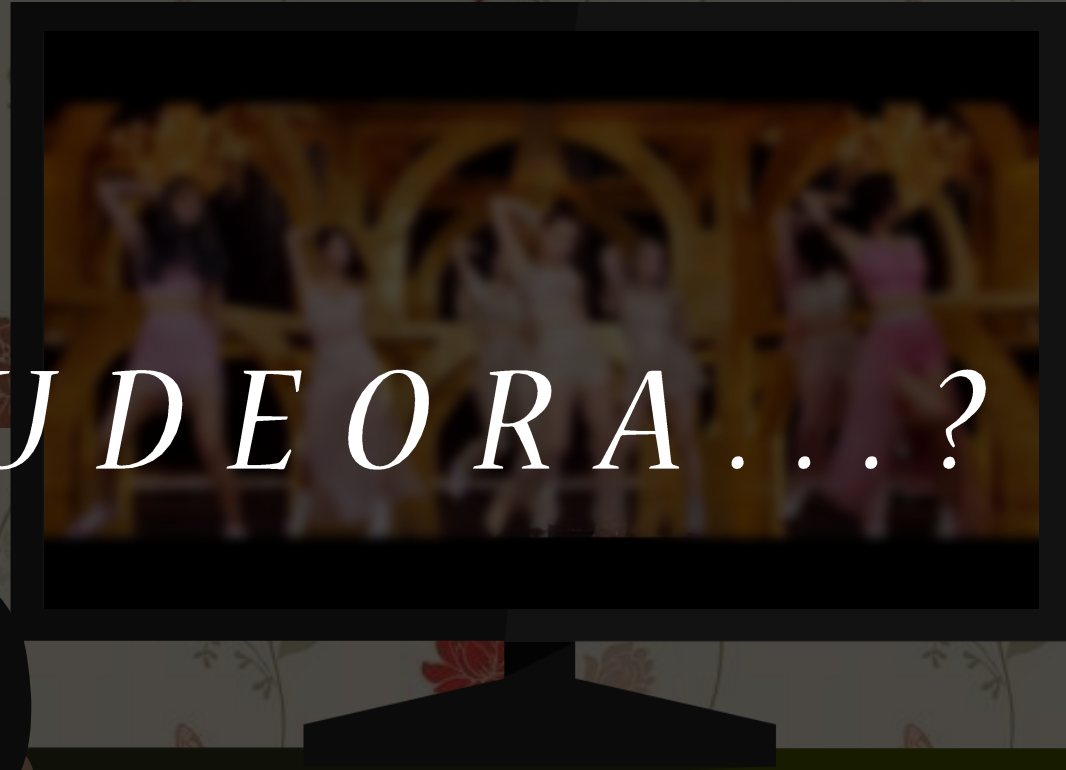




NUGUDEORA...?



NUGUDEORA



NAVER BTS rm

통합검색 이미지 블로그 뉴스 카페 포스트 지식iN 동영상 더보기

정렬 기간 영역 옵션유지 깨짐 썸네일 상세검색

인물 정보

RM (김남준) 가수

출생 1994년 9월 12일

소속그룹 방탄소년단

소속사 빅히트 엔터테인먼트

데뷔 2013년 방탄소년단 싱글 앨범 [2 COOL 4 SKOOL]

수상 2018년 화관문화훈장

사이트 공식홈페이지, V LIVE

내 인물정보 수정

본인참여 2019.04.15. 인물정보 더보기

내어바 TV 앨범 공연 방송 더보기

- 01 _ 데이터 수집
- 02 _ 데이터 전처리
- 03 _ 데이터 학습
- 04 _ 모델 적용
- 05 _ 개선점

1. 데이터 수집

GoogleImageCrawler

다음 기준으로 검색결과 좁히기...

