On Unsupervised Object-Centric Representation Learning: Advantages and Shortcomings

Yarden Goraly
Kaylene Stocking, Ed.
Claire Tomlin, Ed.

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2025-112
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2025/EECS-2025-112.html

May 16, 2025

Acknowledgement

I'd like to thank everyone who made this work possible, including my program advisor,
Professor Claire Tomlin. I'm very grateful to have been advised by such a knowledgeable
and kind professor who could guide me through making the most out of this program. I'm
also extremely grateful for the incredible work of Kaylene Stocking in mentoring me over the
past couple of years. She truly went above and beyond in mentoring me through both class
projects and personal projects. Every day of this program I was thankful to be under such
great guidance, and I appreciate all the interesting conversations we were able to have.

On Unsupervised Object-Centric Representation Learning: Advantages and Shortcomings

by Yarden Goraly

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Claire Tomlin
Research Advisor

May 16, 2025
(Date)

* * * * * * *

Professor Sosale Shankar Sastry
Second Reader

May 16, 2025
(Date)

On Unsupervised Object-Centric Representation Learning: Advantages and Shortcomings

by

Yarden Goraly

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Claire Tomlin, Chair
Professor Sosale Shankar Sastry

Spring 2025

On Unsupervised Object-Centric Representation Learning: Advantages and Shortcomings

Abstract

On Unsupervised Object-Centric Representation Learning: Advantages and Shortcomings

by

Yarden Goraly

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Claire Tomlin, Chair

Unsupervised object-centric representation learning is an active area of research with promising applications to robotics and computer vision. These models go beyond the ability to segment objects in a scene. The goal is for these models to develop a disentangled internal representation of objects in latent space. Some models can even encode specific interpretable properties of these objects, such as position, size and shape, in the latent space. In this work, we review the current literature and history of unsupervised object-centric learning and evaluate the impact of each model and how they compare to human perception. We then look at the current theory related to object-centric latent disentanglement and suggest avenues for future research. Finally, we look into a few novel experiments that improve the segmentation performance of these methods and solve sim-to-real problems. We found that it is possible to improve segmentation performance of unsupervised object-centric models using knowledge distillation while retaining latent encoding of object properties. We also uncover unique ways in which the type of dataset can affect reconstruction quality for real and synthetic inputs.

Contents

Contents **i**

List of Figures **ii**

1 Introduction and Motivation **1**
1.1 Object-Centric Perception in Animals 1
1.2 What is Unsupervised Object-Centric Learning? 2

2 Preliminaries **4**
2.1 Variational Autoencoders 4
2.2 β-VAE 5
2.3 Latent Disentanglement 6

3 Approaches **7**
3.1 Supervised Methods 7
3.2 Core Object-Centric Works 8
3.3 Variants on Core Works 12
3.4 Takeaways 14

4 Object-Centric Disentangled Representation Learning Theory **15**
4.1 Object-Identifiability Theory in 2D scenes 15
4.2 Theory of Slot Identifiability in Videos 18

5 Improving Object-Centric Methods **21**
5.1 Improving Object-Centric Representations and Segmentations using Knowledge Distillation 21
5.2 How Datasets Affect Real World Performance 26

Bibliography **32**

List of Figures

2.1 **Connecting posterior overlap with minimizing the KL divergence and reconstruction error.** Broadening the posterior distributions and/or bringing their means closer together will tend to reduce the KL divergence with the prior, which both increase the overlap between them. But, a datapoint $\tilde{x}$ sampled from the distribution $q_\psi(z_2|x_2)$ is more likely to be confused with a sample from $q_\psi(z_1|x_1)$ as the overlap between them increases. Hence, ensuring neighbouring points in data space are also represented close together in latent space will tend to reduce the log likelihood cost of this confusion 6

3.1 SAM is a foundation model for segmentation by using three interconnected components: a promptable segmentation task, a segmentation model (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a data engine for collecting SA-1B, a dataset of over 1 billion masks. 8

3.2 Schematic of MONet. **(a)** Overall compositional generative model architecture. The attention net recurrently generates the masks over a sequence of steps to condition the component VAE, labelling which pixels to focus on representing and reconstructing for that component. **(b)** Recursive decomposition process: the attention network at a particular step is conditioned on the image and the current scope, which is what currently remains to be explained of the scene (with the initial scope $s_0 = 1$). The attention mask outputted will be some portion of the scope, and the scope for the next step is then what still remains to be explained after accounting for this attention mask. **(c)**. The component VAE receives both the image and a mask as input, and is pressured only to model the masked region by applying the mask to weight the component likelihood in the loss. Thus the reconstruction component is unconstrained outside of the masked region, enabling it for example to fill in occluded regions. The VAE also models the masks themselves. 9

3.3 Architecture of the SIMONe inference network ε_ϕ. The transformers integrate information jointly across space and time to infer (the posterior parameters of) the object and frame latents. 13

4.1 When can unsupervised object-centric representations provably be learned? We assume that observed scenes $\mathbf{x}$ comprising K objects are rendered by an unknown generator f from multiple ground-truth latent slots $\mathbf{z}_1, ..., \mathbf{z}_K$ (here, $K = 3$). We assume that this generative model has two key properties, which we call compositionality and irreducibility. Under this model, we prove theorem 1: An invertible inference model with a compositional inverse yields latent slots $\hat{\mathbf{z}}_i$ which identify the ground-truth slots up to permutation and slot-wise invertible functions $\mathbf{h}_i$. To measure violations of compositionality in practice, we introduce a contrast function which is zero if and only if a function is compositional, while to measure invertibility, we rely on the reconstruction loss in an auto-encoder framework. . 19

5.1 Results showing comparison between (a) teacher-student model and (b) original SOCS model. Both models were trained the exact same hyperparameters other than those associated with mask loss, which was incorporated in (a) but not in (b). 24
5.2 Reconstruction loss from training the teacher-student model (loss vs. iteration) . 24
5.3 Distribution loss from training the teacher-student model (loss vs. iteration) . . 25
5.4 Mask loss from training the teacher-student model (loss vs. iteration) 25
5.5 Total combined loss from training the teacher-student model (loss vs. iteration) 26
5.6 An overview of the proposed model architecture. From left to right on the top: video sequences are inputted into the CNN encoder, which outputs patches that are inputted the transformer. The transformer outputs slot tokens which are used to parameterize the VAE distribution. The object slots are then sampled from this distribution. These slots are then pooled over time and inputted into a query decoder. Finally the query decoder outputs per-slot probabilities, which can then be thresholded to get the segmentations From left to right on the bottom: Input images are also fed into the SAM2 model, which produces teacher masks. Then a subset of pixels from these masks are queried and compared to the VAE segmentation to calculate a mask loss. 27
5.7 **Synthetic Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right) 29
5.8 **Real World Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right) 29
5.9 **Combined Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right) 30
5.10 **Reconstruction Loss:** Real world (green), Synthetic (black), Combined (pink) 30
5.11 **Distribution (KL) Loss:** Real world (green), Synthetic (black), Combined (pink) 31

Acknowledgments

I'd like to thank everyone who made this work possible, including my program advisor, Professor Claire Tomlin. I'm very grateful to have been advised by such a knowledgeable and kind professor who could guide me through making the most out of this program. I'm also extremely grateful for the incredible work of Kaylene Stocking in mentoring me over the past couple of years. She truly went above and beyond in mentoring me through both class projects and personal projects. Every day of this program I was thankful to be under such great guidance, and I appreciate all the interesting conversations we were able to have.

Chapter 1

Introduction and Motivation

For many years, one of the most enticing goals in robotics has been the general home robot, capable of doing everything around the house, from washing the dishes to doing the laundry [9]. While we are still far from that reality, robots in real-world environments capable of performing general tasks must have the ability to interact with diverse objects. Later in this chapter we will discuss what researchers in neuroscience and psychology currently know about object-centric representations in animals. While researchers in machine learning are often inspired by biology [22, 37, 2, 10, 35], it is important to distinguish between ideas that mirror biology or if they are merely inspired in principle. Nevertheless, authors of unsupervised object-centric papers often claim inspiration from biology [34, 7, 36, 33, 30]. In this chapter, we will explore the current literature on object-centric representations in animals and then in the machine learning literature. We will compare and contrast what we know about these two approaches in order to better understand the extent to which object-centric methods in machine learning are similar to biology.

1.1 Object-Centric Perception in Animals

We live in an object-centric world. Everything we perceive, name, and categorize has to do with objects. This gives us a good opportunity to test how nature solved the object-centric problem in animals. Much research in psychology and neuroscience attempts to do exactly this. A particularly interesting question is the extent to which object-centric perception is innate versus learned. To answer this, we can look into the work of Elizabeth Spelke, who is one of the leading researchers in cognitive psychology who works on object perception in animals and humans. Her book, *What Babies Know* [31] contains a great summary of the recent literature in object-centric psychology. One of her hypotheses is that animals have a partially innate ability to perceive the world in terms of objects. This would have significant implications if true, as this shows that much of how animals observe and interact with objects is not learned at all, but rather built into their cognition. We will next investigate some experiments that attempt to answer this question.

Scientists had discovered that chicks are a valuable animal for testing perception immediately after birth because of their immediate displays of object permanence. The standard

experiment using chicks involves a technique called "imprinting" [15, 16]. In standard practice, imprinting involves making the chicks see a singular object at birth, with the ability to walk around and interact with it. The researchers can then perform perception experiments based on the behavior of the chicks. A convenient fact is that chicks naturally gravitate to the first thing they see from birth [29], which allows for many interesting experiments that test a chick's ability to locate an object. A particularly interesting study in this realm was done by [6]. In this work, the authors performed a variant on a common method for testing perception in young animals. Instead of the imprinting method described above, the authors had the chicks live their first two days of life in complete darkness. Then, they allowed the chicks to see the object for the first time from afar. The object would move around behind two panels, but the chicks could not interact with it. Then the researchers would turn off the lights and put the object behind one of the panels, leaning the panels so that one of them leans on the object while the other is in an impossible position had the object been there. Then the researchers see which side the chicks gravitate towards. The results of the study show that chicks were able to predict the panel under which the object was hiding. These results are incredibly insightful because they show that chicks might have an understanding of the physical properties of objects even having never interacted with one. More research is required in this area, but this result shows that there is value in exploring perception systems that enforce object-centric representations through inductive biases with the hope of increased generalizability. Object-centric methods in machine learning attempt something similar to this by forcing an interpretable latent space that encodes the objects separately, but they still require much more data than animals. It's worth noting the animals also have other sensory inputs that contribute to perception such as sensorimotor, depth, etc, which are largely absent from current object-centric approaches. Thus we have to be careful when saying that object-centric methods are inspired by biology, because in reality these mechanisms are very different.

1.2 What is Unsupervised Object-Centric Learning?

The goal of unsupervised object-centric learning is to build a representation of objects within a latent space for downstream tasks, such as segmentation and inference. The unifying factor behind object-centric models is that they aim to create a latent space via an autoencoder framework that has inductive biases conducive to scenes with objects. In particular, these models aim to disentangle the latent space, leading to an interpretable internal representation of objects within the model. Having a rich representation of objects has numerous potential benefits, especially in the field of robotics. For instance, search and rescue missions require an agent to navigate an unknown environment while potentially interacting with objects to find a target. In this situation, supervised methods might suffer since novel situations may generalize poorly to out-of-distribution inputs. In addition, this rich understanding of objects could allow the robot to generalize better by being able to predict the properties that objects have. Lastly, unsupervised object-centric models do not require any labeled data, so training can be very cheap. While the field is still in its infancy, the object-centric nature of the real world suggests that robots and perceptual systems will likely require robust internal representations of objects in order to achieve full generality.

Chapter 2

Preliminaries

2.1 Variational Autoencoders

Below is a relatively formal introduction to variational autoencoders (VAEs). Many of the common tutorials on VAEs are not precise, so this section offers motivation to some of the mathematical objects present in the object-centric literature. For a more approachable tutorial, reference [8, 25].

Variational autoencoders are a cornerstone of many object-centric learning models [17, 32, 11, 3, 13]. VAEs come from a technique in statistics known as Variational Inference (VI). We will introduce VAEs in the context of classical VI. We set up our problem using a latent variable model. Let $\Omega = Z \times \mathcal{X}$ be an outcome space, where $Z = \mathbb{R}^{k \times d}$ is the latent space and $\mathcal{X} = \mathbb{R}^{h \times w}$ is the set of images of dimension (h, w). $\mathcal{F}$ is a σ-algebra on Ω, and $P : \mathcal{F} \to [0, 1]$ is a probability function. We define a complete probability space $(\Omega, \mathcal{F}, P)$ in which observations $x : \Omega \to \mathbb{R}^{h \times w}$ and latents $z : \Omega \to \mathbb{R}^{k \times d}$ are random variables on $(\Omega, \mathcal{F}, P)$. Here P is a probability measure over $(\Omega, \mathcal{F})$. In addition, we usually assume a gaussian prior on Z. Now we can relate P_θ to its density: $P_\theta((z, x) \in G) = \int_G p_\theta(x, z) dz dx$ for some event $G \in \mathcal{F}$.

