Dyn-O: Building Structured World Models with Object-Centric Representations

Zizhao Wang[1,2*]   **Kaixin Wang**[2]   **Li Zhao**[2]   **Peter Stone**[1,3]   **Jiang Bian**[2]

[1]The University of Texas at Austin   [2]Microsoft Research Asia   [3]Sony AI

zizhao.wang@utexas.edu,pstone@cs.utexas.edu
{kaixwang,lizo,jiang.bian}@microsoft.com

Abstract

World models aim to capture the dynamics of the environment, enabling agents to predict and plan for future states. In most scenarios of interest, the dynamics are highly centered on interactions among objects within the environment. This motivates the development of world models that operate on object-centric rather than monolithic representations, with the goal of more effectively capturing environment dynamics and enhancing compositional generalization. However, the development of object-centric world models has largely been explored in environments with limited visual complexity (such as basic geometries). It remains underexplored whether such models can be effective in more challenging settings. In this paper, we fill this gap by introducing Dyn-O, an enhanced structured world model built upon object-centric representations. Compared to prior work in object-centric representations, Dyn-O improves in both learning representations and modeling dynamics. On the challenging Procgen games, we demonstrate that our method can learn object-centric world models directly from pixel observations, outperforming DreamerV3 in rollout prediction accuracy. Furthermore, by decoupling object-centric features into dynamic-agnostic and dynamic-aware components, we enable finer-grained manipulation of these features and generate more diverse imagined trajectories. The code of Dyn-O can be found at: `https://github.com/wangzizhao/dyn-O`.

1 Introduction

World models have emerged as powerful tools for simulating environment dynamics, enabling agents to predict and plan for the future [16, 17, 20, 21, 39]. A common design in these models is to map high-dimensional observations (*e.g.*, images) into latent features and then model the environment's dynamics in this latent space. However, these latent features are typically monolithic, encoding the entire scene as a whole without accounting for its internal compositional structure. Yet, interactions in most environments are inherently object-centric. This observation motivates the development of object-centric world models, which can offer improved efficiency, interpretability, and compositional generalization.

Building object-centric world models involves two key steps: 1) learning object-centric representations and 2) modeling dynamics on top of them, as illustrated in Figure 1. For the former, the goal is to encode features associated with each object present in the observation. For the latter, the dynamics model should account for the different representation space, in contrast to a monolithic representation, the input and the prediction target is a set of object-centric representations. While several prior approaches have explored object-centric world models, they primarily focus on simple environments consisting of basic shapes [2, 13, 32, 42], or rely on externally provided compositional

---

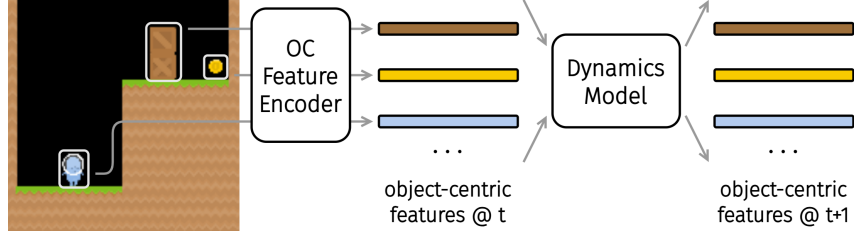[*]work done during an internship at Microsoft Research Asia

Figure 1: A high-level overview of the object-centric (OC) world model framework. The latent features are not monolithic or patch-based, but instead are bound to the objects present in the scene.

signals such as language [46]. It remains underexplored whether object-centric world models can be learned purely from trajectories to capture complex dynamics in more challenging, complex settings.

As a step toward bridging this gap, we introduce **Dyn-O**, a novel object-centric world model with improved designs for both object-centric representation learning and dynamics modeling over prior work. To handle complex visual observations, we draw inspiration from prior work on learning object-centric representations from real-world videos (*e.g.*, SOLV [1]) and adopt an autoencoder-style architecture to embed observations into object-associated features, referred to as *slots*. Specifically, we adopt a pre-trained Cosmos encoder [12], which offers two key advantages compared to the DINO encoder [34] used in SOLV: (1) improved representation quality, and (2) access to a pretrained visual decoder, eliminating the need to train a decoder from scratch for reconstructing pixel observations. To further enhance the quality of the extracted slot features, we incorporate priors from a high-performing pretrained segmentation model, SAM2 [35]. While this incorporation significantly improves performance, it also introduces substantial computational overhead, particularly during inference. To mitigate this issue, we use a scheduling strategy that gradually reduces reliance on the segmentation mask during training, enabling the model to maintain performance at inference time without requiring the mask.

The extracted slot features are modular in nature, with each slot associated with a distinct object. To model transition dynamics in this slot space, we aim for the world model to respect this modular structure. To this end, we adopt a state-space model (SSM) based on the Mamba architecture [15], borrowing the idea from SlotSSM [23]. Unlike SlotSSM, which tightly integrates the slot encoder and decoder within each transformer block of the SSM, we use the pretrained slot representation module introduced above and apply the SSM solely for dynamics modeling. This decoupling of representation learning and dynamics modeling aligns with recent trends in the world model literature [9, 29, 30]. Additionally, we explore disentangling each object's slot feature into a static component capturing time-invariant properties (*e.g.*, texture) and a dynamic component encoding time-varying properties (*e.g.*, position). This disentanglement enables fine-grained manipulation of slot features. For instance, we can modify the static feature of one object while keeping its dynamics unchanged, allowing for diverse data generation during world model rollouts.

We evaluate Dyn-O in seven Procgen [7] environments. Our experiments indicate that Dyn-O learns high-quality object-centric representations, generalizable world models, and disentangled static and dynamic representations. We summarize our contributions as follows.

- We propose a novel object-centric representation learning method by leveraging segmentation masks with a dropout schedule, enhancing representation quality while keeping efficient inference.

- We propose a novel object-centric world model that uses state-space models as backbones and outperforms monolithic models in both rollout quality and generalization.

- We propose a novel object-centric representation that disentangles each object's representation into static and dynamic features, allowing the generation of diverse rollouts by altering static attributes while preserving dynamic behavior.

2   Related Work

World Models   Accurately and efficiently modeling the world's dynamics has been a long-standing topic in reinforcement learning [39]. A learned model of a simulated environment can be used to

facilitate planning or generate imagined data for training a policy. In recent years, world models have seen great progress powered by advances in deep neural networks. Ha and Schmidhuber introduce a generative world model where the dynamics is learned entirely in a latent feature space. In a similar fashion, the Dreamer line of work [17–19] has further advanced the performance of world models. More recently, several works have explored using sequence modeling techniques to train world models within a token representation space [9, 29, 30].

Our works differs in that we build a world model on top of an object-centric representation, while prior work mainly focuses on monolithic representations. Closely related to our work are those that also consider the compositional nature of the world. Specifically, HOWM [45] and Cosmos [36] design object-centric world models to address the compositional generalization problem. DreamWeaver [2] uses a novel Recurrent Block-Slot Unit to discover compositional representations and generate compositional future simulations. However, those approaches mainly focus on small-scale diagnostic environments such as 2D block pushing, without validating in more complicated environments. RoboDreamer [46] learns a compositional world for robot manipulation tasks, but their method relies on language and leverages its natural compositionality. In comparison, our method learns the object-centric world model purely from agent trajectories.

Object-Centric Representations   Object-centric representations have gained increasing attention in recent years [6, 11, 22, 28, 37]. Instead of encoding an image observation as a single monolithic latent vector, object-centric representation learning aims to decompose a scene into objects and to learn different latent representations for each object. A key milestone in this direction was Slot Attention [28], which has since become the foundation for many subsequent methods that learn object-centric features in an unsupervised manner from images [22, 37, 43] or videos [1, 10, 25].

Our work builds on the SOLV framework [1], introducing improvements to enhance the extraction of object-specific latent features. Another closely related approach is the Slot State Space Model (SlotSSM) [23], which uses state-space models to capture long-range temporal dependencies in a structured way. However, it has only been validated on domains with limited complexity, such as basic shapes. Moreover, SlotSSM tightly couples slot representation learning and dynamics modeling, training them jointly. In contrast, we first learn a strong slot feature encoder and then build object-centric world models on top of it.

