Supplementary material of MoSiC: Optimal-Transport Motion Trajectory for Dense Self-Supervised Learning

Mohammadreza Salehi,[1,3*] Shashanka Venkataramanan,[2*] Ioana Simion,[1]
Efstratios Gavves,[1] Cees G. M. Snoek,[1] Yuki M Asano[3]
[1] VIS Lab, UvA [2] Valeo.ai [3] Fundamental AI Lab, UTN

6. Appendix

6.1. Additional Experiments

Comparison to TimeT. Here, we compare MoSiC with TimeT, both initialized from the same DINO backbone. As shown in Figure 5, MoSiC consistently outperforms TimeT across Pascal VOC and ADE20K for both DINO and DINOv2 backbones. Notably, while TimeT struggles to further enhance strong vision backbones such as DINOv2 beyond the baseline, MoSiC achieves consistent improvements across all benchmarks, demonstrating its greater generalizability.

Image classification performance. As shown in Table 8, while MoSiC significantly enhances the dense understanding of DINOv2, it results in only a 0.8% performance reduction—negligible compared to the gains of up to 6% in certain benchmarks.

Table 8. **MoSiC vs. DINOv2 on downstream image classification.** MoSiC greatly improves DINOv2's dense understanding while incurring a minimal 0.8% performance reduction, outweighed by gains of up to 6% in some benchmarks.

Method	Backbone	CIFAR-100	ImgNet-100	CIFAR-10	Average
DINOv2	ViT-B/14	86.2%	90.5%	97.8%	91.5%
MoSiC	ViT-B/14	84.8%	89.9%	97.3%	90.7%

Full finetuning results. In Table 9 and Table 10, we evaluate the performance of MoSiC when used as the backbone for object detection and semantic segmentation in the full finetuning setting. For object detection ViT-Det [38] and for semantic segmentation Segmenter [55] frameworks are used. As shown, our method consistently outperforms DINOv2 across all datasets and evaluation protocols, setting a new state-of-the-art—despite being fine-tuned solely on video data that differs in distribution from the evaluation datasets.

*Equal Contribution. Correspondence: s.salehidehnavi@uva.nl

Since full finetuning updates all model parameters, the superior performance of MoSiC indicates that learning a more structured, object-aware feature space is achievable even without any image dataset, by finetuning on videos alone.

Table 9. **MoSiC vs. DINOv2 on object detection.** We use ViTDet [38] on COCO [8] and report Average Precision (AP) on bounding-box object detection (AP^{box}) and instance segmentation (AP^{mask}).

Method	Backbone	Params	AP^{box}	AP^{mask}
DINOv2	ViT-S/14	21M	42.5	36.7
MoSiC	ViT-S/14	21M	42.5	36.8
DINOv2	ViT-B/14	85M	46.1	41.9
MoSiC	ViT-B/14	85M	46.4	42.0
DINOv2	ViT-L/14	307M	51.6	45.9
MoSiC	ViT-L/14	307M	51.8	46.0

Generalization to large variants. We report the performance for the large variant of MoSiC for all the experiments in the paper, as shown by Table 9, Table 10, Table 11, Table 12a, Table 12b, Table 13, and Table 14. Our model can increase the performance of DINOv2-L even more than the small and base variants, showing the effectiveness of our method on large models.

Tracker ablation. We train and evaluate MoSiC using both RAFT and CoTrackerv2 to investigate the effect of different trackers on our method. From Table 15, MoSiC improves over DINOv2 with both these point trackers. MoSiC is compatible with multiple point trackers, yet the better the tracker, the better MoSiC performs.

6.2. Additional Visualizations

Hummingbird qualitative results for MoSiC. Here, we show the qualitative results of our method for the visual in-context learning evaluation (Hummingbird benchmark) shown by Table 1. As shown, although MoSiC is finetuned

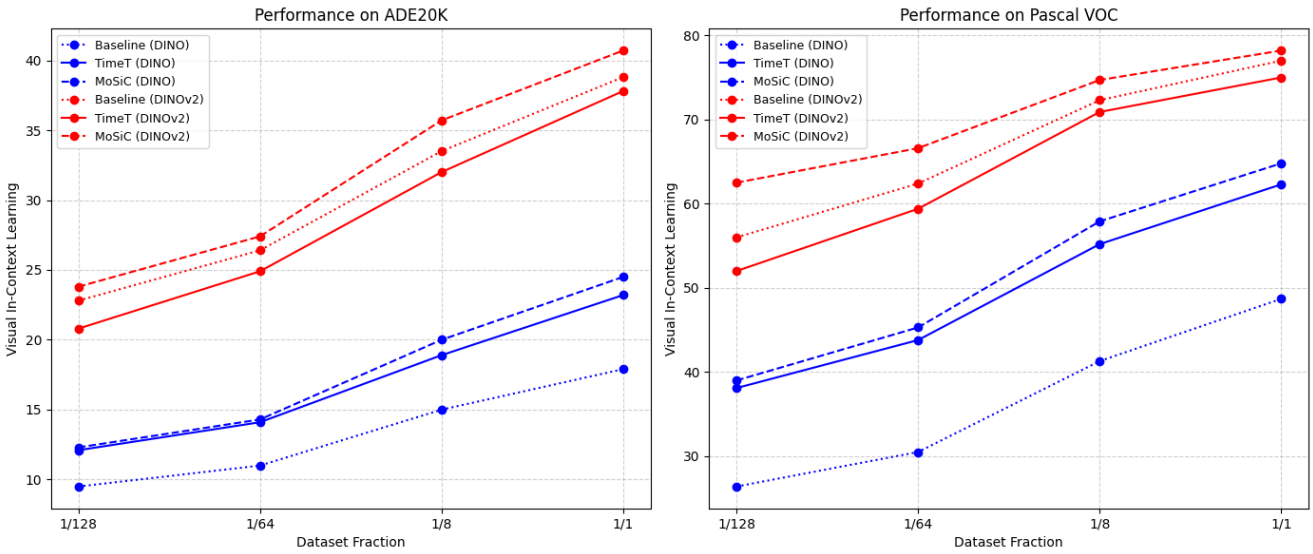

Figure 5. **TimeT vs. MoSiC.** As the figure shows, MoSiC consistently improves both DINO and DINOv2 backbones, while TimeT only improves DINO.

