

무신사 크롤링 데이터를 이용한  
이미지 합성

PLAYDATA°

빅데이터 기반 지능형 **SW** 및 **MLOps** 개발자 양성과정

플레이데이터 **15기** 김가람

플레이데이터 **15기** 김병찬

플레이데이터 **15기** 김준태

플레이데이터 **15기** 김은식

# 목차

## I 서론

1. 주제
2. 배경 및 필요성
3. 관련 분석 사례
4. 관련 기술 소개

## II 본론

1. 본 주제의 차별성
2. 분석 시나리오 및 가설
3. 분석 과정 설계요약
4. 프로젝트 구현

## III 결론

1. 결론
2. 참고문헌

# I 서론

## 1. 주제

- 무신사 사이트 크롤링과 대시보드 구현 및 딥페이크를 활용한 이미지 & 동영상 합성

## 2. 배경 및 필요성

- 인터넷 쇼핑을 할 때 모델의 사진에 본인의 얼굴 사진을 합성을 함으로써, 미리 옷을 입혀보는 효과를 주어 구매자의 쇼핑 만족도를 높임
- 판매자의 입장에서 현재 판매되고 있는 상품들의 최신트렌드 및 인기 상품에 대한 정보를 취합하고 평균치를 낸 그래프를 시각화하여 최근 패션동향에 대한 이해를 높일 수 있도록 설계함.

## 3. 관련 분석 사례

- GAN 모델을 이용한 이미지 합성 기술의 활용 사례
  - GAN(Generative adversarial network, 생성적 적대 신경망)이란 비지도 학습에 사용되는 인공지능 알고리즘으로, 생성자(Generator)와 감별자(Discriminator) 두 개의 신경망 모델을 경쟁시켜 실제에 가까운 거짓 데이터를 생성하도록 만드는 심층학습(Deep Learning) 모델\*1이다. GAN은 2014년 논문\*1이 처음 발표된 이후로, 이미지 합성 및 생성 분야에서 많이 활용되어 왔다.\*1 예를 들어, 2018년에 Nvidia사가 개발한 Style GAN 기술\*2은 앞서 언급한 사례와 같은 실제와 구분하기 어려운 ‘가짜’ 이미지를 만들어 냈다.
  - 그러나 이렇게 고도화 된 ‘가짜 이미지’ 합성 및 생성 기술이 실존 인물을 대상으로 합성하는 포르노나 가짜 정보 생성 및 유통에 악용되는 문제도 발생하고 있다. 여기에 대항해 삼성 SDS 사내벤처가 다시 GAN 기술을 활용하여 이미지 합성을 탐지해내는 기술을 개발한 사례가 있다.\*3
  - 이러한 흐름을 타고, 이미지 합성과 관련한 데이터 및 AI 경연대회에서 GAN 기술 활용을 언급하거나 주제로 삼는 경우도 보이고 있다. 2022년 10월 4일부터 11월 14일까지 열린 ‘월간 데이콘 예술 작품 화가 분류 AI 경진대회’에서는 GAN 기술을 언급하며, 비지도 학습으로 그림 작가를 추론하는 AI 제작을 목표로 제시했다.\*4 Kaggle에서는 ‘I’m Something of a Painter Myself’이라는 이름의 경연대회에서 GAN을 이용하여 모네의 화풍을 따라하는 이미지 제작 AI 개발을 과제로 제시하였다.\*5
- AI 앵커 및 가상 아이돌
  - 위와 같은 딥페이크 기반 기술을 이용하여 실존하는 사람을 모방한 AI 3D 모델을 만들어 TV 뉴스에 활용하거나, 아예 허구의 인물을 창조하여 가상 아이돌, 연예인으로 활용하는 사례가 많아지고 있다. 전자의 경우 MBN의 앵커 김주하 씨의 모습을 기반으로 만든 ‘AI 김주하’의 사례가 있으며\*6, 후자의 경우 신한라이프 등의 광고 등에 출연한 사례가 있는, 일명 ‘버추얼 인플루언서 ‘오로지’ 등이 있다.\*7

## 4. 관련 기술 소개

- 딥페이크를 기반으로 한 SimSwap 프로그램을 사용한다.



