PREFERENCE-CONDITIONED LANGUAGE-GUIDED ABSTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning from demonstrations is a common way for users to teach robots, but it is prone to spurious feature correlations. Recent work constructs *state abstractions*, i.e. visual representations containing task-relevant features, from language as a way to perform more generalizable learning. However, these abstractions also depend on a user's *preference* for what matters in a task, which may be hard to describe or infeasible to exhaustively specify using language alone. How do we construct abstractions to capture these latent preferences? We observe that how humans behave reveals how they see the world. Our key insight is that changes in human behavior inform us that there are differences in preferences for how humans see the world, i.e. their state abstractions. In this work, we propose using language models (LMs) to query for those preferences directly given knowledge that a change in behavior has occurred. In our framework, we use the LM in two ways: first, given a text description of the task and knowledge of behavioral change between states, we query the LM for possible hidden preferences; second, given the most likely preference, we query the LM to construct the state abstraction. In this framework, the LM is also able to *ask the human directly* when uncertain about its own estimate. We demonstrate our framework's ability to construct effective preference-conditioned abstractions in simulated experiments, a user study, as well as on a real Spot robot performing mobile manipulation tasks.

1 INTRODUCTION

In robot learning, we wish to teach robots how to perform tasks that human users want. Learning from demonstrations (LfD) is a common way for doing so, as the user can directly teach the robot desired task behavior. Unfortunately, LfD requires a lot of data and often fails to fully specify all the reasons behind the demonstrated behavior Correa et al. (2022). For example, consider the scenario depicted in figure 1, which shows two demonstrations for the task "throw away the can". Is the user demonstrating moving cans, navigating to a specific goal location, or tossing the can in the trash? Without more data disambiguating the demonstrations, it's difficult for the robot to fully learn what all the features that matter for the task are.

Humans, meanwhile, exhibit extraordinary generalization capabilities in new environments. A key reason why humans can learn so quickly is their ability to construct simplified mental representations over which to plan Ho et al. (2022). Useful abstractions are task-dependent, and prior experience, commonsense reasoning, and direct teaching contribute to humans learning how to best construct these abstractions Ho et al. (2023); Huey et al. (2023). Recent work showed how we can successfully leverage strong priors embedded in LMs to aide in constructing state abstractions for robots Peng et al.. Given a language description of the task, language-guided abstraction (LGA) leverages the strong semantic priors in LMs to model task-relevant features important for decision-making Peng et al..

Unfortunately, LGA is limited when the features that are important to the human are not *fully* specified in language. This presents a challenge in real-world robotics settings where we must adapt to human preferences quickly and efficiently, which can often be expensive or even intractable for preferences inexpressible through natural language. How can we ensure that the robot's state abstractions are strong enough to enable efficient learning Peng et al. (2023); Peng et al. yet flexible enough to learn individual preferences?

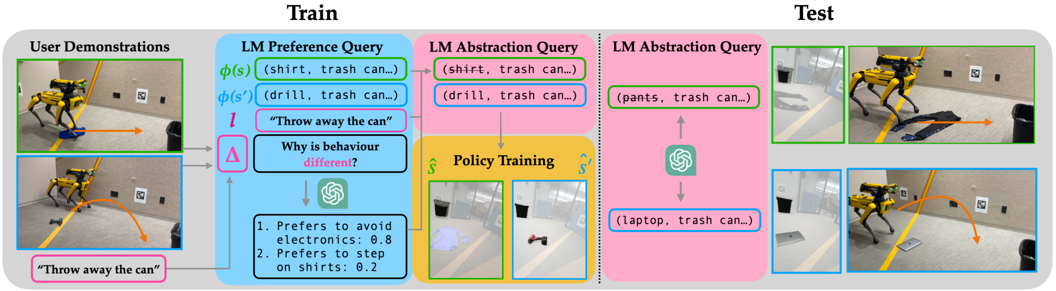
Train
Test
User Demonstrations
"Throw away the can"
LM Preference Query
$\phi(s)$ (shirt, trash can...)
$\phi(s')$ (drill, trash can...)
l "Throw away the can"
Δ
Why is behaviour different?
1. Prefers to avoid electronics: 0.8
2. Prefers to step on shirts: 0.2
LM Abstraction Query
(~~shirt~~, trash can...)
(drill, trash can...)
Policy Training
$\hat{s}$
$\hat{s}'$
LM Abstraction Query
(~~pants~~, trash can...)
(laptop, trash can...)

Figure 1: Preference-Conditioned Language-Guided Abstraction (PLGA). (Left) The robot uses the demonstration pair to identify a behavior change not captured by the language specification. Given this information, we query the LM for potential preferences that could explain this change. Finally, the robot uses its best preference estimate to query the LM for state abstractions and train a policy. (Right) At test time, the robot generalizes to new states and language specifications.

In this work, we propose a framework to use language and behavior to query LMs for their possible abstraction preference. Our observation is how humans behave is indicative of how they see the world, i.e. their state abstraction. If we are able to observe a difference in human behavior, this provides meaningful grounds to infer there are differences in preferences for how their abstractions are constructed. In this work, we introduce Preference-conditioned Language-Guided Abstraction (PLGA), a framework for using this information to infer latent preferences to explain differences in human behavior. In PLGA, we use the LM in two ways: first, given a text description of the task and knowledge of behavior change between states, we query the LM for possible hidden preferences; second, given the most likely preference, we query the LM for the state abstraction. In this framework, the LM is also able to *actively query for human preferences* by asking the human when it is uncertain about its own estimate.

2 Problem Formulation

2.1 Preliminaries

Markov Decision Processes. We model our problem as a Markov Decision Process $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ with states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, transition probability $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, and rewards $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. We define a trajectory τ as a sequence of state-action pairs, $\tau = (s_0, a_0, \cdots, s_T, a_T)$. We wish to learn a policy $\pi_\psi : \mathcal{S} \to \mathcal{A}$, parameterized by ψ, that solves the MDP.

Goal-Conditioned Behavioral Cloning. We consider scenarios where the robot does not know the reward, and instead it learns the policy π_ψ from user demonstrations $\mathcal{D} = \{\tau^i\}_{i=1}^n = \{(s_1^i, a_1^i, ..., s_T^i, a_T^i)\}_{i=1}^n$ and a natural language description $\ell \in \mathcal{L}$ that specifies the goal for each demonstration. Goal-conditioned behavioral cloning (GCBC) Co-Reyes et al. (2018) is a method where the policy can condition on both the current state s and a linguistically-specified goal ℓ to try and imitate human actions. GCBC attempts to learn a policy π that minimizes:

$$\mathcal{L}_{\text{GCBC}} = \mathbb{E}_{(s_t^i, a_t^i, \ell^i) \sim D_{\text{train}}} [||\pi_\psi(s_t^i, \ell^i) - a_t^i||_2^2] \ , \tag{1}$$

However, because at its core the algorithm simply imitates the data it has seen, GCBC alone cannot reliably generalize the policy $\pi_\psi(s_t^i, \ell^i)$ to novel specifications ℓ^i or states s_t^i.