Table 10. **Evaluation of full Finetuning with Segmenter.** Various backbones pre-trained with different self-supervised learning methods are fine-tuned using Segmenter [55]. The table shows the mIoU scores obtained on validation sets across 4 different datasets.

Method	Backbone	Params	Pascal Context	Pascal VOC	COCO-Stuff	ADE20K
DINO	ViT-S/16	21M	46.0	80.3	43.2	43.3
CrOC	ViT-S/16	21M	46.0	80.9	42.9	42.8
TimeT	ViT-S/16	21M	47.4	80.4	43.1	43.5
CrIBo	ViT-S/16	21M	49.3	82.3	43.9	45.2
DINOV2	ViT-S/14	21M	58.0	80.4	42.1	44.0
MoSiC	ViT-S/14	21M	**58.5**	**81.1**	**42.5**	**44.4**
DINO	ViT-B/16	85M	45.8	82.2	44.4	45.0
MAE	ViT-B/16	85M	47.9	82.7	45.5	46.4
CrIBo	ViT-B/16	85M	49.2	83.4	44.6	46.0
DINOV2	ViT-B/14	85M	62.0	85.2	48.3	51.9
MoSiC	ViT-B/14	85M	**62.5**	**85.8**	**48.6**	**52.4**

on YTVOS videos—which differ significantly in distribution from Pascal—it still produces accurate and precise semantic segmentation maps, characterized by distinct IDs and tight boundaries.

Overclustering visualizations for MoSiC. We present the overclustering qualitative results of MoSiC on Pascal VOC for $K = 50$, approximately twice the number of object categories in the dataset, as shown in Figure 7. As shown, MoSiC not only localizes objects precisely but also identify them with different cluster ids demonstrated by different colors. For instance, classes such as birds, motorcycles, dogs, cats, and cars are clearly identifiable.

Failure cases. While MoSiC is robust to occlusions, it can struggle when motion cues are subtle or ambiguous, reducing the effectiveness of clustering supervision. In rare cases of tracker failure—due to extreme motion, prolonged occlusions, or appearance changes—noisy trajectories may propagate incorrect signals (Figure 8). Overly dense tracking (e.g., 32×32 grids) can similarly introduce spurious correspondences (Ablations - Table 5c). These issues arise from dependence on accurate tracking and clustering, though MoSiC mitigates them by focusing losses on visible points.

6.3. Dense Post-Pretraining

Implementation Framework We implement our model in Python using Torch [46].

Table 11. **In-context scene understanding benchmark.** We evaluate dense nearest neighbor retrieval performance across various training data proportions on ADE20K [69] and Pascal VOC [18]. Retrieved cluster maps are compared with the ground truth via Hungarian matching [33]. We report mIoU; higher is better.

			ADE20K				Pascal VOC			
Method	Backbone	Params	1/128	1/64	1/8	1/1	1/128	1/64	1/8	1/1
Trained on Images										
DINO [10]	ViT-S/16	21M	9.5	11.0	15.0	17.9	26.4	30.5	41.3	48.7
CrOC [54]	ViT-S/16	21M	8.7	10.8	15.2	17.3	34.0	41.8	53.8	60.5
SelfPatch [66]	ViT-S/16	21M	10.0	10.9	14.7	17.7	28.4	32.6	43.2	50.8
Leopart [71]	ViT-S/16	21M	12.9	14.8	19.6	23.9	44.6	49.7	58.4	64.5
CrIBo [36]	ViT-S/16	21M	14.6	17.3	22.7	26.6	53.9	59.9	66.9	72.4
DINOv2 [42]	ViT-S/14	21M	22.8	26.4	33.5	38.8	56.0	62.4	72.3	77.0
Finetuned on Videos										
TimeT [50]	ViT-S/16	21M	12.1	14.1	18.9	23.2	38.1	43.8	55.2	62.3
MoSiC	ViT-S/14	21M	**23.8**	**27.4**	**35.7**	**40.7**	**62.5**	**66.6**	**74.7**	**78.2**
Trained on Images										
MAE [23]	ViT-B/16	85M	10.0	11.3	15.4	18.6	3.5	4.1	5.6	7.0
DINO [10]	ViT-B/16	85M	11.5	13.5	18.2	21.5	33.1	37.7	49.8	57.3
Leopart [71]	ViT-B/16	85M	14.6	16.8	21.8	26.7	50.1	54.7	63.1	69.5
Hummingbird [5]	ViT-B/16	85M	11.7	15.1	22.3	29.6	50.5	57.2	64.3	71.8
CrIBo [36]	ViT-B/16	85M	15.9	18.4	24.4	28.4	55.9	61.8	69.2	74.2
DINOv2 [42]	ViT-B/14	85M	24.2	27.6	34.7	39.9	55.7	61.8	72.4	77.1
Finetuned on Videos										
MoSiC	ViT-B/14	85M	**25.4**	**29.3**	**37.3**	**42.6**	**65.5**	**69.8**	**76.9**	**80.5**
DINOv2 [42]	ViT-L/14	307M	22.0	25.2	32.8	37.9	47.9	54.8	68.1	74.4
MoSiC	ViT-L/14	307M	**24.7**	**28.1**	**35.7**	**41.0**	**55.0**	**62.0**	**73.6**	**78.5**

Table 12. **Frozen clustering-based evaluations.** *(a)* We evaluate the models using K-means with various clustering granularities K on the features of Pascal VOC [18] and COCO-Things [8]. Cluster maps are matched to the ground-truth via Hungarian matching [33], We report mIoU; higher is better. *(b)* We post-process these maps for unsupervised semantic segmentation on Pascal VOC. †: ViT-S/16.