Disentangling Static and Dynamic Features   The disentanglement between static and dynamic features in sequences has been studied in computer vision [5, 31, 38]. DSVAE [44] divides the latent representation into the static factors and the dynamic factors and learns them jointly with the ELBO objective. However, as formalized by Locatello et al. [27], unsupervised disentanglement is impossible to achieve without inductive biases. C-DSVAE [3] further adds contrastive learning to encourage disentanglement, yet its learned dynamic features highly depend on the chosen data augmentations and can still capture static information. ContextWM [41] incorporates the notion of time-invariant context and time-varying latent variables into world models. However, using the same ELBO objectives as DSVAE, it may still learn entangled representations.

3   Method

From a high level perspective, Dyn-O models the environment's dynamics in an object-centric manner, enabled by two learning phases (Figure 2).

- **Object-centric representation learning** (Section 3.1): Dyn-O learns an object-centric representation in which coherent objects are each encoded into independent features, referred to as *slots*. This factorized representation, in contrast to a monolithic scene-level encoding, leverages the natural compositionality of objects in the world.
- **Dynamics learning** (Section 3.2): Dyn-O adopts a State Space Model (SSM) to predict transitions from the current slots to the next-step slots, conditioned on the action. Each object's slot feature is further decoupled into static and dynamic components to enable fine-grained manipulation.

3.1   Extracting Object-Centric Representations

In the first learning phase, Dyn-O learns an object-centric representation from image observations. In contrast to a monolithic representation that mixes all objects' information, this factored representa-
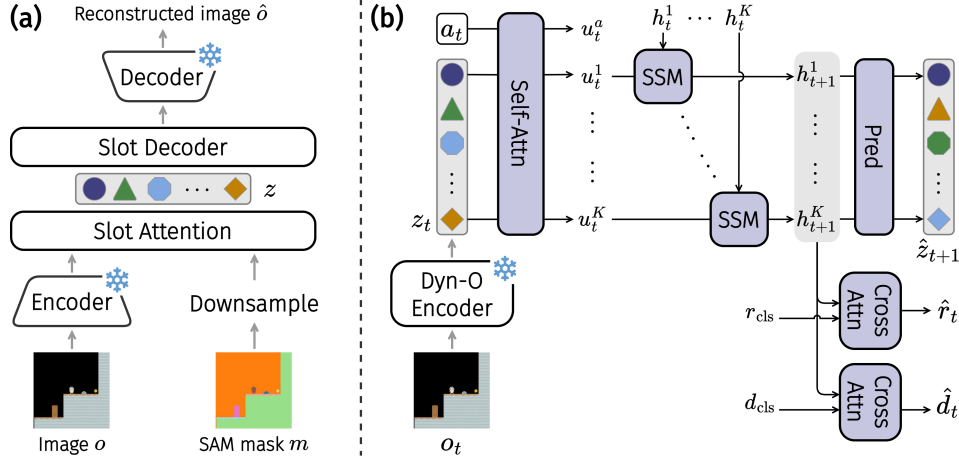
Figure 2: Components of Dyn-O: **(a)** object-centric representation learning; **(b)** dynamics learning. Modules marked with ❄ are fixed, while others are learnable. The "Dyn-O Encoder" in (b) corresponds to the lower half of (a), which maps the image o to the latent slot feature z. See Section 3 for details.

tion enables Dyn-O to modify each object independently and generate novel object combinations. Formally, as shown in Figure 2(a), Dyn-O learns an encoder $\texttt{Enc} : \mathcal{O} \rightarrow \mathcal{Z}$ to extract an object-centric representation $z \in \mathcal{Z}$ from observations o, where the representation consists of K slots, $z = [z^1, \ldots, z^K]$, each corresponding to an object. Here, K is a predefined hyperparameter, and when the number of objects in a scene is smaller than K, some slots may remain unused.

During encoder learning, inspired by SOLV [1], instead of training from scratch with raw pixels, Dyn-O learns on top of the high-quality features extracted by the Cosmos tokenizer [12]. Given an input frame $o \in \mathbb{R}^{H \times W \times 3}$, Dyn-O first applies the Cosmos encoder ($\texttt{CosmosEnc}$) to extract patch-level features $f \in \mathbb{R}^{N \times d_f}$, where N denotes the number of patches in the frame. It then initializes K slots $z_{\text{init}} \in \mathbb{R}^{K \times d_z}$ from a set of learnable vectors. These slots compete to bind to objects using iterative Slot Attention [28], producing $z = \texttt{Slot-Attn}(z_{\text{init}}, f)$. The learning signals for slot extraction arise from reconstructions – a decoder ($\texttt{Slot-Dec}$) reconstructs features $\hat{f}$ from the slots, which are then used to reconstruct the observation via the Cosmos decoder as $\hat{o} = \texttt{CosmosDec}(\hat{f})$. That is,

$$f = \texttt{CosmosEnc}(o),$$
$$z = \texttt{Slot-Attn}(z_{\text{init}}, f),$$
$$\hat{f} = \texttt{Slot-Dec}(z),$$
$$\hat{o} = \texttt{CosmosDec}(\hat{f}).$$

The $\texttt{Slot-Attn}$ and $\texttt{Slot-Dec}$ modules are trained jointly by minimizing the reconstruction error as follows, while the Cosmos encoder-decoder remains frozen:

$$\mathcal{L}_{\text{slot}} = \|f - \hat{f}\|^2 + \|o - \hat{o}\|^2. \tag{1}$$

Empirically, we observe that this fully unsupervised training objective usually results in inaccurate object-slot bindings (see examples in Section 4.1). To improve the quality of object-centric representations, we leverage the prior provided by foundational segmentation models. Specifically, we use a pre-trained SAM2 model [35] to generate a segmentation mask $m \in \{0, 1\}^{H \times W \times K}$ and use it as an attention mask during slot attention, $z = \texttt{Slot-Attn}(z_{init}, f, m)$, binding each slot to one segmented object and constraining it to only attend to patches from that object.

While the segmentation mask enhances object-slot binding, it also introduces substantial computational overhead, especially during inference. This dependency could limit the practicality of Dyn-O. For example, when extracting slots for an agent interacting with the environment in an online setting, we would need to run SAM2 inference at every environment step, which would be prohibitively expensive compared to the typical cost per step. To address this issue, during encoder learning, we
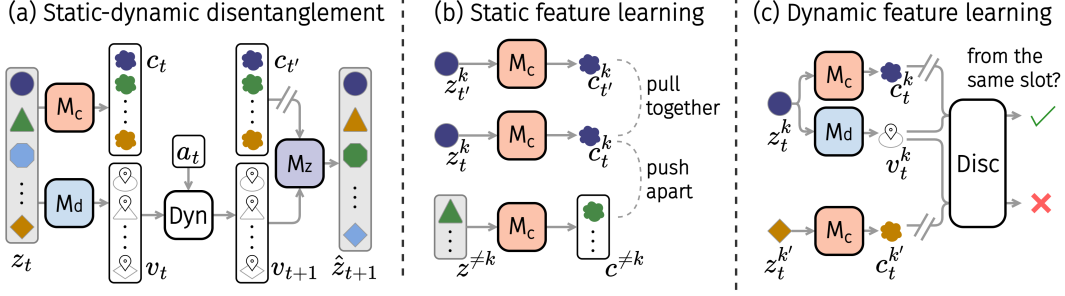
4

Figure 3: Illustration of **(a)** the overall design for disentangling slot features in dynamics modeling, and the training procedures for **(b)** static features and **(c)** dynamic features. // indicates a stop-gradient operation.

introduce an annealing schedule that gradually reduces the reliance on the segmentation mask, aiming to eliminate its use by the end of training. Initially, the segmentation mask is always used to guide slot extraction. As training progresses, the probability of not using the mask increases according to a logarithmic schedule w.r.t. the number of network updates, as $\log(1 + \# \text{ updates}) / \log(1 + \# \text{ total updates})$. This dropout schedule allows the encoder to achieve the best of both worlds – it learns high-quality representations with the initial guidance of segmentation masks, while enabling efficient inference by phasing out the need for them.

3.2 World Model with Object-Centric Representations

After learning object-centric representations as described above, the next phase of Dyn-O focuses on training a world model to reason about object interactions. Given the history of slots $z_{\leq t}$ and actions $a_{\leq t}$, the world model predicts the next slots, the reward, and whether the episode terminates as $\hat{z}_{t+1}, \hat{r}_t, \hat{d}_t = \texttt{Dyn}(z_{\leq t}, a_{\leq t})$.

