

# 여름제선 프로젝트

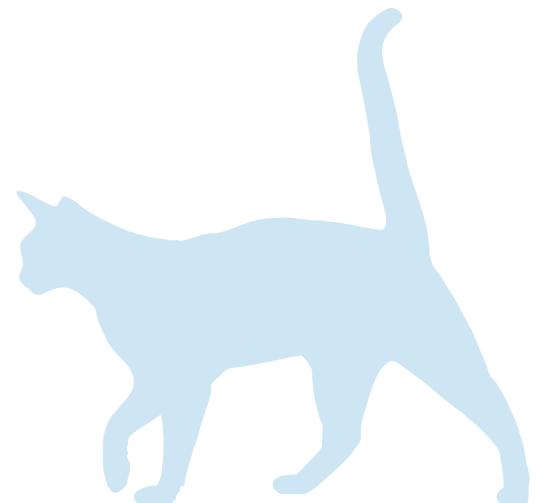
## 고양이 감정분석

권민지 | 김영홍 | 김윤영 | 서준형 | 유민서



# 목 차

- 01** 프로젝트 주제 설명
- 02** 데이터셋
- 03** 모델 소개
- 04** 모델 구축 과정
- 05** 프로젝트 한계 및 의의



01

---

**프로젝트 주제 선정 및  
목표 설정**

# 프로젝트 주제 설명



귀여운 반려동물이  
과연 무슨 생각을 하고 있을까?

기존에 존재하는 유머 앱은  
그저 감정을 랜덤으로 출력

딥러닝의 computer vision 기술과 결합하여  
신빙성 있는 방식으로 문제를 풀어가고자 함

02

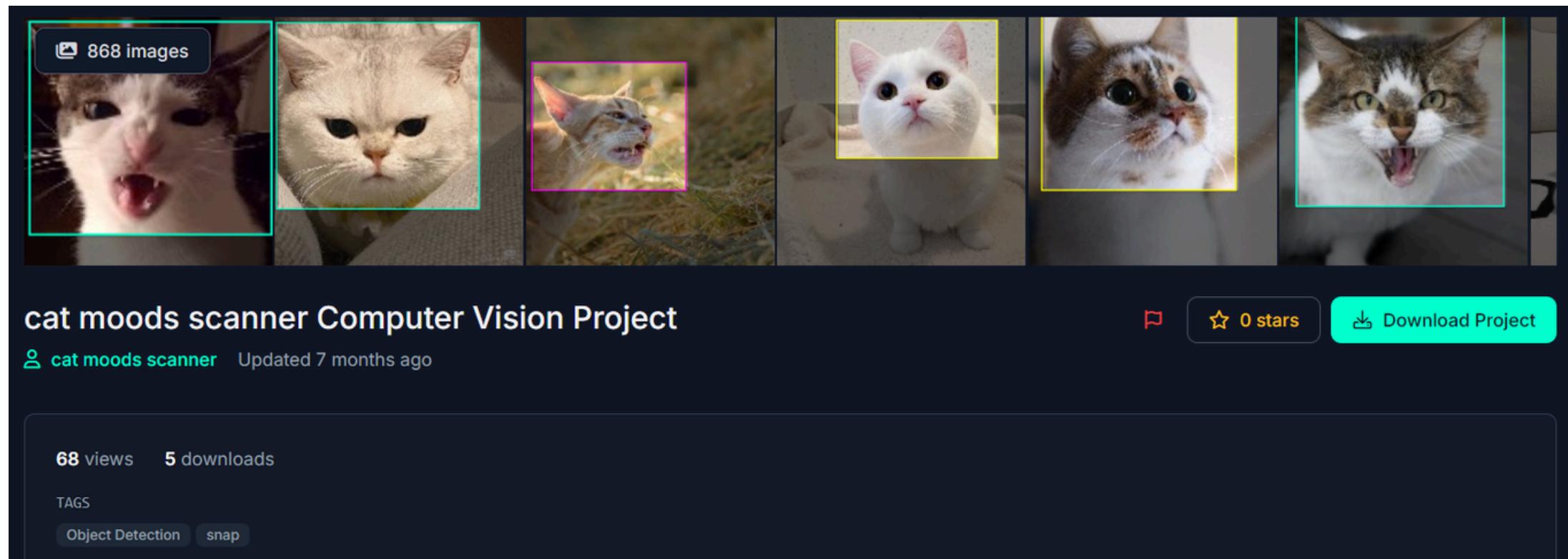
---

비비드



# Yolo 데이터셋

## Roboflow Cat Mood Scanner Dataset



bbox가 존재하는 고양이 이미지 868개, 기존에 감정 라벨링이 있으나 삭제하고 사용  
-> 더욱 정확한 감정분석을 위해 분류기를 따로 연결하기 위해