이미지 크기:

가로/세로 비율:

이미지의 색상: ☒ 모든 색상 ☐ 칼라 ☐ 흑백 ☐ 투명 ☐ 선택 색상:

이미지 유형:

지역:

사이트 또는 도메인:

세이프서치:

파일 형식:

사용 권한:

```
1 print("원하는 그림의 키워드를 넣어주세요: ")
2 kw=input()
```

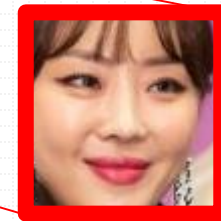
원하는 그림의 키워드를 넣어주세요:

나르샤

```
1 google_crawler=GoogleImageCrawler(
2     feeder_threads=1,
3     parser_threads=1,
4     downloader_threads=4,
5     storage={'root_dir':download_dir})
6
7
8
9 filters=dict(size='small', #이미지 크기
10             color='color', #이미지의 색깔을 필터링
11             license='commercial,modify',
12             #검색하고자 하는 날짜의 범위
13             date=((2017,1,1),(2017,12,31)))
14
15 google_crawler.crawl(keyword=kw1,
16                     filters=filters,
17                     max_num=50,
18                     file_idx_offset=0)
19
```

1. 데이터 수집

Face ROI



100*100px

1. 데이터 수집

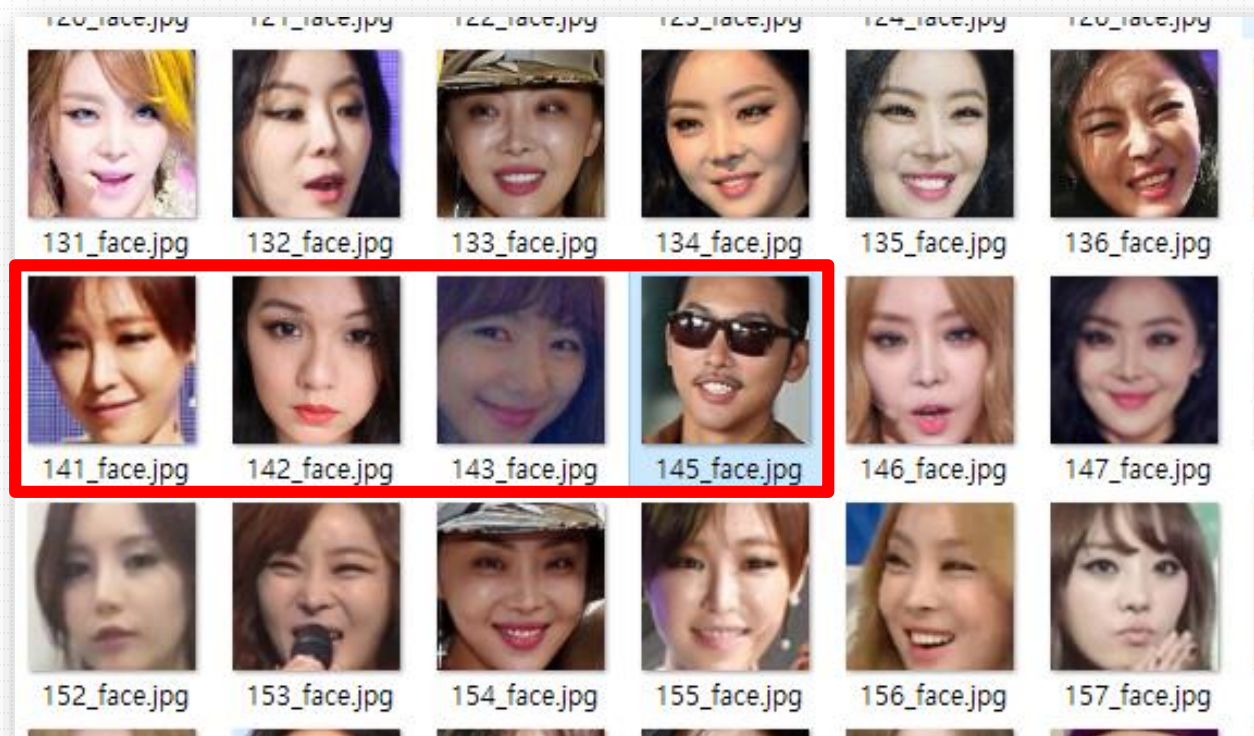
Face ROI

```
1 face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
2 #폴더 내의 이미지 파일명 리스트 반환
3 file_list = os.listdir(download_dir)
4
5 #각 이미지 얼굴만 추출저장
6 for idx, f_name in enumerate(file_list):
7     try:
8         print('processing: ' + str(idx))
9         #file_type = re.findall('[.]\w+', f_name)[0]
10        f_name1 = f_name
11        f_name = download_dir + '/' + f_name
12        img = my_imread(f_name)
13        gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
14        faces = face_cascade.detectMultiScale(gray, 1.3, 7)
15        if type(faces) == tuple:
16            continue
17        tmp_px=20
18        h_tmp_px=tmp_px//2
19        for (x,y,w,h) in faces:
20            img = cv2.rectangle(img,(x-h_tmp_px,y-h_tmp_px),(x+w+h_tmp_px,y+h+h_tmp_px),(255,0,0),2)
21            roi_gray = gray[y:y+h, x:x+w]
22            roi_color = img[y:y+h, x:x+w]
23            roi_color=cv2.resize(roi_color,(100,100),interpolation=cv2.INTER_LINEAR)
24            save_dir = download_dir + '_face'
25            w_name = save_dir + '/' + f_name1
26            if not os.path.exists(save_dir):
27                os.makedirs(save_dir)
28            my_imwrite(w_name, roi_color)
29        except:
30            pass
31    print("done")
```

