Multi-Object Representation Learning via Feature Connectivity and Object-Centric Regularization

Alex Foo **Wynne Hsu** **Mong Li Lee**
School of Computing
National University of Singapore
{alexfoo,whsu,leeml}@comp.nus.edu.sg

Abstract

Discovering object-centric representations from images has the potential to greatly improve the robustness, sample efficiency and interpretability of machine learning algorithms. Current works on multi-object images typically follow a generative approach that optimizes for input reconstruction and fail to scale to real-world datasets despite significant increases in model capacity. We address this limitation by proposing a novel method that leverages feature connectivity to cluster neighboring pixels likely to belong to the same object. We further design two object-centric regularization terms to refine object representations in the latent space, enabling our approach to scale to complex real-world images. Experimental results on simulated, real-world, complex texture and common object images demonstrate a substantial improvement in the quality of discovered objects compared to state-of-the-art methods, as well as the sample efficiency and generalizability of our approach. We also show that the discovered object-centric representations can accurately predict key object properties in downstream tasks, highlighting the potential of our method to advance the field of multi-object representation learning.

1 Introduction

Human understanding of the world relies on objects as compositional building blocks [24], and emulating this through object-centric representations can improve robustness, sample efficiency, generalization to out-of-domain distributions, and interpretability of machine learning algorithms [18, 9]. Recent work utilizes a generative approach, optimizing pixel-based reconstruction to learn object-centric representations [8, 17, 33, 4, 35, 12, 13, 11, 45, 46, 51]. This approach has limitations as it prioritizes pixel accuracy over object discovery and functional feature extraction [30, 11]. This may lead to the failure of discovering objects [35], or obtaining useful object features, such as position, shape, or boundaries between overlapping objects [23]. Additionally, pixel-based reconstruction tends to waste model capacity on less important visual features, such as complex backgrounds [25], making scaling these methods to real-world images a challenge.

To address the fundamental limitations of pixel-based reconstruction, we propose a framework that leverages feature connectivity and design two object-centric regularization terms to directly refine object representations, ensuring sufficient separation and high disentanglement between dimensions. Our method utilizes visual connectedness principles [38], where similar pixels that are connected should belong to the same object, to guide object discovery. The two regularization terms promote disentangled representations and prevent sub-optimal clustering.

We demonstrate that our approach outperforms state-of-the-art methods in discovering multiple objects from simulated, real-world, complex texture and common object images in a fine-grained manner without supervision. The proposed solution attains sample efficiency and is generalizable to out-of-domain images. The learned object representations also accurately predict key object

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

properties in downstream tasks. Our contributions include: (1) a framework that leverages feature connectivity for fine-grained object discovery, (2) introduction of object-centric regularization terms as an alternative to pixel-based reconstruction loss, (3) experimental validation of our solution's superior performance, and (4) demonstration of the usefulness of discovered object representations in downstream tasks.

2 Related Work

Numerous works have demonstrated remarkable success in segmenting real-world images. Most of these works focus on semantic segmentation and object detection by utilizing supervised signals [19, 6]. In contrast, unsupervised approaches like SLIC [1] employ a modified k-means algorithm to cluster pixels into superpixels, similar to Felzenszwalb's algorithm [15] that relies on hand-crafted features for clustering. However, these methods do not focus on learning useful representations for the segmented components.

Various unsupervised methods for learning object-centric representations have been proposed, and can be categorized into three main approaches: spatial attention, sequential attention, and iterative attention. Spatial attention approach utilizes spatial transformer networks [21] to crop out rectangular regions from an image and extract object attributes such as position and scale [14, 31, 8, 33]. They rely on a fixed-size sampling grid which may not be suitable for scenes with varying object sizes, and may compromise training when the sampling grid does not overlap with any object.

The sequential attention approach uses RNN-based models such as MONet [5] and GENESIS [12] to sequentially attend to different regions in an image. These methods employ a deterministic network to perform the attention process, which allows them to capture and represent objects in the scene. However, these methods may neglect smaller objects as they tend to produce a weaker signal during the attention process. This can lead to incomplete or biased representations of scenes with objects of varying sizes. To overcome this, GENESIS-V2 [13] uses a stochastic stick-breaking process to perform attention randomly.

In the iterative attention approach, a set of object representations is randomly initialized and then iteratively refined to bind these objects to different regions of an image. IODINE [17] is a model that can discover objects with disentangled representations. However, it requires long training times and many samples. Slot Attention [35] introduces competition among the object representations by utilizing cross-attention along the object dimension. While this method is fast, versatile and can be extended to handle videos [29], it may fail to discover objects when the training set is diverse, and the resulting representations are highly entangled.

EfficientMORL [11], SLATE [45], SysBinder [46] and BO-QSA [22] are recent developments in the iterative attention approach aimed at addressing some limitations of earlier methods like Slot Attention. EfficientMORL presents an hierarchical variational autoencoder and a lightweight iterative refinement network to increase efficiency without sacrificing representation quality. SLATE increases the non-linear interaction between the slots in Slot Attention with an autoregressive decoder that is conditioned on the slots, resulting in improved reconstructions and object-centric representations. SysBinder enhances the slots of Slot Attention with factor representations called block-slots, which provides within-slot disentanglement between learned factors. BO-QSA initializes the slots of Slot Attention as learnable embeddings instead of sampling from a Gaussian distribution and uses bi-level optimization, resulting in more stable training. Despite the advancements, one drawback remains: the number of clusters are fixed a priori which limits the applicability in real-world scenarios where the number of objects or clusters is not known beforehand.

3 Methodology

Our proposed method OC-Net is designed to extract objects in an image without relying on labeled data or specifying the number of objects present in the image. By not requiring the number of objects to be specified beforehand, OC-Net can generalize better to real-world scenes with varying numbers of objects and handle more complex scenarios.

Figure 1 shows the main components of OC-Net. The main idea behind OC-Net is to learn pixel embeddings that can be clustered to discover objects and their respective object masks and object

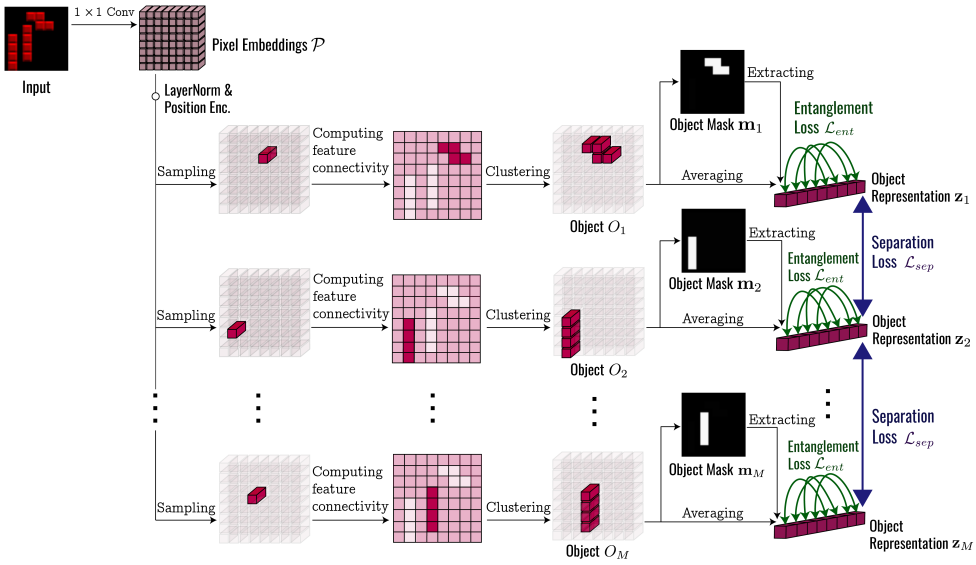


Figure 1: Overview of OC-Net.

representations. This is achieved by passing the input image through a 1×1 convolutional layer to obtain a set of N pixel embeddings $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ of D dimensions each. We leverage on feature connectivity and iteratively cluster the embeddings of neighbouring pixels based on the likelihood that they belong to the same object. The output is a set of objects $\mathcal{O} = \{O_1, \ldots, O_M\}$ where each object is a set of pixel embeddings. We derive the object mask $\mathbf{m}_j$ of each object by setting the pixel corresponding to the embedding in O_j:

$$\mathbf{m}_j[i] = \begin{cases} 1 & \text{if } \mathbf{p}_i \in O_j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $\mathbf{m}_j[i]$ denotes the i^{th} pixel value and $i \in \{1, \ldots, N\}$.

With this, we obtain the matrix of object representations $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M]$ where each $\mathbf{z}_j$ is the sum of extracted mask information and the average of the pixel embeddings in O_j:

$$\mathbf{z}_j[d] = (\mathbf{A} \cdot \mathbf{m}_j)[d] + \frac{1}{|O_j|} \sum_{\mathbf{p}_i \in O_j} \mathbf{p}_i[d] \tag{2}$$

where $\mathbf{z}_j[d]$ denotes the d^{th} value of vector $\mathbf{z}_j$, $d \in \{1, \ldots, D\}$, $\mathbf{A}$ is the mask transformation matrix.

3.1 Object Discovery

The object discovery process iteratively clusters the pixel embeddings based on their feature connectivity and similarity. LayerNorm [2] is applied to normalize all pixel embeddings, and positional encodings are added to the pixel embeddings. The neighbors of a pixel embedding $\mathbf{p}$ are the set of embeddings of the 8 neighbours in the input image. We use Dijkstra's algorithm to compute the shortest distance of a sampled pixel embedding to all other embeddings as follows.

Let $\mathcal{U}$ be the set of pixel embeddings that have not been assigned to an object yet. We uniformly sample a pixel embedding $\mathbf{p}_i \in \mathcal{U}$. The distance from $\mathbf{p}_i$ to itself is set to zero, and the distance to all the other pixels is set to infinity. We select an unvisited pixel embedding $\mathbf{p}_m$ that has the minimum distance to $\mathbf{p}_i$. Let $\mathbf{p}_k$ be the embedding of a neighbour of the pixel corresponding to $\mathbf{p}_m$. We compute the distance between a pair of neighbouring pixels as the similarity between their corresponding embeddings given by:

$$\text{sim}(\mathbf{p}_m, \mathbf{p}_k) = \sqrt{\sum_{d=1}^{D} (\mathbf{p}_m[d] - \mathbf{p}_k[d])^2} \tag{3}$$

where $\mathbf{p}_m[d]$ denotes the d^{th} value of the embedding $\mathbf{p}_m$.

If the distance between $\mathbf{p}_i$ and $\mathbf{p}_k$ is shorter through $\mathbf{p}_m$, we update the shortest distance accordingly. This ensures that we consider the most efficient path between pixel embeddings, leading to better object discovery. We mark $\mathbf{p}_m$ as visited and consider it to be part of the same object as $\mathbf{p}_i$ according to a threshold. The process is repeated for the next unvisited pixel embedding until all pixels have been visited.

3.2 Object-Centric Regularization

We design two object-centric regularization terms $\mathcal{L}_{sep}$ and $\mathcal{L}_{ent}$ to improve the quality of the learned object representations for downstream generalization and object discovery. Given the matrix of object representations $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M]$ corresponding to the object training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{M}$, we quantify downstream generalization performance with the prediction error:

$$\delta_{\mathbf{Z}} = \frac{1}{M} \sum_{i=1}^{M} ||\mathbf{W} \cdot \mathbf{z}_i - \mathbf{y}_i|| \tag{4}$$

where $\mathbf{W}$ is the minimum-norm solution of the downstream predictor.

Since labels $\mathbf{y}_i$ are unknown in the unsupervised setting, $\delta_{\mathbf{Z}}$ cannot be directly minimized. Theorem 3.1 below shows that we can minimize an upper bound of $\delta_{\mathbf{Z}}$ by using the projection matrix $\mathbf{P}_{\mathbf{Z}}$. Proof of the theorem is provided in the supplementary material.