# keypoint 데이터셋

CatFLW



Cat Emotions



- 고양이 얼굴 이미지 2079개
- 고양이 얼굴의 48개의 주요 랜드마크
- 얼굴에 대한 바운딩 박스 라벨링

- 고양이의 다양한 감정을 분류하는 데이터셋
- 총 671개의 이미지
- 고양이의 감정을 나타내는 라벨 부여

03

---

모델 고침



# 분류 모델

**VGG**

깊고 규칙적인  $3 \times 3$  합성곱 필터를 사용하여 심플한 구조로 이미지를 처리하는 CNN 모델

**Res  
Net**

잔차 연결을 도입하여 매우 깊은 신경망에서도 학습이 가능하도록 만든 모델

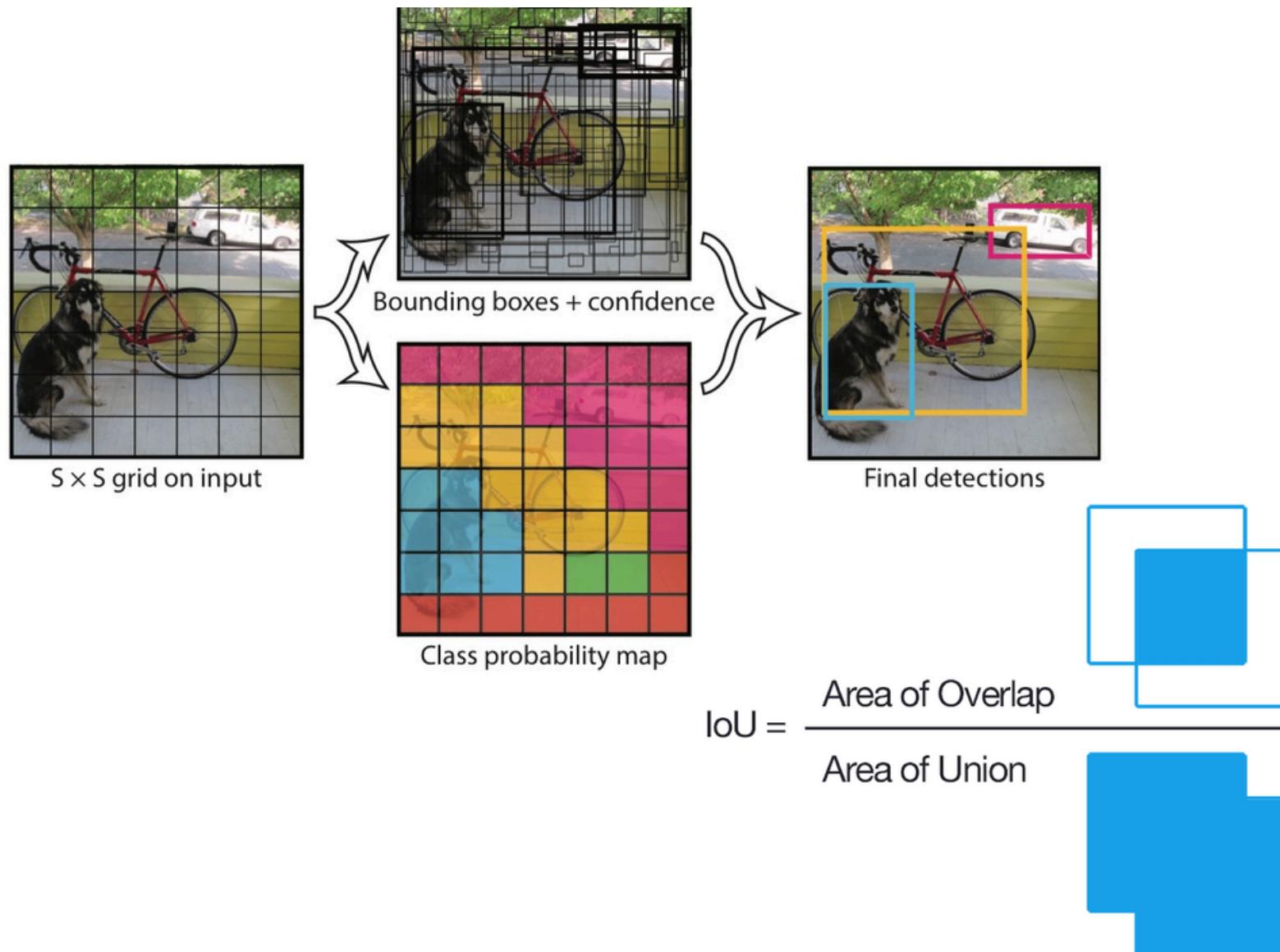
**Dense  
Net**

모든 레이어가 이전 레이어의 출력을 받아들이는 밀집 연결 사용해 특징 전달과 재사용을 극대화한 모델

**Efficient  
Net**

