(SenNet + HOA - Hacking the Human Vasculature in 3D)

신장 내 혈관 분할

CV SEGMENTATION 1팀
김송성 김연규 최유민

(CONTENTS)

목차

(PRE-STUDY)

Detection & Segmentation

Paper Review

- R-CNN
- YOLO
- U-Net
- SSD
- FPN
- ViT
- SegFormer
- SAM

Toy Project

- 위성 이미지 건물 영역 분할 (데이콘)
- Camera-Invariant Domain Adaptation (데이콘)

Kaggle

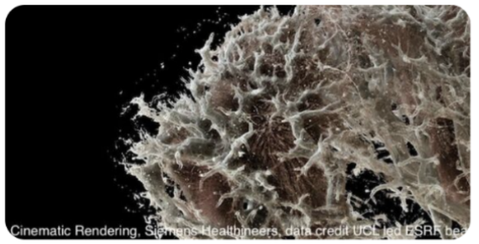
- 3D Segmentation Model 리뷰
- Method for medical domain 리뷰

(OVERVIEW)

대회 소개

SenNet + HOA - Hacking the Human Vasculature in 3D

Segment vasculature in 3D scans of human kidney



3D Hierarchical Phase-Contrast Tomography (HiP-CT)로 촬영된 인체의 신장 데이터에서 **혈관을 분할.**

-> 몸 전체의 혈관 구조 사진을 완성하는 것 도움

-> 인체 조직의 혈관의 크기, 모양, 패턴 등에 대한 연구자의 이해 도움

(OVERVIEW)

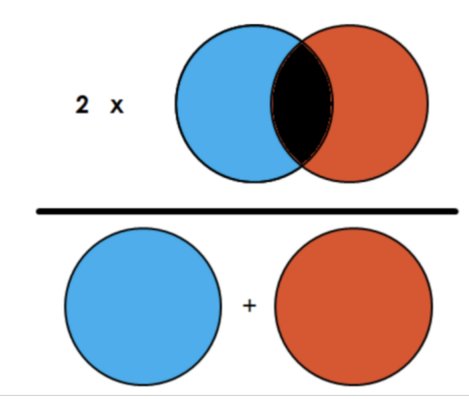
대회 소개

평가지표 | Surface Dice Metric (tolerance : 0.0)

- 두 표면 사이의 겹침을 측정
- 다른 표면과 가장 가까운 거리가 지정된 tolerance보다 작거나 같으면 표면 겹침으로 계산
- 0.0(겹침 없음) ~ 1.0(완전히 겹침)

```
overlap_gt = np.sum(surfel_areas_gt[distances_gt_to_pred <= tolerance_mm])
overlap_pred = np.sum(
    surfel_areas_pred[distances_pred_to_gt <= tolerance_mm])
surface_dice = (overlap_gt + overlap_pred) / (np.sum(surfel_areas_gt) +
                                              np.sum(surfel_areas_pred))
return surface_dice
```

$$Dice = \frac{2 * |A \cap B|}{|A| + |B|} = \frac{2 * TP}{(TP + FP) + (TP + FN)}$$



(DATA)

데이터 소개

- 2D slice of a 3D volume 데이터
- kidney1, 2, 3 세 사람에 대한 image 및 mask
- 각각의 분할된 정보나 비율이 다름

- kidney_5 & kidney_6 : test set

kidney_1

- dense : 50um 해상도의 오른쪽 신장 전체
- voi : 5.2um 해상도에서 kidney_1의 고해상도 부분 집합

kidney_2

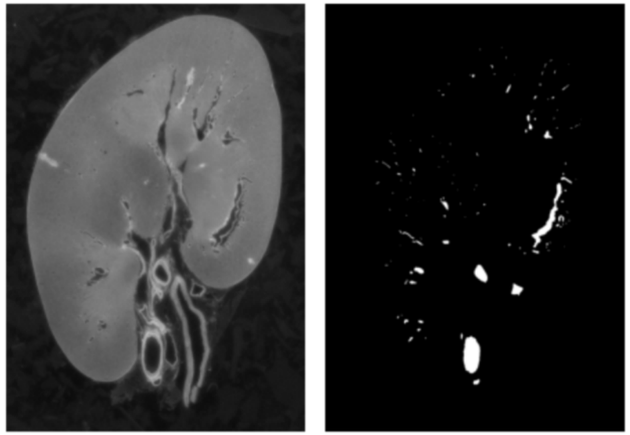
- 50um 해상도의 신장 전체
  Sparsely segmented (about 65%)