### 딥페이크(DeepFake)란?

- 딥러닝(Deep Learning) + 페이크 (Fake)의 합성어로, 과거 포토샵을 활용해 어떤 사진이나 영상물에 다른 사람 얼굴이나 없던 물건을 합성하듯이, 인공지능을 이용해 사람 얼굴을 합성하는 기술을 뜻한다.

### SimSwap

9,000 + identities

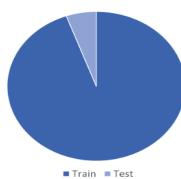
VGGFace2 contains images from identities spanning a wide range of different ethnicities, accents, professions and ages.



Gender Distribution

3.3 million + faces

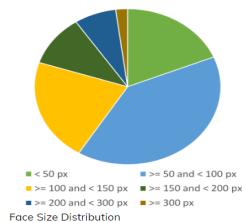
All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions.



Train/Test Split

362 ~ per-subject samples

Face distribution for different identities is varied, from 87 to 843, with an average of 362 images for each subject.



Face Size Distribution

### 훈련데이터셋(VGGFace2 Dataset)

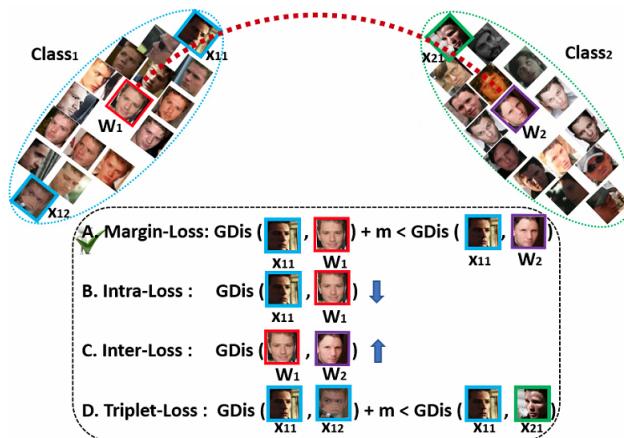
약 9000개의 특징을 가지고 362가지의 주제로 되어있으며, 330만명 정도의 데이터가 담긴 데이터셋으로 모델을 학습 시킴

## 학습 알고리즘모델

	224 × 224		320 × 320 / 299 × 299	
	top-1 err.	top-5 err.	top-1 err.	top-5 err.
ResNet-152 [10]	23.0	6.7	21.3	5.5
ResNet-200 [11]	21.7	5.8	20.1	4.8
Inception-v3 [44]	-	-	21.2	5.6
Inception-v4 [42]	-	-	20.0	5.0
Inception-ResNet-v2 [42]	-	-	19.9	4.9
ResNeXt-101 (64 × 4d) [47]	20.4	5.3	19.1	4.4
DenseNet-264 [14]	22.15	6.12	-	-
Attention-92 [46]	-	-	19.5	4.8
Very Deep PolyNet [51] <sup>†</sup>	-	-	18.71	4.25
PyramidNet-200 [8]	20.1	5.4	19.2	4.7
DPN-131 [5]	19.93	5.12	18.55	4.16
<b>SENet-154</b>	<b>18.68</b>	<b>4.47</b>	<b>17.28</b>	<b>3.79</b>
NASNet-A (6@4032) [55] <sup>†</sup>	-	-	17.3 <sup>‡</sup>	3.8 <sup>‡</sup>
<b>SENet-154 (post-challenge)</b>	-	-	<b>16.88<sup>‡</sup></b>	<b>3.58<sup>‡</sup></b>

위 이미지 처리에 사용 된 알고리즘은 SENet-154이며, 이는 SE block을 개정된 ResNeXt와 통합한 것을 의미합니다. SENet-154는 CNN을 기반으로 한 모델이며, 이미지 분류(image classification) 성능을 평가하는 지표인 top-5 error와 top-1 error에서 이미지 분류를 시행했을때, top-5 error가 4.5% 이하로 동기간 성능을 측정한 알고리즘 중 우월한 성능을 내는 모델(SENet-154)을 사용하고 있다.

