BRIDGING THE GAP TO REAL-WORLD OBJECT-CENTRIC LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Humans naturally decompose their environment into entities at the appropriate level of abstraction to act in the world. Allowing machine learning algorithms to derive this decomposition in an unsupervised way has become an important line of research. However, current methods are restricted to simulated data or require additional information in the form of motion or depth in order to successfully discover objects. In this work, we overcome this limitation by showing that reconstructing features from models trained in a self-supervised manner is a sufficient training signal for object-centric representations to arise in a fully unsupervised way. Our approach, DINOSAUR, significantly out-performs existing image-based object-centric learning models on simulated data and is the first unsupervised object-centric model that scales to real-world datasets such as COCO and PASCAL VOC. DINOSAUR is conceptually simple and shows competitive performance compared to more involved pipelines from the computer vision literature.

1 INTRODUCTION

Object-centric representation learning has the potential to greatly improve generalization of computer vision models, as it aligns with causal mechanisms that govern our physical world (Schölkopf et al., 2021; Dittadi et al., 2022). Due to the compositional nature of scenes (Greff et al., 2020), object-centric representations can be more robust towards out-of-distribution data (Dittadi et al., 2022) and support more complex tasks like reasoning (Assouel et al., 2022; Yang et al., 2020) and control (Zadaianchuk et al., 2020; Mambelli et al., 2022; Biza et al., 2022). They are in line with studies on the characterization of human perception and reasoning (Kahneman et al., 1992; Spelke & Kinzler, 2007). Inspired by the seemingly unlimited availability of unlabeled image data, this work focuses on *unsupervised* object-centric representation learning.

Most unsupervised object-centric learning approaches rely on a reconstruction objective, which struggles with the variation in real-world data. Existing approaches typically implement "slot"-structured bottlenecks which transform the input into a set of object representations and a corresponding decoding scheme which reconstructs the input data. The emergence of object representations is primed by the probabilistic permutation symmetry (Bloem-Reddy & Teh, 2020) of the set bottleneck of models like Slot Attention (Locatello et al., 2020) that group together independently repeating visual patterns across a fixed data set. While this approach was successful on simple synthetic datasets, where low-level features like color help to indicate the assignment of pixels to objects, those methods have failed to scale to complex synthetic or real-world data (Eslami et al., 2016; Greff et al., 2019; Burgess et al., 2019; Locatello et al., 2020).

To overcome these limitations, previous work has used additional information sources, e.g. motion or depth (Kipf et al., 2022; Elsayed et al., 2022). Like color, motion and depth act as grouping signals when objects move or stand-out in 3D-space. Unfortunately, this precludes training on most real-world image datasets, which do not include depth annotations or motion cues. Following deep learning's mantra of scale, another appealing approach could be to increase the capacity of the Slot Attention architecture. However, our experiments (Sec. 4.4) suggest that scale alone is *not* sufficient to close the gap between synthetic and real-world datasets. We thus conjecture that the image reconstruction objective on its own does not provide sufficient inductive bias to give rise to object groupings when objects have complex appearance. But instead of relying on auxiliary external signals, we introduce an additional inductive bias by reconstructing features that have a high

level of homogeneity within objects. Such features can easily be obtained via recent self-supervised learning techniques like DINO (Caron et al., 2021). We show that combining such a perceptual similarity (Dosovitskiy & Brox, 2016) with existing grouping modules such as Slot Attention leads to models that significantly out-perform other image-based object-centric methods and *bridge the gap to real-world object-centric representation learning*. The proposed architecture DINOSAUR (**DINO** and **S**lot **A**ttention **U**sing **R**eal-world data) is conceptually simple and highly competitive with existing unsupervised segmentation and object discovery methods in computer vision.

2 Related Work

Our research follows a body of work studying the emergence of *object-centric representations* in neural networks trained end-to-end with certain architectural biases (Eslami et al., 2016; Burgess et al., 2019; Greff et al., 2019; Lin et al., 2020; Engelcke et al., 2020; Locatello et al., 2020; Niemeyer & Geiger, 2021; Singh et al., 2022a). These approaches implicitly define objects as repeating patterns across a closed-world dataset that can be discovered e.g. via semantic discrete- or set-valued bottlenecks. As the grouping of low-level features into object entities is often somewhat arbitrary (it depends for example on the scale and level of detail considered), recent work has explored additional information sources such as video (Kosiorek et al., 2018; Crawford & Pineau, 2020; Jiang et al., 2020; Veerapaneni et al., 2020; Weis et al., 2021; Singh et al., 2022b; Traub et al., 2022), optical flow (Kipf et al., 2022; Elsayed et al., 2022; Bao et al., 2022), text descriptions of the scene (Xu et al., 2022) or some form of object-location information (e.g. with bounding boxes) (Kipf et al., 2022). In contrast, we completely avoid additional supervision by leveraging the implicit inductive bias contained in the self-supervised features we reconstruct, which present a high level of homogeneity within objects (Caron et al., 2021). This circumvents the scalability challenges of previous work that rely on pixel similarity as opposed to perceptual similarity (Dosovitskiy & Brox, 2016) and enables object discovery on real-world data without changing the existing grouping modules.

In the computer vision literature, the task of structuring natural scene without any human annotations has recently also enjoyed popularity. Ji et al. (2019); Ouali et al. (2020); Cho et al. (2021); Gansbeke et al. (2021; 2022); Zadaianchuk et al. (2022); Hamilton et al. (2022) tackle *unsupervised semantic segmentation*, i.e. splitting the scene into semantically coherent regions. These methods are typically separated into several independent steps: first, dense representations for each pixel in the image are refined from a pretrained self-supervised model; second, those representations are clustered on the whole dataset to assign a class category to each pixel. Instead, we combine those steps into a simple method that groups semantically meaningful regions jointly with learning region representations.

Another line of work focuses on *unsupervised object localization* (Vo et al., 2020; 2021; Simeoni et al., 2021; Melas-Kyriazi et al., 2022; Wang et al., 2022). These methods often group pre-trained, self-supervised features of individual images into initial object proposals, and then aggregate and cluster these object proposals across images to assign them discrete labels. Self-training can then be used on top to achieve multi-object detection on natural scenes. In contrast, our method uses the reconstruction loss to discover objects and makes little to no assumption on the underlying features and how they represent objects.

3 Method

Our approach essentially follows the usual autoencoder-like design of object-centric models and is summarized in Figure 1: a first module extracts features from the input data (the encoder), a second module groups them into a set of latent vectors called *slots*, and a final one (the decoder) tries to reconstruct some target signal from the latents. However, our method crucially differs from other approaches in that instead of reconstructing the original inputs, the decoder is tasked to reconstruct *the encoder's output*. We now describe our choices for each module, with additional details in App. C.1.

ViT as Encoder We use a vision transformer (ViT) (Dosovitskiy et al., 2021) as the feature extractor. ViTs process the image as a set of N square patches, where each patch is linearly mapped to an embedding vector. After adding a learned positional encoding to each patch, the set of patches is transformed through a series of alternating self-attention (Vaswani et al., 2017) and MLP blocks. The output tokens $\boldsymbol{h} \in \mathbb{R}^{N \times D_{\text{feat}}}$ of the last ViT block form the input to the Slot Attention stage.

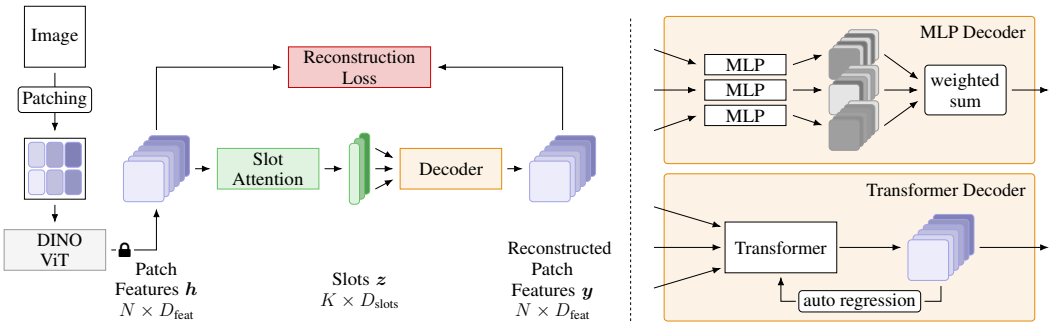

Figure 1: Overview of the proposed architecture DINOSAUR. The image is processed into a set of patch features by a frozen ViT encoder pre-trained using the DINO method. Slot attention groups these features into a set of slots. The model is trained by reconstructing the input features from the slots, either independently per-slot (MLP decoder) or jointly via auto regression (Transformer decoder).

Slot Attention Grouping The grouping stage of our model uses Slot Attention (Locatello et al., 2020) to turn the the set of ViT features $\boldsymbol{h}$ into a set of K slot vectors $\boldsymbol{z} \in \mathbb{R}^{K \times D_{\text{slots}}}$. This follows an iterative process where slots compete for input features using an attention mechanism, starting from randomly sampled initial slots. We largely use the original Slot Attention formulation (including GRU (Cho et al., 2014) and residual MLP modules), with one difference: we do not add positional encodings on the ViT features before Slot Attention, as we found the ViT's initial position encodings to be sufficient to support spatial grouping of the features. Additionally, we add a small one-hidden-layer MLP that transforms each ViT feature before Slot Attention.

Self-Supervised Pre-Training with DINO The architecture as described so far does not suffice for learning object-centric representations on real-world data. Our intuition is that a randomly initialized feature space makes it difficult for the grouping module to find commonly occurring patterns, in turn sending poor gradients to the feature encoder. Therefore, we start with a generic self-supervised pre-training step of the encoder. Self-supervised learning techniques have been successful in learning powerful representations for vision tasks such as classification and object detection (Chen et al., 2020b; Grill et al., 2020; He et al., 2022). While we demonstrate that different self-supervised approaches work for the purpose of object-centric learning (Sec. 4.4), in this work we mainly apply the DINO method (Caron et al., 2021). DINO applies a student-teacher distillation scheme for training, namely by encouraging representations resulting from multiple crops and distortions of the input image to match between student and teacher network. The resulting features have been shown to be highly discriminative of objects (e.g. Amir et al., 2021), making DINO an attractive candidate for pre-training.

Training Signal We hypothesize that reconstruction on the pixel level produces too weak of a signal for object-centricness to emerge; the task focuses (at least initially) strongly on low-level image features such as color statistics. This quickly decreases the reconstruction error, but the resulting model does not discover objects beyond datasets where objects are mostly determined by distinct object colors. Instead, if we had a signal that required higher-level semantic information to reconstruct, there would be pressure on the slots to efficiently encode this information as well. Luckily, we have such a signal available in the form of the features produced by the DINO pre-trained ViT. Thus, instead of images, the slots are trained to *reconstruct the input features* well, by minimizing the following loss:

$$\mathcal{L}_{\text{rec}} = \|\boldsymbol{y} - \boldsymbol{h}\|^2, \qquad \boldsymbol{y} = \text{Decoder}(\boldsymbol{z}). \tag{1}$$

As we use the intermediate features $\boldsymbol{h}$ as the reconstruction target, training the full architecture would lead to training collapse (because of trivial solutions minimizing the loss). We therefore freeze the pre-trained ViT encoder and only optimize Slot Attention and the decoder. This loss is akin to a perceptual similarity (Dosovitskiy & Brox, 2016), optimizing the slots in a space more semantic than pixel space.

Feature Decoding As we apply feature instead of image reconstruction as the training objective, we need a decoder architecture suitable for this purpose. To this end, we consider two different designs: a MLP decoder that is applied independently to each slot, and a Transformer decoder (Vaswani et al., 2017) that autoregressively reconstructs the set of features. We describe both options in turn.

The *MLP decoder* follows a similar design as the commonly used spatial broadcast decoder (Watters et al., 2019). Each slot is first broadcasted to the number of patches, resulting in a set of N tokens for each slot. To make the spatial positions of the tokens identifiable, a learned positional encoding is added to each token. The tokens for each slot are then processed token-wise by the same MLP, producing the reconstruction $\hat{\boldsymbol{y}}_k$ for slot k, plus an alpha map $\boldsymbol{\alpha}_k$ that signifies where the slot is active. The final reconstruction $\boldsymbol{y} \in \mathbb{R}^{N \times D_{\text{feat}}}$ is formed by taking a weighted sum across the slots:

$$\boldsymbol{y} = \sum_{k=1}^{K} \hat{\boldsymbol{y}}_k \odot \boldsymbol{m}_k, \qquad \boldsymbol{m}_k = \underset{k}{\operatorname{softmax}}\, \boldsymbol{\alpha}_k \tag{2}$$

The advantage of this simple design is its computational efficiency: as the MLP is shared across slots and positions, decoding is heavily parallelizable.

The *Transformer decoder* (Vaswani et al., 2017) reconstructs features $\boldsymbol{y}$ jointly for all slots in an autoregressive manner. In particular, the feature at position n is generated while conditioning on the set of previously generated features $\boldsymbol{y}_{<n}$ *and* the set of slots $\boldsymbol{z}$: $\boldsymbol{y}_n = \text{Decoder}(\boldsymbol{y}_{<n}; \boldsymbol{z})$. This decoder design is more powerful than the MLP decoder as it can maintain global consistency across the reconstruction, which might be needed on more complex data. In exchange, it loses the property of independent decoding of slots, often thought to be an architectural bias necessary for object-centric learning. However, we find that object-centric slots still emerge in our experiments. We note that Transformer decoders have also been previously explored by SLATE (Singh et al., 2022a) and STEVE (Singh et al., 2022b), but to reconstruct the discrete token map of a VQ-VAE (van den Oord et al., 2017).

