

CV-11 랩업 리포트

Level-2 Hand Bone Image Segmentation 대회 리뷰
CV-11조 팀 SEE-Beyond-Accuracy

1. Project outline

뼈는 우리 몸의 구조와 기능에 중요한 영향을 미치기 때문에, 정확한 뼈 분할은 의료 진단 및 치료 계획을 개발하는 데 필수적입니다.

Bone Segmentation은 인공지능 분야에서 중요한 응용 분야 중 하나로, 특히, 딥러닝 기술을 이용한 뼈 Segmentation은 많은 연구가 이루어지고 있으며, 다양한 목적으로 도움을 줄 수 있습니다.

질병 진단의 목적으로 뼈의 형태나 위치가 변형되거나 부러지거나 골절 등이 있을 경우, 그 부위에서 발생하는 문제를 정확하게 파악하여 적절한 치료를 시행할 수 있습니다.

수술 계획을 세우는데 도움이 됩니다. 의사들은 뼈 구조를 분석하여 어떤 종류의 수술이 필요한지, 어떤 종류의 재료가 사용될 수 있는지 등을 결정할 수 있습니다.

의료 장비 제작에 필요한 정보를 제공합니다. 예를 들어, 인공 관절이나 치아 임플란트를 제작할 때 뼈 구조를 분석하여 적절한 크기와 모양을 결정할 수 있습니다.

의료 교육에서도 활용될 수 있습니다. 의사들은 병태 및 부상에 대한 이해를 높이고 수술 계획을 개발하는 데 필요한 기술을 연습할 수 있습니다.

1.1 대회 를

- 외부 데이터셋 사용 금지
- 모든 기학습 가중치 사용 허용

1.2 기간

- 11월 11일 (월) 10:00 ~ 11월 28일 (목) 19:00

1.3 Dataset

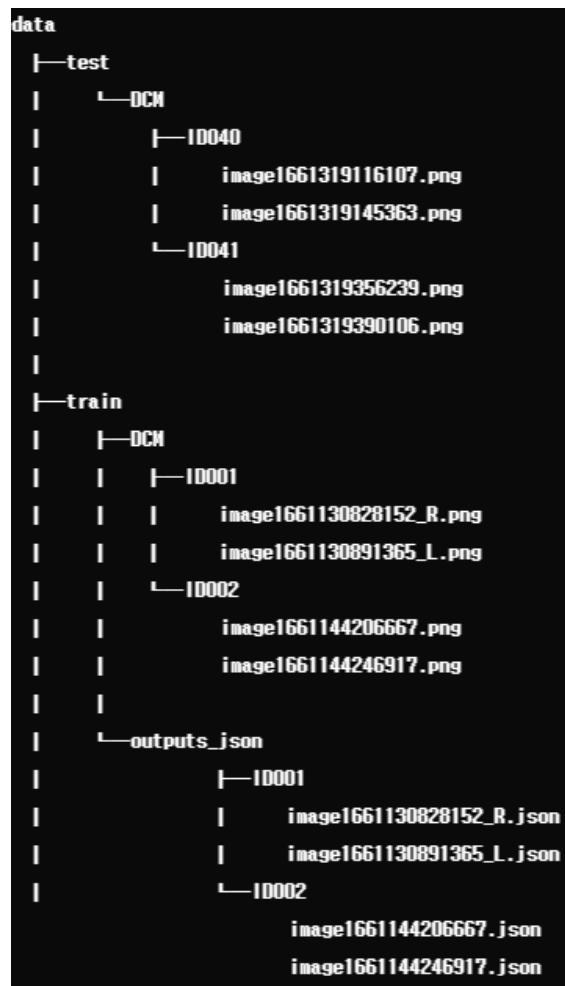


그림 1. dataset 구조도

Total (Train / Test) : 1088 (800 / 288)

29 Classes : finger 1 ~ finger 19, Trapezium, Trapezoid, Capitate, Hamate, Scaphoid, Lunate, Triquetrum, Pisiform, Radius, Ulna

1.4 평가 Metric

- Dice Coefficient

$$DICE = \frac{2 \times |A \cap B|}{|A| + |B|}$$

그림 2. Dice Coeff 계산 식

2. Team Members and Roles

Name	Role
임용섭	EDA, Modeling, MLflow
박재우	EDA, Modeling
이상진	EDA, Modeling, Refactoring, Loss Test, Backend
정지훈	EDA, baseline test, mmseg test
천유동	EDA, Augmentation test
유희석	EDA, Annotation, Refactoring, Kfold

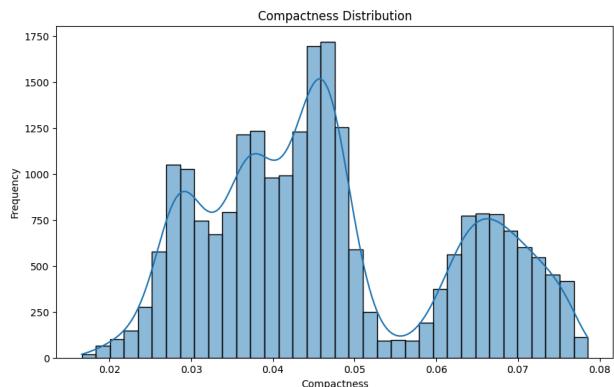


그림 5. 복잡도 분포 히스토그램

3. Project

3.1 EDA

✓ 전체 분포

A. 클래스 별 위치 분포

양손의 데이터이기 때문에 각 손의 위치 분포는 대칭 형태를 띈다.

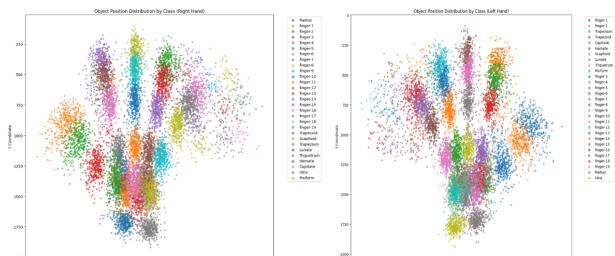


그림 3. 양손의 클래스별 위치 분포 이미지

B. Multi Label, Occlusion

뼈가 겹치는 부분에서 같은 픽셀에 여러 레이블이 존재하는 경우가 생긴다. - 주로 손등 뼈

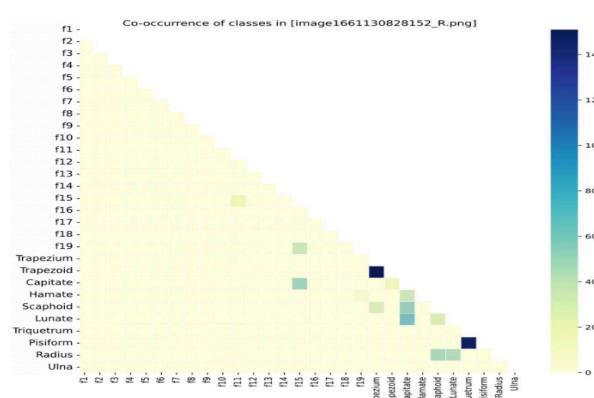


그림 4. Multi Label 분포 그래프

C. 복잡도 분포

D. 밝기 분포

전반적으로 어두운 경향을 보인다.