## Insight Face



InsightFace는 2D & 3D의 이미지 처리를 위하여 PyTorch 와 MXNet을 기반으로 하는 오픈소스이며, 얼굴 분석을 위한 통합 파이썬 라이브러리입니다. InsightFace는 얼굴 인식, 얼굴 감지 및 얼굴 정렬과 같은 다양한 최첨단 알고리즘을 효율적으로 구현한다.

## II 본론

### 1. 본 주제의 차별성

- 취합된 데이터를 특정 형태로 저장하는 스크레이핑(Scraping) 기능
- 스트림릿(StreamLit) 모듈을 사용하여 그래프 형식으로 시각화를 할 수 있는 기능
- 딥페이크(DeepFake) 기술을 활용한 이미지 합성 기능

### 2. 분석 시나리오 및 가설

1. 파이참에서 대시보드 기능을 하는 스트림릿 코드 실행
2. 대시보드 창에서 찾고 싶은 의류의 검색어 입력 및 페이지 수 지정 후 크롤링코드 실행
3. 크롤링 된 데이터들을 바탕으로 판매금액 분포, 판매량 상위 10개 브랜드의 시각화 및 자료 분석 진행
4. 크롤링 된 csv파일에서 url 주소로 이뤄진 데이터를 JPG 파일로 변환하여 저장함
5. 코랩을 통해 변환된 이미지 파일을 같은 사이즈로 리사이징하고 JPG파일을 MP4 파일로 변환
6. 변환된 MP4파일과 본인의 이미지를 합성진행, 이후 합성된 MP4 파일을 프레임별로 나눠서 JPG파일로 저장
7. 저장된 JPG 파일과 MP4파일을 대시보드에 노출 시켜 원본 사진과 합성된 사진을 비교하여 볼 수 있음

### 3. 분석 과정 설계요약



#### 4. 프로젝트 구현

- 크롤링 코드 및 대시보드 화면

The screenshot shows two windows side-by-side. On the left is PyCharm with an open file named 'sin.py' containing Python code for a Streamlit application. The code includes sidebar creation, checkbox logic for filtering products by brand, and button handling. On the right is a web browser displaying the Streamlit application. The title bar says 'sin - Streamlit'. The main content area has a large green header 'MUSINSA'. Below it is a section titled '상품검색' with a sub-section '상품명'. A table lists products with columns for ID, 가격 (Price), 할인가격 (Discount Price), 브랜드 (Brand), and 조회수 (View Count). One row is selected, showing product ID 1, price 143000, discount price 89900, brand 라파이스토어, and view count 37.1만 회 (37.1k views). At the bottom of the Streamlit app is a '크롤링' (Crawling) button.

```
# 상품 검색창
st.sidebar.header("상품검색")
key_word = st.sidebar.text_input('상품명')
d_li.append(key_word)

page_num = st.sidebar.number_input('검색 페이지 수', min_value=1, max_value=50, value=1)
d_li.append(page_num)

# 크롤링 데이터 부르기
key_word2 = st.sidebar.write('어떤 데이터를 크롤링 하시겠습니까?')
checked1 = st.sidebar.checkbox('제품명')
checked2 = st.sidebar.checkbox('가격')
checked3 = st.sidebar.checkbox('할인가격')
checked4 = st.sidebar.checkbox('브랜드')
checked5 = st.sidebar.checkbox('세부정보')

# 버튼클릭 처리
if 'button_1' not in st.session_state:
    st.session_state.button_1 = False
if 'button_2' not in st.session_state:
    st.session_state.button_2 = False

def cb1():
    pass
```

