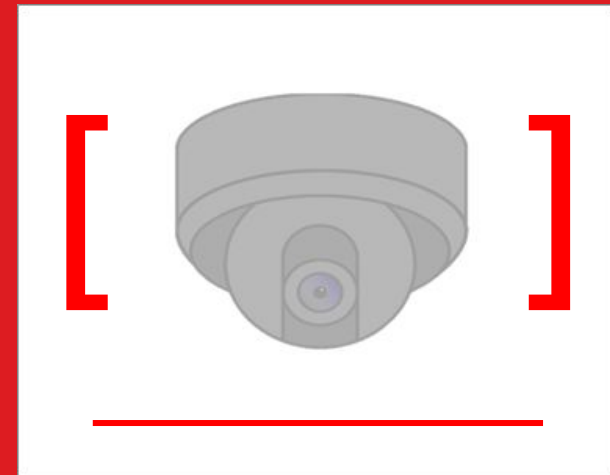


내 집 앞의 당신

이미지 인식 기술을 이용해 낯선 사람을
누구세요..?
알아채는 CCTV 어플



캡스톤디자인B_중간발표_알알이_15조

1771012 김수영

1772131 김수연

1771046 이혜진

목차

01. 개발 필요성

02. 작품 소개

03. 기존 서비스와의 차별점

04. 대표 기능

05. 시스템 구조도

06. 딥러닝 모델

07. 기대 효과



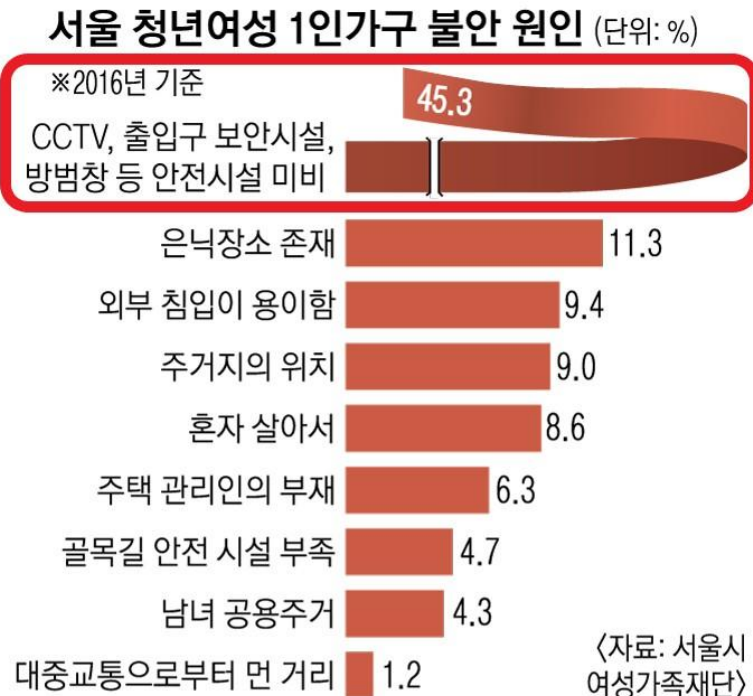
01 개발 필요성



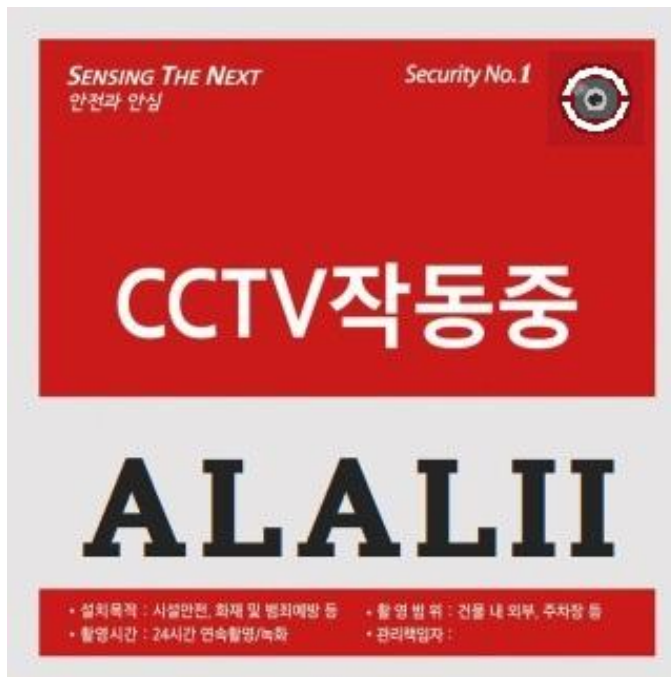
여성 노린 주거침입 범죄

지난 5년간 **1611건**

매년 증가하는 주거침입 성범죄
1인 가구의 불안 증가



02 작품 소개



AI CCTV

위험한 상황에서 실시간으로 낯선 사람을 감지하여 알리고 도움을 요청하는 서비스

이미지 학습을 통해 사용자의 지인과 낯선 사람 구분

+

실시간 영상 / 녹화 영상 시청 기능

+

SOS 버튼으로 도움 요청 기능

+

녹화 영상 / 추가 사진으로 추가 학습

03 기존 서비스와의 차별점

| | |
|---------|------------------|
| 녹화 대상 | 아기 / 반려동물 중심의 녹화 |
| 개인 CCTV | 집 내부의 상황 촬영 |
| 공동 CCTV | 녹화 사각지대 거주지 존재 |
| 서버 관리 | 녹화된 모든 영상 저장 |



