Learning with Language-Guided State Abstractions

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—We describe a framework for using natural language to design state abstractions for robot learning tasks. Policy learning in high-dimensional observation and action spaces is greatly facilitated by well-designed state representations, which can surface important features of an environment and hide irrelevant ones. Today, these state representations must often be designed by hand, or derived from other labor-intensive labeling procedures. Our framework, LGA (*language-guided abstraction*), uses a combination of natural language supervision and background knowledge from language models (LMs) to automatically build state representations tailored to new tasks. In LGA, a user first provides a (possibly incomplete) description of a target task in natural language; next, a pre-trained LM translates this task description into a state abstraction function that masks out irrelevant features; finally, a policy is co-trained using a small number of demonstrations and LM-generated abstract states. Experiments on continuous robotic control tasks show that LGA yields state abstractions similar to human-designed ones, but in a fraction of the time, and that these abstractions improve policy generalization and robustness in the presence of spurious feature correlations and ambiguous demonstrations.**

I. Introduction

In complex environments with many objects, features, and possible goals, learning a policy from a small number of demonstrations can be challenging or impossible [6, 14]. Consider the demonstration in Fig. 1A, which shows a robot arm executing a specific maneuver to perform a desired task. Which task is demonstrated here—grabbing a pan, grabbing a metal object, or placing any object on a stove? Are all pans metallic, and all stoves red, or must learners generalize to other feature combinations when deployed in test environments?

In humans, *state abstraction* is essential for generalizable learning. When learning (and planning), humans reason over simplified mental representations of environment states that hide details and distinctions not needed for action prediction [19]. Useful state abstractions are task-dependent, and a growing body of evidence supports the conclusion that humans dynamically construct such representations to learn new tasks [20, 23]. Importantly, this process does not begin with a blank slate—instead, experience, common-sense knowledge, and direct instruction provide rich sources of prior knowledge about which features matter for which tasks. In Fig. 1A, learning that the demonstrated skill involves *cooking* makes it clear that the object's identity (*pan*), not its appearance (*metallic*), are likely important. Meanwhile, the demonstration provides complementary information (about the desired speed, goal placement location, etc.) that are hard to communicate in language and not encapsulated by the label *cooking* alone.

What would it take to build autonomous agents that can reason about tasks and representations in this way? State abstraction has been a major topic of research from the very earliest days of research on sequential decision-making, with significant research devoted to both unsupervised representation learning [13, 5, 18, 27] and human-in-the-loop state design [4, 7, 3, 17]. But there are few tools for incorporating human priors for constructing abstractions in new domains.

In this paper, we propose to use *natural language* as a source of information about state representations. Our approach, called **Language-Guided Abstraction (LGA)** (Fig. 1B), begins by querying human users for high-level task descriptions, then uses a pre-trained language model (LM) to translate these descriptions into task-relevant state abstractions. Importantly, LGA requires only natural language annotations for state features. Unlike most recent work applying language models to sequential decision-making tasks [22, 1], it does not depend on pre-trained skills, multiple training tasks, or even the ability to fully specify tasks in language. It may be applied generically to any technique for learning from demonstrations, with or without other forms of side-information. Experiments comparing LGA to standard behavior cloning show that generated abstractions improve sample efficiency and distributional robustness in both single- and multi-task settings. They match the performance of human-designed state abstractions while requiring only a fraction of the human effort.

A crucial function of language is to communicate useful information about the structure of the world [10, 41, 40, 32, 29, 35]. LGA integrates this linguistically transmitted representational information into policy learning, enabling state abstraction with human-like flexibility and task-specificity.

II. Preliminaries

In **behavioral cloning (BC)**, we assume access to a set of expert demonstrations $D_{\text{train}} = \{\tau^i\}_{i=0}^n = \{(s_0^i, a_0^i, s_1^i, a_1^i, ...s_t^i, a_t^i)\}_{i=0}^n$ from which we derive an expert policy π_θ [36] by minimizing:

$$\mathcal{L}_{\text{BC}} = E_{(s_t^i, a_t^i)\ \sim D_{\text{train}}}[\|\pi_\theta(s_t^i) - a_t^i\|_2^2] \tag{1}$$

In **goal-conditioned behavioral cloning (GCBC)**, policies additionally condition on goals l [12]. Motivated by the idea that natural language is a flexible, intuitive interface for humans to communicate, we specify the goal through language instruction $l \in L$, resulting in a training objective:

$$\mathcal{L}_{\text{GCBC}} = E_{(s_t^i, a_t^i, l^i)\ \sim D_{\text{train}}}[\|\pi_\theta(s_t^i, l^i) - a_t^i\|_2^2] \tag{2}$$

Above we have highlighted differences from BC in red.

A (GC)BC policy $\pi(s_t^i, l^i)$ must generalize to novel commands l^i and contextualize them against potentially novel states s_t^i. Due to difficulties with sampling representative data and expense of collecting a large number of expert demonstrations, systematic biases may be present in both

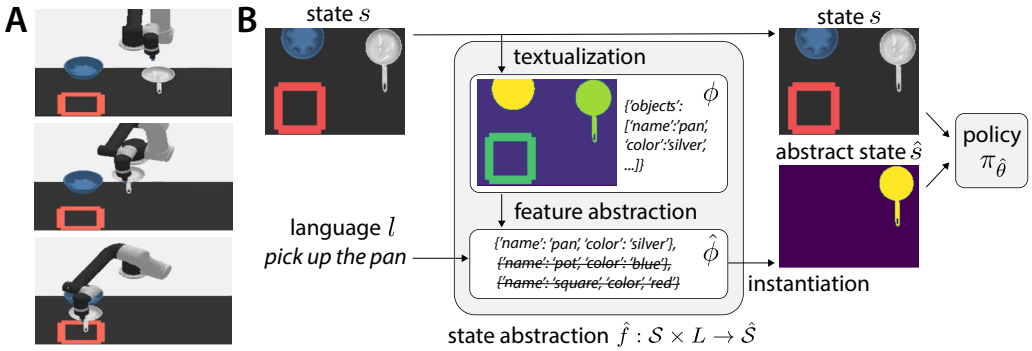


Fig. 1. **A**: Example trajectory in our environment, picking up a pan and placing it on the stove. **B**: Our approach, *Language Guided Abstraction* (LGA), constructs a state abstraction with task-relevant features from an LM. The state abstraction is then co-trained with the original state.

the training demonstrations $(s_0^i, a_0^i, \cdots)$ for a goal, or the commands l^i describing the goal. This can result in brittle policies which fail to generalize. In particular, we identify two possible sources of covariate shift: (1) out-of-distribution states s^i, or (2) out-of-distribution utterances l^i.

III. Language-Guided Abstraction (LGA)

Traditional (GC)BC forces the behavioral policy to learn a joint distribution over language and perceptual states — effectively requiring the robot to develop both language and perceptual scene understanding simultaneously, and to ground language in the current state. Our approach instead *offloads* contextual language understanding to a LM which identifies task-relevant features in the perceptual state. We introduce a *state abstraction function* that takes the raw perceptual inputs and language-specified goal, and outputs a set of task-relevant features. Intuitively, LGA can be seen as a form of language-guided attention [2]: it conditions the agent's perceptual inputs on language, removing the burden of language understanding from the policy. Fig. 1 illustrates our approach.

A. State Abstraction Function

Formally, we define a *state abstraction function* $\hat{f}$ that produces task-relevant state representations: $\hat{f} : \mathcal{S} \times L \to \hat{\mathcal{S}}$. $\hat{f}$ consists of three steps:

Textualization ($s \to \phi$). First, similar to other LM-assisted methods [22, 1] the raw perceptual input s is transformed into a text-based feature set ϕ, representing a set of features (described in natural language) that encapsulate the agent's full perceptual inputs.[1] This text representation may include common visual attributes of the state like object type, color, and texture. In fig. 1B, for example, the textualization step transforms state observations to object types and colors.

