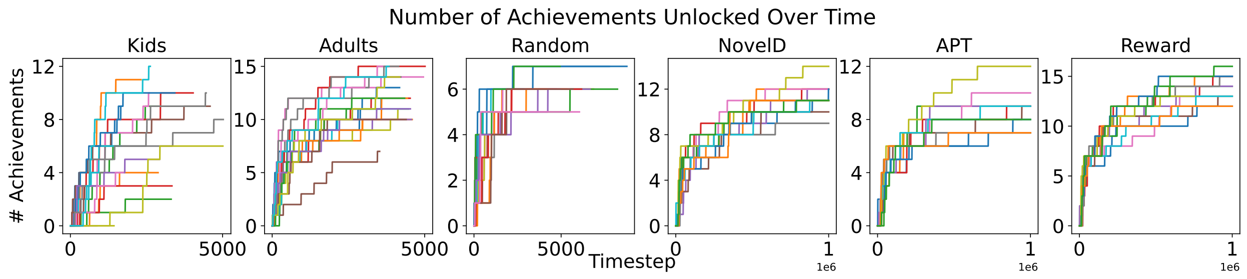


Figure 5.4: The total number of unique achievements unlocked over time.

Finally, we examine whether exploration is more focused on breadth or depth. Figure 5.5 shows a normalized 2D histogram of exploration breadth and depth through the achievement tree. Notably, children show the clearest correlation between the breadth and depth of exploration, whereas all other groups have participants or agents that are more focused on either breadth or depth. This suggests that children may play in a way that explores diverse and increasingly complex skills similarly.

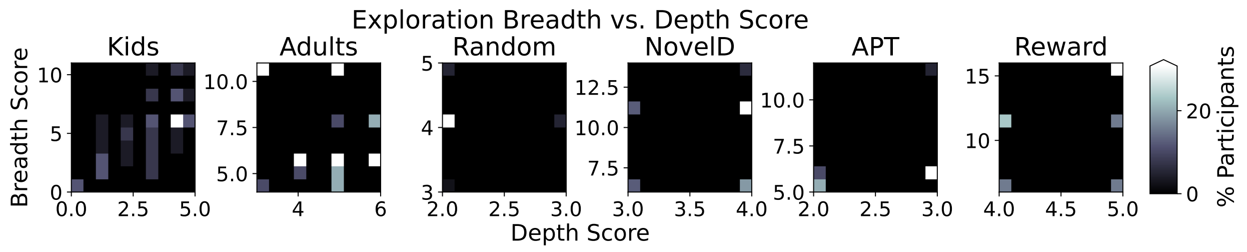


Figure 5.5: Exploration Breadth vs. Depth through the achievement tree.

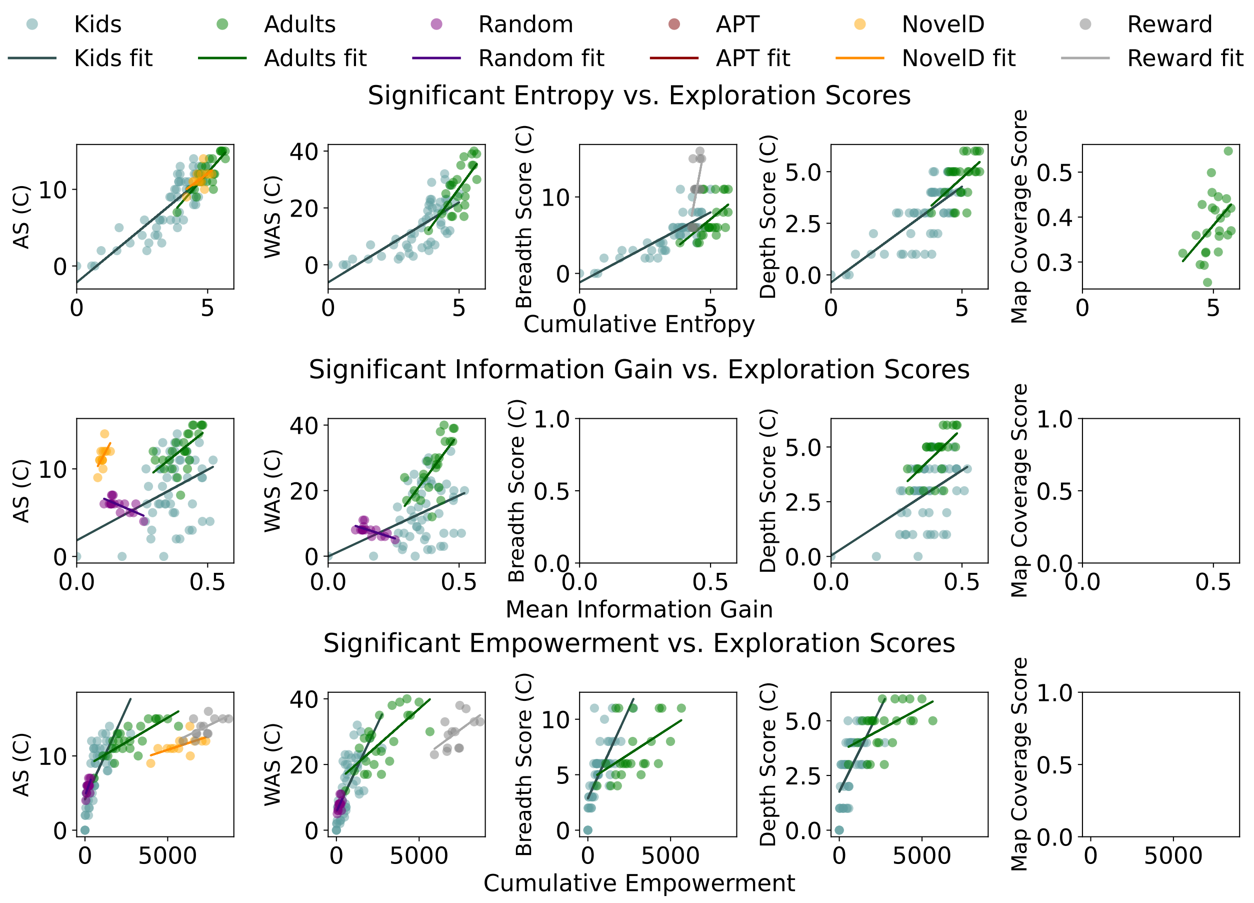


Figure 5.6: Information theoretic objectives vs. exploration scores with significance $p < 0.05$ on the linear fit. We note that only adults and children consistently show a significant positive correlation between the achievement-based scores and the given objectives. We generally do not see a significant relationship between physical map coverage and the objectives. For full plots, see Figure C.3.

Analyzing Information Theoretic Objectives

We verify whether objectives proposed as intrinsic motivation functions are indeed significantly correlated with human and agent exploration. We focus on Entropy, Information Gain, and Empowerment, each of which have been proposed as motivations for exploration in both the AI and cognitive science literature (see Section 5.2). Rather than computing these functions on raw pixel inputs, we construct a state representation s that inherently imbues some prior knowledge by combining the following: the semantic label of the cell the player is currently facing, the contents of their inventory, and the increase in their status from the previous state, if any. This captures aspects of the environment that the player is most likely to be paying attention to and has direct control over, while aiming to avoid meaningless increases in the objectives (*e.g.*, visually novel configurations of the procedurally generated map that are not semantically novel). We use this representation to construct transition tables of each participant's and agent's behaviors, mapping each transition (s, a, s') to the number of times it was experienced.

Entropy. The entropy of the distribution of states visited throughout play is given by

$$\text{Entropy} = \sum_s -p(s)\log(p(s)) \tag{5.1}$$

Concretely, we compute $p(s)$ as $N_s / \sum_s N_s$, where N_s is the number of times a transition in the transition table started with s. We report the cumulative entropy over the person or agent's total experience. This can be interpreted as a measure of the diversity of all visited states.

Information Gain. We measure the total information gain from all experiences of taking action a from state s as the log count of the total number of times that transition has been made (Matusch et al., 2020).

$$IG(s, a) = \log(1 + N_{(s,a)}), \tag{5.2}$$

where $N_{(s,a)}$ is the number of times that same action a has been taken given being in state s. We then report the average amount of information gained per transition (accounting for all past experiences) as the overall information gain of a player's experience.

$$\text{Information Gain} = \frac{\sum_{(s,a)} IG(s,a)}{\sum_{(s,a)} N_{(s,a)}} \tag{5.3}$$

This can be interpreted as a measure of novel transitions encountered, such that the player acquires less information each time they take the same action in a known state. We use the log-count approximation as prior work has found it to perform similarly to more complex measures (Matusch et al., 2020). A similar alternative is to use the square root instead of the logarithm (Brändle et al., 2023).

Empowerment. Empowerment is defined as the channel capacity of the agent's actuation channel (Klyubin et al., 2005a). We compute one-step empowerment as most impactful actions in Crafter are single-step:

$$\text{Empowerment} = \max_{p(a|s)} \mathcal{I}\left[a; s'|s\right] = \max_{p(a|s)} \sum_{\mathcal{A},S} p(s'|a)p(a)\log\frac{p(s'|a)}{\sum_{\mathcal{A}} p(s'|a)p(a)} \tag{5.4}$$

We use the Blahut-Arimoto (Dupuis et al., 2004) algorithm to approximate the channel capacity, as proposed in Klyubin et al. (2005a). We report the cumulative empowerment over the person or agent's total experience. This can be interpreted as a measure of the amount of control the agent has over visited states, or the amount of information the agent could inject into the environment.

In Figure 5.6, we plot each information theoretic objective against the proposed exploration scores and compute a least squares linear fit for each condition. For clarity, we only show the plots with a significant correlation ($p < 0.05$). Interestingly, we find only the humans consistently exhibit significant positive correlation between the exploration scores and the information theoretic objectives (the only exception being Breadth vs. Information Gain). This is despite the non-random agents spanning similar absolute score values (see Figure 5.3), achieving Entropy and Empowerment comparable to adults (see Figure C.3), and the children performing worse on average across all scores. For example, the NovelD agent behavior across all objectives correlates well with Achievement Score but none of the other exploration scores. This suggests human exploration may be advantageously guided by information theoretic intrinsic motivations, while the agents' is not, *even when agents are explicitly trained on intrinsic rewards that aim to approximate those same motivations*. We do not see similar correlations for map coverage, suggesting that the information theoretic objectives are more aligned with exploration in the skill space rather than just physical exploration of the map space.

Analyzing Verbalizations

We investigate whether self-talk might help exploration by examining the relationship between verbalizations and the intrinsic objectives. Following prior works using LLMs for summarizing human data (Rathje et al., 2023), we take transcriptions from each participant and use ChatGPT (`gpt-3.5-turbo`) to classify whether each utterance expresses a question and/or a goal. To improve accuracy, we also ask the LLM to generate reasoning before making each classification.

Questions	Questions about the game, such as "How do I move?" or "What is that skeleton doing?"
Goals	Stated goals for the game, such as "I need to get some water."

Table 5.2: Classes of verbalizations analyzed in our study.

The children talked significantly more during gameplay than the adults (averaging 240 vs 160 words per session, despite adults often playing for longer). To account for different play durations, we normalize the number of utterances by the total number of timesteps played. Our exploratory analysis found, among just the child participants, a significant correlation between the fraction of verbalizations expressing goals or questions, and the cumulative Empowerment (see Figure 5.7), with goals exhibiting by far the highest correlation ($r^2 = 0.28, p = 0.005$ unadjusted). This corroborates with prior findings in psychology that self-talk can help direct and focus problem solving in children, especially by focusing their behavior in a goal-directed manner, and findings in AI that agents that generate goals may explore more effectively (Lee, 2011; Hu et al., 2023; Du et al., 2023c). Inferring which exploration motivations are implied by the choice of verbalized goal is important future work.

