



Term Project

- AI 기반 일상 감정 인식 -
자살 예방과 정신 건강 증진을 위한 접근

과목명	인공지능(가)
교수명	최강타 교수님
학 과	송실대 IT 대학 컴퓨터학부
학 번	20212908
이 름	이 진
제출일	2023.12.21 목요일

I 문제 정의

- 목적

자살 예방 및 정신 건강 증진을 위해 사용자의 일상적인 감정 상태를 인식하고, 일정 기간 동안 긍정적인 감정 표현을 유도하는 시스템을 개발한다.

- 배경

현대 사회에서 정신 건강 문제는 점점 더 주목받는 문제로 부상하고 있다. 스트레스, 우울증 등은 개인의 삶의 질을 저하시킬 뿐만 아니라, 심각한 경우 자살로 이어질 수 있다. 이러한 문제에 대처하기 위해, 감정 인식 기술을 활용하여 개인의 감정 상태를 파악하고 긍정적인 감정 표현을 유도하는 것은 중요한 대안이 될 수 있을 거라 생각하여 이 프로젝트를 계획하게 되었다.

- 도전 과제

인간의 감정은 매우 복잡하며, 얼굴 표정만으로 정확한 감정 상태를 파악하는 것은 큰 도전이다. 또한, 사용자가 긍정적인 감정을 표현하도록 유도하는 것도 중요한 과제이다.

II 데이터 수집

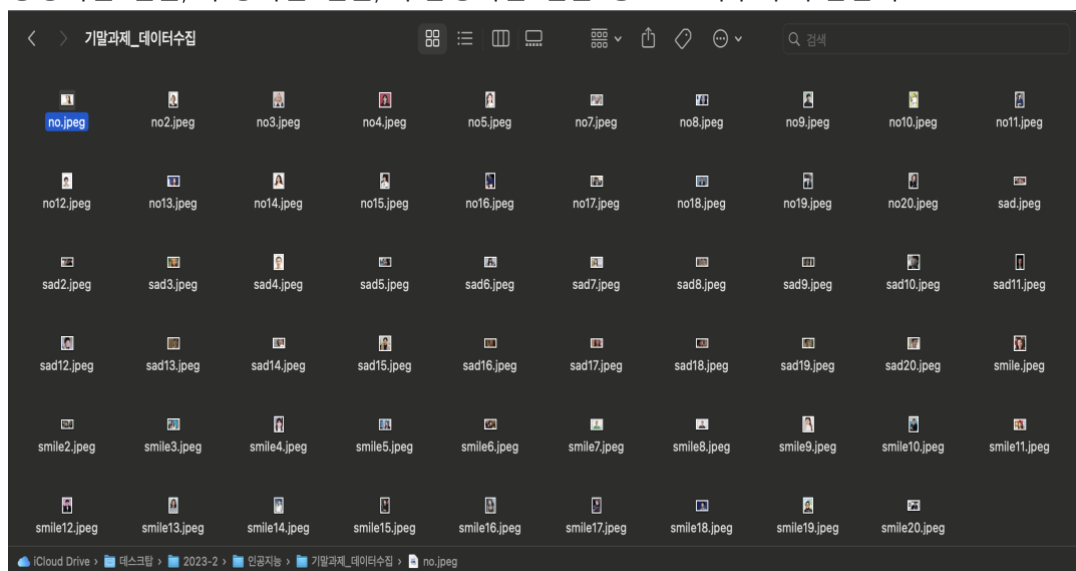
- 데이터 유형

웃는 얼굴, 우는 얼굴, 화난 얼굴 등 다양한 감정 표현을 담은 얼굴 이미지.
감정 상태를 나타내는 다양한 얼굴 표정의 이미지 데이터.

- 수집 방법

온라인 플랫폼 및 SNS 에서 이미지를 수집한다.

긍정적인 얼굴, 부정적인 얼굴, 무감정적인 얼굴 등으로 나누어 수집한다.



- 윤리적 고려사항

개인정보 보호: 사용자의 동의를 얻고, 이미지 수집 시 개인정보 보호를 위한 적절한 절차를 준수한다.

