COMPAS: Compose Actions and Slots in Object-Centric World Models

Daniil Kirilenko[1] , **Vitaliy Vorobyov**[2] , **Alexey K. Kovalev**[3] and **Aleksandr I. Panov**[1,3]
[1]FRC CSC RAS
[2]MIPT
[3]AIRI
{kirilenko.de, vorobev.vitaly.v}@phystech.edu, {kovalev, panov}@airi.net,

Abstract

In this paper, we propose a reinforcement learning world model that leverages the strengths of the state-of-the-art object-centric models. Our approach combines symbol-like object-centric representations, known as slots, with action representations to accurately predict the next state and reconstruct the current state of the environment. A key aspect of our method is the composition of actions and objects using an autoregressive transformer, which enables the model to efficiently capture the complex interactions between objects and actions in a given context. We present a comprehensive evaluation of our approach in various environments, demonstrating that our proposed method outperforms competing models. The source code of our model and training/testing scripts are publicly available at https://anonymous.4open.science/r/compas-1E03.

1 Introduction

The ability to generalize compositionally is the key to generalizing to new problems and understanding new concepts with limited experience [Lin *et al.*, 2023]. The difficulty in compositional generalization is caused by the so-called binding problem [Greff *et al.*, 2020] – the inability of modern artificial neural networks to dynamically and flexibly bind information distributed over the network, which arises in the process of learning on unstructured input data.

A possible solution to this problem could be the use of symbol-like representations. Such representations can be slot representations [Locatello *et al.*, 2020], where the input data is not encoded by a single latent representation, but by a set of such representations (slots). Slots compete with each other to describe a portion of the input data. Such representations have been successfully used for object-centric tasks such as set property prediction [Locatello *et al.*, 2020] and object detection [Locatello *et al.*, 2020] in an image, learning visual dynamics from video [Wu *et al.*, 2022], image generation [Singh *et al.*, 2022], and unsupervised object-centric representation learning for real-world data [Seitzer *et al.*, 2022].

World models serve as core components in a wide variety of machine learning tasks, ranging from robotics and autonomous vehicles to game AI and video synthesis [Wu *et al.*, 2023a; Burgard *et al.*, 2016; Ha and Schmidhuber, 2018]. These models aim to represent the environment's complexities and dynamics to enable agents to understand, predict, and interact with their surroundings effectively. However, creating world models that can handle high-dimensional, continuous, and time-varying data remains a challenging task. Traditional world models often represent the environment as a whole without distinguishing between individual objects. These models struggle to capture the intricate relationships between different objects and their evolving dynamics over time. They often fail to generalize well across different tasks and struggle with scalability when faced with complex scenes with multiple interacting objects.

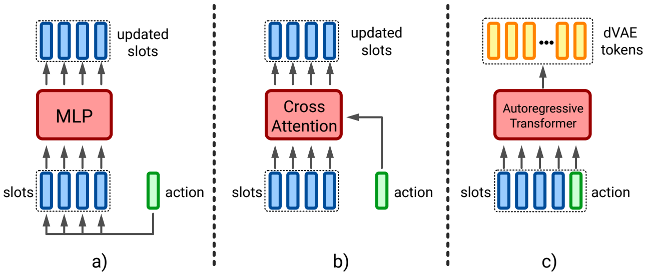


Figure 1: Comparative overview of three approaches to combine actions and slot representations: a) the conventional method, where each slot is combined with the action by addition or concatenation, followed by processing through a multilayer perceptron (MLP) for slot updating; b) a more complex approach, where actions are bound to slots through cross-attention (either hard or soft), and the updated slots are subsequently used for task-specific needs; c) our method, where slot and action representations are jointly fed to an autoregressive transformer, which then predicts the tokens of the corresponding image. Our method facilitates efficient and dynamic scene understanding by explicitly integrating action information into the object-centric world model.

Object-centric world models offer several advantages over traditional methods. First, they allow for more interpretable representations, since the state of the world is described in terms of identifiable objects and their properties. Second, they can handle complex scenes with multiple objects more efficiently, since changes in one object do not necessarily affect the representation of others. Third, object-centric models can be more data efficient because they can potentially reuse learned knowledge about one an object across different scenes or tasks.