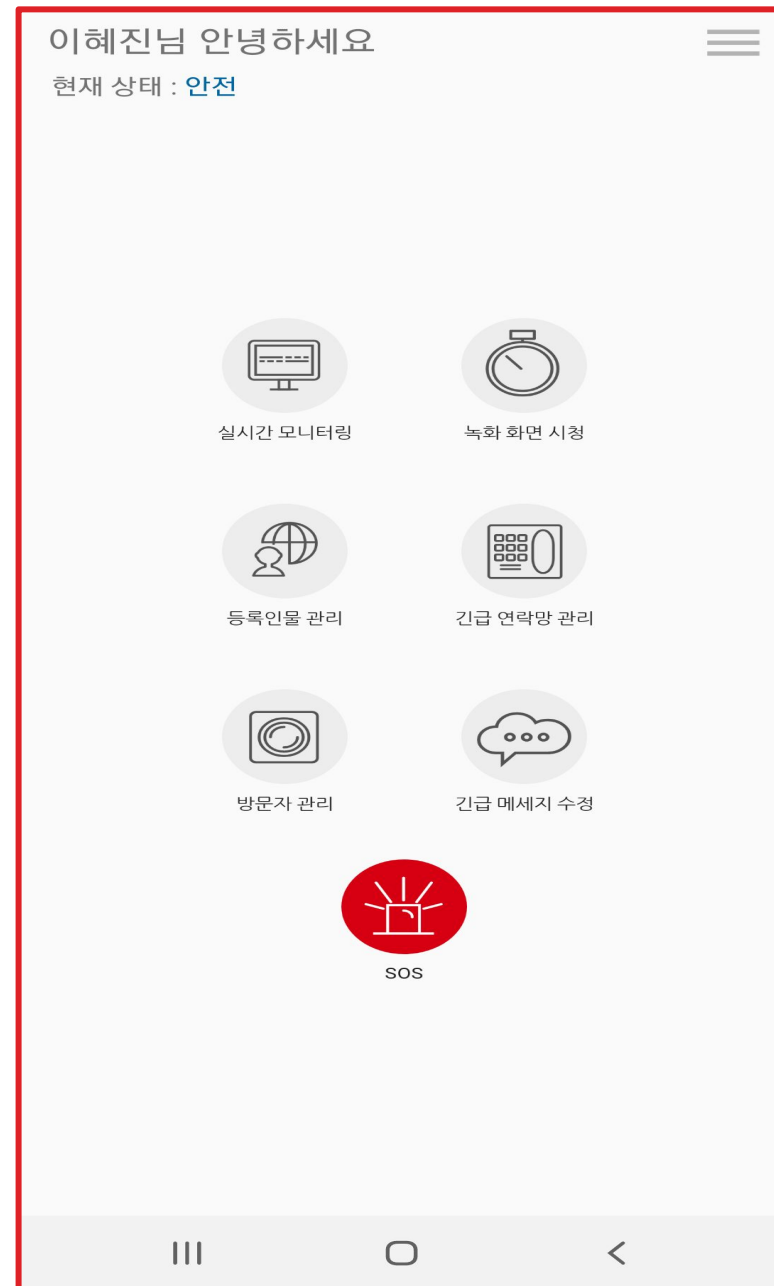
방문자 중심의 녹화
거주지 외부 상황 촬영
개인 CCTV
얼굴 녹화 영상만 저장