Feature abstraction ($\phi \to \hat{\phi}$). Given a feature set ϕ, we achieve **abstraction** by removing features from ϕ which are irrelevant to the task l: $(\phi, l) \to \hat{\phi}$. In fig. 1B, for example, the abstraction step removes distractor objects from the feature set, in this case preserving only the target object (*pan*).

Instantiation ($\hat{\phi} \to \hat{s}$). As a last step, we transform the abstracted feature set back into an (abstracted) perceptual input with only relevant features on display: $\hat{\phi} \to \hat{s}$. This step is required for putting the abstracted feature set into a form understandable to the policy. In fig. 1B, for example, this step transforms the abstracted feature set into a state showing only the relevant object, removing colors.

[1] In the general case, this stage could be implemented via semantic segmentation [25] with an object captioner.

B. Abstraction-Augmented Policy Learning

After receiving an abstract state from the state abstraction function, we *augment* the original state s with the abstracted state $\hat{s}$ and learn a policy on top of both states, yielding $\pi_{\hat{\theta}} : \mathcal{S} \times \hat{\mathcal{S}} \to \mathcal{A}$. [2] $\pi_{\hat{\theta}}$ is trained to minimize the loss:

$$\mathcal{L}_{\text{LGA}} = E_{(s_t^i, a_t^i, l^i) \sim D_{\text{train}}}[||\pi_{\hat{\theta}}(s_t^i, \hat{f}(s_t^i, l^i)) - a_t^i||_2^2] \quad (3)$$

with the differences from GCBC (Eq. 2) highlighted in red. The LGA policy $\pi_{\hat{\theta}}$ never sees the language input l; instead, it operates over the language-conditioned representation, $\hat{s}$.

LGA offers several appealing properties relative to traditional (GC)BC. Feature correlates are mitigated during training because all goal information is highlighted in semantic maps rather than the raw pixels. The LM can help resolve goal ambiguities present at test by effectively converting the raw utterance and test state into a goal-conditioned abstract state that the agent can then operate on. Goal ambiguity is mitigated during deployment, as the LM can "intercede" to determine only the contextually-appropriate task-relevant features.

IV. Experimental Setup

A. Environment and Feature Space

We generate a series of robotic control tasks from VIMA [24], a vision-based simulation environment. We focus on setups where a UR5 arm is tasked with picking up a specified target object and placing it on a goal on a tabletop surface. The environment consists of a large feature space (29 objects, e.g. "bowl", and 81 textures, e.g. "wooden") of various semantic meaning (full list in Appendix). Success is a pick-and-place action of the specified target object within radius ϵ of the goal.

[2] While learning policies directly over the abstract state alone is theoretically possible and computationally cheaper, it also requires that the abstraction preserve *all* task-relevant information. We take a more conservative approach, allowing the policy to observe both the original and abstracted states.

The task-relevant feature set $\hat{\phi}$ is specified via a JSON configuration containing possible instantiations of the target object and texture. Because the full feature space (ϕ) can be combinatorially large (objects of all textures), constructing abstractions that preserve only task-relevant features of an ambiguously specified objective (l), e.g. *something that can hold water*, is challenging.

B. State Abstraction Function

Textualization. In our experiments, we extract a structured feature set ϕ by obtaining the ground truth state segmentation mask and text object descriptions from the simulator.

Feature abstraction. We use two versions of LGA for eliciting relevant feature sets $\hat{\phi}$. Our (vanilla) **LGA** simply uses a language model to specify a relevant feature set and **LGA-HILL** uses an LM together with a human-in-the-loop.

In our experiments, for both LGA methods, we use GPT4 [34] as the LM for feature extraction. For each task, we query the LM for an initial hypothesis of a specified abstract feature set, which may then be interactively refined by a human. In our experiments, we implement this by asking the LM whether each object type and feature in the environment could correspond to the intended task described by the goal utterance. In each query, we give GPT4 a description of the environment (including a list of all possible object types and colors), the goal utterance for the task being considered, and a target object type or feature that we want the model to evaluate (the full prompt and additional details can be found in the Appendix). With these queries, we have the model provide a binary response for whether every object type and feature can apply to the target object specified in each task.

For LGA-HILL, humans are shown the list of features the LM identified as pertinent and can refine $\hat{s}$ by adding or removing features (section IV-E details the user study).

Instantiation. As described in section IV-A, the abstract state representation consists of a distribution over *target object types* and *target object colors*. Given a target object type and color, we use an image editor or simulator to pick out all objects of the salient type and color. In particular, the image editor produces a "goal mask", a binary pixel mask where the goal object is converted to ones and everything else is zeroed out.[3] In our setting, the simulator provides the ground-truth scene segmentation which we use to produce the goal mask.

C. Abstraction-Augmented Policy Learning

We instantiate the abstraction-augmented policy procedure, described in section III-B, in our environment. LGA learns a joint policy by co-training on $(s, \hat{s})$. We implement a dual-CNN architecture that independently processes each state input into embeddings, which we then concatenate and feed through a MLP for action prediction.

[3]Here, we produce a simple binary pixel mask highlighting the target object. In general, however, it may be fruitful to experiment with more general scalar cost maps, such as highlighting areas that should be avoided.

D. Training and Test Tasks

Training Tasks. We are interested in studying how different methods are able to perform task-relevant feature selection, i.e. specify $\hat{\phi}$. We parameterize a task distribution by instantiating possible target features specified in $\hat{\phi}$ (e.g. "pan", "bowl", and "container" and all textures for *something that can hold water*). To create a task, we sample a target object and texture from this distribution as the "pick" object. We then generate a random distractor object. Distractors and target objects are randomly placed in one of three discretized state locations. Last, we place a fixed goal object (pallet) in the state as the "place" goal. We generate corresponding training demonstrations via an oracle.

Test Tasks. In practice, we do not have access to ground truth task distributions – this specification must come from the end user. However, for our experiments, we sample test tasks from a "true" distribution defined by experimenters to evaluate performance. Details are in the Appendix.

E. Comparisons

We compare instantiations of our proposed approach (**LGA** and **LGA-HILL**) against two baselines. Question-specific metrics are described in section IV.

Human (Baseline). The first baseline focuses on having users (instead of LMs) manually specify task-relevant features $\hat{\phi}$ in the feature abstraction stage, to isolate the effect of LM-aided specification. All other components remain the same as in LGA. The features are fed to the same dual-CNN architecture as for LGA and the policy is again learned over the concatenated representations.

GCBC (Baseline): Our second baseline is (goal-conditioned) behavior cloning, described in Eq. 2. We implement GCBC by concatenating an LM embedding of the goal utterance with a CNN embedding of the state and then learning a policy over the joint representation. We generate language embeddings using Sentence-BERT [37].

Human Data Collection. For comparisons that require human data (**LGA-HILL** and **Human**), we conducted a in-person user study to assess the ability of humans to specify task-relevant features $\hat{\phi}$ in the *feature abstraction* step, both with and without an LM. We recruited 14 participants (47% male, aged 22-40). All participants attested to having never worked with machine learning or holding a STEM degree. Our study passed IRB review and all data was anonymized.