Theorem 3.1. *Let $\mathbf{Y}$ be the matrix of training sample labels and $\mathbf{P}_{\mathbf{Z}}$ be the projection matrix of $\mathbf{Z}$:*

$$\mathbf{P}_{\mathbf{Z}} = \mathbf{I} - \mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z} \tag{5}$$

where $\mathbf{I}$ is the identity matrix, $(.)^{\dagger}$ is the pseudoinverse and $\Sigma_{\mathbf{Z}} = \mathbf{Z}\mathbf{Z}^{\top}$ is the unnormalized covariance matrix of $\mathbf{Z}$. Let $||.||_F$ be the Frobenius norm. Then, the following relation holds:

$$\delta_{\mathbf{Z}} \leq ||\mathbf{P}_{\mathbf{Z}}||_F ||\mathbf{Y}||_F \tag{6}$$

Since $||\mathbf{Y}||_F$ in Equation 6 is unknown but fixed, we can minimize $\delta_{\mathbf{Z}}$ by minimizing $||\mathbf{P}_{\mathbf{Z}}||_F$. From the definition of $\mathbf{P}_{\mathbf{Z}}$ in Equation 5, $||\mathbf{P}_{\mathbf{Z}}||_F$ is minimized when the rank of $\Sigma_{\mathbf{Z}}$ is maximized [44, 3]. We achieve this by maximizing the diagonal entries of $\Sigma_{\mathbf{Z}}$ with a separation term $\mathcal{L}_{sep}$ while simultaneously minimizing its off-diagonal entries with an entanglement term $\mathcal{L}_{ent}$, in effect regularizing $\Sigma_{\mathbf{Z}}$ to be a diagonal matrix with a maximum number of nonzero entries.

Maximizing the diagonal entries of $\Sigma_{\mathbf{Z}}$ via $\mathcal{L}_{sep}$ consequently maximizes the distance between object representations in the latent space. This encourages the model to learn distinct and non-overlapping object representations. Expanding the representation space also ensures that objects with varying features and properties can be accurately represented and distinguished from one another, enhancing both downstream generalization and fine-grained object discovery. We define $\mathcal{L}_{sep}$ as:

$$\mathcal{L}_{sep} = \frac{1}{D} \sum_{d=1}^{D} \max(0, 1 - \sqrt{\sigma_d + \tau}) \tag{7}$$

where σ_d is the variance of the d^{th} dimension across the vectors $\mathbf{z}_1, \cdots, \mathbf{z}_M$ and τ is a small constant to maintain numerical stability.

The entanglement term $\mathcal{L}_{ent}$ minimizes the off-diagonal entries of $\Sigma_{\mathbf{Z}}$ and consequently minimizes the correlation between dimensions in the latent space $\mathbf{Z}$, thereby achieves more disentangled object representations. Such representations are easier to manipulate and analyze, as each dimension captures a distinct object property, such as position, scale, or color. $\mathcal{L}_{ent}$ is defined as follows:

$$\mathcal{L}_{ent} = \frac{1}{D \times (M-1)} \sum_{i \neq j} \Sigma_{\mathbf{Z}}[i, j] \tag{8}$$

4 Performance Study

We conduct experiments to evaluate the performance of OC-Net in terms of quality, sample efficiency and generalizability. We use a diverse range of datasets to demonstrate its effectiveness across various scenarios:

Table 1: Summary of dataset characteristics

Dataset	Type	Ground Truth	Image Size	# Samples
Multi-dSprites	Simulated	Pixel Mask	64×64	1M
Tetrominoes-NM	Simulated	Pixel Mask	35×35	1M
SVHN	Real-World	Bounding Box	Varied	530K
IDRiD	Real-World	Pixel Mask	4288×2848	81
CLEVRTEX	Complex Texture	Pixel Mask	128×128	50K
CLEVRTEX-OOD	Complex Texture	Pixel Mask	128×128	10K
Flowers	Common Object	Pixel Mask	128×128	7K
Birds	Common Object	Pixel Mask	128×128	11K
COCO	Common Object	Pixel Mask	128×128	12K

1. Simulated datasets Multi-dSprites [23] and Tetrominoes-NM. The former consists of multiple oval, heart or square-shaped sprites with some occlusions, while the latter is a subset of the original Tetrominoes dataset [23] where images whose ground truth segmentation requires knowledge of the object shapes are filtered out.
2. Real-world multi-object datasets SVHN [36] and IDRiD [40]. SVHN consists of street view images of house numbers while IDRiD is the Indian Diabetic Retinopathy Image Segmentation Dataset.
3. Complex texture datasets CLEVRTEX [26] and CLEVRTEX-OOD. CLEVRTEX features scenes with diverse shapes, textures and photo-mapped materials while CLEVRTEX-OOD is the CLEVRTEX out-of-distrbution test set with 25 new materials and 4 new shapes.
4. Common object datasets Flowers [37], Birds [48] and COCO [50]. The Flowers dataset features 17 diverse flower classes with large variations viewpoint, scale, illumination and background. Birds is the most widely-used CUB-200-2011 dataset for fine-grained visual categorization. COCO is the variant of the Microsoft Common Objects in Context dataset used for large-scale object segmentation [32].

Table 1 shows the dataset characteristics. Following [35, 11], we use the first 60K samples in Multi-dSprites, Tetrominoes-NM and SVHN for training and hold out the next 320 samples for testing. For IDRiD, we split this dataset into 54 images for training and 27 images for testing. For CLEVRTEX, we use the first 40K samples for training and last 5K samples for testing. For CLEVRTEX-OOD, we use 10K samples for testing. For Flowers, we use the first 6K samples for training and last 1K samples for testing. For Birds, we use the first 10K samples for training and last 1K samples for testing. For COCO, we use the first 10K samples for training and last 2K samples for testing.

We compare OC-Net with SLIC [1], Felzenszwalb [11], Slot Attention [35], EfficientMORL [11], GENESIS-V2 [13], SLATE [45], SysBinder [46] and BO-QSA [22]. We train OC-Net for 1000 iterations with a batch size of 64 using Adam [28] with a learning rate of 1×10^{-3}. We carried out an initial experiment to choose the clustering threshold. The results show that the value can range from 0.2 to 2.0 without affecting the performance of OC-Net. As such, we set the threshold to $\epsilon = 0.7$ so that two pixels will belong to the same object if their normalized feature similarity is more than 50%. If a pixel is assigned to multiple objects, we assign it to the mask of the first object in that list and ignore its membership in other objects. Training on 64-by-64 images from Multi-dSprites on a single V100 GPU with 32GB of RAM takes about 10 minutes.

For all methods, we set the maximum number of foreground objects to 6 and 4 for Multi-dSprites and Tetrominoes respectively. Training is carried out for 300,000 iterations with a batch size of 64, using the Adam optimizer with a base learning rate of 4×10^{-4}. We set the size of the latent space to be $D = 64$ for all models. For SVHN and COCO, the number of objects is set to 6. For IDRiD, the number of objects is set to 20 and we train them for 100,000 iterations. For CLEVRTEX and CLEVRTEX-OOD, the number of objects is set to 11. For Flowers and Birds, the number of objects is set to 2.

We use the Adjusted Rand Index (ARI) to measure the quality of objects discovered [20]. The ARI is a measure of similarity between two data clusterings that takes into account the permutation-invariant nature of the predicted segmentation masks and their corresponding ground-truth masks. We also

use the Dice similarity coefficient, along with the Intersection-over-Union (IoU) between the best matching object masks X and Y as follows:

$$\text{Dice}(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|} \quad ; \quad \text{IoU}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} \tag{9}$$

where X the set of object pixels extracted and Y is the set of annotated object pixels in the ground truth. We compute the mean Dice and the mean IoU scores, denoted as mDice and mIoU respectively, by averaging the individual Dice and IoU scores across all matches. For the complex textures dataset, background discovery is included in the computation of the scores.

4.1 Experiments on Quality of Discovered Objects

We first evaluate the ability of OC-Net to discover objects from images with multiple objects. Table 2(a) shows the average ARI, mDice and mIoU scores based on the discovered foreground objects in the simulated datasets after 3 runs. We observe that OC-Net outperforms all other methods by a large margin in Multi-dSprites, and even achieves perfect score for all Tetrominoes-NM test samples.

Table 2(b) shows the results on the real-world multi-object image datasets. For SVHN, OC-Net outperforms all methods, even when the ground truth is provided in the form of bounding boxes. This implies that we need to expand the discovered object masks into their corresponding bounding boxes which are often rough fits, and is the reason for the close difference in mDice and mIoU scores between OC-Net and BO-QSA. For IDRiD, which contains multiple small objects, OC-Net significantly improves the ARI scores and more than doubles the mDice and mIoU scores over all methods, demonstrating its robustness in challenging object discovery tasks.

Table 2(c) shows the results on CLEVRTEX and CLEVRTEX-OOD, which contains complex textured objects and backgrounds. Here, OC-Net again shows superior performance in all metrics, illustrating its capability to effectively segment complex objects. Although a general decrease in performance is observed across all methods in the CLEVRTEX-OOD test set, likely due to a change in data distribution, OC-Net's performance drop is slight and it still outperforms the closest baseline.

Finally, Table 2(d) shows the results on Flowers, Birds and COCO common object datasets. Here, OC-Net again shows superior performance in all metrics, illustrating its capability to effectively segment commonly seen natural objects. Notably, OC-Net outperforms all other methods by a large margin in COCO, demonstrating its robustness in handling objects with highly varied appearances.

Figure 2 visualizes the objects discovered by the various methods for sample images. OC-Net is able to identify large and small objects in Multi-dSprites even when these objects are significantly occluded. Moreover, in the Tetrominoes-NM dataset, despite the presence of shadow effects that often confuse existing methods, OC-Net still manages to separate each tile. For SVHN, only OC-Net is able to segment the character objects out in a fine-grained manner. EfficientMORL tend to group all the characters together while the other methods segment the objects in a coarse-grained manner. For IDRiD, OC-Net is able to segment out the optic disc and small lesions which other methods fail to discover. For CLEVRTEX and CLEVRTEX-OOD, OC-Net is able to segment out the various objects from the complex-textured backgrounds in a fine-grained manner. Finally, for Flowers, Birds, and COCO, only OC-Net is able to segment out the complex-shaped and multi-part objects from the backgrounds in a fine-grained manner.

4.2 Experiments on Sample Efficiency

One obstacle to unsupervised object discovery is the availability of a sufficiently large number of suitable training samples. Sample efficiency refers to a model's ability to learn effectively from a relatively small number of examples. Figure 3 shows the mIoU scores as we decrease the number of training samples in Multi-dSprites, SVHN and CLEVRTEX. OC-Net is able to achieve near-optimal performance even with a significantly smaller training set (1,000 samples) compared to all the other methods. The high sample efficiency of OC-Net reduces the need for large, potentially costly or difficult-to-obtain datasets. This makes OC-Net a more practical solution for real-world applications.

Table 2: Evaluation scores for the discovered foreground objects.

(a) Simulated datasets

Method	Multi-dSprites			Tetrominoes-NM		
	ARI	mDice	mIoU	ARI	mDice	mIoU
SLIC	67.9±0.0	78.5±0.0	70.8±0.0	53.0±0.0	66.1±0.0	53.6±0.0
Felzenszwalb	97.4±0.0	98.6±0.0	95.0±0.0	95.0±0.0	98.0±0.0	96.9±0.0
Slot Attention	91.3±0.3	45.7±0.7	32.6±0.6	99.8±0.1	41.5±0.8	26.6±0.7
EfficientMORL	85.2±0.5	30.1±1.3	19.5±1.1	99.0±1.7	42.5±2.3	27.6±1.9
GENESIS-V2	85.0±1.3	81.5±1.9	72.2±1.4	97.6±0.5	47.1±1.1	31.0±0.8
SLATE	89.5±1.2	82.5±0.9	72.6±1.1	84.5±1.5	57.8±0.9	44.3±0.8
SysBinder	72.3±1.2	30.6±1.1	19.6±1.0	90.7±1.7	41.8±1.9	27.0±1.7
BO-QSA	90.4±1.1	91.6±1.1	88.0±1.2	99.3±0.3	40.9±1.4	25.8±1.2
OC-Net	**99.8±0.0**	**99.5±0.0**	**99.1±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**