데이터 보안: 수집된 데이터의 보안을 유지하고, 무단 접근을 방지한다.

III 데이터 전처리 (옵션)

- 이미지 정규화 및 정제

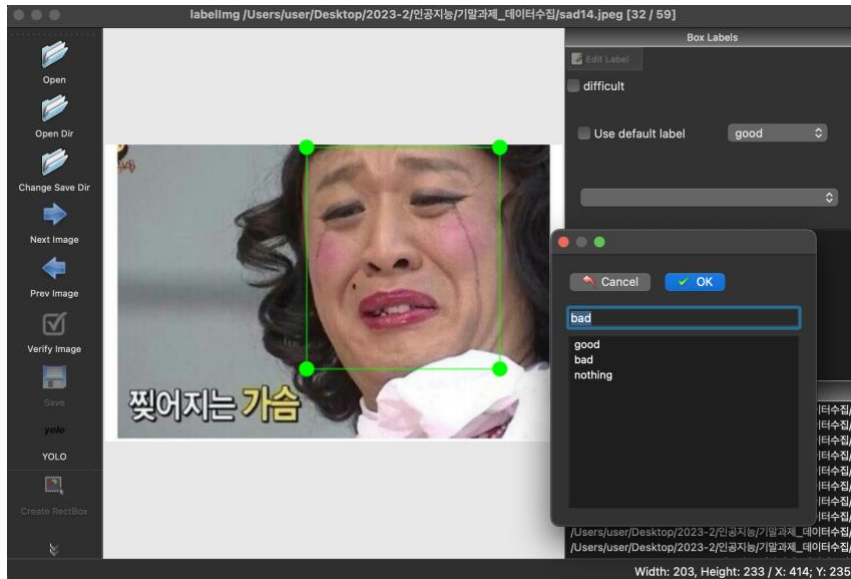
이미지 크기 및 형식을 통일하여 효율적인 처리를 보장한다.

불완전하거나 품질이 낮은 이미지를 제거하여 데이터의 품질을 보장한다.

- 라벨링

각 이미지에 대한 정확한 감정 상태 라벨링을 수행한다.

labellmg 를 사용하여 수집한 데이터를 직접 긍정(good), 부정(bad), 무표정(nothing)으로 분류한다.



IV 데이터 분석

가. 분석에 사용된 방법에 대한 소개

- 사용된 기술 및 알고리즘

YOLOv5 (You Only Look Once): YOLO 는 실시간 객체 인식을 위해 널리 사용되는 딥러닝 알고리즘이다. 이미지 전체를 한 번에 보면서 객체를 탐지하고 분류한다. 이 프로젝트에서는 YOLO 를 사용하여 얼굴 이미지에서 다양한 감정 상태를 인식한다.



- 분석 과정

데이터 준비: 수집된 얼굴 이미지에 대해 라벨링을 수행하고, 이미지 크기 및 형식을 통일한다.

