Affordance-based Robot Object Retrieval

Thao Nguyen, Nakul Gopalan, Roma Patel, Matt Corsaro, Ellie Pavlick, Stefanie Tellex
{thaonguyen, romapatel, matthew_corsaro, ellie_pavlick}@brown.edu
nakul_gopalan@gatech.edu, stefie10@cs.brown.edu

***Abstract*—Natural language object retrieval is a highly useful yet challenging task for robots in human-centric environments. Previous work has primarily focused on commands specifying the desired object's type such as "scissors" and/or visual attributes such as "red," thus limiting the robot to only known object classes. We develop a model to retrieve objects based on descriptions of their usage. The model takes in a language command containing a verb, for example "Hand me something to *cut*," and RGB images of candidate objects; and outputs the object that best satisfies the task specified by the verb. Our model directly predicts an object's appearance from the object's use specified by a verb phrase, without needing an object's class label. Based on contextual information present in the language commands, our model can generalize to unseen object classes and unknown nouns in the commands. Our model correctly selects objects out of sets of five candidates to fulfill natural language commands, and achieves a mean reciprocal rank of 77.4% on a held-out test set of unseen ImageNet object classes and 69.1% on unseen object classes *and* unknown nouns. Our model also achieves a mean reciprocal rank of 71.8% on unseen YCB object classes, which have a different image distribution from ImageNet. We demonstrate our model on a KUKA LBR iiwa robot arm, enabling the robot to retrieve objects based on natural language descriptions of their usage[1]. We also present a new dataset of 655 verb-object pairs denoting object usage over 50 verbs and 216 object classes[2].**

I. Introduction

A key bottleneck in widespread deployment of robots in human-centric environments is the ability for non-expert users to communicate with robots. Natural language is one of the most popular communication modalities due to the familiarity and comfort it affords a majority of users. However, training a robot to understand open-ended natural language commands is challenging since humans will inevitably produce words that were never seen in the robot's training data. These unknown words can come from paraphrasing such as using "saucer" instead of "plate," or from novel object classes in the robot's environments, for example a kitchen with a "rolling pin" when the robot has never seen a rolling pin before.

We aim to develop a model that can handle open-ended commands with unknown words and object classes. As a first step in solving this challenging problem, we focus on the natural language object retrieval task — selecting the correct object based on an indirect natural language command with constraints on the object's functionality. More specifically, our work focuses on fulfilling commands requesting an object for a task specified by a verb such as "Hand me a box cutter to **cut**." Being able to handle these types of commands is highly useful for a robot agent in human-centric environments, as people usually ask for an object with a specific usage in mind. The robot would be able to correctly fetch the desired object for the given task, such as cut, without needing to have seen the object, a box cutter for example, or the word representing the object, such as the noun "box cutter." In addition, the robot has the freedom to substitute objects as long as the selected object satisfies the specified task. This is particularly useful in cases where the robot cannot locate the specific object the human asked for but found another object that can satisfy the given task, such as a knife instead of a box cutter to cut.

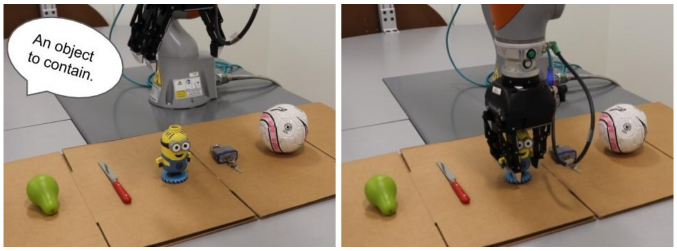


Fig. 1. Our robot receives segmented RGB images of the objects in the scene and a natural language command "An object to contain," and correctly retrieved the Minion (yellow cartoon character)-shaped container.

There has been much prior work on natural language object retrieval [16, 14, 3, 4] and similar areas such as image captioning and image retrieval [23, 19, 30, 32]. However, previous work primarily focuses on natural language commands that either specify the object class such as "scissors" or describe its visual attributes such as "red," "curved," "has handle," and cannot handle unknown object classes or words. Our work, in contrast, anchors the desired object to its usage (which is specified by a verb) and reasons about the verb to handle unknown objects and nouns on-the-fly.

Our work demonstrates that an object's appearance provides sufficient signals to predict whether the object is suitable for a specific task, without needing to explicitly classify the object class and visual attributes. Our model takes in RGB images of objects and a natural language command containing a verb, generates embeddings of the input language command and images, and selects the image most similar to the given command in embedding space. The selected image should represent the object that best satisfies the task specified by the verb in the command. We train our model on natural language command-RGB image pairs. The evaluation task for the model is to retrieve the correct object from a set of five

[1] Video recordings of the robot demonstrations can be found at https://youtu.be/WMAdGhMmXEQ.

[2] The dataset and code for the project can be found at https://github.com/Thaonguyen3095/affordance-language.

images, given a natural language command. We use ImageNet images and language commands generated from verb-object pairs extracted from Wikipedia to train our model. Our model achieves a mean reciprocal rank of 77.4% on a held-out test set of unseen ImageNet object classes and 69.1% on unseen object classes *and* unknown nouns. Our model also achieves a mean reciprocal rank of 71.8% on unseen YCB object classes, which have a different image distribution from ImageNet. In addition, we demonstrate our model on a KUKA LBR iiwa robot arm, enabling the robot to retrieve objects based on natural language commands, and present a new dataset of 655 verb-object pairs denoting object usage over 50 verbs and 216 object classes.

Portions of this work were previously published at Robotics: Science and Systems 2020 [22]. This version extends the model evaluation by adding a qualitative evaluation of the learned task embeddings. In addition to the previously reported retrieval accuracies, we calculate and report the model's mean reciprocal ranks. We also evaluate our model's retrieval performance over varying numbers of candidate objects.

II. Related Work

Natural language object retrieval refers to the task of finding and recovering an object specified by a human user using natural language. The computer vision and natural language grounding communities attempt to solve object retrieval by locating or *grounding* the object specified in an image using natural language [16, 14, 26]. Krishnamurthy and Kollar [16] use a dataset of RGB images with segmented objects and their natural language descriptions to learn the grounding of words to objects in the image by exploiting repeated occurrences of segmented objects within images, along with their descriptions in natural language. Hu et al. [14] use a similar approach albeit using deep neural networks to avoid parsing and feature construction by hand. Schlangen et al. [26] learn individual classifiers for words and compose them to resolve object references. Similar to Krishnamurthy and Kollar [16] and Hu et al. [14], they focus on words that specify object categories or visual attributes such as "cup," "red," "right," and verbs that describe human activities such as "stand," "run," and cannot handle unknown object classes. Chen et al. [3] learn joint embeddings of language descriptions and colored 3D objects for text-to-shape retrieval and generation of colored 3D shapes from natural language. Cohen et al. [4] learn joint embeddings of language descriptions and segmented depth images of objects for object retrieval within instances of the same object class. Our work, in contrast, learns an embedding across object classes based on their suitability for a given task specified using natural language. Our object embeddings are not conditioned on the output class of objects, but on the relevancy of the object for the specified task. This, therefore, allows us to retrieve objects based on descriptions of their usage and importantly allows handling of unknown nouns and unseen object classes.

Another relevant line of work is image captioning and image retrieval, which also aims to jointly model a natural language sequence and image content. The SUN scene attribute dataset [23] maps images to attributes such as "hills," "houses," "bicycle racks," "sun," etc. Such understanding of image attributes provides scene category predictions and high level scene descriptions, for example "human hiking in a rainy field." Methods based on recurrent neural networks (RNNs) [19, 30, 32] trained to directly model the probability distribution of generating a word given previous words and an image have shown to be effective in image caption generation, natural language image retrieval, and image caption retrieval. Our work is most similar to earlier attribute based image retrieval work. However, these models are trained on attributes that are directly specified and not inferred from indirect task based queries. Implicit task-based object attributes are harder to learn but are also more general than directly specifiable object visual attributes, and are useful for a natural language object retrieval system to have in its toolbox.

