Conditional Generative Adversarial Networks for Sketch to Face transformation

Group Number : S21DL20

Korupoulu Saideepthi - S20180010087

Varakala Sowmya - S20180010187

Manjju Shree Devy - S20180010055

Swathi Kedarasetty - S20180010172

In [43]:
```
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 /tmp/p
ip-req-build-69xhxf5p
  Running command git clone -q https://www.github.com/keras-team/keras-
contrib.git /tmp/pip-req-build-69xhxf5p
Requirement already satisfied (use --upgrade to upgrade): keras-contrib
==2.0.8 from git+https://www.github.com/keras-team/keras-contrib.git in
/usr/local/lib/python3.7/dist-packages
Requirement already satisfied: keras in /usr/local/lib/python3.7/dist-p
ackages (from keras-contrib==2.0.8) (2.4.3)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.7/dist-
packages (from keras->keras-contrib==2.0.8) (3.13)
Requirement already satisfied: scipy>=0.14 in /usr/local/lib/python3.7/
dist-packages (from keras->keras-contrib==2.0.8) (1.4.1)
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.
7/dist-packages (from keras->keras-contrib==2.0.8) (1.19.5)
Requirement already satisfied: h5py in /usr/local/lib/python3.7/dist-pa
```

```
ckages (from keras->keras-contrib==2.0.8) (2.10.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-pac
kages (from h5py->keras->keras-contrib==2.0.8) (1.15.0)
Building wheels for collected packages: keras-contrib
  Building wheel for keras-contrib (setup.py) ... done
  Created wheel for keras-contrib: filename=keras_contrib-2.0.8-cp37-no
ne-any.whl size=101065 sha256=178259cd1cffccb3db413c6b74186072bee25812c
76210105d34f13611e0304c
  Stored in directory: /tmp/pip-ephem-wheel-cache-f8v56itq/wheels/11/2
7/c8/4ed56de7b55f4f61244e2dc6ef3cdbaff2692527a2ce6502ba
Successfully built keras-contrib
```