World Model For the world model $\texttt{Dyn}$, we adopt state-space models (SSMs) for their strength at capturing long-range temporal dependencies. Meanwhile, the model needs to account for the *permutation equivariance* among slots – if the input slots are randomly permuted, the slots' prediction should follow the same permutation. Therefore, as shown in Figure 2(b), we design $\texttt{Dyn}$ as follows. We first apply self-attention to extract information about object interactions:

$$u_t^k = \texttt{Self-Attn}(\texttt{q} = z_t^k, \texttt{kv} = [z_t^1, \ldots, z_t^K, a_t]),$$

where no position encoding is used to maintain permutation invariance. Next, each slot is processed by a shared SSM to update its hidden state, which is then used by Dyn-O to predict the next slots, reward, and whether the episode terminates using separate prediction networks as follow:

$$h_{t+1}^k = \texttt{SSM}(h_t^k, z_k^t), \quad \hat{z}_{t+1}^k = \texttt{Pred}(h_{t+1}^k),$$
$$\hat{r}_t = \texttt{Cross-Attn}(\texttt{q} = r_{\text{cls}}, \texttt{kv} = [h_{t+1}^1, \ldots, h_{t+1}^K]),$$
$$\hat{d}_t = \texttt{Cross-Attn}(\texttt{q} = d_{\text{cls}}, \texttt{kv} = [h_{t+1}^1, \ldots, h_{t+1}^K]),$$

where h^k is the hidden state of the SSM that tracks past information for the k-th slot, and r_{cls} and d_{cls} learnable query tokens used to extract reward and termination signals from the hidden states. Finally, the world model is optimized by minimizing the following prediction loss (Alg. 1 line 4) :

$$\mathcal{L}_{\text{wm}} = \sum_{t=1}^{T-1} \|\hat{z}_{t+1} - z_{t+1}\|^2 + \sum_{t=1}^{T} \left(\|\hat{r}_t - r_t\|^2 + \text{CE}(\hat{d}_t, d_t) \right), \qquad (2)$$

where CE stands for cross-entropy loss for binary classification.

Static-Dynamic Disentanglement The object-centric representation learned in Section 3.1 enables the creation of scenes with novel object combinations. However, when synthesizing data, it is often desirable to only modify object appearance (*e.g.*, colors) while preserving their dynamic properties

5

(*e.g.*, positions). For example, modifying dynamic information could result in invalid scenes, such as a table being moved while the objects originally on it remain suspended in midair.

To this end, as shown in Figure 3(a), Dyn-O disentangles each slot z^k into two components: static features $c^k \in \mathbb{R}^{d_c}$ that captures time-invariant properties and dynamic features $v^k \in \mathbb{R}^{d_v}$ that encodes time-variant properties. This decomposition allows us to modify an object's static features while keeping its dynamic features unchanged, generating novel scenarios with consistent dynamics. Formally, Dyn-O maps slots to static and dynamic features with two separate neural networks: $c^k = \mathtt{M}_c(z^k)$ and $v^k = \mathtt{M}_v(z^k)$. We describe how these features are learned below.

Static features are intended to capture time-invariant properties. Thus, for an object present at timestep t, its static feature should remain the same across all timesteps. Therefore, a natural training objective is to minimize the difference between static features across all timesep pairs. However, this objective alone admits a degenerate solution where all static features collapse to a constant. To prevent this collapse, Dyn-O incorporates contrastive learning to encourage feature diversity, as illustrated in Figure 3(b): static features should be similar if they belong to the same slot, and distinct otherwise. Dyn-O therefore learns static features by optimizing the following loss (Alg. 1 line 5):

$$\mathcal{L}_{\text{stat}} = \underbrace{\sum_{k,t \neq t'} \|c_t^k - c_{t'}^k\|^2}_{\text{time invariance}} + \underbrace{\sum_{k,t} -\log \frac{\cos(c_t^k, c_{t'}^k)}{\cos(c_t^k, c_{t'}^k) + \sum_{\tilde{c} \in \tilde{C}} \cos(c_t^k, \tilde{c})}}_{\text{contrastive learning}}, \tag{3}$$

where $\cos$ denotes cosine similarity, and the second term corresponds to the InfoNCE loss [33]. For each c_t^k, the positive sample $c_{t'}^k$ comes from the same slot at a different timestep, while the negative samples $\tilde{C}$ are drawn from other slots in the batch.

On the other hand, Dyn-O learns dynamic features by reconstructing the slot content while ensuring their disentanglement from static features. Specifically, Dyn-O uses a reconstruction network $\mathtt{M}_z$ to output $\hat{z}^k = \mathtt{M}_z(\mathtt{sg}(c^k), v^k)$, where $\mathtt{sg}$ denotes stop-gradient, preventing the static feature c^k from being updated by the reconstruction loss. (For simplicity, in this paragraph, we omit the subscript t in z, c and v). However, minimizing reconstruction loss alone may lead the dynamic features to encode time-invariant information as well, bypassing the need for c^k. To avoid this effect, Dyn-O promotes disentanglement by minimizing the mutual information $\sum_{t,k} I(c_t^k, v_t^k)$ via adversarial training, as illustrated in Figure 3(c). A discriminator $\mathtt{Disc}$ is trained to distinguish whether a pair of static and dynamic features comes from the same slot. While $\mathtt{Disc}$ minimizes the discrimination loss, the dynamic feature extractor $\mathtt{M}_v$ is trained to maximize it, thereby reducing the static information encoded in dynamic features [14]. For stability, we adopt the Wasserstein distance as the discrimination loss and apply LeCam regularization [40] (Alg. 1, lines 6-7):

$$\mathcal{L}_{\text{dyn}} = \sum_k \underbrace{\|\hat{z}^k - z^k\|^2}_{\text{reconstruction}} + \underbrace{\mathtt{Disc}(c^k, v^k)}_{\text{disentanglement}}, \tag{4}$$

$$\mathcal{L}_{\text{disc}} = \sum_k \underbrace{\left(-\mathtt{Disc}(c^k, v^k) + \mathtt{Disc}(c^k, d^{k'}) \right)}_{\text{discrimination}} + \underbrace{\mathtt{LeCam}(\mathtt{Disc})}_{\text{regularization}}, \tag{5}$$

where (c^k, v^k) are from the same slot, and $(c^k, d^{k'})$ are from different slots.

4 Experimental Evaluation

We evaluate Dyn-O to answer the following questions: Q1: Can Dyn-O learn accurate object-centric representations (Sec. 4.1)? Yes. Q2: Can Dyn-O learn accurate world models (Sec. 4.3)? Yes. Q3: Are static and dynamic features learned by Dyn-O truly disentangled (Sec. 4.4)? Yes.

Our experiments are conducted in **Procgen** [7], a set of procedurally-generated 2D video game environments. We use 7 Procgen environments: bigfish, coinrun, caveflyer, dodgeball, jumper, ninja, and starpilot. In each Procgen environment, a PPG policy [8] is trained and used to collect an offline dataset of 1M transitions from the first 200 levels for the learning of all methods.

Algorithm 1 Dyn-O World Model Learning

1: Collect a dataset of (o_t, a_t, r_t, o_{t+1}). Initialize object-centric encoder (`Enc`), mappings to static features (`M`$_c$), dynamic features (`M`$_v$), and slots (`M`$_z$), the world model (`Dyn`), and the discriminator (`Disc`).
2: Train object-centric encoder `Enc` with Eq. (1).
3: Train the world model with joint optimization:
4:   Update the world model `Dyn` by minimizing prediction losses in Eq. (2).
5:   **if** learn static-dynamic disentanglement **then**
6:     Update static features from `M`$_c$ by enforcing time-invariance in Eq. (3).
7:     Update `Disc` with Eq. (4) to estimate mutual information.
8:     Update dynamic feature from `M`$_v$ via reconstruction `M`$_z$ and disentanglement loss in Eq. (5).