In this framework, the aim in VI is to compute $p_\theta(z|x) = p_\theta(z, x)/p_\theta(x)$. However, computing the normalizing factor, $p_\theta(x) = \int p_\theta(x, z) dz$ is intractable in practice (because it would require integrating along an infinitely large set). Thus, variational methods get around this problem by approximating the posterior, $p_\theta(x)$ using $q_\psi(z)$ defined as the following [25]:

$$q_\psi = \arg\min_{q_\psi \in \mathcal{Q}} D_{\mathrm{KL}} \left(q_\psi(z) \,\|\, p_\theta(z \mid x) \right) \tag{1}$$

Here $\mathcal{Q}$ is the family of approximating distribution. The term "variational" comes from the fact that we are minimizing over functions, namely distributions q. An interesting connection is that this exact concept has roots in the calculus of variations, in which first-order perturbations of a cost function are minimized via a differential equation.

VAEs are a technique for finding q by jointly optimizing $q_\psi(z|x)$ and $p_\theta(x|z)$. To see how we can do that we start with writing the objective more explicity:

$$D_{KL}(q_\psi(z) \,||\, p_\theta(z \,|\, x) = \mathbb{E}_{q_\psi(z)}[\log q_\psi(z) - \log p_\theta(z \,|\, x)] \tag{2}$$

This is the definition of KL divergence. We can use Bayes rule to get:

$$D\left[q_\psi(z) \,\|\, p_\theta(z \mid x)\right] = \mathbb{E}_{q_\psi(z)}\left[\log q_\psi(z) - \log p_\theta(x \mid z) - \log p_\theta(z)\right] + \log p_\theta(x) \tag{3}$$

Here $\log p_\theta(x)$ comes out of the expectation since it doesn't depend on z. We can negate rearrange the equation into the following form:

$$\log p_\psi(x) - D\left[q_\psi(z) \,\|\, p_\theta(z \mid x)\right] = \mathbb{E}_{q_\psi(z)}\left[\log p_\theta(x \mid z)\right] - D\left[q_\psi(z) \,\|\, p_\theta(z)\right] \tag{4}$$

Since q_ψ is free to be anything we want, it makes sense for it to be conditioned on x so it can better approximate $p_\theta(z \mid x)$ as conditioning improves the approximation of the true posterior. Thus we get the following equation:

$$\log p_\theta(x) - D\left[q_\psi(z \mid x) \,\|\, p_\theta(z \mid x)\right] = \mathbb{E}_{q_\psi(z)}\left[\log p_\theta(x \mid z)\right] - D\left[q_\psi(z \mid x) \,\|\, p_\theta(z)\right] \tag{5}$$

Since we negated (3), the goal of the VAE is to maximize this objective. Note that this is exactly the Evidence Lower-Bound (ELBO), which is a statistical bound for the quantity $p_\theta(x)$. Thus, maximizing the VAE objective is equivalent to maximizing the ELBO. Intuitively, in the LHS of (5), maximizing $p_\psi(x)$ means our model is more likely to product training samples, and minimizing $D\left[q_\psi(z \mid x) \,\|\, p_\theta(z \mid x)\right]$ means that $q_\psi(z \mid x)$ is a good approximation of $p_\theta(z \mid x)$, as desired. However, both these metrics are intractable. Luckily, the RHS of (5) is something we can optimize through stochastic gradient descent given that we choose q_ψ to be differentiable. In VAEs, we usually choose $q \sim \mathcal{N}(\mu_\psi, \sum_\psi)$ where μ_ψ and $\sum_\psi$ are learned. We can think of $q_\psi(z \mid x)$ as an encoder and $p_\theta(x \mid z)$ as a decoder, which inspires many of the VAE-based models we will look at in the next chapter.

2.2 β-VAE

β-VAE is a modification of the standard VAE that introduces a tunable hyperparameter, β, to the standard VAE objective [14]:

$$\mathcal{L}(\theta, \psi; x, z, \beta) = \mathbb{E}_{q_\psi(z|x)}\left[\log p_\theta(x \mid z)\right] - \beta\, D_{\mathrm{KL}}\left(q_\psi(z \mid x) \,\|\, p(z)\right) \tag{6}$$

When $\beta = 1$, this is equivalent to the standard VAE, but it having $\beta > 1$ puts extra pressure on the posterior $q_\psi(z \mid x)$ to match the Gaussian prior $p_\theta(z)$. It is hypothesized that higher values of β encourage disentanglement of the latent space by pressuring z to be factorized while still being sufficient to reconstruct the data x [4]. Figure 2.1 demonstrates the intuition. Decreasing $D_{\mathrm{KL}}\left(q_\psi(z \mid x) \,\|\, p(z)\right)$ encourages posteriors corresponding to different latent axes to be closer to each other. However, this would cause a new datapoint $\tilde{x}$ sampled from, say $q_\psi(z_1 \mid x_2)$, is more likely to be confused with a different posterior. Thus, if a latent variable must deviate from the mean, the posterior is encouraged to represent neighboring points in the data space to avoid this confusion.

The next question is how does this increased latent factorization lead to a more disentangled latent space? There is no definitive proof for why this is seen in practice, but a hypothesis is that interpretable disentangled factors naturally provide the most variation in reconstruction quality [4]. This can be understood intuitively when thinking about a dataset of objects. In practice, often one of the first factors to be disentangled is the position. This makes sense, as reconstructing an object in the wrong position leads to extremely high error in $p_\theta(x)$. This suggests that curating datasets in certain ways can affect how the model learns what the most important factors of variation are.

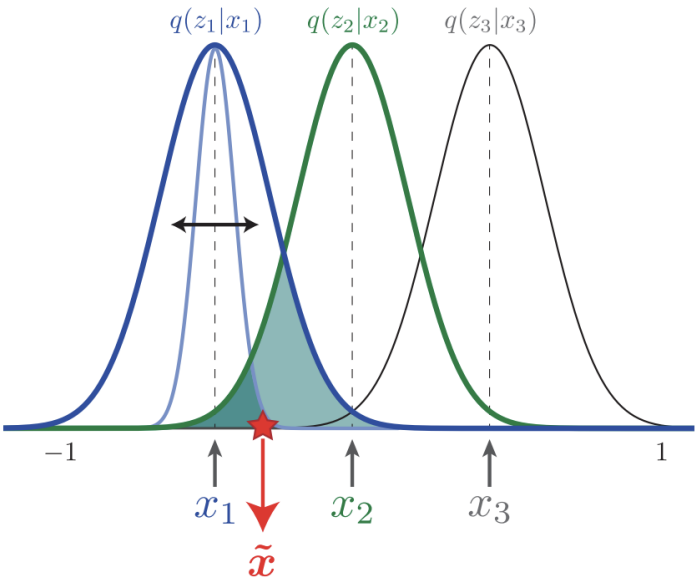


Figure 2.1: **Connecting posterior overlap with minimizing the KL divergence and reconstruction error.** Broadening the posterior distributions and/or bringing their means closer together will tend to reduce the KL divergence with the prior, which both increase the overlap between them. But, a datapoint $\tilde{x}$ sampled from the distribution $q_\psi(z_2|x_2)$ is more likely to be confused with a sample from $q_\psi(z_1|x_1)$ as the overlap between them increases. Hence, ensuring neighbouring points in data space are also represented close together in latent space will tend to reduce the log likelihood cost of this confusion

2.3 Latent Disentanglement

One key concept in object-centric learning is latent disentanglement. "Disentangled" in this context doesn't have a unifying definition, but one can describe a latent space as being disentangled when variation to a single element in the latent space corresponds to an interpretable outcome in the output space [23]. As we will see later, the β-VAE was created in order to optimize for this disentanglement property, and this has inspired many object-centric models that we will cover in this work.

Chapter 3

Approaches

In this section we discuss various approaches to object-centric learning in the literature. We start with an overview of common supervised methodology inspired by the rise of foundation models [28, 27] in order to give context to the unsupervised approaches, which promise to improve upon the interpretability and generalizability of supervised approaches.

3.1 Supervised Methods

Before we delve into the various approaches of unsupervised object-centric representation learning, we will take a look at some related supervised models that can deal with objects as a baseline.

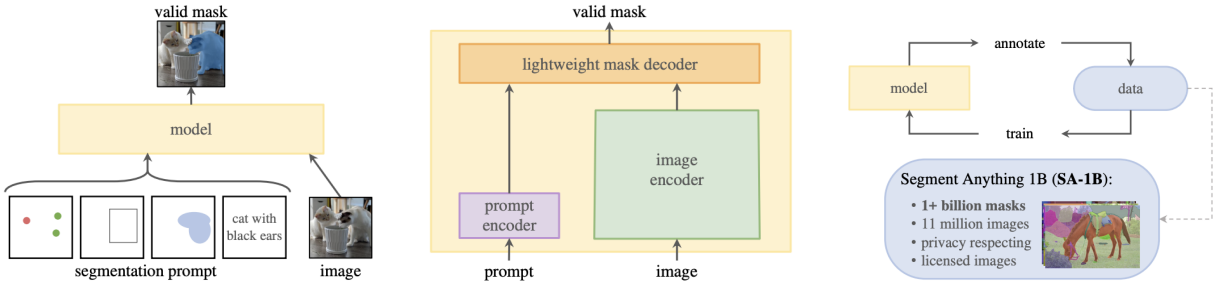
One of the most famous of these models is known as Segment Anything Model (SAM) [21]. SAM is a semi-supervised large vision model which takes in both images and prompts as input and outputs a relevant segmentation mask. The authors also allow the model to segment objects without prompts, or zero-shot. Their results show impressive segmentation results even for out-of-distribution data. The SAM methodology can be divided into three parts, as shown in Figure 3.1.

Task: The overall task in the SAM pipeline is to output a valid segmentation mask given an image and a prompt. The prompt can be not only text, but also background/foreground points, bounding boxes, masks, etc.

Model: SAM has three components: an image encoder, a flexible prompt encoder, and a fast mask decoder. The image encoder consists of a mask autoencoder and a vision transformer. The prompt encoder creates an embeddings from the prompts. For text prompts they use CLIP[26] and for visual prompts, they use positional encodings and convolutions. The lightweight mask decoder maps the image and prompt embeddings into masks. This module uses a transformer decoder block followed by an MLP classifier that takes in the transformer tokens and returns the mask foreground probability at each image location.

Data Engine: One of the most difficult tasks for SAM to address is how to get enough

labeled segmentation data. Labeling segmentation masks is expensive and unreasonable for creating entire large datasets. The authors of SAM propose a three-stage approach. The first stage is the assisted-manual stage in which a team of professional annotators label parts in an image that they could describe as an object. The second stage is the semi-automatic stage in which SAM creates masks that have high confidence, and the professional annotators are tasked with creating masks for any additional unannotated object. Finally, in the fully automatic stage, annotation is automatic, and the model ends up creating a dataset of 1.1 billion masks in 11 million images. The authors had to incorporate a few design decisions in order to ensure that SAM is generating reasonable masks by itself. For instance, they prompt the model with a grid of points so that each point predicts a set of masks that could all be valid interpretations of the scene. They also enforce model stability by selecting only masks which are invariant to small perturbations of the probability masks. These design choices are important because the definition of an object, even for humans, can be ambiguous. For instance, when describing a car, it can be ambiguous whether to consider the entire car an object or separate it into smaller objects such as wheel, door, etc. SAM handles this multi-modality by predicting multiple objects that a mask could belong to. The end result is that one can prompt SAM to segment an image at different "resolutions" of object specificity. In other words, one can prompt it to segment bigger objects, like cars, all the way to smaller objects, such as the wheels.



(a) Task: promptable segmentation

(b) Model: Segment Anything Model (SAM)

(c) Data: data engine and dataset

Figure 3.1: SAM is a foundation model for segmentation by using three interconnected components: a promptable segmentation task, a segmentation model (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a data engine for collecting SA-1B, a dataset of over 1 billion masks.

3.2 Core Object-Centric Works

MONet

Learning disentangled representations is one of the most most important goals in the object-centric literature. The aim is to curate a latent space such that objects are represented as slots, and each object contains a set of interpretable attributes. One of the first methods to

attempt this is known as Multi-Object Network (MONet)[3].