We first introduce the user to the environment and full feature space ϕ. To additionally help human participants, we provide both a text and visual representation of features (the latter of which the LM does not see). We also walk the user through an example task specification. In the **Human** condition, we introduce task objectives (detailed in section IV) sequentially to the user and ask for them to specify $\hat{\phi}$ by typing their selected features into a task design file (we provide the full feature list for easy access). In the **LGA-HILL** condition, we show the same objectives, but with LM answers prefilled as the "raw" $\hat{\phi}$. To minimize ordering bias, we randomly assign

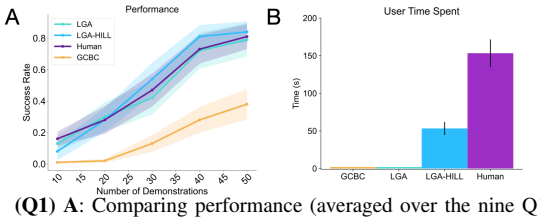


Fig. 2. **(Q1) A**: Comparing performance (averaged over the nine Q1 tasks) of each method when controlling the number of training demonstrations. **B**: Comparing the amount of time (averaged over the nine Q1 tasks) that human users spent specifying task-relevant features for each method. LGA outperforms naive GCBC on task performance while significantly reducing user time spent compared to manual feature specification ($p < 0.001$).

half of our participants to begin with each condition. For both conditions, we measure the time spent per task.

V. Results

We investigate three questions related to our approach: (Q1) Does LGA help users specify single-task policies faster and more effectively than baselines? (Q2) Is LGA more robust to observational covariate shift than baselines? (Q3) Is LGA more robust to language ambiguity in multi-task settings?

A. Q1: Improving ease and performance of task specification

Metrics. We measure two variables: *task performance* and *user specification time*. Task performance is assessed as the success rate of the policy on 20 sampled test tasks, and user specification time is measured as the time in seconds each user spent specifying the task-relevant feature set for each task.

Task Objectives. We construct nine tasks with a wide range of true abstract feature sets: *red heart*, *heart*, *letter*, *tiger-colored object*, *letter from the word letter*, *consonant with a warm color*, *vowel with multiple colors*, *something to drink water out of*, and *something from a typical kitchen*. We chose these tasks to 1) test the ability of the LM in LGA to semantically reason over which target objects and features satisfy human properties, e.g. *to drink water out of*, and 2) explore how potentially laborious performing manual feature specification from large feature sets can be for human users.

Results. We vary the number of demonstrations that each policy is trained on from 10 to 50. We visualize the resulting policy performance in Figure 2 (A). In Figure 2 (B), we visualize how much time users spent specifying the task-relevant features for each method[4]. From these results, it is clear that the feature-specification methods (i.e. LGA, LGA-HILL, and Human) are more sample-efficient than GCBC with consistently higher performance at each number of training demonstrations. Furthermore, LGA methods require significantly less user time than the Human baseline ($p < 0.001$ using a paired t-test for all tasks). We note that despite comparable task performance between LGA and LGA-HILL, the latter remains a valuable instantiation for tasks that require human input, such as high-stakes decision-making scenarios or personalization to individual user preferences.

Summary. *LGA requires significantly less human time than manually specifying features, but still leads to better sample-efficient performance compared to GCBC.*

[4]While we assign zero time spent specifying features for GCBC in our experiments, in practice, user time would be instead spent specifying initial state configurations for generating every demonstration, which only disadvantages naive GCBC even more when compared to methods that specify features.

Fig. 3. **(Q2: *Texture Shift*) A**: An example training task for the *heart* task. **B**: An example test task, where the texture of the target object has shifted (red box). **C**: LGA outperforms GCBC on both tested tasks, confirming our hypothesis that providing a non-ambiguous state abstraction as additional policy training information results in policies that are most robust to observational covariate shift.

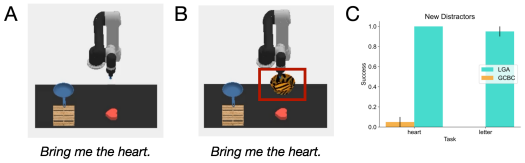


Fig. 4. **(Q2: *New Distractors*) A**: An example training task for the *heart* task. **B**: An example test task, where an additional distractor appears (red box). **C**: LGA outperforms GCBC on both tested tasks, additionally validating the robustness of the policy when deployed in cluttered scenes.

B. Q2: Improving single-task policy robustness to observational covariate shift

We have shown that LGA enables faster feature specification when compared to hand-engineering features from scratch, and also leads to more sample-efficient policies when compared to naive GCBC. We now ask whether the state abstractions learned by LGA policies result in performance that is more robust to state covariate shift.

Task Objectives. We evaluate on two previously defined tasks in Q1: *heart* and *letter*. For each task, we now construct two types of state covariate shift to test policy robustness:

Texture Shift. For *heart*, we define a training distribution where "heart" is parameterized only as the textures "red", "green", and "blue". At test time, we evaluate by sampling tasks from a distribution where "heart" is instead parameterized by "yellow", "tiger", and "rainbow". We do the same with all the letters present in the *letter* task'. These tasks are intended to evaluate LGA's flexibility in including (and excluding) appropriate task-relevant features.

New Distractors. We define the same training distributions as above, but evaluate by sampling test tasks where an additional distractor (randomly sampled from the full feature set) is placed randomly in the state. These tasks are intended to evaluate LGA's ability to be robust to cluttered scenes.

Results. We train on 50 demonstrations sampled from the train distribution for each task, then evaluate on 20 tasks sampled from the test distribution. As shown in Fig. 3, policies trained with either of the LGA methods are more robust to state covariate shifts than those trained with GCBC. This

Fig. 5. **(Q3) A**: An example training task for the *bowl* and *star* multi-task. **B**: An example test task, where two valid target objects (bowl and star) appear. LGA can resolve linguistic ambiguity, even though it has never seen the utterance "something I can find in a typical kitchen", to identify the correct target object (red box). **C**: LGA outperforms GCBC in multi-task settings with task ambiguity.

confirms our hypothesis that providing a non-ambiguous state abstraction that already precludes non-relevant task features from the state result in policies that are less brittle to spuriously correlated state features that may occur in test environments.

Summary. *Single-task policies trained with LGA are more robust to observational shift than BC.*

C. Q3: Improving multi-task robustness to linguistic ambiguity

The previous experiments evaluated LGA in single-task settings where each policy corresponds to a singular task. We now evaluate how LGA performs in the multi-task case where test environments allow for multiple tasks to be performed, and conditioning on language is necessary to resolve the ambiguity.

Task Objectives. We now define two new tasks: *bowl* and *star* and train a multi-task policy on both tasks. We define our training distribution as states that have either a bowl or a star, (with no distractors). At test time, we present the policy with states that have both a bowl and a star. We condition the policy on the two linguistic utterances it has seen before: "Bring me a bowl." and "Bring me a star.", and one that it has not: "Bring me something I can find in a typical kitchen." The first two test tasks evaluate whether the policy has successfully learned a multi-task policy that is robust to state shift ambiguity from seeing both objects at test time (after seeing only one object at a time during train). The third test task evaluates zero-shot generalization performance of the policy on linguistic shifts.

Results. For each method, we train on 50 demonstrations sampled from the train distribution for each task, then evaluate on 20 task instances from the test distribution. We visualize the results in Figure 5. From these results, it is clear that the LGA methods strongly outperform GCBC in the multi-task setting, both in the case of state covariate shift and in the case of linguistic utterance shift.

Summary. *Multi-task policies trained with LGA can better resolve task ambiguity compared to GCBC as they are more robust to both observational covariate shift and changes in the linguistic utterance.*

REFERENCES

[1] 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.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations*, 2015.

[3] Andreea Bobu, Marius Wiggert, Claire Tomlin, and Anca D. Dragan. Feature expansive reward learning: Rethinking human input. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '21, page 216–224, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382892. doi: 10.1145/3434073.3444667. URL https://doi.org/10.1145/3434073.3444667.

[4] Andreea Bobu, Marius Wiggert, Claire Tomlin, and Anca D Dragan. Inducing structure in reward learning by learning features. *The International Journal of Robotics Research*, 41(5):497–518, 2022.

