OBJECT-CONTRASTIVE NETWORKS: UNSUPERVISED OBJECT REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Discovering objects and their attributes is of great importance for autonomous agents to effectively operate in human environments. This task is particularly challenging due to the ubiquitousness of objects and all their nuances in perceptual and semantic detail. In this paper we present an unsupervised approach for learning disentangled representations of objects entirely from unlabeled monocular videos. These continuous representations are not biased by or limited to a discrete set of labels determined by human labelers. The proposed representation is trained with a metric learning loss, where nearest neighbors in embedding space are pulled together while being pushed against other objects. We show these unsupervised embeddings allow robots to discover object attributes that generalize to previously unseen environments. We quantitatively evaluate performance on a large-scale synthetic dataset with 12k object models, as well as on a real dataset collected by a robot and show that our unsupervised object understanding generalizes to previously unseen objects. Specifically, we demonstrate the effectiveness of our approach on robotic manipulation tasks, such as pointing at and grasping of objects. An interesting and perhaps surprising finding in this approach is that given a limited set of objects, object correspondences will naturally emerge when using metric learning without requiring explicit positive pairs. Videos of robotic experiments are available at sites.google.com/view/object-contrastive-networks

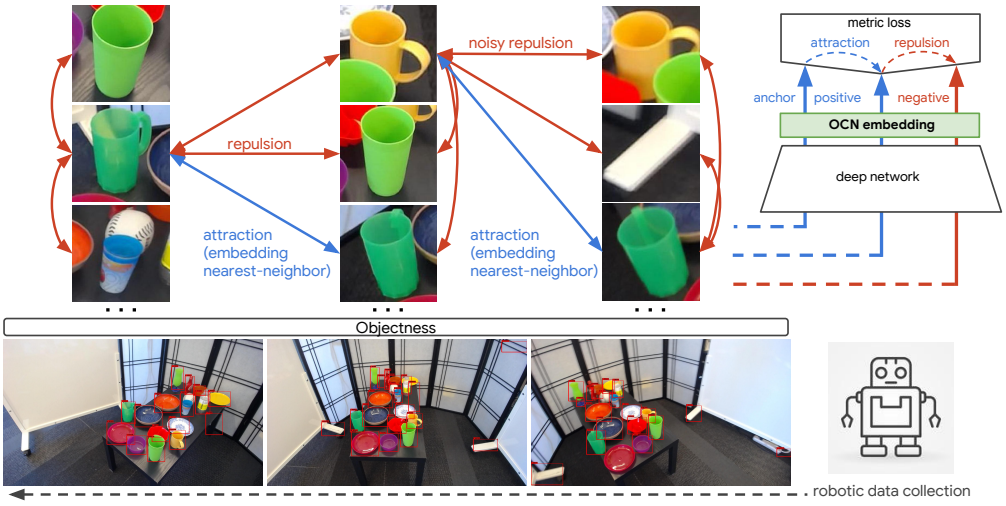

Figure 1: **Object-Contrastive Networks (OCN)**: by attracting embedding nearest neighbors and repulsing others using metric learning, continuous object representations naturally emerge. In a video collected by a robot looking at a table from different viewpoints, objects are extracted from random pairs of frames. Given two lists of objects, each object is attracted to its closest neighbor while being pushed against all other objects. Noisy repulsion may occur when the same object across viewpoint is not matched against itself. However the learning still converges towards disentangled and semantically meaningful object representations which can be useful in autonomous robotics applications.

1 INTRODUCTION

The ability to autonomously train to recognize and differentiate previously unseen objects as well as infer general properties and attributes is an important skill for robotic agents. Increased autonomy leads to robustness, one of the main challenges real-world robotics faces. It also renders scaling up data collection practical. Additionally, removing human supervision from the loop has the potential to enable learning richer and less biased continuous representations than ones supervised by a limited set of discrete labels. Unbiased representations can prove useful in unknown future environments different from the ones seen during supervision, a typical challenge for robotics.

In this work we present an unsupervised method that learns representations that disentangle perceptual and semantic object attributes such as class, function, and color. We automatically acquire training data by capturing videos with a real robot; a robot base moves around a table to capture objects in various arrangements. Assuming a pre-existing objectness detector, we extract objects from random frames within a same scene containing the same objects, and let the metric learning system decide how to assign positive and negative pairs of embeddings. Representations that generalize across objects naturally emerge despite not being given groundtruth matches. Unlike previous methods, we abstain from employing additional self-supervisory training signals such as tracking or depth. The only inputs to the system are monocular videos. This simplifies data collection and allows our embedding to integrate into existing end-to-end learning pipelines. We demonstrate that a trained Object-Contrastive Network (OCN) embedding allows us to reliably identify object instances based on their visual features such as color and shape. Moreover, we show that objects are also organized along their semantic or functional properties. For example, a cup might not only be associated with other cups, but also with other containers like bowls or vases.

The key contributions of this work are: (1) an unsupervised algorithm for learning representations of objects (naturally encoding attributes like class, color, texture and function) which generalize to previously unseen objects; (2) showing monocular videos are sufficient to contrast similar and dissimilar objects pairs naturally without requiring explicit correspondences; (3) demonstrating the autonomy of the system, using a robot from data collection to tasks such as pointing and grasping similar objects to ones presented to it.

2 RELATED WORK

Object discovery from visual media. Identifying objects and their attributes has a long history in computer vision and robotics (Tuytelaars et al., 2009). Traditionally, approaches focus on identifying regions in unlabeled images to locate and identify objects (Sivic et al., 2005; Russell et al., 2006; Arora et al., 2007; Fritz & Schiele, 2008; Kim et al., 2008). Discovering objects based on the notion of 'objectness' instead of specific categories enables more principled strategies for object recognition (Uijlings et al., 2013; Romea et al., 2011). Several methods address the challenge to discover, track, and segment objects in videos based on supervised (Wang et al., 2014) or unsupervised (Kwak et al., 2015; Schulter et al., 2013; Haller & Leordeanu, 2017) techniques. The spatio-temporal signal present in videos can also help to reveal additional cues that allow to identify objects (Wang & Gupta, 2015; Jain et al., 2017). In the context of robotics, methods also focus on exploiting depth to discover objects and their properties (Mishra et al., 2012; Karpathy et al., 2013).

Many recent approaches exploit the effectiveness of convolutional deep neural networks to detect objects (Ren et al., 2015; Liu et al., 2016; Lin et al., 2017) and to even provide pixel-precise segmentations (He et al., 2017). While the detection efficiency of these methods is unparalleled, they rely on supervised training procedures and therefore require large amounts of labeled data. Self-supervised methods for the discovery of object attributes mostly focus on learning representations by identifying features in multi-view imagery (DeTone et al., 2017; Lin et al., 2015) and videos (Wang & Gupta, 2015), or by stabilizing the training signal through domain randomization (Doersch et al., 2015; Zhang et al., 2018).

Some methods not only operate on RGB images but also employ additional signals, such as depth (Florence et al., 2018; Pot et al., 2018) or egomotion (Agrawal et al., 2015) to self-supervise the learning process. It has been recognized, that contrasting observations from multiple views can provide a view-invariant training signal allowing to even differentiate subtle cues as relevant features that can be leveraged for instance categorization and imitation learning tasks (Sermanet et al., 2018).