Object functionalities or *affordances* have been studied for a long time. Gibson [9] defines affordances as the "action possibilities" available to an agent, which covers a large range of actions. Chao et al. [2] mine the web for the knowledge of semantic affordance — given an object, determining whether an action can be performed on it. In contrast, our work focuses on actions that can be performed *with* the objects, which is similar to the affordances studied by Myers et al. [21], Do et al. [5], and Fulda et al. [8]. However, Myers et al. and Do et al. focus on detecting the affordances of object parts, and only work with a small number of object classes and affordances. Fulda et al. extract verb-noun pairs from word embeddings trained on Wikipedia, but do not take into account objects' visual appearance and cannot handle unknown objects.

Also related to our work are methods on interactive object retrieval [31] and language grounding [27, 10]. These methods perform inference over dialogue to deduce the right object based on the specifications provided by the human user. However, they use known object corpora and directly specify the object attributes. Our work, in contrast, retrieves objects based on contextual information about the task specified by the language command. We are not performing inference over dialogue, but it is a natural next step for our work where our joint embedding can prove useful in the case of novel objects.

Similar to previous work, our work aims to learn joint object representations from visual and language information using RNNs. However, previous work primarily focuses on language commands specifying the object type or visual attributes such as "scissors," "red," "has handle." In contrast, our work focuses on fulfilling commands requesting an object for a task specified by a verb, for example "Hand me something to **cut**." To handle such commands, a possible approach is to rely on accurate classification of the object type and visual attributes and an external knowledge base to query for valid verb-object or verb-attribute pairings. However, that approach would be limited to known objects and words. Our work, on the other hand, bypasses classification of object type and attributes, and directly maps object use that is specified by the verb to object appearance that is captured by the image. Our work uses the

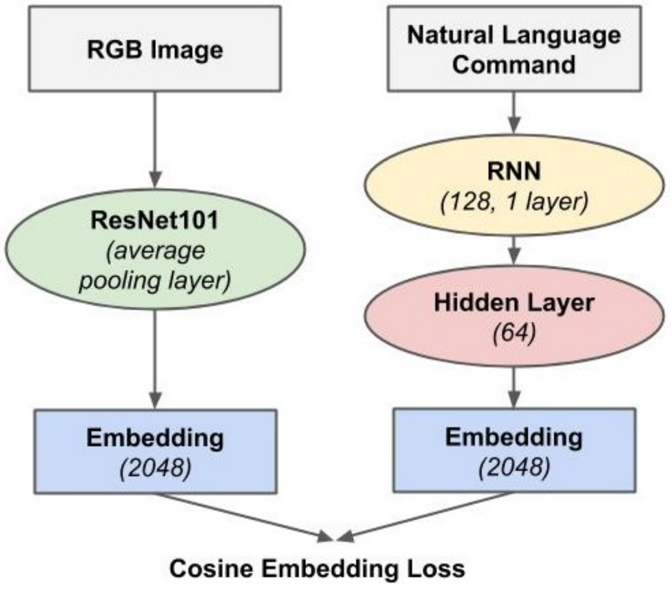


Fig. 2. Diagram of the language-vision embedding model. The model encodes given natural language commands and RGB images, and minimizes the cosine embedding loss between the language and image embeddings during training. At inference time, the model calculates the cosine similarities between the embeddings of the language command and candidate images, and selects the image most similar to the command in embedding space.

context of the verb to implicitly infer the necessary object attributes for the task, and can generalize to unseen object classes and unknown nouns in the language commands.

III. Approach

To fulfill natural language commands requesting an object for a task specified by a verb, our model generates embeddings for the language command and candidate objects and selects the object that is closest to the command in embedding space. Our model is trained using pairs of natural language object requests containing verbs and ground truth objects that satisfy the requests. We describe our model and data collection process in detail in Sections III-A and III-B.

A. Model

Given a natural language command and images of candidate objects, we want our model to correctly select the object that best satisfies the command. Our model does this by generating embeddings for the input natural language command and images, calculating the cosine similarities between the image embeddings and the language embedding, and selecting the image most similar to the command in embedding space. A diagram of our model is shown in Figure 2. Our model consists of separate image and language encoders, each in charge of generating embeddings for the input natural language commands and RGB images, respectively. During training, our model minimizes the cosine embedding loss between the embeddings of language command-RGB image pairs, thus maximizing the likelihood of the target image given the command. We describe the component image and language encoders and our model training process below.

TABLE I
Example verb-object pairs from our dataset

contain – bucket	hit – hammer	wear – necklace
contain – wardrobe	hit – racket	wear – suit
cut – cleaver	play – baseball	wrap – cloak
cut – hatchet	play – violin	wrap – handkerchief
eat – banana	serve – plate	write – notebook
eat – pizza	serve – tray	write – quill

1) Image Encoder

To encode each RGB image, we use the average pooling layer of a pretrained ResNet101 [11]. We chose ResNet101 due to ResNet's good performance on RGB images captured by common robot sensors [18]. The use of deep pretrained representations enables our model to leverage prior information of complex image features to allow for better encoding of the visual information from the images. We get an embedding of size 2048 from the pretrained ResNet model for each image.

2) Language Encoder

To encode each natural language command, our language encoder consists of a recurrent neural network (RNN) [6] followed by a fully connected layer. We randomly initialize word embeddings for each language command that are then trained from scratch. The language encoder produces an embedding vector that is the same size as the embedding produced by the image encoder.

3) Training Process

We train our model to optimize an objective function that attempts to bring the corresponding language and image embeddings closer to each other in embedding space. We achieve this by reducing the cosine embedding loss between the embedding produced by the image encoder, which takes in an RGB image of the object, and the embedding produced by the language encoder, which takes in the referring natural language command.

We describe the training data in Section III-B3. Positive training samples consist of pairs of natural language commands, each containing one verb, and RGB images of objects that can be paired with that verb. We obtain negative samples by randomly sampling an image of a different object that does not correspond with the verb and pairing the image with the language command containing the verb, resulting in a dataset of equally balanced positive and negative samples. We use Adam [15] as an optimiser with a learning rate of 0.0001 and train for 50 epochs until convergence.

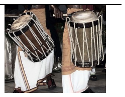
B. Data Collection

To train and evaluate our model, we require pairs of natural language commands containing verbs and RGB images of objects. To obtain these command-image pairs, we need verb-object pairs denoting valid object usage such as "cut" for a "knife." We also require RGB images for the objects. Since we are interested in testing our model's generalization capability on unseen object classes and nouns, we require a large number of object classes that can be paired with the verbs for a sufficient number of held-out object classes.

TABLE II
TEMPLATES FOR LANGUAGE COMMAND GENERATION

Verb only	An item that can <verb>.
	An object that can <verb>.
	Give me something that can <verb>.
	Give me an item that can <verb>.
	Hand me something with which I can <verb>.
	Give me something with which I can <verb>.
	Hand me something to <verb>.
	Give me something to <verb>.
	I want something to <verb>.
	I need something to <verb>.
Verb and object	Give me the <object> to <verb>.
	Hand me the <object> to <verb>.
	Pass me the <object> to <verb>.
	Fetch the <object> to <verb>.
	Get the <object> to <verb>.
	Bring the <object> to <verb>.
	Bring me the <object> to <verb>.
	I need the <object> to <verb>.
	I want the <object> to <verb>.
	I need a(n) <object> to <verb>.
	I want a(n) <object> to <verb>.

