SORNet: Spatial Object-Centric Representations for Sequential Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Sequential manipulation tasks require a robot to perceive the state of an environment and plan a sequence of actions leading to a desired goal state, where the ability to reason about spatial relationships among object entities from raw sensor inputs is crucial. Prior works relying on explicit state estimation or end-to-end learning struggle with novel objects. In this work, we propose **SORNet** (**S**patial **O**bject-Centric **R**epresentation **Net**work), which extracts object-centric representations from RGB images conditioned on canonical views of the objects of interest. We show that the object embeddings learned by SORNet generalize *zero-shot* to *unseen* object entities on three spatial reasoning tasks: spatial relationship classification, skill precondition classification and relative direction regression, significantly outperforming baselines. Further, we present real-world robotic experiments demonstrating the usage of the learned object embeddings in task planning for sequential manipulation.

Keywords: Object-centric Representation, Spatial Reasoning, Manipulation

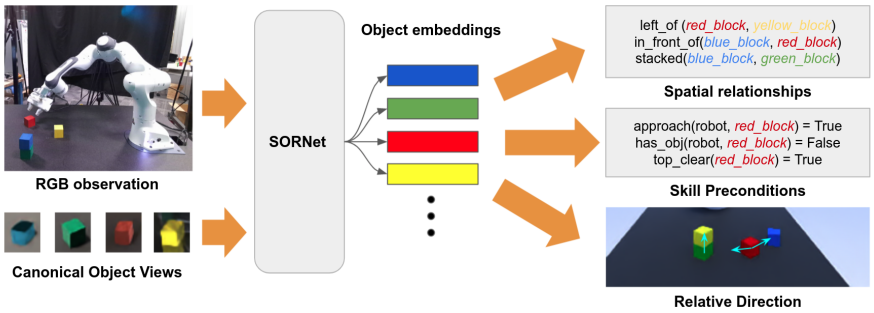


Figure 1: We propose **SORNet** (**S**patial **O**bject-Centric **R**epresentation **Net**work) that learns object embeddings useful for spatial reasoning tasks such as predicting spatial relationships, classifying skill preconditions, and regressing relative direction between objects.

1 Introduction

A crucial question for complex multi-step robotic tasks is how to represent relationships between entities in the world, particularly as they pertain to preconditions for various skills the robot might employ. In goal-directed sequential manipulation tasks with long-horizon planning, it is common to use a state estimator followed by a task and motion planner or other model-based system [1, 2, 3, 4, 5, 6]. A variety of powerful approaches exist for explicitly estimating the state of objects in the world, e.g. [7, 8, 9, 10]. However, it is challenging to generalize these approaches to an arbitrary collection of objects. In addition, the objects are often in contact in manipulation scenarios, where works explicitly addressing the problem of generalizing to unseen objects [11, 12] still struggle.

Fortunately, knowing exact poses of objects may not be necessary for manipulation. End-to-end methods [13, 14, 15] leverage that fact and build networks that generates actions directly from sensor inputs without explicitly representing objects. Nevertheless, these networks are very specific to the

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.

tasks they are trained on. For example, it is non-trivial to use a network trained on stacking blocks to unstack blocks.

In this work, we take an important step towards a manipulation framework that generalizes zero-shot to novel tasks with unseen objects. Specifically, we propose a neural network that extracts object-centric embeddings directly from raw RGB images, which we call **SORNet**, or **S**patial **O**bject-Centric **R**epresentation **Net**work. The design of SORNet allows it to generalize to novel objects and tasks without retraining or finetuning. Trained from large amounts of simulated robotic manipulation data, the object-centric embeddings produced by **SORNet** can be used to inform a task and motion planner with implicit object state information relevant to goal-directed sequential manipulation tasks, e.g. logical preconditions for primitive skills or continuous 3D directions between the entities in the scene.

To summarize, our contribution are: (1) a method for extracting object-centric embeddings from RGB images that generalizes zero-shot to different number and type of objects; (2) a framework for learning object embeddings that capture continuous spatial relationships with only logical supervision; (3) a dataset containing sequences of RGB observations labeled with spatial predicates during various tabletop rearrangement manipulation tasks.

We empirically evaluate the object-centric embeddings produced by **SORNet** on three different downstream tasks: 1) classification of logical predicates for action preconditions; 2) visual question and answering for spatial relationships; 3) prediction of relative 3D direction between entities in manipulation scenes, in all cases testing on held-out objects that did not appear in training data. In all tasks, **SORNet** obtains significant improvements over the baseline methods. Finally, we evaluated SORNet with real-world robot experiments to showcase the transfer of the learned object-centric embeddings to real-world observations.

