VIP Detection Sensor

딥러닝을 활용한 영상에서의 특정 인물 인식

Agenda

01

주제 소개

주제 선정 이유, VIP Detection Sensor 소개, 프로젝트 일정

02

모델 구현

데이터 수집, 데이터 정제, 모델 훈련, 최적화, 최종 모델

03

시뮬레이션

테스트 영상 구현

04

결론

프로젝트 의의

주제 선정

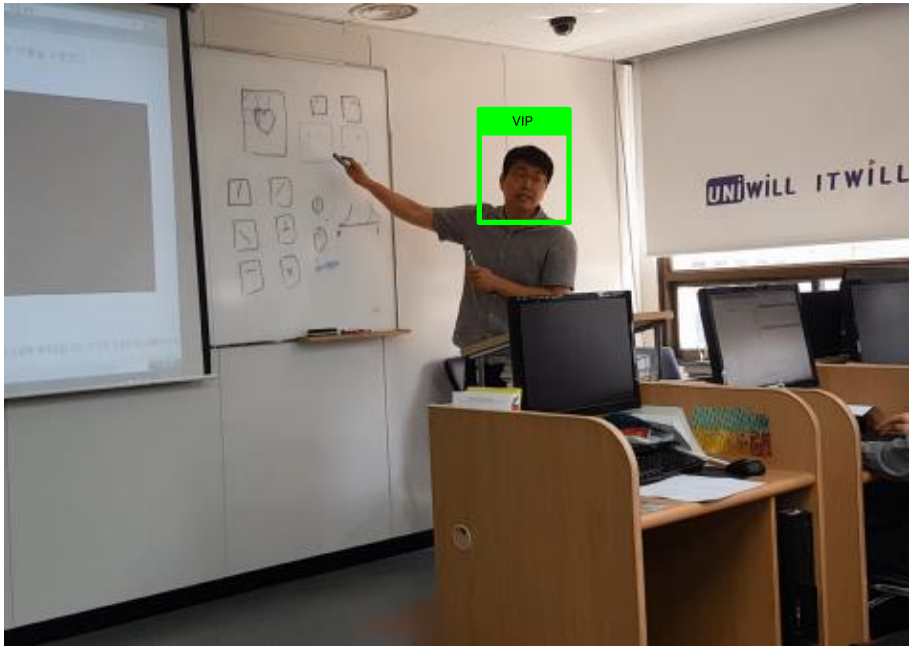
Twin or Not?

지난 2017년 Apple의 IPHONE X 가 출시되고, 얼굴인식으로 관리하는 Face ID가 쌍둥이를 구분해낼 수 있는지에 대한 이슈가 있었습니다.

과연 영상에서 특정인물을 인식하여 다른 사람과 분류하고, Face ID 이슈처럼 유사 인물과 구분하여 식별할 수 있는지 구현해보고 싶었습니다.

VIP Detection Sensor 소개

VIP Detection Sensor란,
라벨링한 특정인물을 다른 사람과 구별하여 영상에서 인식하는 모델을 말합니다.



VIP Detection Sensor 특징

- 실시간 타겟 감지가 가능합니다. (휴대용 웹캠 이용)
- 인식하고자 하는 타겟을 변경하여도 라벨 학습이 가능하도록 모델을 설계하였습니다.
- Face_recognition-OpenCV library를 기반으로 합니다.
  : 앞, 옆모습에 상관없이 얼굴을 잘 인식합니다.
- 모델의 정확도가 높고, GPU 기반으로 빠른 처리가 가능합니다.

프로젝트 일정

시간 자원을 적절히 분배하기 위해 프로젝트 일정을 만들어 진행하였습니다.
프로젝트 일정에 대해 Deadline을 설정하여 기한을 지키려 하였습니다.

