

어떤 옷을 입을지 고민 될 때,
사진, 한 장이면 ‘패스’!

팀명 : 패스
팀원 : 임지수, 박준원, 이해준



Contents

01. 프로젝트 배경

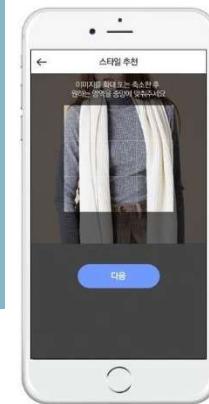
02. 프로젝트 팀 구성/ 역할

03. 프로젝트 수행 과정

1. 분류 모델 학습
 - 1) 데이터 전처리
 - 2) 모델 구현 및 학습
2. 코디 추천 매트릭스 구축

04. 프로젝트 수행 결과

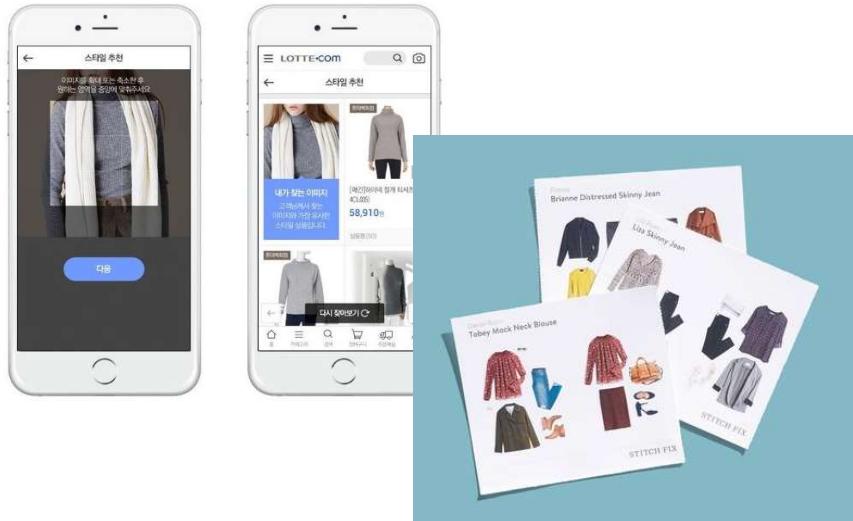
01. 프로젝트 배경



패션과 스타일에 대한 관심 증가로
AI 패션 큐레이팅 서비스 등 '[스타일 테크](#)'가 부상

롯데 닷컴의 스타일 추천
스티치 픽스의 AI 스타일리스트

01. 프로젝트 배경



현재 코디 서비스들은
비슷한 상품을 이미지로 검색해주거나,
고객의 평소 스타일을 분석해 그에 맞는 옷을
한 벌로 제안해주는 서비스가 대부분

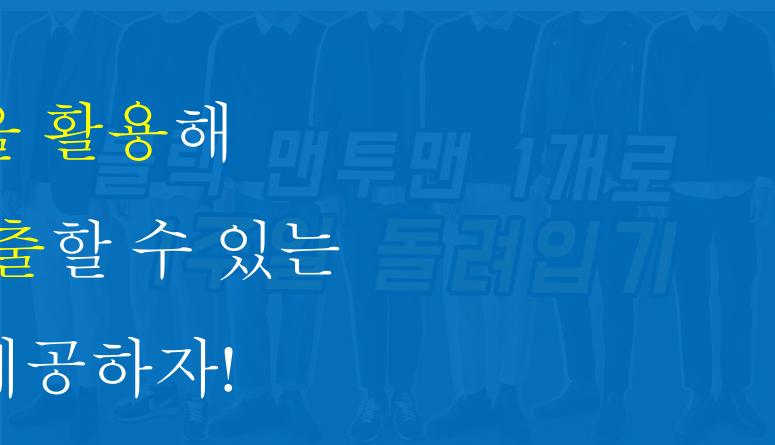


하지만, 새로운 한 벌의 구매보다
자신이 가진 옷을 활용해 다양한 스타일을 연출하고자 하는
Needs

01. 프로젝트 배경



가지고 있는 옷을 활용해
다양한 스타일을 연출할 수 있는
코디 서비스를 제공하자!



현재 코디 서비스들은
비슷한 상품을 이미지로 검색해주거나,
고객의 평소 스타일을 분석해 그에 맞는 옷을
한 벌로 제안해주는 서비스가 대부분

하지만, 새로운 한 벌의 구매보다
자신이 가진 옷을 활용해 다양한 스타일을 연출하고자 하는
Needs

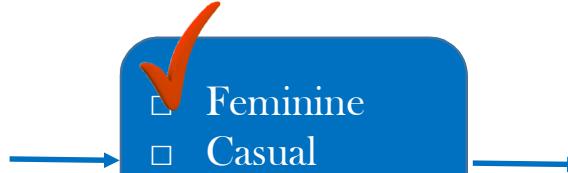
01. 프로젝트 배경

<사용자 관점>



상의 사진 입력

- 원하는 Style 선택
- Feminine
 - Casual
 - Formal



하의 추천



평소 편하게 입던 반팔 티
여성스럽게 연출할 수 있구나!

평소 편하게 입던 반팔 티
여성스럽게 연출할 수 있을까?

01. 프로젝트 배경

<개발자 관점>



상의 사진 입력

→
Shape : 반팔티
Pattern : 그래픽

입력값의 shape과
Pattern 분류

→
 Feminine
 Casual
 Formal

Feminine 추천 매트릭스에서 그
래픽_반팔티와 가장 많이
matching한 하의 찾기

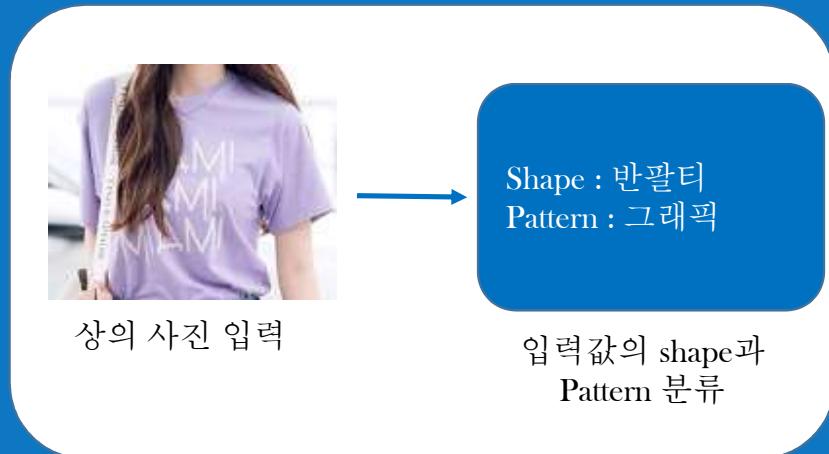
→
Shape : 쉬폰스커트
Pattern : 체크



체크_쉬폰스커트 사진
return

01. 프로젝트 배경

<개발자 관점>



1. 옷의 shape과 pattern 분류 모델

래픽_반팔티와 가장 높이

입력 받은 옷의 shape과 pattern 특성을 추출할 수 있도록



체크_쉬폰스커트 사진
return

01. 프로젝트 배경

<개발자 관점>

2. Style별 상하의 shape과 pattern 조합

추천 매트릭스

상의 사진입니다
입력값의 shape과
Pattern 분류
쇼핑몰 코디 이미지를 크롤링 해
어떤 상의에 어떤 하의를 많이 matching 하는지 매트릭스로 구축.
→ style에 따른 매트릭스를 만들기 위해 style 분류 모델도 필요!