(b) Real-world datasets

Method	SVHN			IDRiD		
	ARI	mDice	mIoU	ARI	mDice	mIoU
SLIC	5.3±0.0	50.1±0.0	34.5±0.0	32.2±0.0	12.7±0.0	8.8±0.0
Felzenszwalb	31.7±0.0	51.6±0.0	39.8±0.0	14.7±0.0	19.0±0.0	15.4±0.0
Slot Attention	38.9±1.5	51.7±1.8	36.7±1.7	28.7±1.1	8.6±1.7	5.0±1.6
EfficientMORL	32.2±1.7	49.2±2.0	34.0±1.8	16.8±1.5	11.1±2.7	7.0±1.8
GENESIS-V2	28.6±1.4	60.8±1.5	45.9±1.4	18.3±1.6	8.8±1.9	5.4±1.6
SLATE	21.2±1.2	57.0±1.3	41.7±1.5	35.6±2.1	8.1±1.2	4.7±1.8
SysBinder	15.8±1.6	49.5±1.9	34.1±1.8	25.2±1.3	16.6±1.7	11.1±1.8
BO-QSA	24.3±1.2	62.0±1.6	48.3±1.3	27.7±2.0	7.0±1.9	4.5±1.7
OC-Net	**39.7±0.1**	**64.6±0.1**	**49.9±0.1**	**39.0±0.4**	**38.1±0.2**	**31.2±0.2**

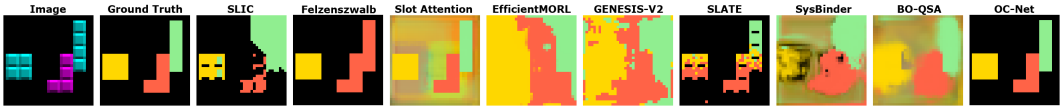
(c) Complex textures dataset

Method	CLEVRTEX			CLEVRTEX-OOD		
	ARI	mDice	mIoU	ARI	mDice	mIoU
SLIC	27.4±0.0	20.0±0.0	13.0±0.0	25.8±0.0	21.7±0.0	14.0±0.0
Felzenszwalb	57.3±0.0	33.6±0.0	26.8±0.0	44.6±0.0	29.6±0.0	23.4±0.0
Slot Attention	58.6±1.6	35.0±1.6	26.7±1.5	51.3±1.9	34.1±1.4	25.1±1.3
EfficientMORL	59.5±1.7	37.7±1.5	31.1±1.4	53.9±2.5	32.2±2.4	25.3±2.8
GENESIS-V2	65.6±1.8	36.9±1.4	30.4±1.4	67.6±1.6	34.2±1.5	27.6±1.9
SLATE	57.5±1.8	33.3±1.6	24.4±1.5	56.6±1.3	34.7±2.1	25.3±1.8
SysBinder	61.4±1.7	31.3±1.5	23.1±1.4	61.0±2.4	32.3±2.0	23.8±1.8
BO-QSA	**70.9±1.9**	42.9±1.8	34.7±1.7	66.1±1.3	42.8±1.4	33.9±1.3
OC-Net	**70.7±0.9**	**45.1±0.9**	**37.5±0.7**	**69.8±0.8**	**43.5±0.7**	**35.0±0.6**

(d) Common objects datasets

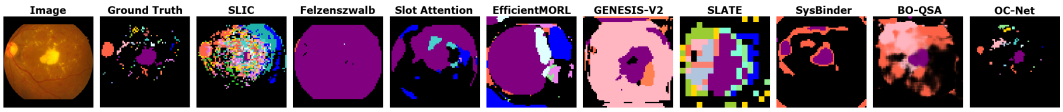
Method	Flowers		Birds		COCO	
	Dice	IoU	Dice	IoU	mDice	mIoU
SLIC	30.5±0.0	18.4±0.0	33.1±0.0	20.3±0.0	36.2±0.0	24.4±0.0
Felzenszwalb	43.7±0.0	30.4±0.0	34.3±0.0	23.0±0.0	36.6±0.0	27.1±0.0
Slot Attention	43.0±1.5	28.6±1.2	42.9±2.0	27.9±1.8	24.8±2.0	15.0±1.7
EfficientMORL	59.5±2.1	45.2±2.2	44.0±1.9	30.8±1.8	28.4±2.3	18.9±2.1
GENESIS-V2	63.7±2.2	50.2±2.2	41.4±1.7	27.7±1.5	25.1±2.1	16.1±1.7
SLATE	55.6±1.2	40.8±1.8	39.5±1.5	25.9±1.8	37.0±1.9	24.4±1.8
SysBinder	45.0±1.8	30.8±1.6	33.7±1.3	21.1±2.0	18.4±1.6	10.7±1.4
BO-QSA	65.8±1.9	51.7±1.9	44.6±1.7	30.3±1.5	34.9±1.1	23.6±0.9
OC-Net	**67.2±0.2**	**54.4±0.2**	**47.8±0.2**	**33.5±0.2**	**48.2±0.2**	**35.6±0.2**

(a) Multi-dSprites



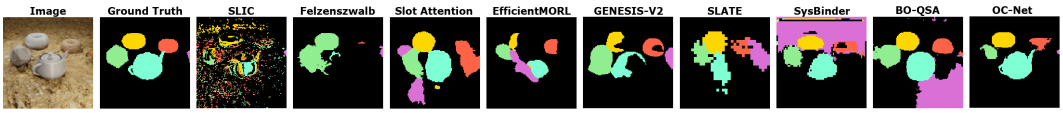
(b) Tetrominoes-NM

(c) SVHN

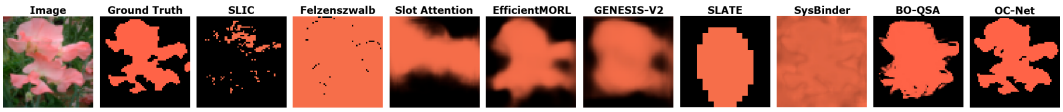


(d) IDRiD

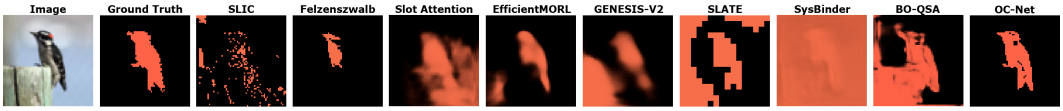
(e) CLEVRTEX



(f) CLEVRTEX-OOD



(g) Flowers



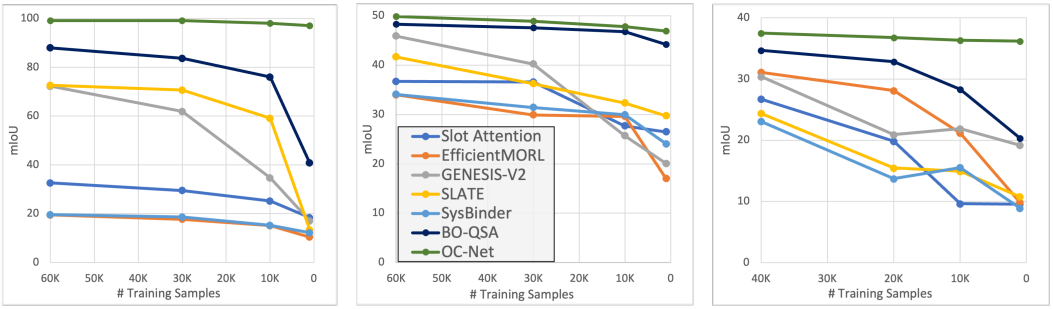
(h) Birds

(i) COCO

Figure 2: Visualization of discovered objects.

4.3 Experiments on Model Generalizability

Ideally, an unsupervised object discovery model should be trained to understand common visual appearances such as the difference between foreground vs background, so as to discover objects from



(a) Multi-dSprites (b) SVHN (c) CLEVRTEX

Figure 3: mIoU scores vs decreasing number of training samples.

Table 3: mIoU scores for model generalizability after training on Multi-dSprites.

Method	Tetrominoes-NM	SVHN	IDRiD	CLEVRTEX	CLEVRTEX-OOD
Slot Attention	21.8±3.5	19.5±3.8	7.5±2.5	12.2±2.2	12.3±2.2
EfficientMORL	21.2±3.8	23.4±2.8	6.5±2.6	12.7±3.2	15.2±2.0
GENESIS-V2	42.9±4.9	31.1±2.8	8.5±2.4	21.9±1.6	21.3±2.5
SLATE	51.4±1.6	21.1±2.0	10.0±1.7	12.7±2.2	12.9±1.8
SysBinder	28.5±1.8	23.8±1.1	13.9±1.8	10.6±1.5	11.6±1.9
BO-QSA	41.8±1.8	24.3±2.0	4.0±1.5	24.4±1.4	22.8±2.5
OC-Net	**100.0±0.0**	**47.5±0.5**	**29.1±0.5**	**31.7±0.6**	**31.3±0.6**

out-of-domain images. In this set of experiments, we compare the generalization ability of OC-Net with the baselines by training the models on Multi-dSprites and testing them on the other datasets.

Table 3 shows the results. For Tetrominoes-NM, there is a decrease in performance for all methods while OC-Net still obtains perfect score. For SVHN, CLEVRTEX and CLEVRTEX-OOD, the performance of all models decrease due to the shift in data distribution. However, OC-Net experiences the smallest drop in performance and still significantly surpasses the best performing method. For IDRiD, the methods show improvement in performance. One possible reason is that the larger training set in Multi-dSprites enables the circular shape of the optic disc to be better segmented. Despite this, OC-Net remains the top performer.

4.4 Ablation Studies

Next, we examine the effect of feature connectivity and regularization terms $\mathcal{L}_{sep}$ and $\mathcal{L}_{ent}$ on the performance of OC-Net. We implemented three variants of OC-Net: (a) w/o connectivity. Here, we do not require that a path should exist between i and k when computing dist$[i, k]$; (b) w/o $\mathcal{L}_{ent}$. This network is trained without the entanglement regularization term; (c) w/o $\mathcal{L}_{sep}$. The separation regularization term is not used in the training of OC-Net.

Table 4 shows the mIoU scores for all the datasets. Without feature connectivity, we observe a drop in performance across all datasets since it is common for images to have multiple identical objects, that is, same color and shape. As such, OC-Net w/o connectivity tend to cluster these blocks as a single object.

Removing the entanglement term (OC-Net w/o $\mathcal{L}_{ent}$) leads to a slight decrease in the performance as the object representations may still be separated even when the dimensions are entangled. The largest performance drop in all datasets is seen when the object representation separation term is removed (OC-Net w/o $\mathcal{L}_{sep}$), indicating the importance of having a well-separated object representation space for effective object discovery.

Table 4: mIoU scores for variants of OC-Net.

Method	Multi-dSprites	Tetro-NM	SVHN	IDRiD	CTEX	CTEX-OOD
OC-Net	**99.1±0.0**	**100.0±0.0**	**49.9±0.1**	**31.2±0.2**	**37.5±0.7**	**35.0±0.6**
w/o connectivity	98.8±0.1	89.2±0.1	36.6±0.1	17.3±0.1	20.3±0.9	17.7±0.9
w/o $\mathcal{L}_{ent}$	98.7±0.3	99.6±0.1	48.8±0.1	25.8±0.2	26.4±0.8	18.9±0.4
w/o $\mathcal{L}_{sep}$	49.3±0.3	51.2±0.2	35.0±0.2	4.0±0.1	18.5±0.6	11.0±0.4

5 Prediction based on Learned Object Representation

One characteristic of an effective object-centric representation is its ability to encode object properties such as color, position and shape [42]. In this section, we show that the learned object representations are disentangled and can be used to predict the values of these properties.

Given the representations and their corresponding ground truth values of a target object property, we employ them as features to train a gradient boosted tree (GBT) [16]. To evaluate how well the properties of unseen objects are predicted by the GBT, we use the coefficient of determination R^2 [49]. We perform an experiment using the learned object representations from the simulated datasets to predict the properties of object such as color, position and shape. We use the mIoU score to match the discovered object to the ground truth object. We split the 320 test images equally into two sets, one for training the GBT model for each object property, and the other for evaluation.

Table 5 shows the average R^2 scores of the GBT models on the evaluation set. The GBT models trained with OC-Net representations achieved the highest R^2 scores compared to the models trained using the representations from other methods. This suggests that the object representations learned by OC-Net are effective in encoding the object properties.

