# Learning Perceptual Concepts by Bootstrapping from Human Queries

Andreea Bobu[1], Chris Paxton[2], Wei Yang[2], Balakumar Sundaralingam[2],
Yu-Wei Chao[2], Maya Cakmak[2,3], and Dieter Fox[2,3]

***Abstract*— Robots need to be able to learn *concepts* from their users in order to adapt their capabilities to each user's unique task. But when the robot operates on high-dimensional inputs, like images or point clouds, this is impractical: the robot needs an unrealistic amount of human effort to learn the new concept. To address this challenge, we propose a new approach whereby the robot learns a low-dimensional variant of the concept and uses it to generate a larger data set for learning the concept in the high-dimensional space. This lets it take advantage of semantically meaningful *privileged information* only accessible at training time, like object poses and bounding boxes, that allows for richer human interaction to speed up learning. We evaluate our approach by learning prepositional concepts that describe object state or multi-object relationships, like *above*, *near*, or *aligned*, which are key to user specification of task goals and execution constraints for robots. Using a simulated human, we show that our approach improves sample complexity when compared to learning concepts directly in the high-dimensional space. We also demonstrate the utility of the learned concepts in motion planning tasks on a 7-DoF Franka robot.**

## I. Introduction

As robots are increasingly interfacing with human environments, they have to be able to adapt their task execution to assist the people they interact with. This necessitates teaching the robot new skills and *concepts* on the fly. For example, in the scenario in Fig. 1, the user wants the robot to place the mug near the plate. However, for the robot to be helpful, the person has to first teach it what the concept of *near* means.

In order for the robot to then use such concepts in a task, they need to be compatible with its motion planning and execution pipeline. In most modern robotics applications, this requires operating on high-dimensional input spaces (e.g. images, point clouds, etc.) [1], [2], [3], [4], so, consequently, the robot should learn high-dimensional concepts capable of supporting them. Recent work in deep learning suggests it could do so by directly asking a human teacher for labeled examples demonstrating the concept [1]. Unfortunately, because of the high dimensionality of the space, the robot would need a large and diverse enough dataset of human labels, making it impractical to have a novice user teach a new concept.

In this work, we observe that, while the robot is constrained to high-dimensional inputs during task execution, that restriction can be relaxed at training time. In particular, we give the robot access to *privileged information* in the form of a low-dimensional input space that should capture the desired concept with much less human input. In the example in Fig. 1, this privileged information could be the two object poses – a much simpler representation than their point cloud equivalent. Our insight is that instead of learning the concept from high-dimensional data from the get-go, the robot should first learn the concept on the privileged space, then use it as the labeler in place of the human. This allows the robot to generate large amounts of data for training the concept in the original high-dimensional space without needing additional human input. As these low-dimensional spaces are often semantically meaningful, this approach also allows for richer human interaction, such as directly asking if a dimension is relevant for the concept, that can further accelerate learning.

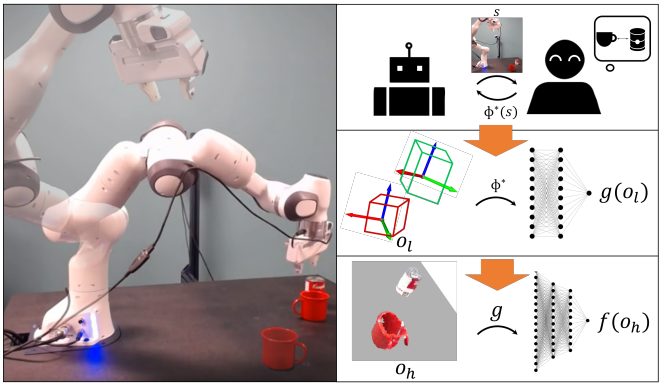


Fig. 1: (Left) The robot moves the cup to be *near* the can using our learned concept. (Right). We propose a new learning approach whereby the robot collects queries about the concept from the human (top), learns a low-dimensional concept $g$ on the *privileged information* space (e.g. poses and bounding boxes) (middle), then uses it to label data necessary for learning the high-dimensional concept $f$ (bottom). Additional qualitative results available at https://sites.google.com/nvidia.com/active-concept-learning.

Our contributions are three-fold: (1) we investigate three types of human input that can be used to quickly learn a concept on the privileged space – demonstration, label, and feature queries; (2) we show that these concepts can label large amounts of simulator-synthesized high-dimensional data, resulting in 9 higher quality spatial concepts than the baseline; (3) we test our concepts on a 7DoF Franka Panda robot in a pick-and-place task with real data. While our evaluation focuses on prepositional concepts, which are key to user specification of robot task goals, our method can be extended to any concepts where privileged information is available.

[1] EECS at UC Berkeley abobu@berkeley.edu
[2] NVIDIA Robotics, USA {cpaxton, weiy, balakumars, ychao, dieterf}@nvidia.com
[3] University of Washington mcakmak@cs.washington.edu
We thank Weiyu Liu for an aligned version of the Shapenet dataset.