Method	FR-ARI ($\uparrow$)
Oracle	0.96
Dyn-O (ours)	0.80
SOLV	0.54

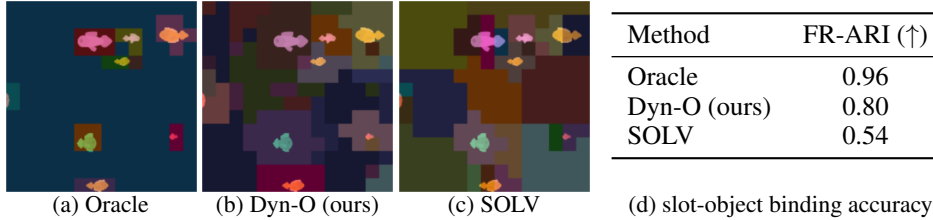(a) Oracle   (b) Dyn-O (ours)   (c) SOLV   (d) slot-object binding accuracy

Figure 4: Evaluation of the object-centric representation learning in **bigfish**. **(a)** Using segmentation masks during inference, Oracle achieves the most accurate slot-object binding, but also suffering from the high computational overhead. **(b)** By using segmentation masks only during training, Dyn-O learns to accurately assign each fish to one slot (shown as colored patches), which avoiding inference-time overhead. **(c)** In contrast, learning without the guidance of segmentation masks, SOLV often inaccurately splits a fish into multiple separate slots.

4.1 Evaluating Object-Centric Representation

As Dyn-O's world model is learned on top of the object-centric representations, the quality of learned representation is critical to its prediction accuracy. We compare Dyn-O's representation learning against the following ablations:

- **Oracle**: our method but always using the segmentation mask during both training and inference.
- **SOLV** [1]: the same as our method but does not use segmentation mask for training and inference.

As shown in Fig. 4, compared to SOLV that often splits an object into multiple slots, Dyn-O achieves more accurate object-slot binding, assigning each object to a single slot. We also evaluate the foreground adjusted rand index (FG-ARI) which is a widely used metric in the object-centric literature that measures the similarity of the discovered objects masks to ground-truth masks. Again, Dyn-O outperforms SOLV, suggesting the benefit of using segmentation masks to guide representation learning.

4.2 Evaluating World Model Accuracy

The promise of Dyn-O is to learn accurate and and generalizable world models based on object-centric representations. Therefore, the most critical evaluation of our work focuses on the world model quality, and we compare Dyn-O against the following baselines and ablations:

- **DreamerV3** [19]: one of the state-of-the-art model-based RL methods.
- **Dreamweaver** [2]: an object-centric world model designed to discover hierarchical and compositional representations.
- **Dyn-O without object-centric representations** (denoted as **Dyn-O w/o OC**): our method but without object-centric representations. The dynamics model is learned on top of 14x14 patch-level features extracted by Cosmos encoder, where each patch is treated as a "slot".

For a fair comparison, we use the same frozen Cosmos tokenizer for DreamerV3 and Dreamweaver as Dyn-O during world model learning.
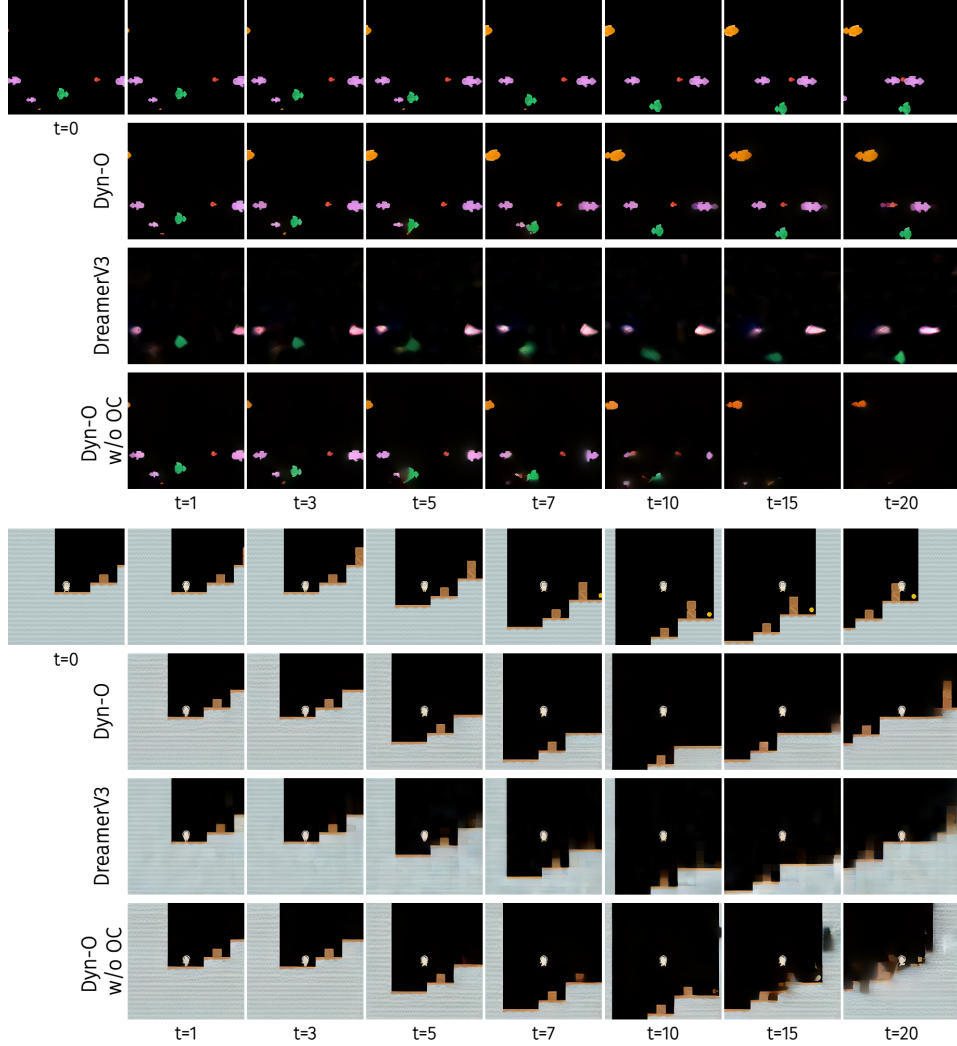
Figure 5: Dyn-O generates more accurate rollouts than Dreamer in **bigfish** and **coinrun**. In each environment, the first row displays a trajectory collected in the real environment. The second row depicts the prediction inside the world model by Dyn-O (ours). The third and fourth rows show the prediction of Dreamer and Dyn-O w/o OC respectively. In bigfish, our method keeps consistent prediction for each fish until the 15th step, while baselines lose track of multiple small fish before the 10th step. Similarly, in coinrun, compared to baselines, Dyn-O generates predictions with clearer floor and boxes.

To evaluate the generalizability of each method, we use the learned world model to generate 20-step rollouts in 500 unseen levels. The evaluation metric covers pixel-level quality (PSNR) as well as temporal and spatial coherence (FVD, LPIPS, and SSIM). The results are shown in Table 1, where we average the evaluation at the last step across all environments. The results for each environment can be found in Appendix B.3. Dyn-O significantly outperforms dreamer which uses a monolithic representation, demonstrating the advantage of predicting in the object-centric representation space. Meanwhile, in contrast to Dyn-O w/o OC whose latent representation consist of 196 patch-level "slots", Dyn-O only uses 31 or 47 object-level slots (where each object-level slot has the same dimension as a patch-level "slot") and achieves higher performance, demonstrating the superior efficiency of object-centric representations. The results for Dreamwaver can be found in Appendix B.3. To complement the analysis with qualitative examples, Fig. 5 shows the generated rollouts of Dyn-O and its comparison with baselines.
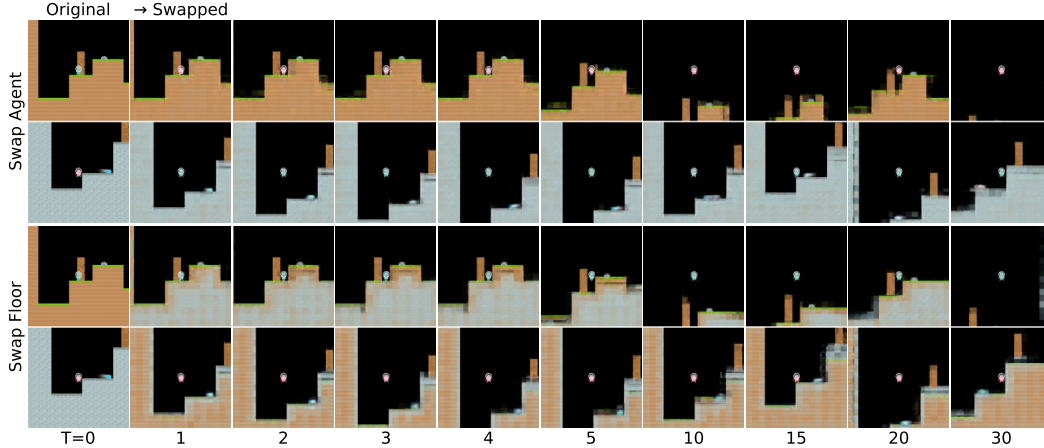
Figure 6: Dyn-O generates dynamically consistent rollouts after exchanging static features. In the top two rows, we swap the static features of the avatar (the small agent in the center of the image) between two initial states and generate 30-step rollouts using Dyn-O. The results show that only the avatar's color changes, while all other objects remain unchanged. In the bottom two rows, we exchange the static features of the floor, and Dyn-O consistently swaps their colors while keeping all other objects intact.

