Bootstrapping Top-down Information for Self-modulating Slot Attention

Dongwon Kim[1] **Seoyeon Kim**[1] **Suha Kwak**[1,2]
Dept. of CSE, POSTECH[1] Graduate School of AI, POSTECH[2]
{kdwon, syeonkim07, suha.kwak}@postech.ac.kr

Abstract

Object-centric learning (OCL) aims to learn representations of individual objects within visual scenes without manual supervision, facilitating efficient and effective visual reasoning. Traditional OCL methods primarily employ bottom-up approaches that aggregate homogeneous visual features to represent objects. However, in complex visual environments, these methods often fall short due to the heterogeneous nature of visual features within an object. To address this, we propose a novel OCL framework incorporating a *top-down pathway*. This pathway first bootstraps the semantics of individual objects and then modulates the model to prioritize features relevant to these semantics. By dynamically modulating the model based on its own output, our *top-down pathway* enhances the representational quality of objects. Our framework achieves state-of-the-art performance across multiple synthetic and real-world object-discovery benchmarks.

1 Introduction

Object-centric learning (OCL) is the task of learning representations of individual objects from visual scenes without manual labels. The task draws inspiration from the human perception which naturally decomposes a scene into individual entities for comprehending and interacting with the real world visual environment. Object-centric representations provides improved generalization and robustness [9], and have been proven to be useful for diverse downstream tasks such as visual reasoning [44], simulation [45], and multi-modal learning [24]. In this context, OCL which learns such representations without labeled data has gained increasing attention.

A successful line of OCL builds upon slot attention [30]. This method decomposes an image into a set of representations, called slots, that iteratively compete with each other to aggregate image features. Reconstructing the original image from the slots, they are encouraged to capture entities constituting the scene. This simple yet effective method has been further advanced by novel encoder or decoder architectures [33, 19, 46, 42], optimization technique [18, 6], and new query initialization strategies [18, 24].

It is worth noting that all these methods are fundamentally considered bottom-up models, as they rely on aggregating visual features without incorporating high-level semantic information from the beginning. This bottom-up approach assumes that visual features within an object are homogeneous and can be clustered in the feature space, which only holds for simplistic objects that can be identified using low-level cues such as color [20]. In complex real-world scenarios where visual entities of the same semantics exhibit diverse appearances, this homogeneity often breaks down, leading to suboptimal object representations [22, 47]. Thus, we take an approach different from the previous line of research: *introducing top-down information into slot attention*, such as object categories and semantic attributes.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3296

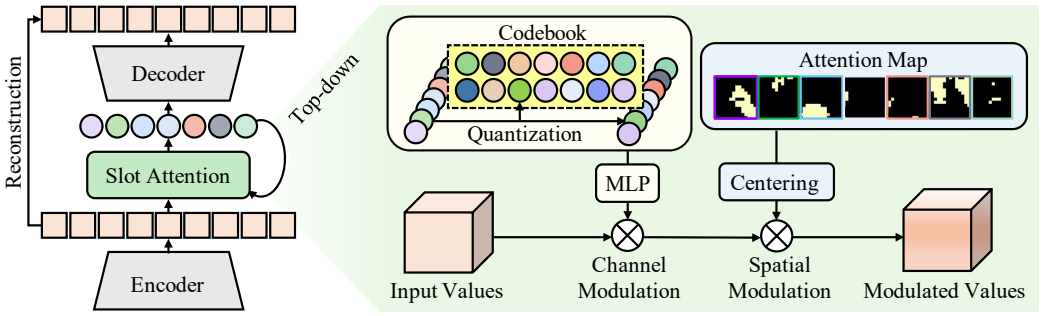

Figure 1: The overall pipeline of our framework. A *top-down pathway* is introduced into slot attention to utilize top-down information. The pathway consists of two parts: bootstrapping top-down knowledge and exploiting them. Firstly, semantic information is bootstrapped from slot attention outputs by mapping slots to discrete codes from a learned codebook through vector quantization. Secondly, slot attention is modulated using these codes and its attention maps, transforming it into a self-modulating module. Inner activations are modulated across channels with codes and across space with centered attention maps. Slot attention is then repeated with these modulated activations, yielding more representative slots.

Incorporating top-down information enables slot attention to specialize in discerning objects within specific semantic categories. For instance, identifying vehicles in a complex urban environment can be challenging due to the diverse and cluttered nature of the scene. Top-down information can guide the model to prioritize vehicle-specific features, such as wheels and windows. This inhibits the contributions of irrelevant features when computing slots, and enhances the aggregation of visual features of individual vehicles into slots. Nevertheless, devising such a top-down approach is not straightforward since OCL assumes an unsupervised setting without any labeled data, making it hard to identify and exploit the high-level semantics typically obtained from annotated datasets.

We propose a novel framework that incorporates a *top-down pathway* into slot attention to provide and exploit top-down semantic information; Fig. 1 illustrates of our framework. The pathway consists of two parts: bootstrapping semantics and exploiting them for better representations. Firstly, top-down semantic information is bootstrapped from the output of slot attention itself, by mapping continuous slots to discrete codes selected from a finite learned codebook. Such an approach allows the codebook to learn prevalent semantics in the dataset, with each code representing a specific semantic concept. Thus, semantic information can be bootstrapped without any object-level annotations and used to provide top-down semantic information. Secondly, slot attention is modulated using bootstrapped top-down cues obtained from the first phase, which we call self-modulation. In this phase, the *top-down pathway* dynamically guides the slot attention by re-scaling its inner activations based on the top-down information. This self-modulation process enables the model to focus on feature sub-spaces where object homogeneity is more consistent, thereby improving its performance in diverse and realistic settings.

Our contributions are threefold:

- We introduce a method to bootstrap top-down semantic information from the output of slot attention, without requiring any object-level annotations. This allows the extraction of high-level semantic cues from an unsupervised learning process.
- We propose a self-modulation scheme that dynamically guides the slot attention's inner activations to enhance object representation, successfully incorporating the top-down semantic cues extracted.
- By integrating the proposed *top-down pathway* into slot attention, we demonstrate that the performance of object discovery is largely improved on various OCL benchmarks.

2 Related Work

Object-centric learning OCL aims to learn representations of individual objects within an image. The 'object-centric' dimension is orthogonal to the conventional representation learning which learns representations independent of the composition of the image. The structured nature of

object-centric representations offers improved generalization [9], making it valuable for various applications, including visual reasoning [44], dynamics simulation [45], and multi-modal learning [23, 24]. A foundational method in this field is slot attention [30], which introduced a simple yet effective framework that employs a competitive attention mechanism between slots. Following slot attention, many recent works have proposed improvements by introducing novel encoder or decoder formulations [33, 19, 46, 35], optimization techniques [18, 6], additional slot refinement modules [37, 25, 3], and expansions to video modality [27, 11, 36]. These methods are primarily bottom-up models, while our approach proposes to bootstrap and incorporate top-down information.