(a) Clustering

	Pascal VOC			COCO-Things		
Method	K=100	K=300	K=500	K=100	K=300	K=500
Trained on Images						
DINO [10]†	10.1	13.9	17.3	14.4	18.8	19.2
CrOC [54]†	10.2	16.4	20.0	22.4	14.7	18.1
iBOT [70]†	16.5	23.8	31.1	15.5	26.6	28.0
EVA-CLIP [56]	31.7	37.4	41.4	30.5	38.0	39.8
DINOv2R-S14 [16]	34.8	46.7	49.5	32.0	38.9	41.2
Leopart [71]†	39.2	46.5	51.2	38.3	47.8	53.2
CrIBo [36]†	40.3	51.3	54.5	40.2	46.0	48.3
DINOv2-S14 [42]	43.2	55.1	58.6	43.8	51.6	53.1
Finetuned on Videos						
TimeT-S16 [50]†	34.6	43.6	46.2	34.9	42.7	44.6
MoSiC-S14	50.0	58.8	60.2	45.8	53.2	54.9
MoSiC-B14	52.5	62.1	65.7	47.8	56.3	58.6
MoSiC-L14	**54.8**	**69.6**	**71.3**	**51.1**	**59.0**	**60.8**

(b) Semantic segmentation

Method	mIoU
Trained on Images	
MaskConstrast [59]†	35.1
DINOv2R-S14 [16]	35.1
DINOv2-S14 [42]	37.5
DINOv2-B14 [42]	36.7
DeepSpectral [40]†	37.2
DINOSAUR† [52]	37.2
Leopart [71]†	41.7
COMUS [67]†	50.0
Finetuned on Videos	
TimeT [50]†	41.1
MoSiC-S14	51.2
MoSiC-B14	54.4
MoSiC-L14	**57.1**

Datasets Our pretraining datasets include COCO [8] and the ImageNet-100 subset of the original ImageNet [49]. COCO comprises approximately 118,000 scene-centric images, while ImageNet-100 contains around 100,000 object-centric images.

Network Architecture We use vision transformers as our backbone, specifically training on ViT-Small and ViT-Base [17]. Following [10, 20], we adopt a student-teacher setup, where the teacher's weights are updated via the exponential moving average of the student's weights.

Projection Head As in [10], our projection head comprises three linear layers with a hidden dimensionality of 2048, Gaussian error linear units (GELU) as the activation function [25], and an output dimensionality of 256.

Table 13. **Unsupervised video semantic segmentation results** for clustering and over-clustering on DAVIS [48] and Youtube-VOS (YTVOS) [65]. For clustering, the Hungarian algorithm [33] matches clusters (K) to ground truth (GT) per frame (F), clip (C), or dataset (D). For over-clustering: K=10 (F, C), K=200 (D, DAVIS), K=500 (D, YTVOS). We report mIoU; higher is better.

	CLUSTERING						OVER-CLUSTERING					
	YTVOS			DAVIS			YTVOS			DAVIS		
	F	*C*	*D*	*F*	*C*	*D*	*F*	*C*	*D*	*F*	*C*	*D*
TRAINED ON IMAGES												
DINO [10]†	39.1	37.9	1.9	30.2	31.0	1.6	66.2	65.4	4.0	56.9	54.9	17.9
Leopart [71]†	39.2	37.9	11.7	30.3	30.2	16.5	64.5	62.8	15.5	54.9	54.4	26.7
DINOv2-S14 [42]	56.3	55.5	12.8	57.4	57.4	13.6	62.8	62.5	17.3	57.9	58.5	25.5
DINOv2R-S14 [16]	56.8	55.4	14.7	57.3	57.8	14.3	64.0	63.5	21.5	58.1	59.5	27.2
FINETUNED ON VIDEOS												
STEGO [22]†	41.5	40.3	2.0	31.9	31.0	3.2	58.1	54.3	5.1	47.6	46.3	10.4
DINO [10]†	37.2	36.1	1.2	29.3	29.2	2.4	53.1	50.9	1.3	45.4	44.0	8.6
Leopart [71]†	41.5	40.5	7.7	37.5	36.5	12.6	60.8	59.8	6.8	53.7	53.1	16.8
TimeT [50]†	50.2	51.4	12.8	49.5	49.2	12.8	60.6	59.4	18.1	55.9	57.6	22.0
MoSiC-S14	**60.6**	**59.6**	18.4	58.9	**60.8**	23.4	66.8	65.6	24.4	**58.4**	59.8	29.0
MoSiC-B14	59.5	58.6	18.7	59.5	59.2	24.2	**67.2**	64.1	23.1	57.9	59.3	32.0
MoSiC-L14	60.2	59.1	**19.4**	**59.7**	59.1	**26.3**	67.1	**65.9**	**26.9**	58.3	**60.2**	**33.5**

Table 14. **Linear segmentation performance.** A linear segmentation head is trained on top of the frozen spatial features obtained from different feature extractors. We report the mIoU scores achieved on the validation sets of 4 different datasets.