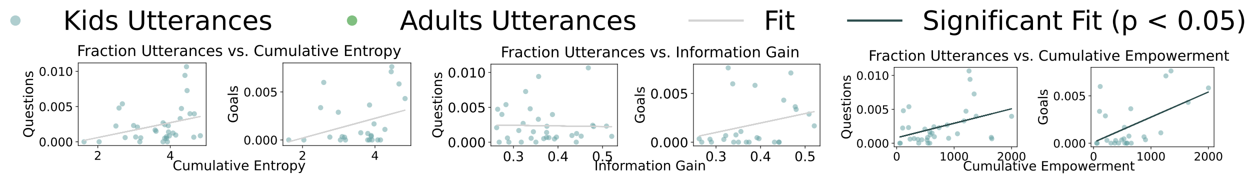


Figure 5.7: Fraction of verbalized questions and goals vs. cumulative entropy, information gain, and empowerment for children. We find that the relationship between the fraction of uttered goals and empowerment has the highest correlation and largest significance ($r^2 = 0.28, p = 0.005$ unadjusted). No significant relationship was found in the adult data (full plots in Figure C.1).

5.5 Conclusion

Conclusions Our goal in this work is to develop an understanding of human exploratory behaviors in an open-ended environment. To this end, we propose a framework for studying human and agent behaviors in a shared, open-ended environment within Crafter, with various scores for measuring exploration quality. We find that both children and adult exploration success consistently correlates with Entropy, Information Gain, and Empowerment. On the other hand, we find surprisingly that this is not the case for either intrinsically or extrinsically motivated RL agents, despite the intrinsically motivated agents using objectives that approximate maximizing Entropy or Information Gain. This suggests that perhaps humans are making use of the information theoretic objectives for exploration more effectively than current RL agents are able to. We emphasize that finding this relationship is important, as one possible explanation for the gap between human and agent exploration is simply the degree of prior knowledge—but human behavior being correlated with the different objectives suggests that humans, unprompted, can explore in ways that optimize these objectives. In some preliminary analyses of verbalizations, we find that goal-based utterances in children are significantly correlated with Empowerment. This suggests that goal-setting may be an important component of exploration, with further verbalization analyses left for future work.

Limitations and Future Work. One limitation of our work is the small sample size, limiting broader conclusions about human exploration in general. We also note that the analyses on verbalizations were exploratory, and need to be confirmed with a larger sample in a preregistered study format. That said, we hope this work inspires interest in the intersection between cognitive science and AI, laying ground for future work that can collect larger datasets in richer and more naturalistic settings.

CHAPTER 6 CONCLUSION

In this thesis, we propose that addressing reward design for increasingly capable reinforcement learning agents will require novel ways of incorporating human input. We begin with the domain of learning reward models from human feedback, where we propose leveraging large-scale pretrained vision-language models to robustify learned reward models and better handle naturalistic vision and language distribution shifts. Next, we propose novel forms of human input for training assistive agents and for guiding exploration. Within the domain of human assistance, we propose using human empowerment as an input to agent reward functions, enabling us to circumvent some challenges of existing goal inference based approaches for assistance. Within the domain of intrinsic reward design for exploration, we first propose using human priors and general world knowledge as input for rewards that elicit more human-aligned exploration in open-ended domains. Next, we explicitly study the exploratory behaviors of human adults and children in a unified environment with unsupervised reinforcement learning agents in order to identify differences in exploration objectives between humans and current agents.

We study a range of different rewards and behaviors in this work—task-specific language-conditioned agents, real-world robotic manipulation, simulated assistive agents, and agents exploring in open-ended worlds. In studying human-centric reward design in each of these scenarios, many open questions remain. While the discussed approaches push the frontier of incorporating human input in a range of settings, the scope of tasks we hope intelligent machines will eventually be able to accomplish span an even wider scope. If we are to continue incorporating human supervision through reward design, one crucial direction for future work is to better understand which forms of human input are usable, effective, and necessary in different domains. While human feedback over trajectories and demonstrations currently dominate as forms of human input for training intelligent agents, these forms of feedback have their own limitations and may not be generally applicable to all behaviors and types of agents. How can we provide more immediate forms of feedback or corrections, enabling active learning and teaching? What tools can we build that facilitate better communication between humans and agents in interaction? Beyond direct human feedback as input, are there other objectives (as we found with human empowerment) that act as meaningful reward inputs? Alternatively, AI-provided feedback has emerged as a means of providing fuzzily human-aligned feedback without the inefficiencies and cost of human-in-the-loop feedback. How can we ensure that the AI-provided feedback and evaluations align with actual human feedback?

An orthogonal direction to developing new forms of human input is to further our understanding of existing effective forms of input. When learning reward models from human preferences that aim to capture general human intent, open questions remain as to whether the learned reward models are well-aligned with the intended human objectives. While some initial work in this area show reward models for large language models can be well-calibrated (Bai et al., 2022), further work is needed to develop more transparency into such reward models and to better understand their failure modes. Can we formulate methods for correcting and adjusting reward models over time, as objectives change and model failures are detected?

The general question of how we can use human input to guide agent behaviors carries important sociotechnical implications. Ideally, human inputs into reward design should be representative of the objectives of people that are affected by such a system, directly or indirectly. Future work in this direction should build on interdisciplinary insights; drawing on lessons from domains such as HCI, where there exists bodies of prior work on interpreting and making use of human preferences, and psychology, where better understanding human objectives and behaviors can inform reward design for agents. While some of these interdisciplinary works exist, many challenges remain. We hope that the work presented in this thesis paves a way forward for reward design that actively accommodates different forms of human input, allowing us to better elicit desired behaviors in intelligent systems.

BIBLIOGRAPHY

Daniel Aarno, S. Ekvall, and Danica Kragic. In *Adaptive Virtual Fixtures for Machine-Assisted Teleoperation Tasks*, volume 2005, pp. 897– 903, 05 2005. ISBN 0-7803-8914-X. doi: 10.1109/ROBOT.2005.1570231.

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

Josh Abramson, Arun Ahuja, Iain Barr, Arthur Brussee, Federico Carnevale, Mary Cassin, Rachita Chhaparia, Stephen Clark, Bogdan Damoc, Andrew Dudzik, et al. Imitating interactive intelligence. *arXiv preprint arXiv:2012.05672*, 2020.

Josh Abramson, Arun Ahuja, Arthur Brussee, Federico Carnevale, Mary Cassin, Felix Fischer, Petko Georgiev, Alex Goldin, Tim Harley, et al. Creating multimodal interactive agents with imitation and self-supervised learning. *arXiv preprint arXiv:2112.03763*, 2021.

Josh Abramson, Arun Ahuja, Federico Carnevale, Petko Georgiev, Alex Goldin, Alden Hung, Jessica Landon, Jirka Lhotka, Timothy Lillicrap, Alistair Muldal, et al. Improving multimodal interactive agents with reinforcement learning from human feedback. *arXiv preprint arXiv:2211.11602*, 2022a.

Josh Abramson, Arun Ahuja, Federico Carnevale, Petko Georgiev, Alex Goldin, Alden Hung, Jessica Landon, Timothy Lillicrap, Alistair Muldal, Blake Richards, et al. Evaluating multimodal interactive agents. *arXiv preprint arXiv:2205.13274*, 2022b.

Caspar Addyman and Denis Mareschal. Local redundancy governs infants' spontaneous orienting to visual-temporal sequences. *Child development*, 84(4):1137–1144, 2013.

Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Legged locomotion in challenging terrains using egocentric vision. In *Conference on Robot Learning*, pp. 403–415. PMLR, 2023.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as I can, not as I say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

P. Aigner and B. McCarragher. Human integration into robot control utilising potential fields. In *Proceedings of International Conference on Robotics and Automation*, volume 1, pp. 291–296 vol.1, 1997.

Riad Akrour, Marc Schoenauer, and Michèle Sebag. APRIL: Active preference learning-based reinforcement learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2012.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L Littman. Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257*, 2019.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Arthur Aubret, Laetitia Matignon, and Salima Hassas. A survey on intrinsic motivation in reinforcement learning. *arXiv preprint arXiv:1908.06976*, 2019.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Chris Baker, Rebecca Saxe, and Joshua B Tenenbaum. Bayesian models of human action understanding. In *Advances in neural information processing systems*, pp. 99–106, 2006.

Gianluca Baldassarre, Tom Stafford, Marco Mirolli, Peter Redgrave, Richard M Ryan, and Andrew Barto. Intrinsic motivations and open-ended development in animals, humans, and robots: an overview. *Frontiers in psychology*, 5:985, 2014.

Nir Baram, Oron Anschel, Itai Caspi, and Shie Mannor. End-to-end differentiable adversarial imitation learning. In *International Conference on Machine Learning*, 2017.

Adrien Baranes and Pierre-Yves Oudeyer. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73, 2013.

Adrien F Baranes, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. The effects of task difficulty, novelty and the size of the search space on intrinsically motivated exploration. *Frontiers in neuroscience*, 8:317, 2014.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Daniel E Berlyne. Novelty and curiosity as determinants of exploratory behaviour. *British journal of psychology*, 41(1):68, 1950.

Richard Blahut. Computation of channel capacity and rate-distortion functions. *IEEE transactions on Information Theory*, 18(4):460–473, 1972.

Andreea Bobu, Andrea Bajcsy, Jaime F. Fisac, and Anca D. Dragan. Learning under misspecified objective spaces. *CoRR*, abs/1810.05157, 2018. URL http://arxiv.org/abs/1810.05157.

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

Franziska Brändle, Lena J Stocks, Joshua Tenenbaum, Samuel J Gershman, and Eric Schulz. Intrinsically motivated exploration as empowerment. 2022.

Franziska Brändle, Lena J Stocks, Joshua B Tenenbaum, Samuel J Gershman, and Eric Schulz. Empowerment contributes to exploration behaviour in a creative video game. *Nature Human Behaviour*, pp. 1–9, 2023.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International Conference on Machine Learning*, 2019.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020a.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020b. URL https://arxiv.org/abs/2005.14165.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *Seventh International Conference on Learning Representations*, pp. 1–17, 2019.

Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyu Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. In *Robotics: Science and Systems Conference*, 2020.

Maya Cakmak and Manuel Lopes. Algorithmic and human teaching of sequential decision tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pp. 1536–1542, 2012.

Rui Camacho and Donald Michie. Behavioral cloning a correction. *Ai Magazine*, 16(2): 92–92, 1995.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Harris Chan, Yuhuai Wu, Jamie Kiros, Sanja Fidler, and Jimmy Ba. Actrce: Augmenting experience via teacher's advice for multi-goal reinforcement learning. *arXiv preprint arXiv:1902.04546*, 2019.

Annie S Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021a.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021b.

Kristy Choi, Chris Cundy, Sanjari Srivastava, and Stefano Ermon. LMPriors: Pre-trained language models as task-specific priors. *arXiv preprint arXiv:2210.12530*, 2022.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. Gep-pg: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *International conference on machine learning*, pp. 1039–1048. PMLR, 2018.

Cédric Colas, Tristan Karch, Nicolas Lair, Jean-Michel Dussoux, Clément Moulin-Frier, Peter Dominey, and Pierre-Yves Oudeyer. Language as a cognitive tool to imagine goals in curiosity driven exploration. *Advances in Neural Information Processing Systems*, 33: 3761–3774, 2020.