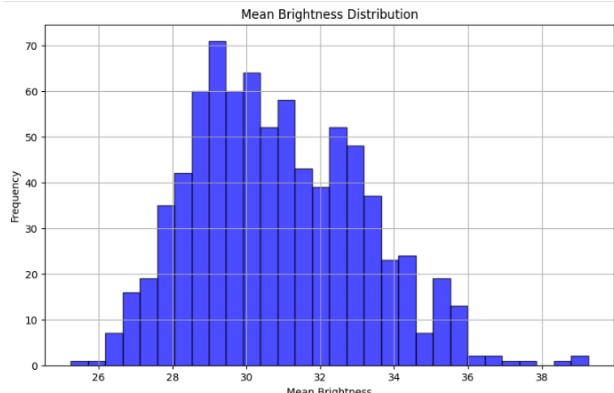


그림 6. 평균 밝기 분포 히스토그램

E. 대비 분포

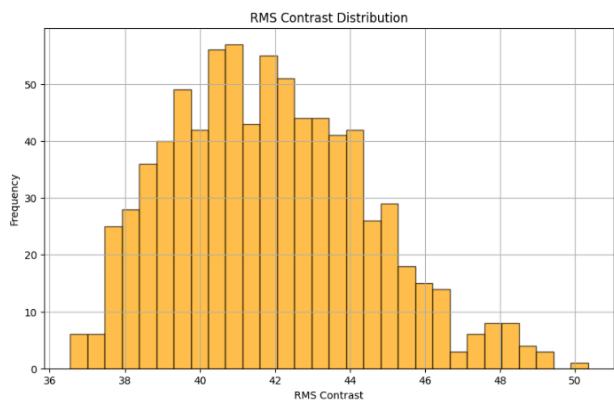


그림 7. 대비 분포 히스토그램

F. 배경 비율

밝기가 존재하는 부분 (손)이 30~40% 정도로 전체 이미지에서 배경이 차지하는 부분이 넓다.

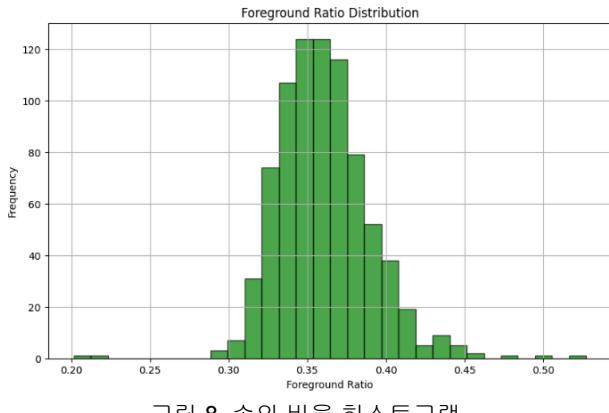


그림 8. 손의 비율 히스토그램

G. 손의 기울기

중지의 기울기의 분포를 통하여 얼마나 손이 기울어져 있는지 파악

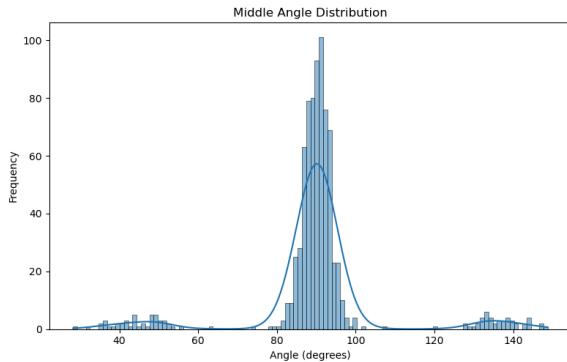


그림 9. 손의 기울기 분포 히스토그램

✓ 이상치

1. 이미지에 이물질이 들어간 경우

Train ID 363에서 반지가 끼워져 있는 경우가 있는데 반지에 annotation이 되어 있다.

Train ID 387에서 손톱에 보석처럼 네일아트가 박혀 있다.

Train ID 487에서 손목 뼈에 붙어 있는 보철물이 있다.

2. 라벨링 오류

엄지손가락 뼈와 손등의 뼈 간에 위치가 바뀌어서 라벨링이 되어 있다.

Train ID 058에서 Finger 1과 Finger 3의 위치가 바뀌었고 손등의 뼈들의 위치가 거의 전부 바뀌어 있는 문제가 있다.

3.2 K-Fold

데이터셋 확보가 어려운 의료 데이터다 보니 학습 데이터셋을 온전히 활용해야겠다고 판단해 K-Fold기법을 활용해 총 5개의 fold로 분리 했다.

모든 fold를 모두 학습해야 하기 때문에 학습

시간이 오래 걸린다는 단점이 있다. 이로 인해 5개의 fold 중 public score와 가장 비슷한 분포를 가진 fold 하나만을 이용해 실험. 최종 제출 시 모든 fold를 학습해 양상을 하기로 결정했다.

FOLD	DICE	Public score(DICE)	Diff
0	0.3751	0.3555	0.0196
1	0.4182	0.3981	0.0201
2	0.4864	0.4558	0.0306
3	0.4725	0.4518	0.0207
4	0.4612	0.4464	0.0148

표 1. 동일 조건에서 30 Epoch으로 실험했을 때 결과. FOLD 4번에서 가장 적은 점수 차이를 보였다.

3.3 Data Preprocessing

✓ Annotation 수정

Supervised Learning 특성상 완전히 잘못된 정답을 주고 학습을 시키면 학습의 효율이 떨어진다고 판단했다. 잘못된 라벨링이 되어 있는 Train ID 363, Train ID 058에 대해서 Annotation을 수정했다.

Annotation	DICE
수정 전	0.9638
수정 후	0.9644

표 2. Annotation 수정 전, 후의 성능차이를 확인하기 위해서 fcn_resnet50 모델에서 50 Epoch 동일 조건에서 실험을 진행 Annotation 수정 후 DICE score가 소폭 상승했다.

✓ Preprocessing

(1) Resize

Size (width = height)	DICE
512 (default)	0.9476
1024	0.9673
1536	0.9629

입력 이미지를 3가지 크기(512, 1024, 1536)로 조정하여 테스트한 결과, **1024**에서 가장 높은 Dice Score를 기록했다.

512의 경우, 해상도가 낮아 중요한 경계와 세부 정보를 누락하거나 학습하지 못해 성능이 저하될 것으로 예상했으며, 결과 또한 그에 부합하게 나타났다.

그러나 **1536**으로 1024보다 큰 이미지를 입력으로 모델에 넣었을 때, 오히려 성능이 떨어지는 현상이 발생했다. 이는 큰 입력 이미지가 모델이 과도하게 경계와 세부 정보에 집중하도록 만들어, 오히려 일반화 성능이 저하되었을 가능성이 있다고 예상된다.

(2) Augmentation

DuckNet에서 512*512의 이미지 입력 크기와 10번의 에포크를 기반으로 다양한 상황에서 검출 성능을 유지하기 위한 증강기법들을 조합해서 학습을 진행하였다.

Method	avg Dice
None	0.9249
Normalization	0.9394
Horizontal Flip	0.5373
CLAHE	0.9396
RandomBrightness	0.9343
ElasticTransform	0.9399
Rotate 16	0.4102
CLAHE, Normalization	0.9375
Normalization + Center Crop	0.9385
ElasticTransform, Normalization	0.9403
ElasticTransform, CLAHE	0.9397
ElasticTransform, CLAHE, Normalization	0.9384

*CLAHE : Contrast Limited Adaptive Histogram Equalization (타일 별 히스토그램 균일화)

Normalization : 주어진 Hand Bone Dataset이 Grayscale 이미지이므로 평균과 표준편차를 0.5로 적용하여 표준화를 진행하였다.

- 결과 : 기존 Base에서 1.5%p 정도의 성능 향상을 보였다.

Horizontal Flip : Dataset 한 쌍의 팔 이미지를 가지고 있어 Flip을 사용하여 다양성을 증가시키려 사용하였다.

- 결과 : 팔의 다양성을 증가시키려 했지만 오히려 점수가 떨어지는 현상이 발생하였다. Flip 후 데이터가 실제 팔의 자연스러운 대칭 구조를 반영하지 못하여 Dice Score가 떨어지지 않았나 예상한다.

CLAHE : Xray Dataset의 경우 배경이 어두워서 우리의 GT인 뼈를 강조하며, 뼈의 디테일함을 조정하기 위해 사용하였다.

- 결과 : 기존 Base에서 1.5%p 정도의 성능 향상을 보였다.

Random Brightness : 전반적으로 이미지 Data들이 어두운 편이기 때문에 밝기를 조정하여 성능을 향상시키기 위해 사용하였다.

- 결과 : 기존 Base에서 1%p 정도의 성능 향상을 보였다.