In this paper, we propose a novel approach **COMPAS** (**COMP**ose **A**ctions and **S**lots) to encode these dynamics, focusing on object-centric world models that integrate actions and slots. Our model takes advantage of the power of discrete variational autoencoders (dVAEs) [Van Den Oord *et al.*, 2017], slot attention mechanisms [Locatello *et al.*, 2020], and autoregressive transformers [Vaswani *et al.*, 2017] to produce robust and versatile representations of complex scenes. Our approach, as illustrated in Figure 1, differs from other existing methods in its joint treatment of actions and slot representations by the autoregressive transformer.

Conventional models, shown in Figure 1a), typically combine the action with each slot by either addition or concatenation, which is then passed through a multilayer perceptron (MLP) to update the slots. This approach, while relatively straightforward, may not fully capture the complex relationships between actions and objects in a scene. A more advanced approach, depicted in Figure 1b), binds the action to the slots through a cross-attention mechanism, either hard or soft. This enables the model to pay differential attention to each slot depending on the action.However, this method still may not fully encapsulate the dynamics of a complex scene. In contrast, our method, illustrated in Figure 1c), passes the slot representations along with the action representation to an autoregressive transformer. This transformer then generates a prediction of the tokens of the corresponding image. By integrating action information directly into the model and using a transformer for prediction, our method is able to capture complex scene dynamics and generate accurate predictions. This approach fundamentally differs from other methods in its explicit integration of action information and its use of an autoregressive transformer for prediction.

2 Related works

2.1 World models

World models is the one of the most important research areas for sample-efficient reinforcement learning (RL). The main purpose of world models is to represent the environment in the latent state and predict the next latent states from the action. The dynamic prediction of next latent states is the one central problem in world models because it directly affects the quality of reinforcement learning algorithms. One approach is to use graph structures as in [Silver *et al.*, 2016; Silver *et al.*, 2017; Schrittwieser *et al.*, 2019; Hubert *et al.*, 2021]. These models represent states as nodes and transitions as edges in the graph. The disadvantage of graph world models is that they cannot work efficiently with continuous action spaces.

Another approach is to predict dynamics through recurrent neural networks (RNN). This family of models [Ha and Schmidhuber, 2018; Hafner *et al.*, 2018; Hafner *et al.*, 2020; Wu *et al.*, 2023a] is able to solve complex environments, from Minecraft to robotics tasks. The most recent advances are related to the use of transformers [Micheli *et al.*, 2023]. It allows to train world models faster because of input parallelization. But the disadvantage of these models is higher memory consumption. One of the other challenges in world models is state representation. A possible solution to this problem is to represent state by objects within a scene. This opens an opportunity for algorithms to reason about object interaction. Thus, there is a need for an algorithm that can extract the objects states from the scene.

2.2 Unsupervised object-centric representation learning

Many modern object-centric representation models are based on the slot attention module [Locatello *et al.*, 2020]. It implements an iterative attention mechanism, based on soft k-means clustering, for autoregressive slots refinement. Recent improvements of this method are based on better optimization [Chang *et al.*, 2022], learnable slots initialization [Jia *et al.*, 2023] or slots structure augmentation [Singh *et al.*, 2023]. However, these methods use simple convolutional neural network (CNN) encoders and decoders, which result in worse image reconstructions. Authors of the SLATE [Singh *et al.*, 2022] proposed to use discrete latent space, which is extracted from the image by dVAE [Van Den Oord *et al.*, 2017]. The computed slots are then passed through a transformer that predicts the latent token. This token is used in the dVAE decoder for image reconstruction. This results in better quality reconstructions than other models.

2.3 Object-centric world models

Previous object-centric world models have used other algorithms, than slot-attention for objects states extraction. SQAIR [Kosiorek *et al.*, 2018] uses a special detection model to detect objects in the scene and track their trajectories through RNNs. SCALOR [Jiang *et al.*, 2020] and SILOT [Crawford and Pineau, 2020] are able to scale the number of objects in SQAIR due to parallel inference mechanism. The STOVE [Kossen *et al.*, 2019] introduced a graph neural network (GNN)[Scarselli *et al.*, 2009] for dynamics prediction and per-object interactions.

One of the notable world models is the Generative Structured World Model (G-SWM) [Lin *et al.*, 2020]. It treats foreground objects and background separately by encoding them into two different latent vectors. For next state prediction, it uses two separate RNN, one for foreground latent vector, the other for background. One of the limitations of this model is that it doesn't consider actions taken in the environment when predicting future trajectories.