Evaluation Object-centric methods are commonly evaluated by inspecting masks associated with each slots. Previous approaches reconstructing to image-level typically use the decoder's alpha mask for this purpose; for the MLP decoder, we also make use of this option. The Transformer decoder does not produce an alpha mask. Instead, we have two options: the attention masks of Slot Attention (used by SLATE), or the decoder's attention mask over the slots. We found that the latter performed better (see Sec. B.5), and we use it throughout. As the masks from feature reconstruction are of low resolution, we bilinearly resize them to image resolution before comparing them to ground truth masks.

4 Experiments

Broadly, we pursue two goals: 1) demonstrating that our approach significantly extends the capabilities of object-centric models towards real-world applicability (Sec. 4.1), and 2) showing that our approach is competitive with more complex methods (Sec. 4.2 and Sec. 4.3). Additionally, we ablate key model components to find what is driving the success of our method in Sec. 4.4.

Datasets In this work, we consider two synthetic and two real-world image datasets. As synthetic datasets, we use the MOVi datasets (Greff et al., 2022), recently introduced as challenging testbeds for object-centric methods. In particular, we use the variants MOVi-C and MOVi-E, which contain around 1 000 realistic 3D-scanned objects on HD backgrounds. For our purposes, the main difference is that MOVi-C contains 3–10, and MOVi-E 11–23 objects per scene. Note that we treat the video-based MOVi datasets as image datasets by randomly sampling frames. As real-world datasets, we use PASCAL VOC 2012 (Everingham et al., 2012) and MS COCO 2017 (Lin et al., 2014), commonly used for object detection and segmentation. Whereas PASCAL VOC contains many images with only a single large object, COCO consists of images with at least two and often dozens of objects. Both datasets represent a significant step-up in complexity to what object-centric models have been tested on so far.

Tasks We evaluate on the following tasks: *object discovery* (Sec. 4.1), i.e. finding instances of objects by inferring pixel masks around them, *object localization*[1] (Sec. 4.2), i.e. finding object location and size by predicting bounding boxes, and *unsupervised semantic segmentation* (Sec. 4.3), separating the image into semantically consistent labeled regions. For the latter, we consider two variations: object segmentation, where only foreground objects should get segmented and labeled, and scene decomposition, where each pixel of the image has to be labeled with a semantic class. To be comparable to prior work, each task follows a slightly different evaluation protocol (see App. D). As such, numbers between the different tasks are not directly comparable.

[1]This task is often called "object discovery" in the literature as well, but we term it "object localization" in this work in order to avoid confusion with the task evaluated in the object-centric literature.

Table 1: Object Discovery on Synthetic Datasets (mean ± standard dev., 5 seeds). We report foreground adjusted rand index (FG-ARI) and mean best overlap (mBO). Higher is better.

	MOVi-C		**MOVi-E**	
	FG-ARI	mBO	FG-ARI	mBO
Block Pattern	*42.3*	*17.1*	*41.8*	*18.6*
Slot Attention	43.8 ±0.4	27.2 ±1.0	45.0 ±1.7	23.0 ±1.2
SLATE	43.3 ±1.2	25.2 ±1.2	44.3 ±0.8	22.0 ±0.5
Ours (ViT-S/8)	67.0 ±0.4	35.2 ±0.2	64.6 ±0.7	32.0 ±0.1
Ours (ViT-B/8)	68.9 ±0.4	38.0 ±0.2	65.1 ±1.2	33.5 ±0.1

Training Details We train DINOSAUR using the Adam optimizer (Kingma & Ba, 2014) with a learning rate of $4 \cdot 10^{-4}$, linear learning rate warm-up of 10 000 optimization steps and an exponentially decaying learning rate schedule. Further, we clip the gradient norm at 1 in order to stabilize training and train for 500k steps for the MOVI and COCO datasets and 250k steps for PASCAL VOC. The models were trained on 8 NVIDIA V100 GPUs with a local batch size of 8, with 16-bit mixed precision. For the experiments on synthetic data, we use a ViT with patch size 8 and the MLP decoder. For the experiments on real-world data, we use a ViT with patch size 16 and the Transformer decoder. We analyze the impact of different decoders in Sec. 4.4. The main results are averaged over 5 random seeds; other experiments use 3 seeds. Further model details can be found in App. C.1.

4.1 Unsupervised Object Discovery

The task of object discovery – splitting an image into a set of masks that cover semantically related parts of the image – is the most frequent way object-centric models are evaluated (Burgess et al., 2019; Locatello et al., 2020). Our goal in this section is two-fold: 1) demonstrating that previous object-centric methods fail to produce meaningful results on real-world datasets and struggle even on synthetic datasets, and 2) showcase how our approach of incorporating strong pre-trained models results in a large step forward for object-centric models on both kinds of datasets.

Metrics As common in the object-centric literature, we evaluate this task using foreground adjusted rand index (FG-ARI), a metric measuring cluster similarity. Additionally, we compute a metric based on intersection-over-union (IoU), the mean best overlap (mBO) (Arbeláez et al., 2014). mBO is computed by assigning each ground truth mask the predicted mask with the largest overlap, and then averaging the IoUs of the assigned mask pairs. In contrast to ARI, mBO takes background pixels into account, thus also measuring how close masks fit to objects. On datasets where objects have a semantic label attached (e.g. on COCO), we can evaluate this metric with instance-level (i.e. object) masks, and semantic-level (i.e. class) masks. This allows us to find model preferences towards instance- or semantic-level groupings.

Baselines We compare our approach to a more powerful version of Slot Attention (Locatello et al., 2020) which is based on a ResNet encoder and has been shown to scale to more complex data (Elsayed et al., 2022). Further, we compare with SLATE (Singh et al., 2022a), a recent object-centric model that trains a discrete VQ-VAE (van den Oord et al., 2017) as the feature extractor and a Transformer as the decoder. We refer to App. C.2 for details about baseline configurations.

As it can be hard to gauge how well object-centric methods perform on new datasets solely from metrics, we add one trivial baseline: dividing the image into a set of regular block masks. This *block pattern* (see Fig. 13) thus shows the performance of a method that only follows a geometric strategy to group the data, completely ignoring the semantic aspects of the image. Familiar to practitioners, this is a common failure mode of object-centric methods, particularly of Slot Attention.

Results on Synthetic Datasets (Table 1 and Fig. 2) Both Slot Attention and SLATE struggle on the challenging MOVi datasets. Our model achieves good performance on both MOVI-C and MOVi-E. Visually, the quality is on par with reports from more involved methods that use temporal information and weak supervision (Elsayed et al., 2022; Singh et al., 2022b).

Results on Real-World Datasets (Table 2 and Fig. 3) As expected, Slot Attention can not handle the increased complexity of real-world data and degrades to non-semantic grouping patterns. For SLATE, semantic grouping begins to emerge (e.g. of backgrounds), but not consistently (see Fig. 15).

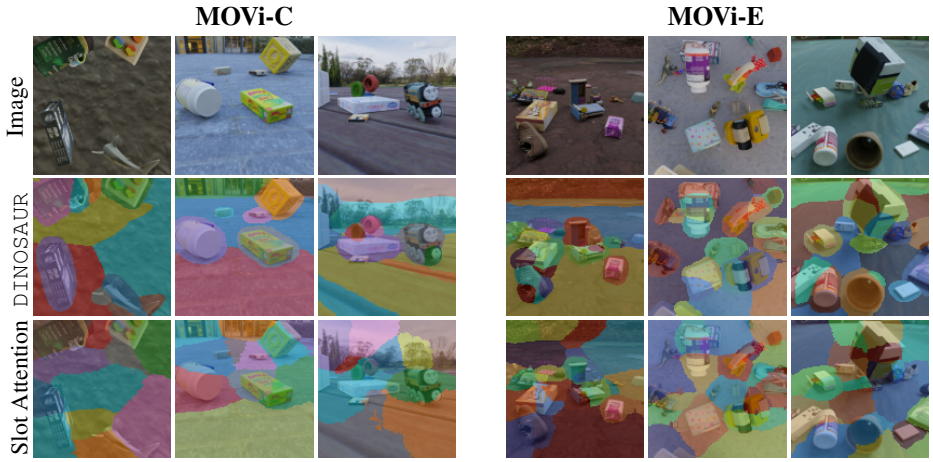

Figure 2: Example results on the synthetic MOVi-C and MOVi-E datasets (Greff et al., 2022).

Figure 3: Masks produced by our method and Slot Attention on COCO 2017, using 7 slots.

In contrast, our method captures a variety of objects of different size, appearance and shape. To the best of our knowledge, we are the first to show a successful version of an object-centric model on unconstrained real-world image data in the fully unsupervised setting. Our result represents a significant step-up in complexity of what object-centric methods can handle.

Table 2: Object Discovery on Real-World Datasets (mean $\pm$ standard dev., 5 seeds). We report foreground adjusted rand index (FG-ARI) and instance/class mean best overlap (mBO^i/mBO^c).

	PASCAL VOC 2012			**COCO**		
	FG-ARI	mBO^i	mBO^c	FG-ARI	mBO^i	mBO^c
Block Pattern	*12.6*	*24.5*	*25.0*	*22.9*	*17.6*	*19.5*
Slot Attention	11.3 $\pm$0.5	23.6 $\pm$0.8	23.9 $\pm$0.8	21.4 $\pm$0.8	17.2 $\pm$0.6	19.2 $\pm$0.4
SLATE	13.0 $\pm$0.1	28.6 $\pm$0.7	30.2 $\pm$0.9	22.8 $\pm$0.1	18.6 $\pm$0.1	21.0 $\pm$0.1
Ours	23.2 $\pm$0.9	43.6 $\pm$0.8	50.8 $\pm$1.0	33.4 $\pm$1.0	29.6 $\pm$0.6	38.7 $\pm$0.9

4.2 Unsupervised Object Localization

In the computer vision literature, the task of object localization refers to finding the location of one or more objects that repeatedly appear in a collection of images (Cho et al., 2015). In particular, objects have to be discovered by proposing bounding boxes that "correctly localize" them. An object

Table 3: Object Localization (mean ± standard dev., 5 seeds). We report CorLoc and detection rate and compare with rOSD (Vo et al., 2020), DINO-Seg (Simeoni et al., 2021), LOST (Simeoni et al., 2021), DeepSpectral (Melas-Kyriazi et al., 2022) and TokenCut (Wang et al., 2022)

	PASCAL VOC 2012		**COCO-20k**	
	CorLoc	DetRate	CorLoc	DetRate
rOSD	51.9	41.2	48.5	12.0
DINO-Seg	46.2	–	42.1	–
LOST	64.0	–	50.7	–
DeepSpectral	66.4	–	52.2	–
TokenCut	72.1	–	58.8	–
Ours	70.4 ±2.5	52.6 ±2.0	67.2 ±1.5	31.1 ±0.9
Ours (COCO Transfer)	77.8 ±1.3	58.2 ±1.1		

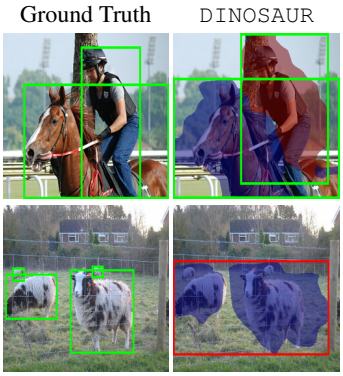

Figure 4: Object Localization on PASCAL VOC 2012.

counts as correctly localized if its bounding box has an IoU-overlap of at least 50% with a proposed bounding box. Like prior work (Vo et al., 2020; Melas-Kyriazi et al., 2022), we report the fraction of images on which at least one object was correctly localized (CorLoc), and the average fraction of objects correctly localized per image (detection rate). As our method produces masks, we use the tightest bounding box around each mask as the proposal. We evaluate on the `trainval` split of PASCAL VOC 2012 and COCO-20k, as is standard in the literature (Vo et al., 2020; Simeoni et al., 2021). COCO-20k is a subset of COCO 2014 filtered to contain no instances annotated as crowd.

Results (Table 3) The results show that our method manages to focus on the salient parts of the data. On the majority of images, at least one object is captured by a slot. Also, our method reaches results comparable or better to what has been reported in the literature on this task. Interestingly, zero-shot transferring a model trained on COCO to PASCAL yields better results than a model trained on PASCAL itself; we explain this by the size of the training data, which is around $10\times$ larger for COCO.

4.3 Unsupervised Semantic Segmentation

In this task, each pixel of the image has to be assigned a semantic class label. As in the unsupervised setting, no direct correspondence to ground truth classes are given, a successful model needs to output segments that consistently cover the same semantic content on different images.

We report results on two different versions of this task: in object segmentation, only foreground classes have to segmented; in scene decomposition, the background has to be split into semantic categories as well. For object segmentation, we evaluate on PASCAL VOC 2012 and COCO 2017, with instance masks merged to semantic class masks. For scene decomposition, we evaluate on the COCO-Stuff dataset (Caesar et al., 2018), in two different variants: COCO-27 uses a reduced set of 27 coarse classes from the COCO class hierarchy, whereas COCO-171 uses the full set of 80 foreground and 91 background classes available in the dataset. To the best of our knowledge, we are the first to report results on this challenging setting. We evaluate with mean intersection-over-union over classes (mIoU) and per-pixel accuracy (pAcc).

As our method does not directly output labels required for semantic segmentation, we obtain them by running k-means clustering on features associated with each slot after training the model, then assigning clusters to ground truth classes by maximizing IoU using Hungarian matching. Similar ways of evaluating are commonly used in the literature (Gansbeke et al., 2021; Melas-Kyriazi et al., 2022); we refer to App. C.3 for more details.

Results on Object Segmentation (Table 4a) This task is not naturally suited for our method, which decomposes the scene into semantically meaningful parts without distinguishing fore- and background. We still obtain competitive results with recently published work (Gansbeke et al., 2021; Melas-Kyriazi et al., 2022). It is also important to note that GroupViT, reporting the strongest results on this task, use additional supervision from text. Moreover, DeepSpectral, MaskDistill and COMUS employ additional steps of training segmentation networks which improves results and allows them

Table 4: Unsupervised Semantic Segmentation (mean ± standard dev., 5 seeds). We report mean intersection over union (mIoU). For scene decomposition, we additionally report per-pixel accuracy (pAcc). STEGO results are presented without CRF post-processing for fair comparison with other methods. Results marked † were produced ourselves using the official implementation.