2 Related Work

Sequential Manipulation In goal-directed sequential manipulation tasks with long-horizon planning, a class of work uses a pipeline approach requiring a state estimator followed by a task and motion planner [1, 2, 3, 4, 5, 6], but the state estimator with an explicit state representation is often restricted to an environment or a set of objects.

Another class of work employs neural networks to learn motor controls directly from raw sensor data, such as RGB images and joint encoder readings [14, 16, 15, 17, 18, 19], but they are often not transferable to new tasks, especially tasks involving long-horizon planning. Our work combines a network that estimates implicit states with a symbolic planner to obtain the best of both worlds: generalization to new objects and to new tasks.

Learning Spatial Relationships Learning spatial relationships between object entities have been studied in the field of 3D vision and robotics. Methods such as [20, 21, 22] predict discrete or continuous pairwise object relations from 3D inputs such as point clouds or voxels, assuming complete observation of the scene and segmented objects with identities. In contrast, our approach does not make any assumptions regarding the observability of the objects and does not require pre-processing of the sensor data. The learning framework by Kase et al. [23] is most related to our approach, which takes a sequence of sensor observations and classifies a set of pre-defined relational predicates which is then used by a symbolic planner to produce a suitable operator. Compared to our approach, theirs is limited by the number of objects in the scene and a fixed set of spatial predicates.

Visual Reasoning Recently, several advancements have been made on visual reasoning benchmarks [24, 25, 26] using transformer networks [27]. Toward solving spatio-temporal reasoning task from CLEVRER [25] and CATER [26], Ding et al. [28] proposed an object-based attention mechanism and Zhou et al. [29] proposed a multi-hop transformer model. Both works assume a segmentation model to produce object segments and performs language grounding to the segments to perform reasoning. Our SORNet architecture is simpler and can solve spatial-reasoning tasks for unseen object instances without requiring a segmentation or object detection module. Furthermore, our work focuses on a relatively complex manipulation task domain involving manipulator in the observations. Although our current work focuses on predicting spatial relations from a single RGB frame, the object-centric embeddings could potentially be used for solving temporal-reasoning tasks.

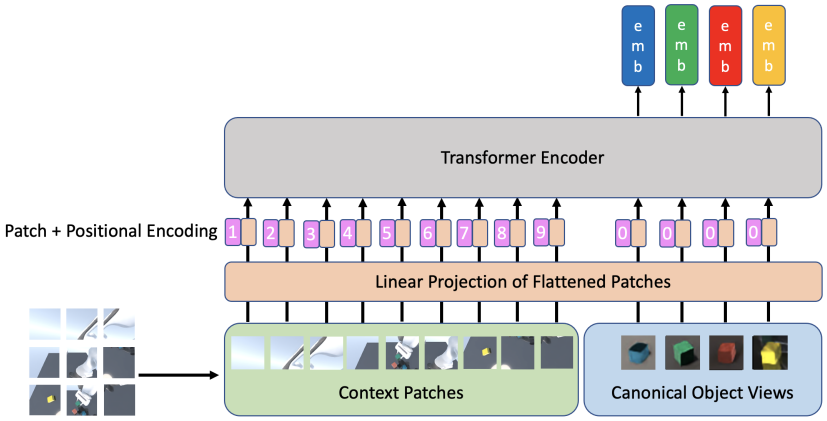


Figure 2: **SORNet** architecture. Input to the network is an RGB image and canonical views of the objects of interest. The RGB image is broken into context patches which have the same size as the canonical views. These patches are flattened and added with positional encoding and passed through a multi-layer multi-head transformer [30]. The embeddings corresponding to the canonical views are used for the downstream tasks.

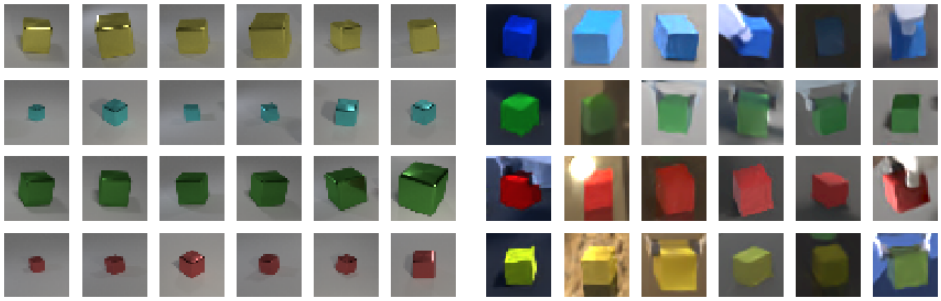


Figure 3: Examples of canonical object views used for training CLEVR (left) and Leonardo(right) dataset. Lighting, texture and object poses vary across different views, and sometimes there is occlusion by the robot.

