

# 프로젝트 명세서

Google Colab 기반의

Generative Adversarial Network

응용 PJT

## 목 차

1. 프로젝트 개요.....	3
2. 기본과제 .....	5
3. 심화과제 .....	11
4. 과제 제출방법.....	25

## 1. 프로젝트 개요

---

2014 년 Ian J. Goodfellow 는 최근 10 년간 Machine Learning 분야에서 나온 가장 멋진 아이디어라고 평가되는 Generative Adversarial Network (생성적 적대 신경망, GAN)을 발표합니다. GAN 은 간단히 말하자면 적대적인 관계에 있는 2 개의 모델들이 상호간의 Output 을 기반으로 학습을 하여 정확도를 높여가는 개념이라고 할 수 있습니다.

하지만 인공지능 학습을 해보지 않은 사람이 GAN 을 하루 만에 학습해서 습득하기는 어렵기 때문에 이번 프로젝트에서는 GAN 에 대한 학습 및 구현이 아닌 GAN 에 근간을 둔 여러가지 재미있는 프로젝트들을 웹브라우저 기반 Python 실행 환경인 Google Colab 을 사용해 진행하고자 합니다.

본 프로젝트를 진행하면서 GAN 에 대한 호기심이 생기셨다면 자기주도학습을 통해 꼭 코드 레벨로 GAN 에 대해 학습해 보시기 바랍니다.



[Fig.1] GAN 을 사용해 합성해낸 인물 영상

본 프로젝트의 목표는 다음과 같습니다.

1. Python 코드 실행 환경인 Google Colab 의 기본적인 사용법을 익혀봅니다.
2. Colab 을 통해 Python 코드를 GPU 가속을 적용해 실행해보고 Google Drive 와 연동해 수행결과를 읽고 쓸 줄 알게 됩니다.
3. Colab 기반으로 OpenCV API 를 사용하고 출력하는 방법을 알게 됩니다.
4. Generative Adversarial Network (GAN)의 개념을 익혀봅니다.
5. GAN 을 사용해 구현한 First Order Model 에 대해 알아보고 Colab 에서 실습해 봅니다.
6. 기존에 사용해 보지 못한 새로운 라이브러리 / 패키지를 레퍼런스 문서를 보고 적용하는 방법을 경험해 봅니다.
7. 본 프로젝트를 통해 GAN 에 대해 관심을 갖게 된 교육생들이 차후 본격적인 자기주도 학습 진행 시 Machine Learning 및 GAN 모델을 활용한 프로젝트를 기획해 볼 수 있게끔 관련 정보를 제공 합니다.

Python 을 한번도 사용해보지 않은 전공학생들은 다음의 링크를 참조하여 Java 와의 차이점을 빠르게 습득해보시기 바랍니다. 코드량이 많지 않으니 부담없이 시도해 보세요.

 <https://haloper.tistory.com/32>

## 2. 기본과제

---



### 2-1. Google Colab

Python 을 학습하는 과정에서 웹 브라우저 상에서 소스코드를 작성하고 한 줄씩 실행 및 디버깅을 할 수 있는 Jupyter Notebook 을 많이 사용하셨을 겁니다.

Colab 은 Jupyter Notebook 에 추가로 Python 소스코드를 Google 의 클라우드 컴퓨팅 환경에서 GPU 와 TPU 를 무료로 사용 할 수 있고 소스코드나 데이터를 Google Drive 를 통해 불러오거나 저장할 수도 있는 개발 환경 입니다. Cloud 기반이므로 별도의 설치과정이 필요 없으며 딥러닝, M/L, 데이터 사이언스 분야에서 사용되고 있습니다.

## Google Colab(oratory)

- For you Jupyter Notebook fans, this is an even better option!

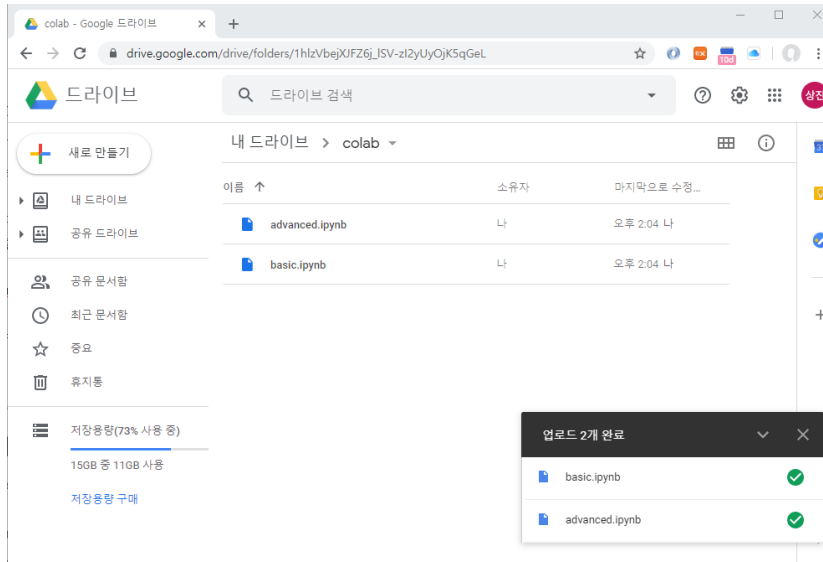
- 1) Hosted by Google
  - a) Extremely fast network speed
- 2) Access to GPU and TPU (Tensor Processing Unit)
  - a) Not something you can buy for your PC!
  - b) TF code works the same on all devices
- 3) Stored in Google Drive (the "cloud")
  - a) You'll never lose it, and it's easy to share
- 4) Many libraries for deep learning / machine learning / data science
  - a) More than I assumed there would be! (Theano, PyTorch)

다음의 링크를 참조하여 Colab 에 대해 학습해 보시기 바랍니다.