	가격	할인가격	브랜드	조회수
패딩	2037173	69900	62910	스파우 <NA>
검색페이지 수	2814804	219000	197100	디스이즈네버댓 28.6만 회
1	1173366	143000	89900	라파이스토어 37.1만 회
	2835510	129000	103200	예일 33.1만 회
<input checked="" type="checkbox"/> 제품명	2704962	99000	79200	예일 76.2만 회
<input type="checkbox"/> 가격	2037167	69900	62910	스파우 43.2만 회
<input type="checkbox"/> 할인가격	2081586	119000	83300	커버낫 27.6만 회
<input type="checkbox"/> 브랜드	2483269	339000	339000	노스페이스 32.1만 회
<input type="checkbox"/> 세부정보	912314	119900	119900	무신사 스탠다드 14.3만 회
<input type="checkbox"/> 크롤링	2877785	173000	128000	라파이스토어 25.2만 회

- 크롤링한 데이터를 **CSV** 파일로 추출

- 크롤링한 csv 파일을 데이터프레임화

## DataFrame 데이터

	제품명	상품url	가격	할인가격	브랜드	조회수
0	베이직 푸퍼_SPJPC4TC11	<a href="https://www.musinsa.com/app/goods/2037173">https://www.musinsa.com/app/goods/2037173</a>	69900	62910	스파오	<NA>
1	PERTEX® T Down Jacket Black	<a href="https://www.musinsa.com/app/goods/2814804">https://www.musinsa.com/app/goods/2814804</a>	219000	197100	디스이즈네버댓	28.6만 회 0
2	[구교환 착용] [무봉제] 덕다운 심리스 미니멀 푸퍼 숏패딩_Blk	<a href="https://www.musinsa.com/app/goods/1173366">https://www.musinsa.com/app/goods/1173366</a>	143000	89900	라퍼지스토어	37.1만 회 0
3	REVERSIBLE WARM UP QUILTING JACKET BLACK / IVORY	<a href="https://www.musinsa.com/app/goods/2835510">https://www.musinsa.com/app/goods/2835510</a>	129000	103200	예일	33.1만 회 0
4	(22FW) WARM+ UP QUILTING JACKET BLACK	<a href="https://www.musinsa.com/app/goods/2704962">https://www.musinsa.com/app/goods/2704962</a>	99000	79200	예일	76.2만 회 0
5	파스텔 푸퍼_SPJPC4TG01	<a href="https://www.musinsa.com/app/goods/2037167">https://www.musinsa.com/app/goods/2037167</a>	69900	62910	스파오	43.2만 회 0
6	RDS 커버라이트 카라리스 다운 점퍼 블랙	<a href="https://www.musinsa.com/app/goods/2081586">https://www.musinsa.com/app/goods/2081586</a>	119000	83300	커버낫	27.6만 회 0
7	NJ1DN75A 남성 1996 에코 눌시 자켓	<a href="https://www.musinsa.com/app/goods/2482269">https://www.musinsa.com/app/goods/2482269</a>	339000	339000	노스페이스	32만 회 이상
8	다운 푸퍼 숏 패딩 재킷 [블랙]	<a href="https://www.musinsa.com/app/goods/912314">https://www.musinsa.com/app/goods/912314</a>	119900	119900	무신사 스탠다드	14.3만 회 0
9	덕다운 아르텍 후드 패딩_Blk	<a href="https://www.musinsa.com/app/goods/2877785">https://www.musinsa.com/app/goods/2877785</a>	173000	128000	라퍼지스토어	25.2만 회 0