Language-Guided Abstraction. Our work builds on LGA (Language-Guided Abstraction) Peng et al., which proposes using LM priors to build abstract state representations. LGA's key novelty is an abstraction function $\hat{f} : \mathcal{S} \times \mathcal{L} \to \hat{\mathcal{S}}$ that contextualizes the state within the language task specification and produces a task-relevant state abstraction $\hat{s} = \hat{f}(s, \ell)$. This extends GCBC to learning policies $\pi_{\hat{\psi}} : \hat{\mathcal{S}} \to \mathcal{A}$ that operate at the abstraction level:

$$\mathcal{L}_{\text{LGA}} = \mathbb{E}_{(s_t^i, a_t^i, \ell^i) \sim \mathcal{D}} [||\pi_{\hat{\psi}}(\hat{f}(s_t^i, \ell^i)) - a_t^i||_2^2] \ . \tag{2}$$

The key to LGA generalizing beyond specific user commands and demonstrations is the rich language prior that determines which states and specifications should be treated similarly in the context of decision-making (e.g. if the robot has learned to "pick up a cup", it should also know to "pick up something to drink with").

In LGA, the abstraction function $\hat{f}^{\mathrm{LGA}}$ consists of 3 steps:

1. In **textualization**, a state captioner $C : \mathcal{S} \to \mathcal{L}^{\phi}$ converts the raw perceptual state s into a text-based feature set $\phi = C(s)$. This text representation may include common visual attributes of the state like object type and color, which are reasonably accessible via segmentation models today Kirillov et al. (2023).
2. **Feature abstraction** passes ϕ and ℓ to the LM and asks for the features relevant for the task, $\hat{\phi} = \mathrm{LM}_{\mathrm{abs}}(\phi, \ell)$. We denote $\mathrm{LM}_{\mathrm{abs}}$ as queries for the abstraction, e.g. "What features in the scene matter for the task $\langle$throw away the can$\rangle$?".
3. Lastly, LGA **instantiates** $\hat{\phi}$ into an abstracted state $\hat{s} = C^{-1}(\hat{\phi})$. We assume that the captioner from step 1 is invertible and can, thus, instantiate (potentially abstracted) perceptual states from feature sets, i.e. $C^{-1} : \mathcal{L}^{\phi} \to \mathcal{S}$. For instance, in figure 1 the captioner converts states to a feature set of object names, and the inverse captioner takes an LM-obtained feature set and converts it into an abstracted state.

Altogether, the LGA abstraction function can be written as $\hat{f}^{\mathrm{LGA}}(s, \ell) = C^{-1}(\mathrm{LM}_{\mathrm{abs}}(C(s), \ell))$.

2.2 Problem Statement

Unfortunately, LGA is limited when the language utterance does not fully specify the desired behavior. For example, in figure 1, without explicitly mentioning "avoid electronics" in the utterance ℓ, there is no recourse for the model to know that "drill" or "laptop" should be captured by the abstraction, and are thus relevant for robot behavior. Consequently, the LGA function $\hat{f}$ will ignore it, leading to learning an incorrect policy $\pi_{\hat{\psi}}$ downstream. In this paper, we present a method to infer and incorporate such unexpressed preferences.

Formally, we assume the human holds a latent preference $\theta \in \Theta$ over what the abstraction $\hat{s}$ should be, i.e. $\hat{s} = \hat{f}(s, \ell, \theta)$ for $\hat{f} : \mathcal{S} \times \mathcal{L} \times \Theta \to \hat{\mathcal{S}}$. In the example above, the user is a cautious person who prefers to "avoid electronics". The challenge is that the robot does not know θ and must infer it in order to build the abstraction.

We observe that in providing demonstrations to the robot, humans reveal information about what matters to them in their tasks. In other words, demonstrations *implicitly* give evidence for what the latent abstraction preference θ is (Jeon et al., 2020). In this paper, we study how we can use demonstrations $\mathcal{D}$ together with the utterance ℓ to learn *preference-conditioned* language-guided abstractions $\hat{s} = \hat{f}(s, \ell, \theta)$, i.e. abstractions that capture *how the human* represents the task, using information from both their linguistic specification and physical behaviors. We expect these preference-conditioned abstractions will allow flexible adaptation to preferences over tasks.

3 Preference-conditioned Language-Guided Abstraction

We present our method for constructing preference-conditioned language guided abstractions (PLGA). We use an LM to give a common-sense prior over abstraction preferences given a language specification and information about user demonstrations. At a high level, our method consists of two steps: 1) estimating the abstraction preference θ and 2) updating the abstraction function $\hat{f}$ with that θ. Our use of the LM is, thus, two-fold: first, given ℓ and information about demonstrations τ, we query the LM for most likely human preference θ; next, given that preference, we query the LM for the abstraction. This framing also allows us to actively query the human for their preference when the LM is uncertain about its set of hypothesized θs. We present the full PLGA procedure in Alg. 1.

We use GPT4 OpenAI (2023) as our LM to query for human preferences and state abstractions given state, language, and trajectory information. Here, we first focus on LM queries for state abstractions. We discuss the use of LMs for querying for human preferences in section 3.2.

Algorithm 1: PLGA

```
Input: $N$ sampled trajectory pairs $(\tau, \tau') \in \mathcal{D}$, specification $\ell$, captioner $C$, entropy threshold
  $\epsilon$, distance threshold $\kappa$
Init: Abstraction model without preferences $\hat{f}^{\mathrm{LGA}}$
for $i \leftarrow 1$ to $N$ do
    // Language can't explain behavior change
    if $\|\tau - \tau'\|_2^2 > \kappa$ and $\hat{f}^{\mathrm{LGA}}(s, \ell) = \hat{f}^{\mathrm{LGA}}(s', \ell)$ then
        // Find hidden preference as in section 3.2
        $\Theta_{LM}, P(\theta \mid s, s', \ell, \Delta = 1) \sim \mathrm{LM}_{\mathrm{pref}}(C(s), C(s'), \ell, \Delta = 1)$
        // LM is confident about preference
        if $H(P(\theta \mid s, s', \ell, \Delta = 1)) < \epsilon$ then
            $\hat{\theta} \leftarrow \arg\max_{\theta}(P(\theta \mid s, s', \ell, \Delta = 1))$
        else
            $\hat{\theta} \leftarrow$ query H // as in section 3.3
// Create updated abstractions as in section 3.1.
$\hat{f}^{\mathrm{PLGA}}(s, \ell, \hat{\theta}) = C^{-1}(\mathrm{LM}_{\mathrm{abs}}(C(s), \ell, \hat{\theta}))$
$\pi_{\hat{\psi}} \leftarrow \mathcal{L}_{\mathrm{PLGA}}(\hat{f}^{PLGA}(s, \ell, \hat{\theta}))$
```

3.1 LMs as Models of State Abstraction

Moving beyond LGA, we want an abstraction function that is preference-conditioned. Here, we assume we already have an estimate of the human's abstraction preference θ, and we discuss the estimation process later in section 3.2. We can use the same captioner from LGA, but the LM must now be queried with preference information as well. Hence, in our feature abstraction step we pass ϕ, ℓ and a language description of the estimate θ to the LM and query it for the preference-conditioned features that are relevant for the task, i.e. $\hat{\phi} = \mathrm{LM}_{\mathrm{abs}}(\phi, \ell, \theta)$. In the figure 1 example, the abstraction query includes not only the scene and task specification, but also the inferred preference "avoid electronics". Overall, our abstraction function can be written as $\hat{f}^{\mathrm{PLGA}}(s, \ell, \theta) = C^{-1}(\mathrm{LM}_{\mathrm{abs}}(C(s), \ell, \theta))$.