faces = face_cascade.detectMultiScale(gray, 1.3, 7)





















1. 데이터 수집

Face ROI



2. 데이터 전처리

Data Argumentation

Add  value=45	Add (per_channel=True)  value=[35, 55]	AdditiveGaussianNoise  scale=0.30*255	AdditiveGaussianNoise (per_channel=True)  scale=0.30*255	AdditiveLaplaceNoise  scale=0.30*255
AdditiveLaplaceNoise (per_channel=True)  scale=0.30*255	AdditivePoissonNoise  lam=64.00	AdditivePoissonNoise (per_channel=True)  lam=64.00	Multiply  value=1.50	Multiply (per_channel=True)  value=[1.40, 1.60]
Dropout  p=0.20	Dropout (per_channel=True)  p=0.20	CoarseDropout (p=0.2)  size_percent=0.05	CoarseDropout (p=0.2, per_channel=True)  size_percent=0.05	ImpulseNoise  p=0.20
SaltAndPepper  p=0.10	Salt  p=0.10	Pepper  p=0.10	CoarseSaltAndPepper (p=0.2)  size_percent=0.10	CoarseSalt (p=0.2)  size_percent=0.10

```
# Affine
seq = iaa.Sequential([iaa.Affine(rotate=(-25, 25))], random_order=True)
images_aug.append(seq.augment_image(image))

# Gaussian Noise
seq = iaa.Sequential([iaa.AdditiveGaussianNoise(scale=(30, 50))], random_order=True)
images_aug.append(seq.augment_image(image))

# Crop
seq = iaa.Sequential([iaa.Crop(percent=(0, 0.3))], random_order=True)
images_aug.append(seq.augment_image(image))

# Laplacian Noise
seq = iaa.Sequential([iaa.AdditiveLaplaceNoise(scale=0.1*255, per_channel=True)], random_order=True)
images_aug.append(seq.augment_image(image))

# Coarse Dropout
seq = iaa.Sequential([iaa.CoarseDropout(p=0.2, size_px=1, per_channel=True)], random_order=True)
images_aug.append(seq.augment_image(image))

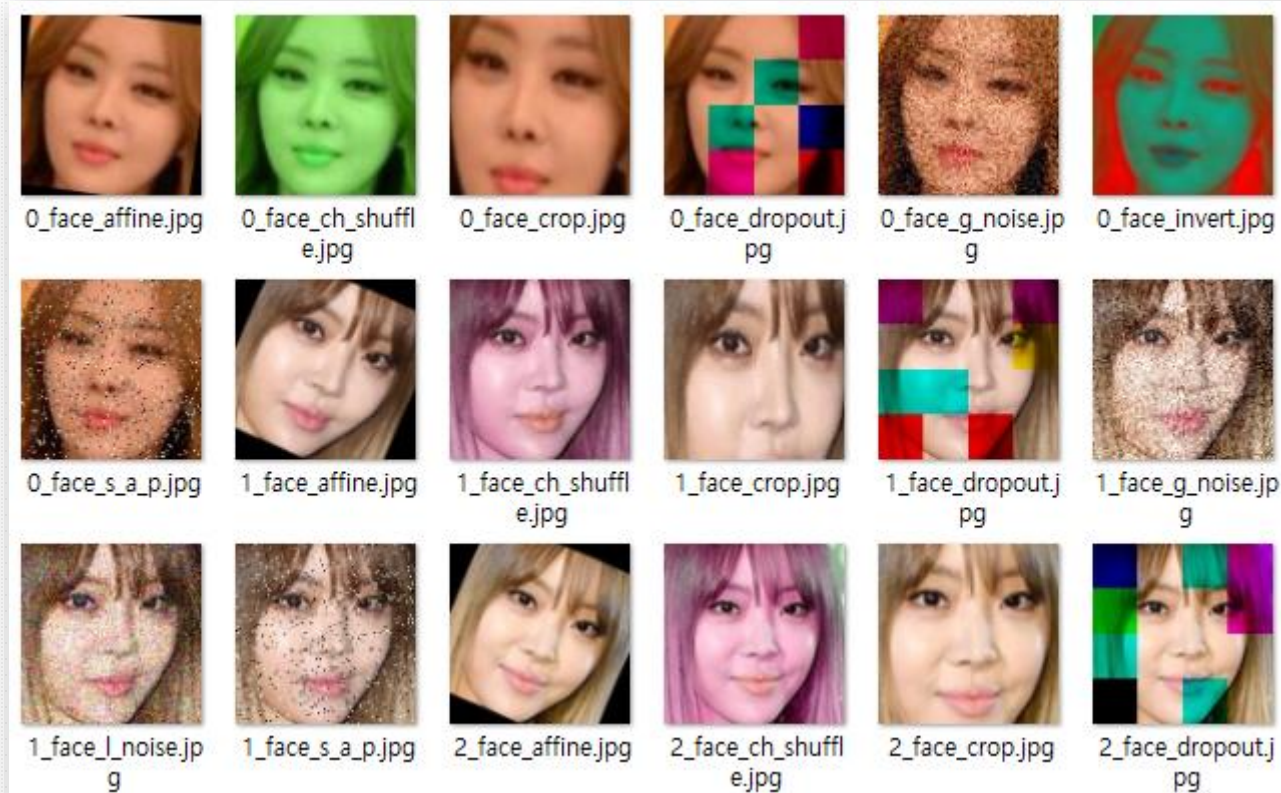
# Invert
seq = iaa.Sequential([iaa.Invert(per_channel=True, p=0.6)], random_order=True)
images_aug.append(seq.augment_image(image))

# Salt and Pepper
seq = iaa.Sequential([iaa.SaltAndPepper(p=0.1)], random_order=True)
images_aug.append(seq.augment_image(image))

# Channel Shuffle
seq = iaa.Sequential([iaa.ChannelShuffle(p=1.0)], random_order=True)
images_aug.append(seq.augment_image(image))
```

