그리기

양승우

이번 장에서 역시 발생할 수 있는 문제를 먼저 해결합니다. (다음 슬라이드 참고)

파일 중복 문제로,

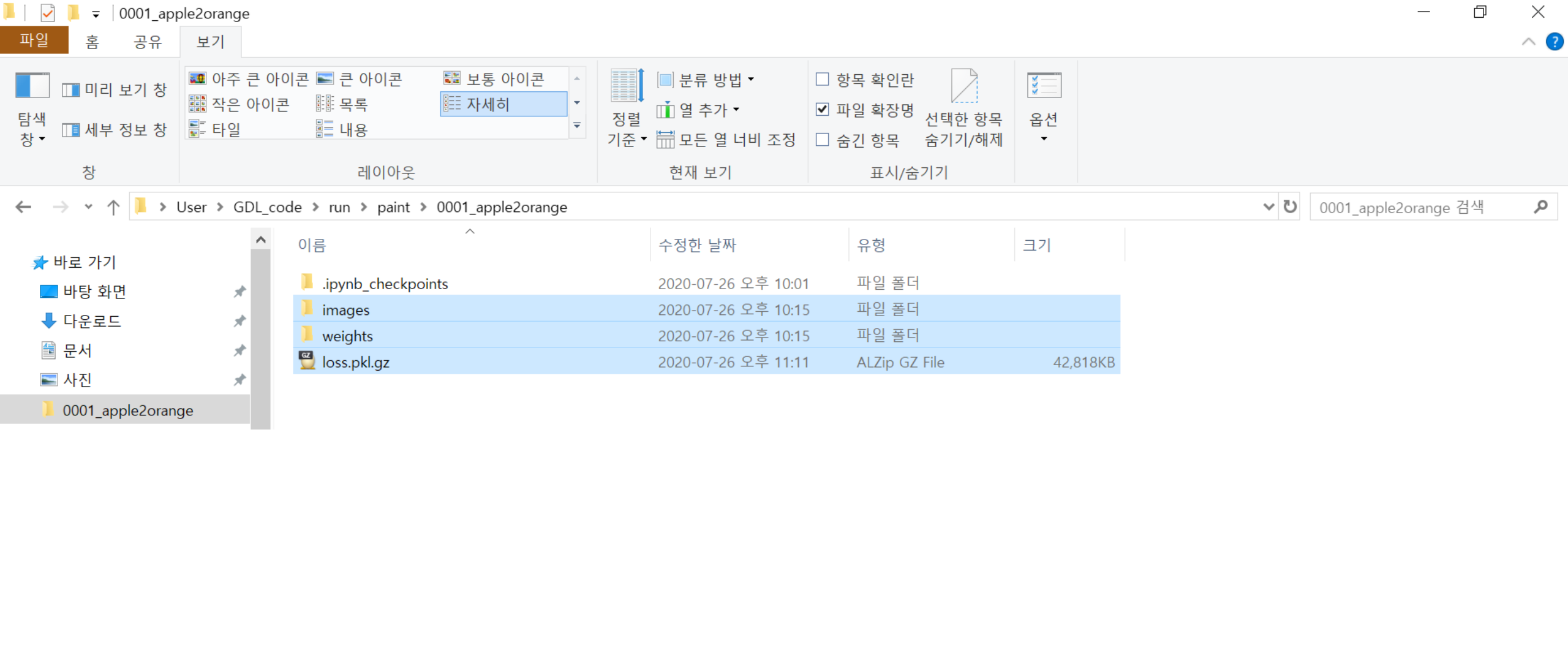
기존의 0001_apple2orange 파일에 있는 폴더들을 모두 복사 후에

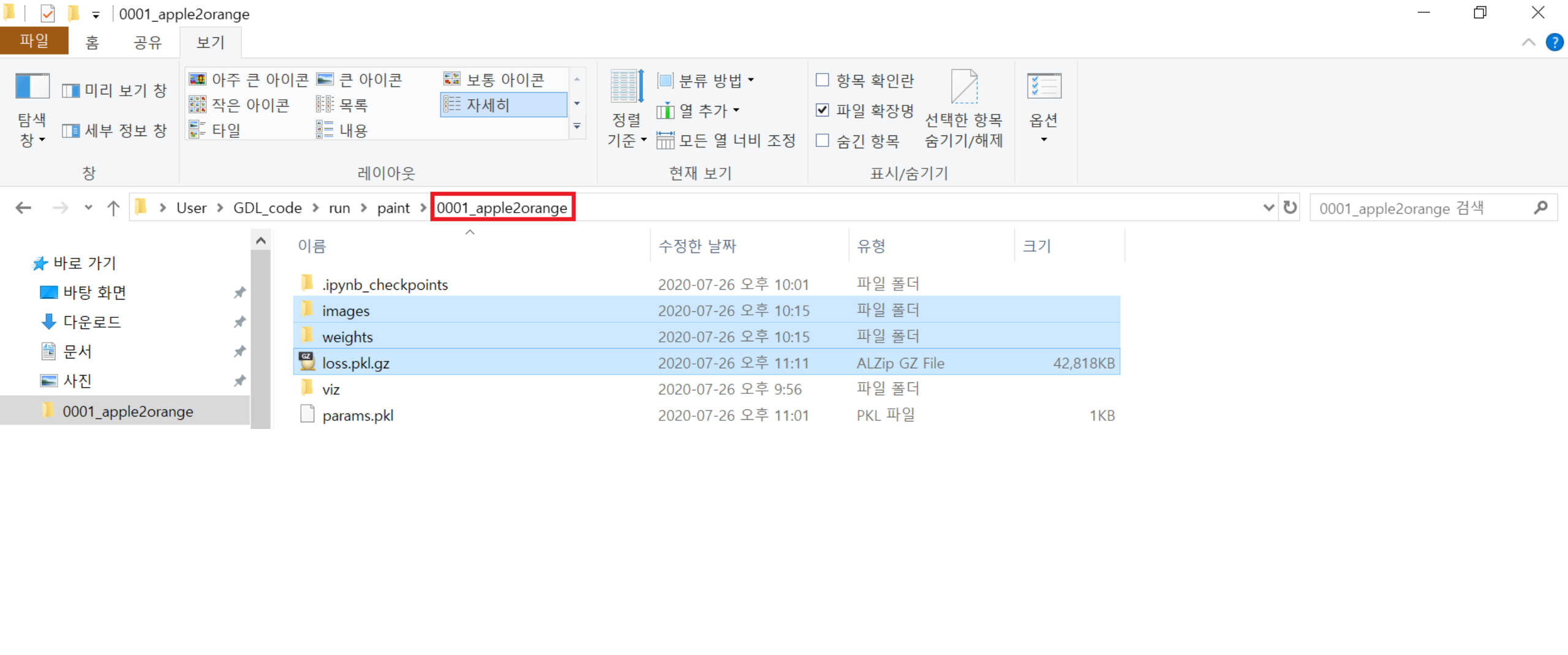
그 후 3번 라인을 실행 후 다시 만들어진 0001_apple2orange 파일에서 viz 폴더와

params.pkl 파일을 제외하고 나머지 파일을 삭제합니다.(중복 방지)

그리고 방금 삭제 했다 생긴 0001_apple2orange 파일에 다시 붙여 넣습니다.

옮기는 것이 불편하거나 오류가 난다면 윈도우에서 직접 하는 것이 편리합니다.


0001_apple2orange
파일
홈
공유
보기
미리 보기 창
세부 정보 창
탐색 창
아주 큰 아이콘
큰 아이콘
보통 아이콘
작은 아이콘
목록
자세히
타일
내용
창
레이아웃
정렬 기준
분류 방법
열 추가
모든 열 너비 조정
현재 보기
항목 확인란
파일 확장명
숨긴 항목
선택한 항목 숨기기/해제
표시/숨기기
옵션
User
GDL_code
run
paint
0001_apple2orange
0001_apple2orange 검색
바로 가기
바탕 화면
다운로드
문서
사진
0001_apple2orange
이름
수정한 날짜
유형
크기
.ipynb_checkpoints
2020-07-26 오후 10:01
파일 폴더
images
2020-07-26 오후 10:15
파일 폴더
weights
2020-07-26 오후 10:15
파일 폴더
loss.pkl.gz
2020-07-26 오후 11:11
ALZip GZ File
42,818KB


0001_apple2orange
파일
홈
공유
보기
미리 보기 창
세부 정보 창
탐색 창
아주 큰 아이콘
큰 아이콘
보통 아이콘
작은 아이콘
목록
자세히
타일
내용
창
레이아웃
정렬 기준
분류 방법
열 추가
모든 열 너비 조정
현재 보기
항목 확인란
파일 확장명
숨긴 항목
선택한 항목 숨기기/해제
표시/숨기기
옵션
User
GDL_code
run
paint
0001_apple2orange
0001_apple2orange 검색
바로 가기
바탕 화면
다운로드
문서
사진
0001_apple2orange
이름
수정한 날짜
유형
크기
.ipynb_checkpoints
2020-07-26 오후 10:01
파일 폴더
images
2020-07-26 오후 10:15
파일 폴더
weights
2020-07-26 오후 10:15
파일 폴더
loss.pkl.gz
2020-07-26 오후 11:11
ALZip GZ File
42,818KB
viz
2020-07-26 오후 9:56
파일 폴더
params.pkl
2020-07-26 오후 11:01
PKL 파일
1KB

준비 완료

CycleGAN 훈련

라이브러리 임포트

Note: 이 노트북의 코드를 실행하려면 `keras_contrib` 패키지를 설치해야 합니다. 다음 셀의 주석을 제거하고 실행하여 패키지를 설치하세요

In [ ]:
```
#!pip install git+https://www.github.com/keras-team/keras-contrib.git
```

In [ ]:
```
import os
import matplotlib.pyplot as plt

from models.cycleGAN import CycleGAN
from utils.loaders import DataLoader
```

In [ ]:
```
# run params
SECTION = 'paint'
RUN_ID = '0001'
DATA_NAME = 'apple2orange'
RUN_FOLDER = 'run/{}/'.format(SECTION)
RUN_FOLDER += '_'.join([RUN_ID, DATA_NAME])

if not os.path.exists(RUN_FOLDER):
    os.mkdir(RUN_FOLDER)
    os.mkdir(os.path.join(RUN_FOLDER, 'viz'))
    os.mkdir(os.path.join(RUN_FOLDER, 'images'))
    os.mkdir(os.path.join(RUN_FOLDER, 'weights'))

mode =  'build' # 'build' #
```

CycleGAN 훈련

Note: 라이브러리 버전 때문에 책의 내용과 결과가 다를 수 있습니다

라이브러리 임포트

Note: 이 노트북의 코드를 실행하려면 keras_contrib 패키지를 설치해야 합니다. 다음 셀의 주석을 제거하고 실행하여 패키지를 설치하세요

