**최종 보고서**

**PlantTraits2024**

이름 : 이재훈

학번 : 120230458

연구실 : 지능형 연결시스템 (ICSL)

1. **Challenge 소개**

* CVPR 2024의 FGVC11 워크숍의 일환인 이 챌린지는, 식물 이미지와 데이터를 통해 6가지 특성을 예측하는 것이다. 챌린지 주최측에서 준 데이터를 가지고 학습하여 test 데이터를 예측하여 결과 값을 제출하는 형태이다. (https://www.kaggle.com/competitions/planttraits2024/overview)
* 총 6가지의 Target Column을 예측하는데 아래와 같다.
  1. X4 (Stem specific density, SSD)
  2. X11 (Specific leaf area, SLA)
  3. X18 (Plant height)
  4. X26 (Seed dry mass)
  5. X50 (Leaf nitrogen content per leaf area)
  6. X3112 (Leaf area)
* 이 challenge의 목적은 crowd-sourced 식물 이미지들과 데이터를 사용하여 식물의 생태적 특성을 파악하고, 다양한 식물 이미지와 기후 데이터를 통해 식물의 주요특성(잎 면적, 식물 높이 등등)을 예측하여 기후 변화가 생태계에 미치는 영향을 분석하는 것이다.
* Train data는 식물 이미지 사진과 이미지에 해당하는 값들인 테이블 데이터로 이루어져 있는데 테이블 데이터는 아래와 같이 크게 세가지의 데이터로 나눌 수 있다.
  1. 기후 데이터
     + WORDLCLIM\_BIO1\_annual\_mean\_teamperature : 연평균 온도
     + WORDCLIM\_BIO12\_annual\_precipitation : 연간 강수량
  2. 토양데이터
     + SOIL\_bdod\_0.5cm\_mean\_0.01\_deg : 0.5cm 깊이에서의 토양밀도
     + SOIL\_bdod\_15.30cm\_mean\_0.01\_deg : 15-30cm 깊이에서의 토양 밀도

3. 위성 데이터

* + - MODIS\_2000.2020\_monthly\_mean\_surface\_reflectance\_band\_01\_.\_month\_m1 : 2000년부터 2020년까지 매월 1월에 수집된 첫번째 밴드(가시광선 영역의 데이터)의 월평균 지표 반사율
    - VOD\_C\_2002\_2018\_multiyear\_mean\_m01 : 2002년부터 2018년까지 매년 1월에 수집된 C 밴드 식생의 밀도와 생물량의 다년 평균값
* Target Column들을 포함하여 총 176개의 column들이 존재한다.
* Evaluation의 경우 아래와 같이 R2 Score로 진행된다.

폰트, 라인, 화이트, 텍스트이(가) 표시된 사진

자동 생성된 설명

* Challenge에는 총 두 가지 Score가 존재한다.
  1. 하나는 Challenge가 끝나기전 모두에게 보여지는 Public Score인데 test data를 모델이 예측한 값들을 .csv 형식으로 만든 데이터중 59퍼센트만 가지고 evaluation한 score이다.
  2. **다른 하나는 private score인데 위에서 evaluation하지 않은 41퍼센트의 데이터를 challenge가 끝난 후 공개하는 score이다. 최종 등수는 private score를 가지고 매겨진다.**

개발 환경 – i9, rtx4090 24GB, RAM 128GB, SSD 1TB

**실행방법**

1. <https://www.kaggle.com/competitions/planttraits2024/overview> 에서 data를 다운받는다.

2. create\_test\_set.ipynb를 실행한다. (데이터가 있는 위치 조정 필요)

3. source\_code\_120230458.ipynb를 실행한다. (데이터가 있는 위치 조정 필요)

**2. 모델 선정**

1. Swin Transformer (swin\_large\_patch4\_window12\_384.ms\_in22k\_ft\_in1k)
   * 대규모 이미지 인식 및 다양한 해상도에서의 작업에 특화
2. Vision Transformer (vit\_large\_patch16\_384)
   * Self-Attention Mechanism으로 이미지 분류에 특화
3. EfficientNetv2 (efficientnetv2\_s)
   * 적은 자원으로도 높은 성능을 내는 자원 효율적인 이미지 인식에 특화