Incorporating top-down information The human visual system perceives scenes by leveraging both top-down and bottom-up visual information [4, 8]. Top-down information represents task-driven contextual cues, such as high-level semantics and prior knowledge about the scene. In contrast, bottom-up information is derived directly from the sensory input. Inspired by this dual-processing mechanism of the human visual system, several studies [34, 1, 48] have attempted to model this approach within deep learning, achieving significant improvements across various tasks. Our work follows in a similar direction, specifically focusing on introducing top-down information into the representative OCL method, slot attention. By incorporating top-down semantic and spatial information, we aim to enhance the performance of slot attention in diverse visual environments, addressing the limitations of previous bottom-up methods

Discrete representation learning Discrete representations within neural networks are considered effective for modeling discrete modalities [50, 38] and tackling generation tasks [13, 28]. Particularly, the pioneering work, VQ-VAE [38], introduced a method for learning discrete latent representations through vector quantization. This model uses a discrete codebook, where the encoder maps input data to discrete codes using nearest-neighbor lookup and the decoder reconstructs the input from the codes. Another notable approach is the Gumbel-softmax trick [17, 32], which provides a differentiable approximation of sampling from a categorical distribution. Recent advancements have aimed to incorporate a more sophisticated formulation of codebooks [49] or improve codebook utilization [16] to better handle discrete representations. Recent work by Wallingford et al. [40] is related to our research in terms of using vector quantization for segmentation task. However, our method differs in that the quantized codes are used to modulate bottom-up slot attention, while in Wallingford et al. [40], the codes are used solely for segmentation labeling.

3 Method

We propose an OCL framework that incorporates top-down semantic information, such as object categories and semantic attributes, into slot attention through a *top-down pathway*. Fig. 1 illustrates the overall pipeline of our framework. Firstly, slot attention is applied to visual features extracted from an image encoder to output slots (Sec. 3.1). Then, a *top-down pathway* leverages the slots to identify semantics in the input image and modulate slot attention. The pathway consists of two parts: bootstrapping top-down semantic information from a learned codebook and attention maps (Sec. 3.2) and modulating the inner activations of slot attention with this semantic information (Sec. 3.3). During the self-modulation stage, slot attention is repeated with the modulated activations, resulting in more representative slots.

3.1 Slot Attention

Slot attention is a recurrent bottom-up module that aggregates image features into slots through an iterative attention process, where each of the resulting slots represent an entity in the input image. Within our framework, these slots are used to bootstrap top-down information in the later stage (Sec. 3.2).

The module takes in the initial slots, $\boldsymbol{S}^0 \in \mathbb{R}^{K \times D}$, and visual features extracted from an image encoder, $\mathbf{x} \in \mathbb{R}^{N \times D_{\text{feat}}}$. The initial slots are obtained by sampling K vectors from a learnable Gaussian distribution using a reparameterization trick. The slots $\boldsymbol{S} = [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_K] \in \mathbb{R}^{K \times D}$, where each represents an individual object in the image, are computed by iteratively updating the initial slots T times as

$$\boldsymbol{S} := \boldsymbol{S}^T, \text{where } \boldsymbol{S}^{t+1} = \texttt{slot\_attn}\big(\mathbf{x}, \boldsymbol{S}^t\big). \tag{1}$$

In each iteration, the slots attend to the visual features, refining their representations through a series of attention-based updates. Let $q(\cdot)$, $k(\cdot)$, and $v(\cdot)$ represent linear projections from dimension d to

 https://doi.org/10.52202/079017-3296

d_h. Then, the attention map $\boldsymbol{A} \in \mathbb{R}^{K \times N}$ is computed as

$$\boldsymbol{A}_{i,j} = \frac{e^{P_{i,j}}}{\sum_{i=1}^{K} e^{P_{i,j}}}, \text{ where } P = \frac{q\big(\boldsymbol{S}^{t-1}\big)k\big(\mathbf{x}\big)^\top}{\sqrt{d_h}}. \tag{2}$$

Unlike the original attention introduced in transformer [39] which normalizes across the keys, the attention in slot attention is normalized across the slots. Such a distinct normalization scheme makes the slots compete with each other to aggregate the visual features, encouraging each slot to represent a distinct object in the scene. Then, the computed attention map $\boldsymbol{A}$ is normalized across the rows into $\tilde{\boldsymbol{A}}$ and used to update the slots as

$$\boldsymbol{S}^t = f_{\texttt{update}}(\boldsymbol{U}, \boldsymbol{S}^{t-1}), \text{ where } \boldsymbol{U} = \tilde{\boldsymbol{A}} v\big(\mathbf{x}\big), \tag{3}$$

such that $\boldsymbol{U}$ is the weighted sum of the visual features $v(\mathbf{x})$. The function $f_{\texttt{update}}$ processes $\boldsymbol{U}$ with consecutive GRU [7] and Multi-Layer Perceptron (MLP) and residually sums it to the previous slots, $\boldsymbol{S}^{t-1}$, to produce the updated slots, $\boldsymbol{S}^t$. For further details, refer to Locatello et al. [30].

3.2 Bootstrapping Top-down Information

Our idea to bootstrap top-down information without annotations is based on our observation that the slots $\boldsymbol{S}$, which are outputs of the bottom-up attention module, contain rough semantic information about objects. We leverage this coarse information to bootstrap both the semantic and spatial top-down information about the objects coarsely represented by the slots. Top-down semantic information pertains to the specific semantic categories or attributes of the objects (*what*), while top-down spatial information indicates the locations or regions within the image where these objects are located (*where*). Incorporating such knowledge can guide slot attention to focus on the features most relevant to the objects expected to appear, enabling it to accurately capture objects that are obscured or have high intra-object variance, such as people with different hairstyles or clothing.

Firstly, we extract the "*what*" information from the slots $\boldsymbol{S}$ using Vector Quantization (VQ), which maps each slot to one of the semantic concepts learned throughout training. Specifically, each slot $\boldsymbol{S}_k$ is mapped to the nearest code in a finite codebook, $\mathbb{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_E] \in \mathbb{R}^{E \times D}$ with size E. The mapped code $\boldsymbol{c}_k^* \in \mathbb{R}^D$ is considered a top-down semantic cue for the slot $\boldsymbol{s}_k$. Formally, this quantization process can be written as

$$\boldsymbol{c}_k^* = \underset{\boldsymbol{c} \in \mathbb{C}}{\arg\min} \|\boldsymbol{s}_k - \boldsymbol{c}\|_2^2. \tag{4}$$

Since the $\arg\min$ operation is non-differentiable, we use the straight-through estimator [2] for backpropagation. During training, the codebook learns to store distinct semantic patterns recurring within the dataset by quantizing continuous slot embeddings into a limited number of discrete embeddings. Thereby, each code can act as automatically discovered top-down semantic information.

Secondly, we obtain the "*where*" information from the attention maps of the last layer of slot attention. For each slot $\boldsymbol{s}_k$, the k-th row vector of the attention map $\boldsymbol{A}$, denoted as $\boldsymbol{a}_k \in \mathbb{R}^N$, is used to aggregate visual features and update $\boldsymbol{s}_k$. This attention map provides useful spatial prior information about where each extracted top-down semantic information is located in the image.