Center Crop : 손등 뼈 중 Occlusion 문제였던 2개의 클래스 쌍에 대한 많은 정보와 배경영역의 축소로 인하여 사용하였다.

- 결과 : Occlusion 문제였던 클래스 쌍에 대한 성능이 소폭 향상되었지만 손등뼈에 집중하는 문제가 있어 다른 클래스의 성능이 떨어지는 현상이 발생하였다.

Elastic Transform : 사람마다 뼈의 모양이 상이하기

때문에 변형을 주어 일반화 성능의 향상을 기대하고 사용하였다.

- 결과 : 기존 Base에서 1.5%p 정도의 성능 향상을 보였다.

Rotate 16 : EDA 과정에서 손의 기울기를 중지의 기울기를 기준으로 계산하였다. 해당 분포의 표준편자는 약 16도로 확인되었으며, 데이터 증강 과정에서 평균 ± 표준편자 범위(약 74 ~ 106도)를 고려하여 증강을 적용시켰다.

- 결과 : 손가락 뼈를 기준으로 기울기 분포를 계산하여 회전 증강을 적용했으나, 이는 손등 뼈와 팔 뼈의 실제 분포 및 특성과 불일치하여 Dice Score 감소로 이어졌을 가능성이 높다.

Augmentation 2+ : Augmentation 중 **Elastic Transform + Normalization**이 제일 성능이 좋았고 3개 이상으로 증가 시키면 오히려 기존 손뼈의 특징을 해치거나 모델이 다양한 Augmentation이 학습에 과도하게 참여하게 되면서 중요한 특징보다 불필요하게 증강에 민감히 반응하여 Dice Score를 낮춘다 판단하였다.

3.4 Loss

DICE Loss : Dice Loss는 클래스 간 불균형한 픽셀 수를 효과적으로 처리하고, 겹쳐있는 경계를 강조하기에 Hand Bone Task에 효과적이라 생각하여 채택하였다.

Binary Cross Entropy with Logits : Dice가 겹치는 영역에 초점을 맞춘다면 BCE는 픽셀 수준의 확률 예측을 보완하기 위해 채택하였다.

IoU Loss : Dice와 마찬가지로 클래스 간 분리도를 개선하고 전체 영역에 대한 학습을 강화하기 위해 채택하였다.

Focal, Focal Tversky Loss : 쉽게 예측된 샘플에 대한 손실을 줄이고 어려운 손등과 같은 곳에 집중하도록 Focal loss를 사용하였으며, 또한 Focal Tversky Loss를 사용하여 더욱 미세한 경계를 학습하도록 하였다.

Method (loss * weight)	Avg Dice
BCE (Binary Cross Entropy with Logits)	0.4864
DICE * 0.5 + BCE * 0.5	0.7782
DICE * 0.8 + BCE * 0.2	0.9063
DICE * 0.9 + BCE * 0.1	0.9200
DICE * 1.0 + BCE * 0.1	0.9182

D * 0.9 + B * 0.1 + IoU * 0.3	0.9258
D * 0.9 + B * 0.1 + IoU * 0.5	0.9263
D * 0.9 + B * 0.1 + IoU * 0.7	0.9273
D * 0.9 + B * 0.1 + IoU * 0.9 (best)	0.9279
D * 0.9 + B * 0.1 + IoU * 1.0	0.9274
Best 3 Loss + Focal * 0.3	0.8446
Best 3 Loss + Focal * 0.5	0.8430
Best 3 Loss + Focal * 0.7	0.8394
Best 3 Loss + Focal * 0.9	0.8363
Best 3 Loss + Focal Tversky * 0.1	0.8463
Best 3 Loss + Focal Tversky * 0.3	0.8479
Best 3 Loss + Focal Tversky * 0.5	0.8499
Best 3 Loss + Focal Tversky * 0.7	0.8502
Best 3 Loss + Focal Tversky * 0.9	0.8502
Best 3 Loss + Focal Tversky * 1.5	0.8513

*D: Dice, B: BCE

다양한 Loss 조합을 테스트한 결과, 각 Loss의 조합과 가중치에 따라 성능이 다르게 나타났다.

최적의 성능을 낸 조합은 **DICE * 0.9 + BCE * 0.1 + IoU * 0.9**로 0.927의 Avg Dice score를 보였다.

특히, 겹쳐 있는 클래스 (**Trapezoid**, **Trapezium**) 및 (**Triquetrum**, **Pisiform**)간의 경계 문제를 해결하기 위해 Focal loss, Focal Tversky Loss를 사용한 결과, 기존 조합 대비 약 **0.01 ~ 0.02** 정도의 성능 개선이 있었다. 하지만 Focal Loss, Focal Tversky loss에 0.9이상의 가중치를 부여할 경우, “**finger-16**” 클래스 예측 확률이 0으로 수렴하는 문제가 발생, 이를 Focal 계열의 Loss가 특정 클래스에 과도하게 집중하여, 상대적으로 다른 클래스의 학습이 이루어 지지 않는거라 판단된다.

따라서, Focal Loss 계열은 안정성과 성능을 동시에 유지하기 위해 **0.3 ~ 0.7** 범위에서 학습을 진행하였다.

3.5 Modeling

SegFormer

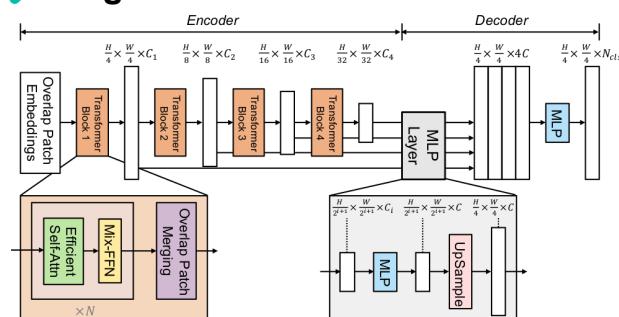


그림 10. SegFormer 아키텍처

Transformer 기반의 Semantic Segmentation 모델로 Global Context와 Local Detail을 학습

손의 전체 이미지와 부분을 학습하면 좋은 성능을 낼 것이라 생각하여 모델 구현

✓ Multi-Scale Feature Learning

해상도를 줄이고, 채널 수를 증가시키며 다양한 크기의 Feature 추출

✓ Efficient Self-Attention

Self-Attention의 시간 복잡도를 줄이기 위해 채널 수를 스케일링 하는 방식을 사용하여 시간 복잡도를 줄임

✓ Overlap Patch Embedding

Local Context와 Global Context를 동시에 학습 및 Local Continuity를 보존하여 정보 손실 감소

✓ ResNet vs SegFormer

Model	Parameter	Pretrained	DICE
FCN-ResNet50	35.3M	True	0.9249
SegFormer	98.6M	False	0.9646

DUCKNet

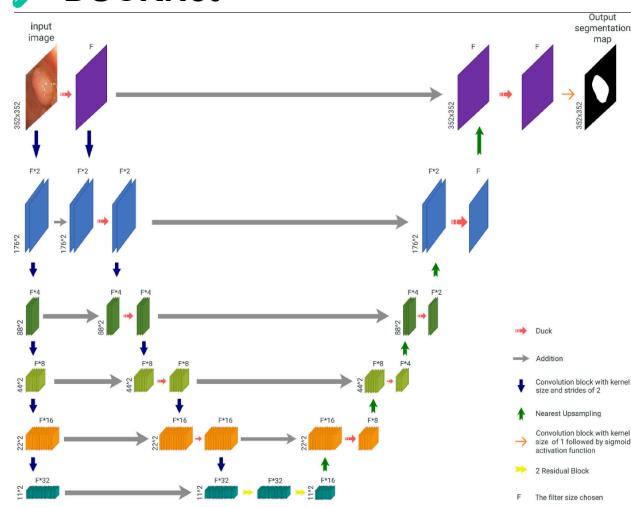


그림 11. DUCKNet 아키텍처

DUCKNet은 대장내시경 용종 분할 작업을 위해 개발된 모델로, 복잡한 경계를 정확히 학습하고 분할하는 데 특화되어 있다. 이 모델은 복잡한 구조를 포착하면서도 해상도를 유지하는 학습 과정을 통해, Hand Bone Task에서의 Occlusion 문제를 해결하고, 뼈 구조 간의 경계를 더욱 뚜렷하게 분리할 수 있을 것으로 예상하여 Hand Bone Task의 학습 모델로 DUCKNet을 채택하였다.