모델 학습: YOLO 알고리즘을 사용하여 감정 인식 모델을 학습시킨다. 이 과정에서 다양한 얼굴 표정을 포함하는 데이터셋을 사용하여 모델의 일반화 능력을 강화한다. 정확도 향상을 위해 학습횟수는 500으로 설정하였다.

```
train: weights=yolov5s.pt, cfg=./models/yolov5s.yaml, data=./data/feeling.yaml, hyp=data/hyps/hyp.scratch-low.yaml, epochs=500
, batch_size=8, imgsz=640, rect=False, resume=False, nosave=False, noval=False, noautoanchor=False, noplots=False, evolve=None
, bucket=, cache=None, image_weights=False, device=, multi_scale=False, single_cls=False, optimizer=SGD, sync_bn=False, worker
s=8, project=runs/train, name=feeling, exist_ok=False, quad=False, cos_lr=False, label_smoothing=0.0, patience=100, freeze=[0]
, save_period=1, seed=0, local_rank=-1, entity=None, upload_dataset=False, bbox_interval=-1, artifact_alias=latest
github: skipping check (not a git repository), for updates see https://github.com/ultralytics/yolov5
YOLOv5 2023-12-1 Python-3.9.6 torch-2.1.2 CPU

hyperparameters: lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=0.05, cls=0.5, cls_pw=1.0, obj=1.0, obj_pw=1.0, iou_t=0.2, anchor_t=4.0, fl_gamma=0.0, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, mosaic=1.0, mixup=0.0, copy_paste=0.0
Comet: run 'pip install comet_ml' to automatically track and visualize YOLOv5 runs in Comet
TensorBoard: Start with 'tensorboard --logdir runs/train', view at http://localhost:6006/
Overriding model.yaml nc=80 with nc=3

from n      params module
cases 0 0 2 0 0 0 34, 10 0 UTF-8 LF Python 3.10.12 64-bit Port: 5500 Prettier

Class      Images      Instances      P      R      mAP50      mAP50-95: 100%| 4/4 [00:24<00:00, 0.000863]
all         59            59      0.00227  0.304  0.00476  0.000863

Epoch      GPU_mem      box_loss      obj_loss      cls_loss      Instances      Size
1/499      0G          0.1077       0.0304       0.04036       7              640: 100%| 8/8 [00:22<00:00, 2.87s/it]
Class      Images      Instances      P      R      mAP50      mAP50-95: 0%| 0/4 [00:00<?, 7it/s]
WARNING Δ NMS time limit 1.300s exceeded
Class      Images      Instances      P      R      mAP50      mAP50-95: 25%| 1/4 [00:05<00:16, 1/4 [00:12<00:13, 2/4 [00:19<00:06, 3/4 [00:21<00:00, 4/4 [00:21<00:00, 0.000775]
Class      Images      Instances      P      R      mAP50      mAP50-95: 100%| 4/4 [00:21<00:00, 0.000775]
all         59            59      0.0024  0.439  0.00291  0.000775

Epoch      GPU_mem      box_loss      obj_loss      cls_loss      Instances      Size
2/499      0G          0.08275     0.02993     0.03478       6              640: 100%| 8/8 [00:23<00:00, 2.94s/it]
Class      Images      Instances      P      R      mAP50      mAP50-95: 25%| 1/4 [00:04<00:14, 1/4 [00:04<00:14, 2/4 [00:04<00:14, 3/4 [00:04<00:14, 4/4 [00:04<00:14, 0.000775]
Class      Images      Instances      P      R      mAP50      mAP50-95: 100%| 4/4 [00:04<00:14, 0.000775]
all         59            59      0.0024  0.439  0.00291  0.000775

10 model = torch.hub.load('ultralytics/yolov5', 'custom', path='./yolov5-master/runs/train/feeling/weights/best.pt')
11
12 image_paths = ['./test_image/test1.jpeg', './test_image/test2.jpeg', './test_image/test3.jpeg', './test_image/test4.jpeg']
13
14 y = []
15 for image in image_paths:
16     results = model(image)
17     results = results.xyxy[0]
18     y.append(results[:, -1])
19
20 y_pred = []
21 for pred in y:
22     if len(pred) == 0:
23         y_pred.append(2)
24     else:
25         y_pred.append(int(pred[-1]))
26 y_true = [0, 2, 1, 2, 1]
27
28 # 성능 지표 계산
29 accuracy = accuracy_score(y_true, y_pred)
30 precision = precision_score(y_true, y_pred, average='macro')
31 recall = recall_score(y_true, y_pred, average='macro')
32 f1 = f1_score(y_true, y_pred, average='macro')
33
34 # 성능 지표 라벨과 값
35 metric_labels = ['accuracy', 'precision', 'recall', 'F1 score']
36 metrics = np.array([accuracy, precision, recall, f1])
37
38 conf_mat = confusion_matrix(y_true, y_pred)
39 classes = ['good', 'bad', 'nothing']
```

나.분석 방법에 대한 선택 이유

- YOLO의 적합성

고속 처리 능력: YOLO는 실시간 처리에 적합한 고속 객체 인식 알고리즘으로, 사용자의 감정 상태를 신속하게 인식할 수 있다.