In [1]:
```
!pip install git+https://www.github.com/keras-team/keras-contrib.git
```

```
Collecting git+https://www.github.com/keras-team/keras-contrib.git
  Cloning https://www.github.com/keras-team/keras-contrib.git to c:₩users₩user₩appdata₩local₩temp₩pip-req-build-qcgpam8k
Requirement already satisfied: keras in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras-contrib==2.0.8) (2.2.4)
Requirement already satisfied: pyyaml in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contrib==2.0.8) (5.3.1)
Requirement already satisfied: h5py in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contrib==2.0.8) (2.10.0)
Requirement already satisfied: keras-applications>=1.0.6 in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contr
ib==2.0.8) (1.0.8)
Requirement already satisfied: six>=1.9.0 in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contrib==2.0.8) (1.1
5.0)
Requirement already satisfied: keras-preprocessing>=1.0.5 in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-cont
rib==2.0.8) (1.1.2)
Requirement already satisfied: scipy>=0.14 in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contrib==2.0.8) (1.
5.0)
Requirement already satisfied: numpy>=1.9.1 in c:₩users₩user₩anaconda3₩envs₩testgan₩lib₩site-packages (from keras->keras-contrib==2.0.8)
(1.18.5)
Building wheels for collected packages: keras-contrib
  Building wheel for keras-contrib (setup.py): started
  Building wheel for keras-contrib (setup.py): finished with status 'done'
  Created wheel for keras-contrib: filename=keras_contrib-2.0.8-py3-none-any.whl size=101658 sha256=58ea3d414f169bcec5189dd476ce91afc2af4a2
084048d6f67027e00b57b71aa
  Stored in directory: C:₩Users₩User₩AppData₩Local₩Temp₩pip-ephem-wheel-cache-boy_nltm₩wheels₩16₩87₩6e₩8e3b73f23fb38163af1c319aa23f14602018
b501ecb91430a2
Successfully built keras-contrib
Installing collected packages: keras-contrib
Successfully installed keras-contrib-2.0.8
```

```
  Running command git clone -q https://www.github.com/keras-team/keras-contrib.git 'C:₩Users₩User₩AppData₩Local₩Temp₩pip-req-build-qcgpam8
k'
```

In [2]:
```
import os
import matplotlib.pyplot as plt

from models.cycleGAN import CycleGAN
from utils.loaders  import DataLoader
```

```
Using TensorFlow backend.
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:526: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:528: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:529: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:530: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩dtypes.py:535: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [3]:
```
# run params
SECTION = 'paint'
RUN_ID = '0001'
DATA_NAME = 'apple2orange'
RUN_FOLDER = 'run/{}/'.format(SECTION)
RUN_FOLDER += '_'.join([RUN_ID, DATA_NAME])

if not os.path.exists(RUN_FOLDER):
    os.mkdir(RUN_FOLDER)
    os.mkdir(os.path.join(RUN_FOLDER, 'viz'))
    os.mkdir(os.path.join(RUN_FOLDER, 'images'))
    os.mkdir(os.path.join(RUN_FOLDER, 'weights'))

mode =  'build' # 'build' #
```

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 사과, 오렌지 데이터셋을 다운로드하세요.

In [ ]:
```
#!./scripts/download_cyclegan_data.sh apple2orange
```

In [ ]:
```
IMAGE_SIZE = 128
```

In [ ]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 사과, 오렌지 데이터셋을 다운로드하세요.

In [4]:
```
! /scripts/download_cyclegan_data.sh apple2orange
```

```
'.'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
배치 파일이 아닙니다.
```

In [5]:
```
IMAGE_SIZE = 128
```

In [6]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

다음과 같은 오류는 경로 문제로

다음과 같은 절대경로로 바꾸어 줍니다.

! C:/Users/User/GDL_code/scripts/download_cyclegan_data.sh apple2orange

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 사과, 오렌지 데이터셋을 다운로드하세요.

In [4]:
```
! C:/Users/User/GDL_code/scripts/download_cyclegan_data.sh apple2orange
```

In [5]:
```
IMAGE_SIZE = 128
```

In [6]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

모델 생성

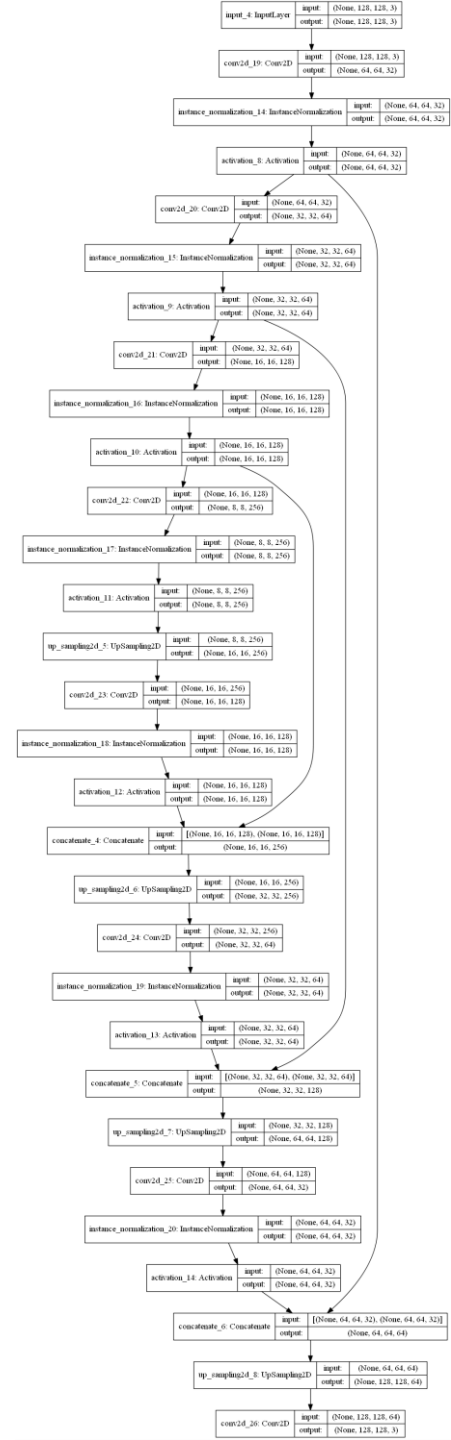
In [7]:
```
gan = CycleGAN(
    input_dim = (IMAGE_SIZE,IMAGE_SIZE,3)
    ,learning_rate = 0.0002
    , buffer_max_length = 50
    , lambda_validation = 1
    , lambda_reconstr = 10
    , lambda_id = 2
    , generator_type = 'unet'
    , gen_n_filters = 32
    , disc_n_filters = 32
    )

if mode == 'build':
    gan.save(RUN_FOLDER)
else:
    gan.load_weights(os.path.join(RUN_FOLDER, 'weights/weights.h5'))
```

```
WARNING:tensorflow:From C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩op_def_library.py:263: colocate_
with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

In [8]:
```
gan_g_BA.summary()
```

Layer (type)	Output Shape	Param #	Connected to
input_4 (InputLayer)	(None, 128, 128, 3)	0	
conv2d_19 (Conv2D)	(None, 64, 64, 32)	1568	input_4[0][0]
instance_normalization_14 (Inst	(None, 64, 64, 32)	0	conv2d_19[0][0]
activation_8 (Activation)	(None, 64, 64, 32)	0	instance_normalization_14[0][0]
conv2d_20 (Conv2D)	(None, 32, 32, 64)	32832	activation_8[0][0]
instance_normalization_15 (Inst	(None, 32, 32, 64)	0	conv2d_20[0][0]
activation_9 (Activation)	(None, 32, 32, 64)	0	instance_normalization_15[0][0]
conv2d_21 (Conv2D)	(None, 16, 16, 128)	131200	activation_9[0][0]
instance_normalization_16 (Inst	(None, 16, 16, 128)	0	conv2d_21[0][0]
activation_10 (Activation)	(None, 16, 16, 128)	0	instance_normalization_16[0][0]
conv2d_22 (Conv2D)	(None, 8, 8, 256)	524544	activation_10[0][0]
instance_normalization_17 (Inst	(None, 8, 8, 256)	0	conv2d_22[0][0]
activation_11 (Activation)	(None, 8, 8, 256)	0	instance_normalization_17[0][0]
up_sampling2d_5 (UpSampling2D)	(None, 16, 16, 256)	0	activation_11[0][0]
conv2d_23 (Conv2D)	(None, 16, 16, 128)	524416	up_sampling2d_5[0][0]
instance_normalization_18 (Inst	(None, 16, 16, 128)	0	conv2d_23[0][0]
activation_12 (Activation)	(None, 16, 16, 128)	0	instance_normalization_18[0][0]
concatenate_4 (Concatenate)	(None, 16, 16, 256)	0	activation_12[0][0] activation_10[0][0]
up_sampling2d_6 (UpSampling2D)	(None, 32, 32, 256)	0	concatenate_4[0][0]
conv2d_24 (Conv2D)	(None, 32, 32, 64)	262208	up_sampling2d_6[0][0]
instance_normalization_19 (Inst	(None, 32, 32, 64)	0	conv2d_24[0][0]
activation_13 (Activation)	(None, 32, 32, 64)	0	instance_normalization_19[0][0]
concatenate_5 (Concatenate)	(None, 32, 32, 128)	0	activation_13[0][0] activation_9[0][0]
up_sampling2d_7 (UpSampling2D)	(None, 64, 64, 128)	0	concatenate_5[0][0]
conv2d_25 (Conv2D)	(None, 64, 64, 32)	65568	up_sampling2d_7[0][0]
instance_normalization_20 (Inst	(None, 64, 64, 32)	0	conv2d_25[0][0]
activation_14 (Activation)	(None, 64, 64, 32)	0	instance_normalization_20[0][0]
concatenate_6 (Concatenate)	(None, 64, 64, 64)	0	activation_14[0][0] activation_8[0][0]
up_sampling2d_8 (UpSampling2D)	(None, 128, 128, 64)	0	concatenate_6[0][0]
conv2d_26 (Conv2D)	(None, 128, 128, 3)	3075	up_sampling2d_8[0][0]

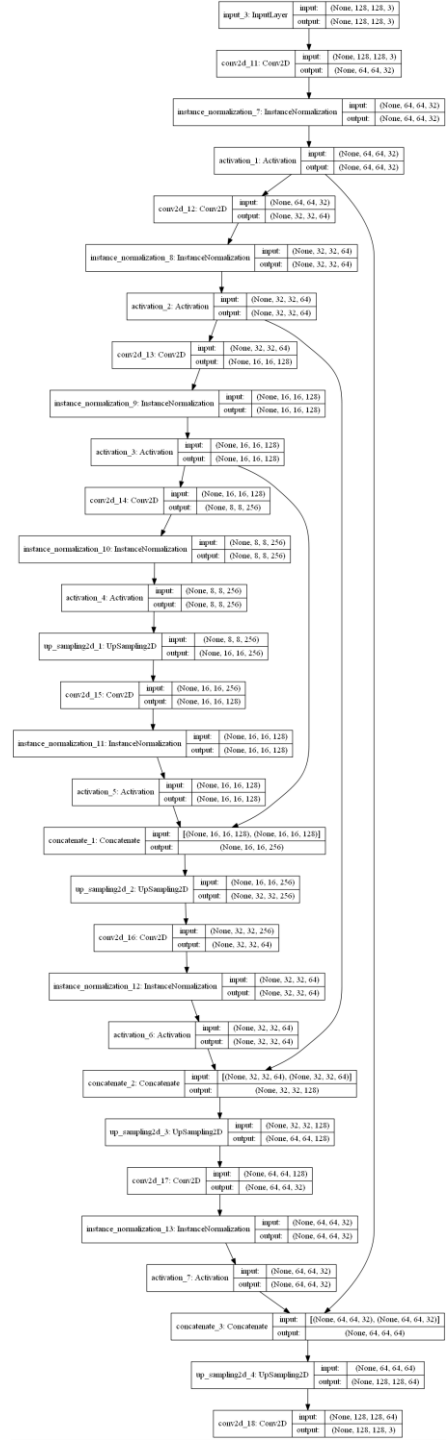
Total params: 1,545,411
Trainable params: 1,545,411
Non-trainable params: 0