TABLE III
EXAMPLE TRAINING DATA (COMMAND-IMAGE PAIRS)

Verb-Object	Language Command	Image
contain – cup	*Give me an item that can contain*	
	I need something to contain	
play – drum	*Hand me something to play*	
	I want an object to play	
wear – kimono	*An item to wear*	
	Give me something to wear	

To the best of our knowledge, no existing datasets of verb-object pairs met our requirements. Chao et al. [2] collected a semantic affordance dataset of verb-noun combinations. However, their dataset only has 20 object classes, and focuses on verbs denoting actions that can be performed on the objects, such as "hunt" a "bird," rather than the objects' usage. Other works on semantic affordances [21, 5] also provide datasets of objects labeled with their affordances. However, these datasets contain fewer than 20 object classes and fewer than 10 affordances. Fulda et al. [8] extracted verb-noun pairs from Wikipedia to enable an agent to play text-based adventure games, but do not consider the objects' visual appearances. We, therefore, decided to collect our own dataset of valid verb-object pairs and use it to generate natural language commands paired with RGB images. We describe our data below.

1) Vision Data

We use RGB images and object classes from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC2012) validation set [25]. We chose this dataset as it has 1000 object classes and a variety of images per object class, and we want our model to work on many different object classes and object instances. The ImageNet object classes such as "violin," "suit," and "quill" are usually nouns that occur frequently in textual data in correspondence with other verbs such as "play," "wear," "write."

2) Language Data

We extracted sentences from Wikipedia containing the ImageNet object classes and used spaCy [13] to parse the sentences and extract corresponding verb-object pairs. We originally sought out to extract verb-object pairs from the common-sense knowledge base ConceptNet [28], which ended up being too small and was missing many valid verb-object pairings. We then decided to use Wikipedia instead for its large text corpus. However, the resulting dataset was highly noisy with 20,198 verb-object pairs, containing many abstract verbs such as "name," "feature," "use" that can be paired with any object, or nouns in the wrong word sense such as "suit" in "follow suit" and "file suit" that did not refer to physical objects and were not relevant for the natural language object retrieval task we were interested in. Therefore, we manually annotated the verb-object pairs to retain only concrete verbs paired with nouns in the correct sense such as "wear" and "suit," where "suit" refers to the clothing item. This resulted in a dataset with 655 verb-object pairs over 50 verbs and 216 object classes. Example verb-object pairs from our dataset are shown in Table I.

3) Training and Testing Data

We use 80% of the 216 object classes and their corresponding 535 verb-object pairs to generate our training data. The training data consist of natural language commands paired with RGB images. For each verb-object pair, we randomly select sentence templates to generate language commands and pair them with different image instances of that object class. Table II lists the templates we use. Examples of the training data are shown in Table III. Rather than using only the verbs and/or nouns from the verb-object pairs as language data for our model, we generate and give our model natural language sentences so that it can handle more realistic language commands, as most people do not ask for objects with one or two-word commands such as "knife" or "knife cut."

We hold out 20% of the object classes for testing. Test examples consisting of language command-set of five images pairs are randomly generated from objects in the test set and their corresponding verb-object pairs. The evaluation task is to select the correct object from a set of five images given a natural language command.

TABLE IV
OBJECT RANKINGS BASED ON THEIR EMBEDDINGS' SIMILARITY SCORE TO THE MODEL'S LEARNED EMBEDDING FOR THE VERB "WEAR"

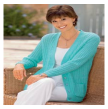
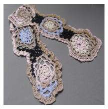
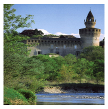
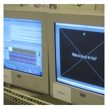
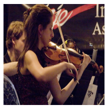
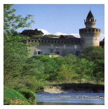
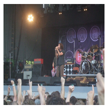
	Top-10			Bottom-10	
Image	Label	Score	Image	Label	Score
	apron	0.3966		stage	-0.0158
	apron	0.3429		monastery	-0.0198
	cardigan	0.3168		castle	-0.0230
	wool	0.3057		mosque	-0.0245
	gown	0.2947		castle	-0.0289
	shield	0.2809		library	-0.0348
	gown	0.2584		library	-0.0374
	shield	0.2522		missile	-0.0460
	diaper	0.2459		library	-0.0477
	puck	0.2437		mosque	-0.0837

TABLE V
OBJECT RANKINGS FOR THE VERB "PERFORM"

	Top-10			Bottom-10	
Image	Label	Score	Image	Label	Score
	sax	0.4795		palace	0.0741
	stage	0.4757		violin	0.0741
	sax	0.4039		castle	0.0733
	violin	0.3862		screen	0.0701
	violin	0.3583		palace	0.0636
	violin	0.3401		castle	0.0536
	cleaver	0.3359		missile	0.0500
	shield	0.3195		monastery	0.0454
	shield	0.2872		mosque	0.0287
	stage	0.2811		missile	-0.001

IV. EXPERIMENTS AND RESULTS

A. Qualitative Evaluation of Learned Task Embeddings

We rank all test objects based on their embeddings' similarity to the model's output embedding for each task (verb) and analyze the ranked lists to qualitatively evaluate the learned task embeddings. We use cosine similarity as the similarity metric. The test set contains 43 object classes and 118 object instances, which belong to the held-out 20% of the object classes. We report the top-10 and bottom-10 ranked objects for 6 verbs: "wear," "perform," "play," "serve," "contain," and "cut" in Tables IV–IX. For each object, we show its image, class label, and embedding similarity score.

Table IV shows the results for "wear." The top-9 objects are all items that can be worn. And while the 10th object is labeled as a puck, its image mostly depicts people in hockey uniforms, which are wearable objects. Our model was able to learn good embeddings for both tasks.

Results for "perform" can be found in Table V. The majority of the top-10 objects are either musical instruments or stages, which are both compatible with "perform." Cleavers and shields are most likely incorrectly included in the top-10 due to their metallic appearances, which is a characteristic shared by several musical instruments in the training set (such as flute, gong, and trombone). In addition, when comparing the top-10 and bottom-10 ranked objects, we see that the images for the top-10 objects have warmer tones, whereas those for the bottom-10 have cool tones. This contrast likely led to the violin with a black-and-white image being misranked and included in the bottom-10.

We report the results for "play" in Table VI. Similar to "perform," most of the top-10 objects are musical instruments

TABLE VI
OBJECT RANKINGS FOR THE VERB "PLAY"

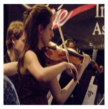
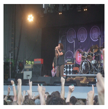
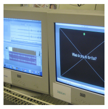
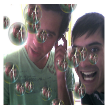
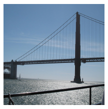
	Top-10			Bottom-10	
Image	Label	Score	Image	Label	Score
	sax	0.5187		corn	0.0663
	stage	0.4542		castle	0.0657
	sax	0.4014		screw	0.0636
	violin	0.3841		burrito	0.0610
	stage	0.3668		library	0.0574
	stage	0.3624		missile	0.0569
	violin	0.3085		screen	0.0531
	library	0.2930		wool	0.0370
	castle	0.2831		cheeseburger	0.0361
	violin	0.2797		screw	0.0339

TABLE VII
OBJECT RANKINGS FOR THE VERB "SERVE"

	Top-10			Bottom-10	
Image	Label	Score	Image	Label	Score
	restaurant	0.5170		palace	0.0652
	burrito	0.4788		castle	0.0609
	goblet	0.4260		palace	0.0558
	goblet	0.4017		mosque	0.0536
	mortar	0.3703		screen	0.0483
	bubble	0.3374		palace	0.0443
	restaurant	0.3322		stage	0.0378
	cheeseburger	0.3296		castle	0.0376
	diaper	0.3252		pier	0.0360
	syringe	0.3163		pier	0.0157

or stages, which are good matches for "play." The 8th and 9th objects' labels are library and castle, respectively, which do not seem to match the task. However, the image for the library depicts a performance or play on an impromptu stage, while the castle's image with bright lights and a clear path resembles the appearance of a stage. In the training set, "play" is also paired with sports equipment such as basketball, baseball, and volleyball. Puck is the only sports equipment in the test set, and a puck is ranked at number 11 in its suitability for "play."

