Improving Robot Success Detection using Static Object Data

Rosario Scalise, Jesse Thomason, Yonatan Bisk, and Siddhartha Srinivasa

***Abstract*— We use static object data to improve success detection for stacking objects *on* and nesting objects *in* one another. Such actions are necessary for certain robotics tasks, e.g., clearing a dining table or packing a warehouse bin. However, using an RGB-D camera to detect success can be insufficient: same-colored objects can be difficult to differentiate, and reflective silverware cause noisy depth camera perception. We show that adding static data about the objects themselves improves the performance of an end-to-end pipeline for classifying action outcomes. Images of the objects, and language expressions describing them, encode prior geometry, shape, and size information that refine classification accuracy. We collect over 13 hours of egocentric manipulation data for training a model to reason about whether a robot successfully placed unseen objects *in* or *on* one another. The model achieves up to a 57% absolute gain over the task baseline on pairs of previously unseen objects.**

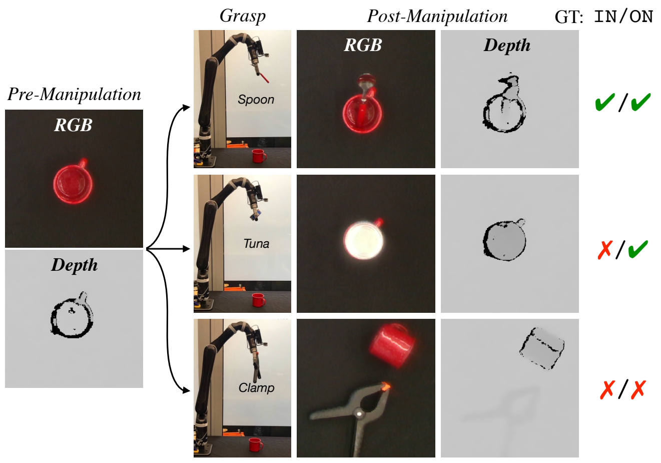


Fig. 1. The task is to classify whether dropping one object onto another resulted in the first being *in* or *on* the second. Robot RGB-D scans of the workspace pre and post action are used these to predict the ground truth.

I. Introduction

We show that static vision and language data about the objects a robot manipulates can improve action outcome detection compared to a robot's egocentric camera data alone. We consider the task of determining whether a robot's stacking action resulted in successfully placing an object *in* or *on* another object given egocentric scene information from a manipulator-mounted camera[1] (Fig. 1) and static vision and language data about the objects themselves in the form of images and referring expressions.

Consider a robot clearing a table of a plate, two cups, and two forks. Rather than clear each item one by one, a more efficient plan is to nest the cups *in* one another, place the forks *in* the cups, and put that collection *on* the plate. Accomplishing this plan requires the robot to detect success after nesting and stacking actions. Before the robot stacks the cups *on* the plate, it must know that it has successfully nested the cups *in* one another.

This problem is non-trivial for unseen objects with a wide range of geometries. In the first row of Fig. 1, the spoon is *in* the cup, but the robot's RGB signal is shallow because the handle and the cup are both red, and the depth signal is noisy because of the reflective surface of the spoon that hangs outside of the cup.

For such applications, the objects in a robot's workspace have properties—such as geometry, size, and shape—that provide a prior for action success. These properties are reflected in static visual and language data about workspace objects. Pictures taken from multiple viewpoints can reveal an object's geometry, while language descriptions can reveal an object's category (e.g., *cup*) and shape (e.g., *long*).

Our insight is to augment RGB-D workspace signals with vision and language priors about the objects themselves to improve classification accuracy of containment relationship after a robot stacking action on unseen objects.[2] In Fig. 1, static images show that the cup has a hollow center, so objects can go *in* it, while language referring expressions (e.g., *silver spoon with a long handle*) reveal a *long* shape that is easy to rest *on* other objects.

We examine whether a grasped object (O_G) lands *in* or *on* a target object (O_T) when a robot performs a simple stacking action. We gather egocentric RGB-D camera streams of stacking pairs of objects from the Yale-CMU-Berkeley (YCB) Object and Model set [1], which reflect over 50 hours of robot operation by a researcher. We propose end-to-end CNN-based models to detect whether object O_G is *in* or *on* object O_T after manipulation given this egocentric data (Fig. 2). We improve performance by adding static object images and referring expressions.

Contributions:

- We show that vision and language data about objects improves detection success for robot action outcomes;
- we provide an end-to-end model for detecting successful placement of one object *in* or *on* another; and
- we create a dataset of over 13 hours of robot stacking sensor streams, more than 4000 human annotations of whether object pairs can be *in* and *on* one another, and over 800 natural language object referring expressions.

Paul G. Allen School of Computer Science and Engineering, University of Washington `rosario@cs.washington.edu`

[1]`http://rosarioscalise.com/IROS19`

[2]All code and data for reproducing our results are available at `https://github.com/thomason-jesse/YCBLanguage`

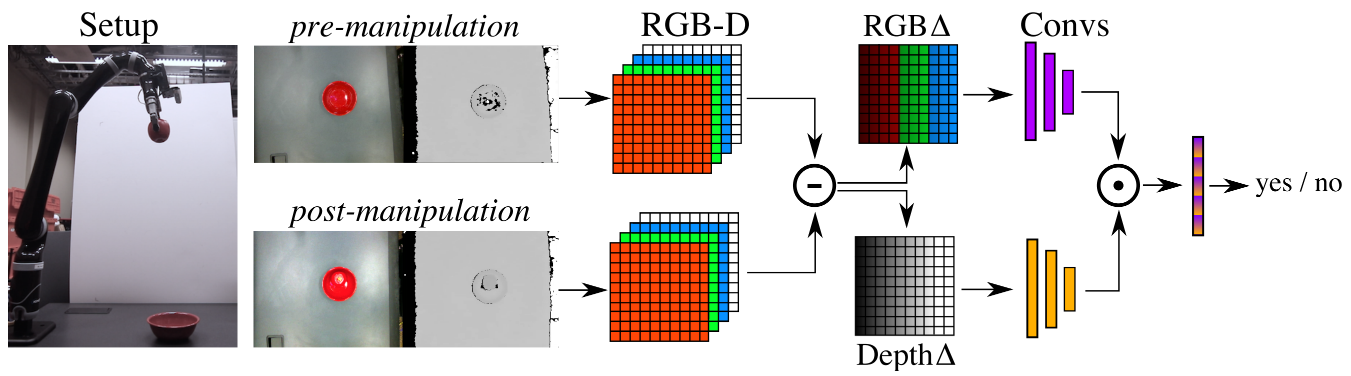


Fig. 2. End-to-end model M^{ego} architecture trained to detect whether the grasped object is either *in* or *on* the target object after manipulation given egocentric scans. The robot takes RGB-D snapshots of the workspace before and after attempting to stack the objects. The color and depth differences between these scenes are processed by separate convolutional neural networks, then element-wise multiplied together and used to predict the outcome through a fully connected layer. Both color and depth are informative for this task since objects can have similar colors and reflective surfaces can mislead depth scans.