Shape : 반팔티

Pattern : 그雷픽

- Feminine
- Casual
- Formal

Shape : 쉬폰스커트

Pattern : 체크

Feminine 추천 매트릭스에서 그
래픽_반팔티와 가장 많이
matching한 하의 찾기



체크_쉬폰스커트 사진
return

02. 프로젝트 팀 구성/역할

팀원	역 할
임지수 (팀 리더)	Keras CNN 모델 구현 홍콩 데이터 전처리 추천 매트릭스 구축 코드 구현 WBS 작성 Pattern 모델 train 데이터 전처리
박준원 (팀 원)	VGG16 모델 구현 Open-cv 전처리 Kakao API 사용 코드 구현 Shape 모델 train 데이터 전처리
이혜준 (팀 원)	핀터레스트, 인스타그램, 구글 크롤링 코드 구현 Style 모델 train 데이터 전처리 PPT 제작

3.1-1. 분류 모델

데이터 확보 및 전처리

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[Shape, pattern 학습 데이터 확보]_ 홍콩대학교 중문대학 딥패션 database



총 289,222장의 의상 사진
옷의 모양 - 옷의 종류 50개
옷의 무늬 - texture 53개

2-in-1_Space_Dye_Athletic_Tank
25_Mesh-Paneled_Jersey_Dress
36_Plaid_Shirt_Dress
1981_Graphic_Ringer_Tee
Above_Average_Linen_Tee
Abstract_Animal_Print_Dress
Abstract_Arrow_Flounce_Romper
Abstract_Asymmetrical_Hem_Top
Abstract_Bodcon_Dress
Abstract_Brushstroke_Pocket_Top
Abstract_Brushstroke_Print_Pencil_Skirt
Abstract_Brushstroke_Sweater

Label 간의 특징이 두드러지도록
비슷한 특성의 label 제거 후
4만장의 사진

hoodie
jeans
long_sleeve_tee
shirts
short_sleeve_tee
sweater
a_line_skirt
blouse
cotton_long_pants
cutoffs
h_line_skirt

Shape -11개

check
dot
floral
graphic
leopard
none
stripe
tribal
zigzag
lace

Pattern -10개

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[Shape, pattern 데이터 전처리] _ 1) 홍콩 데이터 box 좌표로 옷 shape에 맞게 잘라내기

▶ 원본 사진보다 더 명확하게 shape과 pattern label의 특성 차이를 학습할 수 있음.

자른 후 사진의 크기가 150 * 150미만인 이미지는 모델 학습 시 픽셀이 깨질 것을 우려해 삭제

```
import pandas as pd
data = pd.read_csv('./data/원본 데이터/list_bbox.csv')
data

image_name x1 y1 x2 y2
0 img/Sheer_Pleated-Front_Blouse/img_00000001.jpg 72 79 232 273
1 img/Sheer_Pleated-Front_Blouse/img_00000002.jpg 67 59 155 161
2 img/Sheer_Pleated-Front_Blouse/img_00000003.jpg 65 65 156 200
3 img/Sheer_Pleated-Front_Blouse/img_00000004.jpg 51 62 167 182
4 img/Sheer_Pleated-Front_Blouse/img_00000005.jpg 46 88 166 262

for _ in range(0,289222):
    try:
        jpg_file = './data/img/' + data['image_name'][_].split('/')[1] + "/" + data['image_name'][_].split('/')[2]
        img = Image.open(jpg_file,'r')
        dim = (int(data['x1'][_]),
               int(data['y1'][_]),
               int(data['x2'][_]),
               int(data['y2'][_]))
        crop_img = img.crop(dim)
        name = './data/cut_img/' + data['image_name'][_].split('/')[1] + "/" + data['image_name'][_].split('/')[2]
        crop_img.save(name)
        if _ % 1000 ==0:
            print(_)
    except:
        print(_)
        print(name)
```



[원본 사진]

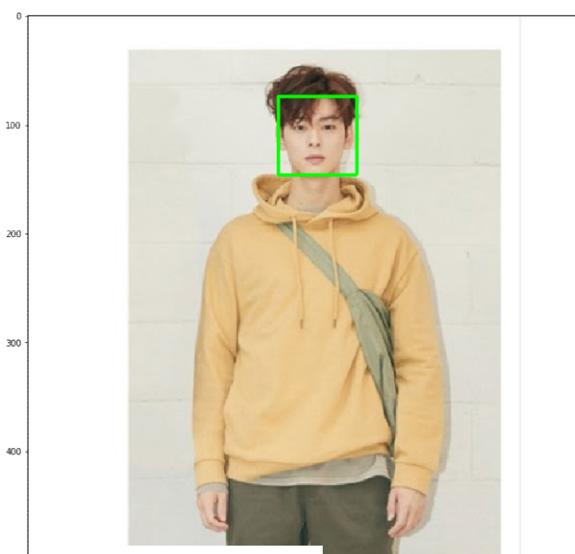


[전처리 후 사진]

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[Shape, pattern 데이터 전처리] _ 2) Crawling 한 데이터 OpenCV로 얼굴 자르기

▶ OpenCV : 인텔이 개발한 라이브러리. 안면과 몸체를 인식해 좌표로 결과값 제공.
haarcascade_frontalface_default로 얼굴 인식 후, 얼굴을 잘라내고 옷 사진만 남김.



[원본 사진]



[전처리 후 사진]



모든 전처리 후,
Shape 약 6000장
Pattern 약 10000장

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[Style 학습 데이터 확보]_ 편터레스트, 인스타그램, 구글 이미지 crawling

▶ 각 label에 대해 아래의 검색어로 2000장씩 이미지 crawling

```
Google 사진 클릭해서 크롤링

## 사이트 초기화
search = input('검색어를 입력하세요: ')
url = f'https://www.google.com/search?q={quote_plus(search)}&source=lnms&tbo=isch&sa=X&ved=2ahUKEwjOia9vJhufoAhUk7GEKHMaASQQ_AUoAXoEg
ur1 = f'https://www.google.com/search?q={quote_plus(search)}&tbo=isch'

driver = webdriver.Chrome()
driver.get(url)

time.sleep(3)

elem = driver.find_element_by_tag_name("body")

## for문에 뒤에 숫자만 바꿔주면 원하는 갯수만큼 가능
for i in range(1,1001):
    ## 이미지 클릭
    click_section = '//*[@id="isrg"]/div[1]/div[' + str(i) + ']/a[1]' #div[0] 안의 숫자가 바뀌면은 클릭할 사진이 바뀌는 거임
    driver.find_element_by_xpath(click_section).click()
    time.sleep(2)

    ## 오른쪽에 드는 이미지 url 가져오기
    img_section = '//*[@id="Sva75c"]/div/div/div[3]/div[2]/div/div[1]/div[1]/div/div[2]/a/img'
    img = driver.find_element_by_xpath(img_section).get_attribute('src')

    ## 이미지 저장
    img_name = 'C:/Users/LHJ/Desktop/img_crawling/병아리' + '/' + 'cutoffs' + str(i) + '.jpg'
    urllib.request.urlretrieve(img, img_name)

except:
    pass

driver.close()
```



Casual	Feminine	Formal
여성 캐쥬얼	Feminine style	#직장인룩
남성 캐쥬얼	여성스러운 패션	남성 정장
lookple.e	Feminine look	여성 정장
스트릿 패션	여성스러운 스타일	오피스룩

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[Style 데이터 전처리]_ OpenCV로 상하의 모두 포함된 사진만 남기기

▶ 상의와 하의가 따로 style을 결정하는 것이 아니라, **상하의의 코디 조합 자체가 style을 결정하는 것이라고 판단.**
OpenCV의 haarcascade_fullbody로 상하의가 모두 포함된 사진만 style 모델 학습에 사용.