Table VII shows the results for "serve." The top-10 objects include food, dishware, and restaurants, all of which are suitable for serving. Although mortar does not seem to be a good match for the task, its image resembles a cup with a spoon, which can be used to serve. The bubble, diaper, and syringe are incorrectly included in the top-10 objects. This is most likely due to the presence of people in their images, as people are often seen in the images of objects that are paired with "serve" in the training data

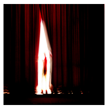
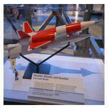
Results for "contain" can be found in Table VIII. Most of the top-10 objects are buildings, which can contain many things, but are not often what people have in mind when asking for objects to contain. However, the 4th and 9th objects, tub and restaurant, are reasonable choices for containers. The restaurant's image depicts a dining table with a glass bowl in the center, as well as multiple wine glasses and bottles, which are all suitable for containing. Goblets, another good match for "contain," are ranked relatively high at numbers 21, 31, and 39 (the majority of objects at number 11-20 are other buildings).

Table IX shows the results for the verb "cut." The top choice, cleaver, has a significantly higher similarity score (0.4985) than other objects in the top-10 (<= 0.3746). It is

TABLE VIII
OBJECT RANKINGS FOR THE VERB "CONTAIN"

Top-10			Bottom-10		
Image	Label	Score	Image	Label	Score
	library	0.4577		swing	0.0977
	library	0.4530		screw	0.0969
	restaurant	0.4320		broom	0.0935
	tub	0.4205		fly	0.0928
	monastery	0.4196		violin	0.0866
	mosque	0.4185		cleaver	0.0830
	library	0.4068		screw	0.0495
	monastery	0.4005		stage	0.0477
	restaurant	0.3985		sax	0.0462
	library	0.3902		screw	0.0312

TABLE IX
OBJECT RANKINGS FOR THE VERB "CUT"

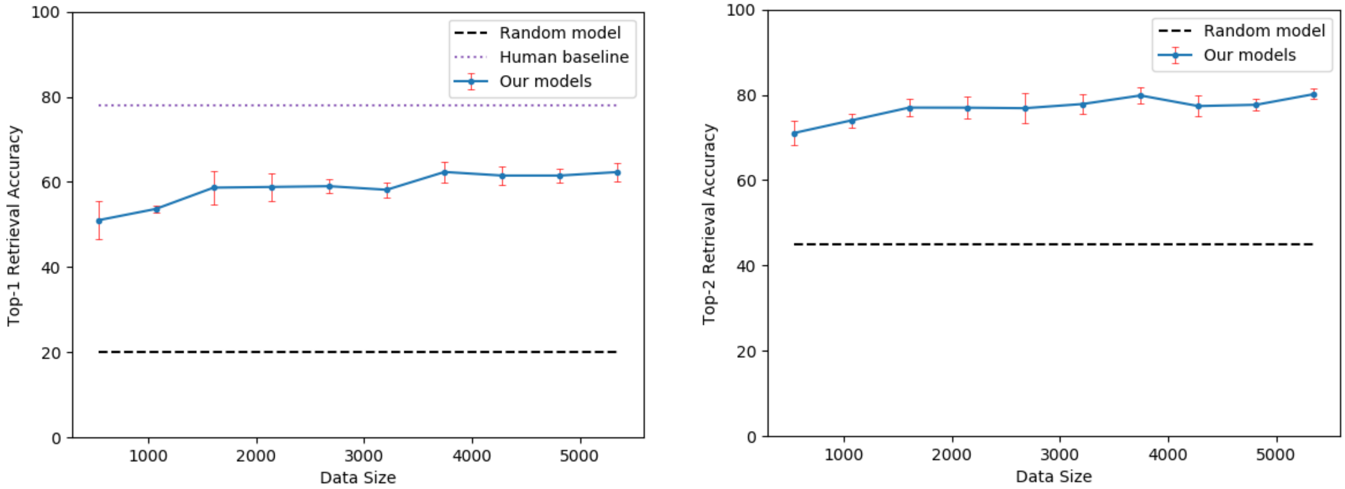
Top-10			Bottom-10		
Image	Label	Score	Image	Label	Score
	cleaver	0.4985		pier	0.0493
	screw	0.3746		screen	0.0479
	broom	0.3494		stage	0.0435
	screw	0.2979		stage	0.0404
	missile	0.2944		restaurant	0.0336
	tub	0.2884		palace	0.0281
	violin	0.2807		mosque	0.0231
	screw	0.2773		library	0.0099
	goblet	0.2635		palace	0.0081
	swing	0.2517		castle	-0.0518

also the only object instance in the test set that can be paired with "cut." While the 6th object is labeled as a tub, its image includes a pair of pliers, which can indeed be used to cut. The image for the 9th object, goblet, shows different goblets with decorative stems, the majority of which has sharp-looking parts that can be mistaken to be fit for cutting. The rest of the top-10 objects mostly have metallic appearances and/or slim bodies, which are characteristics shared by objects that are paired with "cut" in the training set.

Our model was able to learn good embeddings for a number of verbs to perform object retrieval based on descriptions of the object's usage. However, it was unable to learn better embeddings for more difficult verbs such as "serve" and "contain." As our model does not take into account the objects' labels for the retrieval task, it performs well even in cases where the label does not reflect what is shown in the object's image. The downside of this is that the model relies solely on the objects' appearances, which can be affected by extraneous properties such as the image tone and lighting conditions. Furthermore, our model learns to output embeddings that are close to the corresponding objects' embeddings, and thus is confined within the structure of the image encoder's output embedding space. Augmenting the data with depth information and image color augmentation techniques might help the model learn to pay less attention to extraneous image properties and achieve better generalization.

B. *Quantitative Evaluation*

The aim of our evaluation is to test our model's ability to accurately select objects based on natural language descriptions of their usage, given unseen object classes and unknown nouns in the language commands. Generalization to unseen



(a) Top-1 retrieval accuracies (%)

(b) Top-2 retrieval accuracies (%)

Fig. 3. Average retrieval accuracies over sets of 5 candidate objects (from unseen object classes from the held-out ImageNet object set) of models trained on different data sizes, represented by the solid lines, with vertical bars denoting standard errors. Models trained on larger data sizes usually perform better. All our models significantly outperform a random model, represented by the dashed lines. The dotted line represents the human object retrieval baseline.

object classes is much more difficult than to unseen instances of known object classes, as different instances of the same object class such as two bottles would usually look more alike than instances from different object classes such as a bottle and a bowl, even if those object classes can be used for similar tasks such as "contain."

The trained model is tested on natural language object retrieval tasks: given a language command containing a verb, along with five objects, it must output the correct object that can be paired with the verb. The evaluation task is modeling a typical in-the-wild retrieval task where there are a few objects on a table and the robot has to pick the correct one. Retrieval examples consisting of a language command paired with a set of five images are randomly generated from objects in the test set and their corresponding verb-object pairs. We test our model on several different test sets — the held-out ImageNet object classes and YCB object set — and report average top-1 and top-2 retrieval accuracies, as well as the mean reciprocal rank. Top-1 accuracy means that the model's top choice is the correct answer, and top-2 accuracy means that the correct answer is among the model's top-2 choices. Mean reciprocal rank (MRR) is the average of the multiplicative inverse of the correct answer's rank in the model's output ordered list of choices.

1) Held-out ImageNet Object Set

We first test our model on object classes held out from our dataset, which have a similar image distribution to that of the training set, as the images all come from the ILSVRC2012 validation set. We evaluate our model under 2 different settings, representing increasingly difficult scenarios. We describe the test settings and our model's performance in each case below. For both cases, the test object classes are held-out, meaning our model has never seen any instances belonging to the test object classes during training.