We assume oracle-level object grasping (grasp selected by the researcher), and motion planning that results in object O_G being held within 3cm of the target O_T before being dropped. We assume a fixed vantage point from which to survey the scene pre- and post-manipulation, ameliorating one difficulty of manipulator-mounted cameras. We consider pairwise interactions between objects, limiting our task to two objects at a time. We consider nesting objects *in* and stacking them *on* one another as a starting point for a wider class of object containment relations.

II. Related Work

Our task is to detect action success in terms of physical relationships between two objects using a manipulator-mounted camera. This task relates to work in object relations, grasping and stacking, and egocentric vision. We demonstrate improvements by using static data about the objects involved, including human annotations, related to other work in utilizing human judgements for robotics applications.

a) Object Relations: Beyond a functional understanding of which object it can interact with [2], a robot can reason about relationships between objects. Single models that take in information about two objects can reason about multiple relationships between them [3], but this requires large, annotated training data for all inter-object relations. Some work targets relationships between *tools* and target objects [4]. Other work considers *stacking* to be a category of high-level activity within a dataset [5], or learns object-level attributes for *stackable* and *containment* [6]. Still others learn to ground multi-object concepts, like *rows* and *columns* [7]. We use static object information as a prior to improve egocentric detection of whether a robot successfully stacked one object *on* or contained one object *in* another.

b) Grasping and Stacking: Current robot systems are increasingly proficient at robustly grasping objects [8]–[10] and subsequently dropping, placing, and balancing them [11], [12]. The outcomes of these actions vary based on the dexterity of the robot manipulator [13]. Longer-horizon tasks, like stacking multiple objects, require awareness of intermediate action success. Research has addressed: reasoning about visual stability of objects in simulation [14]–[16] and in a robot workspace [17]; detecting intermediate task success [18], [19]; and predicting whether a task will succeed before executing it [20]. We define success in terms of two object relationships: *in* and *on*, and we assume oracle-level grasping and localization during action execution.

c) Egocentric Vision: Existing work detects action outcomes using extrinsic camera viewpoints [21]. However, as robots move from research labs to homes, static cameras become impractical and intrusive. Instead, egocentric cameras can be mounted to the robot chassis [22], [23] or manipulator [24], [25]. Most egocentric vision research centers on identifying human [26], [27] and robot activities [28] or ego-driven scene prediction [29]. These perspectives are often obtained through head- or body-mounted cameras [30]–[32]. We use a manipulator-mounted camera, which may become the norm for future robot platforms (e.g., Kinova Jaco 3). This setup restricts the robot to a single viewpoint at a time.

d) Human Judgement: Attempting to model or utilize human manipulation abilities is a long-studied problem in robotics [33]. Researchers have used crowdsourcing platforms to source data for grasping novel objects [34] and translating natural language commands to actions [35], [36]. Natural language descriptions of objects from humans can help robots learn visual (*red*) [37]–[39] or multi-modal (*heavy*) [40], [41] classifiers between words and workspace objects, as well as inter-object relations [42]. While embeddings learned by language modeling (e.g., [43]) place words like *fork*, *plate*, and *cup* as neighbors in the embedding space despite their referents having different physical properties, language vectors informed by physical properties [41] encode some differences in geometry. However, such vocabularies are noisy and sparse, making them insufficient for an open-vocabulary (unrestricted word choice) domain. We crowdsource open-vocabulary referring expressions (e.g., "yellow long fruit") for objects, and annotations for whether humans think objects can be placed *in* or *on* one another.

III. Task and Data

Our task is to determine whether a robot stacking action successfully placed object O_G *in* or *on* object O_T given egocentric scans from a manipulator-mounted camera.

a) Data Folds: We chose the YCB Object and Model set [1], a standardized collection of physical objects designed for benchmarking robot manipulation, for two reasons. First, this object set is widely used among robotics labs, making the data we collect useful to other researchers. Second, its constituent objects span a diverse space of geometries and potential physical interactions, making the detection of simple physical relations like *in* and *on* after a stacking attempt more challenging.

Among the full set of 90 YCB objects, Y, we designate a subset of 28 *containers* as C (e.g., cups, boxes, and cans). We split Y into *Train*, *Development*, and *Test* data folds. Pairs of objects considered in each fold are considered for the task (e.g., all *Train* object pairs (O_G, O_T) comprise two *Train* objects). Consequently, not all object pairs are included in the data folds since pair i, j cannot be included in any fold if individual objects i and j are assigned to different folds. This split also means that all objects are unseen at inference time. For testing O_G *in* O_T, we consider pairs where $(O_G, O_T) \in Y \times C$, all objects paired with all containers; for O_G *on* O_T, we consider pairs where $(O_G, O_T) \in Y \times Y$, all objects paired with all objects. We evaluate on the subset of object pairs in those splits for which we collect egocentric, RGB-D data from robot trials (**Robot Pairs**), and we define an auxiliary task over all pairs in these splits (**All Pairs**). Table I gives the number of pairs per data fold.

b) Robot Pairs: For a subset of **All Pairs**, we collect RGB-D directly from the robot's egocentric viewpoint. For each such **Robot Pair**, five trials were gathered, resulting in $191 \times 5 = 955$ training examples. Our robot is the widely used Kinova Jaco2 [44]. We use an Intel Realsense D415 RGB-D camera [45] mounted on the end-effector.

To collect egocentric camera data for each pair (O_G, O_T), object O_T is first set on an origin position on the table. Then, the robot moves to a designated survey position and captures a *pre-manipulation* RGB-D frame. An operator manually places object O_G within the robot's gripper, and the gripper is closed. Then, the robot gripper moves to a height of 3cm above O_T and opens. Finally, the robot returns to the survey position and captures a *post-manipulation* RGB-D frame.