Table 5: R^2 scores for object property prediction on simulated datasets

	Multi-dSprites			Tetrominoes-NM		
Method	Color	Position	Shape	Color	Position	Shape
Slot Attention	72.2±12	96.8±0.1	38.2±0.0	86.5±6.5	98.7±0.6	36.3±0.0
EfficientMORL	86.5±6.2	95.8±0.1	61.7±0.0	94.9±3.2	97.9±0.7	68.5±0.0
GENESIS-V2	78.1±7.5	97.1±0.7	75.8±0.0	88.1±5.8	94.6±2.6	37.9±0.0
SLATE	87.5±0.7	90.6±4.4	31.7±0.0	85.5±3.9	89.6±0.7	10.5±0.0
SysBinder	73.6±1.0	69.3±3.4	33.3±0.0	97.9±0.6	77.8±2.7	19.9±0.0
BO-QSA	96.3±1.6	97.4±0.1	75.2±0.0	98.1±0.7	98.9±0.2	52.5±0.0
OC-Net	**98.0±0.6**	**98.3±0.1**	**78.1±0.0**	**100.0±0.0**	**99.4±0.1**	**98.7±0.0**

6 Conclusion

In this work, we have described a framework called OC-Net that learns object-centric representations in a fine-grained manner without supervision. OC-Net leverages on feature connectivity and two new regularization terms to learn disentangled representations and to ensure the representations of different objects are well-separated. From the results of experiments conducted on simulated, real-world, complex texture and common object images, we have demonstrated the superior quality of the object representations over current state-of-the-art. Moreover, we have highlighted the sample efficiency and generalizability of OC-Net. Finally, we have shown how the discovered object representations can be used to predict object properties in a downstream task, indicating its potential use for other computer vision applications where samples and ground truth labels are limited.

There are still obstacles that have to be overcome for successful application of our framework to the full visual complexity of the real world. A natural next step would be to extend OC-Net to handle real-world scenes containing objects with more complex part-whole hierarchies. It is also promising to explore explicit representation of the discovered objects into a dictionary of prototypes to better handle occlusion between objects. Lastly, real-world scenes with multiple objects is still of higher visual complexity than the datasets considered here and reliably bridging this gap is an open problem.

Acknowledgments and Disclosure of Funding

This research is supported by the National Research Foundation Singapore under its AI Singapore Programme (Award Number: AISG-GC-2019-001-2A).

References

[1] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. doi: 10.1109/TPAMI.2012.120.

[2] Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *Advances in NIPS 2016 Deep Learning Symposium*, 2016.

[3] Bardes, A., Ponce, J., and LeCun, Y. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[4] Bear, D., Fan, C., Mrowca, D., Li, Y., Alter, S., Nayebi, A., Schwartz, J., Fei-Fei, L. F., Wu, J., Tenenbaum, J., et al. Learning physical graph representations from visual scenes. *Advances in Neural Information Processing Systems*, 33:6027–6039, 2020.

[5] Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M. M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *CoRR*, 1901.11390, 2019.

[6] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

[7] Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

[8] Crawford, E. and Pineau, J. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3412–3420, 2019.

[9] Dittadi, A., Papa, S. S., Vita, M. D., Schölkopf, B., Winther, O., and Locatello, F. Generalization and robustness implications in object-centric learning. In *International Conference on Machine Learning, ICML*, 2022.

[10] Eastwood, C. and Williams, C. K. I. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

[11] Emami, P., He, P., Ranka, S., and Rangarajan, A. Efficient iterative amortized inference for learning symmetric and disentangled multi-object representations. In *International Conference on Machine Learning*, pp. 2970–2981, 2021.

[12] Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. GENESIS: generative scene inference and sampling with object-centric latent representations. In *8th International Conference on Learning Representations, ICLR 2020*, 2020.

[13] Engelcke, M., Parker Jones, O., and Posner, I. Genesis-v2: Inferring unordered object representations without iterative refinement. *Advances in Neural Information Processing Systems*, 34: 8085–8094, 2021.

[14] Eslami, S., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Hinton, G. E., et al. Attend, infer, repeat: Fast scene understanding with generative models. *Advances in Neural Information Processing Systems*, 29, 2016.

[15] Felzenszwalb, P. F. and Huttenlocher, D. P. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004.

[16] Friedman, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

[17] Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433, 2019.

[18] Greff, K., Van Steenkiste, S., and Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

[19] He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[20] Hubert, L. and Arabie, P. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[21] Jaderberg, M., Simonyan, K., Zisserman, A., et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[22] Jia, B., Liu, Y., and Huang, S. Improving object-centric learning with query optimization. In *The Eleventh International Conference on Learning Representations*, 2023.

[23] Kabra, R., Burgess, C., Matthey, L., Kaufman, R. L., Greff, K., Reynolds, M., and Lerchner, A. Multi-object datasets. https://github.com/deepmind/multi-object-datasets/, 2019.

[24] Kahneman, D., Treisman, A., and Gibbs, B. J. The reviewing of object files: Object-specific integration of information. *Cognitive psychology*, 24(2):175–219, 1992.

[25] Karazija, L., Laina, I., and Rupprecht, C. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In *Proceedings of the Neural Information Processing Systems*, 2021.

[26] Karazija, L., Laina, I., and Rupprecht, C. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In Vanschoren, J. and Yeung, S. (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Curran, 2021.

[27] Kim, J., Choi, J., Choi, H.-J., and Kim, S. J. Shepherding slots to objects: Towards stable and robust object-centric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19198–19207, 2023.

[28] Kingma Diederik, P. and Adam, J. B. A method for stochastic optimization. *International Conference on Learning Representations ICLR*, 2015.

[29] Kipf, T., Elsayed, G. F., Mahendran, A., Stone, A., Sabour, S., Heigold, G., Jonschkowski, R., Dosovitskiy, A., and Greff, K. Conditional Object-Centric Learning from Video. In *International Conference on Learning Representations (ICLR)*, 2022.

[30] Kipf, T. N., van der Pol, E., and Welling, M. Contrastive learning of structured world models. In *8th International Conference on Learning Representations, ICLR*, 2020.

[31] Kosiorek, A., Kim, H., Teh, Y. W., and Posner, I. Sequential attend, infer, repeat: Generative modelling of moving objects. *Advances in Neural Information Processing Systems*, 31, 2018.

[32] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

[33] Lin, Z., Wu, Y., Peri, S. V., Sun, W., Singh, G., Deng, F., Jiang, J., and Ahn, S. SPACE: unsupervised object-oriented scene representation via spatial attention and decomposition. In *8th International Conference on Learning Representations, ICLR 2020*, 2020.

[34] Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

[35] Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

[36] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[37] Nilsback, M.-E. and Zisserman, A. Delving deeper into the whorl of flower segmentation. *Image and Vision Computing*, 28(6):1049–1062, 2010.

[38] Palmer, S. and Rock, I. Rethinking perceptual organization: The role of uniform connectedness. *Psychonomic bulletin & review*, 1(1):29–55, 1994.

[39] Paszke, A. e. a. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 2019.

[40] Porwal, P., Pachade, S., Kamble, R., Kokare, M., Deshmukh, G., Sahasrabuddhe, V., and Meriaudeau, F. Indian diabetic retinopathy image dataset (idrid), 2018.

[41] Rezende, D. J. and Viola, F. Taming vaes. *ArXiv*, abs/1810.00597, 2018.

[42] Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021. doi: 10.1109/JPROC.2021.3058954.

[43] Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[44] Shwartz-Ziv, R., Balestriero, R., Kawaguchi, K., Rudner, T. G., and LeCun, Y. An information-theoretic perspective on variance-invariance-covariance regularization. *arXiv preprint arXiv:2303.00633*, 2023.

[45] Singh, G., Deng, F., and Ahn, S. Illiterate dall-e learns to compose. In *International Conference on Learning Representations*, 2021.

[46] Singh, G., Kim, Y., and Ahn, S. Neural systematic binder. In *The Eleventh International Conference on Learning Representations*, 2023.

[47] Wang, X., Girdhar, R., Yu, S. X., and Misra, I. Cut and learn for unsupervised object detection and instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3124–3134, 2023.

[48] Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-ucsd birds 200. 2010.

[49] Wright, S. Correlation and causation. *Journal of agricultural research*, 1921.

[50] Yang, Y. and Yang, B. Promising or elusive? unsupervised object segmentation from real-world single images. *Advances in Neural Information Processing Systems*, 35:4722–4735, 2022.

[51] Yuan, J., Chen, T., Li, B., and Xue, X. Compositional scene representation learning via reconstruction: A survey. *arXiv preprint arXiv:2202.07135*, 2022.

A Additional Implementation Details

A.1 Object Discovery Algorithm

We further describe the details of the object discovery process (see Algorithm 1). The input is the set of pixel embeddings for an image. We apply LayerNorm [2] to normalize all pixel embeddings and add positional encodings to the pixel embeddings (Line 3). We use Dijkstra's algorithm to compute the shortest path distance of a sampled pixel embedding to all other embeddings as follows. We uniformly sample an unprocessed pixel embedding $\mathbf{p}_i \in \mathcal{U}$ that has not been assigned to an object (Line 6). We initialize the distance from $\mathbf{p}_i$ to itself as zero and to all the other pixels as infinity (Lines 7-10). Then we select an unvisited pixel embedding $\mathbf{p}_m$ that has the minimum distance to $\mathbf{p}_i$ (Line 13). For each neighbour of the pixel corresponding to $\mathbf{p}_m$, let $\mathbf{p}_k$ be its embedding (Lines 14-15). Here, we consider the neighbours of a pixel to be the 8 surrounding pixels and the distance between a pair of neighbouring pixels is the similarity measure between their corresponding embeddings given by:

$$\text{sim}(\mathbf{p}_m, \mathbf{p}_k) = \sqrt{\sum_{d=1}^{D} (\mathbf{p}_m[d] - \mathbf{p}_k[d])^2}, \tag{10}$$

where $\mathbf{p}_m[d]$ denotes the d^{th} value of the embedding $\mathbf{p}_m$.

We update the shortest path distance between $\mathbf{p}_i$ and $\mathbf{p}_k$ if the distance is shorter through $\mathbf{p}_m$ (Lines 16-18). We mark $\mathbf{p}_m$ as visited (Line 20) and consider $\mathbf{p}_m$ to be part of the same object as $\mathbf{p}_i$ (Lines 21-23). We repeat the process for the next unvisited pixel embedding until all pixels have been visited (Line 24). The discovered object O_c is added to the set of objects found thus far $\mathcal{O}$ (Line 25). The entire process is repeated till all pixel embeddings have been assigned to some object (line 26).

Algorithm 1 Object Discovery Process.

1: **Input:** Set of pixel embeddings $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$
2: **Output:** Set of objects $\mathcal{O} = \{O_1, \ldots, O_M\}$
3: $\mathcal{P} \leftarrow \text{LayerNorm}(\mathcal{P}) + \text{Position Enc.}$
4: $\mathcal{U} \leftarrow \mathcal{P}$; $\mathcal{O} \leftarrow \emptyset$; $c \leftarrow 1$ // Initialization
5: **repeat**
6: $\mathbf{p}_i \leftarrow \text{UniformSampling}(\mathcal{U})$
7: dist[i,i] $\leftarrow 0$
8: **for** $j = 1$ to N, $j \neq i$ **do**
9: dist[i, j] $\leftarrow \infty$
10: **end for**
11: $O_c \leftarrow \emptyset$; $\mathcal{V} \leftarrow \emptyset$ // Initialization
12: **repeat**
13: let $\mathbf{p}_m \in \mathcal{P} - \mathcal{V}$ be the pixel embedding with minimum distance to $\mathbf{p}_i$
14: **for** each neighbor of the pixel corresponding to the embedding $\mathbf{p}_m$ **do**
15: let $\mathbf{p}_k$ of be the embedding of the neighbour
16: **if** dist[i, k] > dist[i, m] + sim($\mathbf{p}_m, \mathbf{p}_k$) **then**
17: dist[i, k] = dist[i, m] + sim($\mathbf{p}_m, \mathbf{p}_k$)
18: **end if**
19: **end for**
20: $\mathcal{V} \leftarrow \mathcal{V} \cup \{\mathbf{p}_m\}$
21: **if** $dist[i, m] < \epsilon$ **then**
22: $O_c \leftarrow O_c \cup \{\mathbf{p}_m\}$; $\mathcal{U} \leftarrow \mathcal{U} - \{\mathbf{p}_m\}$
23: **end if**
24: **until** $|\mathcal{V}| == N$
25: $\mathcal{O} \leftarrow \mathcal{O} \cup \{O_c\}$; $c \leftarrow c + 1$
26: **until** $|\mathcal{U}| == \emptyset$

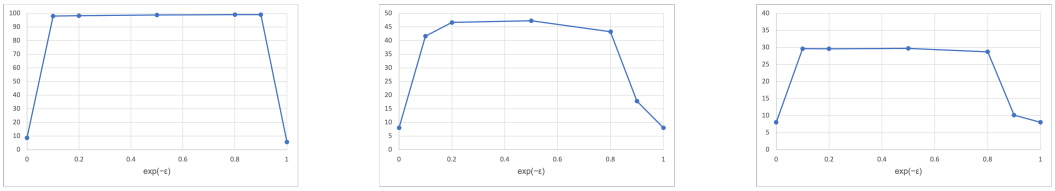
A.2 Positional Encodings

We add fixed positional encondings to the last two pixel embedding dimensions, where the encondings have values from 0 to 1 according to the pixel embedding's relative location to the top and left of the

image. The top-left pixel has positional encondings of values $(0, 0)$, while the bottom-right pixel has values $(1, 1)$. All other postional encondings have uniformly spaced values between $[0, 1]$ according to the height and width of the image.

