Chameleon: A Data-Efficient Generalist for Dense Visual Prediction in the Wild

Supplementary Material

This document provides our implementation details and additional results that could not be accommodated in the main paper due to space limitations.

A Meta-Training Details and More Results

This section describes details of meta-training. We first describe the dataset details and then describe the implementation details.

A.1 Datasets

We use six existing datasets to construct our unified meta-training dataset. We summarize the statistics of each dataset in Table A.4.

1. **Taskonomy**: The Taskonomy dataset [68] comprises an indoor scene dataset annotated with various vision task labels. We utilize a small subset consisting of ∼380K images collected from 35 buildings with various camera properties, such as camera pitch, roll, or FoV. Following Kim et al. [24], we select 10 dense prediction tasks, including semantic segmentation, surface normal, Euclidean distance, Z-buffer depth, texture edge, occlusion edge, 2d keypoints, 3d keypoints, reshading, and principal curvature. Note that the 2d and 3d keypoint labels in the Taskonomy dataset are obtained by descriptor-based algorithms [6, 53] and differ from the joint keypoints we describe in the following datasets. Additionally, three single-channel tasks (texture edge, occlusion edge, and Euclidean distance) are pre-processed to multi-channel labels. Since labels of the texture edge task can be generated by a deterministic edge detection algorithm [23], we include the unsupervised task in all the following sub-datasets along with two additional tasks (autoencoding and denoising).

2. **COCO**: The COCO dataset [30] consists of ∼120K images of everyday objects, which contains semantic/instance segmentation annotations of various object categories and human keypoint annotations of 17 keypoint classes. With three unsupervised tasks included in the Taskonomy dataset, we include five types of tasks: semantic segmentation, panoptic segmentation, interactive segmentation, joint-specific human keypoint detection, and joint-agnostic human keypoint detection. For semantic segmentation and panoptic segmentation, we use 80 object categories from COCO 2017 split and five selected categories (tree, wall-concrete, sky-other, building-other, and grass)

from COCO-Stuff [8] split, respectively. For the joint-specific keypoint detection task, we use the human keypoint labels from COCO 2017 split. We also add a joint-agnostic keypoint detection task, whose objective is to predict all human joint locations within an image without distinguishing specific ones. We categorize this task as continuous signal prediction in Table A.4. Finally, we include an interactive segmentation task using the instance segmentation labels from COCO 2017 split, which consists of the same 80 object categories used for the semantic segmentation task. First, we randomly choose $k \in [1, \lfloor K/2 \rfloor]$ object instances of a specific class from each image, where K denotes the total number of instances within the image. Then the model should segment the chosen instances by using the interactive guide given as a second image, where the guide is generated by a Mixture of Gaussian density whose centers are randomly sampled at $p \in [1, 5]$ pixels from each chosen instance.

3. **MidAir**: The MidAir dataset [16] consists of ∼420K aerial video frames recorded in a synthetic environment. It includes two splits (KITE and PLE), each containing videos from multiple trajectories under four distinct weather conditions (sunny, sunset, cloudy, and foggy) and three distinct seasons (spring, fall, winter), respectively. For each weather/season condition, we employ the first three trajectories of the KITE split and the first two trajectories of the PLE split as our training data. The remaining final trajectory from each split serves as our validation data. We select three monocular dense prediction tasks (semantic segmentation, depth estimation, and surface normal) and two binocular dense prediction tasks (stereo depth estimation and stereo surface normal). We use images of a left camera for monocular tasks and images of left and right cameras for binocular tasks. For semantic segmentation, we use eight categories (trees, dirt ground, ground vegetation, rocky ground, boulders, water plane, road, train track) out of twelve, as the remaining four categories occupy a tiny portion of pixels in the entire dataset. We also include three unsupervised tasks as in Taskonomy.

4. **KP-3**: We include three additional datasets consisting of joint keypoint labels. **MPII** [2] dataset consists of ∼25K human images annotated with 16 human joints, whose joint definition differs from the COCO dataset. **DeepFashion** [32] dataset consists of ∼120K fashion images annotated with 8 fashion landmarks as joints. Finally, **FreiHand** [70] dataset consists of ∼130K hand images captured from 32 subjects, annotated with 21 hand joints. Similar to the COCO dataset, we both include joint-specific and joint-agnostic keypoint detection tasks in all of the three keypoint-specific datasets, as well as the unsupervised tasks. In Table 3, we refer to this dataset together with the COCO keypoint dataset as KP-4.

Table A.4: Statistics of six datasets contained in the meta-training dataset. When counting the number of tasks, we consider different channels of a multi-channel task as different tasks. For example, the number of segmentation tasks corresponds to the number of classes, and the number of joint-specific keypoint tasks corresponds to the number of joints. Numbers in parentheses denote the number of task groups obtained by considering different channels of a multi-channel task and tasks from different source datasets as a single group.

