Active Robot Learning for Sequential Object-directed Tasks

Kalesha Bullard and Andrea L. Thomaz

***Abstract*—We explore active learning for high-level task learning. The active learning paradigm enables a robot to both resolve unintended ambiguities during the learning process and explore new parts of the state space, in order to create a more *generalized* task representation. This research builds upon a recent framework of *embodied queries*.We apply the embodied query framework to a new learning problem: sequential object-directed tasks. Our preliminary results show that given a new set of objects, capable of achieving the task goal, the learner is able to iteratively select the most informative query, refine its model, based upon the teacher's response, and autonomously stop making queries as it reaches ground truth.**

I. Introduction

Our work is on Learning from Demonstration (LfD) [3], addressing the problem of efficiently learning high level tasks by example. We focus on high-level tasks like setting a table or making a sandwich. In demonstrating such tasks, a good teacher conveys the allowable variance (e.g., different bread or sandwich meat, etc.), enabling a robot to model the different ways to achieve the task. In this work, use Active Learning (AL) [8, 3] to efficiently query the human teacher for examples that maximize information gain in this generalization problem. A traditional AL query for robot task learning would simply request an action "label" for a state, e.g. "What action do I do here?" We build on the recently introduced *embodied queries*, which get more detailed information from teachers through three additional query types [2]. (1) **Partial label queries:** where the robot executes part of a task and asks for confirmation. (2) **Demonstration queries:** where the robot finds a configuration of the environment that its model does not cover, and asks for a demonstration from here. (3) **Feature queries:** the robot asks about the variance or invariance of particular features of that action.

Prior work in embodied queries focused on low-level skill learning, where the learning problem concerned modeling acceptable variance of the robot's end effector pose, e.g., during a pour action. Here our contribution is showing how to formulate embodied queries for the new learning problem of sequential object-directed tasks.

II. Approach

A. Modeling Sequential Object-directed Tasks

We define a high-level task, $t \in T$, as a sequence of primitive parameterized actions, $a_1, a_2, \ldots, a_n \in A$. Each action a_i has a set of precondition *criteria*, C_i, that must be satisfied before a_i can be executed and a set of postcondition *expectations*, E_i, that must be satisfied for successful execution of a_i. The sequence of actions, as well as the model C_i and E_i for each action constitutes the model of task t. In this initial work, we focus on assembly-like tasks, which are sequences of *pick-and-place(object, location)* actions.

The two example tasks we use are making a sandwich and serving coffee (abstracted into a sequence of actions on colored blocks, see Fig. 1). Each object has discrete attributes: *color* $\{red, yellow, blue, green\}$, *size* $\{small, large\}$, and *shape* $\{circle, triangle, rectangle, semicircle\}$. In these tasks, each action's *criteria* is a classifier selecting which set of objects may be used for that action, and *expectations* is a model of the location where the object can be placed.

(a) Sandwich (b) Coffee

Fig. 1: Example of each task goal. Table I shows the defined ground truth C and E models for each action. Location is a 2-D coordinate (*cm*) in a tabletop workspace.

For each task, the robot receives training demonstrations of how to pick and place objects to achieve the task goal. In each demonstration, object features may change, in ways consistent with the ground truth. The learner builds hypothesis models of the criteria and expectations for each action in the sequence; h_i^C and h_i^E for all actions in each task. A hypothesis h_i^C is a conjunction of discrete object attribute values. Each discrete attribute is either relevant or not, and if relevant, has a single target value for that attribute. If more than one value has been observed for an attribute in the demonstrations, then the attribute is marked as irrelevant as a criterion for this action. Hence the space of candidate hypotheses consists of the subset of *all* hypotheses that are *consistent* with all observed positive demonstrations of the action. A hypothesis is defined as *consistent* if it correctly classifies all seen training examples [6]. Given the predefined object attribute values, the maximum size of the version space for each action's criteria is 74. This means the size of the version space for a task $t \in T$ is $|V| = 74 * |A|$, where $|A|$ is the number of actions in t.