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* + **Swin Transformer가 셋 중에서 가장 좋은 성능을 보여주기에 모델로 선정**

**3. Loss Function 과 Optimizer 선정**

* + Smooth L1 Loss (Huber Loss)

텍스트, 폰트, 라인, 화이트이(가) 표시된 사진

자동 생성된 설명

* + 1. 작은 오차에 대해서는 MSE를 사용하고, 큰 오차에 대해서는 MAE를 사용하여 안정적으로 학습을 할 수 있다.
    2. 데이터셋에 Outlier가 존재할 수도 있고 작은 오차를 최소화하여 예측 정확도를 높이는 것이 이번 프로젝트의 목표이기 때문에 선정.
    3. 후에 approach에서 기술할 예정이지만, Smooth L1 Loss을 수정하여 weighted Smooth L1 Loss를 사용하였다.
  + AdamW (Adaptive Moment Estimation with Weight Decay)
    1. Weight Decay(가중치 감쇠)
       1. 가중치가 지나치게 증가하는 것을 방지하고 모델의 일반화 성능을 향상
    2. Adaptive Learning Rate (적응형 학습률)
       1. 고정 LR이 아닌 적응형 LR을 사용하여 학습이 빠르고 안정적으로 이루어지도록 한다.
    3. Adaptive LR과 Momentum을 사용하여 기존 SGD에 비해 효율적으로 최적화를 수행하고 일반화 성능을 향상시키기 때문에 식물 특성 예측에서 강점을 발휘할 것이라 생각하였다.

**4. Approaches to Improve Performance**

* **Transfer Learning**
  + 이미 사전 학습된 가중치를 사용하여 모델을 처음부터 학습시키는 것보다 적은 데이터와 시간을 소요하여 좋은 성능을 이끌어 내었다.
  + Microsoft의 ImageNet-22K 데이터셋으로 pre-trained하였고 ImageNEt-1K 데이터셋으로 fine-tuning을 한 모델인 swin\_large\_patch4\_window12\_384.ms\_in22k\_ft\_in1k 를 사용하였다.
* **Weighted Loss Function (Custom)**
  + Validation set(training set의 10%)를 사용하여 GT와 예측값 차이를 기록하였을 때 아래와 같았다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* + 표는 epoch 20, 25일 때 GT와 모델의 예측값 차이를 나타낸 것이다. X18은 예측이 잘 되는 반면 X50은 예측이 잘 되지 않는 것을 볼 수 있다.
  + 그렇기 때문에 Loss Function에 학습이 더딘 Target Column에 가중치를 주고 학습이 비교적 잘되는 Target Column에 가중치를 덜 준다면 다른 column들과 같이 0.4 에 수렴할 것이라는 생각으로 시작하였다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* + 이렇게 weight를 주어 X50 column에 대해 학습이 더 잘 될 수 있도록 조정하였다. 기본 적인 형태는 smooth l1 loss지만 마지막에 loss에 weight를 곱하여 학습이 타 target column들보다 더딘 target column이 학습이 더 잘될 수 있도록 설정하였다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 위와 같이 고정된 weight가 아니라 epoch마다 예측값 차이대로 weight를 dynamic하게 수정할 수 있도록 설정을 해보았는데 아래와 같이 static이 dynamic보다 훨씬 높은 score를 보여준다. (diffs는 validation set을 이용하여 예측값과 GT의 차이를 0-1사이 값으로 표준화한 값이다.)

텍스트, 폰트, 라인, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* + 그 이유는 매 epoch마다 X50에 대한 weight가 높아지고 다른 target column(특히 X18)에 대한 weight가 0.1이라는 최저 weight로 계속 낮아지기 때문에, 다른 target column들이 제대로 학습이 되지 않아 score가 static보다 더 낮게 나오는 것이다.
* **Train Image Blending**
  + 학습 데이터셋에 비해 많은 feature들이 존재하기 때문에 더 많은 데이터셋의 필요성을 느껴 Image Blending을 생각하였다.
  + 투명도 조절을 통해 서로 다른 이미지를 합성하는 것인데 투명도가 너무 낮으면 feature가 잘 보이지 않기 때문에 적절하게 0.4:0.6으로 설정하여 합성을 하였다
  + Opencv의 addweighted라는 함수에 ration를 0.4로 조정하여 새로운 이미지를 만들었다. 합성과 동시에 테이블데이터도 투명도의 비율에 맞게 아래와 같이 추가하였다.
    - **0.4\*이미지1의 테이블데이터 + 0.6\*이미지2의 테이블데이터 = 새로운 이미지의 테이블 데이터**
  + 아래는 생성 코드와 결과물 중 하나이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

