OpenCV와 딥러닝을 이용한 얼굴인식 동영상 처리

최석재 *lingua@naver.com*

동영상 처리

- 동영상을 처리하는 것도 크게 다르지 않다

- 기본적으로 OpenCV의 VideoCapture() 를 사용하여 프레임을 나누고
- 각 프레임 별로 앞과 동일한 이미지 처리를 한다

구글 드라이브 연결

```
from google.colab import drive
drive.mount('/content/gdrive')

# 경로 변경
%cd /content/gdrive/MyDrive/pytest_img/opencv/
```

시간 및 경로 설정

- import cv2
- import numpy as np
- import dlib
- import os

- seconds = 1 # 동영상을 읽을 시간을 초 단위로 설정

- video_path = '/content/gdrive/MyDrive/pytest_img/opencv/eye_opening.mp4' # video path
- output_dir = '/content/gdrive/MyDrive/pytest_img/opencv/results' # 결과를 저장할 폴더

예측 파일 로드

얼굴 영역 인식 함수

- detector = dlib.get_frontal_face_detector()

얼굴의 68개 지점 인식 모델 로드

- sp = dlib.shape_predictor("/content/gdrive/MyDrive/pytest_img/dlib/shape_predictor_68_face_landmarks.dat")

눈 위치와 EAR 함수 정의

- 눈의 위치만 정의한다

- RIGHT_EYE = list(range(36, 42)) # 오른쪽 눈. 36~41까지 생성
- LEFT_EYE = list(range(42, 48)) # 왼쪽 눈. 42~47까지 생성

- def eye_aspect_ratio(eye_points):

```
    A = np.linalg.norm(eye_points[1] - eye_points[5])  # 벡터의 길이 계산
    B = np.linalg.norm(eye_points[2] - eye_points[4])
    C = np.linalg.norm(eye_points[0] - eye_points[3])

    EAR = (A + B) / (2.0 * C)
    return EAR
```

예측 함수 정의 (1/2)

※ 프레임 단위로 반복해야 하므로 예측 부분을 함수로 구현한다

- def predict(frame, predictor, LEFT_EYE, RIGHT_EYE):

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)        # 잘 인식하도록 그레이스케일로 변환
    dets = detector(gray, 1)                              # 얼굴 감지

    for det in dets:
        sp = predictor(gray, det)
        points = np.array([[p.x, p.y] for p in sp.parts()])

        left_eye_EAR = eye_aspect_ratio(points[LEFT_EYE])     # 왼쪽 눈 EAR
        right_eye_EAR = eye_aspect_ratio(points[RIGHT_EYE])   # 오른쪽 눈 EAR
```

얼굴 주요 지점 인식 모델(sp)를 사용하여 각 얼굴영역에서 얼굴 랜드마크를 찾고, 이 포인트들의 x, y 좌표를 NumPy 배열로 생성

예측 함수 정의 (2/2)

- frame 단위 이미지의 (왼쪽 눈 X 위치, 오른쪽 눈 Y 위치 +20) 포지션에 EAR 값을 출력한다

```
        # 눈 위치에 EAR 값 표시 (왼쪽 눈, 오른쪽 눈)
        cv2.putText(frame, f"{left_eye_EAR:.2f}", (points[LEFT_EYE[0]][0], points[LEFT_EYE[0]][1] + 20), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)
        cv2.putText(frame, f"{right_eye_EAR:.2f}", (points[RIGHT_EYE[0]][0], points[RIGHT_EYE[0]][1] + 20), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)

    return frame
```

cv2.putText(image, text, text_position, font, font_scale, text_color, thickness)

```python
def predict(frame, predictor, LEFT_EYE, RIGHT_EYE):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    dets = detector(gray, 1)  # 얼굴 감지

    for det in dets:
        sp = predictor(gray, det)
        points = np.array([[p.x, p.y] for p in sp.parts()])

        left_eye_EAR = eye_aspect_ratio(points[LEFT_EYE])
        right_eye_EAR = eye_aspect_ratio(points[RIGHT_EYE])