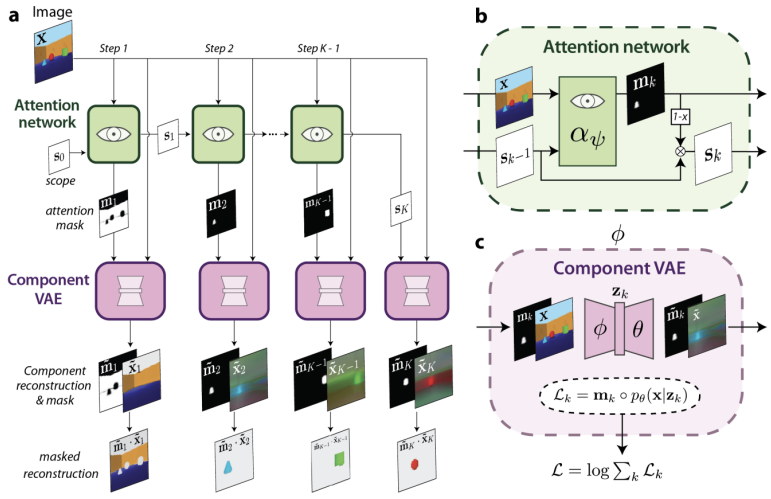


Figure 3.2: Schematic of MONet. **(a)** Overall compositional generative model architecture. The attention net recurrently generates the masks over a sequence of steps to condition the component VAE, labelling which pixels to focus on representing and reconstructing for that component. **(b)** Recursive decomposition process: the attention network at a particular step is conditioned on the image and the current scope, which is what currently remains to be explained of the scene (with the initial scope $s_0 = 1$). The attention mask outputted will be some portion of the scope, and the scope for the next step is then what still remains to be explained after accounting for this attention mask. **(c)**. The component VAE receives both the image and a mask as input, and is pressured only to model the masked region by applying the mask to weight the component likelihood in the loss. Thus the reconstruction component is unconstrained outside of the masked region, enabling it for example to fill in occluded regions. The VAE also models the masks themselves.

The MONet architecture features two main components, those being the attention network and the component VAE, as shown in Figure 3.2. Each iteration of the algorithm goes through K steps, each ideally representing an object. For step k, an input image, $\mathbf{x}$, and scope, s_k are fed into the attention network. The output of the attention network is a spatial mask, $\mathbf{m}_k$ corresponding to part of the scene designated by s_k. The scope is calculated using the following equation: $s_{k+1} = s_k(1 - \alpha_\psi(\mathbf{x}; s_k))$, where $\alpha_\psi(\mathbf{x}; s_k)$ is the output of a recurrent neural network that determines which section of the scene remains to be explained. The initial condition, $s_0 = 1$, representing that the entire image has yet to be explained.

Next, these attention masks are fed into a VAE, which is also conditioned on the image. Following the notation from section 2.1, we define the VAE encoder to be $q_\phi(\mathbf{z}_k|\mathbf{x}, \mathbf{m}_k)$, and the decoder is $p_\theta(\mathbf{x}|\mathbf{z}_k)$. In addition to images, the VAE models the attention masks. If we let $p(\mathbf{c}|\mathbf{z}_k)$ be the distribution of masks given the set of possible latents and $q(\mathbf{c}|\mathbf{x})$ be the

distribution of masks given an image, then the total VAE loss is the following:

$$\mathcal{L}(\phi;\theta;\psi;\mathbf{x}) = -\log \sum_{k=1}^{K} \mathbf{m}_k p_\theta(\mathbf{x} \mid \mathbf{z}_k) + \beta D_{\mathrm{KL}} \left(\prod_{k=1}^{K} q_\phi(\mathbf{z}_k \mid \mathbf{x}, \mathbf{m}_k) \,\|\, p(\mathbf{z}) \right) + \gamma D_{\mathrm{KL}} \left(q_\psi(\mathbf{c} \mid \mathbf{x}) \,\|\, p_\theta\left(\mathbf{c}(\{\mathbf{z}_k\})\right) \right) \quad (3.1)$$

Where the first two terms come from the standard VAE loss and the last term compares $q_\psi(\mathbf{c}|\mathbf{x})$, which is learned from the attention module, with $p_\theta(\mathbf{c}|\mathbf{z}_k)$ being learned from the VAE. Note that $p_\theta(\mathbf{x}|\mathbf{z}_k)$ is weighted by mask $\mathbf{m}_k$ to further promote disentanglement. MONet is one of the first papers to attempt to create a disentangled object-centric latent space using an unsupervised model. While MONet displayed impressive performance for its time, other more recent methods, especially ones that use transformers, exhibit better performance when it comes to segmentation and disentanglement. That being said, many of the strategies in MONet, such as attention modules and β-VAEs, are present in modern day object-centric models.

Slot Attention

Slot attention is a key work by Locatello et al. [24], which has provided the basis of many other unsupervised slot-based object-centric models. Slot-attention is a simple modification of a standard cross-attention module. Its simplicity and effectiveness is part of what makes it so appealing.

Slot attention uses an iterative attention mechanism. Algorithm 1 contains the method in pseudo-code. Slots are initialized as random $k \times d$ vectors where k is the number of slots and d is the slot dimension. At each iteration, the slots "compete" for explaining parts of an image. The attention matrix is then processes using a softmax operation over the slot dimension. As a result, each input image will generate a slot matrix in which the values in each row represent the probabilities that the object can be explained by that slot.

Comparison to Cross Attention in LLMs

Slot attention is a simple modification of cross attention, which is present in nearly every transformer. In cross attention (within a language context), the input embedding (which could represent a sentence, for instance) is compared to another string of text. Through many training iterations, the model learns how to predict the next word in a sequence given the context from data. In a sense, the weight matrices are learning to encode similarity of each word to all other words based on context. Slot attention, on the other hand, compares the input embedding (which in this case can be an image passed through a CNN) to the slot matrix, which is initialized at random. At each iteration, each input embedding is given a probability that it belongs to each slot through a softmax operation. The model is incentivized have each slot explain as much variation in the input image as possible as a result of the slot-competition process. It turns out that object-centric properties are generally the ones that can explain most of the variation in an image. This is an extremely useful result that has also been observed in other methods that we will soon look at.

Algorithm 1 Slot Attention Module

Input: $\texttt{inputs} \in \mathbb{R}^{N \times D_{\text{inputs}}}$, $\texttt{slots} \sim \mathcal{N}(\mu, \text{diag}(\sigma)) \in \mathbb{R}^{K \times D_{\text{slots}}}$

Layer params: k, q, v: linear projections for attention; GRU; MLP; LayerNorm (x3)

$\texttt{inputs} \leftarrow \texttt{LayerNorm(inputs)}$
for $t = 0 \dots T$ **do**
  $\texttt{slots\_prev} \leftarrow \texttt{slots}$
  $\texttt{slots} \leftarrow \texttt{LayerNorm(slots)}$
  $\texttt{attn} \leftarrow \texttt{Softmax}\left(\frac{1}{\sqrt{D}} k(\texttt{inputs}) \cdot q(\texttt{slots})^\top, \text{axis='slots'}\right)$ ▷ norm. over slots
  $\texttt{updates} \leftarrow \texttt{WeightedMean}(\texttt{weights=attn} + \epsilon, \texttt{values=}v(\texttt{inputs}))$ ▷ aggregate
  $\texttt{slots} \leftarrow \texttt{GRU(state=slots\_prev, inputs=updates)}$ ▷ GRU update (per slot)
  $\texttt{slots} \mathrel{+}= \texttt{MLP(LayerNorm(slots))}$ ▷ optional residual MLP (per slot)
end for
return slots

SIMONe

So far, we've only seen models that take in images as data and attempt to create object-representations of them. However, none of them do anything special given a video of a objects. Videos can be a very powerful source of data for unsupervised object-centric methods since object consistency can allow the model to learn the properties of the objects based on how their states evolve over time. SIMONe [17] aims to make use of this object-consistency by using solely RGB videos as inputs to create object-centric representations that encode properties like position, size, and even trajectory of motion.

SIMONe is another variational autoencoder with some modifications. An overview of the architecture is contained in 3.3. The key design decisions are a) a latent factorization of objects and frame latents, and b) a spatio-temporal handling of data using two transformers. As per the architecture diagram, a sequence of images is fed into a CNN, which outputs spatial patches. These patches are inputted into a series of transformers. The first transformer integrates spatio-temporal information, while the second transformer pools the output of the first transformer to match the intended slot dimension, and optimizes for object and frame disentanglement. The output of the transformers are fed through an MLP to get the variational latent mean and variance. VAE decoder inputs are sampled from the normal distribution with those means and variances.

Encoder

More precisely, let $X := \{\mathbf{x}_t\}_{t=1}^T$ be the sequence of frames. We want our VAE to infer the object latents, $\mathbf{O} := \{\mathbf{o}_t\}_{t=1}^T$ and frame latents, $\mathbf{F} := \{\mathbf{f}_t\}_{t=1}^T$. Thus the VAE posterior, $p(\mathbf{O}, \mathbf{F}|X)$ is approximated by a learned distribution $q_\phi(\mathbf{O}, \mathbf{F}|\mathbf{X})$. To estimate $\mathbf{O}$ and $\mathbf{F}$. In order to estimate $\mathbf{O}$ and $\mathbf{F}$, we use the following pipeline: X is fed into a CNN that outputs 64 spatial image patches for each timestep. These patches are then fed into transformer 1, $\mathcal{T}_1$. The output is a matrix of size $T \times 64$ representing the images with compressed spatio-temporal data through the transformer attention mechanisms. This matrix is then spatially

pooled so its size is $T \times K$ where K is the number of slots. This pooled matrix is then fed into transformer 2, $\mathcal{T}_2$. The purpose of $\mathcal{T}_2$ is to refine the token specifically for object representations. The output of $\mathcal{T}_2$, denoted as $\hat{\mathbf{e}}$, and $\hat{\mathbf{e}}_{t,k}$ is the output for frame t for slot k. In order to parameterize q, the authors "average" $\hat{\mathbf{e}}$ along the frame and object latents. More precisely, we get $\lambda_{\mathbf{o}_k} := mlp_o(1/T \sum_t \hat{\mathbf{e}}_{t,k})$ and $\lambda_{\mathbf{f}_t} := mlp_f(1/K \sum_k \hat{\mathbf{e}}_{t,k})$. We then sample $\mathbf{o}_k$ and $\mathbf{f}_t$ using the following distributions: $\mathbf{o}_k \sim \mathcal{N}(\lambda^{\mu}_{\mathbf{o}_k}, \exp(\lambda^{\sigma}_{\mathbf{o}_k})\mathbf{1})$ and $\mathbf{f}_t \sim \mathcal{N}(\lambda^{\mu}_{\mathbf{f}_t}, \exp(\lambda^{\sigma}_{\mathbf{f}_t})\mathbf{1})$. Here $\lambda^{\mu}_{\mathbf{o}_k}$ and $\lambda^{\sigma}_{\mathbf{o}_k}$ are the mean and variance of $\lambda_{\mathbf{o}_k}$ and $\lambda^{\mu}_{\mathbf{f}_t}$ and $\lambda^{\sigma}_{\mathbf{f}_t}$ are the mean and variance of $\lambda_{\mathbf{f}_t}$ respectively.

Decoder

The query decoder allows us to model each pixel as a gaussian mixture such that we can get the probability that slot k represents pixel i for each pixel. This decoder takes in the latent variables and a pixel mask and outputs mixture logits $\hat{m}_{k,t,i}$ and RGB reconstruction means $\boldsymbol{\mu}_{k,t,i}$. We have the following equations:

$$\hat{m}_{k,t,i},\ \boldsymbol{\mu}_{k,t,i} = \mathcal{D}_\theta(\mathbf{o}_k, \mathbf{f}_t; \mathbf{l}_i, t) \tag{8}$$

$$p(\mathbf{x}_{t,i} \mid \mathbf{o}_1, \ldots, \mathbf{o}_K, \mathbf{f}_t; t, \mathbf{l}_i) = \sum_k m_{k,t,i}\, \mathcal{N}(\mathbf{x}_{t,i} \mid \boldsymbol{\mu}_{k,t,i}; \boldsymbol{\sigma}_x) \tag{9}$$

Here $\mathbf{l}_i \in [0, 1]$ is the i'th component of a mask of queried pixels with the same shape as the input image. Querying allows us to choose which pixels get considered for the loss function or for evaluation. During training, we want to only use a subset of the total pixels since using all of them at each iteration is too computationally intensive. When performing evaluation, we simply have $\mathbf{l}_i = 1$ for all i. $m_{k,t,i} = softmax_k(\hat{m}_{k,t,i})$ is normalized in order to allow our gaussian mixture to be a valid probability. In an ideally disentangled latent space, we would have $m_{k,t,i} = 1$ if $k = j$ and $m_{k,t,i} = 0$ if $k \neq j$.