Unsupervised representation learning. Unlike supervised learning techniques, unsupervised methods focus on learning representations directly from data to enable image retrieval (Paulin et al., 2015), transfer learning (Zhang et al., 2017a), image denoising (Vincent et al., 2008), and other tasks (Dumoulin et al., 2016; Kumar et al., 2015). Using data from multiple modalities, such as imagery of multiple views (Sermanet et al., 2018), sound (Owens et al., 2016; Aytar et al., 2016), or other sensory inputs (Dehzangi et al., 2017), along with the often inherent spatio-temporal coherence (Doersch et al., 2015; Radford et al., 2015), can facilitate the unsupervised learning of representations and embeddings. For example, Zagoruyko & Komodakis (2015) explore multiple architectures to compare image patches and Pathak et al. (2017b) exploit temporal coherence to learn object-centric features. Gao et al. (2016) rely of spatial proximity of detected objects to determine attraction in metric learning, OCN operates similarly but does not require spatial proximity for positive matches, it does however take advantage of the likely presence of a same object in any pair of frames within a video. Zhang et al. (2017b) also take a similar unsupervised metric learning approach for tracking specific faces, using tracking trajectories and heuristics for matching trajectories and obtain richer positive matches. While our approach is simpler in that it does not require tracking or 3D matching, it could be augmented with extra matching signals.

In robotics and other real-world scenarios where agents are often only able obtain sparse signals from their environment, self-learned embeddings can serve as an efficient representation to optimize learning objectives. Pathak et al. (2017a) introduce a curiosity-driven approach to obtain a reward signal from visual inputs; other methods use similar strategies to enable grasping (Pinto & Gupta, 2016) and manipulation tasks (Sermanet et al., 2018), or to be pose and background agnostic (Held et al., 2015). Mitash et al. (2017) jointly uses 3D synthetic and real data to learn a representation to detect objects and estimate their pose, even for cluttered configurations. Hickson et al. (2018) learn semantic classes of objects in videos by integrating clustering into a convolutional neural network.

3 Unsupervised Learning of Object Representations

We propose an unsupervised approach to the problem of object understanding for multiple reasons: (1) make data collection simple and scalable, (2) increase autonomy in robotics by continuously learning about new objects without assistance, (3) discover continuous representations that are richer and more subtle than the discrete set of attributes that humans might provide as supervision which may not match future new environments. All these objectives require a method that can learn about objects and differentiate them without supervision. To bootstrap our learning signal we leverage two assumptions: (1) we are provided with a general objectness model so that we can attend to individual objects in a scene, (2) during an observation sequence the same objects will be present in most frames (this can later be relaxed by using an approximate estimation of ego-motion). Given a video sequence around a scene containing multiple objects, we randomly select two frames I and $\hat{I}$ in the sequence and detect the objects present in each image. Let us assume N and M objects are detected in image I and $\hat{I}$, respectively. Each of the n-th and m-th cropped object images are embedded in a low dimensional space, organized by a metric learning objective. Unlike traditional methods which rely on human-provided similarity labels to drive metric learning, we use a self-supervised approach to mine synthetic synthetic similarity labels.

3.1 Objectness Detection

To detect objects, we use Faster-RCNN (Ren et al., 2015) trained on the COCO object detection dataset (Lin et al., 2014). Faster-RCNN detects objects in two stages: first generate class-agnostic bounding box proposals all objects present in an image (Fig. 1), second associate detected objects with class labels. We use OCN to discover object attributes, and only rely on the first *objectness* stage of Faster-R-CNN to detect object candidates. Examples of detected objects are illustrated in Fig. 1.

3.2 Metric Loss for Object Attribute Disentanglement

We denote a cropped object image by $x \in \mathcal{X}$ and compute its embedding via a convolutional neural network $f(x) : \mathcal{X} \rightarrow K$. Note that for simplicity we may omit x from $f(x)$ while f inherits all superscripts and subscripts. Let us consider two pairs of images I and $\hat{I}$ that are taken at random from the same contiguous observation sequence. Let us also assume there are n and m objects

detected in I and $\hat{I}$ respectively. We denote the n-th and m-th objects in the images I and $\hat{I}$ as x_n^I and $x_m^{\hat{I}}$, respectively. We compute the distance matrix $D_{n,m} = \sqrt{(f_n^I - f_m^{\hat{I}})^2}$, $n \in 1..N$, $m \in 1..M$. For every embedded *anchor* f_n^I, $n \in 1..N$, we select a *positive* embedding $f_m^{\hat{I}}$ with minimum distance as *positive*: $f_{n+}^{\hat{I}} = argmin(D_{n,m})$. Given a batch of (*anchor*, *positive*) pairs $\{(x_i, x_i^+)\}_{i=1}^N$, the n-pair loss is defined as follows (Sohn, 2016):

$$\mathcal{L}_{N-pair}\left(\{(x_i, x_i^+)\}_{i=1}^N; f\right) = \frac{1}{N}\sum_{i=1}^{N} log\left(1 + \sum_{i \neq j} exp(f_i^\top f_j^+ - f_i^\top f_i^+)\right)$$

The loss learns embeddings that identify ground truth anchor-positive pairs from all other anchor-negative pairs in the same batch. It is formulated as a sum of softmax multi-class cross-entropy losses over a batch, encouraging the inner product of each anchor-positive pair (f_i, f_i^+) to be larger than all anchor-negative pairs (f_i, $f_{j\neq i}^+$).

The final OCN training objective over an observation sequence is the sum of npairs losses over all pairs of individual frames:

$$\mathcal{L}_{OCN} = \mathcal{L}_{N-pair}\left(\{(x_n^I, x_{n+}^{\hat{I}})\}_{n=1}^N; f\right) + \mathcal{L}_{N-pair}\left(\{(x_m^{\hat{I}}, x_{m+}^I)\}_{m=1}^M; f\right)$$

3.3 Architecture

OCN takes a standard ResNet50 architecture until layer *global_pool* and initializes it with ImageNet pre-trained weights. We then add three additional ResNet convolutional layers and a fully connected layer to produce the final embedding. The network is trained with the n-pairs metric learning loss as discussed in Sec. 3.2. Our architecture is depicted in Fig. 1 and Fig. 2.

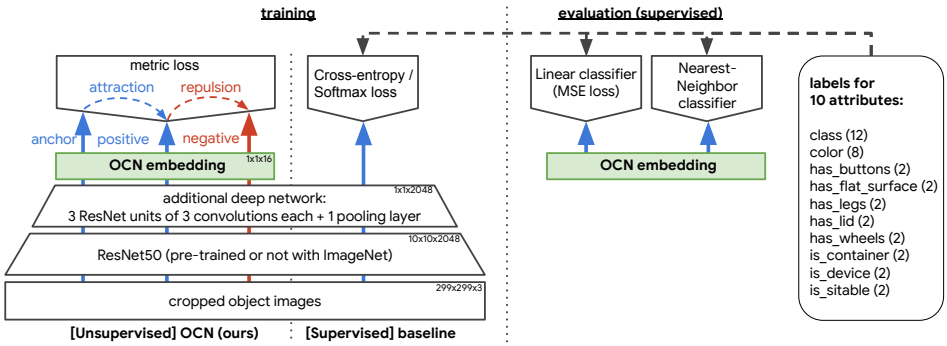

Figure 2: **Models and baselines:** for comparison purposes all models evaluated in Sec. 5 share the same architecture of a standard ResNet50 model followed by additional layers. While the architectures are shared, the weights are not across models. While the unsupervised model (left) does not require supervision labels, the 'softmax' baseline as well as the supervised evaluations (right) use attributes labels provided with each object. We evaluate the quality of the embeddings with two types of classifiers: linear and nearest neighbor.