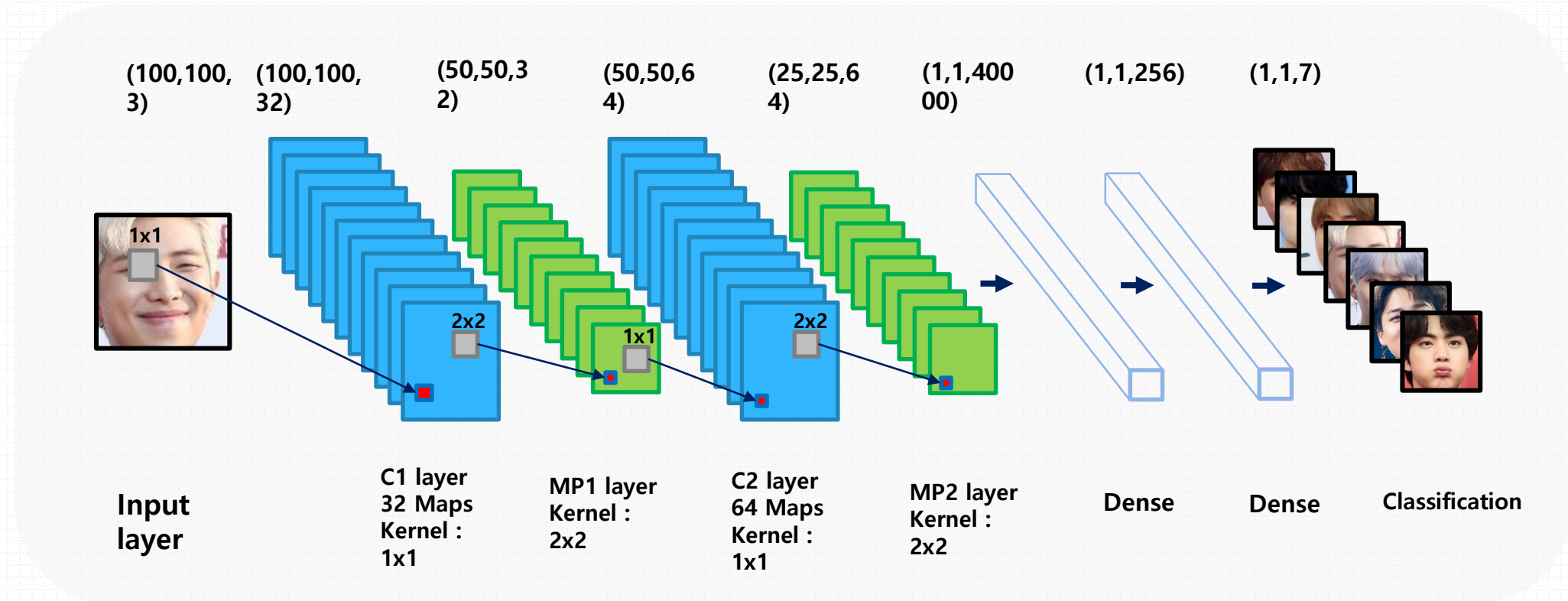
2. 데이터 전처리

Data Argumentation



3. 데이터 학습

Model



3. 데이터 학습

Sequential

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 100, 100, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 50, 50, 32)	0
dropout_1 (Dropout)	(None, 50, 50, 32)	0
conv2d_2 (Conv2D)	(None, 50, 50, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 25, 25, 64)	0
dropout_2 (Dropout)	(None, 25, 25, 64)	0
flatten_1 (Flatten)	(None, 40000)	0
dense_1 (Dense)	(None, 256)	10240256
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 9)	2313