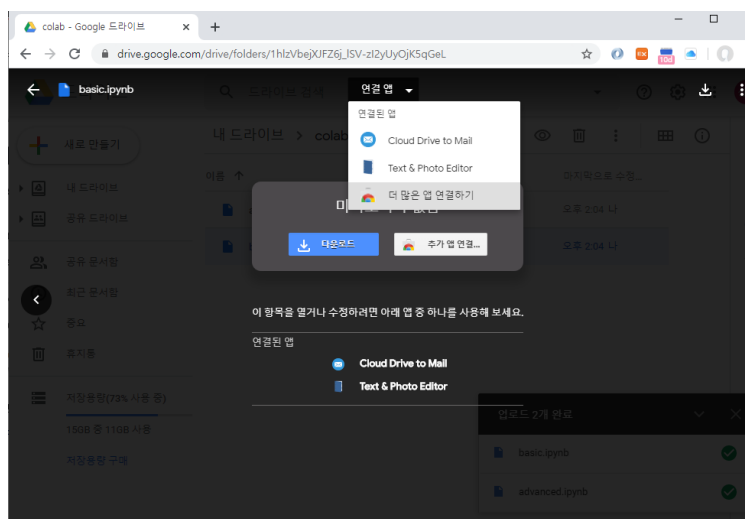
🔗 [https:// colab.research.google.com/](https://colab.research.google.com/)

## 2-2. Google Drive 의 ipynb 파일과 Colab 연결하기

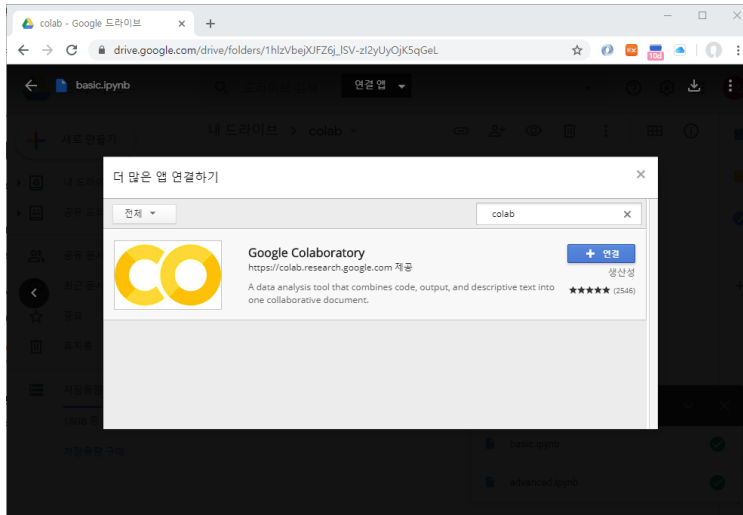
Google Drive 에 colab 이라는 디렉토리를 생성한 후, 전달받은 basic.ipynb 와 advanced.ipynb 파일을 업로드 합니다.



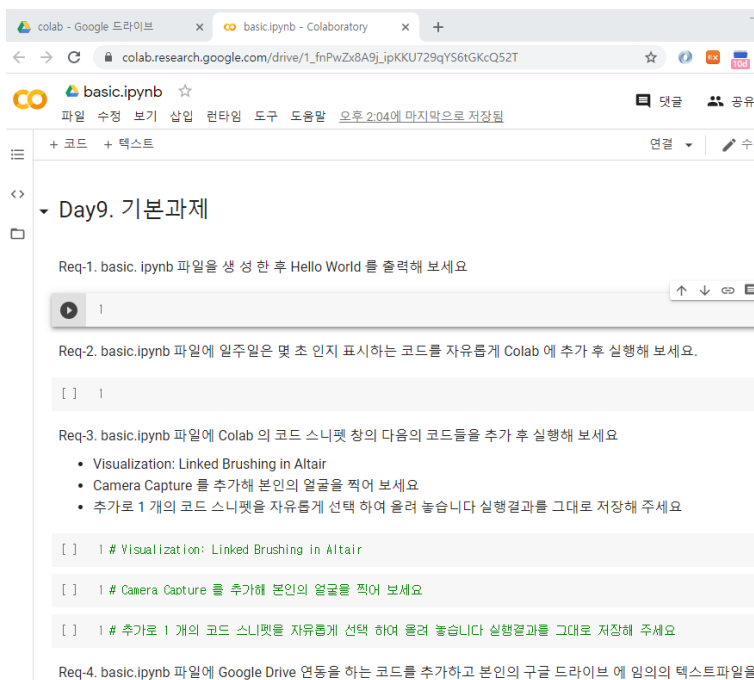
이후, 업로드 된 basic.ipynb 을 더블클릭 한 후 상단의 “연결 앱” 버튼을 누르면 다음과 같이 “더 많은 앱 연결하기” 를 선택 할 수 있습니다.



이후 다음과 같이 검색창에서 “colab” 을 검색하고 “연결” 버튼을 누르면 구글 드라이브의 ipynb 파일을 colab 과 연결 할 수 있습니다.




연결 완료후 ipynb 파일을 더블클릭하면 다음과 같이 Colab 으로 연동이 됩니다.



Req-1	제공된 basic.ipynb 파일상에서 Hello World 를 출력해 보세요.
-------	--

Req-2	basic.ipynb 파일에 일주일은 몇 초 인지 표시하는 코드를 자유롭게 Colab 에 추가 후 실행해 보세요.
-------	---

## 2-2. Google Colab - 코드 스니펫

Colab 에서는 재사용이 가능한 코드와 다양한 예제가 좌측의  모양의 아이콘을 클릭하면 나옵니다. 이를 코드 스니펫 창이라고 합니다. Req-3 을 수행해 봅니다.

Req-3	<p>basic.ipynb 파일에 Colab 의 코드 스니펫 창의 다음의 코드들을 추가 후 실행해 보세요.</p> <ul style="list-style-type: none"> <li>- Visualization: Linked Brushing in Altair</li> <li>- Camera Capture 를 추가해 본인의 얼굴을 찍어 보세요.</li> <li>- 추가로 1 개의 코드 스니펫을 자유롭게 선택하여 올려 놓습니다.</li> </ul> <p>실행결과를 그대로 저장해 주세요.</p>
-------	---

## 2-3. Google Colab - Google Drive 연동

Colab 에서는 간단한 인증과정을 거쳐 Google Drive 와 연동을 하여 데이터를 읽어오거나 연산결과를 저장 할 수 있습니다.