- 코랩에서 필요 모듈 설치 및 드라이브 마운트

```
# 필요한 모듈 설치
!pip install ffmpeg
!pip install insightface==0.2.1 onnxruntime moviepy
!pip install googleapiclient
!pip install imageio==2.4.1

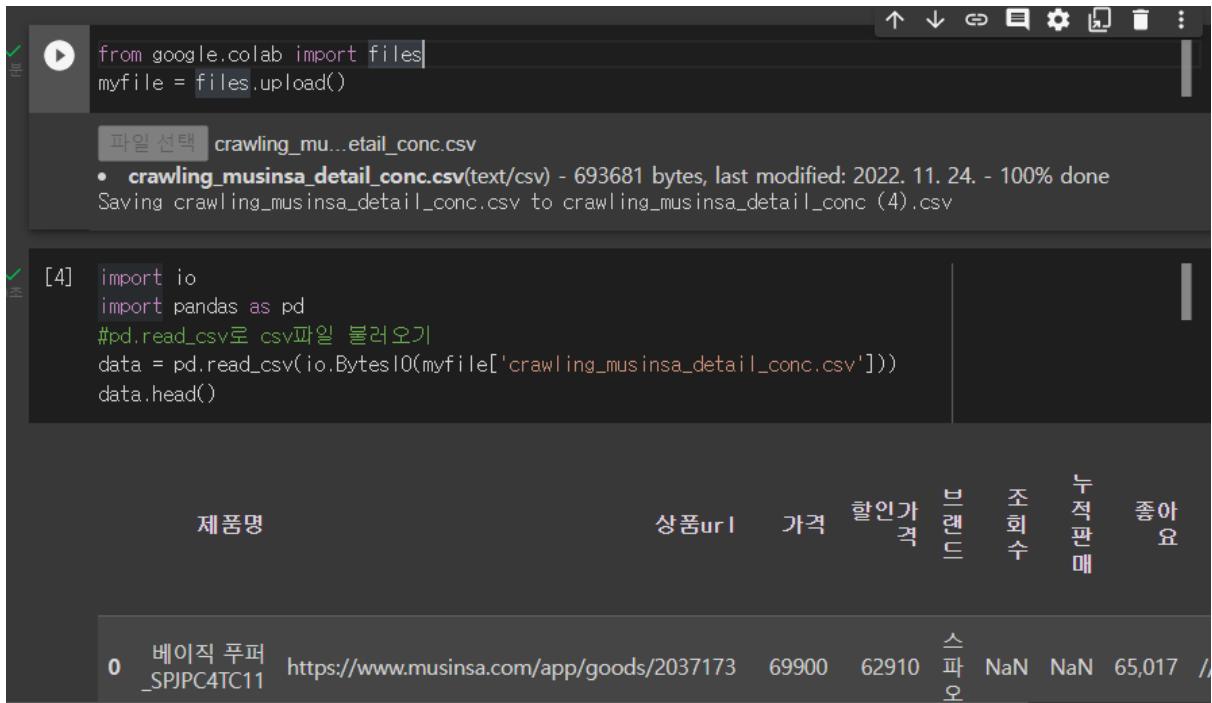
# 드라이브 마운트
from google.colab import drive
drive.mount('/content/drive')

# 작업위치 변경
%cd drive/MyDrive/pro/SimSwap_colab

# 모듈 불러오기
import cv2
import torch
import fractions
import numpy as np
from PIL import Image
import torch.nn.functional as F
from torchvision import transforms
from models.models import create_model
from options.test_options import TestOptions
from insightface_func.face_detect_crop_multi import Face_detect_crop
from util_videoswap import video_swap
from util.add_watermark import watermark_image
import moviepy.video.io.ImageSequenceClip
import glob
import os
from PIL import Image
```

The image shows a screenshot of a web browser window displaying the Google OAuth login page. The URL in the address bar is accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?access\_type=offline&cli... The main content is a 'Google 계정으로 로그인' (Log in with Google) button. Below it, there's a '계정 선택' (Select account) section with a 'Google Drive for desktop(으)로 이동' (Move to Google Drive for desktop) link. A user profile for '김병찬' (Kim Byungchan) is shown, with the email 'qudcks14644@gmail.com'. At the bottom, there's a note about continuing the process by granting permissions to Google Drive for desktop.