In [9]:
```
gan,g_AB,summary()
```

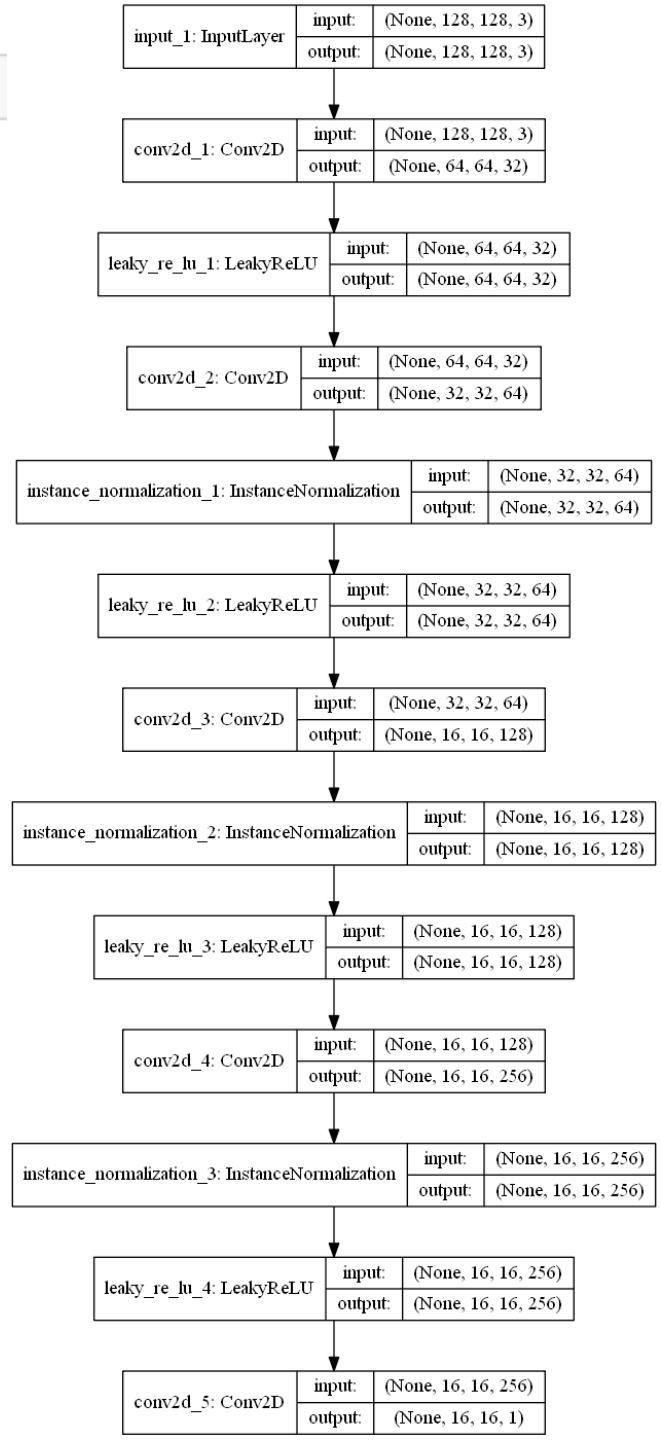
Layer (type)	Output Shape	Param #	Connected to
input_3 (InputLayer)	(None, 128, 128, 3)	0	
conv2d_11 (Conv2D)	(None, 64, 64, 32)	1568	input_3[0][0]
instance_normalization_7 (Insta	(None, 64, 64, 32)	0	conv2d_11[0][0]
activation_1 (Activation)	(None, 64, 64, 32)	0	instance_normalization_7[0][0]
conv2d_12 (Conv2D)	(None, 32, 32, 64)	32832	activation_1[0][0]
instance_normalization_8 (Insta	(None, 32, 32, 64)	0	conv2d_12[0][0]
activation_2 (Activation)	(None, 32, 32, 64)	0	instance_normalization_8[0][0]
conv2d_13 (Conv2D)	(None, 16, 16, 128)	131200	activation_2[0][0]
instance_normalization_9 (Insta	(None, 16, 16, 128)	0	conv2d_13[0][0]
activation_3 (Activation)	(None, 16, 16, 128)	0	instance_normalization_9[0][0]
conv2d_14 (Conv2D)	(None, 8, 8, 256)	524544	activation_3[0][0]
instance_normalization_10 (Inst	(None, 8, 8, 256)	0	conv2d_14[0][0]
activation_4 (Activation)	(None, 8, 8, 256)	0	instance_normalization_10[0][0]
up_sampling2d_1 (UpSampling2D)	(None, 16, 16, 256)	0	activation_4[0][0]
conv2d_15 (Conv2D)	(None, 16, 16, 128)	524416	up_sampling2d_1[0][0]
instance_normalization_11 (Inst	(None, 16, 16, 128)	0	conv2d_15[0][0]
activation_5 (Activation)	(None, 16, 16, 128)	0	instance_normalization_11[0][0]
concatenate_1 (Concatenate)	(None, 16, 16, 256)	0	activation_5[0][0] activation_3[0][0]
up_sampling2d_2 (UpSampling2D)	(None, 32, 32, 256)	0	concatenate_1[0][0]
conv2d_16 (Conv2D)	(None, 32, 32, 64)	262208	up_sampling2d_2[0][0]
instance_normalization_12 (Inst	(None, 32, 32, 64)	0	conv2d_16[0][0]
activation_6 (Activation)	(None, 32, 32, 64)	0	instance_normalization_12[0][0]
concatenate_2 (Concatenate)	(None, 32, 32, 128)	0	activation_6[0][0] activation_2[0][0]
up_sampling2d_3 (UpSampling2D)	(None, 64, 64, 128)	0	concatenate_2[0][0]
conv2d_17 (Conv2D)	(None, 64, 64, 32)	65568	up_sampling2d_3[0][0]
instance_normalization_13 (Inst	(None, 64, 64, 32)	0	conv2d_17[0][0]
activation_7 (Activation)	(None, 64, 64, 32)	0	instance_normalization_13[0][0]
concatenate_3 (Concatenate)	(None, 64, 64, 64)	0	activation_7[0][0] activation_1[0][0]
up_sampling2d_4 (UpSampling2D)	(None, 128, 128, 64)	0	concatenate_3[0][0]
conv2d_18 (Conv2D)	(None, 128, 128, 3)	3075	up_sampling2d_4[0][0]

Total params: 1,545,411
Trainable params: 1,545,411
Non-trainable params: 0



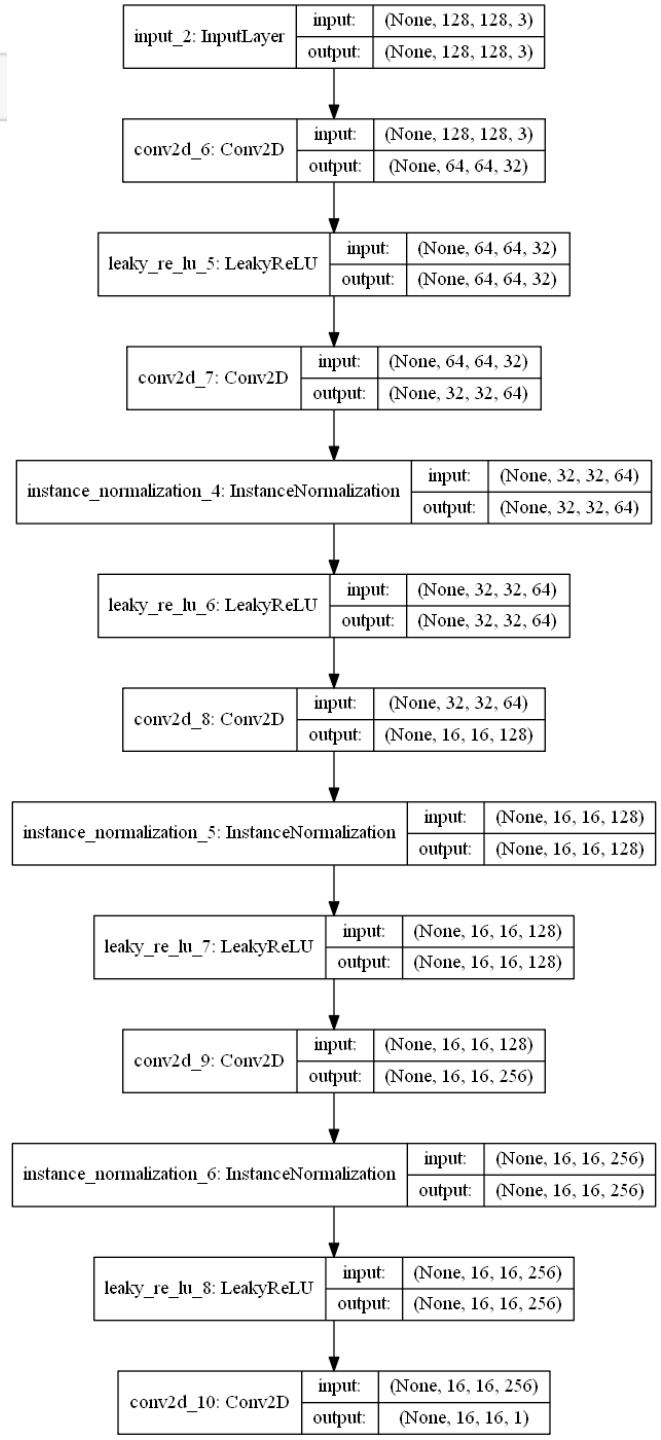
In [10]:
```
gan.d_A.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 128, 128, 3)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 64, 64, 32)        1568
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 32, 32, 64)        32832
_________________________________________________________________
instance_normalization_1 (In (None, 32, 32, 64)        0
_________________________________________________________________
leaky_re_lu_2 (LeakyReLU)    (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 128)       131200
_________________________________________________________________
instance_normalization_2 (In (None, 16, 16, 128)       0
_________________________________________________________________
leaky_re_lu_3 (LeakyReLU)    (None, 16, 16, 128)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 16, 16, 256)       524544
_________________________________________________________________
instance_normalization_3 (In (None, 16, 16, 256)       0
_________________________________________________________________
leaky_re_lu_4 (LeakyReLU)    (None, 16, 16, 256)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 16, 16, 1)         4097
=================================================================
Total params: 694,241
Trainable params: 694,241
Non-trainable params: 0
_________________________________________________________________
```



In [11]:
```
gan.d_B.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 128, 128, 3)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 64, 64, 32)        1568
_________________________________________________________________
leaky_re_lu_5 (LeakyReLU)    (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 32, 32, 64)        32832
_________________________________________________________________
instance_normalization_4 (In (None, 32, 32, 64)        0
_________________________________________________________________
leaky_re_lu_6 (LeakyReLU)    (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 16, 16, 128)       131200
_________________________________________________________________
instance_normalization_5 (In (None, 16, 16, 128)       0
_________________________________________________________________
leaky_re_lu_7 (LeakyReLU)    (None, 16, 16, 128)       0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 16, 16, 256)       524544
_________________________________________________________________
instance_normalization_6 (In (None, 16, 16, 256)       0
_________________________________________________________________
leaky_re_lu_8 (LeakyReLU)    (None, 16, 16, 256)       0
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 16, 16, 1)         4097
=================================================================
Total params: 694,241
Trainable params: 694,241
Non-trainable params: 0
_________________________________________________________________
```



모델 훈련

Note: CycleGAN 훈련 도중 주피터 커널이 예기치 않게 종료될 수 있습니다. 이럴 때는 쉘에서 05_01_cyclegan_train.py 파일을 실행하여 훈련하세요.

In [ ]:
```
BATCH_SIZE = 1
EPOCHS = 200
PRINT_EVERY_N_BATCHES = 10

TEST_A_FILE = 'n07740461_14740.jpg'
TEST_B_FILE = 'n07749192_4241.jpg'
```

Note: 이 훈련은 시간이 매우 오래 걸립니다. 깃허브에 훈련된 가중치와 손실이 저장되어 있으므로 훈련을 건너 뛰고 다음 셀을 실행해도 됩니다.