데이터 수집
- 예상 소요 기간: 3일
- Deadline : 9/6
- 영상 촬영
- 웹크롤링

데이터 정제
- 예상 소요 기간: 2일
- Deadline : 9/8
- 데이터 전처리
① 사이즈 조정
② 불필요 파일 제거

모델 설계 및 구현
- 예상 소요 기간: 6일
- Deadline : 9/14
- GPU 환경 구축
- CNN 모델 설계
- 데이터 학습

최적화
- 마지막까지 계속 진행
- 데이터 수 변경
- CNN 망 깊이 변경
- Parameter 조정

PPT 제작 착수
- 예상 소요 기간: 3일
- D-3

모델 시물레이션
- D-2

마무리
- D-1

최종 완성
- 9/20

Agenda

01

주제 소개

주제 선정 이유, VIP Detection Sensor 소개, 프로젝트 일정

02

모델 구현

데이터 수집, 데이터 정제, 모델 훈련, 최적화, 최종 모델

03

시뮬레이션

테스트 영상 구현

04

결론

프로젝트 의의

모델 제작 과정

모델은 4가지의 프로세스를 통해 제작하였습니다.

데이터 수집

- OpenCV
영상에서 얼굴을 인식하고, Cropping 하여 데이터를 수집하였습니다.

- Google 웹크롤링을
통해 학습을 위한 얼굴 이미지 데이터를 수집하였습니다.

데이터 정제

- 전처리 (Python)
일괄적으로 데이터를 정제하기 위해 2가지 프로세스를 거치도록 코드로 구현하였습니다.

① 사이즈 조절
② 비정상 파일 제거

모델 설계 및 훈련

- Keras 기반 모델 설계
간결하고 직관적인 Keras를 프레임워크로 선택하였습니다.

- CNN 모델의 구조는
VGG-net을 기본으로 설계하였습니다.

모델 최적화

- Hyper-parameter 조정
BN, Dropout, 가중치 초기화 값을 변경하였습니다.

- 데이터 오버피팅 문제를
해결하기 위해 이미지 증식 기법을 사용하였습니다.

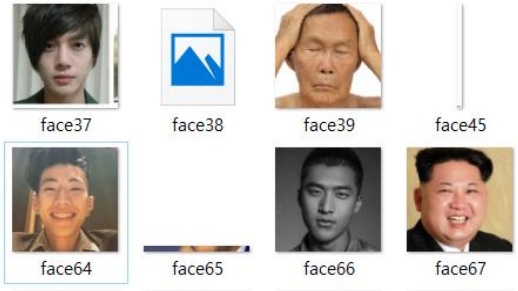
데이터 수집

데이터 수집은 라벨/비라벨로 나누어 진행하였습니다.

라벨 데이터

- 영상에서 OpenCV-Face_recognition을 통한 얼굴 인식 후 Cropping 단계를 거쳐 약10,000 장의 데이터 수집.
- 사진마다 다른 픽셀 크기를 일괄적으로 64x64 사이즈로 정제.