Cédric Colas, Tristan Karch, Olivier Sigaud, and Pierre-Yves Oudeyer. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022.

C. Cook, N. D. Goodman, and L. E. Schulz. Where science starts: Spontaneous experiments in preschoolers' exploratory play. *Cognition*, 2011.

J. W. Crandall and M. A. Goodrich. Characterizing efficiency of human robot interaction: a case study of shared-control teleoperation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pp. 1290–1295 vol.2, 2002.

Yuchen Cui, Scott Niekum, Abhinav Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics and Control Conference*, pp. 893–905. PMLR, 2022.

Wenliang Dai, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. Enabling multimodal generation on clip via vision-language knowledge distillation. *arXiv preprint arXiv:2203.06386*, 2022.

Ildefons Magrans de Abril and Ryota Kanai. A unified strategy for implementing curiosity and empowerment driven reinforcement learning. *CoRR*, abs/1806.06505, 2018. URL http://arxiv.org/abs/1806.06505.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Daniel Dewey. Reinforcement learning and the reward engineering principle. In *2014 AAAI Spring Symposium Series*, 2014.

Anca D Dragan and Siddhartha S Srinivasa. A policy-blending formalism for shared control. *Int. J. Rob. Res.*, 32(7):790–805, June 2013. ISSN 0278-3649. doi: 10.1177/0278364913490324. URL https://doi.org/10.1177/0278364913490324.

Yuqing Du, Stas Tiomkin, Emre Kiciman, Daniel Polani, Pieter Abbeel, and Anca Dragan. Ave: Assistance via empowerment. *Advances in Neural Information Processing Systems*, 33:4560–4571, 2020.

Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. *Conference on Lifelong Learning Agents*, 2, 2023a.

Yuqing Du, Eliza Kosoy, Li Dayan, Maria Rufova, Pieter Abbeel, and Alison Gopnik. What can ai learn from exploration? intrinsically-motivated humans and agents in open-world exploration. *Intrinsically Motivated Open-ended Learning Workshop at NeurIPS*, 2023b.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. *International Conference on Machine Learning*, 2023c.

Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In *Advances in neural information processing systems*, pp. 1087–1098, 2017.

Rachit Dubey, Pulkit Agrawal, Deepak Pathak, Thomas L Griffiths, and Alexei A Efros. Investigating human priors for playing video games. *arXiv preprint arXiv:1802.10217*, 2018.

Frédéric Dupuis, Wei Yu, and Frans MJ Willems. Blahut-arimoto algorithms for computing channel capacity and rate-distortion with side information. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, pp. 179. IEEE, 2004.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

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

Jaime F. Fisac, Chang Liu, Jessica B. Hamrick, S. Shankar Sastry, J. Karl Hedrick, Thomas L. Griffiths, and Anca D. Dragan. Generating plans that predict themselves, 2018.

Jaime F Fisac, Monica A Gates, Jessica B Hamrick, Chang Liu, Dylan Hadfield-Menell, Malayandi Palaniappan, Dhruv Malik, S Shankar Sastry, Thomas L Griffiths, and Anca D Dragan. Pragmatic-pedagogic value alignment. In *Robotics Research*, pp. 49–57. Springer, 2020.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *International Conference for Learning Representations*, 2018.

Johannes Fürnkranz, Eyke Hüllermeier, Weiwei Cheng, and Sang-Hyeun Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine learning*, 89:123–156, 2012.

Samuel J Gershman. Deconstructing the human algorithms for exploration. *Cognition*, 173: 34–42, 2018.

Jennifer C Gidley Larson and Yana Suchy. The contribution of verbalization to action. *Psychological Research*, 79:590–608, 2015.

Amelia Glaese, Nat McAleese, Maja Trebacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Soňa Mokrá, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William Isaac, John Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks, and Geoffrey Irving. Improving alignment of dialogue agents via targeted human judgements. arXiv:2209.14375, 2022.

Deepak Gopinath, Siddarth Jain, and Brenna D Argall. Human-in-the-loop optimization of shared autonomy in assistive robotics. *IEEE Robotics and Automation Letters*, 2(1): 247–254, 2016.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Giovanni Granato, Anna M Borghi, and Gianluca Baldassarre. A computational model of language functions in flexible goal-directed behaviour. *Scientific reports*, 10(1):21623, 2020.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18995–19012, 2022.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Christian Guckelsberger, Christoph Salge, and Simon Colton. Intrinsically motivated general companion npcs via coupled empowerment maximisation. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. IEEE, 2016.

Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Danijar Hafner. Benchmarking the spectrum of agent capabilities. 2022.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pp. 2681–2691. PMLR, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Felix Hill, Andrew Lampinen, Rosalia Schneider, Stephen Clark, Matthew Botvinick, James L McClelland, and Adam Santoro. Environmental drivers of systematicity and generalization in a situated agent. *arXiv preprint arXiv:1910.00571*, 2019.

Felix Hill, Sona Mokra, Nathaniel Wong, and Tim Harley. Human instruction-following with deep reinforcement learning via transfer-learning from text. *arXiv preprint arXiv:2005.09382*, 2020.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

Edward S Hu, Richard Chang, Oleh Rybkin, and Dinesh Jayaraman. Planning goals for exploration. *arXiv preprint arXiv:2303.13002*, 2023.

Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling up vision-language pre-training for image captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17980–17989, 2022.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pp. 9118–9147. PMLR, 2022a.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *Conference on Robot Learning*, 2022b.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Shervin Javdani, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization. *Robotics science and systems: online proceedings*, 2015, 2015.

Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research*, 37(7):717–742, 2018.

Hong Jun Jeon, Smitha Milli, and Anca Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. *Advances in Neural Information Processing Systems*, 33:4415–4426, 2020.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Perekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, 2021.

Tobias Jung, Daniel Polani, and Peter Stone. Empowerment for continuous agent—environment systems. *Adaptive Behavior*, 19(1):16–39, 2011.

Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. Housekeep: Tidying virtual households using commonsense reasoning. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 355–373, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19842-7.

Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. All else being equal be empowered. In *European Conference on Artificial Life*, pp. 744–753. Springer, 2005a.

Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. Empowerment: A universal agent-centric measure of control. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pp. 128–135. IEEE, 2005b.

W Bradley Knox and Peter Stone. Tamer: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE international conference on development and learning*, pp. 292–297. IEEE, 2008.

Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding language models to images for multimodal generation. *arXiv preprint arXiv:2301.13823*, 2023.

Yu Kong and Yun Fu. Human action recognition and prediction: A survey. *International Journal of Computer Vision*, 130(5):1366–1401, 2022.

Eliza Kosoy, Jasmine Collins, David M Chan, Sandy Huang, Deepak Pathak, Pulkit Agrawal, John Canny, Alison Gopnik, and Jessica B Hamrick. Exploring exploration: Comparing children with rl agents in unified environments. *arXiv preprint arXiv:2005.02880*, 2020.

Eliza Kosoy, Adrian Liu, Jasmine L Collins, David Chan, Jessica B Hamrick, Nan Rosemary Ke, Sandy Huang, Bryanna Kaufmann, John Canny, and Alison Gopnik. Learning causal overhypotheses through exploration in children and computational models. In *Conference on Causal Learning and Reasoning*, pp. 390–406. PMLR, 2022.

Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=10uNUgI5Kl.

Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 2022.

Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *International Conference on Machine Learning*, 2021.

Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*, 2023.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

Scott Weng Fai. Lee. Exploring seven-to eight-year-olds' use of self-talk strategies. *Early Child Development and Care*, 2011.

C. H. Legare. Exploring explanation: Explaining inconsistent evidence informs exploratory, hypothesis-testing behavior in young children. *Child development*, 2012.

Joel Lehman, Kenneth O Stanley, et al. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pp. 329–336, 2008.

Joel Lehman, Jeff Clune, Dusan Misevic, Christoph Adami, Lee Altenberg, Julie Beaulieu, Peter J. Bentley, Samuel Bernard, Guillaume Beslon, David M. Bryson, Nick Cheney, Patryk Chrabaszcz, Antoine Cully, Stephane Doncieux, Fred C. Dyer, Kai Olav Ellefsen, Robert Feldt, Stephan Fischer, Stephanie Forrest, Antoine Fŕenoy, Christian Gagńe, Leni Le Goff, Laura M. Grabowski, Babak Hodjat, Frank Hutter, Laurent Keller, Carole Knibbe, Peter Krcah, Richard E. Lenski, Hod Lipson, Robert MacCurdy, Carlos Maestre, Risto Miikkulainen, Sara Mitri, David E. Moriarty, Jean-Baptiste Mouret, Anh Nguyen, Charles Ofria, Marc Parizeau, David Parsons, Robert T. Pennock, William F. Punch, Thomas S. Ray, Marc Schoenauer, Eric Schulte, Karl Sims, Kenneth O. Stanley, François Taddei, Danesh Tarapore, Simon Thibault, Richard Watson, Westley Weimer, and Jason Yosinski. The Surprising Creativity of Digital Evolution: A Collection of Anecdotes from the Evolutionary Computation and Artificial Life Research Communities. *Artificial Life*, 26(2):274–306, 05 2020. ISSN 1064-5462. doi: 10.1162/artl_a_00319. URL https://doi.org/10.1162/artl_a_00319.

Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: A research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

Yunzhu Li, Jiaming Song, and Stefano Ermon. InfoGAIL: Interpretable imitation learning from visual demonstrations. In *Advances in Neural Information Processing Systems*, 2017.

Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Cam Linke, Nadia M Ady, Martha White, Thomas Degris, and Adam White. Adapting behavior via intrinsic reward: A survey and empirical study. *Journal of Artificial Intelligence Research*, 69:1287–1332, 2020.

Emily G Liquin, Frederick Callaway, and Tania Lombrozo. Developmental change in what elicits curiosity. In *Proceedings of the annual meeting of the cognitive science society*, volume 43, 2021.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34, 2021.

Manuel Lopes, Francisco Melo, and Luis Montesano. Active learning for reward estimation in inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 31–46. Springer, 2009.

Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. *arXiv preprint arXiv:1906.03926*, 2019.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv:2002.06353*, 2020.

Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *Robotics: Science and Systems*, 2021. URL https://arxiv.org/abs/2005.07648.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *International Conference on Machine Learning*, pp. 2285–2294. PMLR, 2017.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. Count-based exploration with the successor representation. *arXiv preprint arXiv:1807.11622*, 2018.

Owen Macindoe, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Pomcop: Belief space planning for sidekicks in cooperative games. 2012. URL https://www.aaai.org/ocs/index.php/AIIDE/AIIDE12/paper/view/5461.

Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022.

Jim Mainprice and Dmitry Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 299–306. IEEE, 2013.

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

P. Marayong, A. Bettini, and A. Okamura. Effect of virtual fixture compliance on human-machine cooperative manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pp. 1089–1095 vol.2, 2002.

Brendon Matusch, Jimmy Ba, and Danijar Hafner. Evaluating agents without rewards. *arXiv preprint arXiv:2012.11538*, 2020.

Björn Meder, Charley M Wu, Eric Schulz, and Azzurra Ruggeri. Development of directed and random exploration in children. *Developmental science*, 24(4):e13095, 2021.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingam, Geoffrey Irving, and Nat McAleese. Teaching language models to support answers with verified quotes. arXiv:2203.11147, 2022.

Josh Merel, Yuval Tassa, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. Learning human behaviors from motion capture by adversarial imitation. arXiv:1707.02201, 2017.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *arXiv preprint arXiv:2111.01243*, 2021.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 2125–2133, 2015.

