Bootstrapping Linear Models for Fast Online Adaptation in Human-Agent Collaboration

Benjamin A. Newman
Carnegie Mellon University, Meta
Pittsburgh, Pennsylvania, USA
newmanba@cmu.edu

Chris Paxton
Meta
Pittsburgh, Pennsylvania, USA
cpaxton@meta.com

Kris Kitani
Carnegie Mellon University, Meta
Pittsburgh, Pennsylvania, USA
kkitani@cmu.edu

Henny Admoni
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
henny@cmu.edu

ABSTRACT

Agents that assist people need to have well-initialized policies that can adapt quickly to align with their partners' reward functions. Initializing policies to maximize performance with unknown partners can be achieved by bootstrapping nonlinear models using imitation learning over large, offline datasets. Such policies can require prohibitive computation to fine-tune *in-situ* and therefore may miss critical run-time information about a partner's reward function as expressed through their immediate behavior. In contrast, online logistic regression using low-capacity models performs rapid inference and fine-tuning updates and thus can make effective use of immediate in-task behavior for reward function alignment. However, these low-capacity models cannot be bootstrapped as effectively by offline datasets and thus have poor initializations. We propose BLR-HAC, Bootstrapped Logistic Regression for Human Agent Collaboration, which bootstraps large nonlinear models to learn the parameters of a low-capacity model which then uses online logistic regression for updates during collaboration. We test BLR-HAC in a simulated surface rearrangement task and demonstrate that it achieves higher zero-shot accuracy than shallow methods and takes far less computation to adapt online while still achieving similar performance to fine-tuned, large nonlinear models. For code, please see our project page https://sites.google.com/view/blr-hac

KEYWORDS

Assistive Robotics; Online Assistance; Human-Robot Interaction; Collaborative Assistance

ACM Reference Format:
Benjamin A. Newman, Chris Paxton, Kris Kitani, and Henny Admoni. 2024. Bootstrapping Linear Models for Fast Online Adaptation in Human-Agent Collaboration. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 10 pages.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Agents that collaborate with people to complete a person's preferred goal cannot always know this preference in advance of an interaction. Though people may initially state these preferences, they may drift, sometimes changing entirely, over the course of multiple interaction episodes. While there may be no continued explicit communication between collaborative partners, people's *in-situ* behaviors are goal-driven and thus can reveal the up-to-date preference. This means that updating agent policies based on *in-situ* behaviors is critical for assisting people during collaborations, i.e. ensuring that robot actions are deferential to user goals [21].

Much current research in human-agent collaboration aims to learn zero-shot collaboration policies from offline datasets that are either collected from human-human demonstrations [8] or generated synthetically [28]. Instead of using an individual's *in-situ* behavior to update a model online to improve performance with respect to that individual's preference, these approaches train agents offline in collaboration with the population of partner agents represented by the training dataset. They then target good performance in aggregate on task metrics. At test time, these approaches assume the preferences and behavior of a new human collaborator will fall within the distribution of the collaborators represented by the training data. While these approaches have been shown to be effective on task metrics in general collaboration settings, they do not necessarily transfer to the stricter criteria of assistive collaborations where success in a task is dictated by a personal preference and people's goals and behaviors can drift away from the training distribution.

Furthermore, the population of personal preferences is substantial and diverse, making it difficult to ensure sufficient coverage during training time. Collecting large datasets of human-human data is time-consuming and expensive, while collaboration among populations of procedurally generated agents can yield data that do not tightly match the distribution of the human population. Furthermore, as people repeatedly execute a collaborative task, they may develop new preferences that are unlikely to be captured by the distribution of collaboration data represented in offline datasets.

We propose a method that takes advantage of these advancements in zero-shot coordination and applies them to algorithms for fast, online adaptation from *in-situ* behavior. In this way, we hope to achieve both good initial performance when assisting a

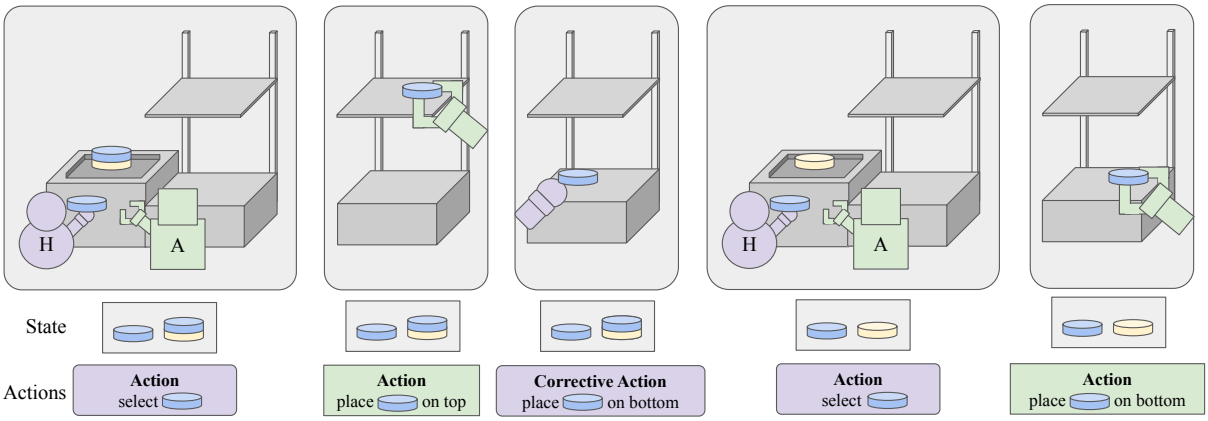

Figure 1: One step of an example surface rearrangement task: cupboard organization. From left to right: a person (H) picks an object to place in the dishwasher; the agent (A) initially places this incorrectly; the person corrects the placement. From this, the agent learns that the user likes to place blue objects on the bottom shelf and can place the next, similar object correctly.

new partner, but also to continue to adapt to their preference over continued exposure.

