한국공학대학교
TECH UNIVERSITY OF KOREA

집중 분석 **AI** 프로젝트

AI소프트웨어학과

2023년**06**월

목차

I서론 ——————————————————————

1장 - 학습 목표

2장 - 사전 준비

3장 - 사전 지식 쌓기

II본론 ——————————————————————

4장 - 구현

5장 - 추후 추가 기능

III결론——————————————————————

6장 - 요약정리

1. 학습 목표

현대 사회에서는 많은 정보와 변화가 빠르게 이루어지고 있다. 특히 IT 분야의 개발자들은 이러한 변화에 빠르게 적응하고, 새로운 기술과정보를
계속해서 습득해야 한다. 그러나 무수히 많은 정보 속에서 필요한 정보를 찾고, 효과적으로 습득하고 활용하는 것은 쉬운 일이 아니다. 따라서 시간을 어떻게 효과적으로 활용할 수 있을지를 고민하였다. 그런 고민 속에서 나온 결론이 바로 이 프로젝트이다. 이 프로젝트는 개발자들이 시간을 더 효율적으로 사용하고, 그 결과로 더 나은 개발자가 될 수 있도록 도움을 주는 도구로 계획되었다. 또한, 이 프로젝트를 통해 더욱 집중력 있는 시간을 보낼 수 있도록 돕고, 그 시간을 통해 더 많은 것을 배울 수 있도록 도와주는 것이 핵심 목표이다.

구현 순서

구글에서 제공하는 Teachable Machine으로 정상 모습인지 엎드린 모습인지를 판별하는 모델을 학습합니다. OpenCV를 통해서 카메라로 촬영을 합니다. 촬영된 모습이 30초 이상 엎드린 모습으로 유지되면, 소리로 알람을 주고 카카오 메시지를 전송합니다.

2. 사전 준비

[언어 & 환경(IDE)]

Python & Jupter notebook

1) '나에게 카카오톡 보내기'의 사전 준비

카카오톡 개발자 사이트에 접속하여 앱만들기를 하고 API를 받아와 준비한다.

2) 웹캠

보통 노트북에 내장되어 있는 웹캠을 많이사용하여 노트북 웹캠을 기준으로 하였다.

3) 카카오 애플리케이션에 사이트 도메인 등록하기

30초 이상 집중 이외에 모습이 유지되면, 3가지 조치를 취하려 한다. 그중 한가지 방법이 카톡으로 메시지로 '졸음 방지 소리' 유튜브 링크를 보내는 것이다.
그것을 위한 방법의 순서는

1. 카카오 개발자 사이트 접속한다.
2. "내 애플리케이션" 클릭한다.
3. 앱설정 클릭한다.
4. 플랫폼 클릭한다.
5. Web에서 [수정] 버튼 클릭한다.
6. www.youtube.com 추가한다.

4) 라이브러리 설치

$ pip install opencv-python
$ pip uninstall tensorflow
$ pip install tensorflow==2.3
$ pip install beepy
$ pip install pandas

$ pip install opencv-python
집중하는 동안의 모습과 다른 모습을 비교하려면 카메라를 사용해야 한다. 이를 위해 OpenCV라는 라이브러리를 사용한다. OpenCV란 실시간 컴퓨터 비전을 위한 라이브러리로, 파이썬 환경에서 opencv-python으로 설치할 수 있다.

$ pip uninstall tensorflow
$ pip install tensorflow==2.3
이 라이브러리는 딥러닝과 같은 기계학습에 자주 사용된다. Google Teachable Machine을 이용하기 위해서도 필요하다. 다양한 버전이 있지만, tensorflow.keras가 지원되는 버전인 tensorflow 2.3 이상을 사용해야 한다. 참고로, 이는 2020년 11월 이후 버전을 말한다.

$ pip install beepy
이 라이브러리는 업무에 집중하지 않는 상태에서 30초 이상 동작이 없으면, 컴퓨터 내장 음악을 재생하여 알림을 제공하는라이브러리이다.

$ pip install pandas
데이터 처리에 최적화되어 있고, 데이터 분석에서 자주 사용되는 기능이다.