A.3 Clustering Threshold

We carried out initial experiments to choose the clustering threshold. Figure 4 shows the mIoU scores on Multi-dSprites, SVHN and CLEVRTEX when we vary the negative exponent of the threshold value. We see that the value can range from 20% ($\epsilon = 1.6$ to $\epsilon = 2.0$) to 80% (about $\epsilon = 0.2$) without affecting the performance of OC-Net. As such, we set the threshold to $\epsilon = 0.7$ so that two pixels will belong to the same object if their normalized feature similarity is more than 50%.



(a) Multi-dSprites (b) SVHN (c) CLEVRTEX

Figure 4: mIoU scores with different values of clustering threshold.

A.4 Mask Information Extraction

We define the object representation as the sum of the extracted mask information and the average of the pixel embeddings in $\mathcal{O}_j$. We extract mask information by transforming the object mask $\mathbf{m}_j$ by a mask transformation matrix $\mathbf{A}$. This extraction is performed in a positionally-invariant manner by initializing the rows of $\mathbf{A}$ with a set of unit vectors which point from the object center to the border pixels. After initialization, $\mathbf{A}$ is refined via gradient descent.

Based on initial experiments shown in Table 6, when compared to random initialization (OC-Net w/ random $\mathbf{A}$ init), we see that initializing $\mathbf{A}$ with unit vectors enhances the learned representation especially in terms of object shape information.

Table 6: R^2 scores for object property prediction on simulated datasets

Method	Multi-dSprites			Tetrominoes-NM		
	Color	Position	Shape	Color	Position	Shape
OC-Net w/ random $\mathbf{A}$ init	97.7±0.8	98.0±0.1	77.1±0.0	100.0±0.0	99.1±0.2	89.6±0.0
OC-Net	**98.0±0.6**	**98.3±0.1**	**78.1±0.0**	**100.0±0.0**	**99.4±0.1**	**98.7±0.0**

A.5 Computational Resources

An 8X Tesla V100 (32GB) GPU server is used to train OC-Net and all comparison baselines for our experiments.

B Datasets

1. Multi-dSprites [23] (Apache-2.0 License): This simulated dataset consists of sprites-based images of 64×64 size with ground truth segmentation masks and sprite properties available at https://github.com/deepmind/multi_object_datasets. Each image consists of multiple oval, heart or square-shaped sprites with some occlusions. We use the variant where 2-5 randomly colored sprites appear on a randomly sampled grayscale background.

2. Tetrominoes-NM [23] (Apache-2.0 License): This dataset is a subset of the original simulated Tetrominoes dataset where each 35×35 image consists of 3 Tetris-like shapes sampled from 6 colors and 17 shapes. We download the data from https://github.com/deepmind/multi_object_datasets. Ground truth segmentation masks and properties are provided. We filter out images whose ground truth segmentation requires knowledge of the object shapes for testing.

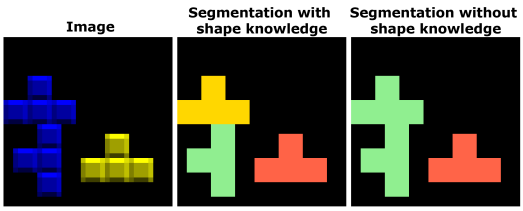


Figure 5: Images in Tetrominoes dataset that require knowledge of object shapes for segmentation

3. Street View House Numbers (SVHN) [36] (CC0 Public Domain): This real-world dataset consists of house numbers in Google Street View images with character level ground truth bounding boxes. We use the dataset labeled as 'extra' [36], available at http://ufldl.stanford.edu/housenumbers. All original images and their ground truths are resized and cropped to size 64×64. The dimensions of the raw images in this set vary, hence we resize all heights of the image to 64 pixels first before cropping the width to 64. We modify the ground truth bounding boxes according to the same transformations. Since bounding boxes are provided as ground truths, we expand the object masks predicted by each method into their best-fit bounding boxes according to the masks' convex hull before computing the ARI, mDice and mIoU scores.
4. Indian Diabetic Retinopathy Image Segmentation Dataset (IDRiD) [40] (CC-BY 4.0): This is a publicly available real-world dataset with 81 retinal images where each image retina image has multiple red lesions (microaneurysms and hemorrhages) or yellow lesions (hard exudates and soft exudates). We obtain the 81 images with fine-grained segmentation ground truths from https://ieee-dataport.org/open-access/indian-diabetic-retinopathy-image-dataset-idrid. Each image has a manually labelled segmentation mask of the lesions and also the optic disc. We filter out the red lesions and resize all raw images and their ground truths to size 64×64. This dataset contains retina images which have more severe diabetic retinopathy, hence the number of possible lesions to be detected is large. We filter out the relatively smaller red lesions and focus the object discovery task on the optic disc and yellow lesions. We cluster the ground truth pixel masks into connected regions as the different objects to be discovered by the models.
5. CLEVRTEX [26] (CC-BY 4.0): This dataset features scenes with diverse shapes, textures and photo-mapped materials, created using physically based rendering techniques. The data is available at https://www.robots.ox.ac.uk/ vgg/data/clevrtex/. Each image contains 3-10 objects of 4 possible shapes randomly arranged on a background. The objects and backgrounds can take materials from a total of 60 possible materials. The scenes contain realistic reflections, highlights, shadows and lighting effects.
6. CLEVRTEX-OOD [26] (CC-BY 4.0): For further evaluation, we use the CLEVRTEX-OOD (out-of-distribution) test set containing 10K images with 25 new (unseen) materials and 4 new shapes (cone, torus, icosahedron, and a teapot) that are not part of CLEVRTEX. The data is available at https://www.robots.ox.ac.uk/ vgg/data/clevrtex/.
7. Flowers [37] (CC0 Public Domain): The dataset has 17 flower classes (e.g. buttercup, daffodil, iris, pansy), with photographs exhibiting typical (large) variations in viewpoint, scale, illumination and background. Segmenting such photographs is challenging due to both the variety of colours and the variety of shapes. We download the data from https://www.robots.ox.ac.uk/ vgg/data/flowers/.
8. Birds [48] (CC0 Public Domain): The Caltech-UCSD Birds-200-2011 (CUB-200-2011) dataset is the most widely-used dataset for fine-grained visual categorization task. It contains 11,788 images of 200 subcategories belonging to birds. Each image has detailed

annotations: 1 subcategory label, 15 part locations, 312 binary attributes and 1 bounding box http://www.vision.caltech.edu/datasets/cub_200_2011/.

9. COCO [32] (CC-BY 4.0): This is the Microsoft Common Objects in Context dataset used for large-scale object segmentation. We use the variant that focuses on multi-object segmentation as proposed by [50], which we download from this link https://www.dropbox.com/sh/u1p1d6hysjxqauy/AACgEh0K5ANipuIeDnmaC5mQa?dl=0

C Baseline Models

1. SLIC [1] is a clustering algorithm that clusters pixels into superpixels by using an efficient adaptation of the k-means algorithm. We use the python implementation and for each dataset, we perform a grid search for the optimal hyperparameters which produce the best results. For example, for the Tetrominoes dataset, we set the optimal clustering threshold value as 10 and the initial number of clusters as 12.
2. Felzenszwalb's Algorithm [11] is a graph-based segmentation algorithm that groups pixels together through a hand-crafted boundary detection procedure. We use the python implementation and for each dataset, we perform a grid search for the optimal hyperparameters which produce the best results. For example, for the Tetrominoes dataset, we set the optimal clustering threshold 1000, the minimum cluster size as 10, and the image smoothening value as 0.1.
3. Slot Attention [35] initializes a set of random object representations called slots which are iteratively refined by slot-normalized cross-attention on the outputs of a simple convolutional neural network (CNN). The slots are then decoded individually and combined to reconstruct the input. All Slot Attention baselines are trained with 500,000 iterations. We use the default training hyperparameters from the official reference implementation.
4. EfficientMORL [11] uses a hierarchical variational auto-encoder to extract disentangled object representations and refine the representations by a lightweight network before reconstructing the input. For all real-world and complex textures datasets, we fine-tune the per-pixel GECO reconstruction target to -2.206 which significantly outperformed the suggested settings in both the original paper and CLEVRTEX experiments[26].
5. GENESIS-V2 [13] obtains pixel embeddings through a U-Net which are then clustered using a stochastic stick-breaking process. The clusters are then decoded to reconstruct the input. We similarly fine-tune the model to use the output standard deviation of 0.7 and the equivalent per-pixel GECO reconstruction target as EfficientMORL and achieved much higher results than those reported in the CLEVRTEX paper [26].
6. SLATE [45] replaces the decoder in Slot Attention with a more expressive transformer-based autoregressive decoder conditioned on the slots. The original model uses patch sizes of 4×4 pixels as inputs and obtains poor scores for the simulated Multi-dSprites and Tetrominoes datasets which require fine-grained understanding of each pixel. Hence, we extend the model to have 1×1 input for these datasets.
7. SysBinder [46] enhances the slots of Slot Attention with factor representations called block-slots which provides within-slot disentanglement. Similar to SLATE, the original model uses patch sizes of 4×4 pixels as inputs and obtains poor scores for the simulated datasets. Hence, we extend the model to have 1×1 input for these datasets.
8. BO-QSA [22] initializes Slot Attention's object representations as learnable embeddings instead of sampling from a learnable Gaussian distribution and supplements the training with bi-level optimization. All BO-QSA baselines are trained with 500,000 iterations. We use the default training hyperparameters from the official reference implementation. We train all datasets with both the mixture-based decoder and autoregressive transformer-based decoder and report the highest scores.

D Upper Bound for Downstream Generalization Error

In this section, we present the proof of Theorem 3.1.

Proof. Let the matrix of object representations be $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M] \in \mathbb{R}^{D \times M}$ corresponding to the object training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^M$, and let the matrix of unknown labels be $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_M]^\top \in \mathbb{R}^{M \times R}$.

Recall that $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_R]^\top \in \mathbb{R}^{R \times D}$ in Equation 4 is the minimum norm solution of the downstream predictor:

$$\mathbf{W} = \underset{\mathbf{W}'}{\text{minimize}} \, ||\mathbf{W}'||_F \text{ s.t. } \mathbf{W}' \in \arg\min_{\hat{\mathbf{W}}} \frac{1}{M} \sum_{i=1}^{M} ||\hat{\mathbf{W}} \cdot \mathbf{z}_i - \mathbf{y}_i||^2. \tag{11}$$

To solve for $\mathbf{W}$, we first define the vectorization:

$$\mathbf{w} = \text{vec}(\mathbf{W}) = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_R \end{bmatrix} \in \mathbb{R}^{DR}. \tag{12}$$

With this we can express:

$$\mathbf{W}\mathbf{z}_i = (\mathbf{z}_i^\top \otimes \mathbf{I})\mathbf{w} = \tilde{\mathbf{Z}}_i \mathbf{w}, \tag{13}$$

where $\otimes$ is the Kronecker product, $\tilde{\mathbf{Z}}_i = (\mathbf{z}_i^\top \otimes \mathbf{I}) \in \mathbb{R}^{R \times DR}$ and $\mathbf{I}_R \in \mathbb{R}^{R \times R}$ is the identity matrix. Then, we obtain our optimization objective

$$f(\mathbf{W}) = \sum_{i=1}^{M} ||\mathbf{W} \cdot \mathbf{z}_i - \mathbf{y}_i||^2 = \sum_{i=1}^{M} ||\mathbf{y}_i - \tilde{\mathbf{Z}}_i \mathbf{w}||^2. \tag{14}$$