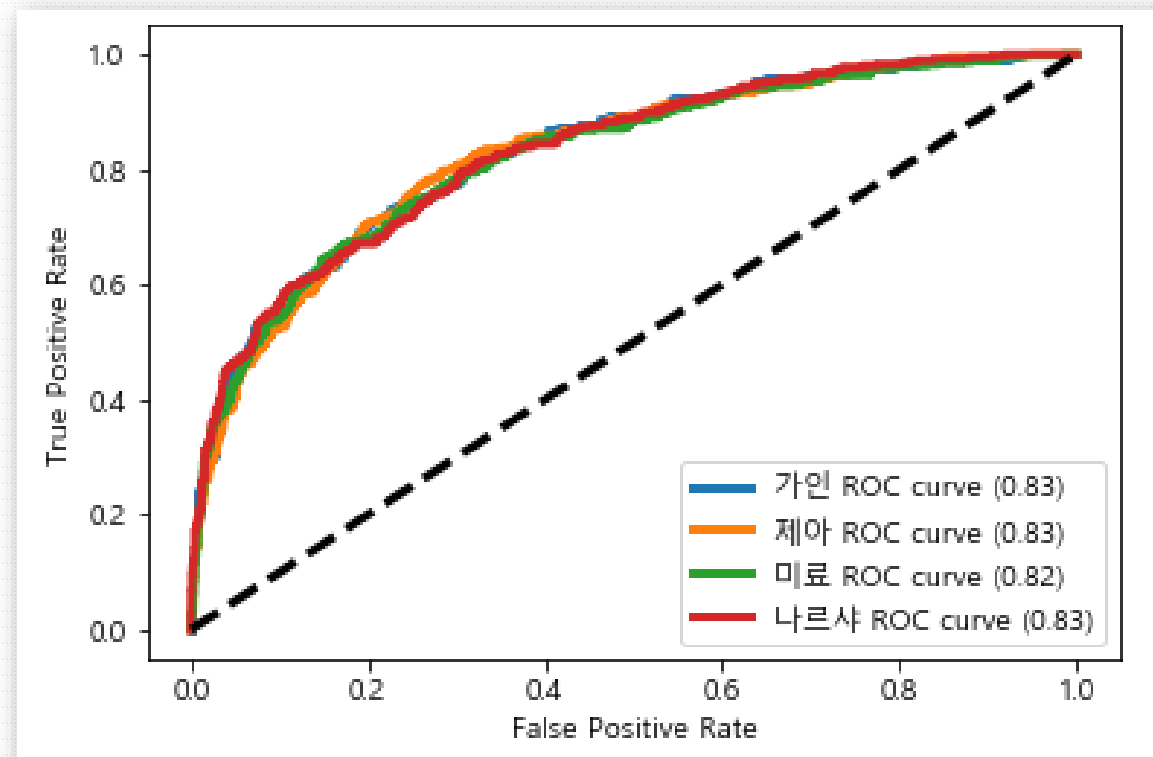
Total params: 10,261,961

Trainable params: 10,261,961

Non-trainable params: 0

3. 데이터 학습

ROC curve



4. 모델 적용

저장된 모델 불러오기

```
# 학습된 모델 불러오기 (****수정하세요****)
model = joblib.load('BTS_classification_model2.pkl')

# (****categories만 수정하세요****)
categories = ["브아걸_제아", "브아걸_나르샤", "브아걸_미료", "브아걸_가인"]
roi_color = [(255, 255, 255), (0, 255, 255), (255, 0, 255), (255, 255, 0),
              (0, 0, 255), (0, 255, 0), (255, 0, 0)]

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
fontpath = "fonts/gulim.ttc"
font = ImageFont.truetype(fontpath, 20)

count=0
frame_count=0

# (****경로 수정하세요****)
cap=cv2.VideoCapture('img/Brown_Eyed_Grils_mv.mp4')>>
```