3. 사전 지식 쌓기

3-1) OpenCV로 카메라 입력받기

OpenCV는 이미지 처리에 필요한 다양한 기능을 제공한다. 내장 카메라를 사용하여 촬영 테스트를 진행해볼 예정이다. 프로그램을 실행하면 카메라가 켜지고 작은 윈도우 창이 열린 후 카메라 촬영 화면이 표시된다.

```
import cv2

# 카메라 캡쳐 객체, 0=내장 카메라
capture = cv2.VideoCapture(0)

# 캡쳐 프레임 사이즈 조절
capture.set(cv2.CAP_PROP_FRAME_WIDTH, 320)
capture.set(cv2.CAP_PROP_FRAME_HEIGHT, 240)

while True: # 특정 키를 누를 때까지 무한 반복
    # 한 프레임씩 읽기
    ret, frame = capture.read()
    if ret == True:
        print("read success!")

    # 이미지 뒤집기, 1=좌우 뒤집기
    frame_fliped = cv2.flip(frame, 1)

    # 읽어들인 프레임을 윈도우창에 출력
    cv2.imshow("VideoFrame", frame_fliped)

    # 1ms동안 사용자가 키를 누르기를 기다림
    if cv2.waitKey(1) > 0:
        break

# 카메라 객체 반환
capture.release()

# 화면에 나타난 윈도우들을 종료
cv2.destroyAllWindows()
```

실행 결과

윈도우 작업표시줄에 Python 윈도우 창이 생성되었다. 클릭하면 카메라로 촬영된 영상이 화면에 나타난다. Python 윈도우 창에서 아무 키나 누르면 촬영이 중지되고 윈도우 창이 종료된다.

3-2) Google Teachable Machine 이란?

수업 시간에 사용해본 것은 기계 학습 과정을 간편하고 빠르게 도와주면서, 최종적으로 학습이 완료된 모델을 제공하는 도구이다. 이 과정은 간단히 3단계로 나눌 수 있다.

3-3) Google Teachable Machine 사용 방법

1단계 Gather : 컴퓨터가 학습을 하기 위한 데이터를 모은다.

2단계 Train : 컴퓨터가 학습을 한다.

3단계 Export : 학습을 마친 후 model을 다운로드한다.

3-4) Google Teachable Machine로 모델 만들기

이미지 프로젝트를 클릭한다. 이후

1단계 **Gather**
컴퓨터가 학습할 수 있도록 집중하는 사진들과 기타 행동들을 준비한다. 팀원들과 지인들에게 부탁해 많은 사진 자료를 수집했다. 준비가 완료되면 [업로드] 버튼을 클릭하여 사진을 업로드한다. 그 후 'Class 1'을 '엎드림'으로 수정한다. 'Class 2', 'Class 3', 'Class 4'도 동일한 방법으로 업로드하고, 각각 '집중', '자리 비움', '폰 사용'으로 이름을 변경한다.

2단계 **Train**
[Train Model] 버튼을 클릭하여 학습을 시작한다. 진행 상황은 프로그레스 바로 확인할 수 있다. 잠시 기다리면 학습이 완료된다.

3단계 **Export**
프로그램으로 모델을 다운로드하여 사용할 수 있지만, 이번에는 모델이 잘 동작하는지만 확인한다. [File] 버튼을 클릭한 후, 학습에 사용하지 않은 이미지를 업로드한다. 하단에 기계가 구분한 결과가 표시된다. '엎드림'의 경우 100%로 나타난다. 정확성을 위해 '폰 사용' 이미지도 업로드하여 확인해본다. 이번에도 100%로 결과가 나온다.

4. 구현

구현할 순서는 구글에서 제공하는 Teachable Machine으로 업무에 집중하는 모습인지 그외에 모습들을 알려주는 모델을 학습한다. OpenCV를 통해서 카메라 촬영을 한다. 30초 이상 집중 이외에 다른 행동을 유지되면, 소리로 알람을 주고 카카오 메시지를 전송한다.

[구현 순서**]**
점진적으로 구현을 완성해 갈 수 있도록 아래 순서로 진행되며, Step3에서는 소스코드를 완성한다.

Step1	엎드린 모습을 판별하는 모델 생성하기
Step2	정신차리는 방법 설정
Step3	카메라로 엎드린 상태 감지하기

Step1) 집중하는 모습을 판별하는 모델 생성하기

Teachable Machine 웹사이트에 접속한다 > [Get Started]를 클릭한다 > [Image Project]를 선택하여 아래와 같은 화면으로 이동한다

3-4) Google Teachable Machine로 모델 만들기

원하는 결과가 잘 판별될 경우, Tensorflow에서 'Download my model'을 클릭하여 모델을 다운로드한다. 이때 다운로드 된 파일은 'converted_keras.zip'이라는 이름을 가지고 있다. 해당 파일을 리소스 관리용 폴더로 복사한 후 압축을 해제한다. 'converted_keras' 폴더 안에는 두 개의 파일이 존재한다.

1) 'keras_model.h5'는 완료된 학습 모델 파일이다.

2) 'labels.txt' 파일을 확인해보면, class1 (엎드림)은 0을 의미하고, class2 (집중)은 1을 의미하는 라벨 값들이 적혀있다.
모델을 다운로드한 후, 웹캠이 꺼지도록 미리보기의 입력을 'off'로 변경한다.

