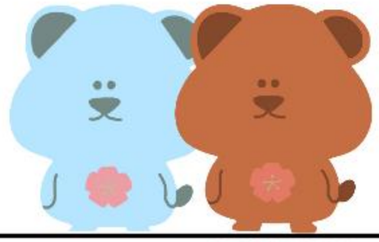
Segment Anything

고급심화 차수빈

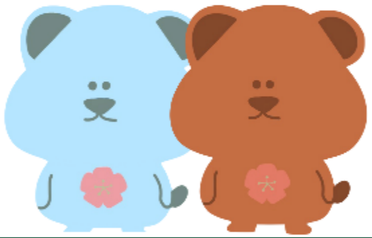
EWHA EURON

Contents



EWHA EURON

1. Introduction

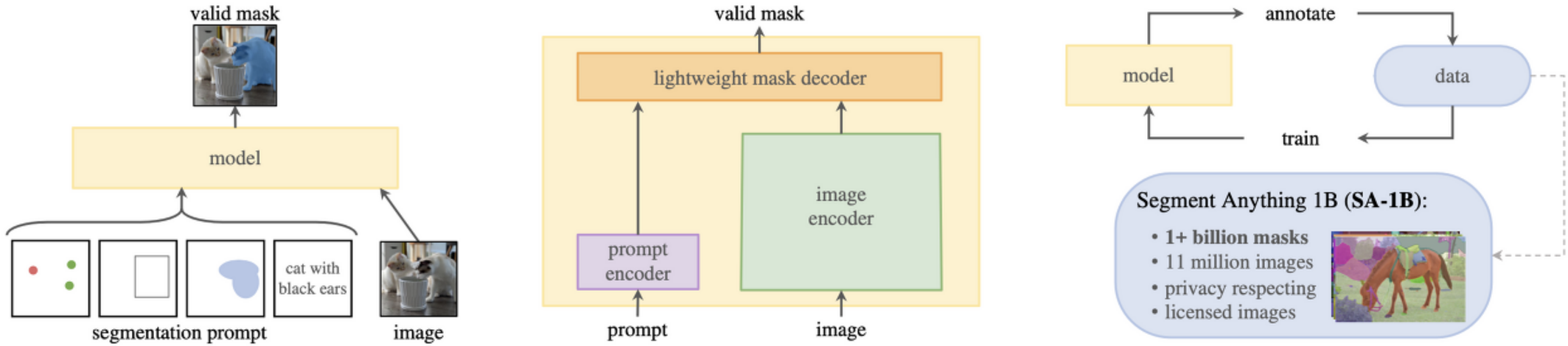


EWHA EURON

#1. Introduction

- **SA(Segment Anything) 프로젝트**
  - 대용량 데이터셋을 가지고 자연어 및 다양한 지시(= prompt)로 Image Segmentation task를 수행할 수 있도록 하는 프로젝트
  - 목표: image segmentation 분야에서의 foundation model을 제안하겠다!

  foundation model
  - 훈련 중에 본 적이 없는 작업 및 데이터 분포에도 적용할 수 있는 범용 모델
  - zero-shot learning



(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

Figure 1: We aim to build a foundation model for segmentation by introducing three interconnected components: a promptable segmentation *task*, a segmentation *model* (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a *data* engine for collecting SA-1B, our dataset of over 1 billion masks.

#1. Introduction

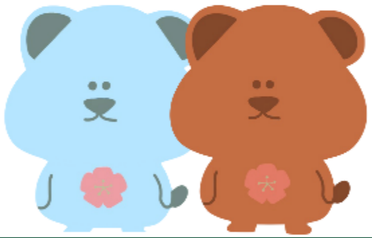
- SA 프로젝트

1. What task will enable zero-shot generalization?

2. What is the corresponding model architecture?

3. What data can power this task and model?

2. Segment Anything Task

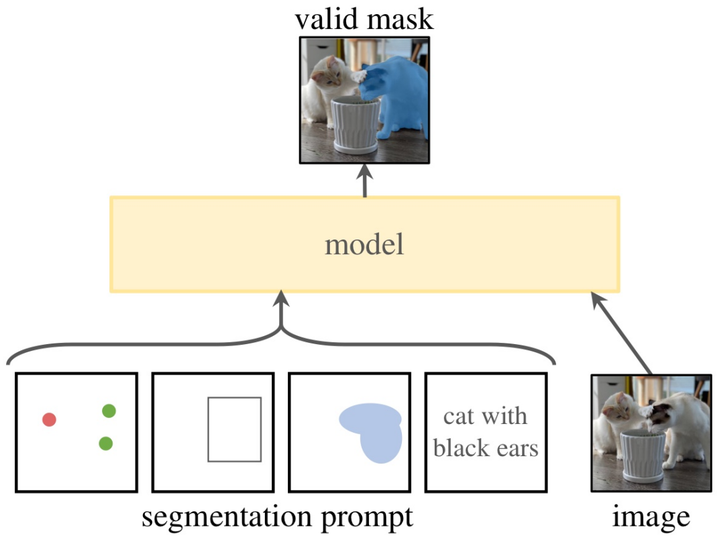


EWHA EURON

#2. Segment Anything Task

"prompt enginnering을 Image Segmentation에도 적용해 보자!"