4. 모델 적용

지정한 영상에 모델 적용

```
# 학습된 모델 불러오기 (****수정하세요****)
model = joblib.load('BTS_classification_model2.pkl')

# (****categories만 수정하세요****)
categories = ["브아걸_제아", "브아걸_나르샤", "브아걸_민정", "브아걸_지민", "브아걸_정국", "브아걸_남준", "브아걸_뷔", "브아걸_지수", "브아걸_태연", "브아걸_정현", "브아걸_윤하", "브아걸_아이유", "브아걸_소피", "브아걸_트와이스", "브아걸_블랙핑크", "브아걸_레드벨벳", "브아걸_몬스타엑스", "브아걸_에픽하이", "브아걸_투모로우바이투게더", "브아걸_더보이즈", "브아걸_스트레이 키즈", "브아걸_엔하이픈", "브아걸_뉴진스", "브아걸_르세라핌", "브아걸_이슬", "브아걸_김민서", "브아걸_김수현", "브아걸_김수민", "브아걸_김수정", "브아걸_김수지", "브아걸_김수현", "브아걸_김수민", "브아걸_김수정", "브아걸_김수지"]
roi_color = [(255, 255, 255), (0, 255, 255), (255, 0, 255), (0, 0, 255), (0, 255, 0), (255, 0, 0)]

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
fontpath = "fonts/gulim.ttc"
font = ImageFont.truetype(fontpath, 20)

count=0
frame_count=0

# (****경로 수정하세요****)
cap=cv2.VideoCapture('img/Brown_Eyed_Grills_mv.mp4')

tmp_px=20
h_tmp_px=tmp_px//2
while True:
    is_ok, frame=cap.read()
    roi=[]
    if not is_ok:
        break
    gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    if len(faces)!=0:
