

# Supplementary Materials for FORLA: Federated Object-Centric Representation Learning with Slot Attention

## A Extended Related work

To provide additional context for our approach, we expand on key areas of related work in this section.

### A.1 Federated Learning (FL)

FL enables collaborative training across decentralized datasets while preserving privacy [42]. Prior work addresses challenges like non-IID data [31] and communication efficiency [29], but most focus on classification or segmentation tasks without emphasis on learning representations. Early work on unsupervised representation learning for FL showed that naive FedAvg misaligns local feature spaces under non-IID data. Zhang et al. [69] addressed this with FURL/FedCA, which maintains a shared memory bank and a contrastive alignment module. Michieli et al. [44] proposed Prototype-Guided FL, where clients periodically exchange class-agnostic prototypes to pull their feature spaces into a common geometry. Tan et al. [57] also leverage prototypes in FedPCL in a contrastive learning scheme. This idea is later extended by clustering-based methods such as ORCHESTRA [40] and FedRLC [43]. Knowledge distillation has also emerged as a powerful alignment tool. FedX [19] employs two-sided knowledge distillation with a contrastive objective, allowing clients to share rich representation information without exchanging data while FedUKD [63] further reduces domain bias via bilateral distillation in an unsupervised FL framework. Despite this progress, object-centric representation learning – which aims to decompose scenes into entity-specific features – has not yet been explored in federated settings, and no prior work explicitly combines these paradigms to date, despite its potential to leverage domain-specific data.

### A.2 Object-Centric Learning

Slot Attention (SA) [38] pioneered the use of iterative attention to decompose scenes into object slots. Subsequent work improved its scalability [56] and incorporated transformers [51], but all assume centralized training on mixed data. ORL [64] demonstrated that exploiting object-level correspondences can improve unsupervised learning on multi-object images. Following the introduction of DINOSAUR [52], SA began to be applied to real-world scenarios using foundation models. A growing body of work has since built on this approach to improve performance—either by augmenting foundation features [39], or adding modality [67, 2], or by fine-tuning the foundation models themselves [12]. Recent techniques like contrastive learning objective [41], patch permutation augmentation [26] and using language supervision signals [10] have been demonstrated to improve the scene decomposition performance. However, no prior work in federated learning has combined object-centric models with distributed training across heterogeneous datasets. We argue that centralized training of object-centric models struggles with concept entanglement, a challenge that federated learning inherently mitigates by preserving local data coherence.

### A.3 Concept Entanglement in Multi-Domain Learning

Concept entanglement represents a significant challenge in the field of multi-domain learning, particularly when training machine learning models on datasets that contain overlapping visual semantics. Early work tackles this problem by explicitly splitting features into shared and private subspaces, as in Domain Separation Networks [4]. Subsequent approaches refine the idea: [37] propose a Unified Feature Disentangler that learns domain-invariant codes via adversarial training and [24] maximizes interaction information to carve representations into domain-specific and domain-agnostic parts. Latent-domain methods reduce entanglement without domain labels—for example using a sparse adapter for Latent Domain Learning [9]. [32] show that distilling representations from multiple task- and domain-specific networks using alignment with small domain-specific adapters can lead to efficient universal representations. Despite these advances, object-centric disentanglement has yet to be explored in any centralized or federated setting, leaving a gap that our work addresses.

#### A.4 Adaptation Layers and Feature Harmonization

Adaptation layers are widely used in transfer learning [49] and multi-task learning [59] to align feature spaces. In federated learning (FL), [8] employs adapters for personalized federated learning. Didolkar et al. [12] investigate the transferability of foundation models in object-centric representation learning and introduce a fine-tuning framework (FT-DINOSAUR) for the task of object discovery, achieving strong unsupervised transfer. Crucially, their results suggest that object-centric learning can be improved by training on one domain and fine-tuning on another, but that improvements cannot be achieved merely by scaling up data. Our work goes further by demonstrating how federated learning can scale to heterogeneous datasets through cross-client feature harmonization, ensuring global model stability despite diverse local inputs. One technique for adapting foundation models is the application of lightweight feature adapters [16, 70], which introduce trainable components into the network while keeping most of the model frozen. Other methods include fine-tuning only the bias parameters of pretrained models [6, 68], and learning low-rank adaptations [22]. We also experiment with adaptation modules correlated to a masking approach based on frozen foundation models [60]. An advantage of this approach is that multiple foundation models can be used in parallel, without any query on which one to fine-tune.

#### A.5 Knowledge Distillation

Knowledge distillation has been used to perform federated learning, for instance, when a global student calculated with FedAvg learns for a local teacher [71, 61]. In large-scale self-supervised learning, such as DINO [7], self-distillation enables the model to implicitly ensemble knowledge over time, with the teacher network being updated through an exponential moving average (EMA) of the student, akin to Polyak-Ruppert averaging [45]. Inspired by this approach, our framework updates certain modules of the teacher by averaging them with the student, but without relying on a conventional knowledge distillation (KD) loss. This is because slot prediction is inherently permutation-invariant [38], making direct teacher-to-student supervision inapplicable. Instead, we synchronize the teacher and student Slot Attention modules through periodic averaging—effectively a local FedAvg operation between the two branches. This strategy is also related to co-distillation [1], where student-teacher models distill knowledge from one another to efficiently ensemble knowledge from large-scale, distributed data. Unlike co-distillation, where student and teacher models are typically identical, we share only certain modules between the two. Moreover, our framework operates in a fully unsupervised setting.

### B Feature adapters

We experimented with three types of adapter, *MLP adapter* [50], *mixture-of-experts (MoE)* [53, 15], and *Attention-based Feature Modulation (AFM)* [23]. Extended details on those three adapters are listed below:

- *MLP adapter* [50]: a simple MLP that maps  $\mathbf{F}$  to  $d$  channels. This treats the concatenated features as one vector per location and learns a single set of weights to combine and reduce them.
- *Mixture-of-experts* [53, 15]: a set of  $E$  expert projection layers  $\{g^{(e)}\}_{e=1}^E$  (each a learned linear projection) corresponding to each foundation. An auxiliary gating network produces spatially varying weights  $\alpha^{(e)}(u, v)$  for each foundation model at each location  $(u, v)$ . The adapted feature is then  $\mathbf{F}_{\text{adapt}}(u, v) = \sum_{e=1}^E \alpha^{(e)}(u, v) g^{(e)}(\mathbf{F}(u, v))$ . This allows the model to dynamically select different combinations of foundation features in different images.
- *Attention-based Feature Modulation* [23]: instead of combining all features, the adapter learns to *suppress or amplify* channels from  $\mathbf{F}$ . We implement this as a set of learnable mask parameters  $\{m_c\}$  for  $c = 1, \dots, C_{\text{tot}}$  applied to  $\mathbf{F}$ . The mask effectively select which foundation model features to pass through for each location. Following the channel masking, a linear projection is applied to obtain  $d$  channel feature  $\mathbf{F}_{\text{adapt}}$ .



