Attribute Based Object Identification

Yuyin Sun, Liefeng Bo and Dieter Fox

***Abstract*— Over the last years, the robotics community has made substantial progress in detection and 3D pose estimation of known and unknown objects. However, the question of how to identify objects based on language descriptions has not been investigated in detail. While the computer vision community recently started to investigate the use of attributes for object recognition, these approaches do not consider the task settings typically observed in robotics, where a combination of appearance attributes and object names might be used in referral language to identify specific objects in a scene. In this paper, we introduce an approach for identifying objects based on natural language containing appearance and name attributes. To learn rich RGB-D features needed for attribute classification, we extend recently introduced sparse coding techniques so as to automatically learn attribute-dependent features. We introduce a large data set of attribute descriptions of objects in the RGB-D object dataset. Experiments on this data set demonstrate the strong performance of our approach to language based object identification. We also show that our attribute-dependent features provide significantly better generalization to previously unseen attribute values, thereby enabling more rapid learning of new attribute values.**

I. Introduction

Identifying objects in complex scenes is a crucial capability for an autonomous robot to understand and interact with the physical world and be of use in everyday life scenarios. Over the last years, the robotics community has made substantial progress in detection and 3D pose estimation of known and unknown objects [14], [18]. The development of features and algorithms for combining color and depth information provided by RGB-D cameras further increased the accuracy of object detection [14]. So far, virtually all work on object instance recognition assumes that each object has a unique ID by which it is referenced. However, this is not how people would use language to identify objects in everyday settings [8]. Since not every object in an environment has a unique language identifier, people often use object name and additional *attributes* such as color (blue), shape (round), size (large), material (metal) and so on to refer to specific objects. For instance, a person might say "Bring me my coffee mug; it's the blue one", or "Pick up the Oreos on the kitchen counter" (see Fig. 1). While the first situation requires a color attribute to identify the correct object, the object name "Oreos" is sufficient in the second situation.

The computer vision community recently started to investigate the use of attributes for object recognition [10]. Their work showed that it is possible to recognize new object types solely by their appearance attributes [19], or that attributes improve recognition of fine-grained object classes such as bird species [9]. However, these approaches do not consider the task settings typically observed in robotics, where a combination of appearance attributes and object names might be used together to identify specific objects in a scene, rather than describing general object categories. Recent work in semantic natural language processing introduced a joint model for language and visual attributes for the purpose of object identification [22]. Since the focus of that work was on language learning, however, only very simple objects such as uniformly colored plastic toys were considered.

Command could be "Bring me my coffee mug; it's the blue one".

Command could be "Pick up the Oreos on the kitchen counter".

Fig. 1. Object identification: Identify and visually recognize language attributes that refer to the desired object (marked by red rectangle).

In this paper, we introduce an approach for identifying objects based on appearance and name attributes (while people might use additional cues such as gestures and spatial attributes, we here focus on intrinsic attributes only). Specifically, we consider the following *object identification task*: A robot perceives a set of segmented objects, is given a sentence describing attributes of one of the objects, and has to identify the particular object being referred to. Attributes are grouped into different types: shape, color, material, and name. The attribute type "name" contains all words people

Yuyin Sun and Dieter Fox are with the Department of Computer Science & Engineering, University of Washington, Seattle, WA 98195, USA. {sunyuyin,fox}@cs.washington.edu

Liefeng Bo is with ISTC-Pervasive Computing Intel Labs, Seattle, WA 98195, USA. liefeng.bo@intel.com

might use to name an object. For instance, a name could be an object type, such as "box", or a specific product, such as "Mini Oreo". To learn rich RGB-D features needed for attribute classification, we build on recently introduced sparse coding techniques [5]. Our approach first learns codebooks from color and depth images captured by an RGB-D camera, and then uses group lasso regularization to select the most relevant codewords for each type of attribute. These attribute dependent codewords generalize much better than fully unsupervised learned codebooks, especially when only limited training samples are available.

To evaluate our approach, we collected an extensive RGB-D attribute dataset for a subset of objects taken from the RGB-D object dataset developed by Lai and colleagues [17]. Attribute values are gathered via Amazon Mechanical Turk. This dataset provides a rich selection of attribute words and names people use to describe objects. In addition to describing individual objects, we also collect a SCENE dataset, which contains multiple objects and simulates a task in which a person might command a robot to pick up an object from among multiple objects placed on a table. Experiments demonstrate that our learned appearance and name attributes are extremely well suited to identify objects.

This paper is organized as follows. After discussing related work, we present attribute types and our object attribute dataset in Section III. Then, in Section IV, we introduce our approach to attribute dependent feature learning, followed by experimental results. We conclude in Section VI.

II. RELATED WORK

This research focuses on attribute based object identification. To handle attributes used in referral language, we use attribute dependent feature learning. To the best of our knowledge, the paper presents the first study on combining attribute (and object name) learning with feature learning for object identification. In this section, we review related work on the model components.

Referral Language: Using language to refer an object in a scene to a hearer (other people or an autonomous robot) has been studied extensively by linguistics and cognitive scientists. Dale and Reiter [8] found that for the goal of identification, people tend to use easily perceivable attribute-value pairs, like *color-red*, which don't lead to false implications.

