**Assessment 6**

***GitHub Link:*** https://github.com/manansuthar55/CSE6037_20MAI0016/tree/main/Assessment_6

# Image Colorization Using Autoencoders

In [1]:

```
import os
import sys
import random
import warnings
import numpy as np
import pandas as pd
import cv2
import matplotlib.pyplot as plt
from tqdm import tqdm
from itertools import chain
import skimage
from PIL import Image
from skimage.io import imread, imshow, imread_collection, concatenate_images
from skimage.transform import resize
from skimage.util import crop, pad
from skimage.morphology import label
from skimage.color import rgb2gray, gray2rgb, rgb2lab, lab2rgb
from sklearn.model_selection import train_test_split
from keras.applications.inception_resnet_v2 import InceptionResNetV2, preprocess_input
from keras.models import Model, load_model,Sequential
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Input, Dense, UpSampling2D, RepeatVector, Reshape
from keras.layers.core import Dropout, Lambda
from keras.layers.convolutional import Conv2D, Conv2DTranspose
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate
from keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from keras import backend as K
import tensorflow as tf

warnings.filterwarnings('ignore', category=UserWarning, module='skimage')
seed = 42
random.seed = seed
np.random.seed = seed
```

```
Using TensorFlow backend.
```

# Read in Data

**The dataset I am using is ~2000 classic paintings which we will remove the color from and attempt to teach a nearal network to recolorize them.**

In [2]:

```
IMG_WIDTH = 256
IMG_HEIGHT = 256
IMG_CHANNELS = 3
INPUT_SHAPE=(IMG_HEIGHT, IMG_WIDTH, 1)
TRAIN_PATH = '../input/art-images-drawings-painting-sculpture-engraving/dataset/dataset_updated/training_set/painting/'
train_ids = next(os.walk(TRAIN_PATH))[2]
```

In [3]:

```
%%time
```

```
X_train = np.zeros((len(train_ids)-86, IMG_HEIGHT, IMG_WIDTH, IMG_CHANNELS), dtype=np.ui
nt8)
missing_count = 0
print('Getting train images ... ')
sys.stdout.flush()
for n, id_ in tqdm(enumerate(train_ids), total=len(train_ids)):
    path = TRAIN_PATH + id_+''
    try:
        img = imread(path)
        img = resize(img, (IMG_HEIGHT, IMG_WIDTH), mode='constant', preserve_range=True)
        X_train[n-missing_count] = img
    except:
        missing_count += 1
X_train = X_train.astype('float32') / 255.
print("Total missing: "+ str(missing_count))
```

```
Getting train images ...
```

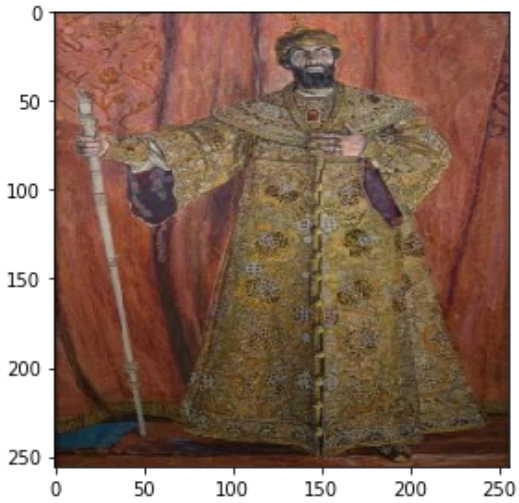
```
100%|██████████| 2128/2128 [00:39<00:00, 54.10it/s]
```

```
Total missing: 86
CPU times: user 44.8 s, sys: 25.1 s, total: 1min 9s
Wall time: 40.4 s
```

In [4]:

```
imshow(X_train[5])
plt.show()
```

```
/opt/conda/lib/python3.6/site-packages/skimage/io/_plugins/matplotlib_plugin.py:51: Futur
eWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` i
s deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  out_of_range_float = (np.issubdtype(image.dtype, np.float) and
/opt/conda/lib/python3.6/site-packages/matplotlib/axes/_base.py:1428: MatplotlibDeprecati
onWarning: The 'box-forced' keyword argument is deprecated since 2.2.
  " since 2.2.", cbook.mplDeprecation)
```




In [5]:

```
X_train, X_test = train_test_split(X_train, test_size=20, random_state=seed)
```

# Create the Model

In [6]:

```
inception = InceptionResNetV2(weights=None, include_top=True)
inception.load_weights('../input/inception-resnet-v2-weights/inception_resnet_v2_weights_
tf_dim_ordering_tf_kernels.h5')
inception.graph = tf.get_default_graph()
```