비라벨 데이터

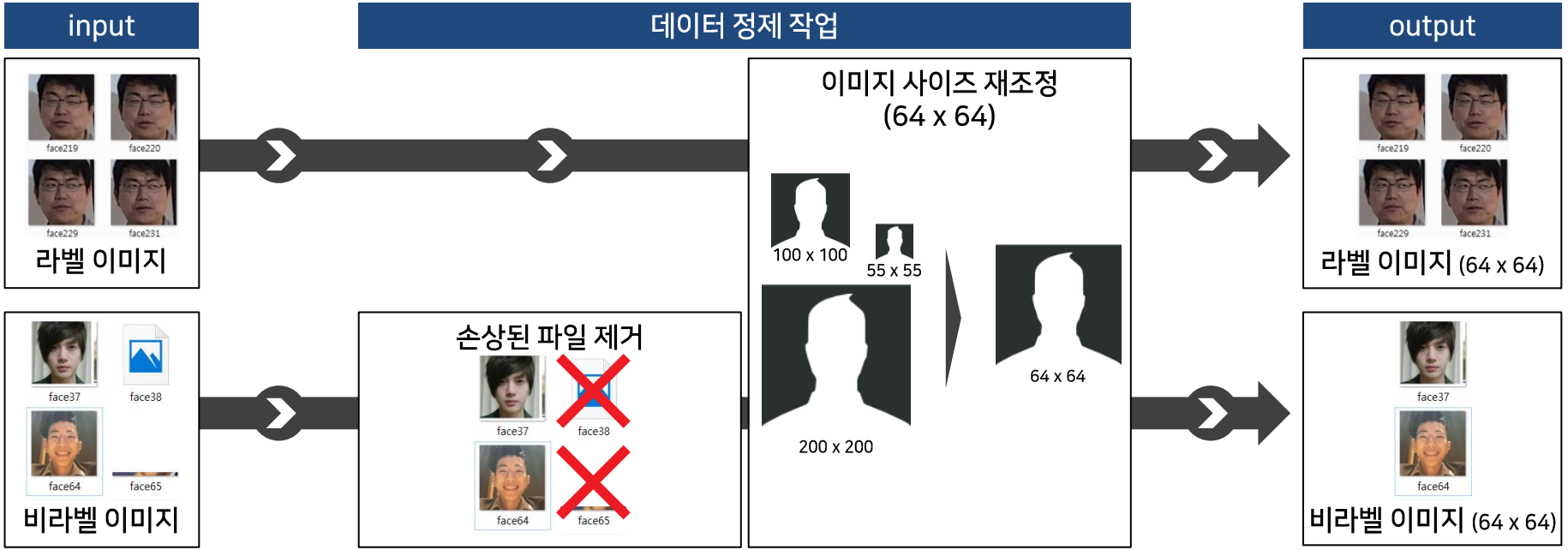


- Google 웹크롤링을 통해 약 8,000장 수집.

- "얼굴"로 검색된 이미지 중, 손상된 파일을 걸러내고 픽셀 사이즈를 64X64 사이즈로 정제.

데이터 정제

데이터 정제 작업으로 사이즈 재조정과 손상된 파일제거를 일괄적으로 진행하였습니다.



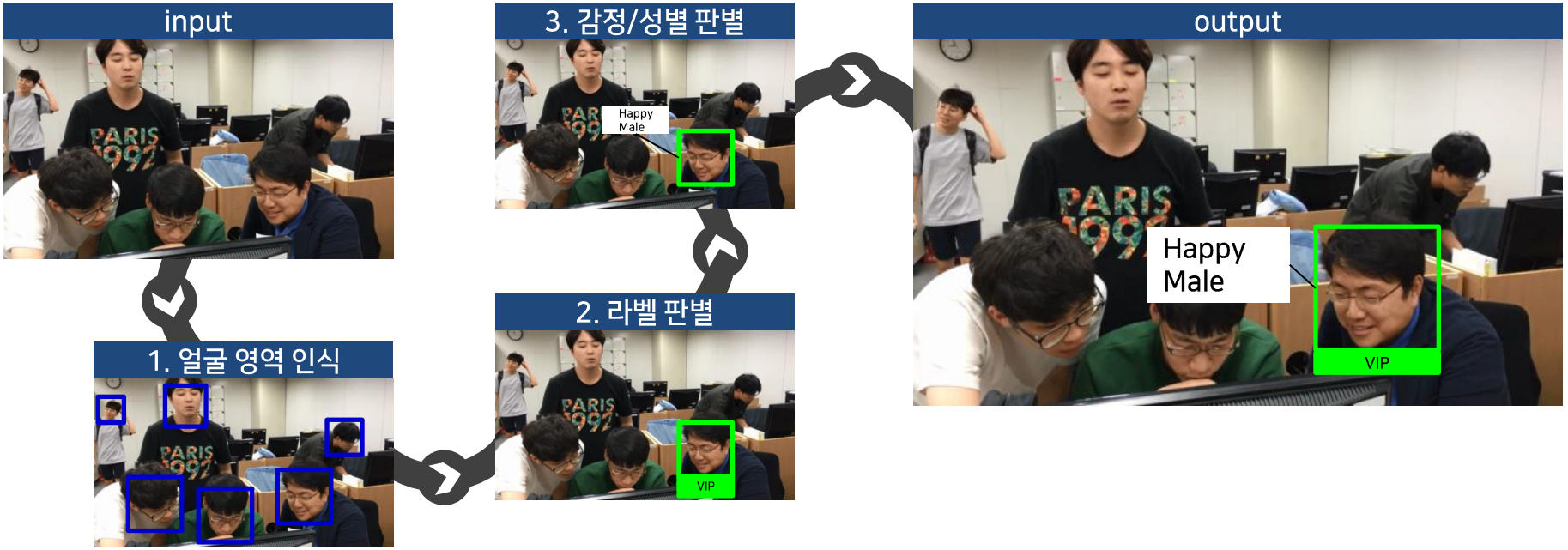
운영환경구축

VIP Detection Sensor 를 구현하기 위한 운영환경을 구축하였습니다.
시행착오와 트러블 슈팅 끝에, 가장 호환이 잘 되는 버전으로 설치하였습니다.

