P-Stage 2 Object Detection 대회 개인 회고

T2234 최현진

1. 프로젝트 목표

- 주어진 쓰레기 객체가 담긴 이미지와 bbox 정보를 학습데이터로(4882장) 사용해 객체를 감지할 수 있는 Object Detection Model 개발
- Detection Model 학습에 사용되는 다양한 라이브러리의 사용법, 모델 이론 및 개념을 학습하여 모델 성능 개선
- 전체적인 pipe line 구축과 협업 과정에서 필요한 Tools 개발해 보면서 MLOps의 역량 갖추기

2. 목표 달성 계획

개인 학습

- Object Detection Task에서 사용되는 2-Stage, 1-Stage 모델의 이론과 코드 이해
- MMdetection, Detectron2, YOLOv5 라이브러리 사용법을 습득하고 모델 생성
- EDA, Model 결과 및 분석, Tools 사용을 통해 성능을 향상 시킬 수 있는 방법 도출

공동학습

- Object Detection에 대한 개념 학습
- 문제 해결을 위한 다양한 접근 방법 공유 (Customizing, Augmentation, Ensemble,...)
- Detection 과제 SOTA 논문 스터디 (Swin Transformer, EfficientDet, ...)

3. 모델 개선 방안

- Model Search

  Detection Task에서 많이 사용되는 모델들을 쓰레기 분류 문제에 적용해 보고 비교 하여 Baseline Model 찾기

- Bounding Box

  데이터셋 EDA를 통해 다양한 크기의 bbox 확인. Anchor Box 크기를 다양하게 설정하여 학습 해보기

- Augmentation

  Kaggle의 Detection 과제에서 상위권 개발자들의 Augmentation 기법을 참고하여 적용하기(Mosaic, Mix-up, Color, ...)

- Ensemble

  최종 모델의 다양성을 확보하기 위해 1-Stage 모델, 2-Stage 모델 Ensemble

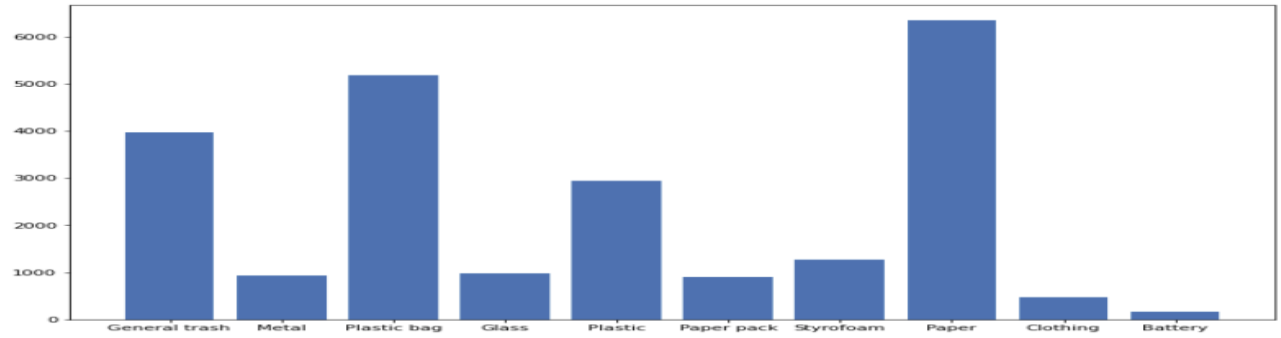
  팀원들과 Seed를 다르게 하여 같은 baseline 위에서 생성한 모델들을 Ensemble (Seed, Snapshot)