- task에 대한 guide를 제공하는 역할을 하는 prompt와 함께 모델을 학습
  - ㄴ 다양한 downstream에 대해서 zero-shot transfer가 가능하게 됨
- 모호한 prompt에 대해서도 학습
  - ㄴ ambiguous prompt: prompt가 지칭하는 대상(부분)이 명확하지 않은 prompt



(a) **Task**: promptable segmentation

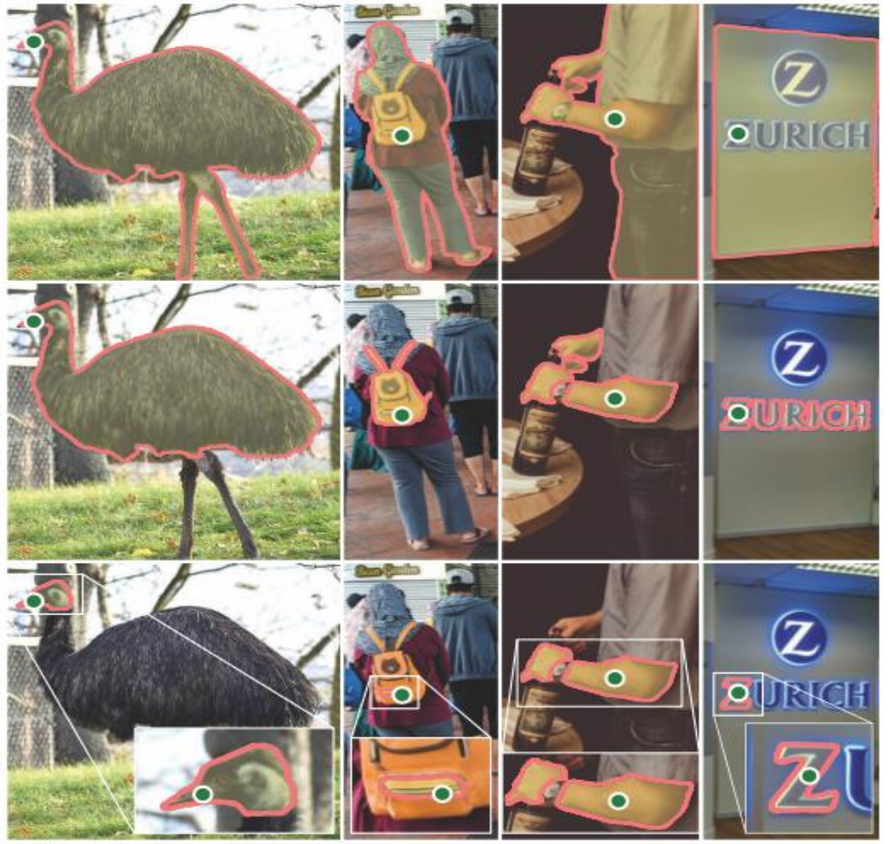
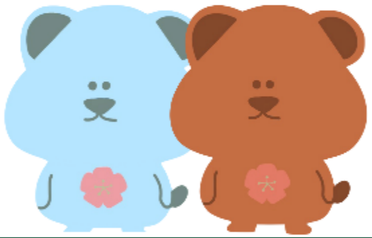


Figure 3: Each column shows 3 valid masks generated by SAM from a single ambiguous point prompt (green circle).

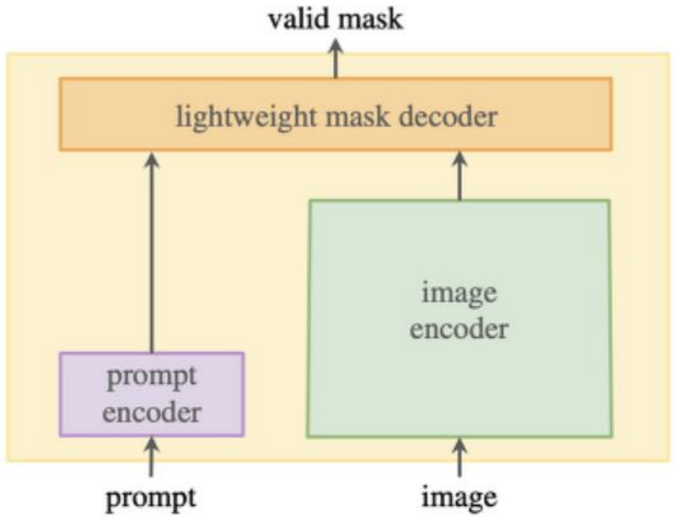
3. Segment Anything Model



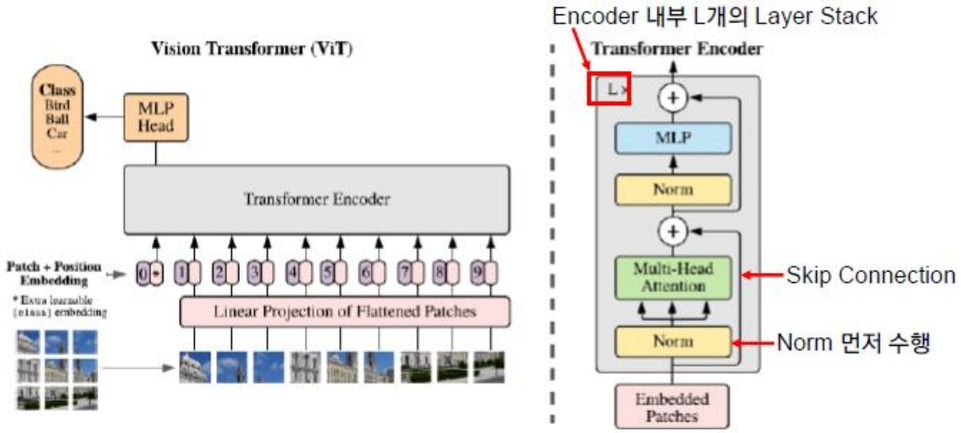
EWHA EURON

#3. Segment Anything Model

- **SAM**(Segment Anything Model)은 크게 **세 부분**으로 구성됨
  ⇒ 이미지 인코더, prompt 인코더, mask 디코더