운영체제	GPU 환경	언어 환경	플랫폼 / 소프트웨어
Ubuntu 16.04	GEFORCE GTX 1080	Python 3.5.6	Anaconda Python 3.5 ver.
	NVIDIA CUDA toolkit9.0		OpenCV 3.43.18
	cuDNN		Dlib 19.15.99

모델 흐름도

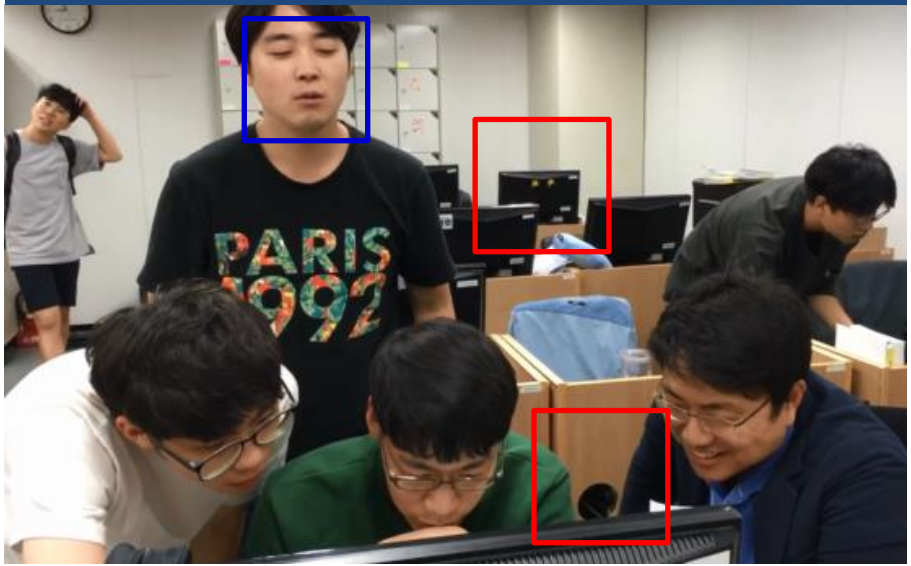
모델은 3가지의 프로세스로 진행됩니다.
얼굴 인식 후 라벨인지 판별하며, 동시에 감정/성별을 판별하는 프로세스가 진행됩니다.

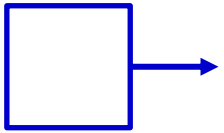


모델 흐름도_1.얼굴영역인식-오류

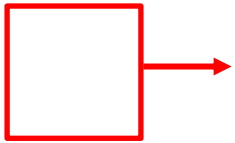
얼굴 인식 초기모델 OpenCV와 haarcascade frontalface XML 파일을 사용하였는데,
정면 얼굴만 인식하고 원형, 사각형의 사물을 인식하는 문제를 발생하였습니다.

VIDEO





정면 얼굴 출현시에만
얼굴영역을 인식.

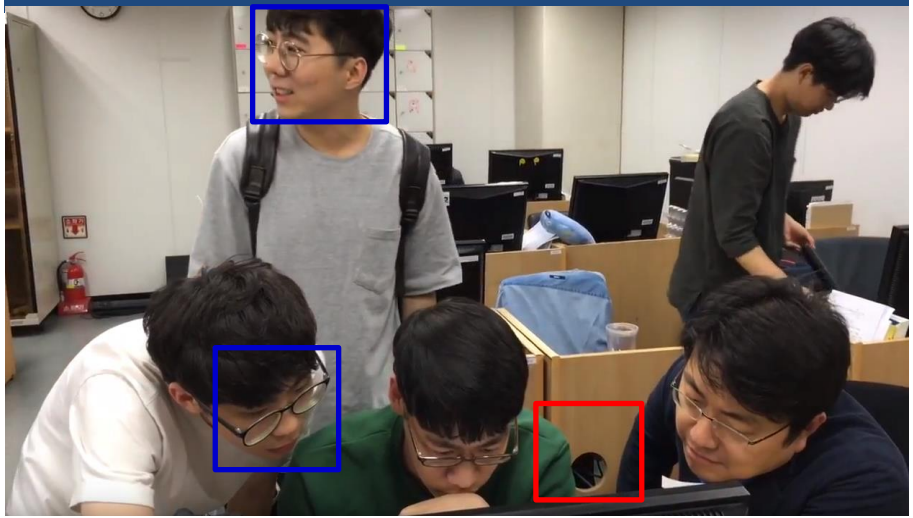


원형, 사각형의 사물이
얼굴로 인식.