## C Slot Attention Models

Slot Attention [38] is an architecture for learning object-centric representations. It maps a set of  $N$  input feature vectors (e.g., image pixels or CNN features) to  $K$  slot vectors, each aiming to represent an object in the scene. The  $K$  slots are initialized (e.g., randomly) and iteratively updated by an attention mechanism that binds slots to parts of the input. At each iteration  $t = 1 \dots T$ , attention weights  $W_{ik}$  are computed between each input feature  $x_i$  and each slot  $s_k$  (for  $i = 1, \dots, N$  and  $k = 1, \dots, K$ ). For example, using dot-product attention with learned projections  $W^Q, W^K$  for queries/keys, one can write:

$$W_{ik} = \frac{\exp\left(\frac{(x_i W^Q)(s_k W^K)^\top}{\sqrt{d}}\right)}{\sum_{k'=1}^K \exp\left(\frac{(x_i W^Q)(s_{k'} W^K)^\top}{\sqrt{d}}\right)}, \quad (6)$$

which is a normalized attention weight indicating how much slot  $k$  attends to input  $i$ . Using these weights, each slot is updated by aggregating the inputs:

$$s_k \leftarrow \text{GRU}\left(s_k, \sum_{i=1}^N W_{ik} (x_i W^V)\right), \quad (7)$$

where  $W^V$  is a learned value projection and GRU is a gating recurrent unit that combines the current slot value with the weighted input. This process is repeated  $T$  times, and includes an MLP-based refinement per iteration [38]. The result is  $K$  output slot vectors  $\{s_k\}_{k=1}^K$  that are exchangeable (permutation-invariant) and ideally each slot specializes to one object or background component.

In unsupervised object discovery, Slot Attention is typically trained by reconstructing the original image from the slots. A decoder (e.g., a CNN, spatial broadcast decoder, or transformer decoder) uses the slots to output  $K$  component reconstructions and masks, which are combined to match the input image. The model thus learns to partition the scene into objects. However, such purely unsupervised training can struggle on complex real-world images. Thus, recent work of applying slot attention to such images is all based on foundational models [52, 39, 67, 11], and the reconstruction target is feature map instead of image.

## D Additional Details on FORLA

### D.1 Two-stage Training and Local-FedAvg

We extend the FORLA framework details in Algorithm 1. The FedAvg between clients is performed every 100 iterations, while the FedAvg between teacher and student is performed every 1000 iterations. We use Adam optimizer with learning rate of  $4 \times 10^{-4}$ , and weight decay of  $4 \times 10^{-4}$ . For all data the batch size is 16.

We set an empirical switching criteria as 90K iterations, as this number allows the reconstruction loss of slot attention models on clients’s student branch to converge to plateau. We also found that training for a larger number of iterations before switching from EMA (larger than 90K) can lead to better initial convergence of the student branch, but in the long term this advantage is offset by the second stage training of the student, making the additional time spent on initial training obsolete. In future, the empirical threshold can be replaced by a dynamic threshold based on tracking of student loss.

### D.2 Foundation Model Specifications

In computer vision community, a significant amount of time and resources have been used to train foundation models from large image datasets of everyday scenes and objects (*Natural vision domain*). We provide specifications of four vision foundation models used in FORLA. These four models were selected because of their complementary capabilities in semantic understanding, segmentation, reconstruction, and multimodal alignment. Table 14 summarizes key technical specifications, while we elaborate on their architectural and functional characteristics below:

- DINO (self-Distillation with NO labels) [7]:

- *Architecture*: Vision Transformer (ViT-B/16, ViT-B/14 for DINO-v2 ) with 12 layers, 768D embeddings
- *Pretraining Objective*: Self-supervised distillation using image augmentations without labels
- *Training Data*: ImageNet-1k (1.28M images)
- *Key Strengths*: Captures high-level semantic relationships through global self-attention; produces spatially consistent feature maps ideal for object discovery
- **SAM (Segment Anything Model) [28]:**
  - *Architecture*: ViT-B/16 image encoder with mask decoder
  - *Training Objective*: Supervised promptable segmentation on 1B+ masks
  - *Training Data*: SA-1B dataset (11M licensed images)
  - *Key Strengths*: Specialized in boundary-aware feature extraction.
- **MAE (Masked Autoencoder) [20]:**
  - *Architecture*: Asymmetric ViT-B/16 with 75% patch masking
  - *Pretraining Objective*: Self-supervised, pixel reconstruction of masked image regions
  - *Training Data*: ImageNet-1k (1.28M images)
  - *Key Strengths*: Excels at texture/structure recovery; provides complementary low-level features to DINO’s semantics.
- **CLIP (Contrastive Language-Image Pretraining) [48]:**
  - *Architecture*: ViT-B/16 image encoder (text branch disabled)
  - *Pretraining Objective*: Contrastive alignment of 400M image-text pairs, language is used as weak-supervision signal
  - *Training Data*: Web-crawled multimodal corpus
  - *Key Strengths*: Cross-modal concept grounding; robust to distribution shifts

Our experiments use frozen foundation models without fine-tuning: ViT-B/16 variants of DINO, MAE, and CLIP with  $14 \times 14$  patch size and  $224 \times 224$  input resolution. The CLIP text branch is excluded. For SAM, we use only the image encoder, downsampling positional embeddings to align spatial resolution with other models; the decoder is omitted as our focus is solely on representation learning. The specific hyperparameters of other modules are included in Table 8. We follow DINOSAUR [52] in setting the slot count for COCO and PASCAL to 6 slots, and for other datasets we use 7 slots. Both teacher and student branches are trained with the same number of slots. Using a teacher and a student with a different number of slots would be interesting to explore in future work.

