MUFASA: A Multi-Layer Framework for Slot Attention

Supplementary Material

Sebastian Bock[*1,2] Leonie Schüßler[*1,2]
Krishnakant Singh[1] Simone Schaub-Meyer[1,3] Stefan Roth[1,2,3]
[1]TU Darmstadt [2]Zuse School ELIZA [3]hessian.AI [*]equal contribution

A. Implementation Details

In this section, we provide a more detailed overview of the training and implementation details for DINOSAUR-M and SPOT-M. The relevant hyperparameters for every dataset are summarized in Tab. 8. In general, we train for 1120 epochs on VOC, 100 epochs on COCO, and 95 epochs on MOVi-C. While this remains consistent between DINOSAUR-M and SPOT-M, the total number of epochs is split between the teacher and student model if self-training is employed. For the VOC and COCO datasets, the total number of epochs is evenly distributed between teacher and student, whereas for MOVi-C, the teacher is trained for 65 epochs following SPOT. We utilize the Adam optimizer [62] with $\beta_0 = 0.9$, $\beta_1 = 0.999$ and no weight decay. Learning rate scheduling is employed using a linear warm-up to 10 000 training steps and subsequent cosine annealing. For SPOT-M on COCO, we empirically found the student to perform better with an increased warm-up for 30 000 training steps. The learning rates are defined through a main value η_{main} and a lower boundary η_{low}. In SPOT-M, the learning rates of the students are set to match the teacher on VOC, whereas the peak value for COCO is reduced to $\eta_{\text{main}} = 3 \times 10^{-4}$ and the lower boundary for MOVi-C is set to $\eta_{\text{low}} = 1.5 \times 10^{-4}$. During self-training (*i.e.* SPOT-M), the knowledge distillation is incorporated into the reconstruction loss as the cross-entropy loss between aligned slot-attention masks of the teacher and student, weighed by some constant λ. We assign a greater weight to this loss as opposed to [26] with $\lambda = 0.01$, which we empirically found to work best for SPOT-M. All experiments are conducted on a single NVIDIA RTX A6000 GPU with 48 GB of memory.

A.1. MLP decoder

In our ablations, we investigate the use of an MLP decoder for MUFASA. Following previous work [26, 47], we implement it using a spatial broadcast decoder [53]. Here, each of the K slots in the fused slot representation is independently broadcast onto N image patches. These patches correspond to the flattened $H_{\text{emb}} \times W_{\text{emb}}$ grid of the encoder, requiring the addition of learned positional encodings to convey the notion of order within it. Then, an MLP processes the image patches of each slot independently, converting them into meaningful feature information. This MLP is shared across all slots. In addition to the features that are constructed for every token, the MLP predicts unnormalized alpha values, determining how much a slot contributes to each image patch. This results in an independent feature reconstruction from each slot. To obtain attention masks for the MLP decoder, we normalize these alpha values across the slot dimension using softmax. Finally, the complete reconstruction is generated through a weighted linear combination of the slot features for every image patch, using the alpha masks as weights.

Table 6. **Slot-attention and decoder metrics using an MLP decoder.** UOS results (in %, higher is better) of MUFASA and baselines on COCO using an MLP decoder. (↓) denotes the relative decrease in comparison to the transformer decoder. (*) indicates reproduced results. Decoder metrics degrade more substantially than slot-attention metrics when a weaker decoder is used.

Model (MLP Dec.)	mBO^c	mBO^i	mIoU
Slot-Attention Metrics			
DINOSAUR	31.4* (↓17.6%)	27.7* (↓8.6%)	26.4*
DINOSAUR-M (ours)	34.0 (↓20.2%)	27.1 (↓17.2%)	25.9 (↓15.1%)
SPOT	32.3* (↓25.1%)	27.6* (↓18.1%)	26.3* (↓17.3%)
SPOT-M (ours)	34.7 (↓23.7%)	30.2 (↓13.0%)	28.9 (↓11.1%)
Decoder Metrics			
DINOSAUR	30.5* (↓23.2%)	26.9* (↓14.9%)	25.7*
DINOSAUR-M (ours)	31.0 (↓27.9%)	27.1 (↓17.2%)	25.9 (↓15.1%)
SPOT	32.4 (↓26.9%)	28.4 (↓18.2%)	27.0 (↓17.4%)
SPOT-M (ours)	32.0 (↓25.4%)	27.5 (↓18.2%)	26.3 (↓16.9%)

A.2. Visualization of segmentation masks

To visualize segmentation masks (*e.g.*, Fig. 8), the fused attention mask $\mathcal{A}^{\text{Slot}}_{\text{fused}}$ is first reshaped to a spatial grid and upsampled to the image size with bilinear interpolation. Each pixel is assigned to the slot that attended most to it, where each slot is represented with a unique color. The resulting segmentation mask is then overlayed onto the image.