In [ ]:
```
gan.train(data_loader
        , run_folder = RUN_FOLDER
        , epochs=EPOCHS
        , test_A_file = TEST_A_FILE
        , test_B_file = TEST_B_FILE
        , batch_size=BATCH_SIZE
        , sample_interval=PRINT_EVERY_N_BATCHES)
```

결과


Original
Translated
Reconstructed
ID
Original
Translated
Reconstructed
ID

손실

Note: 앞에서 훈련을 직접 실행하지 않았다면 다음 셀의 주석을 제거하고 실행하세요

In [ ]:
```
#!gunzip run/paint/0001_apple2orange/loss.pkl.gz
```

In [ ]:
```
# import pickle

# loss = pickle.load(open(os.path.join(RUN_FOLDER, "loss.pkl"), "rb"))

# gan.d_losses = loss['d_losses']
# gan.g_losses = loss['g_losses']
```

In [ ]:
```
fig = plt.figure(figsize=(20,10))

# plt.plot([x[0] for x in gan.g_losses], color='black', linewidth=0.25)
plt.plot([x[0] for x in gan.d_losses], color='black', linewidth=0.1) #discriminator loss

plt.plot([x[1] for x in gan.g_losses], color='green', linewidth=0.1) #validation loss
# plt.plot([x[2] for x in gan.g_losses], color='orange', linewidth=0.1)

plt.plot([x[3] for x in gan.g_losses], color='blue', linewidth=0.1) #reconstr loss
# plt.plot([x[4] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.plot([x[5] for x in gan.g_losses], color='red', linewidth=0.1) #id loss
# plt.plot([x[6] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.xlabel('batch', fontsize=18)
plt.ylabel('loss', fontsize=16)

plt.ylim(0, 5)

plt.show()
```

손실

Note: 앞에서 훈련을 직접 실행하지 않았다면 다음 셀의 주석을 제거하고 실행하세요

In [12]:
```
!gunzip run/paint/0001_apple2orange/loss.pkl.gz
```

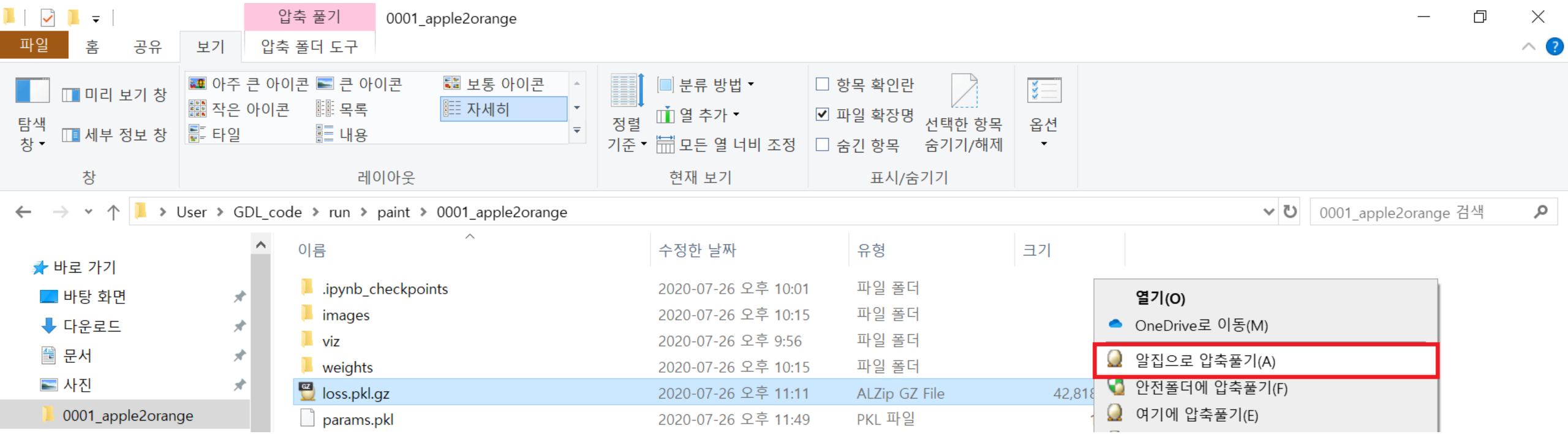
```
'gunzip'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
배치 파일이 아닙니다.
```

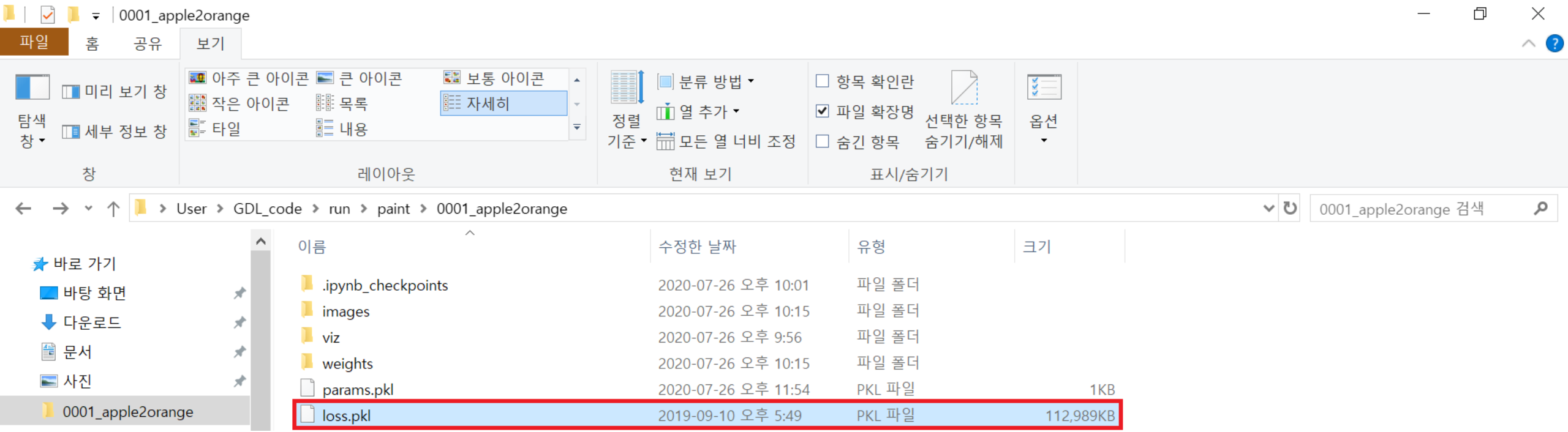
In [ ]:
```
import pickle

loss = pickle.load(open(os.path.join(RUN_FOLDER, "loss.pkl"), "rb"))

gan.d_losses = loss['d_losses']
gan.g_losses = loss['g_losses']
```

In [ ]:
```
fig = plt.figure(figsize=(20,10))

plt.plot([x[0] for x in gan.g_losses], color='black', linewidth=0.25)
plt.plot([x[0] for x in gan.d_losses], color='black', linewidth=0.1) #discriminator loss

plt.plot([x[1] for x in gan.g_losses], color='green', linewidth=0.1) #validation loss
plt.plot([x[2] for x in gan.g_losses], color='orange', linewidth=0.1)

plt.plot([x[3] for x in gan.g_losses], color='blue', linewidth=0.1) #reconstr loss
plt.plot([x[4] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.plot([x[5] for x in gan.g_losses], color='red', linewidth=0.1) #id loss
plt.plot([x[6] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.xlabel('batch', fontsize=18)
plt.ylabel('loss', fontsize=16)

plt.ylim(0, 5)

plt.show()
```

다음과 같은 오류는

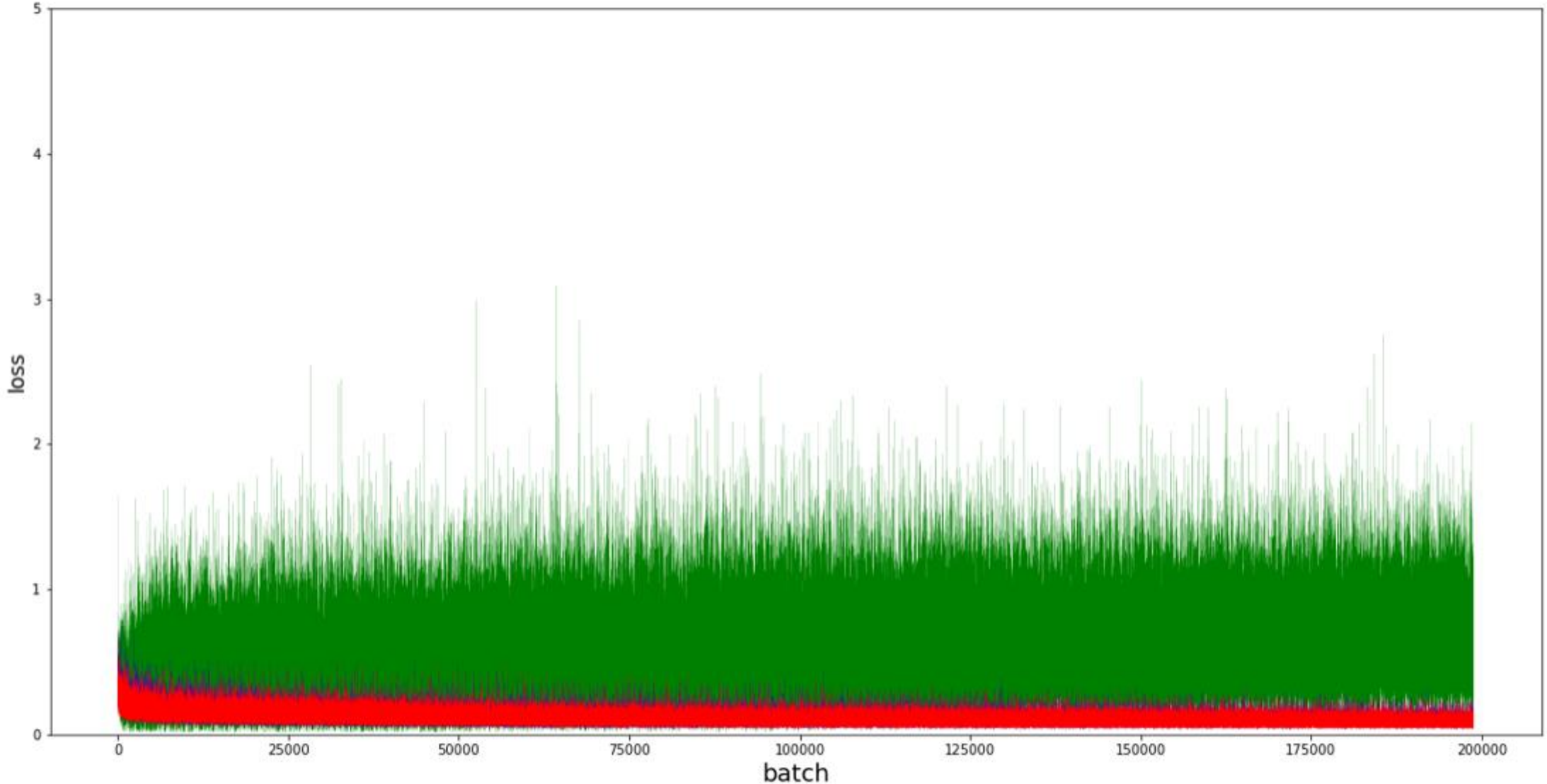
리눅스 환경에서 압축을 해제하거나 다음과 같이 알집으로 해제가 가능합니다.