[5] Andreea Bobu, Yi Liu, Rohin Shah, Daniel S. Brown, and Anca D. Dragan. SIRL: similarity-based implicit representation learning. *CoRR*, abs/2301.00810, 2023. doi: 10.48550/arXiv.2301.00810. URL https://doi.org/10.48550/arXiv.2301.00810.

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

[7] 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*, pages 17–24, 2012.

[8] Thomas Carta, Pierre-Yves Oudeyer, Olivier Sigaud, and Sylvain Lamprier. Eager: Asking and answering questions for automatic reward shaping in language-guided rl. *Advances in Neural Information Processing Systems*, 35:12478–12490, 2022.

[9] Nick Chater and Paul Vitányi. Simplicity: a unifying principle in cognitive science? *Trends in cognitive sciences*, 7(1):19–22, 2003.

[10] Sahil Chopra, Michael Henry Tessler, and Noah D Goodman. The first crank of the cultural ratchet: Learning and transmitting concepts through language. In *CogSci*, pages 226–232, 2019.

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

[12] 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.

[13] Adam Coates, Andrew Y Ng, et al. Learning feature representations with k-means., 2012.

[14] André Correia and Luís A Alexandre. A survey of demonstration learning. *arXiv preprint arXiv:2303.11191*, 2023.

[15] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models, 2023.

[16] Prasoon Goyal, Scott Niekum, and Raymond J Mooney. Using natural language for reward shaping in reinforcement learning. *arXiv preprint arXiv:1903.02020*, 2019.

[17] Donald Joseph Hejna III and Dorsa Sadigh. Few-shot preference learning for human-in-the-loop rl. In *Conference on Robot Learning*, pages 2014–2025. PMLR, 2023.

[18] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2017.

[19] 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.

[20] Mark K Ho, Jonathan D Cohen, and Tom Griffiths. Rational simplification and rigidity in human planning. *PsyArXiv*, Mar 2023. doi: 10.31234/osf.io/aqxws. URL psyarxiv.com/aqxws.

[21] 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*, 2022.

[22] 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*, 2022.

[23] 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.

[24] 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.

[25] 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.

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

[27] Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

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

[29] Gary Lupyan and Benjamin Bergen. How language programs the mind. *Topics in cognitive science*, 8(2): 408–424, 2016.

[30] William P McCarthy, Robert D Hawkins, Haoliang Wang, Cameron Holdaway, and Judith E Fan. Learning to communicate about shared procedural abstractions. *arXiv preprint arXiv:2107.00077*, 2021.

[31] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3D environments with visual goal prediction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2667–2678, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1287. URL https://aclanthology.org/D18-1287.

[32] Thomas JH Morgan, Natalie T Uomini, Luke E Rendell, Laura Chouinard-Thuly, Sally E Street, Hannah M Lewis, Catherine P Cross, Cara Evans, Ronan Kearney, Ignacio de la Torre, et al. Experimental evidence for the co-evolution of hominin tool-making teaching and language. *Nature communications*, 6(1):6029, 2015.

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

[34] OpenAI. GPT-4 technical report, 2023.

[35] Prashant Parikh. *The use of language*. CSLI Publications Stanford, CA, 2001.

[36] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[37] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[38] 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)*, pages 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.

[39] Theodore R Sumers, Mark K Ho, Robert D Hawkins, Karthik Narasimhan, and Thomas L Griffiths. Learning rewards from linguistic feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6002–6010, 2021.

[40] Theodore R Sumers, Mark K Ho, Robert D Hawkins, and Thomas L Griffiths. Show or tell? exploring when (and why) teaching with language outperforms demonstration. *Cognition*, 232:105326, 2023.

[41] Michael Henry Tessler, Jason Madeano, Pedro A Tsividis, Brin Harper, Noah D Goodman, and Joshua B

Tenenbaum. Learning to solve complex tasks by growing knowledge culturally across generations. *arXiv preprint arXiv:2107.13377*, 2021.

[42] Lucas Tian, Kevin Ellis, Marta Kryven, and Josh Tenenbaum. Learning abstract structure for drawing by efficient motor program induction. *Advances in Neural Information Processing Systems*, 33:2686–2697, 2020.

[43] 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.

APPENDIX

A. Discussion

Limitations. We assume state abstractions for training good policies can be captured by visual features present in the initial state (and therefore, only require querying the LM for task-relevant features at the beginning of each trajectory). An exciting direction for future work would be to learn how to build state abstractions for trajectory-relevant features. We also assume features themselves are expressible in language, i.e. a LM (or human) can understand how each feature may or may not relate to the task. In more complex scenarios, many real-world tasks may be dependent on features that are less expressible in language, e.g. lighting, and are difficult to encapsulate in an abstraction. We are excited to explore additional feature representations (perhaps multimodal visualization interfaces in addition to language representations) that can be jointly used to construct state abstractions.

Broader impact. LGA involves using an LM to guide task abstraction. As in many other machine learning settings, there is a risk that pre-trained models may reflect or even amplify social biases found in their training data. For settings where this and other automation safety risks exist, we suggest using the human-in-the-loop version of our framework (LGA-HILL) to ensure that the LM-suggested features are validated by humans before being deployed in real-world settings.

B. Related Work

Abstraction in Human Learning. There is substantial evidence to suggest much of the flexibility of human learning and planning can be attributed to information filtering of task-relevant features [30, 23, 19, 42, 9]. For example, visual abstractions have been found to prioritize functional properties, i.e. downstream task use, at the expense of visual fidelity, i.e. reconstruction [23]. This suggests that flexibly creating task-conditioned abstractions is important for fast downstream learning, particularly in low-data regimes [33, 6]. Inspired by this, we seek to train neural agents that also flexibly construct and deploy task abstractions.

Language-Aided Task Design. Reinforcement learning from human feedback (RLHF) [11] leverages LMs by fine-tuning reward models on large amounts of human data. Several works [16, 26, 39, 8] instead use LMs to shape or learn reward models by training a policy to complete intermediate or higher-level tasks and asking the LM to annotate resulting policy behavior. In contrast, we instead leverage the rich semantic priors provided by LMs to learn *representations* of the desired task.

LMs for Planning. Large language models, trained on large amounts of text data, contain commonsense information about object properties, functions, and their salience / relevance to various tasks. Past work on leveraging this information for robotics have predominantly focused on using LMs to generate plans or high-level action sequences [38, 1, 21, 22]. Other approaches for policy generation include chaining together a sequence of multimodal LMs [43], using LM probabilities as part of a structured probabilistic model [28]. Aside from generating policies, LMs have also been used in other places in the RL pipeline, including for reward design [26], or for guiding exploration [33, 15].

In this work, we specifically study LMs for *constructing state abstractions*. Furthermore, as natural language is a flexible and intuitive interface for human interaction, we also study how humans can be integrated into our system to help guide the creation of these state abstractions.

Perhaps most similar to our approach is [31], which conditions on language and raw visual observations to create a *binary goal mask*, specifying goal location within an observation. However, our proposed approach is more flexible (potentially allowing for relevant abstract state features that aren't the goal object location) and more personalizable (our approach allow humans to interact with the system, meaning they could potentially refine the representation based on individual preference).

C. Environment Details

States are fully-observable images of dimension $80\times80\times3$ and represent a RGB fixed-camera topdown view of the scene. The action space is continuous of dimension 4 and consists of high-level pick-and-place actions parameterized by the pose of the end effector. There are 4 possible objects that can be spawned: the target (manipulated) object, goal object, and (up to) two possible distractors. There are two types of target features (object type and texture) and 29 possible instantiations of object type and 81 instantiations of texture. Visual depictions of the features can be found in the user study below.