A.3. Training stability for MAE and DINOv2

Naively implementing MAE [19] and DINOv2 [43] as feature encoders leads to training collapse if no self-training is employed (*e.g.*, in DINOSAUR and DINOSAUR-M). We mitigate this issue by using trainable initial slots instead of random initialization along with bi-level optimization (BO-QSA [61]). This strategy was originally introduced by [26] to stabilize training during image encoder fine-tuning.

B. Discussion on Test-Time Ensembling

In their work, SPOT [26] employ test-time ensembling within the decoder by averaging predictions over nine decoder passes, one for each patch-order permutation. This yields marginal improvements in their reported results at the cost of increased inference time. Given the minimal gains, we consider the additional inference cost unwarranted and therefore do not apply test-time ensembling. To ensure a fair comparison, we do not utilize test-time ensembling in either MUFASA or SPOT in our experiments.

C. Discussion on Slot *vs.* Decoder Masks

Empirically, integrating MUFASA yields stronger results when evaluating on segmentation masks derived from the slot attention module. In contrast, in previous models [26, 47], the decoder-produced masks were found more suitable for segmentation tasks. However, metrics computed based on decoder segmentations (*decoder metrics*) are sensitive to the specific decoder architecture deployed. As shown in Tab. 6, when a weaker MLP decoder is used, the decoder metrics degrade substantially more than metrics computed on slot-attention segmentations (*slot metrics*). This highlights the decoder capacity as a confounding factor. As a consequence, we observe that the decoder metrics do not reliably reflect the quality of the slot-object binding itself. By integrating MUFASA, we reduce this dependence and thereby reliably improve slot representations for UOS. Despite these limitations of decoder metrics, we report the maximum over both slot and decoder metrics in the main paper in accordance with prior work to enable a fair comparison.

D. Comparison to Additional SOTA Models

We provide further comparisons of MUFASA against additional state-of-the-art models in the task of unsupervised object segmentation in Tab. 7. Notably, [63] relies on additional training signals beyond the reconstruction loss and [59] leverages diffusion models pre-trained on caption-annotated data, while MUFASA does not require any of these. Yet, our method outperforms them over multiple datasets and metrics.

E. Visualization of ViT Layers

The feature representations at different layers of the DINO ViT [4] serve as the foundation for MUFASA's multi-layer slot attention. In this section, we analyze how their structural properties and encoding characteristics evolve across layers. To do so, we conduct a principal component analysis (PCA), visualized for all layers that were investigated in our ablations on layer choice. Following [1], we project the high-dimensional feature representations to three principal components, which are then mapped to RGB channels for visualization. In Fig. 7 (a), we observe a grid-like structure at layer 3, devoid of any object-specific shape. This suggests that such early layers are unsuitable as input to slot attention, as they lack object-centric information, visually confirming our ablations in Fig. 6. In the intermediate layers (Fig. 7 (b) – (d)), the object structure gradually emerges, and background textures become distinguishable. These layers provide information about the object localization. At last, the latest layers (Fig. 7 (e) – (g)) exhibit semantic information, such as the stripe in the fur at the penguin's head (first row) or the small items on the table (third row). At this stage, the characteristics are now semantically meaningful features to form object-centric representations.

Table 7. **Comparison to additional SOTA methods in UOS.** We compare our approach with current SOTA OCL methods on PASCAL VOC, COCO, and MOVi-C. The metrics (in %, higher is better) are computed from slot-attention and decoder masks; the maximum across both is reported. For MUFASA, we report mean over three seeds. "–" indicates that results were not reported in the respective paper. We evaluate [9] on the same resolution as MUFASA. Best results are in **bold**, the 2nd-best underlined.