Deciphering people's exact preferences can be difficult, however, as these preferences are often not explicitly stated and can change over the course of an interaction. Fortunately, *in-situ* behaviors are goal-directed and can implicitly reveal information about a person's current preference or goal, even when it is not expressly communicated. We suggest that agents engaging in assistive collaborations utilize these goal-directed behaviors to infer and act towards a person's current goal, thereby enabling personalized assistance.

To do develop a model that can utilize these goal-directed behaviors for collaborative assistance, we introduce BLR-HAC: Bootstrapped Logistic Regression for Human Agent Collaboration. This model is trained using a two-stage approach: first, we pretrain a transformer [30] to learn to produce the parameters of a shallow, parameterized policy that second, is updated throughout a human-agent collaboration using online logistic regression. To test BLR-HAC, we first introduce a formalization of a specific instance of a rearrangement task, which we call assistive surface rearrangement. We then compare BLR-HAC's performance in a simulated version of this task against two baselines: 1) a traditional transformer trained with behavior cloning and 2) a traditional shallow policy trained with online logistic regression.

Our chosen domain of surface rearrangement models household tasks, like dishwasher loading, which have complex, long-term dependencies determined by a combination of a person's environment and their strongly held preferences. For example, a person may prefer to place large dishes before small ones to maximize capacity. Such high dimensional state and preference spaces lead to an almost infinite number of diverse and equally valid solutions for completing any given household chores. For example, choosing to load a dishwasher based on dish material is just as valid as loading based on dish size; it is a matter of personal preference. Given this diversity, household tasks make especially good testbeds for studying algorithms that require aligning robot policies with people's reward functions, thus mimicking many use cases for assistive robotics.

While some prior approaches to developing autonomous assistants for household tasks rely on people providing full task demonstrations in advance of a collaboration, BLR-HAC aims to operate in real time, utilizing information from each action as it is taken by a person. Furthermore, approaches relying on full task demonstrations can introduce additional burden on a person and be redundant to the goal-directed behavior people exhibit when completing tasks [6]. In contrast, training shallow, low-capacity models with logistic regression through MaxEntIRL to utilize *in-situ* behavior has been shown to effectively and quickly adapt to people's objectives in areas such as robot teleoperation [16] and motion planning [18].

We test BLR-HAC in a simulated version of our surface rearrangement task. We find that BLR-HAC outperforms baseline low-capacity models and large, nonlinear models trained with behavior cloning in zero-shot coordination. We also find that BLR-HAC achieves similar performance but requires a fraction of the compute of a transformer that is fine-tuned online. This finding holds true when considering both preferences that remain the same over time, i.e. are stationary, and those that drift, i.e. are nonstationary. Taken together, these results show how BLR-HAC is able to take advantage of the strengths of both zero-shot and fast online adaptation methods. It does this by pretraining a large, nonlinear model to learn the parameters of a shallow policy that can be updated with online logistic regression. This results in a collaborative agent that is both well-initialized and highly adaptable.

In this paper we make the following contributions:

- a formalization of common household tasks as collaborative IRL tasks, which we call surface rearrangement,
- a novel model, BLR-HAC, that combines the strengths of pretrained large, nonlinear models with low-capacity models trained online via logistic regression for efficient learning in human-robot collaborations, and
- evidence from experiments in simulation that BLR-HAC outperforms its component models.

2 RELATED WORK

First, we present work in state and action-conditioned models that do not explicitly learn about their human partner during task execution. We follow this by reviewing work in adaptive collaborations.

2.1 State and Action-Conditioned Collaboration

Prior approaches to solving long-horizon tasks with complex temporal dependencies and under specified solutions, such as those present in our surface rearrangement domain, can rely on resolving ambiguities through a combination of teleoperation and preprogrammed routines [12], or by suggesting optimal, predetermined solutions [20]. Solutions following the former method can place undue burden on a person to explicitly express their preferences, and render robot action redundant when a demonstration completes. They also require people to continually demonstrate their desired solution as the constraints of the task, such as a person's preference or the environment, vary. Methods following the latter example do not allow for full freedom of expression from the user and assume all users have the same "optimal" solution.

Zero-shot coordination is a recent field of research aiming to develop models that can successfully and immediately interact with novel partners. This can be done by pretraining models in simulation against agents designed to mimic human behavior [8] or over a diverse population of simulated agents [28]. Using these methods, though, can lead to overly specific solutions. Others have used large language models trained with web-scale data to propose task plans that are then executed by robots [2]. These task plans are not adapted to an individual user, whose reward function may or may not fit well within the distribution seen during training. These methods place the burden on the person to either accept a less preferred robot behavior or continue to provide actions that increase the likelihood of the robot behavior exhibiting behavior in line with the person's preferences. In this work, we focus on combining these good initializations with online adaptation.

Often, approaches relying solely on large, pretrained deep neural networks require people to generate explicit descriptions of their preferences which can be decoded by the model into robot action [2]. Actions produced from this process are not guaranteed to align with a person's task objective. While deep networks can potentially be adapted to meet individual preferences through fine-tuning [10, 14], doing so with large models can lead to challenging, unstable learning that results in variable performance [19]. In this work, we focus on developing an algorithm that can quickly adapt to people's naturally expressed, task-oriented behavior.

2.2 Adaptive Collaborations

Using IRL for robot control can be difficult, in part, due to the ambiguity that arises from traditional IRL [1]. Maximum entropy IRL facilitates this by using the principle of maximum entropy to order solutions according to how well they match observed user behavior [32]. This solution has also been used in behavioral science to model people's ability to infer others' goals from their behavior as exhibited during goal-directed plans [6].

These insights have been applied to robot trajectory optimization for shared control. In the difficult task of teleoperating a high-degree of freedom robot arm with a low-degree of freedom input device, such as a joystick, a robot can observe user input commands and infer the user's most likely goal from a set of predetermined goals. The robot then assists the user by moving along a path towards the predicted goal [16]. MaxEntIRL can also be used to interpret less direct forms of user behavior, such as physically pushing a robot out of the way to determine which path the user prefers the robot to take, for example to carry a coffee mug around a laptop computer instead of over it [18], using naturalistic eye gaze in combination with joystick signals to control a robot arm [4, 5, 22], or using corrective actions to learn about features of the environment that relate to a person's preference to increase generalizability and sample efficiency [24]. We are interested in adapting online MaxEntIRL for determining high-level task plans consistent with user preferences in household collaborations from in-task corrective behavior.