Installing tensorflow version 2.2.0 using pip

In [44]:
```
pip install tensorflow==2.2.0
```

```
Requirement already satisfied: tensorflow==2.2.0 in /usr/local/lib/pyth
on3.7/dist-packages (2.2.0)
Requirement already satisfied: gast==0.3.3 in /usr/local/lib/python3.7/
dist-packages (from tensorflow==2.2.0) (0.3.3)
Requirement already satisfied: wheel>=0.26; python_version >= "3" in /u
sr/local/lib/python3.7/dist-packages (from tensorflow==2.2.0) (0.36.2)
Requirement already satisfied: tensorflow-estimator<2.3.0,>=2.2.0 in /u
sr/local/lib/python3.7/dist-packages (from tensorflow==2.2.0) (2.2.0)
Requirement already satisfied: numpy<2.0,>=1.16.0 in /usr/local/lib/pyt
hon3.7/dist-packages (from tensorflow==2.2.0) (1.19.5)
Requirement already satisfied: astunparse==1.6.3 in /usr/local/lib/pyth
on3.7/dist-packages (from tensorflow==2.2.0) (1.6.3)
Requirement already satisfied: grpcio>=1.8.6 in /usr/local/lib/python3.
7/dist-packages (from tensorflow==2.2.0) (1.32.0)
Requirement already satisfied: scipy==1.4.1; python_version >= "3" in /
usr/local/lib/python3.7/dist-packages (from tensorflow==2.2.0) (1.4.1)
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in /usr/local/lib/p
ython3.7/dist-packages (from tensorflow==2.2.0) (2.10.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/pytho
n3.7/dist-packages (from tensorflow==2.2.0) (1.1.0)
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.
7/dist-packages (from tensorflow==2.2.0) (1.12.1)
```

```
Requirement already satisfied: keras-preprocessing>=1.1.0 in /usr/loca
l/lib/python3.7/dist-packages (from tensorflow==2.2.0) (1.1.2)
Requirement already satisfied: google-pasta>=0.1.8 in /usr/local/lib/py
thon3.7/dist-packages (from tensorflow==2.2.0) (0.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/pyth
on3.7/dist-packages (from tensorflow==2.2.0) (3.3.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.7/
dist-packages (from tensorflow==2.2.0) (1.15.0)
Requirement already satisfied: absl-py>=0.7.0 in /usr/local/lib/python
3.7/dist-packages (from tensorflow==2.2.0) (0.12.0)
Requirement already satisfied: protobuf>=3.8.0 in /usr/local/lib/python
3.7/dist-packages (from tensorflow==2.2.0) (3.12.4)
Requirement already satisfied: tensorboard<2.3.0,>=2.2.0 in /usr/local/
lib/python3.7/dist-packages (from tensorflow==2.2.0) (2.2.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/d
ist-packages (from protobuf>=3.8.0->tensorflow==2.2.0) (54.2.0)
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/pyth
on3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0)
(1.0.1)
Requirement already satisfied: google-auth<2,>=1.6.3 in /usr/local/lib/
python3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->tensorflow==2.
2.0) (1.28.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /us
r/local/lib/python3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->te
nsorflow==2.2.0) (0.4.3)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python
3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0)
(3.3.4)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/lo
cal/lib/python3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->tensor
flow==2.2.0) (1.8.0)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/py
thon3.7/dist-packages (from tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.
0) (2.23.0)
Requirement already satisfied: rsa<5,>=3.1.4; python_version >= "3.6" i
n /usr/local/lib/python3.7/dist-packages (from google-auth<2,>=1.6.3->t
ensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (4.7.2)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/li
b/python3.7/dist-packages (from google-auth<2,>=1.6.3->tensorboard<2.3.
0,>=2.2.0->tensorflow==2.2.0) (4.2.1)
```

```
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/
python3.7/dist-packages (from google-auth<2,>=1.6.3->tensorboard<2.3.0,
>=2.2.0->tensorflow==2.2.0) (0.2.8)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/l
ib/python3.7/dist-packages (from google-auth-oauthlib<0.5,>=0.4.1->tens
orboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (1.3.0)
Requirement already satisfied: importlib-metadata; python_version < "3.
8" in /usr/local/lib/python3.7/dist-packages (from markdown>=2.6.8->ten
sorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (3.8.1)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.
7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.3.0,>=2.2.0->t
ensorflow==2.2.0) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->te
nsorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/pyth
on3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.3.0,>=2.2.
0->tensorflow==2.2.0) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/pyt
hon3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.3.0,>=2.
2.0->tensorflow==2.2.0) (2020.12.5)
Requirement already satisfied: pyasn1>=0.1.3 in /usr/local/lib/python3.
7/dist-packages (from rsa<5,>=3.1.4; python_version >= "3.6"->google-au
th<2,>=1.6.3->tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python
3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<
0.5,>=0.4.1->tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (3.1.0)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/di
st-packages (from importlib-metadata; python_version < "3.8"->markdown>
=2.6.8->tensorboard<2.3.0,>=2.2.0->tensorflow==2.2.0) (3.4.1)
Requirement already satisfied: typing-extensions>=3.6.4; python_version
< "3.8" in /usr/local/lib/python3.7/dist-packages (from importlib-metad
ata; python_version < "3.8"->markdown>=2.6.8->tensorboard<2.3.0,>=2.2.0
->tensorflow==2.2.0) (3.7.4.3)
```

In [45]:
```
# Importing all the necessary libraries
from __future__ import print_function, division

from keras_contrib.layers.normalization.instancenormalization import In
```

```
stanceNormalization
from tensorflow.keras.layers import Input, Dense, Reshape, Flatten, Dro
pout, Concatenate, BatchNormalization, Activation, ZeroPadding2D
from tensorflow.python.keras.layers.advanced_activations import LeakyRe
LU
from tensorflow.python.keras.layers.convolutional import UpSampling2D,
Conv2D
from tensorflow.python.keras.models import Sequential, Model
from tensorflow.keras.optimizers import Adam

from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img

from sklearn.utils import shuffle
import matplotlib.pyplot as plt
import numpy as np
import datetime
import natsort
import scipy
import sys
import os
import cv2
import numpy
```

In [46]:
```
# Mounting the drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remoun
t, call drive.mount("/content/drive", force_remount=True).
```

In [47]:
```
%cd drive/MyDrive/Face-Sketch-to-Image-Generation-using-GAN-master/
```

```
[Errno 2] No such file or directory: 'drive/MyDrive/Face-Sketch-to-Imag
e-Generation-using-GAN-master/'
/content/drive/MyDrive/Face-Sketch-to-Image-Generation-using-GAN-master
```