3.4 Object-centric Embedding Space

By using multiple views of the same scene and by attending to individual objects, our architecture allows us to differentiate subtle variations of object attributes. Observing the same object across different views facilitates learning invariance to scene-specific properties, such as scale, occlusion, lighting, and background, as each frame exhibits variations of these factors. The network solves the metric loss by representing object-centric attributes, such as shape, function, texture, or color, as these are consistent for (anchor, positive)-pairs, and dissimilar for (anchor, negative)-pairs.

3.5 Why Should This Work?

One might expect that this approach may only work if it is given a good enough initialization so that matching the same object across multiple frames is more likely than random chance. While ImageNet pretraining certainly helps convergence as shown in Table 1, it is not a requirement to learn meaningful representations as shown in Sec. 8. When all weights are random and no labels are

provided, what can drive the network to consistently converge to meaningful embeddings? We estimate that the co-occurrence of the following hypotheses drives this convergence: (1) objects often remains visually similar to themselves across multiple viewpoints, (2) limiting the possible object matches within a scene increases the likelihood of a positive match, (3) the low-dimensionality of the embedding space forces the model to generalize by sharing abstract features across objects, (4) the smoothness of embeddings learned with metric learning facilitates convergence when supervision signals are weak, and (5) occasional true-positive matches (even by chance) yield more coherent gradients than false-positive matches which produce inconsistent gradients and dissipate as noise, leading over time to an acceleration of consistent gradients and stronger initial supervision signal.

4 DATA COLLECTION, HYPERPARAMETERS, AND TRAINING

To evaluate the effectiveness of OCN embeddings we generated two datasets of real and synthetic objects. For the (unlabeled) real data we arrange objects in table-top configurations and capture frames from continuous camera trajectories. The (labeled) synthetic data is generated from renderings of 3D objects in a similar configuration. Details about the datasets are reported in Table 4.

4.1 SYNTHETIC DATA GENERATION

To generate diverse object configurations we use 12 categories (airplane, car, chair, cup, bottle, bowl, guitars, keyboard, lamp, monitor, radio, vase) from ModelNet (Wu et al., 2015). The selected categories cover around 8k models of the 12k models available in the entire dataset. ModelNet provides the object models in a 80-20 split for training and testing. We further split the testing data into models for test and validation, resulting in a 80-10-10 split for training, validation, and test. For validation purposes, we manually assign each model labels describing the semantic and functional properties of the object, including the labels 'class', 'has lid', 'has wheels', 'has buttons', 'has flat surface', 'has legs', 'is container', 'is sittable', 'is device'. Fig. 9 shows an example scene.

We randomly define the number of objects (up to 20) in a scene and select half of the objects from two randomly selected categories. The other half is selected from the remaining object categories. We further randomly define the positions of the objects and vary their sizes, both so that they do not intersect. Additionally, each object is assigned one of eight predefined colors. We use this setup to generate 100K scenes for training, and 50K scenes for each, validation and testing. For each scene we generate a number (n = 10) of views and select random combination of two views for detecting objects. In total we produce 400K views (200 pairs) for training and 50K views (25K pairs) for each, validation and testing.

4.2 AUTOMATIC REAL DATA COLLECTION

Our real object data set consists of 187 unique object instances spread across six categories including 'balls', 'bottles & cans', 'bowls', 'cups & mugs', 'glasses', and 'plates'. Table 5 provides details about the number of objects in each category and how they are split between training, test, and validation. Note that we distinguish between cups & mugs and glasses categories based on whether it contains a handle. Fig. 3 provides a snapshot of our entire object dataset.

We automated the real world data collection through using a mobile robot equipped with an HD camera (Fig. 8). At each run, we place about 10 objects on the table and then trigger the capturing process by having the robot rotate around the table by 90 degrees (see Fig. 8). In average 130 images are captured at each run. We select random pairs of frames for each trajectory during training of the OCN. We performed 345, 109, and 122 runs of data collection for training, test, and validation dataset, respectively. In total 43084 images were captured for OCN training and 15061 and 16385 were used for test and validation, respectively.

4.3 TRAINING

An OCN is trained based on two views of the same synthetic or real scene. We randomly pick two frames of a camera trajectory around the scene to ensure the same objects are present; the frames are selected based on their time stamps so that they are as far apart as possible. We set the n-pairs regularization to λ = 0.002. The distance matrix $D_{n,m}$ (Sec. 3.2) is constructed based on the

Figure 3: We use 187 unique object instance in the real world experiments: 110 object for training (left), 43 objects for test (center), and 34 objects for evaluation (right).

individually detected objects for each of the two frames. The object detector was not specifically trained on any of our datasets. Furthermore, we only used scenes where at least 5 objects were detected in each frame. Operating on less objects results in a more noisy training signal as the n-pairs loss cannot create enough meaningful (anchor, negative)-pairs for contrasting them with the (anchor, positive)-pair. As the number of detected objects per view varies, we reciprocally use both frames to find anchors and their corresponding positives as discussed in Sec. 3.2. Across our experiments, the OCN training converged after 600k-1.2M iterations.

5 Experimental Results

To evaluate the effectiveness of an OCN embedding as representation for object attribute disentanglement, we performed experiments on a large-scale synthetic dataset and two robotic tasks of pointing and grasping in a real-world environment. Moreover, the experiments are designed in a way to directly showcase the usefulness of OCN on real robotics applications.

5.1 Attributes Classification

One way to evaluate the quality of unsupervised embeddings is to train attribute classifiers on top of the embedding using labeled data. Note however this may not entirely reflect the quality of an embedding because it is only measuring a discrete and small number of attributes while an embedding may capture more continuous and larger number of abstract concepts.

Classifiers: We consider two types of classifiers to be applied on top of existing embeddings in this experiment: linear and nearest-neighbor classifiers. The linear classifier consists of a single linear layer going from embedding space to the 1-hot encoding of the target label for each attribute. It is trained with a range of learning rates and the best model is retained for each attribute. The nearest-neighbor classifier consists of embedding an entire 'training' set, and for each embedding of the evaluation set, assigning to it the labels of the nearest sample from the training set. Nearest-neighbor classification is not a perfect approach because it does not necessarily measure generalization as linear classification does and results may vary significantly depending on how many nearest neighbors are available. It is also less subject to data imbalances. We still report this metric to get a sense of its performance because in an unsupervised inference context, the models might be used in a nearest-neighbor fashion (e.g. as in Sec. 5.3).