Another important approach is contrastively structured world models (C-SWM) [Kipf *et al.*, 2020]. It uses contrastive loss function on slots level, instead of basic image reconstruction loss. The model predicts the trajectory using GNN. Some notable improvements of the C-SWM are negative sampling [Biza *et al.*, 2021], which improves the loss function by either selecting negative samples from different time steps in the same episode or the same time step in different episodes. Another improvement is two types of action attention [Biza *et al.*, 2022]. Soft attention uses simple self-attention with a single head of transformers [Vaswani *et al.*, 2017]. Hard attention calculates the expectation of all possible assignments to objects and takes the index of the object with the highest probability and maps the action to that object.

The latest model – SlotFormer [Wu *et al.*, 2023b] uses ether SLATE [Singh *et al.*, 2022] or Slot-Attention [Locatello *et al.*, 2020] to extract slots from a sequence of images or video. The slots are then passed through a transformer to predict future frames of the video. This approach has achieved better results in dynamics prediction than previous models. However, the main limitation of this method is that, unlike our approach, it cannot work efficiently in action-dependent environments.

3 Method

Our method draws inspiration from the SLATE [Singh *et al.*, 2022] model, which has gained prominence due to its superior ability to construct meaningful representations of complex scenes. SLATE combines the best of object-centric representation learning, robust composition-based generalization, and effective representations. Its decoder is particularly robust, adept at handling new slot configurations and producing accurate compositions, a feature that enhances its adaptability in complex RL environments. Moreover, it excels at zero-shot image generation, a critical capability for diverse and unpredictable RL scenarios. SLATE also ensures global consistency in its image compositions, a factor that contributes to the stability and reliability of the RL world model. Finally, its attention maps effectively localize individual objects, enhancing its object-centric capabilities, which is crucial for RL tasks that require precise object interaction and manipulation.

The proposed approach involves a three-stage process of image encoding, slots representation extraction, and conditional generation with an autoregressive transformer. Our proposed method is visually summarized in Figure 2.

3.1 Observation encoding

Our first step is to encode the input image (observation $\mathbf{x_t}$) into a discrete feature map $\mathbf{z_t}$, a process carried out by a discrete variational autoencoder (dVAE) [Van Den Oord *et al.*, 2017]. The dVAE converts high-dimensional input data $\mathbf{x_t}$ into a lower-dimensional representation $\mathbf{e}$ extracting essential features from the frame:

$$\mathbf{e} = \text{encoder}(\mathbf{x_t}). \tag{1}$$

These features are then discretized by transforming the continuous latent space into a discrete one via approximate sampling using Gumbel-Softmax [Jang *et al.*, 2016]:

$$\mathbf{z_t} \sim \text{GumbelSoftmax}(\mathbf{e}, \tau). \tag{2}$$

We slowly decrease the temperature parameter τ from 1 to 0.1 during training. This process enables more straightforward manipulation of the image data in the subsequent stages of our model. The dVAE also includes the decoder part. We minimize the Mean Squared Error (MSE) between input and reconstructed images during training:

$$L_{rec} = \text{MSE}(\mathbf{x_t}, \hat{\mathbf{x}}_\mathbf{t}), \quad \hat{\mathbf{x}}_\mathbf{t} = \text{decoder}(\mathbf{z_t}). \tag{3}$$

3.2 Slots representation extraction

Next, we use a slot attention [Locatello *et al.*, 2020] mechanism to derive object-centric representations from the discretized feature map $\mathbf{z_t}$. TThis mechanism works by iteratively assigning each token to slots that are randomly sampled from a normal distribution with trainable parameters. This is done using a special cross-attention mechanism and a recurrent neural network, GRU [Chung *et al.*, 2014]. This process generates an abstract scene representation that focuses on a set of individual representations rather than the entire scene.

$$\text{slots} \sim \mathcal{N}(\mu, \Sigma), \quad M = \frac{1}{\sqrt{D}} k(\mathbf{z_t}) q(\text{slots})^T, \tag{4}$$

$$\text{attn}_{i,j} = \frac{e^{M_{i,j}}}{\sum_{j=1}^{K} e^{M_{i,j}}}, \quad W_{i,j} = \frac{\text{attn}_{i,j}}{\sum_{i=1}^{N} \text{attn}_{i,j}}, \tag{5}$$

$$\text{updates} = W^T v(\mathbf{z_t}) \in R^{K \times D}, \tag{6}$$

$$\text{slots} = \text{GRU}(\text{input=updates, hidden=slots}), \tag{7}$$

where $\mu,\ \Sigma$ are trainable parameters, and Σ is a diagonal covariance matrix; K is a number of slots; N – number of tokens $\mathbf{z_t}$; D – slot dimensionality; $k,\ q,\ v$ are trainable matrix projections.