kidney_3

- dense : BM05를 사용한 50.16um 해상도의 신장 부분(500 slice)
  Densely segmented
- spare : kidney_3의 나머지 분할 mask
  Sparsely segmented (about 85%).

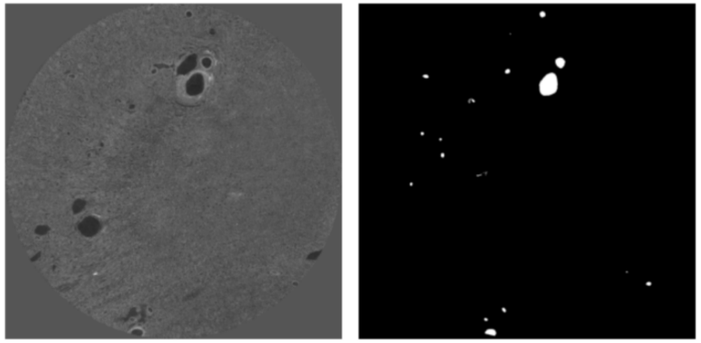
(DATA)

데이터 소개

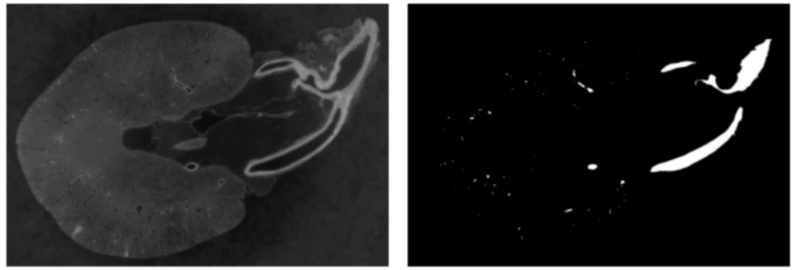


kidney_1_dense
(2279, 1303, 912)

Z X Y



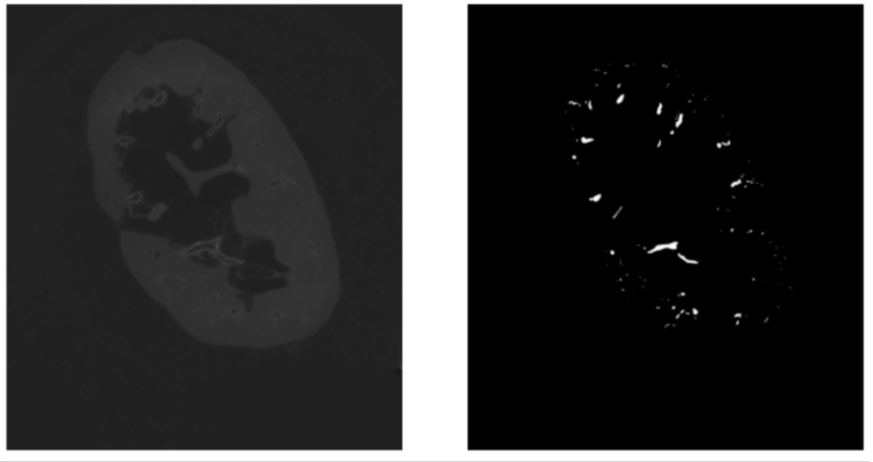
kidney_1_voi
(1397, 1928, 1928)