- CSV 파일 불러오기



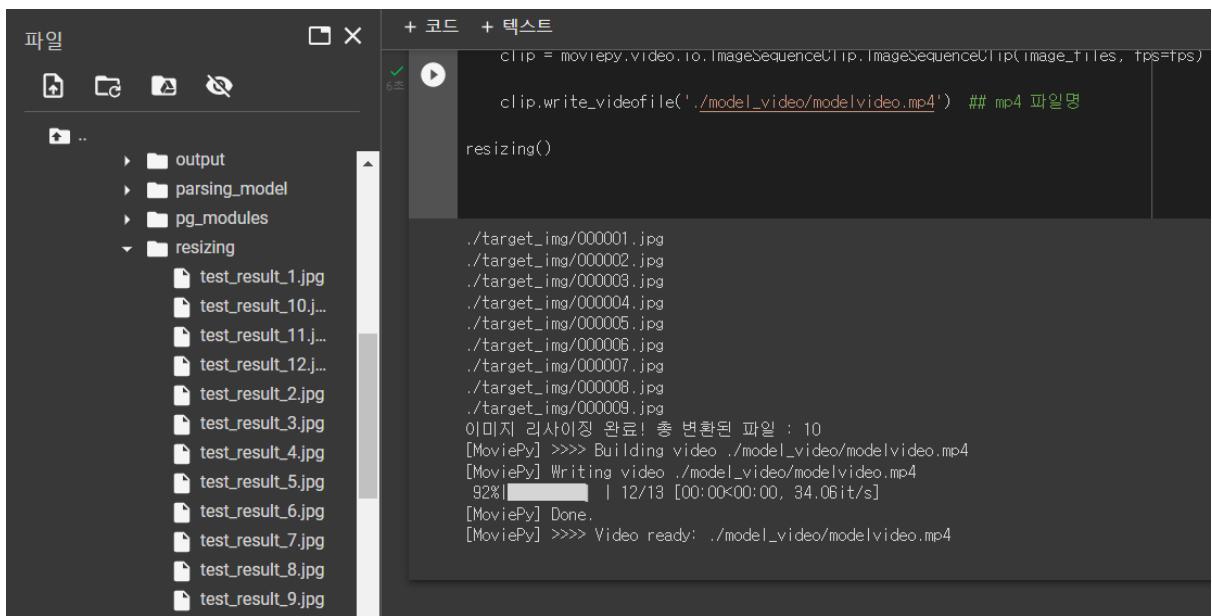
from google.colab import files  
myfile = files.upload()

파일 선택 crawling\_mu...etail\_conc.csv  
• crawling\_musinsa\_detail\_conc.csv(text/csv) - 693681 bytes, last modified: 2022. 11. 24. - 100% done  
Saving crawling\_musinsa\_detail\_conc.csv to crawling\_musinsa\_detail\_conc (4).csv

```
[4]: import io
import pandas as pd
#pd.read_csv로 csv파일 불러오기
data = pd.read_csv(io.BytesIO(myfile['crawling_musinsa_detail_conc.csv']))
data.head()
```

제품명	상품url	가격	할인가격	브랜드	조회수	누적판매	좋아요
베이직 푸퍼 _SPJPC4TC11	<a href="https://www.musinsa.com/app/goods/2037173">https://www.musinsa.com/app/goods/2037173</a>	69900	62910	스파오	NaN	NaN	65,017

- JPG 파일들 리사이징 & JPG파일 MP4변환



+ 코드 + 텍스트

```
clip = moviepy.video.io.ImageSequenceClip(image_files, fps=fps)
clip.write_videofile('./model_video/modelvideo.mp4') ## mp4 파일명
```

resizing()

```
./target_img/000001.jpg
./target_img/000002.jpg
./target_img/000003.jpg
./target_img/000004.jpg
./target_img/000005.jpg
./target_img/000006.jpg
./target_img/000007.jpg
./target_img/000008.jpg
./target_img/000009.jpg
이미지 리사이징 완료! 총 변환된 파일 : 10
[MoviePy] >>> Building video ./model_video/modelvideo.mp4
[MoviePy] Writing video ./model_video/modelvideo.mp4
92% [███████] | 12/13 [00:00<00:00, 34.06it/s]
[MoviePy] Done.
[MoviePy] >>> Video ready: ./model_video/modelvideo.mp4
```