This process allows us to handle each object in the scene independently, facilitating object-level transformations in the subsequent transformer stage.

3.3 Conditional generation with an autoregressive transformer

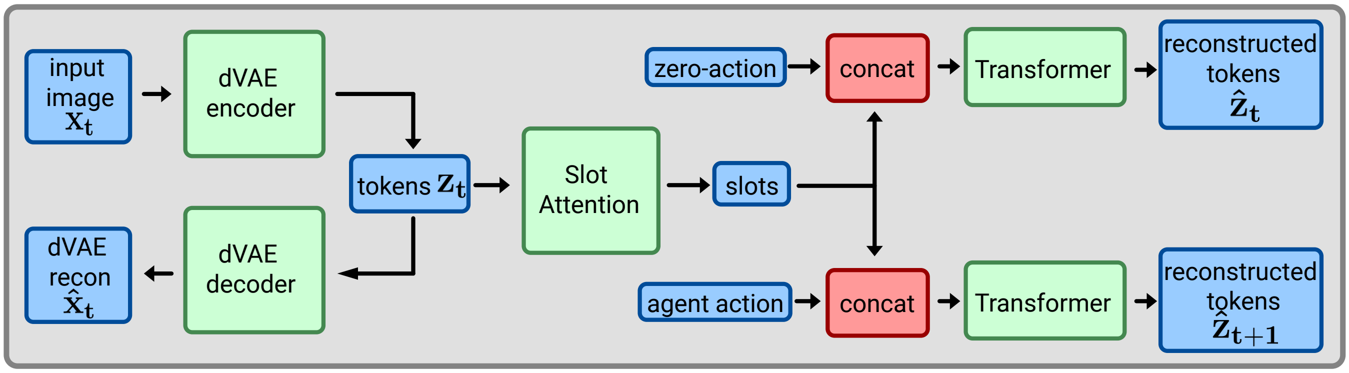
Finally, the slot representations and an action representation are combined and fed into an autoregressive transformer [Vaswani *et al.*, 2017]. The autoregressive transformer then conditionally generates a prediction of the source tokenized feature map. This generation occurs in two regimes, based on the nature of the action representation. In both cases, we use the cross-entropy of the prediction of each successive token as the training signal.

1. Non-action regime: If the action representation is an auxiliary embedding that represents a non-action (zero-action) vector, the task of the transformer is to predict the tokens of the frame at the current timestamp. This is essentially a "do nothing" action, and the transformer should ideally reproduce the original input.

$$\hat{z}_{i,t} = \text{Transformer}(\hat{z}_{<i,t};\ \text{slots};\ \text{zero-action}), \tag{8}$$

$$L_t = \text{CrossEntropy}(\mathbf{z_t}, \hat{\mathbf{z}}_\mathbf{t}) \tag{9}$$

2. Agent-action regime: If the action is a representation of an action made by an agent, the transformer aims to predict the tokens of the frame at the next timestamp. This requires the model to understand the dynamics of the scene and to predict how the agent's action will change the scene.



input image $\mathbf{x_t}$
dVAE encoder
tokens $\mathbf{z_t}$
dVAE recon $\hat{\mathbf{x}}_\mathbf{t}$
dVAE decoder
Slot Attention
slots
zero-action
concat
Transformer
reconstructed tokens $\hat{\mathbf{z}}_\mathbf{t}$
agent action
concat
Transformer
reconstructed tokens $\hat{\mathbf{z}}_\mathbf{t+1}$

Figure 2: Overview of the proposed method. This figure illustrates the three-stage process: image encoding into a discretized feature map using a dVAE, extraction of object-centric slot representations through a slot attention mechanism, and conditional generation of source tokenized feature map predictions using an autoregressive transformer, conditioned on the slot representations and an action representation. The two regimes of operation – zero-action and agent-action – are also depicted.

