Lite-SAM Is Actually What You Need for Segment Everything

Jianhai Fu[1,*], Yuanjie Yu[1,2,*], Ningchuan Li[1,†], Yi Zhang[1], Qichao Chen[1], Jianping Xiong[1], Jun Yin[2], and Zhiyu Xiang[2,†]

[1] Zhejiang Dahua Technology Co., Ltd., Hangzhou, China
[2] Zhejiang University, Hangzhou, China

A More quantitative and qualitative evaluation and results

A.1 Image Classifcation: serving as a pre-trained model

To understand the effectiveness of LiteViT backbone in image classifcation, we train our models on ImageNet following the standard training strategy. We summarize the results and compare our models with SOTA image classifcation models in Tab. 1. We have demonstrated that our LiteViT achieves an optimal balance of performance and accuracy in classification, establishing a new state-of-the-art (SOTA) standard. LiteViT achieves an optimal balance between performance and accuracy. With only 1.16M parameters and 1.2G computational cost, it rivals the accuracy of models with over 20M parameters, even at a size of 224. This remarkable feat demonstrates the efficiency and effectiveness of LiteViT.

A.2 Class-wise comparative analysis of Lite-SAM with other SAM models

As a supplement to Section 4.5 (Table 5), we have compared the performance of our approach on COCO across 80 object classes with other three SAM architectures in Tab. 2. This comparison showcases the competitive results of Lite-SAM in relation to other SAM models.

*Equal contribution.
†Corresponding author.

Table 1: LiteViT Performance on ImageNet Classification. 1) All these models are only trained on the ImageNet-1K training set and the accuracy on the validation set is reported. RSB-ResNet means the results are from "ResNet Strikes Back".

Models	Top1 Acc(%) ↑	Params(M) ↓	MACs(G) ↓	Input Size
RSB-ResNet-18 [5, 16]	70.6	12	1.8	224
RSB-ResNet-34 [5, 16]	75.5	22	3.7	224
RSB-ResNet-50 [5, 16]	79.8	26	4.1	224
MoblieViT-S [11]	78.4	6	1.5	256
TinyViT-5M [17]	79.1	5.4	1.3	224
GLiT-Tiny [2]	76.3	7	1.5	224
ViTAS-DeiT-A [12]	75.5	6	1.3	224
PoolFormer-S12 [18]	77.2	12	1.8	224
EfficientViT [1]	82.7	24	2.1	256
CoAtNet-0 [3]	81.6	25	4.2	224
ConvNeXt-T [10]	82.1	29	4.5	224
DeiT-S [14]	79.8	22	4.6	224
PVT-Tiny [15]	75.1	13	1.9	224
PVT-Small [15]	79.8	25	3.8	224
ResMLP-S12 [13]	76.6	15	3.0	224
Swin-Mixer-T/D24 [9]	79.4	20	4.0	256
gMLP-S [8]	79.6	20	4.5	224
ViT-L/16* [4]	76.1	307	63.6	224
LiteViT(ours)	78.5	**1.16**	**1.2**	224

Table 2: Lite-SAM has achieved competitive results in both overall and class-wise performance. The best results in each class are displayed in red.

Category	Model				
	SAM-B [6] (r1024)	EfficientViT L0-SAM [1] (r1024)	MobileSAM [19] (r1024)	Lite-SAM (r640)	Lite-SAM (r1024)
overall	0.566	0.561	0.540	0.558	0.565
person	0.544	0.532	0.498	0.498	0.522
bicycle	0.294	0.276	0.247	0.283	0.295
car	0.576	0.539	0.511	0.567	0.588
motorcycle	0.420	0.438	0.365	0.364	0.390
airplane	0.570	0.612	0.576	0.525	0.543
bus	0.779	0.767	0.758	0.765	0.748
train	0.717	0.739	0.725	0.722	0.717
truck	0.684	0.660	0.638	0.659	0.661
boat	0.456	0.444	0.420	0.474	0.504
traffic light	0.527	0.498	0.509	0.595	0.592