네트워크의 깊이, 너비, 해상도를 균형 있게 확장하여 높은 성능을 달성한 효율적인 CNN 모델

# You Only Look Once: YOLO



**이미지를 보고 한번에 물체의 위치를  
감지할 수 있는 알고리즘**

Step 1: 입력 이미지를  $S \times S$  크기의 cell로 분할

Step 2: cell에서 bbox, confidence score과 클래스 확률 예측

$$\text{Confidence Score} = Pr(\text{Object}) * IOU_{pred}^{truth}$$

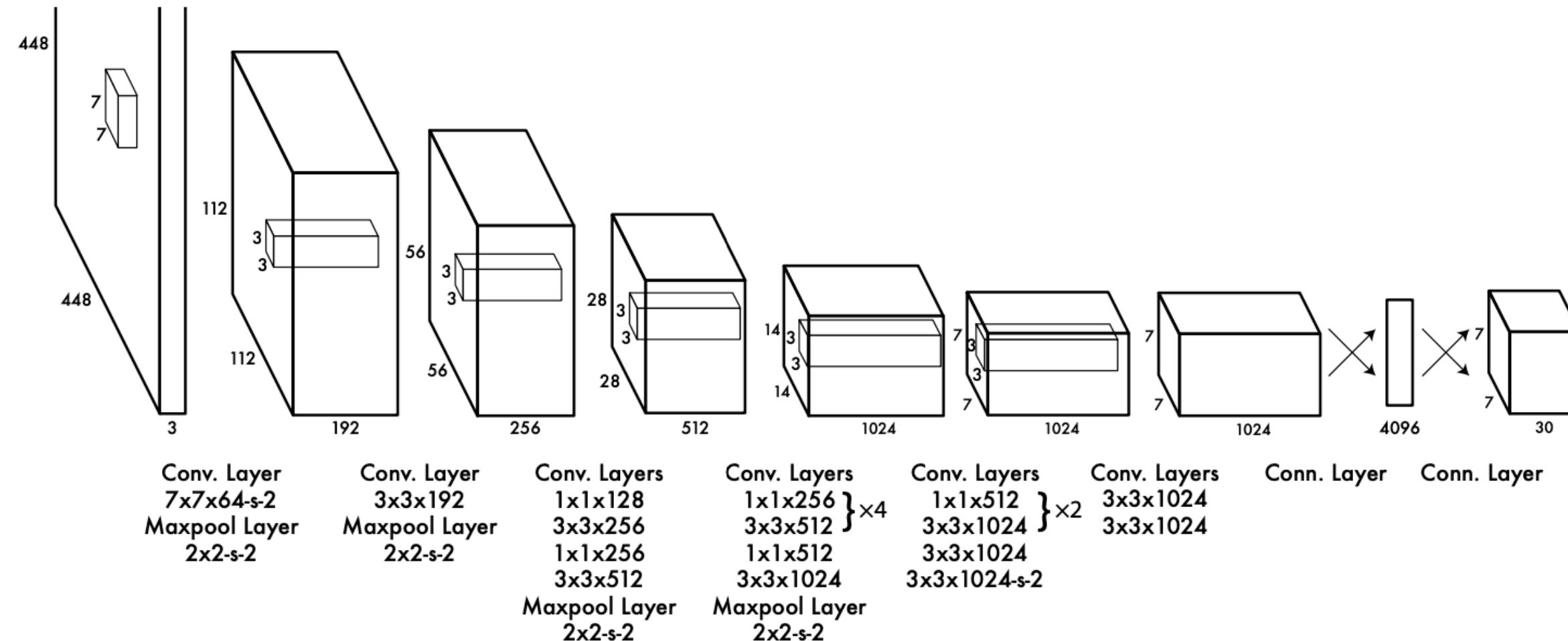
Class Probability =

$$PR(\text{Class}_i | \text{Object}) PR(\text{Object}) IOU = Pr(\text{Class}_i) | IOU$$

# You Only Look Once: YOLO

Learn More!