$$\hat{z}_{i,t+1} = \text{Transformer}(\hat{z}_{<i,t+1};\ \text{slots};\ \text{action}), \quad (10)$$

$$L_{t+1} = \text{CrossEntropy}(\mathbf{z_{t+1}}, \hat{\mathbf{z}}_\mathbf{t+1}) \quad (11)$$

3.4 Loss function

When computing the loss function (Formula 12), we weight L_t and L_{t+1} in favor of L_{t+1} since it is more important for the world model to make accurate predictions for the following timestamps. At the same time, L_t ensures the better disentanglement of objects to slots.

$$Loss = L_{rec} + 0.2L_t + 0.8L_{t+1} \quad (12)$$

The main blocks of the model are trained separately. Namely, dVAE receives training signal only from the L_{rec} component, while transformer and slot attention get gradient updates regarding L_t and L_{t+1}.

4 Experiments

In our experiments, we primarily compare our model to Contrastive Structured World Models (C-SWM) [Kipf *et al.*, 2020] and its attention-based modifications [Biza *et al.*, 2022], which we consider to be our main competitors in the field of object-centric world modeling. For a comprehensive overview of the hyperparameters used in our experiments, we refer the reader to the Appendix A.

Our research focuses on the precision training of our model to predict a single step into the future, an aspect that significantly enhances the efficiency of our methodology. Upon evaluation, we highlight the remarkable ability of our system to extrapolate accurate predictions beyond the range initially encountered during the training process.

4.1 Environments

To evaluate the effectiveness of our proposed model, we perform experiments using trajectories randomly sampled from two environments: Atari Pong and Causal World [Ahmed *et al.*, 2020].

Atari Pong is a classic video game environment that provides a relatively simple setting for our experiments. It is a two-dimensional environment with a discrete action space. In this game, the agent must learn to control a paddle to hit a ball and prevent it from reaching the edge of the screen. Despite its simplicity, Pong presents a reasonable challenge in terms of predicting the dynamics of the ball and paddle based on the current state and actions.

At the other end of the complexity spectrum, we use the Causal World environment [Ahmed *et al.*, 2020], which presents a much more complicated and challenging scenario. Causal World is a physically simulated environment with three robotic arms interacting with objects. It provides a continuous action space and requires the model to understand complex physical interactions and the impact of nuanced actions on the state of the world.

By conducting experiments in both environments, we aim to demonstrate the versatility and scalability of our model – from simpler, discrete action environments like Pong to more complex, continuous action environments such as Causal World.

4.2 Metrics

Following the evaluation setup of [Biza *et al.*, 2022] we predict future slots for 1, 5 and 10 steps forward and compare them with the real slot representations at the same predicted time step. The HITS@1 calculates the proportion of cases where the predicted slots were nearer to the real slots than to any of the other predictions:

$$\frac{|U|}{|D|}, \quad (13)$$

where D is the evaluation dataset, $U \subseteq D$ – set of nearest true samples to the predicted ones.

The Mean Reciprocal Rank (MRR) metric computed as:

$$\frac{1}{|D|}\sum_{n=1}^{|D|}\frac{1}{\text{rank}_n}, \quad (14)$$

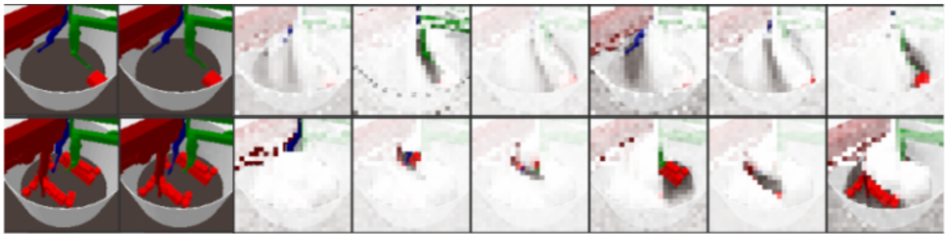


Figure 3: Each row represents a different task, with the first row illustrating the Push task of the Causal World environment and the second showcasing the Stack task. The first column in each row represents the current observation, while the second column shows its reconstruction by our model. Subsequent columns display the attention maps between slots and visual features, highlighting the specific object-slot interactions being processed by our model.

