Wrap up

▼ 목차

2주에 걸친 Stage 1 - Image Classification이 끝이 났다.

그동안 시도했던 방법들, 아쉽거나 부족했던 점들에 대해 적어보고자 한다.

💡기술적인 도전

최종 순위

- Public LB - 32 / 224

  F1 Score 0.7660

  Accuracy 81.1111%

- Private LB - 37 / 224

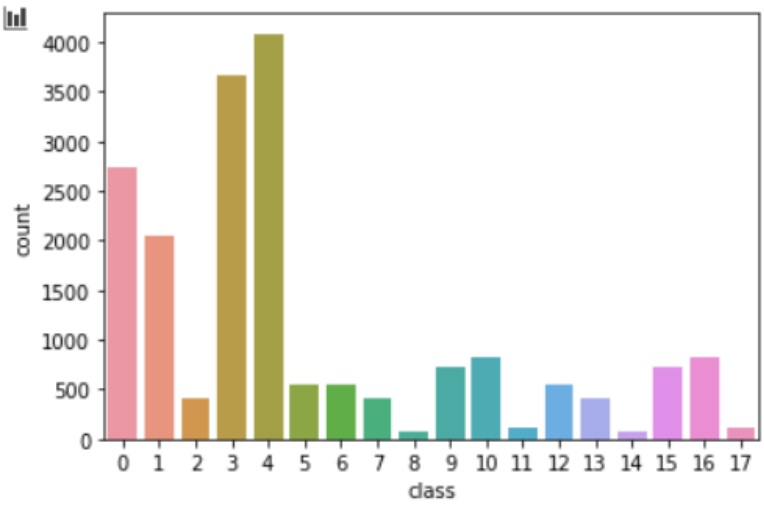
  F1 Score 0.7506

  Accuracy 80.4762%

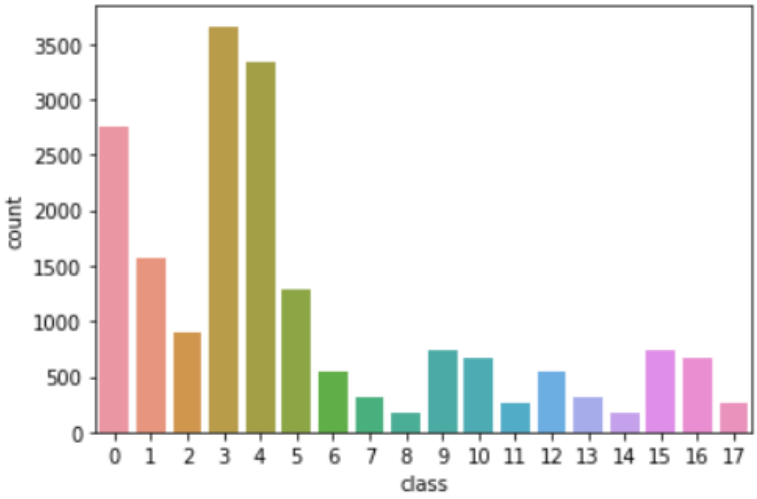
가장 큰 issue라고 생각한 내용

Class간 분포의 불균형

- 나이, 성별, 마스크 착용 여부를 고려한 결과 class간의 분포가 너무 불균형했다 (83 ~ 3660)



- 나이 기준을 60세 → 58세를 기준으로 변경 (179 ~ 3660)



- 해결 방법
  - 인위적으로 작은 개수의 class의 image를 augmentation해서 늘려주거나 (`Oversampling`), 큰 개수의 image를 작게 해주는 방법(`Undersampling`)이 있지만,
  - Loss 자체에서 해결해주는 방법을 사용했다 (Focal loss)
  - 인위적으로 image 개수를 조정하는 방법이 들인 노력에 비해 효과적이지 않다는 조언을 참고했다. (실제로 테스팅해보지 않아서 정확한 성능의 차이는 확인하지 못했다.)
  - EDA결과 성별 등 라벨이 이상한 부분은 수정해서 학습 진행

검증(Validation) 전략

- 적합한 parameter를 찾을 때는, train set : validation set = 8 : 2의 비율로 두고 진행했다
- 그 중 적합한 parameter가 나올 때에는 train set : validation set = 9.5 : 0.5의 비율로 두고 제출용 모델을 만들었다. 그 이유는 최종 제출할 때는 모든 데이터들이 학습되는 것이 이득이기 때문인데, 모든 데이터는 train set에 올인하면 그 중 어떤 모델이 좋은지 알 수 없기 때문이었다.
- K-Fold validation을 사용하는 것이 보통 사용하는 방법이고, 정석(?)이라고 생각하는데, 사용하지 않은 이유는 하나의 최종 모델을 만드는데 많은 시간이 소모된다고 느꼈기 때문이었다.