Step2) 정신차리는 방법 설정

집중하지 않는 모습이 30초 이상 지속되면, 잠을 깨기 위해 2가지 조치를 취하려 한다. 첫 번째 방법은 컴퓨터로부터 소리를 발생시키는 것이다.
두 번째 방법은 카카오톡 메시지로 '졸음 방지 베타파'가 포함된 유튜브 링크를 전송하는 것이다.

```
import beepy
import kakao_utils

# 컴퓨터에 내장된 소리를 출력
def beepsound():
    beepy.beep(sound=6)

# 카카오톡 메시지로 '졸음 방지 베타파' 영상 링크를 전송
def send_music_link():
    KAKAO_TOKEN_FILENAME =
"res/kakao_message/kakao_token.json"# "<kakao_token.json 파일이
있는 경로를 입력하세요.>"
    KAKAO_APP_KEY = "<REST_API 앱키를 입력하세요>"
    tokens = kakao_utils.update_tokens(KAKAO_APP_KEY,
KAKAO_TOKEN_FILENAME)


    # 텍스트 메시지 보내기
    template = {
        "object_type": "text",
        "text": "당신은 30초 이상 집중하지 않았습니다. 집중 하세요!!!!",
        "link": {
          "web_url": "https://www.youtube.com/watch?v=7Q2N7919o5o",
          "mobile_web_url":
"https://www.youtube.com/watch?v=7Q2N7919o5o"
        },
        "button_title": "정신차리는 노래 듣기"
    }

    # 카카오 메시지 전송
    res = kakao_utils.send_message(KAKAO_TOKEN_FILENAME,
template)
    if res.json().get('result_code') == 0:
```

```
        print('텍스트 메시지를 성공적으로 보냈습니다.')
    else:
        print('텍스트 메시지를 보내지 못했습니다. 오류메시지 : ',
res.json())
```

실행 결과

beepsound()를 호출하면, 컴퓨터에 내장된 소리가 출력된다.
send_music_link()를 호출하면, 아래와 같은 메시지가 전송된다.

Step3) 카메라로 집중외에 상태 감지하기 및 그래프로 표현

3-1) OpenCV로 카메라 입력받기'에서 배운 내용을 기반으로 카메라를 통해 현재 상태를 촬영한다. 촬영된 프레임을 학습 모델에 입력값으로 제공하여, 사용자가 집중 상태인지 아닌지를 확인한다.

```
import cv2
import tensorflow.keras
import numpy as np
import beepy
import kakao_utils
import pyecharts.options as opts
from pyecharts.charts import Pie
import webbrowser

# 시스템 사운드 출력 함수
def beepsound():
    beepy.beep(sound=6)

# 카카오톡 나에게 메세지 보내는 함수
def send_music_link():
    KAKAO_TOKEN_FILENAME = "C:/Users/user/Desktop/Python work
spapce/kakao_token.json"
    KAKAO_APP_KEY = "3b34f2e7f557ef84bf043fccf76e1791"
    tokens = kakao_utils.update_tokens(KAKAO_APP_KEY, KAKAO_TOKEN_FILENAME)

# 이미지 전처리
```