Req-4	basic.ipynb 파일에 Google Drive 연동을 하는 코드를 추가하고 본인의 구글 드라이브에 임의의 텍스트파일을 저장하는 코드를 작성해보세요. 코드 스니펫의 Mounting Google Drive in your VM 을 사용해도 무방합니다.
-------	--



## 2-4. Colab 에서 GPU / TPU 사용

Colab 에서는 Google 에서 제공하는 GPU 와 TPU 를 몇가지 제약사항이 있지만 무료로 사용해 코드를 실행 할 수 있습니다.

Req-5	<p>다음의 GPU 가속 예제코드를 실행해 보고 CPU 사용시 처리시간과 GPU 사용시 처리시간을 비교해 보고 마지막 블럭인 “<b>Observe TensorFlow speedup on GPU relative to CPU</b>”의 결과를 복사해서 basic.ipynb 의 Req-5 다음에 텍스트 형태로 붙여 주세요.</p> <p><a href="https://colab.research.google.com/notebooks/gpu.ipynb">https://colab.research.google.com/notebooks/gpu.ipynb</a></p>
-------	---

## 2-5. Colab 에서 OpenCV 라이브러리를 사용한 그래픽 처리

OpenCV 는 무료로 공개된 컴퓨터 비전 라이브러리로서 대부분의 OS 를 지원하며 C/C++ 프로그래밍 언어로 개발 되었으나 Python, Java 및 MATLAB 에 바인딩 된 인터페이스가 있어 Windows, Linux, Android 및 MacOS 를 지원합니다. 영상 관련 라이브러리로서 사실상 표준의 지위를 가지고 있다고 할 수 있으며 조금이라도 영상처리가 들어간다면 필수적으로 사용하게 되는 라이브러리 입니다.

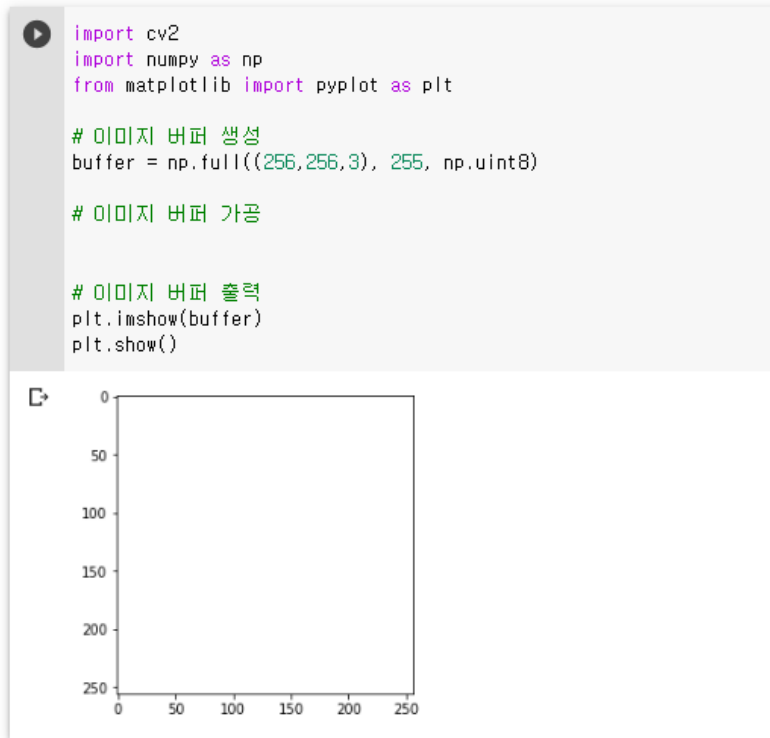
라이브러리에는 2500 개가 넘는 최적화 된 알고리즘이 있으며 여기에는 클래식 및 최신 컴퓨터 비전과 머신 러닝 알고리즘이 모두 포함됩니다. 이 알고리즘은 얼굴 감지하고 인식하고, 물체를 식별하고, 비디오에서 사람의 행동을 분류하고, 카메라 움직임을 추적하고, 움직이는 물체를 추적하고, 물체의 3D 모델을 추출하고, 스테레오 카메라에서 3D 포인트 클라우드를 생성하고, 이미지를 연결하여 고해상도를 생성하는 데 사용될 수 있습니다.

본 명세서에서는 OpenCV 의 기능 중 단순히 인식한 얼굴에 사각형을 그리는 수준의 Drawing API 만 사용합니다. 따라서 잠시 언급하고 넘어가는 수준 입니다만, OpenCV 는 여러모로 활용도가 높은 라이브러리 이므로 어떤 기능들이 있는지 다음의 링크들을 참조하여 OpenCV 에 대해서 살펴보고 추후 프로젝트 진행시 활용해 보시기 바랍니다.

<https://opencv.org/> - OpenCV 공식 사이트

[https://docs.opencv.org/master/d6/d00/tutorial\\_py\\_root.html](https://docs.opencv.org/master/d6/d00/tutorial_py_root.html) - OpenCV 파이썬 튜토리얼

Colab에서는 OpenCV로 가공된 그래픽을 matplotlib를 사용해 이미지 버퍼 출력 부분을 바꿔 사용하는 방식으로 사용됩니다. 다음처럼 생성된 buffer에 cv2를 사용해 필요한 이미지 처리를 하신 후 plt.imshow() 함수와 plt.show()를 통해 간단히 화면에 출력 할 수 있습니다.



matplotlib는 위와 같이 좌표가 나오기 때문에 처리결과를 표시하기 좋습니다.