검증(Validation) 전략에서 아쉬운 점

- Baseline(Python project) 코드를 사용했는데, 마지막날에 생각해본 내용이라서 적용할 시간이 부족했지만, Train set과 Validation set을 나누는 기준이 random이기 때문에,

Validation set의 클래스 분포가 전체 데이터셋의 클래스 분포를 반영하지 못한다는 단점이 있다.

(`MaskSplitByProfileDataset` 기준으로는 약간 커버된다고 생각한다. 마스크 클래스에 대해서는 비율이 보존되기 때문, 그래도 반영하지 못한다)

- Validation metric을 accuracy를 사용했는데, 문제의 목표를 이해한다면, maximum f1 score & minimum loss를 기준으로 모델을 저장하는 방법을 사용해야 한다고 생각한다.

Model Architecture & Hyper parameters

- 공통 사항
  - Optimizer : `AdamP`

    → `SGD`는 학습 속도가 너무 느리고, `Adam`은 직접 테스트해보지 못했지만, `AdamP`가 더 우수한 성능을 나타낸다는 공통적인 이야기가 있어서 선정했다.
  - Dataset : `MaskSplitByProfileDataset`
  - Learning rate : 5e-4, Steplr

1. 아키텍처 : `EfficientNet - b5`
   - LB 점수 : F1 score - 0.7575, Accuracy - 80.5873%
   - Traning time augmentaion : `Center Crop`, `CLAHE`
     - CLAHE를 사용한 이유 : 얼굴의 특징을 강조할 수 있기 때문 → 나이 판별에 효과적일 것이라고 생각

- Img size : Center Crop한 이미지 크기 사용 (320 * 256)
- Loss : `F1 loss` + `Focal loss` (gamma = 5)
- 특이 사항

  다른 EfficientNet b5 backbone 모델보다 성능이 더 잘나왔던 이유는, Loss값을 두 개 사용했기 때문이라고 생각한다.

  Epoch은 30으로 설정하였고, 14 epoch 지점에서 최고 Validation Accuracy 85.74%를 기록한 모델이다.

2. 아키텍처 : `EfficientNet - b5`
   - LB 점수 : F1 score - 0.7478, Accuracy - 79.5714%
   - Traning time augmentaion : `Center Crop`, `CLAHE`

- Img size : Center Crop한 이미지 크기 사용 (320 * 256)
- Loss : `F1 loss` + `Focal loss` (gamma = 5)
- 특이 사항

  1번 모델과 차이점은 learning rate인데, 너무 빠르게 training set에 과적합되어서 (4 ~ 5에폭이면 거의 90% 중반의 accuracy 수렴) 일부러 lr을 1e-5로 하고, lr decay를 5epoch마다 주었던 모델이다.

  Epoch은 30으로 설정하였고, 최고 Validation Accuracy 85.54%를 기록한 모델이다.

3. 아키텍처 : `Resnext50`
   - LB 점수 : F1 score - 0.7317, Accuracy - 78.1587%
   - Traning time augmentaion : `Center Crop`, `CLAHE`
   - Img size : Center Crop한 이미지 크기 사용 (320 * 256)
   - Loss : `F1 loss` + `Focal loss` (gamma = 5)
   - 특이 사항

     EfficientNet과 다른 모델 구조를 사용했다.
4. 아키텍처 : `EfficientNet - b6`
   - LB 점수 : F1 score - 0.7134, Accuracy - 76.8730%
   - Traning time augmentaion : `Resize`, `Normalize`
   - Img size : Normalize한 이미지 크기 사용 (128 * 96)
   - Loss : `Focal loss` (gamma = 2)
   - 특이 사항

     Augmentation이 Base Augmentation이다.
5. 아키텍처 : `ViT` (Vision Transformer)
   - LB 점수 : F1 score - 0.7026, Accuracy - 76.2381%
   - Traning time augmentaion : `Center Crop`, `CLAHE`
   - Img size : Center Crop한 이미지 크기 사용 (384 * 384)
   - Loss : `F1 loss` + `Focal loss` (gamma = 5)
   - 특이 사항

     다른 모델과 달리 Transformer가 적용된 모델

Soft voting하면 기존의 모델과 구조가 다르기 때문에 성능이 좋아질 수 있다고 생각하지만, Hard voting을 시도했기 때문에 앙상블에서도 큰 성능 향상이 이루어지지 않았다.