We collect 5 such trials per object pair. For each pair, the operator provides an annotation in {*Yes*, *No*} of the outcome, noting whether object O_G was *in* or *on* object O_T for all five trials.[3] This data collection is expensive to conduct because it uses operator-selected oracle grasping.

c) Data Augmentation: To augment expensive robot pair data collection, we gather complementary data about objects and object pairs via crowdsourcing through Amazon Mechanical Turk (AMT)[4] conditioned only on static images

[3]For cases where the trials had different annotated outcomes (e.g., *on* 2 times and not 3 times), the label was decided as *No*. Similarly, annotator disagreement from Mechanical Turk was rounded to *No*.

[4]`https://www.mturk.com/`

TABLE I
Dataset folds sizes.

Fold	Objects		Robot Pairs		All Pairs	
	Y	C	*in*	*on*	*in*	*on*
Train	51	17	191	191	800	2500
Dev	20	5	47	58	100	400
Test	19	6	60	60	114	361

of the objects. The YCB dataset provides standardized images of each object from multiple angles. For each object, we choose the 0° yaw, 30° pitch viewpoint as a representative image to show AMT workers (examples in Figure 3).

Each AMT worker annotates 35 pairs of side-by-side object images. For 9 of these pairs, we ask annotators whether object O_G could be put *in* object O_T. For the remaining 26 pairs, we ask annotators if object O_G could be placed *on* object O_T. For each object pair, we collect at least three annotations for each of *in* and *on* and assign labels {*Yes*, *No*} based on them.

Subsequently, for each YCB object from Y, we collect three referring expressions from three different annotators (nine in total per object). These expressions, for example *small black sphere* in the first row of Table III, provide linguistic clues about object size and shape properties.

IV. Models

We introduce an end-to-end model that uses egocentric camera inputs to determine whether a stacking action resulted in an *in* or *on* relationship between two objects (Fig. 2). We then add additional object data inputs: images and natural language referring expressions of the objects.

a) Egocentric Model M^{ego}: During a robot execution trial, the robot's egocentric camera provides *pre-manipulation* and *post-manipulation* RGB-D inputs: R_{pre} and R_{post} of size 640×480. These images are downsampled once via an average pooling layer to size 64×48. We use the difference between the images (R_Δ) as input to the model.

We learn two egocentric classification models, M^{ego}_{in} and M^{ego}_{on}, that independently learn whether *in* and *on* hold between pairs of objects. These models consist of two 3-layer convolutional nets, one for processing RGB and one for processing D inputs, with max-pooling followed by a singular fully connected layer. Specifically, these models take 48×64 RGB (3 color channels + 1 L_2 channel) and depth image (1 depth channel) differences as input and output a label $c \in \{Yes, No\}$. To capture changes in RGB space, we include the L_2 distance between R_{post} and R_{pre}. L_2 lets us capture change across all three dimensions, while subtraction assumes the color channels are independent. Fig. 2 depicts the egocentric model architecture.

b) Egocentric plus Object Data Model $M^{\text{ego+obj}}$: We augment the *Egocentric Model* with two additional inputs. First, we add five images of each object (O_G and O_T), independently representing front, back, left, right, and top-

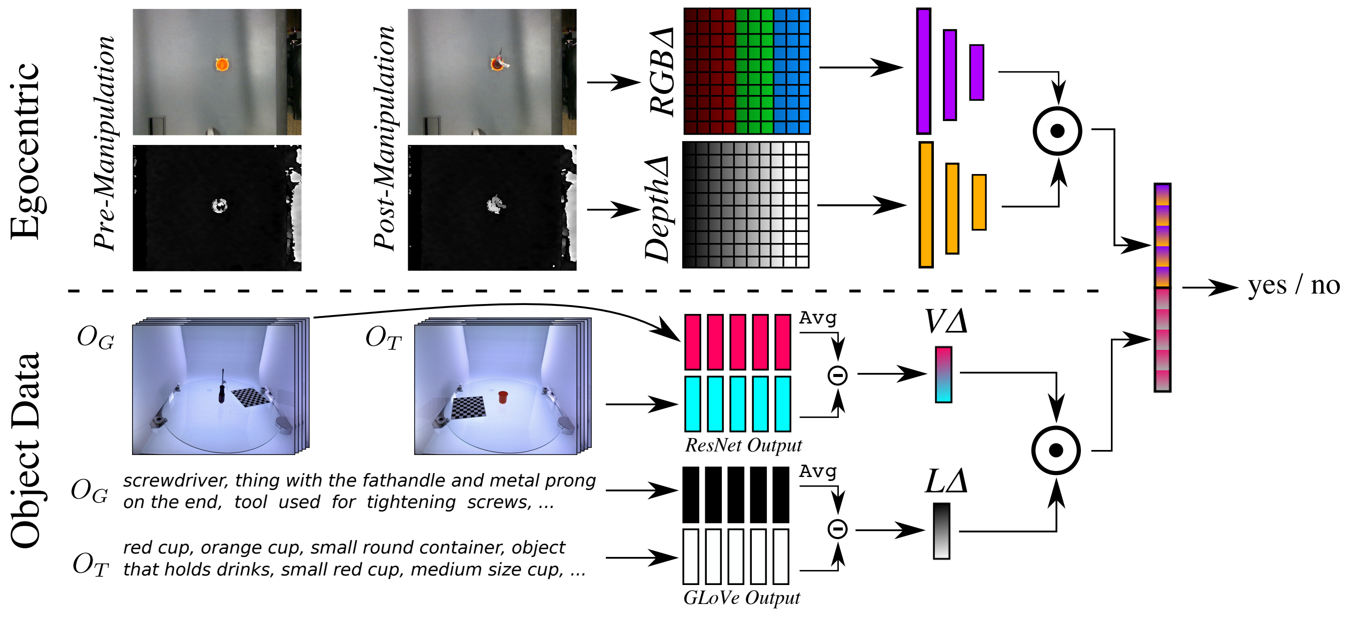

Egocentric
Pre-Manipulation
Post-Manipulation
RGBΔ
DepthΔ
yes / no
Object Data
O_G
O_T
Avg
VΔ
ResNet Output
O_G screwdriver, thing with the fathandle and metal prong on the end, tool used for tightening screws, ...
O_T red cup, orange cup, small round container, object that holds drinks, small red cup, medium size cup, ...
Avg
LΔ
GLoVe Output

Fig. 3. End-to-end model $M^{\text{ego+obj}}$ takes egocentric visual input, vision embeddings from multiple static image viewpoints of each object, and language embeddings from referring expressions for each object. The vision and language embeddings Δs for each object pass through fully connected layers, whose outputs are element-wise multiplied and then concatenated with the output of the egocentric pipeline. This concatenation is used to classify the pair label.

down views.[5] These static images provide multiple vantage points from which to extract the objects' geometry features. Second, we add natural language referring expressions for each object, which provide linguistic clues about physical properties (e.g., "small" and "long").