Continued on next page

Table 2 – continued from previous page

Category	SAM-B [6] (r1024)	EfficientViT-L0-SAM [1] (r1024)	MobileSAM [19] (r1024)	Lite-SAM (r640)	Lite-SAM (r1024)
fire hydrant	0.716	0.717	0.709	0.720	0.703
stop sign	0.790	0.777	0.753	0.807	0.766
parking meter	0.743	0.728	0.741	0.784	0.758
bench	0.402	0.398	0.371	0.367	0.398
bird	0.434	0.419	0.391	0.340	0.416
cat	0.683	0.770	0.744	0.673	0.675
dog	0.710	0.745	0.715	0.667	0.676
horse	0.483	0.478	0.440	0.421	0.438
sheep	0.551	0.571	0.511	0.533	0.534
cow	0.587	0.587	0.534	0.540	0.564
elephant	0.649	0.680	0.637	0.611	0.608
bear	0.748	0.784	0.777	0.764	0.744
zebra	0.575	0.606	0.562	0.525	0.535
giraffe	0.549	0.571	0.537	0.454	0.493
backpack	0.502	0.487	0.448	0.503	0.513
umbrella	0.633	0.630	0.605	0.586	0.612
handbag	0.457	0.443	0.423	0.437	0.450
tie	0.484	0.439	0.413	0.437	0.482
suitcase	0.648	0.675	0.655	0.680	0.676
frisbee	0.708	0.710	0.691	0.728	0.708
skis	0.068	0.062	0.051	0.029	0.065
snowboard	0.328	0.315	0.311	0.320	0.386
sports ball	0.612	0.590	0.580	0.628	0.647
kite	0.506	0.476	0.470	0.421	0.487
baseball bat	0.436	0.400	0.371	0.339	0.387
baseball glove	0.619	0.619	0.612	0.650	0.637
skateboard	0.338	0.331	0.317	0.313	0.337
surfboard	0.472	0.450	0.422	0.460	0.493
tennis racket	0.562	0.559	0.532	0.518	0.516
bottle	0.633	0.605	0.582	0.637	0.641
wine glass	0.452	0.437	0.405	0.430	0.461
cup	0.698	0.675	0.669	0.721	0.710
fork	0.262	0.282	0.201	0.190	0.240
knife	0.339	0.311	0.266	0.296	0.356
spoon	0.374	0.334	0.302	0.321	0.374
bowl	0.618	0.505	0.561	0.664	0.629
banana	0.597	0.567	0.581	0.600	0.604
apple	0.653	0.645	0.637	0.671	0.656
sandwich	0.739	0.702	0.718	0.746	0.725
orange	0.670	0.653	0.654	0.697	0.680
broccoli	0.483	0.463	0.465	0.534	0.496
carrot	0.555	0.543	0.512	0.544	0.563
hot dog	0.561	0.578	0.555	0.610	0.594

Continued on next page

Table 2 – continued from previous page

Category	SAM-B [6] (r1024)	EfficientViT-L0-SAM [1] (r1024)	MobileSAM [19] (r1024)	Lite-SAM (r640)	Lite-SAM (r1024)
pizza	0.666	0.667	0.655	0.666	0.652
donut	0.724	0.719	0.702	0.746	0.728
cake	0.685	0.672	0.684	0.708	0.680
chair	0.421	0.433	0.413	0.425	0.433
couch	0.562	0.583	0.555	0.584	0.589
potted plant	0.429	0.455	0.447	0.467	0.465
bed	0.465	0.415	0.461	0.520	0.479
dining table	0.220	0.214	0.228	0.274	0.279
toilet	0.689	0.711	0.700	0.690	0.684
tv	0.753	0.757	0.739	0.763	0.738
laptop	0.674	0.716	0.671	0.675	0.675
mouse	0.707	0.679	0.701	0.721	0.711
remote	0.528	0.489	0.470	0.487	0.532
keyboard	0.691	0.694	0.691	0.715	0.703
cell phone	0.597	0.557	0.531	0.559	0.585
microwave	0.776	0.801	0.771	0.797	0.762
oven	0.615	0.567	0.582	0.628	0.632
toaster	0.825	0.823	0.738	0.836	0.815
sink	0.611	0.603	0.578	0.648	0.638
refrigerator	0.786	0.776	0.772	0.786	0.733
book	0.416	0.410	0.361	0.452	0.476
clock	0.745	0.732	0.715	0.803	0.774
vase	0.685	0.653	0.657	0.687	0.672
scissors	0.324	0.305	0.247	0.284	0.277
teddy bear	0.659	0.693	0.667	0.633	0.647
hair drier	0.473	0.600	0.477	0.418	0.455
toothbrush	0.369	0.328	0.307	0.299	0.359