높은 정확도: YOLO는 다양한 객체와 환경에서 높은 정확도를 보여준다. 이는 감정 인식에서도 정확한 결과를 도출하는 데 중요하다.

- 다른 방법들과의 비교

전통적인 머신 러닝 대비: 전통적인 머신 러닝 알고리즘들은 특징 추출 과정이 복잡하고, 실시간 처리에는 적합하지 않다. YOLO는 이러한 단점을 극복하고 빠른 처리 속도와 높은 정확도를 제공한다.

다른 딥러닝 알고리즘 대비: 다른 딥러닝 기반의 객체 인식 알고리즘들은 처리 속도가 느리거나 복잡한 설정이 필요할 수 있다. YOLO 는 이러한 문제를 최소화하면서도 우수한 성능을 제공한다.

다.분석 결과

- 주요 발견 사항

감정 인식의 정확도: YOLO 를 사용한 감정 인식 모델이 다양한 얼굴 표정에 대해 높은 정확도를 보여준 결과를 확인할 수 있었다.

감정 상태별 인식 성능: 특정 감정 상태(예: 기쁨, 슬픔, 분노 등)에 대한 인식 성능이 다른 감정 상태보다 높거나 낮은 경향을 분석했다.

- 결과의 의미

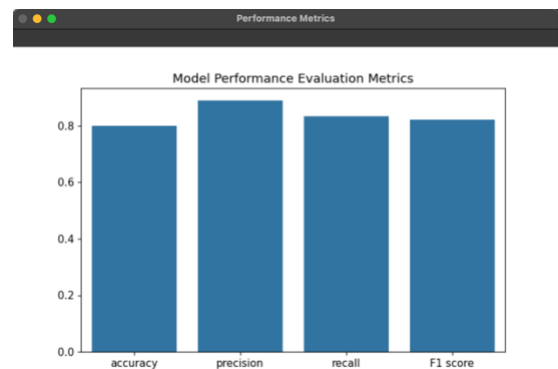
시스템의 실용성: 높은 정확도는 시스템이 실제 환경에서 효과적으로 사용될 수 있음을 시사한다.

향후 개선 방향: 일부 감정 상태에서 낮은 인식 성능을 보인 경우, 이를 개선하기 위한 추가적인 연구와 개발이 필요함을 나타낸다.

V 데이터 시각화

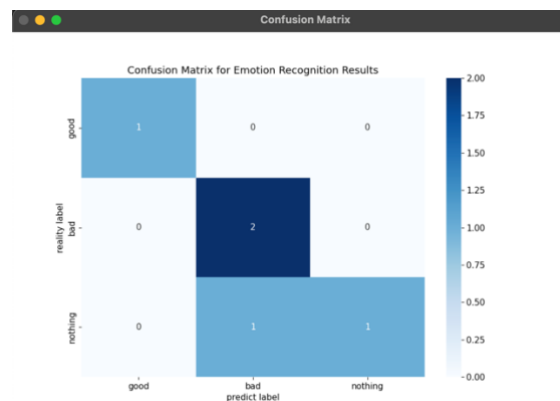
- 감정 인식 결과 시각화

각 감정 상태별 인식 결과를 차트, 그래프 또는 히트맵 형태로 시각화하여, 모델의 성능을 직관적으로 이해할 수 있게 한다. 혼동 행렬은 모델의 예측이 실제 라벨과 어떻게 일치하는지 보여준다. 이 경우, 모델은 'good', 'bad', 'nothing' 세 가지 감정 상태를 예측해야 하는 것으로 보인다. 혼동 행렬에서 대각선(왼쪽 위에서 오른쪽 아래로)에 위치한 값들은 모델이 올바르게 예측한 경우의 수를 나타낸다. 여기서 'good'과 'nothing' 상태는 각각 1 회씩 올바르게 예측되었고, 'bad' 상태는 2 회 올바르게 예측되었다. 다른 셀들은 0 으로, 잘못된 예측이 없음을 나타낸다.