Ron Mokady, Amir Hertz, and Amit H. Bermano. Clipcap: Clip prefix for image captioning, 2021a. URL https://arxiv.org/abs/2111.09734.

Ron Mokady, Amir Hertz, and Amit H Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021b.

Jesse Mu, Victor Zhong, Roberta Raileanu, Minqi Jiang, Noah Goodman, Tim Rocktäschel, and Edward Grefenstette. Improving intrinsic exploration with language abstractions. *arXiv preprint arXiv:2202.08938*, 2022.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback. arXiv:2112.09332, 2022.

Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 663–670, 2000.

S. Nikolaidis and J. Shah. Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 33–40, 2013.

Georg Ostrovski, Marc G Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2721–2730. JMLR. org, 2017.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2007.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. pp. 5062–5071, 2019.

Stefania Pellegrinelli, Henny Admoni, Shervin Javdani, and Siddhartha Srinivasa. Human-robot shared workspace collaboration via hindsight optimization. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 831–838. IEEE, 2016.

M. Pelz. The elaboration of exploratory play. *Phil. Trans. R. Soc*, 2020.

J. Piaget. Children's philosophies. a handbook of child psychology. *A handbook of child psychology*, 1933.

Silviu Pitis, Harris Chan, Stephen Zhao, Bradly Stadie, and Jimmy Ba. Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International Conference on Machine Learning*, pp. 7750–7761. PMLR, 2020.

Francesco Poli, Marlene Meyer, Rogier B Mars, and Sabine Hunnius. Contributions of expected learning progress and perceptual novelty to curiosity-driven exploration. *Cognition*, 225:105119, 2022.

Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep rl: A short survey. *arXiv preprint arXiv:2003.04664*, 2020.

C. Pérez-D'Arpino and J. A. Shah. Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6175–6182, 2015.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

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

Steve Rathje, Dan-Mircea Mirea, Ilia Sucholutsky, Raja Marjieh, Claire Robertson, and Jay J Van Bavel. Gpt is an effective tool for multilingual psychological text analysis. 2023.

Siddharth Reddy, Anca D Dragan, and Sergey Levine. Shared autonomy via deep reinforcement learning. *arXiv preprint arXiv:1802.01744*, 2018a.

Siddharth Reddy, Anca D. Dragan, and Sergey Levine. Where do you think you're going?: Inferring beliefs about dynamics from behavior. *CoRR*, abs/1805.08010, 2018b. URL http://arxiv.org/abs/1805.08010.

Siddharth Reddy, Anca Dragan, Sergey Levine, Shane Legg, and Jan Leike. Learning human objectives by evaluating hypothetical behavior. In *International Conference on Machine Learning*, pp. 8020–8029. PMLR, 2020.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. 11 2019. URL http://arxiv.org/abs/1908.10084.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668. JMLR Workshop and Conference Proceedings, 2010.

Azzurra Ruggeri, Madeline Pelz, Eric Schulz, et al. Toddlers search longer when there is more information to be gained. 2021.

Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 101–103, 1998.

Dorsa Sadigh, Anca D. Dragan, Shankar Sastry, and Sanjit A. Seshia. Active preference-based learning of reward functions. In *Robotics: Science and Systems Conference*, 2017a.

Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. *Active preference-based learning of reward functions.* 2017b.

Christoph Salge and Daniel Polani. Empowerment as replacement for the three laws of robotics. *Frontiers in Robotics and AI*, 4:25, 2017.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Changing the environment based on empowerment as intrinsic motivation. *CoRR*, 2014a.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pp. 67–114. Springer, 2014b.

Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Learning to fly. In *Machine Learning Proceedings 1992*, pp. 385–393. Elsevier, 1992.

Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 44(1.2):206–226, 2000.

Jürgen Schmidhuber. Curious model-building control systems. In *Proc. international joint conference on neural networks*, pp. 1458–1463, 1991.

Marc Schoenauer, Riad Akrour, Michele Sebag, and Jean-Christophe Souplet. Programming by feedback. In *International Conference on Machine Learning*, 2014.

L.E. Schulz. The origins of inquiry: Inductive inference and exploration in early childhood. *Trends in Cognitive Sciences*, 2012.

L.E. Schulz and E. B. Bonawitz. Serious fun: Preschoolers play more when evidence is confounded. *Developmental Psychology,*, 2007.

Dale H Schunk. Verbalization and children's self-regulated learning. *Contemporary Educational Psychology*, 11(4):347–369, 1986.

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

Daniel Shapiro and Ross Shachter. User-agent value alignment. In *Proc. of The 18th Nat. Conf. on Artif. Intell. AAAI*, 2002.

Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. Skill induction and planning with latent language. *arXiv preprint arXiv:2110.01517*, 2021.

Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. In *Robotics: Science and Systems Conference*, 2019.

Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4):301–321, 2003.

Charles D Smock and Bess Gene Holt. Children's reactions to novelty: An experimental study of" curiosity motivation". *Child Development*, pp. 631–642, 1962.

Aleksandar Stanić, Yujin Tang, David Ha, and Jürgen Schmidhuber. Learning to generalize with object-centric agents in the open world survival game crafter. *arXiv preprint arXiv:2208.03374*, 2022.

Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. From show to tell: a survey on deep learning-based image captioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

Fabrizio Taffoni, Eleonora Tamilia, Valentina Focaroli, Domenico Formica, Luca Ricci, Giovanni Di Pino, Gianluca Baldassarre, Marco Mirolli, Eugenio Guglielmelli, and Flavio Keller. Development of goal-directed action selection guided by intrinsic motivations: an experiment with children. *Experimental brain research*, 232:2167–2177, 2014.

Allison C Tam, Neil C Rabinowitz, Andrew K Lampinen, Nicholas A Roy, Stephanie CY Chan, DJ Strouse, Jane X Wang, Andrea Banino, and Felix Hill. Semantic exploration from language abstractions and pretrained representations. *arXiv preprint arXiv:2204.05080*, 2022.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, pp. 2753–2762, 2017.

Alexandr Ten, Pramod Kaushik, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. Humans monitor learning progress in curiosity-driven exploration. *Nature communications*, 12(1): 5972, 2021.

Alexandr Ten, Pierre-Yves Oudeyer, and Clément Moulin-Frier. Curiosity-driven exploration. *The Drive for Knowledge: The Science of Human Information Seeking*, pp. 53, 2022.

Stas Tiomkin, Daniel Polani, and Naftali Tishby. Control capacity of partially observable dynamic systems in continuous time. *arXiv preprint arXiv:1701.04984*, 2017.

Anthony Meng Huat Tiong, Junnan Li, Boyang Li, Silvio Savarese, and Steven CH Hoi. Plug-and-play vqa: Zero-shot vqa by conjoining large pretrained models with zero training. *arXiv preprint arXiv:2210.08773*, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Pedro A Tsividis, Joao Loula, Jake Burga, Nathan Foss, Andres Campero, Thomas Pouncy, Samuel J Gershman, and Joshua B Tenenbaum. Human-level reinforcement learning through theory-based modeling, exploration, and planning. *arXiv preprint arXiv:2107.12544*, 2021.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6, 2020.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Lev S Vygotsky. *Thought and language.* MIT press, 1962.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Ziyu Wang, Josh S Merel, Scott E Reed, Nando de Freitas, Gregory Wayne, and Nicolas Heess. Robust imitation of diverse behaviors. *Advances in Neural Information Processing Systems*, 30, 2017.

H. James Wilson and Paul R. Daugherty. *Human Machine: Reimagining Work in the Age of AI.* Harvard Business Review, 2018a.

H. James Wilson and Paul R. Daugherty. Collaborative intelligence: humans and ai are joining forces. *Harvard Business Review*, Jul 2018b.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5288–5296, 2016.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. Keep calm and explore: Language models for action generation in text-based games. *arXiv preprint arXiv:2010.02903*, 2020.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. In *International Conference on Machine Learning*, pp. 11920–11931. PMLR, 2021.

Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee. A survey of modern deep learning based object detection models. *Digital Signal Processing*, pp. 103514, 2022.

Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu, Kurt Keutzer, Joseph E Gonzalez, and Yuandong Tian. Noveld: A simple yet effective exploration criterion. *Advances in Neural Information Processing Systems*, 34, 2021.

Ruihan Zhao, Stas Tiomkin, and Pieter Abbeel. Learning efficient representation for intrinsic motivations. 2019.

Yuke Zhu, Ziyu Wang, Josh Merel, Andrei A. Rusu, Tom Erez, Serkan Cabi, Saran Tunyasuvunakool, János Kramár, Raia Hadsell, Nando de Freitas, and Nicolas Heess. Reinforcement and imitation learning for diverse visuomotor skills. In *Robotics: Science and Systems Conference*, 2018.

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

APPENDIX A

VISION-LANGUAGE MODELS AS SUCCESS DETECTORS

A.1 Simulated household domain

To evaluate agent policies on the standardized set of scenarios (STS), each agent is first given a period of context to replay up to a "continuation point", after which the agent policy is used to complete the trajectory. Each continuation is then evaluated offline by human annotators as either successful or failure, along with the point at which success or failure occurs. These human annotations are then used to rank agent policies, using the proportion of successful annotations they receive. For more details on the evaluation procedure, see Abramson et al. (2022b).

Baseline Evaluation Models

While human evaluations provide the ground truth signal for assessing agent capabilities, the cost of annotations scales directly with the number of evaluations for each new task and agent. Thus, there has been interest in automating the evaluation protocol to enable evaluation to scale over time. Ideally, an automated evaluation model will condition on an episode of agent behavior and the input task utterance, and output a classification whether or not the task is successful.

Currently two baseline evaluation models have been developed for the STS: whole-episode and autoregressive models. In both cases, the reward annotations for a particular episode are aggregated using majority voting.

Whole episode evaluation models.

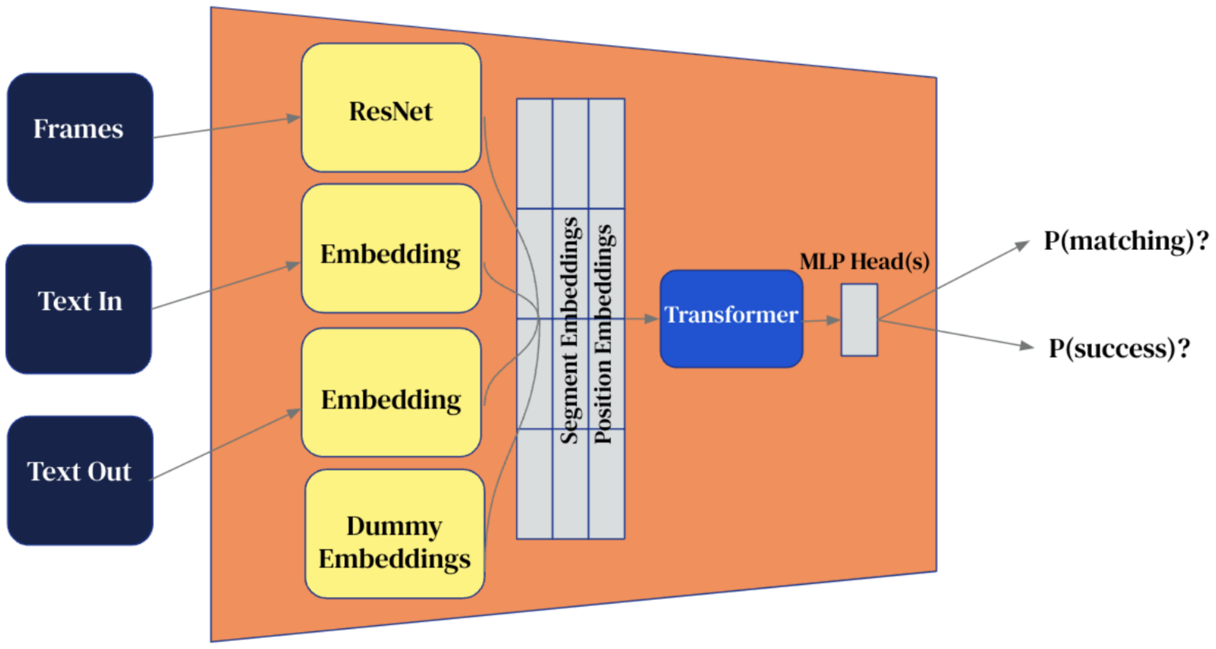


Figure A.1: Whole Episode Bespoke Evaluation Model

For these models, we first preprocess an STS episode by downsampling it to 32 frames and tokenizing the text instruction and agent responses. The images are then embedded with a ResNet-101, the input and output text are embedded, and these embeddings are concatenated together and fed to a transformer with 16 layers and 16 attention heads. The transformer output is fed through two MLP heads: one to predict the likelihood of the episode being successful, P(success), and an auxiliary contrastive loss, P(matching). P(success) is supervised with the aggregated reward annotations, and P(matching) is trained to predict whether an instruction matches the episode or has been shuffled.