3 Methods

3.1 SORNet: Spatial Object-Centric Representation Network

Our object embedding network (**SORNet**) (Fig. 2) takes an RGB image and an arbitrary number of canonical object views and outputs an embedding vector corresponding to each input object patch. The architecture of the network is based on the Visual Transformer (ViT) [30]. The input image is broken into a list of fixed-sized patches, which we call *context patches*. The context patches are concatenated with the canonical object views to form a patch sequence. Each patch is first flattened and then linearly projected into a token vector, then positional embedding is added to the sequence of tokens. Following [30], we use a set of learnable vectors with the same dimension as the token vectors as positional embeddings. The positional-embedded tokens are then passed through a transformer encoder, which includes multiple layers of multi-head self-attention. The transformer encoder outputs a sequence of embedding vectors. We discard the embedding for context patches and keep those for the canonical object views.

We apply the same positional embedding to the canonical object views to make the output embeddings permutation equivariant. We also mask out the attention among canonical object views and the attention from context patches to canonical object views to ensure the model uses information from the context patches to make predictions. In this way, we can pass in an arbitrary number of canonical object views in arbitrary order without changing model parameters during inference.

Intuitively, the canonical object views can be viewed as queries where the context patches serve as keys to extract the spatial relationships' values. Note that the canonical object views are *not* crops from the input image, but arbitrary views of the objects that may not match the objects' appearance in the scene. Our model learns to identify objects even under drastic change in lighting, pose and occlusion. Fig. 3 shows some examples of canonical object views used in our experiments.

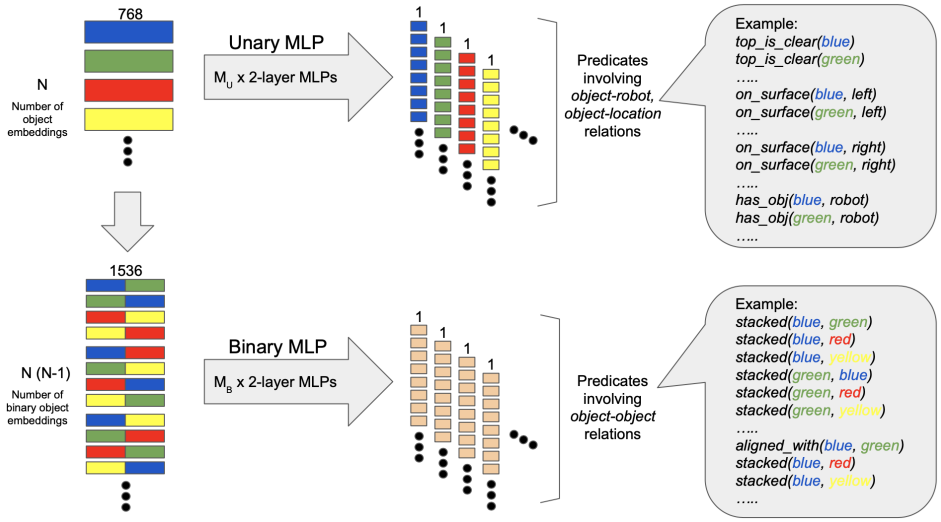


Figure 4: The embeddings from the **SORNet** is used to predict logical statements on spatial relations, which can serve as preconditions for primitive skills in manipulation tasks. This part of the framework is flexible to accommodate any number of objects in the scene. M_U and M_B denote the number of unary and binary predicate types respectively.

3.2 Predicate Classifier

The predicate classifier (Fig. 4) is responsible for predicting a list of predicates using object embeddings. The predicates can be logical statements about spatial relationships, e.g., whether the blue block is on the left part of the table, or continuous quantities such as which direction should the end effector move to reach the red block. The predicate classifier consists of a collection of 2-layer MLPs, one for each type of relationship. Here we focus on unary and binary predicates. Unary predicates involve a single object or an object and the environment, which could be the robot or a region on the table. Binary predicates involve two objects and, optionally, the environment. In principle, our framework is extensible to predicates involving more than two objects, but we leave that for future work.

The MLP for unary predicates takes the list of object embeddings produced by **SORNet** and outputs predicates pertaining to the object that the embedding is conditioned on. Taking the `top_is_clear` classifier for an example, if the input embedding is conditioned on the blue block, the MLP will output whether there is any object on top of the blue block. If the input embedding is conditioned on the red block, the MLP will output whether there is any object on top of the red block.