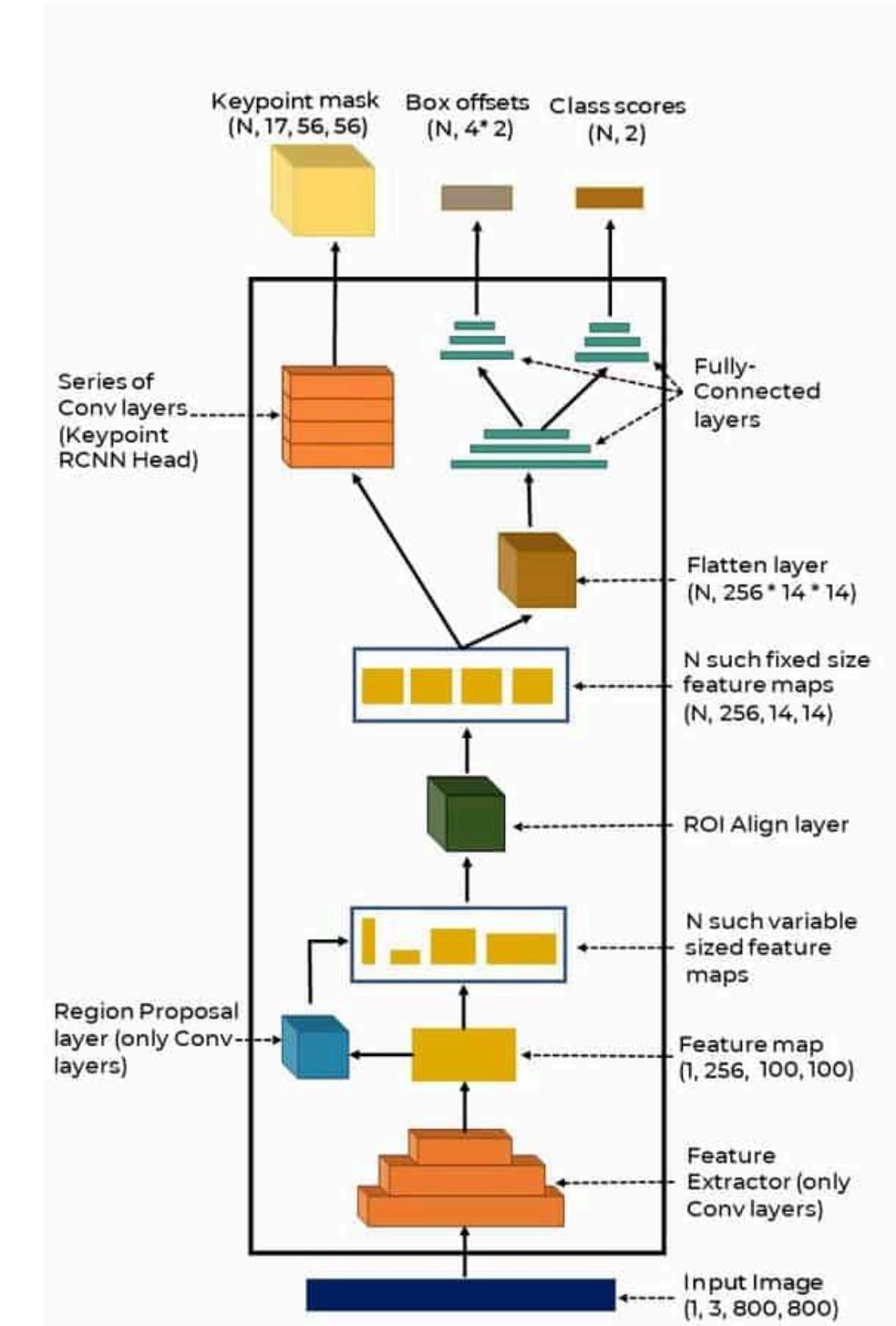
<Model Architecture> YOLO v1



# Keypoint RCNN

## <Keypoint 작동과정>

- Region Proposal Network(RPN) → 객체의 위치 예측
- ROI-Align → 각 객체를 동일한 크기의 feature map으로 변환  
→ 객체의 클래스와 위치를 예측
- Keypoint R-CNN Branch 추가
  - keypoint Head를 통해 각 사람의 신체 부위별로 keypoint를 예측하는 것