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
import matplotlib.image as mpimg
import os
from PIL import Image

# fullbody.xml 불러오기
body_cascade = cv2.CascadeClassifier(
    './opencv-master/data/haarcascades/haarcascade_fullbody.xml') # 런칭!

# body 갯수 세는 함수
def count_body(x):
    body = body_cascade.detectMultiScale(x, 1.01, 10, minSize=(30,30)) # 너무 조
    return len(body)

# body 갯수가 1이상이면 저장
for img_name in os.listdir('C:/Users/student/Desktop/fullbody'):
    img = mpimg.imread('C:/Users/student/Desktop/fullbody/' + '/' + img_name)
    mpimg.imsave(new_name, img)

    try:
        count_body(img)
        new_name = 'C:/Users/student/Desktop/fullbody_2' + '/' + img_name
        mpimg.imsave(new_name, img)
    except:
        pass
```



Style -3개

- casual - 517장
- feminine - 358장
- formal - 332장

모든 전처리 후,
Style 1207장

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[최종 train_data_set]

- ▶ Shape : 특성 구분이 어려운 블라우스와 셔츠를 합하고 , 모양 자체로 특성을 구분하기 어려운 sweater 제거
- ▶ Pattern : 특성이 뚜렷하고, 평상시 많이 착용하는 패턴 label만 남김



Shape -9개(5361)



Pattern -7개(6200)



Style -3개
(1207)

3.1-2. 분류 모델

모델 구현 및 학습

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[1차. 모델] _ keras CNN을 활용한 분류 모델

▶ Conv2D와 MaxPooling2D 층을 4개의 층으로 컨브넷 구성

전체 train set의 33%를 validation set으로 잡고 학습에 관여

validation loss에 대한 Patience를 20으로 해서 과적합 방지(EarlyStopping)

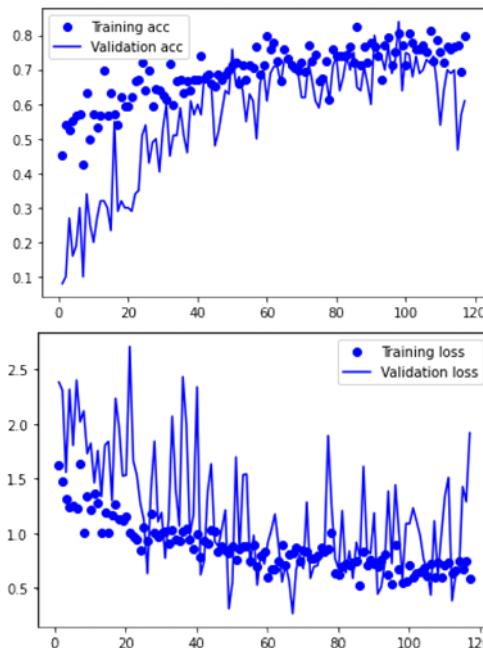
```
1 model = Sequential()
2 model.add(Conv2D(16, (3, 3), padding='same', use_bias=False, input_shape=(150,150, 3)))
3 # model.add(BatchNormalization(axis=3, scale=False))
4 model.add(Activation("relu"))
5 model.add(MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same'))
6 model.add(Dropout(0.2))
7
8 model.add(Conv2D(32, (3, 3), padding='same', use_bias=False))
9 # model.add(BatchNormalization(axis=3, scale=False))
10 model.add(Activation("relu"))
11 model.add(MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same'))
12 model.add(Dropout(0.2))
13
14 model.add(Conv2D(64, (3, 3), padding='same', use_bias=False))
15 # model.add(BatchNormalization(axis=3, scale=False))
16 model.add(Activation("relu"))
17 model.add(MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same'))
18 model.add(Dropout(0.2))
19
20 model.add(Conv2D(128, (3, 3), padding='same', use_bias=False))
21 # model.add(BatchNormalization(axis=3, scale=False))
22 model.add(Activation("relu"))
23 model.add(Flatten())
24 model.add(Dropout(0.2))
25
26 model.add(Dense(512, activation='relu'))
27 model.add(Dense(3, activation='softmax'))
28 model.summary()                                     # label, class 갯수
29
```

Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 150, 150, 16)	432
activation_5 (Activation)	(None, 150, 150, 16)	0
max_pooling2d_4 (MaxPooling2D)	(None, 38, 38, 16)	0
dropout_5 (Dropout)	(None, 38, 38, 16)	0
conv2d_6 (Conv2D)	(None, 38, 38, 32)	4608
activation_6 (Activation)	(None, 38, 38, 32)	0
max_pooling2d_5 (MaxPooling2D)	(None, 10, 10, 32)	0
dropout_6 (Dropout)	(None, 10, 10, 32)	0
conv2d_7 (Conv2D)	(None, 10, 10, 64)	18432
activation_7 (Activation)	(None, 10, 10, 64)	0
max_pooling2d_6 (MaxPooling2D)	(None, 3, 3, 64)	0
dropout_7 (Dropout)	(None, 3, 3, 64)	0
conv2d_8 (Conv2D)	(None, 3, 3, 128)	73728
activation_8 (Activation)	(None, 3, 3, 128)	0
flatten_2 (Flatten)	(None, 1152)	0
dropout_8 (Dropout)	(None, 1152)	0
dense_3 (Dense)	(None, 512)	590336
dense_4 (Dense)	(None, 3)	1539
<hr/>		
Total params: 689,075		
Trainable params: 689,075		
Non-trainable params: 0		
<hr/>		

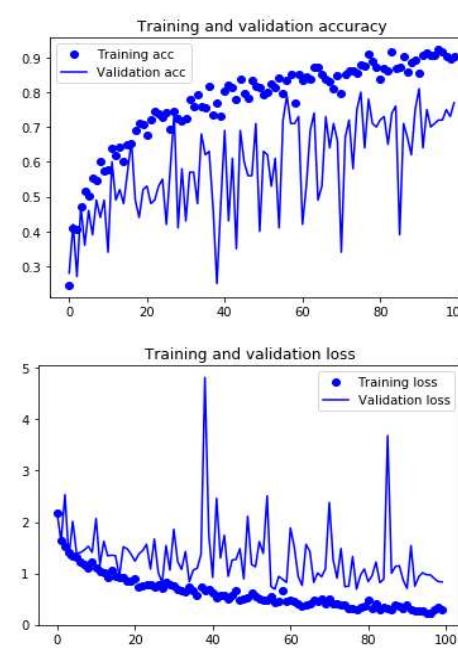
03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[1차. 모델 학습 결과]

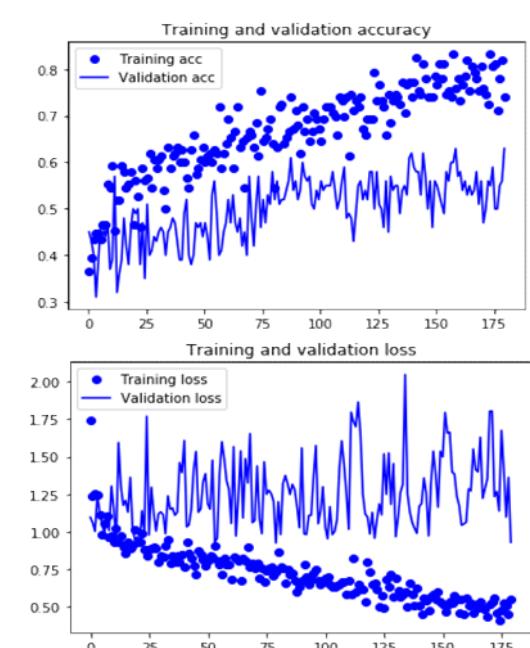
<shape>



<pattern>



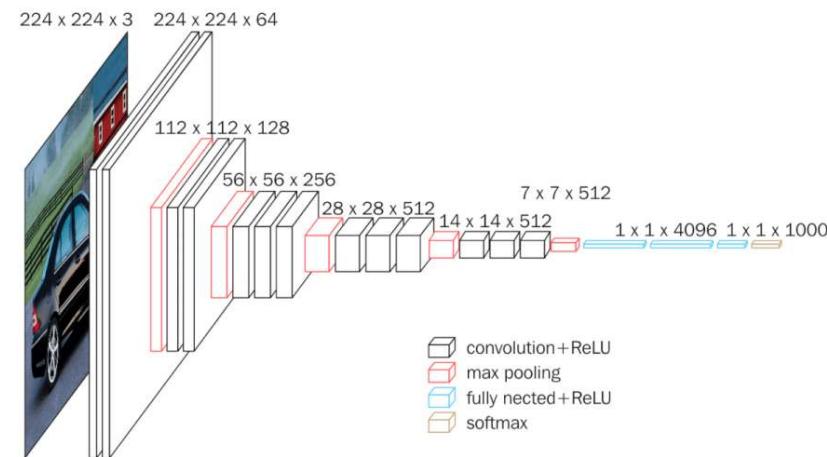
<style>



03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[2차. 모델 보완] _사전 훈련된 컨브넷 사용_ VGG16

▶ VGG16은 1400만장의 이미지와 1000개의 class. 16개의 층으로 사전 학습되어 있는 모델.
이미지의 특징을 추출하는 신경망의 능력을 그대로 이용하고, 마지막 출력 layer만 변경하여 이 변경된 layer만 재학습.
데이터 수가 적을 때도 높은 정확도의 모델 학습이 가능하고, 학습 속도도 빠름.



03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[2차. 모델 보완] _사전 훈련된 컨브넷 사용_ VGG16

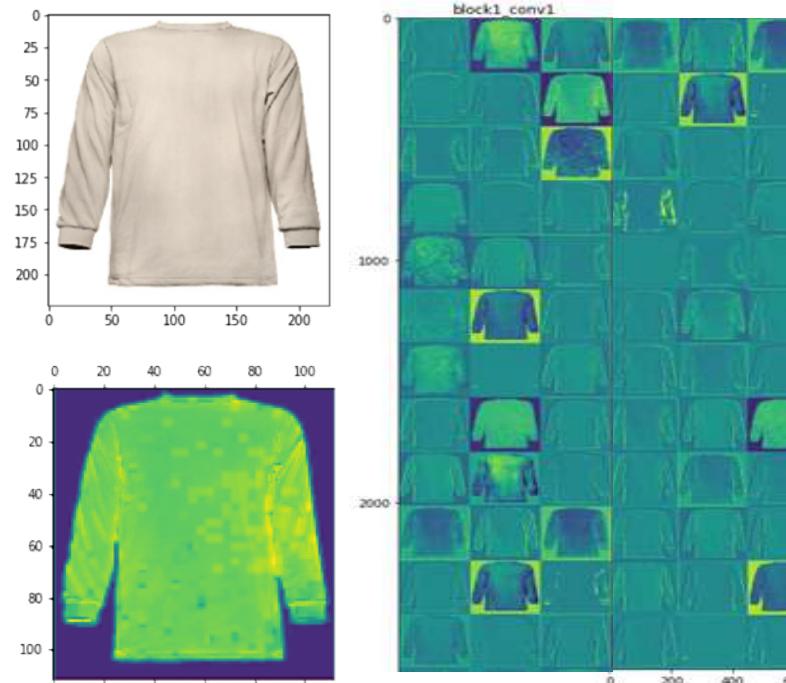
▶ VGG16의 중간층 활성화 시각화 :

이미지를 입력했을 때 이미지가 컨브넷의 Conv층과 Maxpooling 층을 지나며 어떤 특성에 의해 학습이 이루어지는지 확인 가능.
히트맵 시각화를 통해 이미지의 어느 부분이 최종 분류 결정에 기여했는지를 확인