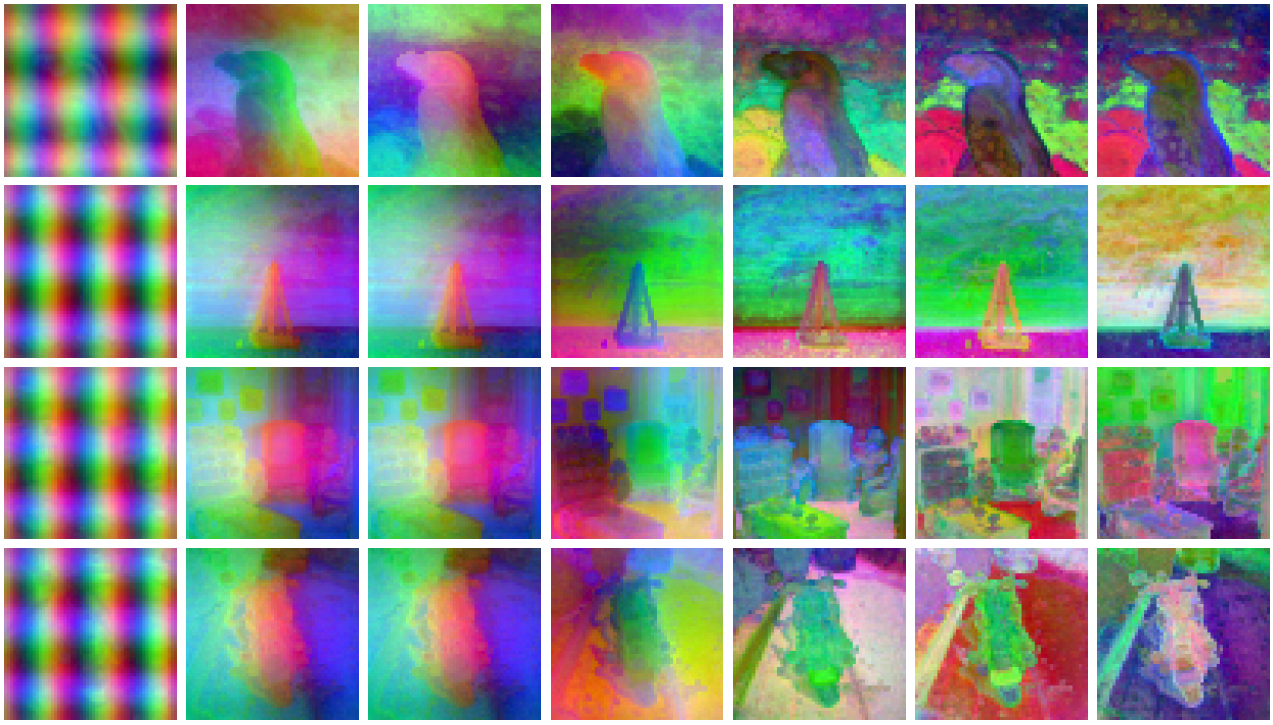
Model	Pascal VOC			COCO			MOVi-C	
	mBO^c	mBO^i	mIoU	mBO^c	mBO^i	mIoU	mBO^i	mIoU
SlotAdapt [59]	51.9	**51.5**	–	39.2	35.1	–	–	–
Multi-Query SA [45]	–	39.7	39.4	–	–	–	–	–
FT-DINOSAUR [9]	–	–	–	–	32.0	–	–	–
SPOT-FS-RC [63]	56.5	49.3	–	45.3	**35.7**	–	49.0	47.8
DINOSAUR-M *(ours)*	57.6	49.2	47.2	43.0	32.7	30.5	**49.2**	**48.3**
SPOT-M *(ours)*	**59.8**	51.3	**49.4**	**45.5**	34.8	**32.5**	**49.2**	48.2

F. Additional Visual Examples

We provide further segmentation masks for DINOSAUR-M and SPOT-M compared against their respective baselines, as well as the ground truths for PASCAL VOC in Fig. 8, COCO in Fig. 9, and MOVi-C in Fig. 10. They provide an extended overview over different settings and motives, such as close-up objects, landscapes, or a composition of multiple small objects to emphasize MUFASA's abilities to decompose various kinds of scenes into meaningful entities.

Table 8. **Hyperparameters of MUFASA on the VOC, COCO, and MOVi-C datasets.** Learning rates and warmup epochs may differ between teacher and student if self-training is employed; students utilizing a different learning rate and warmup schedule than their teacher are denoted with †. "–/–" denotes identical hyperparameters across datasets.