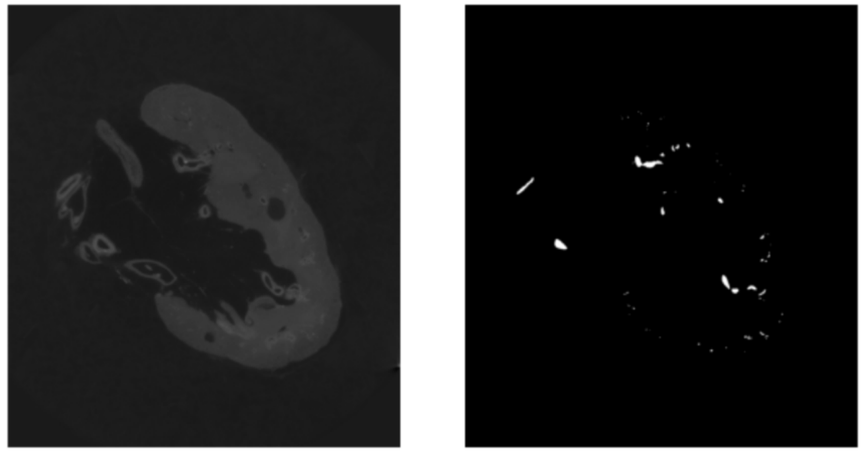
kidney_2
(2217, 1041, 1511)

(DATA)

데이터 소개



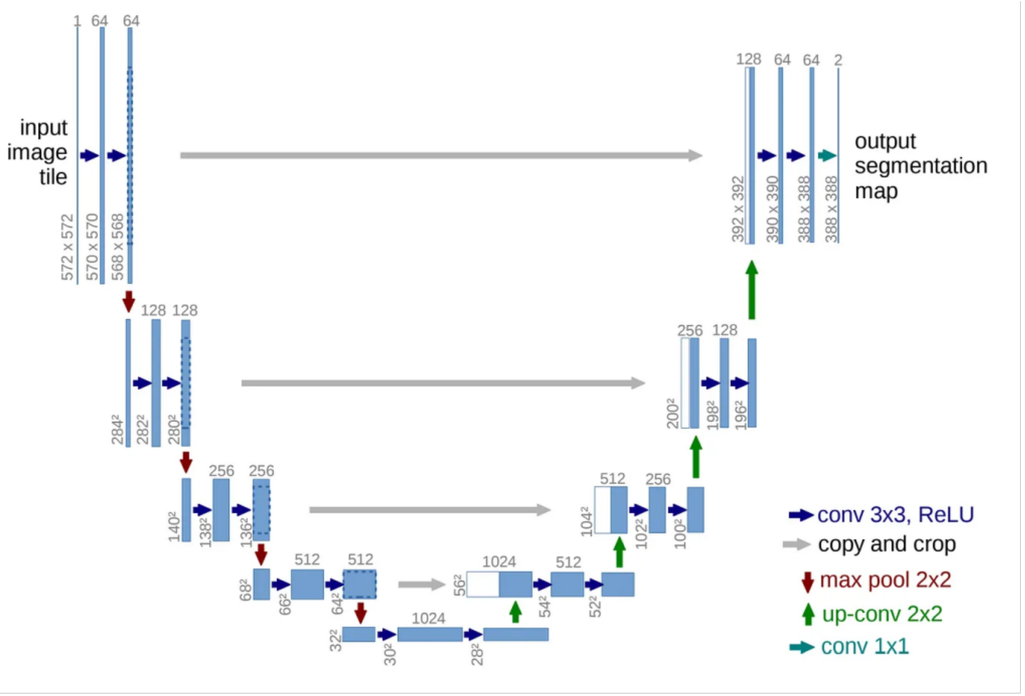
kidney_3_dense
(501, 1706, 1510)



kidney_3_sparse
(1035, 1706, 1510)

(MODELING)

