Semantic Segmentation Wrap-up Report

CV-13조 (SMiLE)

김영일, 안세희, 유한준, 윤일호, 이재혁

1. 프로젝트 개요

가. 프로젝트 주제

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

Bone Segmentation은 인공지능 분야에서 중요한 응용 분야 중 하나로, 특히 딥러닝 기술을 이용한 뼈 Segmentation은 많은 연구가 이뤄지고 있으며, 많은 목적으로 도움을 줄 수 있음

따라서, X-ray 이미지에서 사람의 손 뼈마디를 정확하게 Segmentation하는 모델을 개발함으로써 질병 진단 및 의료 목적으로 활용 가능성을 높이는 것을 목표로 함

나. 활용 장비 및 재료

(1) 서버

- CPU: Intel® Xeon® Gold 5120
- GPU: V100 32GB 1EA

(2) 의사소통

- 데일리 스크럼 및 피어세션 시간에 Zoom을 활용하여 실험 진행 방향 공유
- Slack기반 실험 결과 공유

(3) 협업

- Git를 활용하여 협업 진행
- Team Wandb 기반 실험 진행상황 공유

2. 프로젝트 팀 구성 및 역할

가. 팀 소개

저희 팀의 이름은 'SMiLE'로, 어떤 일을 하게 되든 **항상 웃으면서** 포기하는 다같이 열심히 달려보자는 포부를 가지고 있음

나. 역할

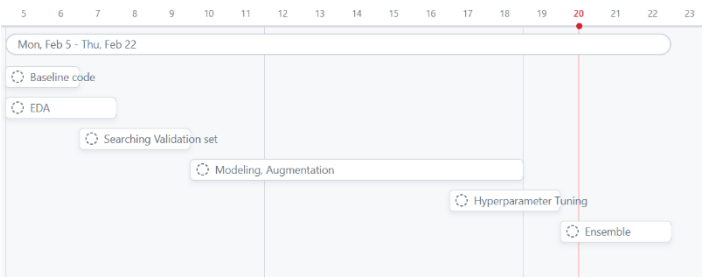
이름	역할
김영일 (팀장)	데이터 시각화 공유, Psuedo labeling 실험, DeepLabV3+ 기반 모델 실험, Instance Segmentation 실험
안세희 (팀원)	Smp 모델 실험 코드 구현, Encoder/Augmentation 실험
유한준 (팀원)	MMSegmentation 실험환경 구축, Optimizer/Scheduler 실험환경 구축, Encoder/Augmentation 실험
윤일호 (팀원)	Loss 실험환경 구축 및 실험, Encoder/Optimizer/Augmentation 실험
이재혁 (팀원)	협업 환경 구축, Encoder/Augmentation 실험

3. 프로젝트 수행 절차 및 방법

가. 프로젝트 공동 목표

- 1주차: EDA 및 적절한 Validation 탐색
- 2주차: Loss/Optimizer/Scheduler 실험
  Augmentation 실험
- 3주차: Parameter Tuning 및 Ensemble

나. 타임라인



다. 협업 Rule

- 1 Issue, 1 branch, 1 PR을 기준으로 진행
- default branch를 develop으로 생각하여 작업을 진행하며, feature branch를 생성하고 작업이 완료되면 develop에 PR 진행
- 진행 해볼만한 실험이나 진행할 실험 내용은 노션 및 깃허브 이슈로 공유
- 데일리 스크럼시 당일 할 작업에 대해 공유하고, 진행 상황을 피어세션 때 공유

4. 프로젝트 수행 절차 및 방법

가. EDA (Exploratory Data Analysis)

- 예측해야 할 클래스가 총 29개이며, 각 이미지마다 29개의 클래스가 모두 존재한다는 특징을 확인함으로써 제공받은 데이터셋이 클래스 균형임을 확인함
- 3번 클래스의 경우 손가락 옆 작은 뼈가 붙어있을 경우 레이블링이 되어있으며, 떨어져 있을 경우 레이블링이 안 되어있는 것을 확인함으로써 해당 부분이 결과에 영향을 미칠 수 있다고 판단함
- 손바닥 뼈 부분의 경우 멀티 레이블로 존재하는 픽셀이 존재하는 것을 파악했으며, 해당 부분을 잘 해결하여 segmentation 하는 것이 본 대회의 핵심일 것이라는 인사이트를 얻게 됨