Dataset	# Images	# Tasks				
		Segmentation			Continuous Signals	
		semantic	panoptic	interatctive	monocular	binocular
Taskonomy	381,916	12 (1)	0	0	19 (8)	0
COCO	118,287	80 (1)	5 (1)	80 (1)	1 (1)	0
MidAir	423,676	8 (1)	0	0	4 (2)	4 (2)
MPII	24,984	0	0	0	1 (1)	0
DeepFashion	123,016	0	0	0	1 (1)	0
FreiHand	130,240	0	0	0	1 (1)	0
Total	1,202,119	100 (1)	5 (1)	80 (1)	27 (8)	4 (2)

Dataset	# Tasks				Total
	Keypoints	Low-Level			
	joint-specific	autoencoding	denoising	texture edge	
Taskonomy	0	3 (1)	3 (1)	3 (1)	40 (12)
COCO	17 (1)	3 (1)	3 (1)	3 (1)	192 (8)
MidAir	0	3 (1)	3 (1)	3 (1)	25 (8)
MPII	16 (1)	3 (1)	3 (1)	3 (1)	26 (5)
DeepFashion	8 (1)	3 (1)	3 (1)	3 (1)	18 (5)
FreiHand	21 (1)	3 (1)	3 (1)	3 (1)	31 (5)
Total	62 (1)	18 (1)	18 (1)	18 (1)	332 (17)

A.2 Implementation Details

Task Sampling. We train Chameleon with 400,000 episodic training iterations. At each iteration of meta-training, we construct 16 episodic batches, where 8 of them consist of a uni-modal task and the remaining consist of a bi-modal task sampled from the unified dataset. Then we use either batch for computing the loss at each iteration, where the batch is randomly chosen with probability proportional to the number of uni-modal and bi-modal tasks (246 : 84). Both episodic batches are sampled as the following procedure.

1. First, we sample the type of tasks, which is one of categorical (both segmentation and joint-specific keypoints), continuous, and low-level, with a sampling rate $3:3:1$.
2. Second, we sample the source dataset, where the sampling rate is proportional to either the number of tasks (for categorical and continuous tasks) or the number of images (for low-level tasks) included in each dataset.
3. Third, we uniformly sample four tasks from the task pool filtered by the chosen task type and source dataset, where multi-channel tasks are disassembled to separate single-channel tasks.
4. Finally, we sample four support pairs and four query pairs from the source dataset with the selected task, thus simulating a 4-shot learning episode with four queries for each task.

Loss Function. We use three different loss functions for meta-training: L1 loss, binary cross-entropy (bce) loss, and spatial softmax loss. We use L1 loss as default while using bce loss for segmentation tasks and spatial softmax loss for joint-specific keypoint detection tasks. Spatial softmax loss is computed by first applying the softmax function on the prediction along the spatial axis (both horizontal and vertical), then applying the binary cross-entropy loss. We normalize the spatial softmax loss by the sum of the target label and do not use any other hyper-parameters for weighting three loss functions.

Task-Specific Parameters. Following VTM [24], we use task-specific bias parameters of the image encoder for each task, which results in a total of 332 bias sets for meta-training. In Section 3.1 and Section 3.2, we have introduced the position bias and the feature re-weighting matrix, which are also task-specific. During meta-training, we share the position bias for all tasks and fine-tune it task-specifically for downstream tasks with multi-modal inputs. We share the feature weighting matrix among tasks that originate from the same multi-channel task (*e.g.,* three channels of surface normal) since they would require similar correspondence between image and label features. We also use the shared matrix for semantic segmentation and panoptic segmentation in COCO. This results in 42 matrices of size 4×4, where we use $L = 4$ feature levels. In Figure A.11, we plot the learned feature weights averaged on all tasks and tasks within each task category in the meta-training dataset. It clearly shows that different task groups require different feature correspondences, where low-level tasks mainly require low-level features while the other tasks require both low-level and high-level features.

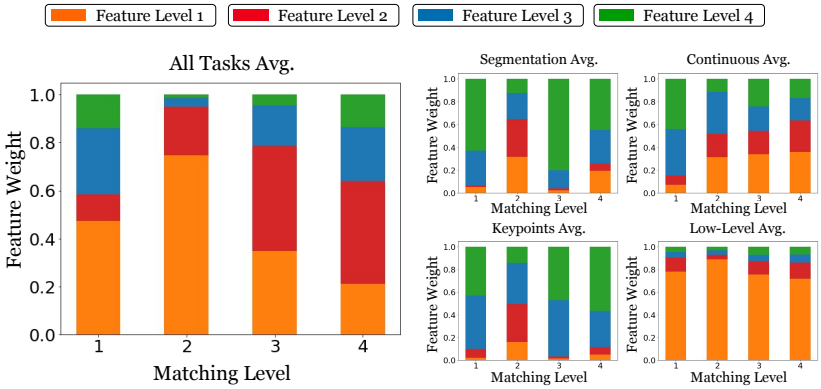

Fig. A.11: Learned feature weights in training tasks.

B Downstream Details and More Results

In this section, we describe the detailed settings of six downstream tasks and provide additional qualitative results for each task.

B.1 AP-10K

Detailed Settings. We fine-tune and evaluate our model on the AP-10K train and test split, respectively. Since the ground-truth bounding box labels are provided in the dataset, we center-crop the images and labels, then resize them to 256×256 resolution. During fine-tuning, we apply a random crop of crop size 224×224 on the resized data. For evaluation, we further resize the data to 224×224 to obtain the prediction, then translate the keypoint locations back to the original resolution. We use the spatial softmax loss described in Section A.
Additional Results. In Figure A.12, we provide additional qualitative results on AP-10K. We can observe that Chameleon successfully adapts to different animal species with distinctive appearances and joint configurations.