KEYPOINT RCNN

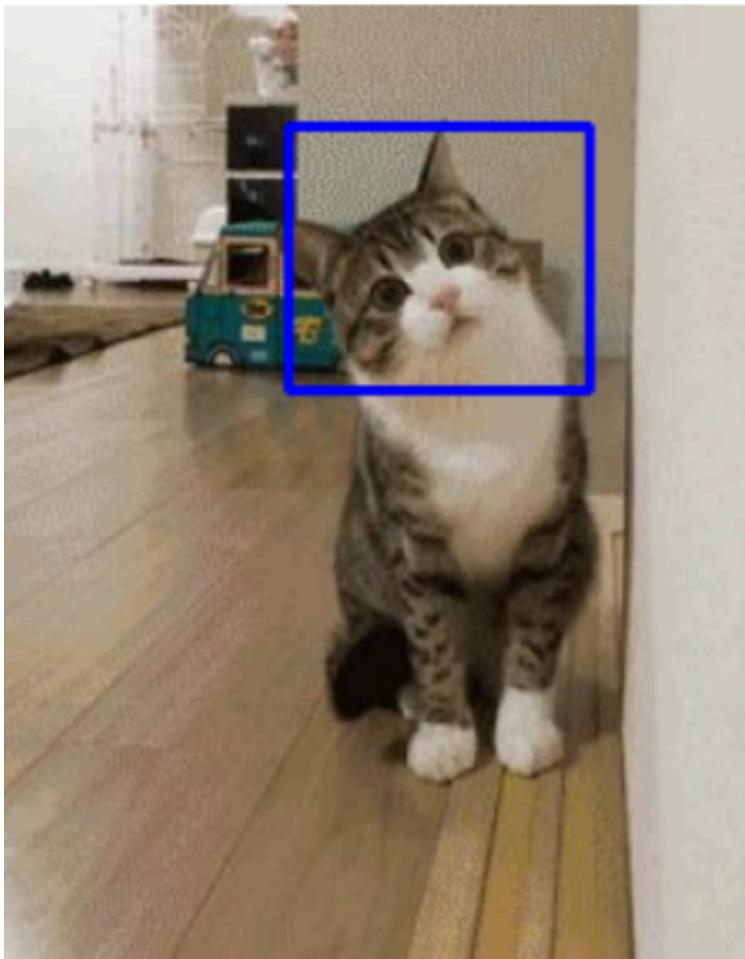
04

---

**모델 구축 과정 1**  
**: YOLO + Efficient Net**

# 고양이의 얼굴 영역 학습

## Step 1: Yolo 모델로 얼굴영역 예측



```
Validating runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.2.78 🚀 Python-3.10.12 torch-2.3.1+cu121 CUDA:0 (NVIDIA L4, 22700MIB)
Model summary (fused): 218 layers, 25,844,392 parameters, 0 gradients, 78.7 GFLOPs
      Class    Images  Instances     Box(P)        R      mAP50    mAP50-95):
        all      249      254    0.261    0.537    0.326    0.198
        Anger     28       28    0.345    0.786    0.675    0.49
        Beg       31       31    0.241    0.613    0.381    0.234
        Frightened   5       5    0.0556    0.8    0.0673    0.0514
        Happy      67      67    0.514    0.19    0.335    0.188
        Scare      10      10    0.0994    0.5    0.314    0.152
        Sick       36      36    0.279    0.667    0.337    0.2
        Sleepy      9       9    0.0617    0.333    0.104    0.0479
        Wonder     63      68    0.489    0.408    0.394    0.223
Speed: 0.2ms preprocess, 6.3ms inference, 0.0ms loss, 1.2ms postprocess per image
Results saved to runs/detect/train
ultralytics.utils.metrics.DetMetrics object with attributes:
