

# 약용작물 흰 곰팡이 이미지 탐지 프로그램

2조 김진, 박시우, 조경림, 최한결

# OUR TEAM



**김진**

데이터 전처리, 어노테이션,  
모델링, 실행파일 작성

Git : <https://github.com/>

E-mail : camuscheer33@gmail.com



**박시우**

데이터 전처리, 어노테이션,  
모델링, 가이드 영상 제작

Git : <https://github.com/PerfectTruth>

E-mail : truestar0807@gmail.com



**조경림**

팀장, 데이터 전처리,  
어노테이션, 모델링, 실행파일

설명서 작성, 발표

Git : <https://github.com/00ong>

E-mail : philosophic.code@gmail.com



**최한결**

데이터 전처리, 모델링

Git : <https://github.com/Choihankyul>

E-mail : chk765@naver.com

# 목차

01

## 프로젝트 개요

- 1) 프로젝트 목적
- 2) 분석 환경 및 도구

02

## 제작 과정

- 1) 데이터 전처리
- 2) 어노테이션

03

## 학습 및 평가

- 1) 모델 학습(YOLO v5)
- 2) 평가 결과
- 3) 사용방법
- 4) 프로젝트 활용 방안

# 프로그램 개요

# 01

1) 프로그램 목적

2) 분석 환경 및 개요

## 프로그램 목적



인공적인 실험실 환경에서

자라는 약용 작물의

**CCTV** 이미지를 확보 후,

확보된 이미지에서

흰 곰팡이 탐지

# 프로그램 개요



mold 0.7



# 어노테이션 (ANNOTATION)

AI 모델을 통해  
객체탐지(Object Detection)를 하기 위해  
학습용 이미지에  
바운딩 박스(Bounding Box) 처리,  
탐지하고자 하는 이미지에 레이블을 다는 작업

## 어노테이션 과정

- 사용 툴 : ROBOFLOW
- 어노테이션 데이터셋 생성 과정 :
  - 1) 학습 및 테스트 용 데이터 업로드
  - 2) 학습 비율 설정
  - 3) 바운딩 박스 처리
  - 4) 결과로 옵션 설정
  - 5) 어노테이션 데이터 셋 생성



어노테이션  
클래스  
(nc, Number of Classes)



Mold

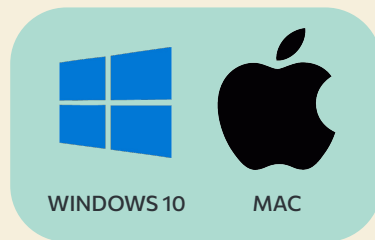
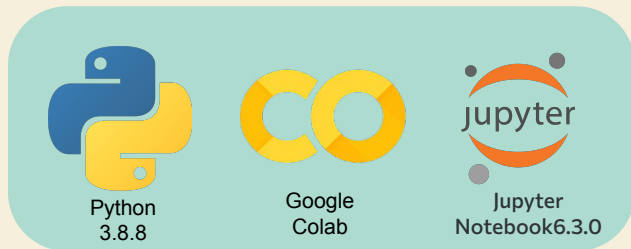
흑과 명확히 구분되는 곰팡이



Blur

비교적 선명하지 않은  
곰팡이

# 개발 및 환경도구



사용 모델

모델 명
YOLOv5
Faster RCNN
EfficientNet

사용 라이브러리

라이브러리 명	버전	라이브러리 명	버전	라이브러리 명	버전
Pytorch	-	scipy	1.4.1	PyYAML	5.3.1
tensorboard	2.4.1	matplotlib	3.2.2	seaborn	0.11.0
tensorflow	2.4.1	Pillow	7.1.2		
torchvision	-	open cv	4.1.2		

# TOOLS



어노테이션 툴  
ROBOFLOW



개발 언어  
Python 3.8.8



사용 모델  
YOLO v5



## YOLO v5

(You Only Look Once)

가장 빠른 객체 탐지 모델 중 하나.  
Input 이미지를  $S \times S$ 의 그리드로 나눈 후,  
각각의 그리드 셀들은 바운딩 박스를 통해 신뢰도  
(Confidence score)를 예측한다.