SIMONe is one of the first methods to incorporate sequences as input while also using a transformer architecture in combination with the β-VAE. This work is part of a general trend to include transformers in the encoding step. Transformers have been show to better capture complex features in visual data, and their effectiveness in other fields of computer vision have inspired and boosted performance in the object-centric domain.

3.3 Variants on Core Works

In this section we will introduce some variants on the above core frameworks, paying special attention to how the authors adapted the state-of-the-art methods and what additional design choices they made.

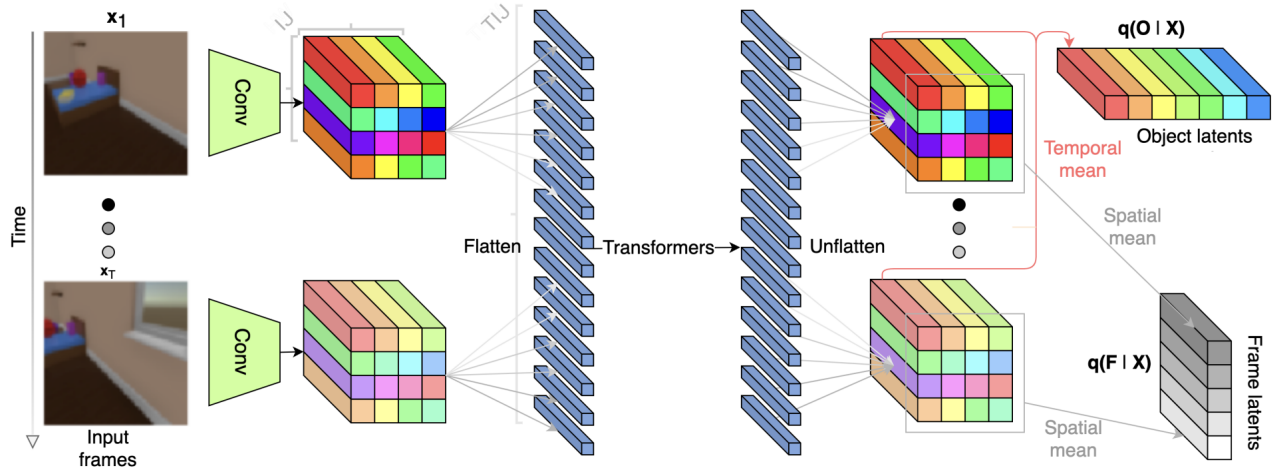


Figure 3.3: Architecture of the SIMONe inference network ε_ϕ. The transformers integrate information jointly across space and time to infer (the posterior parameters of) the object and frame latents.

SPOT

SPOT stands for *Self-Training with Patch-Order Permutation for Object-Centric Learning with Autoregressive Transformers* [18]. In this work they used a slot-attention based encoder. First they generate image patches using DINO features [5], which have been shown to be more effective than CNN's for unsupervised vision tasks. For the decoder, they use an autogressive transformer, which predicts features based on the features that came before it. Additionally, SPOT employs a teacher-student knowledge distillation approach to enhance object-centric learning. In this teacher-student model, they initially train the teacher on purely reconstruction loss, and then that model teaches the student model, which introduces an attention mask loss.

One hurdle that the authors of SPOT had to resolve was the fact that tokens in the first row and column of the image contributed significantly more to the gradients than any other patch. This is likely because those patches were always used first in the autoregressive transformer, and thus every subsequent patch is informed by the first one. To combat this problem, the authors introduce a patch-permutation strategy in which the order in which the autoregressive transformer predicts patches is randomized. They found that SPOT had superior segmentation performance for complex scenes compared to other method. This work truly shows how promising vision transformers are when combined with slot attention.

Linking Vision and Motion for Self-Supervised Object-Centric Perception

This work by Stocking et al. [32] introduces the Self-Supervised Object-Centric Segmentation (SOCS) model, which builds off of SIMONe to provide high-quality object-centric represen-

tations for real-world driving videos. In this paper, instead of learning frame latents, like SIMONe, they provide the ground-truth transformation matrix for each input image, which is possible when the camera motion and positions are predictable. They also convert the reconstruction decoder from a unimodal gaussian distribution to a gaussian mixture, allowing for better handling of multi-colored objects.

3.4 Takeaways

State of the art unsupervised object-centric methods generally take the following approach: an encoder transforms an image into a latent representation that aims to disentangle objects into slots, and then using a decoder to reconstruct the image from the latents. For the encoder, most works use either variational methods or slot-attention. The advantage of β-VAE methods is that they are able to disentangle not only objects, but also their properties, such as size and shape. However, the attention-based methods generally have superior segmentation performance. Future work will continue to find superior architectures that build off of slot-attention and VAEs, but maybe down the line there will be an alternative to the slot-based methods that have become so common.

In Chapter 1, we talked about the current research on object-centric perception in animals. While slot-based methods claim to be inspired by biological perception, there are still some major differences that affect generalization. For instance, some animals have been shown to have object-centric representations from birth, after very little time looking at objects. However, for these object-centric representations to be successful they require hundreds of thousands of images of objects from different angles. This innate abject-centric ability in animals has not yet been reproduced in machine-learning, and solving this problem would massive implications for generalizability in this field.

Chapter 4

Object-Centric Disentangled Representation Learning Theory

In the previous chapter, we looked at an assortment of key papers in the unsupervised object-centric literature. One unifying theme for these methods is that they aim to encode scenes into a disentangled latent space, in which each slot represents a single object. However, the reason for why this disentangled behavior emerges has historically not been well understood. As a result, researchers have started to look into the theory of object disentanglement. A key work by Brady et al. [1] investigates under which circumstances one can prove that a sufficient model can disentangle a scene. In this chapter, we will explore this work by Brady et al. and theorize how this can be extended to sequences of 3D objects.

4.1 Object-Identifiability Theory in 2D scenes

The aim of disentanglement theory is to prove identifiability results. Identifiability is defined as the ability for an inference model to uniquely recover a set of ground truth latents given an observation. Figure 4.1 gives a diagram of the general setup. Theory on disentanglement learning is largely based on generative methods, such as the VAE [20]. Consistent with the VAE, these models assume that latents are generated from some prior $p(z)$, in which the aim is to parameterize $p(\mathbf{x}|\mathbf{z})$. We call this probability the generator and refer to it as $\mathbf{f}$. It has been proven that identifiability is impossible without assumptions on $\mathbf{f}$ [23]. However, this does not mean that identifiability is impossible in general. Some works guarantee identifiability through imposing restrictions on the prior latent distribution [19]. The authors of [1], however, make no such assumptions and instead impose inductive biases on $\mathbf{f}$. Specifically, the necessary properties of $\mathbf{f}$ are compositionality and irreducibility. Informally, compositionality implies that every pixel in a scene corresponds to only one object. Irreducibility requires that information is shared across different parts of the same object, but not between objects.

Figure 4.1 shows a diagram of our setup. We assume that there exist a set of ground truth latents $\mathbf{z}_k$, where $1 \leq k \leq K$. Then our scene, $\mathbf{x}$, is generated from these latents by $\mathbf{f}$, which is some ground truth generator. We will see later that $\mathbf{f}$ is required to be a bijection,

meaning that each $\mathbf{z}$ produces a unique scene $\mathbf{x}$ and vice versa. It's worth noting that this is a rather strong assumption, that is impossible to guarantee from real data. This is one limitation in the current disentanglement theory that has caused experiments to be in simulation, which allows researchers to have ground truth latents and generators that can be used for evaluation. Once we have $\mathbf{x}$, we train an encoder $\hat{\mathbf{g}}$ to output the inferred latents $\hat{\mathbf{z}}_k$, where $1 \leq k \leq K$. We also have a trained decoder $\hat{\mathbf{f}}$ which aims to approximate $\mathbf{f}$. Proving identifiability in this setup would mean that each $\mathbf{z}_i$ corresponds to one $\hat{\mathbf{z}}_j$ and vice versa. In other words, the object represented by $\mathbf{z}_i$ must also be represented by some $\hat{\mathbf{z}}_j$ where i may more may not be equal to j. The reverse also has to be true so there is a one-to-one mapping between ground-truth and inferred latents.

Mathematical Formulation

Let $\mathcal{Z} = \mathcal{Z}_1 \times ... \times \mathcal{Z}_K = \mathbb{R}^{KM}$ be a latent space where $\mathcal{Z}_k = \mathbb{R}^M$. Formulated this way, the latent space matches our theory in Chapter 3 with K representing the number of slots and M being the slot dimension. $\mathbf{z}$ is sampled from some probability $p_{\mathbf{z}}$, which is fully supported on $\mathcal{Z}$. Here, $\mathbf{z} = (\mathbf{z}_1, ...\mathbf{z}_K) \in \mathcal{Z}$ We define $\mathbf{x} = \mathbf{f}(\mathbf{z})$. In order to impose restrictions on $\mathbf{f}$, it is useful to define a set that stores all of the pixels affected by the latent in slot k. We define:

$$I_k(\mathbf{z}) := \left\{ n \in [N] : \frac{\partial f_n}{\partial \mathbf{z}_k}(\mathbf{z}) \neq \mathbf{0} \right\}. \tag{10}$$

Here N is the number of pixels in the image and n is the index of the flattened image. f_n is the generator for the nth pixel. In words, this set contains all the pixels that are affected by slot $\mathbf{z}_k$ once they are generated by $\mathbf{f}$. Using this definition, we can formally define compositionality:

Definition 1 (Compositionality). *Let $\mathbf{f} : \mathcal{Z} \to \mathcal{X}$ be differentiable. $\mathbf{f}$ is said to be compositional if*

$$\forall\, \mathbf{z} \in \mathcal{Z} : \quad k \neq j \implies I_k(\mathbf{z}) \cap I_j(\mathbf{z}) = \varnothing. \tag{11}$$

This compositionality constraint is actually a significant constraint. The implication is that $\mathbf{x}$ cannot have objects overlap since that would cause the same pixel to be able to be explained by multiple latent slots. With the compositionality constraint there's still another issue. Remember the goal is to prove that for all $\mathbf{z}_i$ there exists a unique corresponding $\hat{\mathbf{z}}_j$ and vice versa. This requires us to have to think about what it means for pixels to be part of an object. Intuitively, humans have an understanding of what objects are. But in certain cases it can be ambiguous whether an object is part of a larger object or not. For instance, when you look at a car, you recognize it as one object, but you can also describe the car using smaller objects, such as wheels, doors, and so on. Thus, the authors had to create a constraint that would ensure that pixels that one would consider as part of the same object are also considered as the same object according to the model. This is where the irreducibility constraint comes in. Intuitively, we want to remove the case that one latent slot describes two or more objects. We formalize this concept below:

First, we define a mechanism:

Definition 2 (Mechanism). $\forall\, \mathbf{z} \in \mathcal{Z}$, $k \in [K]$, *we define the* k^{th} *mechanism of* $\mathbf{f}$ *at* $\mathbf{z}$ *as the Jacobian matrix* $\mathbf{Jf}_{I_k}(\mathbf{z})$.

Here we define $\mathbf{Jf}_{ij} = (\frac{\partial f_i}{\partial z_j})$ and $\mathbf{Jf}_{I_k}(\mathbf{z})$ contains only the pixels in the set $I_k(\mathbf{z})$. In words, the kth mechanism is a sub-matrix of $\mathbf{Jf}$ in which the rows correspond to pixels affected by slot k. We also define a submechanism:

Definition 3 (Sub-Mechanism). $\mathbf{Jf}_S(\mathbf{z})$ *is said to be a sub-mechanism of* $\mathbf{Jf}_{I_k}(\mathbf{z})$, *if* $S \subseteq I_k(\mathbf{z})$ *and* S *is nonempty.*

The authors use these mechanisms to define a notion of independence between two subsets of pixels in the observation. The aim here is to determine whether the latent capacity necessary to generate $S_1 \cup S_2$ is the same as that required to generate S_1 and S_2 individually. This can be done through a independence of sub-mechanisms as in definition 4.