## E Dataset details

Here we extend details of all datasets used in this research:

*Abdominal dataset* is a public dataset from animal, phantom, and simulator abdominal surgeries [72]. We utilize 739260 frames for training, and 3000 frames with segmentation masks evaluation.

*Cholec dataset* [58] consists of 80 laparoscopic cholecystectomy videos. Following [35], we use 15,000 frames for training. 8,000 frames with segmentation annotations for evaluation.

*Thoracic dataset* includes data from 40 robot-assisted right upper lobectomies (RULs) for lung cancer, performed at Toronto General Hospital between 2014 and 2023. We use a total of 51,900 images for training and 800 manually annotated frames for evaluation.<sup>2</sup>

*COCO* (Common Objects in Context) [36] is a widely used benchmark for object detection, segmentation, and image captioning, consisting of 80 object categories. We use the 2017 split, with 118,000 images for training and 5,000 for validation.

*PASCAL VOC 2012* [13] provides 11,530 images with segmentation masks for 20 object categories. Following standard protocol, we use 10,582 images for training and 1,449 for validation.

<sup>2</sup>An open-access version of this proprietary dataset is integrated as part of the federated learning benchmark in this work. See: <https://github.com/PCASOlab/FORLA>

Table 8: Hyperparameters of different network components.

	Name	Type	Model file size	Feature dim
Foundation models	DINO	ViT-B/16	346 MB	768
	SAM	ViT-B/16	375 MB	256
	MAE	ViT-B/16	327 MB	768
	CLIP(image encoder)	ViT-B/16	437 MB	768
	Type	Input dim	Model file size	Feature dim
Adapters	MLP	2560	56MB	256
	MoE	2560	107MB	256
	AFM	2560	158MB	256
	Slot num	Slot dim	Model file size	Iteration
Slot attention	6 / 7	256	2.3 MB	3
	MLP hidden dim	Input dim	Model file size	Output dim
Teacher decoder	1024	256	13M	262 / 263
Student decoder	1024	256	23M	2566 / 2567

*YTVIS* (YouTube-VIS) [66] is a benchmark for video instance segmentation, containing 8,858 videos spanning 40 object categories, with pixel-level masks across frames. We train on the 2,985-video training split from the 2021 version (78,810 frames) and evaluate on 4,210 validation frames.

*YTOBJ* (YouTube-Objects) [47] consists of 126 YouTube videos across 10 object categories, annotated with sparse bounding boxes for object tracking. We extract 388,050 frames from 100 videos for training and evaluate on 9,000 frames from 26 held-out videos.

## F Additional Implementation Details

### F.1 Additional Details on Metrics

We evaluate our approach based on the quality of the slot attention masks using four primary metrics: Foreground Adjusted Rand Index (FG-ARI) [17], Mean Best Overlap (mBO) [46], Mean Best Hausdorff Distance (mBHD), and CorLoc [3]. FG-ARI, widely adopted in object-centric research, measures the similarity between predicted object masks and ground truth segmentation, specifically focusing on foreground regions. mBO evaluates the overlap between predicted and ground truth masks using intersection-over-union (IoU). It computes the average IoU after Hungarian matching between each ground truth and its best-matching predicted mask. While FG-ARI emphasizes segmentation quality regardless the permutation, mBO offers a broader assessment by channel matched IoU. To further assess spatial precision, we compute mBHD, which captures boundary-level accuracy and is particularly important for clinical applications. CorLoc measures localization accuracy by counting predicted object instances whose bounding boxes achieve  $\text{IoU} > 0.5$  with a ground truth object.

### F.2 Additional Details on Experiment setup

Our experiments are conducted under three training regimes: (i) individual training, where each dataset is trained independently; (ii) centralized mixed training, where data from multiple datasets (surgical or natural) are pooled; and (iii) federated training, where each dataset is treated as a separate client without data sharing.

As each dataset can be considered as residing on a single client, we have a maximum of seven clients corresponding to seven distinct sub-domains. All images are resized as 224×224 for input (14×14 patches after ViT-B/16). For SAM, only the image encoder is used; positional embeddings are down-sampled to match the spatial resolution of the other models, and the decoder is omitted since representation learning is our focus. All adapter variants reduce the input feature dimensionality to 256 and use a slot embedding size of 256, following common practice in recent slot attention models [52, 62]. Each federated client is trained for a minimum of 100 epochs, with early stopping triggered if the student’s reconstruction loss does not improve over 30 consecutive epochs. This stopping criterion was consistently met, and no late convergence was observed. The teacher’s reconstruction loss is not used for early stopping, as it tracks a dynamic and more fluctuating target. Individual and

Table 9: Performance on surgical data using single foundation model’s raw features for slot attention.

Model	Abdominal				Cholec				Thoracic			
	mBO	mHD↓	FG-ARI	CorLoc	mBO	mHD↓	FG-ARI	CorLoc	mBO	mHD↓	FG-ARI	CorLoc
DINO	<b>47.33</b>	<b>51.497</b>	<b>57.87</b>	52.38	<b>28.7</b>	<b>62.13</b>	<b>38.9</b>	30.4	30.96	112.324	19.3	30.45
SAM	47.0	55.84	53.7	<b>56.2</b>	25.75	79.43	32.62	<b>31.81</b>	<b>50.28</b>	<b>71.94</b>	<b>41.14</b>	<b>54.88</b>
MAE	35.0	71.17	42.6	34.0	14.37	107.21	19.23	18.7	35.08	100.57	36.35	26.53
CLIP	23.0	75.14	28.9	19.8	10.9	97.26	13.41	10.39	14.99	117.26	8.71	11.75

centralized training baselines are run for 130 epochs. Our experiments were conducted using four NVIDIA RTX 6000 GPUs, with some GPUs assigned multiple clients. Each running client consumes approximately 6 GB of GPU memory after feature caching. Feature caching accelerates training by a factor of 10–20 for each client. A complete run of federated learning across all mixed sub-domains takes approximately 12 hours.

Centralized training on 1.4 million data points (frames) across 7 datasets with 130 epochs and batch size 16 takes 21.6 hours which is 1.8 times slower than FL. As the data can not be distributed across too many clients since unsupervised object centric learning needs a certain amount of data to discover meaningful concept, it would indeed be interesting to explore in more detail how one can achieve an optimal speedup/performance ratio in a FL training regime.