Autoregressive evaluation models. The autoregressive evaluation models use the same architecture as the agents, which takes inputs on a per-frame basis, rather than at the episode level. The model embeds the images and language for each frame, passes the embeddings to a multimodal transformer followed by an LSTM, and is asked to predict success or no-success on a per frame basis. The success of an entire episode is then determined by whether or not any single frame was predicted to be successful.

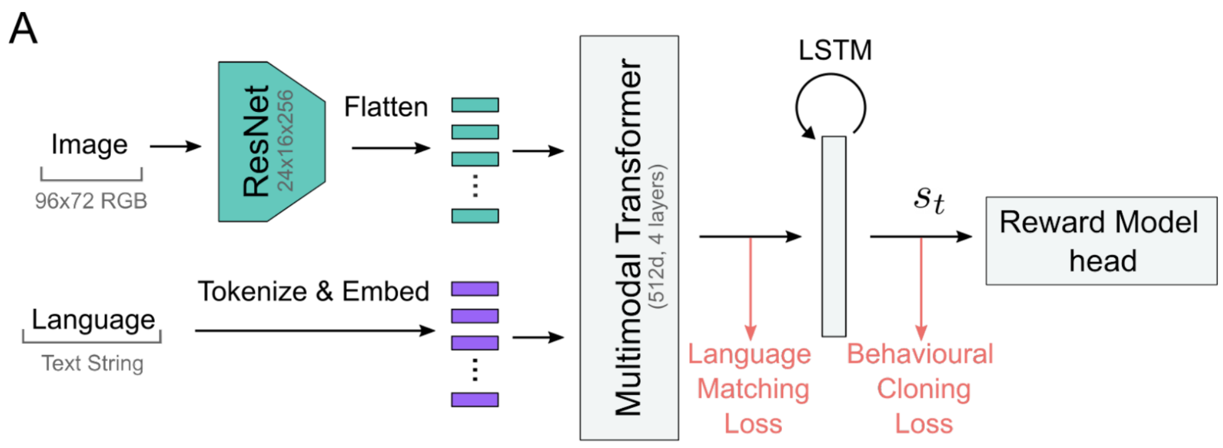


Figure A.2: Autoregressive Bespoke Evaluation Model

Model	Test 1: unseen episodes	Test 2: unseen behavior	Test 3: unseen language
Baseline Whole Episode Model	80.6%	**85.4%**	49.9%
Baseline Autoregressive Model	71.7%	70.4%	(not tested)
Flamingo 3B	50%	50%	50%
FT Flamingo 3B	**83.4%**	85.0%	**59.3%**

Table A.1: Zero-shot episode-level balanced accuracies for IA Playroom STS evaluation models. For reference, human level balanced accuracy is around 88%.

A.2 Robotics domain

Ground truth in robotics domain

Figure A.3 shows how the ground truth success and failure labels are assigned to the full episodes. For an episode to be successful, it must start in a failure state and terminate in a success state.

Data Efficiency in robotics domain

We investigate whether the pretraining used for Flamingo makes it more amenable to accurate success detection in the lower-data regime. For this set of experiments, we train on only 100-200 episodes (100x less than the tens of thousands of episodes used in the above experiments) per task and evaluate on the same in-domain test set. As shown in Table A.2, for five of the six tasks the Flamingo-based model is less affected by the smaller dataset than the ResNet-based model.

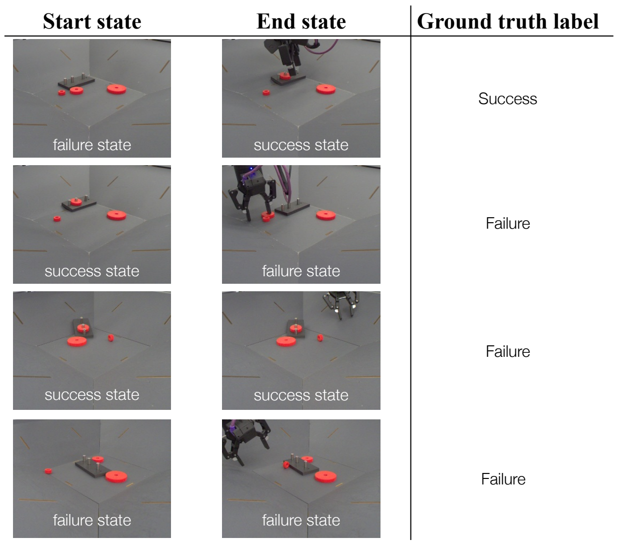


Figure A.3: **Ground truth labels for robotics tasks.** The episode is considered positive only when it starts in a failure state and ends in a success state, all other episodes are considered as negative.

Balanced Accuracy	Insert Small	Insert Medium	Insert Large
bespoke SD	68.7% (-29.2%)	70.2% (-28.3%)	89.7% (-9.4%)
FT Flamingo 3B	**77.6%** (-18.3)	**85.3%** (-9.1%)	**93.2%** (-1.8%)
	Remove Small	Remove Medium	Remove Large
bespoke SD	**86.7%** (-10.6%)	**95.3%** (-3.4%)	**95.7%** (-2.7%)
FT Flamingo 3B	70.5% (-11.6%)	86.7% (+3.3%)	87.1% (-0.0%)

Table A.2: Data Efficiency – train on 100-200 episodes, evaluate on 50-60k episodes.

Policy Evaluation

We further verify that the **FT Flamingo 3B** success detector can be used to train useful policies using reward-filtered behavior cloning (BC). In filtered BC, we first use **FT Flamingo 3B** to classify demonstration episodes as successes or failures for a particular task. Then, we use only the episodes classified as success for BC training. Table A.3 shows the average success rates of the policies evaluated on 20 episodes with manual resets. In manual resets no extra gears are pre-inserted on the pegs for the insert task and only the one relevant gear is pre-inserted for the remove tasks. The success rates vary between 50% and 75%, suggesting that the accuracy of the success detector models is sufficient for some amount of policy training. To compare with the bespoke SD model, we also conduct filtered BC training with the bespoke SD reward model and evaluate an insert large gear policy over 100 episodes with automated resets. In automated resets, policies for different tasks are run in sequence one after another and any number of gears might be already inserted at the start of the episode. In this case, the success rate with FT Flamingo 3B is 30% and with bespoke SD it is 33%. This provides a preliminary proof-of-concept that the difference in reward model accuracy did not lead to a large difference in policy performance. We leave more detailed policy evaluations to future work.

	Small	Medium	Large
Insert	55%	65%	70%
Remove	60%	75%	60%

Table A.3: **Policy success rates.** Policies are trained with filtered behavior cloning where only successful episodes are used for trianing and success is determined by **FT Flamingo 3B**.

APPENDIX

B

GUIDING PRETRAINING IN REINFORCEMENT LEARNING WITH LARGE LANGUAGE MODELS

B.1 Crafter Downstream Training

We finetune on seven downstream Crafter tasks plus the Crafter game reward:

- **Place Crafting Table** - agent must chop a tree and then create a crafting table. This is an easy task most agents will have seen during pretraining.
- **Attack Cow** - agent must chase and attack a cow. This is also an easy task often seen during pretraining in most methods.
- **Make Wood Sword** - agent must chop a tree, use it to make a crafting table, chop a second tree, use the wood at the crafting table to make a wood sword. This task could be achieved during the pretraining env, but many agents rarely or never achieved it because of the sheer number of prerequisites.
- **Mine Stone** - agent must chop a tree, use it to make a crafting table, chop a second tree, use the wood at the crafting table to make a wood pickaxe, seek out stone, and then mine stone. This task is so challenging that we replaced the fully sparse

reward (where all pretraining methods fail) with a semi-sparse reward for achieving each subtask.

- **Deforestation** - agent must chop 4 trees in a row. This task tests whether having goal conditioning improves performance by directing the agent. During pretraining most agents will have chopped a tree, but novelty bias should deter agents from regularly chopping 4 trees in a row.
- **Gardening** Like above, this task tests the value of goal conditioning. The agent must first collect water and then chop the grass. Both skills maybe have been learned during pretraining, but never in sequence.
- **Plant Row** - agent must plant two plants in a row. This task is challenging because even a highly skilled ELLM agent cannot have learned this task 0-shot because the state captioner has no concept of a "row".

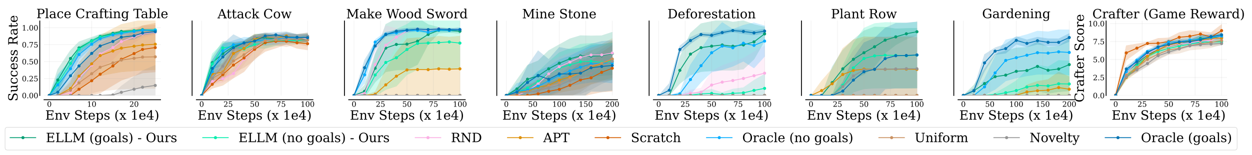


Figure B.1: Goal completion success rate for different tasks in the Crafter environment. RL training uses sparse rewards. Each method trains an agent from scratch while using a pretrained policy for exploration. Each line shows the mean across 5 seeds with shaded stds.

B.2 Crafter Env Modifications

The default Crafter action space contains an all purpose "do" action which takes different actions depending on what object the agent is facing - for instance attacking a skeleton, chopping a tree, or drinking water.