Attribute Learning: Visual attributes have received increasing attention in the computer vision community over the past few years. Learning visual attributes has been shown to be beneficial not only for improving performance of object recognition but also for transferring learned knowledge to new categories. Ferrari and Zisserman [12] learn to localize color and texture attributes from annotations captured by image search. Farhadi et al. [10] describe objects by their attributes and showed that attribute based approaches generalize well across object categories. Kumar et al. [15] propose attribute and similar classifiers for face verification and showed that such classifiers offer complementary recognition cues over low-level patch features. Lampert et al. [19] show that attributes are useful for detecting unseen object categories. Parikh et al. [24] propose to model relative attributes and showed their advantages over traditional binary attributes. Duan et al. [9] show that attributes improve recognition of fine-grained object classes such as bird species. Matuszek et al. [22] present grounded attribute learning for jointly learning visual classifiers and semantic parsers to produce rich, compositional models that span directly from sensors to meaning. In this paper, we investigate how attributes are used to identify specific objects from among a set of objects. This setting is more relevant to the robotics scenario in which people want to use language to command a robot to, for instance, pick up an object.

Feature Learning: Over the past decade, there has been increasing interest in deep learning and unsupervised feature learning for object recognition. Deep belief nets [13] learn a hierarchy of features, layer by layer, using the unsupervised restricted Boltzmann machine. The learned weights are then further adjusted to the current task using supervised information. Alternatively, hierarchical sparse coding [26] and hierarchical matching pursuit [4] have been proposed for building rich features from scratch, layer by layer, using sparse codes and spatial pooling. Very recently, such unsupervised feature learning approaches have been adapted to depth maps and 3-D point clouds for RGB-D object recognition [2], [5].

State-of-the-art attribute learning [12], [10], [15], [19], [24], [22] is still based on hand-designed features, such as SIFT [21], HOG [7], color histograms, and kernel descriptors [3]. However, these features may not be sufficiently discriminative for general attribute learning since they usually only capture a small set of recognition cues from raw data. In contrast to existing approaches, we build our work on top of unsupervised feature learning [5] and propose attribute dependent feature learning. The approach adapts the group lasso to sparsify the codebooks learned via sparse coding techniques based on supervised attribute type information.

III. OBJECT ATTRIBUTE DATASET

We developed a new RGB-D Object Attribute Dataset for training attribute classifiers. To do so, we selected 110 objects in 12 categories from the RGB-D Object Dataset [17]: *Ball*, *Coffee Mug*, *Food Bag*, *Food Box*, *Food Can*, *Garlic*, *Instant Noodle*, *Marker*, *Sponge*, *Stapler*, *Tomato* and *Water Bottle*. Fig. 2 shows the 110 objects from our dataset.

We collect labels for the attributes of objects using Amazon Mechanical Turk (AMT). In particular, we ask workers on AMT to describe color, shape, material, and name attributes of objects using simple but precise words or phrases. According to [8], those attributes are most frequently used in referral language.

We found that annotations for color, shape and material attributes are rather consistent since different workers tend to use the same words for the same objects. For these attribute types, we select the dominant words used to describe each attribute for each object instance. Overall, we collected 11 color words for the color attribute, 3 shape words for the shape attribute, and 6 material words for the material attribute

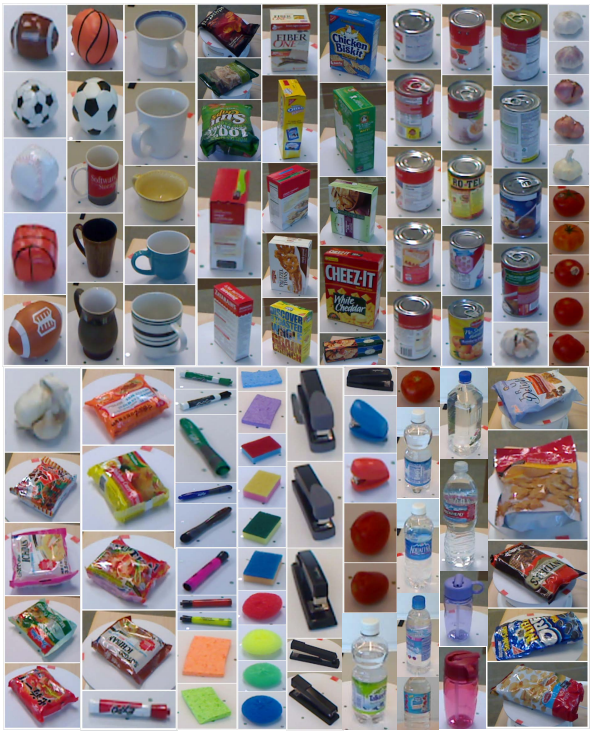


Fig. 2. 110 objects in the RGB-D Object Attribute Dataset. Each image shown here belongs to a different object.

