Supplementary - SegGPT: Towards Segmenting Everything In Context

A. Additional Implementation Details

Training. We use various segmentation datasets during training. The sampling weight for each dataset is 0.22 (COCO instance), 0.15 (ADE20K semantic), 0.15 (COCO panoptic semantic), 0.07 (Cityscapes semantic), 0.07 (COCO stuff semantic), 0.07 (LIP person semantic), 0.07 (PASCAL VOC semantic), 0.07 (PACO semantic), 0.06 (iSAID and loveDA aerial semantic), and 0.06 (CHASE_DB, DRIVE, HRF and STARE retinal vessel). For semantic segmentation data, we use a probability of 0.5 for using the transformation of the input image as the in-context examples and then conduct random color selection. For instance segmentation data, the probability is 1.0, *i.e.*, we always use two transformed views of the same image as the in-context pair. Almost all the segmentation sub-tasks can be grouped into two types, *i.e.*, to segment a category or an instance (not limited to objects). To avoid the ambiguity between category and instance, we initialize two learnable embeddings which are associated with category-level and instance-level coloring tasks respectively.

Evaluation. For quantitative evaluation on the existing benchmarks, the examples are either from the support samples, the training set, the first frame in a video, or a learned prompt. Take ADE20K semantic segmentation as an example. Given a tuned prompt, we directly stitch the prompt with each test image to obtain the predictions. Without the tuned prompt, for each category, we randomly sample several images from the training set which contain that category. These examples are used together via context ensemble to obtain the predictions for this category across all test images.

B. Additional Results

ADE20K semantic segmentation. In Table S1, we provide the example-based semantic segmentation results on ADE20K. Different from the in-context tuning, we only randomly select several samples in the training set as examples, and use `Feature Ensemble` to ensemble the examples. Specifically, for each category, we randomly sample without replacement from all images with that category. Since the selection of the examples can affect performance, we sample with different random seeds {1000, 2000, 3000, 4000} and report the best results. We can see that more examples significantly boost the performance, *e.g.*, +13.1% mIoU from 1 to 16 examples, although there is still a gap with the tuned prompt. These experiments inspire us to explore in the future what makes good examples and how many examples we need to approach the results of in-context tuning.

Context ensemble. Here we qualitatively demonstrate the effectiveness of our context ensemble approach in Figure S1. Given a video clip and its first annotated frame, it is difficult to distinguish the instances in a crowd when using only the first frame as an example. With the context ensemble of several previous frames and their pseudo-labels, SegGPT segments each object successfully.

Visualizations. We provide more visualizations in Figure S3, including semantic segmentation on ADE20K, instance segmentation on COCO, and arbitrary segmentation in the wild.

examples	mIoU	mAcc
1	18.8	27.4
2	25.0	34.4
4	28.3	37.7
8	30.1	38.9
16	31.9	40.4
32	33.0	42.0
tuned	39.6	50.7

Table S1: Example-based results on ADE20K semantic segmentation. More examples boost the performance.

(a)

(b)

Figure S1: Context ensemble helps segment objects across frames. a) Incorrect predictions for objects in a crowd when only the first frame is used as the example. b) Correct predictions using `Feature Ensemble` with previous frames.

Results on medical imaging. As shown in Table S2, we evaluate SegGPT on out-of-domain medical imaging dataset

CVC-ClinicDB via few-shot inference. The results demonstrate that our model outperforms Painter on out-of-domain medical image segmentation data, approaching the performance of specialist models trained on medical data. In Figure S2, we also visualize the results of SegGPT on medical imaging data.

method	mIoU
ResUNet++ [2]	79.6
FCB-SwinV2 [1]	82.6
Painter	12.5
SegGPT	76.3

Table S2: Medical imaging results on CVC-ClinicDB.

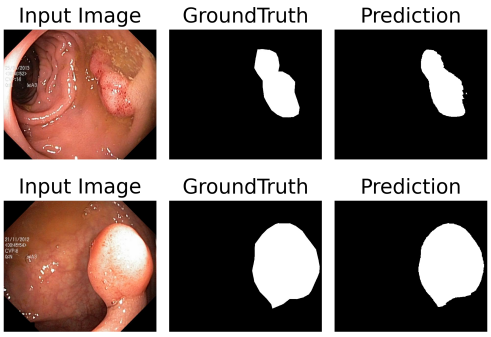


Figure S2: Medical imaging visualizations.

Results on smaller backbone. We train SegGPT with a smaller ViT-B backbone and report the FSS-1000 result in Table S3. When the parameter count was reduced from 307 million to 87 million, our performance on the FSS-1000 dataset decreased from 85.6 to 81.1. Given the notable decrease in parameter count, this drop in performance is acceptable.