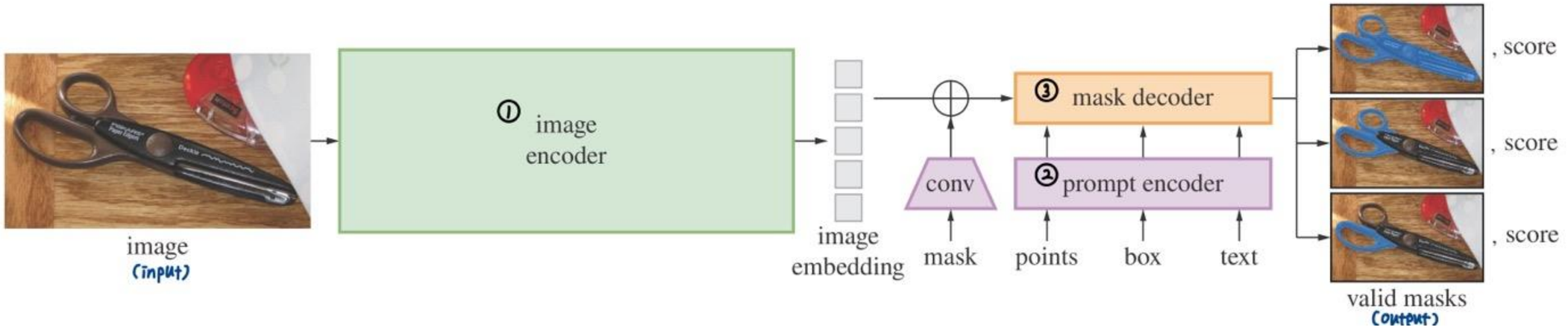
- Vision Transformer(ViT) 모델을 기반으로 하되 real-time 성능을 위한 trade-off를 가지고 있음
  ㄴ ViT: 이미지 인식 및 처리 작업에서 높은 성능을 발휘하는 딥러닝 모델로, 자연어 처리(NLP) 분야에서 성공을 거둔 Transformer 구조를 컴퓨터 비전 분야에 적용한 모델



[Vision Transformer (ViT) 란? - 정의, 원리, 구현, 응용분야]

#3. Segment Anything Model

- **모델 구성**
  - Image Encoder
    - ㄴ 이미지가 들어오면 embedding을 얻는 부분
  - Prompt Encoder
    - ㄴ prompt를 embedding하는 부분
    - ㄴ sparse와 dense prompt로 구분됨
      - sparse: 점(points), 박스(boxes), 텍스트(text)
      - dense: 마스크(Mask)
  - Mask Decoder
    - ㄴ image embedding, prompt embedding, output token을 받아서 segmentation map(mask)을 만드는 부분
    - ㄴ Transformer Decoder를 변형하여 활용



#3. Segment Anything Model

- **모델 구성**
  - Prompt Encoder
    - ㄴ **Sparse** Prompt
      - ▶ 점(points): Positional encodings summed with learned embeddings
      - ▶ 박스(boxes): Positional encodings summed with learned embeddings
      - ▶ 텍스트(text): CLIP output

```
def _embed_boxes(self, boxes: torch.Tensor) -> torch.Tensor:
    """Embeds box prompts."""
    boxes = boxes + 0.5  # Shift to center of pixel
    coords = boxes.reshape(-1, 2, 2)
    corner_embedding = self.pe_layer.forward_with_coords(coords, self.input_image_size)
    corner_embedding[:, 0, :] += self.point_embeddings[2].weight
    corner_embedding[:, 1, :] += self.point_embeddings[3].weight
    return corner_embedding

sparse_embeddings = torch.empty((bs, 0, self.embed_dim), device=self._get_device())
if points is not None:
    coords, labels = points
    point_embeddings = self._embed_points(coords, labels, pad=(boxes is None))
    sparse_embeddings = torch.cat([sparse_embeddings, point_embeddings], dim=1)
if boxes is not None:
    box_embeddings = self._embed_boxes(boxes)
    sparse_embeddings = torch.cat([sparse_embeddings, box_embeddings], dim=1)
```