3.2 LMs as Models of Preference

We now discuss how PLGA estimates the human's latent abstraction preference parameter θ. Given s and ℓ, we could query an LM for potential human preferences θ_i corresponding to that state and task specification, i.e. $\theta_i \sim \mathrm{LM}_{\mathrm{pref}}(C(s), \ell)$, but the space of possible preferences may be intractably large. For example, in figure 1 the more objects in the scene, the combinatorially more preferences for caring or not caring about each one of them the LM could find.

We observe that given demonstrations τ, we can derive additional insights about the abstraction preference beyond the language specification: human behavior (i.e. demonstrations) implicitly reveals information about what the human cares about in the world (i.e. the abstraction). If we had a language description of the demonstrations, we could include it in our query to the LM. Unfortunately, behaviors are particularly challenging to caption Rana et al. (2023) and asking the human to narrate every demonstration they give is too burdensome.

Instead of giving the LM a description of the behavior the human demonstrates, we indicate initial scenes where behaviors are *different* in ways that the language utterance does not specify. Given a trajectory pair (τ, τ') corresponding to initial states s and s' and the specification ℓ, we introduce a binary variable $\Delta(s, s', \ell)$ that indicates whether the desired human behaviors in s and s' are different in ways not directly specified by ℓ.

Intuitively, Δ signals that an unknown human preference θ is impacting behavior. If Δ is 0, then behaviors τ and τ' are either the same despite starting in different states or different but in a way conveyed by ℓ. If Δ is 1, then τ and τ' differ beyond the language specification. In the figure 1 example, the user demonstrations differ despite the specification "Throw away the can" not explicitly indicating that they should. Our hypothesis is that the context change between s and s' can reveal the human preference θ that resulted in the behavior change in τ and τ'.

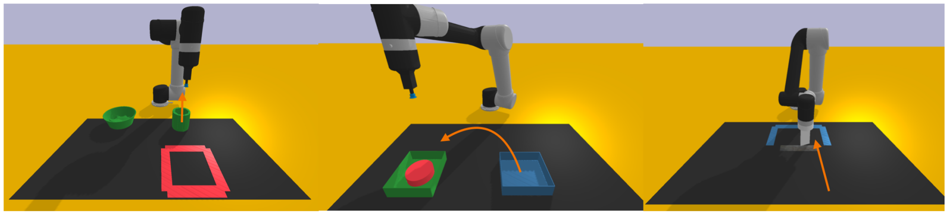

Figure 2: We evaluate on three tabletop manipulation tasks: pick, place, and sweep.

When $\Delta = 1$, we query the LM for potential human preferences θ_i that explain the change in behavior for the two scenes, i.e. $\theta_i \sim \text{LM}_{\text{pref}}(C(s), C(s'), \ell, \Delta = 1)$. We denote the set of "sampled" preferences $\Theta_{LM} = \{\theta_i\}_{i=0}^{k}$. The PLGA estimate $\hat{\theta}$ should be the most likely in Θ_{LM}. To generate that, we ask the LM to also assign a normalized probability for how likely it is that θ_i is the hidden preference, resulting in a distribution $P(\theta \mid s, s', \ell, \Delta = 1)$ with support on Θ_{LM}. In passive PLGA, we simply select $\hat{\theta}$ to be the preference in Θ_{LM} with the highest probability.

3.3 Querying Preferences with Language

If the LM model is uncertain about which of the hypothesised preferences θ_i is the most likely explanation for the behavior change, PLGA enters an active learning stage where it queries the user directly for the cause of behavior change. This scenario may apply when the human preference cannot be captured by a general LM prior, e.g. "pick up my favorite object" where the robot is uncertain about what the user's "favorite object" may be. In such cases, we expect none of the probability values to stand out. In other words, the entropy of the LM-queried distribution $P(\theta_i \mid s, s', \ell, \Delta = 1)$ is high. We propose that when this is the case, the robot should query the human directly for a language description of their preference $\hat{\theta}$.

3.4 Policy Learning with PLGA

Once the robot has a preference estimate $\hat{\theta}$, our abstraction function is simply $\hat{f}^{\text{PLGA}}(s, \ell, \hat{\theta}) = C^{-1}(\text{LM}_{\text{abs}}(C(s), \ell, \hat{\theta}))$. We can use this to train our policies $\pi_{\hat{\psi}}$, similar to LGA:

$$\mathcal{L}_{\text{PLGA}} = \mathbb{E}_{(s_t^i, a_t^i, \ell^i) \sim \mathcal{D}}[||\pi_{\hat{\psi}}(\hat{f}^{\text{PLGA}}(s_t^i, \ell^i, \hat{\theta})) - a_t^i||_2^2] \ . \tag{3}$$

with differences from LGA highlighted in red.

4 Investigating Passive PLGA as a Prior for General Preferences

We begin our evaluation by testing PLGA's ability to leverage the semantic priors in LMs to generate human preferences that explain changes in behavior. We first conduct simulated experiments to demonstrate passive PLGA in cases where the LM should be able to confidently identify the human preference. For cases where the LM may be unsure about the hidden preference, we will test the active component of PLGA with real users in section 5. Here, we present results for nine different scenarios across three different tasks.

Environment. We generate a series of robotic control manipulation tasks from the simulated environment VIMA Jiang et al. (2022) (figure 2). VIMA is a vision-based simulator where a UR5 arm is tasked with manipulating a specified target object into a desired goal configuration. Observations are top-down RGB images of the manipulation space and actions are continuous pick and place poses each consisting of a 2D coordinate and a rotation expressed as a quaternion. We modify the VIMA feature space to contain up to 48 potential objects (e.g. bowl) and 17 colors/textures (e.g. glass) (see list in Appendix).