## II. Related Work

Traditionally, concepts are hand-engineered by the system designer [5], [6], [7], [8], [9] prior to robot deployment. Unfortunately, because they are hand-specified, these concepts are practically limited to low-dimensional input spaces, like robot and object poses, that the designer can understand and work with. Moreover, since they are defined *a priori*, the robot has no way of learning new concepts that are specific to the end-user's needs. Recent methods attempt to address this by either inferring concepts directly from task demonstrations [10], [11], [12], [13] or learning them from human input [14]; however, these methods too have been demonstrated only on low-dimensional input spaces.

As an alternative to concept learning or specification, deep learning approaches in inverse reinforcement learning and imitation learning attempt to automatically extract the concepts implicitly from expert human demonstrations [2], [15], [16]. These methods can perform well on the training distribution even when the input space is high-dimensional (e.g. images, pointclouds, etc.), but they require a large amount of diverse data from the human in order to generalize [17], [3].

Instead of learning the concept implicitly, other work learns object semantic relationships directly from point cloud data [1], [18]. The disadvantage of this approach is that it still requires large amounts of data, making it unsuitable for learning the concept from a human. We look at how we can quickly and efficiently teach similar high-dimensional spatial concepts that the robot can use for planning.

## III. Method

Our goal is to learn concepts that are useful for robot manipulation tasks in high-dimensional input spaces, like segmented point clouds. We assume that the robot may query a human for labeled examples about the concept. However, we wish to learn these concepts as efficiently as possible, minimizing human effort as much as possible.

Unfortunately, training high-dimensional concepts is typically data intensive [1], [18]. Instead of learning the concept directly from this high-dimensional input, we propose to learn it on a simpler, lower dimensional space, like object poses or bounding boxes. We can then use this low-dimensional variant to label as much randomly-generated data as needed to train the concept in the original high-dimensional space.

### A. Preliminaries

Formally, a concept is an arbitrary complex mapping over states, $\phi(s) : \mathbb{R}^d \to [0, 1]$, representing how much concept $\phi$ is expressed at state $s \in \mathbb{R}^d$. In our setting, we assume that the human teacher already knows the ground truth concept $\phi^*$, and can, therefore, answer queries about it.

While at training time the robot has access to the entire state $s$, at test time it only receives high-dimensional observations $o_h \in \mathbb{R}^h$ given by a transformation of the state representation $\mathcal{F}(s) : \mathbb{R}^d \to \mathbb{R}^h$. For example, in Fig. 1, $s$ captures the objects' and robot's pose, mesh, color, etc, whereas $o_h$ is the segmented point cloud of the scene from a fixed camera view. The robot seeks to learn a high-dimensional concept mapping over these observations $\phi_h(o_h) : \mathbb{R}^h \to [0, 1]$, so that it can use it in manipulation tasks later on.

To do so, we assume the robot can ask the person for state-label examples $(s, \phi^*(s))$, forming a dataset $\{s, o_h, \phi^*(s)\} \in \mathcal{D}_\phi$. Since the high-dimensional observation $o_h$ directly corresponds to state $s$, this dataset has the crucial property that the same label $\phi^*(s)$ applies to both $s$ and $o_h$:

$$\phi^*(s) = \phi_h(o_h), \forall s, o_h = \mathcal{F}(s) \ . \tag{1}$$

From here, one natural idea to learn $\phi_h$ is to treat it as a classification or regression problem and directly perform supervised learning on $(o_h, \phi^*(s))$ pairs. Unfortunately, to learn anything useful, this approach would require very large amounts of data from the person, making it impractical to have a user teach a new concept.

Instead, we assume the robot can use *privileged information* in the form of a low-dimensional observation $o_l \in \mathbb{R}^l$, as given by a transformation $\mathcal{G}(s) : \mathbb{R}^d \to \mathbb{R}^l$. We think of this information as privileged because the robot has access to it during training but not at task execution time. For instance, in Fig. 1, $o_l$ only needs the object poses to determine whether one object is near the other. The set of collected human examples then includes the low-dimensional observation: $\{s, o_l, o_h, \phi^*(s)\} \in \mathcal{D}_\phi$, which allows the robot to learn a low-dimensional variant of the concept, $\phi_l(o_l) : \mathbb{R}^l \to [0, 1]$, by extending the property in Eq. (1):

$$\phi^*(s) = \phi_h(o_h) = \phi_l(o_l), \forall s, o_h = \mathcal{F}(s), o_l = \mathcal{G}(s) \ . \tag{2}$$

We hypothesize that learning the low-dimensional concept $\phi_l$ on top of the privileged information should require less human input than learning $\phi_h$ from the get-go. Furthermore, Eq. 2 allows the learned $\phi_l$ to act as a labeler, bypassing the need for additional human input. As such, we break down the concept learning problem into two steps: leverage the human queries to learn a low-dimensional concept $\phi_l$, then use it to ultimately learn the original high-dimensional $\phi_h$.