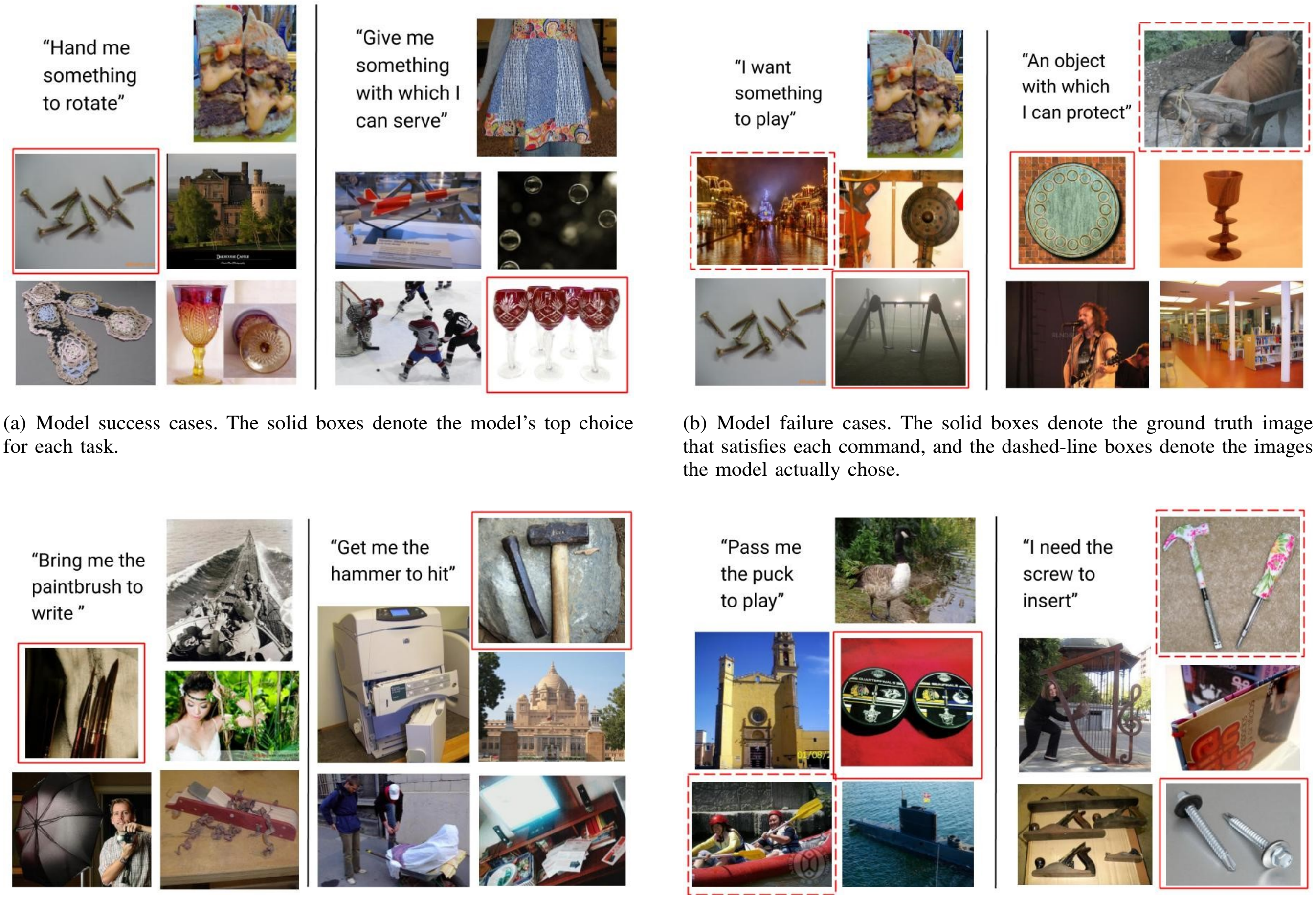
TABLE X
RETRIEVAL ACCURACIES (%) OVER SETS OF 5 CANDIDATES FROM HELD-OUT IMAGENET OBJECT CLASSES

Model	Top-1 *(Std. Error)*	Top-2 *(Std. Error)*
Random	20.0	45.0
Data size 535	51.0 *(4.50)*	71.0 *(2.84)*
Data size 1070	53.7 *(0.86)*	74.0 *(1.60)*
Data size 1605	58.7 *(3.85)*	77.0 *(2.13)*
Data size 2140	58.8 *(3.31)*	77.0 *(2.50)*
Data size 2675	59.0 *(1.68)*	76.9 *(3.55)*
Data size 3210	58.2 *(1.70)*	77.8 *(2.30)*
Data size 3745	**62.3** ***(2.48)***	79.8 *(1.94)*
Data size 4280	61.5 *(2.25)*	77.4 *(2.44)*
Data size 4815	61.5 *(1.71)*	77.7 *(1.43)*
Data size 5350	**62.3** ***(2.18)***	**80.2** ***(1.23)***
Human baseline	78.0 *(1.72)*	

a) Unseen Object Classes

We first train and test our model on language commands containing only the verbs from the verb-object pairs, such as "I want something to `<verb>`." The templates for these commands are listed under the "Verb only" row in Table II. This simplified setting where the test commands look like those in the training data, for example "Give me something to contain," helps us look at how well the model has learned to generalize the concepts associated with the verbs, such as "contain" requires objects with convexity. It removes the additional challenges that might arise with seeing unknown nouns in the commands, such as the fact that the embeddings for these nouns would be untrained. However, this is still a challenging problem because the model has never seen the object classes in the test set before.

We train separate models on increasing sizes of training data generated from the 535 verb-object pairs in the training set. Data was augmented by generating different natural language commands containing the verbs, and pairing the commands with different images of the objects from the verb-object pairs.


"Hand me something to rotate"
"Give me something with which I can serve"
"I want something to play"
"An object with which I can protect"
"Bring me the paintbrush to write "
"Get me the hammer to hit"
"Pass me the puck to play"
"I need the screw to insert"

(a) Model success cases. The solid boxes denote the model's top choice for each task.

(b) Model failure cases. The solid boxes denote the ground truth image that satisfies each command, and the dashed-line boxes denote the images the model actually chose.

(c) Model success cases. The solid boxes denote the model's top choice for each task.

(d) Model failure cases. The solid boxes denote the ground truth image that satisfies each command, and the dashed-line boxes denote the images the model actually chose.

Fig. 4. Example success and failure cases for our best models on object retrieval tasks with unseen object classes from the held-out ImageNet object set, shown in (a) and (b), and with both unseen object classes *and* unknown nouns, shown in (c) and (d). The models were given a natural language command and selected the object that they determine to best satisfy the command out of a set of five objects.