3.3 Self-modulating Slot Attention

In the original slot attention (Sec. 3.1), the slot updates are driven purely by visual features extracted from the input without incorporating higher-level semantic information that can provide additional context. To address this limitation, we introduce self-modulating slot attention, which modulates the computation of the slot updates based on the top-down information obtained in the bootstrapping stage (Sec. 3.2). This bootstrapped top-down information is used to dynamically amplify or inhibit specific channel dimensions or regions of the value-projected visual features, while keeping the model parameters unchanged. Formally, self-modulating slot attention can be represented by conditioning slot attention with the vector quantized slots $\boldsymbol{c}_k^*$ and their corresponding slot-wise attention map $\boldsymbol{a}_k$:

$$\hat{\boldsymbol{S}} := \hat{\boldsymbol{S}}^T, \text{where } \hat{\boldsymbol{S}}^{t+1} = \texttt{slot\_attn}\big(\mathbf{x}, \hat{\boldsymbol{S}}^t; [\boldsymbol{c}_i^*]_{i=1}^K, [\boldsymbol{a}_i]_{i=1}^K\big), \tag{5}$$

Algorithm 1 Self-modulating Slot Attention. The module inputs visual features extracted from an encoder, initial slots sampled from a learned Gaussian distribution, and K vector quantized slots and their corresponding K slot-wise attention maps output from the original slot attention. The number of iterations, T, is set to three.

Input: Visual features $\mathbf{x}$, initial slots $\hat{\boldsymbol{S}}^0$, vector quantized slots $[\boldsymbol{c}_i^*]_{i=1}^K$, and slot-wise attention maps $[\boldsymbol{a}_i]_{i=1}^K$.
Output: Slots after T-iteration of self-modulating slot attention $\hat{\boldsymbol{S}}^T$.

for $k = 1$ to K in parallel **do**
  $\boldsymbol{m}_k^c = \texttt{MLP}(\boldsymbol{c}_k^*)$ // Compute channel-wise modulation vector
  $\boldsymbol{m}_k^s = 1 + (\boldsymbol{a}_k - \overline{\boldsymbol{a}_k})$ // Compute spatial-wise modulation vector
  $\boldsymbol{M}_k = \boldsymbol{m}_k^s \otimes \boldsymbol{m}_k^c$ // Compute modulation map
for $t = 1$ to T **do**
  $\tilde{\boldsymbol{A}} = \texttt{softmax}\big(q\big(\hat{\boldsymbol{S}}^{t-1}\big)k\big(\mathbf{x}\big)^\top / \sqrt{D_h}\big)$
  $\hat{\boldsymbol{S}}^t = f_{\texttt{update}}([\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K], \hat{\boldsymbol{S}}^{t-1})$, where $\boldsymbol{u}_k = \tilde{\boldsymbol{A}}_k(\boldsymbol{M}_k \odot v\big(\mathbf{x}\big))$ // Modulated slot update
return $\hat{\boldsymbol{S}}^T$

where $\hat{\boldsymbol{S}}$ represents the slots from the self-modulating slot attention, different from the slots of the original slot attention denoted by $\boldsymbol{S}$. Note that the original slot attention (Sec. 3.1) and self-modulating slot attention share parameter weights and initial slots, such that $\boldsymbol{S}^0 = \hat{\boldsymbol{S}}^0$.

Specifically, we modulate slot attention with a modulation map $\boldsymbol{M}_k \in \mathbb{R}^{N \times D}$ computed from $\boldsymbol{c}_k^*$ and $\boldsymbol{a}_k$. Each element of the modulation map represents the relevance score between the corresponding visual feature element and the top-down information of the expected object. This modulation map can be used to guide the update of each slot $\hat{\boldsymbol{S}}_k$ (Eq. 3), by prioritizing specific value elements with the high relevance scores. In the self-modulating slot attention, computation of the slot update $\boldsymbol{U}$ is replaced with:

$$\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K] \in \mathbb{R}^{K \times D}, \;\; \boldsymbol{u}_k = \tilde{A}_k(\boldsymbol{M}_k \odot v\big(\mathbf{x}\big)), \tag{6}$$

where $\odot$ represents Hadamard product. Such re-scaling of the value features with the modulation map ensures that the specific channel dimension or regions contribute more to the update of each slot, based on the semantics or locations encoded in bootstrapped top-down information.

The modulation map $\boldsymbol{M}_k$ is computed by taking the outer product between channel-wise and spatial-wise modulation vectors, which are predicted using $\boldsymbol{c}_k^*$ and $\boldsymbol{a}_k$, respectively:

$$\boldsymbol{M}_k = \boldsymbol{m}_k^s \otimes \boldsymbol{m}_k^c \in \mathbb{R}^{N \times D}. \tag{7}$$

For predicting channel-wise modulation vector $\boldsymbol{m}_k^c$, quantized slot $\boldsymbol{c}_k^*$ is used, which tells us "*what*" the object appearing in the image is. The channel-wise scaling is designed to enforce the model to focus on certain feature subspaces closely correlated to the semantic concept identified. Specifically, channel-wise modulation vector $\boldsymbol{m}_k^c$ can be obtained by feeding quantized slot $\boldsymbol{c}_k^*$ to the MLP, which is represented as:

$$\boldsymbol{m}_k^c = \texttt{MLP}(\boldsymbol{c}_k^*) \in \mathbb{R}^D. \tag{8}$$

The spatial-wise modulation vector $\boldsymbol{m}_k^s$ can be obtained by further processing the attention map of each slot, which contains the top-down information on the location of the semantic concept:

$$\boldsymbol{m}_k^s = 1 + (\boldsymbol{a}_k - \overline{\boldsymbol{a}_k}) \in \mathbb{R}^N, \tag{9}$$

where $\overline{\boldsymbol{a}_k}$ is for the average of the attention score of $\boldsymbol{a}_k$. Using an attention map as is for modulation will make all values down-scaled, while some regions likely to contain the object should be highlighted for effective incorporation of the spatial top-down information. Thus, we use the attention map shifted to have a mean value of 1 for the spatial-wise modulation map.

3.4 Training

Slot attention is trained within an autoencoding framework, using a decoder that reconstructs visual features output by the image encoder [35, 33] or the original image [30] from the slots. In this paper, we choose the visual feature reconstruction as our training objective since it is known to provide more robust training signals for real-world datasets [33]. We also employ a vector quantization objective

 https://doi.org/10.52202/079017-3296

for the codebook $\mathbb{C}$ only, which thereby learns to minimize the mean-squared error between the slot and the sampled codes. The reconstruction objective $\mathcal{L}_{\text{recon}}$ and vector quantization objective $\mathcal{L}_{\text{VQ}}$ are given by

$$\mathcal{L}_{\text{recon}} = \|\texttt{Dec}(\hat{\boldsymbol{S}}; \mathbf{x}) - \mathbf{x}\|_2^2, \quad \mathcal{L}_{\text{VQ}} = \|\texttt{sg}(\boldsymbol{S}) - \boldsymbol{C}^*\|_2^2, \tag{10}$$

where $\texttt{sg}(\cdot)$ represents stop gradient operation and $\boldsymbol{C}^* = [\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, \ldots, \boldsymbol{c}_K^*] \in \mathbb{R}^{K \times D}$. For the decoder, we utilized the autoregressive slot decoder [35, 33]. The reconstruction objective ensures that the learned slot representations capture essential information about the objects in the scene, while the vector quantization objective encourages the codebook to capture recurring semantic concepts in the dataset.