Baselines: we compare multiple baselines in Table 1 and Table 6. The 'Softmax' baseline refers to the model described in Fig. 2, i.e. the exact same architecture as for OCN except that the model is trained with a supervised cross-entropy/softmax loss. The 'ResNet50' baseline refers to using the unmodified outputs of the ResNet50 model (He et al., 2016) (2048-dimensional vectors) as embeddings and training a nearest-neighbor classifier as defined above. We consider 'Softmax' and 'ResNet50' baselines as the lower and upper error-bounds for standard approaches to a classification task. The 'OCN supervised' baseline refers to the exact same OCN training described in Fig. 2, except that the positive matches are provided rather than discovered automatically. 'OCN supervised' represents the metric learning upper bound for classification. Finally we indicate as a reference the error rates for random classification.

Results: we quantitatively evaluate our unsupervised models against supervised baselines on the labeled synthetic datasets (train and test) introduced in Sec. 4. Note that there is no overlap in object instances between the training and the evaluation set. The first take-away is that unsupervised performance closely follows its supervised baseline when trained with metric learning. As expected the cross-entropy/softmax approach performs best and establishes the error lower bound while the ResNet50 baseline are upper-bound results. Note that the dataset is heavily imbalanced for the

Table 1: **Attributes classification errors:** using attribute labels, we train either a linear or nearest-neighbor classifier on top of existing fixed embeddings. The supervised OCN is trained using labeled positive matches, while the unsupervised one decides on positive matches on its own. All models here are initialized and frozen with ImageNet-pretrained weights for the ResNet50 part of the architecture (see Fig. 2), while the additional layers above are random and trainable. Attributes are defined in Sec. 4.1.

Method	Class (12) Attribute Error	Color (8) Attribute Error	Binary Attributes Error	Embedding Size
[baseline] Softmax	2.98%	0.80%	7.18%	-
[baseline] OCN supervised (linear)	7.49%	3.01%	12.77%	32
[baseline] OCN supervised (NN)	9.59%	3.66%	12.75%	32
[ours] OCN unsupervised (linear)	10.70%	5.84%	13.76%	24
[ours] OCN unsupervised (NN)	12.35%	8.21%	13.75%	24
[baseline] ResNet50 embeddings (NN)	14.82%	64.01%	13.33%	2048
[baseline] Chance	91.68%	87.50%	50.00%	-

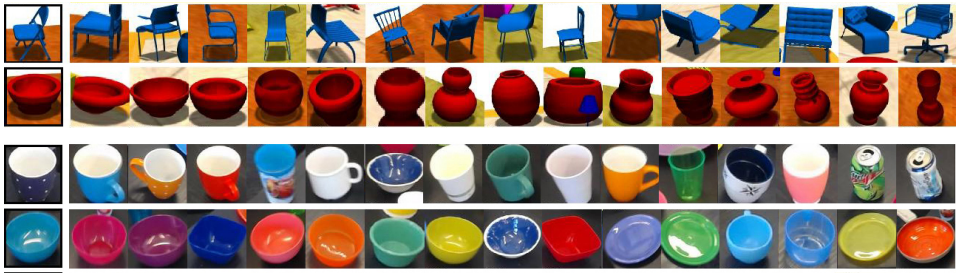

Figure 4: An OCN embedding organizes objects along their visual and semantic features. For example, a red bowl as query object is associated with other similarly colored objects and other containers. The leftmost object (black border) is the query object and its nearest neighbors are listed in descending order. The top row shows renderings of our synthetic dataset, while the bottom row shows real objects.

binary attributes reported in Table 1 and Table 6 and require balancing for linear classification. In Fig. 4 and Sec. 9, 11, we show qualitative results of nearest neighbor objects discovered by OCN.

5.2 Instance Detection and Tracking

An OCN embedding can be used to match instances of the same object across multiple views and over time. This is illustrated in Fig. 5, where objects of one view (anchors) are matched against the objects of another view. We can find the nearest neighbors (positives) in the scene through the OCN embedding space as well as the closest matching objects with descending similarity (negatives). We report the quality of finding corresponding objects in Table 2 and differentiate between *attribute errors*, that indicate a mismatch of specific attributes (e.g. a blue cup is associated with a red cup), and *object matching errors*, which measure when objects are not of the same instance. An OCN embedding significantly improves detecting object instances across multiple views.

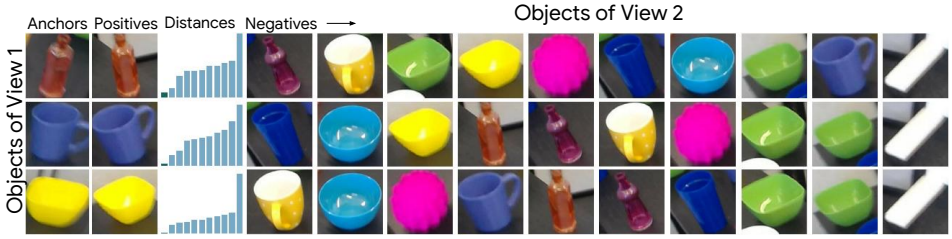

Figure 5: View-to-view object correspondences: the first column shows all objects detected in one frame (anchors). Each object is associated to the objects found in the other view, objects in the second column are the nearest neighbors. The third column shows the distances of all objects, all other objects are shown from left to right in descending order according to their distances to the anchor.

5.3 Robot Experiments

Pointing: We evaluate performance of OCN on a pointing robotic task (Fig. 6). The robot has to point to an object that it deems most similar to the object directly in front of him on the small table.

Table 2: Object correspondences errors: attribute error indicates a mismatch of a particular attribute of an object, while an object matching error is measured when the matched objects are not the same instance.

Method	Attribute Error	Object Matching Error
OCN supervised	4.53%	16.28%
OCN unsupervised	5.27%	18.15%
Resnet50 embeddings	19.27%	57.04%

The objects on the big table are randomly selected from each of the six object categories (Table 5). We consider two sets of these target objects. The quantitative experiment in Table 3 uses three query objects per category and is ran three times for each combination of query and target objects ($3 \times 2 \times 18 = 108$ experiments performed). The full set of experiments for one of the three runs is illustrated in Fig. 15.

Table 3 quantifies OCN performance of this experiment. We report on errors related to 'class' and 'container' attributes (note that the other ten attributes described in Sec. 4.1 are not relevant to the real object data set). While the trained OCN model is performing well on the most categories, it has particularly some difficulty on the object classes 'cups & mugs' and 'glasses'. These categories are generally mistaken with the category 'bowls'. As a result the network performs much better in the attribute 'container' since all the three categories 'bowls', 'bottles & cans', and 'glasses' refer to the same attribute.

Grasping: We qualitatively evaluate the OCN performance on a grasping task in an environment unseen during training. First, a person holds and shows an object to the robot, then the robot picks up the most similar object from a set of objects on a table (see Fig. 7). In this experiment, we focus on evaluating OCN with objects that have either similar shape or color attribute. Using OCN the robot can successfully identify and grasp the object that has the closest color and shape attributes to the query object. Note training data did not contain objects held by hand.