```
from keras.applications import VGG16
conv_base = VGG16(weights = 'imagenet',
                  include_top=False,
                  input_shape=(150,150,3))

conv_base.summary()
```

| Layer (type) | Output Shape | Param # |
|----------------------------|----------------------|---------|
| Input_1 (Input Layer) | (None, 150, 150, 3) | 0 |
| block1_conv1 (Conv2D) | (None, 150, 150, 64) | 1792 |
| block1_conv2 (Conv2D) | (None, 150, 150, 64) | 36928 |
| block1_pool (MaxPooling2D) | (None, 75, 75, 64) | 0 |
| block2_conv1 (Conv2D) | (None, 75, 75, 128) | 73856 |
| block2_conv2 (Conv2D) | (None, 75, 75, 128) | 147584 |
| block2_pool (MaxPooling2D) | (None, 37, 37, 128) | 0 |
| block3_conv1 (Conv2D) | (None, 37, 37, 256) | 295168 |
| block3_conv2 (Conv2D) | (None, 37, 37, 256) | 590080 |
| block3_conv3 (Conv2D) | (None, 37, 37, 256) | 590080 |
| block3_pool (MaxPooling2D) | (None, 18, 18, 256) | 0 |
| block4_conv1 (Conv2D) | (None, 18, 18, 512) | 1180160 |
| block4_conv2 (Conv2D) | (None, 18, 18, 512) | 2359808 |
| block4_conv3 (Conv2D) | (None, 18, 18, 512) | 2359808 |
| block4_pool (MaxPooling2D) | (None, 9, 9, 512) | 0 |
| block5_conv1 (Conv2D) | (None, 9, 9, 512) | 2359808 |
| block5_conv2 (Conv2D) | (None, 9, 9, 512) | 2359808 |
| block5_conv3 (Conv2D) | (None, 9, 9, 512) | 2359808 |
| block5_pool (MaxPooling2D) | (None, 4, 4, 512) | 0 |
| Total params: | 14,714,688 | |
| Trainable params: | 14,714,688 | |
| Non-trainable params: | 0 | |

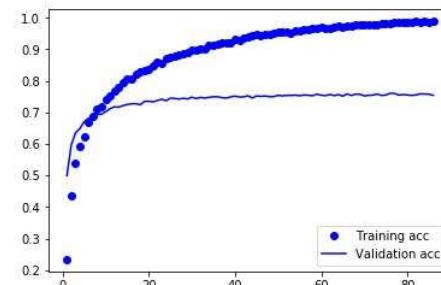


03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