METHOD	BACKBONE	PARAMS	COCO-THINGS	COCO-STUFF	PASCAL VOC	ADE20K
TRAINED ON IMAGES						
DINO [10]	ViT-S/16	21M	43.9	45.9	50.2	17.5
iBOT [70]	ViT-S/16	21M	58.9	51.5	66.1	21.8
CrOC [54]	ViT-S/16	21M	64.3	51.2	67.4	23.1
CrIBo [36]	ViT-S/16	21M	64.3	49.1	71.6	22.7
DINOv2 [42]	ViT-S/14	21M	81.4	58.3	78.9	37.9
FINETUNED ON VIDEOS						
TimeT [50]	ViT-S/16	21M	58.2	48.7	66.3	20.7
MoSiC	ViT-S/14	21M	**82.3**	**61.0**	**79.7**	**39.6**
TRAINED ON IMAGES						
DINO [10]	ViT-B/16	85M	55.8	51.2	62.7	23.6
MAE [23]	ViT-B/16	85M	38.0	38.6	32.9	5.8
iBOT [70]	ViT-B/16	85M	69.4	55.9	73.1	30.1
CrIBo [36]	ViT-B/16	85M	69.6	53.0	73.9	25.7
EVA-CLIP [56]	ViT-B/14	86M	75.9	48.0	70.4	34.6
DINOv2 [42]	ViT-B/14	85M	84.0	58.9	80.3	42.6
FINETUNED ON VIDEOS						
MoSiC	ViT-B/14	85M	**85.8**	**61.4**	**81.5**	**43.6**
DINOv2 [42]	ViT-L/14	307M	83.8	58.0	79.7	41.8
MoSiC	ViT-L/14	307M	**85.7**	**61.5**	**81.8**	**44.7**

Table 15. **Choice of Point Trackers.** In-context scene understanding results (mIoU) on PVOC and ADE20K. Higher is better.

Method	**PVOC (mIoU)**	**ADE20K (mIoU)**
DINOv2	77.0	38.8
RAFT	77.6	39.7
CoTracker2	78.0	40.2
CoTracker3	**78.2**	**40.7**

Optimization We train both network sizes using a cosine learning rate schedule that decays to 0 over 8 epochs for DINOv2 and 100 epochs for DINO, except for the ablation studies, where we use 50 epochs on DINO. The initial learning rate for the projection head is set to $1e{-4}$ across all experiments, while the backbone's learning rate is $1e{-5}$. The teacher's weights are updated using an exponential moving average with the coefficient 0.99. We optimize our model with Adam [30], applying a cosine weight decay schedule.

6.4. Evaluation Setup

Visual In-Context Learning The Dense Nearest Neighbor Retrieval Evaluation is a retrieval-based scene understanding benchmark introduced by [5]. It aims to assess the scene

understanding capabilities of a dense image encoder. The evaluation follows these steps:

1. **Memory Bank Construction**: Given a dataset of images with dense annotations, two memory banks are created. One stores image patch features extracted from the spatial output of a dense encoder applied to the training images, while the other stores the corresponding patch labels from the dataset annotations.
2. **Query Processing**: For each image in the validation set, the spatial output of the dense image encoder is computed. For each patch representation, the k nearest neighbors are retrieved from the feature memory bank. The labels of these nearest neighbors are then aggregated to construct the query's annotation.
3. **Comparison**: The generated annotation for the image is compared against the ground truth annotation to evaluate performance.

Since the original implementation by [5] is unavailable, we use the open-source implementation from [43]. This implementation adheres to the original authors' methodology, including the use of the ScaNN library [21] for efficient nearest neighbor retrieval. For our experiments, we follow the setup used by the Hummingbird Model authors [5], utilizing a memory size of $10,240,000$ and configuring ScaNN with 30 nearest neighbors, consistent with the Hummingbird model evaluation.

Final results are reported as mean Intersection over Union (mIoU) on four different fractions of two datasets: Pascal VOC 2012 [18] and ADE20K [69]. The sub-sampling factors considered are 1, 8, 64, and 128. For factors greater than 1, results are averaged over five different seeds. These dataset subsets are created by randomly and uniformly selecting a distinct set of images from the training split, ensuring an approximately equal number of unique images per annotation label. For instance, for the $1/128$ fraction of Pascal VOC 2012, we sample around 83 images, ensuring that each of the 20 labels (excluding the background) appears in at least 4 different images within the subset.

Overclustering Following [71], we conduct the Overclustering experiment by applying K-Means clustering (using FAISS [28]) to all spatial tokens from our backbone, omitting the projection head. The resulting clusters are then mapped to the dataset's ground-truth classes using a two-step process: first, greedy matching based on pixel-level precision, followed by Hungarian matching [33] on the combined cluster maps. This procedure ensures that the evaluation metric remains permutation-invariant [27].

Input images are resized to 448×448, while overclustering is performed on downsampled 100×100 masks to accelerate Hungarian matching. The final results are reported as the average mean Intersection over Union (mIoU) over five different seeds across four datasets: COCO-Thing and COCO-Stuff [8], Pascal VOC 2012 [18], and ADE20K [69].

Linear Segmentation For linear segmentation, we follow the setup from Leopart [71]. Specifically, we process 448×448 images through our backbone to extract spatial feature outputs, apply bilinear interpolation to align them with the mask resolution, and use a linear head to generate segmentation predictions. These predictions are then compared to the ground-truth segmentation masks and optimized using cross-entropy loss.

To speed up training, we downsample the segmentation masks to 100×100. The linear head is optimized using Stochastic Gradient Descent with a weight decay of 0.0001, a momentum of 0.9, and a step-based learning rate scheduler. We find that a learning rate of 0.01 works well across the backbone models we evaluate. The linear heads are fine-tuned for 20 epochs.