We modify the action space to increase the exploration problem by turning the general 'do' action into more precise combinations of action verbs + noun arguments. Whereas 'do' previously was an all purpose action that could attack a skeleton, chop a tree, or drink water, the agent must now learn to choose between the actions as arbitrary verb + noun combinations, '`attack skeleton`', '`chop tree`', '`drink water`.' The exploration problem becomes more difficult as this larger combinatorial action space is not restricted to admissible actions and the agent could try to `drink skeleton` or `attack water`. Whereas the old action space was 17-dimensional, our new combinatorial one contains 260 possible actions. One way to impose human priors is to design the agent's action space explicitly to disallow invalid combinations (e.g. `'drink' + 'furnace'`). However, manually designing and imposing such

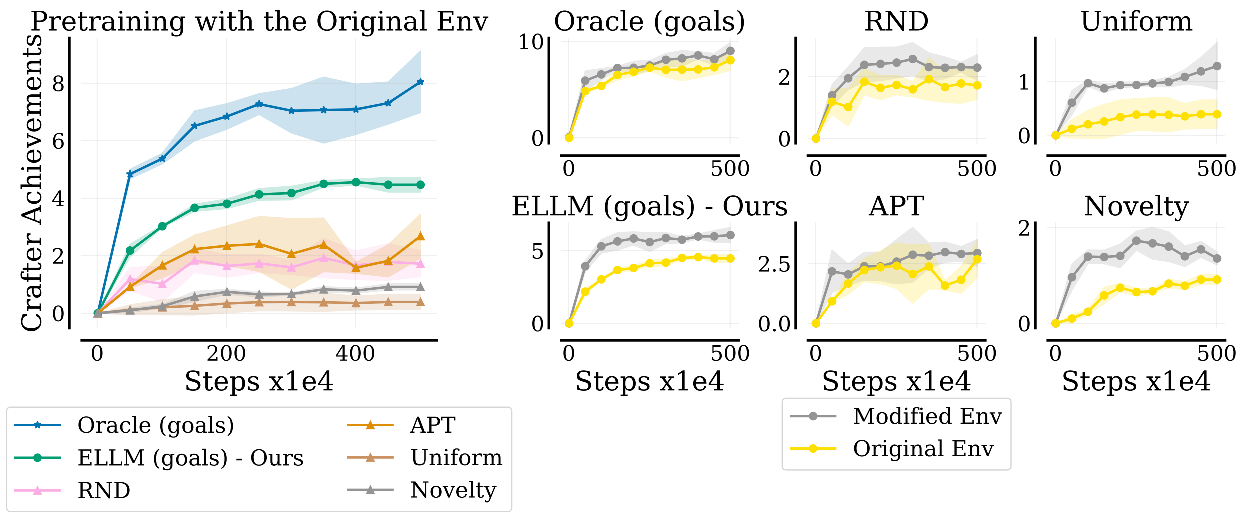


Figure B.2: Training without the environment simplifications described in Section 4.4. Left: pretraining results (comparable to Figure 4.4). Right: original vs modified env performance. Curves average over 3 seeds with std shading. We see minor performance changes across most algorithms but no change in the rank-order of methods.

constraints is also unlikely to be scalable. We hypothesize that our method, guided by common-sense knowledge from LLMs, will focus on learning to use only meaningful action combinations. For the purposes of the Novelty and Uniform baselines, which reward agents for achieving even nonsensical goals, we consider a goal "achieved" if the agent takes an action in front of the appropriate target object (e.g taking "drink furnace" in front of a furnace).

B.3 Crafter Prompt

```
Valid actions: sleep, eat, attack, chop, drink, place, make, mine
   You are a player playing a game. Suggest the best actions the player can take
based on the things you see and the items in your inventory. Only use valid
actions and objects.
   You see plant, tree, and skeleton. You are targeting skeleton. What do you
do?
   - Eat plant
   - Chop tree
   - Attack skeleton
   You see water, grass, cow, and diamond. You are targeting grass. You have in
your inventory plant. What do you do?
   - Drink water
   - Chop grass
   - Attack cow
```

`- Place plant`

In total, the actions present in the prompt make up:

- 6 / 10 (60%) of the good actions the ELLM agent receives.
- 6 / 21 (28.6%) of all rewarded actions the agent receives.
- 7 / 15 (50%) of all good action suggested.
- 7 / 51 (13.7%) of all actions suggested.

In future work, it would be interesting to explore how performance changes with fewer actions included in the prompt. As a preliminary experiment, we have found that pretraining performance is maintained if you provide a prompt with only one example of a list of valid goals. The list only contains two goals. Instead, we use more extensive instructions to tell the agent what good suggestions look like. See the prompt below and pretraining comparison in Figure B.3. This new prompt comes with a decrease in the fraction of "Good" suggestions (shown in Table B.1, showing that suggestion accuracy is not perfectly correlated with success.

New prompt: `Valid actions: sleep, eat, attack, chop, drink, place, make, mine`

```
You are a player playing a Minecraft-like game. Suggest the best actions the player can take according to the following instructions.
1. Make suggestions based on the things you see and the items in your inventory.
2. Each scene is independent. Only make suggestions based on the visible objects, status, and inventory in the current scene.
3. Each suggestion should either be a single valid action, or a phrase consisting of an action and an object. (example: "Eat plant").
4. Do not make suggestions which are not possible or not desirable, such as “Eat skeleton”.
5. Only make suggestions which are reasonable given the current scene (e.g. only “Eat plant” if a plant is visible).
6. You may suggest multiple actions with the same object, but do not duplicate list items.
7. Use your knowledge of Minecraft to make suggestions.
8. Prioritize actions which involve the object you are facing or which the agent hasn’t achieved before.
9. Each scene will include a minimum and maximum number of suggestions. Stick within this range.
New scene: You see plant, cow, and skeleton. You are facing skeleton. What do you do (include 1-2 suggestions)?
- Eat plant
```

```
   - Attack skeleton

   New scene: You see [INSERT CURRENT SCENE DESCRIPTION.] What do you do (include
2-7 suggestions)?
```

	Suggested	Rewarded
Context-Insensitive	21.0%	0.8%
Common-Sense Insensitive	20.5%	54.8%
Good	34.1%	44.4%
Impossible	24.5%	0%

Table B.1: Fractions of suggested and rewarded goals which are good, generated with the modified two-example prompt.

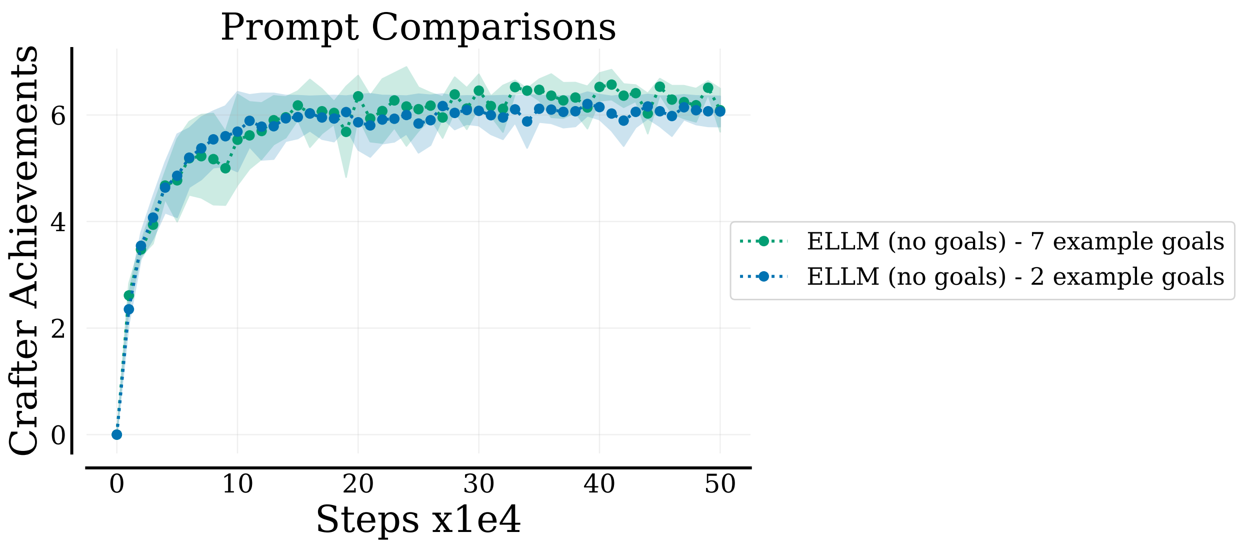


Figure B.3: Comparison between performance of the prompt containing 7 suggested goals (used throughout Chapter 4) and a modified prompt which only includes 2 examples.

B.4 Crafter Action Space

We expand the action space of Crafter to increase exploration difficulty and study if ELLM can learn to avoid nonsensical or infeasible actions. The full action space consists of just verbs (for actions that do not act on anything, such as `sleep`) or verb + noun combinations as follows:

- Verbs: `do nothing` (no noun), `move left` (no noun), `move right` (no noun), `move up` (no noun), `move down` (no noun), `sleep` (no noun), `mine`, `eat`, `attack`, `chop`, `drink`, `place`, `make`
- Nouns: `zombie, skeleton, cow, tree, stone, coal, iron, diamond, water, grass, crafting table, furnace, plant, wood pickaxe, stone pickaxe, iron pickaxe, wood sword, stone sword, iron sword`

For example, an action can be `drink water` or `drink grass`.

B.5 Crafter Pretraining Ablation

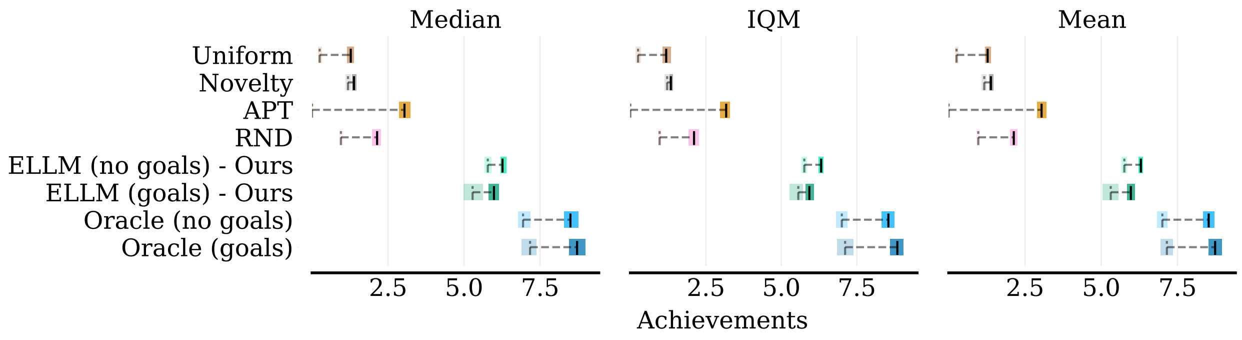


Figure B.4: Number of ground truth achievements unlocked per episode at the end of pretraining. We show the median, interquartile mean (IQM) and mean of the achievements measured in 10 evaluation trials, each averaged over 10 episodes and 5 seeds (50 points) Agarwal et al. (2021). Opaque bars represent variants leveraging textual observations in addition of visual ones and dashed lines represent the gap with vision-only variants (less opaque). We report results for each method described in Table 4.1. Results show that providing textual observations increases performance across all conditions.

B.6 Housekeep Tasks

The original Housekeep benchmark features a large set of different household scenes and episodes with different objects and receptacles possibly instantiated. The ground truth correct object-receptacle placements were determined by crowdsourcing humans. However, since our focus is on RL pretraining, we do not make use of the mapping and planning methods from the original benchmark. To scope the problem for RL, we focus on the first 4 tasks with 5 different misplaced objects per task.

	Misplaced Objects
Task 1	peppermint, lamp, lantern, herring fillets, vase
Task 2	lamp, sparkling water, plant, candle holder, mustard bottle
Task 3	pepsi can pack, electric heater, helmet, golf ball, fruit snack
Task 4	chocolate, ramekin, pan, shredder, knife

Table B.2: Objects per task

B.7 Housekeep Prompt

```
You are a robot in a house. You have the ability to pick up objects and place
them in new locations. For each example, state if the item should be stored
in/on the receptacle.
   Should you store a dirty spoon in/on the chair: No.
   Should you store a mixing bowl in/on the dishwasher: Yes.
   Should you store a clean sock in/on the drawer: Yes.
```

B.8 Algorithmic Details

We make use of DQN Mnih et al. (2013), with double Q-learning Van Hasselt et al. (2016), dueling networks Wang et al. (2016), and multi-step learning Sutton et al. (1998).