### B. Learning a low-dimensional concept

To learn $\phi_l$, the robot first needs to query the human for $\mathcal{D}_\phi$. To ensure the robot can learn the concept with little data, we want a query collection strategy that balances being informative and not placing too much burden on the human. We, thus, consider two types of input that are easy to provide and commonly used in the HRI literature [19]: *demonstration queries* and *label queries*. Since users may struggle to label continuous values, we simplify the labeling scheme to consist of 0 (negative) or 1 (positive) for low and high concept values. Note that despite the labels being discrete, they can still be used to learn a model that predicts continuous values.

*Demonstration queries*, or *demo queries*, involve creating a new scenario and asking the human to select states $s$ that demonstrate the concept and label them according to $\phi^*$. For example, for the concept in Fig. 1, the person could place the mug near the plate and label the state 1.0, symbolizing a high concept value. Here, the robot can only manipulate the constraints of the scenario (e.g. which objects are involved) and the human has complete control over the selection of

the rest of the state. This method, thus, requires an interface that allows the person to directly manipulate the state of the environment and label it.

Under the assumption of a pedagogic human, demonstration queries provide the robot with an informative dataset of examples that should allow it to learn the low-dimensional concept quickly. Unfortunately, this data collection method can be quite slow due to the fact that the person has to spend time both deciding on an informative state and manipulating the environment to reach it. This makes it challenging to use in data intensive regimes (like when training $\phi_h$ from the get-go) but ideal in the low-data ones we are interested in.

*Label queries* are a less time-consuming alternative where the robot synthesizes the full query state $s$, and the person simply has to label it. This type of query is much easier and faster for the person to answer, but places the burden of informative state generation entirely on the robot. Importantly, simply randomly sampling the state space might not result in the most informative dataset for the concept of interest. For example, for a concept like *above*, placing the objects at random locations in the scene will rarely result in examples where the two are above one another. For this reason, we additionally employ an active learning [20], [21] approach to aid the robot in selecting more useful queries.

Following the batch active learning framework [22], the robot interleaves asking for queries with learning the low-level concept $\phi_l^t$ from the $t$ examples received so far. This way, the robot can use the partially-learned $\phi_l^t$ to inform the synthesis of a more useful batch of queries. We enable the robot to choose among three query synthesis strategies: 1. *random*: randomly generate a state $s \in \mathbb{R}^d$; 2. *confusion*: pick the state that maximizes confusion, or, in other words, is at $\phi_l^t(s)$'s decision boundary, i.e. $s = \arg\min_s(\|\phi_l^t(s) - 0.5\|)$; 3. *augment*: select a state that was previously labeled as a positive (or negative, whichever is rarer) and add noise to it.

A *random* query serves as a proxy for exploring novel areas of the state space, and we generate it by randomizing all the parameters of the state in a simulator (e.g. object meshes, poses, etc). The *confusion* query disambiguates areas of the state space where the current concept $\phi_l^t$ has high uncertainty, and we optimize it using the cross-entropy method [23], [24]. The *augment* query is useful for concepts where positives (or negatives) are rarer, like in the *above* example.

Focusing on the low-dimensional concept first offers us the benefit of a richer human interaction via active learning, which would be difficult to achieve with the high-dimensional variant because of its longer training cycles. Another advantage is that, while the transformation $\mathcal{F}$ cannot be modified because the robot is constrained to operate on $o_h$ at test time, we have more flexibility over what $\mathcal{G}$ and $o_l$ can be. We exploit this with a third type of human input called *feature queries* [19].

*Feature queries* typically involve asking the person whether an input space feature is important or relevant for the target concept. However, this query is only useful in as much as the feature itself is meaningful to the human. As such, we adapt feature queries and ask the person a few intuitive questions about the concept such that the answer informs the choice of the transformation $\mathcal{G}$. For example, a negative answer to the question "Does the size of the objects matter?" lets the robot know that $o_l$ does not benefit from containing object bounding box information. These queries let us choose an appropriate $\mathcal{G}$, which can further speed up the learning of $\phi_l$.

Given a (possibly partial) dataset of labeled human examples $\mathcal{D}_\phi$, the robot can now train a low-dimensional concept $\phi_l$. To allow for arbitrarily complex non-linear concepts, we approximate $\phi_l$ by a neural network $g(o_l;\psi) : \mathbb{R}^l \to [0,1]$, where $\psi$ denotes the parameters to learn. We treat concept learning as a classification problem, and train $g$ on the $(o_l, \phi^*(s))$ examples in $\mathcal{D}_\phi$ using a binary cross-entropy loss.

