|  |
| --- |
| **계절에 따른 포획 금지 어종 판별 프로젝트**  **AWS 클라우드 컴퓨팅 실무실습과정 (19.12~20.01)** |

|  |  |  |  |
| --- | --- | --- | --- |
| 팀명 | Sansung | | |
| 작성일 | 2020-01-21 | 버전 | 1.1 |
| 작성자 | 이진 이정은 이현선 백지영 김유정 | | |

**<차례>**

**1. 개요**

**1.1 프로젝트 개요 ----------------------------------3**

**1.2 목표 ----------------------------------3**

**1.3 개발 환경 ----------------------------------3**

**1.4 프로젝트 산출물 ----------------------------------3**

**1.5 조직 구성 ----------------------------------4**

**2. 프로젝트 세부 사항**

**2.1 서버 구축 ----------------------------------4**

**2.2 데이터 전처리 ----------------------------------5**

**2.3 모델 생성 및 학습 ----------------------------------6**

**3. 결론 및 전망**

**1. 개요**

**1.1 프로젝트 개요**

일반적으로 대상 어종에 섞여서 함께 어획되는 비대상 어종을 혼획(by-catch)이라 일컫는다. 정부, 업계, 학계 관련자들은 어획 과정에서 그물에 목적 어종 외 잡어가 들어올 수밖에 없기에 자연 혼획은 불가피하다는 것에 대체로 공감한다. 미래 먹거리인 바다 자원을 지키고 지속 가능한 이용 방안을 생각해보는 관점에서 보았을 때, 불가피한 혼획을 막는 것은 수산업의 최대 과제라고 볼 수 있다.   
그런데 어부들이 사진 또는 영상으로 생선을 찍어 올렸을 때 해당 생선이 혼획된 생선인 지를 분류해 주는 프로젝트를 구현하면 불가피한 혼획을 막을 수 있다는 생각에서 해당 프로젝트를 기획하게 되었다.

**1.2 목표**

해양수산부 기간별 금지어종을 기준으로 하여 사진을 모델에 입력하면 어느 어종인 지 분류해주는 모델을 학습하려고 하였다.

**1.3 개발 환경**

Python 3.6.9

YOLOv3

Jupyter Notebook

Labelbox

aws ec2

**1.4 프로젝트 산출물**

프로젝트가 끝나면 다음과 같은 결과를 산출한다.

* 물고기 6종류(문치가자미, 돌돔, 명태, 연어, 참홍어, 황복)의 각 100개씩 총 600개의 jpg 파일 이미지
* 물고기 6종류의 각 100개씩 총 600개의 YOLO 형식의 (class id, x1, y1, x2, y2) txt 파일
* /darknet/cfg/coco.data : 클래스 개수, train과 validation 폴더의 경로, voc.names, 백업 파일의 경로가 담긴 파일
* /darknet/data/voc.name: \n으로 분리된 물고기 6종류 이름
* /darknet/cfg/yolov3.cfg: 네트워크 구조가 담긴 파일(하이퍼 파라미터 조정 가능)
* /darknet/darknet53.conv.74: 미리 train 된 가중치가 담긴 파일
* /darknet/output : 테스트의 결과가 담긴 파일

**1.5 조직 구성**

- 팀장: 이정은

- 팀원: 김유정, 백지영, 이진, 이현선

- 조직 계획: 분산형 팀 구성

**2. 프로젝트 세부 사항**

**2.1 서버 구축**

Deep Learning을 하기 위하여 AWS의 EC2를 이용하여 서버를 구축하였다. Deep Learning Base AMI (Ubuntu 18.04, Version 21.0) AMI를 사용하였다. 또한, GPU를 사용하기 위하여 p2.xlarge 인스턴스 유형을 사용하여 이미지 분석 및 모델 학습의 효율성을 높였다.

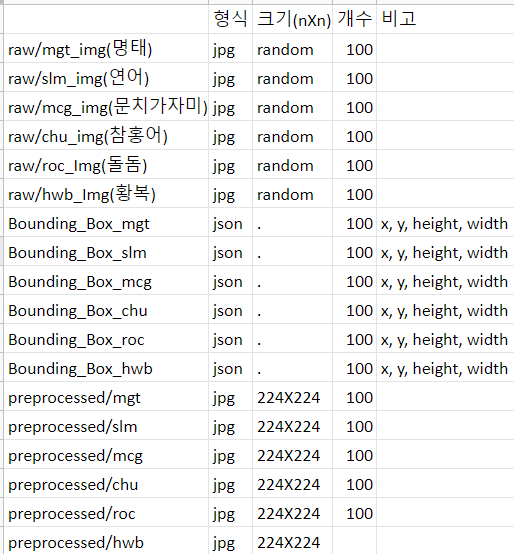
탄력적 IP 를 인스턴스와 연결시켜 서버로의 ssh접근이 용이하도록 환경을 구축하였다.