Image의 input size가 모델에 종속적이다.

앙상블 방법

- 제출한 submission 기준으로 Public LB에서 가장 f1-score가 높았던 8개의 모델을 hard voting을 시도했다.
- 그 결과, 최고 Single model의 f1 score(0.7575)보다 성능이 향상되었다(0.7660).
- 8개의 모델 이외에 더 숫자를 줄이거나, 늘리거나, 가장 성능이 좋은 모델에 가중치를 두고 앙상블을 시도해도 첫 번째로 시도했던 성능을 넘지 못했다.
- 오히려 마지막날 제출기회를 거의 다 소모해버리기만 했다..!
- Soft voting을 시도하면 성능이 어떻게 변했을지 궁금한데 시도해보지 못해 아쉬웠다.

시도했으나 잘 되지 않았던 것들

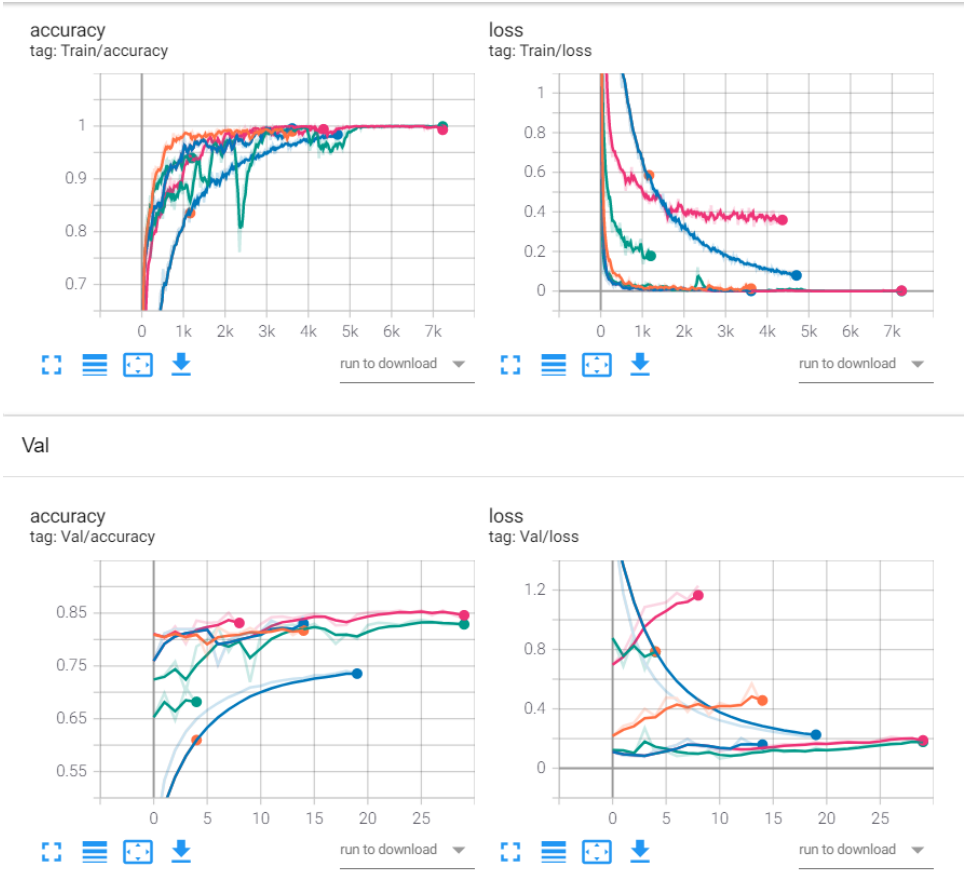
- 초반 30 epochs는 Focal loss로 진행하고, 그 중 best 모델을 저장하고, 그 뒤에는 F1 loss로 10 epoch정도 진행했다.