IRL has also been applied to learn robot policies in other types of human-robot interactions. For example, to learn people's preferences from observations of independent task demonstrations [31], or by learning assistive social actions for therapy by combining a therapists' expertise with expert demonstrations [3], or for social health, such as a robot receptionist learning to give hygiene advice in a shopping mall [11]. Our formulation learns preferences from *in-situ*, collaborative behavior for collaborative rearrangement tasks.

Finally, another important aspect of maintaining assistive human-agent collaborations is to maintain collaborative fluency [15]. Maintaining principles of collaborative fluency, such as minimizing agent and human idle time, allows human-agent collaborations to function similarly to human-human collaborations, thereby reducing friction on people to interact with autonomous agents. Furthermore, robots assisting people to complete collaborative tasks has been shown to affect a person's ultimate decision [23], making it important to continually monitor and assess people's goals during collaboration. In this work, we will use these ideas as justification for our desire to develop an algorithm that adapts to user preferences in real-time.

3 METHODS

We formalize the task of surface rearrangement, a specific instance of rearrangement problems [7, 29], as a decentralized partially observable Markov decision problem (DEC-POMDP).

3.1 Defining Surface Rearrangement

To study assistive collaborations, we introduce *assistive surface rearrangement*, a collaborative pick and place task where two agents work together to arrange a set of objects O into a set of locations L. In this task, the assistive agent aims to help a person rearrange objects into locations. Importantly, the agent's goal is to achieve the final state that is desired by the person, which is initially unknown to the agent.

A single episode of this task consists of an object repository containing objects $o \in O$. The initial state of the episode is L randomly chosen objects from O, and L vacant locations. Each location has a capacity for a single object. Progress in the task is made by placing objects o into locations $l \in L$. A task is completed when all objects

Algorithm 1 Surface Rearrangement

Require: $\pi_\theta, \pi_{\hat{\theta}}$, env, O, L

$s^0 \leftarrow$ env.reset ()
$\xi \leftarrow [\ s^0\ ]$
while ξ.length $< L$ **do**
  $a_h^t \leftarrow \pi_\theta\left(\cdot|s^{t-1}\right)$
  $a_r^t \leftarrow \pi_{\hat{\theta}}\left(\cdot|a_h^t, s^{t-1}\right)$
  $s^t, a_c^t \leftarrow$env.step$\left(a_h^t, a_r^t, s^{t-1}\right)$
  ξ.append$\left(\left[a_h^t, a_r^t, a_c^t, s^t\right]\right)$
end while

o have been placed into a location l. For simplicity, we assume that $N \leq |L|$ and that placing o in l occurs instantaneously.

Two agents interact in an episode in the following way. The human agent π_θ first picks an object given the current state s^{t-1}. Then, the robot agent $\pi_{\hat{\theta}}$ places this object into a location. The environment then returns the next state s^t and the human corrects the robot's action, returning a_c^t. An episode ξ can be represented as the following tuple: $\left(s^0, a_h^1, a_r^1, a_c^1, s^1, \ldots a_h^L, a_r^L, a_c^L, s^L\right)$.

3.2 Formalizing Surface Rearrangement

Given this description, we can model assistive surface rearrangement as a decentralized partially observable Markov decision problem (DEC-POMDP) which is a tuple of $(S, \Pi, A, T, Z, O, r, \gamma)$. Our objective is to train a policy π_r that solves this DEC-POMDP:

- S is the set of all possible states. As in prior work [18], we assume that a particular state $s \in S$ is a tuple of observable and unobservable features: $s = (x, \{\theta_i\})$. Observable state features are represented as a tuple of all possible locations and all possible objects. Locations are represented by their ID and their current occupancy. Objects are represented by their ID and the location they currently occupy, if any. The unobservable portion of the state, θ_i describes the learnable parameters of the reward function consistent with the human's preference in the task.
- Π is the set of agents. In our initial version of this problem, we assume two agents: a human agent and an assistive agent.
- A_i is the set of actions for a particular agent a_i. We assume that the person both selects objects and corrects object placements, while the robot can only make object placements.
- Z_i is the set of observations used to infer θ. The assistive agent's observation space is the person's action space. In this work we assume that the human does not infer the robot's preference.
- $T(s^{t-1}, \mathbf{a}^{t-1}, s^t)$ denotes the transition dynamics that model the probability of entering a particular state given the current state and both agents' actions. As in prior work [18], changes in T are dictated by θ. We assume this to be constant and deterministic within a single episode.
- $O_i(s^{t+1}, u_i^t, z^{t+1})$, the observation distribution for agent π_i.
- $r_i(s^t, \{a_i\}^t)$ is the reward function for the each agent. We assume an assistive setting where the agent is trying to estimate and maximize the person's reward function. We therefore assume all agents have the same reward function.
- γ, a discounting factor.

Given that we assume two agents and that we are only optimizing one (because the other is assumed to be a person over whose policy we have no control), this problem reduces to a single agent problem, allowing it to be decomposed to a POMDP. Since POMDPs are computationally intractable to solve exactly, we use the QMDP approximation [17]. Prior work in online human robot collaboration [18] has shown how a QMDP can be solved online using online gradient descent, adapted for our purpose in Alg. 3.

4 APPROACH

Our ultimate goal is to learn an assistive policy that collaborates with a person during a surface rearrangement task. Given that we want our policy to be assistive, it should take actions that are aligned with the person's underlying preference for completing the task. We interpret this as a regret minimization problem, where the policy aims to minimize the regret of its actions with respect to the actions that would be exhibited under the person's true preference for completing the task. Importantly, we assume that the policy does not have prior knowledge of this preference and that the person does not immediately or explicitly reveal it. Additionally, we assume that the space of possible preferences the person could hold to be extremely large, making disambiguation from limited interaction with the person difficult.