For the Natural image datasets we used optimal slot counts reported in the literature [11, 52]. For the surgical datasets we tuned the slot count to achieve a best score. We’ve also used some evidence from training SA models with both types of data [34, 33]. Choosing an optimal slot number is indeed important for slot attention algorithms. Our method is also fully compatible with SA approaches that use an adaptive slot count, but we decided to use more traditional approaches in this work for easier evaluation.

## G Additional results

In this section, we extend experiments on using single or different combination of frozen/dapted foundation models, distillation dynamics on different dataset using different adapters, evaluation the performance of teacher branch (student is reported in the main paper), Comparison on Zero-shot transfer from natural to surgical domain, FORLA inference on videos when compared to DINO and SAM backbone, and more qualitative demonstration of feature adaptation and slot attention on different domains.

### G.1 Comprehensive Foundation Model Benchmarking

Tables 9 and 10 provide a detailed benchmark of four foundation models across surgical and natural domains. In this experiment we use only frozen features and no adapter or fine-tuning is applied. Each model is train on a single dataset individually. Three key insights emerge: 1) **Specialization-utility tradeoff**: DINO’s self-supervised features excel on Cholec instruments (28.7 mBO) and COCO (23.96 mBO), indicating strong general object semantics. SAM with its segmentation-focused pretrained features achieves 50.28 mBO on Thoracic data (Table 9), outperforming DINO’s 30.96 mBO, demonstrating out of domain generalization advantages. Reconstruction-based MAE underperforms on surgical data (-17.2 mBO vs. SAM) but shows unexpected competence on YTVIS (20.2 mBO) which could due to YTVIS requiring less high-level semantic features. 2) **Modality and domain mismatch**: CLIP’s text-image alignment provides limited value for surgical domains (14.99 mBO Thoracic), suggesting medical imaging diverges from its web-scale pretraining. MAE trained on natural domain also transfers poorly to surgical scenes. 3) **Complementary Strengths**: SAM achieves highest CorLoc (56.2) while DINO leads in FG-ARI (57.87) on Abdominal data, and no single model dominates all metric on all data.

This analysis confirms that foundation models exhibit specialized capabilities aligned with their pre-training objectives and training data. FORLA’s feature integration strategy (Eq. 1) allows synergistic combination of these complementary representations.

Table 10: Performance on natural data using single foundation model’s raw features for slot attention.

Model	COCO				PASCAL				YTVIS				YTOBJ	
	mBO	mHD ↓	FG-ARI	CorLoc	mBO	mHD ↓	FG-ARI	CorLoc	mBO	mHD ↓	FG-ARI	CorLoc	mBO	CorLoc
DINO	<b>23.96</b>	<b>81.929</b>	<b>29.2</b>	20.13	34.79	78.853	35.17	52.31	<b>33.09</b>	<b>68.238</b>	<b>36.64</b>	<b>54.02</b>	42.77	54.78
SAM	22.61	93.192	25.02	<b>20.5</b>	<b>36.98</b>	<b>73.992</b>	<b>35.59</b>	<b>56.04</b>	32.62	68.704	35.89	53.66	<b>42.86</b>	<b>54.95</b>
MAE	17.08	101.592	18.83	8.59	25.95	91.41	22.93	31.87	20.2	87.701	20.81	15.94	33.86	29.91
CLIP	12.73	109.433	14.14	3.46	14.52	109.054	9.82	27.47	14.37	107.217	19.23	23.15	22.27	9.02

Table 11: Adaptation of features from a single foundation model on Abdominal, YTVIS and YTOBJ.

Adapter	Model	Abdominal				YTVIS				YTOBJ	
		mBO	mHD↓	FG-ARI	CorLoc	mBO	mHD↓	FG-ARI	CorLoc	mBO	CorLoc
MLP	Concat	<u>51.85</u>	<u>43.025</u>	<u>59.04</u>	<u>69.25</u>	35.16	67.22	38.25	58.08	<u>48.62</u>	<u>60.31</u>
	DINO	<b>45.35</b>	<b>53.653</b>	<b>50.55</b>	<b>63.6</b>	<b>36.42</b>	66.584	<b>40.91</b>	62.13	<b>46.10</b>	<b>60.02</b>
	SAM	41.78	69.091	46.08	57.72	35.93	<b>64.650</b>	39.61	<b>63.16</b>	37.72	51.92
	MAE	40.84	59.286	47.28	52.3	21.94	83.329	22.97	20.67	33.52	26.83
AFM	Concat	<u>50.34</u>	<u>44.398</u>	<u>59.04</u>	<u>70.73</u>	33.94	66.637	37.54	54.59	<u>48.98</u>	<u>65.11</u>
	DINO	41.99	63.738	47.46	56.73	<b>36.52</b>	65.752	<b>41.01</b>	61.18	<b>48.17</b>	<b>62.00</b>
	SAM	41.73	66.985	46.05	56.65	36.06	<b>64.939</b>	39.51	<b>64.04</b>	41.09	51.35
	MAE	<b>42.85</b>	<b>59.433</b>	<b>49.70</b>	54.37	21.20	85.221	22.66	19.36	34.87	27.58

Table 12: Evaluation of Frozen and Adapted models using 3 vs. 4 foundation models.FM: Foundation Model, Surgical:Abdominal, Natural:YTOBJ.

Setting	FM number	Surgical			Natural		
		mBO	FG-ARI	Cor-Loc	mBO	FG-ARI	Cor-Loc
Frozen	3	39.78	51.28	54.26	42.92	33.74	<u>51.85</u>
	4	<u>48.65</u>	<u>54.07</u>	<u>64.30</u>	<u>43.33</u>	<u>38.78</u>	50.56
Adapted	3	49.58	58.52	69.58	47.32	41.86	58.97
	4	<b>50.34</b>	<b>59.04</b>	<b>70.73</b>	<b>48.98</b>	<b>45.32</b>	<b>65.11</b>

## G.2 Efficacy of Adaptation Layer for Single Foundation Models

Next, to assess the benefit of feature adaptation, we evaluate the performance when using a single foundation model augmented with a lightweight adapter module. In these experiments, we attach either a simple **MLP** adapter or the proposed **AFM** adapter on top of the frozen foundation model, then train the adapter (keeping the foundation model weights fixed) for the downstream object-discovery. Here MOE is not used as it is only applicable for multiple foundation models. Table 11 (middle) reveals the performance: All foundation model show improvement with both adapter on YTVIS data, particularly AFM boost SAM’s CorLoc from 53.66 to 64.65. Adapters recover 54% of MAE’s performance gap vs. SAM on abdominal data (35.0 to 42.8 mBO). Importantly, the performance of foundation models when used individually lags the performance of concatenated adapted baselines, in particular when applied to a domain that is new to foundation models (+5 mBO, +9 FG-ARI, +9 Cor-Loc).