The MLP for binary predicates takes a list of binary object embeddings created by concatenating pairs of object embeddings produced by SORNet and outputs predicates corresponding to a pair of objects, e.g., whether the blue block is on top of the red block. Thus, with N object embeddings, there will be $N(N-1)$ binary object embeddings.

Parameters of the predicate classifier are independent of the number of objects. The number of output predicates dynamically changes with the number of input object embeddings. For example, when are 7 unary MLPs and 2 binary MLPs, with 4 objects, the model predicts $7\times4+2\times4\times(4-1) = 52$ predicates; with 5 objects, the model predicts $7 \times 5 + 2 \times 5 \times (5-1) = 75$ predicates. In this way, our model generalizes zero-shot to scenes with an arbitrary number of objects.

4 The Leonardo Dataset

To test the generalization of our model to new objects and tasks, we created a simulated tabletop environment named Leonardo, where a Franka Panda robot manipulates a set of randomly colored blocks. The robot is given a goal formulated as a list of predicates to be satisfied, and then use a simple task planner [5] to find a sequence of actions to achieve that goal and a sample-based motion planner [31] to generate trajectories and choose grasps. As we know the ground truth poses of the blocks in the simulator, we can compute ground-truth logical predicates at every step of the planning

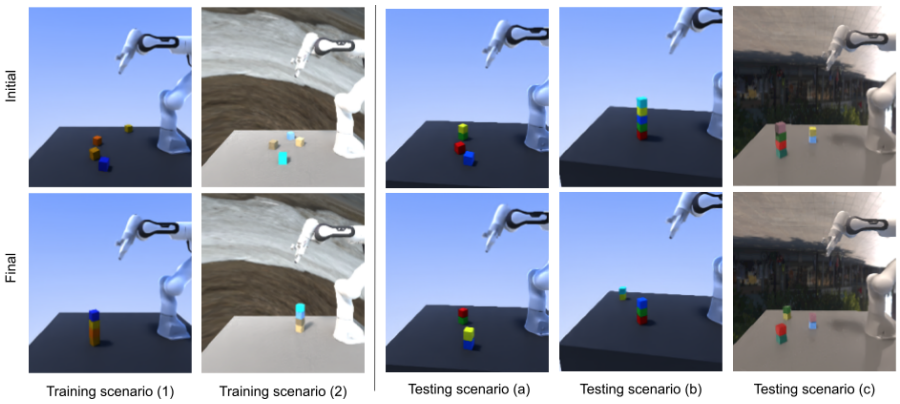


Figure 5: Sample scenes from training and testing scenarios in Leonardo dataset. Top row shows the initial configuration of a task scenario and the bottom row shows the goal condition. The training scenarios contains 4 blocks with random colors with a single tower stack as a goal condition. The testing scenarios contain 4-7 blocks with specific colors with various goal conditions involving multi-tower stacking scenarios.

process. We used NVISII [32] to render the RGB observations. Domain randomization including random lighting, background and perturbations to the camera position is applied while rendering.

The training data contains 133796 sequences of a single task - stacking 4 blocks in a tower. The block colors are randomly chosen from 405 xkcd colors[1]. The testing data contains 9526 sequences with 4-7 blocks, chosen from colors that are not included in the training data: *red, green, blue, yellow, aqua, pink, purple*. The testing tasks consists of 7 tasks different from training. Fig. 5 shows some examples. Please see the supplement for full description of the testing tasks.

5 Results

5.1 CLEVR-CoGenT

We first evaluate our approach on a variant of the CLEVR dataset [24], a well established benchmark for visual reasoning. CLEVR contains rendered RGB images with at most 10 objects per image. There are 96 different objects in total (2 sizes, 8 colors, 2 materials, 3 shapes). Each image is labeled with 4 types of spatial relationships (right, front, left, behind) for each pair of objects.

Specifically, we use the CoGenT version of the dataset, which stands for Compositional Generalization Test, where the data is generated in two different conditions. In condition A, cubes are gray, blue, brown, or yellow and cylinders are red, green, purple, or cyan. Condition B is the opposite: cubes are red, green, purple, or cyan and cylinders are gray, blue, brown, or yellow. Spheres can be any color in both conditions. The models are trained on condition A and evaluated on condition B. The training set (trainA) contains 70K images and the evaluation set (valB) contains 15K images. Several prior works [33, 34] show significant generalization gap on CLEVR-CoGenT caused by the visual model learning strong spurious biases between shape and color.