Since $f(\mathbf{W})$ is convex, setting its derivative to zero obtains the following equation:

$$0 = \nabla_{\mathbf{W}} \sum_{i=1}^{M} ||\mathbf{y}_i - \tilde{\mathbf{Z}}_i \mathbf{w}||^2 = \sum_{i=1}^{M} 2 \cdot \tilde{\mathbf{Z}}_i^\top (\mathbf{y}_i - \tilde{\mathbf{Z}}_i \mathbf{w}) = \sum_{i=1}^{M} (\tilde{\mathbf{Z}}_i^\top \mathbf{y}_i - \tilde{\mathbf{Z}}_i^\top \tilde{\mathbf{Z}}_i \mathbf{w}). \tag{15}$$

From this equation we derive:

$$\sum_{i=1}^{M} \tilde{\mathbf{Z}}_i^\top \mathbf{y}_i = \sum_{i=1}^{M} \tilde{\mathbf{Z}}_i^\top \tilde{\mathbf{Z}}_i \mathbf{w} \implies \tilde{\mathbf{Z}}^\top \mathbf{y} = \tilde{\mathbf{Z}}^\top \tilde{\mathbf{Z}} \mathbf{w}, \tag{16}$$

where

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_M \end{bmatrix} \in \mathbb{R}^{MR} \quad \text{and} \quad \tilde{\mathbf{Z}} = \begin{bmatrix} \tilde{\mathbf{Z}}_1 \\ \tilde{\mathbf{Z}}_2 \\ \vdots \\ \tilde{\mathbf{Z}}_M \end{bmatrix} \in \mathbb{R}^{MR \times DR}, \tag{17}$$

and taking the pseudoinverse $(\tilde{\mathbf{Z}}^\top \tilde{\mathbf{Z}})^\dagger$, we obtain the solution:

$$\mathbf{w} = (\tilde{\mathbf{Z}}^\top \tilde{\mathbf{Z}})^\dagger \tilde{\mathbf{Z}}^\top \mathbf{y}. \tag{18}$$

Since $\tilde{\mathbf{Z}} = (\mathbf{Z}^\top \otimes \mathbf{I}_R) \in \mathbb{R}^{MR \times DR}$, we substitute this in to equation 18 and use properties of the Kronecker product to obtain:

$$\begin{aligned} \text{vec}(\mathbf{W}) = \mathbf{w} &= ((\mathbf{Z}^\top \otimes \mathbf{I}_R)^\top (\mathbf{Z}^\top \otimes \mathbf{I}_R))^\dagger (\mathbf{Z}^\top \otimes \mathbf{I}_R)^\top \mathbf{y} \\ &= ((\mathbf{Z}^\top \otimes \mathbf{I}_R)^\top (\mathbf{Z}^\top \otimes \mathbf{I}_R))^\dagger (\mathbf{Z}^\top \otimes \mathbf{I}_R)^\top \mathbf{y} \\ &= (\mathbf{Z}\mathbf{Z}^\top \otimes \mathbf{I}_R)^\dagger (\mathbf{Z} \otimes \mathbf{I}_R)\mathbf{y} \\ &= ((\mathbf{Z}\mathbf{Z}^\top)^\dagger \mathbf{Z} \otimes \mathbf{I}_R)\mathbf{y} \\ &= \text{vec}(\mathbf{Y}^\top \mathbf{Z}^\top (\mathbf{Z}\mathbf{Z}^\top)^\dagger) \end{aligned} \tag{19}$$

Therefore, denoting the unnormalized covariance as $\Sigma_{\mathbf{Z}} = \mathbf{Z}\mathbf{Z}^\top$, we have:

$$\mathbf{W} = \mathbf{Y}^\top \mathbf{Z}^\top (\mathbf{Z}\mathbf{Z}^\top)^\dagger = \mathbf{Y}^\top \mathbf{Z}^\top (\Sigma_{\mathbf{Z}})^\dagger. \tag{20}$$

Substituting this into the downstream prediction error $\delta_{\mathbf{Z}} = \frac{1}{M}\sum_{i=1}^{M}||\mathbf{W}\cdot\mathbf{z}_i - \mathbf{y}_i||$ from equation 4, we have,

$$\begin{aligned}
\delta_{\mathbf{Z}} &= \frac{1}{M}\sum_{i=1}^{M}||\mathbf{W}\cdot\mathbf{z}_i - \mathbf{y}_i|| \\
&= \frac{1}{M}\sum_{i=1}^{M}\sqrt{\sum_{r=1}^{R}((\mathbf{W}\cdot\mathbf{z}_i)[r] - \mathbf{y}_i[r])^2} \\
&\leq \sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{r=1}^{R}((\mathbf{W}\cdot\mathbf{z}_i)[r] - \mathbf{y}_i[r])^2} \\
&= \frac{1}{\sqrt{M}}||\mathbf{W}\mathbf{Z} - \mathbf{Y}_{\mathcal{S}}^{\top}||_F \\
&= \frac{1}{\sqrt{M}}||\mathbf{Y}^{\top}\mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z} - \mathbf{Y}_{\mathcal{S}}^{\top}||_F \qquad \text{[From Eqn (20)]} \\
&= \frac{1}{\sqrt{M}}||\mathbf{Y}^{\top}(\mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z} - \mathbf{I})||_F \\
&= \frac{1}{\sqrt{M}}||(\mathbf{I} - \mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z})\mathbf{Y}||_F,
\end{aligned} \tag{21}$$

where $\mathbf{I} \in \mathbb{R}^{M\times M}$.

Defining the projection matrix $\mathbf{P}_{\mathbf{Z}} = \mathbf{I} - \mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z}$, we arrive at our upper bound:

$$\begin{aligned}
\delta_{\mathbf{Z}} &\leq \frac{1}{\sqrt{M}}||(\mathbf{I} - \mathbf{Z}^{\top}(\Sigma_{\mathbf{Z}})^{\dagger}\mathbf{Z})\mathbf{Y}||_F \\
&\leq \frac{1}{\sqrt{M}}||\mathbf{P}_{\mathbf{Z}}\mathbf{Y}||_F \\
&\leq ||\mathbf{P}_{\mathbf{Z}}\mathbf{Y}||_F \leq ||\mathbf{P}_{\mathbf{Z}}||_F||\mathbf{Y}||_F.
\end{aligned} \tag{22}$$

□

Elaborating on section 3.2, we minimize $\delta_{\mathbf{Z}}$ by minimizing the term $||\mathbf{P}_{\mathbf{Z}}||_F$. Since $M > D$ and $\mathbf{Z}$ is a real matrix, $||\mathbf{P}_{\mathbf{Z}}||_F$ is minimized when the row rank of $\Sigma_{\mathbf{Z}}$ is maximized [44, 3]. We note that the rank of a diagonal matrix is equal to the number of non-zero eigenvalues, and that the eigenvalues of a diagonal matrix is its diagonal entries. From this, we maximize the rank of $\Sigma_{\mathbf{Z}}$ by maximizing the values of its diagonal entries with a separation term $\mathcal{L}_{sep}$ while regularizing it to be a diagonal matrix by minimizing its off-diagonal terms with an entanglement term $\mathcal{L}_{ent}$. In this work, we demonstrate the benefits of these object-centric regularization terms as an alternative to reconstruction loss for multi-object representation learning.

E Object Property Prediction based on Learned Object Representation

Additional Scores. In Table 7, we show the additional results of the object property prediction task on the CLEVRTEX dataset.

Informativeness Scores. In Table 8, we show the detailed results of the object property prediction task on Multi-dSprites and Tetrominoes-NM.

Disentanglement and Completeness Estimates. To further evaluate the quality of the obtained object representations, we follow [34, 11] and estimate the disentanglement and completeness scores [10]. Given K properties and D object representation dimensions, we use features learned by the GBT to derive an importance matrix $\mathbf{I}$ of K rows and D columns, where every row denotes the importance of each dimension of the input object representation in predicting the property. We use this to compute the entropy of predictive importance of each dimension across all properties and

Table 7: R^2 scores for object property prediction on CLEVRTEX

Method	CLEVRTEX	
	Position	Shape
Slot Attention	47.5±19.6	30.6
EfficientMORL	21.8±2.0	18.5
GENESIS-V2	79.8±8.0	35.2
SLATE	62.0±8.0	30.5
SysBinder	38.2±3.1	29.6
BO-QSA	66.5±0.3	28.7
OC-Net	**80.7±2.4**	**36.1**

Table 8: Detailed R^2 scores for object property prediction on simulated datasets

(a) Multi-dSprites

Method	Red	Green	Blue	X-coord	Y-coord	Shape
Slot Attention	62.06	68.44	86.19	96.73	96.95	38.20
EfficientMORL	84.13	81.79	93.53	95.90	95.77	61.67
GENESIS-V2	86.40	76.08	71.73	97.57	96.55	75.82
SLATE	86.85	88.17	87.60	87.47	93.63	31.74
SysBinder	73.87	72.54	74.42	71.73	66.89	33.33
BO-QSA	97.64	94.54	96.62	97.63	97.17	75.20
OC-Net	**98.71**	**97.82**	**97.47**	**98.33**	**98.29**	**78.07**

(b) Tetrominoes-NM

Method	Red	Green	Blue	X-coord	Y-coord	Shape
Slot Attention	85.00	80.83	93.54	99.15	98.32	36.25
EfficientMORL	91.45	95.20	97.91	98.35	97.38	68.54
GENESIS-V2	84.58	94.79	84.79	96.38	92.77	37.92
SLATE	81.24	86.50	88.79	89.13	90.13	10.5
SysBinder	97.30	98.43	97.98	75.88	79.71	19.94
BO-QSA	97.30	98.43	98.65	98.79	99.03	52.46
OC-Net	**100.0**	**100.0**	**100.0**	**99.50**	**99.34**	**98.65**

define the disentanglement value as the weighted sum:

$$\text{DIS}(\mathbf{I}) = \sum_{d=1}^{D} w_d(1 - \text{H}(\mathbf{I}[1,d], \ldots, \mathbf{I}[K,d])), \tag{23}$$

where $\text{H}(.)$ is the entropy and $w_d = \frac{\sum_{k=1}^{K} \mathbf{I}[k,d]}{\sum_{d=1}^{D}\sum_{k=1}^{K} \mathbf{I}[k,d]}$ is the weight of d. Conversely, we compute the entropy of the rate that each property is captured by a dimension and define the completeness value as:

$$\text{COM}(\mathbf{I}) = \sum_{k=1}^{K} w_k(1 - \text{H}(\mathbf{I}[k,1], \ldots, \mathbf{I}[k,D])), \tag{24}$$

where $w_k = \frac{\sum_{d=1}^{D} \mathbf{I}[k,d]}{\sum_{d=1}^{D}\sum_{k=1}^{K} \mathbf{I}[k,d]}$.

We show the results in Table 9. We find that the object representations obtained by OC-Net achieve both higher disentanglement and completeness than all baselines.

Table 9: Disentanglement and completeness scores for object representations

	Multi-dSprites		Tetrominoes-NM	
Method	Disentanglement	Completeness	Disentanglement	Completeness
Slot Attention	68.97	49.40	56.52	44.44
EfficientMORL	60.97	67.94	54.47	61.59
GENESIS-V2	71.59	64.70	39.58	54.13
SLATE	84.06	55.57	52.10	31.61
SysBinder	80.12	51.45	66.40	61.74
BO-QSA	90.05	77.01	69.47	59.89
OC-Net	**93.99**	**86.36**	**99.22**	**77.00**

F Additional Experiments

F.1 Additional Experiments on Quality of Discovered Objects

We further evaluate OC-Net against additional works that that explore image segmentation by modelling a graph on top of hand-crafted features and learned features.

The Normalized Cut (Ncut) algorithm [43] performs unsupervised image segmentation by treating image segmentation as a graph partitioning problem and uses a hand-crafted criterion to measure both the total dissimilarity between the different groups as well as the total similarity within the groups in order to determine the final segmentation groups.

MaskCut [47] was recently proposed to perform unsupervised image segmentation by leveraging on a pre-trained model. MaskCut first extracts learned features of the input image by using the DINO model, which was pre-trained on the ImageNet dataset using self-supervised learning techniques[7], before applying the Ncut algorithm on the extracted features to determine the segmentation.