✓ DUCK Block

1. 3x3 Residual Block

- 세부적인 패턴 탐지
2. 3x3, 7x7 Midscope Block
- 중간 규모 패턴 탐지
3. 15x15 Widescope Block
- 넓은 범위 패턴 탐지
4. 1xN, Nx1 Separated Block
- 가로, 세로 방향 패턴 탐지

위 4가지 Block이 병렬적으로 작동하여 다양한 크기와 방향의 특징을 동시에 학습하고 각 블록의 출력을 모두 더해 다양한 스케일과 방향 정보를 모두 포함하는 feature map이 생성된다.

✓ Residual Downsampling

기본적인 Downsampling 연산에 Residual Connection을 결합한 구조로, 다양한 해상도의 정보를 효율적으로 처리하면서 세부 정보를 유지하도록 설계.

✓ BASE vs DUCKNet

DUCKNet은 FCN-ResNet50 보다 약 2M개 파라미터가 적음에도 불구하고, **0.03%p** 더 높은 성능을 기록하였다.

Model (filter)	Parameter	Pretrained	DICE
FCN-ResNet50	35.3M	True	0.9249
DUCKNet (16)	33.7M	False	0.9583

✓ UNet

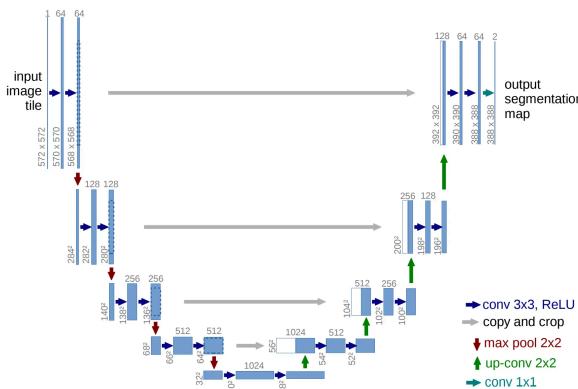


그림 12. UNet 아키텍처

UNet 모델은 세포 경계를 세밀하게 검출하고 구분하기 위해 고안된 아키텍처로, medical segmentation 작업에서 현재까지도 준수한 성능을 보여주고 있다. 이러한 특성 때문에, **Hand Bone Task**에서 발생하는 손등 뼈의 **occlusion** 문제를 해결하고, 각 뼈의 형태를 정확히 검출할 수 있을 것으로 기대하고 채택하였다.

✓ BASE vs UNet

FCN-ResNet50보다 Parameter수가 $\frac{1}{3}$ 적은 대로 불구하고 **0.03%p** 높은 성능을 보인다.

Model	Parameter	Pretrained	DICE
FCN-ResNet50	35.3M	True	0.9249
UNet	23.4M	False	0.9548

✓ Nested UNet

UNet이 준수한 성능을 보였기에, 이를 확장한 Nested UNet을 활용하여 성능을 더욱 향상시키고자 하였다. Nested UNet은 **deep supervision** 기법을 적용하여 segmentation task에서 더 세밀한 예측이 가능하며, 복잡한 구조의 데이터에서도 높은 성능을 기대할 수 있어 채택하였다.

✓ Deep Supervision

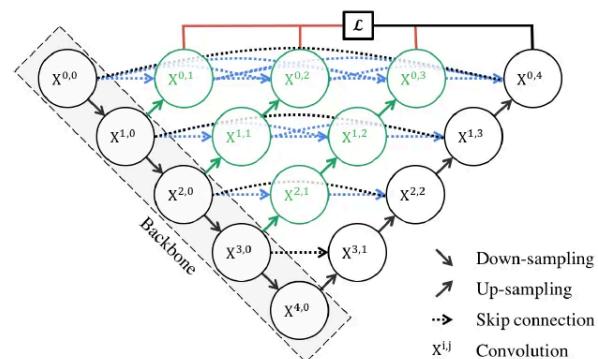


그림 13. Nested UNet 아키텍처

Deep Supervision은 모델의 중간 계층에 손실 함수를 추가하여 학습 신호를 제공한다. 이를 통해 기울기 소실 문제를 완화하고 학습의 안정성을 높이며, segmentation 성능을 향상시킨다 예상하여 채택하였다.

✓ BASE vs Nested UNet

Nested UNet에서 필터수를 [32, 64, 128, 256, 512]로 줄여 경량화 했음에도 FCN-ResNet50에 비해 약 4배 적은 파라미터 수를 가지면서도 DICE 점수에서 **0.03%p** 더 높은 성능을 기록했다.

하지만, Deep Supervision(DS)을 활성화했을 때는 DICE 점수가 **0.7582**로 감소하였다. 이는 학습 시 중간 계층에 분산되면서 출력 계층의 최적화가 방해받았거나, 모델이 중간 계층 출력에 과도하게 의존했기 때문으로 판단된다. DS는 복잡한 데이터셋에서는 긍정적인 영향을 미칠 수 있으나, 상대적으로 단순한 구조를 가진 **Hand Bone Dataset**에서는 부적합하다고 결론지었다.

Model	Parameter	Pretrained	DICE
-------	-----------	------------	------

FCN-ResNet50	35.3M	True	0.9249
Nested UNet	9.164M	False	0.9548
Netest UNet(DS)	9.167M	False	0.7582

UNet3Plus

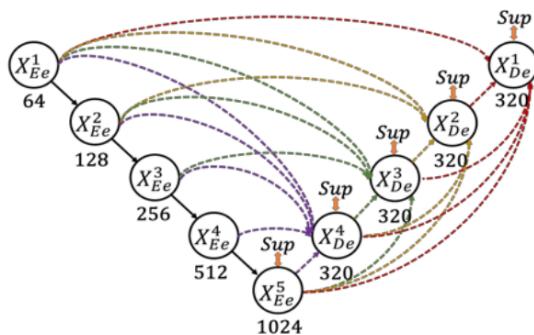


그림 14. UNet3Plus 아키텍처

기존 UNet과 Nested UNet의 한계를 보완하고자 UNet3Plus를 활용하였다. UNet3Plus는 segmentation task에서 더 정확한 경계와 디테일한 예측을 제공하기 위해 멀티 스케일 feature aggregation과 skip connection을 강화한 구조를 가지고 있어 채택하였다.

✓ BASE vs UNet3Plus

UNet3Plus는 약 27.57M의 파라미터를 가지고 있으면서도 Pretrained 모델을 사용하지 않은 상태에서 DICE 점수 0.9573을 기록하였다.

Model	Parameter	Pretrained	DICE
FCN-ResNet50	35.3M	True	0.9249
UNet3Plus	27.57M	False	0.9573

Custom UNet

Custom UNet은 UNet 구조를 기반으로 설계되었으며, 기존 UNet이 제공하는 feature map보다 더 정교하고 유의미한 정보를 추출할 수 있도록 개선하려 하였다. 이를 위해 Spatial Attention과 Channel Attention 기법을 도입하여 중요 영역과 유의미한 채널 간 상호작용을 강조하고자 하였다.

✓ UNet Feature Map

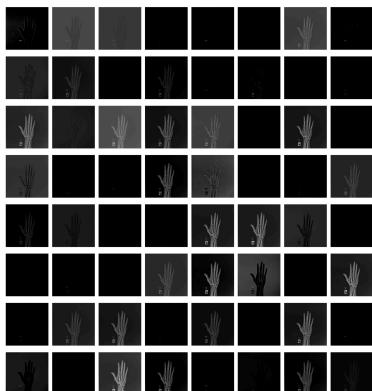


그림 15 : UNet Feature Map

기존의 UNet feature map은 skip connection을 통해 저해상도와 고해상도의 정보를 결합하지만, 공간적 중요도나 채널 간 상호작용을 고려하지 않아 복잡한 구조의 데이터를 처리할 때 세밀한 예측이 어렵다고 판단하였습니다. 또한, UNet의 feature map은 배경 정보로 인해 의미 없는 feature가 다수 포함될 가능성이 있어, 중요한 특징을 효과적으로 추출하는데 한계가 있다고 보았습니다.