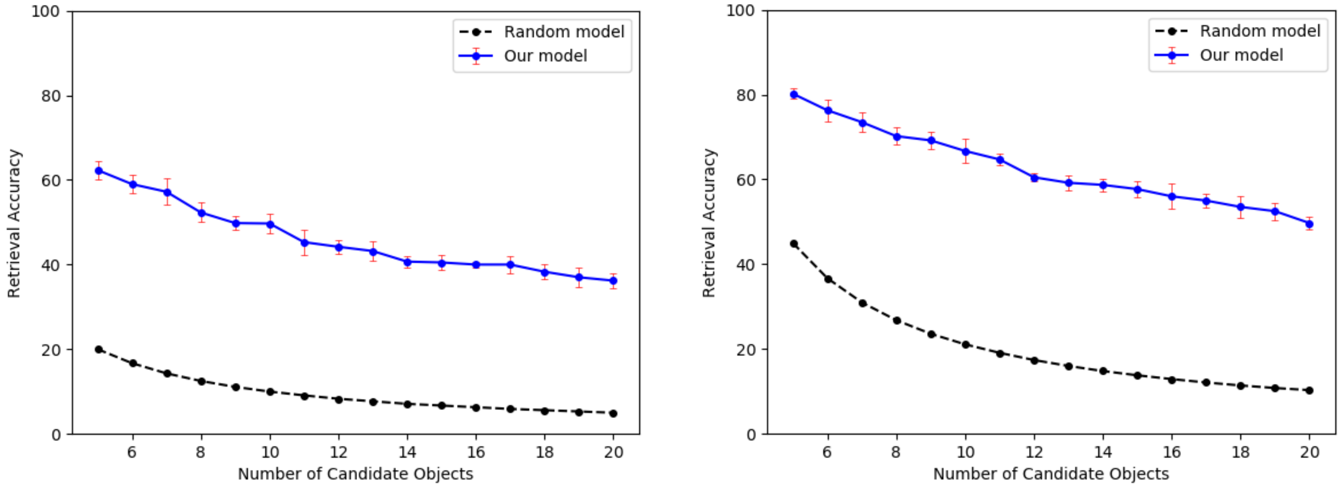
Examples of the training data are shown in Table III. The test set with 43 held-out object classes and 120 corresponding verb-object pairs and retrieval examples are fixed for all models.

We test each model trained on different data sizes 5 times and report their average top-1 and top-2 retrieval accuracies and standard errors in Table X and Figure 3. Model performance generally increases with larger training size. All our models significantly outperform a random model (which has 20% top-1 and 45% top-2 retrieval accuracy) and achieve accuracies in the 50% – 62% range for top-1, and 70% – 80% for top-2. Our best average retrieval accuracy is 62.3% for top-1 and 80.2% for top-2 with standard errors of 2.18% and 1.23%, respectively. The best mean reciprocal rank is 77.4% with a standard error of 1.68%, as shown in Table XI. Our models were able to generalize to unseen object classes.

Our models were able to select the correct object to satisfy the task specified by most verbs in our dataset such as "contain," "write," "don," "rotate," "hit," etc. The objects paired with these verbs usually have similar visual appearances and attributes. For example, a gown and a suit can both be paired with "don" and are both made of fabric. However, our models performed not as well on more abstract verbs such as "play" and "protect," as it is less obvious which object attributes are required for the tasks specified by these verbs. The objects that can satisfy the tasks come in a larger variety of visual appearances, for example a harp, a volleyball, and a swing can all be paired with "play."

Example success and failure cases for our best model are shown in Figures 4(a) and 4(b), respectively. Our model correctly selected images of screws and goblets to satisfy natural language commands "Hand me something to rotate" and "Give me something with which I can serve." The model failed to select the swing given the command "I want something to play," and did not select the shield in response to "An object with which I can protect." However, the instance of a shield in the object retrieval task shown in the right image in Figure 4(b) does not look like a shield but more like a plate, and such object instance outliers can definitely throw the model off.

We also evaluate our best model's retrieval performance



(a) Top-1 retrieval accuracies (%)

(b) Top-2 retrieval accuracies (%)

Fig. 5. Average retrieval accuracies of our best model over varying number of candidate objects (from the held-out ImageNet object classes), represented by the solid lines, with vertical bars denoting standard errors. Our model significantly outperforms a random model, represented by the dashed lines.

TABLE XI
BEST RETRIEVAL ACCURACIES AND MEAN RECIPROCAL RANKS (%) OVER VARYING NUMBERS OF CANDIDATE OBJECTS FROM HELD-OUT IMAGENET OBJECT CLASSES

Candidate number	Top-1	Top-2	MRR
5 objects	62.3 *(2.18)*	80.2 *(1.23)*	77.4 *(1.68)*
6 objects	59.0 *(2.16)*	76.3 *(2.54)*	74.2 *(1.93)*
7 objects	57.2 *(3.13)*	73.5 *(2.23)*	72.0 *(2.71)*
8 objects	52.3 *(2.30)*	70.2 *(2.01)*	68.6 *(1.14)*
9 objects	49.8 *(1.67)*	69.2 *(2.01)*	66.5 *(0.95)*
10 objects	49.7 *(2.35)*	66.7 *(2.77)*	65.5 *(2.21)*
11 objects	45.3 *(3.03)*	64.7 *(1.39)*	62.5 *(2.00)*
12 objects	44.2 *(1.70)*	60.5 *(0.91)*	60.7 *(1.09)*
13 objects	43.2 *(2.35)*	59.2 *(1.86)*	59.3 *(2.02)*
14 objects	40.7 *(1.39)*	58.7 *(1.45)*	57.8 *(0.49)*
15 objects	40.5 *(1.78)*	57.7 *(1.94)*	57.0 *(1.10)*
16 objects	40.0 *(0.62)*	56.0 *(2.96)*	56.5 *(0.97)*
17 objects	40.0 *(2.04)*	55.0 *(1.72)*	56.2 *(1.40)*
18 objects	38.3 *(1.78)*	53.5 *(2.61)*	54.4 *(1.27)*
19 objects	37.0 *(2.27)*	52.5 *(2.01)*	53.3 *(1.17)*
20 objects	36.2 *(1.72)*	49.7 *(1.39)*	51.9 *(0.99)*

over varying numbers of candidate objects. The model's average top-1 and top-2 retrieval accuracies, mean reciprocal ranks, and corresponding standard errors can be found in Table XI. Our model's performance is the best with five candidates, and decreases as the number of candidate objects increases. However, the average rates of decline in our model's retrieval accuracies are only 3.5% and 3.1% for top-1 and top-2, which are lower than those of a random model (8.8% and 9.3%, respectively), as visualized in Figure 5. Therefore, our model still significantly outperforms the random model and achieves reasonable performance levels with larger numbers of candidate objects for retrieval. For example, with 20 candidates, the random model would have a 5.0% top-1 and 10.3% top-2 retrieval accuracy, while our model achieved an average top-1 accuracy of 36.2% and 49.7% for top-2.

b) Unseen Object Classes and Unknown Nouns

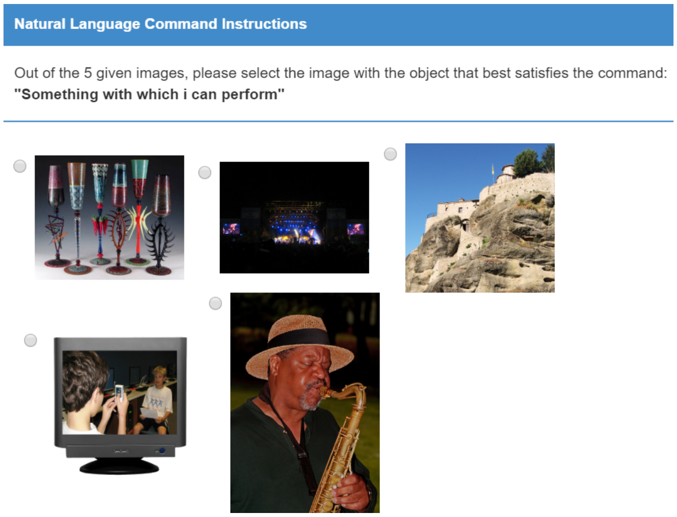
Next, we train and test our model on natural language commands containing both verbs and objects, for example "Give me the `<object>` to `<verb>`." The templates for these commands are listed under the "Verb and object" row in Table II. The model is tested on object retrieval tasks with both unseen object classes *and* unknown nouns. Testing our model in this setting is necessary because when a deep net such as our model is dealing with an unknown word, it will map the word to a random, untrained embedding. That random embedding could completely throw off the model's understanding, for example the model might pick the image for whatever noun the random embedding happens to be closest to. We need to know how our model would behave with truly unknown words in the input to get a sense of how it would work in the real world.

