Panoptic SegFormer

Yongwon Jo

School of Industrial and Management Engineering, Korea University

KOREA UNIVERSITY DMQA

Contents

Research Purpose

❖ **Panoptic SegFormer**

- 난징 대학, 홍콩 대학, NVIDIA, Caltech 소속 연구원들이 발표한 논문
- Panoptic SegFormer는 Panoptic Segmentation을 End-to-End 방식으로 진행하는 방법론

Panoptic SegFormer

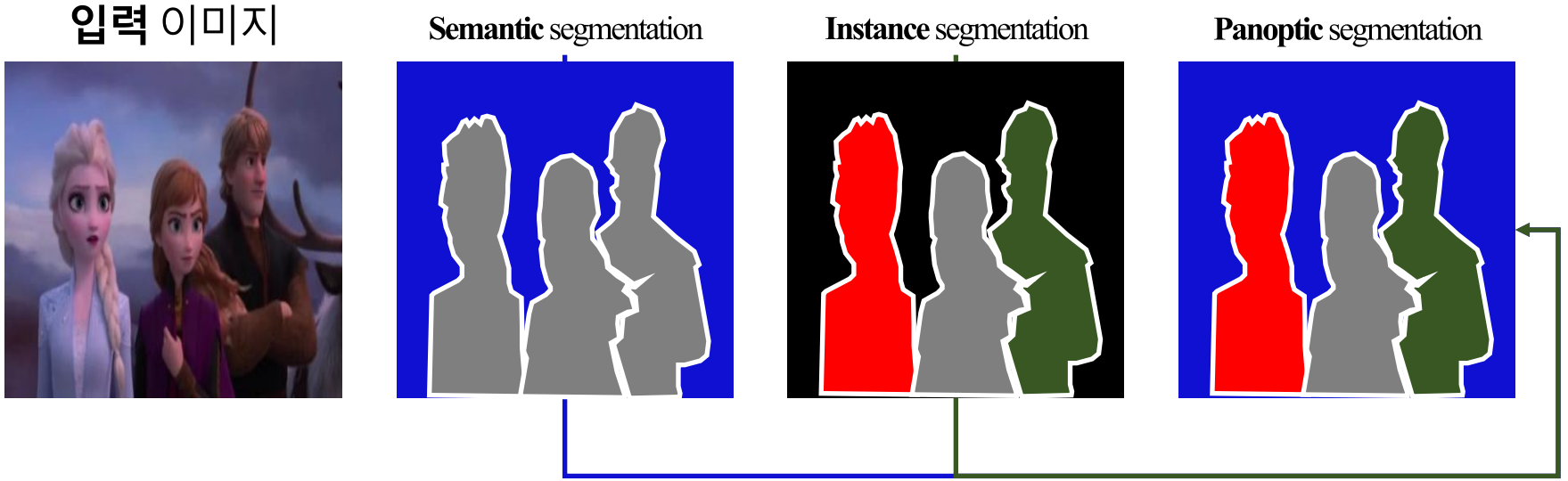
Zhiqi Li[1], Wenhai Wang[1], Enze Xie[2], Zhiding Yu[3],
Anima Anandkumar[3,4], Jose M. Alvarez[3], Tong Lu[1], Ping Luo[2]

[1]Nanjing University [2]The University of Hong Kong [3]NVIDIA [4]Caltech

Research Purpose

❖ **Panoptic segmentation**

- 배경에 대한 픽셀에 대해서는 배경이라고 분류 진행(픽셀에 특정 범주 할당-**Stuff**)
  - ➢ 파란색은 하늘을 의미하는 픽셀이며 회색은 사람을 의미하는 픽셀
- 객체들 끼리 서로 다름을 인식하며 픽셀별 분류 진행(객체간 구별-**Things**)
  - ➢ 빨간색은 엘사, 회색은 안나, 초록색은 크리스토퍼를 의미하며 사람간 구별
- 즉, Semantic segmentation을 수행하며 Instance segmentation 동시 수행