Helper Function

In [48]:
```
# loading the files using natsort to sort the paths
def load_filename(path):
    dirFiles = os.listdir(path)
    for i, file in enumerate(dirFiles):
        dirFiles[i] = path + file
    return natsort.natsorted(dirFiles ,reverse=False)

# load all images in a directory into memory
def load_images(list_path, size=(256, 256)):
    img_list = list()
    # enumerate filenames in directory, assume all are images
    for filename in list_path:
        # load and resize the image
        pixels = load_img(filename, target_size=size)
        # convert to numpy array
        pixels = img_to_array(pixels)
        pixels = (pixels - 127.5) / 127.5
        img_list.append(pixels)
    return np.asarray(img_list)
```

In [49]:
```
# select a batch of random samples, returns images and target
def generate_real_samples(dataset, n_samples, patch_shape):
    # unpack dataset
    trainA, trainB = dataset

    # choose random instances
    ix = np.random.randint(0, trainA.shape[0], n_samples)

    # retrieve selected images
    X1, X2 = trainA[ix], trainB[ix]

    # generate 'real' class labels (1)
    y = np.ones((n_samples, patch_shape, patch_shape, 1))

    return [X1, X2], y
```

```
# generate a batch of images, returns images and targets
def generate_fake_samples(g_model, samples, patch_shape):
    # generate fake instance
    X = g_model.predict(samples)

    # create 'fake' class labels (0)
    y = np.zeros((len(X), patch_shape, patch_shape, 1))

    return X, y
```

In [50]:

```
# generate samples and save as a plot and save the model
def summarize_performance(step, g_model, d_model, dataset, target_dir=
'', n_samples=3):
    if target_dir and not os.path.exists(target_dir):
        os.mkdir(target_dir)
    # select a sample of input images
    [X_realA, X_realB], _ = generate_real_samples(dataset, n_samples, 1
)
    # generate a batch of fake samples
    X_fakeB, _ = generate_fake_samples(g_model, X_realA, 1)
    # scale all pixels from [-1,1] to [0,1]
    X_realA = (X_realA + 1) / 2.0
    X_realB = (X_realB + 1) / 2.0
    X_fakeB = (X_fakeB + 1) / 2.0
    # plot real source images
    for i in range(n_samples):
        plt.subplot(3, n_samples, 1 + i)
        plt.axis('off')
        plt.imshow(X_realA[i])
    # plot generated target image
    for i in range(n_samples):
        plt.subplot(3, n_samples, 1 + n_samples + i)
        plt.axis('off')
        plt.imshow(X_fakeB[i])
    # plot real target image
    for i in range(n_samples):
        plt.subplot(3, n_samples, 1 + n_samples*2 + i)
        plt.axis('off')
        plt.imshow(X_realB[i])
```

```
    # save plot to file
    filename1 = 'plot_%06d.png' % (step+1)
    plt.savefig(target_dir + filename1)
    plt.close()
    # save the generator model
    g_model.save(target_dir + 'g_model.h5')

    # save the discriminator model
    d_model.save(target_dir + 'd_model.h5')

    print('>Saved: %s and %s' % (filename1, 'g_model & d_model'))
```