Table 1: Quantitative results for rollout trajectories, averaged across all environments.

Method	LPIPS ($\downarrow$)	FVD ($\downarrow$)	SSIM ($\uparrow$)	PSNR ($\uparrow$)
DreamerV3	0.42	692.5	0.56	15.70
Dyn-O w/o OC	0.41	538.4	0.53	16.10
Dyn-O (ours)	**0.33**	**361.3**	**0.62**	**16.34**

In addition to Procgen, we further compare Dyn-O against DreamerV3 on the CLEVR dataset [24] and two ALE environments [4], and the results are in Appendix B.3.

4.3 Evaluating Representation Effectiveness for Policy Learning

Next, we evaluate whether the learned object-centric representations can facilitate policy learning. To do so, we fix the representation backbone and train only the policy head using the PPG algorithm on three Procgen games: Bigfish, Starpilot, and Coinrun, with three random seeds for each game. We use the "easy" difficulty setting with an unlimited number of levels.

We compare the performance of policies using object-centric representations learned by Dyn-O against policies using representations directly output by the Cosmos encoder, and the policy performance (measured as reward) are shown in Table. 3. Policies using object-centric representations achieve much higher reward than policies with raw patch-level features, demonstrating the effectiveness of Dyn-O's representations for downstream task learning.

Table 2: Probing accuracy ($\uparrow$), in percentage (%), on **coinrun** privilege properties, shown the mean and standard deviation . Static features have much higher prediction accuracy than dynamic features for static properties (i.e., RGB values), while dynamic features have higher accuracy on dynamic properties (i.e., position and area), demonstrating that Dyn-O achieves effective static-dynamic disentanglement

	Static Properties			Dynamic Properties		
	R value	G value	B value	x position	y position	area
slots	**83.5** $\pm$ 0.0	**81.7** $\pm$ 0.0	**89.3** $\pm$ 0.0	**91.8** $\pm$ 0.0	**94.5** $\pm$ 0.0	**97.1** $\pm$ 0.0
dynamic features	47.7 $\pm$ 7.3	45.7 $\pm$ 7.3	47.0 $\pm$ 8.9	78.1 $\pm$ 5.1	75.2 $\pm$ 6.6	83.3 $\pm$ 3.1
static features	67.3 $\pm$ 1.4	70.9 $\pm$ 1.9	81.7 $\pm$ 2.2	34.9 $\pm$ 1.2	37.7 $\pm$ 2.0	75.3 $\pm$ 0.4
random features	25.8 $\pm$ 0.0	27.3 $\pm$ 0.0	23.2 $\pm$ 0.0	29.3 $\pm$ 0.0	28.3 $\pm$ 0.0	73.3 $\pm$ 0.0

Table 3: Reward of policies using different representations, measured by the mean and standard deviation across three random seeds.

Representation	Bigfish	Starpilot	Coinrun
Cosmos encoder	0.90 ± 0.02	8.80 ± 0.47	8.00 ± 0.18
Dyn-O (ours)	$\mathbf{7.67} \pm 4.02$	$\mathbf{19.19} \pm 0.88$	$\mathbf{8.95} \pm 0.38$

4.4 Evaluating Static-Dynamic Disentanglement

To examine whether the static and dynamic features learned by Dyn-O are disentangled, We probe the model with environment-privileged information, evaluating whether static features only capture time-invariant properties and whether dynamic features only capture time-varying properties. Specifically, we use SAM to segment out objects in 10K transitions and extract the following properties for each object as probing targets: area (i.e., the number of pixels), xy positions, and average RGB values across object pixels. Then we compute each object's **slot**, **static** feature, and **dynamic** feature as probing inputs. During probing, we discretize each property into 10 bins and use a linear classifier to predict the target. To establish a reference point for interpretation, we also perform the same prediction using randomly initialized features, representing the performance expected when no meaningful information is available in the inputs. Table 2 shows the prediction accuracy of all features in the coinrun environment (see results for other environments in the Appendix). Dyn-O shows strong disentanglement results, particularly as the prediction accuracy of static features for position and area is close to that of random features, indicating that static features capture little to no information about dynamic properties.

We further illustrate the disentanglement between static and dynamic features with qualitative examples. Fig. 6 presents rollouts generated after swapping static features between two initial states. As shown, rollouts from the same initial states (the first and third rows) remain nearly identical, except for the color changes in the avatar and floor. The same pattern is observed in the second and fourth rows, demonstrating that Dyn-O preserves dynamics while modifying only static properties.

5 Conclusion

We present Dyn-O, a world modeling method that builds on object-centric representations. By leveraging segmentation masks during training with a schedule, Dyn-O learns high-quality object-centric representations while avoiding the overhead of segmentation computation at inference time. Additionally, Dyn-O further disentangles each object feature into static, time-invariant components (e.g., color) and dynamic, time-variant components (e.g., position), allowing it to generate diverse predictions. Dyn-O predicts in latent space, with a pretrained decoder used for visualization. Exploring more sophisticated decoding methods, such as diffusion models, could further enhance the visual quality of generated rollouts.

Acknowledgements

References

[1] G. Aydemir, W. Xie, and F. Güney. Self-supervised Object-centric Learning for Videos. In *Advances in Neural Information Processing Systems*, 2023.

[2] J. Baek, Y.-F. Wu, G. Singh, and S. Ahn. Dreamweaver: Learning compositional world representations from pixels. *arXiv preprint arXiv: 2501.14174*, 2025.

[3] J. Bai, W. Wang, and C. P. Gomes. Contrastively disentangled sequential variational autoencoder. *Advances in Neural Information Processing Systems*, 34:10105–10118, 2021.

[4] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

[5] N. Berman, I. Naiman, I. Arbiv, G. Fadlon, and O. Azencot. Sequential disentanglement by extracting static information from a single sequence element. *arXiv preprint arXiv:2406.18131*, 2024.

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

[7] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.

[8] K. W. Cobbe, J. Hilton, O. Klimov, and J. Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pages 2020–2027. PMLR, 2021.

[9] L. Cohen, K. Wang, B. Kang, and S. Mannor. Improving token-based world models with parallel observation prediction. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Lfp5Dk1xb6.

[10] G. F. Elsayed, A. Mahendran, S. van Steenkiste, K. Greff, M. Mozer, and T. Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. *Neural Information Processing Systems*, 2022. doi: 10.48550/arXiv.2206.07764.

[11] M. Engelcke, A. R. Kosiorek, O. P. Jones, and I. Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv: 1907.13052*, 2019.

[12] N. et. al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[13] S. Ferraro, P. Mazzaglia, T. Verbelen, and B. Dhoedt. Focus: Object-centric world models for robotics manipulation. *arXiv preprint arXiv: 2307.02427*, 2023.

[14] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[15] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv: 2312.00752*, 2023.

[16] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

[17] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv: 1912.01603*, 2019.

[18] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *International Conference on Learning Representations*, 2020.

[19] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[20] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, Apr. 2025.

[21] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[22] J. Jiang, F. Deng, G. Singh, and S. Ahn. Object-centric slot diffusion. *Neural Information Processing Systems*, 2023. doi: 10.48550/arXiv.2303.10834.

[23] J. Jiang, F. Deng, G. Singh, M. Lee, and S. Ahn. Slot state space models. *arXiv preprint arXiv: 2406.12272*, 2024.

[24] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[25] 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. *arXiv preprint arXiv:2111.12594*, 2021.

[26] B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34, 2021.

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

[28] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf. Object-centric learning with slot attention, 2020.

[29] V. Micheli, E. Alonso, and F. Fleuret. Transformers are sample-efficient world models. *International Conference on Learning Representations*, 2022. doi: 10.48550/arXiv.2209.00588.

[30] V. Micheli, E. Alonso, and F. Fleuret. Efficient world models with context-aware tokenization. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=BiWIERWBFX`.