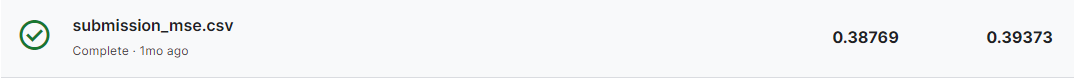
텍스트, 스크린샷이(가) 표시된 사진

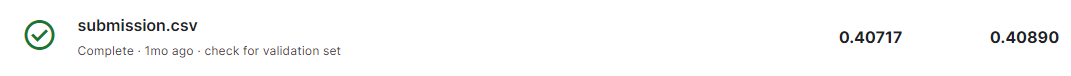
자동 생성된 설명

식물, 나무, 야외, 꽃이(가) 표시된 사진

자동 생성된 설명

* + 이미지에는 꽃 이미지와 나뭇잎 이미지를 블렌딩하여 새로운 이미지를 만든 것이다.
  + 이렇게 총 25000개정도의 학습 데이터를 더 생성하였고 해당 데이터로 학습한 결과 아래와 같이 Score가 상승하였다.





* **Randomforest Regression**
  + Random Forest는 여러 개의 의사결정 트리를 모아서 사용하는 앙상블 학습 기법이다. 각 트리는 무작위로 선택된 데이터와 feature를 사용하여 학습을 진행하며, 최종적으로 개별 트리 예측의 평균을 통해 이루어진다. 이번 프로젝트에서 Random Forest Regression을 사용하여 Target Columns를 예측하는데 사용하였다.
  + 즉, RandomForest를 활용하여 Target Column과 가장 중요도가 높은 상위 feature들의 정보를 추출한 후, 해당 feature들에 대해 학습이 더 잘되도록 기존 train 데이터에 해당 feature들만 이루어진 데이터를 더 추가한다. 코드는 아래와 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* + - 각 Target Column에 대해서 실행하는데 순서는 아래와 같다.

1. Target column 및 불필요한 column을 제외한 feature data를 추출한다.

2. Random Forest Regressor 모델을 초기화한다. (n\_jobs=-1은 병렬설정)

3. feature data(x)와 target column데이터(y)를 사용하여 모델을 학습시킨다.

4. 모델로부터 각 feature의 중요도를 추출하고 중요도를 기준으로 데이터프레임에서 내림차순 정렬한다.

5. 각 target column별로 중요한 상위 20개 feature들을 저장 및 중복 제거 후, train 데이터에 추가한다.

* + - 적용한 Score는 아래와 같다.

텍스트, 폰트, 라인, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* **Data Augmentation**
  + Albumentation이라는 라이브러리를 사용해 데이터 증강을 하여 모델의 일반화 성능을 향상시켰다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

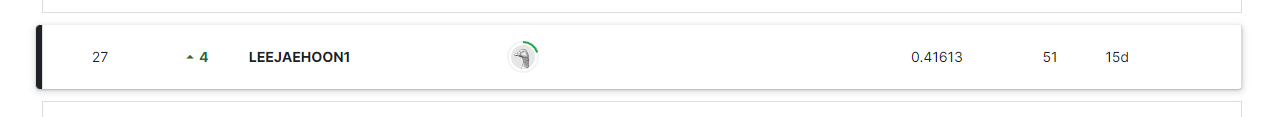
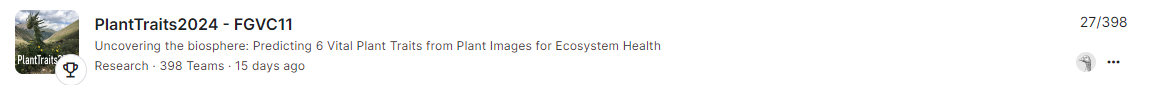
자동 생성된 설명