4 Experiments

4.1 Experimental Settings

Datasets To verify the proposed method in diverse settings, including synthetic and authentic datasets, we considered four object-centric learning benchmarks: MOVI-C [15], MOVI-E [15], PASCAL VOC 2012 [14], and MS COCO 2017 [29]. MOVI-C and MOVI-E are synthetic datasets, adopted for validating our method in relatively simple visual environments. MOVI-C contains 87,633 images for training and 6,000 images for evaluation, while MOVI-E contains 87,741 and 6,000, respectively. To evaluate the proposed model in real-world settings, we leverage the VOC and COCO datasets. Following DINOSAUR [33], we use the `trainaug` variants, containsing 10,582 training images, for VOC dataset. For the evaluation, we use the validation split containing 1,449 images. The COCO dataset consists of 118,287 training images and 5,000 images for evaluation. While the VOC dataset includes some images with a single object, images of the COCO dataset always contain 2 or more objects, making it the most challenging. MOVI datasets are licensed under apache license 2.0 and COCO is licensed under CC-BY-4.0.

Metrics We evaluate our method with three metrics: foreground adjusted random index (FG-ARI), mean best overlap (mBO), and mean intersection over union (mIoU). The FG-ARI is the ARI metric computed for foreground regions only (objects), which measures the similarity between different clustering results. The mBO and mIoU are both IoU-based metrics, computed for all regions including the background. The mBO computes the average IoU between ground truth and prediction pairs, obtained by assigning each prediction to the ground truth mask with the largest overlap. The mIoU is computed as the average IoU between ground truth and prediction pairs obtained from Hungarian matching. For COCO and VOC, mBO^i and mBO^c indicate the mBO metric computed using semantic segmentation and instance segmentation ground truth. We use the instance segmentation ground truth for other metrics. Following previous work [33], the internal attention maps of the autoregressive decoder are used as the mask prediction results of the slots.

Implementation details To assess the effectiveness of the proposed top-down pathway, our model is implemented based on DINOSAUR [33], a representative slot-based OCL method. For the encoder and decoder, we use a DINO [5] pretrained ViT-B/16 [10] and an autoregressive transformer decoder [35, 33], respectively. The model is trained using an Adam optimizer [26] with an initial learning rate of 0.0004, while the encoder parameters are not trained. The number of slots K is set to 11, 24, 7, and 6 for MOVI-C, MOVI-E, COCO, and VOC, respectively. The codebook size E is set to 128 for synthetic datasets (MOVI-C and MOVI-E) and 512 for authentic datasets (COCO and VOC). The model is trained for 250K iterations on VOC and for 500K iterations on the others. For the ablation study and analysis, models are trained for 200K iterations on COCO, as this was enough to reveal overall trends given limited computational resources. Full training of the model takes 26 hours using a single NVIDIA RTX 3090 GPU.

Codebook size E selection The performance of the proposed *top-down pathway* depends on codebook size E (Sec. 4.4), necessitating a principled selection method. We determine E automatically by monitoring the perplexity of code usage distribution during training, requiring only the training set without validation data. Perplexity—the exponent of entropy—indicates how uniformly the codes are being used. While perplexity typically increases with codebook size, it plateaus when E exceeds the number of distinct semantic patterns in the data, as some codes become unused [16, 49, 28]. To find the optimal size, we start with $E = 64$ and double it until the perplexity plateaus after 250K iterations. For example, on COCO, the perplexity when the codebook size is 256, 512, and 1024 are

Table 1: Comparison with DINOSAUR [33] on synthetic datasets: MOVI-C [15] and MOVI-E [15]. We include both the reported and reproduced performance of DINOSAUR for fair comparison.

Method	MOVI-C			MOVI-E		
	FG-ARI	mBO^i	mIoU	FG-ARI	mBO^i	mIoU
DINOSAUR [33]	55.7	42.4	-	-	-	-
DINOSAUR reprod	54.7±4.1	41.9±1.8	41.0±2.1	53.8±2.1	34.5±1.7	33.6±1.9
Ours	**58.9**±5.1	**46.8**±2.4	**45.9**±2.5	**59.7**±3.1	**39.3**±1.8	**38.3**±1.9

Table 2: Comparison with DINOSAUR [33] on real-world datasets: COCO [29] and VOC [14]. We include both the reported and reproduced performance of DINOSAUR for fair comparison.

Method	COCO				VOC			
	FG-ARI	mBO^i	mBO^c	mIoU	FG-ARI	mBO^i	mBO^c	mIoU
DINOSAUR [33]	34.1±1.0	31.6±0.7	39.7±0.9	-	24.8±2.2	44.0±1.9	51.2±1.9	-
DINOSAUR reprod [33]	34.1±0.9	31.4±0.5	39.5±0.1	29.4±0.6	**27.0**±3.2	41.2±3.4	48.2±3.8	39.0±3.6
Ours	**37.4**±0.0	**33.0**±0.3	**40.3**±0.2	**31.2**±0.3	26.7±4.7	**43.9**±2.6	**51.0**±2.5	**42.0**±2.8

176.9, 253.9, and 242.8, respectively, where 512 was chosen as the final size. This procedure enables efficient hyperparameter selection using only training data, eliminating the need for validation set tuning. Following this approach, we set $E = 128$ for synthetic datasets (MOVI-C and MOVI-E) and $E = 512$ for real-world datasets (COCO and VOC).

4.2 Quantitative Analysis

DINOSAUR [33] is the first successful OCL method that scales slot attention to real-world datasets by introducing the use of a self-supervised image encoder [5], an autoregressive decoder, and a feature reconstruction objective. Notably, DINOSAUR uses the vanilla slot attention mechanism without modifications, making it a perfect baseline for validating the effectiveness of our proposed *top-down pathway*. Thus, we adopt DINOSAUR as a baseline and compare its performance with and without our proposed method. For a fair comparison, we report both the reported and reproduced performance of DINOSAUR.

Tab. 1 demonstrates that incorporating the proposed *top-down pathway* into DINOSAUR largely improves performance in every metric. Specifically, our method improves FG-ARI by 5.9 on MOVI-E, which is the most challenging synthetic dataset. In Tab. 2, performances on authentic datasets, COCO and VOC, are reported. Our method largely surpasses the reproduced baseline on most metrics. The only metric for which our method does not show improvement is the FG-ARI on VOC. We hypothesize that this is because VOC images frequently contain single objects only, and FG-ARI is computed solely with foreground pixels so that the performance is less affected by the top-down information.

Tab. 3 presents a comparison between our method and recent state-of-the-art methods. It is notable that our proposed method achieves competitive performance even to recent methods using advanced diffusion-based decoders [19, 46]. Moreover, our approach focuses on incorporating top-down information into slot attention, which is orthogonal to the line of work advancing decoders to provide better training signals to slot attention.

4.3 Qualitative Results

Codebook visualization To validate whether the proposed codebook learns meaningful semantic concepts, we present visualizations of the codebook in Fig. 2. The index of the code and the mask prediction obtained from the slots modulated by the code are presented together, revealing the semantic entity each code represents. Visualization demonstrates that the codebook successfully discovers and stores distinct semantic concepts without using any annotations. Moreover, the codes are mapped to objects with various appearances and layouts, which demonstrates that the codes learn high-level semantic information and not low-level structural or positional information.

Prediction visualization In Fig. 3, we visualize the original image, mask predictions, and slot attention maps $\boldsymbol{A}$ before and after self-modulation. Results demonstrate that the modulation dynamically

https://doi.org/10.52202/079017-3296

Table 3: Comparison with state of the arts [33] on COCO [29], VOC [14], MOVI-C [15], and MOVI-E [15]. Performances of Slot Attention on MOVI-C and -E are reproduced by [33] and that of SLATE by [19]. Those on COCO and VOC are from [46].