압축 풀기
0001_apple2orange
파일
홈
공유
보기
압축 폴더 도구
미리 보기 창
세부 정보 창
탐색 창
아주 큰 아이콘
큰 아이콘
보통 아이콘
작은 아이콘
목록
자세히
타일
내용
창
레이아웃
정렬 기준
분류 방법
열 추가
모든 열 너비 조정
현재 보기
항목 확인란
파일 확장명
숨긴 항목
선택한 항목 숨기기/해제
표시/숨기기
옵션
User
GDL_code
run
paint
0001_apple2orange
0001_apple2orange 검색
바로 가기
바탕 화면
다운로드
문서
사진
0001_apple2orange
이름
수정한 날짜
유형
크기
.ipynb_checkpoints
2020-07-26 오후 10:01
파일 폴더
images
2020-07-26 오후 10:15
파일 폴더
viz
2020-07-26 오후 9:56
파일 폴더
weights
2020-07-26 오후 10:15
파일 폴더
loss.pkl.gz
2020-07-26 오후 11:11
ALZip GZ File
42,818
params.pkl
2020-07-26 오후 11:49
PKL 파일
열기(O)
OneDrive로 이동(M)
알집으로 압축풀기(A)
안전폴더에 압축풀기(F)
여기에 압축풀기(E)


0001_apple2orange
파일
홈
공유
보기
미리 보기 창
탐색 창
세부 정보 창
창
아주 큰 아이콘
큰 아이콘
보통 아이콘
작은 아이콘
목록
자세히
타일
내용
레이아웃
정렬 기준
분류 방법
열 추가
모든 열 너비 조정
현재 보기
항목 확인란
파일 확장명
숨긴 항목
선택한 항목 숨기기/해제
표시/숨기기
옵션
User › GDL_code › run › paint › 0001_apple2orange
0001_apple2orange 검색
바로 가기
바탕 화면
다운로드
문서
사진
0001_apple2orange
이름
수정한 날짜
유형
크기
.ipynb_checkpoints 2020-07-26 오후 10:01 파일 폴더
images 2020-07-26 오후 10:15 파일 폴더
viz 2020-07-26 오후 9:56 파일 폴더
weights 2020-07-26 오후 10:15 파일 폴더
params.pkl 2020-07-26 오후 11:54 PKL 파일 1KB
loss.pkl 2019-09-10 오후 5:49 PKL 파일 112,989KB

In [14]:

```
fig = plt.figure(figsize=(20,10))

#plt.plot([x[0] for x in gan.g_losses], color='black', linewidth=0.25)
#plt.plot([x[0] for x in gan.d_losses], color='black', linewidth=0.1) #discriminator loss

plt.plot([x[1] for x in gan.g_losses], color='green', linewidth=0.1) #validation loss
#plt.plot([x[2] for x in gan.g_losses], color='orange', linewidth=0.1)

plt.plot([x[3] for x in gan.g_losses], color='blue', linewidth=0.1) #reconstr loss
#plt.plot([x[4] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.plot([x[5] for x in gan.g_losses], color='red', linewidth=0.1) #id loss
#plt.plot([x[6] for x in gan.g_losses], color='orange', linewidth=0.25)

plt.xlabel('batch', fontsize=18)
plt.ylabel('loss', fontsize=16)

plt.ylim(0, 5)

plt.show()
```


5
4
3
2
1
0
loss
0
25000
50000
75000
100000
125000
150000
175000
200000
batch

CycleGAN 모네 그림 그리기

라이브러리 임포트

Note: 이 노트북의 코드를 실행하려면 `keras_contrib` 패키지를 설치해야 합니다. 다음 셀의 주석을 제거하고 실행하여 패키지를 설치하세요 잘 설치되지 않는다면 프롬프트 창에서 실행해 주세요.

- scipy==1.1.0 패키지로 변경해야 합니다. 그런 후 scipy.misc 라이브러리를 임포트 합니다.

In [1]:
```
import os
import matplotlib.pyplot as plt
import scipy.misc

from models.cycleGAN import CycleGAN
from utils.loaders import DataLoader
```

```
Using TensorFlow backend.
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:526: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:528: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:529: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:530: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/luxmaris16/anaconda3/envs/testGPU/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:535: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [2]:
```
# run params
SECTION = 'paint'
RUN_ID = '0001'
DATA_NAME = 'cezanne2photo'
RUN_FOLDER = 'run/{}/'.format(SECTION)
RUN_FOLDER += '_'.join([RUN_ID, DATA_NAME])

if not os.path.exists(RUN_FOLDER):
    os.mkdir(RUN_FOLDER)
    os.mkdir(os.path.join(RUN_FOLDER, 'viz'))
    os.mkdir(os.path.join(RUN_FOLDER, 'images'))
    os.mkdir(os.path.join(RUN_FOLDER, 'weights'))

mode =  'build' # 'build' #
```

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 모네 데이터셋을 다운로드하세요.

In [ ]:
```
#!./scripts/download_cyclegan_data.sh monet2photo
```

In [ ]:
```
IMAGE_SIZE = 256
```

In [ ]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 모네 데이터셋을 다운로드하세요.

In [3]:
```
! C:/Users/User/GDL_code/scripts/download_cyclegan_data.sh monet2photo
```

```
WARNING: timestamping does nothing in combination with -O. See the manual
for details.

--2020-07-26 16:50:23--  https://people.eecs.berkeley.edu/~taesung_park/CycleGAN/datasets/monet2photo.zip
Resolving people.eecs.berkeley.edu (people.eecs.berkeley.edu)... 128.32.189.73
Connecting to people.eecs.berkeley.edu (people.eecs.berkeley.edu)|128.32.189.73|:443... connected.
ERROR: The certificate of ‘people.eecs.berkeley.edu’ is not trusted.
ERROR: The certificate of ‘people.eecs.berkeley.edu’ has expired.
mkdir: cannot create directory ‘./data/monet2photo/’ : File exists
Archive:  ./data/monet2photo.zip
  End-of-central-directory signature not found.  Either this file is not
  a zipfile, or it constitutes one disk of a multi-part archive.  In the
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of ./data/monet2photo.zip or
        ./data/monet2photo.zip.zip, and cannot find ./data/monet2photo.zip.ZIP, period.
```

In [4]:
```
IMAGE_SIZE = 256
```

In [5]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

데이터 적재

노트북을 처음 실행할 때 다음 셀의 주석을 제거하고 실행하여 모네 데이터셋을 다운로드하세요.

- 셀 스크립트가 제대로 작동하지 않을 수 있으니 확인 후 정상 작동하지 않는다면 따로 받아서 준비해 주세요
  https://people.eecs.berkeley.edu/~taesung_park/CycleGAN/datasets/

In [3]:
```
#!./scripts/download_cyclegan_data.sh monet2photo
```

In [4]:
```
IMAGE_SIZE = 256
```

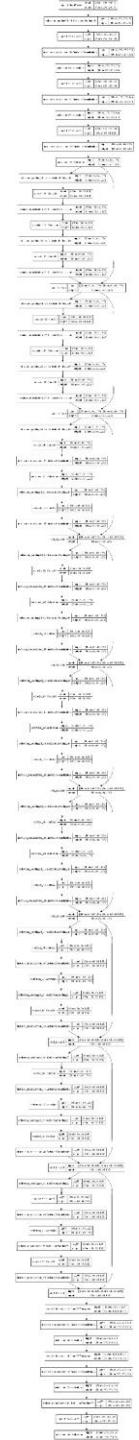
In [5]:
```
data_loader = DataLoader(dataset_name=DATA_NAME, img_res=(IMAGE_SIZE, IMAGE_SIZE))
```

In [7]:
```
gan.g_BA.summary()
```

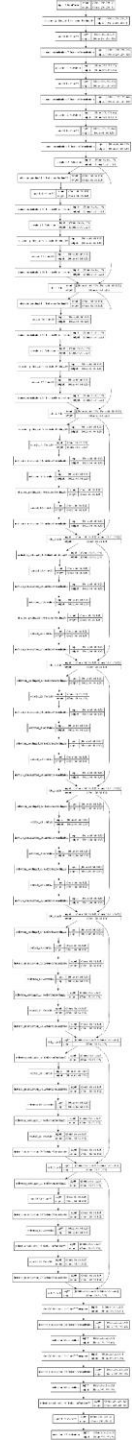
```
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected to
=================================================================================================
input_4 (InputLayer)         (None, 256, 256, 3)       0
_________________________________________________________________________________________________
reflection_padding2d_21 (Reflec (None, 262, 262, 3)    0         input_4[0][0]
_________________________________________________________________________________________________
conv2d_33 (Conv2D)           (None, 256, 256, 32)      4736      reflection_padding2d_21[0][0]
_________________________________________________________________________________________________
instance_normalization_30 (Inst (None, 256, 256, 32)   0         conv2d_33[0][0]
_________________________________________________________________________________________________
activation_16 (Activation)   (None, 256, 256, 32)      0         instance_normalization_30[0][0]
_________________________________________________________________________________________________
conv2d_34 (Conv2D)           (None, 128, 128, 64)      18496     activation_16[0][0]
_________________________________________________________________________________________________
instance_normalization_31 (Inst (None, 128, 128, 64)   0         conv2d_34[0][0]
_________________________________________________________________________________________________
activation_17 (Activation)   (None, 128, 128, 64)      0         instance_normalization_31[0][0]
_________________________________________________________________________________________________
```

In [8]:
```
gan.g_AB.summary()
```