✓ Spatial Attention & Channel Attention



그림 16 : CA Feature Map(좌), SA Feature Map(우)

Spatial Attention(SA)과 Channel Attention(CA)는 Encoder와 Decoder 사이의 **Skip Connection**에서 사용되어, 저해상도에서 고해상도로 정보를 전달할 때 공간적 중요도와 채널 간 상호작용을 강조하였습니다.

- Spatial Attention : 공간적 중요도를 강조하여 영역별 정보의 가중치를 조정
- Channel Attention : 채널 간 상호작용을 통해 유의미한 feature를 선택적으로 강화
- 두 기법을 결합하여 더 정교한 feature representation을 생성

✓ FusionBlock

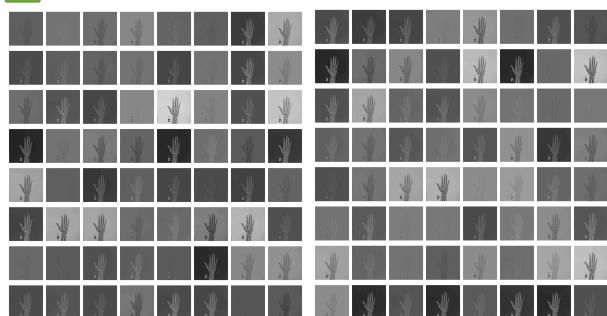


그림 17 : Cat Feature Map(좌), Sum Feature Map(우)

- Spatial Attention과 Channel Attention의 출력을 결합하기 위해 Sum과 Concat 방식 지원
- 모델의 표현력을 유지하면서 계산 효율성 증가

✓ Dropout & GELU

• 학습 진행 시 모델이 과적합되는 현상이 발생하여 중간 레이어와 디코딩 단계에서 Dropout을 적용하여 과적합을 방지하고자 하였다.

- BatchNorm2d 이후 활성화 함수로 GELU를 사용하여 ReLU 대비 계산 Cost가 늘어나지만 학습의 안정성을 높이고자 사용하였다.

Custom UNet vs UNet 기반 모델들

1024x1024 크기의 이미지를 대상으로, 모든 모델에 동일한 학습 조건을 설정한 후 총 30 epoch 동안 비교 평가를 진행하였다.

Model	Parameter	Pretrained	DICE
UNet	23.38M	False	0.9548
Nested UNet	9.164M	False	0.9544
UNet3Plus	27.572M	False	0.9573
DUCKNet	33.7M	False	0.9673
Custom UNet	19.09M	False	0.9706

Hand Bone Dataset에서 Dice 점수 **0.9706**을 기록하며 5개의 모델 중 가장 높은 성능을 보였습니다. 필터 수를 줄인 Nested UNet보다는 높은 Parameter를 가지고 있지만, 다른 UNet 기반 모델들보다는 적은 Parameter로 더 우수한 성능을 달성하였다.

특히, Occlusion 문제가 있던 클래스 2개 쌍 중 1개 쌍에서는 약 **0.03%p**의 성능 개선을 이루었지만, 다른 1개 쌍에서는 성능 개선이 나타나지 않았습니다.

Custom UNet의 한계

1. 연산 비용 증가

- Skip Connection에서 SA Block과 CA Block의 출력을 결합할 때 Sum 방식은 연산 효율이 높으나, Concat을 하였을 때는 계산 비용이 늘어나 고해상도의 데이터를 학습 시 학습 속도가 느리고 GPU 메모리가 부족할 수 있다.

2. Skip Connection 정보 활용의 제한성

- Encoder와 Decoder간의 정보를 연결할 때 단순 Sum 또는 Concat 방식으로 결합함에 따라, 복잡한 데이터 분포를 충분히 반영하지 못할 가능성이 있다.

3. 데이터 의존적 성능 변화

- Hand Bone과 같은 단순한 Dataset에서는 높은 성능을 보였지만, 다른 도메인 데이터에서는 동일한 구조가 최적의 결과가 아닐 수 있다.

4. Overfitting

- 상대적으로 작은 데이터셋에서는 Attention 기법이 다양한 Feature를 학습하면서 모델이 과적합이 되어 일반화 성능이 낮아질 거라 예상된다.

3.6 Ensemble

Test Score와 모델의 다양성을 기준 Ensemble

Dice 점수와 모델의 다양성을 기준으로 상위 모델들을 선택하여 앙상블을 구성하였으며, 다양한 특징을 가진 모델들의 결합을 통해 성능을 극대화하고, 일반화 성능을 향상시키는 것을 목표로 하였다.

지금까지 학습한 모델들 중 상위의 점수를 기록한 모델들을 hard voting 하는 방식으로 여러 가지 섞어보면서 앙상블 진행

Method	Dice
7 model vote 2	0.9710
7 model vote 3	0.9715
7 model vote 4	0.9709
10 model vote 3	0.9710
10 model vote 4	0.9714
10 model vote 5	0.9713
12 model vote 4	0.9714
12 model vote 5	0.9716
15 model vote 6	0.9718

7 models : FCN-Resnet50, DUCK-Net, U-Net, SegFormer, FCN-Resnet50 Inter area, DUCKNet Inter area, FCN-Resnet50 K-Fold Ensemble

10 models : 7 models + DUCK-Net Loss, FCN-Resnet50 Elastic Norm

12 models : 10 models + Custom U-Net + FCN-Resnet50 bce dice iou

15 models : 12models + DUCK-Net K-Fold Ensemble + Custom U-Net threshold 0.5, Custom U-Net threshold 0.7

4. Final Score

최종결과

- Public Score

- 20 / 23**
- DICE : **0.9718**

- Private Score

- 21 / 23**
- DICE : **0.9729**

Public Score보다 Private Score에서 점수는 상승하였지만 등수는 하락하였다.

최종순위 **21위** 기록

5. Cooperation

A. 협업 방법

- Google Spreadsheet을 통한 프로젝트 진행 과정 및 실험 결과를 공유

- ii. Git과 Github를 활용하여 개인이 작성한 코드를 공유하고 서로 리뷰
- iii. Zoom을 활용하여 실시간으로 소통하면서 협업진행
- iv. Slack, Kakaotalk, SpreadSheet를 통해서 서버 사용현황 실시간 공유
- v. Notion을 통해 프로젝트 일정 관리

B. 개발 환경 및 Tool

- i. 개발언어 : Python
- ii. 개발환경 : V100 32GB GPU
- iii. Frame Work : PyTorch
- iv. 협업툴 : MLFlow, Slack, Notion, Git, SpreadSheet

6. 팀 회고 및 개선방안

잘했던 점

- 협업 툴을 적극적으로 활용해 효율적으로 프로젝트를 진행할 수 있었다.
- 개개인의 성장을 위해 최종 순위가 아닌 개개인의 목표를 설정했다.
- 사전학습된 모델만 사용하지 않고, 직접 모델을 수정해보면서 layer간의 인과관계를 분석해보며 새로운 모델을 만들어 봤다.
- 단순 SOTA 모델을 가져오거나 라이브러리를 사용하지 않고 직접 구현하며 Segmentation Task의 이해도를 높였다.

아쉬웠던 점

- 연이은 프로젝트로 인해 팀원들이 번아웃을 경험한 점이 아쉬웠다. 프로젝트 일정 조율을 통해 팀원들의 피로도를 관리하는 방안이 필요했다.
- 의료 도메인이라는 특수성을 더욱 잘 활용하기 위해, 엑스레이 데이터나 손 뼈 구조에 대한 기초적인 학습을 병행했더라면 모델의 성능과 해석력을 높일 수 있었을 거라 생각한다.
- UNet과 같은 CNN 아키텍처에서 Encoder 단에 Backbone 모델을 활용해 Feature를 추출하거나, Decoder 단에도 Backbone의 구조적 이점을 활용하지 못한 점이 아쉬움으로 남는다.