## G.3 Using Different Numbers of Foundation Models

We further examine the impact of using different numbers of foundation models within FORLA. When employing four foundation models (including ViT-B/16, the smaller ViT variant), the computational overhead increases modestly from 1.4 ms (using only DINO) to 3.8 ms per image. Considering recent domain-specific foundation models such as RADIOv2.5 [21], one may question whether fewer models could provide sufficient representational power, particularly in the surgical domain.

To investigate this, we evaluated configurations using either three (DINO, SAM, and CLIP) or four foundation models, under both frozen and adapted settings. As shown in Table 12, the configuration with four foundation models consistently achieved the best performance, especially on surgical images, which are typically underrepresented in the pretraining of generic foundation models. Despite the modest increase in computational cost (an additional 2.4 ms per image), the performance gains justify the use of four foundation models for achieving stronger generalization and robustness.



Table 13: Additional results of self-distillation on the single data with MLP and MOE adapters.

Adapter	Method	Abdominal				Thoracic			
		mBO	mHD↓	FG-ARI	CorLoc	mBO	mHD↓	FG-ARI	CorLoc
MLP	w/o self-distill	51.85	43.025	59.04	69.25	54.00	72.358	43.59	58.00
	w self-distill	56.06	35.005	63.09	79.23	44.86	85.707	36.73	49.28
MOE	w/o self-distill	51.83	42.758	57.54	72.02	51.10	74.670	41.87	55.89
	w self-distill	<b>57.05</b>	<b>35.039</b>	<b>63.27</b>	<b>79.62</b>	<b>62.16</b>	<b>52.544</b>	<b>47.96</b>	<b>72.32</b>

Table 14: Decoder performance of the teacher branch in comparison to student branch in FORLA.

Domain	Sub-domain	mBO		mHD ↓		FG-ARI		CorLoc	
		Teacher	Student	Teacher	Student	Teacher	Student	Teacher	Student
Surgical	Abdominal	51.29	57.86	38.458	34.371	58.31	64.88	77.73	80.30
	Cholec	32.52	34.20	57.247	56.942	42.31	44.35	52.13	54.49
	Thoracic	56.86	61.86	52.702	50.771	43.60	47.58	76.04	75.42
	Average	46.89	51.31	49.469	47.36	48.07	52.27	68.63	70.07
Natural	COCO	26.72	27.22	93.473	93.541	28.41	30.11	66.88	64.94
	PASCAL	39.36	40.83	76.277	75.642	36.41	39.51	65.38	64.84
	YTVIS	37.96	38.51	61.396	62.472	41.63	43.08	65.50	64.92
	YTOBJ	51.06	51.92	–	–	–	–	62.03	66.87
	Average	38.78	39.62	77.05	77.22	35.48	37.57	64.95	65.39

#### G.4 Distillation Dynamics

Table 13 (bottom) shows additional effects of self-distillation on abdominal and thoracic data with MLP and MOE adapter: 1) **MLP overfitting**: Self-distillation improves Abdominal CorLoc (improved performance score by 10) but harms Thoracic performance (-8.7 points performance), suggesting MLP could over-fit to smaller specialized dataset with less constrains for feature reconstruction. 2) **MOE Robustness**: MOE adapters gain +16.4 CorLoc on Thoracic with distillation, leveraging expert gates to preserve generalizability. This once again confirmed that MLP could be the least suitable adapter choice in our FORLA federated object-centric learning framework, as demonstrated in Table 2.

#### G.5 Teacher Decoder Analysis

In FORLA, teacher and student branch share the same global adapter and Slot attention module, while having different decoders for reconstructing features and slot attention masks. We analyzed the performance of the decoder branches by comparing the teacher and student models across all datasets. As shown in Table 14, the teacher branch performance is consistently competitive and often close to the performance of the student decoder, despite the fact that its adapter is not directly optimized but rather updated through EMA (early stage) or local FedAvg (later stage).

Table 14 reveals nuanced performance differences between teacher and student decoders in federated learning: 1) **Dominance on surgical domain**: Student decoder achieves +4.42 average mBO improvement (51.31 vs. 46.89), demonstrating superior object discovery from federated knowledge aggregation; it maintains boundary precision with 4.7% lower mHD (47.36 vs. 49.47), crucial for anatomical structures; 2.44 FG-ARI gain highlights better foreground-background separation. 2) **Gains on natural domain**: Student leads marginally in mBO (+0.84) FG-ARI (+2.09), and CorLoc (0.44). This validates the effectiveness of our teacher-student design, where the student benefits from both local and global knowledge transfer via FL and reconstruct more constrained features, while the teacher adapts more aggressively and encourages the student to re-discover more specialized and transferable object-centric representations.

#### G.6 Compared to Zero-shot transfer performance of slot attention

We performed additional testing that confirmed our hypothesis which was that, if the domain gap is small, the zero-shot and transfer learning will guarantee good performance. However, FORLA FL will outperform transfer learning when the domain gap is large (Table 15). We first tested zero-shot transfer of slot attention trained on Natural images (PASCAL and YTOBJ) to surgical images (Abdominal). In this case zeroshot performance is significantly lower compared to FORLA and even compared to models individually trained on abdominal data.

Table 15: Comparison of zero-shot transfer from Natural to surgical, individual training, and FORLA.