A.3 Ablation study on the selection of training data

In Section 4.2, our choice of using 18% of the SA-1B data was based on a trade-off between training time and accuracy. The ablation study results regarding the selection of training data size and backbones, are presented in Tab. 3.

A.4 Results for Segment Anything and Everything.

The qualitative "SegEvery" outcomes of SAM [6] , Semantic-SAM [7], Fast-SAM [21], Mobile-SAM [19], and EfficientSAM [1], and our proposed approach are depicted in Fig. 1. The visualization illustrates that Lite-SAM achieves comparable results to SAM-B [6] and exhibits superior performance over both Fast-SAM [21] and Mobile-SAM [19]. We also provide "SegAny" visualized results and comparisions for box and point prompt in Fig. 3.

Table 3: Ablation study on the selection of training data size. See Section 4.2 for Implementation details. The evaluation metrics is 1-box prompt mAP on COCO2017(val) dataset. We finally chose 18% of SA-1B as training data, exemplying an optimal balance between training time and accuracy. It should be noted that the results for SAM-B, EfficientViT-L0-SAM, and MobileSAM are all reproduced by our own, without any open-source SAM training code avaliable. Therefore there may be minor inconsistencies with the original papers or models.

Model	Metric & Training time (4 epochs)	Training images of SA-1B			
		1M (9%)	2M (18%)	5M (45%)	11M (100%)
SAM-B [6] (r1024)	mAP(%)	51.9	54.4	57.4	59.0
	Hours	47	96	230	513
EfficientViT-L0-SAM [1] (r1024)	mAP(%)	52.1	55.6	56.7	57.9
	Hours	40	83	203	427
MobileSAM [19] (r1024)	mAP(%)	51.5	53.9	54.8	55.6
	Hours	38	75	197	402
Lite-SAM (r640)	mAP(%)	52.3	55.8	56.4	57.1
	Hours	26	50	130	272
Lite-SAM (r1024)	mAP(%)	53.9	56.5	57.6	58.2
	Hours	37	68	181	403

A.5 Zero-Shot Image Segmentation Results on ARI-TEST2024

Private data. To further evaluate the zero-shot generalization in real-world scenarios, we introduce a novel dataset termed **ARI-TEST2024**. This dataset contains 10,000 meticulously annotated high-resolution images (1024×1024) from varied locations, such as storage units, reservoirs, restaurant kitchens, transformer substations, gas stations, and garbage recycling facilities. Representative samples are presented in Fig. 2.

We demonstrate the robust generalization and stability of our proposed Lite-SAM algorithm in comparison with eight different algorithms. Lite-SAM achieves mIoU scores of 68.3% and 54.5% using the 1-box and 1-point prompt respectively, as detailed in Tab. 4.

To assess the effectiveness of our model in generating segmentation masks influenced by prompts, we utilize both our model and other models based on the SAM framework to conduct instance segmentation. This includes both point-based and box-based prompt segmentation methodologies. In Fig. 3, it is evident that Fast-SAM [21] fails to produce any results in the scene shown in column (a). This behavior can be attributed to the approach employed by the algorithm, where the input point or box is treated solely as a post-processing strategy rather than being utilized as an actual cue. In contrast, our Lite-SAM generates a satisfactory mask prediction that closely resembles the output obtained from SAM-B [6].