[31] Đ. Miladinović, M. W. Gondal, B. Schölkopf, J. M. Buhmann, and S. Bauer. Disentangled state space representations. *arXiv preprint arXiv:1906.03255*, 2019.

[32] A. Nakano, M. Suzuki, and Y. Matsuo. Interaction-based disentanglement of entities for object-centric world models. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=JQc2VowqCzz`.

[33] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[34] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

[35] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL `https://arxiv.org/abs/2408.00714`.

[36] A. Sehgal, A. Grayeli, J. J. Sun, and S. Chaudhuri. Neurosymbolic grounding for compositional world models. *International Conference on Learning Representations*, 2023. doi: 10.48550/arXiv.2310.12690.

[37] 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. *arXiv preprint arXiv: 2209.14860*, 2022.

[38] M. C. Simon, P. Frossard, and C. D. Vleeschouwer. Sequential representation learning via static-dynamic conditional disentanglement. In *European Conference on Computer Vision*, pages 110–126. Springer, 2024.

[39] R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.

[40] H.-Y. Tseng, L. Jiang, C. Liu, M.-H. Yang, and W. Yang. Regularing generative adversarial networks under limited data. In *CVPR*, 2021.

[41] J. Wu, H. Ma, C. Deng, and M. Long. Pre-training contextualized world models with in-the-wild videos for reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[42] Z. Wu, N. Dvornik, K. Greff, T. Kipf, and A. Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. *International Conference on Learning Representations*, 2022. doi: 10.48550/arXiv.2210.05861.

[43] Z. Wu, J. Hu, W. Lu, I. Gilitschenski, and A. Garg. Slotdiffusion: Object-centric generative modeling with diffusion models. *Advances in Neural Information Processing Systems*, 36: 50932–50958, 2023.

[44] L. Yingzhen and S. Mandt. Disentangled sequential autoencoder. In *International Conference on Machine Learning*, pages 5670–5679. PMLR, 2018.

[45] L. Zhao, L. Kong, R. Walters, and L. L. Wong. Toward compositional generalization in object-oriented world modeling. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26841–26864. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/zhao22b.html`.

[46] S. Zhou, Y. Du, J. Chen, Y. Li, D. Yeung, and C. Gan. Robodreamer: Learning compositional world models for robot imagination. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=kHjOmAUfVe`.

Table 4: The Architecture and Hyperparameters of Encoder Learning.

Name	Value						
	bigfish	caveflyer	coinrun	dodgeball	jumper	ninja	starpilot
# slots	31	63	31	31	31	47	47
slot dimension	256						
# slot attention iterations	3						
image size	[224, 224, 3]						
patch size	16						
optimizer	Adam						
learning rate	4e-4						
data size	1M						
epoch	15						
batch size	64						
Cosmos model	Cosmos-0.1-Tokenizer-CI16x16						

Table 5: The Architecture and Hyperparameters of Dynamics Learning.

Name	Value
dynamic feature dimension	256
static feature dimension	256
self-attention # layers	1
self-attention model size	512
self-attention # heads	8
SSM # layers	2
SSM model size	512
SSM d_{state}	64
SSM d_{conv}	4
optimizer	Adam
learning rate	1e-4
batch size	32

A   Method Details

A.1   Encoder

The architecture and hyperparameter used during encoder training are shown in Table. 4.

A.2   Dynamics

During dynamic feature training, the slot reconstruction loss and the disentanglement loss may have conflicting gradient and hinder model learning. To mitigate this issue, we leverage gradient modification [26] to enhance the balance between two objectives.

The architecture and hyperparameter used during encoder training are shown in Table. 5.

B   Experiment Details

B.1   Environment Details

Among the 16 Procgen environments, we selected 7 environments based on their relevance to object-centric learning and visual complexity. Specifically, we looked for environments with **moving objects**, **dynamic layouts**, and **minimal sensitivity to color**. Based on these criteria, we excluded maze-like environments such as Maze, Heist, Chaser, and Miner, as well as Plunder, where the ship's color plays a critical role. Among the remaining games, we randomly selected 7 while ensuring diversity across game types, given resource constraints.

Table 6: slot-object binding accuracy, measured by FR-ARI (↑).

Environments	Oracle	Dyn-O (ours)	SOLV
bigfish	0.96	0.80	0.54
coinrun	0.33	0.27	0.10
dogeball	0.79	0.48	0.17
starpilot	0.86	0.47	0.49
average	0.74	0.51	0.33

(a) Oracle   (b) Dyn-O (ours)   (c) SOLV

(d) Oracle   (e) Dyn-O (ours)   (f) SOLV

(g) Oracle   (h) Dyn-O (ours)   (i) SOLV

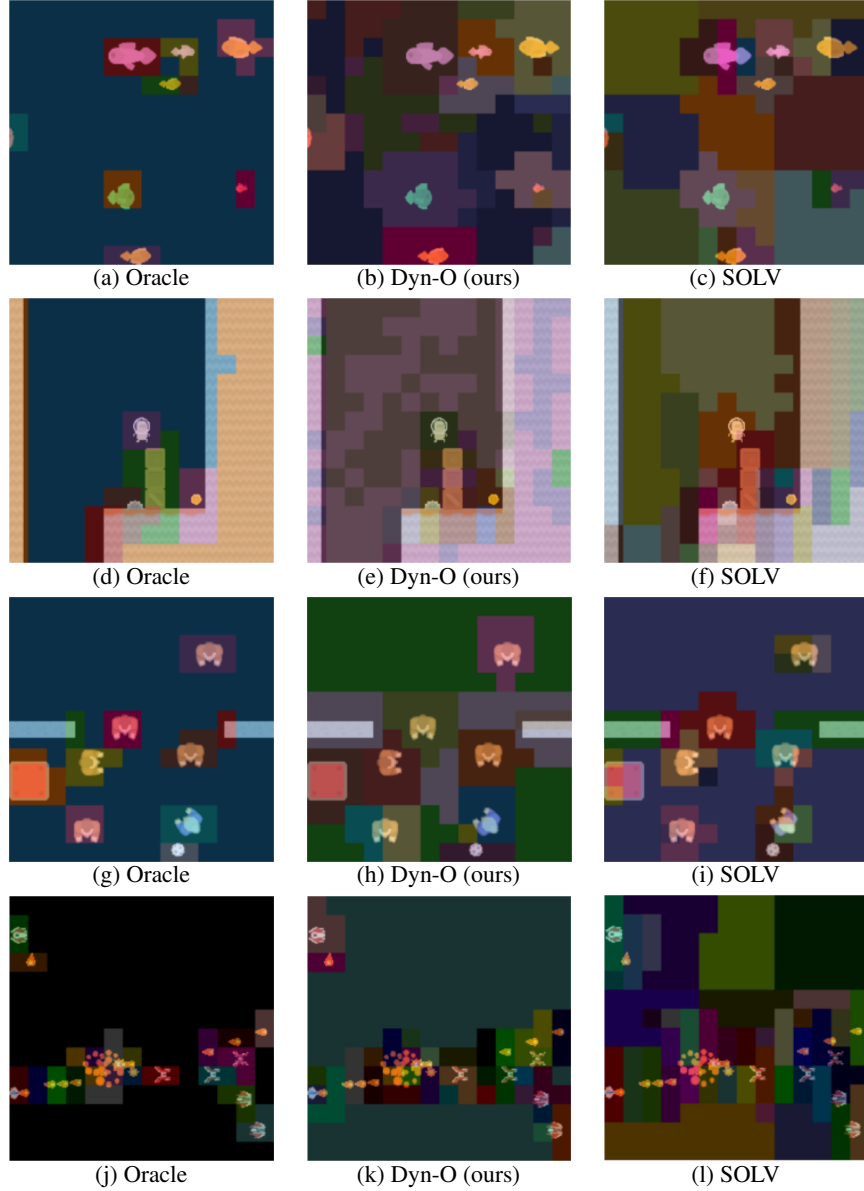(j) Oracle   (k) Dyn-O (ours)   (l) SOLV

Figure 7: Qualitative evaluation of the object-centric representation learning in **bigfish**, **coinrun**, **dodgeball**, and **starpilot**.

B.2 Evaluating Object-Centric Representation

The object-centric representation evaluation for 4 procgen environments are shown in Table. 6 and Fig. 7. By leveraging segmentation masks only during training, Dyn-O significantly outperforms SOLV in all environments, in terms of slot-object binding accuracy.