	1 Step		5 Step		10 Step	
Models	MRR	HITS@1	MRR	HITS@1	MRR	HITS@1
C-SWM	0.33 ± 0.05	0.18 ± 0.05	0.27 ± 0.06	0.15 ± 0.04	0.14 ± 0.05	0.06 ± 0.02
C-SWM Hard Attention	0.11 ± 0.03	0.04 ± 0.01	0.07 ± 0.03	0.03 ± 0.02	0.04 ± 0.01	0.01 ± 0.00
C-SWM Soft Attention	0.35 ± 0.08	0.20 ± 0.06	0.26 ± 0.06	0.15 ± 0.03	0.11 ± 0.02	0.06 ± 0.01
COMPAS (our)	**0.85 ± 0.10**	**0.77 ± 0.13**	**0.29 ± 0.09**	**0.16 ± 0.10**	**0.24 ± 0.04**	**0.11 ± 0.02**

Table 1: Comparative results on Causal World (Push) environment. Mean ± std for 3 seeds.

	1 Step		5 Step		10 Step	
Models	MRR	HITS@1	MRR	HITS@1	MRR	HITS@1
C-SWM	0.08 ± 0.02	0.03 ± 0.01	0.08 ± 0.02	0.02 ± 0.01	0.07 ± 0.01	0.03 ± 0.01
C-SWM Hard Attention	0.02 ± 0.01	0.01 ± 0.00	0.03 ± 0.00	0.01 ± 0.00	0.04 ± 0.00	0.01 ± 0.00
C-SWM Soft Attention	0.12 ± 0.02	0.04 ± 0.01	0.12 ± 0.04	0.04 ± 0.02	0.08 ± 0.03	0.03 ± 0.01
COMPAS (our)	**0.99 ± 0.01**	**0.98 ± 0.02**	**0.49 ± 0.11**	**0.37 ± 0.14**	**0.34 ± 0.07**	**0.17 ± 0.05**

Table 2: Comparative results on Causal World (Stack) environment. Mean ± std for 3 seeds.

	1 Step		5 Step		10 Step	
Models	MRR	HITS@1	MRR	HITS@1	MRR	HITS@1
C-SWM	0.17 ± 0.01	0.18 ± 0.02	0.01 ± 0.01	0.01 ± 0.02	0.03 ± 0.03	0.03 ± 0.03
C-SWM Hard Attention	0.18 ± 0.02	0.16 ± 0.01	0.04 ± 0.06	0.03 ± 0.04	0.02 ± 0.02	0.01 ± 0.01
C-SWM Soft Attention	0.17 ± 0.00	0.16 ± 0.02	0.04 ± 0.04	0.03 ± 0.03	0.05 ± 0.05	**0.05 ± 0.05**
COMPAS (our)	**0.42 ± 0.14**	**0.32 ± 0.08**	**0.23 ± 0.03**	**0.14 ± 0.01**	**0.19 ± 0.05**	0.03 ± 0.07

Table 3: Comparative results on Atari Pong environment with random policy. Mean ± std for 3 seeds.

rank_n refers to the position of the actual instances within a list of distances that includes both real and predicted examples, sorted by the distance between the predicted and real latent states, from lowest to highest values.

Since slot attention has a tendency to permute slots at different time steps, we have used the Hungarian algorithm to match predicted and real slots for the distance matrix computation.

The higher the values of HITS@1 and MRR, the better the model is at predicting future states.

4.3 Training and evaluation setup

We've collected datasets of 1200 episodes with 100 time steps in each trajectory for train and 1000 episodes with the same trajectory length for evaluation.

Each C-SWM model was trained for 100 epochs with the physics simulation configuration [Kipf *et al.*, 2020] for Casual Worlds and the corresponding Atari Pong architecture from [Biza *et al.*, 2022]

4.4 Results

In the context of the Causal World environment, we conducted a thorough evaluation of COMPAS, C-SWM, and C-

SWM attention-based modifications on two specific Causal World tasks: Push and Stack. Figure 3 shows examples of how the model's slots attend to different elements depending on the specific task. In the Push task, the slots focus on different robot arms and the cube, reflecting the key components involved in this task. Conversely, in the Stack task, the slots primarily aim to distinguish between two stacks of blocks. This illustrates the model's adaptability in dynamically adjusting its attention according to the unique demands of each task, and highlights its potential for addressing a wide range of object-centric world modeling challenges.

As highlighted in Table 1, COMPAS demonstrated superior predictive accuracy relative to C-SWM in the Push task. This finding is a testament to the robustness of our approach in handling complex, physically simulated environments.