Method	COCO		VOC		MOVI-C		MOVI-E	
	FG-ARI	mBO^i	FG-ARI	mBO^i	FG-ARI	mBO^i	FG-ARI	mBO^i
Slot Attention [30]	21.4	17.2	12.3	24.6	43.8±0.3	26.2±1.0	45.0±1.7	24.0±1.2
SLATE [35]	32.5	29.1	15.6	35.9	49.54±1.4	39.4±0.8	46.06±3.3	30.2±1.7
DINOSAUR [33]	34.1±1.0	31.6±0.7	24.8±2.2	44.0±1.9	55.7	42.4	-	-
Rotating Features [31]	-	-	-	40.7±0.1	-	-	-	-
SlotDiffusion [46]	37.2	31.0	17.8	**50.4**	-	-	**60.0**	30.2
LSD [19]	35.0	30.4	-	-	52.0	45.6	52.2	39.0
Ours	**37.4**±0.0	**33.3**±0.3	**26.7**±4.7	43.9±2.6	**58.9**±5.1	**46.8**±2.4	59.7±3.1	**39.3**±1.8

Figure 2: Visualization of the codebook $\mathbb{C}$ on COCO [29]. The results show that the codebook learns to capture recurring semantic concepts in the dataset, such as 'pizza' (code 124), 'sign' (code 496), 'clock' (code 235), 'zebra' (code 207), 'motorcycle' (code 341), 'surfer' (code 352), 'dog' (code 359), and 'skier' (code 343).

refines the attention maps, depending on how well they have captured the scene. For example, when the attention map is well-structured but coarse, the modulation process refines the boundaries of the attention map without changing the overall layout (first row). However, if the attention maps fail to delineate objects, the modulation process recomposes the attention maps to differentiate objects (second to fourth row). By providing top-down semantic and spatial information via self-modulation, the attention maps are enhanced to capture the object within complex real-world environments.

4.4 In-depth Analysis

Effect of codebook size In our framework, vector quantization maps slots to distinct top-down semantic information stored in the codebook, as shown in Fig. 2. In Tab. 4, we report the performances across different codebook sizes, E. The results show that a codebook size too large ($E = 1024$) or small ($E \leq 256$) leads to performance degradation. When the codebook size is too small, the codes cannot sufficiently learn distinct semantic information, which degrades the quality of the bootstrapped top-down semantic information. On the other hand, a codebook size too large may cause the codes to capture irrelevant details such as appearance variance or positional information rather than meaningful semantic concepts. However, we determine the optimal codebook size automatically using the perplexity of codebook usage during training (Sec. 4.1), eliminating the need for extensive hyperparameter tuning using validation split and benchmark metrics.

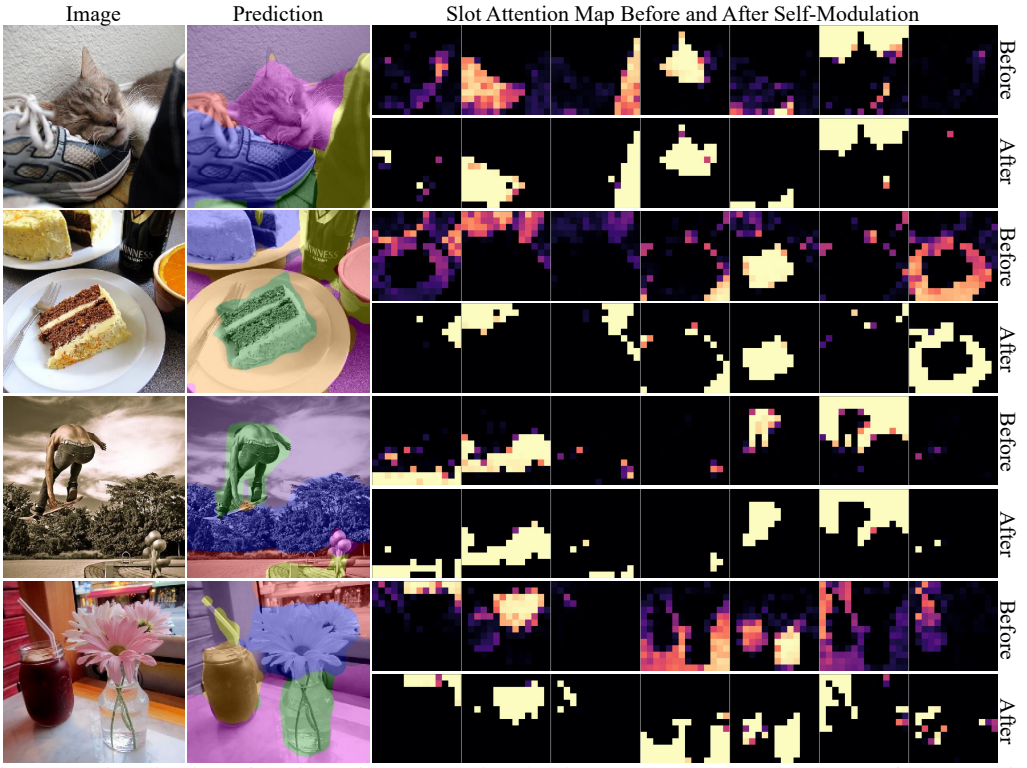

Figure 3: Visualization of the input image, predicted object mask, and attention maps of slot attention before and after self-modulation on COCO [29]. Lighter the color, higher the attention score.

Table 4: Comparison between different codebook sizes on COCO [29].

	128	256	512	1024
FG-ARI	33.1	32.6	**37.3**	31.4
mBO	30.2	30.5	**32.7**	29.8

Table 5: Comparison with DINOSAUR [33] using six iterations in slot attention on COCO [29].

	FG-ARI	mBO
DINOSAUR 6-iter	28.5	28.2
Ours	**37.3**	**32.7**

Impact of increased iterations Since our method requires repeating slot attention with self-modulation, we investigate whether our performance improvement simply comes from the increased iterations of slot attention. In Tab. 5, we compare the results of our model with the DINOSAUR baseline model using six iterations of slot attention, which is twice as many iterations as the default setting. Notably, both our model and the DINOSAUR 6-iteration model leverage the same number of iterations. The results show that the DINOSAUR model with six iterations actually performs worse compared to the default 3 iterations. This indicates that merely increasing the number of iterations does not guarantee improvement. Our method's superior performance is thus attributed to the self-modulation mechanism rather than the increased number of iterations.

Computation overhead of *top-down pathway* While our model requires one more forward pass for slot attention, the additional computation cost is negligible. In DINOSAUR, slot attention accounts for only 0.64% of the total FLOPs, compared to 71.26% for the encoder and 28.10% for the decoder. Consequently, our model requires 47.62 GFLOPs versus 47.32 GFLOPs of DINOSAUR—a mere increase below 1%. In practice, processing the entire COCO 2017 val split on a single NVIDIA RTX 3090 GPU takes 71.4 seconds for our model compared to 70.5 seconds for DINOSAUR, demonstrating minimal impact on inference time.