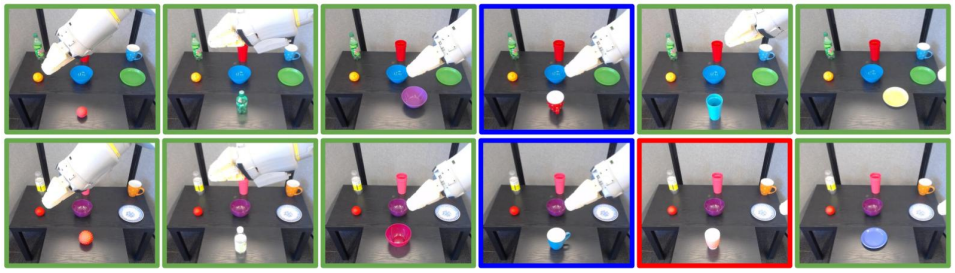

Figure 6: The robot experiment of pointing to the best match to a query object (placed in front of the robot on the small table). The closest match is selected from two sets of target object list, which are placed on the long table behind the query object. The first and the second row respectively correspond to the experiment for the first and second target object lists. Each column also illustrates the query objects for each object category. Image snapshots with green frame correspond to cases where both the 'class' and 'container' attributes are matched correctly. Image snapshots with blue frame refer to the cases where only 'container' attribute is matched correctly. Images with red frames indicates neither of attributes are matched.

Table 3: Quantitative evaluation on the robot pointing experiment. We report on two attribute errors: 'class' and 'container'. See Sec. 5.3 for more information about the experiment.

Attributes	Balls	Bottles & Cans	Bowls	Cups & Mugs	Glasses	Plates	Total
Class error	11.1 ±7.9%	0.0 ±0.0%	22.2 ±15.7%	88.9 ±7.9%	38.9 ±7.9%	5.6 ±7.9%	27.8 ±3.9%
Container error	11.1 ±7.9%	0 ±0.0%	16.7 ±13.6%	16.7 ±0.0%	16.7 ±13.6%	5.6 ±7.9%	11.1 ±2.3%

6 CONCLUSION

We introduced a novel unsupervised representation learning algorithm that allows us to differentiate object attributes, such as color, shape, and function. An OCN embedding is learned by contrasting the features of objects captured from two frames of single view camera trajectories of table-top indoor environments. We specifically attend to individual objects by detecting object bounding boxes and leverage a metric learning loss to disentangle subtle variations of object attributes. The resulting embedding space allows to organize objects along multiple dimensions and serves as representation for robotic learning. We show that an OCN embedding can be used on real robotic tasks such as

Figure 7: Robot experiment of grasping the object that is closest to the query object (held by hand). Images on the left are captured by the robot camera, and the images on the right are the video frames from a third person view camera. The leftmost object (black border) is the query object and its nearest neighbors are listed in descending order. The top row and the bottom row show the robot successfully identifies and grasps the object with similar color and shape attribute respectively.

grasping and pointing, where it is important to differentiate visual and semantic attributes of individual object instances. Finally, we show that an OCN can be trained efficiently from RGB videos that are automatically obtained from a real robotic agent.

REFERENCES

Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *ICCV*, 2015.

H. Arora, N. Loeff, D. A. Forsyth, and N. Ahuja. Unsupervised segmentation of objects using efficient learning. In *CVPR*, pp. 1–7, June 2007.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *NIPS*, 2016.

Omid Dehzangi, Mojtaba Taherisadr, and Raghvendar ChangalVala. Imu-based gait recognition using convolutional neural networks and multi-sensor fusion. *Sensors*, 17(12), 2017.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. *CoRR*, abs/1712.07629, 2017.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. *CoRR*, abs/1606.00704, 2016.

Peter R. Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. 2018.

M. Fritz and B. Schiele. Decomposition, discovery and detection of visual categories using topic models. In *CVPR*, pp. 1–8, 2008.

Ruohan Gao, Dinesh Jayaraman, and Kristen Grauman. Object-centric representation learning from unlabeled videos. *CoRR*, abs/1612.00500, 2016. URL `http://arxiv.org/abs/1612.00500`.

Emanuela Haller and Marius Leordeanu. Unsupervised object segmentation in video by efficient selection of highly probable positive features. In *ICCV*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pp. 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

David Held, Sebastian Thrun, and Silvio Savarese. Deep learning for single-view instance recognition. *CoRR*, abs/1507.08286, 2015.

Steven Hickson, Anelia Angelova, Irfan A. Essa, and Rahul Sukthankar. Object category learning and retrieval with weak supervision. *CoRR*, abs/1801.08985, 2018.

Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In *CVPR*, pp. 2117–2126, 2017.

Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2146–2153. IEEE, 2009.

Andrej Karpathy, Stephen Miller, and Li Fei-Fei. Object discovery in 3d scenes via shape analysis. In *ICRA*, 2013.

Gunhee Kim, C. Faloutsos, and M. Hebert. Unsupervised modeling of object categories using link analysis techniques. In *CVPR*, pp. 1–8, 2008.

B. G. Vijay Kumar, Gustavo Carneiro, and Ian D. Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. *CoRR*, abs/1512.09272, 2015.

S. Kwak, M. Cho, I. Laptev, J. Ponce, and C. Schmid. Unsupervised object discovery and tracking in video collections. In *ICCV*, 2015.

T. Lin, Yin Cui, S. Belongie, and J. Hays. Learning deep representations for ground-to-aerial geolocalization. In *CVPR*, pp. 5007–5015, 2015.

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 David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *ECCV*, pp. 740–755, 2014.

Tsung-Yi Lin, Piotr Dollr, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

A. K. Mishra, A. Shrivastava, and Y. Aloimonos. Segmenting simple objects using rgb-d. In *ICRA*, pp. 4406–4413, May 2012.

Chaitanya Mitash, Kostas E Bekris, and Abdeslam Boularias. A self-supervised learning system for object detection using physics simulation and multi-view pose estimation. In *IROS*, pp. 545–551. IEEE, 2017.

A Owens, P Isola, J H McDermott, A Torralba, E H Adelson, and W T Freeman. Visually indicated sounds. In *CVPR*, pp. 2405–2413, June 2016.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017a.

Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017b.

M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid. Local convolutional features with unsupervised training for image retrieval. In *ICCV*, pp. 91–99, Dec 2015.

L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, pp. 3406–3413, May 2016.

Etienne Pot, Alexander Toshev, and Jana Kosecka. Self-supervisory signals for object discovery and detection. *CoRR*, abs/1806.03370, 2018.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pp. 91–99. 2015.

Alvaro Collet Romea, Manuel Martinez Torres, and Siddhartha Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *IJRR*, 30(10):1284 – 1306, 2011.

Bryan C. Russell, William T. Freeman, Alexei A. Efros, Josef Sivic, and Andrew Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, pp. 1605–1614, 2006.

Andrew M Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *ICML*, pp. 1089–1096, 2011.

Samuel Schulter, Christian Leistner, Peter Roth, and Horst Bischof. Unsupervised object discovery and segmentation in videos. In *BMVC*, 2013.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. In *ICRA*, 2018.