Research Purpose

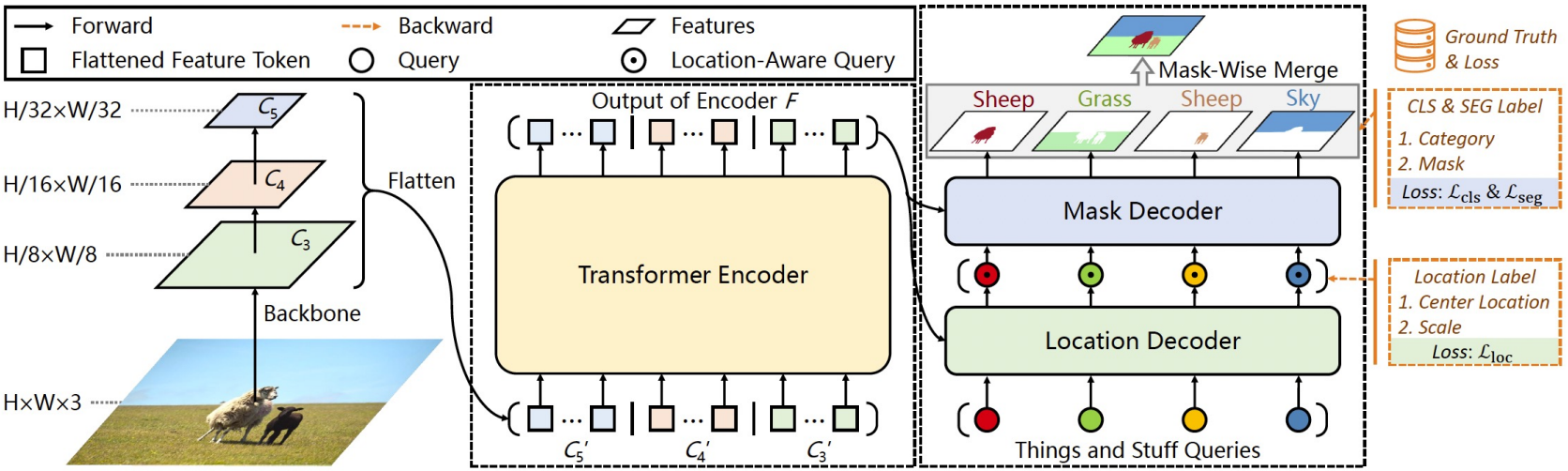
❖ **Panoptic segmentation 기존 알고리즘**

- Semantic segmentation 또는 Instance segmentation 알고리즘을 변형
- 예를 들어, Semantic segmentation 수행 후 Instance 끼리 구별을 진행하는 방식
- 또한, 동일한 특징 벡터에 대해 두 테스크 각각에 대한 Decoder구성
- 위와 같은 방식은 많은 연산량과 학습 시간이 오래 걸린다는 문제 존재

Research Purpose

❖ Panoptic SegFormer

- 개별 문제를 위한 Decoder가 아닌 두 역할을 동시에 학습하는 Decoder 존재
- End-to-End 모델이라 할 수 있으며 Panoptic segmentation을 특별한 후처리 방식 제안
- DETR 과 Deformable DETR을 기반으로 만들어진 방법론
- Stuff에 대해서는 하나의 객체를 가지는 범주로 정의하며 Instance segmentation과 유사