Definition 4 (Independent/Dependent Sub-Mechanisms). *Let* $S_1, S_2 \subseteq [N]$ *and* $\mathbf{z} \in \mathcal{Z}$. *The sub-mechanisms* $\mathbf{Jf}_{S_1}(\mathbf{z})$ *and* $\mathbf{Jf}_{S_2}(\mathbf{z})$ *are said to be independent if:*

$$\operatorname{rank}\left(\mathbf{Jf}_{S_1 \cup S_2}(\mathbf{z})\right) = \operatorname{rank}\left(\mathbf{Jf}_{S_1}(\mathbf{z})\right) + \operatorname{rank}\left(\mathbf{Jf}_{S_2}(\mathbf{z})\right). \tag{12}$$

Conversely, they are said to be dependent if:

$$\operatorname{rank}\left(\mathbf{Jf}_{S_1 \cup S_2}(\mathbf{z})\right) < \operatorname{rank}\left(\mathbf{Jf}_{S_1}(\mathbf{z})\right) + \operatorname{rank}\left(\mathbf{Jf}_{S_2}(\mathbf{z})\right). \tag{13}$$

With these definitions, we are finally able to define irreducibility, which would enforce that the generator is not able to assign one slot to multiple objects.

Definition 5 (Irreducibility). $\mathbf{f}$ *is said to have irreducible mechanisms, or is irreducible, if for all* $\mathbf{z} \in \mathcal{Z}$, $k \in [K]$ *and any partition of* $I_k(\mathbf{z})$ *into* S_1 *and* S_2, *the sub-mechanisms* $\mathbf{Jf}_{S_1}(\mathbf{z})$ *and* $\mathbf{Jf}_{S_2}(\mathbf{z})$ *are dependent in the sense of Defn 4.*

The next step in the theory is to understand under what conditions an inference model $\hat{\mathbf{g}} : \mathcal{X} \to \mathcal{Z}$ can capture the ground truth object representations. Ideally we would have that $\hat{\mathbf{g}} = \mathbf{g} = \mathbf{f}^{-1}$. However, it's not necessary to have this strict of a condition. Instead, the authors develop a notion of slot identifiability in which each inferred latent spot captures information from exactly one ground truth latent slot. This notion is formalized as follows:

Definition 6 (Slot Identifiability). *Let* $\mathbf{f} : \mathcal{Z} \to \mathcal{X}$ *be a diffeomorphism. An inference model* $\hat{\mathbf{g}} : \mathcal{X} \to \mathcal{Z}$ *is said to slot-identify* $\mathbf{z} = \mathbf{g}(\mathbf{x})$ *via* $\hat{\mathbf{z}} = \hat{\mathbf{g}}(\mathbf{x}) = \hat{\mathbf{g}}(\mathbf{f}(\mathbf{z}))$ *if for all* $k \in [K]$ *there exist a unique* $j \in [K]$ *and a diffeomorphism* $\mathbf{h}_k : \mathcal{Z}_k \to \mathcal{Z}_j$ *such that* $\hat{\mathbf{z}}_j = \mathbf{h}_k(\mathbf{z}_k)$ *for all* $\mathbf{z} \in \mathcal{Z}$.

What this definition is saying is that if an inference model is slot identifiable, then each inferred latent spot corresponds with exactly one ground-truth slot. Note that if the generator

is also irreducible, then the reverse is also true. However, this definition does not require $\hat{\mathbf{z}}_k$ to be equal to some $\mathbf{z}_j$. Rather, there just has to be some bijective mapping between the two latents. This ensures that both slots are encoding the information necessary to generate the same object, even if that information is encoded differently in each slot. The authors then use this definition to state a theorem, which they prove in their paper. The theorem is below:

Theorem 1. *Let $\mathbf{f} : \mathcal{Z} \to \mathcal{X}$ be a diffeomorphism that is compositional (Defn. 1) with irreducible mechanisms (Defn. 5). If an inference model $\hat{\mathbf{g}} : \mathcal{X} \to \mathcal{Z}$ is (i) a diffeomorphism with (ii) compositional inverse $\hat{\mathbf{f}} = \hat{\mathbf{g}}^{-1}$, then $\hat{\mathbf{g}}$ slot-identifies $\mathbf{z} = \mathbf{g}(\mathbf{x})$ in the sense of Definition 6.*

This theorem shows that using our definitions for compositionality and irreducible mechanisms, we can guarantee slot identifiability regardless whether there are any distributional assumptions on $\mathbf{z}$. It is interesting to note that despite the fact no state-of-the-art currently optimizes for compositionally, which would lead to slot identifiability, many of them naturally do this based on inductive biases.

Theorems such as this are promising for allowing researchers to understand exactly why unsupervised object-centric models in some situations and not others. However, there are some limitations to this theory. First, this theory does not account for any time an object has to overlap with another object, or if there is an object behind a translucent object. Additionally, in practice, it is desirable for $\mathbf{f}$ to be permutation invariant, which would require it to not be invertible. In the next section we will hypothesize on ways one might develop new theory to mitigate some of these concerns.

4.2 Theory of Slot Identifiability in Videos

In the previous section, we showed how theory guarantees slot identifiability with certain restrictions on the generator $\mathbf{f}$. However, one clear limitation is that it's not possible to account for overlapping objects in the observation $\mathbf{x}$. This is because one slot could have two objects associated with it, which would violate compositionality. However, one might think about extending this theory to videos, or sequences of consecutive frames. It makes intuitive sense that if we have access to multiple frames of an object moving that we would be able to uniquely identify that object with an inferred latent slot as per Definition 6. Thus, the end goal might be to come up with a version of Theorem 1 that no longer requires $\mathbf{f}$ to be compositional in each individual image, but still allow it to be slot identifiable.

It's easiest to start thinking about this problem first in terms of assumptions. First of all, if two objects do not move throughout the entire sequence, slot identifiability is no longer guaranteed for a similar reason as lack of compositionality in the image case. Second, it is necessary that each pixel corresponding to a specific object is visible at some point during the sequence. Otherwise slot identifiability is impossible since there would be infinitely many ways to generate the hidden content.

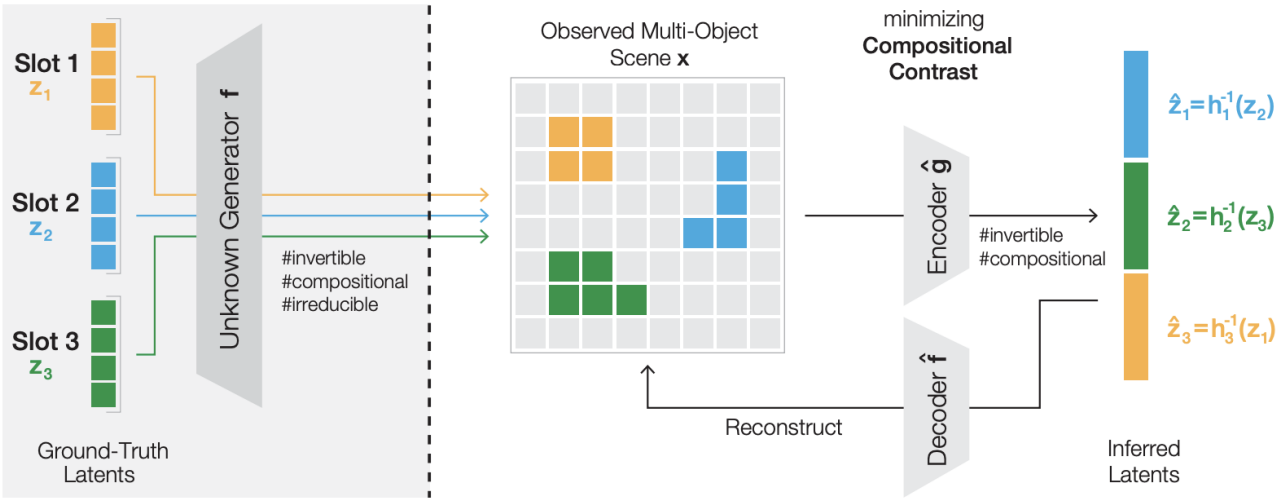


Figure 4.1: When can unsupervised object-centric representations provably be learned? We assume that observed scenes $\mathbf{x}$ comprising K objects are rendered by an unknown generator f from multiple ground-truth latent slots $\mathbf{z}_1, ..., \mathbf{z}_K$ (here, $K = 3$). We assume that this generative model has two key properties, which we call compositionality and irreducibility. Under this model, we prove theorem 1: An invertible inference model with a compositional inverse yields latent slots $\hat{\mathbf{z}}_i$ which identify the ground-truth slots up to permutation and slot-wise invertible functions $\mathbf{h}_i$. To measure violations of compositionality in practice, we introduce a contrast function which is zero if and only if a function is compositional, while to measure invertibility, we rely on the reconstruction loss in an auto-encoder framework.

Let's say the set of images is $\mathcal{X} = \mathbb{R}^{T \times H \times W}$ where T is the number of frames in the sequence, and $H \times W$ is the resolution of each image. One hypothesis is that, for each image in a sequence, it's possible to "lift" each object into its own image, then the problem reduces into standard slot-identifiability as presented in the previous section. More formally, say there exists some function $\mathcal{F} : \mathcal{X} \to \mathcal{X}^+$ where $\mathcal{X}^+ = \mathbb{R}^{T \times K \times H \times W}$. This transformation representing a lifting of each object into its own image. A proper proof of a new theorem would be more precise about what situations guarantee that a function like $\mathcal{F}$ exists. However, if we assume $\mathcal{F}$ exists, it's trivial to prove that slot identifiability exists on the observation space $\mathcal{X}^+$, since compositionality still holds on a generator that creates an observation in $\mathcal{X}^+$.

We leave the proof of this to future work. The possible implications of this result could be useful for identifying situations in which slot assignments might be ambiguous. Having the extra context from videos is something that is often realistic to have access to, and it allows this theory to be applicable to even more situations. Another interesting avenue for future work beyond this would be to prove a theorem with a relaxed notion of slot identifiability that can mitigate some of the assumptions necessary for compositionality to hold. For example, incorporating uncertainty to make guarantees on exactly what the model doesn't know can be an interesting direction. This could also have ties into partial observability, and a future robotics system might be able to make certain guarantees based on what it's seen that can

inform its decisions in order to learn more about what its uncertain about.

Chapter 5

Improving Object-Centric Methods

This section will describe some of my work improving and discovering properties about unsupervised object-centric methods. These projects are based off of SOCS, which was described in section 3.3.

5.1 Improving Object-Centric Representations and Segmentations using Knowledge Distillation

Introduction and Motivation

To reiterate from Chapter 1, unsupervised object-centric learning is incredibly promising in the field of robotics. However, while most current object-centric methods purely use simulation or real-world data, not many of them have yet been deployed on a robot in real time. Despite this, even though this technology is in its infancy, it is reasonable to believe that future robotic perception systems will have to be object-centric, as we have observed this in biology. One of the reasons that current object-centric models aren't effective enough for real-world use is that their segmentation performance isn't as good as state-of-the-art segmentation models.

As we saw in Chapter 2, supervised models, such as SAM, have impressive performance when it comes to zero-shot object segmentation. However, they suffer from a few drawbacks. First, the performance of the model relies on a vast amount labeled data, which is expensive to obtain. Second, the model doesn't have any internal representation of objects in latent space. This limits the potential of the model to generalize to new tasks, especially ones that require sustained knowledge of objects, as would be the case of search-and-rescue and pick-and-place tasks. Unsupervised object-centric models, on the other hand, do not require any labeled data and promise improved generalizability as a result. In addition, they also include latent spaces that are far more interpretable than those offered by large foundation models. However, these object-centric methods currently struggle with segmentation performance compared to SAM. The aim of this work is to bridge the gap between supervised and unsupervised object-centric and segmentation models. We do this by combining the knowledge of a large supervised segmentation model, such as SAM, with the object-centric

representations of an unsupervised model using a knowledge distillation approach. We show in this work that we got improved results from the baseline in terms of segmentation, with a tradeoff between reconstruction quality and segmentation performance.

Method

The teacher-student model is based on SOCS, which we briefly described in section 3.3. Our architecture is shown in Figure 5.6. The top route of this diagram comes from SOCS using one viewpoint. Here, image sequence are fed into a CNN encoder that outputs patches which go through a series of transformers. In section 3.2, we talked more in depth about SIMONe and how their transformers work, which is also true in this work. The transformers output slot tokens that parameterize the distribution of a VAE (this is $q_\psi(z \mid x)$. A sample of this distribution represents the object slots, which are then pooled per object and sent to a query decoder. The query decoder takes a randomly sampled grid of the pixels are uses them to compute a loss. We can also create a reconstruction of the input using a query of all the pixels in an image.