(a) Object Segmentation. We compare with MaskContrast (Gansbeke et al., 2021), DeepSpectral (Melas-Kyriazi et al., 2022), MaskDistill (Gansbeke et al., 2022), COMUS (Zadaianchuk et al., 2022), and GroupViT (Xu et al., 2022).

	PASCAL VOC 2012	COCO
MaskContrast	35.0	–
DeepSpectral	37.2	–
MaskDistill	45.8	–
COMUS	47.3	19.6
GroupViT	51.2	20.9
Ours	35.3 ±1.6	14.6 ±0.2

(b) Scene Decomposition. We compare with IIC (Ji et al., 2019), SegDiscover (Huang et al., 2022), PiCIE (Cho et al., 2021), SlotCon (Wen et al., 2022), and STEGO (Hamilton et al., 2022).

	COCO-Stuff 27		COCO-Stuff 171	
	mIoU	pAcc	mIoU	pAcc
IIC	6.7	21.8	–	–
SegDiscover	14.3	56.5	–	–
PiCIE + H.	14.4	50.0	–	–
SlotCon	18.3	42.4	–	–
STEGO	26.8	54.8	10.0†	32.5†
Ours	24.1 ±0.7	45.0 ±1.2	13.2 ±0.2	27.2 ±0.5

to run at the original image resolution. In contrast, the masks we evaluate are only of size 14×14; we leave it to future work to improve the resolution of the produced masks.

Results on Scene Decomposition (Table 4b) We are competitive with the state-of-the-art method STEGO on the setting with 27 coarse classes. On the novel setting with all 171 classes, our method improves upon STEGO by 3.2 IoU points. This demonstrates that DINOSAUR is able to handle the full complexity of real-world scenes better.

4.4 Analysis

In this section, we analyze different aspects of our approach: the importance of feature reconstruction, the impact of the type of pre-training, and the role of the decoder. Additional experiments are included in App. B, for example analyzing the number of slots (Sec. B.1), or the slot representations (Sec. B.8).

Importance of Feature Reconstruction (Table 5) First, we analyze the choice of image vs. feature reconstruction under different encoder training modes: training from scratch, freezing, or finetuning DINO pre-trained weights. To this end, we use a ViT-B/16 encoder on COCO with Slot Attention's spatial broadcast decoder for image reconstruction and MLP decoder for feature reconstruction. Training from scratch results in divergence of the training process with both training signals. Predictably, feature reconstruction with finetuning results in training collapse, as explained in Sec. 3. Image reconstruction under both the frozen and finetuning setting fails to yield meaningful results, as visible by the striped mask patterns (see Fig. 6). Thus, even when starting from features that are highly semantic, image reconstruction does not give enough signal towards semantic grouping. The only setting that "works" (in the sense of discovering objects) is feature reconstruction with a frozen ViT encoder.

Second, we analyze whether the success of our approach stems from using *pre-trained* encoders, or from feature reconstruction. To do so, we train *ResNet34 encoders* from scratch using feature reconstruction as a training signal, and find that the performance is overall similar to using a pre-trained ViT encoder (see Table 12). This suggests that the feature reconstruction signal is the key component that allows object-centricness to emerge on real-world data. We expand in App. B.2.

Choice of Pre-Training Method (Table 6) We compare the object discovery performance on COCO using different backbones and pre-training schemes. ViT encoders use the Transformer decoder, ResNet50 encoders the MLP decoder. Besides DINO, we evaluate three other self-supervised pre-training methods: MoCo-v3 (Chen et al., 2021), MSN (Assran et al., 2022), and MAE (He et al., 2022). All choices perform better than training with image reconstruction. Remarkably, all self-supervised schemes perform well for the task of object discovery (cf. Fig. 20). This demonstrates that self-supervised training on ImageNet translates into a useful general bias towards discovering objects. In comparison, supervised training via ImageNet classification is a worse source of features. Moreover, there is a gap between ViT and ResNet features: visual inspection shows that ResNet-trained models seem to rely on a mix of semantic and geometric grouping of the image (cf. Fig. 21).

Table 5: Comparing image and feature reconstruction on COCO under different training settings for a ViT-B/16 encoder.

Training	Image Recon.	Feature Recon.
Scratch	~~diverges~~	~~diverges~~
Frozen	stripe pattern	works
Finetuning	stripe pattern	~~collapses~~

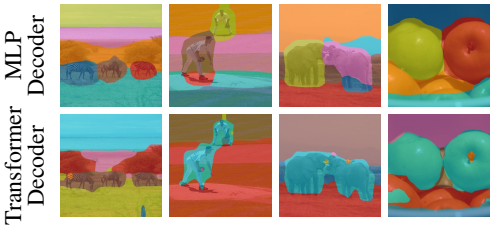

Figure 5: MLP and Transformer decoder have different biases in how they group objects.

Table 6: Comparing different ImageNet pretrained networks for feature extraction and reconstruction on COCO object discovery.

Encoder	Pre-training	ARI	mBO^i	mBO^c
ResNet50	Supervised	35.1	21.0	23.6
ResNet50	DINO	35.1	21.8	24.6
ResNet50	MoCo-v3	32.6	20.0	22.6
ViT-B/16	Supervised	31.7	24.9	30.8
ViT-B/16	DINO	33.4	29.6	38.7
ViT-B/16	MoCo-v3	34.1	29.5	37.5
ViT-B/16	MSN	34.5	29.4	36.4
ViT-B/16	MAE	32.3	28.1	32.3

Table 7: Comparing different decoders on object discovery, with a ViT-B/16 encoder. We also list mean squared reconstruction error (MSE).

Dataset	Decoder	ARI	$mBO^{(i,c)}$		MSE
MOVi-C	MLP	65.6	33.6		0.24
	Transformer	55.3	46.1		0.14
PASCAL	MLP	24.6	39.3	40.8	0.33
	Transformer	23.2	43.6	50.8	0.17
COCO	MLP	39.4	26.1	30.0	0.31
	Transformer	33.4	29.6	38.7	0.16

In Table 8, we extend the results, also studying differently sized ViTs. We conclude from this experiment that self-supervised training is important for real-world grouping to emerge, but that the specific method is less important, and that scaling the encoder yields better results.

Choice of Decoder (Table 7) We compare the choice of MLP vs. Transformer decoder for object discovery. Both options use a ViT-B/16 encoder. Generally, we find that the MLP decoder is better on ARI whereas the Transformer decoder is better on mBO. For MOVi-C, visual inspection (see Fig. 14) shows that the Transformer tends to produce tighter masks and a cleaner background separation, but uses excess slots to split objects. For PASCAL and COCO, what's striking is the Transformer decoders' improvement of 9–10 class mBO. This reveals that the Transformer decoder is biased towards grouping semantically related instances into the same slot, which we suggest stems from its global view on the image, but also from the generally increased expressiveness of the architecture (cf. the lower reconstruction loss). In contrast, the MLP decoder is able to separate instances better (see Fig. 5 and also Fig. 19), which is reflected in the higher ARI scores. Researching how different decoder designs affect semantic vs. instance-level grouping is an interesting avenue for future work. In App. B.3 and App. B.4, we further study different decoder properties.

5 Conclusion

We presented the first image-based fully unsupervised approach for object-centric learning that scales to real-world data. Our experiments demonstrate significant improvements on both simulated and real-world data compared to previously suggested approaches and even achieve competitive performance with more involved pipeline methods from the computer vision literature.

This work only takes a first step towards the goal of representing the world in terms of objects. As such, some problems remain open. One issue concerns semantic vs. instance-level grouping. As evident from the presented examples, our approach covers a mix of both, with semantically related objects sometimes being grouped into a single slot. While we found the type of decoder to influence this behavior, more fine-grained control is needed. A related issue is the detail of the decomposition, e.g. whether objects are split into parts or stay whole. We found this to be dependent on the number of slots, with a fixed number often not being appropriate (see App. B.1). How models can dynamically choose a suitable level of detail while staying unsupervised but controllable will be an important challenge to fully master the ambiguities the real world inherently presents.

In this work, we mainly focused on object discovery. Future work could further examine the properties of the learned slot representations, for instance robustness to distribution shifts, generalization and usefulness for downstream tasks. Another interesting direction is how our approach can be combined with image generation to build flexible and compositional generative models of natural data.

REPRODUCIBILITY STATEMENT

Appendix C.1 contains detailed information about the DINOSAUR architecture and all hyperparamers used for all experiments. Appendix C.2 contains details about how baselines were trained. Appendix D contains information about task evaluation and datasets. All datasets used in this work (MOVi, PASCAL VOC 2012, COCO) are public and can be obtained on their respective web pages. Source code will be released upon acceptance latest.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022.

Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep ViT Features as Dense Visual Descriptors. *ArXiv*, abs/2112.05814, 2021.

P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale Combinatorial Grouping. In *Computer Vision and Pattern Recognition*, 2014.

Rim Assouel, Pau Rodriguez, Perouz Taslakian, David Vazquez, and Yoshua Bengio. Object-centric Compositional Imagination for Visual Abstract Reasoning. In *ICLR2022 Workshop on the Elements of Reasoning: Objects, Structure and Causality*, 2022.

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Michael G. Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *ECCV*, 2022.

Zhipeng Bao, Pavel Tokmakov, A. Jabri, Yu-Xiong Wang, Adrien Gaidon, and Martial Hebert. Discovering objects that can move. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11779–11788, 2022.

Ondrej Biza, Robert Platt, Jan-Willem van de Meent, Lawson LS Wong, and Thomas Kipf. Binding Actions to Objects in World Models. *arXiv preprint arXiv:2204.13022*, 2022.

Benjamin Bloem-Reddy and Yee Whye Teh. Probabilistic Symmetries and Invariant Neural Networks. *J. Mach. Learn. Res.*, 21:90–1, 2020.

Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. COCO-Stuff: Thing and stuff classes in context. In *Computer vision and pattern recognition (CVPR), 2018 IEEE conference on*. IEEE, 2018.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021.

Mark Chen, Alec Radford, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative Pretraining From Pixels. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020b.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9620–9629, 2021.

Jang Hyun Cho, Utkarsh Mall, Kavita Bala, and Bharath Hariharan. PiCIE: Unsupervised Semantic Segmentation Using Invariance and Equivariance in Clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16794–16804, June 2021.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1201–1210, 2015.

Eric Crawford and Joelle Pineau. Exploiting Spatial Invariance for Scalable Unsupervised Object Tracking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:3684–3692, 04 2020. doi: 10.1609/aaai.v34i04.5777.

Andrea Dittadi, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, and Francesco Locatello. Generalization and robustness implications in object-centric learning. In *Proceedings of the International Conference in Machine Learning (ICML)*, 2022.

Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. *Advances in neural information processing systems*, 29, 2016.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Gamaleldin F Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael C Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. *arXiv preprint arXiv:2206.07764*, 2022.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkxfaTVFwH.

SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *Advances in Neural Information Processing Systems*, 29, 2016.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Unsupervised Semantic Segmentation by Contrasting Object Mask Proposals. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10032–10042, 2021.

Wouter Van Gansbeke, Simon Vandenhende, and Luc Van Gool. Discovering Object Masks with Transformers for Unsupervised Semantic Segmentation. *ArXiv*, abs/2206.06363, 2022.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=mLcmdlEUxy-.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T. Freeman. Unsupervised Semantic Segmentation by Distilling Feature Correspondences. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=SaKO6z6Hl0c.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16000–16009, June 2022.

Haiyang Huang, Zhi Chen, and Cynthia Rudin. SegDiscover: Visual Concept Discovery via Unsupervised Semantic Segmentation. *ArXiv*, abs/2204.10926, 2022.

Olivier J. Hénaff, Skanda Koppula, Evan Shelhamer, Daniel Zoran, Andrew Jaegle, Andrew Zisserman, João Carreira, and Relja Arandjelovi'c. Object discovery and representation networks. *ArXiv*, abs/2203.08777, 2022.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Andrew Brock, Evan Shelhamer, Olivier J. H'enaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver io: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations*, 2022.

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9865–9874, 2019.

Jindong Jiang, Sepehr Janghorbani, Gerard de Melo, and Sungjin Ahn. SCALOR: Generative World Models with Scalable Object Representations. In *Proceedings of ICLR 2020*. OpenReview.net, 2020. URL https://openreview.net/pdf?id=SJxrKgStDH.

Daniel Kahneman, Anne Treisman, and Brian J Gibbs. The reviewing of object files: Object-specific integration of information. *Cognitive psychology*, 24(2):175–219, 1992.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas Kipf, Gamaleldin Fathy Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional Object-Centric Learning from Video. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=aD7uesX1GF_.

Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential Attend, Infer, Repeat: Generative Modelling of Moving Objects. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/7417744a2bac776fabe5a09b21c707a2-Paper.pdf.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. doi: https://doi.org/10.1002/nav.3800020109. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. SPACE: Unsupervised Object-Oriented Scene Representation via Spatial Attention and Decomposition. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkl03ySYDH.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-Centric Learning with Slot Attention. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11525–11538. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/8511df98c02ab60aea1b2356c013bc0f-Paper.pdf.

Davide Mambelli, Frederik Träuble, Stefan Bauer, Bernhard Schölkopf, and Francesco Locatello. Compositional Multi-Object Reinforcement Learning with Linear Relation Networks. *arXiv preprint arXiv:2201.13388*, 2022.

Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Deep Spectral Methods: A Surprisingly Strong Baseline for Unsupervised Semantic Segmentation and Localization. In *CVPR*, 2022.

Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The Role of Context for Object Detection and Semantic Segmentation in the Wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing Scenes As Compositional Generative Neural Feature Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11453–11464, June 2021.