Under these conditions, we have two main ways to perform regret minimization. First, we can ensure our policy takes good initial actions that are likely to align with the person's preference, often referred to as zero-shot performance. Second, we can adapt the policy online as a history of behavior is accumulated.

Action inference and policy adaptation do not operate within a vacuum, but rather within the course of the interaction. The common metric in human-robot interaction of collaborative fluency [15], for example, is critical to people considering an interaction with a robot to be "good." An important facet of this metric is related to the amount of time the robot sits idle during task execution. This makes frequently updating large models during an interaction challenging, as both action inference and policy updating require large amounts of computation, leading to high robot idle times. We aim to develop a method that can take advantage of the good performance of large nonlinear models while being able to quickly adapt to user preferences, as expressed through their *in-situ* behavior, without causing the robot to idle.

To learn an assistive policy that solves the DEC-POMDP discussed in Sec. 3.2, we first generate a simulated dataset of diverse, high-level preferences (Sec. 4.1.1). Using these preferences, we collect a dataset of collaborative demonstrations in a simulated surface rearrangement task over a range of difficulties (Sec. 4.1.2). We then train our two-stage algorithm by first, learning to mimic the collected expert demonstrations (Sec. 4.2) and second, using the preference representations learned in Sec. 4.3 to perform fast, online adaptation.

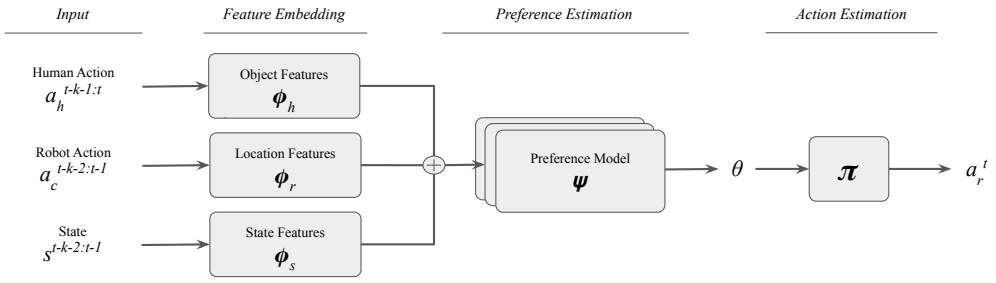

Figure 2: BLR-HAC Overview From left to right, we first embed the input state and actions using ϕ. These are then concatenated and fed into the preference estimator ψ. This learns to output reward parameters, θ which are used to initialize an online learning policy using the policy π, which determines the robot's action a_r.

Algorithm 2 Expert Demonstration Collection

Require: $\Theta, \pi, \text{env}, O, L$
1: $D = [\,]$
2: **for** θ in Θ **do**
3: $\quad \xi \leftarrow \texttt{surfaceRearrangement}(\pi_\theta, \text{env}, \text{O}, \text{L})$
4: $\quad D.\text{append}(\xi)$
5: **end for**

Algorithm 3 Learning Priors for Online Linear Regression

Require: $D, \mathcal{M}, \phi_h, \phi_r, \phi_s$
1: **while** training **do**
2: $\quad$ **for** (s, a_h, k) in D **do**
3: $\quad\quad \hat{\theta} \leftarrow \mathcal{M}\left(\phi_s\left(s\right), \phi_h\left(a_h\right), \phi_r\left(a_c\right)\right)$
4: $\quad\quad a_r \leftarrow \arg\max_{a_r \in A_r} \phi_h(a_h^t) \cdot \hat{\theta} \cdot \phi_r(A_r)$
5: $\quad\quad \texttt{loss} \leftarrow p(a_c) \log q(a_r)$
6: $\quad\quad \text{training} \leftarrow \mathcal{M}.\texttt{update}(\text{loss})$
7: $\quad$ **end for**
8: **end while**

4.1 Datasets

4.1.1 Modeling a Diverse User Population. The two key ideas of our method to develop assistive robots for household collaborations is that the method should be able to both effectively use a large population of preference data to pretrain good initializations and be able to quickly adapt to a particular preference when presented with information about that preference.

To capture these ideas in our experiments, we develop a simulated dataset of preferences. First, we sample a large set of preferences, representing a population, as encoded by θ. We assume preferences from within this population are drawn normally from one of several modes, each of which indicates a subpopulation of similar preferences. We sample three preference datasets: *train*, , and *test*. From each set of preferences, we sample episodes of surface rearrangement episode rollouts, thus creating three datasets: D_{train}, D_{eval}, and D_{test}. D_{train} consists of 1000 simulated preferences, sampled from four modes, with 1000 episodes per preference. D_{eval}, and D_{test} each contain 100 simulated preferences, with 20 episodes per preference.

4.1.2 Environments for Surface Rearrangement. To test the efficacy of our approach at varying difficulties, we develop three environments. Each environment scales problem difficulty by increasing the size of the state space. We have a small environment, with five possible objects and five locations, a medium environment, with ten objects and ten locations, and finally a large environment, with 25 objects and 25 locations.

To collect a demonstration dataset for each environment, we use Alg. 2. Importantly, to collect expert demonstrations, we set $\theta = \hat{\theta}$ and use a linear policy $\pi = \phi_h(a_h) \cdot \theta \cdot \phi_r(A_r)$, where all ϕ are implemented as one-hot embedding layers. For each environment we collect 100 demonstrations from each preference generated in Sec. 4.1.1.

4.2 Learning Preferences in a Diverse User Population

The first step of our proposed algorithm aims to minimize regret by achieving good zero-shot performance. Ultimately, we want to model $p(a_r|s, a_h)$. This problem, however, is ill-posed, as two policies parameterized by different preferences will correctly take two different actions a_r given the same state and human action. To account for this ambiguity, we include a history of k prior state and action pairs taken under the current preference and maximize $p(a_r|s^{t-k-2:t-1}, a_h^{t-k-1:t}, a_c^{t-k-2:t-1})$. For the sake of brevity, we will slightly abuse notation and refer to this distribution as $p(a_r|s, a_h, k)$.