### C. Learning a high-dimensional concept

Learning a high-dimensional concept requires a large amount of labeled high-dimensional data. Generating this dataset is a two-step process: the robot needs to synthesize a large and diverse set of states $s$, which it then has to acquire labels for. However, as opposed to the low-dimensional case, this dataset need not be directly labeled by the human: the learned low-dimensional concept itself can act as a labeler.

Since at training time the robot has access to the simulator, for the data synthesis step we randomly explore the state space, much like we did for the *random* queries. Using the property in Eq. (2), we can use the low-dimensional concept $\phi_l$ to automatically label the states, generating the dataset $\{s, o_l, o_h, \phi_l(o_l)\} \in \mathcal{D}_{\phi_l}$. Given $\mathcal{D}_{\phi_l}$, the robot can now learn a high-dimensional concept $\phi_h$. Once again, we approximate $\phi_h$ by a neural network $f(o_h;\theta) : \mathbb{R}^h \to [0,1]$. We train $\theta$ via classification on the $(o_h, \phi_l(o_l))$ examples in $\mathcal{D}_{\phi_l}$ using a simple cross-entropy loss.

### D. Implementation details

We used a multilayer perceptron (3 layers, 256 units) and a standard PointNet [25], [26] to represent the low- and high-dimensional concepts, respectively. Our concepts involved relationships between objects, so we represented the high-dimensional observation $o_h$ with the relevant objects' segmented point clouds from the camera view, and the low-dimensional one $o_l$ with object poses and bounding boxes.

For data generation, we modified the objects in the ShapeNet dataset [27] such that they are consistently aligned and scaled. When synthesizing states $s \in \mathbb{R}^d$, we spawned pairs of two objects in the Isaac Gym simulator [28], and manipulated their poses and the camera pose.

## IV. Analysis: Learning Low-Dimensional Concepts

Three aspects are crucial when learning low-dimensional concepts from humans: the query synthesis strategy, the privileged input space, and how easily humans can answer queries. In this section, we analyze each of these and seek to answer the following: **Q1:** Does querying via demonstration – the most informative type of query but also the most expensive – benefit learning when compared to random label queries? **Q2:** Does modifying the privileged information space via feature queries speed up learning? **Q3:** Can we choose label queries – the cheaper version of demo queries – that are more

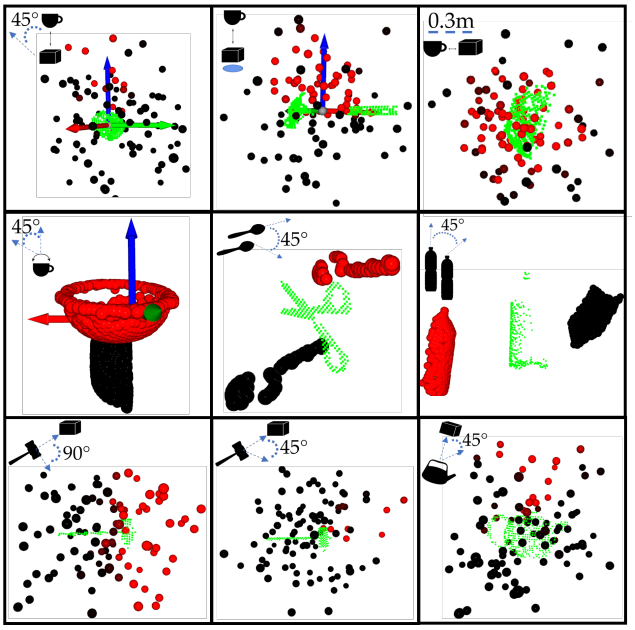


Fig. 2: Visual representation of the 9 concepts (icon in the top left of each box). The *anchor* is accompanied by examples of the *moving* object represented as either partial object point clouds (middle: *upright*, $aligned_{horiz}$, $aligned_{vert}$) or object point cloud centers (top: *above*, $above_{bb}$, *near*; bottom: *forward*, *front*, *top*). We color the examples according to concept values predicted by our method, from red (high) to black (low). For the concepts that are defined with respect to the world coordinate frame, we additionally plot the frame.

informative than random via active learning? **Q4:** How does labeling noise affect the quality of the learned concepts?

Throughout our experiments, we synthesize queries in our simulator by manipulating pairs of objects: an *anchor* and a *moving* object, to which the concept is applied with respect to the anchor. We investigate 9 prepositional concepts that arise in robot motion planning: 1. *above*: angle between the objects' relative position and the world $z$-axis, cut off after $45°$; 2. $above_{bb}$: intersection area of the two objects' bounding box projections on the world $xy$-plane; 3. *near*: inverse distance between the objects, cut off at 0.3m; 4. *upright*: angle between *moving*'s $z$-axis and the world's, cut off after $45°$; 5. $aligned_{horiz}$: angle between the objects' $x$-axes, cut off after $45°$; 6. $aligned_{vert}$: angle between the objects' $z$-axes, cut off after $45°$; 7. *forward*: angle between the anchor's $x$-axis and the objects' relative position; 8. *front*: angle between the anchor's $x$-axis and the objects' relative position, cut off after $45°$; 9. *top*: angle between the anchor's $z$-axis and the objects' relative position, cut off after $45°$. Fig. 2 showcases qualitative visualizations of our concepts.