        faces_n,_=faces.shape
        for (x,y,w,h) in faces:
            #frame = draw_face(frame, model, w, y, w, h)
            #roi_gray = gray[y:y+h, x:x+w]
            roi = frame[y:y+h, x:x+w]
            roi=cv2.resize(roi,(100,100),interpolation=cv2.INTER_LINEAR)
            pred_face=roi.reshape(1,100,100,3)
            i = np.argmax(model.predict(pred_face))

            frame = cv2.rectangle(frame,(x-h_tmp_px,y-h_tmp_px),
                                   (x+w+h_tmp_px,y+h+h_tmp_px),roi_color[i],2)
            frame_pil = Image.fromarray(frame)
            draw = ImageDraw.Draw(frame_pil)
            draw.text((x,y+h+40), categories[i], font=font, fill=roi_color[i])
            frame = np.array(frame_pil)
            #frame=cv2.putText(frame,categories[i],(x,y+h+40),
            #                  cv2.FONT_HERSHEY_PLAIN,2,roi_color[i],4)

    count+=1
    frame_count+=1
    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

4. 모델 적용

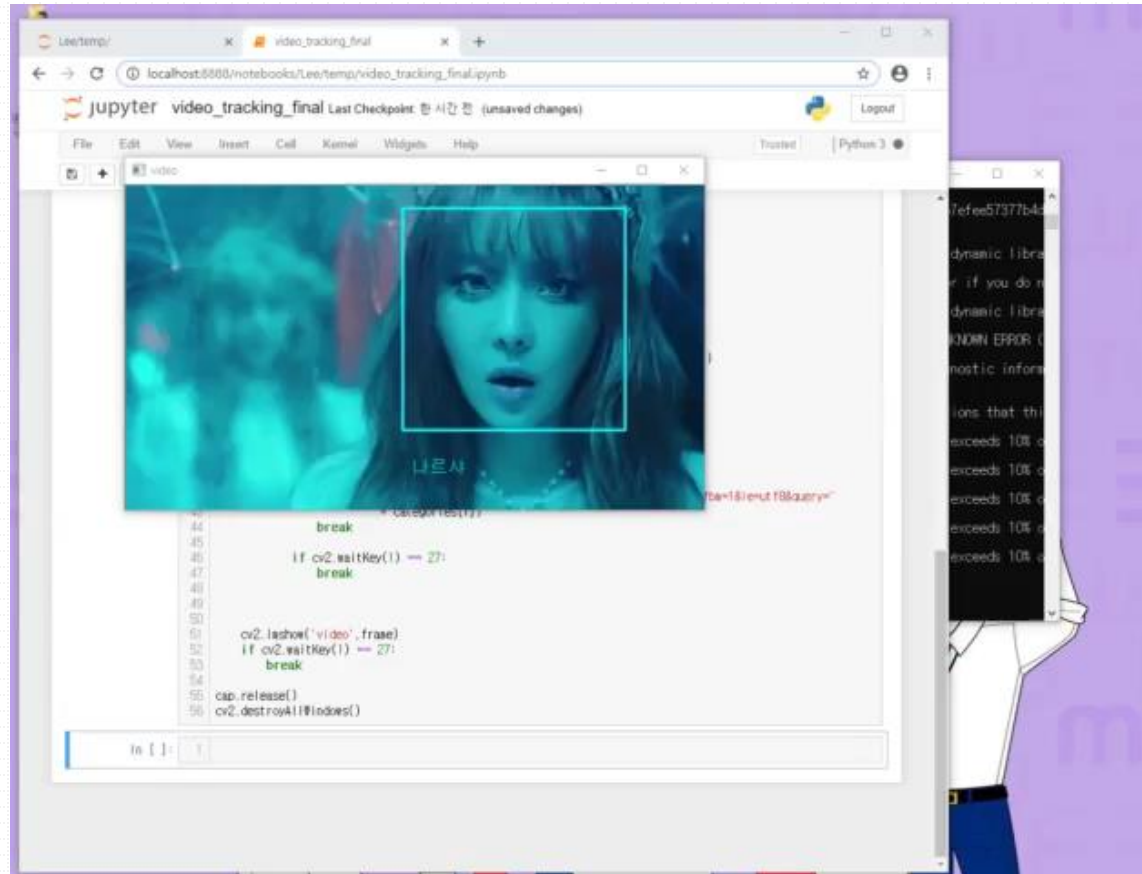
‘Enter’키를 통해 원하는 인물 검색