Yassine Ouali, Céline Hudelot, and Myriam Tami. Autoregressive Unsupervised Image Segmentation. *ArXiv*, abs/2007.08247, 2020.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley D. Edwards, Nicolas Manfred Otto Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *ArXiv*, abs/2205.06175, 2022.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Oriane Simeoni, Gilles Puy, Huy V. Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing Objects with Self-Supervised Transformers and no Labels. In *BMVC*, 2021.

Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate DALL-E Learns to Compose. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=h0OYV0We3oh.

Gautam Singh, Yi-Fu Wu, and Sungjin Ahn. Simple Unsupervised Object-Centric Learning for Complex and Naturalistic Videos. *ArXiv*, abs/2205.14065, 2022b.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

Manuel Traub, Sebastian Otte, Tobias Menge, Matthias Karlbauer, Jannik Thümmel, and Martin V Butz. Learning What and Where–Unsupervised Disentangling Location and Identity Tracking. *arXiv preprint arXiv:2205.13349*, 2022.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. In *NIPS*, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, pp. 1439–1456. PMLR, 2020.

Huy V. Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *ECCV*, 2020.

Huy V. Vo, Elena Sizikova, Cordelia Schmid, Patrick Pérez, and Jean Ponce. Large-Scale Unsupervised Object Discovery. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, 2021.

Yangtao Wang, Xi Shen, Shell Xu Hu, Yuan Yuan, James L. Crowley, and Dominique Vaufreydaz. Self-Supervised Transformers for Unsupervised Object Discovery Using Normalized Cut. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14543–14553, June 2022.

Nicholas Watters, Loïc Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial Broadcast Decoder: A Simple Architecture for Learning Disentangled Representations in VAEs. *ArXiv*, abs/1901.07017, 2019.

Marissa A Weis, Kashyap Chitta, Yash Sharma, Wieland Brendel, Matthias Bethge, Andreas Geiger, and Alexander S Ecker. Benchmarking Unsupervised Object Representations for Video Sequences. *J. Mach. Learn. Res.*, 22:183–1, 2021.

Xin Wen, Bingchen Zhao, Anlin Zheng, X. Zhang, and Xiaojuan Qi. Self-Supervised Visual Representation Learning with Semantic Grouping. *ArXiv*, abs/2205.15288, 2022.

Jiahao Xie, Xiaohang Zhan, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Unsupervised object-level representation learning from scene images. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=X2K8KVEaAXG.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On Layer Normalization in the Transformer Architecture. In *ICML*, 2020.

Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and X. Wang. GroupViT: Semantic Segmentation Emerges from Text Supervision. *ArXiv*, abs/2202.11094, 2022.

Jianwei Yang, Jiayuan Mao, Jiajun Wu, Devi Parikh, David D Cox, Joshua B Tenenbaum, and Chuang Gan. Object-centric diagnosis of visual reasoning. *arXiv preprint arXiv:2012.11587*, 2020.

Andrii Zadaianchuk, Maximilian Seitzer, and Georg Martius. Self-supervised Visual Reinforcement Learning with Object-centric Representations. In *International Conference on Learning Representations*, 2020.

Andrii Zadaianchuk, Matthaeus Kleindessner, Yi Zhu, Francesco Locatello, and Thomas Brox. Unsupervised Semantic Segmentation with Self-supervised Object-centric Representations. *ArXiv*, abs/2207.05027, 2022.

APPENDIX

A EXTENDED RELATED WORK

Unsupervised Semantic Segmentation. The challenging task of decomposing natural scenes into semantically meaningful regions without any human annotations can be tackled from different directions. One line of work (Ji et al., 2019; Ouali et al., 2020; Cho et al., 2021; Gansbeke et al., 2021; 2022; Hamilton et al., 2022; Zadaianchuk et al., 2022) first learns a dense representation for each pixel in the image. Such dense representations can then be clustered to form discrete categories and thus allow to decompose the scene to semantically coherent regions. Methods that try to learn and cluster dense representations without additional assumptions (Ji et al., 2019; Ouali et al., 2020) were shown to be effective only on small-scale datasets with narrow visual domains. Furthermore, incorporation of geometric consistency (Cho et al., 2021) or unsupervised saliency detection (Gansbeke et al., 2022; Zadaianchuk et al., 2022) simplifies dense representation learning and the grouping task. Additionally, using self-supervised DINO representations (Caron et al., 2021) by contrasting them (Hamilton et al., 2022) or to self-train a semantic segmentation model (Zadaianchuk et al., 2022; Gansbeke et al., 2022; Melas-Kyriazi et al., 2022) allows obtaining a more accurate image decomposition to semantic categories.

Object Localization is another research direction that also aims at structuring the scene by predicting object information such as bounding boxes and object category from images. Recent works (Vo et al., 2020; 2021; Simeoni et al., 2021; Melas-Kyriazi et al., 2022; Wang et al., 2022) showed the potential benefit of using networks pre-trained in a supervised self-supervised way for both object localization (i.e, bounding box prediction) and object clustering. First, rOSD (Vo et al., 2020; 2021) was using supervised classifier features to localize salient objects in the image. Next, LOST (Simeoni et al., 2021) showed that dense DINO features could be used for localization of the most prominent object in the image. They proposed some heuristic estimate of the object seed and showed that patches that correlated with the seed patch can form an object proposal mask. Finally, DeepSpectral (Melas-Kyriazi et al., 2022) and TokenCut (Wang et al., 2022) showed that additional transformation of the original DINO features to a spectral embedding can improve the quality of the object proposals significantly. While the proposed methods can be combined with self-training of object detectors like R-CNN, they are limited by the quality of the set of salient objects used as pseudo-labels. In contrast, `DINOSAUR` splits each scene to a set of object proposals enabling multi-object scene decomposition without an additional recognition network.

Architectures making use of slot-like components have also enjoyed popularity aside from the research explicitly focused on object-centric learning. For instance, the Perceiver model learns to distill a large set of input tokens from different modalities into a reduced set of embeddings, which can be seen as slots (Jaegle et al., 2022). Prominent recent successes of large-scale multi-modal modeling such as Flamingo (Alayrac et al., 2022) or Gato (Reed et al., 2022) are based on the Perceiver architecture. In dynamics modeling, recurrent independent mechanisms (Goyal et al., 2021) use a competition between independent modules to read inputs into a slot-structured latent state.

Recent work on contrastive learning such as SlotCon (Wen et al., 2022) or Odin (Hénaff et al., 2022) also can be seen as learning slot representations, although with a focus on semantic (class-based) slots rather than instances. In the case of SlotCon, this is due to the reliance on prototypes which naturally lead to semantically focused representations whereas in the case of Odin, the contrastive objective does not encourage differentiation of one or multiple instances. Further, Odin is positioned as a strategy to pre-train a backbone which can then be finetuned on downstream tasks and is not intended to be an unsupervised method that directly leads to instance-level grouping. Note that even though Odin also evaluates object discovery on COCO using the mBO metric, the obtained numbers are not comparable to ours as Odin generates 255 proposal masks by running k-means 128 times with different numbers of clusters. Finally, ORL (Xie et al., 2021) addresses the reliance of many contrastive methods on ImageNet-specific biases, which leads to intra-image contrasting of the same object under different augmentations. This is done by identifying correspondences of objects in different images using models trained via intra-image contrasting and subsequently training the model using inter-image contrasting by selecting objects of the same type from different images.

Figure 6: Finetuning a DINO pre-trained ViT encoder using image reconstruction as a training signal fails to yield meaningful results and results in striped mask patterns (see discussion in Sec. 4.4).

B Additional Results

In this section, we include several experiments that did not fit into the main part. First, Table 8 and Table 9 complement the encoder and decoder analysis from Sec. 4.4 with results on object localization and unsupervised semantic segmentation. We also refer to App. E, where example model predictions for these analyses are included. Second, in Sec. B.1, we discuss how the number of slots affects the quantitative and qualitative results of our method. Third, in Sec. B.2, we compare the effect of training the encoder from scratch vs. using a pre-trained encoder. Fourth, in Sec. B.4, we analyze how the Transformer decoder behaves with different encoders. Fifth, in Sec. B.5, we compare the performance of masks from slot attention and the Transformer decoder. Sixth, in Sec. B.6, we investigate the question of whether or not our method performs semantic segmentation on the MOVi datasets. Seventh, in Sec. B.7, we add an unsupervised segmentation baseline for object discovery on the MOVi datasets. Finally, in Sec. B.8, we evaluate the usefulness of the learned slot representations for downstream tasks by predicting object properties from them on COCO.

B.1 Number of Slots

MOVi-E (Table 10) On the synthetic MOVi-E dataset, we found that changing the number of slots can greatly impact performance. We attribute this to the strong impact object splitting (distributing an object over two or more slots) can have on the ARI score. While setting the number of slots to the maximum number of objects that can appear in the scene (23 on MOVi-E) allows the model to fully cover crowded scenes, most scenes have fewer objects, leading to impaired performance on those. We show examples on MOVi-E with 11 slots for all tested methods in Fig. 17.

PASCAL (Table 11) & COCO (Fig. 8, Fig. 9) On both PASCAL VOC 2012 and COCO, our method performs well over a range of slots, as long as a minimum number of slots is given. Slot

Table 8: Extended results of encoder analysis (Table 6) (mean ± standard dev., 3 seeds), using the Transformer decoder and 7 slots. For discovery, results are given on COCO. For localization, results are given on COCO-20k. For segmentation, results are given for object segmentation (COCO) and scene decomposition (Stuff-27, Stuff-171).

		Object Discovery			**Object Localization**		**Semantic Segmentation (mIoU)**		
Encoder	Pretraining	ARI	mBO^i	mBO^c	CorLoc	DetRate	COCO	Stuff-27	Stuff-171
ResNet50	Supervised	35.1 ±0.1	21.0 ±0.0	23.6 ±0.1	44.2 ±0.6	15.0 ±0.3	8.7 ±0.3	13.8 ±0.2	7.6 ±0.1
ResNet50	DINO	35.1 ±0.6	21.8 ±0.3	24.6 ±0.3	44.3 ±2.9	15.4 ±1.3	6.5 ±0.1	12.8 ±0.1	6.3 ±0.0
ResNet50	MoCo-v3	32.6 ±0.4	20.0 ±0.2	22.6 ±0.2	35.2 ±1.0	11.2 ±0.5	5.5 ±0.2	11.8 ±0.1	5.4 ±0.1
ViT-S/16	Supervised	25.5 ±0.4	22.8 ±0.2	29.5 ±0.1	56.2 ±0.7	24.5 ±0.3	10.5 ±0.2	16.0 ±0.2	9.5 ±0.2
ViT-S/16	DINO	36.3 ±0.8	28.0 ±0.7	32.9 ±1.0	68.6 ±1.9	30.2 ±1.3	11.9 ±0.1	20.5 ±0.2	10.7 ±0.3
ViT-B/16	Supervised	31.7 ±1.0	24.9 ±0.3	30.8 ±0.3	59.1 ±1.1	25.5 ±0.4	10.7 ±0.4	17.9 ±0.2	10.1 ±0.2
ViT-B/16	DINO	33.4 ±1.0	29.6 ±0.6	38.7 ±0.9	67.2 ±1.5	31.1 ±0.9	14.6 ±0.2	24.1 ±0.7	13.2 ±0.2
ViT-B/16	MoCo-v3	34.1 ±0.4	29.5 ±0.1	37.5 ±0.4	70.0 ±0.5	32.3 ±0.1	11.7 ±0.3	20.7 ±0.5	10.6 ±0.1
ViT-B/16	MSN	34.5 ±0.2	29.4 ±0.2	36.4 ±0.4	69.7 ±0.7	31.9 ±0.4	12.9 ±0.3	20.6 ±0.6	12.2 ±0.2
ViT-B/16	MAE	32.3 ±3.2	28.1 ±1.8	32.3 ±1.7	68.9 ±5.2	29.8 ±3.0	6.2 ±0.1	9.9 ±0.4	5.7 ±0.1
ViT-S/8	DINO	34.0 ±0.6	30.6 ±0.2	37.8 ±0.2	73.0 ±0.2	33.8 ±0.0	15.1 ±0.1	24.7 ±0.4	13.3 ±0.2

Table 9: Additional Decoder Analysis Results for Fig. 7 on *Object Localization* and *Unsupervised Semantic Segmentation* (mean ± standard dev., 3 seeds), using a ViT-B/16 encoder and 6 slots (PASCAL VOC 2012) or 7 slots (COCO). For localization, results are given on PASCAL VOC 2012 `trainval` (training on PASCAL VOC 2012) and COCO-20k (training on COCO). For segmentation, results are given for object segmentation (PASCAL VOC 2012, COCO) and scene decomposition (Stuff-27, Stuff-171); when the training dataset is different from the evaluation dataset, this tests zero-shot transfer (i.e. PASCAL → COCO and COCO → PASCAL).

		Object Localization		**Semantic Segmentation (mIoU)**			
Training	Decoder	CorLoc	DetRate	COCO	Stuff-27	Stuff-171	PASCAL
PASCAL	MLP	76.0 ±0.3	56.3 ±0.2	12.6 ±0.2	21.1 ±0.1	11.5 ±0.1	31.7 ±0.3
	Transformer	70.4 ±2.5	52.6 ±2.0	11.7 ±0.4	22.6 ±0.5	11.4 ±0.3	35.3 ±1.6
COCO	MLP	69.3 ±0.3	29.8 ±0.2	11.9 ±0.1	20.1 ±0.2	11.6 ±0.1	31.1 ±0.5
	Transformer	67.2 ±1.5	31.1 ±0.9	14.6 ±0.2	24.1 ±0.7	13.2 ±0.2	35.9 ±0.9

Attention, in contrast, fails to produce meaningful results for all tested number of slots. For our method on COCO, the analysis shows that there are different sweet spots for instance-level (9–11 slots) and class-level segmentation (7 slots). Visual inspection (see Fig. 7) suggests that the optimal number of slots (unsurprisingly) depends on the complexity of the specific image: using few slots tends to lead to grouping of related objects, whereas using many slots leads to oversegmentation of objects into parts. Choosing the number of slots per-dataset rather than per-image thus only optimizes results for the average scene. An interesting question for future work is how models could deal with the variance of visual complexity and number of instances that they are faced with in real-world situations.