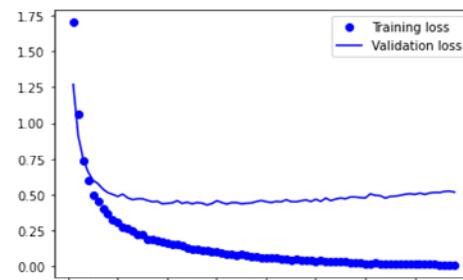
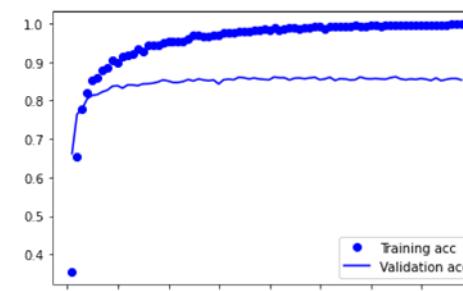
[2차. 모델 학습 결과] ► Training acc가 98%이상, Validation acc도 각 8% 이상 증가

Style의 경우 validation 정확도가 기준 모델이 더 높아서 이전 모델 선택

<shape>

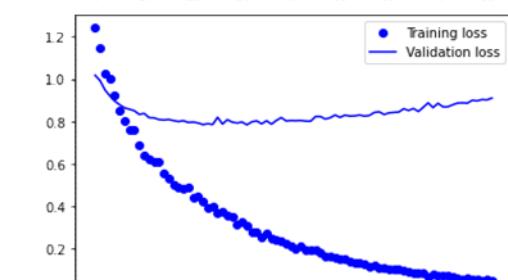
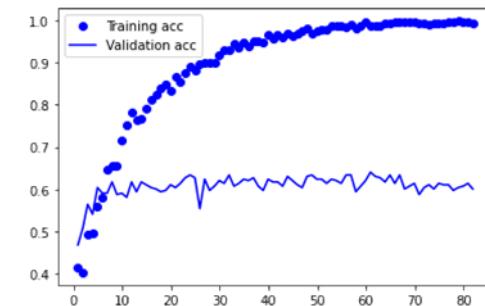


<pattern>



Training acc : 98%
Validation acc : 75%

<style>



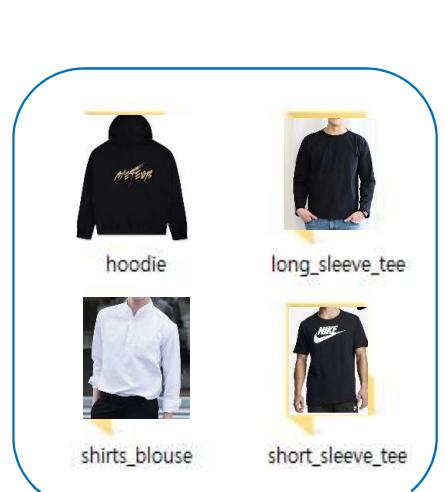
Training acc : 99%
Validation acc : 60%

03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

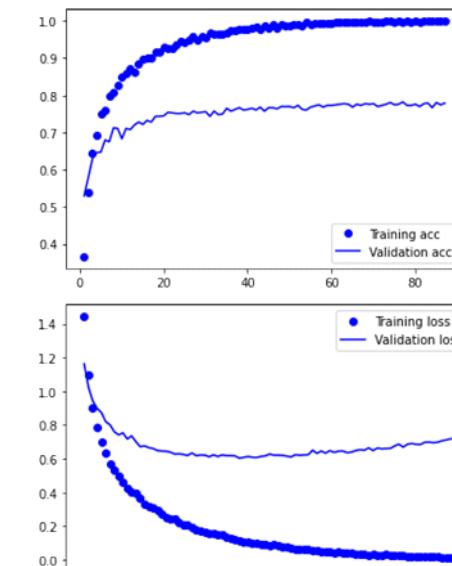
[3차. Shape 모델 보완]_상하의 분리해서 모델 학습

▶ 모양이 비슷한 반팔티를 반바지로 인식하는 문제 발생

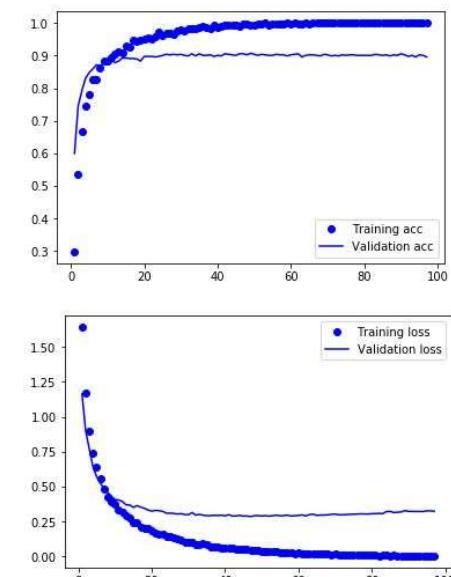
-> 상의 데이터와 하의 데이터를 분리해 학습. 결과적으로 정확도가 상승하며 문제 해결.



<상의 모델>



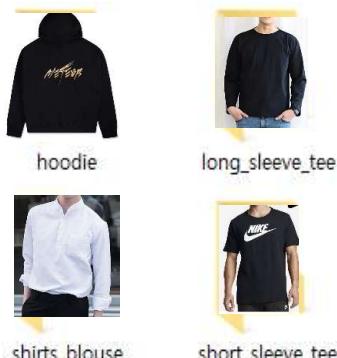
<하의 모델>



03. 프로젝트 수행 과정 _ 1. shape, pattern, style 분류 모델

[최종 분류 모델]

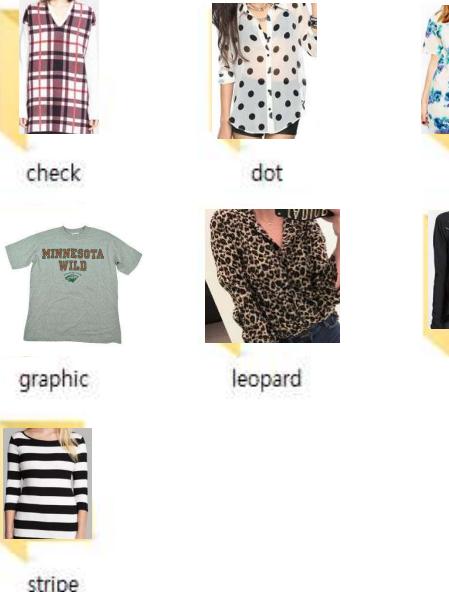
상의 분류 모델
- 78%



하의 분류 모델
- 90%



패턴 분류 모델
- 85%



스타일 분류 모델
- 63%



3.2. 코디 추천 매트릭스

03. 프로젝트 수행 과정 _ 2. 코디 추천 매트릭스

[매트릭스 구축 기반 데이터]

1. 최신 트렌드와 전문가의 코디 반영을 위해

2030의 인기 쇼핑몰인 무신사, 우신사, 코디북, hisfit 인스타그램 크롤링 (2024장)

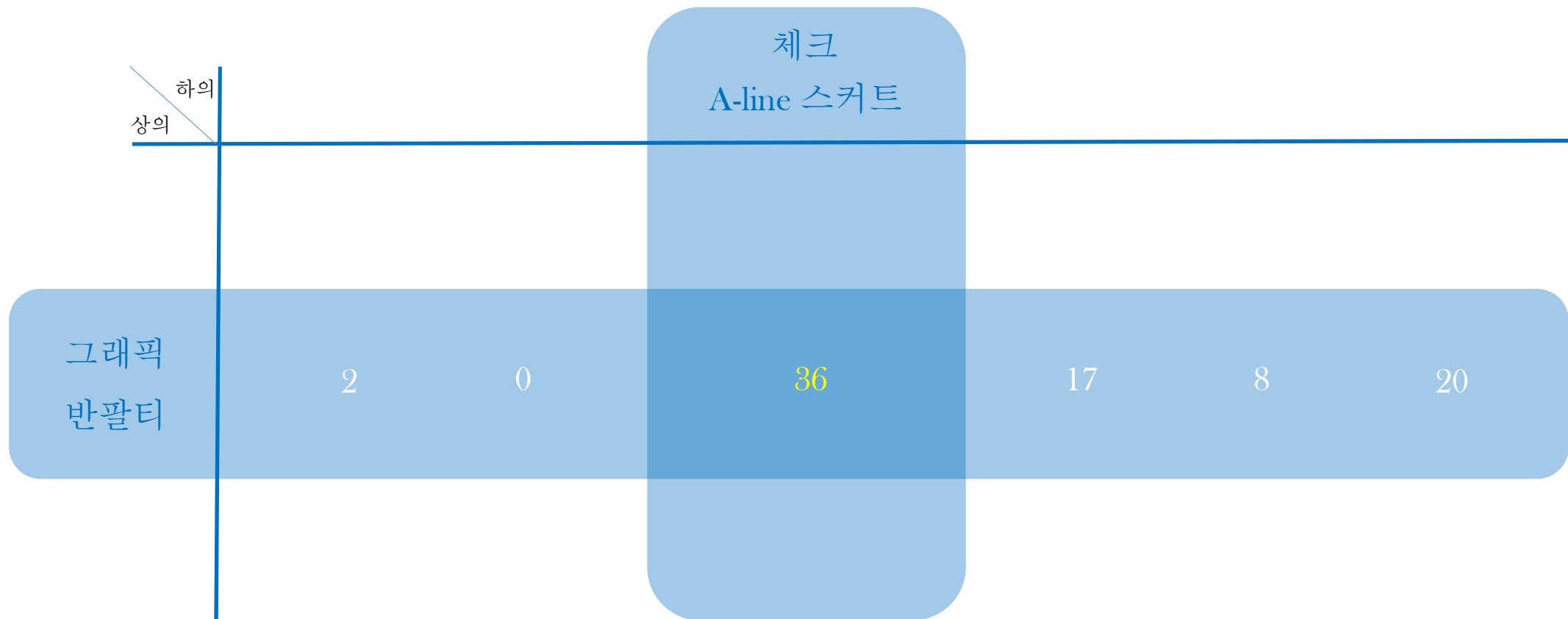


2. 모든 패턴_쉐입별 입력값에 대해 추천이 가능하도록 하기 위해 각 패턴_쉐입 당 10개의 이미지 확보 ($10 \times 70 = 700$ 장)

03. 프로젝트 수행 과정 _ 2. 코디 추천 매트릭스

[코디 추천 매트릭스 예시]

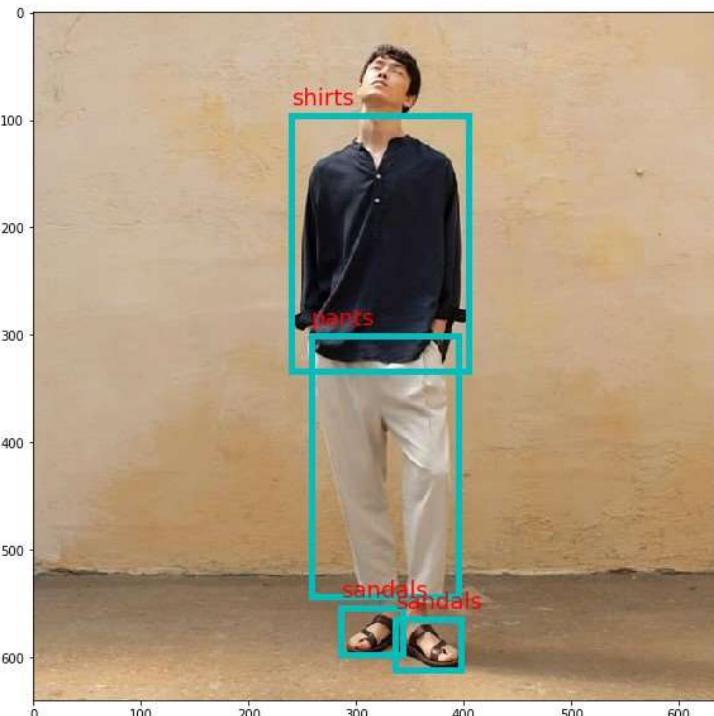
▶ 각 상의와 하의의 pattern_shape 매칭 빈도를 style 별로 3개의 매트릭스로 구축하여, 입력된 상의의 pattern_shape에 가장 잘 어울리는 하의의 pattern_shape 값을 뽑아 낼 수 있도록 한다.



03. 프로젝트 수행 과정 _ 2. 코디 추천 매트릭스

[추천 매트릭스 구축 과정]

▶ 카카오 Vision 상품검출 API : 사진을 입력하면, Shirts, t-shirts, pants, skirt 좌표를 뽑아주는 API
→ 상의와 하의를 분리하는데 사용