The representation for the *expectations* hypothesis, h_i^E, for a_i is a bivariate Gaussian distribution, with dimensions on the horizontal (x) and vertical (z) axes of the robot's workspace.

B. Task Refinement

After the human teacher provides a minimum number of demonstrations, and the robot has initial models of the task

TABLE I: ASSEMBLY TASKS GROUND TRUTH

Name	Object Label	Constrained Object Attributes	Attribute Values	Location Mean	Location Std Dev
Make Sandwich	Bread (top)	Shape	Rectangle	$(0, 35)$	$\sigma_x = 5, \sigma_z = 2$
	Meat	Color, Shape	Circle, Red	$(0, 32)$	$\sigma_x = 5, \sigma_z = 2$
	Cheese	Color, Size	Yellow, Small	$(0, 27)$	$\sigma_x = 5, \sigma_z = 2$
	Bread (bottom)	Shape	Rectangle	$(0, 24)$	$\sigma_x = 5, \sigma_z = 2$
Make Coffee	Coffee	Color, Shape	Blue, Triangle	$(0, 34)$	$\sigma_x = 5, \sigma_z = 2$
	Cup	Shape	Semicircle	$(0, 28)$	$\sigma_x = 5, \sigma_z = 2$
	Saucer	Size	Large	$(0, 23)$	$\sigma_x = 5, \sigma_z = 2$

as descried above, the robot is given a new configuration of objects to execute the task. The challenge we address is how the robot can determine what information is best to request from the human teacher, to minimize its uncertainty about the models h_i^C and h_i^E for all a_i in the given task.

1) Label Queries: Label queries, analogous to membership queries, take as input an unlabeled instance and request a label for it [8, 1]. In task learning, we consider two types of label queries: *criteria* and *expectations* label queries. A criteria label query asks, "Can I use this object?" for a specific action. An expectations label query asks, "Can it go here?" after the place action is complete.

Criteria label queries consider a set of discrete features, and we use *query-by-committee* [9] for query selection. For a particular action a_i, the version space of consistent hypotheses, V_i, serves as the committee of competing hypotheses, $C_i = \left\{h_i^{C(1)}, \ldots, h_i^{C(|V_i|)}\right\}$. Equation 1 computes the information gain for the *criteria* of action a_i, given candidate object o:

$$I(C_i|o) = -\sum_l \frac{v(l)}{|V_i|} log \frac{v(l)}{|V_i|} \tag{1}$$

where $v(l)$ is the number of votes for a label $l \in L$, the set of all labels, and $|V_i|$ is the size of the version space for a_i. In our case, this is a binary classification, that the object meets the criteria or does not. Maximizing the information gain means maximal disagreement among the committee of hypotheses. This is computed for every candidate object o in the robot's workspace. For each action, we want the object that provides the most information: $\arg\max_o I(C_i|o)$.

Expectation label queries are made in a continuous space, with a different approach for query selection. We define a utility function to select a query point and then evaluate informativeness of the selected point. Given the expectation hypothesis model h_i^E (2-D gaussian location model), a label query may be generated to assess whether placing the object in a new location meets the action's expectations. This new location, or query point, should deviate enough from the location distribution mean, to increase the distribution variance.

Prior work [1] defines the utility of a point q in the state space S as a query candidate for action as follows:

$$U_i(q) = P(q|D_i)(|\Sigma_{D_i \cup \{q\}}| - |\Sigma_{D_i}|) \tag{2}$$

where D_i represents the location distribution for action a_i. Given the utility, we aim to find a query point q, for the given action, that maximizes the utility: $\arg\max_q U_i(q)$. As in prior work [1], we use Mahalanobis distance to measure the deviation of the selected query point (a possible location to move the object) from the target concept (object's target location distribution). In addition to the query point, we specify a distance threshold, θ_{dist}, to define the cutoff boundary for which any point $s \in S$ is too far from the mean to even be considered as a query candidate. So given that $dist(q, D_i) \leq \theta_{dist}$, our measure of deviation from action a_i location distribution is given by $deviation_i(q) = \frac{dist(q,D_i)}{\theta_{dist}}$ This formulation normalizes a query point's distance from the location distribution's mean, against the distance threshold.