All concept values are normalized between 0 and 1. Some concepts only apply to a subset of the objects (a mug has a *front*, but a box doesn't), so we selected objects accordingly. By default, the privileged space consists of the object poses, relative pose, positional difference, and bounding boxes.

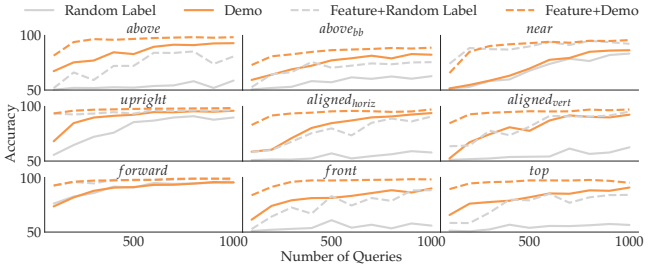


Fig. 3: Comparing different query and input space strategies. *Demo* queries outperform *random label* for concepts with few positives, and *feature* queries improve learning speed.

### A. Benefits of Demonstration, Label, and Feature Queries

Our first experiment compares the three types of human queries across two dimensions: the query selection strategy and the privileged input space. For the former, while the robot could randomly synthesize states and ask the human to label them (i.e. random label queries), for some concepts such a strategy would rarely find states with positive concept values. In contrast, demonstration queries allow the human to select the states themselves, so they can balance the amount of positives and negatives the data set contains to be informative. As for the privileged input space, by default it contains many features that are correlated with one another or irrelevant to some concepts altogether. These redundant dimensions can make it more difficult to learn the concept. Feature queries, with just a few simple and intuitive questions, can eliminate some dimensions of the input space that are unnecessary.

**Manipulated variables.** To answer **Q1** and **Q2**, we use a $2 \times 2$ factorial design. We manipulate the *query strategy* (*random label* and *demo*), and the *input space strategy* (*feature* and *no feature*). For both query strategies, we randomly generate a dataset of labeled states as described in Sec. III-D, and simulate human input by sampling examples randomly for *random label* or in a way that balances the positives and negatives for *demo*. Thus, the practical difference is in the positives-to-negatives ratio: while for *random label* that may be low for certain concepts (other than *near* and *forward* the average ratio is 0.08), for *demo* it should be close to 1.

For *feature*, we ask three intuitive questions: 1. Does the concept concern a single object? 2. Does the concept care about the objects' absolute poses or their relative one? 3. Do the object sizes matter? Q1 discards dimensions from the redundant object (useful for concepts like *upright*). Q2 gets rid of correlated features (absolute or relative pose). Q3 drops bounding box information if the concept doesn't require it.

**Dependent measures.** After training a concept network $g$, we compare it to the ground truth $\phi^*$. We use $\phi^*$ to generate a test set $\mathcal{D}_{test}$ of 20,000 state-label pairs such that they have an equal number of positives and negatives. This way, we probe whether the learned concepts perform well on both labels and don't bias to one. We measure $g$'s accuracy as the percentage of datapoints in $\mathcal{D}_{test}$ predicted correctly.

**Analysis.** In Fig. 3, we show results with varying amounts of queries from 100 to 1000. Comparing the solid lines, we immediately see that, for most concepts, demo queries

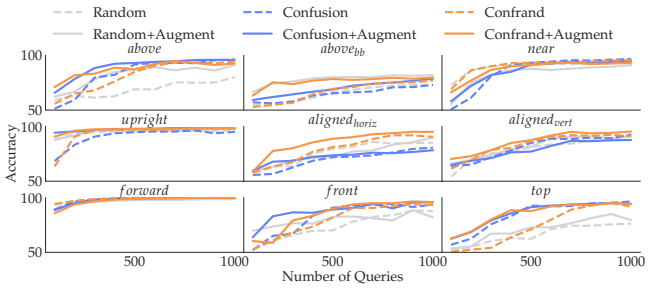


Fig. 4: Comparison for active labeling and positives selection. *Confrand* is the most consistently beneficial strategy, and *Augment* boosts performance, especially in low data regimes.