        # 눈 위치에 EAR 값 표시
        cv2.putText(frame, f"{left_eye_EAR:.2f}", (points[LEFT_EYE[0]][0], points[LEFT_EYE[0]][1] + 20), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)     # 왼쪽 눈
        cv2.putText(frame, f"{right_eye_EAR:.2f}", (points[RIGHT_EYE[0]][0], points[RIGHT_EYE[0]][1] + 20), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)  # 오른쪽 눈

    return frame
```

파일 로드 및 출력 프레임 계산

- capture = cv2.VideoCapture(video_path) # 비디오 파일 로드

비디오가 마지막 프레임에 도달한 경우, 처음부터 다시 시작하기
CAP_PROP_POS_FRAMES는 비디오의 현재 프레임, CAP_PROP_FRAME_COUNT은 비디오의 총 프레임 수
- if capture.get(cv2.CAP_PROP_POS_FRAMES) == capture.get(cv2.CAP_PROP_FRAME_COUNT):
  capture.set(cv2.CAP_PROP_POS_FRAMES, 0) # 첫 번째 프레임으로 변경

- fps = capture.get(cv2.CAP_PROP_FPS) # 초당 프레임 수 얻기
- total_frames = int(fps * seconds) # 현재 seconds == 1
- print(' total 프레임:', total_frames) # 1초인 경우, 25 프레임

출력 디렉토리 생성
- if not os.path.exists(output_dir):
  os.makedirs(output_dir)

프레임 별 예측 수행 (1/2)

- frame_count = 0
- while frame_count < total_frames:

```
    ok, frame = capture.read()
    if not ok:
        print("프레임 읽기에 실패했습니다. 종료.")
        break

    frame = predict(frame, sp, LEFT_EYE, RIGHT_EYE)                    # 예측 수행

    output_path = os.path.join(output_dir, f'frame_{frame_count:04d}.jpg')      # 파일로 저장
    cv2.imwrite(output_path, frame)
    print("Saved to:", output_path)
    frame_count += 1
```

프레임 별 예측 수행 (2/2)

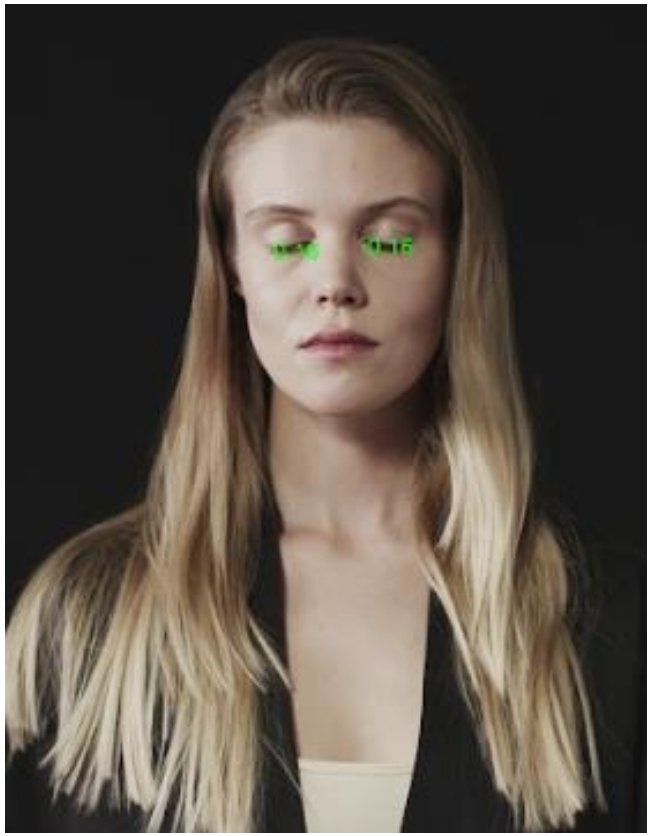
```
    key = cv2.waitKey(10)                                    # 키 입력 대기 및 종료 처리
    if key == ord('q'):
        print("사용자가 종료를 요청했습니다.")
        break
```

- capture.release()
- cv2.destroyAllWindows()