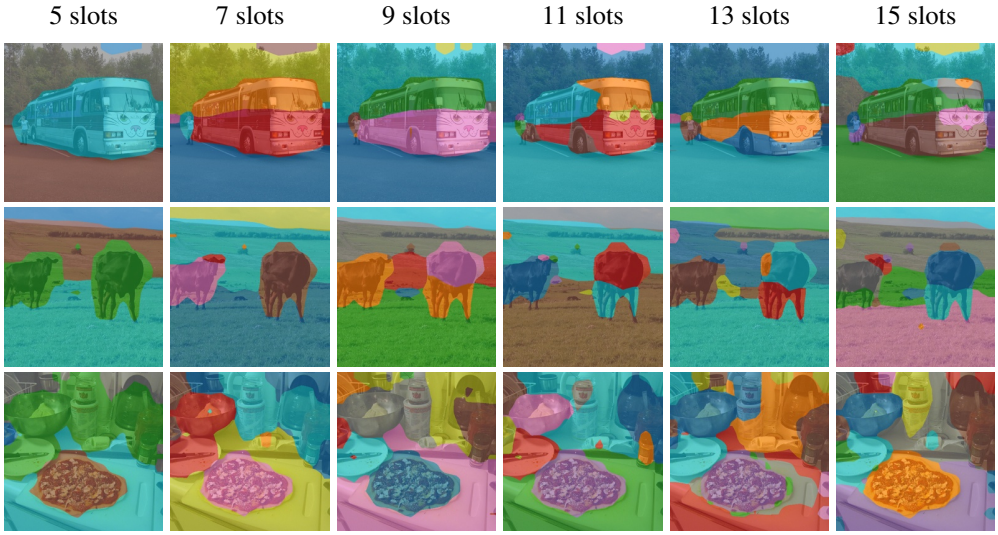

Figure 7: Three scenes of low/medium/high complexity on COCO 2017, varying the number of slots the model is trained with, using a ViT-B/16 encoder and the Transformer decoder. For an appropriate grouping to emerge, the number of slots has to fit the complexity of the scene. However, the groupings found by our method are still semantically meaningful, grouping together related objects (too few slots) or separating objects into their constituents parts (too many slots).

B.2 Training the Encoder from Scratch

In this experiment, we attempt to train an encoder from scratch instead of using a pre-trained, frozen encoder. As discussed in Sec. 4.4, we were not able to train a *ViT encoder* from scratch using feature reconstruction as the objective. ViTs are known to be difficult to optimize, requiring careful training recipes. Combining ViT training with the slot attention module might thus also need non-trivial efforts to train successfully. Here, instead of a ViT, we train a *ResNet34* backbone (the same ResNet

Table 10: Number of slots analysis on *MOVi-E object discovery* (mean over 3 seeds). All methods perform better when training with less slots.

	11 slots		24 slots	
	ARI	mBO	ARI	mBO
Block Pattern	*46.6*	*13.4*	*41.8*	*18.6*
Slot Attention	52.6	16.6	45.0	23.0
SLATE	47.5	14.1	41.9	17.8
Ours (ViT-S/8)	76.7	30.6	64.6	32.0
Ours (ViT-B/8)	79.2	34.0	65.1	33.5

Table 11: Number of slots analysis on *PASCAL VOC 2012 object discovery* (mean ± standard dev., 3 seeds), using a ViT-B/16 encoder and the Transformer decoder.

Slots	ARI	mBO^i	mBO^c
4	29.5 ±1.7	33.9 ±1.0	40.1 ±1.2
5	26.8 ±1.4	38.6 ±0.6	45.5 ±0.7
6	23.2 ±0.9	43.6 ±0.8	50.8 ±1.0
7	19.5 ±0.7	42.5 ±0.7	47.7 ±0.6
9	19.9 ±1.0	43.0 ±0.3	47.1 ±0.5

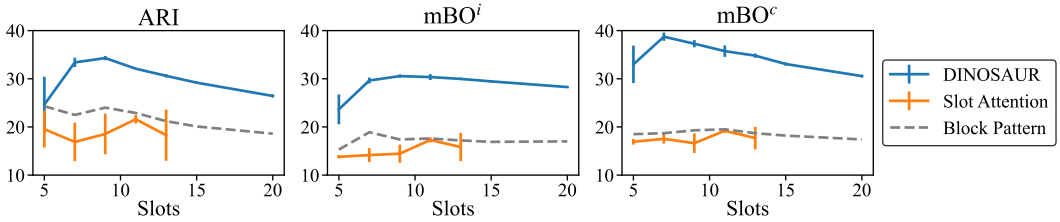

Figure 8: Number of slots analysis on *COCO object discovery*, showing mean and std. over 3 seeds, using a ViT-B/16 encoder and the Transformer decoder. Our method shows a minor trade-off between instance- (more slots) and class-level segmentation (less slots). Slot attention fails to produce meaningful masks over a range of slots.

architecture we use for our Slot Attention baseline) for 300 000 steps. As the reconstruction target, we still use features from a pre-trained ViT. Notably, we were not able to optimize the ResNet from scratch using the Transformer decoder, as the model degraded to only use a single slot for the full image. Instead, we used the MLP decoder. For each dataset and type of target features, we compare with using the target feature network also as a frozen, pre-trained encoder (i.e. our usual setup).

Table 12 lists the results. Example predictions on COCO are included in Table 21 and Table 22. We find that training from scratch is a viable alternative to using a pre-trained encoder, in some cases improving over the pre-trained encoder. We think this is because training from scratch allows adaptation to the domain, whereas the pre-trained encoder may to some degree be misaligned. On COCO, the improvement may also partially stem from the ResNet's higher spatial resolution (28×28) — although the decoder's mask resolution is the same (14×14). However, using a frozen encoder has computational advantages: no backward pass is needed, and only one forward pass through the encoder is needed (instead of one pass through the encoder being trained, and one pass through the encoder providing the target features). Finally, we find that pre-trained ResNet features are a worse reconstruction target then ViT features. Overall, this experiment shows that the crucial element making our approach work is feature reconstruction. While the type of encoder and the use of pre-trained weights can yield further improvements, they are less important compared to the use of feature reconstruction.

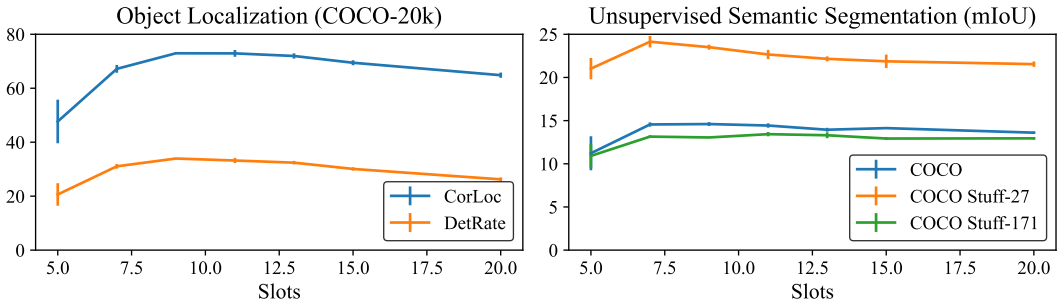

Figure 9: Number of slots analysis on *Object Localization* and *Unsupervised Semantic Segmentation* (mean ± standard dev., 3 seeds), using a ViT-B/16 encoder and the Transformer decoder.

Table 12: Training a ResNet34 encoder *from scratch* vs. using a *pre-trained frozen* ViT encoder on *object discovery* (mean ± standard dev., 3 seeds). Both options use the same target features for reconstruction and the MLP decoder. We find that training from scratch yields overall similar results compared to using a pre-trained encoder.

Dataset	Target Features		Encoder	Training	ARI	$\text{mBO}^{(i,c)}$	
MOVi-C	DINO	ViT-S/8	ResNet34	Scratch	65.2 ±2.1	27.5 ±1.9	
			ViT-S/8	Frozen	67.0 ±0.4	35.2 ±0.2	
MOVi-E	DINO	ViT-S/8	ResNet34	Scratch	64.5 ±0.5	37.1 ±0.2	
			ViT-S/8	Frozen	64.6 ±0.7	32.0 ±0.1	
PASCAL	DINO	ViT-B/16	ResNet34	Scratch	27.5 ±0.7	38.0 ±0.5	38.9 ±0.6
			ViT-B/16	Frozen	24.6 ±0.2	39.3 ±0.1	40.8 ±0.1
COCO	DINO	ResNet50	ResNet34	Scratch	35.9 ±0.4	24.9 ±0.1	28.2 ±0.1
			ResNet50	Frozen	35.1 ±0.6	21.8 ±0.3	24.6 ±0.3
		ViT-B/16	ResNet34	Scratch	39.9 ±0.3	26.5 ±0.1	30.4 ±0.0
			ViT-B/16	Frozen	39.4 ±0.0	26.1 ±0.1	30.0 ±0.2
	MoCo-v3	ResNet50	ResNet34	Scratch	31.3 ±1.1	22.6 ±0.5	26.0 ±0.4
			ResNet50	Frozen	32.6 ±0.4	20.0 ±0.2	22.6 ±0.2
		ViT-B/16	ResNet34	Scratch	39.6 ±0.4	26.6 ±0.1	30.3 ±0.1
			ViT-B/16	Frozen	38.7 ±0.1	26.0 ±0.1	29.8 ±0.0
	MSN	ViT-B/16	ResNet34	Scratch	40.0 ±0.1	26.1 ±0.1	30.0 ±0.1
			ViT-B/16	Frozen	38.1 ±0.1	26.6 ±0.0	30.8 ±0.0
	MAE	ViT-B/16	ResNet34	Scratch	36.8 ±0.1	26.5 ±0.1	30.9 ±0.0
			ViT-B/16	Frozen	41.5 ±0.3	27.6 ±0.2	31.4 ±0.1

B.3 Extended Decoder Analysis

Extending our decoder analysis from Sec. 4.4, we investigate how the capacity of the decoder and the slot dimensionality affects the object discovery performance on COCO. In particular, we compare the MLP with the Transformer decoder: for the MLP decoder, we use a 4-layer MLP and test hidden layer dimensionalities 1024, 2048, and 4096. For the Transformer decoder, we test different layer numbers 2, 3, and 4. For each decoder setting, we test it under slot dimensionalities 64, 128, 256, and 512.

We show the results in Fig. 10. The general trend of Table 7 stays unchanged: the MLP decoder has higher ARI (indicating better instance grouping), whereas the Transformer decoder has higher mBO (indicating better mask quality). For the MLP decoder, we find that scaling the MLP decoder has slight benefits in terms of ARI, and that it is robust to the slot dimensionality (with the exception of slot dimensionality 512). For the Transformer decoder, scaling appears to have little impact. However, we find that the Transformer decoder is sensitive to the slot dimensionality: setting it too low (64, 128) can cause the model to collapse to just use one or two slots to model the whole image. Setting it too high (512) causes training instabilities (i.e. NaNs or loss spikes). These instabilities under slot dimensionality 512 also occur with the MLP decoder. We conjecture this might be related to the use of 16-bit mixed precision and the attention mechanism of slot attention, but we did not investigate further.

B.4 Transformer Decoder with Different Encoders

In Table 13, we compare the Transformer decoder with different ViT encoders on COCO object discovery. We find that scaling up the encoder leads to better results. However, specifically combining the Transformer decoder with the ViT-B/8 encoder leads to poor results – the MLP decoder has no such issues (cf. the results on the MOVi datasets in Table 1). Visual inspection shows that the model collapses to use just one or two slots for the whole image. We attribute this to the *high capacity of the decoder*, which leads to it being able to reconstruct the token map well while using the conditioning information from only a few slots. Note that the capacity of the decoder is related to the encoder through the token dimensionality, and the number of tokens (which has strong influence on a

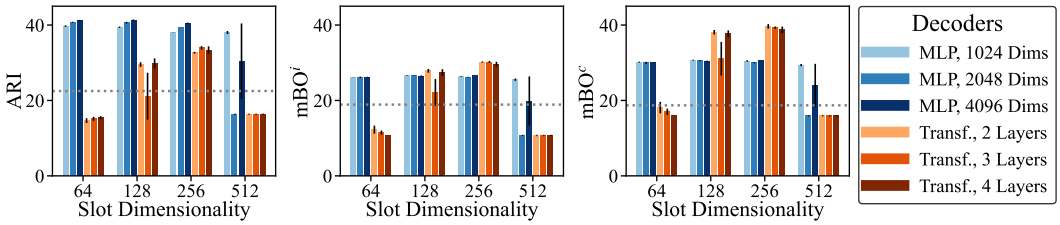

Figure 10: *COCO object discovery* performance when varying decoder capacity and slot dimensionality, using 7 slots (mean ± standard dev., 3 seeds). For the MLP decoder, we vary the dimensionality of a 4-layer MLP, whereas for the Transformer decoder, we vary the number of layers. The dotted line represents the block pattern baseline. The analysis reveals instabilities of the Transformer decoder: for too small or too large slot dimensionalities, the performance degrades below the block pattern baseline. The MLP decoder is generally robust, but also becomes unstable at slot dimensionality 512.

Transformer's capacity as it directly influences the amount of memory and computation available throughout the blocks). Indeed, when reducing the decoder's capacity by either reducing the feature dimensionality (ViT-S/8) or the number of tokens the decoder operates on (using bicubic interpolation on the feature map to reconstruct), the problem disappears. In principle, the issue could also arise from optimization becoming unstable due to switching to a larger model, but the loss curves do not suggest such instabilities.