```
_________________________________________________________________________________________________
instance_normalization_7 (Insta (None, 256, 256, 32)   0         conv2d_11[0][0]
_________________________________________________________________________________________________
activation_1 (Activation)    (None, 256, 256, 32)      0         instance_normalization_7[0][0]
_________________________________________________________________________________________________
conv2d_12 (Conv2D)           (None, 128, 128, 64)      18496     activation_1[0][0]
_________________________________________________________________________________________________
instance_normalization_8 (Insta (None, 128, 128, 64)   0         conv2d_12[0][0]
_________________________________________________________________________________________________
activation_2 (Activation)    (None, 128, 128, 64)      0         instance_normalization_8[0][0]
_________________________________________________________________________________________________
conv2d_13 (Conv2D)           (None, 64, 64, 128)       73856     activation_2[0][0]
_________________________________________________________________________________________________
instance_normalization_9 (Insta (None, 64, 64, 128)    0         conv2d_13[0][0]
_________________________________________________________________________________________________
activation_3 (Activation)    (None, 64, 64, 128)       0         instance_normalization_9[0][0]
_________________________________________________________________________________________________
reflection_padding2d_2 (Reflect (None, 66, 66, 128)    0         activation_3[0][0]
_________________________________________________________________________________________________
conv2d_14 (Conv2D)           (None, 64, 64, 128)       147584    reflection_padding2d_2[0][0]
```



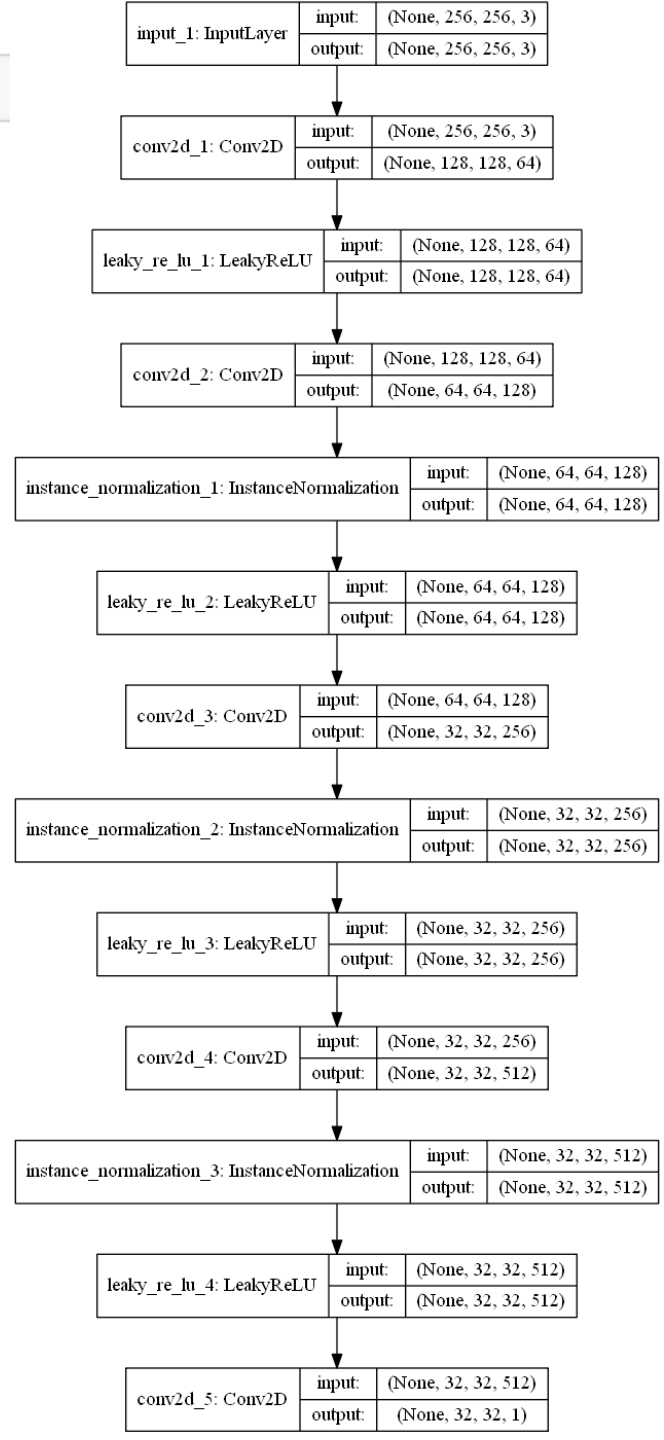
위 모델의 구조입니다.



위 모델의 구조입니다.

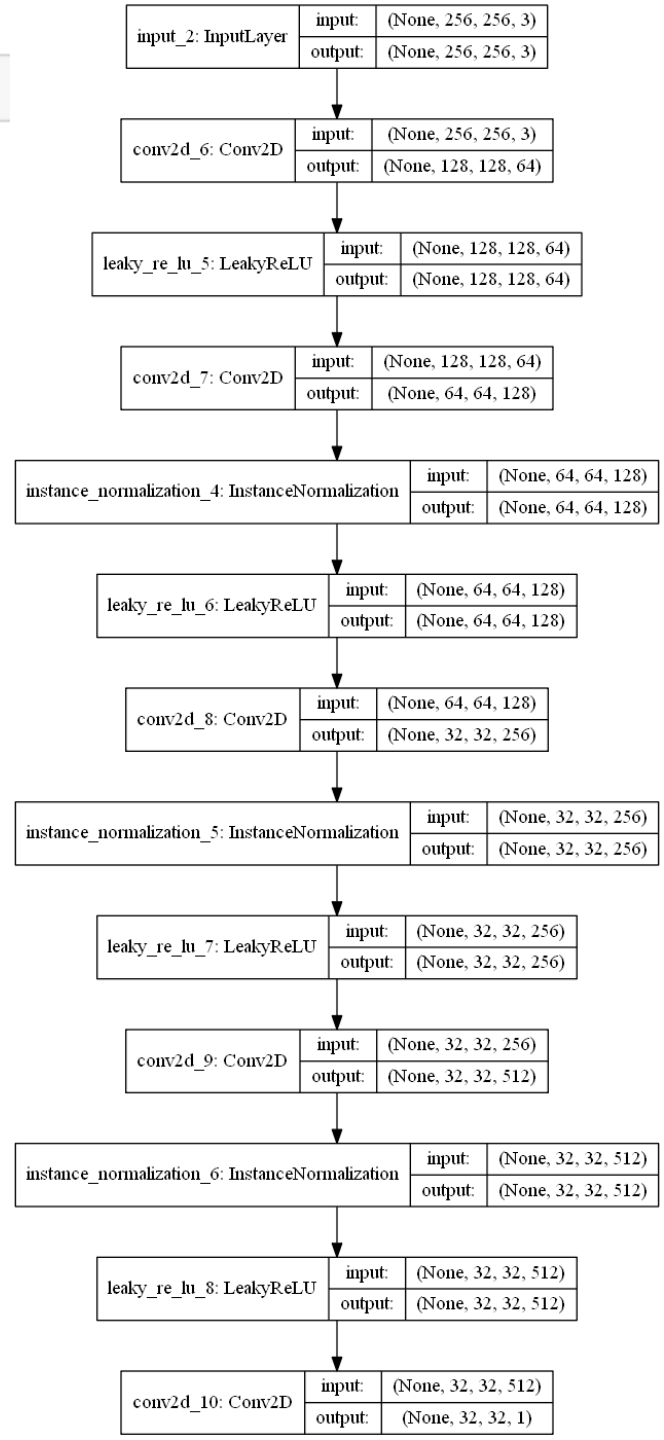
In [9]:
```
gan.d_A.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 256, 256, 3)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 128, 128, 64)      3136
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 128, 128, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 64, 64, 128)       131200
_________________________________________________________________
instance_normalization_1 (In (None, 64, 64, 128)       0
_________________________________________________________________
leaky_re_lu_2 (LeakyReLU)    (None, 64, 64, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 32, 32, 256)       524544
_________________________________________________________________
instance_normalization_2 (In (None, 32, 32, 256)       0
_________________________________________________________________
leaky_re_lu_3 (LeakyReLU)    (None, 32, 32, 256)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 32, 32, 512)       2097664
_________________________________________________________________
instance_normalization_3 (In (None, 32, 32, 512)       0
_________________________________________________________________
leaky_re_lu_4 (LeakyReLU)    (None, 32, 32, 512)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 32, 32, 1)         8193
=================================================================
Total params: 2,764,737
Trainable params: 2,764,737
Non-trainable params: 0
_________________________________________________________________
```



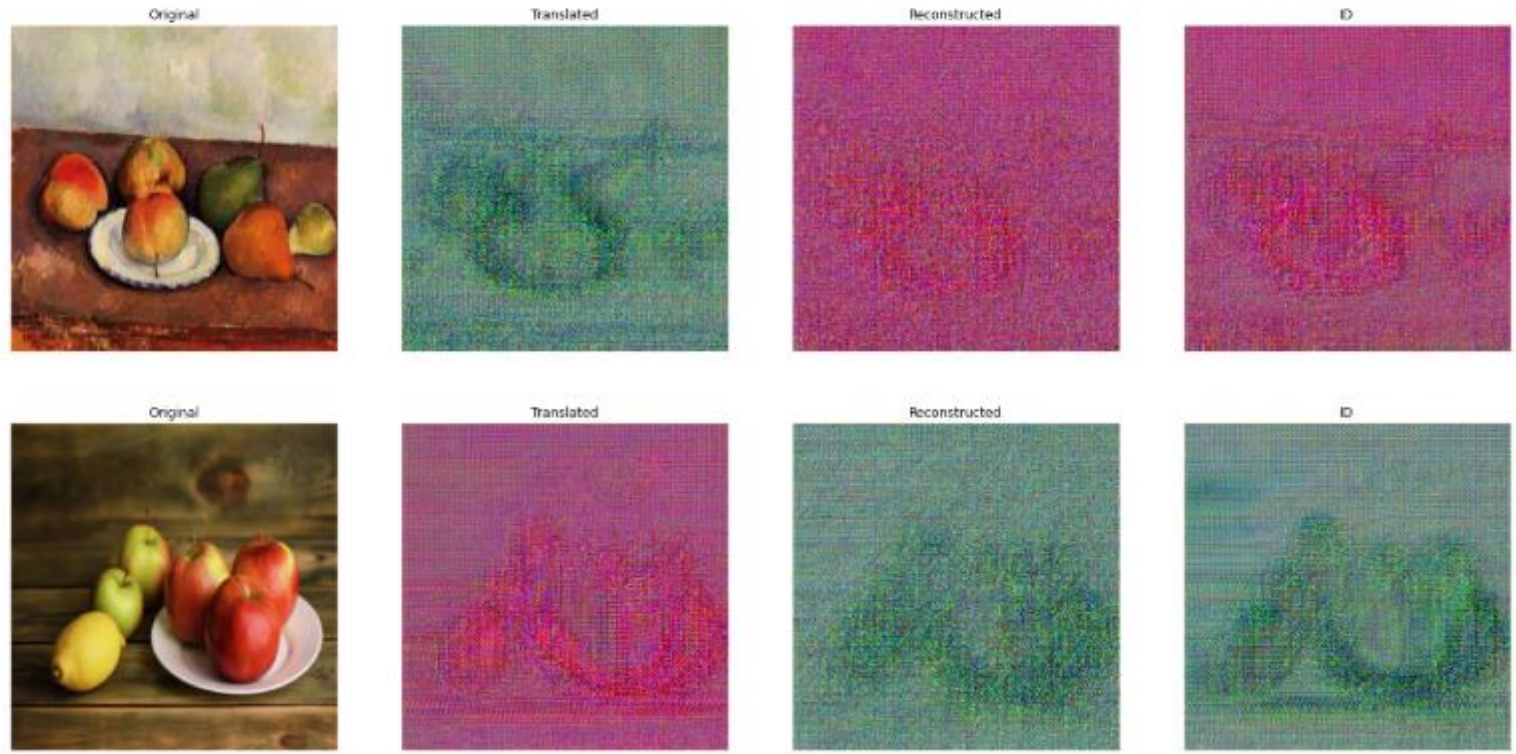
In [10]:
```
gan.d_B.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 256, 256, 3)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 128, 128, 64)      3136
_________________________________________________________________
leaky_re_lu_5 (LeakyReLU)    (None, 128, 128, 64)      0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 64, 64, 128)       131200
_________________________________________________________________
instance_normalization_4 (In (None, 64, 64, 128)       0
_________________________________________________________________
leaky_re_lu_6 (LeakyReLU)    (None, 64, 64, 128)       0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 32, 32, 256)       524544
_________________________________________________________________
instance_normalization_5 (In (None, 32, 32, 256)       0
_________________________________________________________________
leaky_re_lu_7 (LeakyReLU)    (None, 32, 32, 256)       0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 32, 32, 512)       2097664
_________________________________________________________________
instance_normalization_6 (In (None, 32, 32, 512)       0
_________________________________________________________________
leaky_re_lu_8 (LeakyReLU)    (None, 32, 32, 512)       0
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 32, 32, 1)         8193
=================================================================
Total params: 2,764,737
Trainable params: 2,764,737
Non-trainable params: 0
_________________________________________________________________
```

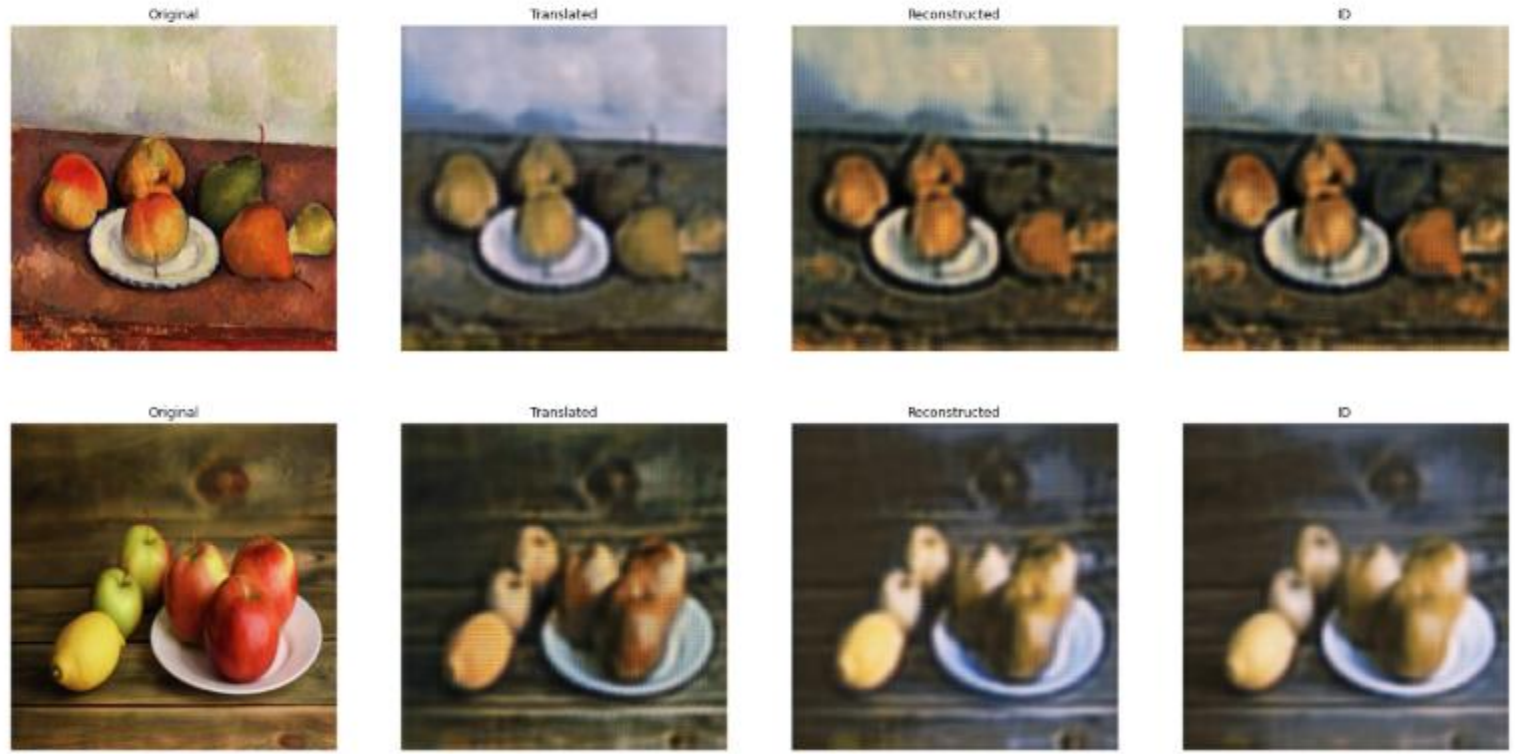


훈련 결과

배치 후


Original
Translated
Reconstructed
ID
Original
Translated
Reconstructed
ID

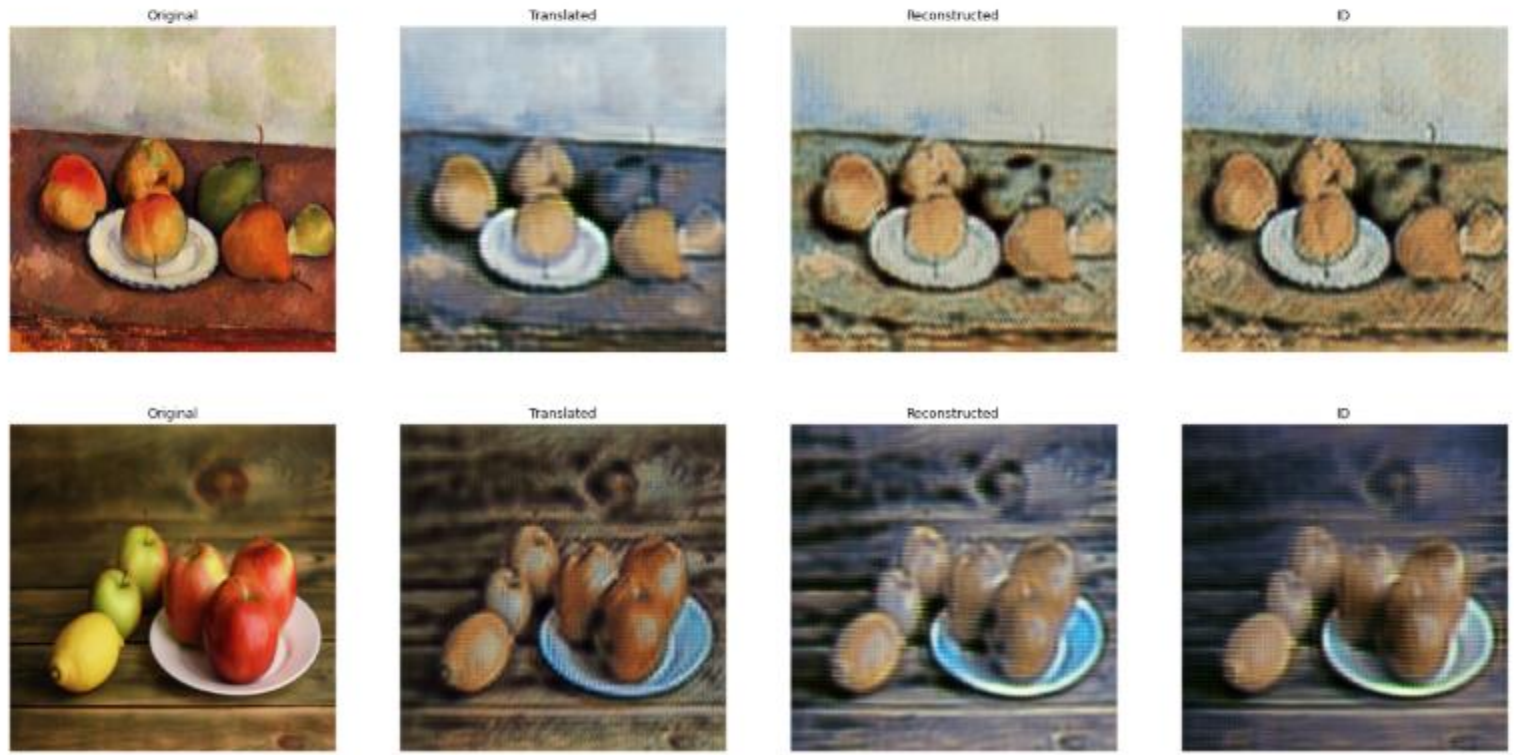
1 에포크


Original
Translated
Reconstructed
ID
Original
Translated
Reconstructed
ID

2 에포크


Original
Translated
Reconstructed
ID
Original
Translated
Reconstructed
ID

3 에포크


Original
Translated
Reconstructed
ID
Original
Translated
Reconstructed
ID

뉴럴 스타일 트랜스퍼

Note: 이 코드는 <케라스 창시자에게 배우는 딥러닝> 8.3절의 주피터 노트북에서 가져왔습니다.

In [ ]:
```
from keras.preprocessing.image import load_img, img_to_array, save_img

