Attribute Based Object Identification

Yuyin Sun[1] **Liefeng Bo**[2] **Dieter Fox**[1]
[1]Department of Computer Science & Engineering
University of Washington, Seattle, WA 98195
[2]Intel Science and Technology Center on Pervasive Computing
Intel Labs, Seattle, WA 98195
{sunyuyin, fox}@cs.washington.edu liefeng.bo@intel.com

Abstract

Over the past 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 to identify specific objects in a scene. In this paper, we introduce an approach for identifying objects based on 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-specific color and depth features. We use Mechanical Turk to collect a large data set of attribute descriptions of objects in the RGB-D object dataset. Our experiments show that learned attribute classifiers outperform previous instance-based techniques for object identification. We also demonstrate that attribute-specific features provide significantly better generalization to previously unseen attribute values, thereby enabling more rapid learning of new attribute values.

1 Introduction

Identifying objects in complex scenes is a crucial capability for an autonomous robot to understand and interact with 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 [9, 14]. The development of features and algorithms for combining color and depth information provided by RGB-D cameras further increases the accuracy of object detection [9]. So far, virtually all work on object recognition assume that each object has a unique instance ID by which it is referenced. However, this is not how people identify objects using natural language. Since not every object in an environment has a unique language identifier, people often use additional *attributes* such as color (blue), shape (round), size (large), or material (metal) to refer 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 the 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 [7]. Their work showed that it is possible to recognize new object types solely by their appearance attributes [15], or that attributes improve recognition of fine-grained object classes such as bird species [6]. However, these computer vision approaches do not consider the task settings typically faced in robotics, where a combination of appearance attributes and object names might be used to identify specific objects in a scene, rather than describing general object categories. Recent

"Blue", "Coffee mug" "Oreos"

Figure 1: Object identification. Imagine objects being on a table or kitchen counter and the command is "Bring me my coffee mug; it's the blue one", or "Pick up the Oreos on the kitchen counter" (red rectangles indicate the objects being referred to).

work in semantic natural language processing introduced a joint model for language and visual attributes for the purpose of object identification [18]. Since the focus of that work was on language learning, however, only very simple objects such as uniformly colored plastic toys were considered.

In this paper, we introduce an approach for identifying objects based on appearance and name attributes. Specifically, we consider the following setting: A robot perceives a set of objects, is given attributes describing 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 name attribute contains all words people might use to name objects. For instance, a name could be an object type, such as "bag", or a specific name, such as "Mini Oreo". To learn rich RGB-D features for attribute classification, we build on recently introduced sparse coding techniques [4]. 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 attributes. These attribute-dependent codewords represent knowledge shared by the same type of attributes, so they generalize much better than full codebook and achieve higher performance with limited samples.

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 [13]. Attribute values are gathered via Amazon Mechanical Turk, which provide a rich selection of attribute words people use to describe objects. We also presented Turkers with sets of objects and asked them to specify attributes that uniquely identify a target object from a group of objects, simulating a task in which a person might command a robot to pick up an object from multiple objects placed on a table. Experiments demonstrate that our learned appearance and name attributes are extremely well-suited to identify objects, even outperforming instance-based classifiers which do not lend themselves to a natural language interface. Furthermore, we show that our attribute-dependent feature learning significantly outperforms task independent features.

2 Related Work

This research focuses on object identification using attribute dependent feature learning. To the best of our knowledge, the paper presents the first study on combining attribute (including object name) learning with feature learning for object identification.

Feature learning: Over the past few years, there has been increasing interest in deep learning and unsupervised feature learning for object recognition. Deep belief nets [10] 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. To make deep belief nets applicable to full-size images, convolutional deep belief nets[16] use a small receptive field and share the weights between the hidden and visible layers among all locations in an image. Alternatively, hierarchical sparse coding [21] and hierarchical matching pursuit [3] 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, 4].

Attribute Learning: 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 [8] learn to localize color and texture attributes from annotations captured by

Table 1: Words used for color, shape, material, and example name attributes in the RGB-D Object Attribute Dataset.