나. 1차 모델 실험

Model (backbone)	**Validation Dice**	**Leaderboard Dice**
FCN (torchvision)	0.9474	0.9450
DeeplabV3+ (xception71)	0.9459	0.9458
UNet++ (xception)	0.9547	0.9512
UNet++ (effinet-b5)	0.9535	0.9506
UNet++ (resnext101)	0.9546	0.9521
UNet++ (hrnet_64)	0.9548	**0.9550**
UNet++ (max-vit)	0.9550	0.9510
UNet++ (tf-effinetv2-xl	**0.9551**	0.9535
UNet++ (halonet50)	0.9550	0.9538
UNet++ (nfnet_l2)	0.9550	0.9537
Upernet (mmseg)	0.0720	N/A
Segformer (mmseg)	0.0652	N/A

- 베이스라인 모델인 FCN 대비 성능 개선 실험을 진행하기 위해 segmentation models pytorch library 활용
- DeepLabV3+이나 FCN에 비해 UNet++ 모델이 성능이 가장 좋았으며, 이후 UNet++을 기준으로 추가 Encoder 실험 진행
- Encoder의 경우 2021년 이후 출간된 논문 중 인용 수가 많았던 모델을 선정하여 실험
- Encoder로 hrnet_64를 활용했을 때 가장 우수한 성능을 보였으며, 해당 모델을 바탕으로 추후 실험 계획 수립

다. 2차 모델실험

Encoder (UNet++, Lion, bce-dice)	**Validation Dice**
Baseline	0.9527
ResNet	0.9532
ResNeXt	0.9540
ResNeSt	0.9550
Res2Ne(X)t	0.9497
RegNet	0.9545
GERNet	0.9508
EfficientNet	0.9552

MobileNet	0.9516
VGGNet_19	**0.9558**
MobileOne	0.9545

- 1차 모델 실험을 바탕으로 선정한 Unet++ 모델에 optimizer와 loss를 각각 Lion, bce-dice로 고정하고 추가 Encoder 실험 진행
- Validation-Dice 점수가 가장 높은 Encoder인 VGG를 활용한 추가 실험 결정

라. 개선 기법

(1) Loss 실험

Loss (FCN, Adam)	**Validation Dice**	**Leaderboard Dice**
BCE	0.9321	0.9273
BCE-DICE	**0.9472**	**0.9455**
BCE-IOU	0.9377	0.9346
BCE-FOCAL	0.9343	0.9311

- 기존 BCE loss를 사용하는 Baseline code를 확장하여 실험을 위해 Dice/IoU/Focal loss를 결합하는 Combined loss 실험을 진행하였으며, 각 loss의 비율을 동일하게 진행
- Dice의 경우 평가 metric이 Dice score이기 때문에 선택하였으며, IoU의 경우 Dice와 비슷한 metric을 가지고 있어 선택
- 예상했던 가설과 동일하게 Dice와 결합하여 진행했던 loss가 가장 좋은 성능을 보였으며, Focal과 결합한 실험은 데이터셋이 클래스 균형하기 때문에 성능 개선을 얻지 못한다는 결과를 얻음

(2) Optimizer 실험

Optimizer (Unet++)	**Validation Dice**	**Leaderboard Dice**
Adam	0.9484	0.9450
AdamW	0.9480	0.9451
Lion	**0.9527**	**0.9508**
RMSprop	0.9496	0.9462

- 대부분의 딥러닝 모델 실험에서 잘 수렴한다고 알려진 Adam과, weight decay를 추가하여 일반화 성능을 높인 AdamW, 그리고 노이즈에 강하고 메모리 효율적이라고 알려진 Lion을 바탕으로 실험을 진행하였으며, Lion optimizer가 가장 좋은 성능을 보임