We generate a question for each spatial relationship in the image, e.g. "*Is the large red rubber cube in front of the small blue metal sphere?*" We filter out any query that is ambiguous, e.g. if there were two large red cubes, one in front and one behind the small blue sphere. This results in around 2 million questions for both valA and valB sets. We compare against MDETR [34], which reports state-of-the-art zero-shot result on CLEVR-CoGenT, i.e. there is no fine-tuning on any example from condition B. The results are summarized in Table 1. Our model performs drastically better on classifying spatial relationships of unseen objects and shows a much smaller generalization gap between valA and valB sets.

Unlike MDETR which takes text queries, our model takes visual queries in the form of canonical object views, and the set of questions we generate are different from the ones provided with CLEVR-CoGenT. To eliminate the influence of those factors, we report the performance of the MDETR model trained on the full CLEVR dataset, denoted as MDETR-oracle. We can see that although SORNet is only trained on condition A, it is able to achieve similar performance to MDETR-oracle.

[1] https://xkcd.com/color/rgb/

	MDETR [34]	MDETR-oracle [34]	**SORNet(ours)**
ValA Accuracy	84.950	97.944	**99.006**
ValB Accuracy	59.627	98.052	**98.222**

Table 1: Zero-shot relationship classification accuracy on CLEVR-CoGenT [24]. The MDETR-oracle model has seen all the objects during training, where as MDETR and SORNet have only see objects in condition A. SORNet takes canonical views as queries whereas MDETR takes text queries.

The zero-shot generalization ability of our model can potentially be combined with other reasoning pipelines to improve generalization performance on other types of queries as well.

5.2 Predicate Classification

Next, we evaluate the task of predicate classification on the Leonardo dataset. We compare against 3 baselines that do not use object conditioning. The first two baselines use a ResNet18 and a ViT-B/32 respectively to directly predict 52 predicates. The last baseline uses the same architecture as ours, but the embedding tokens come from 4 fixed class embedding vectors instead of being conditioned on the canonical object views. We report 3 metrics: accuracy, F-1 score and all-match accuracy for all predicates as well as predicates within a certain category. The accuracy and F-1 score are computed per predicate and averaged. For all-match accuracy, we consider the output predicates as vectors. If a single predicate is wrong, we consider the prediction for the entire category to be wrong. The all-match accuracy most accurately reflect real world performance of the predicate classifier, since the planner uses a vector of predicates instead of a single predicate.

As mentioned in Sec. 4, the models are tested on images where the objects are completely unseen during training. No fine-tuning on the test data is performed. All models except the single-view **SORNet** model are trained on 3 different views of the scene, which serves as a sort of data augmentation. During testing, we aggregate the predictions from 3 views by adding the logits to minimize the effect of occlusion. We denote this multi-view prediction by M-View in Table 2. We also concatenate the gripper state to the object embedding in a variant of **SORNet**, denoted by (G) in Table 2. The opening/closing of the gripper is hard to see visually but provides useful information for predicate classification.

The results are summarized in Table 2. We can see that the non-object-conditioned baselines fail drastically when applied to unseen objects zero-shot. Even after fine-tuning on 100 examples, they still significantly underperform zero-shot SORNet. This demonstrates the generalizability of our model obtained via conditioning on canonical object views.

Further, we tested our model on scenes with 5 to 7 objects, while it has only been trained on 4-object scenes. The performance of our model drops for some types of predicates (e.g. stacked) but remains the same for most predicates. Note that the number of binary predicates increases quadratically with the number of objects in the scene, which also increases complexity of the scene. None of the baselines can even be applied to these scenes with more objects without introducing additional model parameters and retraining the model.

Finally, we run our best performing model on 30 real-world images of a robot performing various manipulation tasks. The quantitative results are in Table 2 and Fig. 6 shows two qualitative examples of the predicates predicted by our model. Our model transfers to real-world without losing much accuracy. It does make some mistakes when encountered with novel scenarios never seen during training, such as one block stacked in between two blocks (right plot in Fig. 6).

5.3 Skill Executability

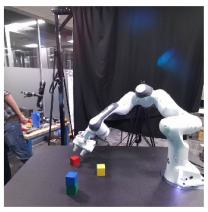
We further evaluate the predicate prediction in the context of task planning. Each frame in the Leonardo dataset is labeled with a primitive skill that will be executed, e.g. grasp the red block. Each skill has a list of preconditions that needs to be satisfied before it can be executed, which can be formulated as a vector of predicate values. In other words, if all of the predicates in the preconditions are classified correctly, we can determine the executability of that skill correctly. In Table 3, we report the accuracy for classifying the executability of skills in the Leonardo test set using the predicate predictions.