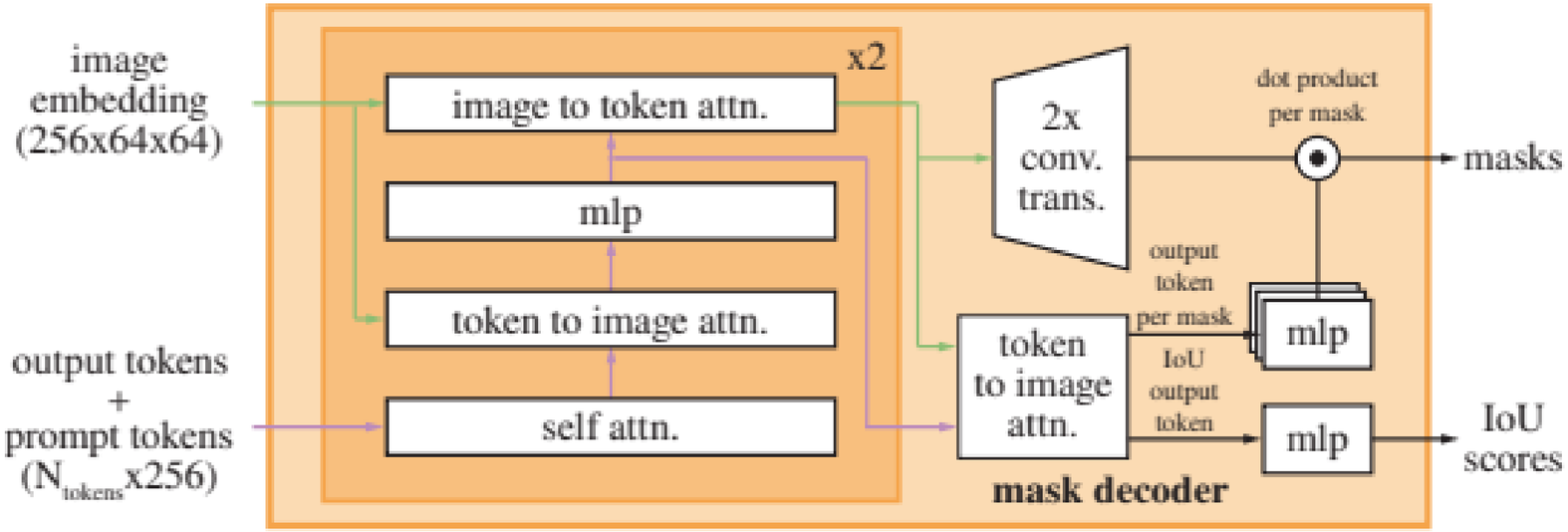
#3. Segment Anything Model

- **모델 구성**
  - Prompt Encoder
    - ㄴ **Dense** Prompt
      - ▶ 마스크(Mask): Image embedding에 convolution 수행

```
self.mask_downscaling = nn.Sequential(
    nn.Conv2d(1, mask_in_chans // 4, kernel_size=2, stride=2),
    LayerNorm2d(mask_in_chans // 4),
    activation(),
    nn.Conv2d(mask_in_chans // 4, mask_in_chans, kernel_size=2, stride=2),
    LayerNorm2d(mask_in_chans),
    activation(),
    nn.Conv2d(mask_in_chans, embed_dim, kernel_size=1),
)
```

#3. Segment Anything Model

- **모델 구성**
  - Mask decoder
    - ㄴ Transformer Decoder를 변형하여 활용함
      head에 dynamic mask prediction이 가능하도록 약간의 수정을 반영
    - ㄴ 전체 임베딩을 업데이트하기 위해 prompt-to-image와 image-to-prompt로 양방향으로 self-attention과 cross-attention을 수행
      양방향의 의미) Prompt-to-image, Image-to-prompt



☆ 레이블을 생성하지는 않음!!
단지 마스크를 만들어 줌

#3. Segment Anything Model

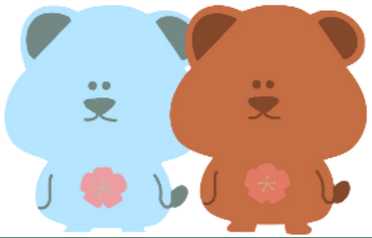
- **모델 설정**

  - 모호성 해결
    - ㄴ 1개의 prompt에 대해 3개의 mask prediction을 하도록 함
      whole, part, and subpart
    - ㄴ back propagation은 마스크들 중에서 minimum loss를 가지고 있는 것에 대해서만 수행

  - Efficiency
    - ㄴ 미리 계산된 image embedding에 대해서 prompt encoder와 mask decoder 작동은 50ms 이내에 가능
      -> real-time interactive prompting이 가능한 범위

  - Loss & 학습
    - ㄴ focal loss와 dice loss를 복합적으로 활용
      - focal loss) “더욱 어려운 객체에 대해 가중치를 주어 학습한다.”
      - Dice loss) IoU(Intersection over Union)와 비슷한 개념, “GT를 얼마나 많이 포함했는가”
    - ㄴ prompt를 랜덤 샘플링해서 mask당 11 round가 돌도록 interactive setup을 수행

4. Segment Anything Data



EWHA EURON

#4. Segment Anything Data

- **Data Engine**

  모델을 학습시켜야 하는데,,어라,,mask 데이터가 충분하지 않네..^^;

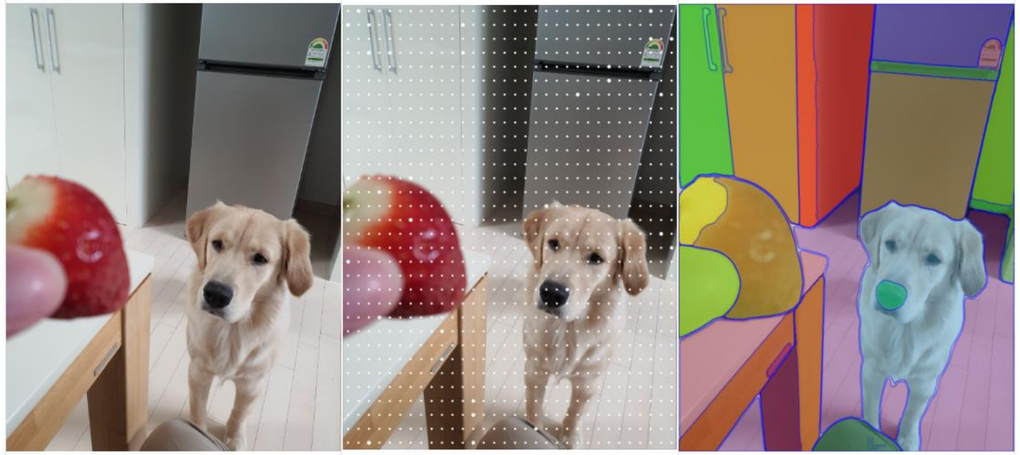
  - SAM 모델을 위한 Data Engine 구축 → SA-1B
  - **3단계**를 거쳐 데이터를 수집 → 점점 **자동화**되는 방식

    1) Assisted-manual stage
       - ㄴ 작업자가 점을 찍으면 SAM 모델이 어느 정도 마스크를 만들어 주는 단계

    2) Semi-automatic stage
       - ㄴ 사람이 직접 라벨링 한 mask GT와 모델이 예측한 mask를 함께 활용하는 단계
       - ㄴ 정확도를 유지시키면서도 human cost를 줄여가는 단계

    3) Fully-automatic stage
       - ㄴ 모든 것을 모델이 예측한 mask로 학습하는 단계
       - ㄴ prompt로 grid point를 제공함으로써 ambiguity에 대응
       - ㄴ Post-processing 단계를 거치며 mask를 정교화