J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *ICCV*, 2005.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *NIPS*, pp. 1857–1865. 2016.

T. Tuytelaars, C. H. Lampert, M. B. Blaschko, and W. Buntine. Unsupervised object discovery: A comparison. *IJCV*, 2009.

J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Deep image prior. *CoRR*, abs/1711.10925, 2017. URL `http://arxiv.org/abs/1711.10925`.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pp. 1096–1103. ACM, 2008.

Le Wang, Gang Hua, Rahul Sukthankar, Jianru Xue, and Nanning Zheng. Video object discovery and co-segmentation with extremely weak supervision. In *ECCV*, 2014.

Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pp. 1912–1920. IEEE Computer Society, 2015. ISBN 978-1-4673-6964-0.

Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, June 2015.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, 2017a.

Shun Zhang, Jia-Bin Huang, Jongwoo Lim, Yihong Gong, Jinjun Wang, Narendra Ahuja, and Ming-Hsuan Yang. Tracking persons-of-interest via unsupervised representation adaptation. *CoRR*, abs/1710.02139, 2017b. URL `http://arxiv.org/abs/1710.02139`.

APPENDIX

7 DATASET

Figure 8: Consecutive frames captured with our robotic setup. At each run we randomly select 10 objects and place them on the table. Then a robot moves around the table and take snapshots of the table at different angles. We collect in average 80-120 images per scene. We select pairs of two frames of the captured trajectory and train the OCN on the detected objects.

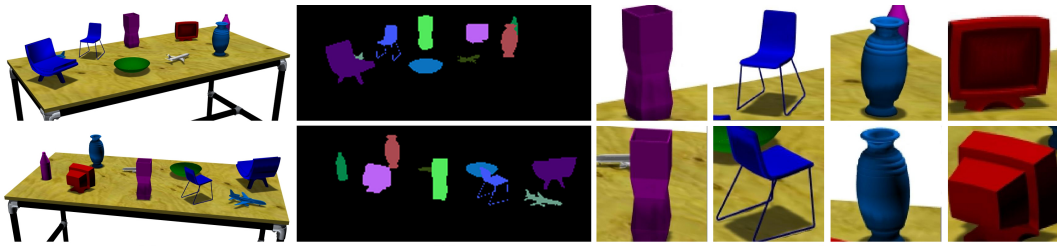

Figure 9: Synthetic data: two frames of a synthetically generated scene of table-top objects (a) and a subset of the detected objects (c). To validate our method against a supervised baseline, we additionally render color masks (b) that allow us to identify objects across the views and to associate them with their semantic attributes after object detection. Note that objects have the same color id across different views. The color id's allow us to supervise the OCN during training.

Table 4: Datasets overview: to train an OCN we use real and synthetic data.

Dataset	#Categories	#Unique Objects	#Scenes	#Views per Scene	#View Pairs
Synthetic	12	4k	250k	2	250K
Real	6	187	576	115-230	400K

Table 5: Real object dataset: we use 187 unique object instances spread across six categories.

	Balls	Bottles & Cans	Bowls	Cups & Mugs	Glasses	Plates
Training	14	13	19	19	22	23
Validation	5	4	8	6	5	6
Test	6	6	10	6	6	9
Total	25	23	37	31	33	38

8 RANDOM WEIGHTS

We find in Table 6 that models that are not pretrained with ImageNet supervision perform worse but still yield reasonable results. This indicates that the approach does not rely on a good initialization to bootstrap itself without labels. Even more surprisingly, when freezing the weights of the ResNet50 base of the model to its random initialization, results degrade but still remain far below chance as well as below the 'ResNet50 embeddings' baseline. Obtaining reasonable results with random weights has already been observed in prior work such as (Jarrett et al., 2009), (Saxe et al., 2011) and (Ulyanov et al., 2017).

Table 6: **Results with random weights** (no ImageNet pre-training)

Method	Class (12) Attribute Error	Color (8) Attribute Error	Binary Attributes Error	Finetuning
[baseline] Softmax	23.18%	10.72%	13.56%	yes
[baseline] OCN supervised (NN)	29.99%	2.23%	20.25%	yes
[baseline] OCN supervised (linear)	34.17%	2.63%	27.37%	yes
[ours] OCN unsupervised (NN)	35.51%	2.93%	22.59%	yes
[ours] OCN unsupervised (linear)	47.64%	4.43%	35.73%	yes
[baseline] Softmax	27.28%	5.48%	20.40%	no
[baseline] OCN supervised (NN)	37.90%	4.00%	23.97%	no
[baseline] OCN supervised (linear)	39.98%	4.68%	32.74%	no
[ours] OCN unsupervised (NN)	43.01%	5.56%	26.29%	no
[ours] OCN unsupervised (linear)	48.26%	6.15%	37.05%	no
[baseline] ResNet50 embeddings (NN)	59.65%	21.14%	34.94%	no
[baseline] Chance	91.68%	87.50%	50.00%	-