B.2 LineMOD

Detailed Settings. We fine-tune and evaluate our model on the conventional train and test split of the LineMOD dataset following literature [28, 67]. As discussed in Section 5, we formulate a 6D pose estimation task as dense prediction by predicting correspondence between each image pixel and the vertex of the CAD model, from which 6D pose is obtained by Perspective-n-point algorithm [26]. To predict the 2D-3D correspondence, we render a 3-channel texture map on all images using the 6D extrinsic camera matrices, where the channels correspond to the X, Y, and Z axes of the relative 3D position of CAD vertices which are normalized to $[-1, 1]$. Then we use the rendered texture maps as dense labels to be predicted by Chameleon. Since the 2D-3D correspondence is defined on the object area, we augment the dense label with a foreground segmentation channel, which represents a non-zero region of the texture maps, and let Chameleon also predict it. We use bce loss to train the segmentation channel and L1 loss to train the texture channels. During fine-tuning, we apply random rotation, random jittering, and random Gaussian blur as data augmentation as well as random cropping with crop size 224×224.

Image Cropping with Object Segmentation. Since the objects occupy a small region of the full image, Chameleon first predicts the object location to crop the images and then predicts the 6D pose in the cropped region as described above. To obtain the object location, we perform an additional fine-tuning stage for foreground segmentation on the full-sized images before predicting texture maps. Then we get the bounding box of the object from the predicted segmentation mask. To fine-tune the object segmentation, we resize the full images and labels to 256×256 and apply random cropping with crop size 224×224, with data augmentation applied in the texture fine-tuning stage. Note that we generate the

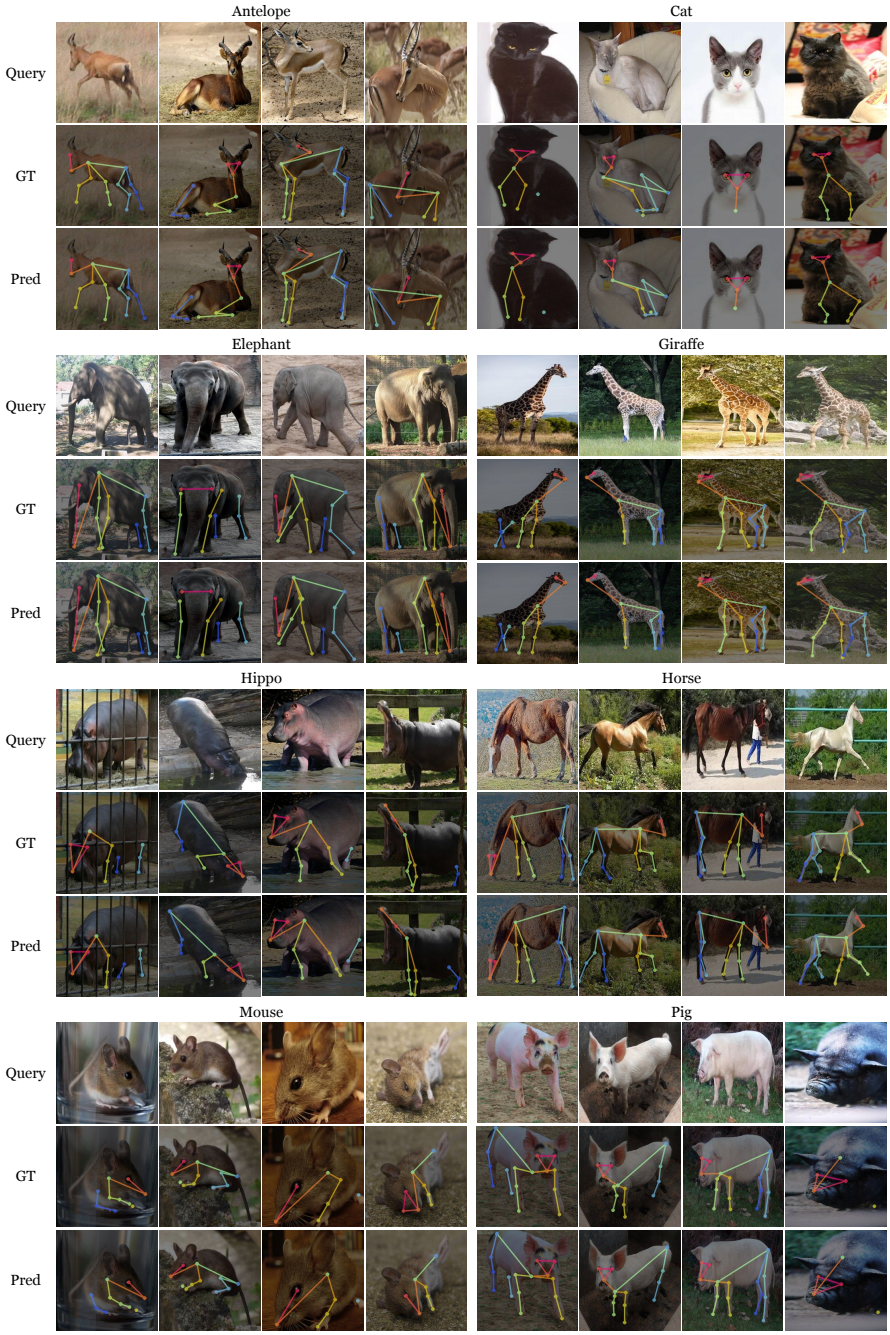

Fig. A.12: Additional qualitative results on AP-10K.