Codes representing broader semantics While most codes in our codebook consistently represent single object categories, we observed interesting cases where codes capture broader concepts, as shown in Fig. 4. Some codes are trained to represent supercategories - for instance, grouping different animal species (code 468) or various human parts into shared codes (code 328). This suggests the codebook can flexibly adapt to different levels of semantic abstraction when beneficial. We also discovered an edge case where certain codes (e.g., code 223) specialize in capturing top-left patches of images. This behavior appears to be influenced by the autoregressive decoding process, which

https://doi.org/10.52202/079017-3296

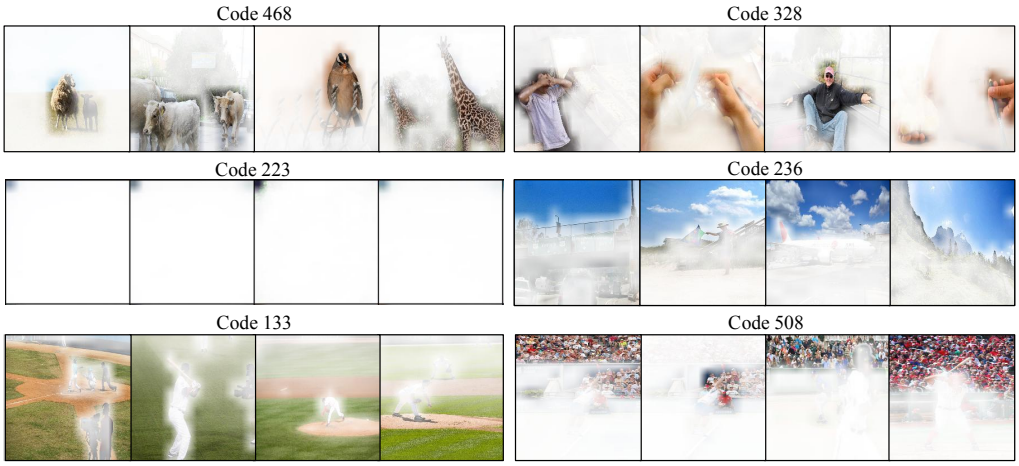

Figure 4: Visualization of the codebook $\mathbb{C}$ on COCO [29]. The results show that the codebook learns to capture broader semantics other than single object categories, such as supercategory (code 468, 328), top-left patch (code 223), and background (code 236, 133, 508).

must reconstruct the top-left patch first without surrounding context. However, these specialized positional codes are rare (1-2 out of 512 codes) and have minimal impact on overall performance. Additionally, we found that certain codes specialize in capturing background elements common in natural scenes. For example, code 236 represents sky regions, code 133 captures sports fields, and code 508 represents crowd scenes.

Ablation study In Tab. 6, we present an ablation study of our method on COCO for channel-wise modulation, vector quantization, spatial-wise modulation, and attention map shifting. In the first row, the result with no modules is presented, which is equivalent to the baseline DINOSAUR model. The last row is the performance of using all four modules, equivalent to our proposed method. The second and third rows are for the ablation of vector quantization and channel-wise modulation, demonstrating that incorporating top-down semantic information significantly improves performance. In the fourth and fifth rows, the ablation of attention map shifting and spatial-wise modulation is presented. The results show that both operations are critical for the effective exploitation of top-down spatial information.

Table 6: Ablation studies on COCO dataset for each module consisting of the proposed top-down pathway: channel-wise modulation ($\boldsymbol{m}^c$), vector quantization (`VQ`), spatial-wise modulation ($\boldsymbol{m}^s$), and shifting attention map (`shift`).

$\boldsymbol{m}^c$	`VQ`	$\boldsymbol{m}^s$	`shift`	FG-ARI	mBO
DINOSAUR				34.8	30.5
✓		✓	✓	36.3	32.3
		✓	✓	35.1	32.5
✓	✓	✓		35.2	31.7
✓	✓			36.0	31.9
✓	✓	✓	✓	**37.3**	**32.7**

5 Conclusion

In this paper, we introduced an OCL framework that incorporates top-down information into the slot attention mechanism through a *top-down pathway*. In this pathway, the output of the slot attention is used to bootstrap high-level semantic knowledge and rough localization cues for existing objects. Using the bootstrapped top-down knowledge, slot attention is modulated to focus on features most relevant to the objects in the scene. Consequently, by incorporating the proposed *top-down pathway* into slot attention, we achieved state-of-the-art performance on various OCL benchmarks, including challenging synthetic and authentic datasets.

Limitation The proposed *top-down pathway* has a limitation in that its overall performance relies on the quality of the codebook learned during training. As shown in Tab. 4, an incorrect choice of codebook size can result in the codes failing to learn distinct semantic concepts or capturing irrelevant details. While we mitigate this limitation through perplexity-based automatic codebook size tuning, the more principled codebook design that can eliminate the need for a pre-defined hyperparameter, such as dynamically expanding codebook during learning [40], will be promising future research direction.

Acknowledgments and Disclosure of Funding

This work was supported by IITP grants funded by the Korea government (MSIT) (RS-2019-II191906 Artificial Intelligence Graduate School Program (POSTECH); RS-2024-00457882 AI Research Hub Project; RS-2024-00509258 Global AI Frontier Lab).

References

[1] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086, 2018.

[2] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[3] O. Biza, S. Van Steenkiste, M. S. Sajjadi, G. F. Elsayed, A. Mahendran, and T. Kipf. Invariant slot attention: Object discovery with slot-centric reference frames. 2023.

[4] T. J. Buschman and E. K. Miller. Top-down versus bottom-up control of attention in the prefrontal and posterior parietal cortices. *science*, 315(5820):1860–1862, 2007.

[5] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[6] M. Chang, T. Griffiths, and S. Levine. Object representations as fixed points: Training iterative refinement algorithms with implicit differentiation. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[7] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[8] M. Corbetta and G. L. Shulman. Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215, 2002.

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

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2021.

[11] G. Elsayed, A. Mahendran, S. Van Steenkiste, K. Greff, M. C. Mozer, and T. Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 35, pages 28940–28954, 2022.

[12] M. Engelcke, A. R. Kosiorek, O. P. Jones, and I. Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations (ICLR)*, 2019.

[13] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[14] 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.

[15] K. Greff, F. Belletti, L. Beyer, C. Doersch, Y. Du, D. Duckworth, D. J. Fleet, D. Gnanapragasam, F. Golemo, C. Herrmann, et al. Kubric: A scalable dataset generator. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3749–3761, 2022.

[16] M. Huh, B. Cheung, P. Agrawal, and P. Isola. Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *International Conference on Machine Learning (ICML)*. PMLR, 2023.

[17] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

 https://doi.org/10.52202/079017-3296

[18] B. Jia, Y. Liu, and S. Huang. Improving object-centric learning with query optimization. In *International Conference on Learning Representations (ICLR)*, 2023.

[19] J. Jiang, F. Deng, G. Singh, and S. Ahn. Object-centric slot diffusion. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[20] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910, 2017.

[21] I. Kakogeorgiou, S. Gidaris, K. Karantzalos, and N. Komodakis. Spot: Self-training with patch-order permutation for object-centric learning with autoregressive transformers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22776–22786, 2024.

[22] L. Karazija, I. Laina, and C. Rupprecht. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[23] D. Kim, N. Kim, and S. Kwak. Improving cross-modal retrieval with set of diverse embeddings. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023. doi: 10.1109/cvpr52729.2023.02243. URL `http://dx.doi.org/10.1109/CVPR52729.2023.02243`.