Generator

In [51]:

```
# Noise and conditional data are combined to create a dense code of the
output image.
# The dense code is "upsampled" via deconvolution into image space.

def generator(img_shape):
    def conv2d(layer_in, n_filter, norm=True):
        d = Conv2D(n_filter, kernel_size=4, strides=2, padding='same')(
layer_in)
        d = LeakyReLU(0.2)(d)
        if norm:
            d = InstanceNormalization()(d)
        return d

    def deconv2d(layer_in, skip_in, n_filter, dropout=0.5):
        d = UpSampling2D(size=2)(layer_in)
        d = Conv2D(n_filter, kernel_size=4, strides=1, padding='same',
activation='relu')(d)
        if dropout:
            d = Dropout(dropout)(d)
        d = InstanceNormalization()(d)
        d = Concatenate()([d, skip_in])
        return d
```

```
    # Input Layer
    in_img = Input(shape=img_shape)

    # Downsampling
    d1 = conv2d(in_img, 64, norm=False)
    d2 = conv2d(d1, 128)
    d3 = conv2d(d2, 256)
    d4 = conv2d(d3, 512)
    d5 = conv2d(d4, 512)
    d6 = conv2d(d5, 512)
    d7 = conv2d(d6, 512)

    # Upsampling
    u1 = deconv2d(d7, d6, 512)
    u2 = deconv2d(u1, d5, 512)
    u3 = deconv2d(u2, d4, 512)
    u4 = deconv2d(u3, d3, 256, dropout=0)
    u5 = deconv2d(u4, d2, 128, dropout=0)
    u6 = deconv2d(u5, d1, 64, dropout=0)
    u7 = UpSampling2D(size=2)(u6)

    out_img = Conv2D(3, kernel_size=4, strides=1, padding='same', activ
ation='tanh')(u7)

    return Model(in_img, out_img, name='generator')
```

Discriminator

In [52]:
```
# Images are transformed via convolution into a dense code in discrimin
ator function.
def discriminator(img_shape):
    def d_layer(layer_in, n_filter, norm=True):
        d = Conv2D(n_filter, kernel_size=4, strides=2, padding='same')(
layer_in)
        d = LeakyReLU(0.2)(d)
        if norm:
            d = InstanceNormalization()(d)
```

```
        return d

    in_src_img = Input(shape=img_shape)
    in_target_img = Input(shape=img_shape)

    merged = Concatenate()([in_src_img, in_target_img])

    d1 = d_layer(merged, 64, norm=False)
    d2 = d_layer(d1, 128)
    d3 = d_layer(d1, 256)
    d4 = d_layer(d1, 512)

    out = Conv2D(1, kernel_size=4, strides=1, padding='same')(d4)

    return Model([in_src_img, in_target_img], out, name='discriminator'
)
```

GAN

In [53]:
```
def GAN(g_model, d_model, img_shape):
    d_model.trainable = False
    in_img = Input(shape=img_shape)
    gen_out = g_model(in_img)
    dis_out = d_model([in_img, gen_out])
    model = Model(in_img, [dis_out, gen_out], name='GAN')
    return model
```

Train GAN model

In [54]:
```
def train(d_model, g_model, gan_model, data, target_dir, n_epochs=100,
n_batch=16):
    # determine the output square shape of the discriminator
    n_patch = d_model.output_shape[1]

    blue_photo = data[0]
```

```
    blue_sketch = data[1]

    for i in range(n_epochs):
        print(' ========== Epoch', i+1, '========== ')

        blue_photo, blue_sketch = shuffle(blue_photo, blue_sketch)

        for j in range(int(len(blue_photo)/n_batch)):

            start = int(j*n_batch)
            end = int(min(len(blue_photo), (j*n_batch)+n_batch))

            dataset = [load_images(blue_photo[start:end]), load_images(
blue_sketch[start:end])]

            # select a batch of real samples
            [X_realA, X_realB], y_real = generate_real_samples(dataset,
n_batch, n_patch)
            # print(type(X_realA))
            # print(type(X_realB))
            # print(type(y_real))
            # print(y_real)
            # X_realA = X_realA[numpy.logical_not(numpy.isnan(X_real
A))]
            # X_realB = X_realB[numpy.logical_not(numpy.isnan(X_real
B))]
            # y_real = y_real[numpy.logical_not(numpy.isnan(y_real))]
            # generate a batch of fake samples
            X_fakeB, y_fake = generate_fake_samples(g_model, X_realA, n
_patch)

            # update discriminator for real samples
            d_loss1 = d_model.train_on_batch([X_realA, X_realB], y_real
)

            # update discriminator for generated samples
            d_loss2 = d_model.train_on_batch([X_realA, X_fakeB], y_fake
)

            d_loss = 0.5 * np.add(d_loss1, d_loss2)
```

```
            # update the generator
            g_loss, _, _ = gan_model.train_on_batch(X_realA, [y_real, X
_realB])

            # summarize performance
            print('Batch : %d, D Loss : %.3f | G Loss : %.3f' % (j+1, d
_loss, g_loss))

        # summarize model performance
#         if (i+1) % 10 == 0:
        summarize_performance(i, g_model, d_model, dataset, target_dir)
```