The mIoU results in Table 10 below show that OC-Net significantly outperforms all other graph-based methods:

Table 10: mIoU scores for discovered foreground objects

Method	Multi-dSprites	Tetro-NM	SVHN	IDRiD	CTEX	CTEX-OOD
Felzenszwalb	95.0±0.0	96.9±0.0	39.8±0.0	15.4±0.0	26.8±0.0	23.4±0.0
Ncut	58.9±0.0	57.4±0.0	32.2±0.0	4.3±0.0	22.9±0.0	18.6±0.0
MaskCut	47.1±0.0	69.3±0.0	31.7±0.0	7.8±0.0	33.5±0.0	34.1±0.0
OC-Net	**99.1±0.0**	**100.0±0.0**	**49.9±0.1**	**31.2±0.2**	**37.5±0.7**	**35.0±0.6**

F.2 Experiments on Model Speed

We perform additional experiments to evaluate the speed of OC-Net. Table 11 shows the average time per iteration and the total training time of the various methods on the Multi-dSprites dataset. We train all models on an 8X Tesla V100 (32GB) GPU server.

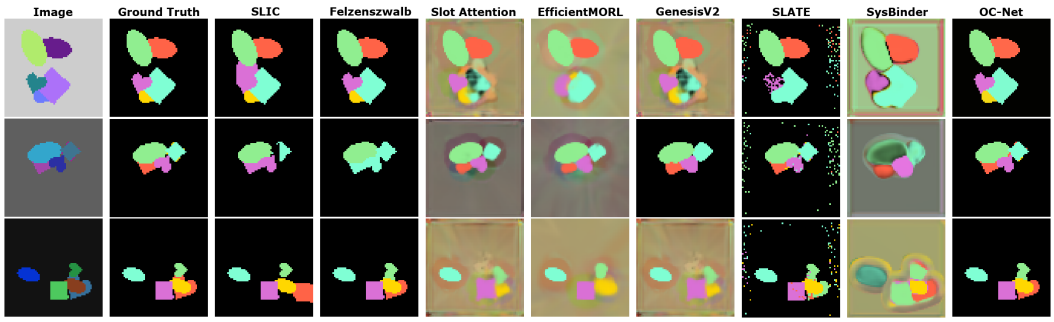
G Additional Visualizations

G.1 Object Discovery

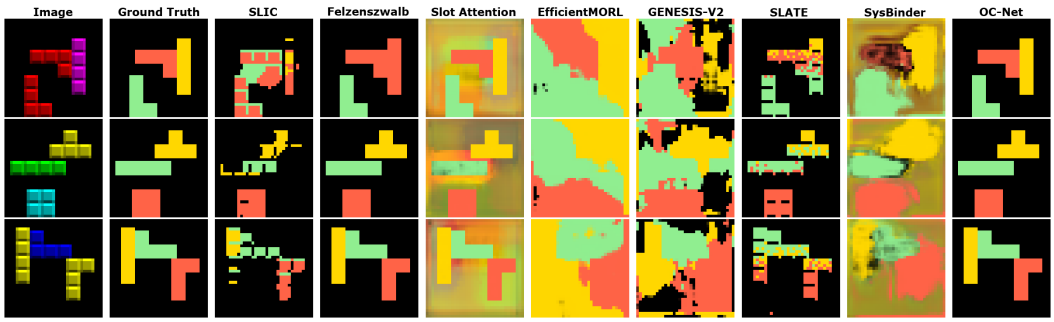
We visualize the results of additional samples from simulated datasets in Figure 6, real-world datasets in Figure 7, complex texture datasets in Figure 8 and common object datasets in Figure 9. For the common object datasets, we perform additional comparison with SLASH [27].

Table 11: Processing speed and training time across various methods on Multi-dSprites

Method	Multi-dSprites	
	Time / Iteration (ms)	Training Time (hours)
Slot Attention	75	5.26
EfficientMORL	139	11.60
GENESIS-V2	132	11.07
SLATE	385	21.4
SysBinder	235	13.06
BO-QSA	66	4.63
OC-Net	137	0.17

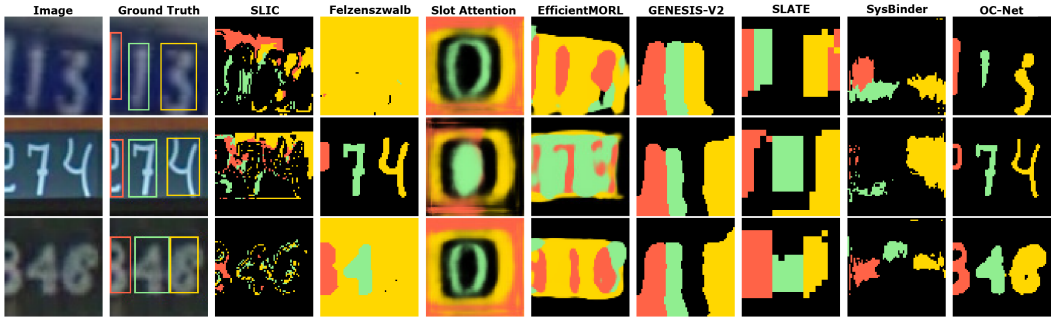


(a) Multi-dSprites

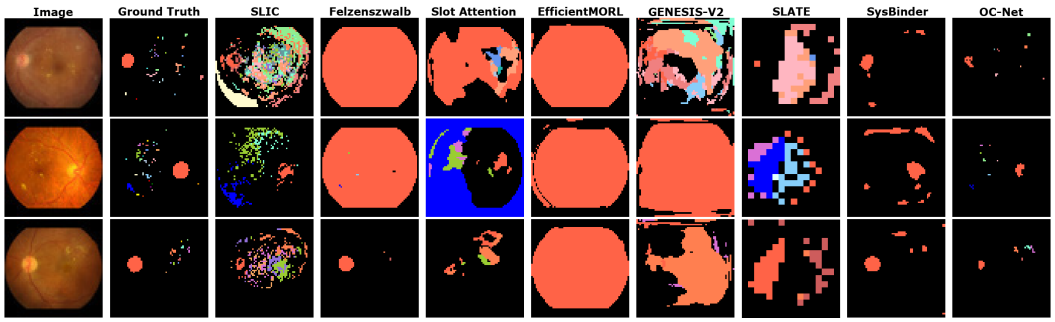


(b) Tetrominoes-NM

Figure 6: Supplementary visualization of discovered objects on simulated datasets.

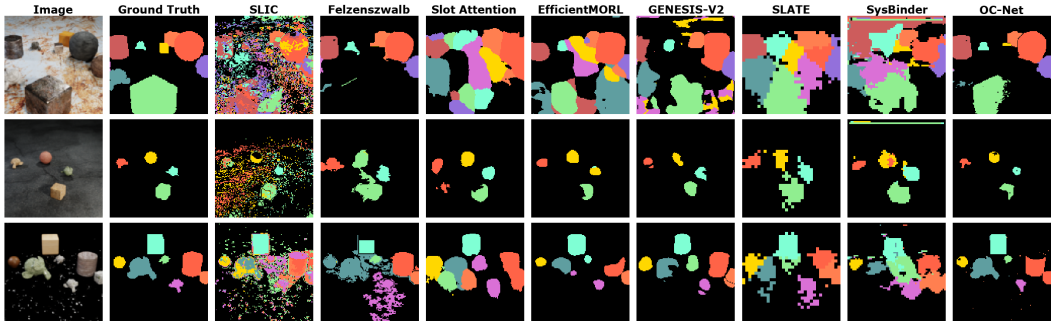


(a) SVHN

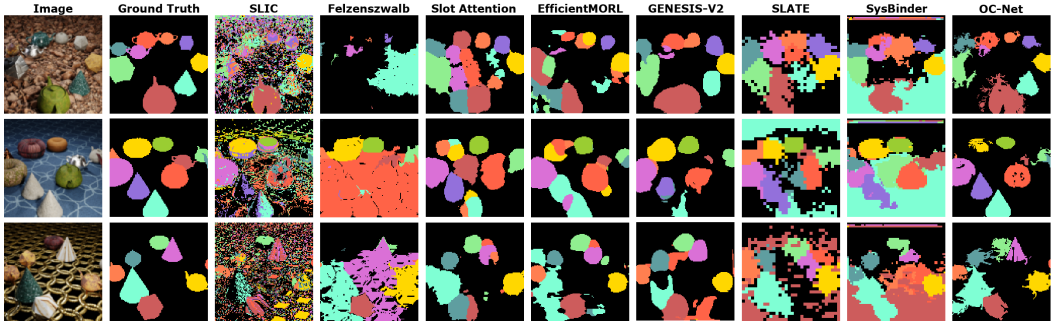


(b) IDRiD

Figure 7: Supplementary visualization of discovered objects on real-world datasets.

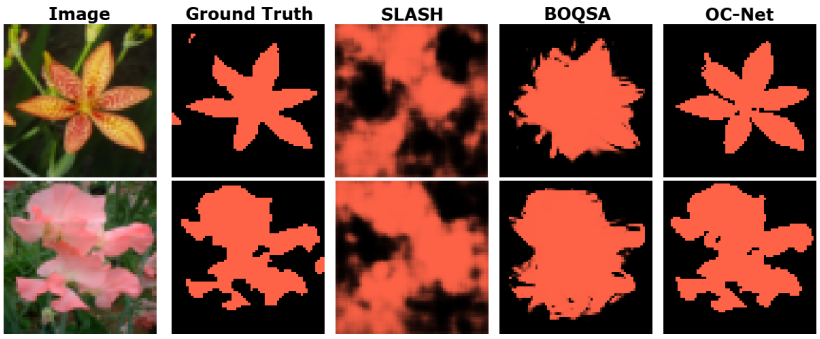


(a) CLEVRTEX

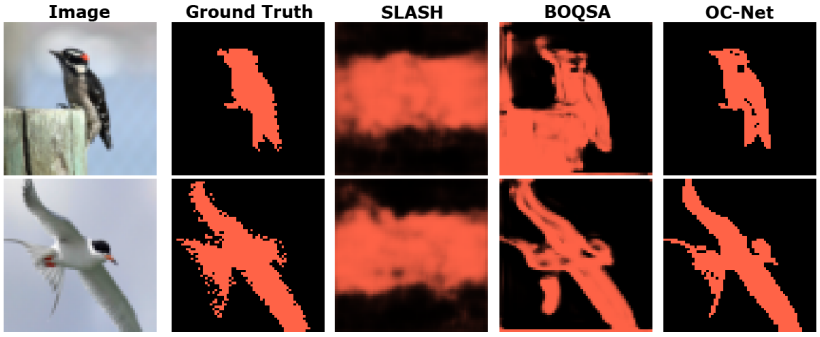


(b) CLEVRTEX-OOD

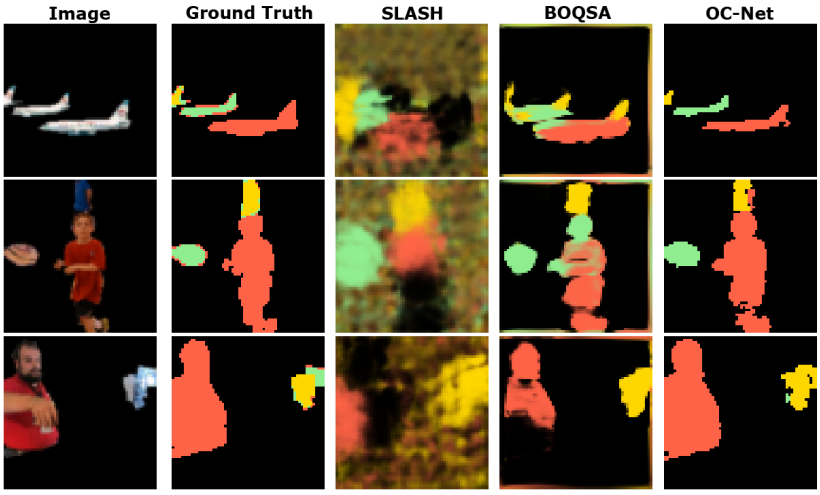
Figure 8: Supplementary visualization of discovered objects on complex texture datasets.



(a) Flowers



(b) Birds



(c) COCO

Figure 9: Supplementary visualization of discovered objects on common object datasets.

G.2 Model Generalizability

We visualize the results of training on Multi-dSprites and testing on Tetrominoes-NM in Figure 10, IDRiD in Figure 11, SVHN in Figure 12, CLEVRTEX in Figure 13 and CLEVRTEX-OOD in Figure 14.