We train and evaluate the linear heads on four datasets: Pascal VOC 2012 [18], subsets of COCO-Thing and COCO-Stuff , and ADE20K [69].

Fully Unsupervised Semantic Segmentation To further assess the scene understanding capabilities of our method, we evaluate it using the Fully Unsupervised Semantic Segmentation Evaluation [71]. This evaluation consists of two components: Cluster-based Foreground Extraction (CBFE) and Overclustering with Community Detection (CD).

CBFE clusters the spatial outputs of a model over a dataset and assigns each cluster as either background (*bg*) or foreground (*fg*). The foreground-background separation is guided by attention maps from a Vision Transformer, which provide cues for distinguishing between fg/bg regions. To construct the final hard fg-bg assignment, we average the attention heads, apply Gaussian filtering with a 7×7 kernel, and retain 70% of the attention mass to generate the binary mask. The rest of the configurations follow the original setup [71].

The CD metric [71] leverages local co-occurrence statistics among clusters to identify and categorize objects. This approach is entirely label-free, relying only on the local co-occurrence of clusters within an image based on an information-theoretic definition of network communities. Our CD evaluation configurations remain consistent with those used in Leopart [71].

We use the implementation from Leopart [71] and apply CBFE and CD to the non-augmented (*train*) split of Pascal VOC 2012 [18], evaluating on its full validation set. For CD, we report the best results across 10 seeds obtained via a hyperparameter search. Our best-performing parameters for CD+CBFE are: *weight_threshold* = 0.07, *markov_time* = 1.6, and *k_community* = 169.

7. Dataset Details

7.1. Image Datasets

Pascal VOC 2012 [18] This dataset, using the latest trainaug split, consists of 10,582 images with annotations spanning 21 classes, including one background class. The validation set contains 1,449 images. Following [59], we ignore unlabeled objects as well as the boundary class. For hyperparameter tuning of the fully unsupervised segmentation method [71] applied to our method, we use the PVOC12 train split with 1,464 images. Figure 10 provides an overview of dataset images overlaid with annotations.

COCO-Stuff 164K [8] COCO-Stuff is a scene-understanding dataset with labels across 91 "stuff" categories and 80 "thing" categories. The training set contains 118,000 images, while the validation set includes 5,000 images. Following [71], we use the COCO benchmark in two ways to better isolate object definitions.

First, we extract stuff annotations, which represent objects without clear boundaries and are often part of the background, using COCO-Stuff annotations [8]. We then consolidate the 91 detailed labels into 15 broader categories as described in [27], assigning the general label “other” to non-stuff objects since this label lacks specific semantic meaning. Non-stuff objects are ignored during training and evaluation. In our work, we refer to this dataset version as COCO-Stuff, which is used for Overclustering and Linear Segmentation (see Appendix 6.4).

Next, we extract foreground annotations using the panoptic labels from [31]. Instance-level annotations are merged into object categories using the authors’ provided script. Additionally, we consolidate the 80 detailed categories into 12 broad object classes. The background class is ignored during training and evaluation. This results in the COCO-Thing dataset version, which we use for Overclustering and Linear Segmentation (see Appendix 6.4).

ADE20K [69] ADE20K is a diverse dataset for semantic segmentation, containing finely detailed labels across 150 unique semantic categories. These include "stuff" labels such as sky and grass, as well as distinguishable objects like people and cars. The dataset features a wide range of scenes, with 20,210 images in the training set and 2,000 images in the validation set, making it one of the most challenging and diverse benchmarks for scene understanding. We use the full dataset in our experiments while ignoring the *others* label.

7.2. Video Datasets

DAVIS17 [48] is a benchmark for video object segmentation, consisting of 150 videos, with 60 allocated for training, 30 for validation, and 60 for testing. Since ground-truth foreground masks are only available for the first frames of the test set, the validation set is used for evaluation. An overview of dataset frames and annotations can be viewed in Figure 11.

YTVOS [65] is another large-scale video object segmentation dataset, significantly larger than DAVIS17, comprising 4,453 videos annotated with 65 object categories. Similar to DAVIS17, ground-truth masks are provided only for the first frames of the test and validation sets. Therefore, a fixed 20% subset of the training set is randomly sampled for evaluation; further details are provided in the supplementary material. Additionally, meta information is used to ensure that objects belonging to the same category retain consistent class IDs throughout the dataset, enabling semantic, category-level assessments. Figure 9 illustrates the object distribution in YTVOS while Figure 12 showcases example frames and annotations.