Method	mBO	FG-ARI	Cor-Loc
Zero-shot (PASCAL $\rightarrow$ Abdominal)	35.61	39.81	35.90
Zero-shot (YTOBJ $\rightarrow$ Abdominal)	33.02	37.69	30.65
Individual	50.34	59.04	70.73
FORLA	<b>57.86</b>	<b>64.88</b>	<b>80.30</b>

### G.7 Inference on Videos

We demonstrate that a slot attention module trained with our FORLA framework can be directly applied to video scene decomposition by leveraging RNN-based slot inference techniques [67]. Specifically, we evaluated our FORLA-trained slot attention model on the YTOBJ video dataset sampled at 1 frame per second (fps). For comparison, we trained individualized slot attention (SA) models directly on YTOBJ using adapted versions of foundation models, including DINO (equivalent to DINOSAUR [52]) and SAM.

As shown in Figures 4 and 5, the slots produced by FORLA maintain strong temporal consistency across different video sequences. The performance of individualized models reveals that SAM- and DINO-based SA models exhibit distinct strengths: the SAM-based model excels at decomposing close-up scenes such as animals or vehicles captured by a near camera, while the DINO-based model performs better in delineating objects from the background in distant or wide-angle scenes.

FORLA however consistently tracks objects both in near and far scenes in a more fine-grained and semantically meaningful manner. Unlike the SAM or DINO-based models, FORLA is less likely to segment objects into implausible parts (e.g., splitting a cat or car into non-semantic regions), demonstrating stronger object-level coherence and generalization.

## H Visualization of Feature Representation

Figures 6 and 7 provide additional visualizations of PCA maps and Slot Attention (SA) masks for surgical and natural image domains, respectively, across different methods. We visualize the first three principal components of the feature representations using RGB channels. These include both frozen features from foundation models and adapted features learned through our FORLA federated learning framework.

In the surgical domain, the frozen foundation model features demonstrate a general understanding of texture-based separation—for example, surgical instruments and tissues often appear as different colors in the PCA map. This indicates some level of semantic separation. However, the foundation features struggle to distinguish between multiple instances of similar instruments, as they are often represented with the same color. In contrast, the FORLA-adapted features effectively assign distinct semantic representations to different instrument instances, enabling clearer separation. Additionally, FORLA representations offer greater differentiation between various tissue textures. This is reflected in the PCA maps, where different tissues are represented by homogenous yet distinct colors, aiding the SA module in grouping regions into semantically meaningful categories.

In the natural image domain, foundation model features can delineate some salient objects, due to their pretraining on natural image datasets, but the representations remain relatively coarse. With FORLA’s unsupervised adaptation, the feature representations become significantly more fine-grained. For example, tiny objects that are otherwise overlooked in foundation features become clearly highlighted in the PCA maps. The model also learns to extract subtle cues from cluttered backgrounds, such as the texture of plants, and can even distinguish between visually similar objects located close to each other. These properties of the adapted representation enable the downstream SA module to decompose scenes into semantically coherent segments.

Such fine-grained representation and semantically meaningful decomposition suggests that FORLA-learned representations can potentially generalize well to downstream tasks, including semi-supervised or weakly supervised segmentation, as well as a variety of prediction tasks that benefit from pre-segmented scene understanding [54].

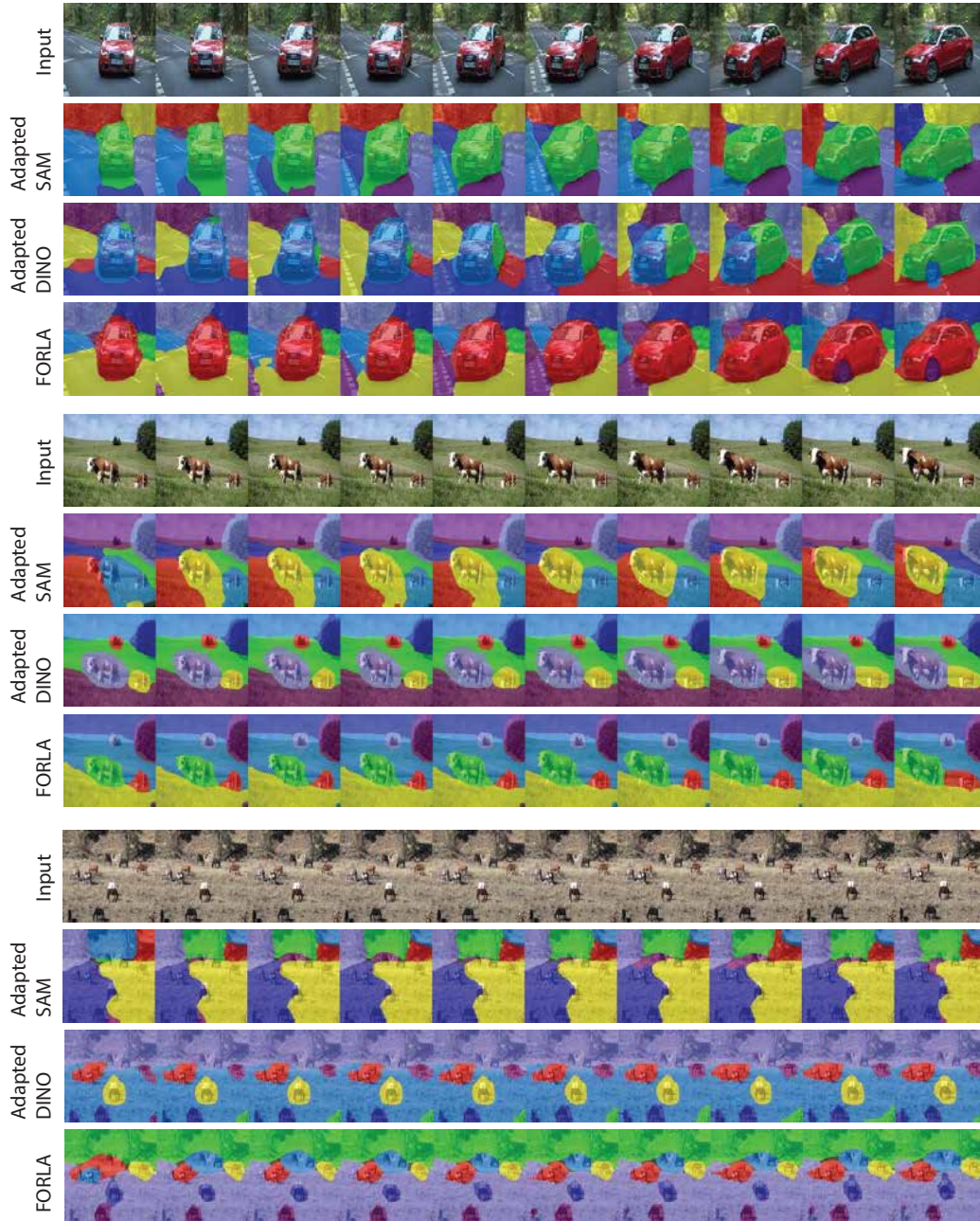


Figure 4: Inference using RNN like slot initialization [67] on YTOBJ videos. We compared to individualized trained SA models on YTOBJ using adapted single foundation model including DINO (as used in DINOSAUR [52]) and SAM.