The bottom route of 5.6 is the teacher-student contribution. Images are fed into a teacher-model, such as SAM2 [27]. For our experiments, instead of SAM2, we simply took the ground-truth mask and sample one of the object masks. Eventually, the goal is to do this with SAM2 masks, but we hypothesis that performance will not be much different because, while SAM2 is not great at segmenting every object in a scene, it is great at segmenting objects that it does detect. Using ground-truth masks allows us to test only the knowledge distillation part of our experiment and not the quality of SAM2 masks. Additionally, since we're only using a single random mask as the teacher segmentation, SAM2 will likely always output a decent mask. Using these teacher masks, we determine which pixels of the initial query are inside the segmented object. Those pixels are compared with the resulting segmentation from SOCS to create a mask loss, which is added to the reconstruction loss and the distribution loss.

The way we compute the mask loss is as follows. We pick a random object (let's call this O_r) from the segmentation mask (we pick only one object for efficiency reasons). Then we choose a set of 10 random pixels within the object and 10 random pixels outside of O_r. Inside our model, we use the queried pixels to determine the slot (S_d) which most likely corresponds to O_r. We do this by figuring out which slot latent outputs the highest probabilities for pixels in our query. Using our queries, we construct the following matrix in two parts: The left side is the predicted RBG probabilities of pixels inside O_r per slot. The right side is the predicted RGB values of the queried pixels outside of O_r. Let's call this matrix P. Next, we construct a "ground truth" matrix (G) with two parts: the left side sets the probability of pixels in O_r and S_d to 1 and 0 respectively otherwise. The right side sets the probability of pixels chose outside of Or in Sd slot to be 0. We don't care about values outside of O_r that are not in S_d, so those can left out of the comparison. To compute the mask loss, we perform the following calculation:

$$\mathcal{L}_{mask} = ||P - G||_2^2 \tag{14}$$

Thus the total loss is:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{mask} + \mathcal{L}_{recon} + \beta\mathcal{L}_{KL}, \tag{15}$$

Where:

$$\mathcal{L}_{KL} = \sum_{k} D_{KL}(q_k(z_k \mid x) \,||\, p_\theta(z)), \tag{16}$$

and

$$\mathcal{L}_{recon} = \frac{-1}{N} \sum_{n} \log p(x^{(n)} \mid o_1, ..., o_K). \tag{17}$$

Here, x are the input frames, $q_k(z_k \mid x) = \mathcal{N}(\mu_k, \sigma_k)$ where μ_k and σ_k are learned from the VAE encoder for slot k. Moreover, n is the current pixel and $o_k \sim q_k(z_k \mid x)$ is the object latent for slot k. α is a hyperparameter that determines the weight of the mask loss. Notice now in equation 15, we weight $\mathcal{L}_{KL}$ by β. This is in spirit of the β-VAE, and setting $\beta > 1$ encourages disentanglement.

Results

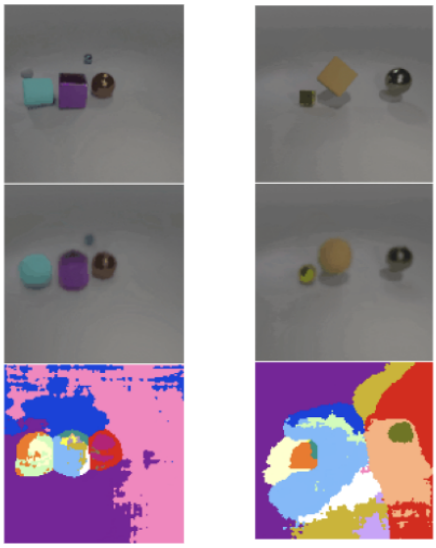
We evaluated the teacher-student model on the MOVI-A dataset [12] and compared the output to the original SOCS model. The result of one of the sequences is in Figure 5.1. The top images correspond to the original sequence, the second row is the VAE reconstruction, and the bottom row contains the instance mask segmentations. In the VAE segmentation output, each pixel is given a color based on which slot most likely describes it. Thus, perfect result would have each object only being represented by one color, indicating that those pixels correspond to one slot. We observe that the teacher-student model did a better job at making out the shapes in the scene rather than confusing them with the background. The teacher-student model was trained with 462,000 steps while SOCS was trained with 1,300,900 steps. This means that the teacher-student model did a better job with fewer training iterations. We noticed this result in a variety of scenes as well, indicating that the teacher-student model consistently outperforms SOCS in terms of segmentation performance while keeping a similar level of reconstruction quality.

For the mask loss we found that performing a scheduling strategy was most effective. This essentially means the weight of mask loss initially starts very low and then increases logarithmically as training progresses. Our hypothesis for why this works better is because early, the model learns to encode larger features such as the background. It's only later in training when individual objects are encoded. Thus it doesn't makes sense to use mask loss early in training since SOCS has not yet learned how to reconstruct the object corresponding to the mask.

We did notice a tradeoff between reconstruction loss and mask loss. In testing we noticed that for the same scene, while we do get slightly better segmentation performance for the teacher-student model, the reconstruction quality is slightly worse. This suggests that tuning

α hyperparameter in the loss function has a large effect on performance.

Figures 5.2-5.5 show loss curves associated with the teacher-student model during training. We find that training loss generally goes down, but can sometimes be erratic due to suboptimal stochastic initialization.



(a) (b)

Figure 5.1: Results showing comparison between (a) teacher-student model and (b) original SOCS model. Both models were trained the exact same hyperparameters other than those associated with mask loss, which was incorporated in (a) but not in (b).

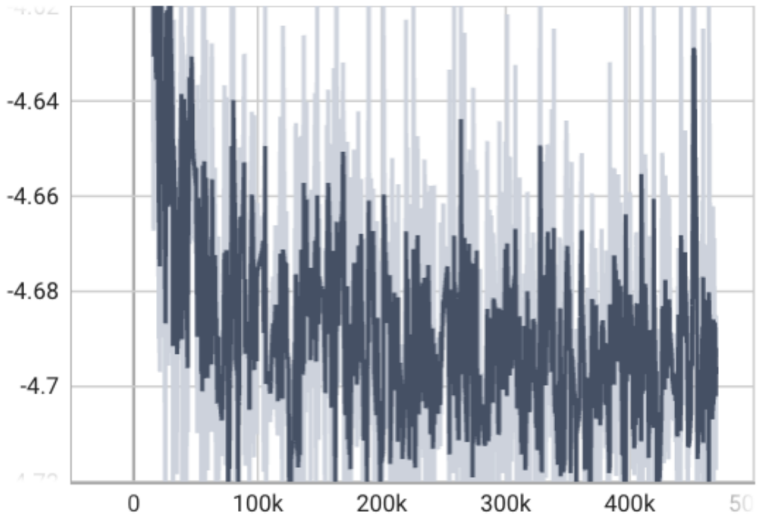


Figure 5.2: Reconstruction loss from training the teacher-student model (loss vs. iteration)

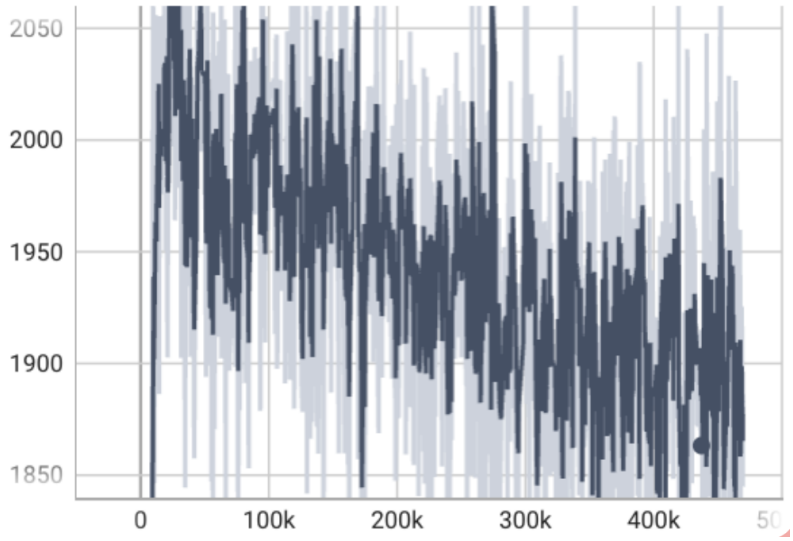


Figure 5.3: Distribution loss from training the teacher-student model (loss vs. iteration)

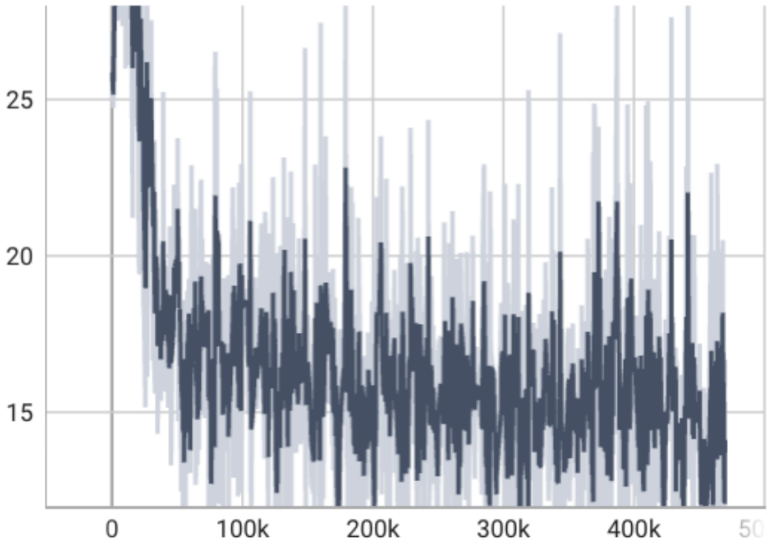


Figure 5.4: Mask loss from training the teacher-student model (loss vs. iteration)

Discussion

These results are promising and show that the segmentation quality of unsupervised object-centric models can be improved by using a better segmentation model as a teacher. Arguably the primary benefit of object-centric models, especially ones based on the β-VAE, is that the latent space contains a representation of objects, including their various properties. Thus, with access to large segmentation models such as SAM that don't have object-centric representations, it's possible to combine the best of both worlds and have a strong segmentation model with object-centric representations. It is also possible to use other segmentation mod-

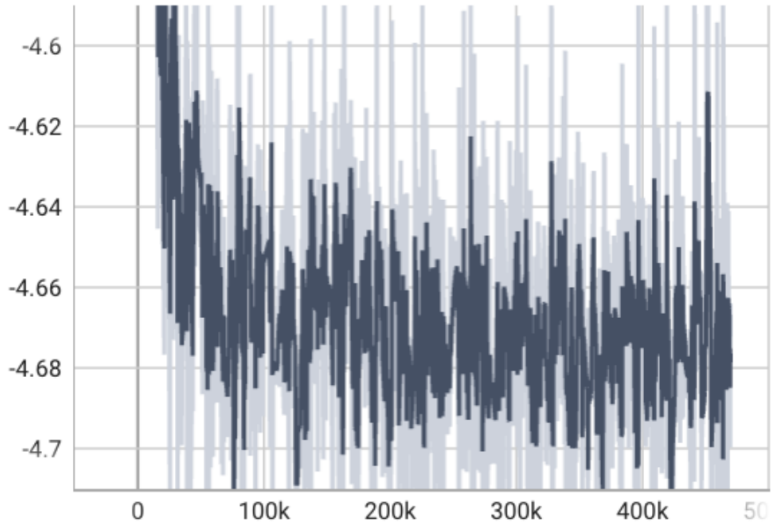


Figure 5.5: Total combined loss from training the teacher-student model (loss vs. iteration)

els, even unsupervised ones, as teachers instead of SAM, as long as we can extract teacher masks. However, the major limitation of this work is the tradeoff between taught segmentation performance and reconstruction loss, and mitigating this could be a good direction for future work.

5.2 How Datasets Affect Real World Performance

Motivation

In this section, we introduce some work aimed at bridging the sim-to-real gap of unsupervised object-centric learning. Most papers in the literature cite robotics as a promising application of unsupervised object-centric learning. However, very few of them actually test their models on real robots. At best they use real-world datasets, such as with SOCS. In this work, we investigate how different types datasets affect reconstruction and segmentation performance for both real-world and synthetic inputs. These datasets include fully synthetic, fully real-world, and combined synthetic/real-world datasets. Our results show that the model attempts to reconstruct the input by piecing together what it's seen in the training data. In some cases, such as with training with the real-world and combined datasets, we see decent segmentation performance. We hypothesize that with more training data and better dataset variety/curation, it could be possible to model any scene by piecing together data elements that the model has seen before.