[24] D. Kim, N. Kim, C. Lan, and S. Kwak. Shatter and Gather: Learning Referring Image Segmentation with Text Supervision. In *IEEE International Conference on Computer Vision (ICCV)*, 2023.

[25] J. Kim, J. Choi, H.-J. Choi, and S. J. Kim. Shepherding slots to objects: Towards stable and robust object-centric learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19198–19207, 2023.

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[27] T. Kipf, G. F. Elsayed, A. Mahendran, A. Stone, S. Sabour, G. Heigold, R. Jonschkowski, A. Dosovitskiy, and K. Greff. Conditional object-centric learning from video. In *International Conference on Learning Representations (ICLR)*, 2022.

[28] D. Lee, C. Kim, S. Kim, M. Cho, and W.-S. Han. Autoregressive Image Generation using Residual Quantization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[30] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf. Object-centric learning with slot attention. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 33, pages 11525–11538, 2020.

[31] S. Löwe, P. Lippe, F. Locatello, and M. Welling. Rotating features for object discovery. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, 2024.

[32] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations (ICLR)*, 2017.

[33] M. Seitzer, M. Horn, A. Zadaianchuk, D. Zietlow, T. Xiao, C.-J. Simon-Gabriel, T. He, Z. Zhang, B. Schölkopf, T. Brox, and F. Locatello. Bridging the gap to real-world object-centric learning. In *International Conference on Learning Representations (ICLR)*, 2023. URL `https://openreview.net/forum?id=b9tUk-f_aG`.

[34] B. Shi, T. Darrell, and X. Wang. Top-down visual attention from analysis by synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2102–2112, 2023.

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

[36] G. Singh, Y.-F. Wu, and S. Ahn. Simple unsupervised object-centric learning for complex and naturalistic videos. volume 35, pages 18181–18196, 2022.

[37] G. Singh, Y. Kim, and S. Ahn. Neural systematic binder. In *International Conference on Learning Representations (ICLR)*, 2023.

[38] A. Van Den Oord, O. Vinyals, et al. Neural discrete representation learning. 2017.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[40] M. Wallingford, A. Kusupati, A. Fang, V. Ramanujan, A. Kembhavi, R. Mottaghi, and A. Farhadi. Neural radiance field codebooks. 2023.

[41] X. Wang, R. Girdhar, S. X. Yu, and I. Misra. Cut and learn for unsupervised object detection and instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3124–3134, 2023.

[42] Z. Wang, M. Z. Shou, and M. Zhang. Object-centric learning with cyclic walks between parts and whole. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

[43] N. Watters, L. Matthey, C. P. Burgess, and A. Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. 2019.

[44] T. Webb, S. S. Mondal, and J. D. Cohen. Systematic visual reasoning through object-centric relational abstraction. volume 36, 2024.

[45] Z. Wu, N. Dvornik, K. Greff, T. Kipf, and A. Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. 2023.

[46] Z. Wu, J. Hu, W. Lu, I. Gilitschenski, and A. Garg. Slotdiffusion: Object-centric generative modeling with diffusion models. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, pages 50932–50958, 2023.

[47] Y. Yang and B. Yang. Promising or elusive? unsupervised object segmentation from real-world single images. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 35, pages 4722–4735, 2022.

[48] Z. Yin, P. Wang, F. Wang, X. Xu, H. Zhang, H. Li, and R. Jin. Transfgu: a top-down approach to fine-grained unsupervised semantic segmentation. In *European Conference on Computer Vision (ECCV)*, pages 73–89. Springer, 2022.

[49] J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu. Vector-quantized image modeling with improved vqgan. In *International Conference on Learning Representations (ICLR)*, 2021.

[50] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

 https://doi.org/10.52202/079017-3296

Appendices

A Additional Qualitative Results

Fig. A6 demonstrates the additional visualizations of original image, mask predictions, and slot attention maps $\boldsymbol{A}$ before and after self-modulation on the COCO dataset.

B Details of Autoregressive Decoder

The proposed *top-down pathway* is implemented based on the DINOSAUR baseline [33], using an autoregressive slot decoder. Singh et al. [35] first proposed to use such an autoregressive decoding scheme for slot attention training. Autoregressive decoder is known to provide better training signal leading to improved performance, compared to the MLP-based broadcast decoder [43] used by the slot attention originally. Autoregressive decoder is the simple variant of the transformer decoder [39], which takes input visual feature $\mathbf{x}$ and slots $\boldsymbol{S}$. The decoder consists of multiple decoding blocks. Let multi-head attention be denoted as $\texttt{MHA}(\boldsymbol{Q}; \boldsymbol{K}; \boldsymbol{V})$, where $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ is for query, key, and value, respectively. Then, the decoding block can be represented as:

$$\texttt{DecBlock}(\mathbf{x}; \boldsymbol{S}) = \texttt{FFN}(\texttt{MHA}(\tilde{\mathbf{x}}; \boldsymbol{S}; \boldsymbol{S})), \tag{11}$$

$$\tilde{\mathbf{x}} = \texttt{MHA}_{<}(\mathbf{x}_{\texttt{[BOS]}}; \mathbf{x}_{\texttt{[BOS]}}; \mathbf{x}_{\texttt{[BOS]}}), \tag{12}$$

where $\texttt{FFN}$ denotes a feedforward layer with MLP and residual connection, $\texttt{MHA}_{<}(\cdot)$ represents multi-head self-attention with causal masking, and $\mathbf{x}_{\texttt{[BOS]}}$ represents the visual feature sequence with a learnable $\texttt{[BOS]}$ token appended at the start of the sequence. By using multiple decoding blocks, we can compute the autoregressive reconstruction of the visual feature $\mathbf{x}$, which is consequently used for the computing reconstruction objective. Following DINOSAUR, we use an autoregressive decoder with four decoding blocks. The number of heads for multi-head attention is set to 8.

C Additional Experiments

***Top-down pathway* without DINO and autoregressive decoder** We have mainly built a *top-down pathway* with the DINO [5] pretrained weight and autoregressive decoder (framework proposed in DINOSAUR [33]) since it is the best working object-centric learning framework in a real-world setting. To see if these settings are essential for the *top-down pathway*, we have implemented our self-modulation technique with the original slot attention setting [30], which includes training encoders from scratch and using an image reconstruction objective with spatial broadcast decoder [43]. Tab. A7 summarizes the results of the CLEVR6 dataset. We observe a significant improvement in mBO, showing that our self-modulation technique is applicable to slot attention and provides complementary benefits. Although FG-ARI decreased, mBO is considered more robust when evaluating model performance [25, 12, 21, 33]. We have also included qualitative results in Fig. A5, which demonstrates that self-modulation markedly improves segmentation. These results indicate our method's effectiveness with different encoder configurations and training objectives.

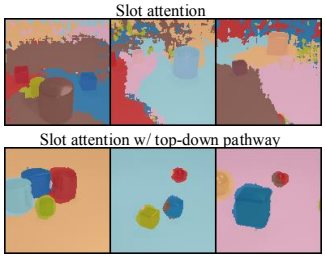

Figure A5: Visualization of the predicted object mask on CLEVR6 [20].