Again, when training assistive agents, achieving low zero-shot performance is not our only objective. We also need an agent that adapts online to incoming user behavior while maintaining collaborative fluency. This means developing a lightweight, low-parameter model capable of performing action inference and policy adaptation in real time.

To do this, instead of learning $p(a_r|s, a_h, k)$ directly, we first learn a latent space that corresponds to the weights of a logistic regression problem. These weights serve as the input to the second step of our algorithm, Sec. 4.3. Thus, we train our model to maximize $M_{\phi,\psi}(s, a_h, k, t) = p(\theta|s, a_h, k, t)$. In this way, we place an inductive bias over the latent space of the model, enticing it to learn a matrix of size $O \times L$, that can be used as the weights of an online logistic

regression problem. We treat this as a classification problem and minimize the cross entropy loss between our model's predictions and the collected expert demonstrations: $L = p(a_c) \cdot \log q(a_r)$ where $q(a_r) = \phi(a_h) \cdot M(s, a_h, k, t) \cdot \phi(A_r)$, as shown in Alg. 3.

4.3 Bootstrapping Shallow Linear Models for Fast, Online Adaptation

The second step of our proposed algorithm aims to minimize regret through online adaptation. Using the output of the model learned in Sec. 4.2, we can employ online logistic regression, which has been shown to work well for teaching human preferences to agents through corrective feedback in robot control tasks. Importantly, since online logistic regression has a very simple update rule to estimate θ that operates over a much smaller number of parameters than a large, nonlinear network, we can adapt this initial estimate of the person's preference *in-situ* without risking large human or robot idle time, thereby maintaining collaborative fluency.

To update our estimate of θ, we use a linear approximation of the QMDP solution to the DEC-POMDP in Section 3.2 and stochastic gradient descent, resulting in the following update rule:

$$\hat{\theta} = \hat{\theta} - \alpha \left(\phi_h(a_h) \cdot \phi_r(a_r) - \phi_h(a_h) \cdot \phi_r(a_c) \right)$$

where α is the learning rate.

5 EXPERIMENTAL DESIGN

To test our algorithm, we design several experiments. First, we validate the need for large, nonlinear models to learn the distribution of preferences embedded in the demonstration dataset, Sec. 5.1. Then we explore how our algorithm fares in its intended use case: fast, online adaptation. We test this in two scenarios. Sec. 5.2.1 analyzes adaptation to a single preference over time, while Sec. 5.2.2 explores how well our algorithm fares when the preference generating the behavior changes without explicit communication to the robot.

5.1 Zero-Shot Coordination

We evaluate our model in each environment over the test set using Alg. 3. While we are in search of an algorithm that performs regret minimization, this metric is relative to a specific preference. To understand model performance in an absolute sense and compare across environments, we report accuracy in terms of the number of correct robot action predictions. This metric is inversely correlated with regret.

We choose our baselines to examine two key questions: 1) are high-capacity, nonlinear models necessary for disambiguation between preferences in a highly diverse preference space, and 2) how does inducing an inductive prior over the latent space affect zero-shot performance?

To answer these questions we introduce baselines across two axes: model complexity and model bias. To determine the effect of high-capacity nonlinear models on zero-shot performance we compare four levels of model complexity in terms of how we implement ψ in Fig. 2:

- **ShallowLinear**. Typical online IRL settings learn a shallow model from scratch using MaxEntIRL. To bootstrap this process, one could perform the same process over the offline dataset, thereby encoding the diverse preference population in the initial model weights. Our intuition, though, is that since demonstrations are drawn from a large, diverse population of preferences, and that the relations between preferences and people are not known a priori, this disambiguation will benefit from a nonlinear function approximator. We expect nonlinear, high-capacity models to outperform this baseline.
- **DeepLinear**. Since the space of preferences is very large, it could simply be that increasing model capacity without introducing nonlinearity may capture the preference distribution. To test this, we introduce DeepLinear, which simply adds additional model parameters in both width and depth. We expect this model to outperform a ShallowLinear model but underperform nonlinear methods.
- **Multi-Layer Perceptron**. To test the importance of modeling the preference distribution with a nonlinear model, we introduce a multi-layer perceptron baseline. We expect this model to outperform both linear methods but underperform attention-based mechanisms.
- **Causal Transformer**. Finally, since we are passing a history of behavior to the model at every time step, we can infer the current preference from this sequence of behaviors. Attention-based mechanisms, specifically causal transformers, have been shown to excel at modeling sequential data. To test this we implement ψ as a transformer, and expect it to outperform all other methods.

The second axis of baselines we develop compares the importance of introducing an inductive bias over the latent space in order to learn θ. We compare an implementation of the above models in which each model minimizes $L = p(a_r) \cdot \log \mathcal{M}(s, a_h, k)$ to our proposed inductive bias, which minimizes $L = p(ar) \cdot \log \phi(a_h) \cdot \mathcal{M}(s, a_h, k) \cdot \phi(A_r)$.

5.1.1 Implementation Details. To implement our models we make the following decisions. We perform a separate parameter sweep for each model and environment for the following parameters and ranges: learning rate $(1e^{-3}, 1e^{-6})$, the dimensionality of hidden layers $(2^5, 2^8)$, and the number of layers in ψ $(3, 5, 7, 10, 12)$. We set the size of the input history to be 50, padding when necessary. For each model, we implement all ϕ as a single, one-hot embedding space of vocabulary size 208, where 0-7 are special characters, 8-107 are location indices, and 108-207 are object indices. To implement ψ, we use PyTorch [25] and base our implementation of a causal transformer on Decision Transformer [9]. We implement π as a simple linear model for inductive bias and as an MLP for no inductive bias. All models are trained using the appropriate training and evaluation sets, which do not overlap with the test set, with 10 epochs of early stopping.