D. Task Details

We provide details regarding all tasks, including ground truth task distributions and full LM prompt.

red heart:
- Task prompt: *Bring me the red heart.*
- True distribution: {objects: heart}, {textures: red, dark red, dark red swirl, red paisley}

heart:
- Task prompt: *Bring me the heart.*
- True distribution: {objects: heart}, {textures: ALL}

tiger-colored object:
- Task prompt: *Bring me the tiger-colored object.*
- True distribution: {objects: ALL}, {textures: tiger}

letter from the word letter:
- Task prompt: *Bring me a letter from the word 'letter'.*
- True distribution: {objects: letter E, letter R, letter T}, {textures: ALL}

consonant with warm-color:
- Task prompt: *Bring me a consonant with a warm color on it.*
- True distribution: {objects: letter G, letter M, letter R, letter T, letter V}, {textures: dark {red—yellow—pink—orange}, dark {red—yellow—pink—orange} and * stripe,

{red—yellow—pink—orange} and * polka dot, dark {red—yellow—pink—orange} and * polka dot, {red—yellow—pink} swirl, dark {red—yellow—pink} swirl, {red—yellow—pink} paisley, tiger, magma, wooden, rainbow, tiles, brick}

vowel with multiple colors:

- Task prompt: *Bring me a vowel with multiple colors on it.*
- True distribution: {objects: letter A, letter E}, {textures: polka dot, tiles, checkerboard, plastic, tiger, magma, rainbow, * and * stripe, * and * polka dot, * swirl, * paisley}

something to drink water out of:

- Task prompt: *Bring me something to drink water out of.*
- True distribution: {objects: bowl, pan, container}, {textures: ALL}

something from a typical kitchen:

- Task prompt: *Bring me something from a typical kitchen.*
- True distribution: {objects: bowl, pan, container}, {textures: ALL}

1) Full Prompt: ChatGPT models (including GPT4) can take in both system prompts and user prompts. We split our prompt into these two parts as follows.

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 pick up objects based on some linguistic command (e.g. "pick up 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 the target object can have based on the given command. Target 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}

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".

E. Architecture and Training Details

Architecture.

For GCBC, Sentence-BERT processes a goal utterance in natural language into an embedding of size 384, which we additionally process through a linear MLP of output size 100. We process the state using a standard Conv2D block consisting of 3 stacked Conv2D layers of output channel sizes 32, 64, 32, and strides 4, 2, and 1. Each output layer is processed by BatchNorm2D as well as a ReLU activation. After the last Conv2D layer, we flatten the output and concatenate with the output of the goal utterance. We feed the concatenated output through a last linear layer for action prediction.

For LGA, we process both the state and the state abstraction through dual Conv2Ds blocks like above. We concatenate the output and feed through a last linear layer for action prediction.

Training.

We train all networks to convergence for a maximum of 300 epochs. All computation was done on two NVIDIA GeForce RTX 3090 GPUs. Rollouts were rendered locally using PyBullet on a Macbook Pro.

F. User Study

In the following pages, we have included the full user study shared with participants. Following standard user study procedure, we initial briefed users by telling them how long the study was (roughly 30 mins) and that they were free to leave at anytime. Demographic information was collected in person. We showed participants the first pdf during the familiarization phase to introduce them to the environment and full feature list. We then randomly chose to begin with either the human-only task (second pdf) or the LM-aided task (third pdf). After the study, users were debriefed and given the email of the study designer to contact if they had any questions.

User Study: Instructions and Dictionary

You are helping teach a robot which properties are important to a desired task.

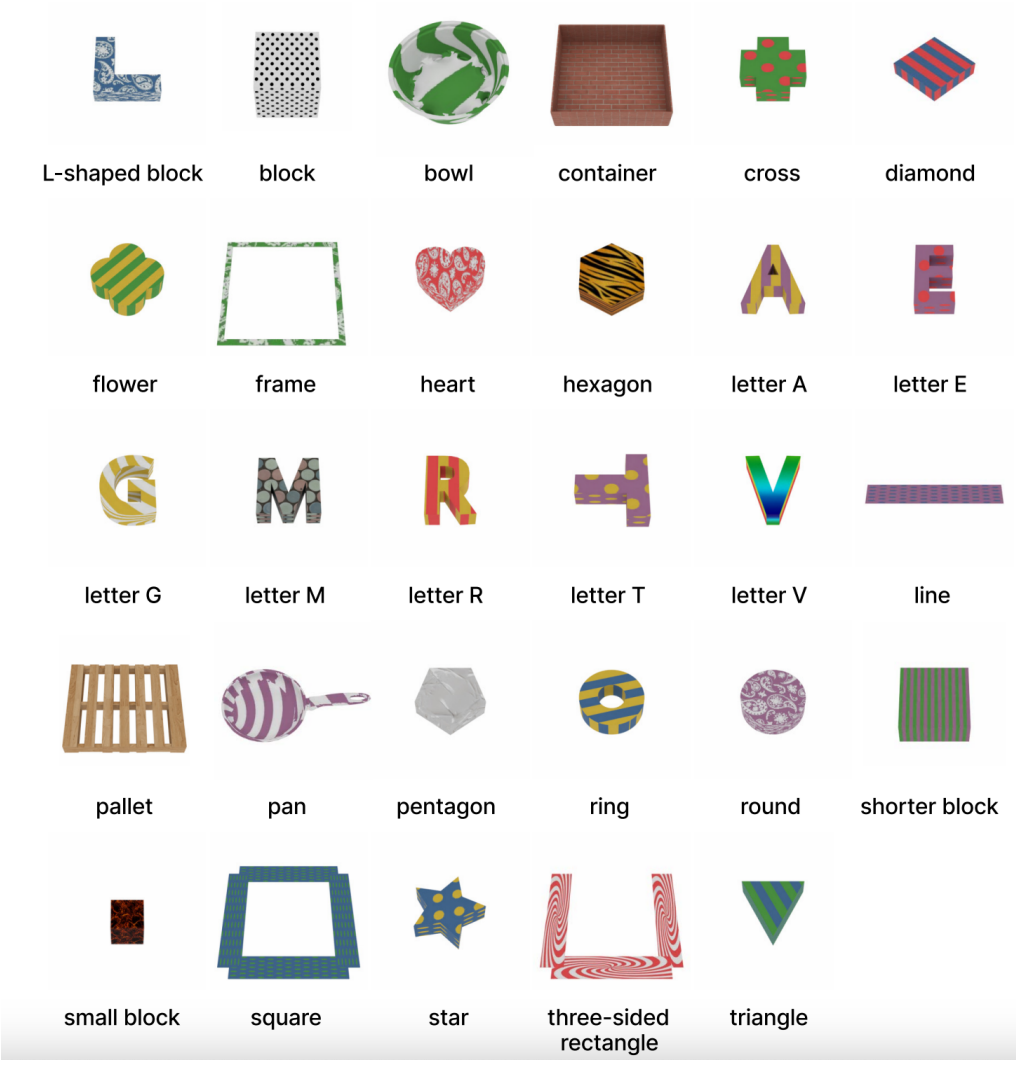
The robot is learning to pick up objects based on some linguistic command (e.g. "pick up red bowl").

At each interaction, we will specify the task that you need to teach the robot. In order to teach the robot, you will need to help design the task by specifying what properties the target object can have based on the given command.