7. 개인 회고

천유동

- 프로젝트 학습 목표

EDA를 통해 증강을 적용하고 최대한 gpt사용을 지양하면서 코딩 실력을 늘리면서 진행하려고 하였다.

- 내 학습 목표를 달성하기 위해 한일.

EDA를 바탕으로 증강을 진행하였다. 효율적인 연산을 위한 정규화 증강, 밝기와 대비를 증강하는 random brightness와 CLAHE, 손의 기울기 분포를 파악하여 진행한 rotate16 등을 진행하였다.

- 나는 어떤 방식으로 모델을 개선했는가?

초기에는 속도가 너무 느려서 mixed precision 적용하여 학습 속도 개선과 모델의 증강 적용을 확인하기 위해 visualize를 적용하여 증강된 이미지를 확인하였다.

- 학습 목표를 달성하기 위한 과정 중 깨달은 점

EDA 때 내린 가설로 항상 개선이 된다면 좋겠지만 꼭 그렇지는 않았다. 오히려 EDA와 상반되는 결과도 있었고 그 때마다 결과 분석을 하면서 실력이 늘어난 것 같다.

또한 데이터 증강을 적용할 때 많이 적용할 수록 오히려 성능이 떨어지는 결과도 확인할 수 있었다. 너무 많은 데이터 증강은 오히려 test set과의 괴리감을 더 발생시켜서 적합하지 못한 데이터를 만들어냈기 때문이라고 생각하였다.

- 아쉬웠던 점

다양한 데이터 증강을 적용하였지만, 새로운 증강을 구현해서 적용하지 못한 점이 아쉬웠다. 또한 주로 데이터 증강 작업을 하였는데, 직접 증강 파라미터들을 조절하며 진행하다보니 시간이 없어서 다른 작업을 하지 못한 점이 살짝 아쉽다. 또한 데이터가 의료 데이터고 아주 정적인 데이터이다 보니 증강이 많이 중요하지는 않았다.

- 총평

마지막 프로젝트를 통해 gpt사용도 지양하며 직접 코딩을 하는데 의의가 있었다. 다양한 증강 작업을 통해 어느 증강이 어떤 데이터에 사용하면 유용할지 알게 되었다. 추후에 있을 마지막 프로젝트에서 이렇게 배운 점들을 활용할 것이다.

임용섭

- 프로젝트 학습 목표.
- 모델 구현 및 실험 관리

- 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

SegFormer 구현, MLflow 사용을 통해서 모델 구현과 실험 관리를 하였습니다.

- 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

아무래도 처음 모델을 구현하는 것 이여서 그런지 논문만 읽고 구현하는 것은 어려웠고, MLflow도 Local Host를 이용하다보니 터널링 프로그램을 새로 찾아야했습니다.

- 나는 어떤 방식으로 모델을 개선했는가?

모델의 추론 결과를 시각화하여 인사이트 공유 및 Transformer 기반의 모델을 사용하여 기존 모델의 한계를 극복하고자 하였다.

- 내가 해본 시도 중 어떠한 실패를 경험했는가?

실패의 과정에서 어떠한 교훈을 얻었는가?

모델이 어려워하는 부분을 찾아도 그 부분을 어떻게 개선해야 할지 모르겠다는 점이 큰 실패였고, 다양한 가설들을 세워서 이런 부분을 극복해야겠다는 교훈을 얻었다.

- 협업 과정에서 잘된 점/ 아쉬웠던 점은 어떤 점이 있는가?

좋았던 점

GitHub 사용을 이번 프로젝트에서 본격적으로 진행하게 되었는데 늦었지만 다양한 이슈 관리 등을 사용하다 보니 재미가 있었고 다음에는 다양한 템플릿을 통해서 GitHub를 좀 더 잘 사용해보고 싶습니다.

모델 구현을 처음으로 시도했는데 아무래도 논문을 보고 구현하다보니까 어려웠고 그 어려운 걸 해내서 좋았습니다. 그리고 프로젝트에서 이런 시간을 가져서 개인적인 성장을 챙길 수 있어서 좋았습니다.

아쉬운 점

많은 가설을 시도하지 못한 점

- 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

EDA의 결과를 토대로 데이터를 관리한 뒤 프로젝트 진행을 하면 좋은 결과를 얻을 수 있을 것 같다는 생각이 들었다. 가설을 세우는 이유가 명확해야한다는 생각이 들었다.

💡 정지훈

- 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

이전에 여러 프로젝트를 수행해본 경험을 바탕으로 이번에는 주도적으로 문제를 탐구하고 해결책을 도출하고자 했다.

데이터를 제대로 이해하지 못하면 적절한 접근법을 설계하기 어렵기 때문에, 모든 데이터를 눈으로 확인하며 전반적인 경향성과 이미지의 특징을 분석했다. 데이터의 분포와 특성을 직접 살펴보는 과정을 통해 보다 직관적으로 문제를 이해하고 해결 방안을 구상할 수 있었다.

제공된 베이스라인 코드를 테스트해보고 그 결과를 기반으로 추가적인 개선점을 직접 만들어보자 다.

라이브러리 활용 경험을 쌓기 위해, 주어진 데이터셋을 MMSeq 프레임워크에서 활용할 수 있도록 데이터 포맷을 변환하고 적용하는데 도전했다. 기존에 사용하던 MMDet과는 다른 구조를 요구했기에, 데이터 전처리와

포맷 변경 등 새로운 시도를 통해 데이터를 MMSeq에 적합하게 가공했다. 이를 통해 프로젝트의 범위를 확장하고 라이브러리에 대한 이해도를 높이고자 했다.

- 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

이론적인 학습을 통해 개념과 원리는 이해하고 있었지만, 이를 실제로 코드로 구현하는 데에는 여전히 부족함이 있었다. 특히, PyTorch와 같은 프레임워크의 문법과 세부적인 클래스 설계 등에 대해 익숙하지 않아 구현 속도가 느렸고, 복잡한 구조를 다루는데 어려움을 겪었다.

GPT와 같은 도구의 도움을 받아 문제를 해결해왔지만, 데이터와 작업의 복잡도가 높아지면서 GPT를 활용하는데에도 한계를 느꼈다. 특히, 고도화된 작업일수록 직접 구현하고 수정할 수 있는 능력이 중요하다는 점을 절실히 깨달았다. 이로 인해, 장기적으로 나 자신의 역량을 키워야겠다는 필요성을 강하게 느꼈다.

MMSeq을 처음 다루면서, 기존에 MMDet에서 사용하던 데이터셋과 다른 요구사항 때문에 많은 부분에서 실패를 경험했다. 특히, MMSeq에 적합한 베이스라인 코드가 없었던 상황에서 초기 구현에 실패했고, 이후 제공된 코드를 활용하여 Segformer를 구현하는 데는 성공했지만, 다른 모델로 확장하는 데 어려움을 겪었다.

- 내가 해본 시도 중 어떠한 실패를 경험했는가?

실패의 과정에서 어떠한 교훈을 얻었는가?

MMSeq을 활용하기 위해 데이터를 변환했으나, 데이터 포맷과 요구사항의 차이로 인해 실패를 경험했다. 이후 오피스아워에서 제공된 코드를 활용해 구현에는 성공했지만, 코드를 깊이 이해하지 못한 상태에서 Segformer 외의 모델로 확장하는 데 실패했다.

이를 통해 프레임워크와 데이터의 구조를 충분히 이해하지 않고 무작정 시도하기보다는, 사전 학습과 계획적인 접근이 필요하다는 점을 배웠다.

PyTorch와 같은 프레임워크의 문법과 클래스 설계를 깊이 이해하지 못해 새로운 작업을 확장하는 데 제약이 있었다. 이는 단순한 실행이 아니라, 프레임워크의 기본 원리를 이해하고 자유롭게 활용할 수 있는 능력을 길러야 함을 깨닫게 했다.

- 협업 과정에서 잘된 점/ 아쉬웠던 점은 어떤 점이 있는가?