```

# 코드 과정 모색

## Step 2: OpenCV로 bbox 영역 crop

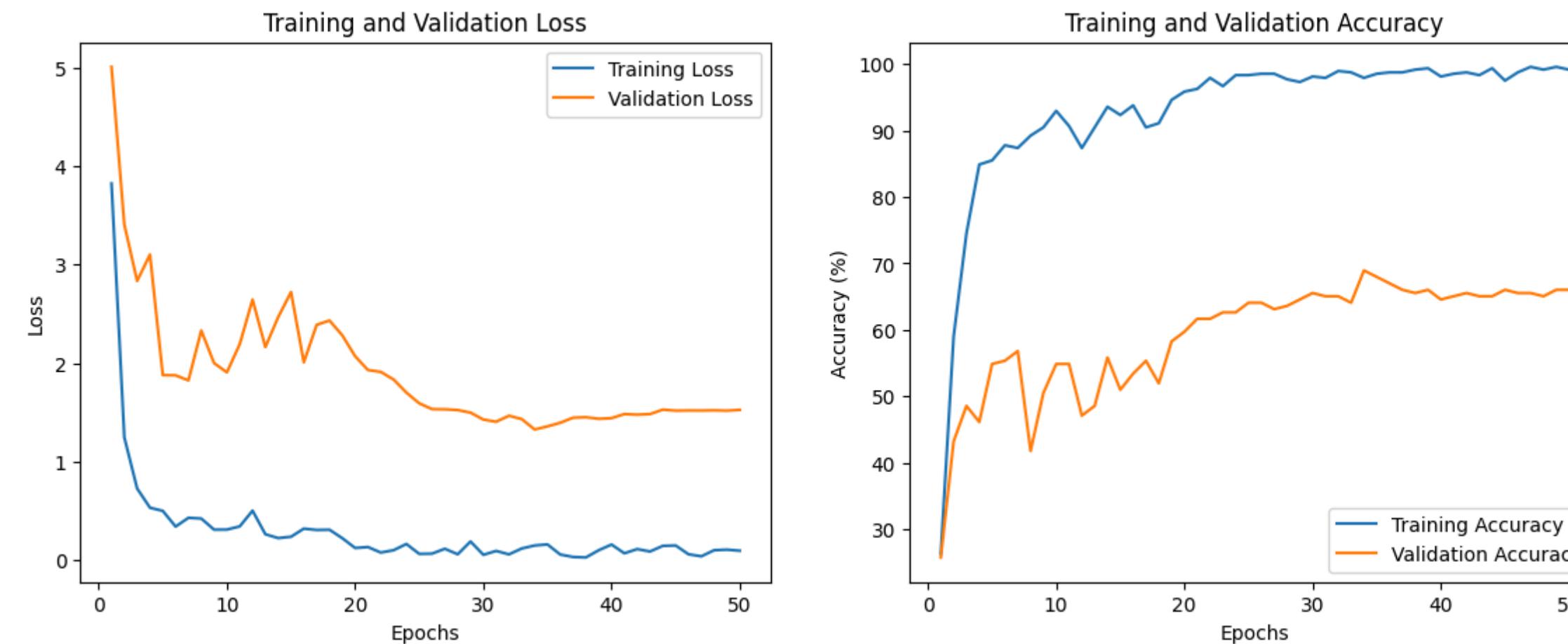


```
1 def cropped_img(input_base_dir, output_base_dir):
2     # 감정 리스트: 직접입력
3     emotions = ['Surprised', 'Scared', 'Sad', 'Normal', 'Happy', 'Disgusted', 'Angry']
4
5     # 감정별 디렉토리 생성
6     for emotion in emotions:
7         os.makedirs(os.path.join(output_base_dir, emotion), exist_ok=True)
8
9     # 각 감정 디렉토리의 이미지를 처리
10    for emotion in emotions:
11        input_dir = os.path.join(input_base_dir, emotion)
12        output_dir = os.path.join(output_base_dir, emotion)
13
14        # 디렉토리 내 모든 파일에 대해 바운딩 박스 예측 및 크롭
15        for img_name in os.listdir(input_dir):
16            img_path = os.path.join(input_dir, img_name)
17            img = Image.open(img_path)
18            results = model(img)
19
20            # 결과에서 바운딩 박스 정보 추출
21            for i, result in enumerate(results):
22                for box in result.bboxes:
23                    # 바운딩 박스 좌표 추출 (정수형으로 변환)
24                    xmin, ymin, xmax, ymax = map(int, box.xyxy[0].tolist())
25
26                    # 바운딩 박스 크롭
27                    cropped_img = img.crop((xmin, ymin, xmax, ymax))
```