Comparison to MaskCut [41] While object-centric learning (OCL) and unsupervised instance segmentation share the goal of discovering objects without supervision, their ultimate objectives differ. OCL aims to learn object-wise representations that support downstream tasks requiring compositionality and systematic generalization, whereas unsupervised instance segmentation focuses primarily on obtaining accurate object masks. Nevertheless, we can directly compare methods from both tasks on their object discovery capabilities. We evaluate our method against MaskCut, the pseudo-mask generation algorithm underlying CutLER [41], on the COCO dataset. As shown in Tab. A8, our method significantly outperforms MaskCut across all metrics, demonstrating its effectiveness for object discovery even when compared to specialized unsupervised segmentation approaches.

https://doi.org/10.52202/079017-3296

Table A7: Comparison with slot attention [30] on CLEVR6

	FG-ARI	mBO
Slot attn (reprod.) [30]	98.7	21.2
Slot attn (reprod.) + top-down pathway	88.7	61.9

Table A8: Comparison with MaskCut [41] on COCO

	FG-ARI	mBO_i	mIoU
MaskCut [41]	31.5	28.9	26.7
Ours	37.4 ± 0.0	33.0 ± 0.3	31.2 ± 0.3

Table A9: Comparison with SPOT [21] on COCO [29], VOC [14], MOVI-C [15], and MOVI-E [15].

Method	**COCO**		**VOC**		**MOVI-C**		**MOVI-E**	
	FG-ARI	mBO^i	FG-ARI	mBO^i	FG-ARI	mBO^i	FG-ARI	mBO^i
SPOT [21]	36.6 ±0.3	**34.7** ±0.1	19.4 ±0.7	**48.1** ±0.4	52.1 ±3.3	**47.0** ±1.2	56.4 ±4.1	**39.9** ±1.1
Ours	**37.4**±0.0	33.3±0.3	**26.7**±4.7	43.9±2.6	**58.9**±5.1	46.8±2.4	**59.7**±3.1	39.3±1.8

Comparison with SPOT [21] In Tab. A9, we present the comparison with SPOT [21], a recent state-of-the-art object-centric learning method. SPOT proposed various ideas that can improve the quality of object-centric representation, such as patch order permutation within autoregressive decoder and self-distillation. We want to emphasize that these ideas are all orthogonal to the proposed *top-down pathway* and can be used together for further improvement, which we will leave as future work.

https://doi.org/10.52202/079017-3296

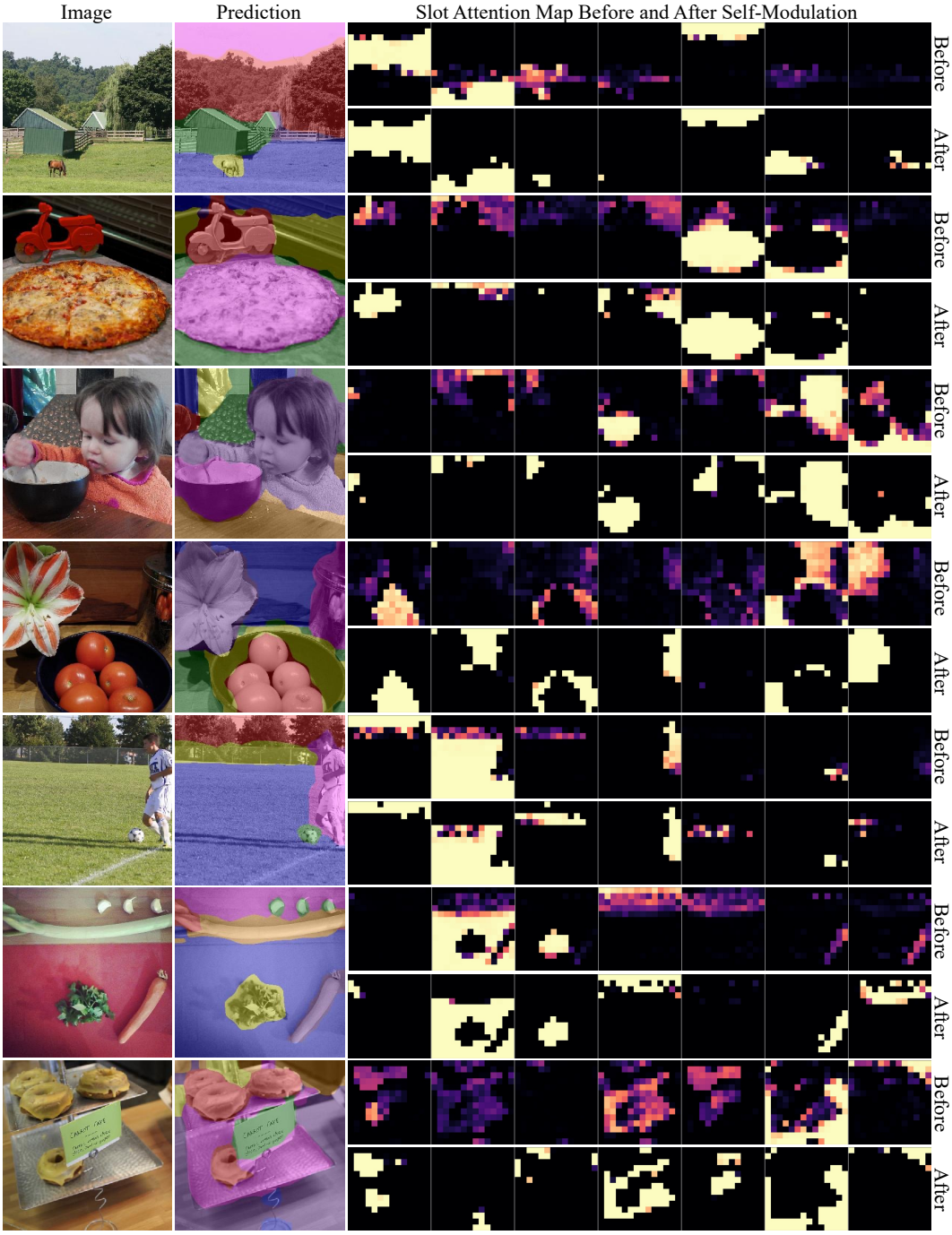

Figure A6: Visualization of the input image, predicted object mask, and attention maps of slot attention before and after self-modulation on COCO [29]. Lighter the color, higher the attention score.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We explain that we tackle object-centric learning (OCL) with a novel top-down pathway that bootstraps top-down information and transforms slot-attention into a self-modulating module. We also explain that we obtain impressive results on major OCL benchmarks.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We explain the limitations of our approach in the last paragraph of the conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

https://doi.org/10.52202/079017-3296

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We describe the model architecture in our method section and also the implementation details in the experiments section.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code will be made public for open access after publication.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We include these in the experiments section: data splits in datasets and hyperparameters in implementation details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide the standard deviation of the reported performances in table 1, 2, and 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

https://doi.org/10.52202/079017-3296

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include the specifications of the GPU we used to run our experiments in the experiments section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We comply with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: To our best knowledge, our work does not have any direct societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our model does not have high risk for misuse and we do not release a dataset.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite the baseline we used (DINOSAUR). We also cite the datasets and provide their licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

https://doi.org/10.52202/079017-3296

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We are not submitting datasets, code, nor models weights.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not conduct crowdsourcing experiments nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: There were no study participants.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-3296