\* 신뢰도 점수 = 실제 물체의 존재 여부와 박스( $S \times S$  grid)가 예측하는 박스의 정확성을 측정하는 지표

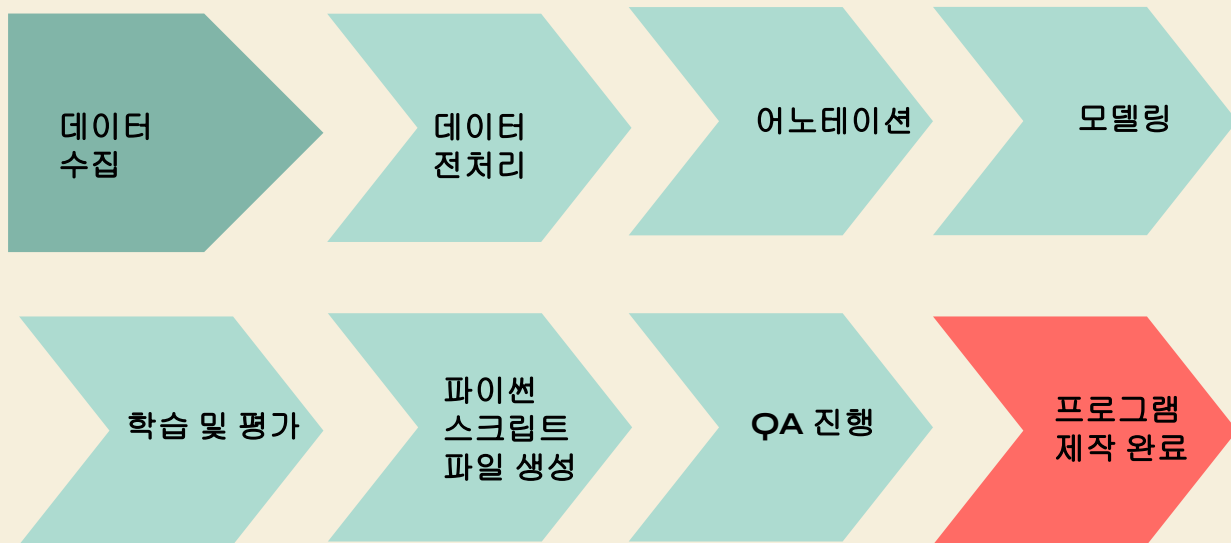
# 제작 과정

# 02

1) 데이터 전처리

2) 어노테이션

## 전체 제작 흐름



## 데이터 전처리 - 원본 데이터 셋 상세 내역



<사용 데이터1>



<사용 데이터2>

- 데이터 셋 수집 기간 : 2021년 9월 17일 ~ 2021년 10월 5일 (총 19일 간)
- 수집 형식 : CCTV 수집된 약용작물 이미지
- 총 이미지 파일 개수 : 15,761개

## 데이터 전처리 - 원본 데이터 셋 상세 내역

- 파일 형식 : jpg
- 이미지 정보 : 640 \* 480 px

장소	카메라 각도	촬영 시작일 (년-월-일-시-분-초)	촬영 종료일 (년-월-일-시-분-초)	파일 개수
A	정면	2021-09-16 18_28_33	2021-10-05 09_00_00	4,055
	측면	2021-09-16 20_58_20	2021-10-05 09_00_02	3,910
B	정면	2021-09-17 09_07_22	2021-10-05 09_10_04	3,890
	측면	2021-09-17 09_07_35	2021-10-05 09_10_04	3,906



## 데이터 전처리 - 제외 데이터



〈제외 데이터1〉



〈제외 데이터2〉



〈제외 데이터3〉

- 모델 학습률 향상을 위해, 흰 곰팡이가 명확하게 보이는 ‘자연광 환경’에서 촬영된 이미지 만 사용
- 모델 학습 시 제외 사진 :
  - 붉은 LED 환경에서 촬영된 사진
  - 어두운 환경에서 촬영된 사진
  - 흰 곰팡이가 없는 정상 밀순 사진
- 최종 학습 이미지 : 123장

# 어노테이션 - 원본 데이터의 한계점 및 극복 방안

## 원본 데이터의 한계점

- 학습에 사용 가능한 총 이미지 개수 123개 학습용으로 비교적 적음
- 640\*480 px, 0.1Mb가 되지 않는 CCTV 화질의 이미지
- 어노테이션 1개만 설정 후 모델링 진행 시 mAP 0.65 이상 향상되지 않는 이슈 발생
- 화질 개선, 이미지 증식 할 경우 mAP가 0.4이하로 하락하는 현상 발생