2) Feature Queries: Feature queries have been used previously to request information about feature relevance in learning a target concept [7, 5]. In a task learning domain, this best translates to the relevance of specific parameters of the task *criteria* and *expectations*. In the generation of criteria feature queries[1], we use information gain, computing how each object feature affects classification of the *criteria*.

Information gain for each feature f is assessed by looking at the ratio of hypotheses in the version space that consider f relevant for classifying the action *criteria*. For example, consider two competing hypotheses for classifying the *criteria* of action a_i: one says an object must be red and small and the other says it must be red. Knowing the color does not change our classification; we only need to know about the relevance of size. Equation 3 computes the information gain of a given feature f_j in trimming the action's version space, V_i:

$$I(C_i|f) = -(\frac{j(f)}{|V_i|} log \frac{j(f)}{|V_i|} + \frac{j(\bar{f})}{|V_i|} log \frac{j(\bar{f})}{|V_i|}) \tag{3}$$

where $j(f)$ is the number of hypotheses that use feature f to judge the action criteria C_i. For each action, we want the feature that maximizes Equation 3: $\arg\max_f I(C_i|f)$

3) Demonstration Queries: Demonstration queries specify some constraint and then request a demonstration of the task. For the task learning domain, we use this query in the case the robot is given a set of new objects and has such low confidence in classifying the objects with its models for a_i, that it cannot proceed with task execution. A task demo query is made if the procedure for pairing objects with actions (specified in the next section) yields a confidence below a certain threshold.

4) Query Selection Algorithm: Given base models for each action's *criteria* and *expectations*, learned from the demonstrations seen thus far, the robot is given some new set of objects in

[1]We focus on criteria feature queries; expectation model feature queries are equivalent to label queries since the model has only the location feature

the workspace. We first compute the optimal pairing between objects in the workspace and constituent actions of the task. We do this by by pairing every permutation of candidate objects with the set of actions $A = \{a_1, \ldots, a_n\}$ where n is the number of action models in task model, t. We take the pairing that yields the *maximum* positive vote probability, as defined by $pos\_vote\_prob_i(o) = P(C_i = met|o) = \frac{v(l_+)}{|V_i|}$, where $v(l_+)$ is the number of votes for a positive label.

The *positive vote probability* for action a_i measures our confidence that object o meets the action's criteria, C_i. If the average of this value, computed using every action $a_i \in t$, is below some confidence threshold θ_{conf}, this implies that the learner is significantly confused about how to pair the candidate objects with the existing actions that will move the objects, and this triggers a demonstration query.

Otherwise, we evaluate each action on an individual basis for generation of feature and label queries. Each candidate query is actually a pair including a *criteria* and *expectation* query, since in practice it is simple to ask both during a single pick-and-place action. For each action a_i in the task representation, it considers the information gain of criteria label and feature queries for each candidate object, o. Added to this is the information gain of the query location that has the greatest utility in exploring the action's *expectations* of where it may place the object. The query pair that yields the most information is selected and generated during task execution.

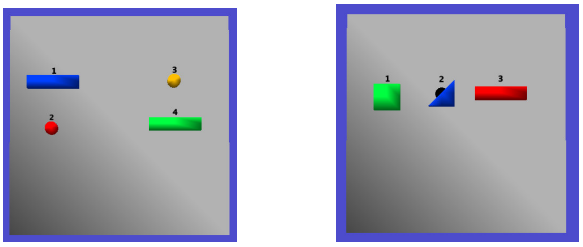
C. Task Execution

Having selected a pair of queries, the robot executes the task sequentially. It first assesses whether, given the set of objects, it has enough confidence to execute the task. If not, it requests a demo query. Otherwise, the robot proceeds with task execution. For each action that is not being queried, the algorithm uses $pos\_vote\_prob_i(o)$ to compute which candidate object it is most confident meets the *criteria*. We assume there is more than one of each object available, so different actions can pair with the same objects. It then picks and places the selected candidate object in the most probable location acceding to h_i^E. We define an *update confidence threshold*, such that if the action's confidence in the selected object meeting its criteria, exceeds this threshold, h_i^C is updated. If not, while the action is still executed, h_i^C is not updated.