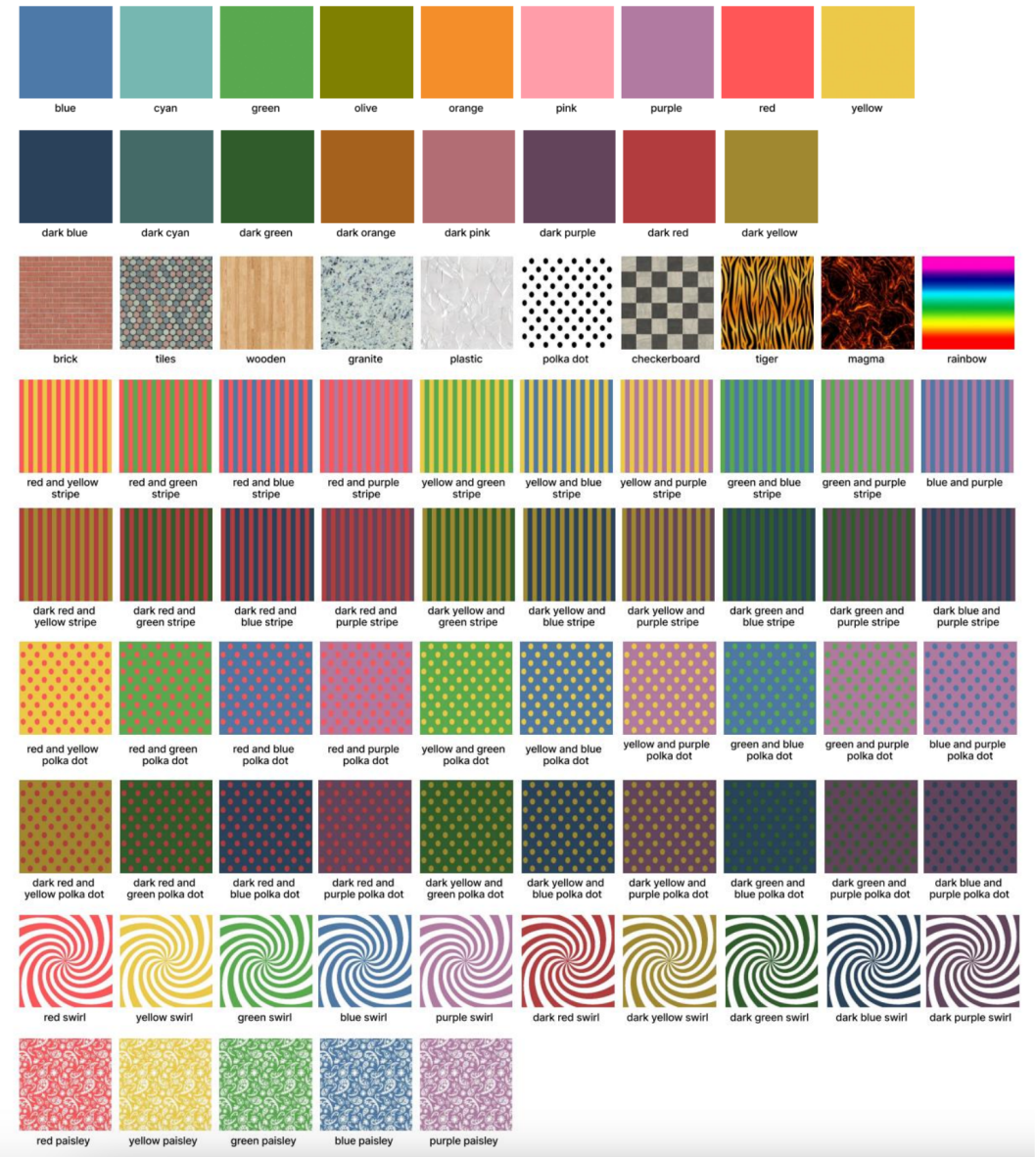
Target objects in this environment have two properties: [*object type, object color*].

Note: 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 colors:



Example task: "Bring me polka dot geometric shape"

In [3]:
```
task_kwargs = {
     'possible_dragged_obj': ['diamond',
                              'triangle',
                              'hexagon',
                              'pentagon',
                              'square'],
     'possible_dragged_obj_texture': ['red and yellow polka dot',
                                      'red and green polka dot',
                                      'red and blue polka dot',
                                      'red and purple polka dot',
                                      'yellow and green polka dot',
                                      'yellow and blue polka dot',
                                      'yellow and purple polka dot',
                                      'green and blue polka dot',
                                      'green and purple polka dot',
                                      'blue and purple polka dot',
                                      'dark red and yellow polka dot',
```

```
                                        'dark red and green polka dot',
                                        'dark red and blue polka dot',
                                        'dark red and purple polka dot',
                                        'dark yellow and green polka dot',
                                        'dark yellow and blue polka dot',
                                        'dark yellow and purple polka dot',
                                        'dark green and blue polka dot',
                                        'dark green and purple polka dot',
                                        'dark blue and purple polka dot',]}
```

Experiment

For your reference, here are the object types and colors (feel free to copy + paste):

In [2]:
```
'possible_dragged_obj': ['L-shaped block',
                         'block',
                         'bowl',
                         'container',
                         'cross',
                         'diamond',
                         'flower',
                         'frame',
                         'heart',
                         'hexagon',
                         'letter A',
                         'letter E',
                         'letter G',
                         'letter M',
                         'letter R',
                         'letter T',
                         'letter V',
                         'line',
                         'pallet',
                         'pan',
                         'pentagon',
                         'ring',
                         'round',
                         'shorter block',
                         'small block',
                         'square',
                         'star',
                         'three-sided rectangle',
                         'triangle']
```

In [ ]:
```
'possible_dragged_obj_texture': ['brick',
                                 'tiles',
                                 'wooden',
                                 'granite',
                                 'plastic',
                                 'polka dot',
                                 'checkerboard',
                                 'tiger',
                                 'magma',
                                 'rainbow',
                                 'blue',
                                 'cyan',
                                 'green',
                                 'olive',
                                 'orange',
                                 'pink',
                                 'purple',
                                 'red',
                                 'yellow',
                                 'dark blue',
                                 'dark cyan',
                                 'dark green',
                                 'dark olive',
                                 'dark orange',
                                 'dark pink',
```

```
                                'dark purple',
                                'dark red',
                                'dark yellow',
                                'red and yellow stripe',
                                'red and green stripe',
                                'red and blue stripe',
                                'red and purple stripe',
                                'yellow and green stripe',
                                'yellow and blue stripe',
                                'yellow and purple stripe',
                                'green and blue stripe',
                                'green and purple stripe',
                                'blue and purple stripe',
                                'dark red and yellow stripe',
                                'dark red and green stripe',
                                'dark red and blue stripe',
                                'dark red and purple stripe',
                                'dark yellow and green stripe',
                                'dark yellow and blue stripe',
                                'dark yellow and purple stripe',
                                'dark green and blue stripe',
                                'dark green and purple stripe',
                                'dark blue and purple stripe',
                                'red and yellow polka dot',
                                'red and green polka dot',
                                'red and blue polka dot',
                                'red and purple polka dot',
                                'yellow and green polka dot',
                                'yellow and blue polka dot',
                                'yellow and purple polka dot',
                                'green and blue polka dot',
                                'green and purple polka dot',
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']
```

Task 1: "Bring me the red heart"

In [4]:

```
task_kwargs = {
     'possible_dragged_obj': [],
```

```
    'possible_dragged_obj_texture': []}
```

Task 2: "Bring me the heart"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 3: "Bring me the tiger object"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 4: "Bring me a letter from the word letter"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 5: "Bring me a consonant that has any warm color on it"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 6: "Bring me a vowel that has multiple colors on it" (white is not a color)

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 7: "Bring me a letter. If there are multiple, bring the one that comes earliest in the alphabet."

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 8: "Bring me something I can drink water out of"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Task 9: "Bring me something I can find in a typical kitchen"

In [ ]:
```
task_kwargs = {
    'possible_dragged_obj': [],
    'possible_dragged_obj_texture': []}
```

Experiment

For your reference, here are the object types and colors (feel free to copy + paste).