# 어노테이션 - 원본 데이터의 한계점 및 극복 시도

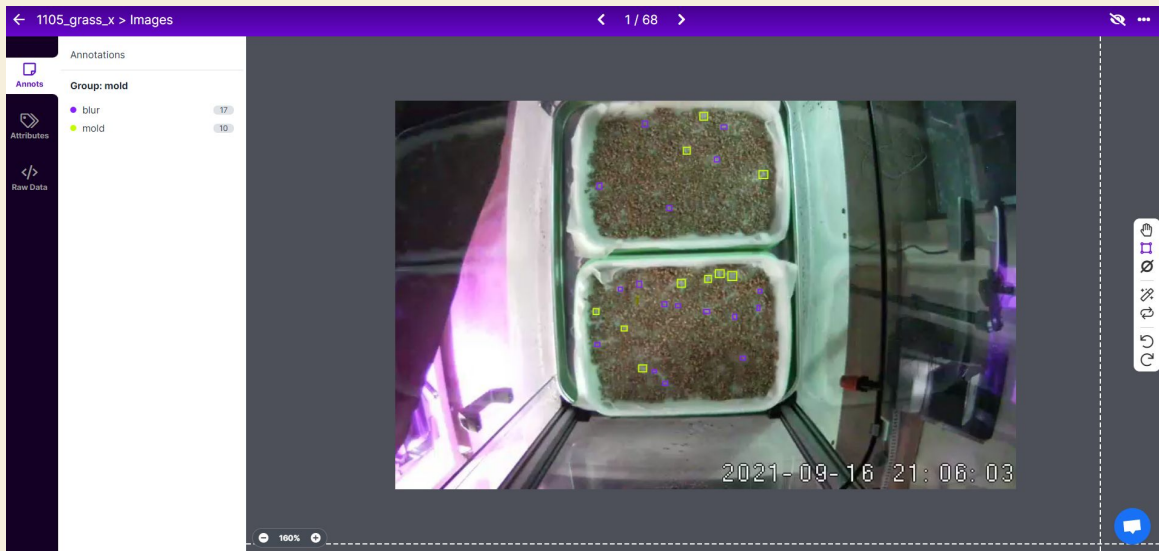
## 극복 방안

- AI가 이미지를 학습할 때 곰팡이와 흙을 더 명확하게 분리 할 수 있는 방안 구상
  - 1) 데이터 셋 선별 : 자연광이 충분하게 들고, 명확하게 곰팡이가 보이는 사진 채택
  - 2) 바운딩 박스 : 곰팡이가 흙과 명확하게 구분되는 부분만 좁게 바운딩박스 치
  - 3) 어노테이션 설정 : 명확하게 나온 곰팡이와 그렇지 않은 곰팡이에 각각 다른 어노테이션 명을 부여

## 어노테이션 명

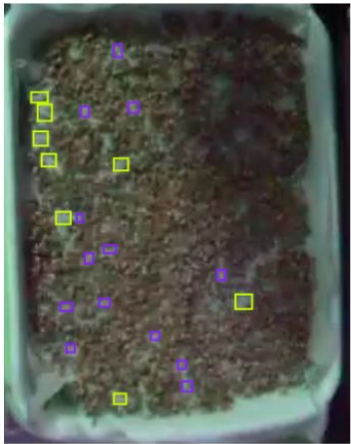
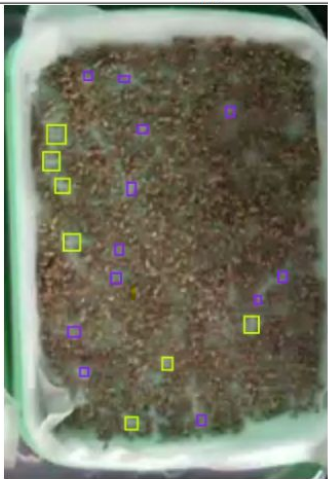
기준	어노테이션 명
흙과 명확하게 구분되는 선명한 곰팡이	mold
비교적 선명하지 않은 곰팡이	blur