We use ResNet-152 [46] and GLoVe vectors [43] (sans fine-tuning) to embed our images[6] and language. The embeddings across the five images and across the referring expression words are averaged to compute embeddings for the auxiliary vision and language input, respectively. We compute the differences between embeddings for O_G and O_T visually and linguistically before projecting them to the same size space and concatenating the output to the penultimate layer of the *Egocentric Model* for classification. These object pair embedding vectors from egocentric scans and from object data concatenated for classification have the same size. Fig. 3 shows the model architecture. We again learn independent $M_{in}^{\text{ego+obj}}$ and $M_{on}^{\text{ego+obj}}$ models.

Object images can capture geometry, e.g., if O_T is wide and flat, O_G is more likely to go *on* it. Referring expressions can reveal how descriptor words like "small" and "large" for O_G encode how likely it is to go *in* O_T.

c) Egocentric plus Pretrained Object Data Model $M^{\text{ego+pre}}$*:* Running the robot through thousands of trials and labeling the outcomes is prohibitively expensive, but object data can be crowdsourced (as described in Section III). Whereas in the previous two models, we trained using the smaller data subset, **Robot Pairs**, here we make use of the **All Pairs** data to pretrain relevant layers of the $M^{\text{ego+obj}}$ architecture to obtain the $M^{\text{ego+pre}}$ model.

We pretrain the $M_{in}^{\text{ego+obj}}$ and $M_{on}^{\text{ego+obj}}$ architecture layers that map from input embeddings to hidden projection layers on an auxiliary task: predicting *in* and *on* Mechanical Turk annotations given only images and referring expressions for objects O_G and O_T (e.g., no RGB-D input information). This is a noisy signal because human intuition about these stacks might not align with the robot's ability, but we can train on the much larger set of **All Pairs**.

We hypothesize that this pretraining will lead to higher task accuracy. We denote $M_{in}^{\text{ego+pre}}$ and $M_{on}^{\text{ego+pre}}$ models with layers for processing image and referring expression inputs instantiated with pretrained weights from this auxiliary task. We compare these against the randomly initializing the weights before training, as done for the $M^{\text{ego+obj}}$ models.

V. Experiments and Results

We evaluate these models by comparing their performance on the *Test* data fold of **Robot Pairs**.[7] We compare model variants against one another and majority class and random baselines. For evaluation, we train each model for 30 epochs, recording the average maximum test data fold accuracy reached in that time across 10 different seeds.[8] During training, all 5 trials per object pair serve as individual training examples, while at inference time a majority vote across the 5 trials of the test pair is taken to assign the label. Table II

[5]Extracted from given YCB object metadata. Accuracy is slightly lower when considering only the front image.

[6]Such models could potentially extract features from egocentric scans, but are trained for ImageNet classification, not task-relevant spatial relations.

[7]We tune all architecture hyperparameters to maximize accuracy on the development data fold. We use an Adam optimizer [47] with learning rate 0.01, dropout of 0.3 and ReLu connections between fully connected layers, and a hidden layer size of 64. For RGB-D, we use a 3×3 kernel, with three layers of alternating convolution and max pooling operations and filters doubling at each.

[8]Random restarts to expose variance in model performance.

TABLE II

MODEL PERFORMANCE ON BOTH *Development* AND *Test* **Robot Pairs**.

	Model (M)		**Detection Correct ↑**	
	Ego	Object	*in*	*on*
Dev Fold	✓		.69 ± .08	.55 ± .11
	✓	✓	.70 ± .09	.59 ± .07
	✓	pre	.73 ± .09	.62 ± .07
	Baseline (MC)		.32 ± .00	.36 ± .00
	Baseline (Rand)		.49 ± .06	.50 ± .06
Test Fold	✓		.77 ± .05	.53 ± .10
	✓	✓	.74 ± .07	.59 ± .08
	✓	pre	.77 ± .05	.59 ± .06
	Baseline (MC)		.20 ± .00	.32 ± .00
	Baseline (Rand)		.52 ± .05	.51 ± .07

✓indicates signal was included, while "pre" indicates models with object features pretrained from **All Pairs** data.

shows model and baseline accuracy and standard deviation on the *Development* and *Test* data folds.

All end-to-end models achieve higher accuracy than the baselines, with a 57% absolute gain for the *in* task between the baseline and M_{in}^{ego} / $M_{in}^{\text{ego+pre}}$ models. When considering the *in* and *on* detection tasks together, models with vision and language processing layers pretrained on an auxiliary task, $M^{\text{ego+pre}}$, score higher accuracy than the others across both the *Test* and *Dev* folds.[9]

VI. DISCUSSION

The proposed models all outperform the baseline at detecting both *in* and *on* relationships. Here, we discuss their relative performance on these tasks, and Table III shows examples of model disagreements between the egocentric M^{ego} models and object data augmented $M^{\text{ego+pre}}$ models.[10]

a) Analysis: The egocentric RGB-D scans are not always sufficient to make an accurate success judgement. The first and second rows of Table III show examples where egocentric data fails to discern that object O_G has been successfully placed in object O_T. In the first row, object O_G is a tiny, barely perceptible marble. In the second row, object O_G is a screwdriver with a shiny tip, resulting in a misleading depth scan (similar to the first row of Fig. 1).

In general, our models have more limited accuracy when detecting *on* than *in* relationships: the *on* relationship is semantically more general than *in*. Specifically, consider that any object O_G *in* a container O_T is also implicitly *on* that container, while the converse does not hold. This leads to at least two distinct modalities for *on*: object O_G on top of object O_T, and O_G is inside of O_T. Each presents differently in depth scans. The third row of Table III shows an object pair exhibiting a hybrid of these two *on* modalities. Generalizing from the *Train* to *Test* data pairs is challenging, especially for the wider *on* relation.[11]

Human intuitions of what relationships can hold between objects are not perfectly consistent with the robot's capabilities. The fourth row of Table III shows that while humans annotate that a banana can balance *on* a spam can, the robot manipulator lacks the dexterity to reliably balance the cylindrical banana. This is reflected in the egocentric M_{on}^{ego} model making the correct decision, while the object data augmented model is mislead by pretraining from human judgement that fruits can be balanced on cans.

b) Limitations and Future Work: To explore the containment detection task, we make several data collection and modeling assumptions (listed in Section I). Relaxing these assumptions is beyond the scope of this work, but we discuss future research towards doing so.