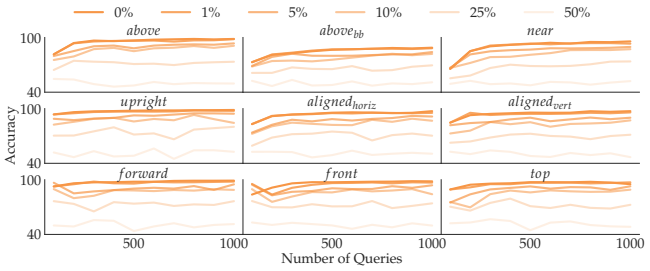


Fig. 5: Comparison for different labeling noise levels. Our method can withstand reasonable noise levels around 1-10%

perform much better than random label queries. The only concepts where this trend doesn't hold are *forward* and *near*, which are concepts where random sampling can already easily find many positives. This result stresses that having enough positives is crucial for learning good concepts. We can also compare the effect of feature queries: whether the method uses demo or label queries, feature queries considerably speed up learning, and this result holds across all 9 concepts.

### B. Active Query Labeling

In Sec. IV-A, we saw that demonstration queries substantially benefit concept learning when compared to random label queries. Unfortunately, demo queries are also very time-consuming to collect, which only makes them feasible in very low-data regimes. In this section, we tackle **Q3** and explore whether we can make label queries more informative by employing active learning techniques, rather than simply randomly selecting them. This way, we can have the benefits of both informative query generation and easy label collection.
**Manipulated variables.** We use a $3 \times 2$ factorial design where we vary the *active strategy* (*random*, *confusion*, and *confrand*) and the *positives selection* (*augment* and *no augment*). As described in Sec. III, *random* generates a query state randomly and *confusion* picks a state at the decision boundary of the currently learned concept. We also introduce *confrand*, which randomly selects between the two strategies, to balance exploration of new areas and disambiguation of the current concept. With an *augment* positives selection, for every query the method also randomly chooses whether to exploit the space of positives it has found so far or just go with the selected active strategy. We use a batch size of 100.
**Dependent measures.** We train $g$ with each strategy and varying number of queries, and report accuracy on $\mathcal{D}_{test}$.
**Analysis.** In Fig. 4, we show results with increasing number of queries across the 6 total label query selection strategies. Right off the bat, we see that active learning helps more the harder it is to find positives. For concepts like *forward* or *near*, random label queries do well from the get-go because the positive-to-negative ratio is already high. For all other concepts, however, active learning helps considerably, certain techniques more than others. A general trend is that using *augment* queries outperforms not using them, especially in lower data regimes, confirming our intuition that finding positives earlier on improves learning. Amongst *random*, *confusion*, and *confrand*, we don't see a clear winner for all concepts, but *confrand*, the combination of novelty and uncertainty exploration, seems to perform the best across.

### C. Noise Ablation

Until now, we assumed the simulated human answered the queries perfectly. Since we use label and demo queries with positive and negative labels – a fairly straightforward type of human input – this assumption is not unreasonable, but it might not necessarily hold for all novice users. We analyze **Q4**, how labeling noise affects our concept learning results.
**Manipulated variables.** We vary the *noise level* with 6 levels: 0%, 1%, 5%, 10%, 25%, and 50%. A "noisy" query has its label flipped. 50% noise is equivalent to a random labeler.
**Dependent measures.** We train $g$ by adding varying noise levels to the queries and report accuracy on $\mathcal{D}_{test}$.
**Analysis.** Fig. 5 reveals that, unsurprisingly, the noisier the queries, the worse the learned concept performs. While unrealistic noise levels like 25% or 50% severely worsen the quality of the learned concepts, our method seems to be able to withstand lower noise levels in most cases.

## V. Analysis: Learning High-Dimensional Concepts

We now present how the low-dimensional concepts $g$ we learned affect learning the high-dimensional concepts $f$. We want to see whether focusing on $g$ and using it to learn $f$ is better than a baseline that learns $f$ from the get-go, first from demonstration queries (**Q5**), then from label queries (**Q6**).

### A. Demonstration Queries

We first focus on the case where the human provides the robot with demonstration queries.
**Manipulated Variables.** To answer **Q5**, we manipulate the *learning method* with 2 levels: our *g-then-f* method and a *baseline* that learns $f$ directly from the queries. For *g-then-f*, we take the concepts we trained using both demonstration and feature queries in Sec. IV-A, and use them to label a large set of 80,000 training states, resulting in $\mathcal{D}_{\phi_l}$. Our method trains $f$ using $\mathcal{D}_{\phi_l}$, while the *baseline* trains the same architecture using the original queries we used to learn $g$. Importantly, both methods use well-balanced demonstration queries.
**Dependent Measures.** We train $f$ with each strategy and varying number of queries, and report two metrics: 1) *Test Accuracy*: how well the concepts can predict labels for a test set of states, and 2) *Optimization Accuracy*: how well the states induced by optimizing these concepts fare under the

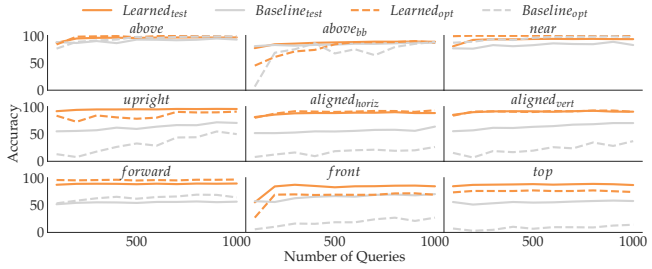


Fig. 6: Baseline comparison when using demo queries. We outperform the baseline on the more complex concepts, on both *Test* (solid) and *Optimization* (dashed) accuracy metrics.