Fig. 1: Qualitative results on "SegEvery". Models demonstrate mask generation capabilities. (1) Note that EfficientViT-SAM's [1] result is based on L1 model. (2) Lite-SAM employs an inference size of 640 × 640, while other comparison algorithms utilize a default size of 1024 × 1024.

Table 4: Zero-Shot Image Segmentation Results on ARI-TEST2024 using mIoU metric.

Model	ARI-TEST2024 mIoU ↑		
	1-box(%)	1-point(%)	Input Size
SAM-B [6]	70.6	53.7	1024^2
SAM-L [6]	72.3	56.4	1024^2
SAM-H [6]	**72.4**	**56.8**	1024^2
Semantic-SAM [7]	N/A	50.3	1024^2
Fast-SAM [21]	62.2	41.5	1024^2
Mobile-SAM [19]	64.0	42.0	1024^2
EfficientViT-L0-SAM [1]	67.9	50.4	1024^2
EfficientViT-L1-SAM [1]	**68.6**	51.6	1024^2
Lite-SAM(r640)	66.6	52.1	640^2
Lite-SAM(r1024)	68.3	**54.5**	1024^2

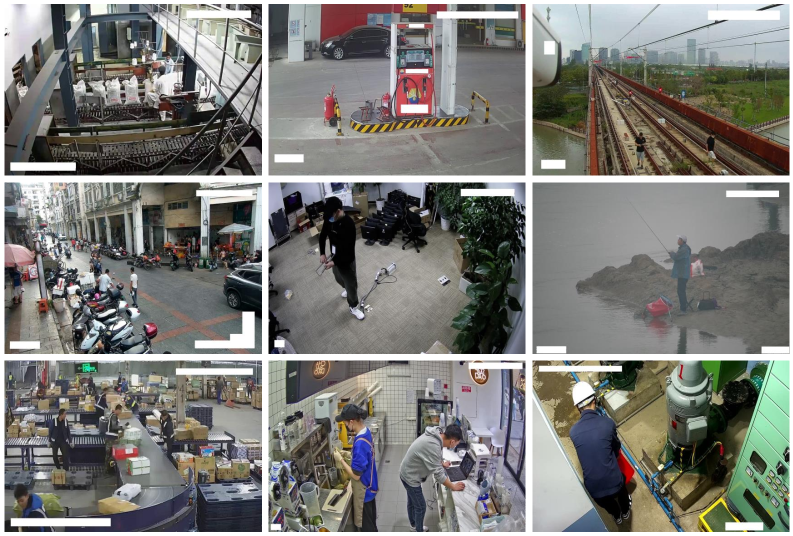

Fig. 2: The proposed **ARI-TEST2024 dataset.** Faces and vehicle license plates have been blurred in the released images. All images are resized to a size of 1024×1024. Each scene contains 1000 images, randomly selected from different videos.

B Other Materials

B.1 Distance Transform: pseudo code

As mentioned in Section 3.3 (2) and Figure 4, we have incorporated the use of distance transforms to estimate the confidence of point prompts. This facilitates the calculation of the distance between a point and its corresponding mask, as depicted in Listing 1.1.

```
alpha = 4.0; eps = 1e-4
h, w, c = image.shape; hmap = zeros(h, w)
masks = sort(masks, key=lambda x: x['area'], reverse=True)
for mask in masks:
    mask_pad  = pad(mask, ((1, 1), (1, 1)), 'constant')
    mask_dist = distanceTransform(mask_pad, kernel=5)
    mask_dist = (mask_dist / (mask_dist.max() + eps))
    mask_dist = mask_dist ** alpha
    hmap = maximum(hmap, mask_dist)
```