```
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0000.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0001.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0002.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0003.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0004.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0005.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0006.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0007.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0008.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0009.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0010.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0011.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0012.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0013.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0014.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0015.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0016.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0017.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0018.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0019.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0020.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0021.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0022.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0023.jpg
Saved to: /content/gdrive/MyDrive/pytest_img/opencv/results/frame_0024.jpg
```

결과 확인

- output_dir '/content/gdrive/MyDrive/pytest_img/opencv/results' 의 결과를 확인한다



나이와 성별

나이와 성별 동영상 처리

- 나이와 성별을 예측하는 부분도 동일하게 처리할 수 있다

- 이번에는 처리된 프레임을 모아서 다시 비디오를 생성하는 부분까지 진행한다
- 기본적으로 OpenCV의 VideoWriter()를 사용한다

경로 설정

- seconds = 1 # 동영상을 읽을 시간을 초 단위로 설정

비디오 경로 설정 및 결과 저장 경로 설정

- video_path = '/content/gdrive/MyDrive/pytest_img/opencv/eye_opening.mp4'
- output_dir = '/content/gdrive/MyDrive/pytest_img/opencv/results2'

구간 정의

나이 구간 정의

- age_list = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)’]

성별 구분 정의

- gender_list = ["Male", 'Female']

인식 함수

얼굴 영역 인식 함수

- detector = dlib.get_frontal_face_detector()

나이 인식 함수

- age_detector = cv2.dnn.readNetFromCaffe("/content/gdrive/MyDrive/pytest_img/dlib/deploy_age.prototxt", "/content/gdrive/MyDrive/pytest_img/dlib/age_net.caffemodel")

성별 인식 함수

- gender_detector = cv2.dnn.readNetFromCaffe("/content/gdrive/MyDrive/pytest_img/dlib/deploy_gender.prototxt", "/content/gdrive/MyDrive/pytest_img/dlib/gender_net.caffemodel")

파일 로드 및 출력 프레임 계산

비디오 파일 로드
- capture = cv2.VideoCapture(video_path)

비디오가 마지막 프레임에 도달한 경우, 처음부터 다시 시작하기
- if capture.get(cv2.CAP_PROP_POS_FRAMES) == capture.get(cv2.CAP_PROP_FRAME_COUNT):
  capture.set(cv2.CAP_PROP_POS_FRAMES, 0)

- fps = capture.get(cv2.CAP_PROP_FPS) # 초당 프레임 수 얻기
- total_frames = int(fps * seconds)
- print('total 프레임:', total_frames)

출력 디렉토리 생성
- if not os.path.exists(output_dir):
  os.makedirs(output_dir)

예측 함수 (1/2)

- def predict(frame, age_detector, gender_detector, age_list, gender_list):

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)        # 잘 인식하도록 그레이스케일로 변환
    dets = detector(gray, 1)                              # 얼굴 감지

    for det in dets:
        x1, y1, x2, y2 = det.left(), det.top(), det.right(), det.bottom()
        face_img = frame[y1:y2, x1:x2].copy()             # 원본 이미지에서 얼굴 영역만 copy
        blob = cv2.dnn.blobFromImage(face_img, scalefactor=1, size=(227, 227),
mean=(78.4263377603, 87.7689143744, 114.895847746), swapRB=False)
```

이미지 전처리를 수행하여 blob(Binary Large OBject) 타입으로 변환
scale factor: 이미지의 픽셀값 조정을 위한 스케일링 인자. 1은 원본 픽셀값. 0.00392를 사용하면 이미지가 0~255일 때 0~1 범위로 조정됨
신경망 입력을 위해 조정하는 고정된 이미지 크기. (227, 227)
mean: 각 색상 채널(BRG)에 대해 빼줄 평균값. 모델 학습 시 사용된 평균값과 일치해야 하므로 고정값
swapRB: 딥러닝 모델이 사용한 것과 같게 BGR 순서를 사용. True는 RGB.