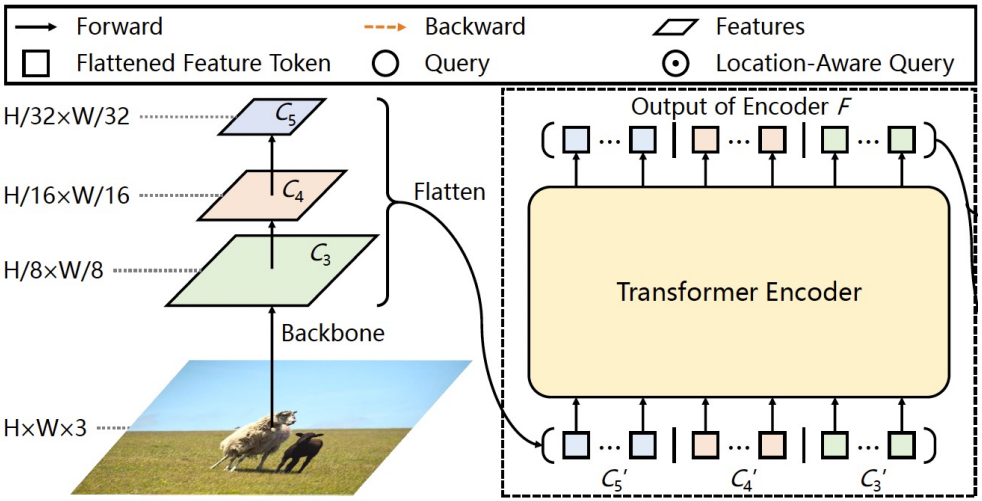


KOREA UNIVERSITY DMQA

Panoptic SegFormer

❖ Transformer encoder

- Backbone 네트워크에서 서로 다른 크기를 가진 Feature map 추출
  - ➢ ResNet 계열이나 PVT_v2 를 Backbone으로 사용
- 서로 다른 크기의 Feature map을 Flatten 후 Fully connected layer를 통과시켜 256차원으로 변환
- 변환된 Feature map 값들과 어떤 단계에서 나왔는지 의미하는 Feature token 준비
- 이들을 결합해 Transformer encoder에 입력하여 특징 벡터 산출
- Thing과 Stuff에 대해 임의로 초기화된 Query 벡터 산출

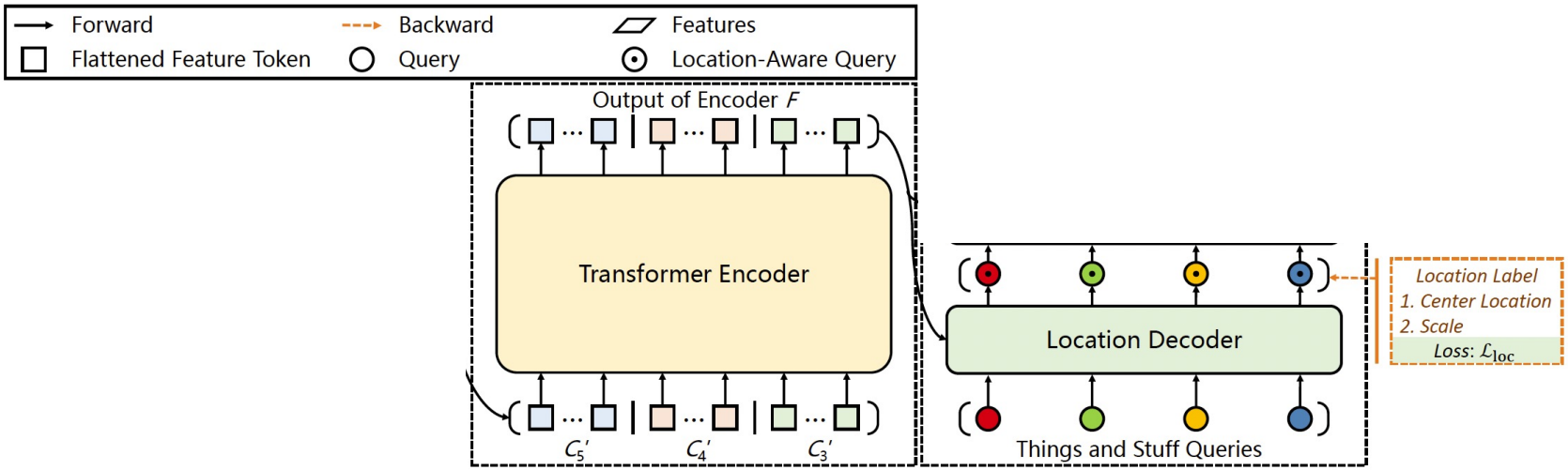


Panoptic SegFormer

❖ Location decoder

- Transformer encoder에서 나온 벡터와 Query 벡터를 Location decoder에 입력
- 개별 Query가 존재할 만 한 영역을 Location decoder가 추천
- 해당 영역이라는 것은 중심의 좌표와 객체의 크기에 대한 값을 제공
- L1 손실 함수를 사용해 학습 진행