04 대표 기능

실시간 모니터링

등록 인물 관리

방문자 관리



녹화 화면 시청

긴급 연락망 관리

긴급 메시지 수정

도움 요청

04 대표 기능

1) 실시간 모니터링



녹화되고 있는 영상
실시간으로 시청

04 대표 기능

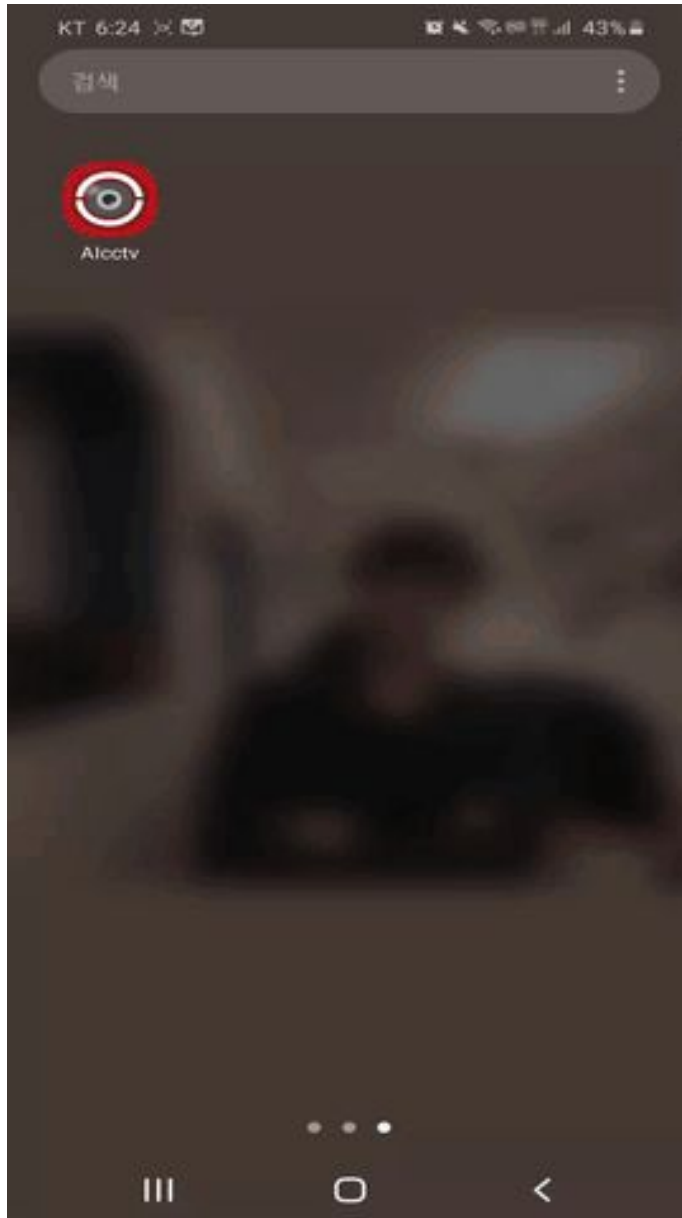
2) 녹화 영상 시청



녹화한 영상 시청
등장한 사람 이름 함께
제시

04 대표 기능

3) 등록 인물 관리



낯선 사람이 아닌
아는 얼굴로 구분할
사람 & 얼굴 사진 등록

04 대표 기능

4) 긴급 연락망 관리



위급한 상황에 연락할
전화번호 등록

04 대표 기능

5) 방문자 관리



녹화 영상 속 검출된
얼굴 사진 사람별 저장

- if) 인식된 사람이 아닌 경우,
1. 다른 사람으로 이름 수정
 2. 사진 삭제

04 대표 기능

6) 긴급 메시지 수정



위급한 상황에 전송될
메시지 내용 등록

04 대표 기능

7) 도움 요청



위급한 상황 시
도움 요청

04 대표 기능

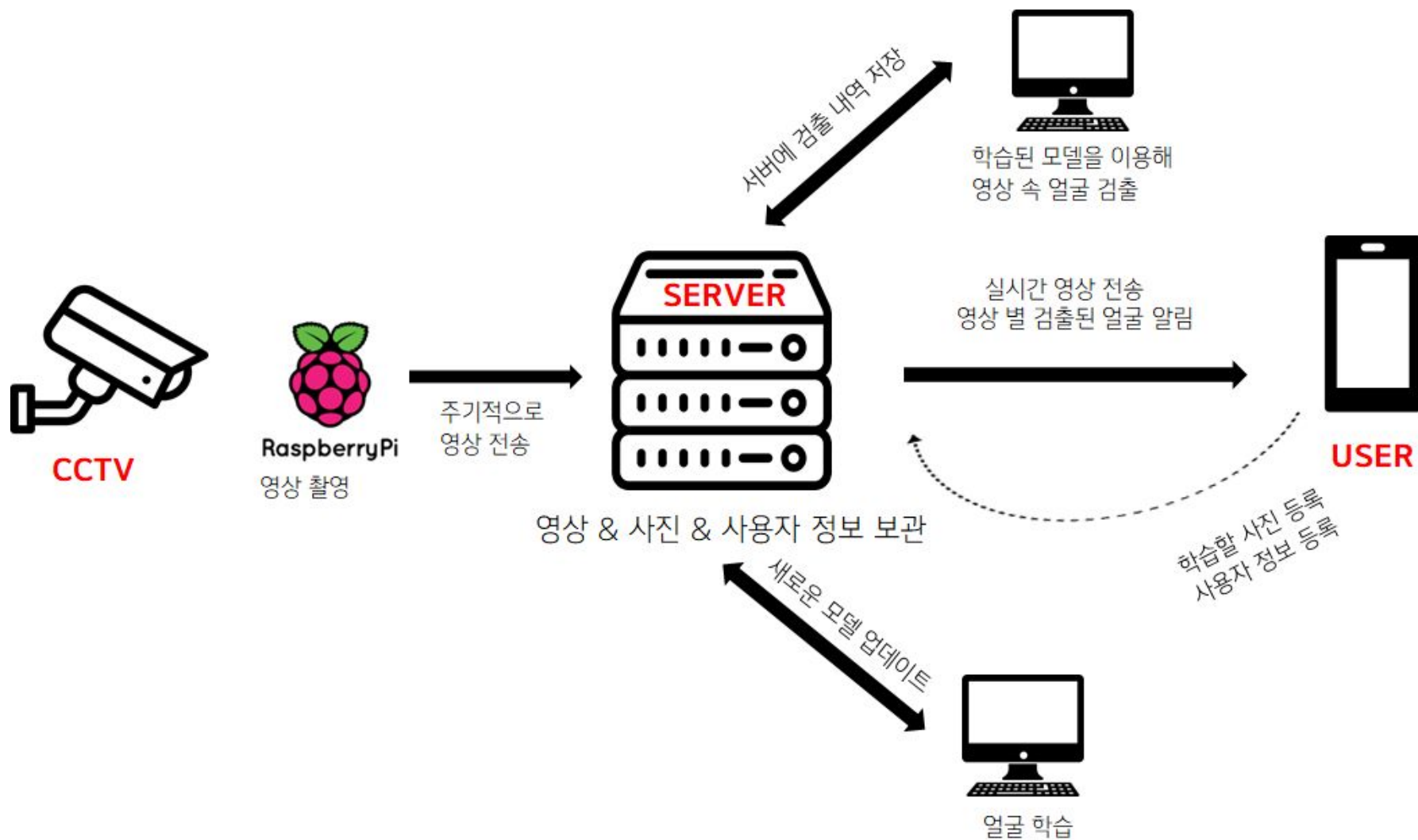
8) 추가 학습



추가 등록된 지인 얼굴
사진

➡ 녹화된 영상 속 사진
추가적 이미지 학습

05 시스템 구조도



06 딥러닝 모델



50장의 사진



70%는
Training

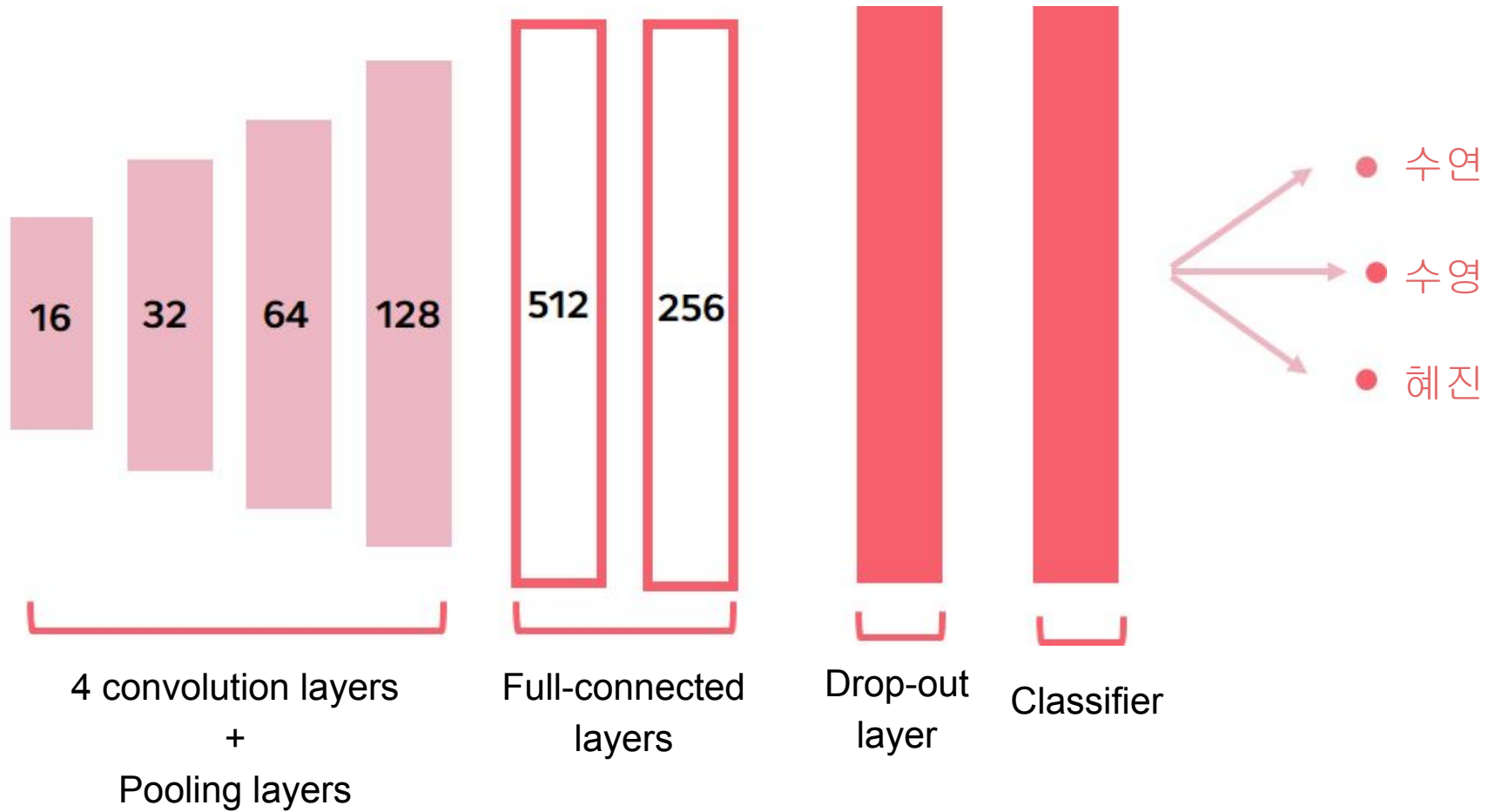
30%는
Validation

Image Augmentation을
이용해 각 사진을 8장씩 늘림

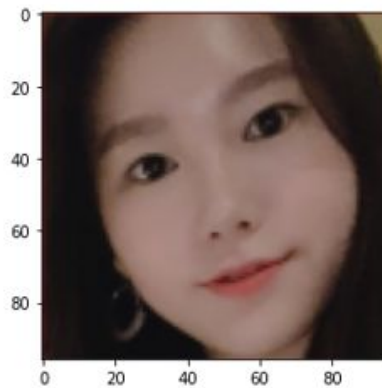
06 딥러닝 모델



96x96 RGB

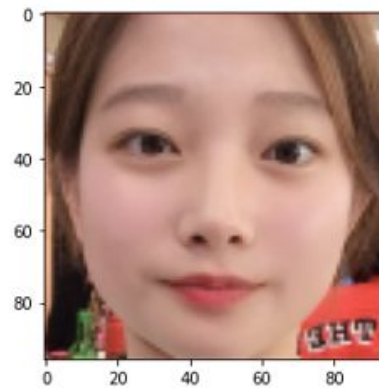