모델 흐름도_1.얼굴영역인식-오류

다른 각도의 얼굴을 인식하기 위해 haarcascade profileface XML 파일을 추가 사용하였으나, 여전히 정확도가 떨어지고 연산속도가 느려지는 문제점을 발견하였습니다.

VIDEO

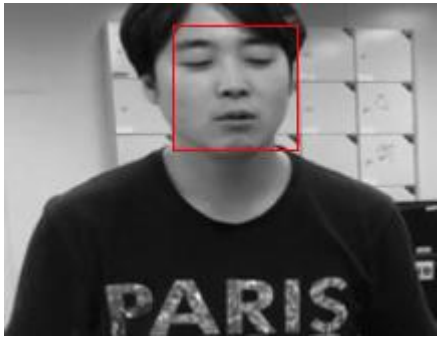


다른 각도의 얼굴 인식 성공
동시에 연산속도 증가.

여전히 사물을 얼굴로 인식.

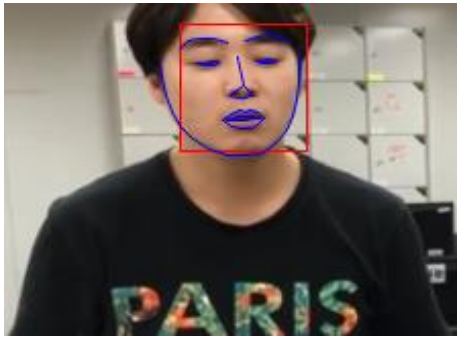
모델 흐름도_1.얼굴영역인식-오류해결

얼굴인식 문제는 Face_recognition 라이브러리를 통해 해결하였습니다.
Face_recognition은 HOG-Landmark-Embedding의 과정을 거쳐 작동합니다.



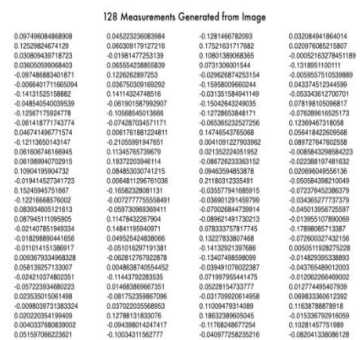
HOG

이미지의 픽셀값에 따라 그래디언트가 결정이 되고, 이미지에서 특징이 되는 부분만 표시되는데, 이때 얼굴 패턴과 비슷한 HOG 패턴을 찾아 얼굴영역을 인식합니다.



Landmark

얼굴에는 총 68개의 랜드마크가 존재하는데, 이를 중심으로 얼굴 영역을 인지하면 이미지가 어떤 각도에 있어도 얼굴을 인식할 수 있도록 합니다.

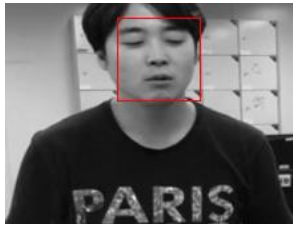


Embedding

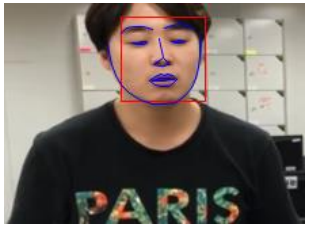
DCNN(Deep Convolutional Neural Network)을 통해 128개의 얼굴 고유정보를 이미지로부터 추출하도록 훈련을 받게되면, 이미지로부터 고유정보를 인식하여 해당하는 인물을 구분하게 됩니다.

모델 흐름도_1.얼굴영역인식-오류해결

Face_recognition의 얼굴인식 기능과 CNN 모델을 결합하여 라벨을 판별하였습니다.