TABLE I
WORDS USED FOR COLOR, SHAPE, MATERIAL, AND EXAMPLE NAME ATTRIBUTES IN THE OBJECT ATTRIBUTE DATASET.

Attributes	Words
Color	red, orange, yellow, green, blue, purple, pink, brown, black, white, transparent
Shape	rectangle, cylinder, ellipse
Material	foam, paper, metal, plastic, food, ceramic
Name (food bag)	bag, bag of chips, barbecue chips, food bag, archer farms potato chips, cracker bag, package, bag of cracker, chicken biscuit, bag of pretzels, Oreo, snack, Snyder's pretzels, bag of Sun chips
Name (instant noodles)	instant noodles, ramen noodles, asian food, bag, Ichiban instant noodles

(see Table. I). The name attribute is more complex, since people use different names to refer to the same object in different scenarios. For instance, some people say "Bag of chips" and others say "Sun chips" for the object instance of "Sun chips". We treat object names in the same manner as appearance attributes. However, we associate *all* object names used by AMT workers with an object instance. As a result, the name attribute has 90 different values (words or phrases) and, for instance, the "Sun chips" object has several possible names like "Sun chips", "Bag of chips", "Bag" and so on. Table I shows some names used for objects in the food bag and instant noodles categories. This data is mainly used for the experiments presented in Sections V-C and V-E.

IV. ATTRIBUTE BASED OBJECT IDENTIFICATION

In our object identification framework, the inputs are values for K attribute types, $A = \{a^1, \ldots, a^K\}$, and a set containing J segmented objects $\{I_1, \ldots, I_J\}$. The goal is to find the specific object j^* referred to by the attributes. We identify j^* by maximizing the likelihood of the attribute values A given object I_{j^*}:

$$j^* = \underset{1 \le j \le J}{\operatorname{argmax}} \, p(A|I_j) = \prod_{k=1}^{K} p(a^k|I_j) \tag{1}$$

where $p(a^k|I_j)$ is the likelihood function. Here, we have factorized $p(A|I_j)$ by assuming that the attributes are independent given the object I_j.

We model the probability of values of each attribute type using multinomial logistic regression. In particular, the probability of object I (we omit the subscript when possible) having attribute value t is defined as

$$p(t|I, W) = \frac{\exp(f^t(I, W))}{\sum_{t'=1}^{T} \exp(f^{t'}(I, W))} \tag{2}$$

where W are the model parameters for the attribute type k learned from training data, and T is the number of attribute values belonging to the attribute type k (we used W and T instead of W^k and T^k for simplicity). The functions $f^t(I, W)$ are discriminative functions. It might be useful to think of $f^t(I, W)$ as a compatibility function that measures how compatible pairs of the attribute value t and the segmented object I are. This is also called a soft-max function in the neural networks literature. The corresponding log likelihood is of the form $\log p(t|I, W) = f^t(I, W) - \log \sum_{t'=1}^{T} \exp(f^{t'}(I, W))$. We will detail the discriminative functions in the next section.

The accuracy of attribute recognition strongly depends on rich features extracted from the color and depth values of object segments. In our object identification framework, we first learn general feature codebooks in an unsupervised way [4], [5], and then sparsify these codebooks to learn attribute dependent features via group lasso optimization [23]. We first describe the general codebook learning approach.

A. Unsupervised Feature Learning

Our attribute dependent feature learning is built on hierarchical sparse coding [4], [5]. The key idea of sparse coding is to learn a codebook, which is a set of vectors, or codes, such that the data can be represented by a sparse, linear combination of codebook entries. In our case, the data are patches of pixel values sampled from RGB-D images. Our codebook learning algorithm uses K-SVD [1] to learn codebooks $D = [d_1, \cdots, d_m, \cdots, d_M]$ and the associated sparse codes $X = [x_1, \cdots, x_n, \cdots, x_N]$ from a matrix Y of observed data by minimizing the reconstruction error

$$\begin{aligned} \min_{D,X} \quad & \|Y - DX\|_F^2 \\ s.t. \quad & \|d_m\|_2 = 1, \ \ \forall m \\ & \|x_n\|_0 \le Q, \ \ \forall n \end{aligned} \tag{3}$$

Here, the notation $\|A\|_F$ denotes the Frobenius norm for matrix A, the zero-norm $\|\cdot\|_0$ counts the non-zero entries in

the sparse codes x_n, and Q is the sparsity level controlling the number of non-zero entries.

With the learned codebooks, sparse codes can be computed for new images using orthogonal matching pursuit or the more efficient batch tree orthogonal matching pursuit [4]. Spatial pyramid max pooling is then applied to the resulting sparse codes to generate object level features. A spatial pyramid partitions an image into multiple levels of spatial cells, and the features of each spatial cell are computed via max pooling, which simply takes the component-wise maxima over all sparse codes within a cell (see [5] for details).