Following standard LGA, we implement a captioner module that extracts the feature set ϕ from the original RGB observation. This captioner uses a ground truth segmentation mask and labels it with text descriptions of objects and their properties (texture, object ID, etc.). Our PLGA algorithm constructs the task-relevant feature subset $\hat{\phi}$ using GPT4 OpenAI (2023) as the LM. We query the

LM by providing a language utterance, description of the scene, estimated preference, and a target feature to evaluate (the full prompt can be seen in the Appendix). The LM returns a binary response indicating whether that feature should be included in the preference-conditioned abstraction $\hat{\phi}$. Finally, we convert $\hat{\phi}$ to $\hat{s}$, a binary pixel mask over the robot observation where all identified task-relevant features are represented as ones (otherwise zero).

Our algorithm requires finding trajectory pairs in the demonstration set where the language specification can't explain the behavior change. To generate them, we randomly sample trajectory pairs from $\mathcal{D}$, compute their Euclidean distance and their corresponding preference-free abstractions $\hat{s} = \hat{f}^{\mathrm{LGA}}(s, \ell)$ and $\hat{s}' = \hat{f}^{\mathrm{LGA}}(s', \ell)$, and check for pairs that are more than κ distance apart while mapping to the same abstraction $\hat{s} = \hat{s}'$. In our experiments, we found $\kappa > 0.2$ was a good metric for differentiating trajectories.

Tasks. We investigate three tasks that arise in the context of personal robotics: 1) *pick up the [target]*, 2) *place grasped object on the [target]*, and 3) *sweep object 1 into object 2* [while avoiding potential obstacle] (brackets denote objects the user may have a preference distribution over). For each task, we test three possible (unspecified) preferences that may impact the desired abstraction.

1. For *pick*: 1) a (*ripe*) tomato, 2) a (*container*) to put food in, 3) a (*dry*) cereal bowl (parentheses denote the hidden preference). The robot must determine the correct target object given behavioral context (e.g. *is a green tomato a target pick object?*).
2. For *place*: 4) a (*non-electronic*) object such as pan, 5) a (*stable*) surface such as coaster, 6) a (*desired content*) container such as recycling or trash. For these tasks, the robot must determine the correct target for the held object to be placed on/in (e.g. *is a laptop a valid place location?*);
3. For *sweep*: 7) a *hot* object such as stove, 8) a *sweepable* object such as rug, and 9) a *sharp* object such as knife. For these tasks, the robot must assess whether objects are potential obstacles to be avoided before executing a sweep motion (e.g. *is a red stove an object to be avoided?)*.

Preferences are instantiated as a distribution over possible object types and colors in the task. These may include preferred pick objects (e.g. red or dark red tomatoes for *ripe*, but not green), preferred place objects (e.g. container or bin for *non-electronic* but not laptop), and avoid obstacles (e.g. a knife for *sharp* but not flower). These are selected to illustrate diversity in preferences that PLGA can infer using strong semantic priors. For each task, the language specification is given without mentioning the preference (e.g. "Sweep the food into the sink"). PLGA therefore must infer the hidden preference from behavioral context (e.g. *avoid hot objects*). Here we assume there is a generic *but unspecified* preference for each scenario (e.g. users generally prefer to avoid hot objects).

For each preference-task pair, we generate a dataset $\mathcal{D}$ via an oracle demonstrator consisting of 20 demonstrations: 10 expressing behavior when the tested feature is present in the scene and 10 when the tested feature is not (e.g. 10 trajectories of the sweeping food around the stove if the stove is hot, and 10 where sweeping food across the stove otherwise). Target objects are randomly sampled from one of three discretized locations. To create additional complexity, we additionally sample a distractor object that is unrelated to the preference (e.g. a *flower* along with a *stove*).

Manipulated Variables. We test PLGA's ability to construct good preference-conditioned abstractions for each task using the LM priors alone. We compare the resulting policies trained via PLGA against two baselines: GCBC (learned directly from raw states and the specified language utterance as per Eq. equation 1) and LGA (learned from state abstractions constructed via querying ϕ against the language utterance alone as per Eq. equation 2). We implement GCBC as a goal-conditioned CNN architecture that independently processes language input ℓ into an embedding via BERT Devlin et al. (2018) and the RGB image into an embedding via a CNN, then concatenates the outputs for action prediction via a MLP. We implement LGA and PLGA as the same CNN architecture processing the state abstraction only.

Dependent Measures. We evaluate success as an executed action via a pick/place/sweep of the target object within radius α of the goal. For these tasks, we constructed a ground truth test distribution reflective of the human preference. We manipulate the training and test distribution such that only a subset of the true preference distribution (e.g. red tomatoes) are seen at training. We

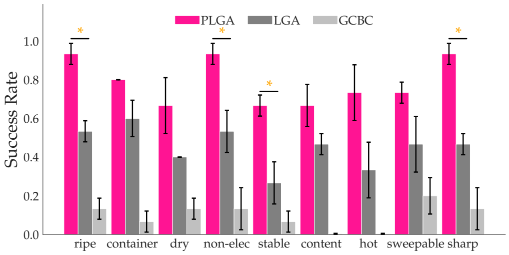

Figure 3: Policy success rate (with standard error) on simulated experiments. PLGA outperforms both LGA and GCBC on task performance, showing better preference-conditioned abstraction construction on downstream task learning.

evaluate performance via success rate of the learned policies on 5 states sampled from the full test distribution during test.

Hypothesis H1: Using information about changes in behavior (PLGA) leads to state abstractions better able to generalize policy learning to preference-conditioned test tasks than abstractions based on language alone (LGA) or no abstractions (GCBC).

Analysis. To compare performance, we show in figure 3 the policy success rates on test scenes for each task. These results illustrate a trend for better PLGA performance compared to baselines (significant for four tasks with a one-sided t-tests $p < 0.05$).

Overall, this illustrates a trend for better PLGA performance than baselines, supporting the notion that preference-conditioned abstractions enable better generalizable learning. However, one-sided t-tests confirm statistical significance only for four of the tasks. The other tasks display high variance at times in the result, indicating that more trials may be necessary to determine significance. Nevertheless, the qualitative trend softly supports **H1**.

5 Investigating Active PLGA for User-Specific Preferences

In section 4 we tested PLGA's ability to construct *generic* preference-conditioned abstractions using only the LM's priors. We now test its ability to construct abstractions when the preferences are more personalized, meaning the LM may not be entirely sure about its sampled hypotheses Θ_{LM}. We study the active component of PLGA with a user study to test the ability of PLGA to recognize uncertainty about a preference estimation, causing it to query for the human preference and update its abstraction model accordingly.

5.1 Experimental Setup

Tasks. We now construct a new scenario for each task.

1. For *pick*: a (*favorite food*);
2. For *place*: a (*preferred dish*) for setting food on;
3. For *sweep*: a (*specific type of object*) to avoid.

These tasks are now intended to study 1) PLGA's ability to measure uncertainty over the LM's inferred preferences, or in other words, know when it does not know the answer and ask for help and 2) PLGA's ability to update its abstraction generation process given a user-specified preference in natural language.