Loss Function

In [55]:

```
import tensorflow as tf
import keras.backend as K
from keras.losses import mean_absolute_error

def pixel_loss(y_true, y_pred):
    return K.mean(K.abs(y_true - y_pred))

def contextual_loss (y_true, y_pred):
    a = tf.image.rgb_to_grayscale(tf.slice(
                                    y_pred,
                                    [0,0,0,0],
                                    [16, 256, 256, 3]))

    b = tf.image.rgb_to_grayscale(tf.slice(
                                    y_true,
                                    [0,0,0,0],
                                    [16, 256, 256, 3]))

    y_pred = tf.divide(tf.add(tf.reshape(a, [tf.shape(a)[0], -1]), 1),
2)
    y_true = tf.divide(tf.add(tf.reshape(b, [tf.shape(b)[0], -1]), 1),
2)
```

```
#     tf.assert_rank(y_true,2)
#     tf.assert_rank(y_pred,2)

    p_shape = tf.shape(y_true)
    q_shape = tf.shape(y_pred)
#     tf.assert_equal(p_shape, q_shape)

    # normalize sum to 1
    p_ = tf.divide(y_true, tf.tile(tf.expand_dims(tf.reduce_sum(y_true,
axis=1), 1), [1,p_shape[1]]))
    q_ = tf.divide(y_pred, tf.tile(tf.expand_dims(tf.reduce_sum(y_pred,
axis=1), 1), [1,p_shape[1]]))

    return tf.reduce_sum(tf.multiply(p_, tf.math.log(tf.divide(p_, q_
))), axis=1)

def total_loss (y_true, y_pred):

    px_loss = pixel_loss(y_true, y_pred)

    ctx_loss = contextual_loss(y_true, y_pred)

    return (0.2 * px_loss) + (0.8 * ctx_loss)
```

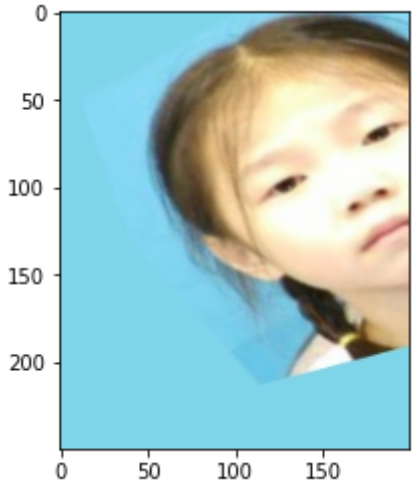
Load Dataset

In [56]:
```
# dataset path
b_photo_path = 'Dataset/Augmented photo/'
b_sketch_path = 'Dataset/Augmented sketch/'

blue_photo = load_filename(b_photo_path)
blue_sketch = load_filename(b_sketch_path)
```

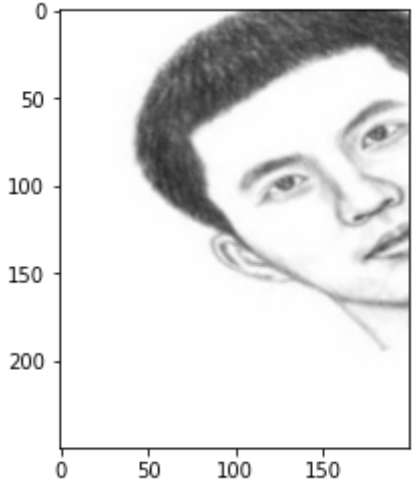
In [57]:
```
plt.imshow(cv2.cvtColor(cv2.imread(blue_photo[1102]).astype('uint8'), c
v2.COLOR_BGR2RGB))
```

Out[57]: <matplotlib.image.AxesImage at 0x7f052b617f90>



In [58]:
```
plt.imshow(cv2.cvtColor(cv2.imread(blue_sketch[1102]).astype('uint8'),
cv2.COLOR_BGR2RGB))
```

Out[58]: <matplotlib.image.AxesImage at 0x7f0527e490d0>



Define GAN Model

In [59]:
```
img_shape = (256, 256, 3)

d_model = discriminator(img_shape)

g_model = generator(img_shape)

gan_model = GAN(g_model, d_model, img_shape)
```