When the algorithm reaches the action for which it has generated a query, it makes the pair of queries. For label queries, it makes a query about the object in question and then updates its criteria version space based upon the teacher's response. It only updates h_i^E if it receives an affirmative response. For feature queries, it selects an object which yields maximum confidence in meeting the action criteria and makes the query about this object. It updates h_i^C and subsequently makes the location feature query.

III. Evaluation

In this work, task demonstrations were performed in simulation. For each task t, we provided two training demonstrations, then presented the robot with a new set of objects, O, capable of achieving the task goal, and allowed it to iteratively select queries until it yielded a model where $|V_i| = 1$, $\forall a_i$. Figure 2 shows images of the initial state of objects for each $t \in T$, for active learning, along with a description of initial demos given. The images in Fig 2 do not depict relative size, but shape and color are displayed accurately. Table III enumerates the values of *all* discrete object attributes associated with each object depicted, as well as which task action the object was intended to be placed by.



(a) Sandwich Task (b) Coffee Task

Fig. 2: New set of objects provided for Task Execution. The initial demonstration for sandwich task used: small green rectangle (top bread), small red circle (meat), small yellow triangle (cheese), and a large blue rectangle (bottom bread). The initial demonstration given for coffee task used: small blue triangle (coffee), small yellow square (cup), and a large green rectangle (saucer). Demo two of the sandwich task changed: color of top slice of bread (green to yellow), size of meat (small to large), shape of cheese (triangle to square), and color of bottom bread (blue to green). Demo two of the coffee task changed: size of coffee (small to large), size of cup (small to large), and shape of saucer (rectangle to square).

TABLE III: Object Attributes for Each Task Execution

Task Name	Obj ID	Intended Label	Object Attributes
Make Sandwich	1	Bread (top)	Small, Blue, Rectangle
	2	Meat	Small, Circle, Red
	3	Cheese	Small, Yellow, Circle
	4	Bread (bottom)	Large, Green, Rectangle
Make Coffee	2	Coffee	Small, Blue, Triangle
	1	Cup	Small, Green Square
	3	Saucer	Large, Red, Rectangle

For each iteration of task execution, the robot is allowed to ask one pair of queries about a specific object in the scene, as it relates to an action in its task representation. Table II illustrates an entire progression of the learning process, from initial training demonstration to arrival at ground truth. The second column of the table represents the total number of hypotheses in the task model at the *beginning* of that iteration.

Take for example, the first active learning iteration of the *make a sandwich* task. For all relevant features of all objects for each action, the learner computes 15 queries yielding the maximum information gain. It randomly selects one, a feature invariance query about the meat to be picked and placed in a_2, asking if $color(o_2)$ must be red. After receiving an affirmative response, it eliminates *all* $h_2^C \in V_2$, that do not use *color* to classify C_2. In the next iteration, there are only two hypotheses left for a_2 whereas there are three hypotheses left for all