Collecting egocentric camera stream information from a robot performing manipulation tasks is expensive in terms of operator hours, but running data collection in a self-supervised fashion is difficult in practice when dealing with unseen objects. Future work towards automatically selecting robust grasps, and planning for correct object placement on the target object, may allow the robot to gather egocentric data autonomously for subsequent experimenter annotation.

We simplify the manipulator-mounted vision input to our model by assuming a fixed vantage point to capture pre- and post-manipulation RGB-D scans. While our data processing makes the model robust to different table heights between pair samples (because we subtract pre- and post- scans), it may be brittle against lighting and axial changes. Standard computer vision techniques (e.g., cropping, translating, color shifting) can increase model robustness, while future autonomous data collection could use the manipulator-mounted camera to capture scans from multiple vantage points.

We consider pairwise interactions between objects when both pre- and post-manipulation scans are available. In many applications contexts (e.g., table clearing), multiple objects interact, and some may already be interacting when the robot arrives. After nesting two cups *in* one another, a robot may need to determine whether they can rest *on* a plate. Similarly, a cup with a fork already *in* it cannot have a cup nested *in* it. Static object data generalizes to the multiple object domain in some cases. Using referring expressions, we can infer that nested *cups* still behave like a *cup*, while a *cup* and *fork* may not. Another approach would be to simulate the multiple objects involved, but that requires high-fidelity object models and physics interactions in simulation.

Nesting objects *in* and stacking them *on* one another exemplify two simple actions. Object manipulation in industrial [48] or home furniture assembly [49], [50] involves additional relationships, like *fastening* one object to another (e.g., by tightening a screw or hammering a nail). We provide a starting point for detecting when manipulation leads to con-

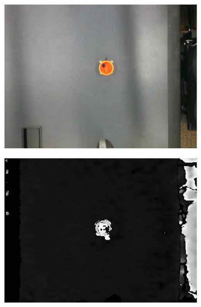
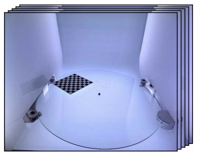
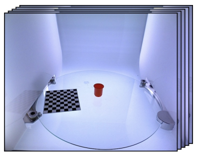
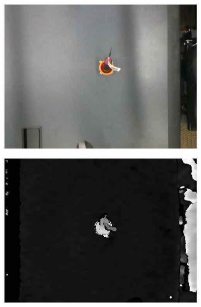
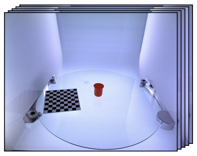
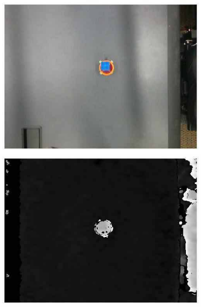
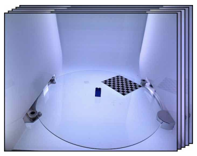
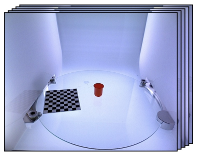
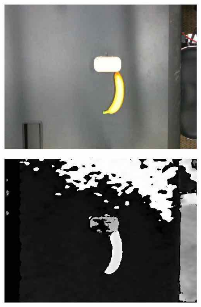
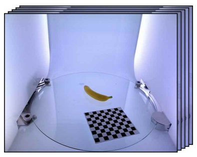
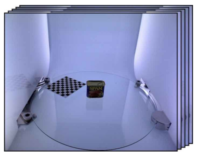
[9]We also tested models pretrained on the auxiliary task but using only object data (e.g., no *Ego* input). These are accurate for detecting the more geometry-informed *in* relation but fall short when classifying *on*, which is conditioned on robot dexterity (e.g., Table III row 4).

[10]Table III shows the best performing model across random restarts.

[11]By contrast, on the *Train* data fold, all models (except the baselines) for both *in* and *on* perform at or above 85% accuracy when measured at the epoch achieving the best *Test*/*Dev* generalization accuracy reported in Table II.

TABLE III

EGOCENTRIC M^{ego} MODEL AND AUGMENTED $M^{\text{ego+pre}}$ MODEL PREDICTIONS. WE DISPLAY ONLY THE POST-MANIPULATION FRAME FOR R_{Δ}.

Egocentric	⇒	Pred	**Egocentric + Pretrained Object**	⇒	Pred	Truth	Reason
		✗In	 *small black sphere*, *round black item*, *small marble*, *the blue object*, *round object*, *tiny object*, *tiny dot*, *blue round object*, *little ball*  *red cup*, *orange cup*, *small round container*, *object that holds drinks*, *small red cup*, *red cup*, *medium size cup without handles*, *red plastic thing*, *red cylinder*		✓In	✓In	With egocentric vision alone, it is difficult to see the small marble. After adding object data, the model classifies the *tiny* marble *in* the *cup* container.
		✗In	*screwdriver*, *thing with the fat handle and metal prong on the end*, *tool used for tightening screws*, *screw driver with long tip*, *screwdriver*, *plastic handle screw driver*, *non phillips screw driver*, *tool*, *black screwdriver*  *red cup*, *orange cup*, *small round container*, *object that holds drinks*, *small red cup*, *red cup*, *medium size cup without handles*, *red plastic thing*, *red cylinder*		✓In	✓In	The reflective screwdriver and its tag spill over the edge of the cup, creating a misleading egocentric depth scan. After adding object data, the model classifies the *screwdriver* as being *in* the *cup* container.
		✗On	 *blue thing*, *blue plastic rectangle*, *blue plastic block*, *blue cube*, *lego piece*, *blue plastic thing*, *blue block*, *small square block*, *little blue block*  *red cup*, *orange cup*, *small round container*, *object that holds drinks*, *small red cup*, *red cup*, *medium size cup without handles*, *red plastic thing*, *red cylinder*		✓On	✓On	The block rests partially nested in the cup, exhibiting a rare hybrid of both *on* modalities (resting on top versus nested inside). After adding object data, the *small block* is classified as resting *on* the *cup* container.
		✗On	 *yellow thing*, *long yellow item*, *soft yellow thing*, *yellow curved cylinder*, *yellow fruit*, *the object that is mostly yellow with slight green at one of the tips*, *yellow long fruit*, *yellow banana*, *banana*  *spam*, *canned meat*, *metal can*, *can of spam*, *aluminum cube*, *blue and gold cube*, *rectangular can*, *square*, *glass circle*		✓On	✗On	Given object data predictions pretrained on crowdsourced labels, the model predicts that *fruit* is balanced *on* the *can*, consistent with human ability. In reality, the robot manipulator lacks the dexterity to reliably balance the banana.

tainment, and hypothesize that prior information about the objects themselves will improve success detection accuracy for other, nuanced physical containment relationships.