Table 7: Rollout accuracy for each Procgen environment at 20-th timestamp, measured as mean and standard error.

Environment	Metric	DreamerV3	Dreamerwaver	Dyn-O w/o OC	Dyn-O (ours)
bigfish	LPIPS (↓)	0.39 ± 0.01	0.52 ± 0.01	0.36 ± 0.01	$\mathbf{0.25} \pm 0.01$
	FVD (↓)	567.20 ± 11.18	831.17 ± 15.55	248.94 ± 16.32	$\mathbf{126.88} \pm 6.42$
	SSIM (↑)	0.73 ± 0.01	0.68 ± 0.00	0.68 ± 0.01	$\mathbf{0.76} \pm 0.01$
	PSNR (↑)	19.34 ± 0.14	19.09 ± 0.12	20.16 ± 0.27	$\mathbf{20.28} \pm 0.30$
caveflyer	LPIPS (↓)	$\mathbf{0.53} \pm 0.01$	–	0.59 ± 0.01	0.61 ± 0.01
	FVD (↓)	1104.50 ± 18.87	–	966.07 ± 28.71	$\mathbf{877.74} \pm 20.99$
	SSIM (↑)	$\mathbf{0.44} \pm 0.01$	–	0.26 ± 0.01	0.26 ± 0.01
	PSNR (↑)	$\mathbf{11.99} \pm 0.12$	–	10.93 ± 0.19	10.82 ± 0.13
coinrun	LPIPS (↓)	0.34 ± 0.01	–	0.36 ± 0.01	$\mathbf{0.28} \pm 0.01$
	FVD (↓)	530.31 ± 10.51	–	583.11 ± 16.07	$\mathbf{266.13} \pm 6.16$
	SSIM (↑)	0.60 ± 0.01	–	0.47 ± 0.01	$\mathbf{0.62} \pm 0.01$
	PSNR (↑)	$\mathbf{14.22} \pm 0.25$	–	12.61 ± 0.14	13.56 ± 0.19
dodgeball	LPIPS (↓)	0.42 ± 0.01	–	0.15 ± 0.01	$\mathbf{0.14} \pm 0.01$
	FVD (↓)	903.51 ± 20.44	–	481.55 ± 20.83	$\mathbf{372.50} \pm 15.07$
	SSIM (↑)	0.50 ± 0.01	–	0.72 ± 0.01	$\mathbf{0.76} \pm 0.01$
	PSNR (↑)	16.10 ± 0.05	–	20.33 ± 0.12	$\mathbf{20.88} \pm 0.15$
ninja	LPIPS (↓)	0.48 ± 0.01	–	0.49 ± 0.01	$\mathbf{0.45} \pm 0.01$
	FVD (↓)	423.27 ± 14.26	–	521.32 ± 15.85	$\mathbf{313.57} \pm 9.73$
	SSIM (↑)	0.45 ± 0.01	–	0.33 ± 0.01	$\mathbf{0.54} \pm 0.01$
	PSNR (↑)	12.58 ± 0.17	–	$\mathbf{12.80} \pm 0.10$	11.95 ± 0.12
starpilot	LPIPS (↓)	0.37 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	$\mathbf{0.22} \pm 0.01$
	FVD (↓)	626.47 ± 14.95	735.24 ± 21.48	429.36 ± 15.46	$\mathbf{211.27} \pm 12.70$
	SSIM (↑)	0.69 ± 0.01	0.68 ± 0.00	0.72 ± 0.01	$\mathbf{0.76} \pm 0.01$
	PSNR (↑)	19.99 ± 0.08	19.48 ± 0.13	19.76 ± 0.23	$\mathbf{20.55} \pm 0.13$

Table 8: Rollout accuracy for CLEVR and ALE environments at 20-th timestamp, measured as mean and standard error.

Environment	Metric	DreamerV3	Dyn-O (ours)
CLEVR	LPIPS (↓)	0.34 ± 0.00	$\mathbf{0.31} \pm 0.00$
	FVD (↓)	1676.18 ± 40.92	$\mathbf{446.57} \pm 8.55$
	SSIM (↑)	0.86 ± 0.00	$\mathbf{0.88} \pm 0.00$
	PSNR (↑)	21.37 ± 0.09	$\mathbf{22.63} \pm 0.06$
Atari Skiing	LPIPS (↓)	0.12 ± 0.00	$\mathbf{0.09} \pm 0.01$
	FVD (↓)	217.84 ± 3.76	$\mathbf{187.76} \pm 8.04$
	SSIM (↑)	0.91 ± 0.00	$\mathbf{0.93} \pm 0.00$
	PSNR (↑)	25.46 ± 0.13	$\mathbf{26.39} \pm 0.16$
Atari Boxing	LPIPS (↓)	0.04 ± 0.00	$\mathbf{0.02} \pm 0.00$
	FVD (↓)	237.90 ± 12.15	$\mathbf{218.68} \pm 15.68$
	SSIM (↑)	0.93 ± 0.00	$\mathbf{0.95} \pm 0.00$
	PSNR (↑)	$\mathbf{30.00} \pm 0.14$	29.46 ± 0.09

B.3 Evaluating World Model Accuracy

The world model accuracy for other procgen environments are shown in Table. 7 and Fig. 8 - Fig. 10. In most environments, Dyn-O significantly outperforms baselines in term of prediction accuracy.

Meanwhile, we further compare Dyn-O against DreamerV3 on the CLEVR dataset [24] and two ALE environments [4] (Skiing and Boxing), and the results are shown in Table. 8.
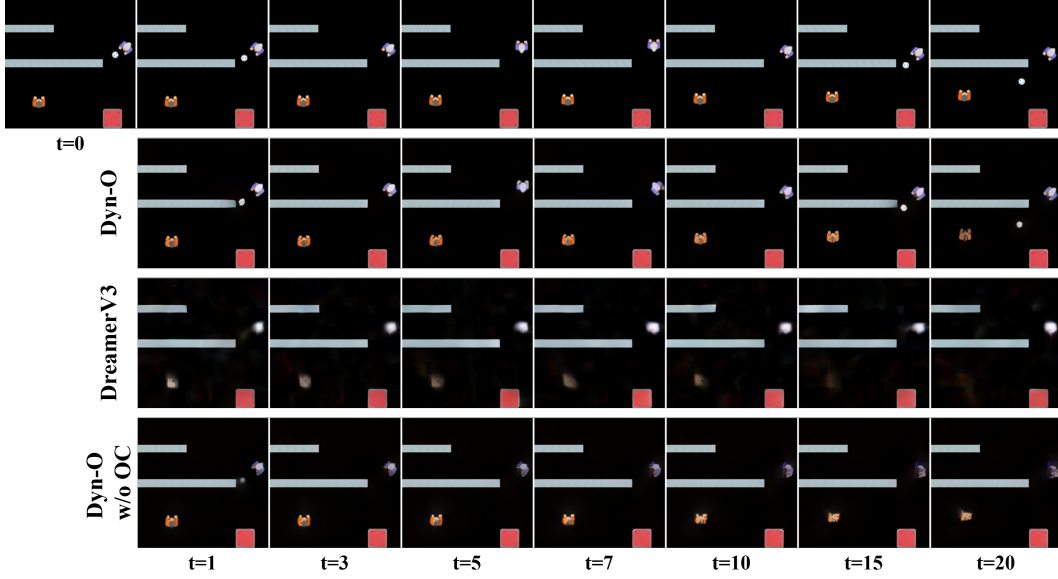
Figure 8: 20-step rollouts in **dodgeball**. **1st** row: ground-truth, **2nd** row: Dyn-O (ours), **3rd** row: DreamerV3, and **4th** row: Dyn-O w/o OC. Dyn-O significantly outperforms dreamer, with sharp player shape and accurate predictions of threw balls.
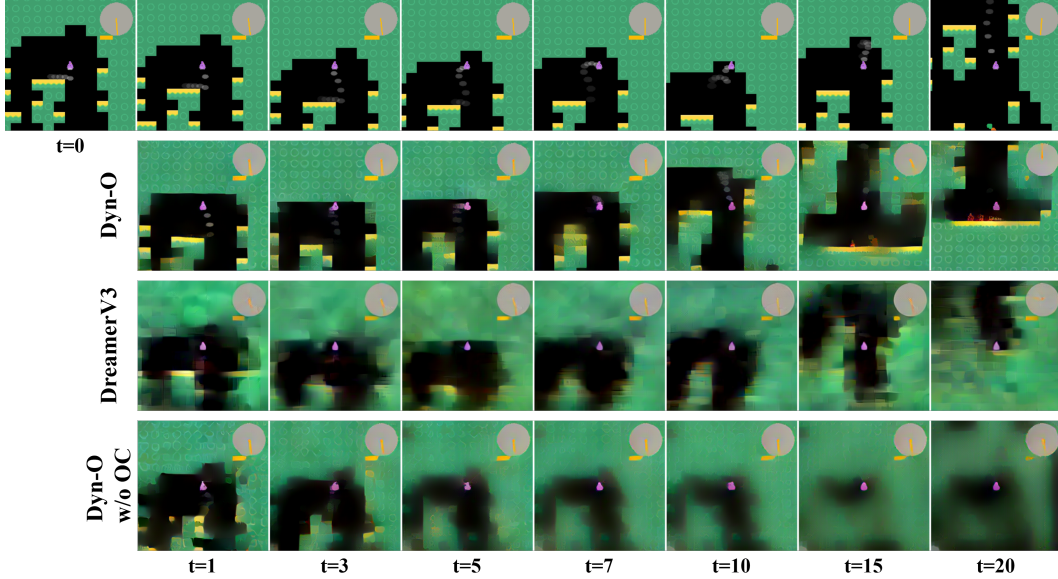