Table 13: Comparing different encoders with the Transformer decoder on *COCO object discovery*. For ViT-B/8, we also list results for *scaling the target token map* by a factor of 0.5 (from 784 to 196 tokens) and by a factor of 0.75 (to 441 tokens). Object-centricness does no longer emerge for ViT-B/8, which we attribute to the high capacity of the decoder: when reducing the capacity (by reducing the feature dimensionality or the number of tokens), the problems disappear.

Encoder	Target Tokens	D_{feat}	ARI	mBO^i	mBO^c
ViT-S/16	196	384	36.3 ±0.8	28.0 ±0.7	32.9 ±1.0
ViT-B/16	196	768	33.4 ±1.0	29.6 ±0.6	38.7 ±0.9
ViT-S/8	784	384	34.0 ±0.6	30.6 ±0.2	37.8 ±0.2
ViT-B/8	196 (scaled)	768	35.3 ±0.4	30.3 ±0.6	37.6 ±0.2
ViT-B/8	441 (scaled)	768	19.2 ±10.0	17.2 ±8.9	23.0 ±9.9
ViT-B/8	784	768	12.5 ±3.0	14.5 ±3.8	20.7 ±5.0

B.5 Mask Type

As noted in Sec. 3, the Transformer decoder offers a choice between attention masks from the slot attention module, and attention masks from the decoder. In Table 14, we compare the two types of masks and find that decoder attention masks are consistently better on all tasks, and especially outperform the masks from slot attention on segmentation. Note that we could in principle also use masks from slot attention with the MLP decoder. We point to Singh et al. (2022b), who analyze this choice and find that alpha masks produced by reconstruction are superior to attention masks.

Table 14: Comparing masks from Slot Attention and the Transformer decoder on *COCO* (mean ± standard dev., 5 seeds), using a ViT-B/16 encoder. For segmentation, we list results for object segmentation (COCO) and scene decomposition (Stuff-27, Stuff-171).

	Object Discovery			**Object Localization**		**Segmentation (mIoU)**		
Mask Type	ARI	mBO^i	mBO^c	CorLoc	DetRate	COCO	Stuff-27	Stuff-171
Slot Attention	31.8 ±0.9	28.8 ±0.3	37.6 ±0.4	66.0 ±1.5	30.2 ±0.8	11.6 ±0.2	18.9 ±0.2	10.6 ±0.2
Decoder	33.4 ±1.0	29.6 ±0.6	38.7 ±0.9	67.2 ±1.5	31.1 ±0.9	14.6 ±0.2	24.1 ±0.7	13.2 ±0.2

B.6 DOES DINOSAUR PERFORM SEMANTIC GROUPING ON MOVI?

On the MOVi datasets, DINOSAUR seemingly separates objects into slots, i.e. instance grouping. As DINOSAUR exhibits some bias towards semantic grouping on real-world datasets, an interesting question is whether our approach does in fact also perform semantic grouping on MOVi. If this would be the case, it would be hard to detect by evaluating on the original MOVi datasets, because objects of the same category rarely appear on the same image on those datasets.

To answer this question, we designed two variants of the MOVi-C dataset. In the first, only objects of *the same semantic category* appear in each image. In the second, each image contains *the exact same object* several times. All other characteristics of the dataset stay unchanged. If our method would perform semantic grouping on MOVi, objects would be grouped together on those datasets. In Table 15, we list the results for training on MOVi-C and testing on the new "semantic" variants. We find that the results do not change significantly compared to evaluating on the original MOVi-C dataset, indicating that our method does in fact perform instance grouping on the MOVi datasets.

Table 15: Testing a model trained on MOVi-C on two variants of MOVi-C: in "semantic objects", each image only contains objects of the same semantic category; in "duplicate objects", each image only contains the same object, but several times. We show results for our method with ViT-B/8 encoder and MLP decoder, with mean ± standard dev. over 5 seeds. We find that the results do not change significantly, indicating that DINOSAUR does not apply semantic grouping on MOVi.

Dataset	FG-ARI	mBO
MOVi-C	68.9 ±0.4	38.0 ±0.2
Semantic Objects	68.6 ±0.3	36.3 ±0.3
Duplicate Objects	68.0 ±0.4	36.1 ±0.3

B.7 UNSUPERVISED SEMANTIC SEGMENTATION BASELINE ON MOVI

An interesting question is to what degree unsupervised semantic segmentation methods (which we compare to on real-word data in Sec. 4.3) can already solve the object discovery task on synthetic datasets designed to benchmark object-centric learning methods. If this were the case, it would showcase that those datasets are flawed in the sense that instance-level grouping (one goal of object-centric learning) is not needed to tackle them. To this end, we test the state-of-the-art scene segmentation method STEGO (Hamilton et al., 2022) on the MOVi datasets.

Table 16: Results for STEGO on the MOVi datasets.

Dataset	Clusters	ARI	mBO
MOVi-C	22	39.9	29.7
	27	39.3	29.4
MOVi-E	22	41.1	15.3
	27	41.8	15.8

STEGO assigns each pixel to one of several clusters/classes, where the clustering is performed over the full dataset. We run STEGO with 22 and 27 clusters, corresponding to the 17 semantic categories of the MOVi dataset (see Sec. D), plus 5 or 10 clusters which could be used for the backgrounds. We use a DINO ViT-B/8 backbone for feature extraction, other hyperparameters are listed in Sec. C.2. To evaluate on the object discovery task, we use the segmentation masks produced by STEGO as if they were object masks. That is, each cluster/class mask is used as a separate object mask.

We list quantitative results in Table 16 and show examples in Fig. 11. STEGO is able to segment objects and background to some degree. However, objects are not properly split, with a single cluster often covering several objects from different semantic categories. Compared to our method, STEGO's masks are also significantly more cluttered and less geometrically consistent. Quantitatively, on MOVi-C, STEGO performs slightly better than our Slot Attention baseline on mBO, but significantly worse on the ARI metric (reflecting the poor object splitting behavior). On MOVi-E, STEGO's mBO is also worse than Slot Attention's, indicating that STEGO struggles with complex scenes of many smaller objects. Overall, this result suggests that a current state-of-the-art unsupervised semantic

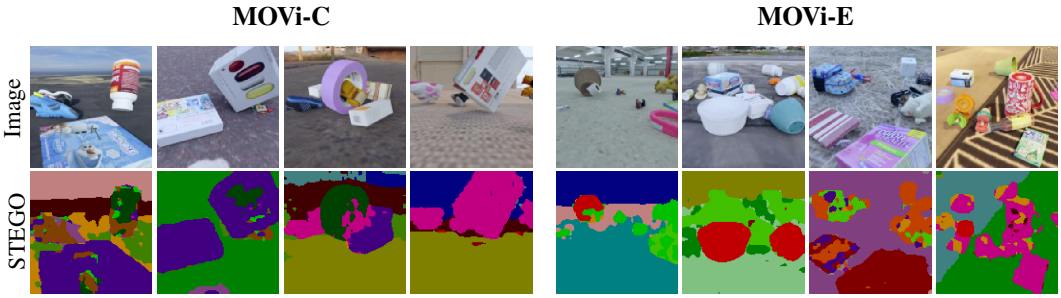

Figure 11: Running the unsupervised semantic segmentation method STEGO (Hamilton et al., 2022) on the MOVi datasets, using 22 clusters for MOVi-C and 27 clusters for MOVi-E. Color codes correspond to the different clusters. Results are presented without CRF post-processing.

segmentation method can not satisfactorily solve the object discovery task, justifying the existence of datasets like MOVi to study object-centric learning methods.

B.8 Object Property Prediction on COCO

We study the usefulness of the learned representations for downstream applications by predicting *object properties* from the slots on COCO. To this end, we supervise shallow predictors using the slot representations from unsupervised object-centric models. As properties, we use the class, and the x- and y-coordinates of each object (center-of-mass).

We closely follow the experimental design of Dittadi et al. (2022), and did not attempt to tune their settings to optimize for performance. As downstream models, we test a linear predictor, and a one-hidden layer MLP. Each model is applied in parallel to all slots for an image, i.e. it receives a slot as input, and outputs a vector of property predictions for this slot. We use the cross-entropy loss for the categorical class prediction, and the MSE loss for the continuous x- and y-coordinates, where the overall loss is just the sum of the individual losses. We match the predicted set of vectors to the target set of vectors using the Hungarian method (Kuhn, 1955), minimizing the total loss of the assignment. As metrics, we report Top-1 and Top-5 average-class accuracy for the 80-way class prediction and R^2 (coefficient of determination) for the x- and y-coordinates.

We use 10 000 random images from the COCO dataset to train, and another 1 000 random images as a validation set. We report metrics on the 5 000 validation images of COCO. Images that contain no objects due to center cropping are filtered out. As COCO images can contain many objects, but our method typically only uses a low number of slots (e.g. 7), we keep only the 6 largest objects in the image. This way, all ground truth objects can in principle be matched by a slot. We train for 15 000 steps, and use the model with the lowest validation loss for testing.

The results are listed in Fig. 12. We compare `DINOSAUR` using a DINO-pretrained ViT-B/16 encoder with a Slot Attention model, all using 7 slots. On class prediction, our method has 25% higher accuracy than Slot Attention. On coordinate prediction, `DINOSAUR` reaches similar performance to Slot Attention on the x-coordinate, but exceeds it on the y-coordinate, where Slot Attention drops-off. We also find that on coordinate prediction, the representations from the Transformer decoder are a bit worse than from the MLP decoder. This may be because the Transformer decoder often captures several disconnected objects in one slot, making it more difficult to predict the center-of-mass. Using a non-linear instead of a linear predictor slightly improves accuracy on all properties; using more hidden layers than one did not improve the performance further in our experiments. Overall, we conclude that the representation `DINOSAUR` learns are more useful than Slot Attention's for downstream tasks on real-world data. However, our method also still leaves room for improvement. Finally, we note that this experiment only constitutes a first step in evaluating the learned representations, and that further investigations could be conducted, e.g. taking into account robustness to distribution shifts.

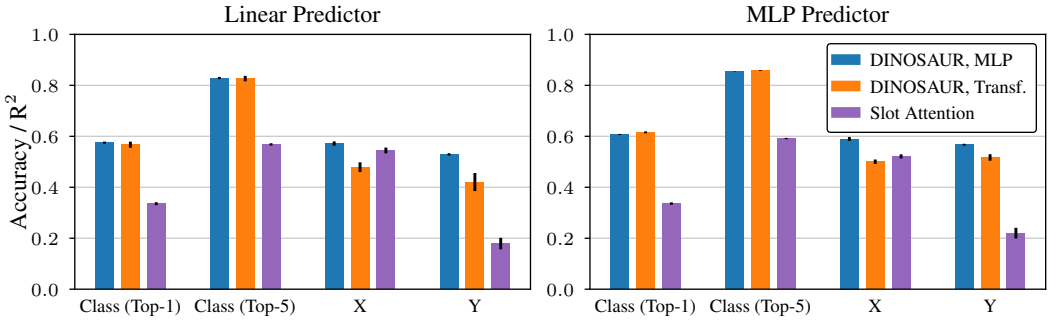

Figure 12: *Object property prediction* on the validation set of COCO, showing mean and standard deviation over 3 seeds. We train a linear predictor (left) and a one-hidden layer MLP (right) to predict object properties from the slot representations. For class prediction, we show Top-1 and Top-5 accuracy. For coordinate prediction, we show R^2 (coefficient of determination, higher is better). We compare our method using MLP and Transformer decoders with Slot Attention.

C Implementation Details

C.1 Architecture and Hyperparameters

In the following, we describe implementation details of the `DINOSAUR` architecture. The hyperparameters for our experiments are given in Table 17.

ViT Encoder We use the Vision Transformer implementation and provided by the timm library[2]. Depending on the experiment, we use the following configurations: ViT-S/16 (token dimensionality 384, 6 heads, patch size 16), ViT-S/8 (token dimensionality 384, 6 heads, patch size 8), ViT-B/16 (token dimensionality 768, 12 heads, patch size 16), ViT-B/8 (token dimensionality 768, 12 heads, patch size 8). The specific timm model names (for DINO weights) used are `vit_small_patch16_224_dino` (ViT-S/16), `vit_small_patch8_224_dino` (ViT-S/8), `vit_base_patch16_224_dino` (ViT-B/16), `vit_base_patch8_224_dino` (ViT-B/8). All models use 12 Transformer blocks, linear patch embedding and additive positional encoding. The output of the last block (not applying the final layer norm) is passed on to the Slot Attention module and used in the feature reconstruction loss, after removing the entry corresponding to the CLS token. We always use images of size 224×224 as input to the ViT; this yields $N = 14^2 = 196$ patches for patch size 16 and $N = 28^2 = 784$ patches for patch size 8. If the image size is not equal to 224, we use bicubic interpolation for resizing. For the pre-trained weights, we use the timm library for DINO and third-party releases for MoCo-v3[3], MSN[4], and MAE[5].

Slot Attention The Slot Attention module largely follows the original formulation from Locatello et al. (2020). After applying a layer norm, the set of N input features is transformed to slot dimensionality D_{slots} by a two-layer MLP with hidden size equal to the feature size D_{feat}, followed by another layer norm. An initial set of K slots is randomly sampled from a normal distribution parametrized by learnable mean $\mu \in \mathbb{R}^{D_{\text{slots}}}$ and log standard deviation $\log \sigma \in \mathbb{R}^{D_{\text{slots}}}$. The slots are then updated over several iterations, where each iteration involves a layer norm step on the current slots, an attention step where slots compete for input features, updating the slots using a GRU (Cho et al., 2014), and applying a residual two-layer MLP. The query/key/value projections' size is the same as the slot dimensionality D_{slots}; they do not use a bias. The key-value dot product is scaled by $D_{\text{slots}}^{-0.5}$, and an epsilon of 10^{-8} is added to the softmax weights for stability reasons. The MLP has a hidden size equal to $4 \cdot D_{\text{slots}}$.