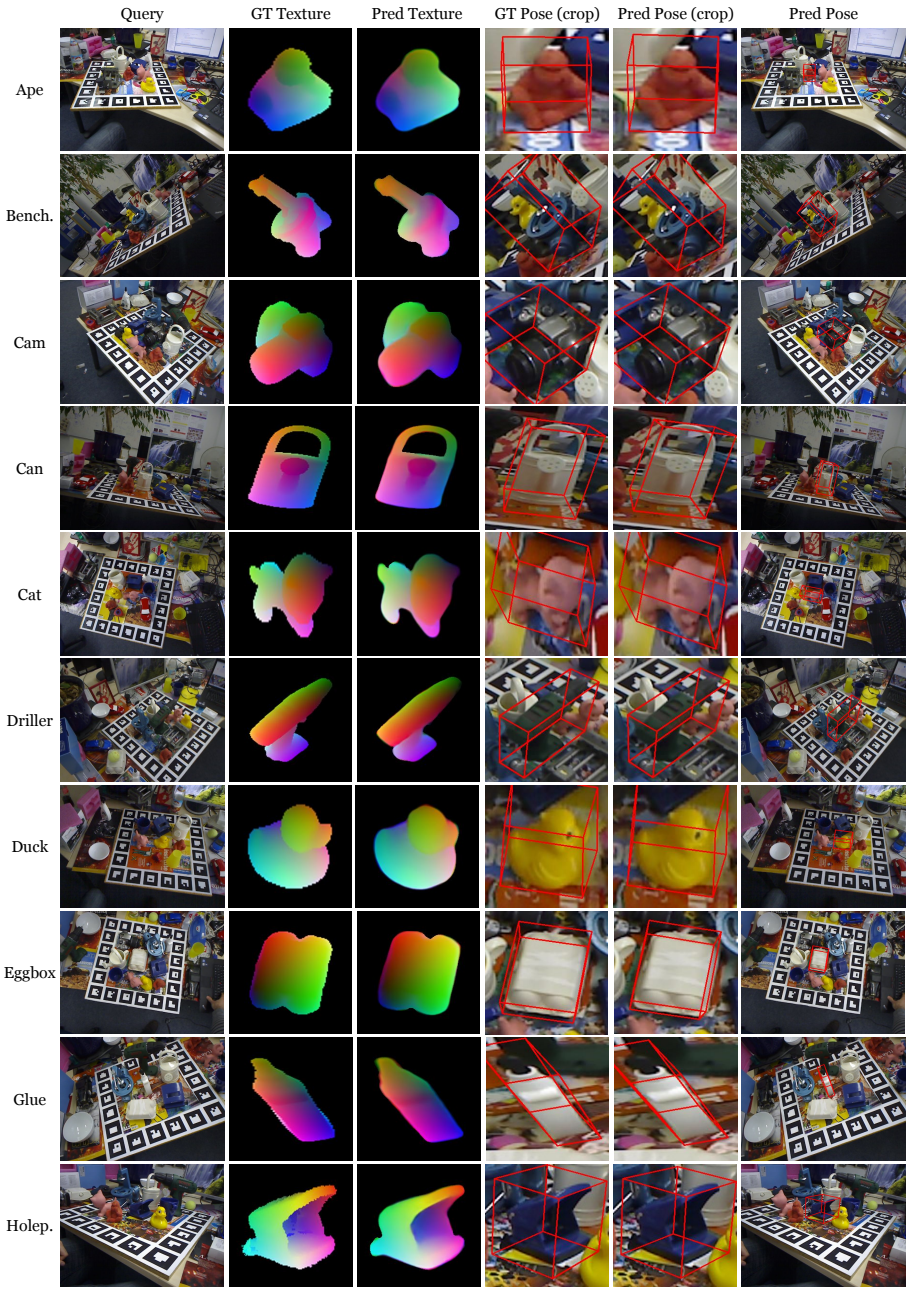

Fig. A.13: Additional qualitative results on LineMOD on ten object classes: Ape, Benchviseblue, Cam, Can, Cat, Driller, Duck, Eggbox, Glue, and Holepuncher.

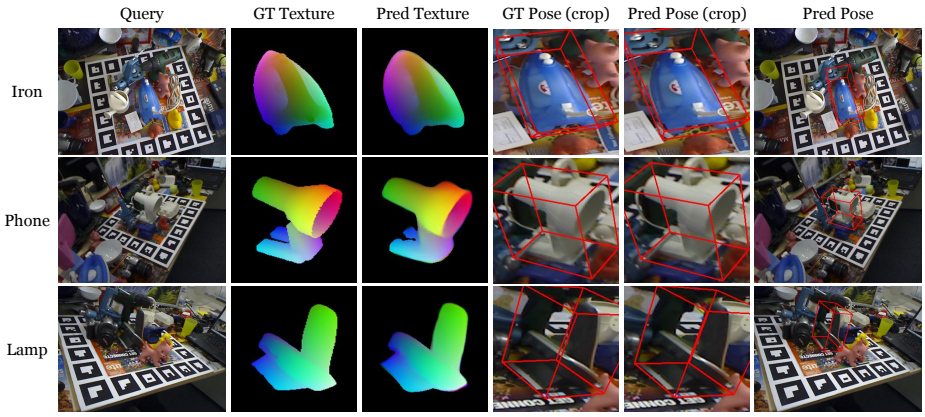

Fig. A.14: Additional qualitative results on LineMOD on Iron, Lamp, and Phone.