5.2 Test-Time Adaptation

Developing assistive policies is not only about achieving good zero-shot performance, however. The space of actual human preferences is almost boundless and likely impossible to capture in advance of an interaction. Therefore, it is important to develop algorithms that can rapidly align themselves with preferences associated with a person's *in-situ* behavior. We study this in two settings. First,

	Small		Medium		Large	
	No Prior	Prior (ours)	No Prior	Prior (ours)	No Prior	Prior (ours)
ShallowLinear	0.413	0.665	0.215	0.518	0.096	0.289
DeepLinear	0.425	0.680	0.199	0.504	0.101	0.303
MLP	0.605	0.759	0.361	0.653	0.120	0.358
Transformer	0.729	**0.771**	0.603	**0.673**	0.160	**0.412**

Table 1: We compare zero-shot performance on the test set of each environment. We have two axes of comparison: model complexity in the rows, and inductive prior in the columns. Results are reported in terms of accuracy. We can see that the highest capacity, attention based model trained with an inductive prior outperforms all other models in every environment.

we analyze our algorithm's ability to adapt to a stationary preference over the course of multiple episodes. Then, we analyze our algorithm's ability to adapt in scenarios where preferences are nonstationary. Here, we are interested in an algorithm's ability to 1) maintain decent performance in the face of the preference change, and 2) rapidly recover after the change in preference.

5.2.1 Stationary Preferences. To test our algorithm's ability to adapt to stationary preferences, we average the performance of our bootstrapped online IRL algorithm over all preferences in the testing set over 20 episodes in each testing environment.

We compare against a linear model that learns from scratch and a method that optimizes over all transformer parameters between episodes but keeps inference computation constant. We measure computation cost in terms of FLOPS and calculate these values empirically using FVCore. We expect to see that the bootstrapped online IRL algorithm achieves similar performance to the online transformer method but at a fraction of the compute.

5.2.2 Nonstationary Preferences. Similar to the stationary preferences experiment, we run IRL over 20 episodes. In this analysis, however, we switch to a different random objective after 10 episodes. Again, we compare against a linear model learning from scratch and an online transformer implementation. We expect to see that the linear method starts with poor performance but adapts quickly when exposed to incoming behavior. We expect to see that the transformer method starts with good performance and adapts more slowly as behavior data is accumulated. Finally, we expect our method to achieve the benefits of both the linear from scratch and the transformer methods: it should start off with reasonable performance and adapt quickly as data is aggregated.

5.2.3 Implementation Details. For both experiments, we do a hyperparameter sweep over the learning rate in the range $(1e^{-2}, 1e^{-5})$ for the transformer and $(1, 5, 10)$ for the linear models. In both cases, we use the maximum learning rate for all experiments. Additionally, we use stochastic gradient descent for optimization in both cases. To train the transformer method, we perform five steps of gradient descent between each episode.

6 RESULTS

From running the experiments outlined in Sec. 5, we have three main results. First, we find support for our hypothesis that nonlinear, high-capacity models trained with inductive biases can learn a diverse population of user preferences. In Tab. 1, we see the attention-based method trained with an inductive prior outperforms all other methods, achieving 77.1%, 67.3%, and 41.2% accuracy on the small, medium, and large environments, respectively. We see that the difference in performance between models trained with and without the inductive prior increases as the difficulty of the problem increases. Additionally, we see the general trend that higher capacity, nonlinear models outperform lower capacity linear models. These results empirically justify our desire to use a high-capacity nonlinear model to bootstrap a linear model in an online logistic regression problem.

Our second set of results is shown in Fig. 3. Here, we plot the test-time adaptation accuracy for three models: linear (in red), BLR-HAC (in green), and an online transformer (in yellow). From these graphs, we can see support for our hypothesis that bootstrapped, shallow linear models trained with IRL achieve good accuracy with low computation. We can see that BLR-HAC and Transformer both start with higher accuracy than Linear in all cases and that this difference increases as the problem complexity increases. Furthermore, we see how BLR-HAC achieves similar performance over episodes as the transformer method, but at a fraction of the computation. While both methods have similar inference compute, of OxL FLOPS, BLR-HAC uses only $2xOxL$ FLOPS, while the Transformer method uses $\sim 400M$ FLOPS during updates.

Finally, we see in Fig. 4 results from test-time adaption with nonstationary preferences. These results show mixed support for our hypothesis that bootstrapped, shallow linear models trained with IRL recover well from unexpected shifts in user behavior. In each graph, episodes 1-10 show similar results to the previous set of experiments. At episode 10, however, the preference shifts, and all models suffer a drop in performance. Interestingly, in all cases, BLR-HAC suffers the smallest drop in performance. While this is a positive result, we also see that as the environment becomes more complex, BLR-HAC suffers in its adaptation rate from episodes 10-20. While it adapts on par with the linear method (though still achieves higher performance due to its better initial performance) it adapts slower than the transformer-based method. This is likely due to the fact that the transformer is able to make better use of the larger amounts of data that are being aggregated in the large environment.

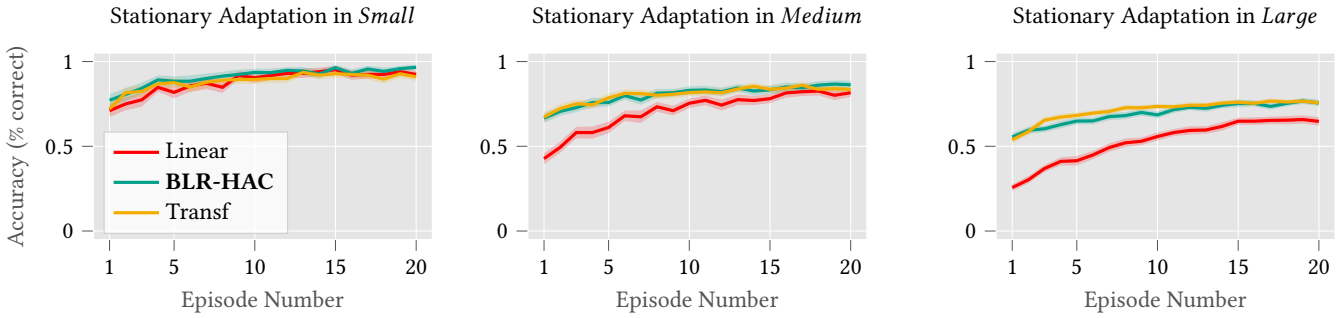

Figure 3: Stationary Test-Time Adaptation. Learning curves for each test environment for each algorithm. We report the average accuracy over each episode. BLR-HAC is able to achieve the low zero-shot performance of the transformer method, and the fast adaptation of the linear method. Additionally, we can see that as the episode length increases, these differences in performance are more notable, with the linear method failing to catch up to the other two methods over the course of 20 episodes.