예측 함수 (2/2)

```
    age_detector.setInput(blob)                                                                                    # 나이 예측 모델의 입력으로 설정
    age_preds = age_detector.forward()                                                                             # 모델을 실행하여 예측 결과를 얻음
    age = age_list[age_preds[0].argmax()]                                                                          # 가장 높은 확률을 선택

    gender_detector.setInput(blob)                                                                                 # 성별 예측
    gender_preds = gender_detector.forward()
    gender = gender_list[gender_preds[0].argmax()]

    # 나이와 성별 표시
    cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 255, 255), 2)
    text = f'{gender}, {age}'
    cv2.putText(frame, text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 0, 0), 10)          # 검정색 배경
    cv2.putText(frame, text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)         # 녹색 텍스트

return frame
```

예측 함수 전체 코드

```python
def predict(frame, age_detector, gender_detector, age_list, gender_list):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)   # 잘 인식하도록 그레이스케일로 변환
    dets = detector(gray, 1)                         # 얼굴 감지

    for det in dets:
        x1, y1, x2, y2 = det.left(), det.top(), det.right(), det.bottom()
        face_img = frame[y1:y2, x1:x2].copy()        # 원본 이미지에서 얼굴 영역만 copy
        blob = cv2.dnn.blobFromImage(face_img, scalefactor=1, size=(227, 227), mean=(78.4263377603, 87.7689143744, 114.895847746), swapRB=True)

        # 나이 예측
        age_detector.setInput(blob)
        age_preds = age_detector.forward()
        age = age_list[age_preds[0].argmax()]

        # 성별 예측
        gender_detector.setInput(blob)
        gender_preds = gender_detector.forward()
        gender = gender_list[gender_preds[0].argmax()]

        # 결과 그리기
        cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 255, 255), 2)
        text = f'{gender}, {age}'
        cv2.putText(frame, text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 0, 0), 4)    # 검정색 배경
        cv2.putText(frame, text, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 3)  # 녹색 텍스트

    return frame
```

비디오 생성

- fourcc = cv2.VideoWriter_fourcc(*'XVID')

비디오 표준 코딩을 사용하는 XVID를 사용하여 비디오 생성
XVID는 좋은 성능을 보이는 오픈소스 MPEG-4 코덱

초당 25 프레임 사용

원본 비디오 프레임의 너비(capture.get(3))와 높이(capture.get(4))를 반환하여 같은 크기로 비디오 생성

- out = cv2.VideoWriter(output_dir+'/output.avi', fourcc, 25.0, (int(capture.get(3)), int(capture.get(4))))

프레임 별 예측 수행 (1/2)

- frame_count = 0
- while frame_count < total_frames:

```
    ok, frame = capture.read()
    if not ok:
        print("프레임 읽기에 실패했습니다. 종료.")
        break

    frame = predict(frame, age_detector, gender_detector, age_list, gender_list)      # 예측 수행
    out.write(frame)                                                                  # 처리된 프레임을 동영상 파일로 저장

    output_path = os.path.join(output_dir, f'frame_{frame_count:04d}.jpg')            # 이미지 파일도 저장
    cv2.imwrite(output_path, frame)
    print("Saved to:", output_path)
    frame_count += 1
```

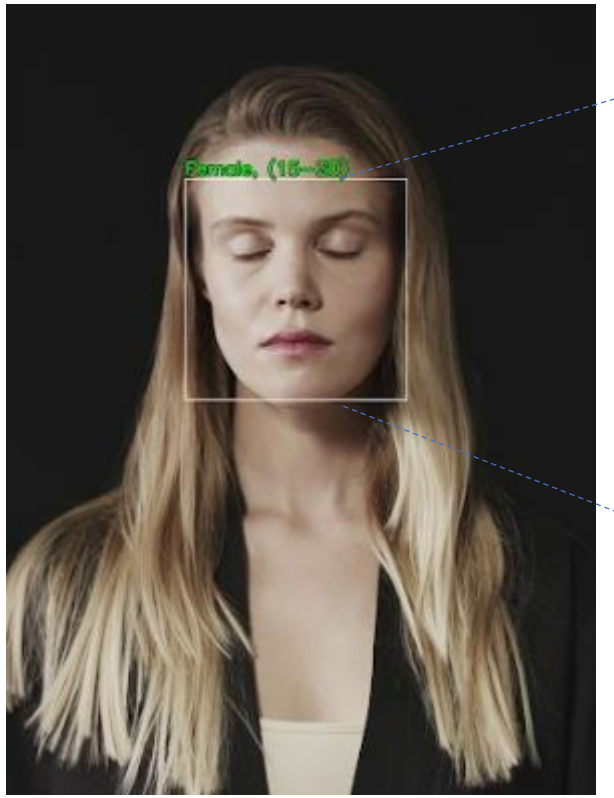
프레임 별 예측 수행 (2/2)