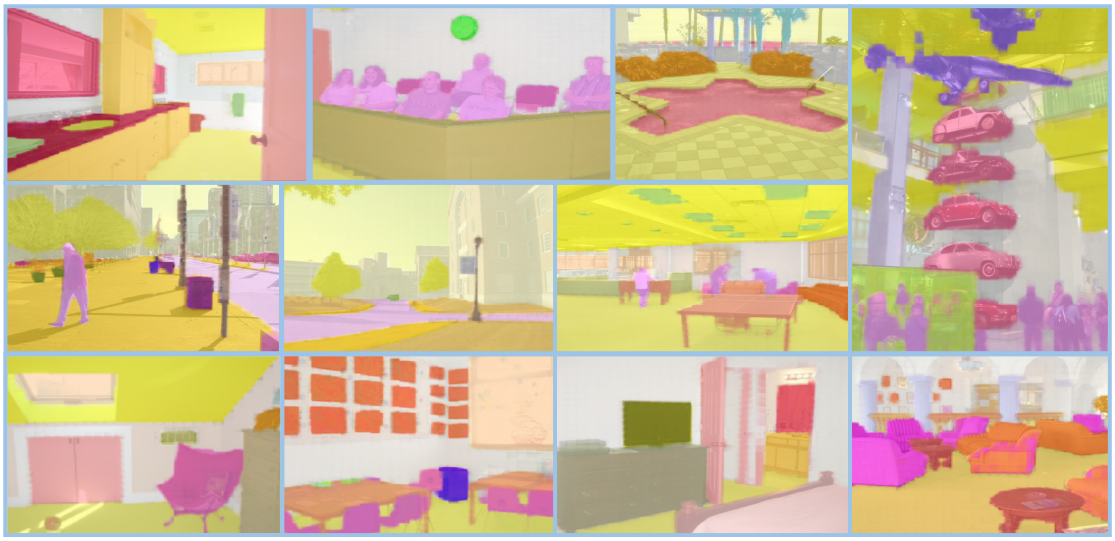
backbone	parameters	FSS-1000 mIoU
ViT-L	307M	85.6
ViT-B	87M	81.1

Table S3: Comparison of different-scale backbones.

References

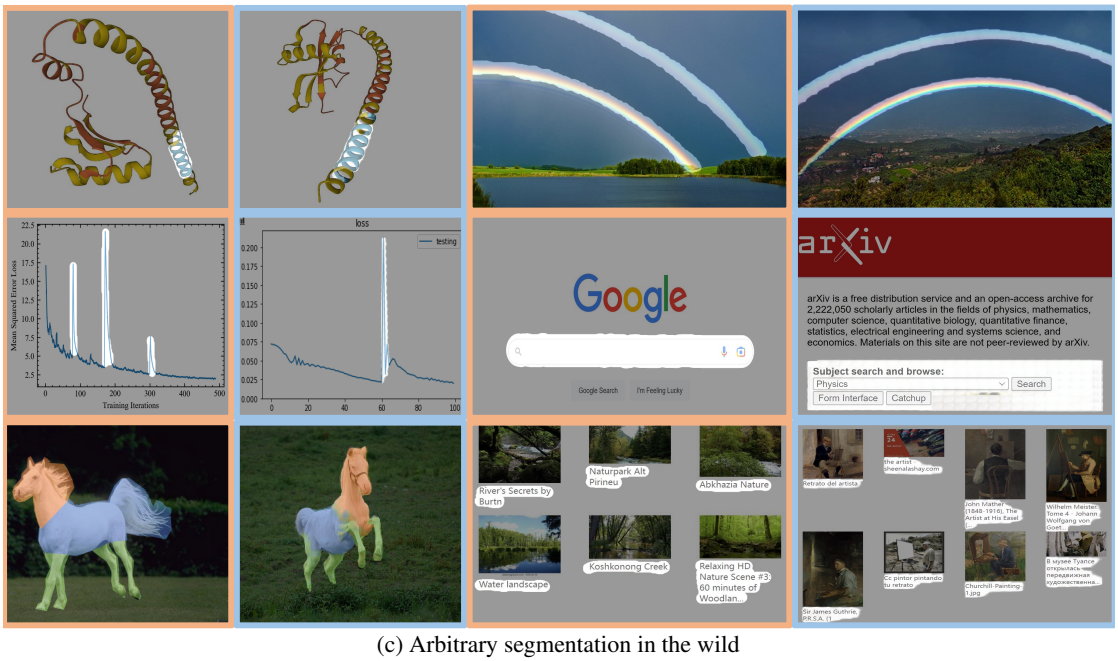
[1] Kerr Fitzgerald and Bogdan J. Matuszewski. Fcb-swinv2 transformer for polyp segmentation. *ArXiv*, abs/2302.01027, 2023. 2

[2] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Dag Johansen, Thomas De Lange, Pål Halvorsen, and Håvard D Johansen. Resunet++: An advanced architecture for medical image segmentation. In *2019 IEEE International Symposium on Multimedia (ISM)*, 2019. 2



(a) Semantic segmentation on ADE20K

(b) Instance segmentation on COCO



(c) Arbitrary segmentation in the wild

Figure S3: More examples of SegGPT applications. Each test image and the corresponding predicted segmentation are combined for better visualization. For (c), the orange box □ on the left displays the example/prompt image and its corresponding mask, while the blue box □ on the right shows the input image and the resulting mask output.