Furthermore, COMPAS' performance increased significantly in the more challenging Stack task (Table 2). In contrast, C-SWM showed a significant performance degradation under these conditions. This stark contrast further highlights the comparative advantages of COMPAS in terms of resilience and adaptability in diverse and complex tasks.

In the context of simpler environments such as Atari Pong, models based on C-SWM have been observed to achieve near-optimal results when employing specific, advanced policies and pre-trained agents during data collection [Biza *et al.*, 2021]. However, the performance of these models deteriorates significantly when a random policy is used, suggesting a degree of instability in their performance.

In contrast, the proposed COMPAS model demonstrates robust performance regardless of the policy used for data collection. As evidenced in Table 3, COMPAS consistently outperforms C-SWM-based models even when a random policy is used for data collection.

5 Conclusion and future works

In this paper, we present a novel method for integrating actions and slots into object-centric world models based on the SLATE model. Our approach uses a discrete variational autoencoder, slot attention, and an autoregressive transformer to produce robust representations of complex scenes. Despite being specifically trained for single-step prediction, our model demonstrates superior consistency over multiple prediction steps compared to analogous models in the field. This confirms the model's robustness, adaptability, and ability to handle different prediction horizons.

Our experimental results, using trajectories from Atari Pong and Causal World environments, demonstrated the superior performance and robustness of our approach. In particular, our model outperformed competing methods in the complex, continuous action environment of Causal World. This demonstrates the strength of our method in handling complex dynamics and predicting the effects of actions in a physically simulated world.

However, our model was less successful in the simpler, discrete action environment of Atari Pong when dealing with short trajectories. This highlights a potential area for improvement in handling environments with simpler dynamics or when operating over shorter time horizons.

Looking to the future, a promising direction is to explore the neuro-symbolic perspectives of object-centric approaches. Neuro-symbolic reasoning combines the strengths of neural networks and symbolic reasoning, allowing models to learn from data while incorporating structured, symbolic knowledge. This could potentially improve the interpretability and robustness of object-centric world models, and allow them to better generalize from learned knowledge. Further improvements could also be made in the integration of actions into the model, potentially through the use of more sophisticated action representations or more advanced mechanisms for binding actions and slots.

In conclusion, our work represents an important step forward in object-centric world models. By integrating action information directly into the model and using an autoregressive transformer for prediction, we have demonstrated a novel way to encode and predict the dynamics of complex environments. We expect that our work will inspire further research and advances in this exciting area of machine learning.

Acknowledgments This work was partially supported by the Russian Science Foundation (Project No. 20-71-10116).

References

[Ahmed *et al.*, 2020] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Manuel Wüthrich, and Stefan Bauer. Causalworld: A robotic manipulation benchmark for causal structure and transfer learning. *arXiv:2010.04296*, 2020.

[Biza *et al.*, 2021] Ondrej Biza, Elise van der Pol, and Thomas Kipf. The impact of negative sampling on contrastive structured world models. *ICML Workshop: Self-Supervised Learning for Reasoning and Perception*, 2021.

[Biza *et al.*, 2022] Ondrej Biza, Robert Platt, Jan-Willem van de Meent, Lawson L. S. Wong, and Thomas Kipf. Binding actions to objects in world models. *ICLR 2022 workshop on Objects, Structure and Causality*, 2022.

[Burgard *et al.*, 2016] Wolfram Burgard, Martial Hebert, and Maren Bennewitz. World modeling. *Springer handbook of robotics*, pages 1135–1152, 2016.

[Chang *et al.*, 2022] Michael Chang, Tom Griffiths, and Sergey Levine. Object representations as fixed points: Training iterative refinement algorithms with implicit differentiation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555*, 2014.

[Crawford and Pineau, 2020] Eric Crawford and Joelle Pineau. Exploiting spatial invariance for scalable unsupervised object tracking. In *AAAI*, pages 3684–3692. AAAI Press, 2020.

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

[Ha and Schmidhuber, 2018] David Ha and Jürgen Schmidhuber. World models. *arXiv:1803.10122*, 2018.

[Hafner *et al.*, 2018] Danijar Hafner, Timothy P. Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *CoRR*, abs/1811.04551, 2018.

[Hafner *et al.*, 2020] Danijar Hafner, Timothy P. Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *CoRR*, abs/2010.02193, 2020.