In [60]:
```
gan_model.summary()
```

```
Model: "GAN"
_________________________________________________________________
_________________________
Layer (type)                 Output Shape              Param #     Connec
ted to
=================================================================
=========================
input_8 (InputLayer)         [(None, 256, 256, 3) 0
_________________________________________________________________
_________________________
generator (Model)            (None, 256, 256, 3)  41825691    input_
8[0][0]
_________________________________________________________________
_________________________
discriminator (Model)        (None, 64, 64, 1)    539203      input_
8[0][0]
                                                               genera
tor[1][0]
=================================================================
=========================
Total params: 42,364,894
Trainable params: 41,825,691
Non-trainable params: 539,203
_________________________________________________________________
_________________________
```

In [61]:
```
opt = Adam(lr=2e-4, beta_1=0.5)

d_model.compile(loss='binary_crossentropy', optimizer=opt, loss_weights
=[0.5])
gan_model.compile(loss=['binary_crossentropy', total_loss], optimizer=o
pt, loss_weights=[1,100])
```

Start Training

In [ ]:
```
train(d_model, g_model, gan_model, [blue_sketch, blue_photo], 'Models/P
ixel[02]_Context[08]/', n_epochs = 100, n_batch=16)
```

```
 ========== Epoch 1 ==========
Batch : 1, D Loss : 3.363 | G Loss : 31.836
Batch : 2, D Loss : 3.334 | G Loss : 28.479
Batch : 3, D Loss : 3.355 | G Loss : 26.867
Batch : 4, D Loss : 3.421 | G Loss : 26.641
Batch : 5, D Loss : 3.356 | G Loss : 27.084
Batch : 6, D Loss : 3.217 | G Loss : 27.565
Batch : 7, D Loss : 3.238 | G Loss : 28.242
Batch : 8, D Loss : 3.384 | G Loss : 25.578
Batch : 9, D Loss : 3.317 | G Loss : 26.753
Batch : 10, D Loss : 3.355 | G Loss : 27.425
Batch : 11, D Loss : 3.522 | G Loss : 27.573
Batch : 12, D Loss : 3.357 | G Loss : 27.286
Batch : 13, D Loss : 3.350 | G Loss : 23.811
Batch : 14, D Loss : 3.369 | G Loss : 26.887
Batch : 15, D Loss : 3.433 | G Loss : 24.908
Batch : 16, D Loss : 3.343 | G Loss : 25.071
Batch : 17, D Loss : 3.359 | G Loss : 25.353
Batch : 18, D Loss : 3.413 | G Loss : 25.453
Batch : 19, D Loss : 3.435 | G Loss : 26.078
Batch : 20, D Loss : 3.422 | G Loss : 24.282
Batch : 21, D Loss : 3.418 | G Loss : 25.266
Batch : 22, D Loss : 3.459 | G Loss : 25.487
Batch : 23, D Loss : 3.473 | G Loss : 25.690
Batch : 24, D Loss : 3.443 | G Loss : 25.430
```

```
Batch : 25, D Loss : 3.220 | G Loss : 25.233
Batch : 26, D Loss : 3.469 | G Loss : 25.350
Batch : 27, D Loss : 3.465 | G Loss : 23.125
Batch : 28, D Loss : 3.474 | G Loss : 26.928
Batch : 29, D Loss : 3.443 | G Loss : 24.043
Batch : 30, D Loss : 3.426 | G Loss : 22.001
Batch : 31, D Loss : 3.359 | G Loss : 23.459
Batch : 32, D Loss : 3.356 | G Loss : 21.903
Batch : 33, D Loss : 3.498 | G Loss : 24.333
Batch : 34, D Loss : 3.430 | G Loss : 25.111
Batch : 35, D Loss : 3.331 | G Loss : 24.489