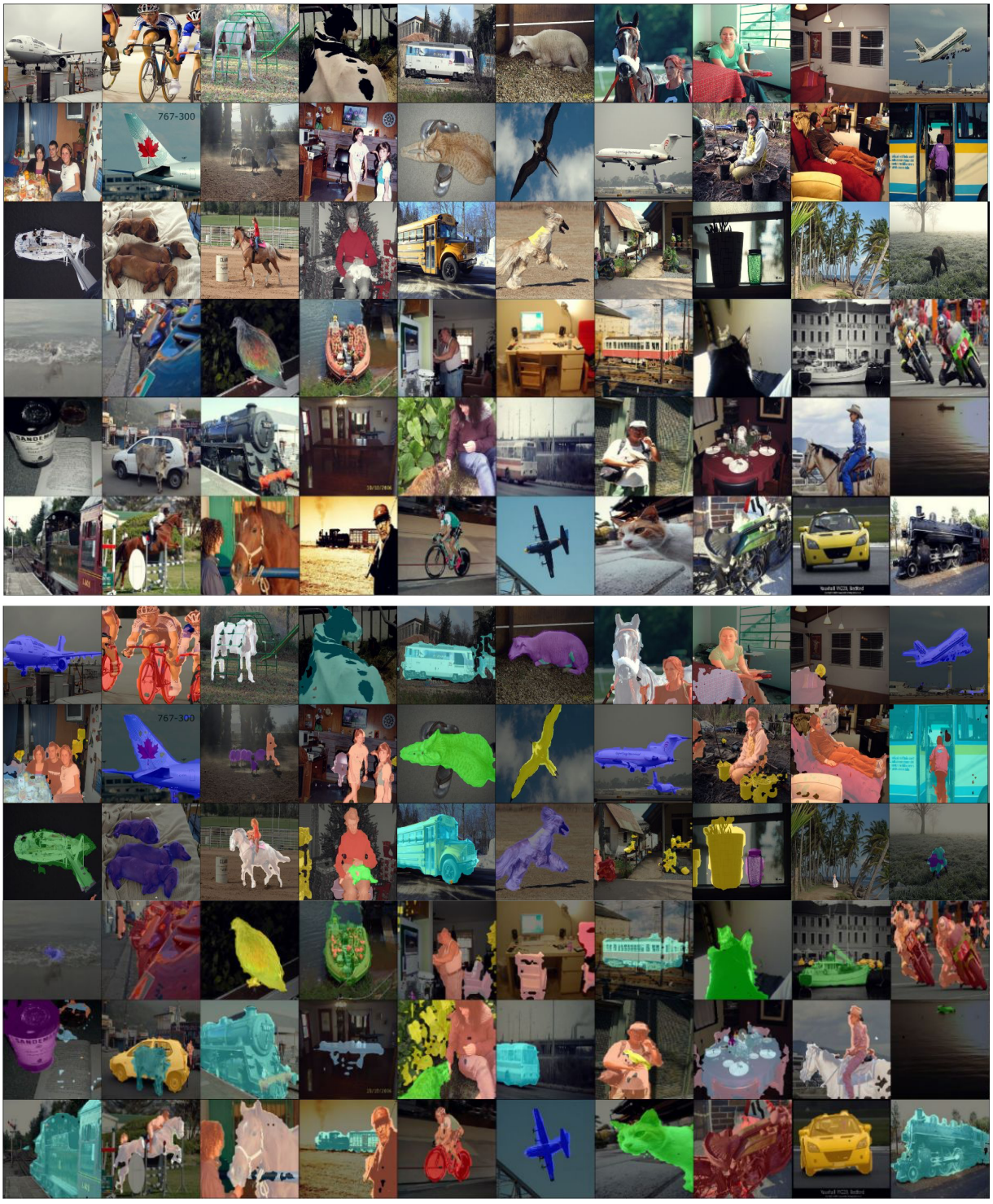

Figure 6. **Hummingbird qualitative results for `MoSiC` on Pascal VOC.** The first group of rows displays the images, the second shows the corresponding per-image masks, and the third overlays the masks on the images, aligned by their semantic IDs. As shown, although `MoSiC` is finetuned on YTVOS videos—which differ significantly in distribution from Pascal—it still produces accurate and precise semantic segmentation maps, characterized by distinct IDs and tight boundaries.

Figure 7. **MoSiC overclustering visualizations on Pascal for $\mathbf{K} = 50$.** MoSiC not only localizes objects precisely but also identify them with different cluster ids demonstrated by different colors. For instance, classes such as birds, motorcycles, dogs, cats, and cars are clearly identifiable.

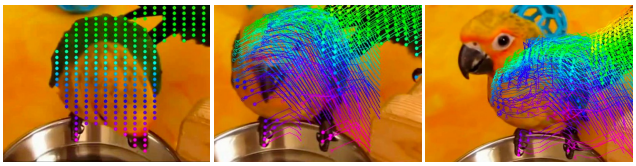

Figure 8. **Incorrect trajectories.** The point tracks that start from the beak of the bird and eventually track the bowl. This happens due to the ambigious motion of the bird's head.

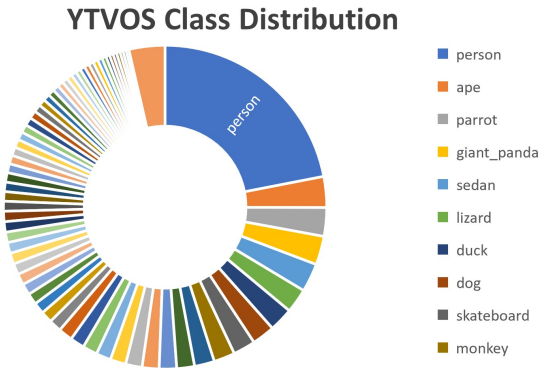

Figure 9. The distribution of classes in YouTube-VOS. Some of the more dominant classes are labeled. The image is taken from [50]