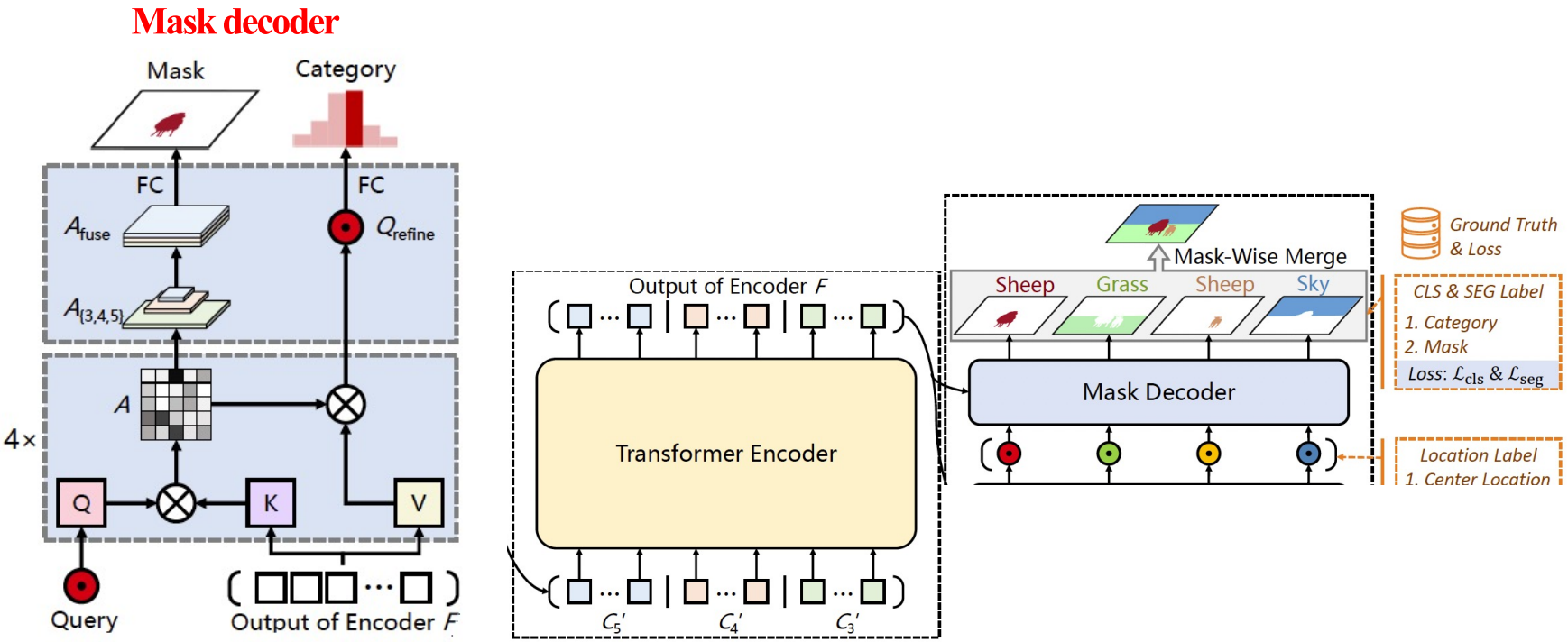
$$Loss_{loc} = \sum_{1}^{N} 1_{\{y_i=\emptyset\}}(L_1(f_c(m_i), \hat{u}_{\sigma(i)}) + (L_1(f_s(m_i), \hat{v}_{\sigma(i)})$$



Panoptic SegFormer

❖ Mask decoder

- Transformer encoder에서 산출된 특징 벡터와 위치 정보를 Mask decoder에 입력
- Mask decoder는 픽셀 별 확률 값을 가지는 출력 값(Mask)과 이에 대한 범주를 출력
- 특징 지도를 Upsampling 및 채널 축 결합을 진행하며 입력 데이터 크기와 동일하게 변경



Panoptic SegFormer

❖ Mask-Wise Merging (후처리 방식)

- Mask decoder 출력 값들을 병합하는 방식
- c: class name| s: mask에 해당되는 범주에 대한 점수| m: mask decoder의 픽셀 별 확률 값

Algorithm 1: Mask-Wise Merging

```
def MaskWiseMergeing(c,s,m):
    # category c ∈ ℝ^N
    # confidence score s ∈ ℝ^N
    # mask m ∈ ℝ^{N×H×W}
    SemMsk = np.zeros(H,W)
    IdMsk = np.zeros(H,W)
    order = np.argsort(-s)
    id = 0
    for i in order:
        # drop low quality results
        if s[i]< thr_cls:
            continue
        # drop overlaps
        m_i = m[i] & (SemMsk==0)
        SemMsk[m_i] = c[i]
        if isThing(c[i]):
            IdMsk[m_i] = id
            id += 1
    return SemMsk,IdMsk
```