  -> 이를 통해 11M의 image로부터 1B masks를 얻을 수 있었음



Input Image | Regular grid | Output mask

#4. Segment Anything Data

- Data Set

Dataset Characteristics

Total number of images: 11M

Total number of masks: 1.1B

Average masks per image: 100

Average image resolution: 1500×2250 pixels

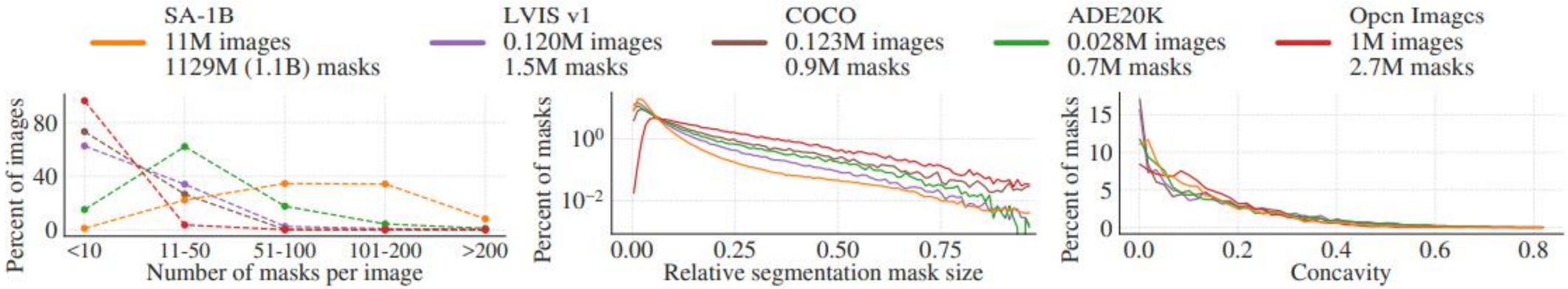
NOTE: There are no class labels for the images or mask annotations.

Labels
Class agnostic mask annotations

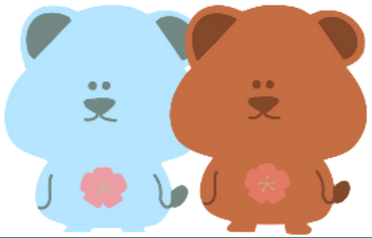
❗ segmentation mask는 instance별로 존재하지만, 각 instance가 무엇인지 명시되어 있지는 않음

- Images
  - ㄴ 사진사가 찍은 11M개의 고해상도 이미지
  - ㄴ 평균 해상도는 3300 x 4950

- Masks
  - ㄴ 대부분의 mask를 fully automatic 방법으로 얻음
  - ㄴ 약 500장의 샘플 데이터로 실험 결과, 수정 전과 후의 IoU가 평균 0.9 수준



5. Results



EWHA EURON

#5. Results

- Results
  - 5개의 하위 task에 대한 SAM의 zero-shot learning 성능을 보여주는 실험 진행
    - Single Point Valid Mask Evaluation
    - Zero-Shot Edge Detection
    - Zero-Shot Object Proposals
    - Zero-Shot Instance Segmentation
    - Zero-Shot Text-to-Mask
  - 모든 결과는 SA-1B와 다른 분포를 가지는 새로운 데이터셋에 대해서 평가함

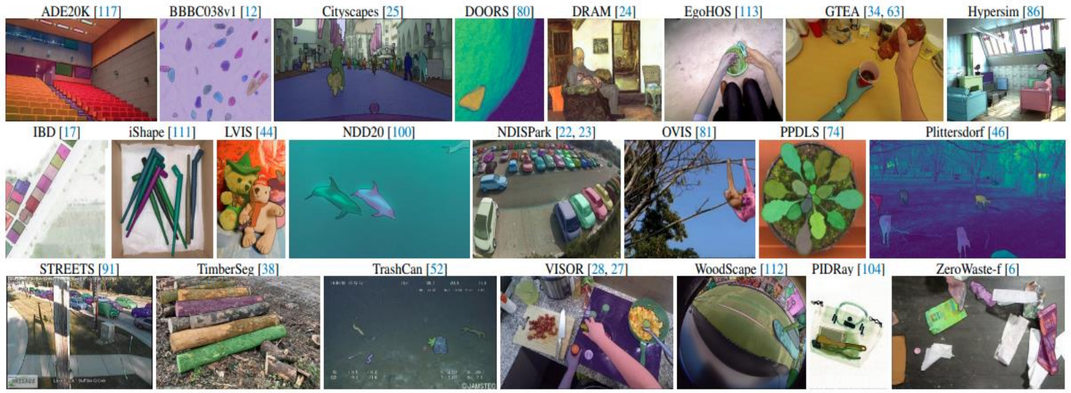


Figure 8: Samples from the 23 diverse segmentation datasets used to evaluate SAM's zero-shot transfer capabilities.