TABLE II: TASK EXECUTION RESULTS

Task 1: Make a Sandwich.						
Iteration #	$\|V\|$	Action #	Object Label	Query Generated	Response	Max Info Gain
TRAINING DEMO 1	296					
TRAINING DEMO 2	28	Task		<Demo, Confidence: 0.5714285714285714 >	Yes	n/a
AL Iteration 1	12	Action 2	Meat	<Feature Invariance, Obj ID 2, Color:Red>	Yes	0.918295834054489
AL Iteration 2	11	Action 2	Meat	<Feature Invariance, Obj ID 2, Shape:Circle>	Yes	1.0
AL Iteration 3	10	Action 1	Bread(top)	<Label, Obj ID 3>	No	0.918295834054489
AL Iteration 4	9	Action 1	Bread(top)	<Feature Invariance, Obj ID 1, Size:Small>	No	1.0
AL Iteration 5	8	Action 3	Cheese	<Feature Invariance, Obj ID 3, Size:Small>	Yes	0.918295834054489
AL Iteration 6	7	Action 3	Cheese	<Label, Obj ID 1>	No	1.0
AL Iteration 7	6	Action 4	Bread(bottom)	<Label, Obj ID 1>	Yes	0.918295834054489
AL Iteration 8	4			NONE		
Task 2: Make Coffee.						
Iteration #	$\|V\|$	Action #	Object Label	Query Generated	Response	Max Info Gain
TRAINING DEMO 1	222					
TRAINING DEMO 2	21	Task		<Demo, Confidence: 0.5238095238095238 >	Yes	n/a
AL Iteration 1	9	Action 1	Coffee	<Feature Invariance, Obj ID 2, Shape:Triangle>	Yes	0.918295834054489
AL Iteration 2	8	Action 1	Coffee	<Feature Invariance, Obj ID 2, Color:Blue>	Yes	1.0
AL Iteration 3	7	Action 2	Cup	<Label, Obj ID 1>	Yes	0.918295834054489
AL Iteration 4	5	Action 3	Saucer	<Feature Invariance, Obj ID 1, Color:Green>	No	0.918295834054489
AL Iteration 5	3			NONE		

other a_i. An additional query about a_2 will yield the highest information gain, since the committee of hypotheses will be split half-half in their vote of some candidate objects. The algorithm only computes one query which yields the maximum information gain of 1.0. As expected, this query regards a_2. The teacher's response allows the robot to make its final hypothesis elimination for a_2 and as expected no subsequent queries are made for this action. In the next iteration, the learner computes 12 queries (for the remaining three actions) yielding the maximum information gain. Of those, it randomly selects a label query for a_1, regarding the fitness of o_3 as the top slice of bread. Once the human invalidates this, the learner eliminates all $h_1^C \in V_1$ that classified o_3 as positive. The learner continues this process until it has no more queries.

In each iteration of the task, the robot is only allowed to make a query about one action. It takes several iterations to completely refine the model such that $|V_i| = 1$, $\forall a_i$. Nonetheless, the maximum number of queries made for any action is two. In some cases, the learner is able to eliminate two hypotheses for a particular action, with one query. If we enabled the learner to make one query for each action, in one task execution, we expect that it should take at most two queries to reach the ground truth, for both tasks demonstrated.

The learner is able to reach the ground truth model in both learning tasks. It autonomously decides to stop making queries because once there is only one hypothesis in the version space of every action, there is no information to be gained by queries. We also observed that after providing only one training demonstration, the learner requests a demo query. This is expected, but was tested as a proof of concept. Since, after one training demonstration, $|V_i| = 7$, $\forall a_i$, it has significant uncertainty about which objects meet an action's criteria. Confidence scores for demo queries are shown in Table II.

IV. CONCLUSION

We have presented an extension of embodied queries to sequential object-directed tasks. We formulated each query type (label, demo, feature) for this problem and showed preliminary results. Our algorithm is able to select the most informative query at each iteration, and refines its model to reach the ground truth and autonomously stop making queries.

REFERENCES

[1] M. Cakmak. Guided teaching interactions with robots: embodied queries and teaching heuristics. 2012.

[2] M. Cakmak and A. 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. ACM, 2012.

[3] S. Chernova and A. L. Thomaz. *Robot Learning from Human Demonstration*. Morgan and Claypool Publishers, 2014.

[4] H. Christensen, T. Batzinger, K. Bekris, K. Bohringer, J. Bordogna, G. Bradski, O. Brock, J. Burnstein, T. Fuhlbrigge, R. Eastman, et al. A roadmap for us robotics: From internet to robotics. *Computing Community Consortium and Computing Research Association, Washington DC (US)*, 2009.

[5] G. Druck, B. Settles, and A. McCallum. Active learning by labeling features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 81–90. Association for Computational Linguistics, 2009.

[6] T. Mitchell. *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill, 1997.

[7] H. Raghavan, O. Madani, and R. Jones. Active learning with feedback on features and instances. *The Journal of Machine Learning Research*, 7:1655–1686, 2006.

[8] B. Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52:55–66, 2010.

[9] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM, 1992.