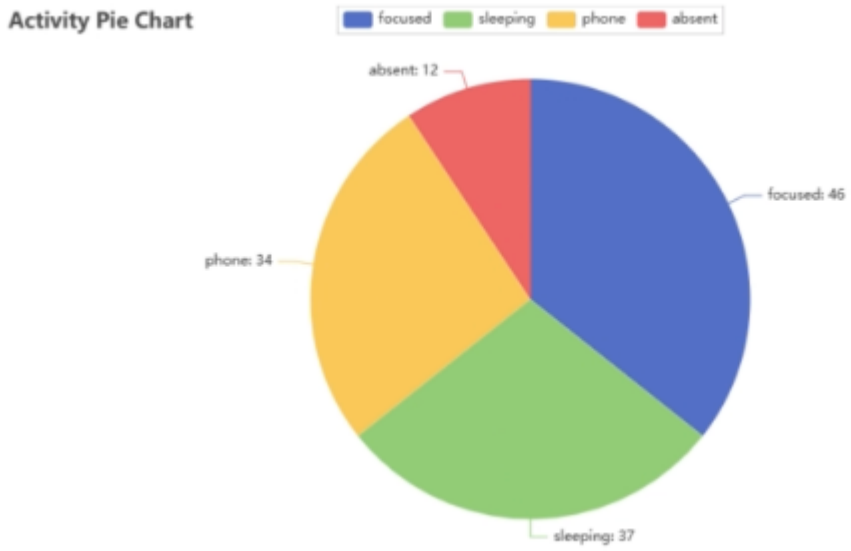
```
def preprocessing(frame):
    # 사이즈 조정
    size = (224, 224)
    frame_resized = cv2.resize(frame, size, interpolation=cv2.INTER_AREA)
    # 이미지 정규화
    frame_normalized = (frame_resized.astype(np.float32) / 127.0) - 1
    # 이미지 차원 재조정 - 예측을 위해 reshape 해줍니다.
    frame_reshaped = frame_normalized.reshape((1, 224, 224, 3))
    return frame_reshaped

## 학습된 모델 불러오기
model_filename = 'C:\\Users\\user\\Desktop\\res\\dont_sleep\\keras_model.h5'
model = tensorflow.keras.models.load_model(model_filename)

# 카메라 캡쳐 객체, 0=내장 카메라
capture = cv2.VideoCapture(0)

# 캡쳐 프레임 사이즈 조절
capture.set(cv2.CAP_PROP_FRAME_WIDTH, 320)
capture.set(cv2.CAP_PROP_FRAME_HEIGHT, 240)

# 각 상태에 대한 카운터
sleep_cnt = 1
phone_cnt = 1
focused_cnt = 1
absent_cnt = 1
activity_dict = {'focused': 0, 'sleeping': 0, 'phone': 0, 'absent': 0}

try:
    while True:
        ret, frame = capture.read()
        if ret == True:
            print("read success!")

        # 이미지 뒤집기
        frame_fliped = cv2.flip(frame, 1)

        # 이미지 출력
        cv2.imshow("VideoFrame", frame_fliped)

        # 데이터 전처리
        preprocessed = preprocessing(frame_fliped)

        # 예측
        prediction = model.predict(preprocessed)

        if prediction[0,0] > prediction[0,1] and prediction[0,0] > prediction[0,2] and
prediction[0,0] > prediction[0,3]:
            print('집중 상태')
            activity_dict['focused'] += 1
            sleep_cnt = 1
        elif prediction[0,1] > prediction[0,0] and prediction[0,1] > prediction[0,2] and
prediction[0,1] > prediction[0,3]:
            print('엎드린 상태')
```

```
            activity_dict['sleeping'] += 1
            sleep_cnt += 1
            if sleep_cnt % 30 == 0:
                sleep_cnt = 1
                print('30초간 졸고 있네요!!!')
                beepsound()
                send_music_link()
        elif prediction[0,2] > prediction[0,0] and prediction[0,2] > prediction[0,1] and
prediction[0,2] > prediction[0,3]:
            print('자리 비움 상태')
            activity_dict['absent'] += 1
            absent_cnt += 1
            if absent_cnt % 30 == 0:
                absent_cnt = 1
                print('30초간 자리를 비우고 있네요!!!')
                beepsound()
                send_music_link()
        else:
            print('휴대폰 하는 상태')
            activity_dict['phone'] += 1
            phone_cnt += 1
            if phone_cnt % 30 == 0:
                phone_cnt = 1
                print('30초간 휴대폰을 하고 있네요!!!')
                beepsound()
                send_music_link()

        # 'q' 키를 누르면 종료
        if cv2.waitKey(200) & 0xFF == ord('q'):
            break

# 강제 종료 시 차트 생성
except KeyboardInterrupt:
    pie = (Pie()
        .add("", [list(z) for z in zip(activity_dict.keys(), activity_dict.values())])
        .set_global_opts(title_opts=opts.TitleOpts(title="Activity Pie Chart"))
        .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {c}"))
    )
    pie.render('activity_pie_chart.html')
    webbrowser.open('activity_pie_chart.html')

finally:
    # 카메라 객체 반환
    capture.release()
    # 화면에 나타난 윈도우들을 종료
    cv2.destroyAllWindows()
```

●실행 중 내용

read success!
집중한 상태
read success!
집중한 상태
read success!
집중한 상태
... (생략)...
read success!
집중한 상태
read success!
집중한 상태
30초간 엎드려 있네요!!!

●종료시 화면



6. 요약정리

집중 감지 AI 프로젝트를 진행하여, 사용자의 집중 이외 상태를 감지하고 알림을 받을 수 있었다. 집중 상태 여부를 판단하기 위해 구글에서 제공하는 딥러닝 알고리즘을 활용했다. 구글의 Teachable Machine 덕분에 복잡한 딥러닝 모델을 쉽게 만들 수 있었다.모델을 통해 예측된 결과를 통해, 집중하지 않는 상태가 지속되면 알람을 전달하여 사용자의 정신을 깨우는 기능을 구현했다. 프로그램을 종료할 때는 그래프를 보여줌으로써 사용자의 집중력이 어느 시점에 약해지는지 확인할 수 있었다. 이를 통해 집중력에 방해되는 요소를 차단하고 도움을 받을 수 있었다.