## 어노테이션 - 바운딩 박스



ROBOFLOW 라는 어노테이션 툴을 사용, 바운딩 박스 작업 진행

## 데이터 전처리 - 어노테이션 진행

항목	예1	예2
사진		
	노란색 바운딩 박스 : mold	
	보라색 바운딩 박스 : blur	

- 바운딩 박스에 노란 박스는 mold, 보라색 박스는 blur로 라벨링 명 지정
- 총 6차에 걸친 어노테이션 작업 진행

# 어노테이션 - 학습 비율 설정 및 모델 출력

- 총 학습 이미지 123개



어노테이션



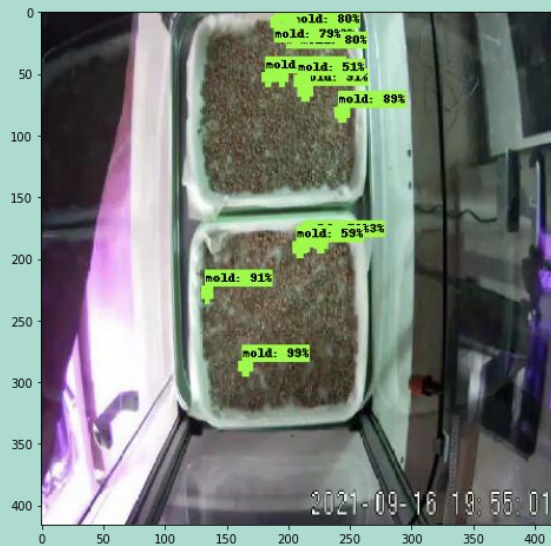
YOLO v5 PyTorch TXT 파일로 포맷

# 학습 및 평가

# 03

- 1) 모델 학습
- 2) 평가 결과
- 3) 사용 방법
- 4) 프로젝트 가치

## 모델 학습 - Faster R-CNN



- Faster R-CNN 사용 결과, 최대 mAP 99% 도출
- 그러나 1회 모델링 실행 속도가 느려 시의성이 중요한 곰팡이 탐지에 맞지 않다고 판단
- 프로그램의 이용성 고려 시 타 모델을 탐색



# 모델 학습 - EfficientNet

```
model.compile(loss='categorical_crossentropy',
              optimizer=optimizers.RMSprop(lr=2e-5),
              metrics=['acc'])

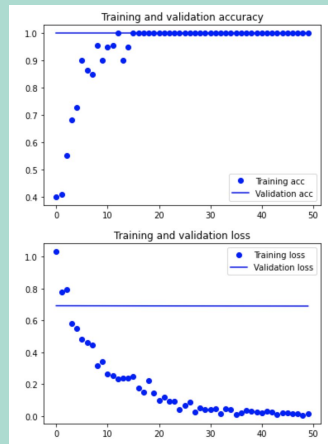
history = model.fit_generator(
    train_generator,
    steps_per_epoch= NUM_TRAIN //batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps= NUM_TEST //batch_size,
    verbose=1,
    use_multiprocessing=True,
    workers=4)
```

Epoch 50/50

5/6 [=====] - ETA: 0s - loss: 0.0144 - acc: 1.0000Epoch 1/50

6/6 [=====] - 1s 148ms/step - loss: 0.6910 - acc: 1.0000

6/6 [=====] - 2s 292ms/step - loss: 0.0158 - acc: 1.0000 - val\_loss: 0.6910 - val\_acc: 1.0000



- EfficientNet 사용 결과, acc 1.0000 으로 과소적합 문제 발생
- 보유한 데이터 셋의 정확도 탐지에 알맞지 않은 모델로 판단
- 보다 알맞은 정확도를 도출 할 수 있는 다른 모델을 탐색

## 모델학습 - YOLO v5

어노테이션 작업을 완료한  
학습용 데이터셋 **YOLO v5**에 학습



## 모델생성 - YOLO v5

```
img_size = 600
batch_size = 32
epochs = 300
weights_size = 'l' # s,m,l,x
name = 'yolov5'
name_fix = name+weights_size
data_path = '/content/drive/MyDrive/2차프로젝트/cakd3_2차프로젝트_2조/datasets/1105/han/data.yaml'
yaml_path = '/content/yolov5/models/{}.yaml'
weights_path = '/content/yolov5/{}.pt'

name = 'mold_{}_results'


# train
!python3 /content/yolov5/train.py --patience 0 --img {img_size} --batch {batch_size} \
--epochs {epochs} --data {data_path} --cfg {yaml_path.format(name_fix)} \
--weights {weights_path.format(name_fix)} --name {name.format(name_fix)} \
```