Batch : 36, D Loss : 3.377 | G Loss : 24.746
Batch : 37, D Loss : 3.469 | G Loss : 23.827
Batch : 38, D Loss : 3.250 | G Loss : 24.067
Batch : 39, D Loss : 3.372 | G Loss : 24.163
Batch : 40, D Loss : 3.440 | G Loss : 25.465
Batch : 41, D Loss : 3.578 | G Loss : 22.316
Batch : 42, D Loss : 3.432 | G Loss : 24.147
Batch : 43, D Loss : 3.465 | G Loss : 24.436
Batch : 44, D Loss : 3.407 | G Loss : 27.336
Batch : 45, D Loss : 3.468 | G Loss : 23.105
Batch : 46, D Loss : 3.376 | G Loss : 23.681
Batch : 47, D Loss : 3.432 | G Loss : 24.340
Batch : 48, D Loss : 3.373 | G Loss : 21.997
Batch : 49, D Loss : 3.386 | G Loss : 23.965
Batch : 50, D Loss : 3.392 | G Loss : 22.973
Batch : 51, D Loss : 3.314 | G Loss : 24.374
Batch : 52, D Loss : 3.373 | G Loss : 23.873
Batch : 53, D Loss : 3.309 | G Loss : 22.628
Batch : 54, D Loss : 3.433 | G Loss : 21.868
Batch : 55, D Loss : 3.374 | G Loss : 26.346
Batch : 56, D Loss : 3.422 | G Loss : 22.222
Batch : 57, D Loss : 3.369 | G Loss : 22.933
Batch : 58, D Loss : 3.374 | G Loss : 23.525
Batch : 59, D Loss : 3.444 | G Loss : 21.699
Batch : 60, D Loss : 3.384 | G Loss : 22.020
Batch : 61, D Loss : 3.320 | G Loss : 23.149
Batch : 62, D Loss : 3.455 | G Loss : 23.153
Batch : 63, D Loss : 3.405 | G Loss : 22.217
Batch : 64, D Loss : 3.421 | G Loss : 22.210
```

```
Batch : 65, D Loss : 3.372 | G Loss : 23.374
Batch : 66, D Loss : 3.283 | G Loss : 21.467
Batch : 67, D Loss : 3.372 | G Loss : 23.266
Batch : 68, D Loss : 3.420 | G Loss : 22.678
Batch : 69, D Loss : 3.327 | G Loss : 22.579
Batch : 70, D Loss : 3.398 | G Loss : 25.896
Batch : 71, D Loss : 3.358 | G Loss : 22.049
Batch : 72, D Loss : 3.436 | G Loss : 25.205
Batch : 73, D Loss : 3.403 | G Loss : 24.074
Batch : 74, D Loss : 3.390 | G Loss : 22.566
Batch : 75, D Loss : 3.387 | G Loss : 21.819
Batch : 76, D Loss : 3.344 | G Loss : 24.116
Batch : 77, D Loss : 3.379 | G Loss : 23.357
Batch : 78, D Loss : 3.433 | G Loss : 22.478
Batch : 79, D Loss : 3.391 | G Loss : 22.613
Batch : 80, D Loss : 3.339 | G Loss : 23.104
Batch : 81, D Loss : 3.456 | G Loss : 22.939
Batch : 82, D Loss : 3.389 | G Loss : 21.320
Batch : 83, D Loss : 3.498 | G Loss : 22.853
Batch : 84, D Loss : 3.407 | G Loss : 21.881
Batch : 85, D Loss : 3.372 | G Loss : 21.300
Batch : 86, D Loss : 3.431 | G Loss : 20.983
Batch : 87, D Loss : 3.427 | G Loss : 21.194
Batch : 88, D Loss : 3.568 | G Loss : 21.066
Batch : 89, D Loss : 3.352 | G Loss : 22.356
Batch : 90, D Loss : 3.455 | G Loss : 21.298
Batch : 91, D Loss : 3.348 | G Loss : 22.686
Batch : 92, D Loss : 3.386 | G Loss : 22.046
Batch : 93, D Loss : 3.471 | G Loss : 23.180
Batch : 94, D Loss : 3.432 | G Loss : 21.806
Batch : 95, D Loss : 3.437 | G Loss : 21.315
Batch : 96, D Loss : 3.321 | G Loss : 22.534
Batch : 97, D Loss : 3.324 | G Loss : 23.584
Batch : 98, D Loss : 3.423 | G Loss : 23.440
Batch : 99, D Loss : 3.346 | G Loss : 22.389
Batch : 100, D Loss : 3.318 | G Loss : 23.232
Batch : 101, D Loss : 3.333 | G Loss : 21.032
Batch : 102, D Loss : 3.316 | G Loss : 21.071
Batch : 103, D Loss : 3.506 | G Loss : 20.573
```