Sanity Check. Before investigating PLGA's active querying of human preferences, we first conduct a sanity check to ensure the measured entropy of the resulting LM preference probability is indeed higher (indicating uncertainty) for these tasks vis-a-vis those less ambiguously defined in the previous section. We perform the same LM query as before (e.g. where the LM is tasked with inferring a hidden *favorite food* from Δ). As shown in figure 4, we do see larger uncertainty for tasks containing more ambiguous preferences, and a one sided t-test ($t(10) = -3.49, p = 0.005$) confirms this observation. Based on these results, we found $\epsilon = 1.0$ to be a good entropy threshold.

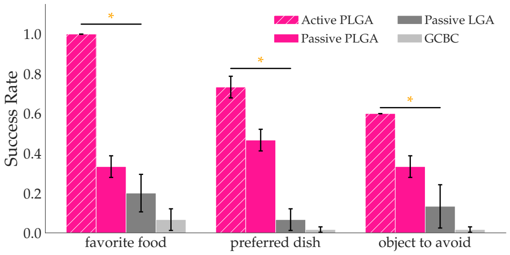

Figure 5: Learned policy success rates for tasks with ground truth preference specified by user study participants. PLGA (active) outperforms PLGA (passive), LGA (passive), and GCBC on task performance, demonstrating an ability to flexibly incorporate natural language human preferences into abstraction construction.

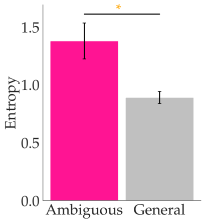

Figure 4: Entropy values show PLGA can model its own uncertainty under preference ambiguity.

Study Design. We conducted a computer-based in-person user study where participants were shown a text description of the task, and asked to give a general preference specified in natural language.

The study is split into three phases: familiarization, scenario generation, and preference querying. During familiarization, we introduce the user to the task context, the simulation interface, and full feature list that is available in the environment. We then show them an example task and text abstraction $\hat{\phi}$. In scenario generation, we introduce six scenarios (two per task), where we describe a background story for each user (e.g. *you are about to have guests over for dinner* or *you now need to figure out how to store food*). This was intended to elicit a natural preference for how each scenario would be interpreted that invoked different downstream preference-conditioned abstractions (e.g. *plate* and *bowl* may be more relevant for the first scenario, while *container* and *box* might be more relevant for the second). In preference querying, we then ask the user to specify, in language, their explicit preferences for the task as our preference query. This preference query is then used by PLGA to explicitly update its abstraction.

Participants. We recruited 12 participants (50% male, aged 18-29) from the greater community. We paid participants $30 for participation. Our study passed institutional IRB review.

5.2 Objective Results: Active PLGA Successfully Learns from Human Preference Queries

Now that we have established active PLGA enables a more natural and less effortful user interaction, we measure whether querying users for their preference in natural language results in good preference-conditioned abstractions as compared to baselines.

Manipulated Variables. We compare the performance of active PLGA to non-interactive abstraction construction algorithms: Passive PLGA (where the LM did not explicitly query the human for their preference and instead used its best estimate $\hat{\theta} \in \Theta_{LM}$), Passive LGA (where the LM builds an abstraction without explicitly modeling preference), and GCBC. We would like the comparison to validate the importance of identifying when the LM is unsure in its hypotheses and asking the human, when compared to taking its best guess (Passive PLGA), not reasoning about preferences at all (Passive LGA), or not even using state abstractions (GCBC).

Dependent Measures. For measuring downstream task success, we report the same success rate as in section 4. Note, instead of assuming ground truth test distributions constructed by the experimenters, we now assume the abstractions explicitly specified by the human manually during the Active LGA querying in section A.1 *are* the ground truth test distributions by which to evaluate. This is a reasonable assumption considering previous work Peng et al. (2023); Bullard et al. (2018); Cakmak & Thomaz (2012) has demonstrated the ability of humans to perform task-specific feature selection to their individualized preferences.

Hypothesis H3: Abstractions learned with human preference queries (Active PLGA) result in better performing policies compared to passive methods (Passive PLGA, Passive LGA, GCBC).

Analysis. figure 5 shows that active PLGA outperforms other passive baselines in learning good preference-conditioned abstractions from human queries in natural language, supporting **H3**. We further confirmed this by running one-sided t-tests (marked with orange asterisks) between Active PLGA and Passive LGA, our strongest competing baseline, confirming significance at $p < 0.05$. This illustrates the ability of PLGA to integrate information queried from the user meaningfully in constructing state abstractions. Moreover, while every method has its natural user effort vs. information gain tradeoff, PLGA's ability to query seamlessly for natural human feedback while reducing user frustration and effort is an exciting testament to the value of strong priors for learning.

6 Investigating PLGA on a Spot Robot

We demonstrate the real world abstraction construction utility of PLGA on a Spot robot[1] performing mobile manipulation tasks.

Robotic Platform. Spot is a mobile manipulation legged robot equipped with six RGB-D cameras (one in gripper, two in front, one on each side, one in back), each producing an observation of size 480x640. We only use observations taken from the front camera.

Tasks and Data Collection. We collected demonstrations of a human teleoperating the robot while performing two mobile manipulation tasks with household objects: *place the drink in the bin* and *throw away the can*. The manipulation action space consists of the following three actions along with their parameters: (*xy, grasp*), (*xy, move*), (*drop*) while the navigation action space consists of a SE(3) group denoting robot waypoints[2]. For *place the drink*, the robot is tasked with bringing an already-grasped soda can to a specified location and dropping it into a trash can. We assume the user has a preference for avoiding electronics in the way, otherwise taking the shortest path. For *throw away*, the robot is tasked with picking up a drink on a table, bringing it to a correct bin (either recycling or trash), and successfully dropping the drink into the bin. We assume the user has a preference for placing cans in a recycling bin if one is available, and otherwise placing them in the trash. Both tasks include possible distractors like drills and brushes.

For *place the drink*, we generate demonstrations of the robot placing a soda can into the recycling if available, otherwise trash. At test time, we evaluate the robot on the scenarios with a water bottle instead. For *throw away the can*, we generate demonstrations of the robot walking directly to the trash can when a shirt is on the ground, but avoiding the drill when it is present. At test time, we evaluate the robot on two new scenes: a laptop (to avoid) and pants (walk across). While the robot sees a trajectory of a user avoiding a drill during train, it is not exposed to laptops prior to test.

Training and Test Procedure. We first extract a segmented image from the observations using Segment Anything (Kirillov et al., 2023) and captioner Dedic (Zhou et al., 2022) to perform a check for behavior Δ (e.g. is the robot taking a different trajectory when a laptop is present in the scene vs. shorts). If the answer is yes, we instantiate the full PLGA pipeline. First, we perform a preference query to the LM with the initial two scenes and task description; next, we use this preference to query the LM to construct a preference-conditioned abstraction; lastly, we map this abstraction back into the observation dimension.