Name	Value (Crafter)	Value (Housekeep)
Frame Stack	4	4
γ	.99	.99
Seed Frames	5000	5000
n-step	3	3
batch size	64	256
lr	6.25e-5	1e-4
target update τ	1.0	1.0
ϵ-min	0.01	0.1
update frequency	4	4

Table B.3: DQN Hyperparameters

For both environments, policies take in 84×84 images which are encoded using the standard Nature Atari CNN Mnih et al. (2015). The image is then passed through a linear layer to output a 512 dimensional vector. If the policy is text-conditioned, we compute the language embedding of the state caption using `paraphrase-MiniLM-L3-v2` SBERT model Reimers & Gurevych (2019), and if the policy is goal-conditioned we similarly compute the language embedding of the goals $g_{1:k}$ using `paraphrase-MiniLM-L3-v2`. We encode all goals as a single text sequence as we did not see any improvement from encoding them each separately and summing or concatenating the embeddings. The image and text embeddings are then concatenated together before being passed to the Q-networks. Each of the value and advantage streams of the Q-function are parametrized as 3-layer MLPs, with hidden dimensions of 512 and ReLU nonlinearities.

In the Crafter environment, we swept over the following hyperparameters for the Oracle and Scratch (no-pretraining) conditions: learning rate, exploration decay schedule, and network update frequency. We then applied these hyperparameters to all conditions, after confirming that the hyperparameters were broadly successful in each case.

For Housekeep pretraining, we swept lr $\in [1e-3, 1e-4, 1e-5]$, ϵ-min $\in [0.1, 0.01]$, and batch size $\in [64, 256]$.

B.9 Hard-coded Captioner Details

Crafter The state captioner is based on the template shown in Figure 4.3 (left). This consists of three components: the observation, the items, and the agent status.

- Observation: We take the underlying semantic representation of the current image from the simulator. Essentially this maps each visible grid cell to a text description (e.g. each tree graphic is mapped to "tree"). We then take this set of descriptions (i.e. not accounting for the number of each object) and populate the "observation" cell of the template.
- Items: We convert each of the inventory items to the corresponding text descriptor, and use this set of descriptions to populate the "item" cell of the template.
- Health status: We check if any of the health statuses are below maximum, and if so, convert each to a corresponding language description (e.g. if the hunger status is < 9, we say the agent is "hungry").

The transition captioner uses the action labels. Each action maps to a predefined verb + noun pairing directly (e.g. "eat cow").

Housekeep The state captioner is based on the template shown in Figure 4.3 (right). We use the simulator's semantic sensor to get a list of all visible objects, receptacles, and the currently held object. The transition captioner is also based on the simulator's semantic sensor, which indicates which receptacles the visible objects are currently in.

B.10 Learned Crafter Captioner

The captioner is trained with a slightly modified ClipCap algorithm (Mokady et al., 2021a) on a dataset of trajectories generated by a trained policy using the PPO implementation from Stanić et al. (2022). Visual observations at timestep t and $t+1$ are embedded with a pretrained and frozen CLIP ViT-B-32 model (Radford et al., 2021) and concatenated together with the difference in semantic embeddings between the two corresponding states. Semantic embeddings include the inventory and a multi-hot embedding of the set of objects present in the local view of the agent. This concatenated representation of the transition is then mapped through a learned mapping function to a sequence of 10 tokens. Finally, we use these 10 tokens as a prefix and pursue decoding using a pretrained and frozen GPT-2 to generate the caption (Radford et al., 2019). We train the mapping from transition representation to GPT tokens on a dataset of 847 human labels and 900 synthetic labels obtained by sampling from a set of between 3 and 8 different captions for each each distinct type of transitions. Instead of the programmatic "`chop tree`" and "`attack zombie`," labeled captions involve fully-formed sentences: "`You collected a sapling from the ground`," "`You built a sword out of wood`," or "`You just stared at the sea`." Because of this additional linguistic diversity, we compare captions to goals with a lower cosine similarity threshold of .5.

Imperfect captioners can cause learning issues in two different ways: (1) they can generate wrong captions all together and (2) they can generate a valid caption that still lead to faulty reward computations. If the caption is linguistically too different from the achievement it captions, the similarity-based reward might not be able to pick it up (false negative reward). This same linguistic variability might cause the reward function to detect the achievement of another achievement that was not achieved (false positive reward). Figure B.5 measures all these issues at once. For each row, it answers: what is the probability that the reward function would detect a positive reward for each of the column achievements when the true achievement is the row label? The false negative rate is 11% on average (1 - the diagonal values), with a much higher false negative rate for *chop grass* (100%). Indeed, human caption mentioned the outcome of that action instead of the action itself (*collect sapling*); which the similarity-based reward fails to capture. The false positive rate (all non diagonal values) is significant here: the agent can get rewarded for several achievements it did not unlock. This often occurs when achievements share words (e.g. wood, stone, collect). This indicates a difficulty of the semantic similarity to differentiate between achievements involving these words.

B.11 Crafter LLM Analysis

Table 4.2 shows that the actions agents are rewarded for are dominated by good actions (66.5%) and bad actions (32.4%). This makes sense; impossible actions can never be achieved. Most context-insensitive cannot be achieved (e.g. "drink water" suggested when no water is present). We consider an action a "success" by checking whether the agent attempted a particular action in front of the right object, so the agent occasionally is rewarded when it takes a context-insensitive action in the appropriate physical location but without the necessary prerequisites (e.g. mining stone without a pickaxe).

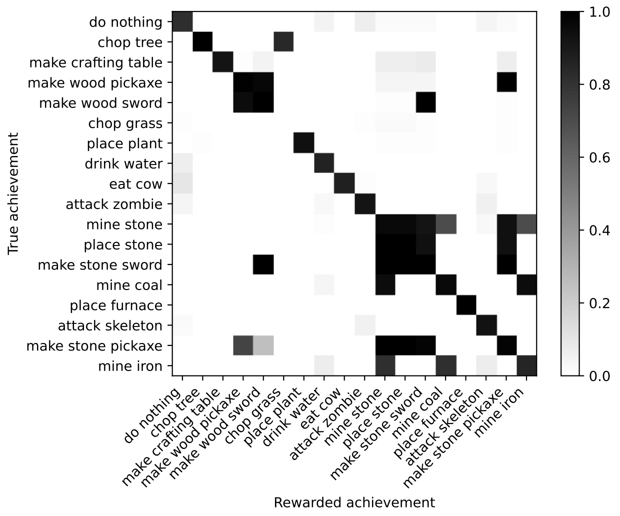


Figure B.5: Reward confusion matrix. Each row gives the probability that any of the column achievement is detected when the row achievement is truly unlocked. For instance, in row 2, when the agent chops a tree, with high probability the agent will be rewarded for the "chop tree" and "chop grass" actions. Tested on trajectories collected from an expert PPO policy, each row estimates probabilities using between 27 and 100 datapoints (27 for *mine iron*, the rarest achievements). Rows do not sum to one, as a given achievement, depending on its particular caption, could potentially trigger several rewards.

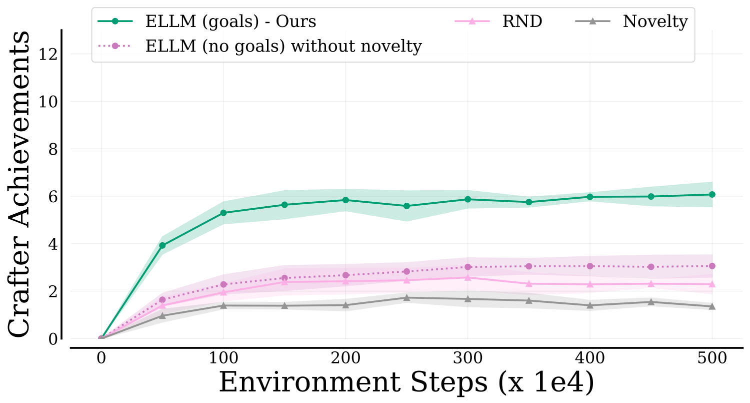
Table B.4 gives examples of LLM suggestions in Crafter.

Suggestion Type	**Examples**
Good	chop tree, attack skeleton, place plant
Context-Insensitive	make crafting table (without wood), mine stone (without a pickaxe or not by stone)
Common-Sense-Insensitive	mine grass, make diamond, attack plant
Impossible	make path, make wood, place lava

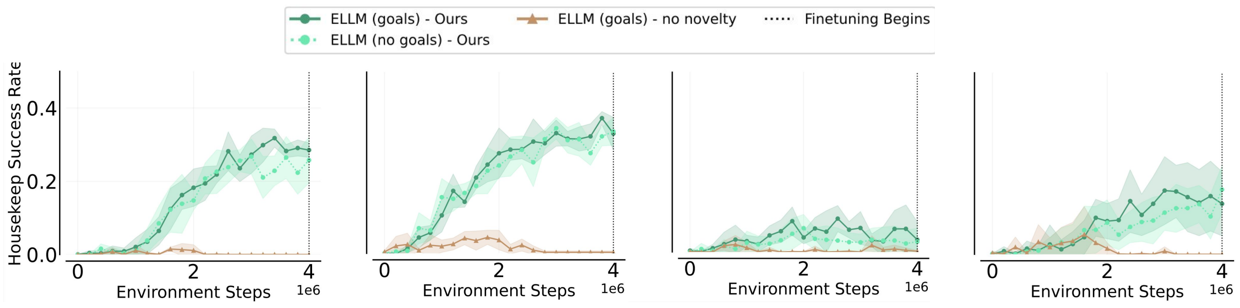
Table B.4: Classification accuracy of LLM for each Housekeep task (left column is true positives, right column is true negatives).

B.12 Novelty Bonus Ablation

We ablate the importance of ELLM's novelty bias in Figure B.6 by allowing the agent to be rewarded repeatedly for achieving the same goal. We see that without the novelty bonus the agent only learns to repeat a small set of easy goals and fails to explore diversely.



(a) Crafter pretraining runs (similar to Figure 4.4), including the "ELLM without novelty" ablation where ELLM's novelty bias is removed.



(b) Housekeep pretraining runs (similar to Figure 4.7a), including the "ELLM without novelty" ablation where ELLM's novelty bias is removed.

Figure B.6: Testing ELLM without a novelty filter—rewarding the agent for accomplishing the same language model goals multiple times.

B.13 Analysis of Downstream Training Approaches

We explored two methods for using exploratory policies: *finetuning*, where the weights of the exploration policy are finetuned and the *guided exploration* method, where a new policy is trained from scratch and the pretrained policy is used for ϵ-greedy exploration.

We found that in Housekeep both methods are effective for ELLM (Figure 4.7a and Figure 4.7b). However, in Crafter we found that the finetuning method performed poorly across all methods (ELLM, baselines, and oracles). Often, we observed that early in finetuning, the agent would unlearn all of its previous useful behaviors, including moving around the environment to interact with objects. We hypothesize that this due to a mismatch in the density and magnitude of rewards between pretraining and finetuning. When the finetuning agent finds it is achieving much lower than the expected return for taking its typical actions, it down-weights the likelihood of taking those actions and unlearns its previous skills. We found that decreasing the learning rate, freezing early layers of the network, manually adjusting finetuning rewards to be at the same scale as pretraining rewards, and decreasing the initial exploration rate partially mitigated this problem. However, these also decrease the sample efficiency and/or performance at convergence of the finetuned policy compared to a training-from-scratch baseline. In Figure B.7), across all methods this method is less reliable than the guided exploration method (Figure 4.5).