A large language model (GPT-4) has given its answers to help you with the task (they are pre-filled in). You may feel free to change/modify the answers, or accept them without change:

In [ ]:
```
'possible_dragged_obj': ['L-shaped block',
                         'block',
                         'bowl',
                         'container',
                         'cross',
                         'diamond',
                         'flower',
                         'frame',
                         'heart',
                         'hexagon',
                         'letter A',
                         'letter E',
                         'letter G',
                         'letter M',
                         'letter R',
                         'letter T',
                         'letter V',
                         'line',
                         'pallet',
                         'pan',
                         'pentagon',
                         'ring',
                         'round',
                         'shorter block',
                         'small block',
                         'square',
                         'star',
                         'three-sided rectangle',
                         'triangle']
```

In [ ]:
```
'possible_dragged_obj_texture': ['brick',
                                 'tiles',
                                 'wooden',
                                 'granite',
                                 'plastic',
                                 'polka dot',
                                 'checkerboard',
                                 'tiger',
                                 'magma',
                                 'rainbow',
                                 'blue',
                                 'cyan',
                                 'green',
                                 'olive',
                                 'orange',
                                 'pink',
                                 'purple',
                                 'red',
                                 'yellow',
                                 'dark blue',
                                 'dark cyan',
```

```
                                'dark green',
                                'dark olive',
                                'dark orange',
                                'dark pink',
                                'dark purple',
                                'dark red',
                                'dark yellow',
                                'red and yellow stripe',
                                'red and green stripe',
                                'red and blue stripe',
                                'red and purple stripe',
                                'yellow and green stripe',
                                'yellow and blue stripe',
                                'yellow and purple stripe',
                                'green and blue stripe',
                                'green and purple stripe',
                                'blue and purple stripe',
                                'dark red and yellow stripe',
                                'dark red and green stripe',
                                'dark red and blue stripe',
                                'dark red and purple stripe',
                                'dark yellow and green stripe',
                                'dark yellow and blue stripe',
                                'dark yellow and purple stripe',
                                'dark green and blue stripe',
                                'dark green and purple stripe',
                                'dark blue and purple stripe',
                                'red and yellow polka dot',
                                'red and green polka dot',
                                'red and blue polka dot',
                                'red and purple polka dot',
                                'yellow and green polka dot',
                                'yellow and blue polka dot',
                                'yellow and purple polka dot',
                                'green and blue polka dot',
                                'green and purple polka dot',
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']
```

Task 1: "Bring me the red heart"

In [ ]:
```
task_kwargs = {
     'possible_dragged_obj': ['heart'],
     'possible_dragged_obj_texture': ['red',
                                      'dark red',
                                      'red swirl',
                                      'dark red swirl'
                                      'red paisley']}
```

Task 2: "Bring me the heart"

In [ ]:
```
task_kwargs = {
     'possible_dragged_obj': ['heart'],
     'possible_dragged_obj_texture': ['brick',
                                      'tiles',
                                      'wooden',
                                      'granite',
                                      'plastic',
                                      'polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'blue',
                                      'cyan',
                                      'green',
                                      'olive',
                                      'orange',
                                      'pink',
                                      'purple',
                                      'red',
                                      'yellow',
                                      'dark blue',
                                      'dark cyan',
                                      'dark green',
                                      'dark olive',
                                      'dark orange',
                                      'dark pink',
                                      'dark purple',
                                      'dark red',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'red and green stripe',
                                      'red and blue stripe',
                                      'red and purple stripe',
                                      'yellow and green stripe',
                                      'yellow and blue stripe',
                                      'yellow and purple stripe',
                                      'green and blue stripe',
                                      'green and purple stripe',
                                      'blue and purple stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark red and purple stripe',
                                      'dark yellow and green stripe',
                                      'dark yellow and blue stripe',
                                      'dark yellow and purple stripe',
                                      'dark green and blue stripe',
                                      'dark green and purple stripe',
                                      'dark blue and purple stripe',
```

```
                                'red and yellow polka dot',
                                'red and green polka dot',
                                'red and blue polka dot',
                                'red and purple polka dot',
                                'yellow and green polka dot',
                                'yellow and blue polka dot',
                                'yellow and purple polka dot',
                                'green and blue polka dot',
                                'green and purple polka dot',
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']}
```

Task 3: "Bring me the tiger object"

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['L-shaped block',
                              'block',
                              'bowl',
                              'container',
                              'cross',
                              'diamond',
                              'flower',
                              'frame',
                              'heart',
                              'hexagon',
                              'letter A',
                              'letter E',
                              'letter G',
                              'letter M',
                              'letter R',
                              'letter T',
                              'letter V',
                              'line',
                              'pallet',
                              'pan',
                              'pentagon',
                              'ring',
                              'round',
                              'shorter block',
```

```
                           'small block',
                           'square',
                           'star',
                           'three-sided rectangle',
                           'triangle'],
     'possible_dragged_obj_texture': ['tiger']}
```

Task 4: "Bring me a letter from the word letter"

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['letter E'
                              'letter R',
                              'letter T'],
     'possible_dragged_obj_texture': ['brick',
                                      'tiles',
                                      'wooden',
                                      'granite',
                                      'plastic',
                                      'polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'blue',
                                      'cyan',
                                      'green',
                                      'olive',
                                      'orange',
                                      'pink',
                                      'purple',
                                      'red',
                                      'yellow',
                                      'dark blue',
                                      'dark cyan',
                                      'dark green',
                                      'dark olive',
                                      'dark orange',
                                      'dark pink',
                                      'dark purple',
                                      'dark red',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'red and green stripe',
                                      'red and blue stripe',
                                      'red and purple stripe',
                                      'yellow and green stripe',
                                      'yellow and blue stripe',
                                      'yellow and purple stripe',
                                      'green and blue stripe',
                                      'green and purple stripe',
                                      'blue and purple stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark red and purple stripe',
                                      'dark yellow and green stripe',
                                      'dark yellow and blue stripe',
                                      'dark yellow and purple stripe',
                                      'dark green and blue stripe',
                                      'dark green and purple stripe',
                                      'dark blue and purple stripe',
                                      'red and yellow polka dot',
                                      'red and green polka dot',
```

```
                                'red and blue polka dot',
                                'red and purple polka dot',
                                'yellow and green polka dot',
                                'yellow and blue polka dot',
                                'yellow and purple polka dot',
                                'green and blue polka dot',
                                'green and purple polka dot',
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']}
```

Task 5: "Bring me a consonant that has any warm color on it"

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['letter G',
                              'letter M',
                              'letter R',
                              'letter T',
                              'letter V'],
     'possible_dragged_obj_texture': ['polka dot',
                                      'orange',
                                      'pink',
                                      'red',
                                      'yellow',
                                      'dark orange',
                                      'dark pink',
                                      'dark red',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark yellow and purple stripe',
                                      'red and yellow polka dot',
                                      'dark red and yellow polka dot',
                                      'red swirl',
                                      'yellow swirl',
                                      'dark red swirl',
                                      'dark yellow swirl',
```

```
                                    'red paisley',
                                    'yellow paisley']}
```

Task 6: "Bring me a vowel that has multiple colors on it" (white is not a color)

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['letter A',
                              'letter E'],
     'possible_dragged_obj_texture': ['polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'yellow',
                                      'dark green',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'red and green stripe',
                                      'red and blue stripe',
                                      'red and purple stripe',
                                      'yellow and green stripe',
                                      'yellow and blue stripe',
                                      'yellow and purple stripe',
                                      'green and blue stripe',
                                      'green and purple stripe',
                                      'blue and purple stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark red and purple stripe',
                                      'dark yellow and green stripe',
                                      'dark yellow and blue stripe',
                                      'dark yellow and purple stripe',
                                      'dark green and blue stripe',
                                      'dark green and purple stripe',
                                      'dark blue and purple stripe',
                                      'red and yellow polka dot',
                                      'red and green polka dot',
                                      'red and blue polka dot',
                                      'red and purple polka dot',
                                      'yellow and green polka dot',
                                      'yellow and blue polka dot',
                                      'yellow and purple polka dot',
                                      'green and blue polka dot',
                                      'green and purple polka dot',
                                      'blue and purple polka dot',
                                      'dark red and yellow polka dot',
                                      'dark red and green polka dot',
                                      'dark red and blue polka dot',
                                      'dark red and purple polka dot',
                                      'dark yellow and green polka dot',
                                      'dark yellow and blue polka dot',
                                      'dark yellow and purple polka dot',
                                      'dark green and blue polka dot',
                                      'dark green and purple polka dot',
                                      'dark blue and purple polka dot',
                                      'yellow swirl',
                                      'green swirl',
                                      'blue swirl',
                                      'dark yellow swirl',
                                      'dark green swirl',
                                      'dark blue swirl',
```

```
                                    'dark purple swirl',
                                    'red paisley',
                                    'green paisley',
                                    'blue paisley',
                                    'purple paisley']}
```

Task 7: "Bring me a letter. If there are multiple, bring the one that comes earliest in the alphabet."