Takeaway. PLGA produced policies capable of successfully completing both tasks consistently, even when faced with new distractor objects, target object colors, or unseen linguistic specifications. Excitingly, we were able to observe non-trivial generalization capabilities, particularly in the avoid task (the robot successfully learned to avoid laptops from only seeing a demonstration of avoiding a drill). The failures we did observe were largely due to captioning errors (e.g. the segmentation model detected the object but was unable to produce a good text description). Our demonstration of PLGA on real robotic hardware indicates an exciting future in using LMs to help generate preference-conditioned state abstractions.

[1] Our Spot's name is Moana.

[2] For ease of data generation, we perform imitation learning over the trajectory rather than each state (i.e. predict a sequence of actions from an initial observation).

REFERENCES

David Abel, John Salvatier, Andreas Stuhlmüller, and Owain Evans. Agent-agnostic human-in-the-loop reinforcement learning. *arXiv preprint arXiv:1701.04079*, 2017.

Gati Aher, Rosa I. Arriaga, and Adam Tauman Kalai. Using large language models to simulate multiple humans and replicate human subject studies, 2023.

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

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351, 2023. doi: 10.1017/pan.2023.2.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, T. J. Henighan, Nicholas Joseph, Saurav Kadavath, John Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Christopher Olah, Benjamin Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *ArXiv*, abs/2204.05862, 2022. URL `https://api.semanticscholar.org/CorpusID:248118878`.

Andreea Bobu, Chris Paxton, Wei Yang, Balakumar Sundaralingam, Yu-Wei Chao, Maya Cakmak, and Dieter Fox. Learning perceptual concepts by bootstrapping from human queries. *IEEE Robotics and Automation Letters*, 7(4):11260–11267, 2022.

Andreea Bobu, Andi Peng, Pulkit Agrawal, Julie Shah, and Anca D Dragan. Aligning robot and human representations. *arXiv preprint arXiv:2302.01928*, 2023.

Daniel S Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pp. 330–359. PMLR, 2020a.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020b. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Kalesha Bullard, Sonia Chernova, and Andrea L Thomaz. Human-driven feature selection for a robotic agent learning classification tasks from demonstration. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6923–6930. IEEE, 2018.

Maya Cakmak and Andrea L Thomaz. Designing robot learners that ask good questions. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pp. 17–24, 2012.

Crystal Chao, Maya Cakmak, and Andrea L Thomaz. Transparent active learning for robots. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 317–324. IEEE, 2010.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017a. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

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, 2017b.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

John D Co-Reyes, Abhishek Gupta, Suvansh Sanjeev, Nick Altieri, Jacob Andreas, John DeNero, Pieter Abbeel, and Sergey Levine. Guiding policies with language via meta-learning. *arXiv preprint arXiv:1811.07882*, 2018.

Carlos G Correa, Mark K Ho, Frederick Callaway, Nathaniel D Daw, and Thomas L Griffiths. Humans decompose tasks by trading off utility and computational cost. *arXiv preprint arXiv:2211.03890*, 2022.

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.

Danica Dillion, Niket Tandon, Yuling Gu, and Kurt Gray. Can ai language models replace human participants? *Trends in Cognitive Sciences*, 27(7):597–600, 2023. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2023.04.008. URL https://www.sciencedirect.com/science/article/pii/S1364661323000980.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislav Fort, Zac Hatfield-Dodds, Tom Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olsson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom Brown, Nicholas Joseph, Sam McCandlish, Chris Olah, Jared Kaplan, and Jack Clark. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, 2022.

Sandra G. Hart and Lowell E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In Peter A. Hancock and Najmedin Meshkati (eds.), *Human Mental Workload*, volume 52 of *Advances in Psychology*, pp. 139–183. North-Holland, 1988. doi: https://doi.org/10.1016/S0166-4115(08)62386-9. URL https://www.sciencedirect.com/science/article/pii/S0166411508623869.

Mark K Ho, David Abel, Carlos G Correa, Michael L Littman, Jonathan D Cohen, and Thomas L Griffiths. People construct simplified mental representations to plan. *Nature*, 606(7912):129–136, 2022.

Mark K Ho, Jonathan D Cohen, and Tom Griffiths. Rational simplification and rigidity in human planning. 2023.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14409–14428, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.806. URL https://aclanthology.org/2023.acl-long.806.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 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. *arXiv preprint arXiv:2207.05608*, 2022b.

Holly Huey, Xuanchen Lu, Caren M Walker, and Judith E Fan. Visual explanations prioritize functional properties at the expense of visual fidelity. *Cognition*, 236:105414, 2023.

Hong Jun Jeon, Smitha Milli, and Anca Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4415–4426. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/2f10c1578a0706e06b6d7db6f0b4a6af-Paper.pdf`.

Jianchao Ji, Zelong Li, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Juntao Tan, and Yongfeng Zhang. Genrec: Large language model for generative recommendation. *ArXiv*, abs/2307.00457, 2023. URL `https://api.semanticscholar.org/CorpusID:259332879`.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2022.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

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.

Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.

Belinda Z. Li, William Chen, Pratyusha Sharma, and Jacob Andreas. LaMPP: Language models as probabilistic priors for perception and action, 2023.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 994–1003, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1094. URL `https://aclanthology.org/P16-1094`.

Jessy Lin, Daniel Fried, Dan Klein, and Anca Dragan. Inferring rewards from language in context. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8546–8560, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.585. URL `https://aclanthology.org/2022.acl-long.585`.

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, and Jiebo Luo. Llm-rec: Personalized recommendation via prompting large language models. *ArXiv*, abs/2307.15780, 2023. URL `https://api.semanticscholar.org/CorpusID:260334587`.

Zhengyi Ma, Zhicheng Dou, Yutao Zhu, Hanxun Zhong, and Ji-Rong Wen. One chatbot per person: Creating personalized chatbots based on implicit user profiles. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, pp. 555–564, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380379. doi: 10.1145/3404835.3462828. URL `https://doi.org/10.1145/3404835.3462828`.

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.

Zhiming Mao, Huimin Wang, Yiming Du, and Kam-Fai Wong. UniTRec: A unified text-to-text transformer and joint contrastive learning framework for text-based recommendation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1160–1170, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.100. URL https://aclanthology.org/2023.acl-short.100.

OpenAI. GPT-4 technical report, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Andi Peng, Ilia Sucholutsky, Belinda Li, Theodore Sumers, Thomas Griffiths, Jacob Andreas, and Julie Shah. Learning with language-guided state abstractions.

Andi Peng, Aviv Netanyahu, Mark K Ho, Tianmin Shu, Andreea Bobu, Julie Shah, and Pulkit Agrawal. Diagnosis, feedback, adaptation: A human-in-the-loop framework for test-time policy adaptation. 2023.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. Assigning personality/profile to a chatting machine for coherent conversation generation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 4279–4285. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/595. URL https://doi.org/10.24963/ijcai.2018/595.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

Krishan Rana, Andrew Melnik, and Niko Sünderhauf. Contrastive language, action, and state pre-training for robot learning. *CoRR*, abs/2304.10782, 2023. doi: 10.48550/arXiv.2304.10782. URL https://doi.org/10.48550/arXiv.2304.10782.

Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. Skill induction and planning with latent language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1713–1726, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.120. URL https://aclanthology.org/2022.acl-long.120.

Haoyu Song, Wei-Nan Zhang, Yiming Cui, Dong Wang, and Ting Liu. Exploiting persona information for diverse generation of conversational responses. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 5190–5196. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/721. URL https://doi.org/10.24963/ijcai.2019/721.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023a.

Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 1929–1937, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539382. URL https://doi.org/10.1145/3534678.3539382.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13484–13508, Toronto, Canada, July 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.754. URL https://aclanthology.org/2023.acl-long.754.

Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. Tidybot: Personalized robot assistance with large language models. *Autonomous Robots*, 2023a.

Likang Wu, Zhilan Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. A survey on large language models for recommendation. *ArXiv*, abs/2305.19860, 2023b. URL https://api.semanticscholar.org/CorpusID:258987581.

Andy Zeng, Maria Attarian, brian ichter, Krzysztof Marcin Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael S Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. Socratic models: Composing zero-shot multimodal reasoning with language. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=G2Q2Mh3avow.

Ruohan Zhang, Faraz Torabi, Lin Guan, Dana H Ballard, and Peter Stone. Leveraging human guidance for deep reinforcement learning tasks. *arXiv preprint arXiv:1909.09906*, 2019.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1205. URL https://aclanthology.org/P18-1205.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2023.

Hanxun Zhong, Zhicheng Dou, Yutao Zhu, Hongjin Qian, and Ji-Rong Wen. Less is more: Learning to refine dialogue history for personalized dialogue generation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5808–5820, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.426. URL https://aclanthology.org/2022.naacl-main.426.

Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, pp. 350–368. Springer, 2022.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. Evaluating commonsense in pre-trained language models. In *AAAI Conference on Artificial Intelligence*, 2019. URL https://api.semanticscholar.org/CorpusID:208310123.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020.

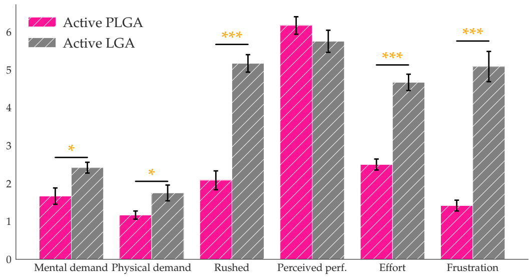

Figure 6: User study interaction results (lower is better for all but perceived performance). The interaction experience with Active PLGA is rated more favorably by users than with Active LGA.

A Appendix

A.1 Subjective Results: PLGA Enables More Natural and Easy User Interaction

We first tested if users can easily and effortlessly specify individualized preferences via natural language to the model in a manner that is less burdensome and frustrating than baseline human-in-the-loop abstraction construction methods.

Manipulated Variables. We are interested in comparing the user experience of PLGA vs. a baseline human-in-the-loop abstraction method. The baseline we select is the active version of LGA where users are first presented with an LM's best guess of the correct abstraction list (without explicitly modeling preference), and then asked to refine the resulting representation via a text-based interface. We implemented this baseline as an additional condition in our user study. In the active LGA condition, the preference querying phase is instead replaced with an explicit abstraction querying phase, where the user is tasked with specifying, in text, the feature list $\hat{\phi}$ that contains all task-relevant aspects for their preferences in each task. We provide a full list of environment features for easy access. We counterbalance conditions and record qualitative task experience post-conclusion of both conditions.

Dependent Measures. For measuring interaction experience, we administered the subjective 7-point Likert Scale survey, inspired by the NASA-TLX Hart & Staveland (1988). We presented the survey after the user completed both conditions, and recorded responses for each.

Hypothesis H2: Describing a language preference (Active PLGA) is a more natural and less effortful user interaction experience than manually filtering relevant abstraction features (Active LGA).

Analysis. figure 6 illustrates our subjective user study results with the NASA TLX scores aggregated across participants. We additionally ran paired t-tests with significance level $\alpha = 0.05$, marked with orange asterisks. We see that users found PLGA to be significantly less mentally ($t(11) = -2.46, p < 0.05$) and physically demanding ($t(11) = -2.54, p < 0.05$), and the results are even more pronounced for feeling rushed ($t(11) = -7.40, p < 0.001$), frustrated ($t(11) = -8.48, p < 0.001$), or expending a great deal of effort ($t(11) = -8.99, p < 0.001$). Meanwhile, we found no statistically significant difference in perceived performance ($t(11) = 1.60, p = 0.14$), suggesting that Active PLGA offers a more natural and effortless interaction experience than Active LGA with no loss in performance quality. Overall, results support our hypothesis **H2**.

The result is not surprising – after all, it is to be expected that giving a natural language utterance is an easier experience than inspecting a list of features and selecting the right subset. However, we wanted to verify that users overall find it easy to explicate their preference in words, and that training the robot this way does not decrease their perception of its performance. From this point of view, the results are positive and even encouraging for future research using natural language to explicate human preferences.

A.2 Discussion

We presented PLGA, a framework for learning preference conditioned state abstractions from language and demonstration information. Particularly, we focused on settings where the language task specification does not list everything the human cares about. We introduced LM preference queries for inferring user preferences present in demonstrations directly from LM priors. Our simulated experiments, user study, and Spot robot demos illustrate that natural language can be a convenient vehicle to communicate hidden preferences for constructing state abstractions, and those abstractions result in improved downstream task performance. Although we demonstrated PLGA's real-world applicability in home manipulation tasks, we are excited about future opportunities in shared autonomy tasks (where the human may have a preference for which aspects of the task the robot assists with), or autonomous driving (where users have a preference for what objects to avoid).

Limitations and Future Work. In our work, we assumed we had no further information regarding differences in user behavior beyond the initial states that induced these behaviors. However, we do not use the information about *how* exactly user behavior changed. A natural direction would be to extend PLGA's preference query abilities to user trajectories, where richer features, like obstacle avoidance distance, can be explored. Such a path would open more meaningful opportunities for grounding natural language to the language of human behavior.

Moreover, while we focused here on using language priors to construct state abstractions for imitation learning, a natural parallel would be to explore this framework in the context of rewards, where rich semantic priors could be extremely meaningful to few-shot downstream learning from demonstrations. Furthermore, our algorithm is not designed to be iterative, which means that there is no opportunity for continual preference learning after repeated exposure to different interactions. However, there are many trajectory-based features that arise in the context of robotics that would require more text-based motion information regarding user actions that we currently do not have.