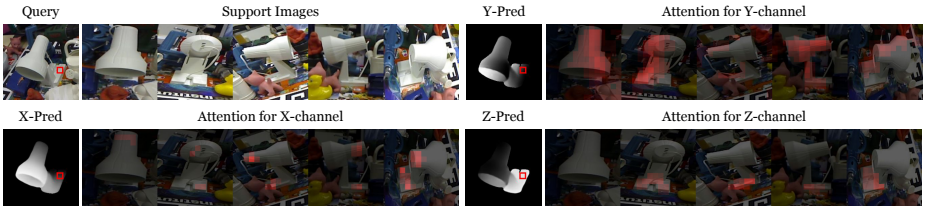

Fig. A.15: Visualization of attention maps between a query (red box in the first image) and support patches in 6D pose estimation. Chameleon captures 3D relationship by attending back, left, and bottom parts in X, Y, and Z channels, respectively.

segmentation labels from the texture map labels contained in the support set (*i.e.*, non-zero region of the texture maps), Chameleon does not use additional supervision for the object detection procedure. After obtaining the bounding box, we center-crop the images and labels and resize them to 256×256.

Additional Results. In Figure A.13 and Figure A.14, we provide additional qualitative results on LineMOD. We visualize the original query image, texture maps, and 6D pose on the cropped region (both ground truth and prediction), and predicted 6D pose on the original image. We observe that Chameleon accurately predicts various 6D poses of all objects. Figure A.15 visualizes the attention maps on this task, which shows that our model successfully captures the 3D relationship required to solve the task.

B.3 ISIC 2018

Detailed Settings. We follow the standard protocol of the literature [19,56] for fine-tuning and evaluation: (1) we perform 5-fold cross-validation on ISIC 2018 train split and report the mean F1 score, and (2) resize the data to 512×512 resolution for evaluation. During fine-tuning, we resize the data to 448×448

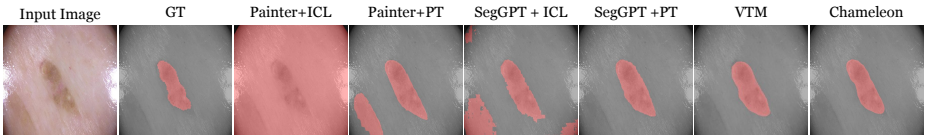

Fig. A.16: Out-of-domain medical semantic segmentation results of generalist models.

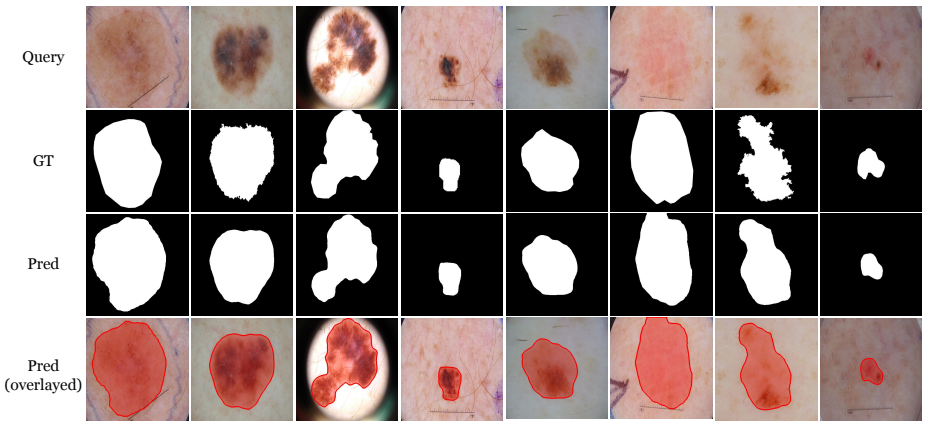

Fig. A.17: Additional Qualitative Results on ISIC 2018.

resolution and apply random cropping with crop size 384×384. During inference, we first resize the data to 384×384 to obtain the prediction, then upsample the prediction to 512×512 resolution for evaluation. We use bce loss for fine-tuning.

Additional Results. In Figure A.16 and Figure A.17, we provide qualitative results on ISIC 2018. We observe that Chameleon accurately segments the lesion boundary, even for ambiguous regions like the first and sixth query examples in the figure. We also note that additional fine-tuning still improves the in-context learning baselines (Painter+PT and SegGPT+PT).

B.4 DAVIS 2017

Detailed Settings. We fine-tune and evaluate our model on the DAVIS 2017 validation split which consists of 30 videos. Notably, we do not use the videos in the train split of DAVIS 2017, unlike the video object segmentation literature [12, 59]. During fine-tuning, we resize the video frames and labels to 448×448, then apply random cropping with crop size 384×384. During inference, we first resize the data to 384×384 to obtain the prediction, then upsample the prediction to the original resolution for evaluation. We treat different instances as different tasks (*e.g.,* we fine-tune five independent task-specific parameters for videos containing five instances). We use cross-entropy loss over predictions on all object instances within the video, where the logits for the background are fixed to zero when computing the loss.