Method	# pred	Accuracy						
		all	on_surface	has_obj	top_clear	stacked	aligned	approach
Majority	52	88.2	79.3	88.4	75.1	91.7	99.2	92.9
ResNet18 M-Head 0-shot	52	88.5	80.0	88.4	76.2	91.7	99.2	92.9
ViT-B/32 M-View 0-shot	52	88.5	80.2	88.4	76.2	91.7	99.2	92.9
ViT-B/32 M-Head M-View 0-shot	52	88.5	80.2	88.4	76.0	91.7	99.2	92.9
ResNet18 M-Head 100-shot	52	88.5	80.0	88.4	76.4	91.7	99.1	92.9
ViT-B/32 M-View 100-shot	52	88.6	79.9	88.3	78.7	91.7	99.1	92.8
ViT-B/32 M-Head M-View 100-shot	52	92.8	89.9	90.4	87.8	92.4	99.2	93.5
SORNet 0-shot	52	97.6	96.9	95.3	96.6	98.7	99.2	96.1
SORNet M-View 0-shot	52	**98.9**	**99.0**	95.9	**99.2**	**99.6**	**99.4**	**97.3**
SORNet M-View (G) 0-shot	52	**98.9**	98.9	**98.8**	98.5	99.5	99.3	96.6
SORNet M-View (G) 5 obj 0-shot	70	98.5	98.5	99.4	95.8	98.2	99.6	97.4
SORNet M-View (G) 6 obj 0-shot	102	98.0	98.3	99.6	93.9	96.8	99.7	97.7
SORNet M-View (G) 7 obj 0-shot	133	97.6	97.9	99.3	91.1	96.0	99.8	97.8
SORNet M-View (G) real-world 0-shot	52	96.3	96.4	96.7	93.3	97.1	96.7	95.6

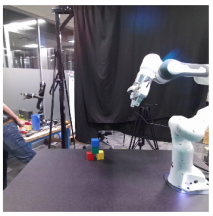
Method	# pred	All-match Accuracy						
		all	on_surface	has_obj	top_clear	stacked	aligned	approach
ResNet18 M-Head 0-shot	52	0.0	0.3	53.5	30.4	30.1	89.8	71.6
ViT-B/32 M-View 0-shot	52	0.0	0.4	53.5	30.3	30.1	89.8	71.6
ViT-B/32 M-Head M-View 0-shot	52	0.0	0.3	53.5	30.3	30.1	89.8	71.6
ResNet18 M-Head 100-shot	52	0.0	0.3	53.5	30.9	30.1	89.8	71.6
ViT-B/32 M-View 100-shot	52	0.2	3.3	55.0	40.5	31.0	89.8	72.5
ViT-B/32 M-Head M-View 100-shot	52	4.1	21.7	62.4	63.8	40.0	89.8	75.0
SORNet 0-shot	52	43.8	66.4	81.4	87.8	87.9	90.5	86.1
SORNet M-View 0-shot	52	60.2	**86.3**	83.5	**96.8**	**95.8**	**92.4**	**89.5**
SORNet M-View (G) 0-shot	52	**63.6**	84.5	**95.8**	94.0	94.3	92.0	86.6
SORNet M-View (G) 5 obj 0-shot	70	45.5	74.8	97.6	81.1	72.6	92.0	87.5
SORNet M-View (G) 6 obj 0-shot	102	29.7	68.9	97.7	70.2	52.4	92.0	87.5
SORNet M-View (G) 7 obj 0-shot	133	17.0	58.2	95.7	55.4	36.1	91.8	86.6
SORNet M-View (G) real-world 0-shot	52	26.7	63.3	93.3	80.0	76.7	90.0	80.0

Method	# pred	F-1 Score						
		all	on_surface	has_obj	top_clear	stacked	aligned	approach
ResNet18 M-Head 0-shot	52	9.7	23.6	0.0	31.5	0.0	0.0	0.0
ViT-B/32 M-View 0-shot	52	11.9	37.4	0.0	28.9	0.0	0.0	0.1
ViT-B/32 M-Head M-View 0-shot	52	12.2	32.5	0.0	27.7	0.0	0.0	0.0
ResNet18 M-Head 100-shot	52	0.0	21.9	0.0	32.6	0.0	0.0	0.0
ViT-B/32 M-View 100-shot	52	0.0	37.7	6.3	46.5	0.0	0.0	7.3
ViT-B/32 M-Head M-View 100-shot	52	0.0	70.5	31.0	73.2	27.2	0.0	23.2
SORNet 0-shot	52	83.2	92.2	79.7	93.0	91.2	63.8	74.9
SORNet M-View 0-shot	52	88.9	**97.5**	82.0	**98.4**	**97.3**	**70.5**	**81.7**
SORNet M-View (G) 0-shot	52	**89.5**	97.1	**94.7**	96.8	96.4	69.9	76.7
SORNet M-View (G) 5 obj 0-shot	70	85.3	96.0	96.7	91.3	83.6	69.8	78.1
SORNet M-View (G) 6 obj 0-shot	102	79.9	95.5	97.0	87.5	69.2	70.0	77.9
SORNet M-View (G) 7 obj 0-shot	133	73.2	94.3	94.5	82.0	58.0	65.4	75.5
SORNet M-View (G) real-world 0-shot	52	76.5	90.7	85.7	80.3	69.1	33.3	68.9