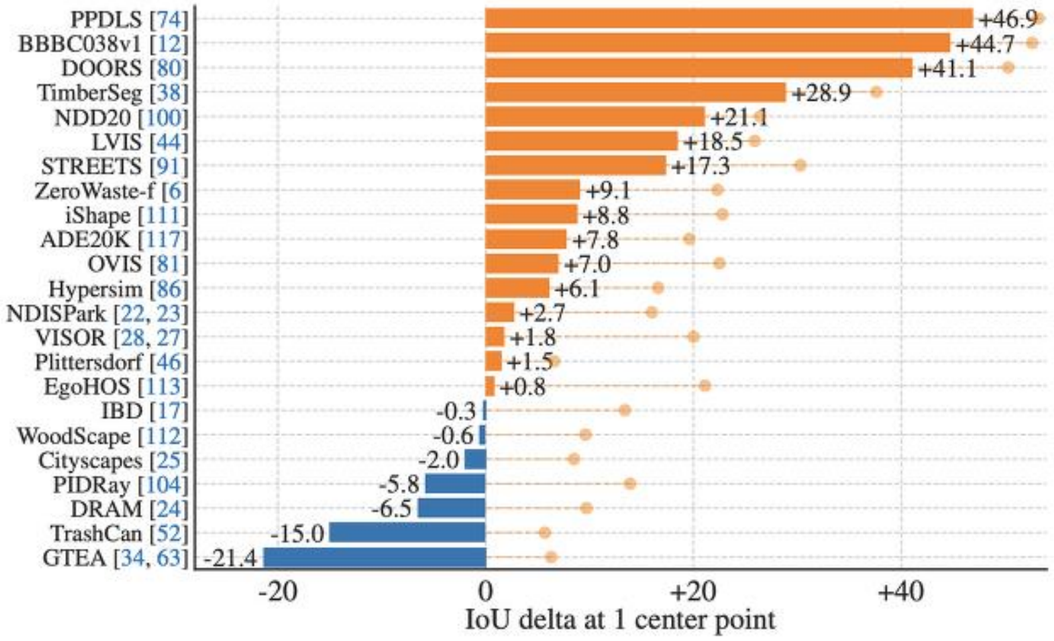
#5. Results

- Results

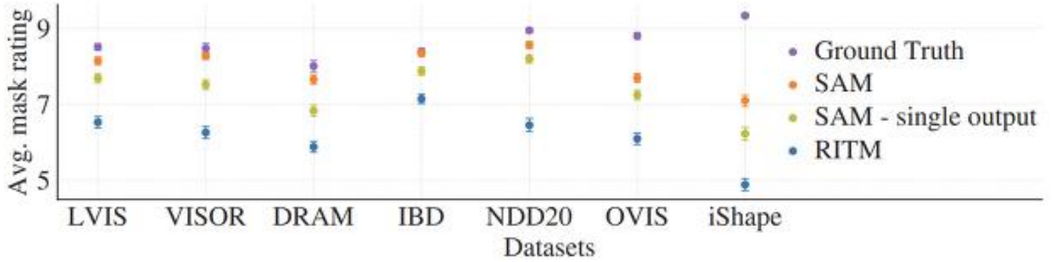
1) Single Point Valid Mask Evaluation

foreground에 해당하는 곳의 중점만 prompt로 주어졌을 때 foreground segmentation mask를 추출해 내는 작업

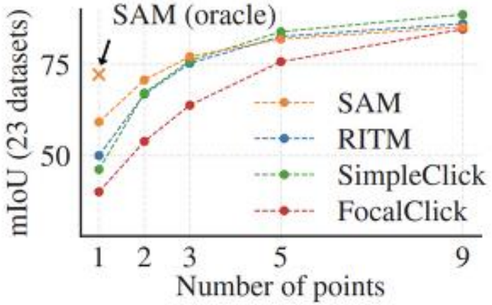
- RITM model과 비교한 결과, 23개의 데이터셋 중 16개에서 높은 성능을 보임
- 몇몇 데이터셋에서의 정량적 평가에서는 RITM보다 낮았지만, 정성적 평가에서는 더 높은 결과를 보이기도 함
- prompt로 주는 점의 개수가 많아질수록 그 차이가 적어지지만 개수가 적을 때는 SAM이 월등한 성능을 보임



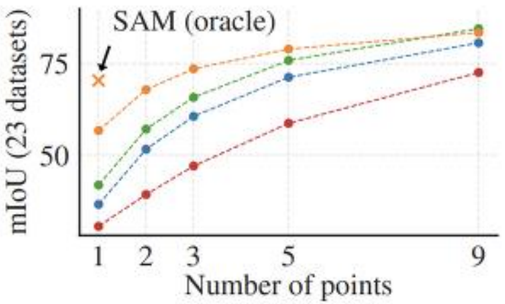
(a) SAM *vs.* RITM [92] on 23 datasets



(b) Mask quality ratings by human annotators



(c) Center points (default)



(d) Random points

#5. Results

- **Results**

  2) Zero-Shot Edge Detection

  전체 이미지에 대해서 segmentation을 진행한 결과에서 Sobel Filter를 써서 edge만 뽑아내는 작업

  - Edge Detection에 대해서 전혀 학습시키지 않았음에도 불구하고 준수한 성능을 보임
  - SAM보다 좋은 성능을 낸 모델들은 테스트 데이터 셋인 BSDS500의 트레이닝 데이터셋으로 학습된 것임을 감안하면, 아주 좋은 성능이라고 볼 수 있음

method	year	ODS	OIS	AP	R50
HED [108]	2015	.788	.808	.840	.923
EDETR [79]	2022	.840	.858	.896	.930
zero-shot transfer methods:					
Sobel filter	1968	.539	-	-	-
Canny [13]	1986	.600	.640	.580	-
Felz-Hutt [35]	2004	.610	.640	.560	-
SAM	2023	.768	.786	.794	.928

Table 3: Zero-shot transfer to edge detection on BSDS500.

#5. Results

- Results

3) Zero-Shot Object Proposals

Object proposal을 뽑아내는 성능을 평가, output을 mask로 뽑아내는 대신 box를 뽑아내도록 함

- ViTDet-H와 비교해 보면, 전반적으로 ViTDet이 더 좋은 성능을 보이긴 함
- SAM이 rare한 케이스에서는 더 좋은 성능을 보이기도 함

	mask AR@1000						
method	all	small	med.	large	freq.	com.	rare
ViTDet-H [62]	63.0	51.7	80.8	87.0	63.1	63.3	58.3
zero-shot transfer methods:							
SAM – single out.	54.9	42.8	76.7	74.4	54.7	59.8	62.0
SAM	59.3	45.5	81.6	86.9	59.1	63.9	65.8

Table 4: Object proposal generation on LVIS v1. SAM is applied zero-shot, *i.e.* it was not trained for object proposal generation nor did it access LVIS images or annotations.