Listing 1.1: distance transform pseudo code

B.2 Q & A

1. *Why do the parameter and computation amounts differ from those mentioned in the reference article?*

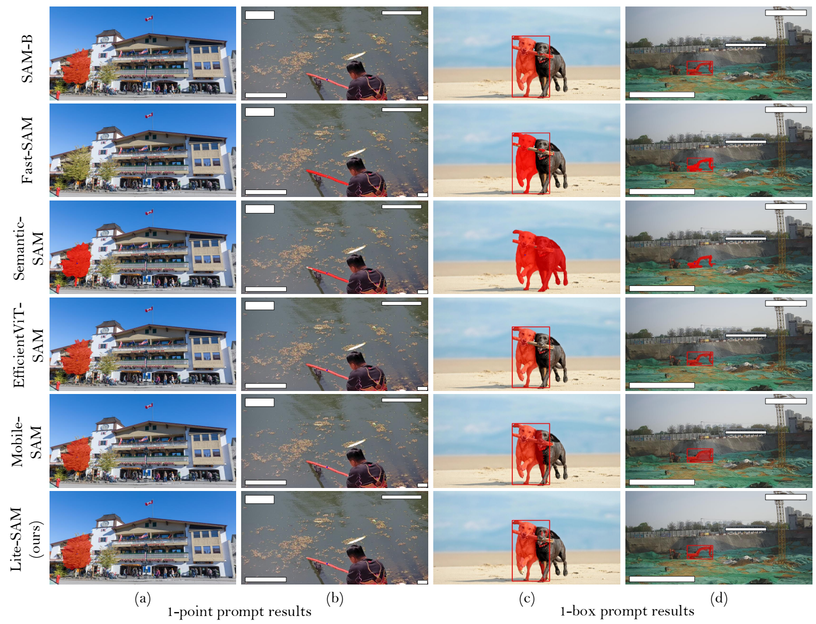

Fig. 3: Qualitative "SegAny" results on COCO2017 and ARI-TEST2024 with bounding box or point as prompt. Please note that the code provided for Semantic-SAM [7] does not include support for box prompts. Therefore, we have used point prompt results instead.

Answer: The reference article does not provide the script to calculate the parameter and computation amounts. Hence, we downloaded their code and model, and employed the same script for an accurate calculation. The script utilized is provided in Listing 1.2. (If the code was unavailable, we used the data provided in the reference article).

```
from ptflops import get_model_complexity_info as cmplx
from segment_anything import sam_model_registry

class Model_1prompt(torch.nn.Module):
  def __init__(self, model):
    super(FlopTestModel, self).__init__()
    self.model = model

  def forward(self, inputs):
    image_embedding, _ = self.model.image_encoder(inputs)

    sparse_embeddings, dense_embeddings = \
        self.model.prompt_encoder(
          points=(torch.randn(1,1,2).cuda(),
                  torch.randn(1,1).cuda()),
          boxes=None,
```

```
          masks=None,
        )

    low_res_masks, iou_predictions = \
        self.model.mask_decoder(
           image_embeddings=image_embedding,
           image_pe=self.model.prompt_encoder.get_dense_pe(),
           sparse_prompt_embeddings=sparse_embeddings,
           dense_prompt_embeddings=dense_embeddings,
           multimask_output=True,
        )
    return low_res_masks, iou_predictions

if __name__ == "__main__":
  model = sam_model_registry["vit_b"]
  model_1prompt = Model_1prompt(model)
  input_size = 640
  flops, params = cmplx(model.image_encoder,
                        (3, input_size, input_size),
                        as_strings=True,
                        print_per_layer_stat=True)
  print("FLOPs: %s, Params: %s " % (flops, params))

  flops, params = cmplx(model_1prompt,
                        (3, input_size, input_size),
                        as_strings=True,
                        print_per_layer_stat=False)
  print("FLOPs: %s, Params: %s " % (flops, params))
```