Table 2: Zero-shot predicate classification results on the Leonardo test data where the objects are held-out from training. SORNet significantly outperforms baselines that are not object-conditioned, and is able to generalize to scenes with a different number of objects without retraining or finetuning.



on_surface(green_block, left)
on_surface(red_block, right)
on_surface(red_block, far)
on_surface(yellow_block, center)
top_is_clear(red_block)
top_is_clear(blue_block)
top_is_clear(yellow_block)
in_approach_region(robot, red_block)
stacked(green_block, blue_block)



on_surface(red_block, right)
on_surface(yellow_block, right)
top_is_clear(red_block)
top_is_clear(blue_block)
stacked(red_block, green_block)
stacked(green_block, blue_block)
stacked(yellow_block, green_block)

Figure 6: Visualization of predicted predicates in real-world scenes. Black means true positive, blue means false positive and red means false negative. True negatives are not shown due to limited space.

Method	Accuracy						
	all	approach	grasp	align	place	lift	go_home
ResNet18 M-View	27.0	88.7	0.0	0.0	0.0	0.0	100.0
ViT-B/32 M-View	27.4	90.6	0.0	0.0	0.0	0.0	100.0
ViT-B/32 M-Head M-View	27.3	90.4	0.0	0.0	0.0	0.0	100.0
SORNet	63.7	64.3	72.4	67.4	99.4	53.0	97.6
SORNet M-View	63.6	65.1	69.1	99.8	52.9	69.7	99.5
SORNet M-View (G)	76.3	98.7	68.1	99.9	99.9	69.7	100.0

Table 3: Skill executability accuracy on the Leonardo test data. SORNet is the only model that is able to correctly determine the executability of skills relevant to the handling of objects.

Method	ResNet18	ResNet18 (MV)	ResNet18 (P)	CLIP-ViT	CLIP-ViT (P)	SORNet (P)	SORNet (P MV)
Obj-Obj	0.4308	0.6068	0.9876	0.9875	0.6145	0.1679	**0.1458**
EE-Obj	0.3251	0.3464	0.5929	0.6544	0.4960	0.1962	**0.1777**

Table 4: Euclidean error on regression of continuous 3D unit vector between entities in the scene. The regressors are trained on 1000 examples with unseen objects. Methods labeled with P are pretrained on the Leonardo dataset. Methods labeled with MV use 3 views.

This evaluation puts more emphasis on the predicates relevant to the manipulation of objects, e.g. `approach`, `aligned`, which are rarely true in the training data. Our model is able to identify these predicates correctly whereas the baselines fail completely on skills relevant to object manipulation, i.e. `grasp`, `align`, `place` and `lift`.

5.4 Open Loop Planning

In this experiment, we incorporate **SORNet** as a part of an open-loop planning pipeline. Specifically, given an initial frame, we use the predicates predicted by SORNet M-View (G) to populate a state vector. A task and motion planner takes the state vector and desired goal (formulated as a list of predicate values to be satisfied), and outputs a sequence of primitive skills. The robot then executes this sequence of skills in an open loop fashion. This demonstrate that how SORNet can be applied to sequential manipulation of unseen object in a zero-shot fashion, i.e. without any fine-tuning on the test objects. Please refer to the supplement for the demo video.

5.5 Relative Direction Prediction

Although the training objective of SORNet is purely logical, with large-scale pre-training, the object embeddings learned by SORNet contains continuous spatial information. We demonstrate this fact by using SORNet embeddings to predict the relative 3D direction between entities. Specifically, we trained a regressor (same architecture as the classifier in Sec. 3.2) on top of frozen SORNet embeddings to predict the continuous direction between two objects (Obj-Obj) or the direction the end effector should move to reach a certain object (EE-Obj). The regressor is trained using L2 loss on a 1000 examples with unseen objects.

The results are summarized in Table 4. SORNet is able to outperform models trained from scratch with pose supervision (ResNet18) as well as models initialized with weights from large-scale language-image pretraining [35] (CLIP-ViT). This demonstrates that our representation learning technique is more suited to manipulation scenarios where more precise spatial information is crucial. In our supplement video, we showed that by predicting the direction for movement in an online fashion, our model can be used for visual servoing to guide the robot to reach a target object.