```
# 카카오 API 정보 설정
info = []
for each in result['result']['objects']:
    each = list(each.values())
    info.append(each)

# 상하의 구분해서 자르고 이미지 저장
for _ in range(len(info)):
    img = Image.open(img_dir)
    category = info[_][4]
    print(category)

    if category in ('shirts','t-shirts'):

        x = info[_][0]*fig_w - 100
        y = info[_][1]*fig_h - 100
        w = info[_][2]*fig_w + 50
        h = info[_][3]*fig_h + 50
        path = 'C:/python_DA/img/cropped_img/top/'
        name = path + category + str(time.time()) + '.jpg'
        img = img.crop((x,y,w,h))
        img.save(name)
        plt.imshow(img)
        img.show()

    elif category in ('pants','skirts'):

        x = info[_][0]*fig_w - 100
        y = info[_][1]*fig_h - 100
        w = info[_][2]*fig_w + 50
        h = info[_][3]*fig_h + 50
        path = 'C:/python_DA/img/cropped_img/bottom/'
        name = path + category + str(time.time()) + '.jpg'
        img = img.crop((x,y,w,h))
        img.save(name)
        plt.imshow(img)
        img.show()
```

03. 프로젝트 수행 과정 _ 2. 코디 추천 매트릭스

[추천 매트릭스 구축 과정]



Style : formal

Style 모델을 지나며
style 출력

원본 이미지



Shape: shirts
Pattern: None

Upper_shape 모델과
Pattern 모델을 지나며
Shape과 pattern 출력



Shape:
long-sleeve-pants
Pattern: None

lower_shape 모델과
Pattern 모델을 지나며
Shape과 pattern 출력

잘라진 이미지는 shape_pattern 별
폴더에 저장되어 추천 이미지로 사용.

| img_dir | style | top shape | top pattern | bottom shape | bottom pattern |
|-----------------------|----------|------------------|-------------|-------------------|----------------|
| 0 codibook (286).jpg | Feminine | Shirts | Check | Jeans | None |
| 1 codibook (284).jpg | Feminine | Long-sleeve-tee | Graphic | Cutoffs | None |
| 2 codibook (271).jpg | Casual | Shirts | None | Cotton-long-pants | None |
| 3 codibook (272).jpg | Casual | Short-sleeve-tee | None | Nan | Nan |
| 4 codibook (273).jpg | Formal | Short-sleeve-tee | None | Chiffon-skirt | Check |
| 5 codibook (282).jpg | Formal | Hoodie | Graphic | Cotton-long-pants | None |
| 6 codibook (283).jpg | Formal | Long-sleeve-tee | Graphic | Cutoffs | None |
| 7 codibook (281).jpg | Formal | Long-sleeve-tee | None | H-line-skirt | Check |
| 8 codibook (280).jpg | Casual | Nan | Nan | Jeans | None |
| 9 codibook (278).jpg | Casual | Long-sleeve-tee | None | Cotton-long-pants | None |
| 10 codibook (279).jpg | Casual | Shirts | None | Jeans | None |
| 11 codibook (277).jpg | Formal | Long-sleeve-tee | Check | Cotton-long-pants | None |
| 12 codibook (269).jpg | Casual | Shirts | None | Nan | Nan |
| 13 codibook (270).jpg | Formal | Long-sleeve-tee | Graphic | H-line-skirt | Graphic |
| 14 codibook (267).jpg | Feminine | Nan | Nan | H-line-skirt | Check |
| 15 codibook (276).jpg | Casual | Nan | Nan | Cotton-long-pants | None |
| 16 codibook (275).jpg | Formal | Long-sleeve-tee | Graphic | Cutoffs | None |
| 17 codibook (274).jpg | Formal | Hoodie | Graphic | Cotton-long-pants | None |
| 18 codibook (268).jpg | Feminine | Nan | Nan | Nan | Nan |
| 19 codibook (266).jpg | Feminine | Shirts | None | Chiffon-skirt | Floral |

원본 이미지의 style,
상의의 pattern, shape,
하의의 pattern, shape를 df로 저장

03. 프로젝트 수행 과정 _ 2. 코디 추천 매트릭스

[추천 매트릭스 구축 과정]

▶ 앞에서 만든 df를 pivot table로 변환시켜 style 별 코디 매트릭스 구축
각 매트릭스에 200개가 넘는 쇼핑몰 코디 데이터가 반영됨.