On MOVi-C and MOVi-E, we use 11 and 24 slots respectively, which corresponds to the maximal number of objects that can appear in the scene plus one slot that can capture the background. On

[2] https://github.com/rwightman/pytorch-image-models, v0.6.7
[3] https://github.com/facebookresearch/moco-v3
[4] https://github.com/facebookresearch/msn
[5] https://github.com/facebookresearch/mae

PASCAL VOC and COCO, we use 6 and 7 slots respectively, as we found those values to perform quantitatively well and give visually appealing results (cf. App. B.1).

MLP Decoder We use a four-layer MLP with ReLU activations, with output dimensionality $D_{\text{feat}} + 1$, where the last dimension is used for the alpha value. The MLP has hidden layer sizes of 1024 for the MOVi datasets, and 2048 for COCO. A learnable positional encoding of size $N \times D_{\text{slots}}$ is added to the slots after broadcasting them to the number of patches N.

Transformer Decoder We use the standard decoder design from Vaswani et al. (2017) with pre-normalization (Xiong et al., 2020), conditioning on the set of slots output by the slot attention module. Similar to ImageGPT (Chen et al., 2020a), the decoder autoregressively reconstructs N patch features of dimensionality D_{feat} starting from the top-left and proceeding row-by-row over the image ("raster order"). To this end, its initial input is the set of target features shifted back by one, removing the last feature and inserting a learned "beginning-of-sentence" feature at the start. Each Transformer block then consists of self-attention on the set of tokens (using causal masking to prevent attending to future tokens), cross-attention with the set of slots, and a residual two-layer MLP with hidden size $4 \cdot D_{\text{feat}}$. Before the Transformer blocks, both the initial input and the slots are linearly transformed to D_{feat}, followed by a layer norm. The Transformer thus operates on tokens of dimensionality D_{feat}. The output of the last Transformer block is directly used as the reconstruction $\boldsymbol{y}$. We do not apply dropout as we did not find it to be beneficial. Also, we do not add positional encoding as the target features forming the input already contain positional information (from the ViT encoder).

As mentioned in the main text, we use the attention mask from the decoder as the slot mask used for evaluation. In particular, the cross attention step involves a softmax over the slots for each of the N tokens, resulting in N attention weights per slot, which can be reshaped to form a $\sqrt{N} \times \sqrt{N}$ mask. Intuitively, each mask entry measures how important the corresponding slot was for reconstructing the patch feature at that position. For the evaluation, this mask is bilinearly resized to the image size. We use the attention mask from the last decoder block, which we found to perform best against the alternatives of using masks from earlier blocks or averaging the masks across blocks.

Details for Specific Experiments Here we list settings for experiments aside from our main experiments.

- Importance of Feature Reconstruction. In this experiment, we train COCO and combine a ViT-B/16 encoder with different decoders: Slot Attention's spatial broadcast decoder (as detailed in App. C.2) for image reconstruction and the MLP decoder for feature reconstruction (as detailed above). Image reconstruction uses 128×128 as the target resolution. For finetuning, we use a lower learning rate of $4 \cdot 10^{-5}$ for the encoder, compared to $4 \cdot 10^{-4}$ for Slot Attention and the decoder. For training from scratch with feature reconstruction, the target signal is produced by a separate pre-trained ViT, not the encoder being trained from scratch. For training a ResNet34 encoder from scratch with feature reconstruction, we use the same encoder architecture as for slot attention (described in App. C.2). All models use 7 slots. Other training and architecture settings are the same as for the main experiments.
- Choice of Pre-Training Method. In this experiment, we use different pre-trained networks for feature extraction and reconstruction. The specific timm model names used are `resnet50` (ResNet50, supervised), `vit_small_patch16_224` (ViT-S/16, supervised), `vit_base_patch16_224` (ViT-B/16, supervised), `vit_small_patch16_224_dino` (ViT-S/16, DINO), `vit_base_patch16_224_dino` (ViT-B/16, DINO). For MoCo-v3, MSN and MAE, we use the ViT-B/16 implementation of timm with third-party weight releases (referenced above). All networks were pre-trained on the ImageNet dataset. The ResNet50 encoder uses the features output from the second last block (feature level 3 in timm), resulting in a 1024-dimensional feature map of size 14×14. Like Kipf et al. (2022), we add linear positional encoding to this feature map before Slot Attention. The ViT encoders use the Transformer decoder whereas for the ResNet encoder, we use the MLP decoder as training with the Transformer decoder did not yield good results. All models use 7 slots.
- Choice of Decoder. This experiment uses 11 slots for MOVi-C, 6 slots for PASCAL VOC 2012, and 7 slots for COCO. For the Transformer decoder on MOVi-C, we use 4 attention heads instead of 8, which produced better results.
- Object Property Prediction (Sec. B.8). We train the downstream models for 15 000 steps with the Adam optimizer with a starting learning rate of 10^{-3} and batch size 64. The

learning rate is halved every 2 000 training steps. The non-linear predictor is a one-hidden layer MLP with 256 hidden neurons and leaky ReLU activation (slope 0.01). Continuous targets are standardized by computing mean and variance over the training dataset. We use center crops as input and filter objects that are not contained in the center crop.

Table 17: Hyperparameters for DINOSAUR used for the main results on MOVi-C, MOVi-E, PASCAL VOC 2012 and COCO datasets.

Dataset		**MOVi-C**	**MOVi-E**	**PASCAL VOC 2012**	**COCO**
Training Steps		500k	500k	250k	500k
Batch Size		64	64	64	64
LR Warmup Steps		10000	10000	10000	10000
Peak LR		0.0004	0.0004	0.0004	0.0004
Exp. Decay Half-Life		100k	100k	100k	100k
ViT Architecture		ViT-B	ViT-B	ViT-B	ViT-B
Patch Size		8	8	16	16
Feature Dim. D_{feat}		768	768	768	768
Gradient Norm Clipping		1.0	1.0	1.0	1.0
Image/Crop Size		224	224	224	224
Cropping Strategy		Full	Full	Random	Center
Augmentations		–	–	Random Horizontal Flip	
Image Tokens		784	784	196	196
Decoder	Type	MLP	MLP	Transformer	Transformer
	Layers	4	4	4	4
	Heads	–	–	4	8
	MLP Hidden Dim.	1024	1024	3072	3072
Slot Attention	Slots	11	24	6	7
	Iterations	3	3	3	3
	Slot Dim. D_{slots}	128	128	256	256
	MLP Hidden Dim.	512	512	1024	1024

C.2 Baselines

Block Pattern The block mask patterns are generated by dividing the image into a specified number of columns, then dividing the columns such that a target number of slot masks is reached (see Fig. 13). For less than 9 masks, 2 columns are used, for 9–15 masks, 3 columns, and for more than 15 masks, 4 columns are used. For the block pattern results given in the main text, MOVi-C, PASCAL VOC 2012 and COCO use 11 masks, and MOVi-E 24 masks. The number of masks is consistent with the number of slots used by the Slot Attention baseline on those datasets.

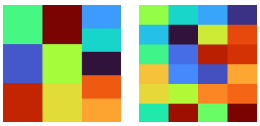

Figure 13: Block patterns for 11 and 24 slots.

Slot Attention The hyperparameters used to run the experiments can be found in Table 18. For the encoder, we use a ResNet34, using the modifications suggested by Kipf et al. (2022) and Elsayed et al. (2022). That is, we modify the basic ResNet34 architecture (`resnet34` in the timm library) to use a stride of 1 in the first convolution, and replace batch normalization with group normalization. This results in a 512-dimensional feature map of size 16×16 for image size 128×128. The slot attention module largely follows the description given in App. C.1 for the DINOSAUR architecture. However, like Kipf et al. (2022), we add a linear positional encoding to the feature map (coordinates in the four cardinal directions scaled to $[-1, 1]$, linearly projected to the feature dimension and added to the features). The number of slots was set to 11 and 24 for MOVi-C and MOVi-E, and 11 for PASCAL and COCO. For PASCAL and COCO, this value was chosen according to what performed best in a scan over several values (cf. Fig. 8); for none of the values, slot attention succeeded in successfully discovering objects on those datasets. For the decoder, we use the same spatial broadcast decoder as (Kipf et al., 2022), i.e. 5 transposed convolutions with 5×5 kernels, channel size 64 and ReLU activations, upsampling from size 8×8 to 128×128. Input and target images are scaled to range $[-1, 1]$.

Table 18: Hyperparameters used for Slot Attention training on MOVi-C, MOVi-E, PASCAL VOC 2012 and COCO datasets.

Dataset		**MOVi-C**	**MOVi-E**	**PASCAL VOC 2012 & COCO**
Training Steps		250k	250k	250k
Batch Size		64	64	64
LR Warmup Steps		2500	2500	2500
Peak LR		0.0002	0.0002	0.0002
Annealing		Cosine	Cosine	Cosine
Gradient Norm Clipping		1.0	1.0	1.0
Image/Crop Size		128	128	128
Cropping Strategy		Full	Full	Random (PASCAL) / Center (COCO)
Augmentations		–	–	Random Horizontal Flip
Slot Attention	Slots	11	24	11
	Iterations	3	3	3
	Slot Dim.	256	256	256
	MLP Hidden Dim.	512	512	512

SLATE We use the official SLATE implementation[6]. The hyperparameters are close to the ones used in SLATE (Singh et al., 2022a) and STEVE (Singh et al., 2022b) and are presented in Table 19. We noticed that SLATE's performance metrics began to degrade significantly after some training time; we picked the training time such that best performance was obtained.

Table 19: Hyperparameters used for SLATE training on MOVi-C, MOVi-E, PASCAL VOC 2012 and COCO datasets.

Dataset		**MOVi-C**	**MOVi-E**	**PASCAL VOC 2012 & COCO**
Training Steps		40k	40k	50k & 40k
Batch Size		64	64	64
LR Warmup Steps		30000	30000	30000
Peak LR		0.0001	0.0001	0.0001
Dropout		0.1	0.1	0.1
Gradient Norm Clipping		1.0	1.0	1.0
DVAE	Vocabulary Size	4096	4096	4096
	Temp. Cooldown	1.0 to 0.1	1.0 to 0.1	1.0 to 0.1
	Temp. Cooldown Steps	30000	30000	30000
	LR	0.0003	0.0003	0.0003
Image Size		128	128	128
Image Tokens		1024	1024	1024
Transformer Decoder	Layers	8	8	8
	Heads	4	4	4
	Hidden Dim.	192	192	192
Slot Attention	Slots	11	24	6 & 11
	Iterations	3	3	3
	Slot Dim.	192	192	192

STEGO To evaluate STEGO on the COCO-Stuff 171 dataset (Sec. 4.3), we modified the official STEGO implementation[7] to use batch k-means clustering with 172 clusters. Other parameters were not changed from the STEGO configuration on the COCO-Stuff 27 dataset as the training data is the same for those two datasets.

For the experiment evaluating STEGO on the MOVi datasets (Sec. B.7), we run a DINO ViT-B with patch size 8, using 22 and 27 clusters, and train for 20 000 steps. We manually tested different

[6] https://github.com/singhgautam/slate
[7] https://github.com/mhamilton723/STEGO

hyperparameters and settled on $\lambda_{\text{self}} = 0.6$, $\lambda_{\text{knn}} = 0.3$, $\lambda_{\text{rand}} = 0.6$, $b_{\text{self}} = 0.26$, $b_{\text{knn}} = 0.21$, $b_{\text{rand}} = 0.4$ for MOVi-C, and $\lambda_{\text{self}} = 0.6$, $\lambda_{\text{knn}} = 0.3$, $\lambda_{\text{rand}} = 0.6$, $b_{\text{self}} = 0.22$, $b_{\text{knn}} = 0.17$, $b_{\text{rand}} = 0.4$ for MOVi-E. We refer to Hamilton et al. (2022) for the meaning of those parameters.

C.3 Unsupervised Semantic Segmentation

For unsupervised semantic segmentation, we need to assign a discrete label to each discovered mask. As common in the literature (Gansbeke et al., 2021; Melas-Kyriazi et al., 2022), we run k-means clustering on features associated with each mask for this purpose. In particular, for each slot generated by the model, we compute a feature vector by taking a weighted average between the slot's mask and features from ViT block 9. For DINO pre-training, we experimented with different features and found the outputs of ViT block 9 to yield the best results. We also use this block with the other pre-training methods (MoCo-v3, MSN, MAE); however, we note that for those methods this block might not be optimal. Each vector is then L2-normalized, and the set of vectors from all images is clustered by running k-means clustering, and each slot is associated with a cluster. Finally, the clusters are assigned to ground truth classes by maximizing the total IoU between clusters and classes using the Hungarian method (Kuhn, 1955). For k-means clustering, to avoid sensitivity to initialization, we run 20 repetitions and use the run with the lowest sum of squared distances to the cluster centers. As we noticed there is still some variance between different evaluation runs, we run the inference and clustering procedure three times and use the average as the final value for the training run.

Results for unsupervised semantic segmentation are dependent on the number of clusters used. Prior methods (Hamilton et al., 2022; Gansbeke et al., 2022) typically use the number of classes labeled in the datasets as the number of clusters. For unsupervised object segmentation, this presents an issue for our method, as it captures entities in both foreground and background, and thus clustering using the number of (foreground) object classes leads to over-merging of classes. To ensure a fairer comparison, we instead estimated the number foreground plus background classes occurring on the datasets and used those as the number of clusters. In particular, for PASCAL VOC 2012, we use 105 clusters (the number of classes occurring on at least 0.5% of images, using the PASCAL-Context (Mottaghi et al., 2014) labeling and class statistics), and for COCO, we use 172 clusters (the number of classes labeled on the COCO-Stuff dataset (Caesar et al., 2018)). As this results in more clusters than ground truth classes, Hungarian matching leaves some clusters unassigned; we treat all unassigned clusters as being assigned to the "background" class. For scene decomposition, we simply use 27 clusters for COCO-Stuff 27 and 172 clusters for COCO-Stuff 172.