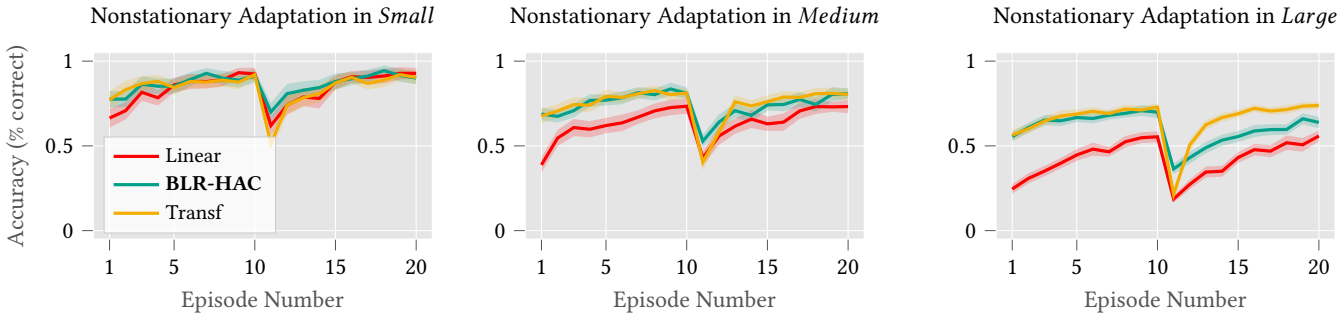

Figure 4: Nonstationary Test-Time Adaptation. Learning curves over each test environment for each algorithm. We report average accuracy over episodes. BLR-HAC is able to perform on par with the transformer method in the small and medium environments and part of the large environment. BLR-HAC outperforms all methods in all environments immediately after the preference switch. In the large environment, though, the transformer recovers more quickly as it has access to more data.

7 DISCUSSION, LIMITATIONS, AND FUTURE WORK

We develop policies for assistive agents that are both well-initialized and highly-adaptable. Through simulated experiments, our method achieves both the good initializations of large, nonlinear models trained with behavior cloning and the fast adaptation to user behavior present in low-capacity models trained with online MaxEntIRL. Importantly, BLR-HAC initializes better than ShallowLinear on test data that is far from the initial distribution, meaning that our approach should ideally allow for faster adaptation to populations for whom it is difficult to collect data for offline pretraining.

Future work should explore applying BLR-HAC to user studies with real people to determine whether the better initializations and faster adaptations of our method hold outside of simulation and are preferred. It is also important to study how the effect of the size of the surface rearrangement problem on these results.

User studies also provide an opportunity to improve our method. Collecting interaction data through interactive simulators, such as AI Habitat [27, 29], deployed on platforms such as Amazon Mechanical Turk [13] or Prolific [26] would allow us to pretrain BLR-HAC with real data.

Finally, our method also assumes a single, synchronized modality of corrective actions: direct state corrections. This makes our learning problem easier by maximizing the correlation between the leader's corrections and their reward function. We would like to extend our approach to account for other modalities of corrections issued asynchronously, such as those expressed in real time through verbal or nonverbal communication.

8 CONCLUSION

In this work, we laid out an argument for why assistive agents should be both well-initialized and highly-adaptable. We introduced a novel formulation of assistive human-agent collaboration as collaborative inverse reinforcement learning and introduced an algorithm BLR-HAC that takes advantage of sophisticated population-level modeling found in deep neural networks with the fast adaptation of shallow, low-capacity inverse reinforcement learning methods. Finally, we verified these claims through simulated experiments.

9 ETHICS STATEMENT

We show we can use offline datasets to bootstrap assistive collaborations by pretraining assistive agents. This method, however, necessitates using specific subpopulations of the larger human population, i.e. those represented by the dataset. This leads to ethical questions such as: Are the preferences present in the dataset representative of the larger population? How does this affect people who hold preferences outside this subpopulation? These questions are especially pertinent in assistive settings, where agents are likely to encounter out-of-distribution phenomena at test-time. It is questions such as these that motivate this work.

We assume a critical part of providing assistance is to reduce unnecessary burden placed on individuals while acting in alignment with their preference. When a person's preferences are well represented by the dataset, pretraining necessarily minimizes a person's burden to bring the agent into alignment with their preference. When a person's preferences are not well represented by the dataset, our method aligns to the person's preference quickly by using their in-situ, goal-directed behavior. Thus, while the model does not have an initial representation of these out-of-domain preferences, it does know how to interpret goal-directed behaviors in order to learn such a representation.

We believe there is ample opportunity for future work to continue to explore solutions to these ethical dilemmas, such as to learn more generalizable features of preferences that allow for better representations of human preferences, or by teaching agents to learn to learn preferences, which would improve an assistive agents ability to adapt to out-of-distribution preferences.

REFERENCES

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

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[3] Antonio Andriella, Carme Torras, Carla Abdelnour, and Guillem Alenyà. 2022. Introducing CARESSER: A framework for in situ learning robot social assistance from expert knowledge and demonstrations. *User Modeling and User-Adapted Interaction* (03 2022). https://doi.org/10.1007/s11257-021-09316-5

[4] Reuben M. Aronson and Henny Admoni. 2022. Gaze Complements Control Input for Goal Prediction During Assisted Teleoperation. *Robotics science and systems* (2022). https://par.nsf.gov/biblio/10327640

[5] Reuben M. Aronson, Thiago Santini, Thomas C. Kübler, Enkelejda Kasneci, Siddhartha Srinivasa, and Henny Admoni. 2018. Eye-Hand Behavior in Human-Robot Shared Manipulation. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction* (Chicago, IL, USA) *(HRI '18)*. Association for Computing Machinery, New York, NY, USA, 4–13. https://doi.org/10.1145/3171221.3171287

[6] Chris L Baker, Joshua B Tenenbaum, and Rebecca R Saxe. 2007. Goal inference as inverse planning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, Vol. 29.