Attrubutes	Words
Color	black, blue, yellow, red, transparent, white, purple, brown, green, pink, orange
Shape	rectangular, cylindrical, circular, ellipsoid
Material	paper, ceramic, fiber, metal, plastic, food, foam
Name (Bag)	bag, bag of chips, True Delights snack, Bear Crackers, bag of pretzels, mini Oreo, Tortilla chips, barbecue potato chips, Archer Farms potato chips, bag of Sun chips
Name (Noodle)	noodle, bag, Instant Noodle, Ichiban Instant Noodle, Ramen Noodle, Beef Noodle, Thai Noodle

image search. Farhadi et al. [7] describe objects by their attributes and showed that attribute based approaches generalize well across object categories. Kumar et al. [11] 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. [15] show that attributes are useful for detecting unseen object categories. Parikh et al [20] propose to model relative attributes and showed their advantages over traditional binary attributes. Duan et al. [6] show that attributes improve recognition of fine-grained object classes such as bird species. Matuszek [18] 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 object from 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.

3 Object Attributes

We developed a new RGB-D Object Attribute Dataset for training attribute classifiers. 110 objects in 12 categories (*Ball*, *Coffee Mug*, *Food Bag*, *Food Box*, *Food Can*, *Instant Noodle*, *Marker*, *Sponge*, *Stapler* and *Water Bottle*, *Garlic* and *Tomato*) from the RGB-D Object Dataset [13] are selected. Attributes of objects are collected using Amazon Mechanical Turk (AMT). In particular, we ask turkers on AMT to describe color, shape, material, and name attributes of single object using simple but precise sentences. We manually extract the attribute words from the AMT annotations. In future work, we intend to use state-of-the-art semantic parsers [12, 18] to make it more automatic. Annotations for color, shape and material attributes are rather consistent since different turkers tend to use the same words for the same objects. For these attribute types, we use the dominant words used for describing certain object. Overall, eleven color words, four shape words, and seven material words, listed in Table 1, are used. The name attribute is more complex, since people use names of different concept levels 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". Fortunately, people only choose one object name from the name hierarchy to identify the objects in a given scenario. This allows us to treat the object names in the same manner as appearance attributes, regardless of the name hierarchy. We thus associate *all* object names used by AMT workers with the corresponding object instance. As a result, the name attribute has 46 different values (words) and, for instance, the "Sun chips" object has three possible names ("Sun chips", "Bag of chips", "Bag"). Table 1 shows words used for objects name in the bag and noodle categories as an example. This data is mainly used for the experiments presented in Sections 5.2 and 5.4.

To collect test data for object identification using attributes, we present turkers with scenes of four objects and ask them to identify the object within the red box using object attributes. We then parse the sentences provided by workers into attributes. This test data consists of 1200 scenes of four objects along with the attributes picked by workers on AMT. Note that in this setting, we did not require that AMT workers provide values for all attributes, but only those they would choose naturally to identify certain object.

4 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 a given set of attributes. We identify j^* by maximizing the likelihood

of the attribute values A given object I_{j^*}:

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

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 I is the segmented object, W is 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. Since we model each attribute type independently, we have 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 following sections.

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 [3, 4], then sparsify these codebooks to learn attribute dependent features via group lasso optimization [19]. We first describe the general codebook learning approach.

4.1 Unsupervised Feature Learning

Our attribute dependent feature learning is built upon a state-of-the-art unsupervised feature learning approach, hierarchical sparse coding [3, 4]. 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

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

Here, the notation $\|\cdot\|_F$ denotes the Frobenius norm, 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 the 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 [3]. 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 [4] for details).

4.2 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 more specific features. For instance, imagine one wants to learn a classifier for the color "red" by providing a small set of red objects. In this case, overly general features might lead to shape specific codewords being used to learn a good classifier from training objects (overfitting). Thus,

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 [19]. We describe this learning approach in the context of our attribute classification.

We learn attribute classifiers using linear decision functions

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

where $\beta(I^p, D)$ are pooled sparse code features over the spatial cell I^p, P is the number of spatial cells drawn from the object I, $w^{p,t}$ and b^t are the weight vectors and the bias term, $W = [w^{11}, \cdots, w^{S1}, \cdots, w^{ST}]$ 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 with sparse code features [4].

We learn the model parameters from the attribute 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

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

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

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

This regularization term is called group lasso, which 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 has effect on the decision boundary and has been removed by the optimization effectively. The log likelihood with $(2, 1)$-norm regularization is a concave function of W, so the optimal parameter settings can be found and there is no local minima. We use the accelerated gradient descent algorithm to solve the above optimization problem, which has been proven to have a fast convergence rate [5, 17].

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 color and shape attributes in Figure 2. It can be seen that the color-specific codebook only contains color codewords, while the shape codebook is dominated by depth codewords (black and white) along with some color gradient codewords.