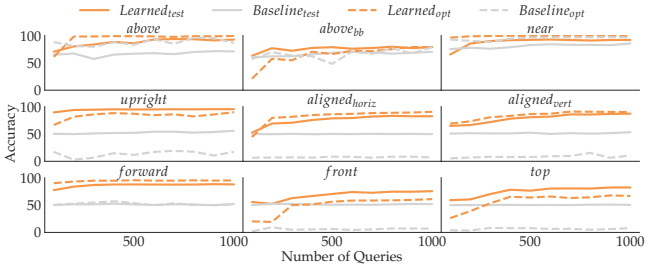


Fig. 7: Baseline comparison when using label queries. We outperform the baseline once again.

ground truth $\phi^*$. *Test Accuracy* is the same accuracy on $\mathcal{D}_{test}$ from Sec. IV. For *Optimization Accuracy*, we sample 1,000 states $\mathcal{S}_{opt}$ with a concept value of 0, and use the learned concepts to optimize them into a new set of states $\mathcal{S}'_{opt}$. We do so by finding a pose transform on the *moving* object that maximizes the concept value, and use the cross-entropy method [23]. Importantly, since this is happening at test time, we use the high-dimensional observation of the state $o_h$ to perform the optimization. We evaluate $\mathcal{S}'_{opt}$ under $\phi^*$ and report the percentage of states that are labeled 1.

**Analysis.** In Fig. 6 we plot both *Test Accuracy* (solid) and *Optimization Accuracy* (dashed). From the *Test Accuracy*, the baseline actually performs well for *above*, *above$_{bb}$*, and *near*. We think this happens because for these concepts it is easy to infer the necessary privileged information just from the positions of the point clouds. For example, for *near*, given the position of the two object point cloud centers, learning a relationship between their distance and the concept value should not require more than a few samples. The other six concepts require rotational information too, which is much more challenging to capture with little data. As a result, our method dramatically outperforms the baseline. *Optimization Accuracy* tells a similar story and shows our concepts can be optimized, which is crucial for their usability in robot tasks.

### B. Active Label Queries

We now look at the case where the human provides the robot with label queries.

**Manipulated Variables.** We manipulate the *learning method*, just like in Sec. V-A. For *g-then-f*, we chose the concepts trained using feature and label queries collected with the *confrand* and *augment* strategies together. Since the *baseline* takes too long to train due to its complex architecture, it is not suitable for active learning. As such, in our experiments the *baseline* uses random label queries. This demonstrates yet another advantage of learning low-dimensional concepts.

**Dependent Measures.** We use the same *Test Accuracy* and *Optimization Accuracy* metrics as in Sec. V-A.

**Analysis.** Fig. 7 shows both *Test Accuracy* and *Optimization Accuracy* just like before. For both metrics, our method beats the baseline on all concepts that require rotational information, and even outperforms it slightly on *above*. Even though we used label queries in this section, we see that active learning helped our method achieve high-quality concepts that can be optimized successfully with an accuracy of over 50%.

## VI. Using Concepts in Motion Planning Tasks

We also performed some tests on the real robot using the learned $f$, as shown in Fig. 1. We used unknown object instance segmentation [29] to segment out the objects and 6-dof graspnet [30] to generate grasps. For more details on the specific strategies used, see prior work [1]. The user selected the concept to test. After objects were segmented out, a user was prompted as to which object should be moved.

We generate goal positions for the moving object using the Cross-Entropy Method [23], using the concept loss as the cost function. To encourage the model to find object poses at reasonable orientations, we added a quaternion-angle cost to the CEM optimization, similarly to the metric used in prior work [31]. This is given as:

$$d(q_1, q_2) = \lambda\,(1 - \langle q_1, q_2 \rangle)$$

where $q_1$ is the pose being optimized and $q_2 = I = (0, 0, 0, 1)$ is an identity quaternion, and $\lambda = 0.001$ was manually-tuned weight. The models worked for finding object positions, even on real-world data of previously unseen objects. In the future, we would incorporate these concepts into a planner such as that proposed in [1], so as to include the robot's kinematic constraints directly in the optimization process.

## VII. Conclusion

In this paper, we presented a method for learning concepts that operate on high-dimensional input spaces. We make it feasible for a human to teach the robot a concept with little data by introducing a privileged information input space accessible only at training time. This allows the robot to quickly learn a concept in the lower dimensional space, which can then be used to generate large amounts of data for training the concept in the original high-dimensional space.