Figure 9: 20-step rollouts in **jumper**. **1st** row: ground-truth, **2nd** row: Dyn-O (ours), **3rd** row: DreamerV3, and **4th** row: Dyn-O w/o OC. Dyn-O significantly outperforms dreamer, with sharp wall and player trail until 10th timestamp.

B.4 Evaluating Static-Dynamic Disentanglement

The probing accuracy for other procgen environments are shown in Table. 9 - 12. The same privilege information may belong to different properties in different environments, which we label out in the tables. In some environments, the same privilege information can be static properties of some objects and can dynamic for other objects. In such case, we mark such privilege information as "mixed".
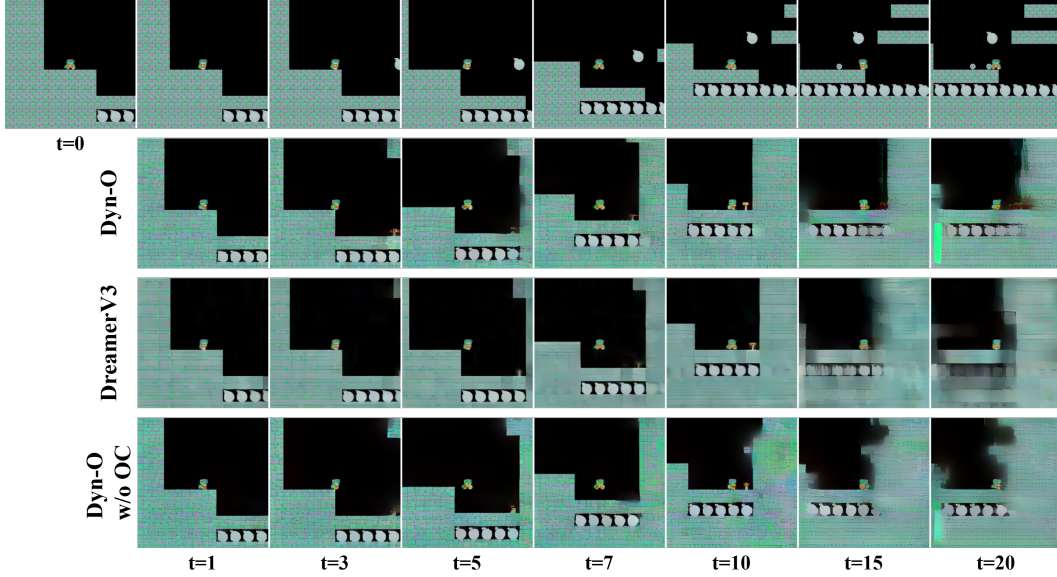
Figure 10: 20-step rollouts in **ninja**. **1st** row: ground-truth, **2nd** row: Dyn-O (ours), **3rd** row: DreamerV3, and **4th** row: Dyn-O w/o OC. Dyn-O significantly outperforms dreamer, with sharper wall shape at 15-th timestamp.

Table 9: Probing accuracy (↑), in percentage (%), on **bigfish** privilege properties.

	mean R values static	mean G values static	mean B values static	x position dynamic	y position static	area static
slots	74.7 ± 0.0	77.9 ± 0.0	83.8 ± 0.0	97.7 ± 0.0	98.3 ± 0.0	100.0 ± 0.0
dynamic features	49.3 ± 3.0	48.4 ± 2.4	47.4 ± 3.7	94.0 ± 1.9	45.2 ± 5.1	95.9 ± 1.2
static features	65.8 ± 4.2	67.6 ± 4.8	77.5 ± 1.2	29.3 ± 0.4	88.8 ± 0.7	100.0 ± 0.0
random features	37.6 ± 0.0	31.8 ± 0.0	37.3 ± 0.0	19.2 ± 0.0	20.8 ± 0.0	88.6 ± 0.0

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .

- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.

Table 10: Probing accuracy (↑), in percentage (%), on **dodgeball** privilege properties.

	mean R values static	mean G values static	mean B values static	x position mixed	y position mixed	area static
slots	90.2 ± 0.0	91.7 ± 0.0	91.2 ± 0.0	98.7 ± 0.0	98.7 ± 0.0	99.8 ± 0.0
dynamic features	66.0 ± 1.6	62.4 ± 1.2	61.1 ± 1.8	55.4 ± 1.8	57.7 ± 2.5	95.3 ± 1.7
static features	73.1 ± 1.2	74.2 ± 1.4	75.5 ± 1.7	74.9 ± 2.4	70.4 ± 2.9	99.3 ± 0.0
random features	53.3 ± 0.0	35.5 ± 0.0	42.9 ± 0.0	20.1 ± 0.0	19.7 ± 0.0	90.1 ± 0.0

Table 11: Probing accuracy (↑), in percentage (%), on **ninja** privilege properties.

	mean R values static	mean G values static	mean B values static	x position dynamic	y position dynamic	area dynamic
slots	87.1 ± 0.0	80.7 ± 0.0	86.1 ± 0.0	95.0 ± 0.0	96.6 ± 0.0	97.3 ± 0.0
dynamic features	60.7 ± 9.7	51.3 ± 8.2	60.6 ± 9.0	86.3 ± 3.0	88.1 ± 0.3	87.6 ± 2.2
static features	77.6 ± 2.9	62.2 ± 0.4	79.6 ± 0.4	35.9 ± 0.9	37.8 ± 1.1	82.2 ± 1.3
random features	32.9 ± 0.0	25.4 ± 0.0	31.3 ± 0.0	25.0 ± 0.0	19.5 ± 0.0	80.5 ± 0.0

Table 12: Probing accuracy (↑), in percentage (%), on **starpilot** privilege properties.

	mean R values static	mean G values static	mean B values static	x position dynamic	y position static	area static
slots	71.5 ± 0.0	79.1 ± 0.0	71.0 ± 0.0	97.2 ± 0.0	97.5 ± 0.0	99.5 ± 0.0
dynamic features	55.6 ± 7.2	66.5 ± 6.4	55.3 ± 6.2	94.8 ± 0.9	77.0 ± 14.2	97.9 ± 1.7
static features	55.9 ± 1.1	70.7 ± 1.1	50.5 ± 1.8	26.8 ± 0.7	76.6 ± 3.2	99.2 ± 0.0
random features	35.9 ± 0.0	50.6 ± 0.0	36.3 ± 0.0	18.3 ± 0.0	22.7 ± 0.0	92.0 ± 0.0

1. **Claims**

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

   Answer: [Yes]

   Justification: Our main claims can be supported by our experiments.

   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 discuss this in the appendix.

   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**

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

   Answer: [NA]

   Justification: Our 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 will provide code and instructions to make sure the results are reproducible.

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: [Yes]

   Justification: We will make the code available.

   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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide these details in Experiment section.

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 this in the appendix.

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.
- 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 provide this in the appendix.

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: The authors have reviewed 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: There is no societal impact of the work performed.

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: We don't foresee such risks.

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 credited existing assets.

    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.
    - 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: The paper does not release new assets

    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: The paper does not involve crowdsourcing 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: The paper does not involve crowdsourcing nor research with human subjects

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.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.