또한 코드와 파일 등을 편집 및 수정하기 위하여 서버에 jupyter notebook을 연결하였다.

다양한 환경에서 test 및 실행하기 위하여 다음과 같이 가상환경을 구축하였다.   
virtualenv 명령어로 python2를 사용하는 p2 가상환경을, python3를 사용하는 p3 가상환경을 생성하였고, ipykernel을 통해 jupyter notebook에 각각의 환경에 맞는 kernel을 구축하였다. 모델이 작동되는지 시험하는 과정에서 해당환경에 따라 kernel을 설정하여 작업할 수 있어, 환경설정 등이 겹쳐 발생하는 문제들을 해결하였다. 구축한 가상환경에서 모델에 맞게 package를 설치하고, version 등의 환경을 구축하였다.

**2.2 데이터 전처리**

본 프로젝트에 쓰인 어종은 총 6종류로 명태, 연어, 문치가자미, 참홍어, 돌돔, 황복이다. 각각 mgt, slm, mcg, chu, roc, hwb라는 코드를 부여하여 작업하였다. (아래 <표 1> 참고)

<표 1>

각 어종마다 100장, 총 600장의 이미지를 모아 해당 데이터를 기반으로 프로젝트를 구현하였다.

이미지 이름을 일관적으로 변경하는 데 사용한 파일은 rename.py이다. 이미지들을 각 폴더에 분류하여 넣어두고, 상위폴더에서 각각의 폴더에 접근하여 이름을 변경하게 하였다.

이미지의 크기를 조정하는데 사용한 파일은 resize.py이다. 모든 이미지를 224 \* 224의 크기로 만들었다.

경계상자는 웹페이지 기반의 Labelbox 서비스를 이용하여 설정해 주었다. 이에 대한 결과는 .json 파일로 받았고, makeYoloData.ipynb을 이용해 YOLO에 적합한 .txt 파일로 가공하였다.

가공한 .txt 파일과 이미지 데이터들을 aws 인스턴스에 올리고 각 데이터들이 해당하는 경로에 위치할 수 있게 구현한 프로그램은 mv\_files.ipynb, mv\_jsons.ipynb, mv\_jsons2.ipynb, mk\_txts.ipynb 이다.

**2.3 모델 생성 및 학습**

2.3.1. Training

YOLO v3는 negative mining이나 기타 다른 방법을 전혀 사용하지 않고, Full Images를 사용하게 된다. 다만 multi-scale training, 많은 data의 augmentation, batch normalization 외의 기타 여러가지 방법들을 사용한다. 실험 환경은 OpenSource인 Darknet Framework에서 수행하였다.

2.3.2. Workflow

yolo의 공식 오픈소스 사이트(<https://pjreddie.com/darknet/yolo/>)에서 실행 과정과 예제 파일을 다운로드 받아 연습을 하였고, 우리 데이터로 훈련하기 위해 예제 코드를 custom 하는 부분은 <https://eehoeskrap.tistory.com/370?category=705416> 을 참고하여 하이퍼 파라미터와 경로를 조정하였다.

2.3.3. Things We Tried But Didn't Work

기대와 달리 작동하지 않았던 부분에 대해 말하려고 한다.   
첫 째, 훈련 시간이 너무 길어 Workshop 기간 내에 훈련을 끝마치지 못했다. 검색 해 본 결과 평균 총 훈련 기간이 3일에서 7일 사이라고 하여 workshop 기간 내에 끝마치는 것이 어려웠다.  
둘 째, Output Data를 보기가 어려웠다. Output Data를 저장하려면 train할 명령어가 들어있는 sh 파일이 필요했다. 권한을 바꾸는 명령을 실행했음에도 불구하고 sh 실행 권한 오류가 반복적으로 발생했기 때문에 결국 Output Data를 다룰 수 없었다.  
다음으로는 라이브러리의 버전을 맞추는 것, 데이터 세트 구하는 것이 어려워 여러 모델을 시도했다가 실패하여 workshop 기간 내에 모델을 택하여 해당 모델의 학습을 마치기가 어려웠다.

**3. 결론 및 전망**

본 프로젝트에서 모델을 학습하고 가중치를 저장하였지만, 앞으로 본 프로젝트를 발전시키기 위해서 모델에 test data를 주어 해당 모델의 성능을 실험하고 가중치 값들에 변화를 주어 모델의 성능을 더 좋게 만들 계획이다.