Img	batch	epochs	data	weights
input image size	batch size	epochs	yaml path	가중치

# 평가지표

## IoU(Intersection over Union)

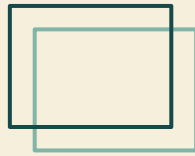
알고리즘이 설정한 바운딩 박스와  
사용자가 설정한 바운딩 박스의 중첩된  
면적을 합집합의 면적으로 나눠준 비율로  
0.5 이상이면 제대로 검출됐다고 판단한다.

$$\text{IoU} = \frac{\text{겹쳐진 영역}}{\text{합쳐진 영역 총합}}$$


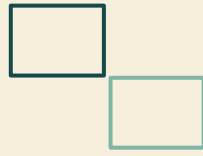
## Sample IoU Scores



0.9



0.5



0.0

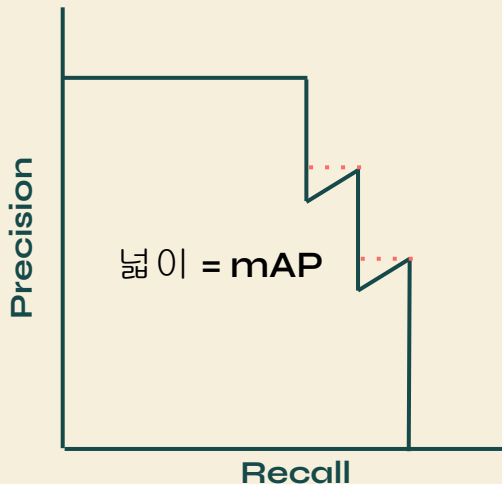
## mAP(mean average precision)

합성곱 신경망(CNN)의 모델 성능 평가를 위해 사용된다. 이용한 Precision-Recall 그래프를 통해 mAP를 도출한다. 1에 가까울수록 좋다.

\* Precision =  $TP / (TP + FP)$ , Recall =  $TP / (TP + FN)$

실제 상황 (Ground Truth)	예측 결과 (Predict Result)	
	Negative	Positive
Negative	TN(True Negative)	FP(False Positive)
Positive	FN(False Negative)	TP(True Positive)

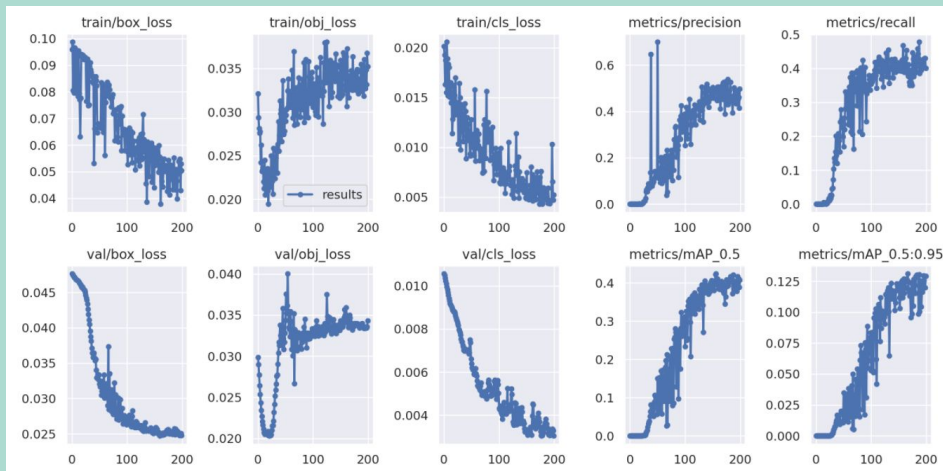
PR Curve



## 모델 검증 결과(1)

Model Summary: 367 layers, 46113663 parameters, 0 gradients, 107.8 GFLOPs

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95	100% 1/1	[00:01<00:00, 1.08s/it]
all	23	439	0.448	0.476	0.421	0.13		
blur	23	194	0.191	0.263	0.119	0.0295		
mold	23	245	0.706	0.69	0.723	0.23		



Mold 탐지 mAP 정확도 : 0.723

## 모델 검증 결과(2)



1) Validation 결과 사진 출력 가능

2) Best.pt 파일 생성 : 총 123개  
이미지를 통해 곰팡이를 학습한 모델

best.pt	2021-11-07 오전 3:16	PT 파일	90,621KB
---------	--------------------	-------	----------

## 모델 Test 결과



best.pt 를 활용해 test 이미지에 적용 결과



## 사용방법 - .py 실행 파일

① 압축 파일을 풀고 파이썬으로 detect.py를 실행시킨다

② yolov5/images 디렉토리에 탐지할 원본 이미지를 넣고 다음과 같이 실행시키면 현재 폴더에 result 디렉토리가 생성되고 그 안에 곰팡이가 탐지된 이미지가 생성된다.