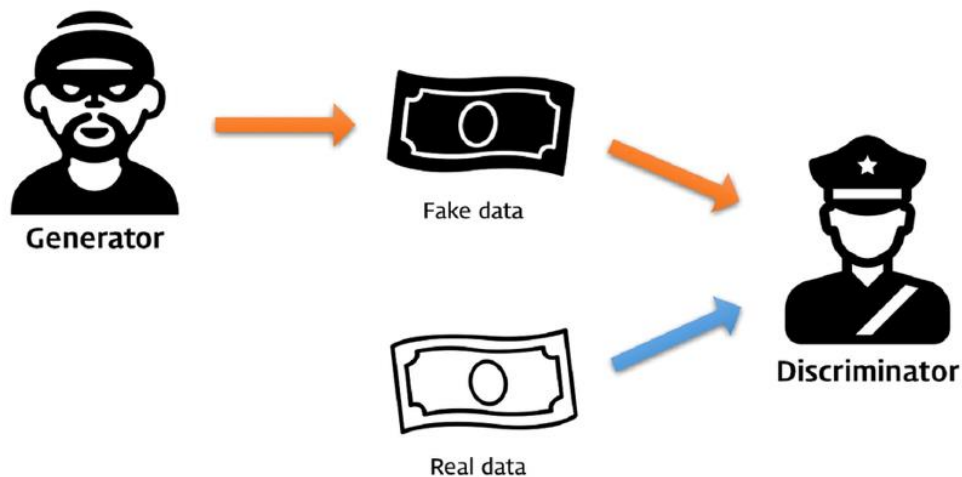
Req-6	basic.ipynb 파일에 matplotlib와 OpenCV의 Drawing API를 이용하여 화면에 여러 가지 도형을 그려주는 코드를 작성해 보세요. (난이도: 쉬움)
-------	---

### 3. 심화과제

#### 3-1. Generative Adversarial Network 의 기본 개념

2014 년 Ian J. Goodfellow 는 2 개의 모델이 서로 적대적인 과정을 통해 동시에 훈련하는 생성적 적대 신경망 (Generative Adversarial Network, GAN)을 발표 합니다.

간단하게 설명을 하자면 위조지폐를 계속해서 만드는 위조지폐범과 주어진 지폐가 위조지폐인지 아닌지를 판별하는 수사관이 있다고 가정할 때 다음과 같은 형태로 학습을 진행 합니다.



[Fig.2] GAN 의 기본 원리

[출처 : [https://files.slack.com/files-pri/T25783BPY-F9SHTP6F9/picture2.png?pub\\_secret=6821873e68](https://files.slack.com/files-pri/T25783BPY-F9SHTP6F9/picture2.png?pub_secret=6821873e68)]

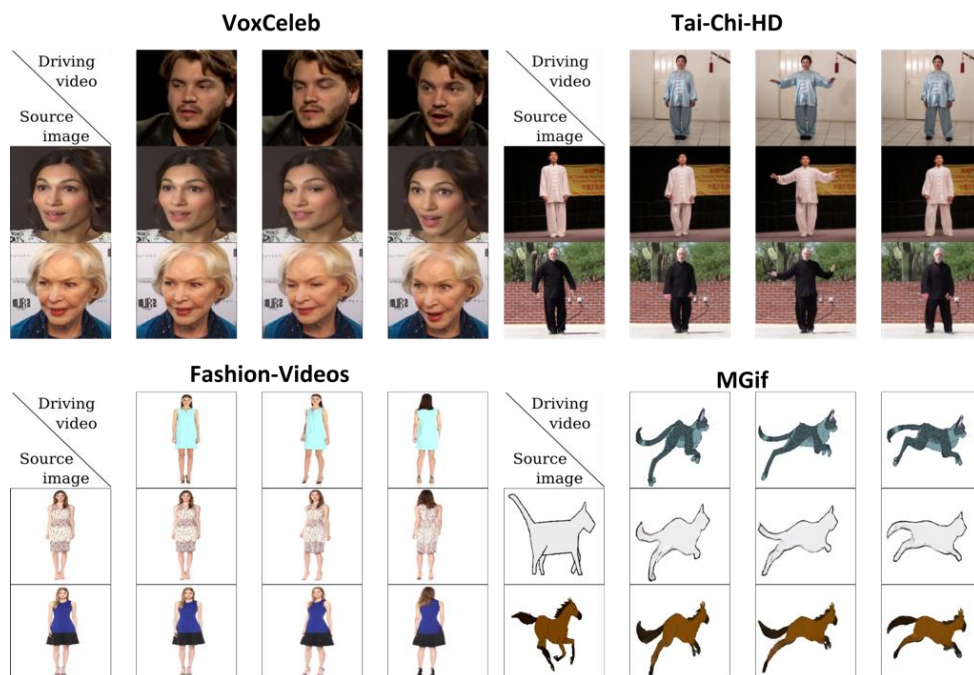
- (1) 위조지폐범은 본인이 만든 위조지폐를 수사관에게 제공해서 피드백을 받으면서 학습을 하여 위조지폐를 보다 정교하게 만듭니다.
- (2)수사관은 위조지폐와 지폐를 가지고 위조지폐 여부를 판단하고 피드백을 받아 학습과정을 거쳐 위조지폐를 판단하는 능력을 보다 정교하게 만듭니다.

여기까지 1 번 수행을 하면 이를 1 epoch 라고 합니다. 이를 계속 반복해 실행하게 되면 각각의 역할을 가진 2 개의 모델은 계속해서 정교해지게 되고 Generator 가 생성한 위조지폐를 Discriminator 가 진짜인지 아닌지 구분하지 못하고 정답 확률이 50%가 되는 순간 학습이 끝나게 됩니다.

## 3-2. First Order Motion Model for Image Animation

GAN 에 대한 기본적인 개념이 이해 되었다면 이번에는 GAN 을 사용해 구현된 기술들에 대해서 알아보겠습니다.

2019년 The University of British Columbia의 Computer Vision Lab 소속의 Aliaksandr Siarohi 외 3 인은 신경정보처리시스템학회(NeurIPS)에 First Order Motion Model for Image Animation 이란 이름의 논문을 발표합니다.



[Fig.3] First Order Motion Model for Image Animation

해당 논문에 포함된 First Order Model 은 GAN 을 활용해 만든 VoxCeleb, Tai-Chi-HD, Fashion-Videos, MGif 라는 4 개의 Dataset 을 포함하고 있습니다. 이번 프로젝트에서는 VoxCeleb 과 Fashion-Videos Dataset 을 가지고 실습해 보겠습니다.

### 3-2-1. VoxCeleb Dataset

VoxCeleb Dataset 은 동영상의 인물을 다른 인물로 바꿔주는 Dataset 으로 단 1 장의






사진으로 대체가 가능합니다.