B. Attribute Dependent Features via Codeword Selection

As we will show in the experiments, the features learned via hierarchical sparse coding give excellent results for attribute classification and object identification, when learned on raw RGB and Depth image patches. However, a limitation of such rich features is that they might not generalize as well as task dependent features. For instance, imagine one wants to train a classifier for the color "red" by providing a small set of red example objects. In this case, overly general features might lead to shape specific codewords being used to learn a good classifier for the examples (overfitting). To avoid this, instead of learning only one, general codebook, we learn subsets of such a codebook containing only codewords useful for specific attribute types. Our approach sparsifies codebooks via group lasso [23]. We now describe this learning approach in the context of attribute classification.

We learn attribute classifiers using linear decision functions

$$f^t(I, W) = \sum_{s=1}^{S} \beta(I^s, D)^\top w^{s,t} + b^t \tag{4}$$

where $\beta(I^s, D)$ are pooled sparse code features over the spatial cell I^s, S is the number of spatial cells drawn from the object I, $w^{s,t}$ and b^t are weight vectors and a bias term, $W = [w^{11}, \cdots, w^{1T}, \cdots, w^{S1}, \cdots, w^{ST}]$, and T is the number of attribute values belonging to one attribute type. Here, $t \in \{1, \cdots, T\}$ denotes a specific attribute value for one attribute type. For instance, attribute values for the shape attribute are circular, cylindrical, ellipsoid and rectangular. Note that previous work has shown that linear classifiers are sufficient to obtain good performance for sparse code features [5].

We learn the model parameters from the training data collected from Amazon Mechanical Turk. For each attribute type, the training data consists of G pairs of the segmented objects I_g and their corresponding attribute values a_g. Here, a_g belongs to one of T attribute values (e.g. one of the 13 color words for color attribute). We want to find the model parameters W by maximizing the log likelihood of training data

$$\sum_{g=1}^{G} \log p(a_g | I_g, W) - \lambda ||W||_{21} \tag{5}$$

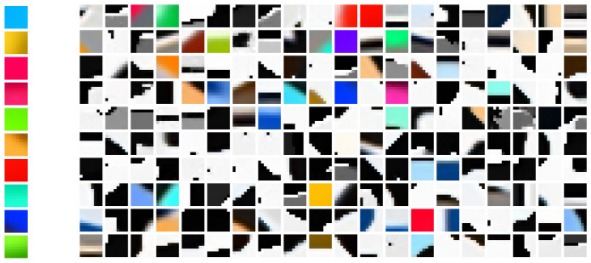


Fig. 3. Sparsified codebooks for color (*left*) and shape (*right*) attributes. Our approach learned that solid color codewords are most relevant to classify colors, while mostly depth codewords (grey scale) are selected for shape classification.

where $||W||_{21}$ is a regularization term, and its intensity is controlled by the parameter λ. Let w^i and w_j denote the i-th row and the j-column of the matrix W, respectively. The matrix norm $(2,1)$ is defined as $\| W \|_{21} = \sum_{m=1}^{M} \| w^m \|_2$. In our case, we have

$$||W||_{21} = \sum_{m=1}^{M} \| w_m^{11}, \cdots, w_m^{S1}, \cdots, w_m^{ST} \|_2 \tag{6}$$

where M is the size of the codebook. This regularization term is called group lasso; it accumulates the weights of the spatial cells and the attribute values for each codeword using 2-norm and then enforces 1-norm over them to drive the weight vectors of individual codewords toward zero. The group lasso thereby sparsifies the codebook and thus leads to an attribute dependent codebook that is usually much smaller than the full codebook. If the group lasso drives an entire weight vector w_m to 0, the corresponding codeword no longer affects the decision boundary and has effectively been removed by the optimization. The $(2,1)$-norm regularized log likelihood is a concave function of W, so the optimal parameter settings can be found without local minima. We use the accelerated gradient descent algorithm to solve the above optimization problem; it has been proven to have a fast convergence rate [6], [20].