D Dataset and Evaluation Settings

MOVi As the MOVi datasets are video datasets, we convert them into an image dataset by sampling 9 random frames per clip. This yields 87 732 images for training and 2 250 images for evaluation. Note that we use the validation split provided by the MOVi dataset for evaluation, and not the test split containing out-of-distribution objects and backgrounds. This is in line with prior work (Kipf et al., 2022; Elsayed et al., 2022).

For the "semantic" MOVi-C variants used in Sec. B.6, we generated 250 videos using the Kubric simulator (Greff et al., 2022), yielding 2 250 images. For the "semantic objects" variant, after sampling the first object in each video, only objects of the same category were allowed to be sampled. The categories are: "Action Figures", "Bag", "Board Games", "Bottles and Cans and Cups", "Camera", "Car Seat", "Consumer Goods", "Hat", "Headphones", "Keyboard", "Legos", "Media Cases", "Mouse", "Shoe", "Stuffed Toys", "Toys". Objects that do not have one of those categories assigned were not considered. For the "duplicate objects" variant, the exact same object was resampled multiple times. Otherwise, the same settings as for MOVi-C were used.

PASCAL VOC 2012 We train on the trainaug variant with 10 582 images. For object discovery, we evaluate on the validation set of trainaug, with 1 449 images. For unsupervised object segmentation, we use the 20 object classes plus one background class. For object localization, we evaluate on the trainval split, with 11 540 images, as standard for this task (Simeoni et al., 2021; Melas-Kyriazi et al., 2022).

COCO We train on the COCO 2017 dataset with 118 287 images, and evaluate on the validation set with 5 000 images. For unsupervised object segmentation, we use the 80 "things" classes plus one

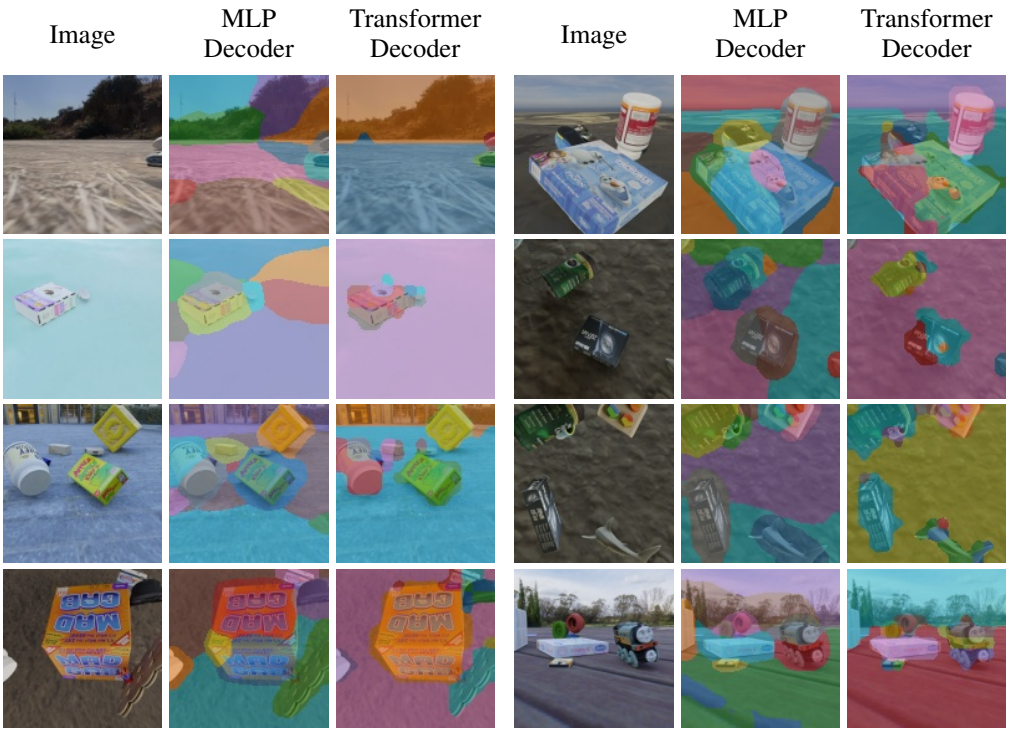

Figure 14: Masks on MOVi-C produced by our method, using 11 slots and a ViT-B/16 encoder. We show predictions from the MLP and the Transformer decoder.

background class. For unsupervised scene decomposition, we use the COCO-Stuff 27 labeling (Ji et al., 2019; Cho et al., 2021), resulting from merging the 80 "things" classes into 12, and the 91 "things" classes into 15 super-classes, and the COCO-Stuff 171 labeling, using all "things" and "stuff" classes. For object localization, we evaluate on COCO-20k, with 19 817 images. COCO-20k is a subset of COCO 2014 training and validation images, filtered to remove images with only crowd instances and removing crowd annotations.

Evaluation Settings For scene decomposition on COCO-Stuff 27 and COCO-Stuff 171, we take a center crop of the image after resizing the minor axis to 320 pixels and evaluate the masks at 320×320 resolution to be consistent with prior work (Ji et al., 2019; Hamilton et al., 2022). For object discovery on PASCAL VOC 2012 and COCO, we use the same settings. For object discovery on MOVi, we use the full image. Again to be consistent with prior work, for object localization on PASCAL VOC 2012 and COCO-20k as well as object segmentation on PASCAL VOC 2012 and COCO, we evaluate the *full* image after resizing to 224×224, ignoring aspect ratio. Our model proved to be robust to the aspect ratio distortion to some degree; however, we believe that results can be further improved, e.g. by taking the distortion into account during training, or by using multi-crop evaluation. Soft probability masks are converted to hard masks using the $\arg\max$.

E Additional Examples

We include additional mask predictions of our model for MOVi-C (Fig. 14), for MOVi-E with 24 (Fig. 16) and 11 slots (Fig. 17), for PASCAL VOC 2012 (Fig. 18) and COCO (Fig. 19, Fig. 20, Fig. 21, Fig. 22). We also show results for SLATE on the images of the main text in Fig. 15.

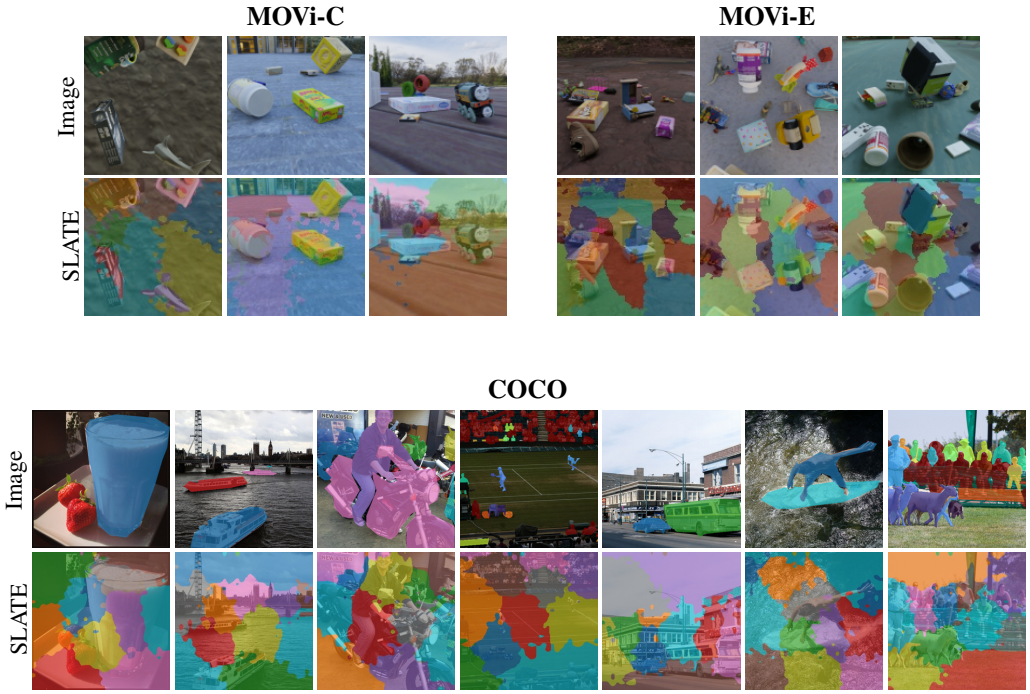

Figure 15: Examples for SLATE on MOVi-C (11 slots), MOVi-E (24 slots), and COCO (11 slots) corresponding to the images in Fig. 2 and Fig. 3.

Figure 16: Masks on MOVi-E using 24 slots. Our method uses a ViT-B/8 encoder and the MLP decoder.

Figure 17: Masks on MOVi-E using 11 instead of 24 slots (quantitative results in Table 10). Our method uses a ViT-B/8 encoder and the MLP decoder.

Figure 18: Masks on PASCAL VOC 2012 produced by our method, using 6 slots and a ViT-B/16 encoder. We show predictions from the MLP and the Transformer decoder.

Figure 19: Masks on COCO 2017 produced by our method, using 7 slots and a ViT-B/16 encoder. We show predictions from the MLP and the Transformer decoder.

Figure 20: Masks on COCO 2017 produced by our method, using different encoders and pre-training schemes on COCO 2017 (cf. Table 6 and Table 8). All models use 7 slots and the Transformer decoder.

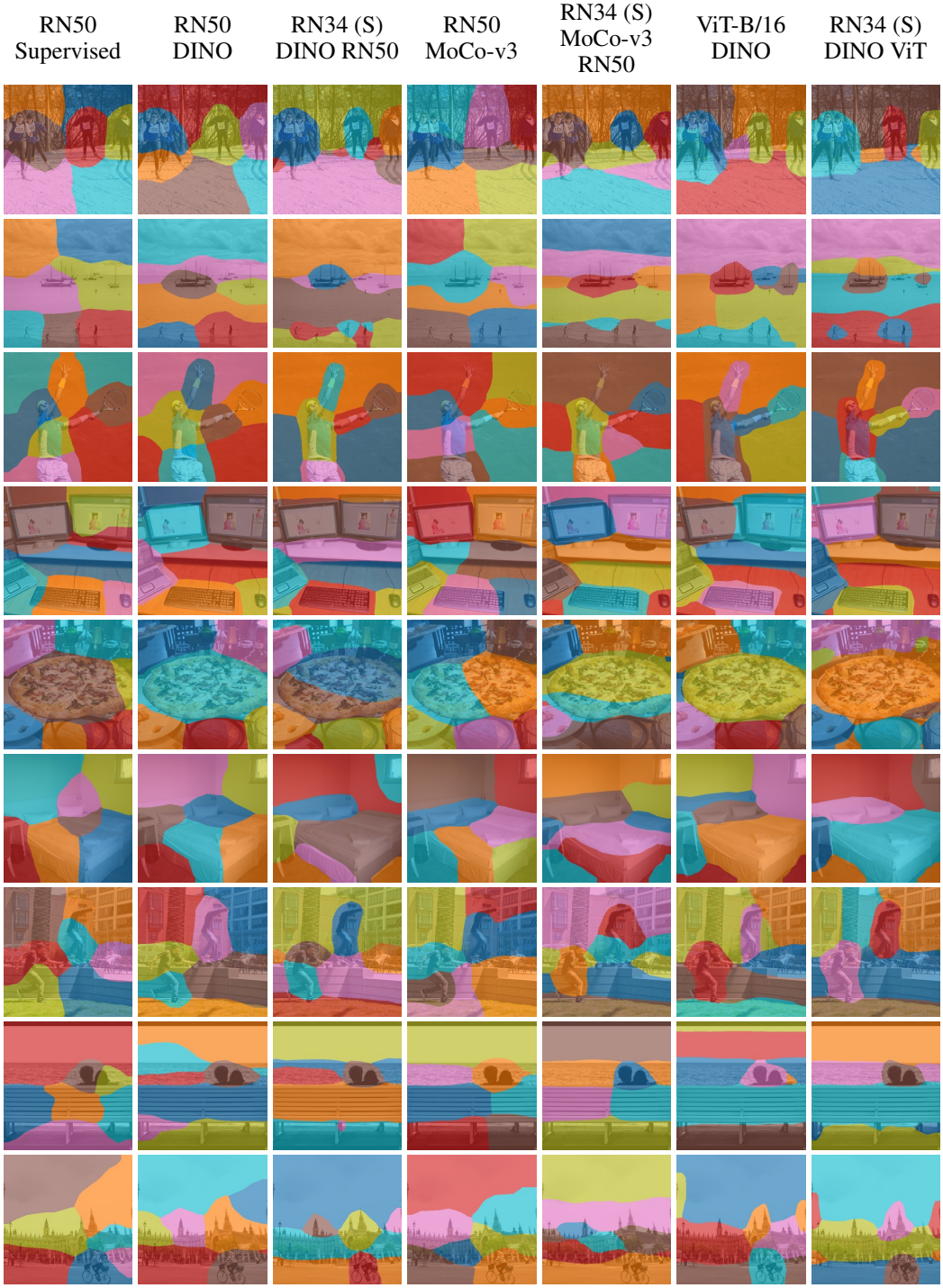

Figure 21: Masks on COCO 2017 produced by our method, using different encoders and pre-training schemes on COCO 2017. ResNet34s (RN34) were trained from scratch ("S"), by reconstruction of ResNet50 (RN50) or ViT-B/16 features (cf. Table 12). All models use 7 slots and the MLP decoder.

Figure 22: Masks on COCO 2017 produced by our method, using different encoders and pre-training schemes on COCO 2017. ResNet34s (RN34) were trained from scratch ("S"), by reconstruction of ViT-B/16 features (cf. Table 12). All models use 7 slots and the MLP decoder.