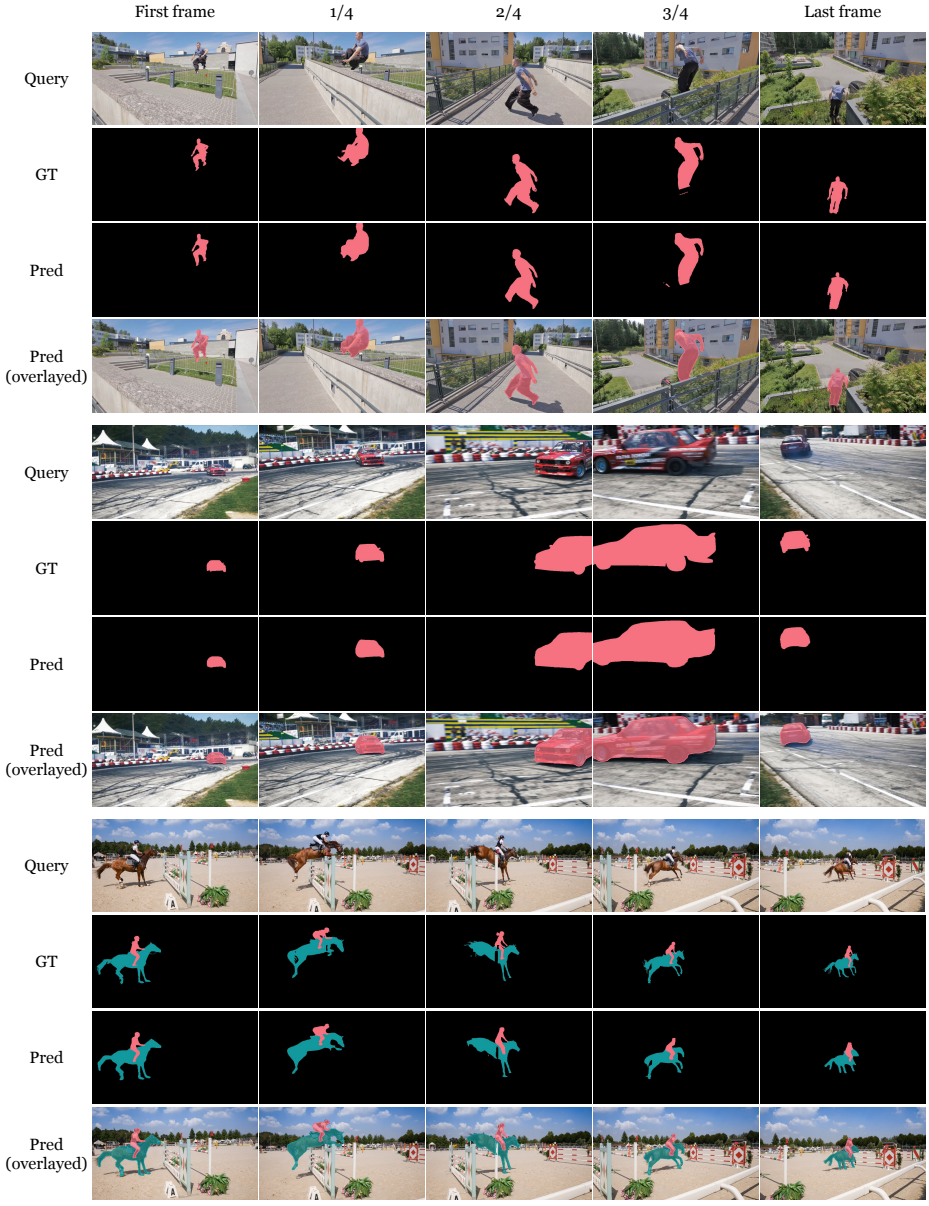

Fig. A.18: Additional qualitative results on DAVIS 2017: parkour, drift-straight, and horse-jump.

Fig. A.19: Additional qualitative results on DAVIS 2017: motocross-jump and shooting.

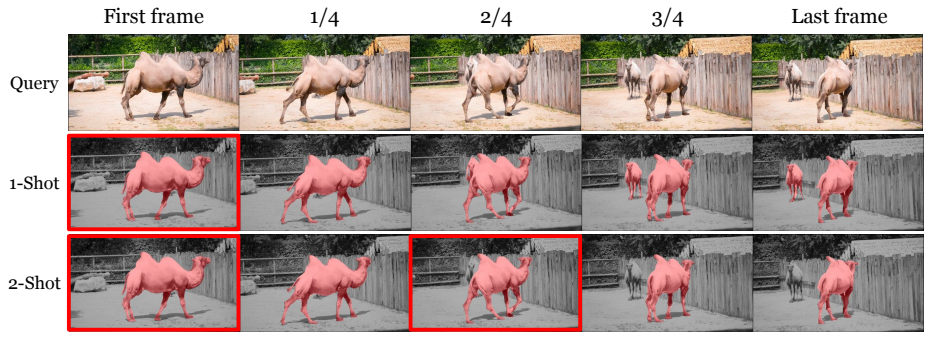

Fig. A.20: Video segmentation results with one and two support frames (red boxes).

Fig. A.21: Additional Qualitative Results on FSC-147.

Additional Results. In Figure A.19 and Figure A.18, we provide additional qualitative results on DAVIS 2017. We can observe that Chameleon accurately segments diverse videos within the benchmark. As discussed in Section 5, Chameleon is robust to the variations in object appearance and the camera view, without using any temporal prior. In Figure A.20, we show the qualitative results of "camel" class, where one of the two camel instances occurs after several frames. With only the first frame (1-shot setting), our model struggles to distinguish two instances, while this can be resolved by just giving an additional frame (2-shot setting).