--source 옵션은 이미지 파일이 있는 디렉토리를 입력한다.

--project 옵션은 결과 파일이 저장될 디렉토리를 입력한다.

--conf 옵션은 곰팡이 탐지 신뢰도의 threshold를 의미한다. 기본값은 0.25이다.

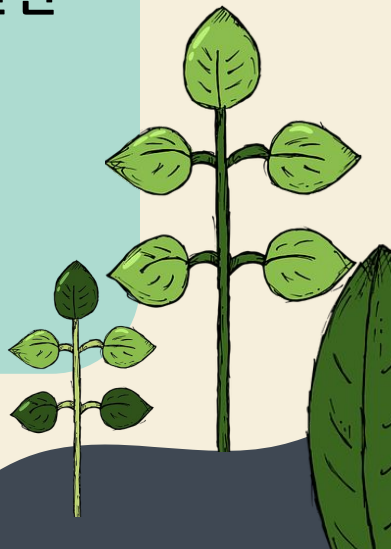
```
Python yolov5/detect.py --source ./yolov5/images --project ./ --conf 0.5
```

## 사용방법 - .py 실행 파일



## 프로그램 활용 방안

1. 약용작물 흰곰팡이 이미지 탐지 프로그램 제공
2. 사용자가 약용작물 데이터 입력 시 프로그램을 통해 관련 종사자들이 작물에 대한 흰 곰팡이 진단 및 예방 가능
3. 곰팡이 방지 및 탐지를 통해 작물의 생산량 증대 예상



# 저작권 등록

## 프로그램등록신청서

※ □ 에는 V표를 합니다.

(앞쪽)

접수번호	접수일자	처리기간
		4일
프로그램 저작물	① 제호(명칭) ② 창작연월일	흰 곰팡이 이미지 탐지 프로그램 2021년 10월 18일 (한글) 허진경 외 4 인
		※ 외국어의 경우 한글을 함께 기재합니다. 멘토 선생님고 팀원 4명 2021년 11월 8일 국적 별지기재

## 접수증

신청인 : 허진경 님이 신청한 접수내역입니다.

접수일자 : 2021년 11월 10일

접수번호	신청내용	신청제호
2021-051628	프로그램 등록	흰 곰팡이 이미지 탐지 프로그램

2021년 11월 10일

한국저작권위원회 위원장



## 프로그램등록신청명세서

프로그램종류코드 : 4 2 8 7 0

1. 제호	흰 곰팡이 이미지 탐지 프로그램
적용분야	농업분야
특징	<ul style="list-style-type: none"> <li>- 인공지능 실험실 환경에서 자라는 약용 작물의 CCTV 이미지를 확보 후, 확보된 이미지에서 흰 곰팡이를 탐지한다.</li> <li>- 구동 환경 및 제작 용</li> <li>- 구동 환경 : Windows, Mac</li> <li>- 제작용 : Colaboratory, Jupyter Notebook, Roboflow, Python, YOLO v5</li> <li>- mAP(정확도) : 곰팡이 탐지를 위해 견차리로 'mold'라는 label에 대해 어노테이션(annotation) 작업을 진행함. 어노테이션 후 YOLO v5를 이용해 IoU가 0.5이상인 것을 대상으로 mAP를 평가했을 때 0.723의 결과를 얻을 수 있었다.</li> <li>- 알고리즘이 설정한 바운딩 박스와 사용자의 설정한 바운딩 박스의 중첩된 면적을 합집합의 면적으로 나누어 비율로 0.5 이상이면 제대로 분류된다고 판단한다.</li> <li>- mAP : 합성곱 신경망(CNN)의 모델 성능 평가를 위해 사용된다. 이용한 Precision-Recall 그래프를 통해 mAP를 도출한다. 1에 가까울수록 객체 탐지에 대한 예측력이 좋다.</li> </ul>
2. 주요 내용	<ul style="list-style-type: none"> <li>- 곰팡이 탐지 : 640*480 pixel 사이즈에 0.1Mb가 되지 않는 CCTV 사진을 통해 약용작물에</li> </ul>

감사합니다!