Semantic mask, Index mask 초기화
점수 내림 차순 정렬

점수가 높은 범주부터 SemMsk에 입력

점수가 일정 수준 이하일 때 제외

Experiments

❖ **기존 Panoptic segmentation 알고리즘(CNN-based, ViT-based)과 비교 (COCO validation)**

- Panoptic SegFormer: Backbone 네트워크를 ResNet, PVT 두가지 사용
- 파라미터 개수가 50M 미만일 경우 중에는 최고 성능
- 파라미터 개수가 100M 이상인 경우에서도 가장 뛰어난 성능

Method	Backbone	Epochs	PQ	PQ^{th}	PQ^{st}	#Param	FLOPs
Panoptic FPN [2]	R50-FPN [24, 39]	36	41.5	48.5	31.1	-	-
SOLOv2 [12]	R50-FPN	36	42.1	49.6	30.7	-	-
DETR [15]	R50	$\sim 150 + 25$	43.4	48.2	36.3	42.8M	137G
Panoptic FCN [13]	R50-FPN	36	43.6	49.3	35.0	37.0M	244G
K-Net [14]	R50-FPN	36	45.1	50.3	37.3	-	-
MaskFormer [17]	R50	300	46.5	51.0	39.8	45.0M	181G
DETR [15]	R101	$\sim 150 + 25$	45.1	50.5	37.0	61.8M	157G
Max-Deeplab-S [16]	Max-S	54	48.4	53.0	41.5	61.9M	162G
MaskFormer [17]	R101	300	47.6	52.5	40.3	64.0M	248G
Max-Deeplab-L [16]	Max-L	54	51.1	57.0	42.2	451.0M	1846G
MaskFormer [17]	Swin-L† [20]	300	52.7	58.5	44.0	212.0M	792G
Panoptic SegFormer	R50	12	46.4	52.6	37.0	47.0M	246G
Panoptic SegFormer	R50	50	50.0	56.1	40.8	47.0M	246G
Panoptic SegFormer	R101	50	50.4	56.3	41.6	65.9M	322G
Panoptic SegFormer	PVTv2-B0 [40]	50	49.6	55.5	40.6	22.2M	156G
Panoptic SegFormer	PVTv2-B2 [40]	50	52.6	58.7	43.3	41.6M	219G
Panoptic SegFormer	PVTv2-B5 [40]	50	54.1	60.4	44.6	100.9M	391G