- **JPG와 MP4 파일 합성**



```

        transforms.Normalize([-0.485, -0.485, -0.485], [1, 1, 1])

    ])

opt = TestOptions()
opt.initialize()
opt.parser.add_argument('-f')
opt = opt.parse()
opt.pic_a_path = './demo_file/meme.jpg' ## 내 얼굴사진 경로
opt.video_path = './model_video/modelvideo.mp4' ## 합성 할 MP4 파일경로
opt.output_path = './output/demo1.mp4'
opt.temp_path = './tmp'
opt.arc_path = './arcface_model/arcface_checkpoint.tar'
opt.isTrain = False
opt.use_mask = True

crop_size = opt.crop_size

torch.nn.Module.dump_patches = True
model = create_model(opt)
model.eval()

app = Face_detect_crop(name='antelope', root='./insightface_func/models')
app.prepare(ctx_id= 0, det_thresh=0.6, det_size=(640,640))

with torch.no_grad():
    pic_a = opt.pic_a_path
    # img_a = Image.open(pic_a).convert('RGB')
    img_a_whole = cv2.imread(pic_a)
    ...

```

- **MP4파일에서 JPG파일로 프레임 나눠주기**



▼ MP4파일에서 JPG파일로 프레임 나눠주기

```

[6] import cv2
import os

vidcap = cv2.VideoCapture('./output/demo1.mp4') #합성된 동영상 경로설정 후 프레임캡쳐
success, image = vidcap.read()
count = 0 #초기값 설정

if os.path.exists('./cp_result'): #폴더 초기화 경로 설정
    for file in os.scandir('./cp_result'):
        os.remove(file.path) #파일 내 문서 삭제

while success:
    cv2.imwrite("./cp_result/%06d.jpg" % count, image) # 프레임을 캡쳐후 jpg로 저장
    success, image = vidcap.read()
    print('Read a new frame: ', success)
    count += 1

print("finish! convert video to frame")

```

Read a new frame: True  
Read a new frame: False  
finish! convert video to frame

- PyCharm에서 스트림릿 실행하여 결과 확인하기

The screenshot shows the PyCharm IDE interface. On the left, the code editor displays Python code for a Streamlit application named 'gui'. The code includes imports for Streamlit (st), session state (st.session\_state), and sidebar buttons. It defines a function cb1() that sets button\_1 to True. The sidebar contains a button labeled '합성' (Composite) with an on\_click handler. A terminal window at the bottom shows the command to run the app: 'You can now view your Streamlit app in your browser.' followed by local and network URLs.

On the right, a browser window shows the Streamlit application. It has a sidebar with a file upload area for composite images. Below it, there are two images: a full-body photo of a person in a black jacket and beige trim, and a close-up of the person's head and shoulders. A video player interface shows a thumbnail of the person and a progress bar at 0:02 / 0:02. A '다음' (Next) button is visible at the bottom right of the image area.

```
22
23     if 'button_1' not in st.session_state:
24         st.session_state.button_1 = False
25     if 'counter' not in st.session_state:
26         st.session_state.counter = 0
27
28
29     def cb1():
30         st.session_state.button_1 = True
31
32
33     # 사이드바 합성버튼 생성
34     st.sidebar.button("합성", on_click=cb1)
35
36     # li = []
37     if 'count' not in st.session_state:
38         st.session_state.count = 0
39
40     if st.session_state.button_1:
41         if image_file is None:
42             st.text('사진을 업로드해주세요')
43         else:
```

You can now view your Streamlit app in your browser.  
Local URL: <http://localhost:8501>  
Network URL: <http://192.168.0.48:8501>

### III 결론

#### 1. 결론

- 상기 프로젝트를 통해 만들어진 프로그램으로 크롤링을 진행하여, 취합된 데이터를 통해 소비자의 구매 경향이나 브랜드점유율, 상품의 가격분포정도를 시각화하여 볼 수 있도록 설계하였고, 이미지합성을 통해 소비욕구를 증진시켜주거나 쇼핑에 대한 만족도를 높혀줌으로서 좀 더 합리적인 판단을 내릴 수 있다

아쉬운 점으로는 코드상으로 사이트를 직접 들어가서 크롤링 코드를 작동해야 하기 때문에 속도가 느리다는 점, 많은 양의 크롤링을 진행할 시에 트래픽과 부하문제로 IP밴을 당할 수 있는 점을 주의해야 한다.