5 Experiments

We evaluate the proposed attribute based object identification on the RGB-D Object Attribute Dataset we collected. We train our attribute classifiers using individually captured objects and evaluate object identification on combinations of the segmented objects. We show different aspects of attribute based object identification and the performance gain achieved by attribute-specific feature learning.

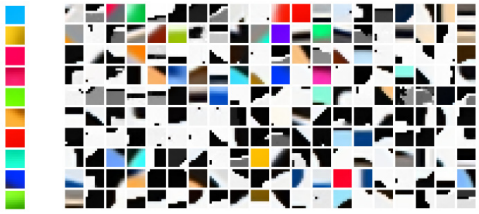
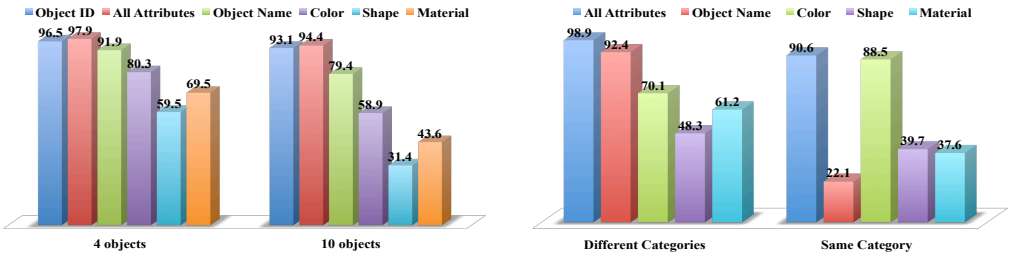


Figure 2: Sparsified codebooks for color (*left*) and shape (*right*) attributes, respectively. Our approach learned that solid color codewords are most relevant to classify colors, while mostly depth codewords (grey scale) are selected for shape classification.



(a) Results for data used in Section 5.2

(b) Results for AMT data

Figure 3: Object identification results (Accuracy) using four types of attributes and their combination.

5.1 Experimental 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. Following the experimental setting in [4], we take video sequences captured from the 30° and 60° elevation angles as training set and ones captured from the 45° angle as test set (leave-sequence-out on the RGB-D dataset [13]). The hyperparameters of sparse coding and multinomial logistic regression are optimized on the RGB-D object attribute training set.

5.2 Attributes for Object Identification

The test data for object identification consists of 1000 scenes, and each scene is generated by randomly picking 4-10 different segmented object instances from the 45° angle test sequences. We consider two experimental settings. In the first setting, we identify the target object from a set of 4 or 10 objects, and use the attribute values provided by Turkers for the target object. We report the identification accuracy results for different attributes and their combination in Figure 3 (a). 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, 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 3 (a) 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, indicating that learning multiple attribute classifiers provides higher performance than a single classifier, even when trained on artificially provided IDs.

In the second setting, we aim at identifying the specific object from a set using a *minimal subset of attributes*, that is, the fewest attributes required to distinguish the target object from the others. To get minimal subsets, we try all combinations of one, two, three, and four attribute types in turn and

Table 2: Object identification results (Accuracy) using all attributes and only a minimal subset of attributes required to identify an object from the other objects in the set.

	# of objects			
	4	6	8	10
All attributes	97.88	96.32	95.80	94.36
Minimal subset of attributes	91.24	89.25	87.57	86.62

Table 3: Object identification results (Accuracy) when the four objects are from different categories or all from the same category. Results are given for raw Mechanical Turk annotations and for manually validated data.

	Overall	Different categories	Same category
Raw Data	81.7	87.2	69.9
Validated Data	90.2	92.1	85.7

stop when the attribute types used are able to distinguish the target object from the others. 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, 0.01% of them have to use three attributes. We report the results in Table 2. As can been seen, our approach obtains very high accuracy even in this setting, suggesting the robustness of attribute based object identification. 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.

5.3 Object Identification Setting

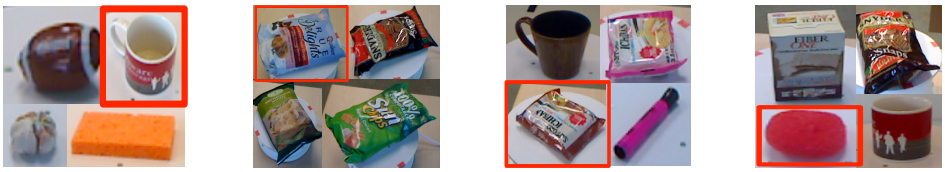
To test if our system can successfully identify the target object based on the subset of attributes provided by people in the context of a set of objects, we present two types of scenes to Amazon Mechanical Turk and ask people to identify the target object based on object properties. The first type of scenes consists of 800 images and each of them contains four objects randomly picked from all objects (here called*different categories*). The second type of scenes consists of 400 images containing four objects randomly picked from the *same category*. The second setting is more difficult than the first one since one typically can not identify the target object using the object name only.