Methods

At a high level, our process consisted of three main components. These were: real world environment and data collection on Turtlebot, segmentation model development and training,

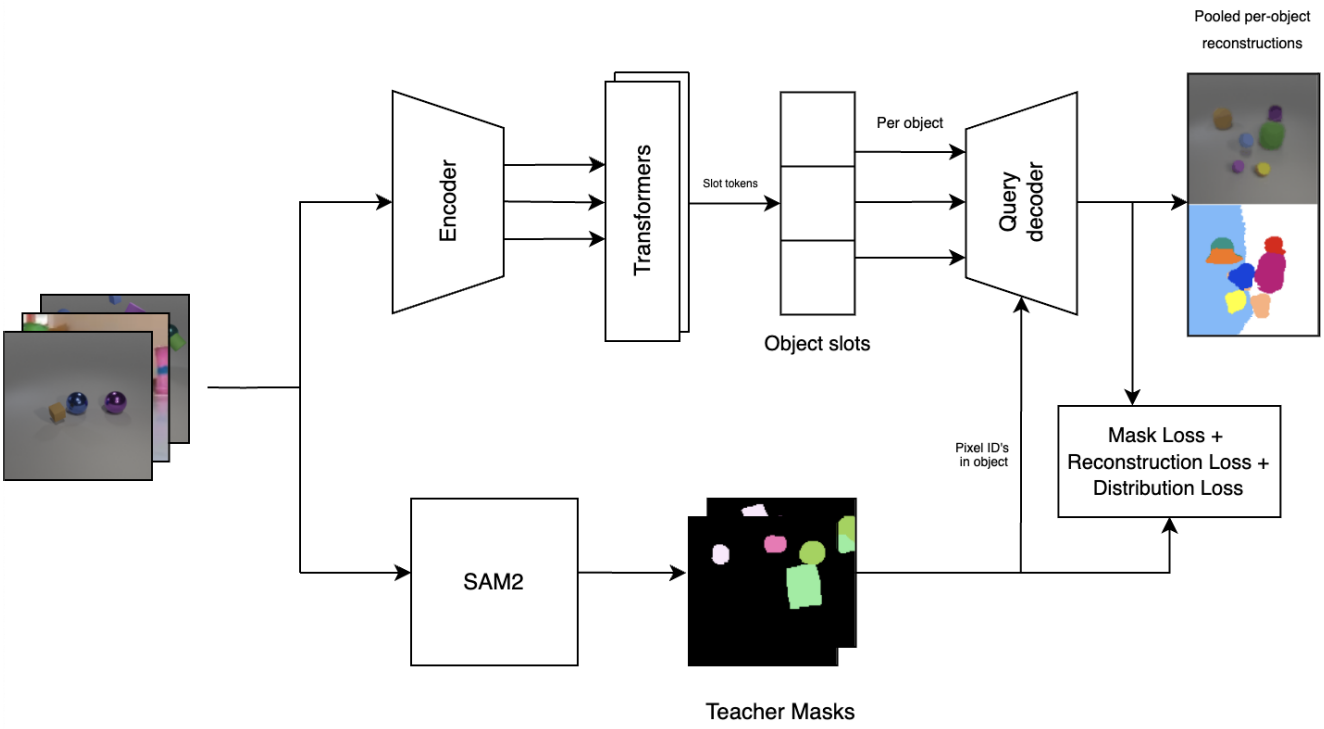


Figure 5.6: An overview of the proposed model architecture. From left to right on the top: video sequences are inputted into the CNN encoder, which outputs patches that are inputted the transformer. The transformer outputs slot tokens which are used to parameterize the VAE distribution. The object slots are then sampled from this distribution. These slots are then pooled over time and inputted into a query decoder. Finally the query decoder outputs per-slot probabilities, which can then be thresholded to get the segmentations From left to right on the bottom: Input images are also fed into the SAM2 model, which produces teacher masks. Then a subset of pixels from these masks are queried and compared to the VAE segmentation to calculate a mask loss.

and model evaluation and application. For the first component, we start with the construction of a real world experimental environment designed to replicate a search and rescue environment using a Turtlebot3. We then utilize the Turtlebot3, with a mounted RealSense camera, to perform numerous traversals through the environment. Our environment was designed such that it contained a variety of random and unique objects, with solid colored cups serving as a stand in for rescue "targets" and thus were the focus of the model evaluation. The positions and orientations of objects, walls, and obstacles were random, and frequently shuffled and changed to provide as unique of a training set as possible. The Turtlebot explored the environment via both autonomous and manual control, recording training data in the process.

To test the SOCS, we decided to train on three different datasets: one with solely synthetic data, one with solely real world data, and one with both. The goal of here was to test

the efficacy of the SOCS model in an indoor search and rescue task, where the lighting and terrain is different than that of the open road dataset which SOCS has previously been trained on. Since manual collection of real world data is a naturally time-intensive process, we decided to supplement it with synthetic data and then see how models trained on different combinations of these datasets would compare to each other. This analysis would be performed by qualitatively examining the video reconstructions and mask accuracy, as well as by quantitatively examining the distribution and reconstruction loss curves. We generated synthetic data using Multi-Object Video (MOVi), creating series of two second videos of several rigid objects falling into a random scene. We specifically made use of MOVi-A, which uses simpler, geometric objects such as spheres, cubes, and cylinders with matte and reflective surfaces. We ultimately generated 70,000 video sequences, each consisting of eight frames for a total of 560,000 frames. We collected real-world data using a Turtlebot3 and an Intel RealSense camera. We set up an enclosed, partially observable environment with several cups ("targets" for the search and rescue task) and other miscellaneous objects. Then we proceeded to record video as the Turtlebot autonomously explores the environment and additionally as we manually drove the robot. We ultimately recorded about 50,000 frames of video, which resulted in 6,250 sequences of eight frames. The combined dataset was then created by taking 6,250 sequences from each of the synthetic and real world datasets for a total of 12,500 sequences (100,000 frames).

Results

We found that the SOCS model trained on solely synthetic dataset performed as expected. When validating on synthetic data, the reconstruction of the video frames appear quite good, albeit some non-spherical shapes end up being reconstructed as spheres and there is some apparent loss in texture of the objects. The segmentations seem to be quite accurate, but are all roughly spherical like the reconstruction and seem to include object shadows as objects themselves. Despite these small irregularities, the model is able to successfully track the objects over the course of videos. When validating on real world data, as expected neither the reconstruction nor the object masks are very good. Interestingly however, the model seems to attempt to recreate colored objects such as cups with similarly colored objects from the synthetic dataset. Given this observation, we hypothesize that given 10 or 100 times the amount of synthetic training data, the model might be able to reconstruct several different types of real world objects.

We found that the SOCS model trained on solely real world data also performed as expected. When validating on real-world data, the image reconstructions were mostly successful, but there are several instances of certain objects not appearing in the reconstruction. However, this interestingly did not always appear to be a bad thing. As seen in the real world results in Figure 5.8, the reconstructed image omits the reflection of the pink cup that appears and as a result, the reflection is not considered when determining the object masks. Like with shadows in the synthetic dataset model, this model grouped object shadows and reflections on the floor along with the original object when producing the masks. Overall the segmentations seemed to accurately track objects but it struggled with the more complex backgrounds. When validating on real world data, both the reconstructed image and the

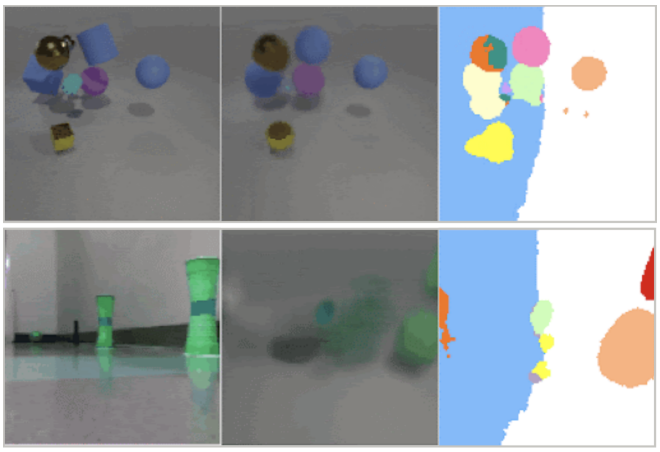


Figure 5.7: **Synthetic Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right)

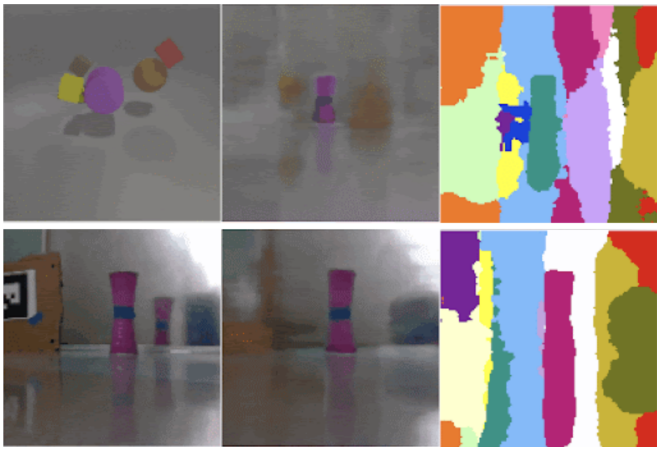


Figure 5.8: **Real World Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right)

segmentation were quite poor. This is much worse than that of real world validation on the synthetic dataset model, as the real world dataset saw much less movement in its image sequences and had more complex object shapes, and therefore cannot generalize as well to objects dissimilar to that in its training set.

We found that the SOCS model trained on the combined synthetic and real world dataset did not perform as expected. While the image reconstruction and segmentation of real world validation data outperformed that of the real world data model, this combined data model, performed significantly worse than the synthetic data model when it came to generating object masks on synthetic validation data. This validates the idea of supplementing real world data with synthetic data to improve real world validation, but not synthetic validation. This was especially surprising considering that the synthetic data reconstruction was overall

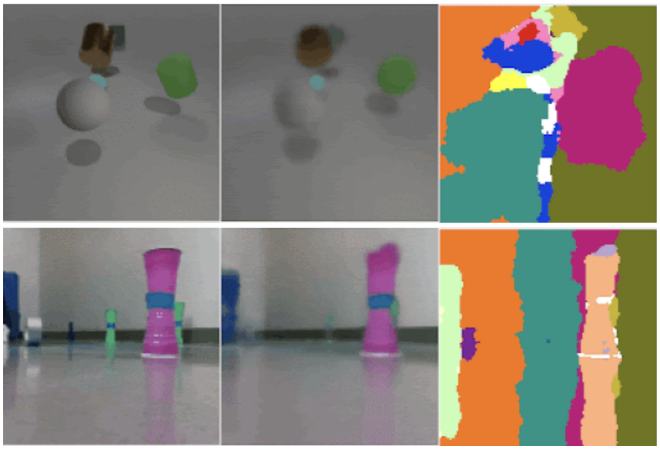


Figure 5.9: **Combined Data Model:** Synthetic (top) & Real World (bottom); Original Image (left), Reconstructed Image (middle), Object Masks (right)

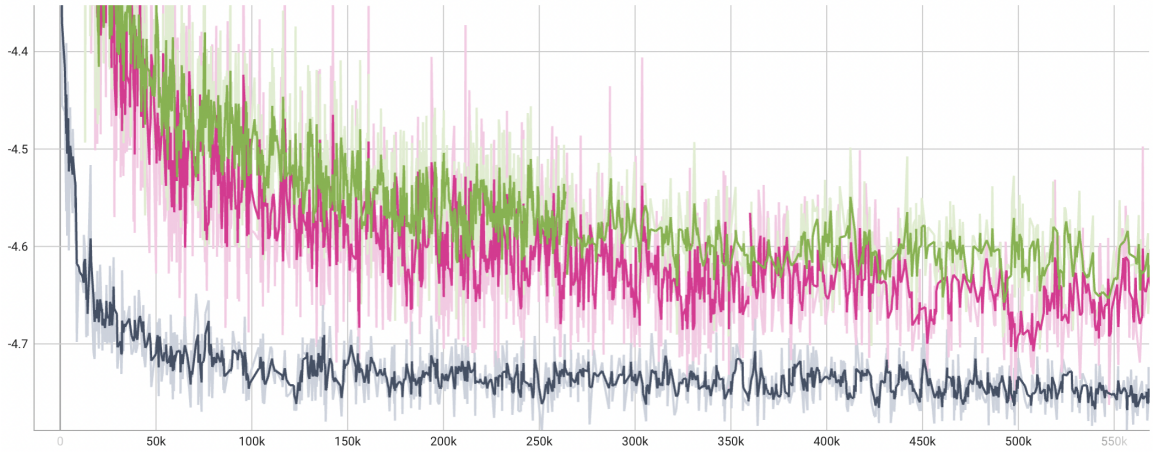


Figure 5.10: **Reconstruction Loss:** Real world (green), Synthetic (black), Combined (pink)

still fairly accurate and the object masks do not seem to match the reconstruction at all. While we do not have a clear answer to this, perhaps this was due to the more complex environment observed in the real world data, causing cost of generating poor masks for synthetic data to be lower by comparison. Additionally we note that the synthetic dataset contains videos of objects falling into a scene with a fixed camera position, while the real world dataset consists videos of objects fixed in a scene with a moving camera. This difference in videos may also have contributed to the combined data model's imbalance of efficacy for the synthetic and real world data.