06 딥러닝 모델



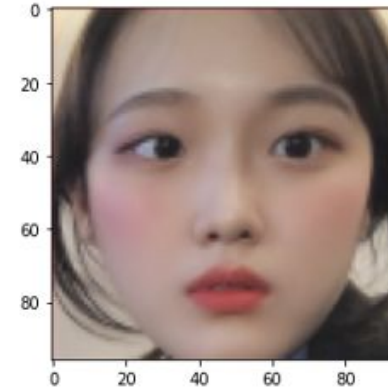
heajin 0.000001
suyeon 0.001354
sooyoung 0.998645

예측 결과: sooyoung



heajin 0.428717
suyeon 0.570943
sooyoung 0.000340

예측 결과: suyeon



heajin 0.740415
suyeon 0.000171
suyeon 0.259414

예측 결과: heajin

```
## time: 3:49:35.116777 steps: 3210  
Prediction loss: 0.06427305 accuracy: 0.98  
Validation loss: 0.72777575 accuracy: 0.91
```

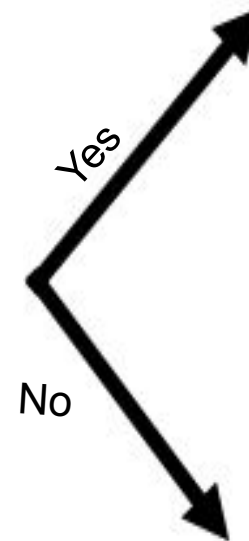
-> 약 4시간 동안 3200 스텝의 학습 과정을 거쳐
90%이상의 정확도 구현

06 딥러닝 모델

낯선 사람(Unknown) 검출 메커니즘



영상 속, 얼굴이 검출된
프레임 개수의 85% 이상을
일관된 사람으로 인식하는가?



등록된 인물



낯선 사람
(Unknown)

07 기대 효과



위급상황에 대한 적절한
대처



서버 관리 용이

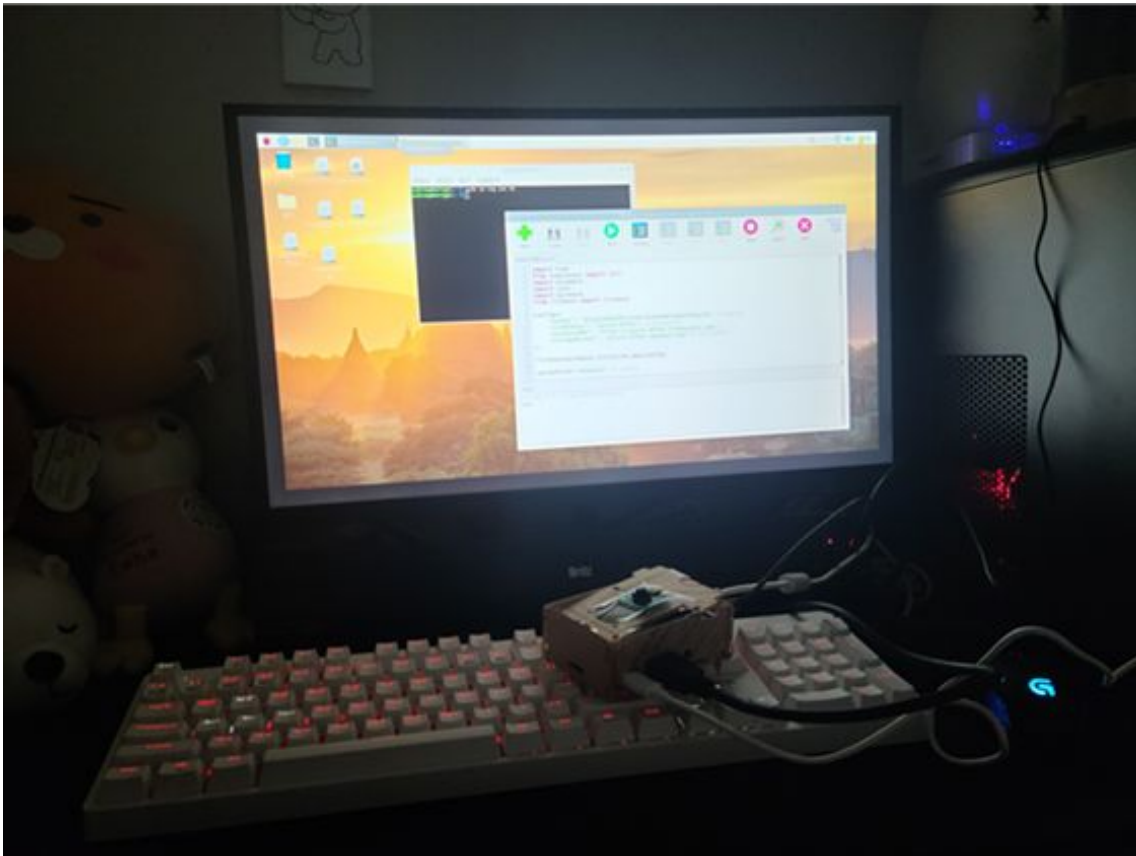


심리적 안정감 제공



Q & A

부 록



추가적인 사항들

1. 라즈베리파이 개발 환경
2. 라즈베리파이 개발 과정
3. 서버 구조
4. 깃허브 관리 과정
5. 안드로이드 UI 디자인
6. 안드로이드 클래스 및 구조
7. 파이썬 개발 코드
8. 역할 분담
9. 앞으로의 계획