U-Net



Segmentation Models

Segmentation Models Pytorch(SMP) Library
https://github.com/qubvel/segmentation_models.pytorch

```
self.encoder = smp.Unet(
    encoder_name=CFG.backbone,
    encoder_weights=weight,
    in_channels=CFG.in_chans,
    classes=CFG.target_size,
    activation=None,
)
```

- U-Net Architecture
- ResNeXt, SE-Net, DenseNet, EfficientNet 등의 encoder
- pre-trained weights (imagenet / instagram)

(MODELING)

U-Net

2.5d Cutting Model

- CFG.in_chans = 5

```
next(iter(train_dataset))[0].shape
✓ 3.4s
torch.Size([16, 5, 256, 256])
```

```
next(iter(train_dataset))[1].shape
✓ 3.7s
torch.Size([16, 256, 256])
```

```
def load_data(path,s):
    data_loader=Data_loader(path,s)
    data_loader=DataLoader(data_loader, batch_size=16, num_workers=2)
    data=[]
    for x in tqdm(data_loader):
        data.append(x)
    return torch.cat(data,dim=0)
```

```
x_index = np.random.randint(0,x.shape[1]-self.image_size)
y_index = np.random.randint(0,x.shape[2]-self.image_size)
```

```
x = x[index:index+self.in_chans,x_index:x_index+self.image_size,y_index:y_index+self.image_size].to(torch.float32)
y = y[index+self.in_chans//2,x_index:x_index+self.image_size,y_index:y_index+self.image_size].to(torch.float32)
```

```
data = self.transform(image=x.numpy().transpose(1,2,0), mask=y.numpy())
x = data['image']
y = data['mask']
```

(MODELING)

U-Net

```
self.encoder = smp.Unet(
    encoder_name=CFG.backbone,
    encoder_weights=weight,
    in_channels=CFG.in_chans,
    classes=CFG.target_size,
    activation=None,
)
```

lr = 6e-5
optimizer=torch.optim.AdamW
torch.optim.lr_scheduler.OneCycleLR

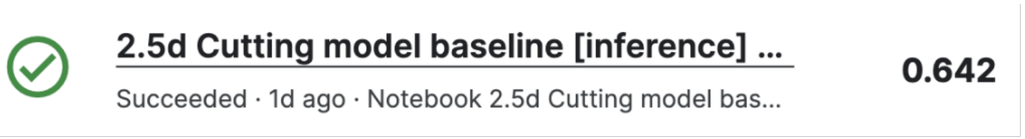
backbone	weight	loss_fn	image_size	epoch	leaderboard
se_resnext50_32x4d (Params : 25M)	imagenet	FocalLoss	256	15	0.252
se_resnext50_32x4d (Params : 25M)	imagenet	BCEWithLogitsLoss	256	15	0.431
se_resnext50_32x4d (Params : 25M)	imagenet	DiceLoss	256	15	0.556
densenet161 (Params : 26M)	instagram	DiceLoss	256	15	0.577
efficientnet-b6 (Params : 40M)	imagenet	DiceLoss	256	15	0.564

(MODELING)

U-Net

```
self.encoder = smp.Unet(
    encoder_name=CFG.backbone,
    encoder_weights=weight,
    in_channels=CFG.in_chans,
    classes=CFG.target_size,
    activation=None,
)
```

lr = 6e-5
optimizer=torch.optim.AdamW
torch.optim.lr_scheduler.OneCycleLR



backbone	weight	loss_fn	image_size	epoch	leaderboard
se_resnext101 _32x4d (Params : 46M)	imagenet	DiceLoss	256	30	0.596
resnext101 _32x8d (Params : 86M)	imagenet	DiceLoss	256	30	0.642
resnext101 _32x16d (Params : 191M)	instagram	DiceLoss	256	40	0.582
resnext101 _32x8d (Params : 86M)	imagenet	DiceLoss	512	40	0.617

(MODELING)

SAM

(MODELING)

SAM

(MODELING)

SCNAS

(MODELING)

SCNAS

(CONCLUSION)

결론