While our results demonstrate that our concepts can be used on a 7DoF Franka arm operating with real point cloud data, we still need to investigate how concepts taught by real people would fare. Our noise analysis in Sec. IV-C suggests that limited random noise might not affect the results too much, but this type of noise might not be a good model for how people make labeling errors. Despite these limitations, we are encouraged to have a tool for quickly learning complex, non-linear concepts for robot motion planning tasks. We look forward to applications of our concept learning ideas beyond prepositional relationships for manipulation robots.

## REFERENCES


[1] C. Paxton, C. Xie, T. Hermans, and D. Fox, "Predicting stable configurations for semantic placement of novel objects," in *Conference on Robot Learning (CoRL)*, 2021, to appear.

[2] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 49–58.

[3] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adverserial inverse reinforcement learning," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rkHywl-A-

[4] J. Fu, A. Singh, D. Ghosh, L. Yang, and S. Levine, "Variational inverse control with events: A general framework for data-driven reward definition," *arXiv preprint*, vol. arXiv:1805.11686, 2018.

[5] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'08. AAAI Press, 2008, pp. 1433–1438. [Online]. Available: http://dl.acm.org/citation.cfm?id=1620270.1620297

[6] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Machine Learning (ICML), International Conference on*. ACM, 2004.

[7] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *The International Journal of Robotics Research*, vol. 34, no. 10, pp. 1296–1313, 2015.

[8] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan, "Inverse reward design," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/32fdab6559cdfa4f167f8c31b9199643-Paper.pdf

[9] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning robot objectives from physical human interaction," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 217–226. [Online]. Available: http://proceedings.mlr.press/v78/bajcsy17a.html

[10] J. Choi and K.-E. Kim, "Bayesian nonparametric feature construction for inverse reinforcement learning," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[11] P. Vernaza and D. Bagnell, "Efficient high dimensional maximum entropy modeling via symmetric partition functions," in *Advances in Neural Information Processing Systems*, 2012, pp. 575–583.

[12] S. Levine, Z. Popovic, and V. Koltun, "Feature construction for inverse reinforcement learning," in *Advances in Neural Information Processing Systems*, 2010, pp. 1342–1350.

[13] N. Ratliff, D. M. Bradley, J. Chestnutt, and J. A. Bagnell, "Boosting structured prediction for imitation learning," in *Advances in Neural Information Processing Systems*, 2007, pp. 1153–1160.

[14] A. Bobu, M. Wiggert, C. Tomlin, and A. D. Dragan, "Feature expansive reward learning: Rethinking human input," in *Proceedings*

[19] M. Cakmak and A. L. Thomaz, "Designing robot learners that ask good questions," in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2012, pp. 17–24.

*of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 216–224. [Online]. Available: https://doi.org/10.1145/3434073.3444667

[15] M. Wulfmeier, D. Z. Wang, and I. Posner, "Watch this: Scalable cost-function learning for path planning in urban environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2089–2095.

[16] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," in *Robotics: Science and Systems*, 2020.

[17] S. Reddy, A. D. Dragan, and S. Levine, "SQIL: imitation learning via reinforcement learning with sparse rewards," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=S1xKd24twB

[18] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "Sornet: Spatial object-centric representations for sequential manipulation," 2021.

[20] S. Reddy, A. Dragan, S. Levine, S. Legg, and J. Leike, "Learning human objectives by evaluating hypothetical behavior," in *ICML*, 2020.

[21] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan, "Information gathering actions over human internal state," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 66–73.

[22] E. Biyik, K. Wang, N. Anari, and D. Sadigh, "Batch active learning using determinantal point processes," *CoRR*, vol. abs/1906.07975, 2019. [Online]. Available: http://arxiv.org/abs/1906.07975

[23] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.

[24] M. Kobilarov, "Cross-entropy randomized motion planning," 06 2011.

[25] E. Wijmans, "Pointnet++ pytorch," *https://github.com/erikwijmans/Pointnet2_PyTorch*, 2018.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[27] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Stanford University — Princeton University — Toyota Technological Institute at Chicago, Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.

[28] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[29] Y. Xiang, C. Xie, A. Mousavian, and D. Fox, "Learning rgb-d feature embeddings for unseen object instance segmentation," in *Conference on Robot Learning (CoRL)*, 2020.

[30] A. Mousavian, C. Eppner, and D. Fox, "6-DOF graspnet: Variational grasp generation for object manipulation," in *International Conference on Computer Vision (ICCV)*, 2019.

[31] W. Yang, C. Paxton, A. Mousavian, Y.-W. Chao, M. Cakmak, and D. Fox, "Reactive human-to-robot handovers of arbitrary objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.