# 변환하려는 이미지 경로
target_image_path = './data/neural_style_transfer/tubingen.jpg'
# 스타일 이미지 경로
style_reference_image_path = './data/neural_style_transfer/starry-night.jpg'

# 생성된 사진의 차원
width, height = load_img(target_image_path).size
img_height = 600
img_width = int(width * img_height / height)

print(img_width, img_height)
```

In [ ]:
```
import numpy as np
from keras.applications import vgg19

def preprocess_image(image_path):
    img = load_img(image_path, target_size=(img_height, img_width))
    img = img_to_array(img)
    img = np.expand_dims(img, axis=0)
    img = vgg19.preprocess_input(img)
    return img

def deprocess_image(x):
    # ImageNet의 평균 픽셀 값을 더합니다
    x[:, :, 0] += 103.939
    x[:, :, 1] += 116.779
    x[:, :, 2] += 123.68
    # 'BGR'->'RGB'
    x = x[:, :, ::-1]
    x = np.clip(x, 0, 255).astype('uint8')
    return x
```

뉴럴 스타일 트랜스퍼

Note: 이 코드는 <케라스 창시자에게 배우는 딥러닝> 8.3절의 주피터 노트북에서 가져왔습니다.

In [ ]:
```
from keras.preprocessing.image import load_img, img_to_array, save_img

# 변환하려는 이미지 경로
target_image_path = 'C:/Users/User/GDL_code/data/neural_style_transfer/gwanghwamun.jpg'
# 스타일 이미지 경로
style_reference_image_path = 'C:/Users/User/GDL_code/data/neural_style_transfer/starry-night.jpg'

# 생성된 사진의 차원
width, height = load_img(target_image_path).size
img_height = 600
img_width = int(width * img_height / height)

print(img_width, img_height)
```

In [ ]:
```
import numpy as np
from keras.applications import vgg19

def preprocess_image(image_path):
    img = load_img(image_path, target_size=(img_height, img_width))
    img = img_to_array(img)
    img = np.expand_dims(img, axis=0)
    img = vgg19.preprocess_input(img)
    return img

def deprocess_image(x):
    # ImageNet의 평균 픽셀 값을 더합니다
    x[:, :, 0] += 103.939
    x[:, :, 1] += 116.779
    x[:, :, 2] += 123.68
    # 'BGR'->'RGB'
    x = x[:, :, ::-1]
    x = np.clip(x, 0, 255).astype('uint8')
    return x
```

뉴럴 스타일 트랜스퍼

Note: 이 코드는 <케라스 창시자에게 배우는 딥러닝> 8.3절의 주피터 노트북에서 가져왔습니다.

In [1]:
```
from keras.preprocessing.image import load_img, img_to_array, save_img

# 변환하려는 이미지 경로
target_image_path = 'C:/Users/User/GDL_code/data/neural_style_transfer/gwanghwamun.jpg'
# 스타일 이미지 경로
style_reference_image_path = 'C:/Users/User/GDL_code/data/neural_style_transfer/starry-night.jpg'

# 생성된 사진의 차원
width, height = load_img(target_image_path).size
img_height = 600
img_width = int(width * img_height / height)

print(img_width, img_height)
```

```
Using TensorFlow backend.
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:526: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:528: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:529: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:530: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/luxmaris16/anaconda3/envs/testGAN/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:535: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

```
900 600
```

In [2]:
```
import numpy as np
from keras.applications import vgg19

def preprocess_image(image_path):
    img = load_img(image_path, target_size=(img_height, img_width))
    img = img_to_array(img)
    img = np.expand_dims(img, axis=0)
    img = vgg19.preprocess_input(img)
    return img

def deprocess_image(x):
    # ImageNet의 평균 픽셀 값을 더합니다
    x[:, :, 0] += 103.939
    x[:, :, 1] += 116.779
    x[:, :, 2] += 123.68
    # 'BGR'->'RGB'
    x = x[:, :, ::-1]
    x = np.clip(x, 0, 255).astype('uint8')
    return x
```