  Resolution 사이즈를 다양하게 설정하여 여러 모델을 생성하고 Ensemble

- Pseudo Labeling

  가장 학습이 잘된 모델로 테스트 데이터를 라벨링 하고 학습데이터로 추가하여 모델 다시 학습

- Class Imbalance

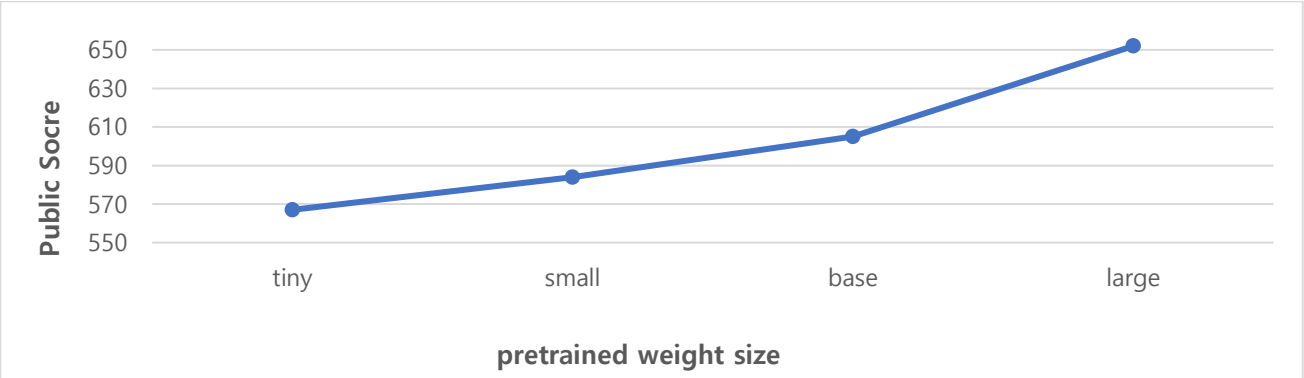


EDA를 통해 객체 class 불균형 발견, 2-Stage model의 classification head에서 Focal Loss 적용

4. 행동 결과 및 깨달음

- Model

  모델 선정 과정에서 다양한 라이브러리 위에서 여러가지 모델의 성능을 비교해 보았습니다. 그 중에서 최근 SOTA로 소개된 **Swin Transformer**를 Cascade R-CNN의 backbone 모델로 사용했을 때 가장 좋은 성능이 나와 baseline model로 선정하였습니다. 모델의 성능을 개선하기 위해서 Swin Transformer의 모델 크기와 **pretrained weight**을 다르게 가져갔을 때 다시 성능 차이가 있었습니다.



  Pretrained weight의 크기가 큰 것을 사용할수록 score 향상이 있었고 최종적으로 가장 큰 사이즈를 사용하였습니다. 하지만 학습하는 시간이 모델이 큰 만큼 많이 소요되었고 여러가지 실험을 하기에는 부적합했습니다. 비교적 학습시간이 적게 소요되는 작은 모델에서 여러가지 실험을 하고 큰 모델에 적용하는 방식으로 했으면 하는 아쉬움이 남습니다.

- Bounding Box

  2-Stage 모델의 Anchor box를 다양하게 설정하여 여러 모델을 학습시키고 Ensemble 하는 방법을 적용해 보았습니다. EDA 과정에서 학습 데이터가 굉장히 작은 bbox도 있다는 것을 알게 되어 적용한 방식입니다. 스코어를 소폭 상승시킬 수 있었습니다.

- Augmentation

  ColorBrightness, Mosaic, Cutmix 같은 다양한 augmentation을 적용했을 때 모델의 일반화가 잘 이루어져 스코어를 소폭 상승시킬 수 있었습니다. 이미지 분류와 마찬가지로 평가 데이터 점수를 향상시키기 위해 모

델 일반화의 중요성을 많이 느꼈습니다.

- Ensemble

  Weigthed Boxes Fusion 기법을 적용해 여러가지 모델을 ensemble을 했을 때 많은 성능 향상이 있었습니다. 특히 1-Stage 모델과 2-Stage 모델을 ensemble한 결과가 가장 큰 성능 향상이 있었는데 Detection 문제에서 다양성을 확보하는 것이 중요하게 작용하는 것으로 알 수 있었습니다.

  학습 모델의 결과를 시각화 하기 위해 자체적으로 제작한 프로그램을 사용했을 때 ensemble한 결과가 한 이미지 내에서 많은 Bounding box를 예측하였습니다. 이 결과로 **mAP**의 특성을 이해할 수 있었습니다.

mAP:0.531	mAP:0.668
	

  왼쪽 모델의 bounding box는 오른쪽 모델보다 사람이 봤을 때 더 합리적인 결과로 보였습니다. 하지만 최종 스코어는 오른쪽 모델이 훨씬 높았는데 **점수 산정 방식(mAP)**에서 더 많은 box를 예측하는 것이 패널티로 작용하지 않는 특성이 있었습니다. 이 결과로 다양한 모델들을 ensemble 했을 때 점수가 높아지는 이유를 알 수 있었고 confidence score threshold를 낮게 설정해 많은 box를 예측하도록 하는 방법 또한 적용해 볼 수 있었습니다.

- Class Imbalance

  클래스 불균형 문제를 해결하기 위해 2-Stage 모델의 classification head에서 사용되는 Loss 함수를 Focal Loss로 사용했습니다. 눈에 띄는 성능 향상은 없었지만 일반적인 Cross Entropy Loss를 사용했을 때 보다 아주 소폭 상승했습니다.