Discussion

The results from our experimentation highlight the successful performance of the segmentation model trained on a combined dataset, comprising both real-world and synthetic data.

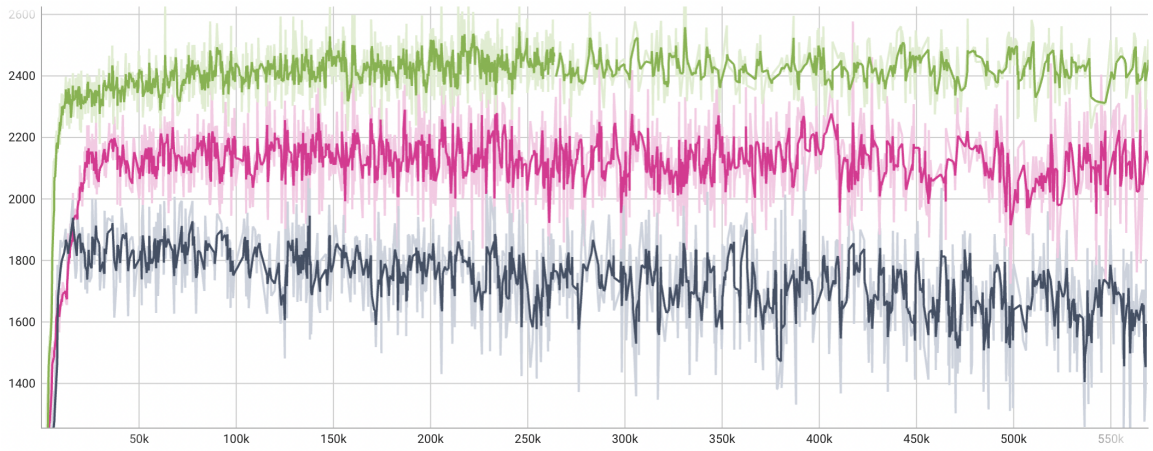


Figure 5.11: **Distribution (KL) Loss:** Real world (green), Synthetic (black), Combined (pink)

This demonstrates the effectiveness of supplementing real-world data with synthetic data without a loss in performance. Notably, synthetic data appears to generalize better to real-world data than vice versa. This can be seen as real world objects never before seen in the synthetic dataset are reconstructed via the general shapes from the synthetic data. While not perfect, this emergent behavior suggests that given enough training on highly diverse synthetic data, the model may be able to generalize well to new and unique objects not seen in training. We imagine that using a dataset with over 1,000,000 sequences trained over 10,000,000 steps could potentially produce impressive results. However, the segmentation model does exhibit some limitations. For example, it often over-segments the background of images, splitting walls and surfaces into numerous segmentations when in reality there was just one. In addition, it groups reflective regions with the area being reflected, which was particularly obvious in our real world gathered data set where the floor was semi-reflective and due to our camera placement much of it was in frame, causing many segmentations to be mirrored down into this region. Future work in this area involves addressing the identified limitations of the segmentation model through further refinement. This includes gathering more diverse real-world training data to improve model generalizability and extending the training duration to enhance performance.

Bibliography

[1] Jack Brady et al. *Provably Learning Object-Centric Representations*. 2023. arXiv: 2305.14229 [cs.LG]. URL: https://arxiv.org/abs/2305.14229.

[2] Rorry Brenner and Laurent Itti. *Perforated Backpropagation: A Neuroscience Inspired Extension to Artificial Neural Networks*. 2025. arXiv: 2501.18018 [cs.NE]. URL: https://arxiv.org/abs/2501.18018.

[3] Christopher P. Burgess et al. *MONet: Unsupervised Scene Decomposition and Representation*. 2019. arXiv: 1901.11390 [cs.CV]. URL: https://arxiv.org/abs/1901.11390.

[4] Christopher P. Burgess et al. *Understanding disentangling in β-VAE*. 2018. arXiv: 1804.03599 [stat.ML]. URL: https://arxiv.org/abs/1804.03599.

[5] Mathilde Caron et al. *Emerging Properties in Self-Supervised Vision Transformers*. 2021. arXiv: 2104.14294 [cs.CV]. URL: https://arxiv.org/abs/2104.14294.

[6] Cinzia Chiandetti and Giorgio Vallortigara. "Intuitive physical reasoning about occluded objects by inexperienced chicks". In: *Proceedings of the Royal Society B: Biological Sciences* 278.1718 (2011), pp. 2621–2627. DOI: 10.1098/rspb.2010.2381.

[7] Carl Doersch. *Tutorial on Variational Autoencoders*. 2021. arXiv: 1606.05908 [stat.ML]. URL: https://arxiv.org/abs/1606.05908.

[8] Carl Doersch. *Tutorial on Variational Autoencoders*. 2021. arXiv: 1606.05908 [stat.ML]. URL: https://arxiv.org/abs/1606.05908.

[9] Haiwei Dong et al. "Bringing Robots Home: The Rise of AI Robots in Consumer Electronics". In: *IEEE Consumer Electronics Magazine* 14.1 (Jan. 2025), pp. 4–6. ISSN: 2162-2256. DOI: 10.1109/mce.2024.3381573. URL: http://dx.doi.org/10.1109/MCE.2024.3381573.

[10] Jarek Duda. *Biology-inspired joint distribution neurons based on Hierarchical Correlation Reconstruction allowing for multidirectional neural networks*. 2024. arXiv: 2405.05097 [cs.LG]. URL: https://arxiv.org/abs/2405.05097.

[11] Gamaleldin F. Elsayed et al. *SAVi++: Towards End-to-End Object-Centric Learning from Real-World Videos*. 2022. arXiv: 2206.07764 [cs.CV]. URL: https://arxiv.org/abs/2206.07764.

[12] Saeed Ghorbani et al. "MoVi: A large multi-purpose human motion and video dataset". In: *PLOS ONE* 16.6 (June 2021). Ed. by Peter Andreas Federolf, e0253157. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0253157. URL: http://dx.doi.org/10.1371/journal.pone.0253157.

[13] Klaus Greff et al. *Multi-Object Representation Learning with Iterative Variational Inference*. 2020. arXiv: 1903.00450 [cs.LG]. URL: https://arxiv.org/abs/1903.00450.

[14] Irina Higgins et al. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=Sy2fzU9gl.

[15] Gabriel Horn. *Memory, Imprinting, and the Brain: An Inquiry into Mechanisms*. Oxford University Press, Dec. 1985. ISBN: 9780198521563. DOI: 10.1093/acprof:oso/9780198521563.001.0001. URL: https://doi.org/10.1093/acprof:oso/9780198521563.001.0001.

[16] M.H. Johnson, J.J. Bolhuis, and G. Horn. "Interaction between acquired preferences and developing predispositions during imprinting". In: *Animal Behaviour* 33.3 (1985), pp. 1000–1006. ISSN: 0003-3472. DOI: https://doi.org/10.1016/S0003-3472(85)80034-8. URL: https://www.sciencedirect.com/science/article/pii/S0003347285800348.

[17] Rishabh Kabra et al. *SIMONe: View-Invariant, Temporally-Abstracted Object Representations via Unsupervised Video Decomposition*. 2021. arXiv: 2106.03849 [cs.CV]. URL: https://arxiv.org/abs/2106.03849.

[18] Ioannis Kakogeorgiou et al. *SPOT: Self-Training with Patch-Order Permutation for Object-Centric Learning with Autoregressive Transformers*. 2024. arXiv: 2312.00648 [cs.CV]. URL: https://arxiv.org/abs/2312.00648.

[19] Ilyes Khemakhem et al. *Variational Autoencoders and Nonlinear ICA: A Unifying Framework*. 2020. arXiv: 1907.04809 [stat.ML]. URL: https://arxiv.org/abs/1907.04809.

[20] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. arXiv: 1312.6114 [stat.ML]. URL: https://arxiv.org/abs/1312.6114.

[21] Alexander Kirillov et al. *Segment Anything*. 2023. arXiv: 2304.02643 [cs.CV]. URL: https://arxiv.org/abs/2304.02643.

[22] Yoshimasa Kubo, Eric Chalmers, and Artur Luczak. *Biologically-inspired neuronal adaptation improves learning in neural networks*. 2022. arXiv: 2204.14008 [cs.NE]. URL: https://arxiv.org/abs/2204.14008.

[23] Francesco Locatello et al. *Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations*. 2019. arXiv: 1811.12359 [cs.LG]. URL: https://arxiv.org/abs/1811.12359.

[24] Francesco Locatello et al. *Object-Centric Learning with Slot Attention*. 2020. arXiv: 2006.15055 [cs.LG]. URL: https://arxiv.org/abs/2006.15055.

[25] Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023. URL: http://probml.github.io/book2.

[26] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020.

[27] Nikhila Ravi et al. *SAM 2: Segment Anything in Images and Videos*. 2024. arXiv: 2408.00714 [cs.CV]. URL: https://arxiv.org/abs/2408.00714.

[28] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV]. URL: https://arxiv.org/abs/1506.02640.

[29] Lucia Regolin and Giorgio Vallortigara. "Perception of partly occluded objects by young chicks". In: *Perception and Psychophysics* 57 (Oct. 1995), pp. 971–976. DOI: 10.3758/BF03205456.

[30] Timothy Schaumlöffel et al. "Caregiver Talk Shapes Toddler Vision: A Computational Study of Dyadic Play". In: *2023 IEEE International Conference on Development and Learning (ICDL)*. IEEE, Nov. 2023, pp. 67–72. DOI: 10.1109/icdl55364.2023.10364409. URL: http://dx.doi.org/10.1109/ICDL55364.2023.10364409.

[31] Elizabeth S. Spelke. "37Objects". In: *What Babies Know: Core Knowledge and Composition Volume 1*. Oxford University Press, Nov. 2022. ISBN: 9780190618247. DOI: 10.1093/oso/9780190618247.003.0002. eprint: https://academic.oup.com/book/0/chapter/371081642/chapter-pdf/57476054/oso-9780190618247-chapter-2.pdf. URL: https://doi.org/10.1093/oso/9780190618247.003.0002.

[32] Kaylene C. Stocking et al. *Linking vision and motion for self-supervised object-centric perception*. 2023. arXiv: 2307.07147 [cs.CV]. URL: https://arxiv.org/abs/2307.07147.

[33] Shagun Uppal et al. *SPIN: Simultaneous Perception, Interaction and Navigation*. 2024. arXiv: 2405.07991 [cs.RO]. URL: https://arxiv.org/abs/2405.07991.

[34] Stefan Sylvius Wagner and Stefan Harmeling. *Object-Aware DINO (Oh-A-Dino): Enhancing Self-Supervised Representations for Multi-Object Instance Retrieval*. 2025. arXiv: 2503.09867 [cs.CV]. URL: https://arxiv.org/abs/2503.09867.

[35] Stanisław Woźniak et al. "Deep learning incorporating biologically inspired neural dynamics and in-memory computing". In: *Nature Machine Intelligence* 2.6 (June 2020), pp. 325–336. ISSN: 2522-5839. DOI: 10.1038/s42256-020-0187-0. URL: http://dx.doi.org/10.1038/s42256-020-0187-0.

[36] Jinyang Yuan et al. *Unsupervised Object-Centric Learning from Multiple Unspecified Viewpoints*. 2024. arXiv: 2401.01922 [cs.CV]. URL: https://arxiv.org/abs/2401.01922.

[37] Tianyu Zheng, Liyuan Han, and Tielin Zhang. *Research Advances and New Paradigms for Biology-inspired Spiking Neural Networks*. 2024. arXiv: 2408.13996 [cs.NE]. URL: https://arxiv.org/abs/2408.13996.