An example task in this setting is the model getting the command "Give me the dax to cut" with "dax" being an unknown word to the model, while also being shown object instances from classes it has never seen before. This task is more difficult than object retrieval with only unseen object classes, as the model has to figure out that unknown words such as "dax" adds no information and avoid being affected by the noise added by the unknown words.

Other than the inclusion of nouns in the language commands, the setup for this experiment is the same as that with only unseen object classes, as described in Section IV-B1a. Each model trained on different data sizes was tested 5 times. As shown in Table XII, our best average retrieval accuracy is 53.0% for top-1 and 72.8% for top-2, with standard errors of 1.33% and 3.11%, respectively. The best mean reciprocal rank is 69.1% with a standard error of 0.87%. Our models were negatively affected by the unknown words in the language commands. However, performance decline is to be expected as this is a more difficult task. Furthermore, our models' perfor-

TABLE XII
BEST RETRIEVAL ACCURACIES AND MEAN RECIPROCAL RANKS (%) OVER SETS OF 5 CANDIDATE OBJECTS FOR EACH EXPERIMENT SETTING

Setting	Top-1	Top-2	MRR
Held-out ImageNet objects	62.3 *(2.18)*	80.2 *(1.23)*	77.4 *(1.68)*
Held-out objects & nouns	53.0 *(1.33)*	72.8 *(3.11)*	69.1 *(0.87)*
Unseen YCB objects	54.7 *(1.99)*	71.9 *(2.40)*	71.8 *(1.50)*



Natural Language Command Instructions

Out of the 5 given images, please select the image with the object that best satisfies the command:
"Something with which i can perform"

○ ○ ○
○ ○

Fig. 6. Amazon Mechanical Turk interface and example task for human baseline experiment.

mance still demonstrates some generalization to unseen object classes *and* unknown nouns. A way to better handle unknown words and boost model performance in this setting would be to use pretrained word embeddings such as Word2vec [20] or GloVe [24] instead of random untrained embeddings.

Example success and failure cases for our best model are shown in Figures 4(c) and 4(d), respectively. Our model was able to correctly select the paintbrush when asked to "Bring me the paintbrush to write," and picked the hammer to satisfy the command "Get me the hammer to hit." Unfortunately, the model incorrectly selected the canoe and hammer given the commands "Pass me the puck to play," and "I need the screw to insert," respectively. However, a canoe can also be paired with "play" as canoeing is a recreational activity. In addition, the image representing the hammer also includes a screwdriver, an object that can be used to "insert."

2) *Human Retrieval Baseline*

We also compare our models' performance to a human baseline for the retrieval task. Humans are experts in natural language understanding and object grounding. The experiment was done on Amazon Mechanical Turk (AMT). We showed AMT workers five images and one language command such as "Give me something to `<verb>`," and asked them to select the image with the object that best satisfies the command. The AMT interface and an example task for the experiment is shown in Figure 6. We collected 5 answers for each of the 120 retrieval tasks. The average top-1 human retrieval accuracy is 78.0% with standard error of 1.72%, shown in Table X and Figure 3(a). The Fleiss' Kappa [7] is 0.66, which reflects a good level of inter-rater agreement. Even human users are not perfect at this task, as the given images of objects are not segmented and thus it can sometimes be confusing as to what object the image is supposed to be capturing, or the image only shows a partial/low-quality view of the object. In addition, the object usage being asked for in the language command can occasionally be unconventional such as using a "spoon" to "cut," and thus might not be obvious to the average AMT worker who is spending very little time on each task. Our models' performances are not as good as the human baseline but not far apart. Furthermore, the imperfect human performance proves how difficult of a task this is and how impressive our models' results are.

3) *YCB Object Set*

Finally, we run an evaluation to test whether the proposed model can perform natural language object retrieval on objects commonly seen and interacted with by real robots. For this evaluation, we test our best model on images of objects from the YCB Object and Model Set [1]. The YCB object set is designed for benchmarking robotic manipulation and consists of objects of daily life with different shapes, sizes, textures, etc. We did not use the YCB object set as our training image set because it has a much smaller number of object classes and only 1 instance per object class in comparison to ImageNet's 1000 object classes and 50 images per class.

Of the 65 object classes with corresponding RGB images in the YCB dataset, we select 33 object classes to test our model on, excluding classes our model has seen during training and picking only one class in the case of identical objects of differing sizes such as "S clamp," "M clamp," "L clamp," "XL clamp." Each object class in the YCB dataset is represented by one object instance, with corresponding RGB images of the object instance from multiple camera angles. We represent each selected object class by a single front-facing image of the object, taken from the YCB dataset. From the 50 verbs our model was trained on, we select 11 verbs (shown in Table XIII) that are most compatible with the 33 YCB objects and annotate valid verb-object pairings among the selected objects and verbs. This resulted in 64 verb-object pairs. Examples of the annotated verb-object pairs are shown in Table XIV. Natural language commands containing only the verbs were generated using templates (listed under the "Verb only" row in Table II), and retrieval examples consisting of sets of five images paired with language commands were randomly generated from the annotated verb-object pairs.

Our best model, without being retrained on images from the YCB dataset, was tested 5 times and achieved average retrieval accuracies of 54.7% and 71.9% with standard errors of 1.99% and 2.40% for top-1 and top-2, respectively. The model also achieved a mean reciprocal rank of 71.8% with a standard error of 1.50%, as reported in Table XII. Although these results are far from perfect, they still demonstrate generalization on a dataset with a different distribution from the model's training data. In addition, these results would enable the robot to

TABLE XIII
VERBS ANNOTATED WITH YCB OBJECTS

construct	eat	open	serve
contain	grow	play	write
cut	hit	rotate	

TABLE XIV
EXAMPLE ANNOTATED VERB-OBJECT PAIRS FOR THE YCB SET

construct – power drill	eat – apple	play – tennis ball
contain – chips can	grow – pear	rotate – adjustable wrench
contain – windex bottle	hit – spoon	serve – bowl
cut – scissors	open – padlock	write – large marker

significantly reduce its search space from all the candidate objects, and employ strategies such as question asking to further disambiguate and retrieve the correct object.

Notably, our model correctly identified a fork, a spoon, and scissors as objects that can be used to cut, while only having seen knife-like object classes such as hatchets paired with the verb "cut" in its training data. In addition, our model selected a chips can and mustard bottle when asked for something to "eat," which in retrospect are very reasonable pairings that were mistakenly left out of our verb-object pair annotations.

C. Robot Demonstrations

We implement our trained model on a KUKA LBR iiwa robot arm with a Robotiq 3-finger adaptive gripper. We pass a natural language command into our model along with manually segmented RGB images of objects in the scene, captured by an Intel RealSense camera. The robot then grasps the observed object with the highest cosine similarity in embedding space to the language command. We use object classes that our model has not seen during training for all the demonstrations.

We test our robot on four object retrieval tasks. Images capturing the tasks are shown in Figure 7. The robot correctly selected the T-shirt for the task of "Give me something to wear." When asked to "Give me an item that can write," it was able to pick out the marker from other distracting objects that are also partly red and have slim bodies. Next, it accurately identified the pear and chips can as the top two items that would satisfy "Hand me something to eat." This is the only test case with more than one possible correct answer. Finally, when asked for "An object to contain," the robot selected the empty Minion-shaped bottle. Video recordings of the robot demonstrations can be found online[3].