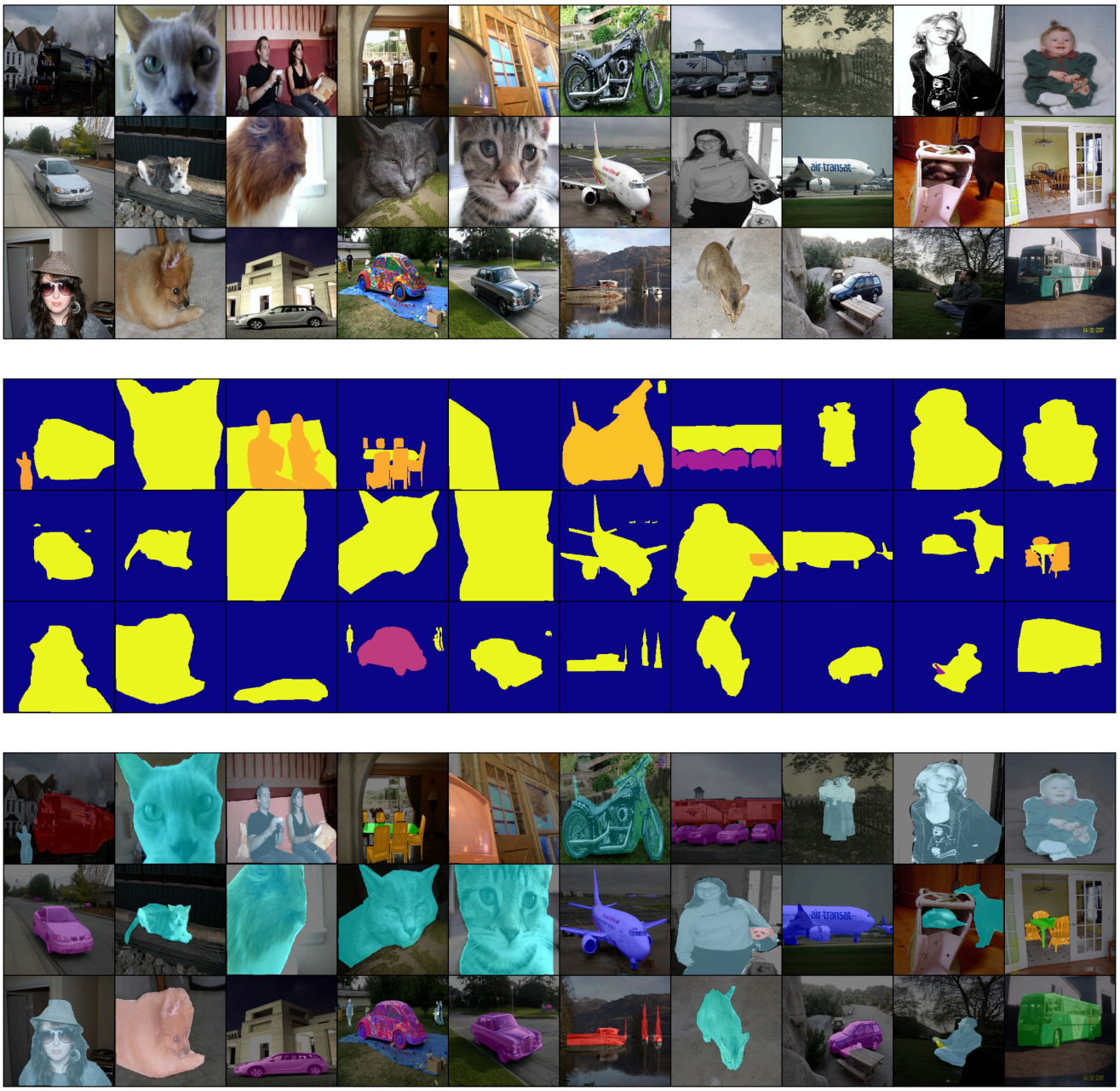

Figure 10. **Pascal VOC visualizations.** The first group of rows displays the images, the second shows the corresponding per-image masks, and the third overlays the masks on the images, aligned by their semantic IDs. These images and their ground truth segmentation maps are used for our tasks, such as visual in-context learning and linear segmentation.

