DeepFashion2

X

Fast R CNN

CP2 자기주도 프로젝트

AI_10_김형대

목차

1. 개요

▶ 프로젝트 주제 - 패션이미지 객체 탐지 모델

옴니어스(패션이미지 안에 객체 분류, 이를 이용한 서비스)기업은 패션 산업에 딥러닝을 도입한 회사입니다. 이 회사는 이미지에 옷을 탐지하고 이에 맞춘 서비스를 기업에게 제공하는 B2B 회사입니다. 이미 패션 산업에서는 딥러닝을 통한 다양한 서비스를 제공하고 있습니다. 옴니어스는 딥러닝 모델을 통해 패션이미지를 분석하고 해쉬태그 생성, 추천, 트렌드 분석 등 다양한 서비스를 제공하고 있습니다. 옴니어스는 동일 제품이지만 각도,빛,장소,촬영자,카메라에 따라 다르게 보이는 사진을 정밀하게 탐지할 수 있는 모델이 필요합니다. 필자는 이 문제를 풀기 위해 2 stage 모델이 적합하다 판단하고 R-CNN 계열 모델을 연구했습니다.

이전 프로젝트에서는 R-CNN 모델을 연구했고 이번 프로젝트에서는 한 단계 업그레이드 버전인 Fast R-CNN 모델을 연구했습니다.

최종 목표는 dectectron2 모델까지 연구하고 R-CNN 이 어떤 과정을 거쳐 진화하였고 어떤 점이 개선되어야 하는 지 파악하고 해결책을 찾아 더 정확하게 탐지하는 모델을 만드는 것입니다.

▶ 데이터 - DeepFashion2

데이터는 DeepFashion2 데이터를 사용했습니다. 이 데이터는 json 파일로 annotation이 기록이 되어있고 약10만개 데이터가 있기 때문에 모델 구현에 적합하다 판단했습니다. 옷 종류를 13개로 분류했고 item 마다 bounding box 데이터 뿐만 아니라 landmark, segmentation 등 다양한 정보들이 있어 데이터 전처리 과정이 용이 했습니다.

▶ Convolution Network - VGG16

중심이 되는 CNN 모델을 VGG16 모델을 선택했습니다. 3X3 필터를 사용하여 연산을 하기 때문에 같은 연산 횟수에서 큰 사이즈의 필터보다 깊은 층을 만들어 낼 수 있고 보다 많은 파라미터를 얻을 수 있습니다. 때문에 효율적으로 학습이 가능하다 판단했습니다.

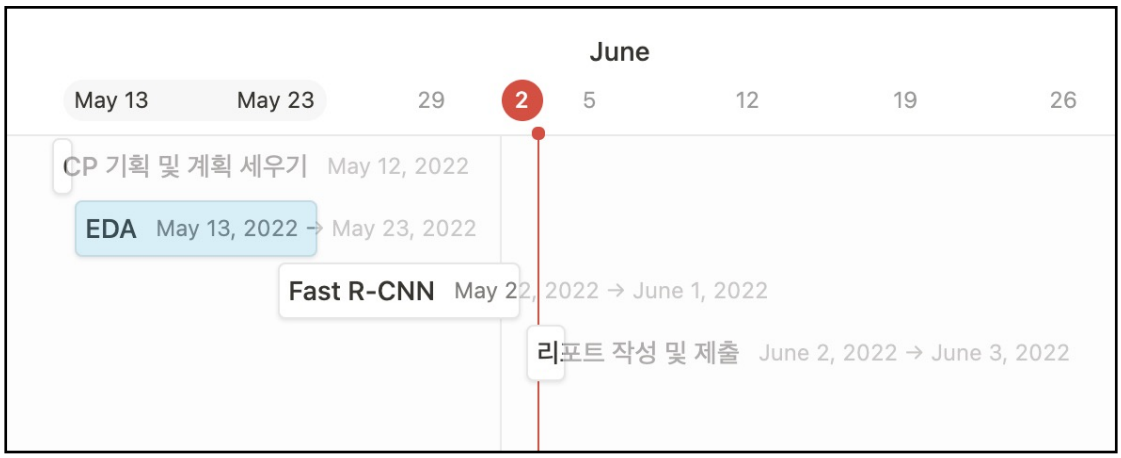
▶ 작업 환경 - Colab

Colab 은 성능이 좋은 GPU를 제공하기 때문에 로컬 GPU 보다 원활한 작업이 가능했습니다. 대부분 Tensorflow.Keras 라이브러리를 사용하였고 CV2 에 있는 selectiveSearch 사용하여 Roi pooling을 만들 때 사용했습니다.