6 Conclusion

We proposed **SORNet** (**S**patial **O**bject-Centric **R**epresentation **Net**work) that learns object-centric representations from RGB images. We show that the object embeddings produced by **SORNet** capture spatial relationships which can be used in a downstream tasks such as spatial relationship classification, skill precondition classification and relative direction regression. Our method works on scenes with an arbitrary number of unseen objects in a zero-shot fashion. With real-world robot experiments, we demonstrate how **SORNet** can be used in manipulation of novel objects.

References

[1] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208, 1971.

[2] M. Fox and D. Long. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research*, 20:61–124, 2003.

[3] S. Balakirsky, Z. Kootbally, C. Schlenoff, T. Kramer, and S. Gupta. An industrial robotic knowledge representation for kit building applications. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1365–1370. IEEE, 2012.

[4] F. Rovida, B. Grossmann, and V. Krüger. Extended behavior trees for quick definition of flexible robotic tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6793–6800. IEEE, 2017.

[5] C. Paxton, N. Ratliff, C. Eppner, and D. Fox. Representing robot task plans as robust logical-dynamical systems. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5588–5595. IEEE, 2019.

[6] Z. Sui, L. Xiang, O. C. Jenkins, and K. Desingh. Goal-directed robot manipulation through axiomatic scene estimation. *The International Journal of Robotics Research*, 36(1):86–104, 2017.

[7] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[8] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 699–715, 2018.

[9] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[10] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox. Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking. *IEEE Transactions on Robotics*, 2021.

[11] C. Xie, Y. Xiang, A. Mousavian, and D. Fox. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation. In *Conference on robot learning*, pages 1369–1378. PMLR, 2020.

[12] Y. Xiang, C. Xie, A. Mousavian, and D. Fox. Learning rgb-d feature embeddings for unseen object instance segmentation. *arXiv preprint arXiv:2007.15157*, 2020.

[13] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[14] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[15] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. In *NIPS*, 2017.

[16] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman. Deep predictive policy training using reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2351–2358. IEEE, 2017.

[17] D.-A. Huang, S. Nair, D. Xu, Y. Zhu, A. Garg, L. Fei-Fei, S. Savarese, and J. C. Niebles. Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8565–8574, 2019.

[18] C. Paxton, Y. Bisk, J. Thomason, A. Byravan, and D. Foxl. Prospection: Interpretable plans from language by predicting the future. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6942–6948. IEEE, 2019.

[19] T. Yu, P. Abbeel, S. Levine, and C. Finn. One-shot hierarchical imitation learning of compound visuomotor tasks. *arXiv preprint arXiv:1810.11043*, 2018.

[20] B. Rosman and S. Ramamoorthy. Learning spatial relationships between objects. *The International Journal of Robotics Research*, 30(11):1328–1342, 2011.

[21] S. Fichtl, A. McManus, W. Mustafa, D. Kraft, N. Krüger, and F. Guerin. Learning spatial relationships from 3d vision using histograms. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 501–508. IEEE, 2014.

[22] M. Sharma and O. Kroemer. Relational learning for skill preconditions. *arXiv preprint arXiv:2012.01693*, 2020.

[23] K. Kase, C. Paxton, H. Mazhar, T. Ogata, and D. Fox. Transferable task execution from pixels through deep planning domain learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10459–10465. IEEE, 2020.

[24] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

[25] K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, and J. B. Tenenbaum. CLEVRER: collision events for video representation and reasoning. In *ICLR*, 2020.

[26] R. Girdhar and D. Ramanan. CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning. In *ICLR*, 2020.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[28] D. Ding, F. Hill, A. Santoro, and M. Botvinick. Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures. *arXiv preprint arXiv:2012.08508*, 2020.

[29] H. Zhou, A. Kadav, F. Lai, A. Niculescu-Mizil, M. R. Min, M. Kapadia, and H. P. Graf. Hopper: Multi-hop transformer for spatiotemporal reasoning. *arXiv preprint arXiv:2103.10574*, 2021.

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[31] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[32] N. Morrical, J. Tremblay, S. Birchfield, and I. Wald. ViSII: Virtual scene imaging interface, 2020. https://github.com/owl-project/ViSII/.

[33] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *arXiv preprint arXiv:1810.02338*, 2018.

[34] A. Kamath, M. Singh, Y. LeCun, I. Misra, G. Synnaeve, and N. Carion. Mdetr–modulated detection for end-to-end multi-modal understanding. *arXiv preprint arXiv:2104.12763*, 2021.

[35] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.