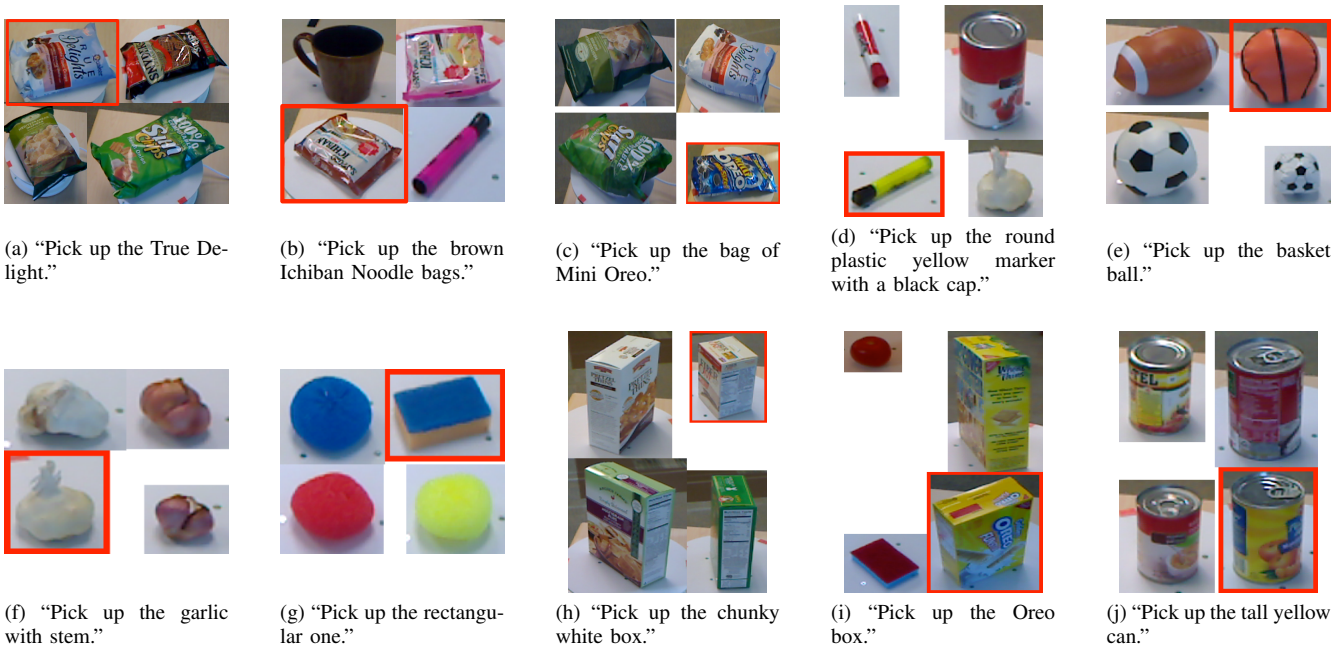
The intuition behind our codebook sparsification is that the full codebook learned by K-SVD consists of many types of codewords while only a small number of codewords is relevant to a given attribute type. To demonstrate this intuition, we visualize the codebooks selected by our algorithm for color and shape attributes in Fig. 3. As can be seen, the color attribute codebook only contains color codewords, while the shape codebook is dominated by depth codewords (black and white) along with some color gradient codewords.

V. Experiments

A. Dataset

All datasets used in our experiments are available at
`http://www.cs.washington.edu/rgbd-object-attributes`

1) Training dataset–Object Attribute Dataset: We use the Object Attribute Dataset described in Section III to train attribute classifier. Following the experimental setting in [5], we take video sequences captured from the 30° and 60° elevation angles as training set and ones captured from



(a) "Pick up the True Delight."

(b) "Pick up the brown Ichiban Noodle bags."

(c) "Pick up the bag of Mini Oreo."

(d) "Pick up the round plastic yellow marker with a black cap."

(e) "Pick up the basket ball."

(f) "Pick up the garlic with stem."

(g) "Pick up the rectangular one."

(h) "Pick up the chunky white box."

(i) "Pick up the Oreo box."

(j) "Pick up the tall yellow can."

Fig. 4. Scenes and sentences collected from Amazon Mechanical Turk. Our system correctly identifies all objects except the ones in (f), (h), and (j). The attributes "stem", "chunky", and "tall" are relative and parts-based attributes not contained in the training data.

TABLE II

STATISTICS FOR THE SCENE DATASET.

	Overall	SCENE-D	SCENE-S
Raw Data	2,400	2,000	400
$V1$	2,027	1,767	260
$V2$	1,767	1,577	190

the $45°$ angle as test set (leave-sequence-out on the RGB-D dataset [17]). For training efficiency, only 1/4 of all images/views are used for training, i.e. around 2,500 images for each classifier. All images used in the evaluation dataset, SCENE, and Section V-C are from sequences of $45°$ angle.

*2) Testing dataset–*SCENE*:* To evaluate attribute based object identification, we collect an additional dataset called SCENE. There are 2,400 scenes in this dataset. Each scene has 4 different objects in it. One of the objects is marked by a red bounding box, which indicates the target object people should refer to. We present each scene to Amazon Mechanical Turk and ask people to write down a sentence based on which a robot can identify the target object. Fig 4 gives some example scenes along with sentences collected via AMT.

SCENE has two parts, SCENE-D and SCENE-S. The first part consists of 2,000 images, each of which contains 4 views of objects randomly picked from all 110 objects. We call this set SCENE-D, since the objects are typically from **d**ifferent categories. Panels (b), (c) and (i) in Fig. 4 are examples of this part. The second part, SCENE-S, consists of 400 scenes containing 4 objects randomly picked from the **s**ame category. The remaining panels in Fig. 4 are taken from SCENE-S.

To automatically extract attributes and names from the AMT sentences, we rely on WordNet [11] and the Stanford Part of Speech (POS) tagger [25]. First, we use POS to tag every word in a referral sentence. All adjective words are treated as appearance attributes and noun phrases are treated as names of objects. However, the scene descriptions might contain attribute words that were not used in the training set. For each unknown adjective, we use WordNet to determine if it is a synonym or hyponym of a known attribute. Since not all new words can be mapped to known words, the test scenes might contain information that our classifiers cannot take advantage of.