Req-7	다음의 과정을 따라 advanced.ipynb 에 VoxCeleb Dataset 을 사용한 코드를 구현해 보세요..
-------	--

### 사전준비

1. Google Drive 에 colab 이란 폴더를 만들고 그안에 first\_motion\_order\_model 폴더를 만듭니다.
2. 해당폴더에 advanced.ipynb 파일을 복사해 넣습니다.
3. [https://drive.google.com/drive/folders/1kZ1gCnpfU0BnpdU47pLM\\_TQ6RypDDqgw](https://drive.google.com/drive/folders/1kZ1gCnpfU0BnpdU47pLM_TQ6RypDDqgw) 에서 동영상 파일 (예, 04.mp4) 하나를 다운로드 받아 구글 드라이브 first\_motion\_order\_model 폴더에 복사해 넣습니다.
4. [https://drive.google.com/drive/folders/1PyQJmkdCsAk0YwUyaj\\_1-10as-iLDgeH](https://drive.google.com/drive/folders/1PyQJmkdCsAk0YwUyaj_1-10as-iLDgeH) 에서 다음의 Dataset 파일을 다운로드 받아 구글 드라이브 first\_motion\_order\_model 폴더에 복사해 넣습니다.
  - o vox-cpk.pth.tar
5. 휴대폰으로 셀카를 찍어 my\_face.png 라는 파일명으로 구글 드라이브 first\_motion\_order\_model 폴더에 올려 넣습니다.

다운로드가 완료되면 다음과 같은 4 개의 파일이 구글드라이브에 위치 하게 됩니다.

내 드라이브 > colab > first_order_motion_model ▾			
<div>  휴지통이 달라졌습니다. 휴지통에 있는 항목이 30일 후 자동으로 완전히 삭제됩니다. <a href="#">자세히 알아보기</a> </div>			
이름 ↑	소유자	마지막으로 수정...	파일 크기
 04.mp4	나	2020. 12. 4. 나	479KB
 advanced.ipynb ★	나	오후 2:05 나	357KB
 my_face.jpg	나	오후 2:08 나	5MB
 vox-cpk.pth.tar	나	2020. 12. 4. 나	695MB

advanced.ipynb 파일을 더블클릭하여 colab 으로 파일을 열면 다음과 같은 화면이 나옵니다.

**심화과제**

**Req.7. 다음의 과정을 따라 advanced.ipynb에 VoxCeleb Dataset을 사용해 결과를 출력 해보세요.**

- [https://drive.google.com/drive/folders/1kZ1gCnpfU0BnpdU47pLM\\_TQ6RypDDqgw](https://drive.google.com/drive/folders/1kZ1gCnpfU0BnpdU47pLM_TQ6RypDDqgw) 에서 동영상 파일 (예, 04.mp4) 하나를 다운로드 받아 구글 드라이브 first\_motion\_order\_model 폴더에 복사해 놓습니다.
- [https://drive.google.com/drive/folders/1PyQJmKdCsAkOYwUyaj\\_H0as-iLDgeH](https://drive.google.com/drive/folders/1PyQJmKdCsAkOYwUyaj_H0as-iLDgeH) 에서 다음의 Dataset 파일을 다운로드 받아 구글 드라이브 first\_motion\_order\_model 폴더에 복사해 놓습니다.
  - vox-cpk.pth.tar
- 휴대폰으로 셀카를 찍어 my\_face.png 라는 파일명으로 구글 드라이브 first\_motion\_order\_model 폴더에 올려 놓습니다. (정사각형 형태로 촬영을 추천 합니다.)

```
[ ] 1 # 구글 드라이브를 /gdrive에 마운트 합니다.
    2 from google.colab import drive
    3 #drive.flush_and_unmount() # 파일목록 갱신이 안될때는 이 부분 주석처리를 해제하고 실행하시기 바랍니다.
    4 drive.mount('/gdrive')
```

Drive already mounted at /gdrive; to attempt to forcibly remount, call drive.mount("/gdrive", force\_remount=True).

```
[ ] 1 # 마운트된 드라이브의 first_order_motion_model 디렉토리의 파일목록을 불러옵니다.
    2 !ls -l /gdrive/MyDrive/colab/first_order_motion_model
```

```
total 717798
-rw-r--r-- 1 root root 490875 Dec 4 01:00 04.mp4
-rw-r--r-- 1 root root 237108 Dec 8 05:53 advanced.ipynb
-rw-r--r-- 1 root root 5529241 Dec 8 05:08 my_face.jpg
-rw-r--r-- 1 root root 728766691 Dec 4 02:12 vox-cpk.pth.tar
```

우선, 런타임유형을 다음과 같이 Python3 / GPU 로 변경합니다.

**노트 설정**

런타임 유형  
 Python 3

하드웨어 가속기  
 GPU

☐ 이 노트를 저장할 때 코드 셀 출력 생략

취소 저장

```
[1] 1 from google.colab import drive
    2 drive.mount('/gdrive')
```

Go to this URL in a browser: <https://accounts.google.com/o/oauth2/auth?scope=drive>

Enter your authorization code:

Mounted at /gdrive

위와 같이 Colab 코드스니펫의 Google Drive 연동코드를 추가한 후 링크를 클릭하여 Authorization code 를 복사해서 붙여 넣어주면 /gdrive 에 구글드라이브를 마운트 해 줍니다.

```
1 ls -l /gdrive/MyDrive/colab/first_order_motion_model
```

```
total 717923
-rw-rw-r-- 1 root root 490875 Dec 4 01:00 04.mp4
-rw-rw-r-- 1 root root 364610 Dec 8 05:32 advanced.ipynb
-rw-rw-r-- 1 root root 5529241 Dec 8 05:08 my_face.jpg
-rw-rw-r-- 1 root root 728766691 Dec 4 02:12 vox-cpk.pth.tar
```