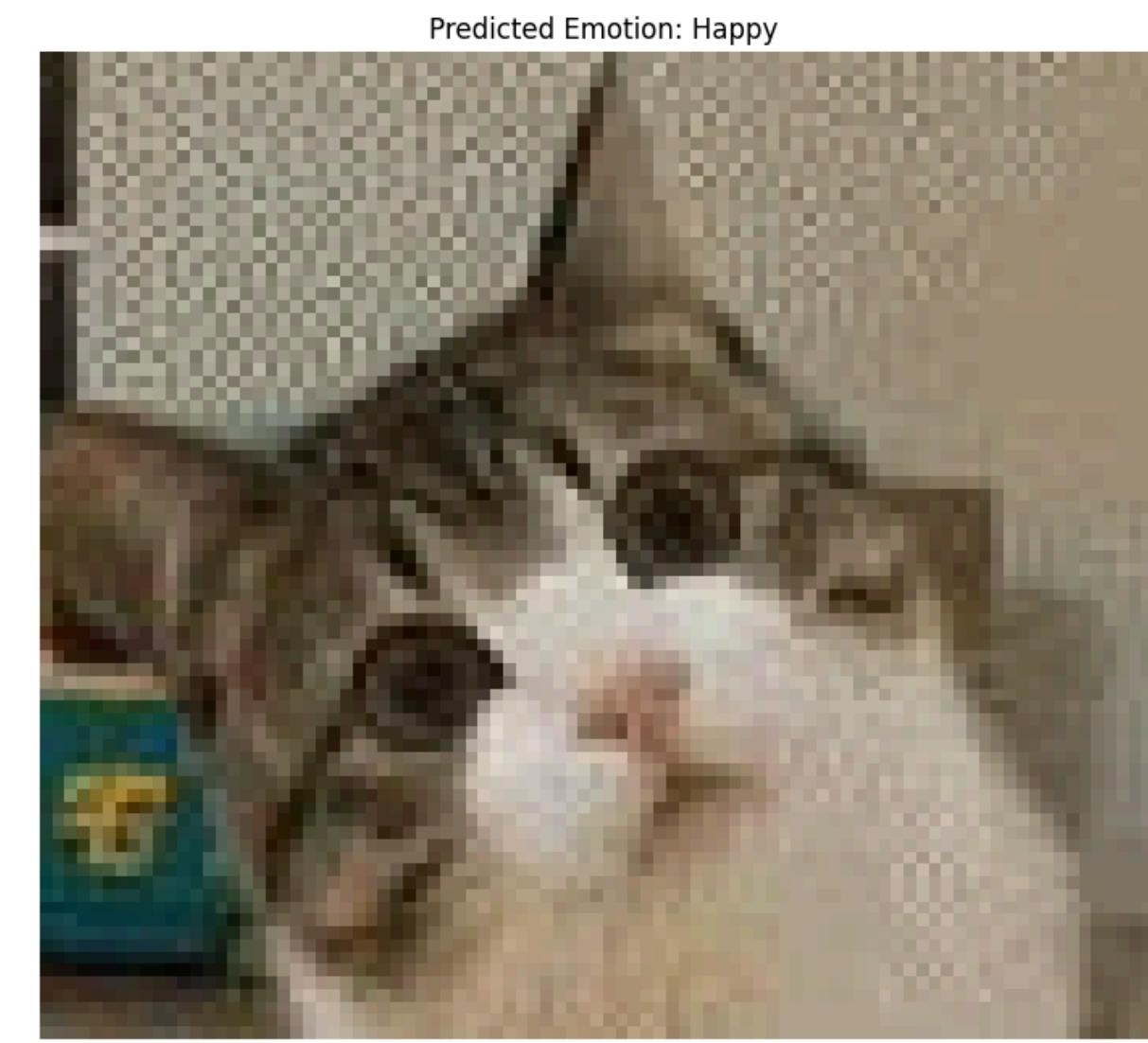
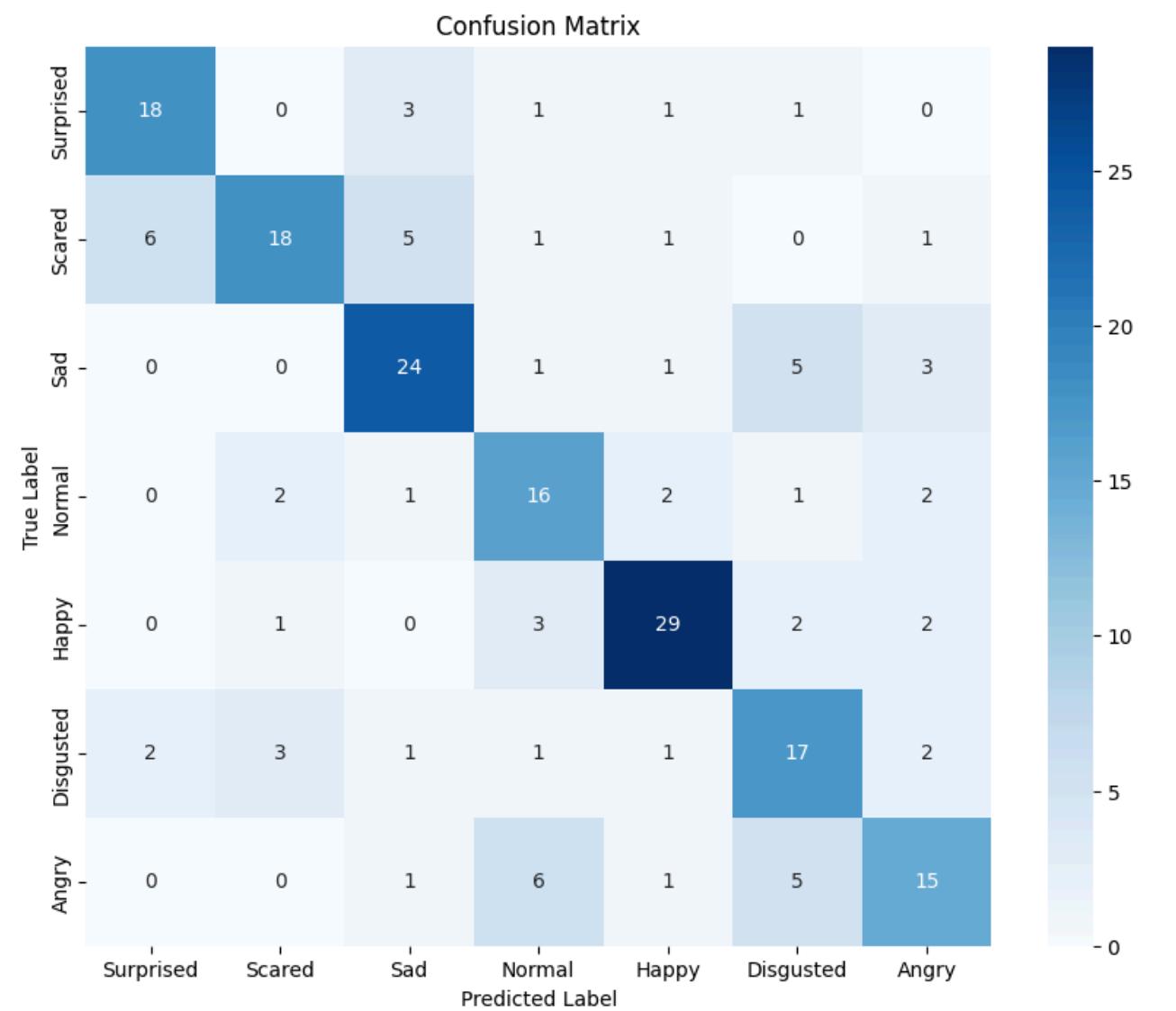
# 고양이 얼굴의 감정 분류