라즈베리파이 개발 환경



- 기본적인 촬영 환경을 세팅하기 위해 모니터, 키보드, 마우스를 이용
- 무선랜 사용(WIFI) 및 전원 공급을 위해 보조 배터리 이용
상용화 할 경우에는 보조 배터리가 아닌 유선 전원 연결이 필요할 것으로 예상
- 와이파이를 사용하기 위해서 부팅시 지역 설정 필요 (sudo iw reg set US)
- 라즈베리파이 내의 개발은 파이썬 2.4 버전을 통해서 이루어짐

라즈베리파이 개발 과정

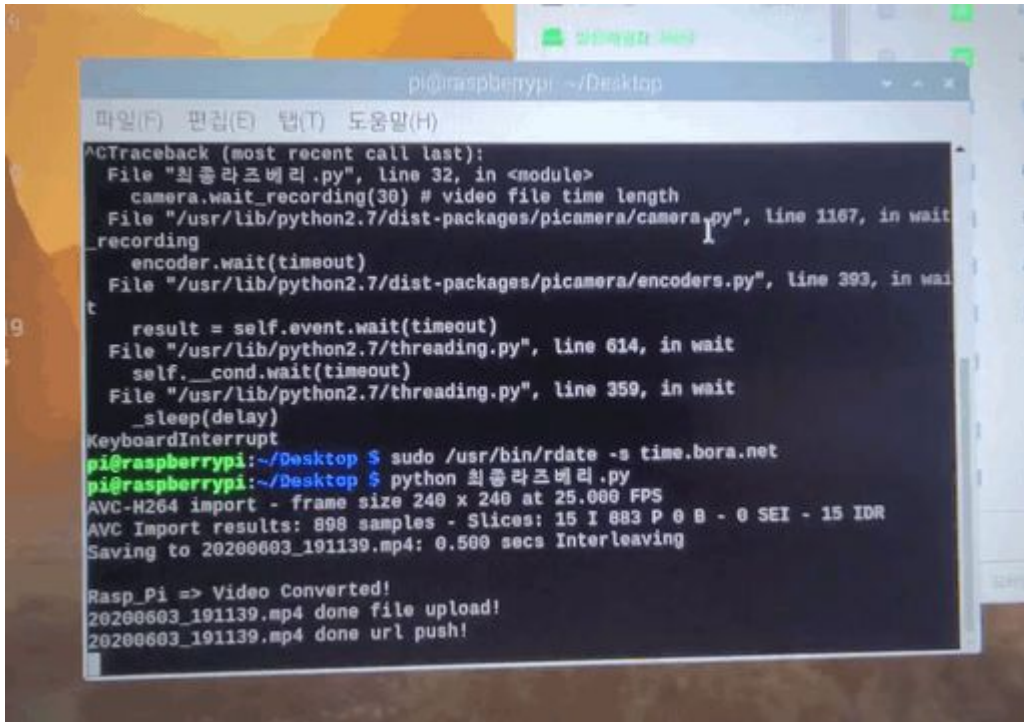
```
pi@raspberrypi: ~/Desktop
파일(F) 편집(E) 탭(T) 도움말(H)
pi@raspberrypi:~ $ sudo iw reg set US
pi@raspberrypi:~ $ cd /home/pi/Desktop
pi@raspberrypi:~/Desktop $ python 최종라즈베리.py
AVC-H264 import - frame size 240 x 240 at 25.000 FPS
AVC Import results: 898 samples - Slices: 15 I 883 P 0 B - 0 SEI - 15 IDR
Saving to 20200525_194008.mp4: 0.500 secs Interleaving

Rasp_Pi => Video Converted!
20200525_194008.mp4 done file upload!
20200525_194008.mp4 done url push!
AVC-H264 import - frame size 240 x 240 at 25.000 FPS
AVC Import results: 899 samples - Slices: 15 I 884 P 0 B - 0 SEI - 15 IDR
Saving to 20200525_194042.mp4: 0.500 secs Interleaving

Rasp_Pi => Video Converted!
20200525_194042.mp4 done file upload!
20200525_194042.mp4 done url push!
```

- 현재 라즈베리파이에서는 영상을 30초 단위로 촬영해 mp4로 변환 후 서버로 전송
- 영상은 30초 영상 전송을 기준으로 평균적으로 2~4초의 지연이 발생
194008, 194042 : 31초의 영상이며 3초의 지연 시간 발생
- 와이파이 환경이 좋지 않았을 때에는 지연 시간이 더욱 길었으나 이를 개선한 이후로 지연 시간 단축 성공 → 유선으로 연결할 경우 지연시간을 더욱 단축시킬 수 있을 것으로 예상
- 파이가 찍은 영상은 240 X 240 해상도를 갖는 영상으로 상용화 할 경우 더 좋은 성능의 카메라 필요