- 성능 평가 매트릭스

정확도, 정밀도, 재현율, F1 스코어 등 다양한 평가 지표를 사용하여 모델 성능을 시각적으로 표현한다. 모델 성능 지표: 모델의 정확도(Accuracy), 정밀도(Precision), 재현율(Recall), F1 스코어(F1 Score)를 나타내는 막대 그래프이다. 정확도가 약 0.8(80%)인 것으로 보이며, 나머지 세 지표도 비슷한

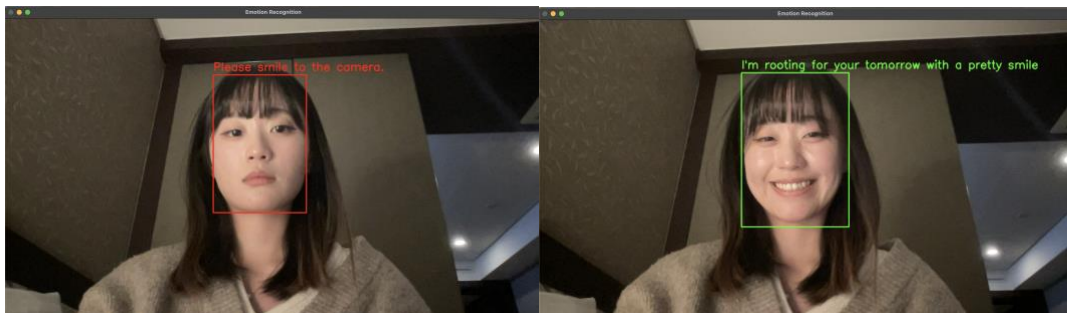


수준이다. 이는 모델이 전체적으로 꽤 괜찮은 성능을 보이고 있음을 나타낸다.

- 사용자 피드백

프로그램을 사용하는 사용자로부터의 반응을 시각적으로 분석하여 바운딩박스를 그린다, 웃고 있지 않으면 미소를 요청한다. 웃고있다면 초록색으로 눈에 띄게 바운딩박스과 글자 색상을 변경하여 응원의 글귀를 남긴다.

```
5 model = torch.hub.load('ultralytics/yolov5', 'custom', path='./yolov5-master/runs/train/feeling/weights/best.pt')
6 cap = cv2.VideoCapture(1)
7 while True:
8     ret, frame = cap.read()
9     if not ret:
10         break
11
12     results = model(frame)
13
14     for *xyxy, conf, cls in results.xyxy[0]:
15         label = model.names[int(cls)] # 클래스 이름 가져오기
16         if label == 'good':
17             text = "I'm rooting for your tomorrow with a pretty smile"
18             color = (0, 255, 0) # 녹색
19         elif label == 'sad':
20             text = "Don't cry, your flower is too beautiful to wither under water"
21             color = (255, 0, 0) # 파란색
22         else:
23             text = 'Please smile to the camera.'
24             color = (0, 0, 255) # 빨간색
25
26         cv2.rectangle(frame, (int(xyxy[0]), int(xyxy[1])), (int(xyxy[2]), int(xyxy[3])), color, 2)
27
28         cv2.putText(frame, text, (int(xyxy[0]), int(xyxy[1]) - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
29
30     cv2.imshow('Emotion Recognition', frame)
```



VI 결론 AND/OR 제언

해당 프로젝트는 일상 속 감정 인식을 통해 정신 건강을 증진하고 자살 방지에 기여할 수 있는 AI 시스템 개발에 중점을 두었다. 다양한 온라인 플랫폼에서 수집한 얼굴 표정 데이터를 활용하여, 개인정보 보호 및 데이터 안전성 확보에 주의를 기울이며 YOLOv5 알고리즘을 적용, 실시간 감정 인식의 정확도를 높였다. 이 기술은 실제 환경에서의 적용 가능성이 매우 높음을 입증하였으나, 특정 감정의 인식률을 향상시키기 위한 지속적인 연구가 필요함을 드러냈다. 실제 사용자 경험을 통해 이 시스템이 긍정적인 정신 건강 효과를 제공한다는 실질적인 증거도 발견되었으며, 이를 기반으로 추가적인 프로그램과의 통합을 통해 보다 광범위한 정신 건강 솔루션으로 발전시킬 수 있는 잠재력이 있음을 시사한다.