We mostly use YCB objects for the demonstrations with the exception of the Minion-shaped bottle in the last case, which was to test our model on an odd-looking object. While these are common objects seen in our daily life, Mask R-CNN [12], the state-of-the-art method for object segmentation and classification, was unable to correctly segment and classify most of them. This is because the objects do not belong to MSCOCO [17], the dataset Mask R-CNN was trained on[4]. The objects that are not part of the 91 object types in COCO are: T-shirt, pear, chips can, marker, clamp, lock, and of course Minion bottle. In addition to these objects, Mask R-CNN was also unable to detect the soccer ball, despite having been trained on sports balls. With such results, relying on accurate classification of objects and querying of an external knowledge base for valid verb-object pairs to select the object that satisfies the language command does not work in these cases. In contrast, our model was able to select the correct object based on the command without needing to explicitly classify the candidate objects or having seen their object classes.

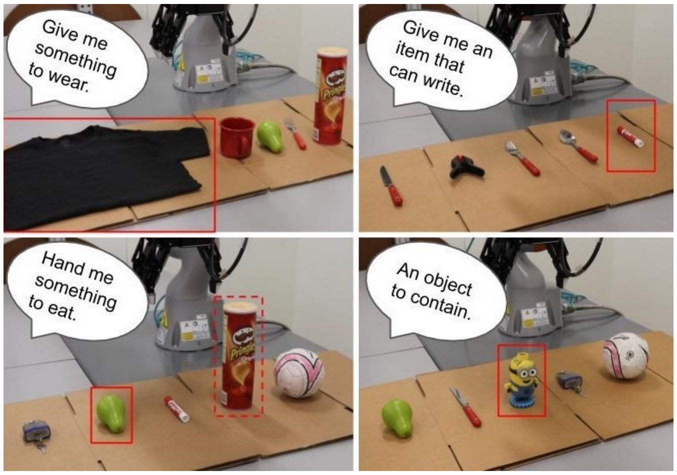

Give me something to wear.
Give me an item that can write.
Hand me something to eat.
An object to contain.

Fig. 7. Natural language object retrieval tasks demonstrated on our robot. The given language commands are: "Give me something to wear" (top left), "Give me an item that can write" (top right), "Hand me something to eat" (bottom left), and "An object to contain" (bottom right). Solid boxes denote the robot's top choice for each task. The dashed-line box denotes the robot's top second choice. Our robot retrieved the correct object for each task.

V. CONCLUSION

Understanding open-ended natural language commands is a challenging but important problem. We address a sliver of the problem by focusing on object retrieval based on descriptions of the object's usage. We propose an object retrieval model that learns from contextual information from both language and vision to generalize to unseen object classes and unknown nouns. Given natural language commands, our model correctly selects objects out of sets of five candidates, and achieves a mean reciprocal rank of 77.4% on a held-out set of unseen ImageNet object classes and 69.1% on unseen object classes *and* unknown nouns. Our model also achieves a mean reciprocal rank of 71.8% on unseen YCB object classes. We demonstrate our model on a KUKA LBR iiwa robot arm, enabling the robot to retrieve objects based on natural language descriptions of their usage. Along with our model, we also present a newly created dataset of 655 verb-object pairs denoting object usage over 50 verbs and 216 object classes, as well as the methods

[3] https://youtu.be/WMAdGhMmXEQ

[4] We did not use the COCO dataset as our training image set as it has a much smaller number of object classes in comparison to ImageNet's 1000 object classes.

used to create this dataset. To the best of our knowledge, this is the first dataset built to perform this task, and could potentially be used for a range of object retrieval tasks.

Our model currently allows us to reduce the problem of task-based object retrieval to an attribute classification problem. There is room to improve our model's performance with a different image encoder such as EfficientNet [29] and pretrained word embeddings. Image color augmentation techniques and the use of depth information may also help the model learn to ignore extraneous image properties and achieve better generalization. Furthermore, a much richer model would perform explicit inference to determine the desired object from oblique language commands. Incorporating dialogue into this framework to perform inference can be a way to incorporate human preference more directly and provide a more intuitive interface.

Acknowledgments

The authors would like to thank Prof. James Tompkin for advice on selecting the image dataset and encoder, and Eric Rosen for help with video editing. This work is supported by the National Science Foundation under award numbers IIS-1652561 and IIS-1717569, NASA under award number NNX16AR61G, and with support from the Hyundai NGV under the Hyundai-Brown Idea Incubation award and the Alfred P. Sloan Foundation.

References

[1] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The YCB object and Model set: Towards common benchmarks for manipulation research. In *Proceedings of the IEEE International Conference on Advanced Robotics*, pages 510–517, 2015.

[2] Yu-Wei Chao, Zhan Wang, Rada Mihalcea, and Jia Deng. Mining Semantic Affordances of Visual Object Categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4259–4267, 2015.

[3] Kevin Chen, Christopher B. Choy, Manolis Savva, Angel X. Chang, Thomas Funkhouser, and Silvio Savarese. Text2Shape: Generating Shapes from Natural Language by Learning Joint Embeddings. In *Asian Conference on Computer Vision*, pages 100–116. Springer, 2018.

[4] Vanya Cohen, Benjamin Burchfiel, Thao Nguyen, Nakul Gopalan, Stefanie Tellex, and George Konidaris. Grounding Language Attributes to Objects using Bayesian Eigenobjects. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2019.

[5] Thanh-Toan Do, Anh Nguyen, and Ian Reid. AffordanceNet: An End-to-End Deep Learning Approach for Object Affordance Detection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1–5, 2018.

[6] Jeffrey L. Elman. Finding Structure in Time. *Cognitive Science*, 14(2):179–211, 1990. doi: 10.1207/s15516709cog1402_1.

[7] Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[8] Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. What Can You Do with a Rock? Affordance Extraction via Word Embeddings. *arXiv preprint arXiv:1703.03429*, 2017.

[9] James J Gibson. The theory of affordances. *Hilldale, USA*, 1(2), 1977.

[10] Jun Hatori, Yuta Kikuchi, Sosuke Kobayashi, Kuniyuki Takahashi, Yuta Tsuboi, Yuya Unno, Wilson Ko, and Jethro Tan. Interactively Picking Real-World Objects with Unconstrained Spoken Language Instructions. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3774–3781, 2018.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[13] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[14] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural Language Object Retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564, 2016.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Jayant Krishnamurthy and Thomas Kollar. Jointly Learning to Parse and Perceive: Connecting Natural Language to the Physical World. *Transactions of the Association for Computational Linguistics*, 1:193–206, 2013.

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[18] Arijit Mallick, Angel P. Del Pobil, and Enric Cervera. Deep Learning based Object Recognition for Robot picking task. In *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication*, pages 1–9, 2018.

[19] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN). *arXiv preprint arXiv:1412.6632*, 2014.

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*, 2013.

[21] Austin Myers, Ching L. Teo, Cornelia Fermüller, and Yiannis Aloimonos. Affordance Detection of Tool Parts from Geometric Features. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1374–1381, 2015.

[22] Thao Nguyen, Nakul Gopalan, Roma Patel, Matthew Corsaro, Ellie Pavlick, and Stefanie Tellex. Robot Object Retrieval with Contextual Natural Language Queries. In *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020. doi: 10.15607/RSS.2020.XVI.080.

[23] Genevieve Patterson and James Hays. SUN Attribute Database: Discovering, Annotating, and Recognizing Scene Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2751–2758, 2012.

[24] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[26] David Schlangen, Sina Zarriess, and Casey Kennington. Resolving References to Objects in Photographs using the Words-As-Classifiers Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1213–1223, 2016. ISBN 9781510827585. URL http://arxiv.org/abs/1510.02125.

[27] Mohit Shridhar and David Hsu. Interactive Visual Grounding of Referring Expressions for Human-Robot Interaction. *arXiv preprint arXiv:1806.03831*, 2018.

[28] Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[29] Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[30] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and Tell: A Neural Image Caption Generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.

[31] David Whitney, Eric Rosen, James MacGlashan, Lawson L.S. Wong, and Stefanie Tellex. Reducing Errors in Object-Fetching Interactions through Social Feedback. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1006–1013, 2017.

[32] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.