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['letter A',
                              'letter E',
                              'letter G',
                              'letter M',
                              'letter R',
                              'letter T'],
     'possible_dragged_obj_texture': ['brick',
                                      'tiles',
                                      'wooden',
                                      'granite',
                                      'plastic',
                                      'polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'blue',
                                      'cyan',
                                      'green',
                                      'olive',
                                      'orange',
                                      'pink',
                                      'purple',
                                      'red',
                                      'yellow',
                                      'dark blue',
                                      'dark cyan',
                                      'dark green',
                                      'dark olive',
                                      'dark orange',
                                      'dark pink',
                                      'dark purple',
                                      'dark red',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'red and green stripe',
                                      'red and blue stripe',
                                      'red and purple stripe',
                                      'yellow and green stripe',
                                      'yellow and blue stripe',
                                      'yellow and purple stripe',
                                      'green and blue stripe',
                                      'green and purple stripe',
                                      'blue and purple stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark red and purple stripe',
                                      'dark yellow and green stripe',
                                      'dark yellow and blue stripe',
                                      'dark yellow and purple stripe',
                                      'dark green and blue stripe',
                                      'dark green and purple stripe',
```

```
                                    'dark blue and purple stripe',
                                    'red and yellow polka dot',
                                    'red and green polka dot',
                                    'red and blue polka dot',
                                    'red and purple polka dot',
                                    'yellow and green polka dot',
                                    'yellow and blue polka dot',
                                    'yellow and purple polka dot',
                                    'green and blue polka dot',
                                    'green and purple polka dot',
                                    'blue and purple polka dot',
                                    'dark red and yellow polka dot',
                                    'dark red and green polka dot',
                                    'dark red and blue polka dot',
                                    'dark red and purple polka dot',
                                    'dark yellow and green polka dot',
                                    'dark yellow and blue polka dot',
                                    'dark yellow and purple polka dot',
                                    'dark green and blue polka dot',
                                    'dark green and purple polka dot',
                                    'dark blue and purple polka dot',
                                    'red swirl',
                                    'yellow swirl',
                                    'green swirl',
                                    'blue swirl',
                                    'purple swirl',
                                    'dark red swirl',
                                    'dark yellow swirl',
                                    'dark green swirl',
                                    'dark blue swirl',
                                    'dark purple swirl',
                                    'red paisley',
                                    'yellow paisley',
                                    'green paisley',
                                    'blue paisley',
                                    'purple paisley']}
```

Task 8: "Bring me something I can drink water out of"

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['bowl',
                              'container'],
     'possible_dragged_obj_texture': ['brick',
                                      'tiles',
                                      'wooden',
                                      'granite',
                                      'plastic',
                                      'polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'blue',
                                      'cyan',
                                      'green',
                                      'olive',
                                      'orange',
                                      'pink',
                                      'purple',
                                      'red',
                                      'yellow',
                                      'dark blue',
                                      'dark cyan',
```

```
                                'dark green',
                                'dark olive',
                                'dark orange',
                                'dark pink',
                                'dark purple',
                                'dark red',
                                'dark yellow',
                                'red and yellow stripe',
                                'red and green stripe',
                                'red and blue stripe',
                                'red and purple stripe',
                                'yellow and green stripe',
                                'yellow and blue stripe',
                                'yellow and purple stripe',
                                'green and blue stripe',
                                'green and purple stripe',
                                'blue and purple stripe',
                                'dark red and yellow stripe',
                                'dark red and green stripe',
                                'dark red and blue stripe',
                                'dark red and purple stripe',
                                'dark yellow and green stripe',
                                'dark yellow and blue stripe',
                                'dark yellow and purple stripe',
                                'dark green and blue stripe',
                                'dark green and purple stripe',
                                'dark blue and purple stripe',
                                'red and yellow polka dot',
                                'red and green polka dot',
                                'red and blue polka dot',
                                'red and purple polka dot',
                                'yellow and green polka dot',
                                'yellow and blue polka dot',
                                'yellow and purple polka dot',
                                'green and blue polka dot',
                                'green and purple polka dot',
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']}
```

Task 9: "Bring me something I can find in a typical kitchen"

In [ ]:

```
task_kwargs = {
     'possible_dragged_obj': ['bowl',
                              'container',
                              'pan'],
     'possible_dragged_obj_texture': ['brick',
                                      'tiles',
                                      'wooden',
                                      'granite',
                                      'plastic',
                                      'polka dot',
                                      'checkerboard',
                                      'tiger',
                                      'magma',
                                      'rainbow',
                                      'blue',
                                      'cyan',
                                      'green',
                                      'olive',
                                      'orange',
                                      'pink',
                                      'purple',
                                      'red',
                                      'yellow',
                                      'dark blue',
                                      'dark cyan',
                                      'dark green',
                                      'dark olive',
                                      'dark orange',
                                      'dark pink',
                                      'dark purple',
                                      'dark red',
                                      'dark yellow',
                                      'red and yellow stripe',
                                      'red and green stripe',
                                      'red and blue stripe',
                                      'red and purple stripe',
                                      'yellow and green stripe',
                                      'yellow and blue stripe',
                                      'yellow and purple stripe',
                                      'green and blue stripe',
                                      'green and purple stripe',
                                      'blue and purple stripe',
                                      'dark red and yellow stripe',
                                      'dark red and green stripe',
                                      'dark red and blue stripe',
                                      'dark red and purple stripe',
                                      'dark yellow and green stripe',
                                      'dark yellow and blue stripe',
                                      'dark yellow and purple stripe',
                                      'dark green and blue stripe',
                                      'dark green and purple stripe',
                                      'dark blue and purple stripe',
                                      'red and yellow polka dot',
                                      'red and green polka dot',
                                      'red and blue polka dot',
                                      'red and purple polka dot',
                                      'yellow and green polka dot',
                                      'yellow and blue polka dot',
                                      'yellow and purple polka dot',
                                      'green and blue polka dot',
                                      'green and purple polka dot',
```

```
                                'blue and purple polka dot',
                                'dark red and yellow polka dot',
                                'dark red and green polka dot',
                                'dark red and blue polka dot',
                                'dark red and purple polka dot',
                                'dark yellow and green polka dot',
                                'dark yellow and blue polka dot',
                                'dark yellow and purple polka dot',
                                'dark green and blue polka dot',
                                'dark green and purple polka dot',
                                'dark blue and purple polka dot',
                                'red swirl',
                                'yellow swirl',
                                'green swirl',
                                'blue swirl',
                                'purple swirl',
                                'dark red swirl',
                                'dark yellow swirl',
                                'dark green swirl',
                                'dark blue swirl',
                                'dark purple swirl',
                                'red paisley',
                                'yellow paisley',
                                'green paisley',
                                'blue paisley',
                                'purple paisley']}
```