  오히려 best 모델보다 성능이 안좋아졌는데, 이 방법은 좋지 않은 것 같다,,!
- EfficientNet의 종류가 b0 ~ b7까지 있고, 숫자가 커질수록 더 규모가 큰(parameter가 많은)모델인데, b7이 좋지 않았다. 최고 점수를 기록한 모델은 b5모델이고, 그 다음은 b6 모델이었는데, 다른 hyper parameter를 고정하고, model만 바꾼 것이 아니기 때문에, 꼭 모델의 요인이라고만 볼 수 없기도 하다.
  하지만 피어 세션에서 다른 캠퍼의 이야기도 있고, hyper parameter의 요인이라기에는 성능 차이가 어느정도 나기 때문에, 꼭 큰 모델이라고 좋은 성능을 내는 것은 아닌 것 같다.

  피어 세션에서도 이야기한 부분이긴 한데, 한 캠퍼분의 이야기는, 더 좋은 모델을 쓰면 성능이 더 좋아질것 이라는 절대적인 룰이 존재한다고 생각했는데, 이번 경험을 계기로 그 생각이 바뀐 것 같다고 말했다. 근데 그 이유는 잘 모르겠다고 하셨다. Underfitting이라기에 적합하지 않은 training - validation score의 흐름이기 때문에, 어떤 이유인지 궁금하다.

💡 학습과정에서의 교훈

- 피어 세션에서 좋은 동료를 만난 덕에 많은 도움을 받았다. 매일 만나는 사람이 다르다보니 분위기 등등 많은 점이 새로웠지만, 그게 매일 랜덤으로 조를 만나는 묘미인것 같다.
- Hard voting하는 좋은 방법이나, loss를 두개 같이 쓰는 방법, CLAHE Augmentation 등등 많은 팁들을 받았고, 그만큼 내가 전달해주지 못한 것에 대한 아쉬움이 있다.
- 피어 세션 뿐만 아니라 토론게시판에서 EDA 결과를 공유하거나, training time을 줄이는 팁과 같이 정말 많은 정보들이 있었다.
- 이 밖에 많은 팁들과 좋은 방법을 제시해주셨지만, 얻은 정보에 비해 내가 들인 노력이 적어서 많은 시도를 하지 못한 것 같다.
- 다음 stage에서는 토론 게시판에 좋은 글을 써보는 것을 목표로 새로운 것을 시도해보면 좋을 것 같다.
- 그래도 얻은 내용은 유의미한 score를 얻었다는 것과, Python project 기반 pipeline의 구조를 이해하는 데 많은 도움이 되었다는 것이다.
- `Tensorboard`라는 tool을 처음 사용해보는데, 사용법에 점점 적응하면서 편하게 사용했다.



💡 마주한 한계와 도전 숙제

아쉬웠던 점들

- 성킴님 마스터 세션에서 올인하라는 내용을 많이 강조하셨는데, 지금 돌아보면 2주동안 올인했나?에 대한 질문에 선뜻 그렇다고 대답하기 어려운 것 같다.
- `nni`, `wandb` 등등 정말 많은 툴들을 소개해주셨는데, 모델 돌리기에 급급해서 많이 적용해보지 못한 것이 아쉽다.
- Baseline 코드가 좋아서 그 코드를 기반으로 진행했는데, 대회가 끝나고 나서 든 생각은 Baseline 코드 안에 갇혀있었다는 느낌을 받았다.
- 새로운 것을 시도하고 동작 방식을 이해하려면 Baseline은 단지 Base로 생각해야 하는데, Baseline에 너무 의존했다는 생각을 했다.
- 순위적으로는 마감 하루 전 밤에 10등이었는데, 어느순간 public이 32등이 되어있고, private는 37등,,
- 순위에 그렇게 신경쓰지 않았지만 막상 아쉽다.

한계/교훈을 바탕으로 다음 스테이지에서 새롭게 시도해볼 것

- 토론 게시판에 글 쓰기
- `nni`, `wandb` 적용해보기
- 새로운 내용 찾아서 시도해보기
- 학습 정리 잘하고, 실험 내용 기록 잘하기 (마지막에 정리하기 어렵다)