2. 수행 방법

▶ Milestone

CP2 프로젝트는 CP1에 이은 프로젝트이기 때문에 기획하는 데 시간을 많이 쓰지 않았습니다. CP1 때 부족했던 부분을 깊게 학습하며 프로젝트를 진행했습니다.



프로젝트 Milestone

▶ 데이터 준비

이미지와 annotation 파일을 DL을 위한 데이터로 가공하는 법을 몰라 시간을 많이 사용했습니다. 코드를 해석하고 관련 공식문서 등을 읽어 보면서 자연스럽게 깊은 학습을 했고 컴퓨터 비전을 위하여 어떻게 데이터를 준비해야하는 지 터득했습니다.

▶ 모델 구축

VGG16 모델을 사용하되 Fast R CNN 에 맞게 구성을 달리 해야 합니다. Input 층은 2개로 해야하며 1개는 학습에 다른 하나는 RoI를 찾고 pooling하는 데 사용합니다.

Output 층 또한 2개로 늘려야 하며 1개는 Bounding box regressor, 다른 하나는 결과값을 Softmax로 출력합니다.

3. 결과

▶ 학습 데이터 전처리

Deepfashion2 - validation data 3만개를 이용했습니다. Train 데이터는 약 10만개이기 때문에 긴 학습시간이 예상되므로 비교적 적은 양의 데이터를 사용하여 빠른 학습과 fine tnuning 에 집중했습니다.

이 데이터는 13개 category_id로 숫자와 이름으로 라벨링이 되어 있고 그 외의 다양한 정보가 있습니다. Boubnding box, category_id, category_name, id 정보를 json 형식 파일에서 가져와 annotation 용 데이터프레임을 만들고 이 순서에 따라 이미지 파일 리스트를 만들었습니다. 한 이미지에 2개의 아이템이 들어있는 경우가 있기 때문에 같은 사진이 다시 한 번 등장하는 경우가 있습니다.

이미지 리스트는 다시 image_to_array 함수를 이용하여 np.array로 변환하여 파일리스트 순서대로 삽입했습니다. 리스트는 id번호가 기준이 되어 서로 쌍을 이루도록 했습니다.

▶ 모델 개요

Fast R-CNN 은 CNN을 한 번만 거친 후에 이미지가 있는 지역을 가져오는 방법으로 이전보다 효율적으로 학습합니다. RoI는 SelectiveSearch 알고리즘으로 찾고 IoU Metrics를 통해 적합한 구역을 선정합니다. 그 후 그 선정된 구역을 합성곱이 끝난 feature map에 projection 합니다. 이 때 Roi window는 원본 이미지 크기가 feature map 크기만큼 줄어든 비율로 곱하여 축소된 boundiong box 값을 적용해야 합니다.

RoI Pooling 은 VGG16 의 5번째 Pooling layer 대신 구축합니다. 5번 째 Pooling layer가 vgg16 모델의 마지막 pooling 이기 때문입니다.

VGG16 모델을 가져왔지만 input layer는 사용하지 않았습니다. 이 모델은 imageNet 가중치를 사용하고 기본적인 구조를 그대로 사용합니다. 때문에 input layer는 vgg16에 맞는 데이터 형식으로 구성했습니다.

4. 자체 평가

▶ Output layer 구축

Fast R CNN 에서 마지막 출력층은 2개로 합니다. 각각 Bounding box regressor와 softmax로 예측하기 위한 출력층입니다. 아직 어떤 구조로 레이어를 구성해야하는지 파악하지 못했습니다.

Relu 함수를 사용하여 어떻게 bbox 값을 조정할 수 있는 손실값을 전달하는 지 모르겠습니다. 논문에는 RoI pooling 층에서 역전파로 전달이 되는 것으로 서술 되어있는데 실제 코드로 어떻게 구현을 방법을 습득하지 못 했습니다.

▶ 프로젝트 이후 - Faster R-CNN

이전 프로젝트를 통해 수확한 지식과 경험을 가지고 다음 버전인 Faster R-CNN 모델을 구현하는 것이 다음 단계입니다. RoI pooling 을 하지 않고 RPN 합성곱으로 더 정확하게 물체를 탐지합니다.