Experiments

❖ **기존 Panoptic segmentation 알고리즘(CNN-based, ViT-based)과 비교 (COCO Test)**

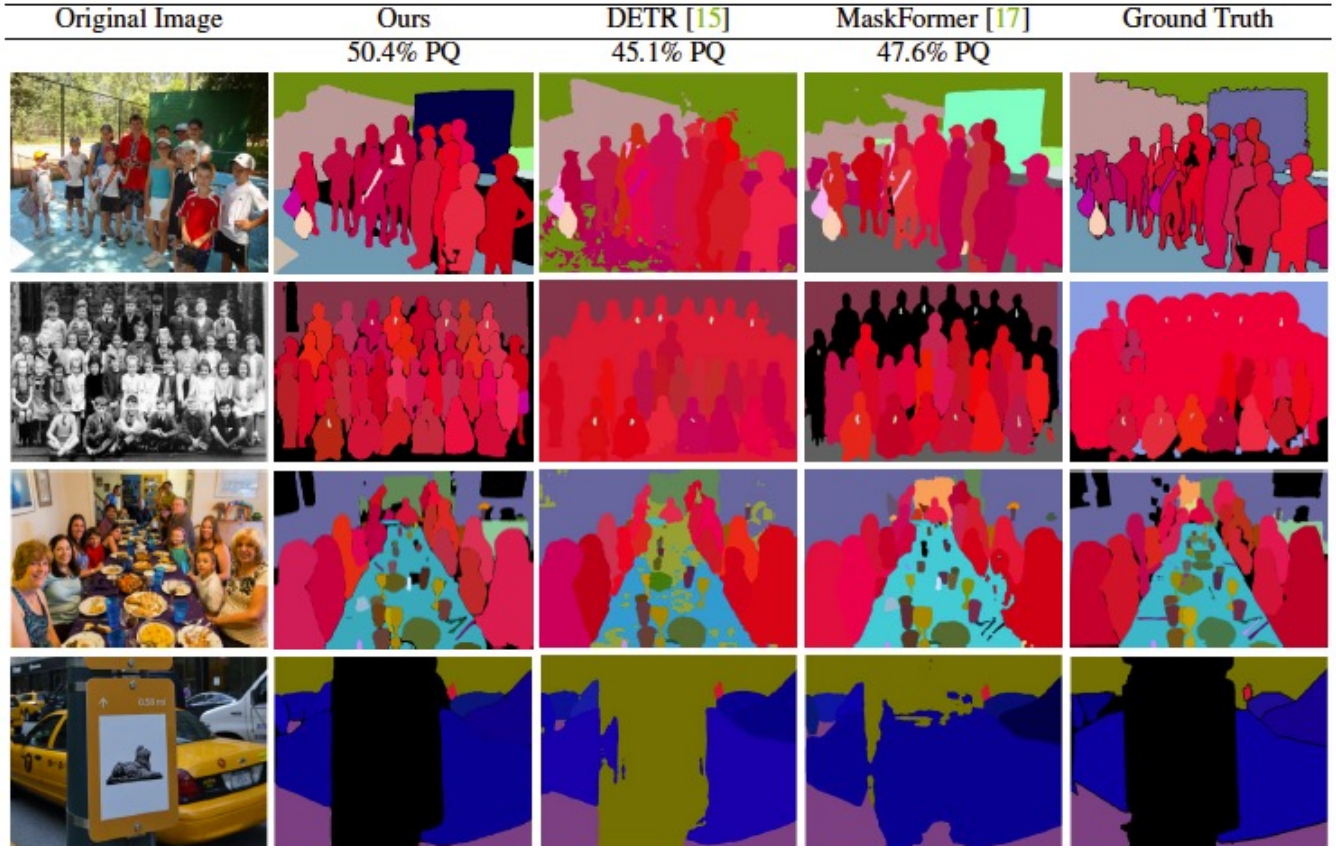
- Panoptic SegFormer는 DETR을 베이스로 개발된 알고리즘
- DETR 대비 적은 Epoch으로도 뛰어난 성능

Method	Backbone	Epochs	PQ	PQ^{th}	PQ^{st}	#Param	FLOPs
Panoptic FPN [2]	R101-FPN	36	43.5	50.8	32.5	-	-
DETR [15]	R101	$\sim 150 + 25$	46.0	-	-	61.8M	157G
Panoptic FCN [13]	R101-FPN	36	45.5	51.4	36.4	56.0M	310G
K-Net [14]	R101-FPN	36	47.0	52.8	38.2	-	-
Max-Deeplab-S [16]	Max-S [16]	54	49.0	54.0	41.6	61.9M	162G
K-net [14]	Swin-$L^{\dagger}$	36	52.1	58.2	42.8	-	-
Max-Deeplab-L [16]	Max-L [16]	54	51.3	57.2	42.4	451.0M	1846G
Innovation [22]	ensemble	-	53.5	61.8	41.1	-	-
Panoptic SegFormer	R50	50	50.0	56.2	40.8	47.0M	246G
Panoptic SegFormer	R101	50	50.9	57.1	41.4	65.9M	322G
Panoptic SegFormer	PVTv2-B5 [40]	50	54.4	61.1	44.3	100.9M	391G