VII. CONCLUSION

We provide a means by which a robot equipped with an egocentric, manipulator-mounted RGB-D camera can determine whether object placement actions result in one object being placed *in* or *on* another. We propose a model solely trained on a robot's observations of outcomes, and then we show that task accuracy can be improved using prior visual and linguistic information about the objects involved. We find that pretraining with crowdsourced annotations of whether objects can be placed *in* or *on* one another bootstraps useful feature extraction from this prior information.

ACKNOWLEDGEMENTS

This work was funded by the National Institute of Health (#R01EB019335), NSF (CPS #1544797, NRI #1637748, #IIS-1524371 & #1703166), the Office of Naval Research, the RCTA, Amazon, Honda, and DARPA's CwC program through ARO (W911NF-15-1-0543).

References

[1] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *ICAR*, July 2015.

[2] C. Ye, Y. Yang, C. Fermüller, and Y. Aloimonos, "What can I do around here? Deep functional scene understanding for cognitive robots," in *ICRA*, 2017.

[3] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *NeurIPS*, 2017.

[4] Y. Zhu, Y. Zhao, and S.-C. Zhu, "Understanding tools: Task-oriented object modeling, learning and recognition," in *CVPR*, 2015.

[5] H. S. Koppula, R. Gupta, and A. Saxena, "Learning human activities and object affordances from RGB-D videos," *IJRR*, vol. 32, no. 8, pp. 951–970, 2013.

[6] A. Aldoma, F. Tombari, and M. Vincze, "Supervised learning of hidden and non-hidden 0-order affordances and detection in real scenes," in *ICRA*, 2012.

[7] R. Paul, J. Arkin, D. Aksaray, N. Roy, and T. M. Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot platforms," *IJRR*, vol. 37, no. 10, pp. 1269–1299, 2018.

[8] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *ICRA*, 2018.

[9] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg, "Learning ambidextrous robot grasping policies," *Science Robotics*, vol. 4, no. 26, 2019.

[10] J. Tobin, L. Biewald, R. Duan, M. Andrychowicz, A. Handa, V. Kumar, B. McGrew, A. Ray, J. Schneider, P. Welinder, *et al.*, "Domain randomization and generative models for robotic grasping," in *IROS*, 2018.

[11] R. Paolini, A. Rodriguez, S. S. Srinivasa, and M. T. Mason, "A data-driven statistical framework for post-grasp manipulation," *IJRR*, vol. 33, no. 4, pp. 600–615, 2014.

[12] Y. Jiang, M. Lim, C. Zheng, and A. Saxena, "Learning to place new objects in a scene," *IJRR*, vol. 31, no. 9, pp. 1021–1043, 2012.

[13] A. Holladay, J. Barry, L. P. Kaelbling, and T. Lozano-Perez, "Object placement as inverse motion planning," in *ICRA*, 2013.

[14] W. Li, A. Leonardis, and M. Fritz, "Visual stability prediction for robotic manipulation," *ICRA*, 2017.

[15] O. Groth, F. B. Fuchs, I. Posner, and A. Vedaldi, "Shapestacks: Learning vision-based physical intuition for generalised object stacking," in *ECCV*, 2018.

[16] J. Hamrick, P. Battaglia, and J. B. Tenenbaum, "Internal physics models guide probabilistic judgments about object dynamics," in *33rd Annual Conference of the Cognitive Science Society*, vol. 2, 2011.

[17] F. Furrer, M. Wermelinger, H. Yoshida, F. Gramazio, M. Kohler, R. Siegwart, and M. Hutter, "Autonomous robotic stone stacking with online next best object target pose planning," in *ICRA*, 2017.

[18] J. Worcester, M. A. Hsieh, and R. Lakaemper, "Distributed assembly with online workload balancing and visual error detection and correction," *IJRR*, vol. 33, no. 4, pp. 534–546, 2014.

[19] G. Xue, T. Fukuda, and H. Asama, "Error recovery in the assembly of a self-organizing manipulator by using active visual and force sensing," *Autonomous Robots*, vol. 1, no. 2, pp. 179–186, 1995.

[20] H. Nguyen and C. C. Kemp, "Autonomously learning to visually detect where manipulation will succeed," *Autonomous Robots*, vol. 36, pp. 137–152, 2014.

[21] M. Dogar, R. A. Knepper, A. Spielberg, C. Choi, H. I. Christensen, and D. Rus, "Multi-scale assembly with robot teams," *IJRR*, vol. 34, no. 13, pp. 1645–1659, 2015.

[22] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *CoRL*, 2018.

[23] A. Stentz, H. Herman, A. Kelly, E. Meyhofer, G. C. Haynes, D. Stager, B. Zajac, J. A. Bagnell, J. Brindza, C. Dellin, *et al.*, "CHIMP, the CMU highly intelligent mobile platform," *Journal of Field Robotics*, vol. 32, no. 2, pp. 209–228, 2015.

[24] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 14–35, 1993.

[25] M. Klingensmith, S. S. Sirinivasa, and M. Kaess, "Articulated robot motion for simultaneous localization and mapping (arm-slam)," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1156–1163, 2016.

[26] P. Duckworth, D. C. Hogg, and A. G. Cohn, "Unsupervised human activity analysis for intelligent mobile robots," *Artificial Intelligence*, vol. 270, pp. 67–92, 2019.

[27] M. S. Ryoo, T. J. Fuchs, L. Xia, J. K. Aggarwal, and L. H. Matthies, "Robot-centric activity prediction from first-person videos: What will they do to me?" *HRI*, 2015.

[28] Z. Li, C. Au, Y. Kakiuchi, K. Okada, and M. Inaba, "What am I doing? Robotic self-action recognition," *International Conference on Humanoid Robots (Humanoids)*, 2016.

[29] D. Jayaraman and K. Grauman, "Learning image representations tied to egomotion," in *ICCV*, 2015.

[30] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray, "Scaling egocentric vision: The EPIC-KITCHENS dataset," *CoRR*, 2018.

[31] R. Luo, O. Sener, and S. Savarese, "Scene semantic reconstruction from egocentric RGB-D-thermal videos," *International Conference on 3D Vision*, 2017.

[32] M. Ma, H. Fan, and K. M. Kitani, "Going deeper into first-person activity recognition," *CVPR*, 2016.