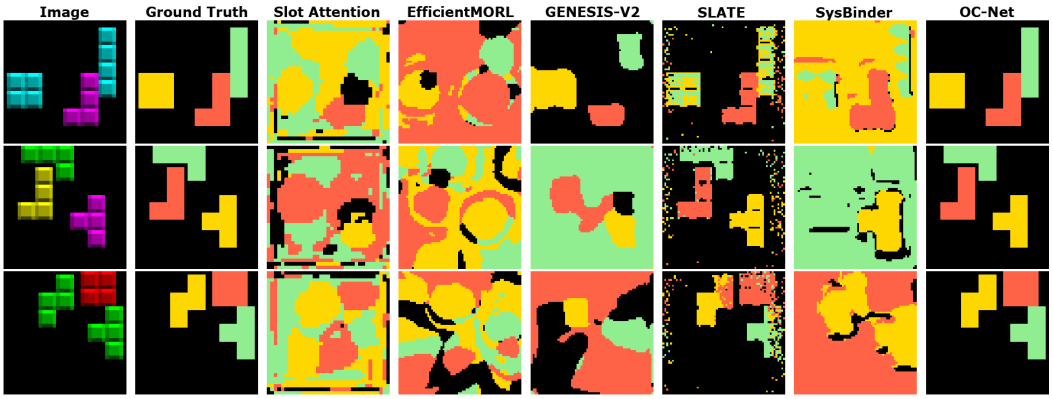


Figure 10: Visualization of model generalizability on Tetrominoes-NM after training on Multi-dSprites.

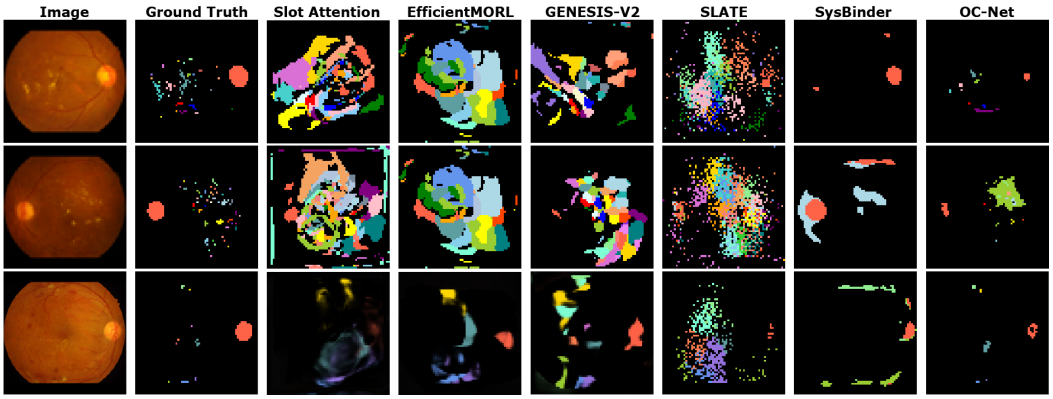


Figure 11: Visualization of model generalizability on IDRiD after training on Multi-dSprites.

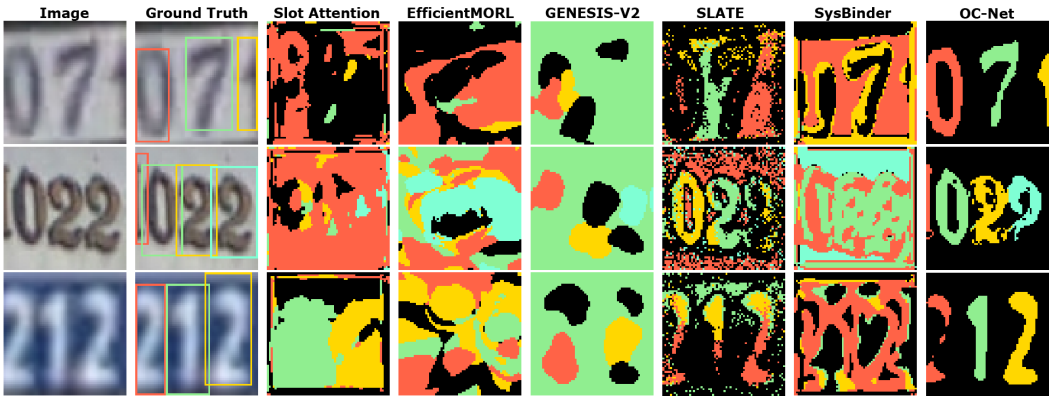


Figure 12: Visualization of model generalizability on SVHN after training on Multi-dSprites.

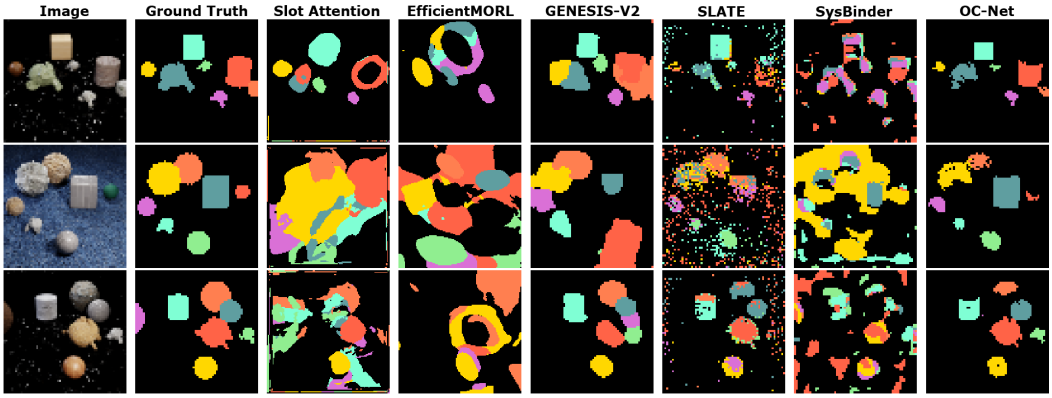


Figure 13: Visualization of model generalizability on CLEVRTEX after training on Multi-dSprites.

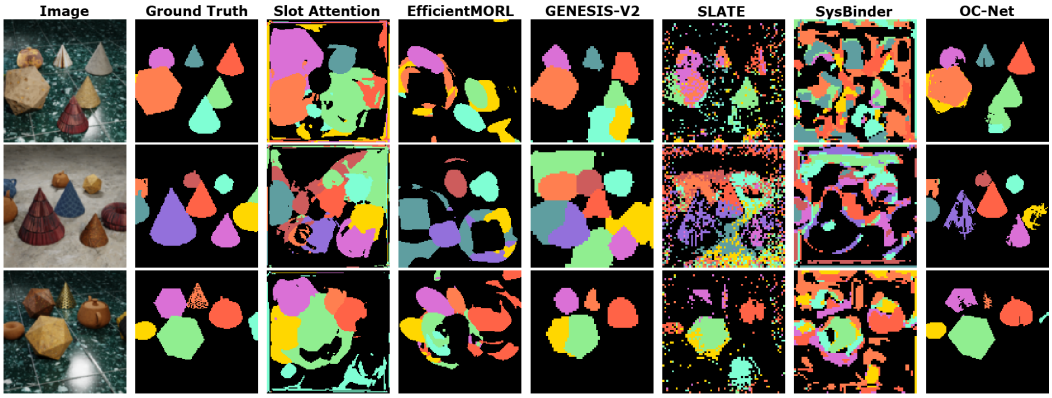


Figure 14: Visualization of model generalizability on CLEVRTEX-OOD after training on Multi-dSprites.

H Additional Discussions and Limitations

H.1 Preservation of Pixel Information for Fine-Grained Object Discovery

OC-Net uses a single 1×1 convolutional layer to obtain pixel embeddings which are individually refined and clustered before the groups of pixel embeddings are processed into object representations. In contrast, existing state-of-the-art methods such as Slot Attention, GENESIS-V2 and Efficient-MORL all use encoder networks with multiple larger sized 3×3 or 5×5 convolutions. Larger sized convolutions are very popular in general computer vision where they are typically used for edge detection especially in the early layers of a deep neural network. For fine-grained segmentation without labels, however, we show the superiority of using paths between pixels to perform both edge/border detection and also segmentation of image components whose borders stretch outside of the image, such as backgrounds. Considering that almost all state-of-the-art methods in multi-object representation learning use larger-sized convolutions as the default, we hypothesize that further exploration of these insights will prove useful to the advancement of this field.

H.2 Generative vs Latent-Space Regularization Approach for Object-Centric Learning

In contrast to the majority of existing methods, in this work we consider a regularization-based approach and focus on the object discovery process instead of optimizing for component-based image reconstruction.

Optimizing for reconstruction demands much more training samples and runtime to ensure that suitable distributions are learned for precise reconstruction. If we assume the usual input image of size $H \times W$ and having 3 channels for Red, Green and Blue, this amounts to a reconstruction objective with at least $H \times W \times 3 \times 255$ possible permutations. When $H = W = 64$, this amounts to more

than 3 million possible outputs. Early works suggested that component-based reconstruction offered an efficient alternative to full-scene image generation if a common repeating structure across samples could be exploited [5]. Inspired by this, existing state-of-the-art works such as EfficientMORL and GENESIS-V2 attempted to handle the large generation space via a combination of component-based reconstruction and constraints such as defining a learnable Gaussian distribution for generated pixels, and use the Generalized ELBO with Constrained Optimization (GECO) objective [41] to assist in stable convergence for multiple initialization seeds, since GECO reformulates the ELBO to allow the Kullback–Leibler (KL) divergence to grow large so that a predefined reconstruction threshold can first be attained. While these tactics are often effective in stabilizing the training of generative networks, they present fundamental limitations for the task of object discovery since the optimization objective primarily aims to reconstruct the input, which may often cause the components that make up the reconstructed image to be simply the most efficient image partitioning instead of semantically meaningful segments [35, 9].

To address these fundamental limitations, we instead proposed a regularization-based approach where feature connectivity guides the discovery of objects and two regularization terms that refine the representation space of the discovered objects. Such regularization-based techniques have been explored in the self-supervised learning domain [3]. We show that by focusing on the quality of objects discovered and their respective representations, such a regularization-based approach can better handle the diversity in real-world scenes in the task of object discovery.

H.3 Limitation: Memorization of Object Shapes

The original Tetrominoes dataset [23] includes samples that require the knowledge of object shapes in order to accurately separate the objects. This reveals a key limitation to our approach especially in cases where the task requires explicit memorization of objects, such as by requesting for a precise generation of an object that was seen previously. One straightforward solution could be to extend OC-Net with a dictionary of object prototypes in order to bridge this gap [51]. However, as discussed in [25] and as demonstrated in this work, demanding the memorization of object shapes in a model could be a barrier to generalizable scene decompositions outside of the training distribution, and hence in this work we decided to focus on generalizable object discovery. Further, state-of-the-art methods follow an implicit memorization approach by attempting to learn patterns within the data distribution during training, which proves to be a fundamental limitation when facing the diversity of image distributions in the real-world. Therefore, there remains fundamental challenges related to the successful design of a visual memory module which can handle both the complexity of real-world images and the complex part-whole hierarchy of objects, which we leave for future work.

I Ethics Statement

Our analysis, which is focused on publicly available multi-object simulated, real-world and complex texture data has no immediate impact on general society. To the best of our knowledge, none of the datasets used in this study contain personally identifiable information or offensive content. As with any model that performs scene understanding, applications with potential negative societal impact such as in the area of surveillance cannot be fully excluded upon future research in this area. At this point in time, however, there remains challenging open questions that need to be reliably addressed before enabling the deployment of OC-Net and its counterparts to real-world settings, and hence a direct application of this method for malicious purposes is currently unlikely.

J Third-Party Assets

OC-Net is implemented using PyTorch [39].In addition to various open-sourced Python packages, we make use of the following third-party assets:

- Locatello et al. [35] (Apache-2.0 license): Implementation of Slot Attention[1],
- Emami et al. [11] (MIT license): Implementation of EfficientMORL[2],
- Engelcke et al. [13] (GPL-3.0 license): Implementation of GENESIS-V2[3],
- Singh et al. [45] (MIT license): Implementation of SLATE[4],
- Singh et al. [46] (MIT license): Implementation of SysBinder[5].
- Jia et al. [22]: Implementation of BO-QSA[6].

[1] https://github.com/google-research/google-research/tree/master/slot_attention
[2] https://github.com/pemami4911/EfficientMORL
[3] https://github.com/applied-ai-lab/genesis
[4] https://github.com/singhgautam/slate
[5] https://github.com/singhgautam/sysbinder
[6] https://github.com/YuLiu-LY/BO-QSA