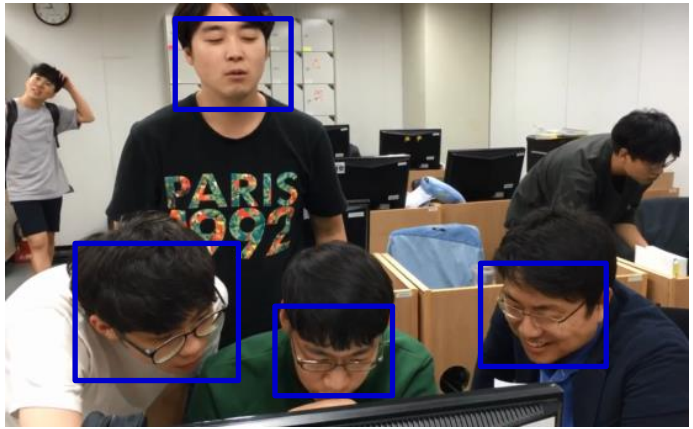


HOG



Landmark

+

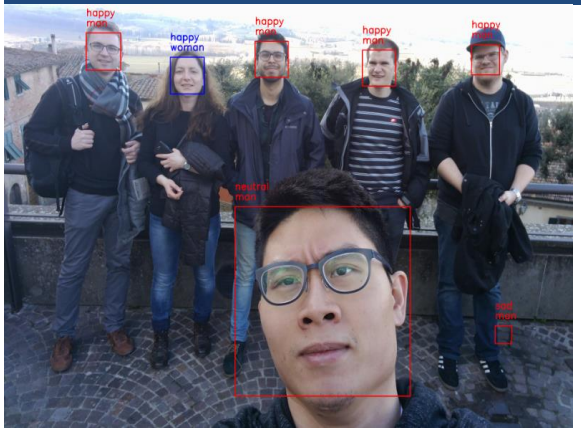


모델 흐름도_2.성별/감정 판별

성별/감정 판별은 오픈소스를 활용하여 적용하였습니다.
이를 통해 얼굴로 인식한 모든 오브젝트에 대하여 성별, 감정이 표시되도록 하였습니다.

VIDEO

Open Source



Face Classification

Gender
Man
Woman

Emotion
Happy
Neutral
Angry
Fear

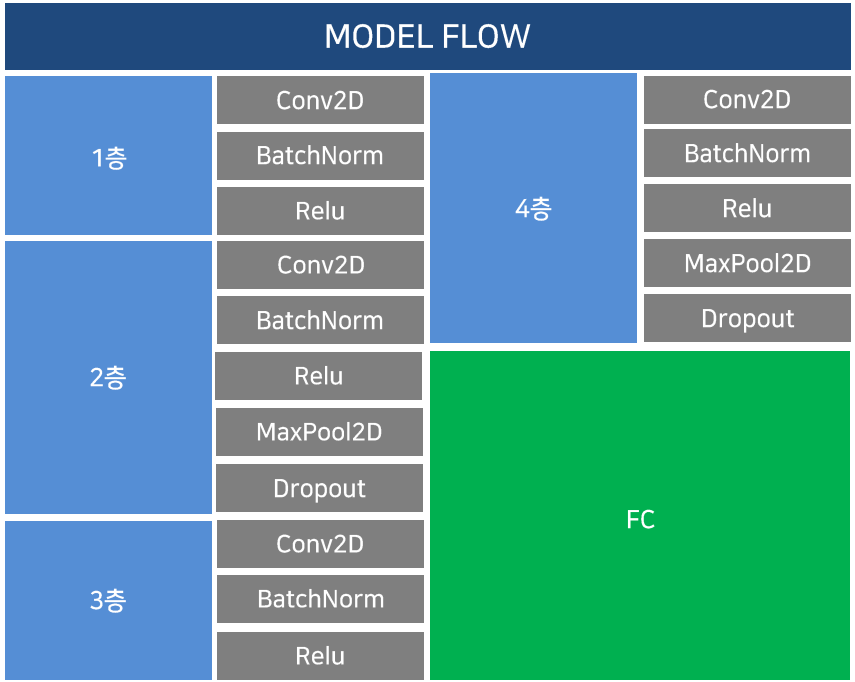
출처: https://github.com/oarriaga/face_classification

모델 구상 및 훈련 요약

CNN 모델 설계를 위해 프레임워크는 Keras를 선택하였고, VGG-Net 모델을 기반으로 최종 모델을 구현하였습니다.