[33] T. Iberall, J. Jackson, L. Labbe, and R. Zampano, "Knowledge-based prehension: Capturing human dexterity," in *ICRA*, 1988.

[34] A. Sorokin, D. Berenson, S. S. Srinivasa, and M. Hebert, "People helping robots helping people: Crowdsourcing for grasping novel objects," *IROS*, 2010.

[35] Y. Bisk, K. Shih, Y. Choi, and D. Marcu, "Learning interpretable spatial operations in a rich 3d blocks world," in *AAAI*, 2018.

[36] J. Thomason, S. Zhang, R. Mooney, and P. Stone, "Learning to interpret natural language commands through human-robot dialog," in *IJCAI*, 2015.

[37] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Visual curiosity: Learning to ask questions to learn visual recognition," in *CoRL*, 2018.

[38] M. Tucker, D. Aksaray, R. Paul, G. J. Stein, and N. Roy, "Learning unknown groundings for natural language interaction with mobile robots," in *International Symposium of Robotics Research*, 2017.

[39] N. Parde, A. Hair, M. Papakostas, K. Tsiakas, M. Dagioglou, V. Karkaletsis, and R. D. Nielsen, "Grounding the meaning of words through vision and interactive gameplay," in *IJCAI*, 2015.

[40] J. Thomason, A. Padmakumar, J. Sinapov, J. Hart, P. Stone, and R. J. Mooney, "Opportunistic active learning for grounding natural language descriptions," in *CoRL*, 2017.

[41] J. Thomason, J. Sinapov, M. Svetlik, P. Stone, and R. Mooney, "Learning multi-modal grounded linguistic semantics by playing "I spy"," in *IJCAI*, 2016.

[42] J. Kulick, M. Toussaint, T. Lang, and M. Lopes, "Active learning for teaching a robot grounded relational symbols," in *IJCAI*, 2013.

[43] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *EMNLP*, 2014.

[44] "Kinova jaco 2 arm," http://www.kinovarobotics.com, 2019.

[45] "Intel realsense d415," https://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html, 2019.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[48] M. Kyrarini, M. A. Haseeb, D. Ristić-Durrant, and A. Gräser, "Robot learning of industrial assembly task via human demonstrations," *Autonomous Robots*, vol. 43, no. 1, pp. 239–257, 2019.

[49] F. Suárez-Ruiz and Q. Pham, "A framework for fine robotic assembly," in *ICRA*, 2016.

[50] R. A. Knepper, T. Layton, J. Romanishin, and D. Rus, "Ikeabot: An autonomous multi-robot coordinated furniture assembly system," in *ICRA*, 2013.

[51] J. Thomason, D. Gordon, and Y. Bisk, "Shifting the Baseline: Single Modality Performance on Visual Navigation & QA," in *NAACL*, 2019.

[52] A. Daruna, W. Liu, Z. Kira, and S. Chernova, "Robocse: Robot common sense embedding," in *ICRA*, 2019.

TABLE A1
OBJECT-ONLY MODEL ABLATIONS FOR THE **ALL PAIRS** TASK.

	Model (M) Object Data		**Prediction Correct** ↑	
	Lang	Vis	*in*	*on*
Dev Fold	✓	$\vec{0}$	$.86 \pm .02$	$.76 \pm .01$
	$\vec{0}$	✓	$.94 \pm .01$	$.79 \pm .01$
	✓	✓	$.86 \pm .04$	$.78 \pm .01$
	Baseline (MC)		$.87 \pm .00$	$.73 \pm .00$
	Baseline (Rand)		$.49 \pm .07$	$.50 \pm .03$
Test Fold	✓	$\vec{0}$	$.86 \pm .01$	$.83 \pm .01$
	$\vec{0}$	✓	$.88 \pm .02$	$.82 \pm .01$
	✓	✓	$.87 \pm .02$	$.83 \pm .01$
	Baseline (MC)		$.84 \pm .00$	$.83 \pm .00$
	Baseline (Rand)		$.51 \pm .06$	$.51 \pm .03$

✓indicates signal was included.

APPENDIX

A. *Model Ablations*

To examine the contribution of each modality among egocentric camera images, object language referring expressions, and object static images, we perform a unimodal ablation analysis [51]. Under this ablation framework, the model architecture under examination remains unchanged, but ablated input modalities are set to zero tensors ($\vec{0}$) at both training and inference time.

Performance of the M^{obj} models on the auxiliary task of predicting Mechanical Turk workers' annotations for whether objects can be stacked *in* or *on* one another given object images and referring expressions is given in Table A1 for the *Test* and *Development* data folds on **All Pairs**. In most cases, the prediction model using both language and vision signals achieves matching or higher accuracy than majority class, with the exception of losing about 1% on the *Development* fold on the *in* annotation prediction task. The language-only and vision-only ablations reveal that either modality is competitive for predicting *in* and *on* annotations, with vision-only having the edge in most cases.

This auxiliary task facilitates pretraining the $M^{\text{ego+pre}}$ model based on the $M^{\text{ego+obj}}$ architecture. Table A2 examines the performance of these models when ablating all three of egocentric, object language, and object vision modalities for detecting the results of actions on the **Robot Pairs**. Note that models with no egocentric input are essentially *predicting* what happens to pairs of objects, as opposed to *detecting* it from available egocentric scene information.

There are three high-level takeaways from these ablations. First, In almost all cases, pretraining improves detection accuracy. Where performance does not improve, it remains the same. Second, performance on the *prediction* task for *in*, that is, when ablating the egocentric modality out, is competitive with the *detection* task of actually surveying the scene. For the more complex *on* relationship, which can take different forms (as discussed in Section VI), using the egocentric camera information improves performance over object priors alone. Finally, as in the **All Pairs** prediction experiments, both the language-only and vision-only ablations of object data perform well, suggesting that either alone may be enough to augment a detection model if the other is unavailable at inference time in practice.

TABLE A2
$M^{\text{ego+obj}}$ AND $M^{\text{ego+pre}}$ MODEL ABLATIONS ON **ROBOT PAIRS**.