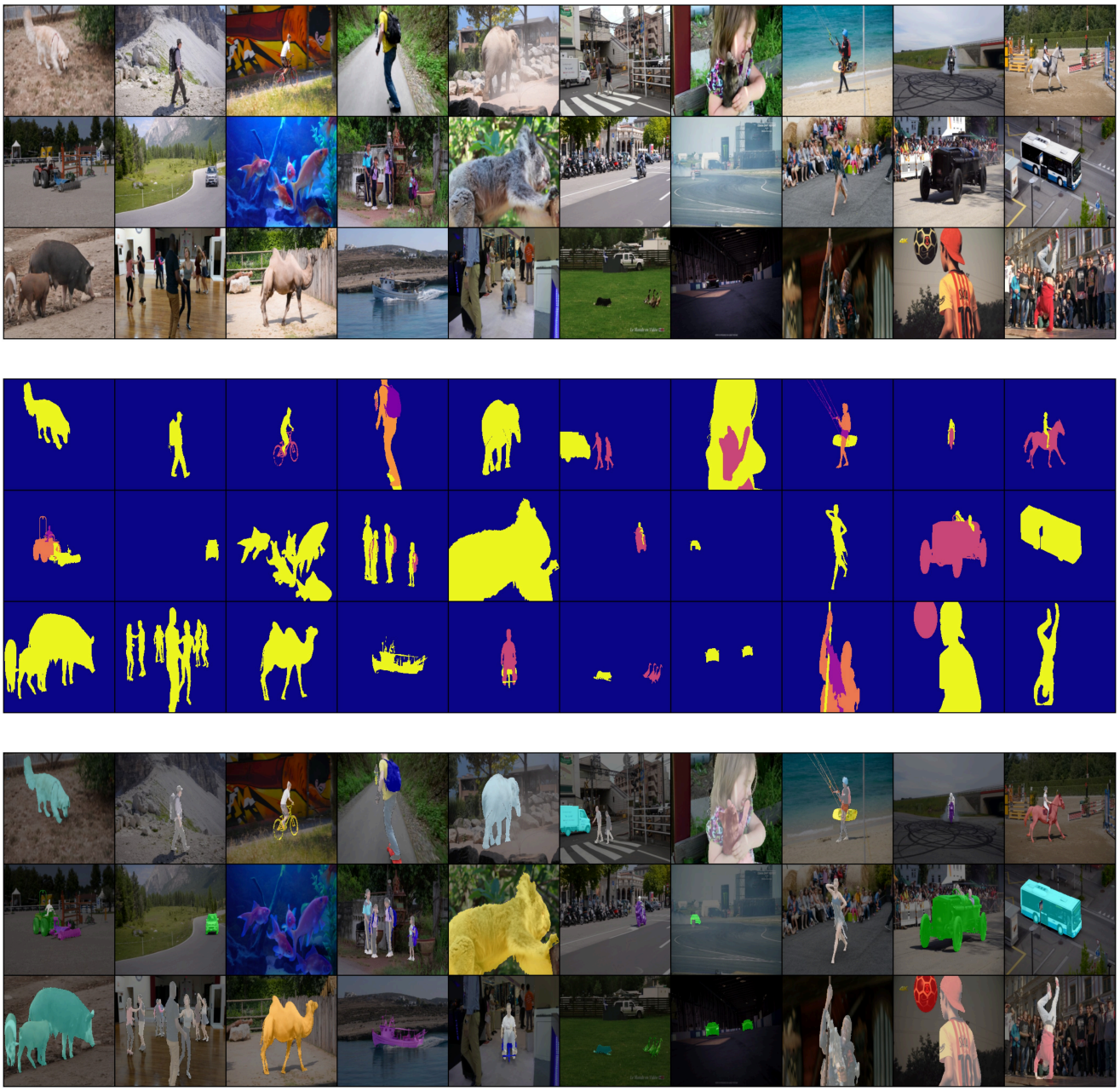

Figure 11. **DAVIS visualizations.** The first group of rows displays the images, the second shows the corresponding per-image masks, and the third overlays the masks on the images, aligned by their semantic IDs. These images and their ground truth segmentation maps are used for our tasks.

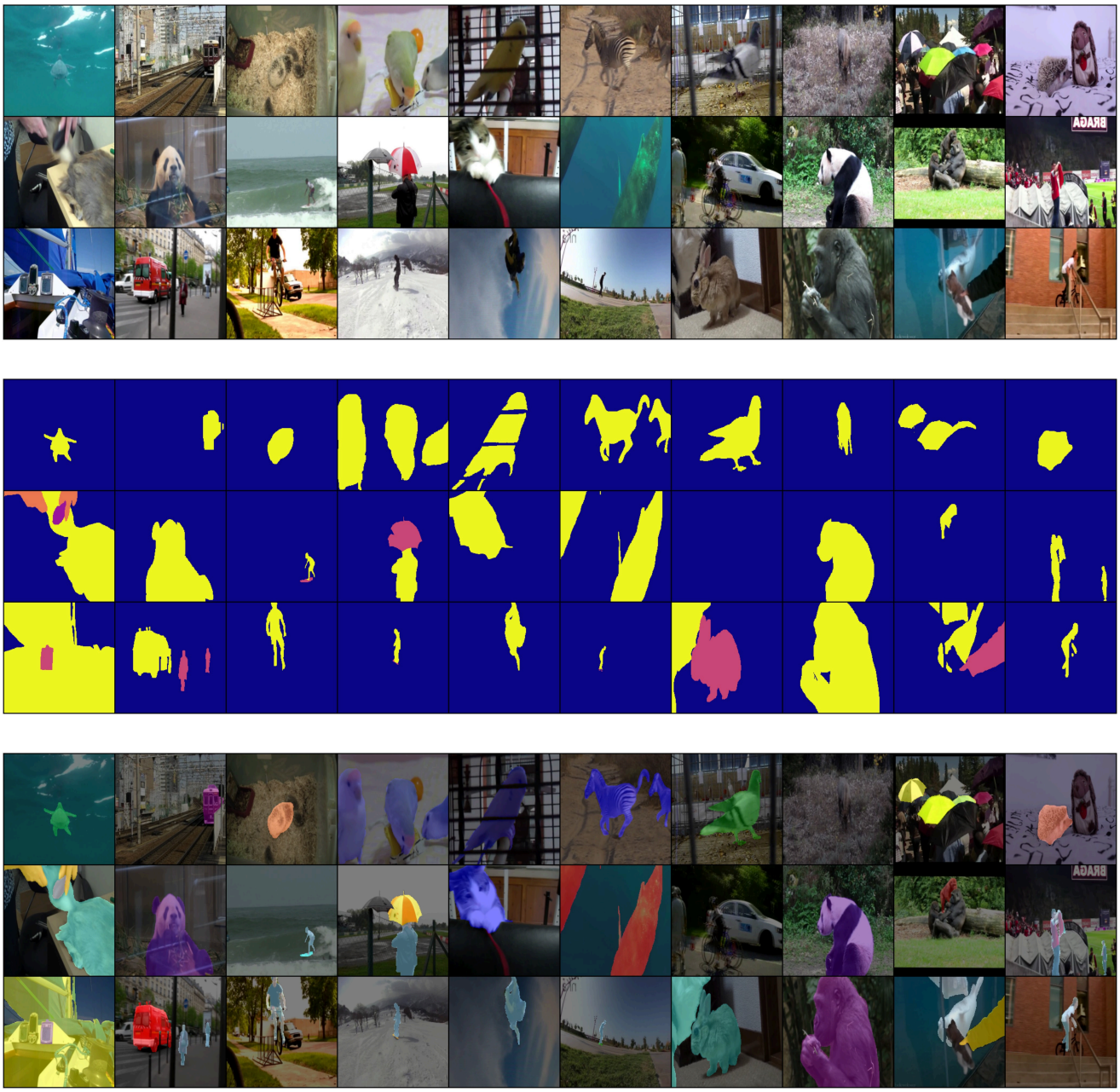

Figure 12. **YouTube VOS visualizations.** The first group of rows displays the images, the second shows the corresponding per-image masks, and the third overlays the masks on the images, aligned by their semantic IDs. These images and their ground truth segmentation maps are used for our tasks.