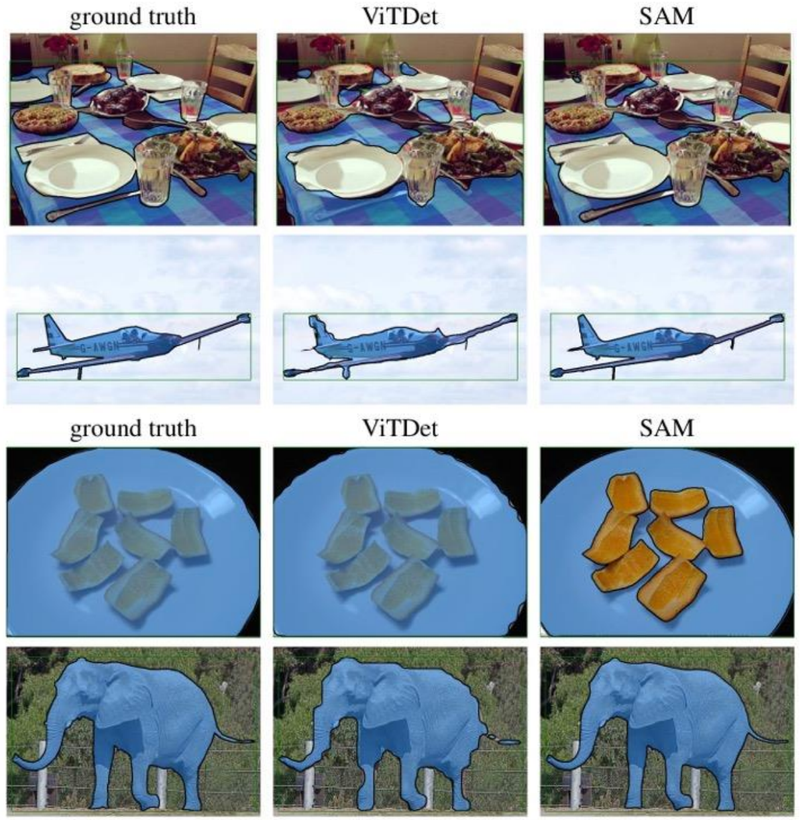
#5. Results

- **Results**

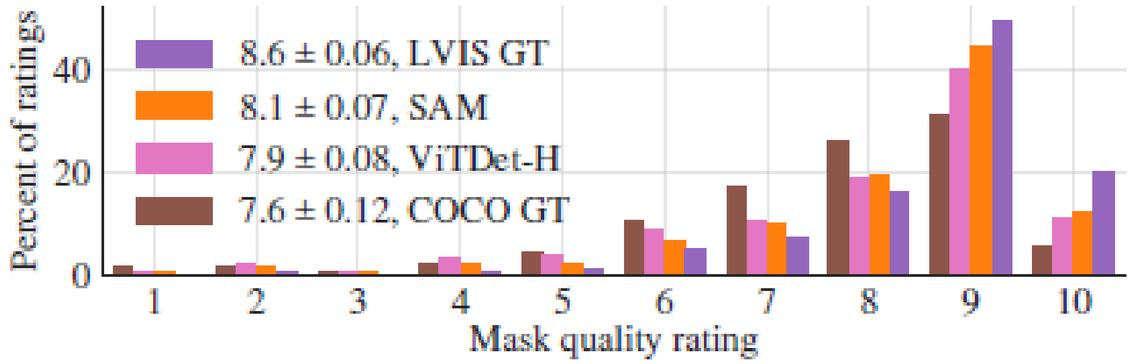
  4) Zero-Shot Instance Segmentation

  간단하게 위에서 얻은 proposal을 box prompt를 통해 segmentation을 진행

  - ViTDet와 segmentation 결과는 거의 비슷함
  - 종종 SAM이 ViTDet의 마스크보다 질적으로 더 우수하며 경계가 뚜렷함



	COCO [66]				LVIS v1 [44]			
method	AP	AP^S	AP^M	AP^L	AP	AP^S	AP^M	AP^L
ViTDet-H [62]	51.0	32.0	54.3	68.9	46.6	35.0	58.0	66.3
zero-shot transfer methods (segmentation module only):								
SAM	46.5	30.8	51.0	61.7	44.7	32.5	57.6	65.5