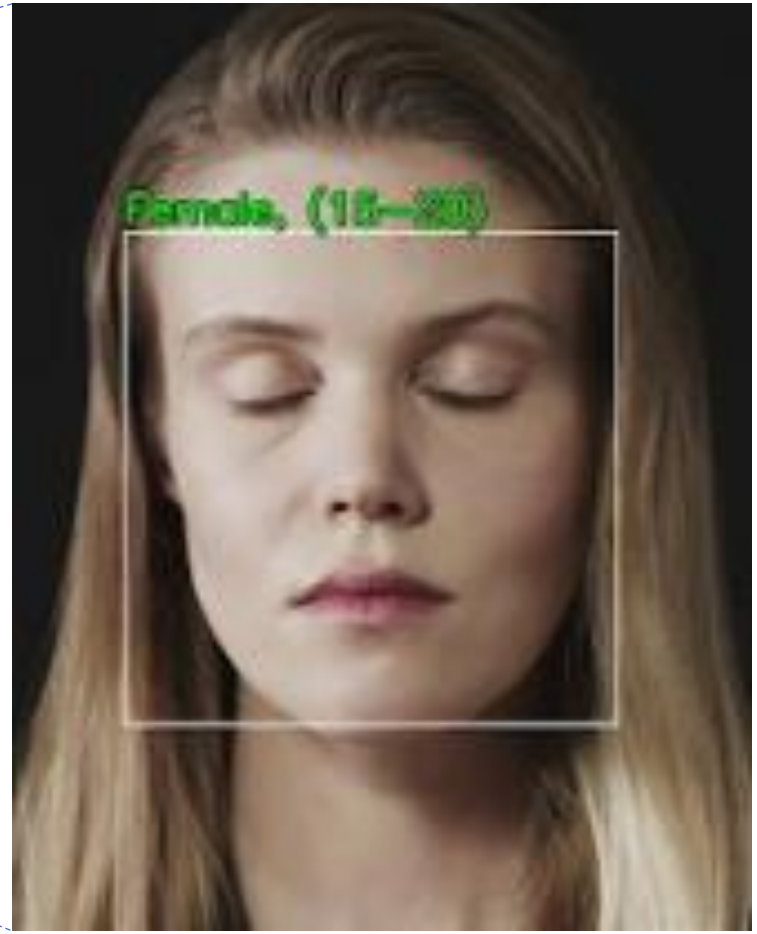
```
# 사용자가 프로세스 중간에 종료할 수 있도록 키 입력 대기 및 종료 처리
key = cv2.waitKey(10)
if key == ord('q'):
    print("사용자가 종료를 요청했습니다.")
    break
```

- out.release()
- capture.release()
- cv2.destroyAllWindows()

```
frame_count = 0
while frame_count < total_frames:
    ok, frame = capture.read()
    if not ok:
        print("프레임 읽기에 실패했습니다. 종료.")
        break

    frame = predict(frame, age_detector, gender_detector, age_list, gender_list)  # 예측 수행
    out.write(frame)                                                              # 처리된 프레임을 동영상 파일로 저장

    output_path = os.path.join(output_dir, f'frame_{frame_count:04d}.jpg')        # 파일로 저장
    cv2.imwrite(output_path, frame)
    print("Saved to:", output_path)
    frame_count += 1

    # 키 입력 대기 및 종료 처리
    key = cv2.waitKey(10)
    if key == ord('q'):
        print("사용자가 종료를 요청했습니다.")
        break

out.release()
capture.release()
cv2.destroyAllWindows()
```

결과 확인





결과 확인

- 1초짜리 동영상도 만들어졌다