(3) Scheduler 실험

Scheduler (Unet++, Lion)	**Validation Dice**
No Scheduler	0.9558
StepLR	0.7351
ReduceLROnPlateau	**0.9564**
CosineAnnealingLR	0.9555

- 딥러닝 실험에 비교적으로 많이 사용되는 CosineAnnealingLR, ReduceLROnPlateau, StepLR를 바탕으로 실험 진행
- 실험을 통해 StepLR을 제외하고 성능이 비슷한 것을 확인함
- Cosine의 경우 이미지 크기가 증가할 때 Gradient Exploding 현상이 발생하여 학습이 잘 이루어지지 못하는 문제가 발생함

(4) Augmentation 실험

Augmentation (resize 512)	**Validation Dice**	**Leaderboard Dice**
Default	0.9564	0.9550
Rotate	0.9567	0.9562
Emboss	0.9560	0.9544
GridDistort	0.9549	0.9542
RandomBrightnessContrast	0.9555	0.9536
CLAHE	0.9570	0.9572
HorizontalFlip	0.9578	0.9574
Normalize	0.7031	N/A
ElasticTransform	0.9520	N/A
Sobel (resize 1024)	0.9667	0.9663

- 이미지에 노이즈나 왜곡을 주는 실험의 경우 성능이 떨어지는 결과를 얻음
- CLAHE의 경우 손가락 뼈와 손목 뼈의 윤곽을 부각시키는 경향성을 보여 실험을 진행하였으며, 0.002 성능 향상 효과를 얻음
- EDA 결과 손목이 꺾인 상태로 존재하는 이미지가 많다는 것을 확인하고 Rotate 실험을 진행하였으며, 0.0012 성능 향상 효과를 얻음
- 오른쪽, 왼쪽 손이 모두 존재하는 데이터의 특성을 바탕으로 HorizontalFlip 실험을 진행하였으며, 0.0024 성능 향상 효과를 얻음

(5) Resize 실험

Resize (UNet++)	Validation Dice	Leaderboard Dice
512x512	0.95703	0.9572
1024x1024	0.9717	0.9713
1536x1536	**0.9720**	**0.9719**

- RLE Encoding을 진행하기 위해서는 추론 결과를 원본 크기로 복원해야 하며, upscale을 많이 할수록 깔끔한 mask를 얻지 못하게 되므로 이미지 크기에 따른 실험 진행
- 이미지 크기를 높여 학습에 활용할수록 결과가 좋아지는 경향성을 파악함

마. 추가 실험

(1) Instance Segmentation

- Segmentation을 진행할 때 bbox도 같이 학습할 경우 탐지한 bbox 영역 내에서 정교하게 mask를 예측할 수 있을 것이라 생각하여 실험 진행
- 원본 JSON파일을 COCO format으로 변환한 후 YOLOv8-seg 모델로 학습 및 추론 진행
- mAP50을 기준으로 mask와 bbox 모두 0.99가 넘는 결과를 얻었지만, Dice score는 0.9436으로 기대보다 낮은 점수를 얻음
- 많은 클래스가 겹쳐져 있는 손바닥의 경우 한 픽셀당 하나의 클래스만 예측하여 multi-class로 학습이 진행되지 않음을 알게 되었으며, 해당 부분이 문제임을 파악함

(2) Pseudo Labeling

Pseudo Labeling (resize 1024)	Validation Dice	Leaderboard Dice
미적용	0.9717	0.9716
적용	0.9714	0.9643

- 가장 높게 나왔던 모델의 결과를 바탕으로 테스트 데이터의 정답을 생성하는 pseudo labeling 진행
- 이미지로부터 mask를 추론했을 때 클래스에 대한 score가 없기 때문에 학습에 사용할 데이터를 추출하는 클렌징 작업 진행
- 대략 200장 정도의 데이터를 추가하여 학습을 진행한 결과, 성능이 오를 것이라고 생각했던 가설과는 다르게 오히려 성능이 떨어지는 결과를 얻음
- Segmentation task 특성상 각 픽셀에 대한 정확한 레이블을 요구로 하기 때문에 오히려 점수가 떨어지는 경향을 얻게 되지 않았을까 생각함