최초 초기설정에서는 JPG 파일과 JPG파일을 합성하여 JPG파일을 만드는 것으로 목적을 두고 프로젝트를 진행하였으나, JPG파일끼리 합성을 하면 다소 부자연스럽게 합성이 되어 JPG파일들을 MP4 파일로 변경하는 과정을 추가하여 처리과정이 매끄럽지 못한점과 크롤링한 사진 중 모델 전체 컷이 아닌 상품사진이나 얼굴이 잘려서 나온사진들을 제거하는 작업을 추가하지 못하여서 추후 동일한 프로젝트 진행 시에 보완 할 점이라고 생각된다.

프로젝트를 마치며, 크롤링 코드를 작성하면서 처리속도에 대한 고민을 하면서 초기 셀레니움으로 시작했던 모델이 bs4와 멀티프로세싱을 추가적으로 적용하여 빠른 속도로 크롤링이 가능하도록 보완을 하면서 크롤링에 대한 이해에 도움이 많이 되었고 이미지 처리과정에서 JPG 파일의 사이즈를 맞추는 것 PyTorch 기반으로 한 cnn모델을 사용함으로써 Layer에 대한 전반적인 이해와 이미지 전체보다는 픽셀부분의 연관성을 살리는 이미지 처리에 대한 공부가 많이 되었던 프로젝트 경험이라고 생각된다.

- 합성 **Before / After**



## 2. 참고문헌

- [네이버 지식백과] [크롤링](#) [crawling] (용어로 알아보는 우리 시대 DATA)
- 딥페이크 모델
  - SimSwap : <https://github.com/neuralchen/SimSwap>
  - StyleGAN : <https://github.com/NVlabs/stylegan>
  - etos-faceswap : <https://github.com/etosworld/etos-faceswap>
- 크롤링 모듈
  - <https://www.selenium.dev/documentation/>
  - <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- 파이썬 공식 문서의 멀티프로세싱
  - <https://docs.python.org/ko/3/library/multiprocessing.html>
- 관련 분석 연구
  - 1 이기범, “[IT열쇳말] GAN(생성적 적대 신경망)”,  
<https://www.bloter.net/newsView/blt201806080001>
  - 2 Wikipedia, “StyleGAN”, 웹페이지  
인용<https://en.wikipedia.org/wiki/StyleGAN>
  - 3 삼성 SDS, “AI를 활용한 멀티미디어 위변조에 대응하는 삼성 SDS 사내벤처 팀나인”, 삼성 SDS 인사이트 리포트, 2022년 2월 11일.  
[https://www.samsungsds.com/kr/insights/220211\\_team9.html](https://www.samsungsds.com/kr/insights/220211_team9.html)
  - 4 Dacon, ‘월간 데이콘 예술 작품 화가 분류 AI 경진대회’, 웹페이지 인용,  
<https://dacon.io/competitions/official/236006/overview/description>
  - 5 Kaggle, ‘I’m Something of a Painter Myself-Use GANs to create art - will you be the next Monet?’, 웹페이지  
인용<https://www.kaggle.com/competitions/gan-getting-started/rules>
  - 6 양재영, “김주하 앵커 아니었어? 소름돋는 AI 아나운서 데뷔 장면”,  
국민일보 <https://m.kmib.co.kr/view.asp?arcid=0015203470>
  - 7 김형자, “누가 진짜 인간? 가상인간이 일상 속으로”, 주간조선,  
<http://weekly.chosun.com/news/articleView.html?idxno=22235>