SegIC: Unleashing the Emergent Correspondence for In-Context Segmentation

Lingchen Meng[1,2] Shiyi Lan[3] Hengduo Li[4]
Jose M. Alvarez[3] Zuxuan Wu[1,2†] Yu-Gang Jiang[1,2]

[1]Shanghai Key Lab of Intell. Info. Processing, School of CS, Fudan University
[2]Shanghai Collaborative Innovation Center of Intelligent Visual Computing
[3]NVIDIA [4]University of Maryland

A Additional Implementation Details

In this section, we provide additional details of the mask decoding and training pipeline.

Mask decoding. Inspired by recent query-based segmentation methods [1–4], we utilize a lightweight query-based mask decoder to effectively map the in-context enhanced image features and object query to an output mask. We employ a learnable object query that will be used for the decoder's output. The workflow of SegIC is concisely summarized in Pytorch-style pseudocode, as presented in Algorithm 1. Initially, the image feature and in-context instructions are projected into the same feature space for mask decoding. Subsequently, the projected in-context features are leveraged to enhance the image feature and object feature. The mask decoder then executes a multi-layered decoding process, structured as a four-step procedure within each layer, including (1) self-attention for the concatenated object query; (2) cross-attention from the image feature to the object query; (3) cross-attention from the query back to the image; and (4) calculating the mask. The mask calculation is performed using the image feature and the first element of the concatenated object feature, which corresponds to the position of the initial object query.

Training pipeline. In our main experiments, we adopt a mixed training scheme using both semantic segmentation datasets (COCO and ADE20k) and instance segmentation datasets (COCO and LVIS), as presented in Algorithm 2. We do not focus intensively on adjusting the dataset ratios, instead opting for uniform dataset-level sampling. For the segmentation datasets, we employ large-scale jittering (ranging from 0.1 to 2.0) for both the target image and in-context examples. These in-context examples are constructed based on the semantic class label of the target image, sampling one class per image during training. In the case of instance segmentation datasets, in-context examples are generated by applying two separate data augmentations to the same image. The instances from these differently augmented views then serve as mutual in-context examples. Our standard data augmentation techniques for this task include random resizing cropping (ranging from 0.3 to 1.0), random color jittering (with a 0.2 probability), and random horizontal flipping (with a 0.1 probability).

B Additional Visualization

In this section, we provide more visualizations of the middle output and the predictions of SegIC.

Propagated mask. As outlined in Section 3.2, the propagated mask **a** is derived from a weighted mean of dense correspondences according to the ground-truth mask of in-context samples. To facilitate visualization, we first apply the sigmoid function to map **a** into $(0, 1)$. Subsequently, this range is transformed into RGB space using the JET color map. As depicted in Figure 1, this process demonstrates that the propagated masks predominantly concentrate on the objects referenced in the in-context examples, providing strong guidance for the subsequent mask decoding process. This observation further demonstrates the emerging potential of pre-trained vision foundation models in the realm of segmentation tasks.

Fig. 1: Visualization of propagated masks. We propagate labels from the in-context examples to the targets to obtain propagated masks by exploring the dense correspondences. We employ DINO-v2-large [5] for the visualization.

More qualitative results on VOS. We further provide more qualitative results on video object segmentation tasks (VOS). Note that SegIC is never

Algorithm 1: Pseudo code for SegIC Mask Decoding.

```
# Inputs: Image Embedding f; In-context Instructions c = {a, p, v, m}
# Variables: Learnable Object Queries q
# Functions: Conv4ImgFeature(), Conv4ProgatedLabel(); Proj4Pos(), Proj4Vis(),
  Porj4Meta(); QuerySelfAttn(), Query2ImgAttn(), Image2QueryAttn(), output()
def InContext_Enhancement(f, a, p, v, m):
    # Project image feature and in-context propagated mask into the hidden space for
      mask decoding.
    f', a'=Conv4ImgFeature(f), Conv4ProgatedLabel(a);
    # Enhance image feature with in-context propagated mask.
    f'=f'+a';
    # Project other in-context instructions into the hidden space for mask decoding.
    q^p, q^v, q^m=Proj4Pos(p),Proj4Vis(v),Proj4Meta(m)
    # Enhance the object query by contacting with the hidden features of in-context
      instructions.
    q'=Concat(q, q^p, q^v, q^m)
def Mask_Decoder(F, Q):
    Q' = QuerySelfAttn(Q) # Query self-attention
    Q^o = Img2QueryAttn(Q', F) # Image-to-query cross-attention
    F^o = Query2ImgAttn(F, Q_o) # Query-to-image cross-attention
    O = output(F^o, Q^o[0]) # Compute mask
def forward(f, a, p, v, m)):
    f', q' = InContext_Enhancement(f, a, p, v, m) # Enhance image feature and object
      query with in-context instructions.
    Q^o, F^o = q', f' # Initialize variables for mask decoding
    for i in range(max_iter):
        O, Q^o, F^o = Mask_Decoder(Q^o, F^o)
```

Algorithm 2: Pseudo code for training pipeline.

```
# training set: mixed dataset D = D_inst ∪ D_sem
def ICL_Preprocess(data):
    I^t, y^t, category = data
    if task_type(data) == 'semantic':
        I^r, y^r = CategoryAwareSample(D_sem, category)
        meta = 'a photo of a {category}.'
    if task_type(data) == 'instance':
        I^r, y^r = DataAug(I^t, y^t) # Individual data aug to build a different view as
          reference
        meta = 'please segment the instances.'
def train_epoch(model, D):
    for data in D:
        I^t, y^t, I^r, y^r, meta = ICL_Preprocess(data)
        loss = model(I^t, y^t, I^r, y^r, meta)
```

trained on video datasets and just treats VOS as in-context segmentation using the first frame as in-context examples. As shown in Figure 2, SegIC well handles challenging scenarios, including (a) occlusions, (b) interwoven objects, and (c) small objects.

Fig. 2: Qualitative results on VOS. SegIC perform well on challenging scenarios in video object segmentation, including (a) occlusions, (b) interwoven objects, and (c) small objects.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020)
2. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: CVPR (2022)
3. Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)
4. 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)
5. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)