9 ADDITIONAL QUALITATIVE RESULTS

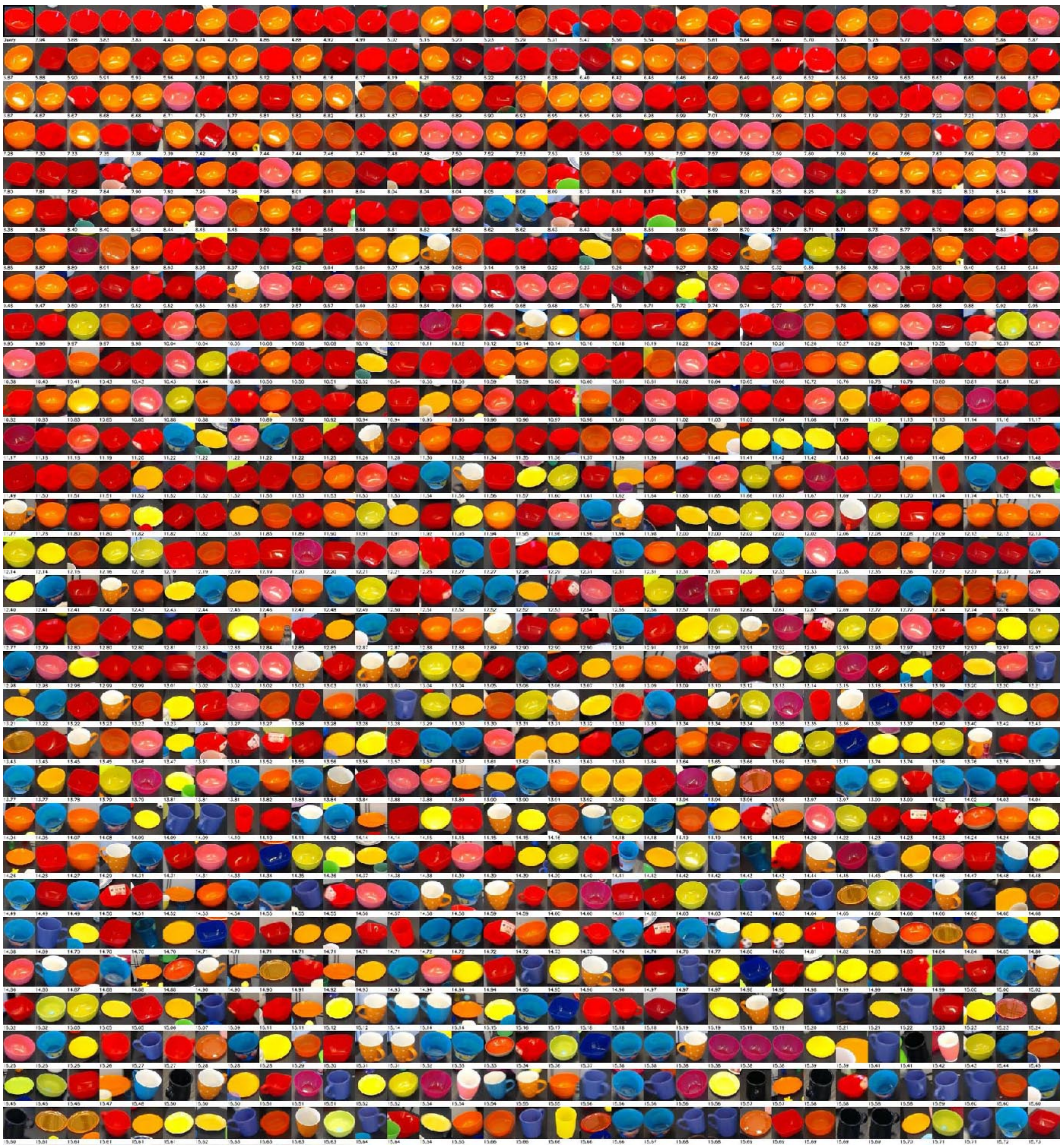

Figure 10: A result showing the organization of real bowls based on OCN embeddings. The query object (black border, top left) was taken from the validation all others from the training data. As the same object is used in multiple scenes the same object is shown multiple times.

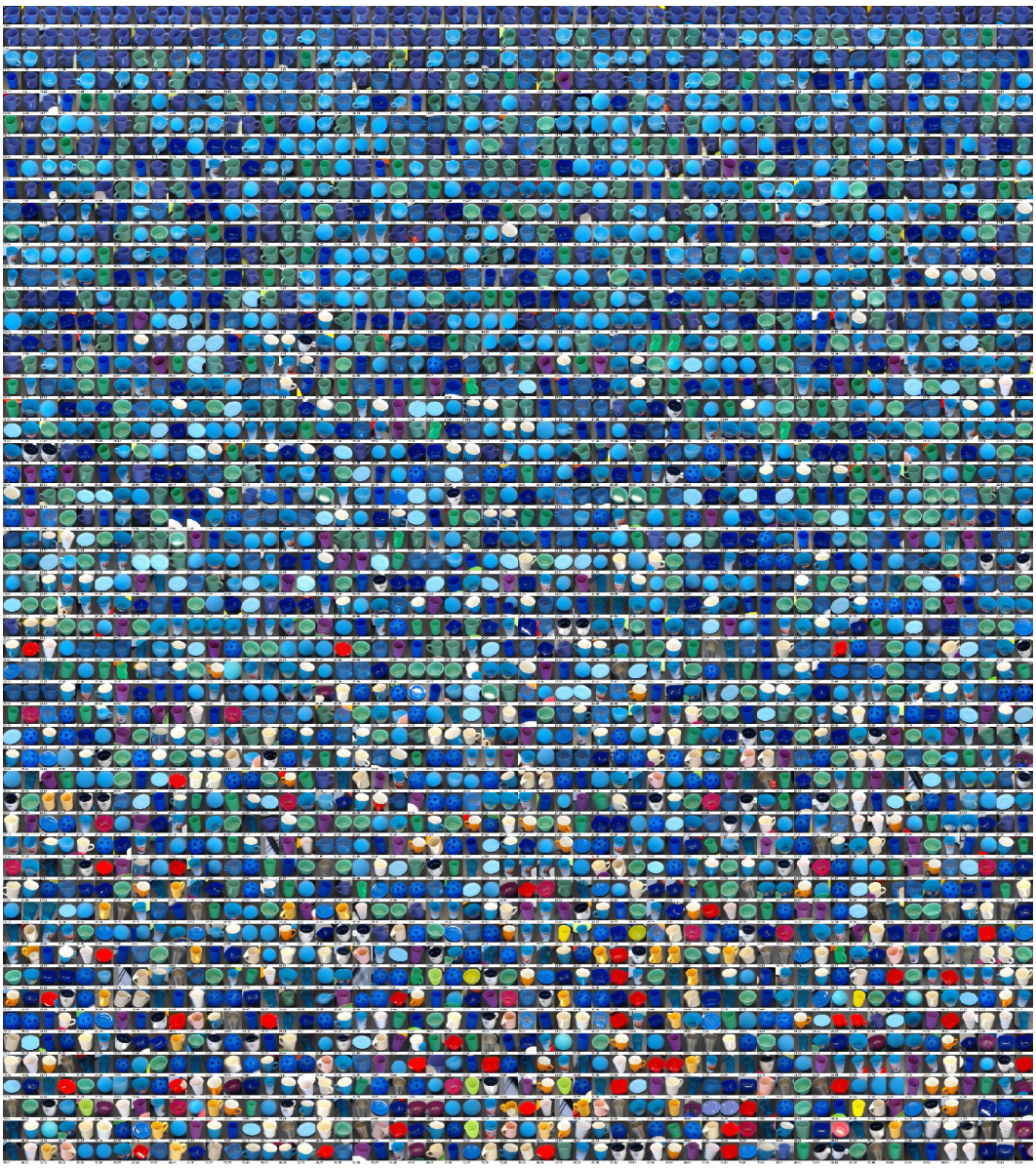

Figure 11: A result showing the organization of real bowls based on OCN embeddings. The query object (black border, top left) was taken from the validation all others from the training data. As the same object is used in multiple scenes the same object is shown multiple times.