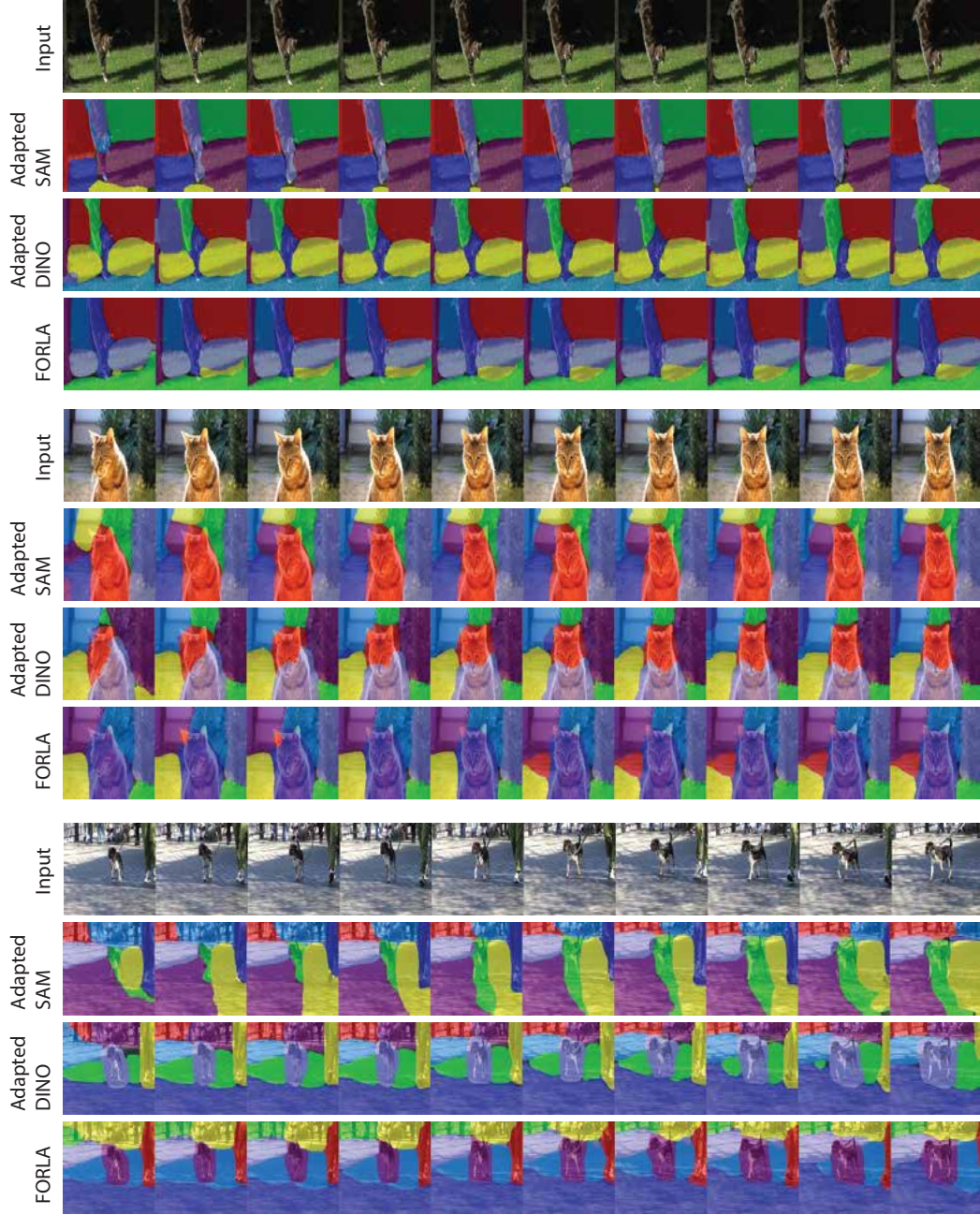


Figure 5: Additional results on YTOBJ videos compared to individualized trained SA models with single foundation model adaptation.