Listing 1.2: complexity pseudo code

2. *Why is the inference time different from other papers?*

Answer: In this paper, we recalibrate the computation time of SegEvery, adopting the calculation method used by Mobile-SAM-v2 [20] for a uniform comparison. The inference time mentioned in the SAM [6] , Semantic-SAM [7], Fast-SAM [21], EfficientViT-SAM [1] , Mobile-SAM [19], and Edge-SAM [22] papers refers to the SegAny time. However, the inference time reported in this paper and Mobile-SAM-v2 [20] is based on SegEvery time. Additionally, Mobile-SAM-v2 [20] is a two-stage model, and the parameter count and inference time of the Object-aware model are not reported in the paper. Therefore, we have recalculated the parameter count, MACs, and SegEvery time for Mobile-SAM-v2 [20].

3. *Does this paper solely support segmentation? Does it have text capabilities?*

Answer: In this paper, benchmarking against lightweight SAM algorithms like MobileSAM [19] and Mobile-SAM-v2 [20], primarily addresses the SegEvery problem. It does not support text capabilities.

References

1. Cai, H., Li, J., Hu, M., Gan, C., Han, S.: Efficientvit: Multi-scale linear attention for high-resolution dense prediction 2, 3, 4, 5, 6, 9

2. Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., Ouyang, W.: Glit: Neural architecture search for global and local image transformer. In: ICCV (2021) 2
3. Dai, Z., Liu, H., Le, Q.V., Tan, M.: Coatnet: Marrying convolution and attention for all data sizes. Advances in Neural Information Processing Systems **34**, 3965–3977 (2021) 2
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 2
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 2
6. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. arXiv preprint arXiv:2304.02643 (2023) 2, 3, 4, 5, 6, 9
7. Li, F., Zhang, H., Sun, P., Zou, X., Liu, S., Yang, J., Li, C., Zhang, L., Gao, J.: Semantic-sam: Segment and recognize anything at any granularity. arXiv preprint arXiv:2307.04767 (2023) 4, 6, 8, 9
8. Liu, H., Dai, Z., So, D.R., Le, Q.V.: Pay attention to mlps. arXiv preprint arXiv:2105.08050 (2021) 2
9. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10012–10022 (October 2021) 2
10. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2022) 2
11. Mehta, S., Rastegari, M.: Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. arXiv preprint arXiv:2110.02178 (2021) 2
12. Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., Xu, C.: Vitas: Vision transformer architecture search. arXiv (2021) 2
13. Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J., et al.: Resmlp: Feedforward networks for image classification with data-efficient training. arXiv preprint arXiv:2105.03404 (2021) 2
14. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. pp. 10347–10357. PMLR (2021) 2
15. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 568–578 (October 2021) 2
16. Wightman, R., Touvron, H., Jégou, H.: Resnet strikes back: An improved training procedure in timm. arXiv preprint arXiv:2110.00476 (2021) 2
17. Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J., Yuan, L.: Tinyvit: Fast pretraining distillation for small vision transformers. In: European Conference on Computer Vision. pp. 68–85. Springer (2022) 2
18. Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., Yan, S.: Metaformer is actually what you need for vision (2022) 2

19. Zhang, C., Han, D., Qiao, Y., Kim, J.U., Bae, S.H., Lee, S., Hong, C.S.: Faster segment anything: Towards lightweight sam for mobile applications. arXiv preprint arXiv:2306.14289 (2023) 2, 3, 4, 5, 6, 9
20. Zhang, C., Han, D., Zheng, S., Choi, J., Kim, T.H., Hong, C.S.: Mobilesamv2: Faster segment anything to everything (2023) 9
21. Zhao, X., Ding, W., An, Y., Du, Y., Yu, T., Li, M., Tang, M., Wang, J.: Fast segment anything. arXiv preprint arXiv:2306.12156 (2023) 4, 5, 6, 9
22. Zhou, C., Li, X., Loy, C.C., Dai, B.: Edgesam: Prompt-in-the-loop distillation for on-device deployment of sam (2023) 9