Lastly, while we broached the subject of active preference elicitation, we did not conduct a deep dive into meaningful ways to interact with the user when trying to learn their preference (opting instead to query them directly if uncertain). Future work can explore different ways of performing preference elicitation with language models, including iterative approaches that perform sequential updates to the reward or preference model.

A.3 Related Work

Learning from Human Input. Existing frameworks for interactive querying for downstream learning, like TAMER (Knox & Stone, 2008) and COACH (MacGlashan et al., 2017), use human feedback to train policies, but are restricted to binary or scalar labeled rewards Abel et al. (2017); Zhang et al. (2019). Another line of work looks at learning from human preferences, often by asking them to compare or rank trajectory snippets (Christiano et al., 2017b; Brown et al., 2020a). There are also works that actively learn from human teachers, where the emphasis is on generating actions or queries that are maximally informative for the human to label (Bobu et al., 2022; Chao et al., 2010). Unfortunately, these approaches all are limited by the fact that the feedback asked of the human is overfit to specific failures or desired data points, and rarely scale well relative to human time or effort Bobu et al. (2023).

Language Models for Human Preferences. LMs are increasingly being used for personalized applications. Prior work has explored using LMs for recommendation systems Wu et al. (2023b); Ji et al. (2023); Lyu et al. (2023); Mao et al. (2023); Wang et al. (2022), user-specific chatbots Zhang et al. (2018); Ma et al. (2021); Li et al. (2016); Qian et al. (2018); Song et al. (2019); Zhong et al. (2022), or even sorting household objects according to personal preferences Wu et al. (2023a).

A range of techniques have been introduced to specify human preferences and inject them into LMs. With the popularization of prompting-based techniques, users simply have to write a textual description (called a *prompt*) specifying their preferred task and condition LMs on this prompt to induce their desired behavior Brown et al. (2020b). In order to encourage LMs to produce outputs in line with users' preferences, recent work has explored techniques such as instruction-tuning Ouyang et al. (2022); Honovich et al. (2023); Wang et al. (2023b); Chung et al. (2022); Zhang et al. (2023) and reinforcement learning from human feedback (RLHF) Bai et al. (2022); Ziegler et al. (2020); Christiano et al. (2017a); Stiennon et al. (2020); Ganguli et al. (2022).

Furthermore, having been pre-trained on large corpora of human-generated text Raffel et al. (2019), LMs often possess sensible priors over "typical"[3]) human preferences and behaviors Li et al. (2023); Brown et al. (2020b); Zhou et al. (2019). Because of this, LMs have at times even been used as *simulations* of humans Aher et al. (2023); Dillion et al. (2023); Argyle et al. (2023). As part of prompting, LMs must implicitly perform language understanding on human-written prompts to infer their preferences. However, LMs have also been used to *explicitly* infer human preferences from linguistic specifications. For example, recent work has examined reward learning using LMs Lin et al. (2022); Kwon et al. (2023).

Language Models in Robotics. LMs hold commonsense knowledge about object properties, functions, and their relevance to various tasks. This is why many recent works have explored using LMs to output plans directly, i.e. generate primitives or high-level action sequences (Sharma et al., 2022; Ahn et al., 2022; Huang et al., 2022a;b). These approaches use priors embedded in LMs to produce better *instruction following* models, or in other words, better compose base skills to generate more complex behavior (Zeng et al., 2023; Li et al., 2023; Ahn et al., 2022; Wang et al., 2023a). In contrast, we use LM priors to learn *it* preferences over relevant features. Recent work (Peng et al.) has also proposed to use LMs to perform *state abstraction* for learning better skills *from scratch*, instead leveraging the LM's priors to identify task-relevant features for state abstraction construction.

A.4 Full Prompt

ChatGPT models (including GPT4) can take in both system prompts and user prompts. We split our prompt into these two parts.

Preference Query. System prompt where {scene_intersection} is replaced by the list of all similar features between two scenes and {scene1_minus_scene2} and {scene2_minus_scene1} are the lists of scene differences.

> There are two scenes. The user takes a different trajectory in the first scene vs. the second.
>
> The first and second scene both have the following features: {scene_intersection}
> The first and second scene differ on the following:
> First scene- {scene1_difference}
> Second scene- {scene2_difference}
>
> What are the most likely high-level preferences to have caused the difference in the user's behavior and why? The user took different trajectories in the two scenes. Please give a list of brief preferences (with only one reason) and assign a confidence score to each answer, in the format [["answer", score], ["answer", score], ...]. Please ensure all scores sum up to 1.

Abstraction Query. System prompt where {object_list} is replaced by the list of all object types in the environment and {object_colors} by the list of all colors and textures:

> You are interfacing with a robotics environment that has a robotic arm learning to manipulate objects based on some linguistic command (e.g. "pick up the red bowl"). At each interaction, the researcher will specify the command that you need to teach the robot. In order to teach the robot, you will need to help design the training distribution by specifying what properties task-relevant objects can have based on the given command. Objects in this environment have two properties: object type, object color. Any object type can be paired with any color, but an object can only take on exactly one object type and exactly one color.
> Object types:
> {object_list}
> Object colors:
> {object_colors}

[3] It is worth noting that text scraped from the internet, which constitutes the bulk of what today's LMs are trained on, is biased and does not capture a representative sample of human preferences globally.

User prompt where {rule} is replaced by one of the task prompts listed above, {group} is replaced by "object color" or "object type", and {candidate} is replaced by each candidate object color or type that we would like the LM to evaluate:

> The command is "{rule}". In an instantiation of the environment that contains only some subset of the object types and colors, could the target object have {group} "{candidate}"? Think step-by-step and then finish with a new line that says "Final answer:" followed by "yes" or "no".

A.5 Task Details

Pick:

ripe tomato:

- Task description: *Bring me a tomato.*
- True distribution: {objects: tomato}, {textures: red, dark red}

food container:

- Task description: *Bring me something to put food in.*
- True distribution: {objects: bowl, container, box}, {textures: ALL}

dry cereal bowl:

- Task description: *Bring me a cereal bowl*
- True distribution: {objects: bowl, drying rack, drying towel, drying cloth}, {textures: ALL}

Place:

non-electronic:

- Task description: *Put down my mug.*
- True distribution: {objects: ALL
  iPad, laptop, phone}, {textures: All}

stable surface:

- Task description: *Put down the pan.*
- True distribution: {objects: pan, coaster, pallet}, {textures: ALL}

desired content:

- Task description: *Put away my food.*
- True distribution: {objects: tomato, pepper, peach, apple, container, box}, {textures: ALL}

Sweep:

hot object:

- Task description: *Sweep the food into the sink.*
- True distribution: {objects: food, sink, stove, pan}, {textures: red, dark red}

sweepable:

- Task description: *Sweep the dust into the container.*
- True distribution: {objects: bin, container, floor}, {textures: wooden, granite}

sharp:

- Task description: *Sweep the food into the sink.*
- True distribution: {objects: pepper, peach, apple, sink, knife, sharp block}, {textures: ALL}