In [3]:
```
from keras import backend as K

target_image = K.constant(preprocess_image(target_image_path))
style_reference_image = K.constant(preprocess_image(style_reference_image_path))

# 생성된 이미지를 담을 플레이스홀더
combination_image = K.placeholder((1, img_height, img_width, 3))

# 세 개의 이미지를 하나의 배치로 합칩니다
input_tensor = K.concatenate([target_image,
                              style_reference_image,
                              combination_image], axis=0)

# 세 이미지의 배치를 입력으로 받는 VGG 네트워크를 만듭니다.
# 이 모델은 사전 훈련된 ImageNet 가중치를 로드합니다
model = vgg19.VGG19(input_tensor=input_tensor,
                    weights='imagenet',
                    include_top=False)
```

```
WARNING:tensorflow:From C:₩Users₩User₩anaconda3₩envs₩testGAN₩lib₩site-packages₩tensorflow₩python₩framework₩op_def_library.py:263: colocate_
with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg19_weights_tf_dim_ordering_tf_kernels_noto
p.h5
80142336/80134624 [==============================] - 19s 0us/step
```

In [4]:
```
def content_loss(base, combination):
    return K.sum(K.square(combination - base))
```

In [5]:
```
def gram_matrix(x):
    features = K.batch_flatten(K.permute_dimensions(x, (2, 0, 1)))
    gram = K.dot(features, K.transpose(features))
    return gram


def style_loss(style, combination):
    S = gram_matrix(style)
    C = gram_matrix(combination)
    channels = 3
    size = img_height * img_width
    return K.sum(K.square(S - C)) / (4. * (channels ** 2) * (size ** 2))
```

In [6]:
```
def total_variation_loss(x):
    a = K.square(
        x[:, :img_height - 1, :img_width - 1, :] - x[:, 1:, :img_width - 1, :])
    b = K.square(
        x[:, :img_height - 1, :img_width - 1, :] - x[:, :img_height - 1, 1:, :])
    return K.sum(K.pow(a + b, 1.25))
```

In [7]:
```
# 층 이름과 활성화 텐서를 매핑한 딕셔너리
outputs_dict = dict([(layer.name, layer.output) for layer in model.layers])
# 콘텐츠 손실에 사용할 층
content_layer = 'block5_conv2'
# 스타일 손실에 사용할 층
style_layers = ['block1_conv1',
                'block2_conv1',
                'block3_conv1',
                'block4_conv1',
                'block5_conv1']
# 손실 항목의 가중치 평균에 사용할 가중치
total_variation_weight = 1
style_weight = 100
content_weight = 20

# 모든 손실 요소를 더해 하나의 스칼라 변수로 손실을 정의합니다
loss = K.variable(0.)
layer_features = outputs_dict[content_layer]
target_image_features = layer_features[0, :, :, :]
combination_features = layer_features[2, :, :, :]
loss += content_weight * content_loss(target_image_features,
                                      combination_features)
for layer_name in style_layers:
    layer_features = outputs_dict[layer_name]
    style_reference_features = layer_features[1, :, :, :]
    combination_features = layer_features[2, :, :, :]
    sl = style_loss(style_reference_features, combination_features)
    loss += (style_weight / len(style_layers)) * sl
loss += total_variation_weight * total_variation_loss(combination_image)
```

```
WARNING:tensorflow:Variable += will be deprecated. Use variable.assign_add if you want assignment to the variable value or 'x = x + y' if y
ou want a new python Tensor object.
```

In [8]:
```
# 손실에 대한 생성된 이미지의 그래디언트를 구합니다
grads = K.gradients(loss, combination_image)[0]

# 현재 손실과 그래디언트의 값을 추출하는 케라스 Function 객체입니다
fetch_loss_and_grads = K.function([combination_image], [loss, grads])


class Evaluator(object):

    def __init__(self):
        self.loss_value = None
        self.grads_values = None

    def loss(self, x):
        assert self.loss_value is None
        x = x.reshape((1, img_height, img_width, 3))
        outs = fetch_loss_and_grads([x])
        loss_value = outs[0]
        grad_values = outs[1].flatten().astype('float64')
        self.loss_value = loss_value
        self.grad_values = grad_values
        return self.loss_value

    def grads(self, x):
        assert self.loss_value is not None
        grad_values = np.copy(self.grad_values)
        self.loss_value = None
        self.grad_values = None
        return grad_values

evaluator = Evaluator()
```

In [ ]:

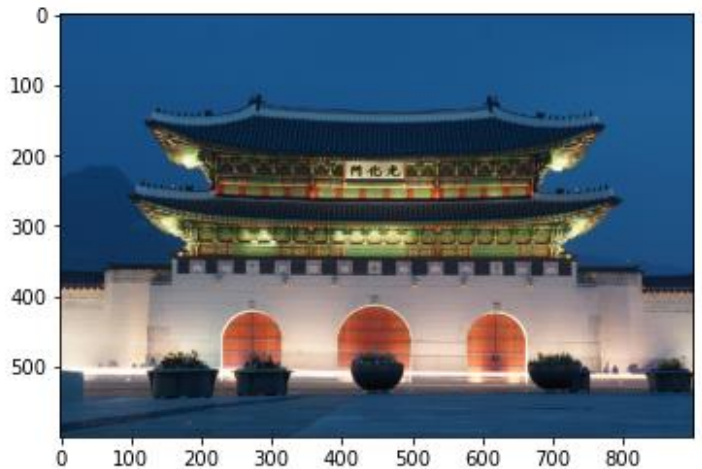
```
from scipy.optimize import fmin_l_bfgs_b

result_file = './data/neural_style_transfer/style_transfer_result.png'
iterations = 1000

# 뉴럴 스타일 트랜스퍼의 손실을 최소화하기 위해 생성된 이미지에 대해 L-BFGS 최적화를 수행합니다
# 초기 값은 타깃 이미지입니다
# scipy.optimize.fmin_l_bfgs_b 함수가 벡터만 처리할 수 있기 때문에 이미지를 펼칩니다.
x = preprocess_image(target_image_path)
x = x.flatten()
for i in range(iterations):
    x, min_val, info = fmin_l_bfgs_b(evaluator.loss, x,
                                     fprime=evaluator.grads, maxfun=20)
    if i % 100 == 0:
        print('.', end=' ')
        print('현재 손실 값:', min_val)

# 생성된 현재 이미지를 저장합니다
img = x.copy().reshape((img_height, img_width, 3))
img = deprocess_image(img)

save_img(result_file, img)
```

In [9]:
```
from scipy.optimize import fmin_l_bfgs_b

result_file = './data/neural_style_transfer/style_transfer_result_test.png'
iterations = 1000

# 뉴럴 스타일 트랜스퍼의 손실을 최소화하기 위해 생성된 이미지에 대해 L-BFGS 최적화를 수행합니다
# 초기 값은 타깃 이미지입니다
# scipy.optimize.fmin_l_bfgs_b 함수가 벡터만 처리할 수 있기 때문에 이미지를 펼칩니다.
x = preprocess_image(target_image_path)
x = x.flatten()
for i in range(iterations):
    x, min_val, info = fmin_l_bfgs_b(evaluator.loss, x,
                                     fprime=evaluator.grads, maxfun=20)
    if i % 100 == 0:
        print('.', end=' ')
        print('현재 손실 값:', min_val)

# 생성된 현재 이미지를 저장합니다
img = x.copy().reshape((img_height, img_width, 3))
img = deprocess_image(img)

save_img(result_file, img)
```

```
. 현재 손실 값: 77277550000.0
. 현재 손실 값: 10718404000.0
. 현재 손실 값: 10587559000.0
. 현재 손실 값: 10582067000.0
. 현재 손실 값: 10582067000.0
. 현재 손실 값: 10582065000.0
. 현재 손실 값: 10582067000.0
. 현재 손실 값: 10582065000.0
. 현재 손실 값: 10582065000.0
. 현재 손실 값: 10582067000.0
```

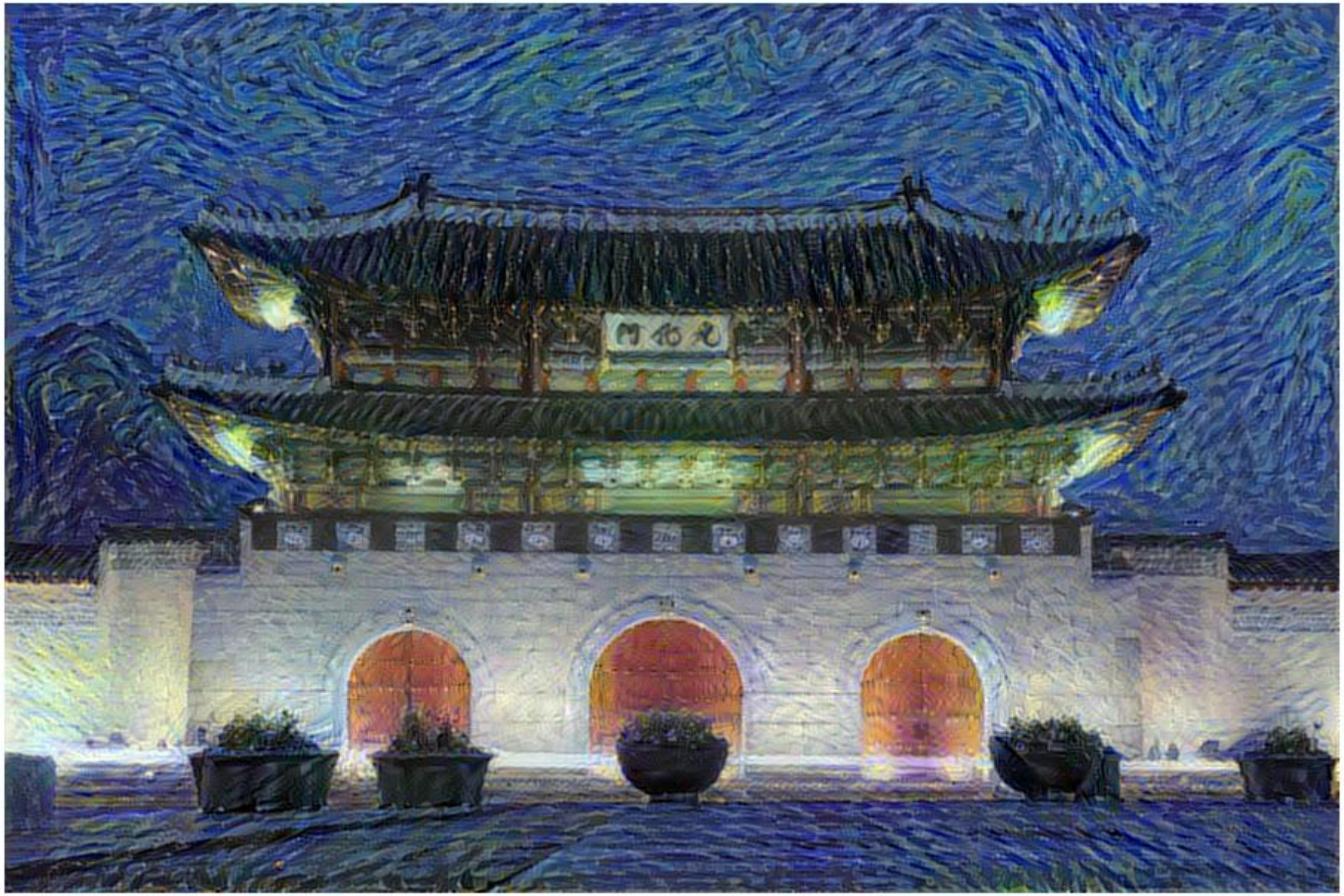
In [10]:
```
from matplotlib import pyplot as plt
```

In [11]:
```
# 콘텐츠 이미지
plt.imshow(load_img(target_image_path, target_size=(img_height, img_width)))
plt.figure()

# 스타일 이미지
plt.imshow(load_img(style_reference_image_path, target_size=(img_height, img_width)))
plt.figure()

plt.show()
```



양승우

결과 이미지



양승우

수고하셨습니다.