바. 앙상블

(1) 모델 후보

② UNet++ (VGGNet-19)	① kfold0: 0.9720
	kfold1: 0.9708
	kfold2: 0.9705
	kfold3: 0.9703
	kfold4: 0.9706

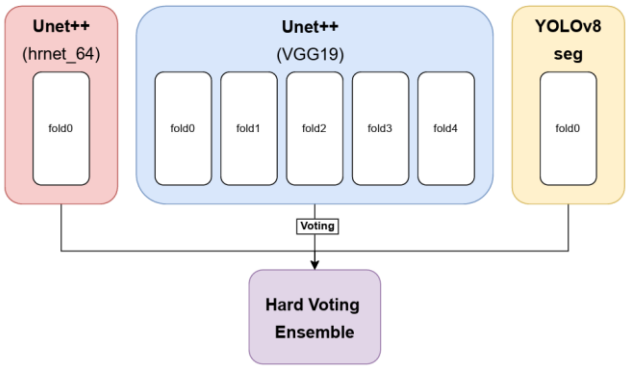
③ UNet++ (HRNet-64)	kfold0: 0.9726

④ YOLOv8-Seg (instance segmentation)	kfold0: 0.9436

(2) 앙상블 결과

Candidates	LB score
② (5-fold)	0.9737
① + ③ + ④	0.9741
② + ③ + ④	**0.9745**

(3) 최종 모델 선택



- 단일 모델로 점수가 높은 모델로 K-Fold를 진행하고, 이를 앙상블에 사용하였음
- YOLO 모델의 경우 타 모델에서 잘 예측하지 못하는 손바닥 클래스를 잘 예측하는 경향성을 가지고 있어 Ensemble에 활용함
- 최종적으로 앙상블을 통해 0.002정도의 성능 향상 결과를 얻음

사. 자체 평가 의견

(1) 잘했던 점

- 서로 진행할 실험을 이야기하고 노션에 공유하며 진행함
- 기본 베이스 세팅을 일정하게 정하고, 실험들을 진행하여 변인 통제가 잘 되었음
- Config 파일로 여러 실험 옵션을 설정할 수 있도록 실험 환경을 구축하여 다양한 실험을 체계적으로 진행하는데 많은 도움이 되었음

(2) 시도했지만 잘 되지 않았던 실험

- Hard-voting 앙상블을 진행하였으나 성능이 잘 나온 모델의 경향성을 따라가 성능 향상이 이뤄지지 않음
- Mmsegmentation을 이용한 실험의 성능이 좋지 못하였음
- Pseudo Labeling을 시도하였지만 유의미한 성능 개선이 이뤄지지 않음

(3) 아쉬웠던 점

- 학습 데이터와 함께 제공된 Metadata를 활용하려고 했지만, 시간 부족으로 인하여 실험을 진행하지 못함
- Instance segmentation 학습을 진행할 때 multi-label로 실험을 진행하지 못했던 부분이 아쉬웠음
- 실험 가설을 세울 때 명확히 개념을 이해한 후 가설을 세우기보다 좋은 선택지라고 판단되는 것을 단지 적용하기만한 부분이 조금 아쉬웠음
- 이전 대회에 비해 데이터를 보고 많은 가설을 세우지 못했던 점이 아쉬웠음

(4) 프로젝트를 통해 배운 점

- Mmsegmentation을 사용하여 multi-label segmentation을 진행하도록 수정하는 방법을 배울 수 있었음.
- 전체적인 실험을 config.yaml에서 관리하는 과정을 배움
- 본 대회에 열심히 참여하면서 Semantic segmentation의 전반적인 이해도를 높일 수 있었으며, 또한 의료 데이터를 다뤄보면서 어떻게 하면 성능을 개선할 수 있을지 많이 고민해볼 수 있는 경험을 하게 됨