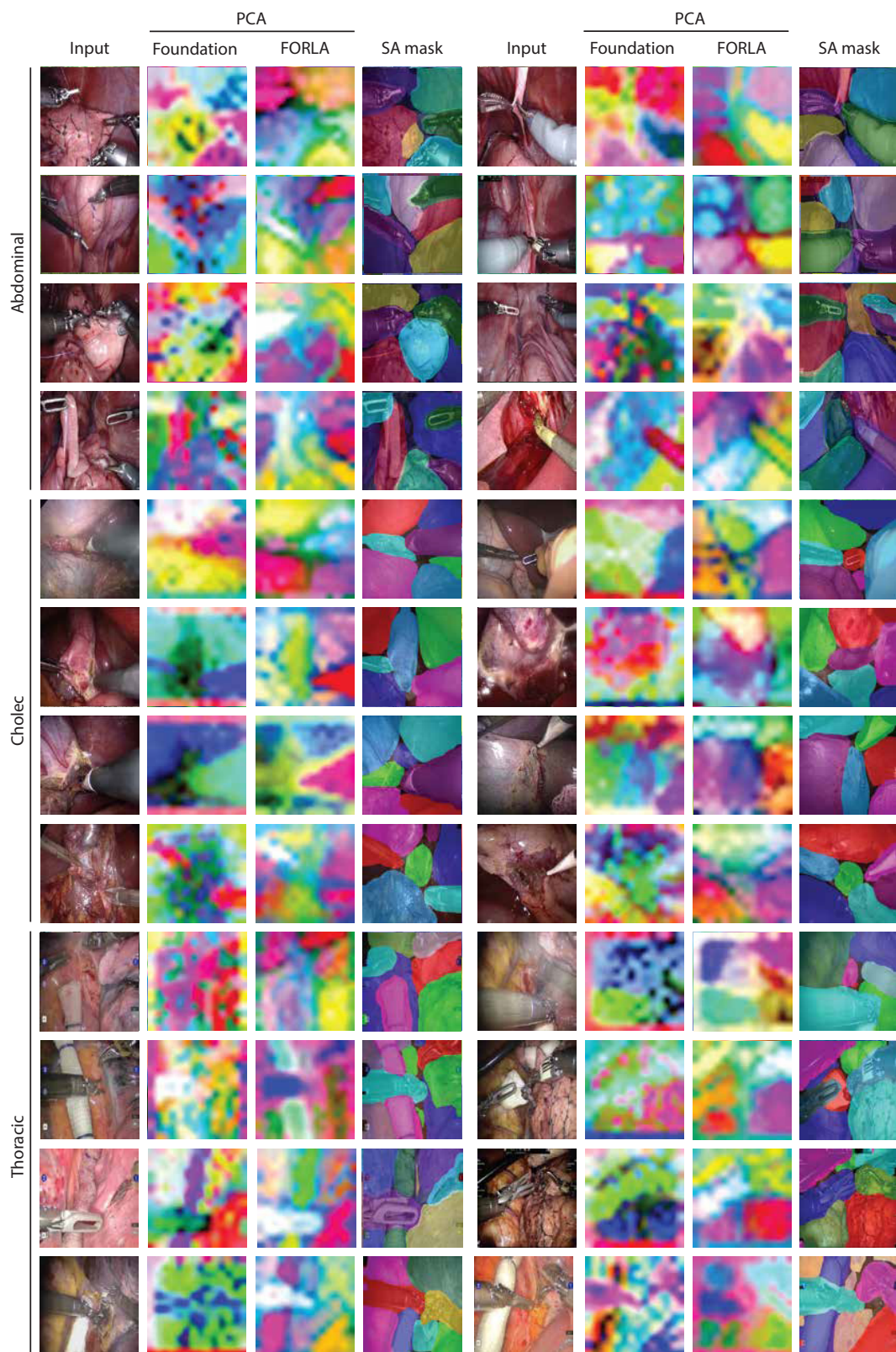


Figure 6: PCA map of raw foundation model features and FORLA representation on surgical image samples. Slot attention (SA) masks of FORLA are shown at 4th and 8th columns.



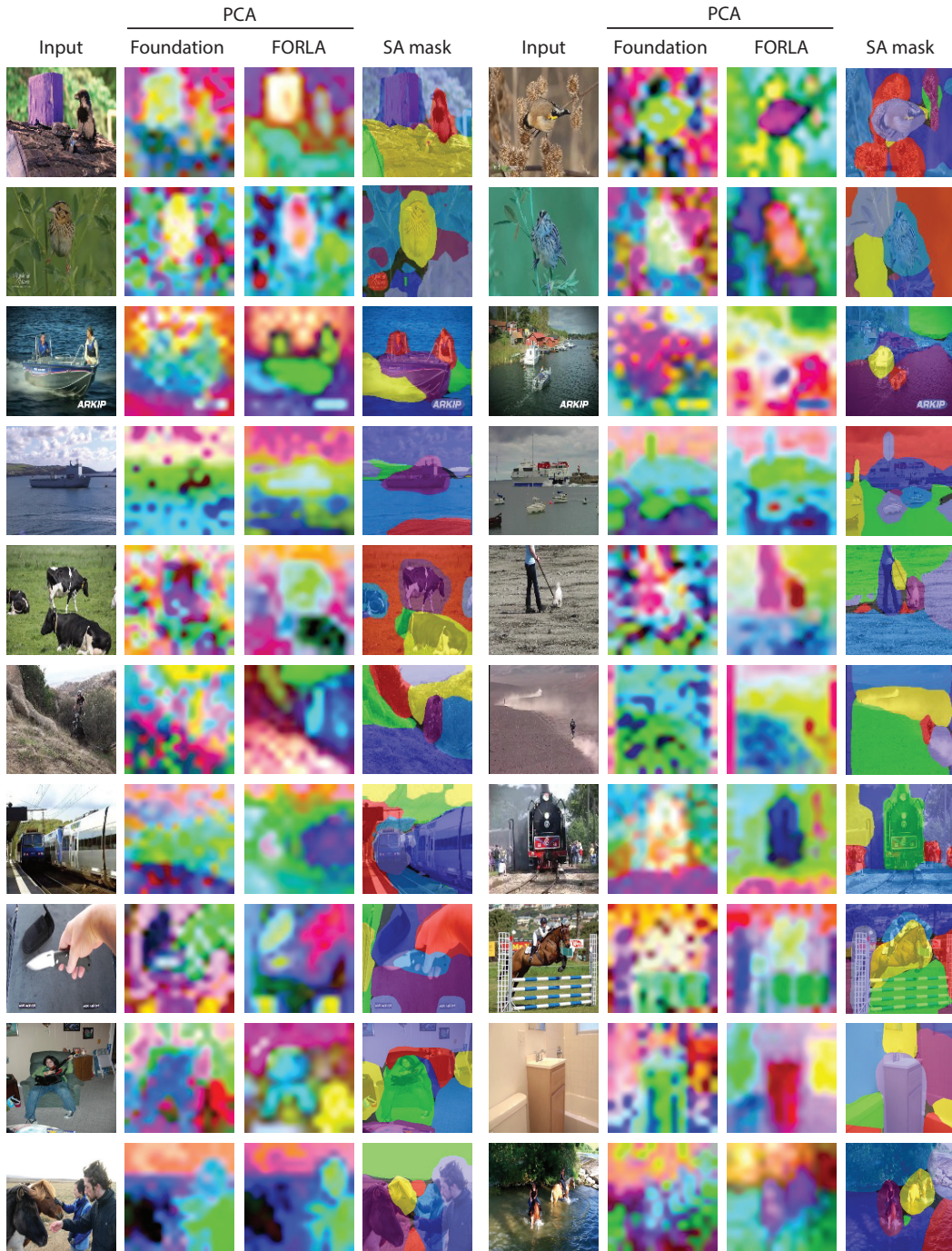


Figure 7: PCA map of raw foundation model features and FORLA representation on natural image samples. Slot attention (SA) masks of FORLA are shown at 4th and 8th columns.