```
key = cv2.waitKey(1)
if key == 13: # enter
    driver = webdriver.Chrome('chromedriver.exe')
    driver.implicitly_wait(1)
    driver.get('https://search.naver.com/search.naver?sm=top_h ty&fbm=1&ie=utf8&query='
              + categories[i])
    break

if cv2.waitKey(1) == 27:
    break
```

4. 모델 적용

‘Enter’키를 통해 원하는 인물 검색



4. 모델 적용

‘Enter’키를 통해 원하는 인물 검색



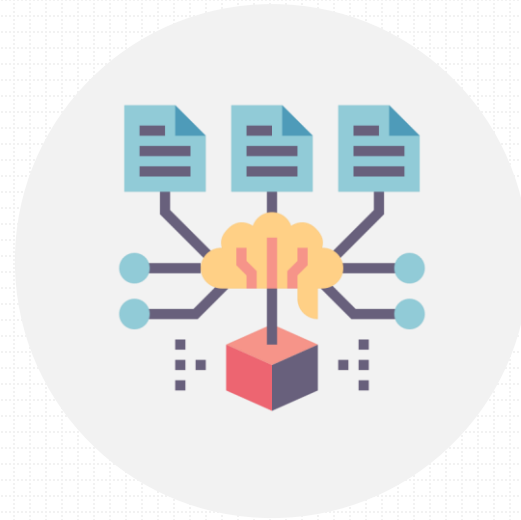
5. 개선점

개선해야 할 2가지 문제



이미지 데이터 수집

다양한 표정을 가진 사진 데이터가
학습시키기에 부족함



최적화된 모델 선정

임의의 모델을 만들어서 적용했기 때문에
최적화된 모델을 찾아야 함

감사합니다.