```
def recommend(df, style, pattern, shape): # dataframe 스타일, identify_clothes(data,0)[0], identify_clothes(data,0)[1]
    if style == 'Casual':
        ca_df = df[df['style']=='Casual']
        ca_df = ca_df[['top shape','top pattern', 'bottom shape', 'bottom pattern']]
        ca_df = ca_df.dropna(axis = 0)
        ca_df['zeros'] = 1
        ca_table=ca_df.pivot_table('zeros',index=['top shape','top pattern'],columns=['bottom shape','bottom pattern'])
        table = ca_table.fillna(0)

    elif style == 'Feminine':
        fe_df = df[df['style']=='Feminine']
        fe_df = fe_df[['top shape','top pattern', 'bottom shape', 'bottom pattern']]
        fe_df = fe_df.dropna(axis = 0)
        fe_df['zeros'] = 1
        fe_table= fe_df.pivot_table('zeros',index=['top shape','top pattern'],columns=['bottom shape','bottom pattern'])
        table = fe_table.fillna(0)

    else:
        fo_df = df[df['style']=='Formal']
        fo_df = fo_df[['top shape','top pattern', 'bottom shape', 'bottom pattern']]
        fo_df = fo_df.dropna(axis = 0)
        fo_df['zeros'] = 1
        fo_table= fo_df.pivot_table('zeros',index=['top shape','top pattern'],columns=['bottom shape','bottom pattern'])
        table = fo_table.fillna(0)

    if shape in ['Hoodie', 'Long-sleeve-tee', 'Shirts', 'Short-sleeve-tee', 'Sweater']:
        matched_lower = table.loc[(shape, pattern)].sort_values(ascending=False).index[0]
        matched_shape = matched_lower[0]
        matched_pattern = matched_lower[1]

    else :
        matched_upper = table.xs((shape,pattern), axis=1).sort_values(ascending=False).index[0]
        matched_shape = matched_upper[0]
        matched_pattern = matched_upper[1]

    # if matched_shape == 'Chiffon-skirt':
    #     matched_shape = 'A-line-skirt'

    return matched_shape, matched_pattern
```

<Casual 매트릭스 909>

| top shape | top pattern | bottom shape | | | | Chiffon-skirt | | | | Cotton-long-pants | | | | Cutoffs | | | |
|-----------------|-------------|--------------|-----|--------|---------|---------------|--------|-------|-----|-------------------|------|--------|-------|---------|--------|---------|-----|
| | | Check | Dot | Floral | Leopard | None | Stripe | Check | Dot | Leopard | None | Stripe | Check | Dot | Floral | Graphic | |
| Hoodie | Check | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Graphic | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 8.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 |
| | None | 0.0 | 2.0 | 0.0 | 1.0 | 4.0 | 0.0 | 0.0 | 0.0 | 0.0 | 25.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Check | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 8.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Dot | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Long-sleeve-tee | Graphic | 0.0 | 0.0 | 0.0 | 0.0 | 4.0 | 0.0 | 0.0 | 0.0 | 0.0 | 28.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | None | 2.0 | 0.0 | 0.0 | 0.0 | 11.0 | 3.0 | 1.0 | 0.0 | 2.0 | 91.0 | 0.0 | 3.0 | 0.0 | 1.0 | 2.0 | 0.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 1.0 | 9.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 |

<Feminine 매트릭스 219>

| top shape | top pattern | bottom shape | | | | Chiffon-skirt | | | | Cotton-long-pants | | | | Cutoffs | | | |
|-----------------|-------------|--------------|-----|--------|---------|---------------|--------|-------|-----|-------------------|------|--------|-------|---------|--------|---------|---------|
| | | Check | Dot | Floral | Leopard | None | Stripe | Check | Dot | Leopard | None | Stripe | Check | Dot | Floral | Graphic | Leopard |
| Hoodie | Check | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 |
| | Graphic | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 7.0 |
| | None | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 5.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 10.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 8.0 |
| | Check | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 |
| | Dot | 3.0 | 1.0 | 2.0 | 2.0 | 4.0 | 3.0 | 0.0 | 0.0 | 1.0 | 10.0 | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 6.0 |
| Long-sleeve-tee | Graphic | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | None | 3.0 | 1.0 | 2.0 | 2.0 | 4.0 | 3.0 | 3.0 | 1.0 | 5.0 | 0.0 | 2.0 | 1.0 | 1.0 | 0.0 | 0.0 | 6.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Check | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| Shirts | Check | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Dot | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Floral | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

<Formal 매트릭스 726>

| top shape | top pattern | bottom shape | | | | Chiffon-skirt | | | | Cotton-long-pants | | | | Cutoffs | | | |
|-----------------|-------------|--------------|-----|--------|---------|---------------|--------|-------|-----|-------------------|---------|-------|--------|---------|-----|--------|-----|
| | | Check | Dot | Floral | Leopard | None | Stripe | Check | Dot | Graphic | Leopard | None | Stripe | Check | Dot | Floral | |
| Hoodie | Check | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Graphic | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 6.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| | None | 4.0 | 0.0 | 0.0 | 0.0 | 4.0 | 2.0 | 0.0 | 1.0 | 0.0 | 0.0 | 15.0 | 0.0 | 1.0 | 2.0 | 0.0 | 0.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Check | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 4.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | Dot | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Long-sleeve-tee | Graphic | 1.0 | 2.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 22.0 | 0.0 | 2.0 | 0.0 | 1.0 | 0.0 |
| | Leopard | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | None | 3.0 | 3.0 | 2.0 | 1.0 | 8.0 | 3.0 | 0.0 | 1.0 | 1.0 | 2.0 | 106.0 | 1.0 | 6.0 | 0.0 | 0.0 | 0.0 |
| | Stripe | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 | 0.0 | 0.0 | 1.0 | 0.0 | 12.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

4. 프로젝트 결과

04. 프로젝트 수행 결과

[프로젝트 수행 결과]



상의 사진 입력

→ Shape : 반팔티
Pattern : 그래픽

입력값의 shape과
Pattern 분류

→ Feminine
 Casual
 Formal

Feminine 추천 매트릭스에서
그래픽_반팔티와 가장 많이
matching한 하의 찾기

→ Shape : 쉬폰스커트
Pattern : 체크



매트릭스를 구축 때 저장한
체크_쉬폰스커트 사진 폴더에서
사진 return

04. 프로젝트 수행 결과

[프로젝트 결과 평가]

- ▶ 1. 입력된 값에 대해 [floral, shirts]라고 **shape**과 **pattern**을 잘 구분함
- 2. 똑같은 상의에 대해 **style** 선택을 다르게 하면 그 **style**에 어울리는 하의를 추천해줌.

<Casual 추천 결과>