B.5 FSC-147

Detailed Settings. We fine-tune and evaluate our model in the train and test split of the FSC-147 dataset, respectively. We convert the object counting to Gaussian density prediction following the literature [9, 13]. After predicting the density map, we detect the modes and use the number of modes as count prediction. As an exception, for outliers whose sum of the predicted density map is more than 3,000, we use the sum of the density map as count prediction. To generate the exemplar guide, we copy and paste the exemplar patches to a black image using their bounding boxes, as shown in Figure A.21 We randomly scale the patch size and paste each patch multiple times to maximize the augmentation effect. For fine-tuning, we first resize the data to 592×592 and apply random cropping of crop size 512×512 during fine-tuning. During inference, we process the images differently depending on the average size of exemplar patches following [9]. For images whose average size of exemplar patches is less than the threshold (13.33 pixels for image size 512×512), we first resize the

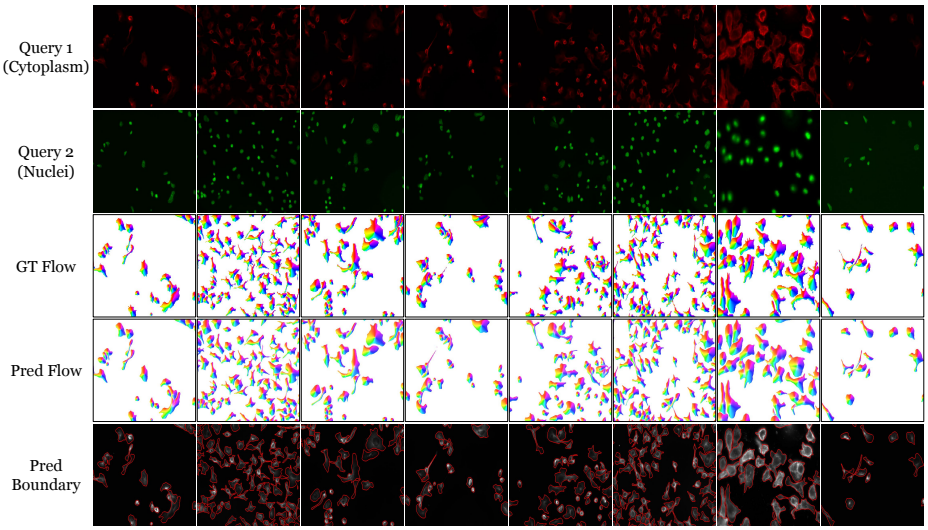

Fig. A.22: Additional Qualitative Results on Cellpose for bi-modal images. We use red and green color codings for the cytoplasm and nuclei images, respectively.

data to 1536×1536 and crop the image into 9 non-overlapping patches of size 512×512, obtain the predictions separately, then merge the predictions. For the other images, we resize the data to 512×512. We use MSE loss for fine-tuning.
Additional Results. In Figure A.21, we provide additional qualitative results on FSC-147. We observe that Chameleon accurately predicts the density map on the query objects by effectively exploiting the exemplar guide.

B.6 Cellpose

Detailed Settings. We fine-tune and evaluate our model in the train and test split of the Cellpose dataset, respectively. Following [54], we formulate a cell instance segmentation task by flow estimation with foreground mask segmentation. We generate a 2-channel flow map where each channel corresponds to vertical and horizontal gradients of each cell towards its center. Then Chameleon predicts the flow map and also a binary segmentation mask to segment all foreground cells, from which we can obtain the instance segmentation mask. We resize the images and labels to 256×256 and apply random resized cropping of crop size 224×224 and scale between 0.75 and 1.25 during fine-tuning. During inference, we first divide an image into overlapping tiles with the size of 224×224 and 50% overlap, then ensemble each prediction by multiplying it with a Gaussian kernel to minimize edge effects. For each tile, we additionally use test-time augmentation where each prediction is obtained by the ensemble of 4 flipped inputs following [54]. We use bce loss for the segmentation channel and L1 loss for the flow channels. We apply random flipping as data augmentation together with random resized cropping.

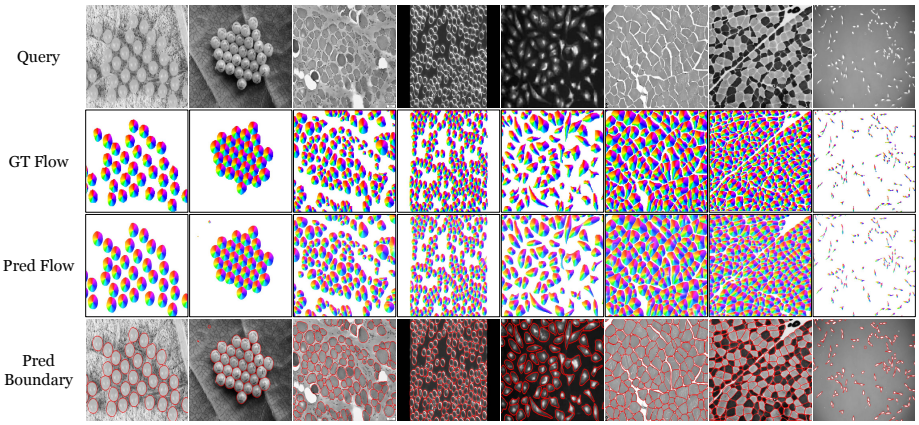

Fig. A.23: Additional Qualitative Results on Cellpose for uni-modal images. We use grayscale color codings for the cytoplasm images.

Additional Results. In Figure A.22 and Figure A.23, we provide additional qualitative results on Cellpose. We observe that Chameleon accurately predicts the cell boundaries for both bi-modal and uni-modal images.

Table A.5: Ablation study on the image encoder backbone.