[Hubert *et al.*, 2021] Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Mohammadamin Barekatain, Simon Schmitt, and David Silver. Learning and planning in complex action spaces. *CoRR*, abs/2104.06303, 2021.

[Jang *et al.*, 2016] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv:1611.01144*, 2016.

[Jia *et al.*, 2023] Baoxiong Jia, Yu Liu, and Siyuan Huang. Improving object-centric learning with query optimization. *ICLR*, 2023.

[Jiang *et al.*, 2020] Jindong Jiang, Sepehr Janghorbani, Gerard de Melo, and Sungjin Ahn. Scalor: Generative world models with scalable object representations. In *ICLR*, 2020.

[Kipf *et al.*, 2020] Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *ICLR*, 2020.

[Kosiorek *et al.*, 2018] Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems*, pages 8606–8616, 2018.

[Kossen *et al.*, 2019] Jannik Kossen, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting. Structured object-aware physics prediction for video modeling and planning. *arXiv:1910.02425*, 2019.

[Lin *et al.*, 2020] Zhixuan Lin, Yi-Fu Wu, Skand Peri, Bofeng Fu, Jindong Jiang, and Sungjin Ahn. Improving generative imagination in object-centric world models. In *ICML*, pages 6140–6149. PMLR, 2020.

[Lin *et al.*, 2023] Baihan Lin, Djallel Bouneffouf, and Irina Rish. A survey on compositional generalization in applications. *arXiv:2302.01067*, 2023.

[Locatello *et al.*, 2020] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention, 2020.

[Micheli *et al.*, 2023] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *ICLR*, 2023.

[Scarselli *et al.*, 2009] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[Schrittwieser *et al.*, 2019] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model, 2019. cite arxiv:1911.08265.

[Seitzer *et al.*, 2022] Maximilian Seitzer, Max Horn, Andrii Zadaianchuk, Dominik Zietlow, Tianjun Xiao, Carl-Johann Simon-Gabriel, Tong He, Zheng Zhang, Bernhard Schölkopf, Thomas Brox, et al. Bridging the gap to real-world object-centric learning. *arXiv:2209.14860*, 2022.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, jan 2016.

[Silver *et al.*, 2017] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017.

[Singh *et al.*, 2022] Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate dall-e learns to compose. In *ICLR*, 2022.

[Singh *et al.*, 2023] Gautam Singh, Yeongbin Kim, and Sungjin Ahn. Neural systematic binder. In *ICLR*, 2023.

[Van Den Oord *et al.*, 2017] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *NeurIPS*, 2017.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[Wu *et al.*, 2022] Ziyi Wu, Nikita Dvornik, Klaus Greff, Thomas Kipf, and Animesh Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. *arXiv:2210.05861*, 2022.

[Wu *et al.*, 2023a] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on Robot Learning*, pages 2226–2240. PMLR, 2023.

[Wu *et al.*, 2023b] Ziyi Wu, Nikita Dvornik, Klaus Greff, Thomas Kipf, and Animesh Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models, 2023.

A Architecture Details

Tables 5, 6, 7 describe architecture details and hyperparameters for our experiments.

Layer	Channels	Activation	Params
Conv2D 4×4	64	ReLU	stride: 4
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	vocab size	ReLU	stride: 1
Position Embedding	-	-	absolute

Table 5: Architecture of the CNN encoder.

Layer	Channels/Size	Activation	Params
Conv2D 3×3	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64 * 2 * 2	ReLU	stride: 1
PixelShuffle	upscale factor = 2	-	-
Conv2D 3×3	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64	ReLU	stride: 1
Conv2D 1×1	64 * 2 * 2	ReLU	stride: 1
PixelShuffle	upscale factor = 2	-	-
Conv2D 1×1	3	-	stride: 1

Table 6: Architecture of the CNN decoder.

Module	Parameter	Value
	Image Size (Causal world/Pong)	96/64
	Encoded Tokens	576/256
	Number of episodes collected	1200
	Number of steps per episode	100
	Number of training epochs	150
	Batch size	128
dVAE	Vocab size	512
dVAE	Temp. Cooldown	1.0 to 0.1
dVAE	Temp. Cooldown Steps	30000
dVAE	LR (no warmup)	0.0003
Transformer	Layers	6
Transformer	Heads	4
Transformer	Hidden Dim.	128
Slot Attention	Iterations	5
Slot Attention	Slot dim.	64

Table 7: Hyperparameters used for our experiments.