To evaluate how well sentences can be automatically parsed by our approach, we analyzed all 2,400 sentences provided via Mechanical Turk and checked which of them were sufficient to identify the referred object. This gave us a subset of cleaned up data, called $V1$. As can be seen in Table II, 2,027 (or 84.5%) of the AMT sentences in the SCENE dataset are sufficient to uniquely identify the object in the scene. Broken down into SCENE-D and SCENE-S, we get 1,767 (88.4%) and 260 (65.0%) sufficient sentences, respectively. The lower percentage of correct sentences in SCENE-S is due to the fact that it contains more difficult scenes, and Mechanical Turkers tend to make more mistakes on these. We further checked among the sentences in $V1$, how many of them our system can parse successfully to known attributes (we call this set $V2$). The numbers in Table II show that our language pipeline can correctly parse 89.2% (1,577 out of 1,767) and 73.1% (190 out of 260) of the valid sentences in SCENE-D and SCENE-S, respectively. A careful analysis of the failure cases showed that for difficult scenes, people use other types of attributes, such as relative attributes (darker, smaller) and localized attributes (has white

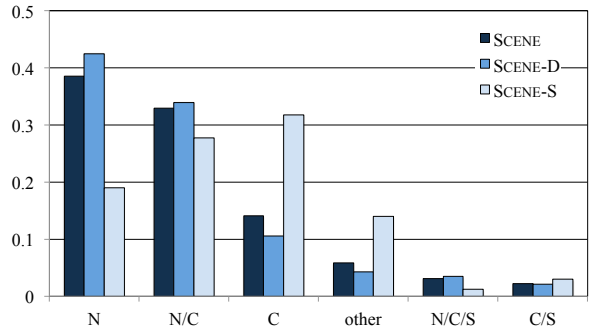


Fig. 5. Frequency of different subsets of attribute types used by people to identify objects. **N** is short for name, **C** is short for color, **S** is short for shape and **M** is short for material. **other** means other attribute types such as relative or parts-based. Only the most frequently used attributes or attribute combinations are shown.

cap), which cannot yet be handled by our system.

We then evaluated which attribute types AMT workers used for object identification in the SCENE dataset. Fig. 5 shows percentages for the most frequently used combinations of attribute types among all tasks. For the SCENE-D set, Name is the most frequently used attribute. The second most frequently is Name with Color (N/C), followed by Color only. This is not very surprising, since Name and Color are very distinctive attributes, especially when the objects belong to different categories (it is quite interesting to see from Fig. 6 that Name and Color are also the two most discriminative attributes for our system). For SENCE-S, Name is not used as often as it is for SCENE-D. People turn to use more Color and other attributes to identify specific objects.

B. Learning Setup

We learn general codebooks of size 1000 with sparsity level 5 on 1,000,000 sampled 8×8 raw patches for both RGB and depth images. We remove the zero frequency component from raw patches by subtracting their means. With these learned codebooks, we compute sparse codes of each pixel (8×8 patch around it) using batch orthogonal matching pursuit with sparsity level 5, and generate object level features by spatial pyramid max pooling over the whole images with 4×4, 2×2, and 1×1 partitions. The final feature vectors are the concatenation of the features over depth and color channels, resulting in a feature size of 42,000 dimensions. The hyperparameters of sparse coding and multinomial logistic regression are optimized on the RGB-D object attribute training set (no test data was used for hyperparameter optimization). Average classification accuracy for individual attribute types are 97.9%(name), 89.1%(color), 94.7%(shape) and 97.0%(material).

C. Attribute Recognition for Object Identification

Here, how useful attributes are for object identification. The test data for object identification consists of 1,000 scenes, each scene is generated by randomly picking 4-10 different segmented object instances from the 45° angle test sequences. We consider three experimental settings.

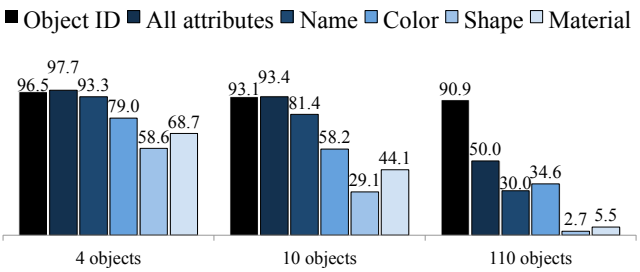


Fig. 6. Object identification results using four types of attributes and their combination.

TABLE III
OBJECT IDENTIFICATION ACCURACY (%) USING DIFFERENT SETS OF ATTRIBUTE TYPES.

	# of objects			
	4	6	8	10
All attributes	97.7	96.1	94.6	93.4
Minimal subset of attributes	91.0	88.8	87.3	86.1

In the first setting, we aim to identify the target object from a set of 4, 10, or all objects using the four attribute types provided by a Turker for the target object. I.e., we assume people mention name, color, shape and material of the target object. We report the results for the different attribute types and their combination in Fig. 6. First of all, we can see that all four types of attributes are helpful for object identification and that their combination outperforms each individual attribute by a large margin. For instance, for 10 objects, the combination of all four attributes achieves more than 10 percent higher accuracy than object Name and more than 20 percent higher accuracy than the appearance attributes Shape, Color, and Material. Not surprisingly, the accuracy of object identification decreases with an increasing number of objects in the set, but the combination of all four types of attributes drops much less than each individual attribute. Fig. 6 also contains results for object instance recognition, in which each object gets a unique Object ID (note that this is not a realistic setting for a natural language interface, since not all objects in an environment can be named uniquely). It is very satisfying to see that our attribute based identification slightly outperforms even this setting for 4 and 10 objects, indicating that learning multiple attribute classifiers provides higher performance than a single classifier, even when trained on artificially provided IDs.