위와 같이 `ls -l /gdrive/MyDrive/colab/first_order_motion_model` 로 4 개의 파일이 출력 되는 것을 확인합니다.

```
[ ] 1 # Aliaksandr Siarohin 의 github에서 First Order Model을 clone 합니다.
    2 !git clone https://github.com/AliaksandrSiarohin/first-order-model
```

```
Cloning into 'first-order-model'...
remote: Enumerating objects: 249, done.
remote: Total 249 (delta 0), reused 0 (delta 0), pack-reused 249
Receiving objects: 100% (249/249), 72.12 MiB | 33.58 MiB/s, done.
Resolving deltas: 100% (123/123), done.
```

이후 Aliakandr Siarohin 의 github 에서 상기와 같이 소스코드 및 데이터 파일을 clone 합니다.

```
[ ] 1 cd /content/first-order-model
```

```
/content/first-order-model
```

이후 clone 받은 소스코드 디렉토리로 cd 를 수행 한 후

```
[ ] 1 # 현재 디렉토리의 내용을 보여줍니다.
    2 !ls -l
```

```
total 1252
-rw-r--r-- 1 root root 4061 Dec 8 07:50 animate.py
-rw-r--r-- 1 root root 12547 Dec 8 07:50 augmentation.py
drwxr-xr-x 2 root root 4096 Dec 8 07:50 config
-rw-r--r-- 1 root root 5876 Dec 8 07:50 crop-video.py
drwxr-xr-x 3 root root 4096 Dec 8 07:50 data
-rw-r--r-- 1 root root 1143852 Dec 8 07:50 demo.ipynb
-rw-r--r-- 1 root root 6730 Dec 8 07:50 demo.py
-rw-r--r-- 1 root root 502 Dec 8 07:50 Dockerfile
-rw-r--r-- 1 root root 7027 Dec 8 07:50 frames_dataset.py
-rw-r--r-- 1 root root 18868 Dec 8 07:50 LICENSE.md
-rw-r--r-- 1 root root 8282 Dec 8 07:50 logger.py
drwxr-xr-x 2 root root 4096 Dec 8 07:50 modules
-rw-r--r-- 1 root root 8619 Dec 8 07:50 README.md
-rw-r--r-- 1 root root 2772 Dec 8 07:50 reconstruction.py
-rw-r--r-- 1 root root 420 Dec 8 07:50 requirements.txt
-rw-r--r-- 1 root root 3299 Dec 8 07:50 run.py
drwxr-xr-x 2 root root 4096 Dec 8 07:50 sup-mat
drwxr-xr-x 2 root root 4096 Dec 8 07:50 sync_batchnorm
-rw-r--r-- 1 root root 4479 Dec 8 07:50 train.py
```

위와 같이 디렉토리의 내용을 확인해 보고 다음처럼 필요한 패키지들을 import 합니다.

```
[ ] 1 # 필요한 패키지들을 import 합니다.
    2 import imageio
    3 import numpy as np
    4 import matplotlib.pyplot as plt
    5 import matplotlib.animation as animation
    6 from skimage.transform import resize
    7 from IPython.display import HTML
    8 import warnings
    9 warnings.filterwarnings("ignore")
    10
    11 from demo import load_checkpoints
    12 from demo import make_animation
    13 from skimage import img_as_ubyte
```

우리가 이번 과제에서 수행하려는 것은 내 사진(my\_face.jpg)를 셀럽의 동영상(04.mp4)과 합성시켜 셀럽 동영상의 얼굴을 내 사진으로 대체 하고자 합니다.

다음과 같이 내 사진과 셀럽의 비디오 파일을 읽어들이어서 256 x 256 사이즈로 축소를 하는 코드를 작성합니다.



```
[ ] 1 # 소스 이미지
2 source_image = imageio.imread('/gdrive/MyDrive/colab/first_order_motion_model/my_face.jpg')
3
4 # 소스 비디오
5 reader = imageio.get_reader('/gdrive/MyDrive/colab/first_order_motion_model/04.mp4')
6
7 # 이미지와 비디오크기를 256x256 픽셀로 변경 합니다
8 source_image = resize(source_image, (256, 256))[..., :3]
9
10 # frames per second 1초당 표시할 프레임수를 가져옵니다
11 fps = reader.get_meta_data()['fps']
```

다음과 같이 driving\_video 를 초기화 하고 소스비디오의 각 프레임을 driving\_video 에 추가해 준 후 256x256 으로 리사이즈를 해주는 코드를 작성 합니다.

```
[ ] 1 #driving_video를 초기화 하고 소스비디오의 각 프레임을 driving_video에 추가 합니다.
2 driving_video = []
3 try:
4     for im in reader:
5         driving_video.append(im)
6 except RuntimeError:
7     pass
8 reader.close()
```

```
[ ] 1 # driving_video를 256x256 으로 리사이즈 합니다.
2 driving_video = [resize(frame, (256, 256))[..., :3] for frame in driving_video]
```

다음과 같이 소스이미지와 기존비디오를 합성해주는 display() 함수를 구현해 준 후 display() 함수를 사용하여 소스이미지와 기본비디오를 그대로 HTML5 비디오 형태로 출력합니다.

```

1 # 소스이미지와 기존 비디오를 합성해주는 display 함수
2 def display(source, driving, generated=None):
3     fig = plt.figure(figsize=(8 + 4 * (generated is not None), 6))
4     ims = []
5     for i in range(len(driving)):
6         cols = [source]
7         cols.append(driving[i])
8         if generated is not None:
9             cols.append(generated[i])
10        im = plt.imshow(np.concatenate(cols, axis=1), animated=True)
11        plt.axis('off')
12        ims.append([im])
13
14    ani = animation.ArtistAnimation(fig, ims, interval=50, repeat_delay=1000)
15    plt.close()
16    return ani

```

실행을 해보면 다음과 같이 내 사진과 동영상이 동시에 출력됩니다.

```

1 # display() 함수를 사용하여 HTML5 비디오 형태로 출력합니다.
2 HTML(display(source_image, driving_video).to_html5_video())

```



이제 학습된 VoxCeleb Dataset 의 Check Points 를 불러오는 load\_checkpoints()