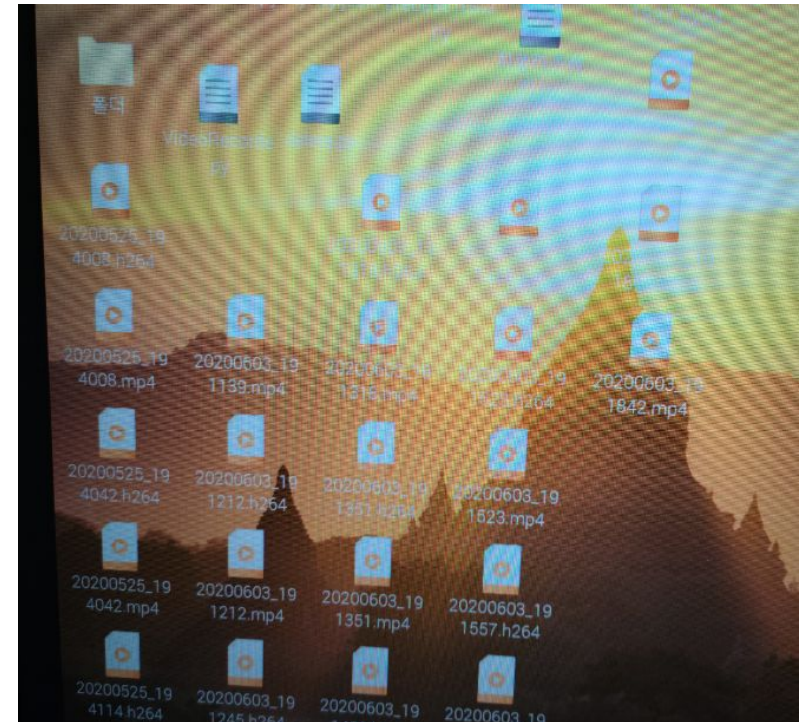
라즈베리파이 개발 과정



A terminal window on a Raspberry Pi showing the execution of a script. The script converts a video file to H264 format and uploads it to a server. The output shows the video is saved and then uploaded successfully.

```
pi@raspberrypi ~/Desktop
파일(F) 편집(E) 탭(T) 도움말(H)
^CTraceback (most recent call last):
  File "최종라즈베리.py", line 32, in <module>
    camera.wait_recording(30) # video file time length
  File "/usr/lib/python2.7/dist-packages/picamera/camera.py", line 1167, in wait_recording
    encoder.wait(timeout)
  File "/usr/lib/python2.7/dist-packages/picamera/encoders.py", line 393, in wait
    result = self.event.wait(timeout)
  File "/usr/lib/python2.7/threading.py", line 614, in wait
    self._cond.wait(timeout)
  File "/usr/lib/python2.7/threading.py", line 359, in wait
    _sleep(delay)
KeyboardInterrupt
pi@raspberrypi:~/Desktop $ sudo /usr/bin/rdate -s time.bora.net
pi@raspberrypi:~/Desktop $ python 최종라즈베리.py
AVC-H264 import - frame size 240 x 240 at 25.000 FPS
AVC Import results: 898 samples - Slices: 15 I 883 P 0 B - 0 SEI - 15 IDR
Saving to 20200603_191139.mp4: 0.500 secs Interleaving

Rasp_Pi => Video Converted!
20200603_191139.mp4 done file upload!
20200603_191139.mp4 done url push!
```



(좌) 실제 라즈베리파이가 실행되는 모습

(우) 실제 영상이 배경화면에 저장되는 모습

서버 구조

aicctv-8f5ac

00gpwls00

Email: "00gpwls00@naver.c

FaceResult

20200603_191523: "혜진"

20200603_191842: "혜진"

20200603_200910: "unknowr"

20200603_220624: "unknowr"

ID: "leeheajin9"

Learning: 1

Nickname: "이혜진"

Password: "dlgpwls9"

PhotoLink

수연

20200603_184743_1: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_10: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_11: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_2: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_3: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_4: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_5: "https://firebasestorage.googleapis.com/v0/b,

20200603_184743_6: "https://firebasestorage.googleapis.com/v0/b,

DownCount: 7

UpdateCount: 7

oneface: 7

State: "안전"

VideoLink

20200603_191139: "https://firebasestorage.googleapis.com/v0/b,

20200603_191212: "https://firebasestorage.googleapis.com/v0/b,

20200603_191245: "https://firebasestorage.googleapis.com/v0/b,

20200603_191318: "https://firebasestorage.googleapis.com/v0/b,

20200603_191351: "https://firebasestorage.googleapis.com/v0/b,

20200603_191423: "https://firebasestorage.googleapis.com/v0/b,

20200603_191523: "https://firebasestorage.googleapis.com/v0/b,

20200603_191842: "https://firebasestorage.googleapis.com/v0/b,

20200603_220624: "https://firebasestorage.googleapis.com/v0/b,

Download: "20200603_220624"

Update: "20200603_220624"

VideoPhoto

- 해당 프로젝트에서는 파이어베이스의 실시간 데이터베이스와 스토리지를 모두 사용
- 각 회원은 아이디, 이메일 주소, 패스워드, 닉네임을 갖고 있으며 구글 아이디를 통해 로그인
- 이 때 닉네임은 메뉴 화면과 메인 화면에 등장하고 이러한 정보들은 변경 가능
- PhotoLink는 이용자가 등록인물관리 메뉴를 통해 등록한 사진으로 접근 할 수 있는 링크를 저장
- 이용자가 메뉴 → 학습 하기 버튼을 누르면 Learning 값이 1로 변경되고 학습을 시작

서버 구조

```
VideoPhoto
├── unknown
│   └── test: "http://"
├── 수연
│   └── test: "http://"
└── 혜진
    ├── 1_1: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_10: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_11: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_12: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_2: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_3: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_4: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_5: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_6: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_7: "https://firebasestorage.googleapis.com/v0/b,"
    ├── 1_8: "https://firebasestorage.googleapis.com/v0/b,"
    └── 1_9: "https://firebasestorage.googleapis.com/v0/b,"
```

```
1_2: "https://firebasestorage.googleapis.com/v0/b,"
1_3: "https://firebasestorage.googleapis.com/v0/b,"
1_4: "https://firebasestorage.googleapis.com/v0/b,"
1_5: "https://firebasestorage.googleapis.com/v0/b,"
1_6: "https://firebasestorage.googleapis.com/v0/b,"
1_7: "https://firebasestorage.googleapis.com/v0/b,"
1_8: "https://firebasestorage.googleapis.com/v0/b,"
1_9: "https://firebasestorage.googleapis.com/v0/b,"
3_1: "https://firebasestorage.googleapis.com/v0/b,"
3_10: "https://firebasestorage.googleapis.com/v0/b,"
3_11: "https://firebasestorage.googleapis.com/v0/b,"
3_12: "https://firebasestorage.googleapis.com/v0/b,"
3_2: "https://firebasestorage.googleapis.com/v0/b,"
3_3: "https://firebasestorage.googleapis.com/v0/b,"
3_4: "https://firebasestorage.googleapis.com/v0/b,"
3_5: "https://firebasestorage.googleapis.com/v0/b,"
3_6: "https://firebasestorage.googleapis.com/v0/b,"
3_7: "https://firebasestorage.googleapis.com/v0/b,"
3_8: "https://firebasestorage.googleapis.com/v0/b,"
```

- **Update**는 영상이 업데이트 될 때 마다 갱신되며, **Download**는 로컬에서 검출을 위해 다운로드 받을 때 마다 갱신 → 두 개의 변수를 바탕으로 동영상을 주기적으로 다운로드하고 얼굴을 검출
- **State**는 현재 위험도를 의미하며 집 앞에 **unkown**의 빈도가 높을 경우 상태를 위험으로 변경
- **VidePhoto**는 집 앞의 영상을 다운로드 받고 검출하는 과정에서 이를 다시 학습에 이용하기 위해서 저장한 사진 스토리지에 접근하기 위한 주소값 → 방문자 관리 메뉴를 통해 삭제 및 수정이 가능
검출한 모든 사진을 저장하며 각 영상은 고유한 넘버를 갖고 있음

깃허브 관리 과정 (Android)

| | | |
|------------------------|------------------|-----------------------------------|
| Sooyoungg final update | | Latest commit 3b804ec 6 hours ago |
| 📁 .idea | UI update | 2 days ago |
| 📁 app | final update | 6 hours ago |
| 📁 gradle/wrapper | UI update | 2 days ago |
| 📄 .gitignore | crop images | 3 months ago |
| 📄 README.md | Create README.md | last month |
| 📄 build.gradle | UI update | 2 days ago |
| 📄 gradle.properties | crop images | 3 months ago |
| 📄 gradlew | crop images | 3 months ago |
| 📄 gradlew.bat | crop images | 3 months ago |
| 📄 settings.gradle | crop images | 3 months ago |