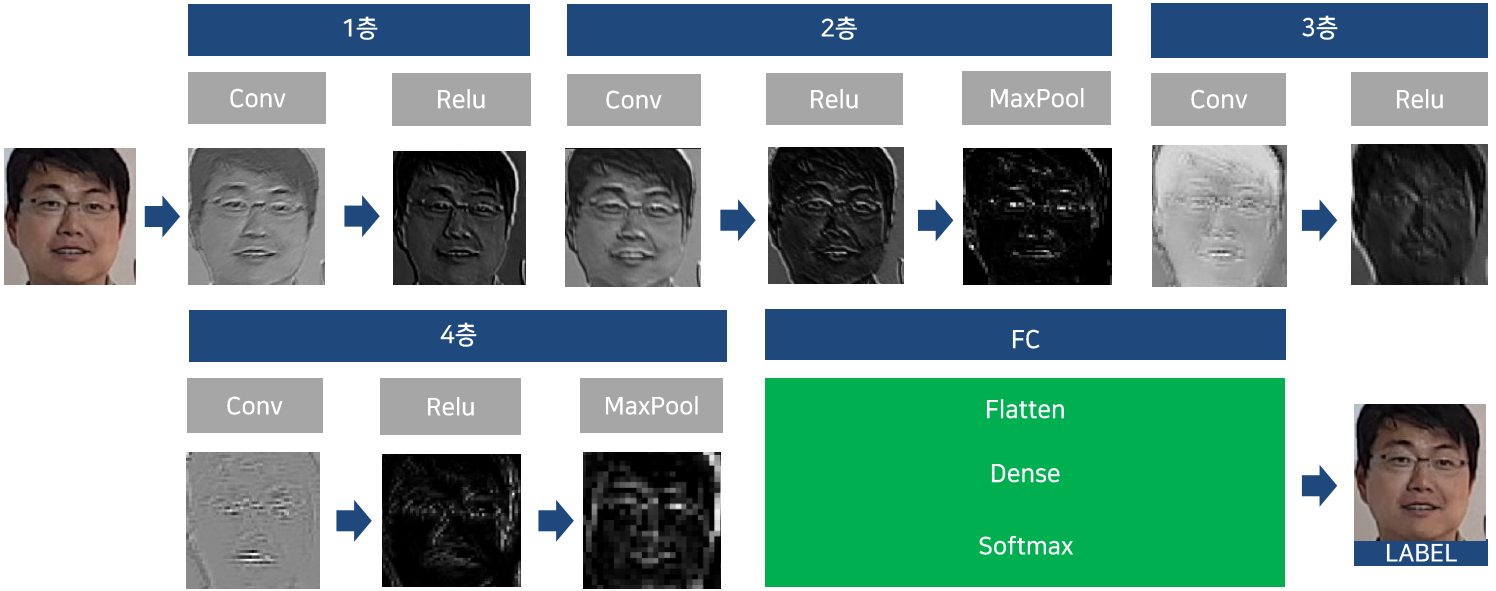
Keras

- Keras는 딥러닝 프레임워크 중에서 가장 간결하고 직관적이며 사용하기 쉽기 때문에 사용이 편리합니다.
- 모듈화가 잘 되어있기 때문에 빠르게 모델을 구현하거나 레이어를 교체할 수 있습니다.
- 내부적으로 tensorflow를 backend로 두고 있기 때문에, tensorflow의 기능도 사용할 수 있습니다.



모델 구상 및 훈련 요약

데이터는 CNN 모델의 각 층을 통과한 뒤 Fully-Connected 층에서 라벨로 판별됩니다.



모델 구상 및 훈련 요약

하이퍼파라미터와 Optimizer를 변경하면서 최적의 모델을 찾고자 하였습니다.
최종적으로 94%의 정확도를 내는 모델을 구현하였습니다.