[7] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. 2020. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975* (2020).

[8] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. 2019. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems* 32 (2019).

[9] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision Transformer: Reinforcement Learning via Sequence Modeling. arXiv:2106.01345 [cs.LG]

[10] Sean Chen, Jensen Gao, Siddharth Reddy, Glen Berseth, Anca D. Dragan, and Sergey Levine. 2022. ASHA: Assistive Teleoperation via Human-in-the-Loop Reinforcement Learning. In *2022 International Conference on Robotics and Automation (ICRA)*. 7505–7512. https://doi.org/10.1109/ICRA46639.2022.9812442

[11] Zhichao Chen, Yutaka Nakamura, and Hiroshi Ishiguro. 2022. Android as a Receptionist in a Shopping Mall Using Inverse Reinforcement Learning. *IEEE Robotics and Automation Letters* 7, 3 (2022), 7091–7098. https://doi.org/10.1109/LRA.2022.3180042

[12] Matei Ciocarlie, Kaijen Hsiao, Adam Leeper, and David Gossow. 2012. Mobile manipulation through an assistive home robot. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 5313–5320. https://doi.org/10.1109/IROS.2012.6385907

[13] Kevin Crowston. 2012. Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars. In *Shaping the Future of ICT Research. Methods and Approaches*, Anol Bhattacherjee and Brian Fitzgerald (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 210–221.

[14] Jerry Zhi-Yang He, Zackory Erickson, Daniel S. Brown, Aditi Raghunathan, and Anca Dragan. 2022. Learning Representations that Enable Generalization in Assistive Tasks. In *6th Annual Conference on Robot Learning*. https://openreview.net/forum?id=b88HF4vd_ej

[15] Guy Hoffman. 2019. Evaluating Fluency in Human–Robot Collaboration. *IEEE Transactions on Human-Machine Systems* 49, 3 (2019), 209–218. https://doi.org/10.1109/THMS.2019.2904558

[16] Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S. Srinivasa, and J. Andrew Bagnell. 2018. Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research* 37, 7 (2018), 717–742. https://doi.org/10.1177/0278364918776060 arXiv:https://doi.org/10.1177/0278364918776060

[17] Michael L Littman, Anthony R Cassandra, and Leslie Pack Kaelbling. 1995. Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*. Elsevier, 362–370.

[18] Dylan P Losey, Andrea Bajcsy, Marcia K O'Malley, and Anca D Dragan. 2022. Physical interaction as communication: Learning robot objectives online from human corrections. *The International Journal of Robotics Research* 41, 1 (2022), 20–44.

[19] Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the Stability of Fine-tuning {BERT}: Misconceptions, Explanations, and Strong Baselines. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nzpLWnVAyah

[20] Benjamin Newman, Kevin Carlberg, and Ruta Desai. 2020. Optimal Assistance for Object-Rearrangement Tasks in Augmented Reality. arXiv:2010.07358 [cs.HC]

[21] Benjamin A. Newman, Reuben M. Aronson, Kris Kitani, and Henny Admoni. 2022. Helping People Through Space and Time: Assistance as a Perspective on Human-Robot Interaction. *Frontiers in Robotics and AI* 8 (2022). https://doi.org/10.3389/frobt.2021.720319

[22] Benjamin A. Newman, Reuben M. Aronson, Siddhartha S. Srinivasa, Kris Kitani, and Henny Admoni. 2022. HARMONIC: A multimodal dataset of assistive human–robot collaboration. *The International Journal of Robotics Research* 41, 1 (2022), 3–11. https://doi.org/10.1177/02783649211050677 arXiv:https://doi.org/10.1177/02783649211050677

[23] Benjamin A. Newman, Abhijat Biswas, Sarthak Ahuja, Siddharth Girdhar, Kris K. Kitani, and Henny Admoni. 2020. Examining the Effects of Anticipatory Robot Assistance on Human Decision Making. In *Social Robotics*, Alan R. Wagner, David Feil-Seifer, Kerstin S. Haring, Silvia Rossi, Thomas Williams, Hongsheng He, and Shuzhi Sam Ge (Eds.). Springer International Publishing, Cham, 590–603.

[24] Benjamin A. Newman, Christopher Jason Paxton, Kris Kitani, and Henny Admoni. 2023. Towards Online Adaptation for Autonomous Household Assistants. In *Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction* (Stockholm, Sweden) *(HRI '23)*. Association for Computing Machinery, New York, NY, USA, 506–510. https://doi.org/10.1145/3568294.3580136

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[26] Prolific. 2014 Online. Prolific. https://www.prolific.co

[27] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9339–9347.

[28] DJ Strouse, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett. 2021. Collaborating with humans without human data. *Advances in Neural Information Processing Systems* 34 (2021), 14502–14515.

[29] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John M Turner, Noah D Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel X Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. 2021. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (Eds.). https://openreview.net/forum?id=DPHsCQ8OpA

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[31] Bryce Woodworth, Francesco Ferrari, Teofilo E. Zosa, and Laurel D. Riek. 2018. Preference Learning in Assistive Robotics: Observational Repeated Inverse Reinforcement Learning. In *Proceedings of the 3rd Machine Learning for Healthcare Conference (Proceedings of Machine Learning Research, Vol. 85)*, Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens (Eds.). PMLR, 420–439. https://proceedings.mlr.press/v85/woodworth18a.html

[32] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. 2008. Maximum Entropy Inverse Reinforcement Learning. In *Proc. AAAI*. 1433–1438.