📖 README.md



AI_CCTV

얼굴인식 기능이 탑재된 CCTV

주요 기능

- 실시간 모니터링: 외출 시에도 대문 앞의 상황을 바로 확인할 수 있도록 언제나 사용자에게 대문 앞의 상황을 모니터링 해준다
- 지인 얼굴 등록: 지인의 사진을 등록해 등록되지 않은 사람이 대문 앞에 포착될 경우 사용자에게 알림 전송
- 긴급 연락: 위급 상황시 지인이나 112 119에 자동으로 연락이 취해질 수 있도록 함

깃허브 관리 과정 (Python)

| | | |
|-----------------------------------|--|-----------------------------------|
| 🌱 leeheajin Add files via upload | | Latest commit e1b0ca8 2 hours ago |
| 📁 face_recognition | Add files via upload | 5 days ago |
| 📄 DownloadPhoto.py | Add files via upload | 3 months ago |
| 📄 DownloadPhoto.txt | Add files via upload | 3 months ago |
| 📄 DownloadVideo&DetectFace.ipynb | Add files via upload | 5 days ago |
| 📄 Download_Video.ipynb | Download_Video | 9 days ago |
| 📄 FullVersion.py | Add files via upload | 3 months ago |
| 📄 README.md | Update README.md | 3 months ago |
| 📄 delete.py | Add files via upload | 9 days ago |
| 📄 down_oneFace.ipynb | find & download images which have only one face | 3 months ago |
| 📄 face_recognition.ipynb | Add files via upload | 2 months ago |
| 📄 opencv_findface&count.ipynb | Add files via upload | 2 months ago |
| 📄 updata_oneFaceCount_FB_DB.ipynb | update the number of images which have only one face on realtime data... | 3 months ago |
| 📄 누르면사진다운최종.ipynb | Add files via upload | 6 hours ago |
| 📄 수영 test.ipynb | Add files via upload | 3 days ago |
| 📄 수영_최종.ipynb | 최종 python 코드 | 2 days ago |
| 📄 얼굴검출&동영상다운&사진올리... | Add files via upload | 2 hours ago |

안드로이드 디자인 UI

- 메인화면 아이콘



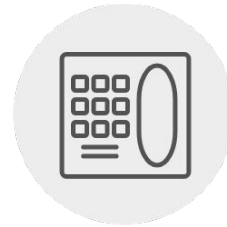
실시간 모니터링



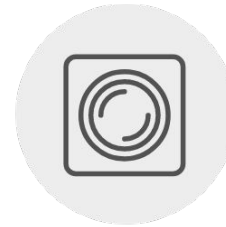
녹화 화면 시청



등록인물 관리



긴급 연락망 관리



방문자 관리

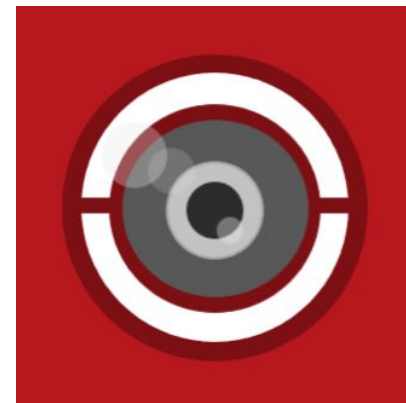


긴급 메시지 수정

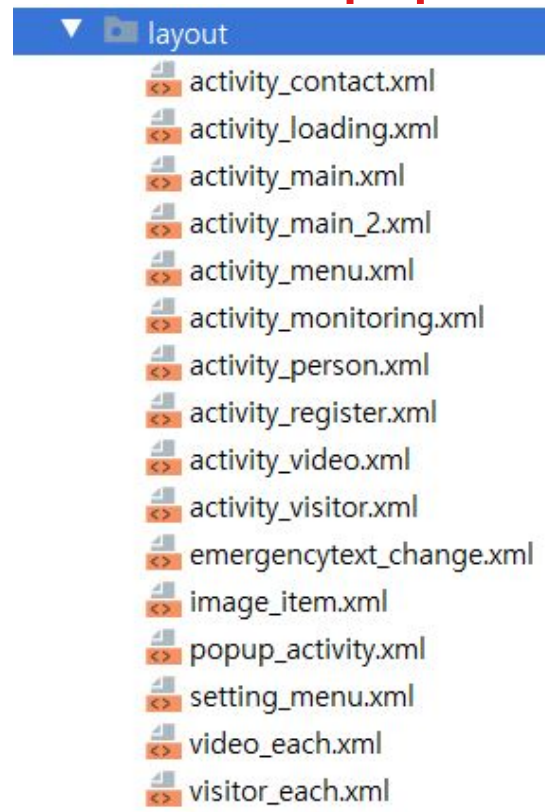
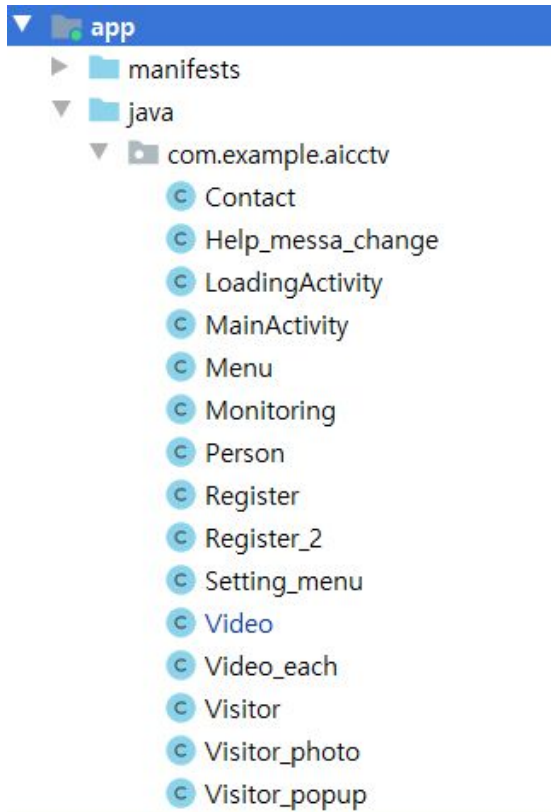
- 로딩 화면 배너



- 앱 아이콘



안드로이드 클래스 및 구조



- 안드로이드 자바 클래스 및 레이아웃 파일들은 위와 같음
- 기능 구현과 디자인 파트로 **공평하게** 나누어서 개발이 진행되었음
- 가상 머신을 사용하지 않고 안드로이드 10 버전을 이용하는 스마트폰을 이용해 개발
- 안드로이드 스튜디오 4.0 버전으로 통합해서 개발 진행
- 깃허브 **AI_CCTV 프로젝트** 내에서 푸시 및 커밋 과정을 통해 협업 진행