Model	CNN
Learning rate	0.001
Batch	100
Optimizer	Adam
가중치 초기값	He
Dropout	0.5
Accuracy	94%

Agenda

01

주제 소개

주제 선정 이유, VIP Detection Sensor 소개, 프로젝트 일정

02

모델 구현

데이터 수집, 데이터 정제, 모델 훈련, 최적화, 최종 모델

03

시뮬레이션

테스트 영상 구현

04

결론

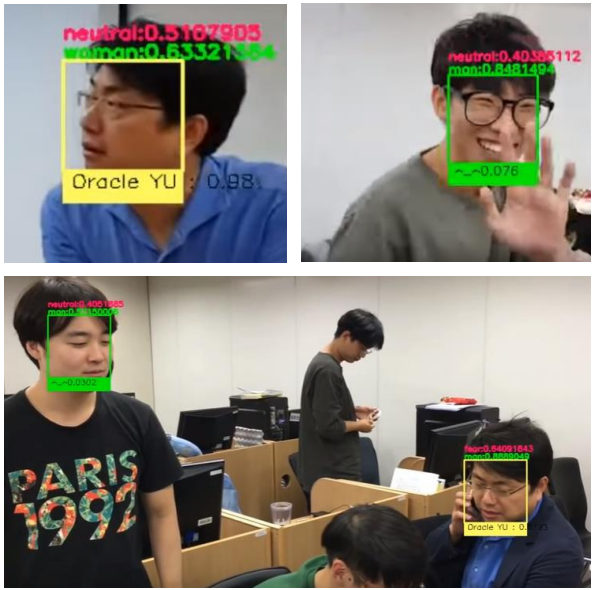
프로젝트 의의

영상 시뮬레이션

영상에서 타겟이 출연하였을 때 라벨로 잘 인식하는지 시뮬레이션 하였습니다.

YouTube https://youtu.be/e52dZfQ9-AA

시뮬레이션 영상



영상 시뮬레이션

타겟과 유사인물을 대상으로, 라벨 판별 시뮬레이션을 실시하였습니다.

타겟

VS

유사인물

시뮬레이션 영상

YouTube https://youtu.be/_GOi78BfK-E

Agenda

01

주제 소개

주제 선정 이유, VIP Detection Sensor 소개, 프로젝트 일정, 자체 평가리스트

02

모델 구현

데이터 수집, 데이터 정제, 모델 훈련, 최적화, 최종 모델

03

시뮬레이션

테스트 영상 구현

04

결론

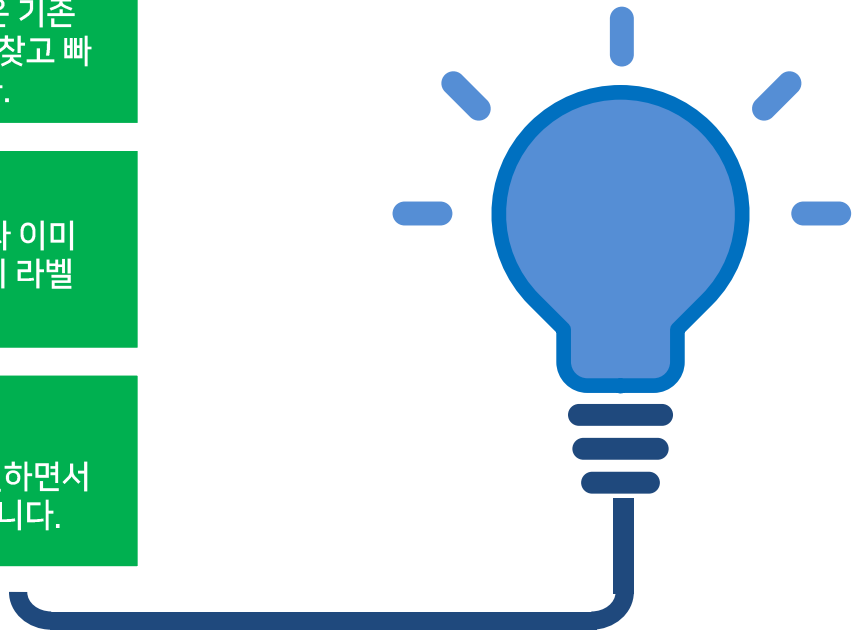
프로젝트 의의

프로젝트 의의

01 얼굴 인식 문제 해결
정확도가 낮고 인식하는데 시간소요가 높은 기존의 방식을 해결하는 과정에서 좋은 대안을 찾고 빠르게 적용하여 프로젝트를 진행하였습니다.

02 라벨 데이터 수집
영상에서 얼굴을 인식하여 추출하는 방식과 이미지 augumentation을 활용하여 많은 양의 라벨 데이터를 빠르게 확보하였습니다.

03 모델 구현
VGG-net 을 기반으로 파라미터 값을 조절하면서 94%의 정확도를 가진 모델을 구현하였습니다.



THANK YOU