함수를 다음과 같이 실행해 보겠습니다.

```
[ ] 1 # VoxCeleb 모델의 체크포인트 불러오기 (700MB 파일이라 오래 걸립니다.)
2 generator, kp_detector = load_checkpoints(config_path='config/vox-256.yaml',
3 checkpoint_path='/gdrive/MyDrive/colab/first_order_motion_model/vox-cpk.pth.tar')
```

make\_animation() 함수를 사용하여 소스이미지, 소스비디오를 입력으로 predictions 을 생성하고 파일로 저장 후 화면에 출력하는 코드를 다음과 같이 구현하고 실행해 봅니다.

```
[ ] 1 # make_animation() 함수를 사용하여 소스이미지, 소스비디오를 입력으로 predictions를 생성합니다.
2 predictions = make_animation(source_image, driving_video, generator, kp_detector, relative=True)
3
4 # generated.mp4 파일로 생성된 결과를 저장
5 imageio.mimsave('./generated.mp4', [img_as_ubyte(frame) for frame in predictions], fps=fps)
6
7 # display 함수를 사용해 로 출력합니다.
8 HTML(display(source_image, driving_video, predictions).to_html5_video())
```



코드를 실행해 보면 상기와 같이 내 사진과 셀럽의 동영상의 동영상이 같이 출력되고 마지막에 내 사진이 동영상에 입혀져서 출력되는 모습을 확인 할 수 있습니다.

### 3-2-2. Fashion-Videos Dataset



[Fig.4] Fashion-Video Dataset 의 학습용 동영상

Fashion-Videos Dataset 은 하나의 촬영된 소스 동영상을 바탕으로 사진 한 장으로 해당 사진의 동영상을 합성해 낼 수 있음을 보여주는 Dataset 입니다. 이를 응용하면 하나의 영상을 촬영 후 이후에는 사진촬영만으로 해당 옷을 입은 동영상을 만들거나 고객이 직접 해당 옷을 입었을 때의 착용 모습을 미리 볼 수 있게 해주는 등 마케팅적으로 여러가지 기능과 아이디어를 활용 할 수 있을 것 입니다.

Req-8	다음의 과정을 따라 advanced.ipynb 에 Fashion-Videos Dataset 을 사용한 코드를 구현해 보세요.
-------	---

#### 사전준비

1. Req-7에서 생성한 Google Drive 의 colab/first\_motion\_order\_model 폴더를 엽니다.
2. [https://drive.google.com/drive/folders/1PyQJmKdCsAkOYwUyaj\\_1-10as-iLDgeH](https://drive.google.com/drive/folders/1PyQJmKdCsAkOYwUyaj_1-10as-iLDgeH) 에서 다음의 Dataset 파일을 다운로드 받아 구글 드라이브 first\_motion\_order\_model 폴더에 복사해 놓습니다.
  - o fashion.pth.tar
3. [https://vision.cs.ubc.ca/datasets/fashion/resources/fashion\\_dataset/fashion\\_train.txt](https://vision.cs.ubc.ca/datasets/fashion/resources/fashion_dataset/fashion_train.txt) 에서 4 개의 mp4 파일을 다운로드 받습니다.

4. 다운로드 받은 mp4 파일을 전체화면으로 재생시킨 후 바로 Pause 버튼을 눌러 모델이 똑바로 앞을 보고 있는 장면에서 멈춘 후 컴퓨터의 화면 캡처 프로그램을 사용해 동영상 화면을 캡처해 m1.png, m2.png, m3.png 등 3 개의 정지 화상을 얻은 후 구글드라이브에 복사해 놓습니다.
5. 나머지 1 개의 동영상 혹은 해당 동영상과 비슷한 구도와 포즈의 전신 샷을 촬영하여 req8\_source.mp4 라는 이름으로 구글드라이브에 복사해 놓습니다.
6. 다음과 같이 Req.8 을 수행하기 위해 필요한 상기 붉은색 박스의 5 개의 파일이 업로드 되었는지 확인 합니다.

내 드라이브 > colab > first\_order\_motion\_model ▾

❗ 휴지통이 달라졌습니다. 휴지통에 있는 항목이 30일 후 자동으로 완전히 삭제됩니다. 자세히 알아보기

이름	소유자	내가 마지막...	↓	파일 크기
 fashion.pth.tar	나	오전 11:18		716MB
 m1.png	나	오전 11:13		346KB
 m3.png	나	오전 11:13		321KB
 m2.png	나	오전 11:13		282KB
 req8_source.mp4	나	오전 11:13		734KB
 advanced.ipynb	나	오전 9:05		143KB
 my_face.jpg	나	2020. 12. 4.		5MB
 vox-cpk.pth.tar	나	2020. 12. 4.		695MB
 04.mp4	나	2020. 12. 4.		479KB

준비가 되었다면 Req.7 을 구현한 advanced.ipynb 파일의 Req.8 부분에 다음을 참고하여 구현 합니다.

```
[ ] 1 from demo import load_checkpoints
    2 from demo import make_animation
    3 from skimage import img_as_ubyte

[ ] 1 # 3개의 이미지와 1개의 동영상을 불러 옵니다.
    2 src_img1 = imageio.imread('/gdrive/MyDrive/colab/first_order_motion_model/m1.png')
    3 src_img2 = imageio.imread('/gdrive/MyDrive/colab/first_order_motion_model/m2.png')
    4 src_img3 = imageio.imread('/gdrive/MyDrive/colab/first_order_motion_model/m3.png')
    5 driving_video = imageio.mimread('/gdrive/MyDrive/colab/first_order_motion_model/req8_source.mp4', memtest=False)
```

상기와 같이 업로드한 3 개의 소스이미지를 읽어들이고 움직임의 토대가 되는 소스 영상인 req8\_source.mp4 를 driving\_video 변수에 읽어 들입니다.