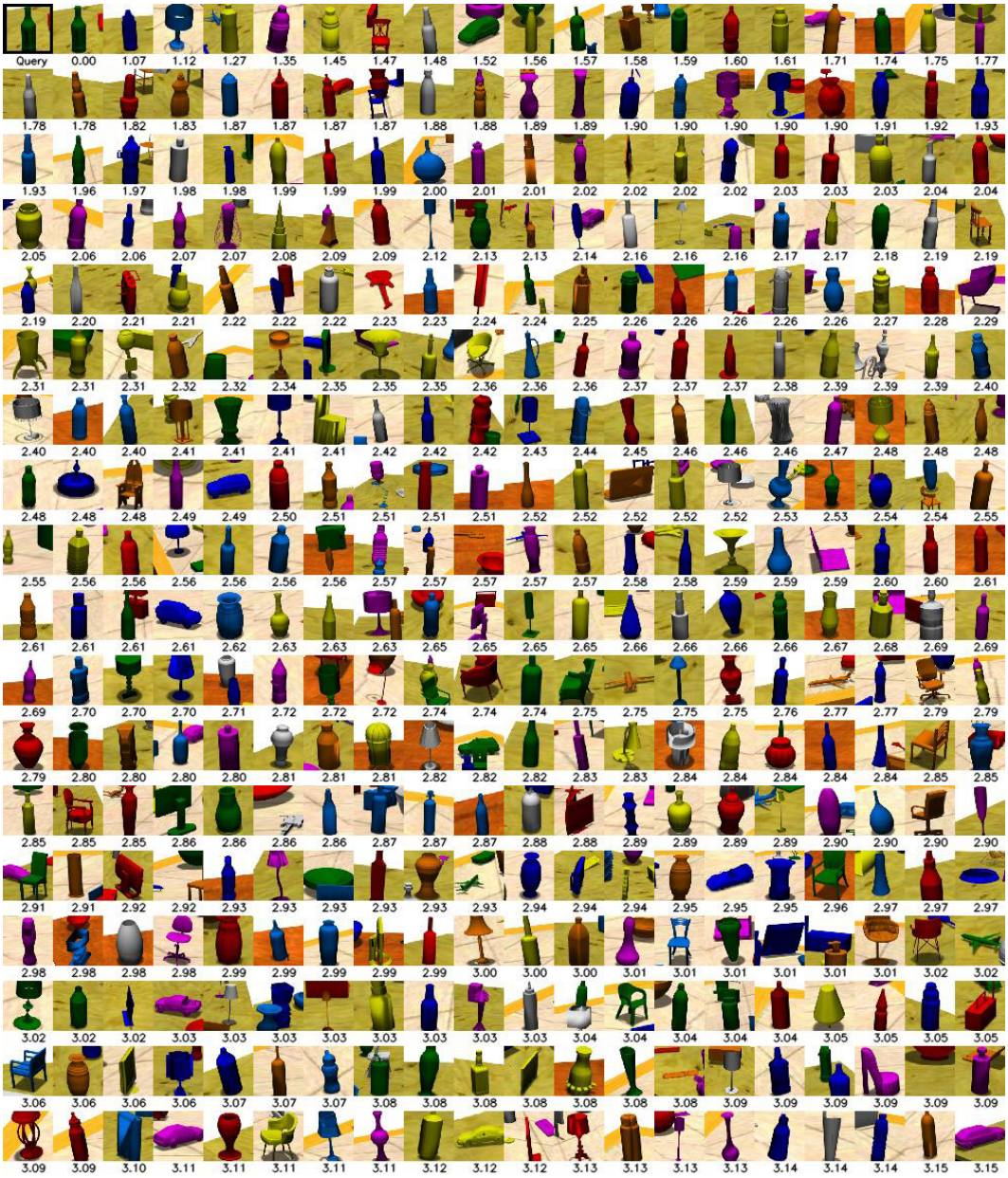

Figure 12: A result showing the organization of bottles from synthetic data based on OCN embeddings. The query object (black border, top left) was taken from the validation all others from the training data.

Figure 13: A result showing the organization of vases from synthetic data based on OCN embeddings. The query object (black border, top left) was taken from the validation all others from the training data.

Figure 14: A result showing the organization of chairs from synthetic data based on OCN embeddings. The query object (black border, top left) was taken from the validation all others from the training data.

10 ADDITIONAL ROBOTIC POINTING RESULTS

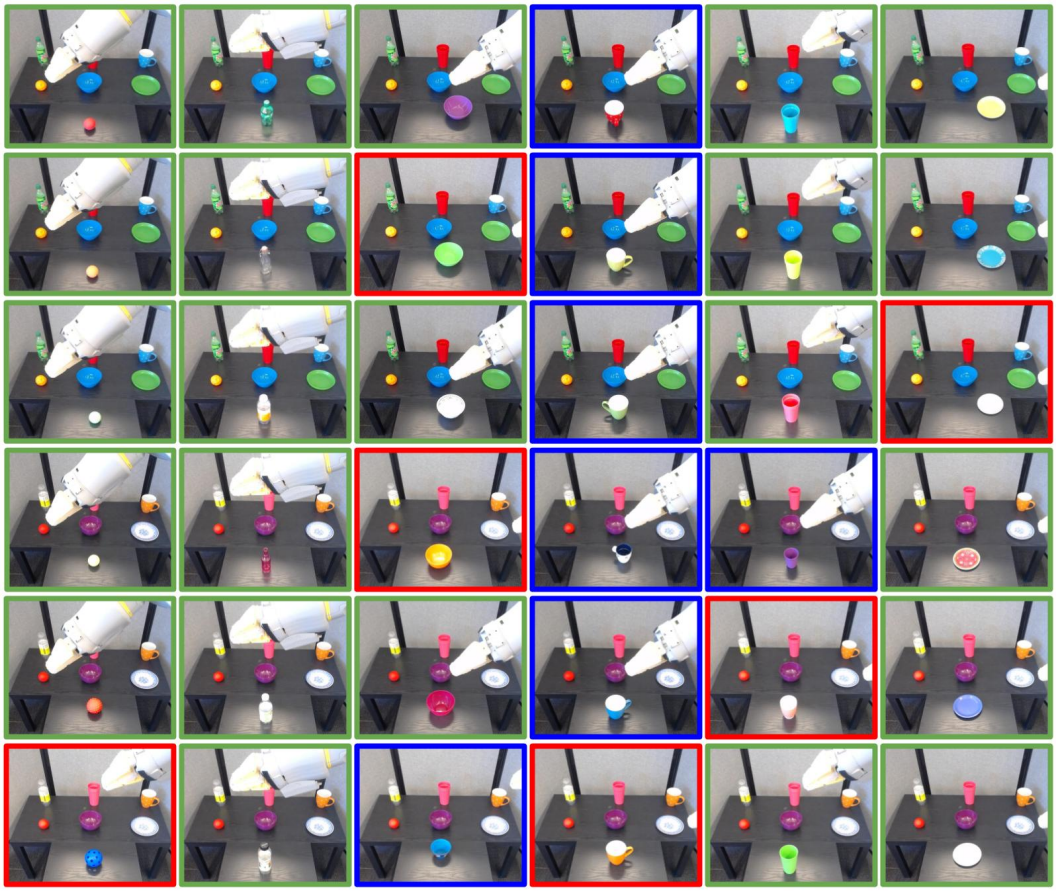

Figure 15: The robot experiment of pointing to the best match to a query object (placed in front of the robot on the small table). The closest match is selected from two sets of target object list, which are placed on the long table behind the query object. The first and the last three rows respectively correspond to the experiment for the first and second target object lists. Each column also illustrates the query objects for each object category. Image snapshots with green frame correspond to cases where both the 'class' and 'container' attributes are matched correctly. Image snapshots with blue frame refer to the cases where only 'container' attribute is matched correctly. Images with red frames indicates neither of attributes are matched.

11 ADDITIONAL NEAREST NEIGHBOR RESULTS

Figure 16: An OCN embedding organizes objects along their visual and semantic features. For example, a red bowl as query object is associated with other similarly colored objects and other containers. The leftmost object (black border) is the query object and its nearest neighbors are listed in descending order. The top row shows renderings of our synthetic dataset, while the bottom row shows real objects. For real objects we removed the same instance manually.