	Model (M)	Object Data		**Detection Correct** ↑	
	Ego	Lang	Vis	*in*	*on*
Dev Fold	$\vec{0}$	✓	$\vec{0}$	$.70 \pm .03$	$.56 \pm .10$
	$\vec{0}$	pre	$\vec{0}$	$.72 \pm .04$	$.57 \pm .09$
	$\vec{0}$	$\vec{0}$	✓	$.71 \pm .08$	$.50 \pm .06$
	$\vec{0}$	$\vec{0}$	pre	$.72 \pm .07$	$.53 \pm .05$
	$\vec{0}$	✓	✓	$.76 \pm .08$	$.58 \pm .05$
	$\vec{0}$	pre	pre	$.78 \pm .08$	$.60 \pm .04$
	✓	✓	$\vec{0}$	$.67 \pm .08$	$.60 \pm .08$
	✓	pre	$\vec{0}$	$.68 \pm .08$	$.62 \pm .08$
	✓	$\vec{0}$	✓	$.70 \pm .10$	$.58 \pm .11$
	✓	$\vec{0}$	pre	$.72 \pm .08$	$.59 \pm .13$
	✓	✓	✓	$.70 \pm .09$	$.59 \pm .07$
	✓	pre	pre	$.73 \pm .09$	$.62 \pm .07$
	Baseline (MC)			$.32 \pm .00$	$.36 \pm .00$
	Baseline (Rand)			$.49 \pm .06$	$.50 \pm .06$
Test Fold	$\vec{0}$	✓	$\vec{0}$	$.79 \pm .02$	$.45 \pm .05$
	$\vec{0}$	pre	$\vec{0}$	$.79 \pm .02$	$.48 \pm .07$
	$\vec{0}$	$\vec{0}$	✓	$.80 \pm .04$	$.46 \pm .09$
	$\vec{0}$	$\vec{0}$	pre	$.81 \pm .04$	$.48 \pm .06$
	$\vec{0}$	✓	✓	$.80 \pm .03$	$.55 \pm .04$
	$\vec{0}$	pre	pre	$.79 \pm .04$	$.55 \pm .04$
	✓	✓	$\vec{0}$	$.75 \pm .06$	$.54 \pm .10$
	✓	pre	$\vec{0}$	$.80 \pm .02$	$.57 \pm .07$
	✓	$\vec{0}$	✓	$.75 \pm .11$	$.57 \pm .10$
	✓	$\vec{0}$	pre	$.80 \pm .05$	$.56 \pm .10$
	✓	✓	✓	$.74 \pm .07$	$.59 \pm .08$
	✓	pre	pre	$.77 \pm .05$	$.59 \pm .06$
	Baseline (MC)			$.20 \pm .00$	$.32 \pm .00$
	Baseline (Rand)			$.52 \pm .05$	$.51 \pm .07$

✓indicates signal was included, while "pre" indicates models with object features pretrained from **All Pairs** data.

B. *YCB Object Details*

We make a number of small changes and omissions from the full YCB Object Set when establishing our included objects Y (Section III). In particular, we:

- exclude objects lacking camera image data used as vision information to the augmented model (Fig. 3);
- exclude `072-*_toy_airplane` parts `b-k`;
- do not split the two similar, *medium-most-sized* `063-f_cups` and `063-e_cups` objects into different folds, to evaluate more conservatively; and
- for `063-j_cups`, we sometimes use a same-sized cup that is light blue instead of yellow during robot trials.

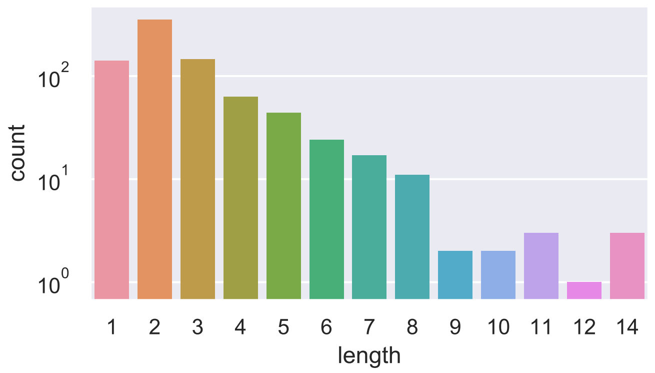


Fig. 4. Distribution of language description lengths.

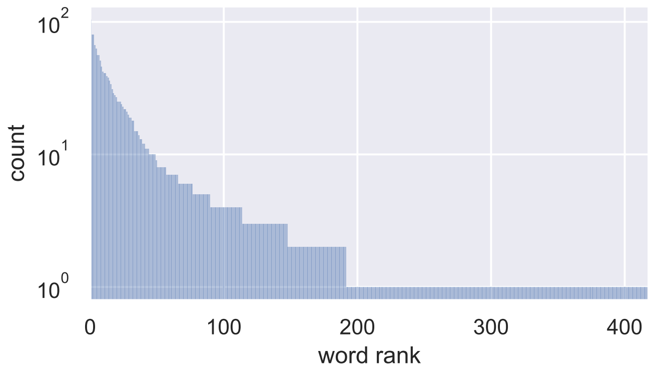


Fig. 5. Frequency count for the nth ranked word in our descriptions.

C. Language Analysis

Our language annotations range up to 14 words in length (Figure 4) and the importance of using pretrained vectors becomes clear as we see a large number of rare words (matching the expected Zipfian distribution of lexical tokens) in Figure 5. First we analyze the description lengths and find that most responses contain fewer than five words.

Short sentences can take the form of a single entity (e.g. *"apple"*) or have a single adjective (e.g. *"blue cylinder"*). In contrast, long descriptions may deviate from descriptions of form to function:

thing you use to see how many spaces you move during a board game

Understanding this requires much richer textual representations about the world. Relatedly, precise descriptions which avoid using the name of an object also result in long descriptions. Here we have a description of physical characteristics of the same die:

white cube with different number of black dots on each side.

A similar pattern emerges when looking at the most and least frequent lexical types. The most common are common adjectives and types:

black, red, blue, yellow, cup, ball, box, plastic, ...

while the least common contain nuanced textures, world references, typos and counting:

bumps, bloch, fleshy, eight, cleaning, avocado, ...

In this work, we use GloVe vectors as our pretrained world representations. We leave the question of building better language representations that account for properties of the physical world to future work [52].

D. Annotation Details

For *on* and *in* labels across the five trials gathered for each *Robot Pair*, the outcome label was annotated in {*Yes*, *No*, *Maybe*}. The *Maybe* annotation denotes that there was mixed success across the five trials. Similarly, for the Mechanical Turk annotations used during the auxiliary task for data augmentation, when annotators disagreed about the annotation in {*Yes*, *No*}, *Maybe* was assigned. Throughout our experiments, we round *Maybe* annotations to the *No* label for both robot trial and Mechanical Turk annotations.

For Mechanical Turk, annotators answer "yes", "no", or "yes, but only if object A is rotated." We rounded this final option down to "no" for our task, but the original label may be useful for other manipulation tasks.