## Step 3: Step2의 이미지로 EfficientNet 분류 학습

Optimizer: Adam (default + lr scheduler) | Loss: Cross Entropy Loss



# yolo+EfficientNet 결과 시각화



04

---

**모델 구축 과정 2**  
**: Keypoint Model**

# 파이프 라인 구성

## DATASET

이미지\_x1

감정\_y1

이미지\_x2

Keypoint\_y2

이미지\_x2

Keypoint 모델

Keypoint\_y2

## STEP2 : Keypoint 라벨 생성

이미지\_x1

keypoint 모델

Keypoint\_y1

## STEP1 : Keypoint 모델 학습

## DATASET 추가

이미지\_x1

감정\_y1

Keypoint\_y1

이미지\_x2

Keypoint\_y2

# 파이프 라인 구성

STEP3 : 감정 분류 multi modal 모델 학습

이미지\_x1

Keypoint\_y1

감정\_y1

keypoint  
& 이미지 감정  
분류 모델

STEP4 : Inference 단계

이미지\_x1

keypoint 모델

이미지\_x1

Keypoint\_y1

keypoint  
& 이미지 감정  
분류 모델

감정\_y1

# keypoint 다양한 pretrain 실험

**PT 1**

Rescale, Normalize, ToTensor 기본 전처리

**PT 2**

기본 전처리 + RandomCrop & RandomHorizontalFlip

**PT 3**

OpenCV의 CascadeClassifier

**PT 4**

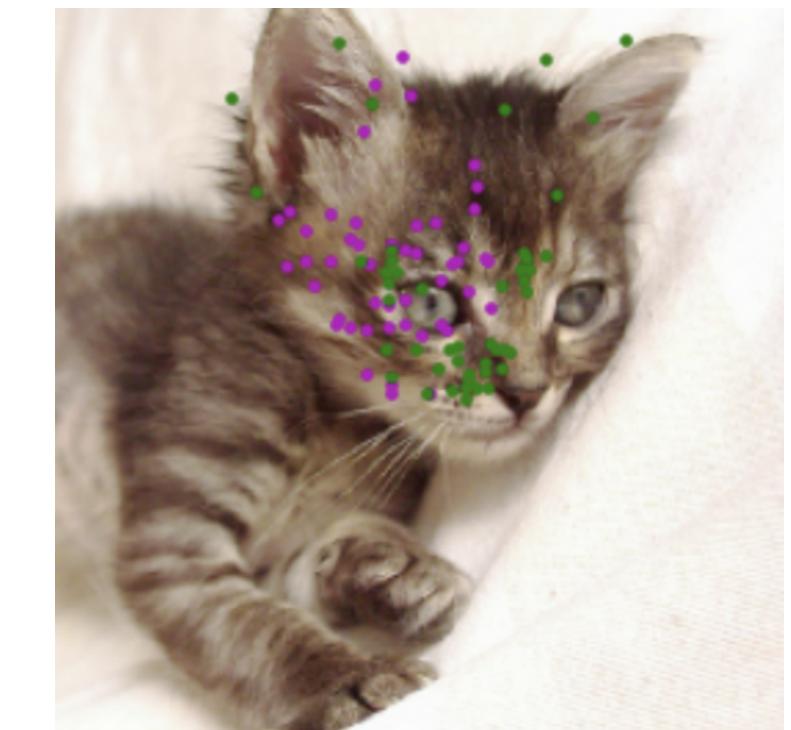
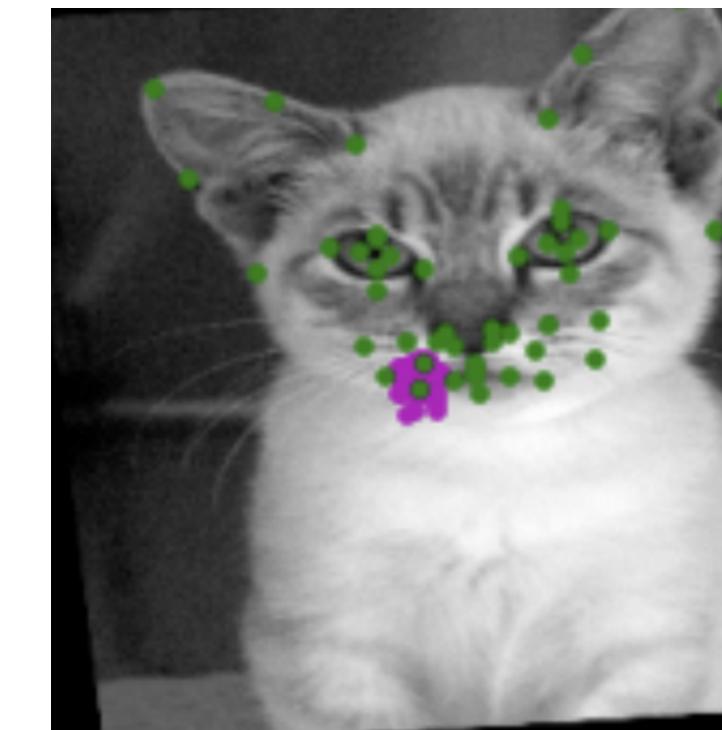
dlib 라이브러리

**PT 5**

CascadeClassifier + 기본 전처리 + RandomCrop &  
RandomHorizontalFlip

PT	LOSS
PT1	0.1265
PT2	0.2585
PT3	0.1362
PT4	0.2763
PT5	0.3142

# pretrain 실험 결과



# Inference 결과



학습 데이터의 양 & 다양성 증가 → 다양한 얼굴 각도와 표정을 잘 반영하도록 함  
데이터 전처리 과정 & 모델의 하이퍼파라미터 조정 → Keypoint 정확도 향상

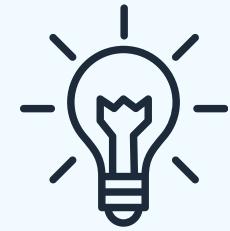


05

---

**프로젝트 한계 및  
의의**

# 프로젝트 한계 및 의의



분류기를 연결했을 때  
에도 accuracy가 80%  
초반에 머물러 모델에  
대한 튜닝이 필요함

분류기 사용 시 데이터  
셋이 많지 않았음. 차후  
시간이 된다면 수기로  
라벨링하여 활용

고양이뿐만 아니라 다  
른 동물에게도 적용할  
수 있는 모델로의 발전  
을 제언함

기술적인 background가 있는 반려동물 감정 분석이 가능했다는 의의

# 감사합니다

비타민 13+14기 여름세션 프로젝트 | 고양이 감정분석