파이썬 개발 코드

```
In [2]: import cv2
import firebase_admin
import re
import json
import time
import urllib.request
import os
from firebase_admin import credentials, firestore, storage
from firebase_admin import db
from subprocess import call
import pyrebase
from firebase import firebase

# Firebase database 인증 및 앱 초기화
cred = credentials.Certificate('myKey.json')
firebase_admin.initialize_app(cred,{
    'databaseURL': "https://aicctv-8f5ac.firebaseio.com/",
    'storageBucket': "aicctv-8f5ac.appspot.com"
})

config={
    "apiKey": "AlzaSyDAA2XR7x3iNsx9cQdhH6FVw8dY3Q3grFU", # webkey
    "authDomain": "aicctv-8f5ac", # projectId
    "databaseURL": "https://aicctv-8f5ac.firebaseio.com/",
    "storageBucket": "aicctv-8f5ac.appspot.com", # storageURL
    "serviceAccount": "myKey.json"
}
fb=pyrebase.initialize_app(config)

storage2=fb.storage()
database=fb.database()

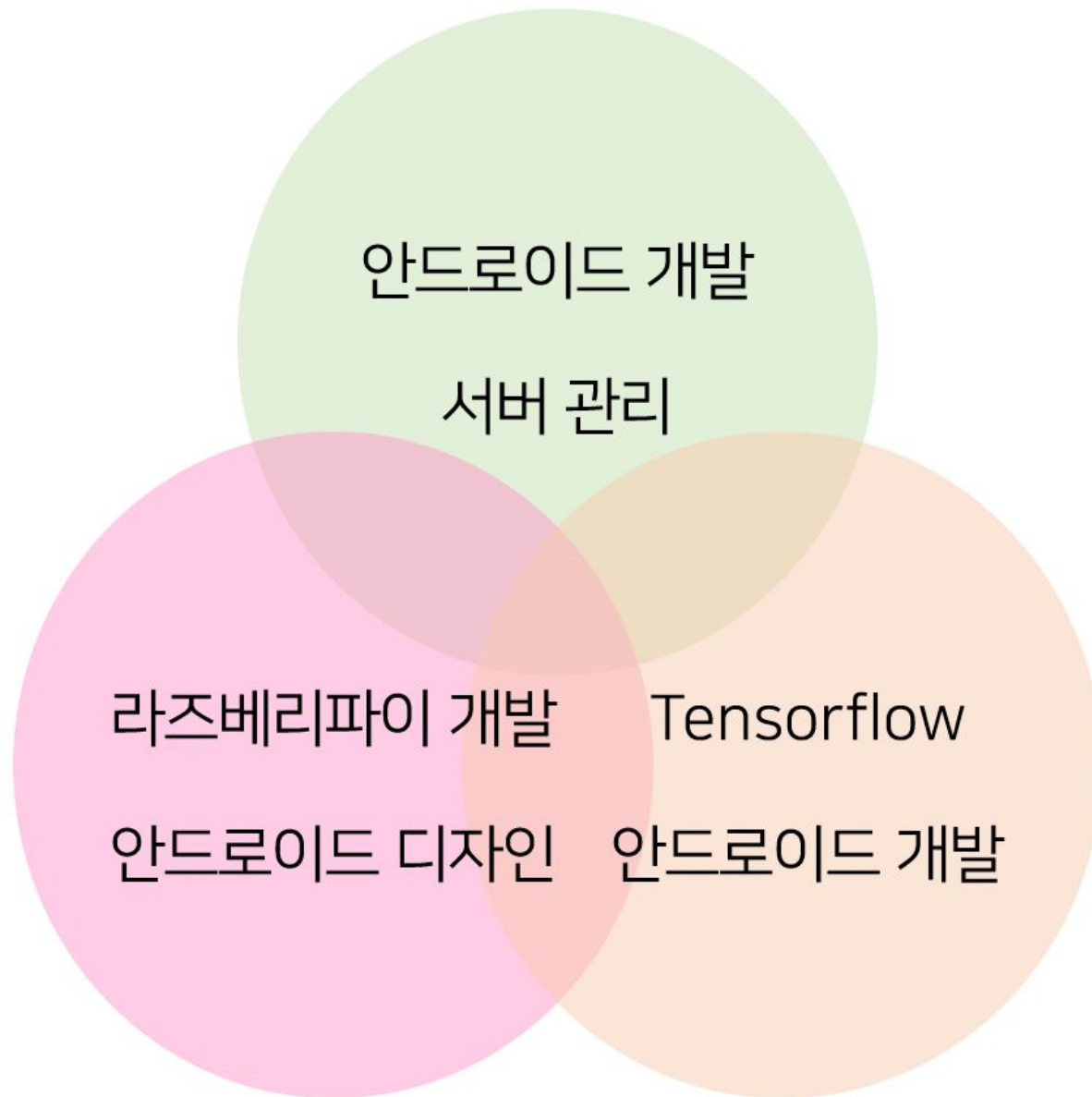
ID = "00gw1s00" # 이용자마다 다르게 코딩해야 하는 부분
ref = db.reference(ID+"/VideoLink") #db 위치 지정

In []: import datetime
# Import UUID4 to create token
from uuid import uuid4

path = 'C:\Users\leeheajin\videophoto'
```

- 파이에서 실시간 영상을 촬영해 전송하는 1개의 코드 / 이용자가 버튼을 누르면 사진을 받아와 재학습 시키는 코드 / 학습된 데이터를 바탕으로 영상에서 얼굴을 검출하고 서버로 전송하는 코드
→ 크게 3개의 코드로 분류
- 깃허브 AI_CCTV2 프로젝트에 모든 코드와 중간 산출물들이 업로드 되어 있음
- https://github.com/leeheajin/AI_CCTV_2.git

역할 분담



앞으로의 진행 계획

| | |
|--------|-----------------|
| 6월 04일 | 중간발표 진행 |
| 6월 05일 | 포스터 제작 및 시연 영상 |
| 6월 07일 | 최종 포스터 제출 |
| 6월 10일 | 최종 보고서 |
| 6월 11일 | 포스터 세션 진행 |
| 6월 16일 | 멘토링 서류 제출 |
| 6월 25일 | 결과물 및 최종 시연 비디오 |

- 현재 총 4회의 멘토링(방학 포함)을 진행하였으며 20회 이상의 회의를 거쳤음
- 6월 4일 최종발표 전 PPT에 대한 피드백을 받을 예정
- 현재 모든 기능을 구현하였음; 초기 프로젝트에서 계획한 것들보다 더욱 많은 내용을 개발함
- 최종 보고서를 매우 자세하게 작성하고 멘토링을 통해 피드백 받을 예정
- 멘토링 서류를 꼼꼼히 검토해 정해진 기한 내에 제출할 수 있도록 함