```
img_dir, style = input().split()
img = Image.open(img_dir)
show_recommend_img(img,style)
```

./data/18.jpg Casual
고객님의 옷의 패턴은 : Floral, 형태는 : Shirts
원하시는 스타일은: Casual 이런 옷은 어떠세요?



<Feminine 추천 결과>



```
img_dir, style = input().split()
img = Image.open(img_dir)
show_recommend_img(img,style)
```

./data/18.jpg Feminine
고객님의 옷의 패턴은 : Floral, 형태는 : Shirts
원하시는 스타일은: Feminine 이런 옷은 어떠세요?



<Formal 추천 결과>



```
img_dir, style = input().split()
img = Image.open(img_dir)
show_recommend_img(img,style)
```

./data/18.jpg Formal
고객님의 옷의 패턴은 : Floral, 형태는 : Shirts
원하시는 스타일은: Formal 이런 옷은 어떠세요?



04. 프로젝트 수행 결과

[시연 영상
- 하의추천]

온 이미지 입력받고 추천

www.BANDIGAM.com

View Insert Cell Kernel Help Trusted [CPU_ENV] O

학습된 모델 불러오기

```
[4]: model_pattern = './model/pattern_9905.h5'
var_pattern = ['Check', 'Dot', 'Floral', 'Graphic', 'Leopard', 'None', 'Stripe']

model_top = './model/upper_9970.h5'
var_top = ['Hoodie', 'Long-sleeve-tee', 'Shirts', 'Short-sleeve-tee', 'Sweater']

model_bottom = './model/lower_0425_9990.h5'
var_bottom = ['Chiffon-skirt', 'Cotton-long-pants', 'Cutoffs', 'H-line-skirt', 'Jumpsuit', 'Leggings', 'Pants', 'Shorts', 'Trousers']

model_style = './model/style_model_1_90_83.h5'
var_style = ['Casual', 'Feminine', 'Formal']

[5]: conv_base = VGG16(weights = 'imagenet', include_top=False, input_shape=(150,150))

top_model = models.Sequential()
top_model.add(layers.Dense(256, activation='relu', input_dim = 4 * 4 * 512))
top_model.add(layers.Dropout(0.5))
top_model.add(layers.Dense(4, activation='softmax'))
top_model.compile(optimizer = optimizers.RMSprop(r=2e-5),
                  loss='categorical_crossentropy',
                  metrics=['acc'])
top_model.load_weights(model_top)

[6]: pattern_model = models.Sequential()
pattern_model.add(layers.Dense(256, activation='relu', input_dim = 4 * 4 * 512))
pattern_model.add(layers.Dropout(0.5))
pattern_model.add(layers.Dense(7, activation='softmax'))
pattern_model.compile(optimizer = optimizers.RMSprop(r=2e-5),
                      loss='categorical_crossentropy',
                      metrics=['acc'])
pattern_model.load_weights(model_pattern)
```

여기에 이미지가 떠요!!

입력 옷 & 추천 옷



04. 프로젝트 수행 결과

[시연 영상
- 상의 추천]

온 이미지 입력받고 추천

www.BANDICAM.com

```
View Insert Cell Kernel Help
Run C Code
data = data.reindex([1,2,3,4,5,6,7,8,9,10])
print('옷과 스타일을 입력하세요.')
print('스타일은 0~9입니다.')
style = int(input())
image = Image.open('shoe.jpg')
image.show()
shape, pattern = recommend(df, style, identify_clothes(data, 1))
recommend_folder_dir = f'/data/recommend/{style}_{pattern}'
if not os.listdir(recommend_folder_dir):
    os.makedirs(recommend_folder_dir)
    random.shuffle(a)
    nua = int(nua)
    for img_name in a[:nua]:
        image = Image.open(recommend_folder_dir + '/' + img_name)
        image.show()

# --- 옷이미지를 입력해주세요 ---
img_dir = input()
print(f'{img_dir} 파일이 {img_dir} 폴더에 있습니다.')
style = input()
print('--- 평민인 Casual, Formal 스타일을 입력해주세요 ---')
style = input()
print('--- 추천 베스트 셔츠 및 수트를 입력해주세요 ---')
nua = input()
print('---')
print('--- 추천 및 추천 ---')
img = Image.open(img_dir)
show_recommend(img, style, nua)
```

여기에 이미지가 떠요!!

입력 옷 & 추천 옷



04. 보완점

1. Grabcut 알고리즘을 사용하면 배경제거가 가능.
-> 배경제거를 한다면 shape에 대한 정확도도 높이고,
색상 추출을 이용한 추천 모델도 가능할 것 같다.
2. Shape, style, pattern 특성을 세분화하면 더 다양한 옷에 대한 추천이 가능할 것
3. Style을 TPO로 확장. 면접록, 소개팅록 등 상황별 코디 추천으로 확장.
4. 모델 구현에 시간을 할애하느라 웹 구현 할 시간이 부족했다. 웹 구현을 해본다면 좋을 것

04. 느낀 점

| 팀원 | 느낀 점 |
|---------------|--|
| 임지수
(팀 리더) | 시작할 때만 하더라도 정말 가능할지, 시간이 충분할지 걱정이 많았다. 하지만 생각하는 것을 하나씩 구현해 나갔고 처음 팀원들과 생각한 그대로를 구현하게 되어 너무 만족스럽다. 특히 새로운 모델을 구현할 때, 혼자라면 쉽게 포기했을 부분을 같이 함으로써 포기하지 않을 수 있었고 많은 부분 배울 수 있었다. 이미지 처리에 대해서 집중적으로 다루면서 자신감도 생긴 것 같다. 다만 처음에 주어진 데이터를 너무 믿고 바로 사용했던 바람에 일정 시간을 허비한 것이 아까웠으며, 이번에 못 다한 구현은 나중에 개인적으로라도 꼭 해보고 싶다 다짐했다. |
| 박준원
(팀 원) | 이터를 확보하는 과정까지 혼난한 과정의 연속 이였습니다. 6개월이라는 시간 동안 많은 것을 배웠다고 생각했지만 공부했던 상황과는 많은 변수들이 존재했으며 데이터를 전처리하고 모델을 구현하는 과정은 생각보다 쉽지 않았습니다. 그래서 해당 주제와 관련된 분류 모델에 대한 심층적인 고민을 하게 되었고 다양한 기법을 공부해가며 점차 프로젝트가 진행될수록 스스로가 성장할 수 있었던 좋은 기회가 되었습니다. 점차 완성도를 높일 수 있을 것 같은 욕심이 생기면서 앞으로 공부 해 나가는데 있어 좋은 동기부여도 되었습니다. 팀 모두가 열심히 노력해서 나름 좋은 결과를 얻은 것 같아 기쁜 경험으로 남을 것 같습니다. |
| 이혜준
(팀 원) | 왜 데이터 분석에서 데이터 확보와 전처리가 70%라고 말씀하셨는지를 알 것 같았다. 모델 구축을 위해서 방대한 양의 이미지를 전처리 하는 것이 어려웠다. 그래도 팀 리더가 적극적으로 이끌어 주고 모든 팀원들이 자기 역할을 다하기 위해 최선을 다해서 좋은 결과를 완성할 수 있었던 것 같다. 이미지로 할 수 있는 전처리는 다 해본 것 같아서 이미지 모델 학습에 대해서 자세히 배울 수 있는 프로젝트였다. 원하던 프로젝트 결과를 이뤄내서 뿌듯하고 만족스럽다. |

감사합니다.