5. 새롭게 시도한 변화 및 효과

- Pseudo Labeling

  처음으로 Competition에서 사용할 수 있는 기법인 pseudo labeling을 적용해 보았습니다. 이미지 분류 대회에서 많은 성능 향상이 있었다는 다른 캠프 분들의 이야기를 듣고 감지 대회에서 적용했는데 큰 효과는 없었습니다. 학습 데이터와 검증 데이터의 분포가 크게 다르지 않았던 것 같습니다. 하지만 개념을 이해하고 검증 데이터를 학습 데이터 형태로 만드는 코드를 만들어 보면서 많이 배웠습니다.

- Tools

  학습한 모델이 평가 데이터의 bbox를 어떻게 예측하는지 편리하게 시각화 할 수 있는 프로그램을 PyQt를 사용해 제작하였습니다. 여러 모델의 결과를 비교할 때 유용하게 사용하였는데 특히 mAP가 높은 모델의 예측 결과를 관찰하면서 점수 산정에 대한 특성을 파악할 수 있었고 뿐만 아니라 팀원들과 모델 개선 방향을 설정할 때 유용하게 사용하였습니다.

  학습 데이터를 편하게 관찰하기 위해 마찬가지로 프로그램을 제작하였습니다. level 1 에서 데이터셋을 눈으로 직접 관찰하는 것이 굉장히 중요하다는 것을 많이 느껴 이번 프로젝트에서 편리하게 관찰할 수 있도록 만들어보고 싶어 제작하였습니다. 학습데이터를 눈으로 직접 관찰하여 EDA 과정에서 많은 도움이 되었고 모델 개선 방향 아이디어도 많이 얻었습니다.

6. 한계 및 아쉬웠던 점

- EfficientDet

  Kaggle의 다양한 Detection Task에서 상위권 개발자들이 사용한 모델로 소개되어 사용해 보려고 하였으나 성능이 좋지 못해 사용하지 않은 모델입니다. 대회가 끝나고 개념적으로 다시 한번 복습하고 코드를 점검해 문제점이 무엇이었는지 파악하려고 합니다.

- Augmentation

  Mosaic, Mix-Up 등 다양하게 적용했을 때 소폭 성능 향상이 있었지만 여전히 자세한 이유를 모르고 사용한 경향이 있습니다. 많은 실험을 통해서 적절한 Augmentation을 사용할 수 있는 직관을 길러야 할 것 같습니다.

- Ensemble

  동일한 모델을 반복적으로 ensemble 하다 보면 bbox는 매우 많이 생기게 되는데 전부 그런 것은 아니지만 bbox가 많을수록 점수향상이 있었습니다. 점수 산정 방식을 이해하고 경진 대회에 참여하는 것도 중요하지만 엔지니어 입장에서 이미지 한장에서 수많은 box를 예측하는 모델이 과연 좋은 모델인지 괴리감이 있었습니다. 결국 모델의 다양성을 위해 아키텍처가 다른 모델에 적용하는 것은 어느 정도 납득이 되어 적용하였지만 반복적인 ensemble은 사용하지 않았습니다.

- Cross Validation 전략

  단순히 scikit-learn의 Stratified Sampling을 적용해 검증 데이터셋을 만들어 사용하였습니다. 검증 데이터도 분포가 잘 이루어졌는지 Validation score가 오르면 Public score가 오르는지 확인하는 시간을 충분히 가져갔어야 하는데 그러지 못한 아쉬움이 있습니다.

- Hyperparameter

  팀원들과 baseline model을 정하고 hyperparameter를 실험하지 못한 아쉬움이 있습니다. 특히 대회 막바지에서 사용했던 모델 (Cascade R-CNN, YOLOv5) 모두 기존에 설정한 epoch보다 증가시켰을 때 성능이 점점 더 좋아지는 것을 확인하였는데 이 부분에서 역할을 분배해 실험했다면 더 좋은 결과가 있었을 것 같습니다.

7. 다음 P-Stage에서 시도해볼 것

- 전체 pipe line을 소스코드로 구현하기
- Kaggle에서 비슷한 Task에 대한 Solution 더 많이 읽어보고 적용해보기
- 완성된 모델을 활용하여 실제 서비스 만들어 보기 (모바일, 웹)
- Pair Programming으로 구현에 어려움이 있는 팀원 없게 하기