Note that the rightmost results represent an extreme case, assuming all 110 objects are placed in the same scene. It can be seen that using language attributes, we can achieve accuracy of 50%, which is significantly worse than using object ID. This shows the potential benefit of introducing additional attribute types such as relative (darker, smaller) and spatial attributes (the box on the left) for large scale scenes.

The Gricean Maxims [8] suggest that people avoid redundancy in referral expressions and use only a subset of attributes for reference. In this second setting, we aim at identifying the specific object from a set using a *minimal*

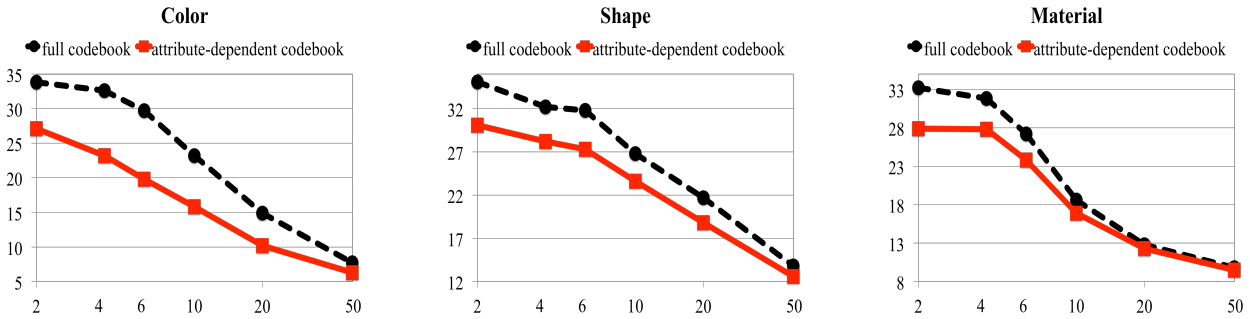


Fig. 7. Error rates for learning new attribute values from different numbers of training examples (# of training samples are in log scale).

TABLE IV
OBJECT IDENTIFICATION RESULTS (ACCURACY %) FOR SCENE, SCENE-D AND SCENE-S.

	SCENE	SCENE-D	SCENE-S
Raw Data	83.7	85.6	64.3
$V1$	88.7	90.2	67.6
$V2$	93.2	94.4	77.3

subset of attributes, that is, only attributes required to distinguish the target object are provided. To get minimal subsets, we try all combinations of one, two, three, and four attribute types in turn and stop when the attribute types used are able to distinguish the target object from the others (using ground truth attribute labels). We found that for scenes containing 10 objects, 87.6% of them can be identified based on one attribute only, 12.6% of them can be identified based on two attributes, and only 0.01% of them have to use three attributes. We report the results in Table III. As can been seen, our approach obtains very high accuracy even in this setting, suggesting the robustness of attribute based object identification. Note that it is not surprising that the minimal subset of attributes works slightly worse than the combination of all four types of attributes since redundant attributes increase the robustness of object identification.

D. Scene Based Object Identification

We now describe our evaluation on the SCENE dataset, which is described in Section V-A.2. We report results on the raw data and on the two validated subsets ($V1$ and $V2$) in Table IV. As can be seen, our object identification framework achieves high accuracy even for the raw data. The accuracy on the verified data is even higher: 90.2% and 94.4% on the first type of test (objects are of different type). The accuracy decreases for SCENE-S, because this task is much harder than the first one. However, even in this most difficult setting, we still achieve 67.6% and 77.3% accuracy on the validated data and 64.3% on the raw AMT sentences (chance would be 25%). The overall performance on both types of tasks is 88.7% and 93.2% on the validated data and 83.7% on the raw data.

E. Sparsified Codebooks for Transfer Learning

To investigate if our attribute dependent codebooks are more suitable for learning new attribute values when compared to the full codebooks, we perform the following experiments. For each type of attribute, we leave two attribute values out as new (or target) attribute values and learn a sparsified codebook using the remaining attribute values. We then train models for the left-out attribute values and test them on the test set for these attribute values. For instance, in one setup we leave the colors blue and yellow out, learn a sparsified color-dependent codebook using the other colors, and then use that codebook to learn classifiers for blue and yellow (see also Fig. 3 for examples of sparsified codebooks for color and shape). This result is compared to learning blue and yellow classifiers using the full codebook.