These findings are consistent with our Housekeep findings. In that environment, the ELLM pretraining task (achieving object placements suggested by a LLM) and the finetuning task (achieving object placements suggested by humans) are similar enough we only see minor dips in performance when finetuning starts. However, the RND and APT baselines have a greater pretrain-finetune mismatch, and we observe those methods did comparatively better with the guided exploration method.

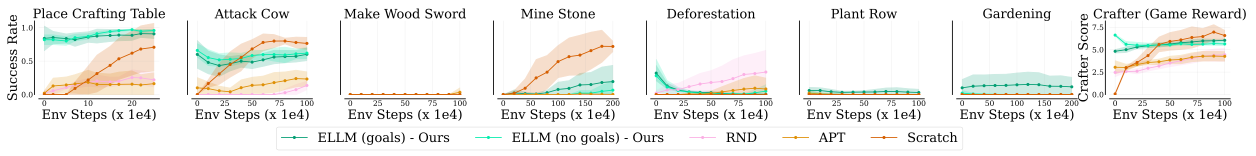


Figure B.7: Success rates across training for each of the seven downstream tasks in the Crafter environment. Each run finetunes the pretrained agent using a lower learning rate than used during pretraining ($2e-5$). Plots show mean $\pm$ std for 5 seeds

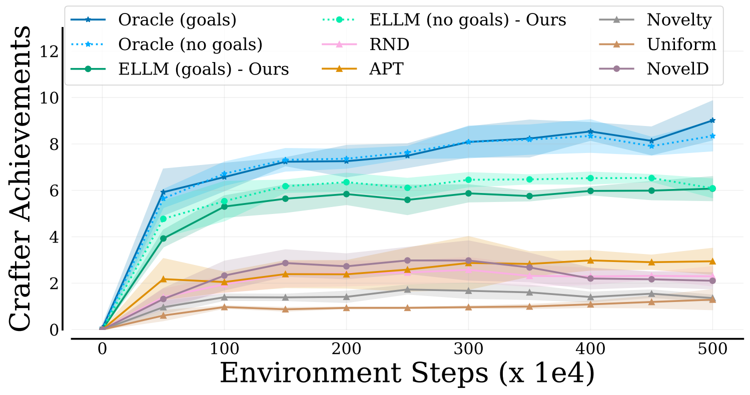
B.14 Additional Baselines

We also include experiments with NovelD Zhang et al. (2021) in Figure B.8, a state-of-the-art exploration method which uses an estimate of state novelty to reward the agent for moving to more novel states. During pretraining, we find it performs similarly to the other prior-free intrinsic motivation methods.

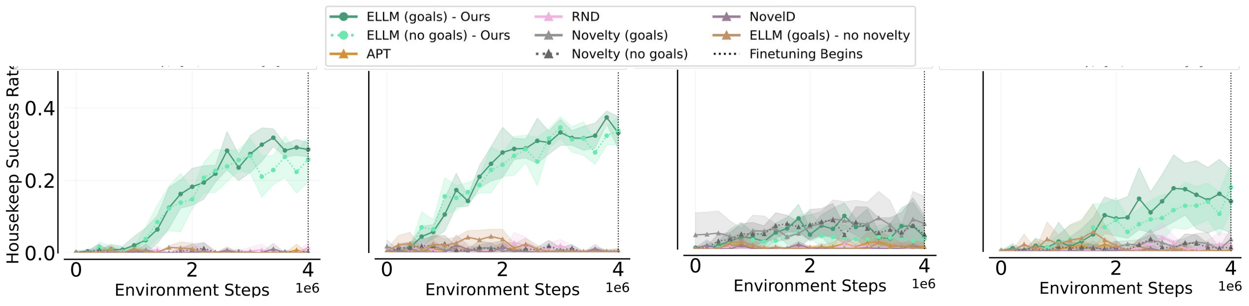
B.15 Code and Compute

All code will be released soon, licensed under the MIT license (with Crafter, Housekeep licensed under their respective licenses).

For LLM access, we use OpenAI's APIs. Initial experiments with the smaller GPT-3 models led to degraded performance, hence choosing Codex and Davinci for our experiments. Codex is free to use and Davinci is priced at \$0.02/1000 tokens. We find caching to be significantly helpful in reducing the number of queries made to the API. Each API query takes .02 seconds, so without caching a single 5-million step training run would spend 27 hours querying the API (and far more once we hit the OpenAI rate limit) and cost thousands of dollars. Since we cache heavily and reuse the cache across runs, by the end of our experimentation, were make almost no API queries per run.



(a) Crafter pretraining curve as in Figure 4.4, including NovelD baseline



(b) Housekeep pretraining curves as in Figure 4.7a, including NovelD baseline

Figure B.8: Additional pretraining curves including NovelD for both Crafter and Housekeep.

We use NVIDIA TITAN Xps and NVIDIA GeForce RTX 2080 Tis, with 2-3 seeds per GPU and running at roughtly 100ksteps/hour. Across all the ablations, this amounts to approximately 100 GPUs for pretraining.

APPENDIX C

INTRINSICALLY-MOTIVATED HUMANS AND AGENTS

C.1 Additional Plots

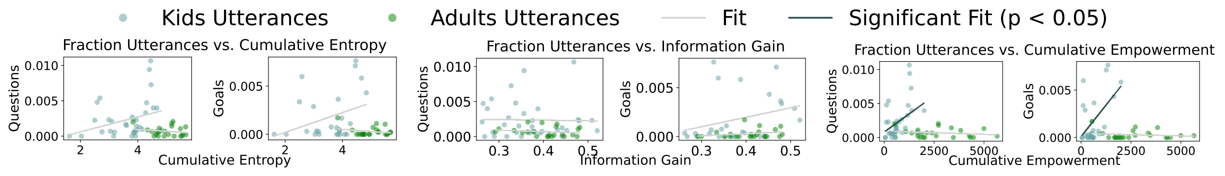


Figure C.1: Fraction of verbalized questions and goals vs. cumulative entropy, information gain, and empowerment for adults and children. We find that the relationship between the fraction of uttered goals and empowerment has the highest correlation and largest significance ($r^2 = 0.28, p = 0.005$ unadjusted).

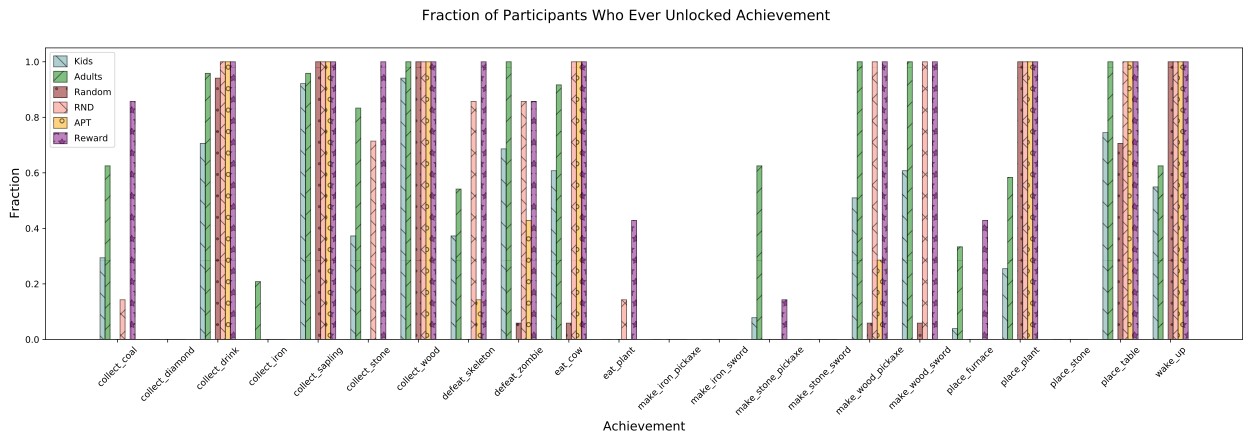


Figure C.2: Bar plot of all possible achievements, showing the fraction of each set of participants or agents that unlocked each achievement at least once.

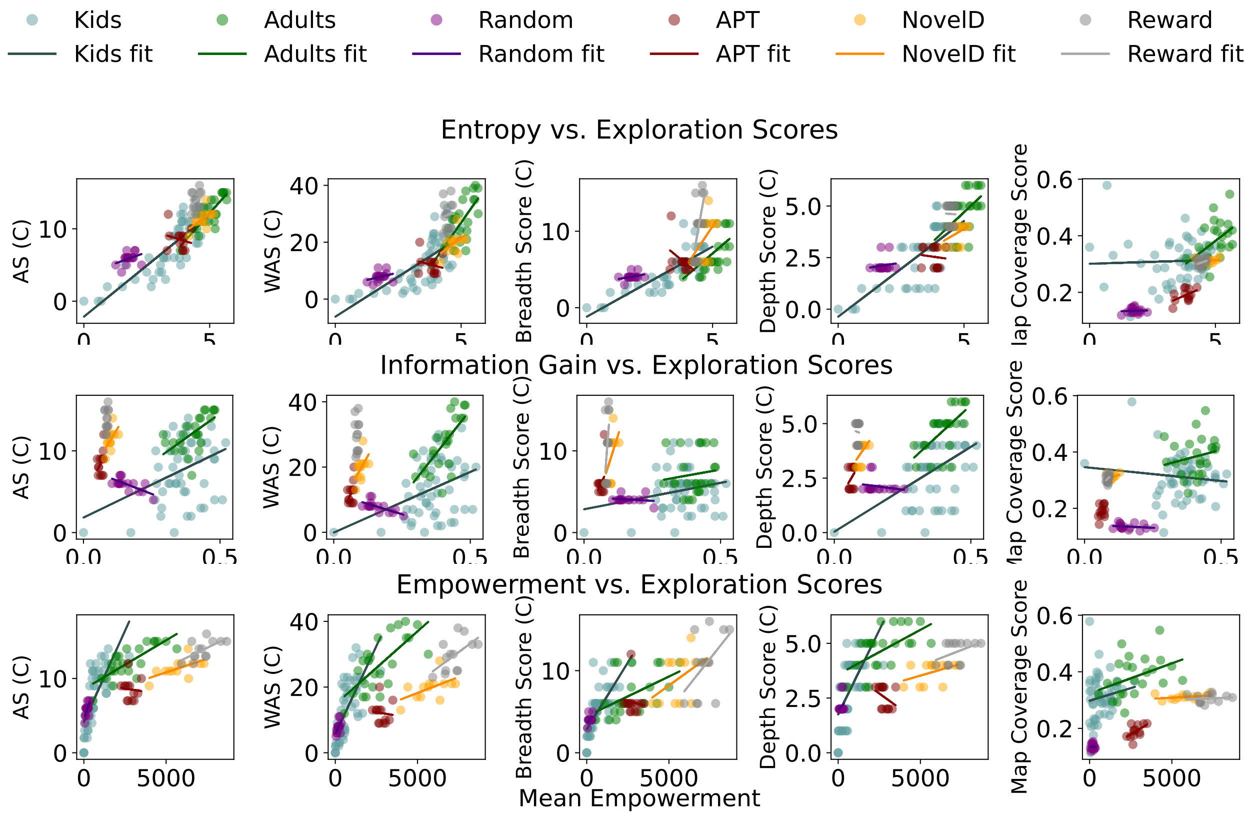


Figure C.3: Information theoretic objectives vs. exploration scores, without filtering for significance.