Dataset →		**PASCAL VOC**	**COCO**	**MOVI-C**
Epochs	Teacher	560	50	65
	Student	560	50	30
	No self-training	1120	100	95
	Warmup	60	5 († : 15)	7
Low LR η_{low}		4×10^{-7}	4×10^{-7}	4×10^{-5} († : 1.5×10^{-4})
Main LR η_{main}		4×10^{-4}	4×10^{-4} († : 3×10^{-4})	2×10^{-4}
Batch size		64	–/–	–/–
Optimizer		Adam ($\beta_0 = 0.9, \beta_1 = 0.999$)	–/–	–/–
Distillation λ		0.01	–/–	–/–
Encoder	Architecture	ViT-B [60]	–/–	–/–
	Patch size	16×16	–/–	–/–
	Feature dimension d_{emb}	768	–/–	–/–
	Weights	DINO [5]	–/–	–/–
ViT decoder	Number of layers	4	–/–	–/–
	Heads	6	–/–	–/–
MLP decoder	Number of layers	4	–/–	–/–
	Hidden size	2048	–/–	–/–
Slot fusion	Strategy	M-Fusion	–/–	–/–
	Layer selection $\mathcal{J}$	9, 10, 11, 12	–/–	–/–
	MLP hidden layers	1	–/–	–/–
	Activation	GELU [20]	–/–	–/–
	MLP hidden size	768	–/–	–/–
Slot attention	Iterations	3	–/–	–/–
	MLP hidden size	1024	–/–	–/–
	Slot dimension d_{slot}	256	–/–	–/–
	Number of slots	6	7	11
Training images		10 582	118 287	87 633
Evaluation images		1449	5000	6000
Crop resolution		224×224	–/–	–/–
Evaluation resolution		320×320	320×320	128×128
Resize strategy		Minor axis to 224	Minor axis to 224	–
Crop strategy		Random	Center	Full
Augmentations		Random flip ($p = 0.5$)	Random flip ($p = 0.5$)	–



(a) Layer 3 (b) Layer 6 (c) Layer 8 (d) Layer 9 (e) Layer 10 (f) Layer 11 (g) Layer 12

Figure 7. **PCA of DINO ViT features**. Layerwise visualization of the DINO ViT features at different layers via principal component analysis (PCA) for four different images. The first three principal components yield red, green, and blue channels. Semantically meaningful information is absent in earlier layers and begins to emerge in intermediate ones, while becoming increasingly rich in deeper layers.

Figure 8. **PASCAL VOC segmentation masks**. Images taken from PASCAL VOC, segmented by SPOT-M *(top row)*, SPOT *(second row)*, DINOSAUR-M *(third row)*, and DINOSAUR *(fourth row)* compared against the ground truth *(bottom row)*. For SPOT and DINOSAUR, segmentation masks derived from the decoder are shown, while for their respective MUFASA variant, segmentation masks from the slot attention module are depicted.

Figure 9. **COCO segmentation masks.** Images taken from COCO, segmented by SPOT-M *(top row)*, SPOT *(second row)*, DINOSAUR-M *(third row)*, and DINOSAUR *(fourth row)* compared against the ground truth *(bottom row)*. For SPOT and DINOSAUR, segmentation masks derived from the decoder are shown, while for their respective MUFASA variant, segmentation masks from the slot attention module are depicted.

Figure 10. **MOVi-C segmentation masks**. Images taken from MOVi-C, segmented by SPOT-M *(top row)*, SPOT *(second row)*, DINOSAUR-M *(third row)*, and DINOSAUR *(fourth row)* compared against the ground truth *(bottom row)*. For SPOT and DINOSAUR, segmentation masks derived from the decoder are shown, while for their respective MUFASA variant, segmentation masks from the slot attention module are depicted.

References

[59] Adil Kaan Akan and Yücel Yemez. Slot-guided adaptation of pre-trained diffusion models for object-centric learning and compositional generation. In *ICLR*, 2025. ii

[60] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. iii

[61] Baoxiong Jia, Yu Liu, and Siyuan Huang. Improving object-centric learning with query optimization. In *ICLR*, 2022. i

[62] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. i

[63] Pinzhuo Tian, Shengjie Yang, Hang Yu, and Alex Kot. Pay attention to the foreground in object-centric learning. In *CVPR*, pages 30281–30290, 2025. ii