협업을 위해서 충분히 협업 툴을 사용하고 서버에 여러 api들을 연결해서 현황을 간접화하도록 만들어 기존에 느꼈던 불편함을 해소했고 팀원과의 소통을 통해서 서로간의 진행 상황을 공유하면서 자유롭게 아이디어를 나눠 더 좋은 결과를 낼 수 있었던 것 같다.

나는 베이스 코드와 약간 독립적으로 작업하다 보니 팀원들의 진행 속도에 맞추지 못한 부분이 있었다. 이는 팀과의 긴밀한 연계성을 유지하는 데 어려움을 초래했다.

- 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트 전에 충분한 학습과 연습을 통해 코드 구현 능력을 키워, 프로젝트 진행 중 최대한 내 힘으로 문제를 해결하고자 한다. 특히 PyTorch와 같은 프레임워크의 심화 학습을 통해 자유롭게 활용할 수 있는 능력을 갖추고 싶다.

다음 프로젝트는 기업 해커톤으로, 최종 발표를 통해 결과물을 검증받아야 하는 만큼 더욱 완성도 높은 결과를 내고자 한다. 이를 위해 초기 설계 단계부터 철저한 계획과 준비를 진행하고자 한다.

팀원들과의 소통과 협업을 강화하여 팀의 목표에 긴밀히 부합하는 작업을 수행하고, 개인 작업과 팀 작업을 균형 있게 수행할 수 있도록 노력할 것이다.

💡 이상진

- 프로젝트 학습 목표.

- OpenAI API를 이용한 자동 EDA 코드를 작성하여 데이터를 더 빠르고 효율적으로 분석하고, 학습 과정에서 잠재적인 문제점이나 최적의 학습 파라미터를 추천받아 학습 효율성을 높이고자 함.

- Segmentation 모델을 직접 구현하여 Segmentation 작업의 코드 구조와 진행 과정을 깊이 이해하고, 이를 실무에 적용할 수 있는 역량을 키우고자 함.

- Custom UNet을 설계하고 구현함으로써 Hand Bone Task에서 중요한 요소들을 파악하고, 문제 해결에 필요한 모델링 역량을 강화하고자 함.

- 개인적 감상

프로젝트를 마무리할 때마다 아쉬움이 남았지만, Custom UNet 개발을 통해 Segmentation 작업에서 발생하는 Occlusion 문제에 대한 해결 가능성을 탐구하며 의미 있는 성과를 얻을 수 있었습니다. OpenAI를 이용한 자동 EDA는 비용 문제로 인해 폐기했지만, 이를 대체할 API 활용 가능성을 고민하며 데이터 분석 자동화에 대한 이해를 더욱 깊이 할 수 있었습니다. 또한, Transformer 기반 모델을 구현하려 했으나 시간 부족으로 최종적으로 완성하지는 못했습니다. 하지만 이 과정에서 Transformer 구조와 Attention 메커니즘이 가진 가능성을 확인하며, 새로운 방법을 고민할 수 있는 계기가 되었습니다.

- 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- DUCKNet 등 여러 SOTA 아키텍처가 Keras 기반으로 제작되었기 때문에, 이를 PyTorch로 구현하기 위해 처음에는 두 프레임워크 간의 레이어 구조 차이를 이해하는 데 어려움을 겪었습니다. 하지만 Keras와 PyTorch 문서를 비교 분석하고, 다양한 예제를 참고하며 단계적으로 코드를 변환할 수 있었습니다.

- Custom UNet을 설계하는 과정에서 기존 UNet보다 적은 파라미터로 더 효율적인 Feature Map을 생성하기 위해 CBAM(Convolutional Block Attention Module)의 Spatial Attention과 Channel Attention을 활용했다.

Spatial Attention은 Feature Map의 공간적 정보를 강조하기 위해 Average Pooling과 Max Pooling을 결합하여 중요한 영역을 강조하고, Channel Attention은 각 채널의 중요도를 학습하도록 설계하여 Feature Map에서 global 특징과 각 채널 간 상관관계를 효율적으로 추출할 수 있도록 구현했습니다. 이를 통해 불필요한 파라미터를 줄이면서도 순차 빼 데이터를 처리할 때 발생하는 Occlusion 문제를 해결하기 위한 모델의 표현력을 개선하고자 했습니다.

- 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- CNN 기반이 아닌 Transformer가 포함된 모델을 직접 구현할 때, Transformer의 Attention 메커니즘이 Segmentation 작업에서 지역적 정보보다 전역적 정보를 더 잘 포착할 수 있도록 설계되어 있음을 이해했지만, 이를 효과적으로 적용하는 과정에서 모델의 복잡도가 증가하고 학습 시간이 길어지는 한계를 마주했습니다. 특히, Hand Bone 데이터에서 발생하는 Occlusion 문제를 해결하기 위해 Transformer 기반 구조를 활용하려 했으나, 데이터

크기와 모델 성능 간의 균형을 맞추는데 어려움을 겪었고, 결국 목표한 성능을 달성하지 못하는 결과로 이어졌습니다.

- Keras의 padding='same' 파라미터가 입력과 출력의 크기를 동일하게 유지하기 위해 자동으로 패딩을 추가해주지만, 이를 PyTorch에서 구현하려 할 때 커널 크기와 입력 크기에 따라 어떻게 패딩을 적용해야 동일한 결과를 얻을 수 있을지에 대한 문제점이 있었습니다.

- 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

- 기존 CBAM은 Spatial Attention(SA)을 통과한 후 Channel Attention(CA)을 적용하여 Feature Map을 생성하는 방식으로 설계되었습니다. 그러나 Custom UNet에서는 SA와 CA의 출력을 각각 sum하거나 concat하여 skip connection에 전달하는 방식으로 활용되었습니다. 이로 인해 기존 CBAM 방식과는 구조적 차이가 있었으며, 이러한 변경이 모델 성능에 미치는 영향을 충분히 분석하지 못한 점이 아쉬움으로 남습니다.

CBAM 논문에서는 SA 블록 이후 CA를 적용하여 글로벌 Feature Map을 강화하는 것이 성능 향상의 핵심이라고 강조하고 있어, 이후 개발 단계에서는 CBAM의 원래 방식을 충실히 따르는 Custom UNet을 구현하고, 이를 통해 Segmentation 작업에서 Attention 순서와 조합이 성능에 미치는 영향을 보다 체계적으로 테스트해보고자 합니다.

- UNet기반 모델에 Encoder 부분에 다른 Backbone을 적용해 테스트해보지 못한 점도 아쉬움으로 남아, 이후 구현에서는 Backbone 네트워크를 Encoder단에 적용해 실험을 해보고 싶습니다.

- Keras와 PyTorch를 합친 하이브리드 프로젝트 또한 더 직관적인 코드와 다양한 프레임워크의 장점을 활용할 수 있을 것으로 기대되어, 앞으로 이러한 방식을 적용해 프로젝트를 구현하고 테스트해보고 싶습니다.

- 협업 과정에서 잘된 점/ 아쉬웠던 점은 어떤 점이 있는가?

- 팀원들과의 원활한 소통과 자유로운 아이디어 교류를 통해 다양한 아이디어를 실험하며 프로젝트의 완성도를 높일 수 있었습니다. 특히, UNet의 Feature Map과 Custom UNet의 Feature Map을 추출해 비교 분석하고, 두 모델이 생성하는 특징의 품질과 차이를 팀원들과 함께 논의하며, 어떤 모델이 더 좋은 Feature Map을 제공하는지 판단한 과정은 매우 좋았습니다. 이러한 논의를 통해 Custom UNet의 설계를 개선하고, Segmentation 작업에서 더 나은 성능을 낼 수 있는 방향으로 모델을 최적화할 수 있었습니다.

또한, 실험 결과와 문제점을 투명하게 공유하고 피드백을 주고받는 과정에서 많은 배움을 얻을 수 있었으며, 팀원들의 다양한 관점이 프로젝트의 완성도를 높이는 데 크게 기여했습니다.