Experiments

❖ Panoptic segmentation 결과 시각화

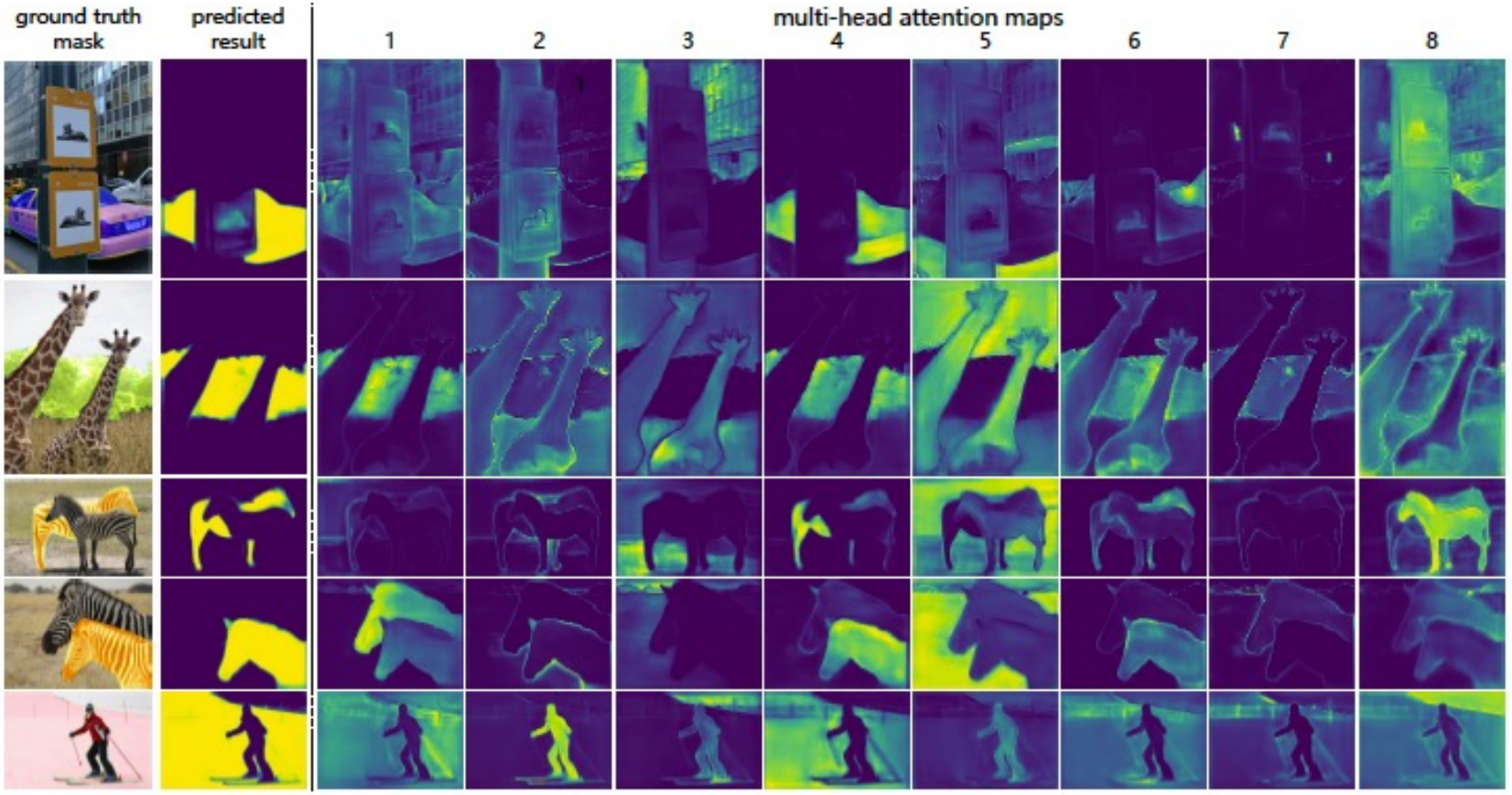
- 논문에서 제안하는 후처리 방식을 통해 사람 사이를 정확히 구분
- 과거 Bottom-Up 방식은 객체간 구별이 어려웠지만 이를 해결한 것으로 보임



Experiments

❖ **Location decoder 내 Multi-head attention map 시각화 결과**

- 객체 가장자리를 인식하고 있음을 확인 가능하며 객체에만 집중하는 Attention
- Thing에 집중할 때는 Thing만, Stuff에 집중할 때는 Stuff에만 집중



Conclusion

❖ Conclusion

- Panoptic SegFormer는 ViT로만 구성된 Panoptic Segmentation 모델
- Transformer encoder, Location decoder, Mask decoder와 Mask-wise merging 후처리 기법 제안
- 기존 State-of-the-art를 뛰어넘는 성능을 보여줌

❖ 본 논문에 대한 나의 생각

- 성능은 매우 뛰어나지만 파라미터 수가 성능 자체에만 집중한 것으로 보임
- 학교에서 연구 진행 시 경량화나 간단한 Backbone 네트워크를 사용한 연구를 해야할 것
- (Semantic, Instance, Panoptic) segmentation 모두 ViT 기반 알고리즘이 주를 이룸
- 대부분 Bilinear Upsampling을 사용하지만 Transposed ViT와 같은 네트워크에 대한 갈망

Thank you