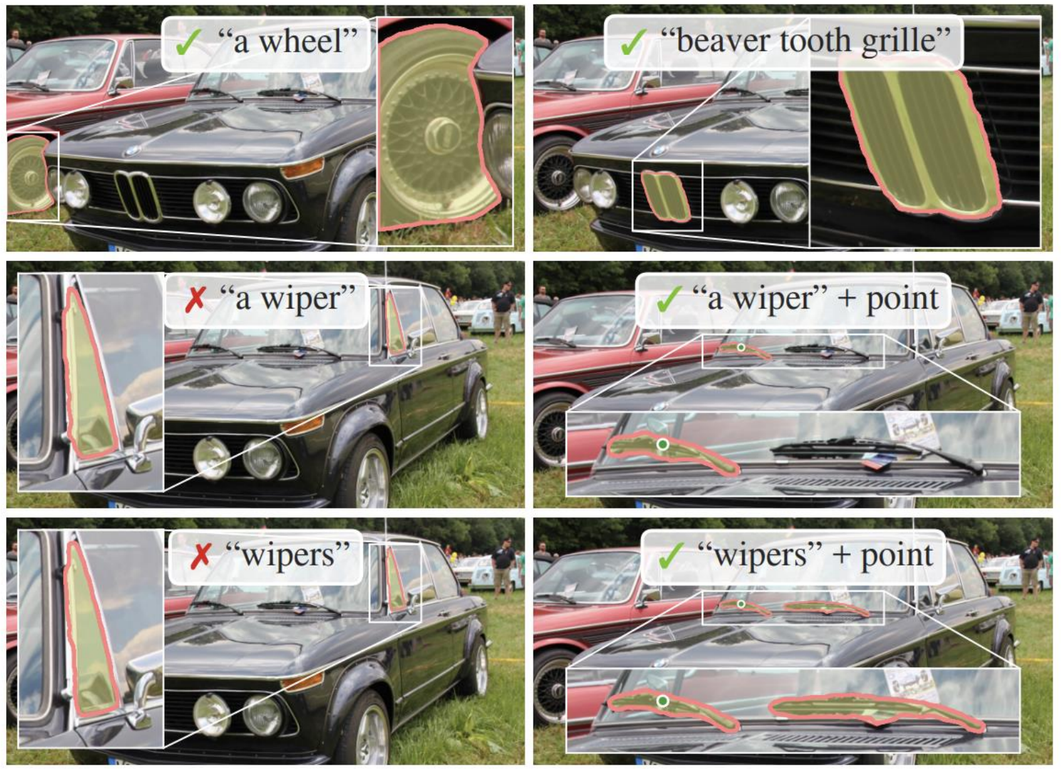
#5. Results

- **Results**

  5) Zero-Shot Text-to-Mask

  텍스트로 주어진 정보를 활용해서 segmentation mask를 생성하는 작업

  - Text prompt를 주기 위해선 새로운 annotation이 필요 -> CLIP의 image encoder 활용
  - 간단한 텍스트 프롬프트 뿐만 아니라 구절에서도 객체를 세그먼트화 할 수 있었음
  - 텍스트 프롬프트만으로 올바른 객체를 선택하지 못하는 경우 추가적인 포인트가 종종 예측을 보완함



#5. Results & Discussion

- Discussion

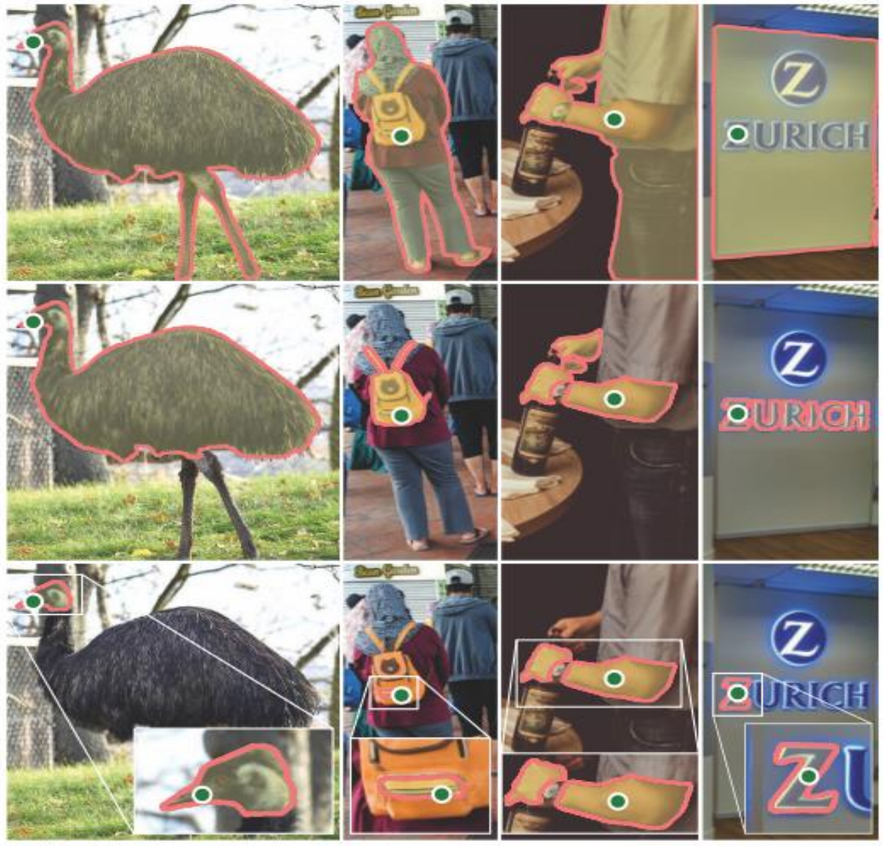
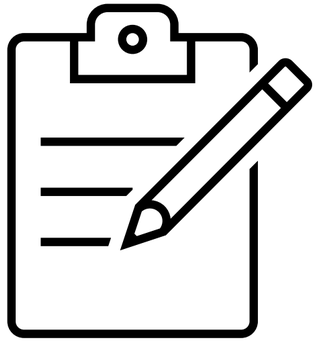


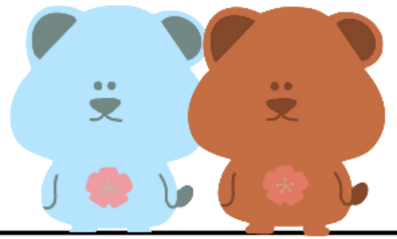
Figure 3: Each column shows 3 valid masks generated by SAM from a single ambiguous point prompt (green circle).



SAM은 일부 성능 한계를 보일 수도 있어.
특히, 많은 포인트가 제공되는 경우 이전의
interactive segmentation 방법보다
성능이 떨어질 수도 있어!

Point는 객체에 대한
정보를 제공해 주는 수단 아닌가?
그러면 point가 많을수록 prompt의
모호성을 보완해 줄 수 있는 거 아닌가?

THANK YOU



EWHA EURON