다음과 같이 읽어들이는 소스의 해상도를 256x256 사이즈로 축소 합니다.

```
1 # 이미지와 비디오를 256x256 사이즈로 변경 합니다.
2 src_img1 = resize(src_img1, (256, 256))[..., :3]
3 src_img2 = resize(src_img2, (256, 256))[..., :3]
4 src_img3 = resize(src_img3, (256, 256))[..., :3]
5 driving_video = [resize(frame, (256, 256))[..., :3] for frame in driving_video]
```

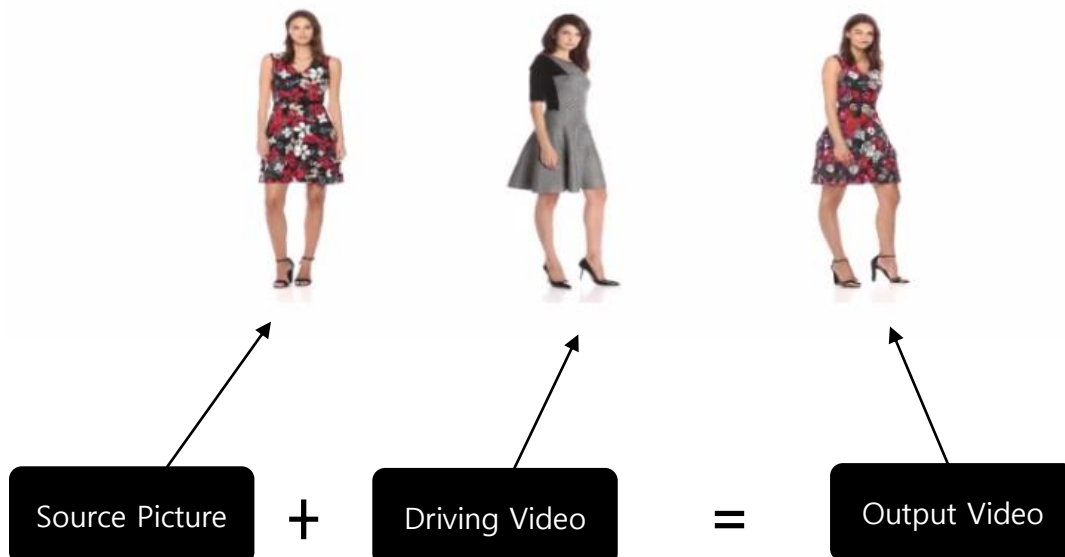
이전에 학습했던 체크포인트를 저장해둔 fashion.pth.tar 을 불러 옵니다.

```
1 # load_checkpoints 함수를 사용해 이전에 학습해둔 fashion.pth.tar을 을 불러 옵니다.
2 generator, kp_detector = load_checkpoints(config_path='config/fashion-256.yaml',
3                                         checkpoint_path='/gdrive/MyDrive/colab/first_order_motion_model/fashion.pth.tar')
4
```

다음처럼 첫번째 모델의 사진과 Driving Video 로 첫번째 모델의 움직이는 비디오를 만들고 재생 합니다.

```
1 # 첫번째 모델의 사진과 Driving Video로 첫번째 모델의 움직이는 비디오를 만들고 재생합니다.
2 predictions1 = make_animation(src_img1, driving_video, generator, kp_detector, relative=True)
3 HTML(display(src_img1, driving_video, predictions1).to_html5_video())
```

100%|■■■■■■■■■■| 394/394 [00:08<00:00, 47.62it/s]



두번째 모델의 사진과 Driving Video 로 두번째 모델의 움직이는 비디오를 만들고 재생합니다.

```
[23] 1 # 두번째 모델의 사진과 Driving Video로 두번째 모델의 움직이는 비디오를 만들고 재생합니다.
      2 predictions2 = make_animation(src_img2, driving_video, generator, kp_detector, relative=True)
      3 HTML(display(src_img2, driving_video, predictions2).to_html5_video())
```

100%|██████████| 394/394 [00:08<00:00, 47.73it/s]



세번째 모델의 사진과 Driving Video 로 세번째 모델의 움직이는 비디오를 만들고 재생합니다.

```
1 # 세번째 모델의 사진과 Driving Video로 세번째 모델의 움직이는 비디오를 만들고 재생합니다.
2 predictions3 = make_animation(src_img3, driving_video, generator, kp_detector, relative=True)
3 HTML(display(src_img3, driving_video, predictions3).to_html5_video())
```

100%|██████████| 394/394 [00:08<00:00, 44.92it/s]



이상으로 간단하게 GAN 을 응용한 영상 합성을 실습해 보았습니다. GAN 을 응용해서 비디오를 합성하는 기술은 대중들에게는 Deepfake 라는 부정적인 명칭으로 더 잘 알려져 있기도 합니다.



[Fig.4] MBN 의 김주하 AI 앵커 소개화면

하지만 국내 방송사 MBN 의 김주하 AI 앵커 개발에 사용되는 등 앞으로도 활용도가 아주 많은 기술이라고 할 수 있을 것 입니다.

오늘 수행한 과제를 바탕으로 Python 기반의 이미지 및 영상처리 방법이나 CNN, RNN, GAN 혹은 Machine Learning, Deep Learning 등에 흥미가 생겨 해당 분야의 학습을 시작하게 되는 계기가 되시길 바랍니다.



## 4. 과제 제출방법

---

다음 제출방법을 잘 읽고 가이드에 따라 과제를 제출 하시기 바랍니다.

- 1) Local Git Repo 의 다음 2 개의 새 노트파일들을 명세서를 참고하여 Google Drive 에 업로드 합니다.

- {PROJECT\_ROOT}/self-project/con10/rep01/**basic.ipynb**
- {PROJECT\_ROOT}/self-project/con10/rep01/**advanced.ipynb**

- 2) 업로드 된 basic.ipynb 을 Colab 으로 열어서 Req-1 ~ 6 를 구현 후 실행 및 저장 합니다.
- 3) 업로드 된 advanced.ipynb 을 Colab 으로 열어서 Req-7 ~ 8 을 구현 후 실행 및 저장 합니다.
- 4) 실행 결과가 저장된 basic.ipynb 와 advanced.ipynb 파일을 다운로드 받아 다음의 Local Git Repo 에 덮어쓰기를 합니다.

- {PROJECT\_ROOT}/self-project/con10/rep01/**basic.ipynb**
- {PROJECT\_ROOT}/self-project/con10/rep01/**advanced.ipynb**

- 5) commit / push 하여 제출 합니다.

- 6) 제출경로

<https://lab.ssafy.com/s10-study/self-project/>의 “산출물 제출 가이드” .docx  
참조