Figure 4 shows example object sets and the attributes collected from AMT. As can be seen, shape, color, material, and object name attributes are frequently used to refer to a specific object. Since not all workers on AMT follow the instructions and some of them actually make mistakes, we manually checked the collected data to see if the collected attributes are sufficient to identify the target objects from the scenes. We found about 15% of the annotations do not refer to a unique target object. We report results on both the raw data and the validated subset (15% insufficient annotations removed) in Table 3. As can bee seen, our object identification framework achieves high accuracy even for the raw data. The accuracy on the verified data is even higher as expected: 92.1% on the first type of test (objects are of different type), 85.7% on the second type of test (all objects are of same type), and 90.2% on the joint set.

To shed light on the value of different attribute types in this setting, we perform these experiments using individual types of attributes and the combination of all attributes. Results are given in Figure 3 (b). Again, the accuracy on the first type of scenes is higher than the second ones, and the combination of attributes works much better than individual attribute. Note the low accuracy achieved by Object Name in the same category setting, caused by the fact that object names are typically not sufficient when several objects of the same type are present. It is also worth emphasizing that the accuracy on attributes collected from AMT is comparable with those obtained by all four types of attributes, suggesting that attribute based object identification is very robust and realistic for real data.

5.4 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 test attributes and learn a sparsified codebook using the remaining attribute values. We then train models for the left-out attribute values and test them



(a) "Pick up the coffee mug." since category name is sufficient for identification.

(b) "Pick up the True Delight." in case instance name is necessary and known.

(c) "Pick up the brown instant noodle." Both color and name attributes are used.

(d) "Pick up the red circular thing." Attributes are used since it's hard to identify the name (sponge) of the object.

Figure 4: Scenes and attributes collected from Amazon Mechanical Turk.

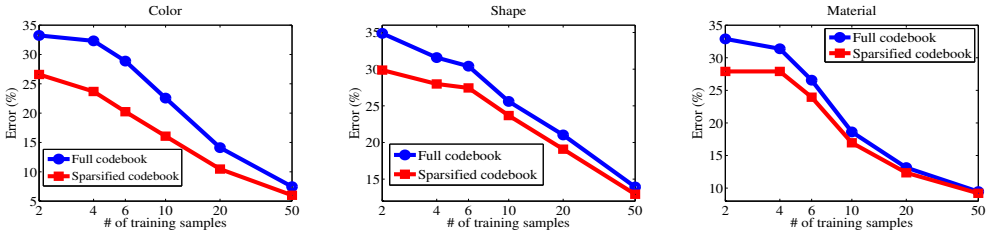


Figure 5: Learning new attribute values using sparsified codebooks. # of training samples are given on log scale. Y-axis is the average error rate over all pairs of new tasks.

on the test set for these attribute values. For instance, in one setup we leave the colors blue and yellow out, learn a sparsified codebook using the other colors, and then use that codebook to learn classifiers for blue and yellow (see also Figure 4 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 Figure 5. As can be seen, the sparsified codebooks significantly outperform the full codebooks especially 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.

6 Discussion

We presented an attribute based approach for object identification which considers four types of attributes: shape, color, material, and name. 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. Our experiments demonstrate that (1) each type of attribute is helpful for object identification and their combination works much better than 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. We manually extract attributes from text provided by AMT workers. We believe that automatic recognition and parsing of input speech is feasible in this context and will greatly increase the usability of our approach. Finally, we only use a flat model for object names, and a more complex, hierarchical model such as the one mentioned in Section 3 should further improve results.

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. Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms. In *Advances in Neural Information Processing Systems*, 2011.

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

[5] 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.

[6] 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.

[7] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

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

[9] S. Hinterstoisser, 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.

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

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

[12] 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.

[13] 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.

[14] 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.

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

[16] H. Lee, R. Grosse, R. Ranganath, and A. Ng. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. In *International Conference on Machine Learning*, 2009.

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

[18] 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.

[19] 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.

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

[21] 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.