- 협업 과정에서 일정 조율과 역할 분배가 다소 부족해 일부 작업이 지연되거나 효율성이 떨어진 부분이 있었습니다. 또한, 학습 자원의 부족으로 인해 모든 팀원이 학습 과정에 고르게 참여하지 못한 점도 아쉬움으로 남습니다.

😊 박재우

- 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

EDA를 통해 데이터를 분석한 결과는 다음과 같습니다. 이전 프로젝트에서 클래스 불균형과 이상치 데이터 문제가 자주 발생했던 점을 고려하여, 이번에도 클래스 분포와 각 챠 수 분포를 우선적으로 확인했습니다. 분석 결과, 29개의 클래스가 모두 균형 있게 분포하고 있음을 확인할 수 있었습니다.

다음으로, 클래스의 위치 분포를 왼손과 오른손으로 나누어 살펴본 결과, 손을 꺾어서 촬영한 이미지들이 포함되어 있다는 점을 확인했습니다. 이러한 특이한 자세는 모델 학습에 다양성을 부여할 가능성이 있어 주목해야겠다고 생각했습니다.

마지막으로, 전체 데이터의 수가 800장으로 비교적 적어, 데이터를 개별적으로 검토했습니다. 이 과정에서 반지를 끈 손이나 네일아트를 한 손과 같은 몇 가지 이상치 데이터도 발견되었습니다. 이러한 데이터들을 추후 전처리 단계에서 적절히 처리해야 할 필요가 있다고 생각했습니다.

- 전과 비교해서 내가 새롭게 시도한 변화는 무엇인가?

이전 프로젝트에서는 기준에 구현된 모델들을 활용하여 학습 및 평가를 수행했지만, 이번에는 직접 Segmentation 모델을 설계하고 구현하여 작동시켜보는 데 중점을 두었습니다. 이를 통해 단순히 모델을 활용하는 것에 그치지 않고, 모델의 구조와 동작 원리에 대한 더 깊은 이해를 얻고자 했습니다.

- 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

논문을 참고하여 직접 모델을 구현하는 과정은 예상했던 것보다 훨씬 큰 어려움으로 다가왔습니다. 관련 지식이 아직 충분하지 않다는 점을 실감했으며, 코드 구현 능력 또한 아직 부족하다는 것을 깨닫게 되었습니다. 결국, 어려움을 극복하기 위해 이번에도 ChatGPT의 도움을 받을 수밖에 없었습니다. ChatGPT를 통해 논문의 내용을 이해하고 구현하는 데 필요한 방향성을 얻었으며, 코드 작성과 디버깅 과정에서도 많은 도움을 받았습니다.

또한 기존에 구현된 모델을 활용했을 때 성능이 압도적으로 우수했던 경험 때문에, 이번 프로젝트에서는 구체적으로 어떤 작업을 수행해야 할지 목표를 설정하는데 어려움을 느꼈습니다. EDA를 통해 데이터의 특성을 어느 정도 파악했지만, 내가 수행하는 작업이 오히려 모델의 성능을 저하시킬지, 아니면 성능에 의미 있는 영향을 미칠지에 대한 확신이 부족했습니다.

이러한 불확실성은 프로젝트를 진행하는 원동력을 일부 상실하게 만들기도 했습니다. 내가 하는 시도가 긍정적인 결과를 가져올지, 아니면 단순히 시간과 노력을 낭비하는 결과로 끝날지 모른다는 점이 부담으로 작용했습니다. 이로 인해 프로젝트를 보다 적극적으로 추진하기가 어려웠던 부분이 있었습니다.

- 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트는 기업 해커톤인 만큼, 원동력이 떨어질 걱정은 없을 것이라 생각합니다. 이번 경험을 바탕으로, 프로젝트를 시작하기 전에 확실한 목표를 설정하고, 이를 중심으로 계획을 세우는 것이 중요하다는 점을 깨달았습니다.

특히, 다음번에는 코드 구현 과정에서 최대한 스스로 해결하려고 노력할 것입니다. 물론 어려움이 있더라도 이 어려움을 통하여 성장할 기회를 얻고, 구현 능력을 향상시킬 수 있을 것이라 믿습니다

💡 유희석

- 내 개인적인 학습 목표

1. Git을 포함한 협업 툴을 활발히 활용
2. 실험 설계와 결과를 꼼꼼히 기록해 분석에 활용하기

- 내 학습 목표를 달성하기 위해 한일

1. EDA & Annotation

익숙하지 않은 의료데이터라는 점과 데이터의 수가 그리 많지 않았기 때문에 모든 이미지와 annotation 데이터를 눈으로 확인했습니다. 그 과정에서 클래스가 이상하게 찍혀있거나, 잘못된 마스킹이 된 이상치 이미지들을 찾을 수 있었고, 이를 수정해 모델의 성능 상승에 기여했습니다.

2. 코드 리팩토링

이번 프로젝트에서 지난 프로젝트와 달리 베이스라인 코드를 수정해서는 안되는 등의 제약이 없었기 때문에 조원들이 편하게 실험을 진행하기 위해서 베이스라인 코드를 모듈화해서 추후 쉽게 기능을 추가할 수 있게 해 편하게 다양한 실험을 할 수 있도록 만들었습니다. 또한 정확한 실험을 위해 변인통제가 중요한데 이런 파라미터들을 config.yaml 파일로 간단하게 조정할 수 있도록 만들었습니다.

3. KFold

데이터 수가 많지 않았고, 추가 데이터셋을 사용할 수 없다는 점으로 인해 주어진 학습 데이터셋을 온전히 사용하기 위해서 KFold 기법을 제안했고, 각 fold별 validation dice score와 public dice score를 비교해 가장 비슷한 분포를 가진 fold를 찾아 다른 조원들은 그 fold를 이용해 실험 결과를 확인하고, 추후에 제출을 할 때에는 모든 fold를 학습시켜 양상불을 진행할 수 있게 했습니다.

- 학습 목표를 달성하기 위한 과정 중 깨달은 점

특수 도메인에서의 task를 진행할 때에 그 특수성이 중요하다는 것을 머리로는 알고 있었지만, 직접 실험해보고 데이터를 만져보면서 왜 중요한지 알 수 있었습니다. annotation format부터 기준에 알고 있는 annotation format이 아니었기 때문에 annotation tool을 활용하기 위해서라도 범용적인 format으로 다시 변환하는 과정을 겪어야 했습니다. 또한 x-ray 데이터라는 특수성으로 인해 범용적인 augmentation 기법에서 오히려 성능이 하락하는 경우를 겪기도 했습니다.

- 아쉬웠던 점

1. 모델간 성능 차이

모델들을 탐색해 보거나 모델에서 조금씩 수정해보면서 모델링을 진행해보았지만, 베이스라인 모델에서 loss에

DICE score를 합쳐 얻은 성능과 크게 차이가 나지 않았습니다. 모델링에서 많은 실험과 노력을 투자했지만 큰 성능 상승을 보이지 못한 것 같아서 아쉽다.

2. 도메인 특성 활용

개인적으로 의료 데이터라는 점과 **x-ray**로 인해 단일 채널 이미지 데이터라는 점을 적극적으로 활용해보지 못한 것이 아닌가라는 생각이 들었습니다. 뒤돌아서 생각해보면 손 뼈 구조나 **x-ray** 이미지 데이터의 숨겨진 특성이나 특징을 알아보려고 하지 않은 것 같았습니다. 강의에서 이런 특성이 해결법을 찾는 과정에서 매우 중요하다는 것을 강조했기에 알고는 있었지만 베이스라인 모델에서 준수한 성능을 보였기 때문에 따로 찾아보려고 하지 않게 된 것 같아 아쉽습니다.

- 다음 프로젝트에서 해보고 싶은 것들

1. 다음 프로젝트에서는 데이터셋 또한 직접 구해야 하고, 그 방법들이 다양한데 그런 것들을 직접 경험해보고 싶다.
2. 내가 맡아 진행하는 것들을 개발일지처럼 왜 이런 판단을 했고, 왜 이런식으로 진행했는지 더 상세하게 기록하면서 프로젝트를 진행하기