Backbone	AP-10K	LineMOD	ISIC 2018	DAVIS 2017	FSC-147	Cellpose
	AP ↑	ADD ↑	F1 ↑	$\mathcal{J}\&\mathcal{F}$ ↑	MAE ↓	AP_{50} ↑
ViT [14]	36.3	66.7	85.5	64.9	26.5	59.9
DINOv2 [38]	63.5	85.1	87.5	70.4	20.1	69.0
BEiTv2 [40]	**67.2**	**85.2**	**88.5**	**77.5**	**12.3**	**70.3**

C Additional Experiments

In this section, we conduct ablation studies on the image encoder backbone and image resolution to analyze their effect on downstream performance. We also provide an additional comparison between VTM and qualitative results in a complex semantic segmentation setting.

C.1 Ablation Study on Image Encoder Backbone

Since the image encoder plays a central role in the matching architecture of Chameleon, it is important to leverage a pre-trained image encoder backbone as initialization for meta-training. To analyze the effect of the image encoder backbone, we compare three different pre-trained transformers: BEiTv2 Large [40] (default setting), ViT Large [14], and DINOv2 Large [38]. BEiTv2 and DINOv2 are pre-trained with self-supervised learning objectives, while ViT is pre-trained with image classification. Also, DINOv2 Large is distilled from the DINOv2 Giant backbone, which is trained on a large-scale dataset containing ImageNet-22k. In Table A.5, we report the performance of Chameleon with three different backbones. We note that BEiTv2 achieves the best performance, while ViT shows inferior performance compared to the self-supervised learning approaches. This can be attributed to the generality of self-supervised features compared to image classification features which may not have fine-grained information for dense prediction.

C.2 Ablation Study on Image Resolution

We also conduct an ablation study on the image resolution. Since we use a larger resolution for three downstream benchmarks (ISIC 2018, DAVIS 2017, and FSC-147) compared to the meta-training (224×224), we analyze the effect of increasing the resolution. We observe that using larger image resolution improves the performance at DAVIS 2017 and FSC-147 to a great extent while having a statistically non-significant effect on ISIC 2018.

Table A.6: Ablation study on the image resolution. Larger resolution corresponds to input image size 384×384 for ISIC 2018 and DAVIS 2017, and 512×512 for FSC-147.

Resolution	ISIC 2018	DAVIS 2017	FSC-147
	F1 ↑	$\mathcal{J}\&\mathcal{F}$ ↑	MAE ↓
224×224	**88.6** ± 1.20	65.7	32.2
Larger Resolution	88.5 ± 0.75	**77.5**	**12.0**

Table A.7: 10-shot learning performance at Taskonomy dataset.

Model	Tasks	
	SS (mIoU ↑)	SN (mErr ↓)
VTM	0.4097	11.4391
Chameleon	**0.4278**	**9.9348**

C.3 Direct Comparison with VTM

To further explore the improvement of Chameleon upon VTM, we directly compare them in the Taskonomy dataset as described in [24]. We use two held-out tasks (semantic segmentation and surface normal), isolating feature re-weighting as the only difference since there are no multi-input tasks. Table A.7 shows clear improvements in both tasks, proving the effectiveness of feature re-weighting.

C.4 Qualitative Results on ADE20K

Finally, we provide qualitative results of more complex semantic segmentation in ADE20K dataset [69]. As a demonstration, we selected two urban scenes from the ADE20K dataset, which contain many salient objects and semantic classes. Figure A.24 shows 20-shot semantic segmentation results using 20 training images per class. Our model segments various unseen objects (*e.g.,* window, headlight, license plate, wheel) reasonably well. In the second example, it even produces complex masks for several intertwined classes (tree, building, window, sky).

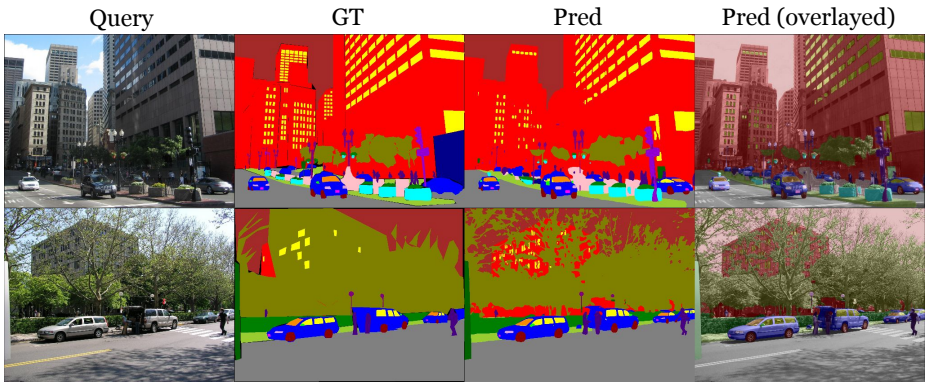

Fig. A.24: 20-shot semantic segmentation results in ADE20K.

D Limitation

Despite its strengths, Chameleon has several limitations. One significant issue is the need for many fine-tuning iterations to achieve optimal performance on new tasks, which can be computationally intensive and time-consuming depending on the application (*e.g.,* real-time services). The fine-tuning stage can still lead to overfitting on few-shot data and requires careful parameter tuning for each new task. Also, the model's reliance on a diverse and extensive meta-training dataset means it may struggle with tasks that have significantly different data distributions not represented in the training data. Finally, overall architecture relies on attention mechanisms that require quadratic complexity, which can be costly on extremely high-resolution images, while these can be alleviated by more efficient implementations.