We report the results obtained by the sparsified codebook and the full codebook in Fig. 7. As can be seen, the sparsified codebooks significantly outperform the full codebooks on small numbers of training samples for color, shape and material attributes. This is because the sparsified codebooks effectively remove the codewords unrelated with the particular attribute type that could confuse the learning procedure when faced with new attribute values, especially for limited training samples. The sparsified codebooks result in much less chance to overfit to the training sets than the full codebooks and thus achieve better accuracy in this scenario. We also tried the sparsified and full codebooks for the object name attribute and found that these two approaches have comparable accuracy on the test set. This is expected since the object name attribute is a highly mixed concept and virtually all codewords are relevant for it.

VI. DISCUSSION

We presented an attribute based approach for object identification. We consider four types of attributes: shape, color, material, and name. We classify each type of attribute via multinomial logistic regression. Our model learns perceptual features used for attribute classification from raw color and depth data. It incorporates sparse coding techniques for codebook learning, and then uses group lasso regularization to select the most relevant codewords for each type of attribute.

To investigate the object identification task on realistic objects, we generate a large attribute data set by collecting descriptions for objects in an existing RGB-D object dataset [17]. Our experiments demonstrate that (1) each type of attribute is helpful for object identification and their combination works much better than each single attribute

including the object name attribute only; and (2) attribute dependent sparsified codebooks significantly improve the accuracy over non-specific codebooks for learning new attribute values when only limited training samples are available. We believe that the capability to learn from smaller sets of examples will be particularly important in the context of teaching robots about objects and attributes.

Attribute based object identification is a very promising area of research in robotics, specifically due to the growing importance of object manipulation and natural human robot interfaces. This work has several limitations that deserve further research. Although treating different attribute types independently achieves excellent performance, considering them jointly might further improve performance. More complex combinations of attributes and gestures are also an important direction for future work. Attributes describing an object's spatial and physical relationship to other objects in a scene are also important. Those localized attributes are useful especially for fine-grained object identification. Finally, we only use a flat model for object names, and a more complex, hierarchical model could further improve results.

Acknowledgments

This work was funded in part by the Intel Science and Technology Center for Pervasive Computing and by ARO grant W911NF-12-1-0197

REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] M. Blum, J. Springenberg, J. Wlfing, and M. Riedmiller. A Learned Feature Descriptor for Object Recognition in RGB-D Data. In *IEEE International Conference on Robotics and Automation*, 2012.

[3] L. Bo, X. Ren, and D. Fox. Depth Kernel Descriptors for Object Recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[4] L. Bo, X. Ren, and D. Fox. Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms. In *Advances in Neural Information Processing Systems*, 2011.

[5] L. Bo, X. Ren, and D. Fox. Unsupervised Feature Learning for RGB-D Based Object Recognition. In *International Symposium on Experimental Robotics*, 2012.

[6] X. Chen, W. Pan, J. Kwok, and J. Carbonell. Accelerated Gradient Method for Multi-task Sparse Learning Problem. In *IEEE International Conference on Data Mining*, 2009.

[7] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

[8] A. Dale and E. Reiter. Computational Interpretations of Gricean Maxims in the Generation of Referring Expressions. *Congnitive Science*, 18:233–263, 1995.

[9] K. Duan, D. Parikh, D. Crandall, and K. Grauman. Discovering Localized Attributes for Fine-grained Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[10] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing Objects by their Attributes. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2009.

[11] C. Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[12] V. Ferrari and A. Zisserman. Learning Visual Attributes. In *Advances in Neural Information Processing Systems*, 2007.

[13] G. Hinton, S. Osindero, and Y. Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554, 2006.

[14] S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal Templates for Real-Time Detection of Texture-less Objects in Heavily Cluttered Scenes. In *IEEE International Conference on Computer Vision*, 2011.

[15] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar. Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision*, 2009.

[16] T. Kwiatkowski, L. Zettlemoyer, S. Goldwater, and M. Steedman. Inducing Probabilistic CCG Grammars from Logical Form with Higher-Order Unification. In *Empirical Methods in Natural Language Processing*, 2010.

[17] K. Lai, L. Bo, X. Ren, and D. Fox. A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In *IEEE International Conference on Robotics and Automation*, 2011.

[18] K. Lai, L. Bo, X. Ren, and D. Fox. A Scalable Tree-based Approach for Joint Object and Pose Recognition. In *the AAAI Conference on Artificial Intelligence*, 2011.

[19] C. Lampert, H. Nickisch, and S. Harmeling. Learning to Detect Unseen Object Classes by Between-Class Attribute Transfer. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2009.

[20] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.

[21] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[22] C. Matuszek, N. FitzGerald, L. Zettlemoyer, L. Bo, and D. Fox. A Joint Model of Language and Perception for Grounded Attribute Learning. In *International Conference on Machine Learning*, 2012.

[23] G. Obozinski, B. Taskar, and M. Jordan. Joint Covariate Selection and Joint Subspace Selection for Multiple Classification Problems. *Statistics and Computing*, 20(2):231–252, 2010.

[24] D. Parikh and K. Grauman. Relative Attributes. In *IEEE International Conference on Computer Vision*, 2011.

[25] K. Toutanova and C. D. Manning. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 2000.

[26] K. Yu, Y. Lin, and J. Lafferty. Learning Image Representations from the Pixel Level via Hierarchical Sparse Coding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.