* + 이미지를 좌우로 뒤집고, [448. 512] 크기 범위에서 무작위로 이미지를 자르고, 특정 이미지 크기로 리사이즈를하여 모델 입력 크기가 일관되어 들어오도록 하고, 이미지의 밝기와 대비를 무작위로 조정합니다.
  + 그리고 이미지를 float 타입으로 변환하여 이후 정규화와 텐서변환을 할 수 있도록 준비합니다.
  + Normalize를 통해 이미지를 정규화하고 ToTensorV2() 를 통해 이미지를 pyTorch 텐서로 변환하여 모델 입력으로 사용합니다.
* **Hyperparameter Tuning**
  + 다양한 Hyperparameter에 대해서 테스트를 진행해보지 못했지만 Batch크기, MAX LR, Weight Decay에 한해서 진행하였고 그 중 Score가 제일 높았던 수치로 고정하였다.
  + Batch Size : 16
    - 8은 Batch 가 너무 작기 때문에 성능이 나오지 않았고 32는 GPU메모리가 부족하기 때문에 실험을 할 수 없었다.
  + MAX LR : 1e-4
    - (1-5)e-4 까지 총 5개를 실험을 해보았으나 1e-4가 가장 Score가 높았다
  + Weight Decay : 0.02
    - AdamW의 Weight Decay hyperparameter이다. 0.01, 0.02, 0.05로 실험하였는데 0.05는 너무나도 안 좋은 score가 나왔으나 0.01과 0.02는 근소하게 0.02가 0.01정도 차이로 성능이 좋았기 때문에 해당 수치로 고정하였다.

**텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명**

**5. 최종 등수 및 결과**

- 위에서 기술한 모델들과 성능을 높인 approach들을 사용하여 아래와 같은 등수를 기록하였다. 

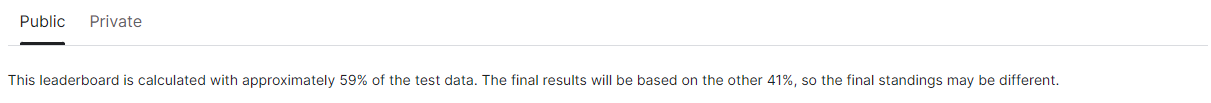
- **398팀 중 27등으로** 마무리하였으며 해당 등수는 아래의 링크에서 확인할 수 있다.

<https://www.kaggle.com/competitions/planttraits2024/leaderboard>?

Public은 final result가 아니라 테스트 데이터의 59퍼센트만 사용하여 측정된 결과이며, private파트가 최종결과이다. Private는 나머지 41퍼센트의 테스트 데이터를 사용하는데 해당 결과는 competition이 끝난 후 공지된 결과이다.

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

**6. Future Work**

* 멀티 모달을 활용하여 성능 높이기
  + 이미지 데이터 백본으로 Swin Transformer, 테이블 데이터 백본을 다른 모델을 사용한 멀티 모달로 성능을 높일 예정이다.



* + 위는 테이블 데이터 백본을 간단한 MLP로 작성하였는데도 불구하고 높은 Score를 보여주는 것을 보아, 다른 모델을 사용할 경우 더 나은 score를 볼 수 있을 것 같다.
* Clipping을 활용하여 학습 데이터를 더 만들 예정이다
  + 한 이미지의 어느 한 부분 (portion)을 자른 다음 다른 이미지의 동일한 위치에 붙이는 방식이다. 동시에, 데이터의 경우 해당 부분이 전체부분에서 차지하는 만큼을 비율로 정하여 더한다음 새로운 데이터를 만들어내는 것이다.

야외, 파켈리아속, 식물, 꽃이(가) 표시된 사진

자동 생성된 설명

* + 하지만 성능이 매우 좋지 않았는데 그 이유는, feature값을 ratio로 병합할 때 feature가 없는 곳을 clip할 경우, 실제 존재하지 않는 feature가 테이블 데이터에 포함될 수 있기 때문에 성능이 좋지 않은 것 같다.
  + 다른 regression모델을 학습시켜, target column를 판단하는데 중요도가 높은 feature가 포함되는 부분을 clip하고 해당 부분의 데이터를 다른 이미지 데이터와 병합할 경우, 성능이 개선될 것이라 생각하여 시험할 예정이다.