In [7]:

```
def Colorize():
```

```
    embed_input = Input(shape=(1000,))
    #Encoder
    encoder_input = Input(shape=(256, 256, 1,))
    encoder_output = Conv2D(128, (3,3), activation='relu', padding='same',strides=1)(enc
oder_input)
    encoder_output = MaxPooling2D((2, 2), padding='same')(encoder_output)
    encoder_output = Conv2D(128, (4,4), activation='relu', padding='same')(encoder_outpu
t)
    encoder_output = Conv2D(128, (3,3), activation='relu', padding='same',strides=1)(enc
oder_output)
    encoder_output = MaxPooling2D((2, 2), padding='same')(encoder_output)
    encoder_output = Conv2D(256, (4,4), activation='relu', padding='same')(encoder_outpu
t)
    encoder_output = Conv2D(256, (3,3), activation='relu', padding='same',strides=1)(enc
oder_output)
    encoder_output = MaxPooling2D((2, 2), padding='same')(encoder_output)
    encoder_output = Conv2D(256, (4,4), activation='relu', padding='same')(encoder_outpu
t)
    encoder_output = Conv2D(256, (3,3), activation='relu', padding='same')(encoder_outpu
t)
    encoder_output = Conv2D(256, (3,3), activation='relu', padding='same')(encoder_outpu
t)
    #Fusion
    fusion_output = RepeatVector(32 * 32)(embed_input)
    fusion_output = Reshape(([32, 32, 1000]))(fusion_output)
    fusion_output = concatenate([encoder_output, fusion_output], axis=3)
    fusion_output = Conv2D(256, (1, 1), activation='relu', padding='same')(fusion_output
)
    #Decoder
    decoder_output = Conv2D(128, (3,3), activation='relu', padding='same')(fusion_output
)
    decoder_output = Conv2D(64, (3,3), activation='relu', padding='same')(decoder_output
)
    decoder_output = UpSampling2D((2, 2))(decoder_output)
    decoder_output = Conv2D(128, (3,3), activation='relu', padding='same')(decoder_outpu
t)
    decoder_output = UpSampling2D((2, 2))(decoder_output)
    decoder_output = Conv2D(64, (4,4), activation='relu', padding='same')(decoder_output
)
    decoder_output = Conv2D(64, (3,3), activation='relu', padding='same')(decoder_output
)
    decoder_output = Conv2D(32, (2,2), activation='relu', padding='same')(decoder_output
)
    decoder_output = Conv2D(2, (3, 3), activation='tanh', padding='same')(decoder_output
)
    decoder_output = UpSampling2D((2, 2))(decoder_output)
    return Model(inputs=[encoder_input, embed_input], outputs=decoder_output)

model = Colorize()
model.compile(optimizer='adam', loss='mean_squared_error')
model.summary()
```

```
_________________________________________________________________________________________
_________
Layer (type)                 Output Shape              Param #     Connected to
=========================================================================================
=========
input_3 (InputLayer)         (None, 256, 256, 1)   0

_________________________________________________________________________________________
_________
conv2d_204 (Conv2D)          (None, 256, 256, 128 1280        input_3[0][0]

_________________________________________________________________________________________
_________
max_pooling2d_5 (MaxPooling2D)  (None, 128, 128, 128 0           conv2d_204[0][0]

_________________________________________________________________________________________
_________
conv2d_205 (Conv2D)          (None, 128, 128, 128 262272      max_pooling2d_5[0][0]
```

| | | | |
|---|---|---|---|
| conv2d_206 (Conv2D) | (None, 128, 128, 128 | 147584 | conv2d_205[0][0] |
| max_pooling2d_6 (MaxPooling2D) | (None, 64, 64, 128) | 0 | conv2d_206[0][0] |
| conv2d_207 (Conv2D) | (None, 64, 64, 256) | 524544 | max_pooling2d_6[0][0] |
| conv2d_208 (Conv2D) | (None, 64, 64, 256) | 590080 | conv2d_207[0][0] |
| max_pooling2d_7 (MaxPooling2D) | (None, 32, 32, 256) | 0 | conv2d_208[0][0] |
| conv2d_209 (Conv2D) | (None, 32, 32, 256) | 1048832 | max_pooling2d_7[0][0] |
| input_2 (InputLayer) | (None, 1000) | 0 | |
| conv2d_210 (Conv2D) | (None, 32, 32, 256) | 590080 | conv2d_209[0][0] |
| repeat_vector_1 (RepeatVector) | (None, 1024, 1000) | 0 | input_2[0][0] |
| conv2d_211 (Conv2D) | (None, 32, 32, 256) | 590080 | conv2d_210[0][0] |
| reshape_1 (Reshape) | (None, 32, 32, 1000) | 0 | repeat_vector_1[0][0] |
| concatenate_1 (Concatenate) | (None, 32, 32, 1256) | 0 | conv2d_211[0][0]<br>reshape_1[0][0] |
| conv2d_212 (Conv2D) | (None, 32, 32, 256) | 321792 | concatenate_1[0][0] |
| conv2d_213 (Conv2D) | (None, 32, 32, 128) | 295040 | conv2d_212[0][0] |
| conv2d_214 (Conv2D) | (None, 32, 32, 64) | 73792 | conv2d_213[0][0] |
| up_sampling2d_1 (UpSampling2D) | (None, 64, 64, 64) | 0 | conv2d_214[0][0] |
| conv2d_215 (Conv2D) | (None, 64, 64, 128) | 73856 | up_sampling2d_1[0][0] |

```
__________
up_sampling2d_2 (UpSampling2D)  (None, 128, 128, 128 0           conv2d_215[0][0]

_____________________________________________________________________________________________
__________
conv2d_216 (Conv2D)             (None, 128, 128, 64) 131136      up_sampling2d_2[0][0]

_____________________________________________________________________________________________
__________
conv2d_217 (Conv2D)             (None, 128, 128, 64) 36928       conv2d_216[0][0]

_____________________________________________________________________________________________
__________
conv2d_218 (Conv2D)             (None, 128, 128, 32) 8224        conv2d_217[0][0]

_____________________________________________________________________________________________
__________
conv2d_219 (Conv2D)             (None, 128, 128, 2)  578         conv2d_218[0][0]

_____________________________________________________________________________________________
__________
up_sampling2d_3 (UpSampling2D)  (None, 256, 256, 2)  0           conv2d_219[0][0]

=============================================================================================
=========
Total params: 4,696,098
Trainable params: 4,696,098
Non-trainable params: 0
_____________________________________________________________________________________________
__________
```

# Data Generator Functions

In [8]:

```
%%time
# Image transformer
datagen = ImageDataGenerator(
        shear_range=0.2,
        zoom_range=0.2,
        rotation_range=20,
        horizontal_flip=True)
#Create embedding
def create_inception_embedding(grayscaled_rgb):
    def resize_gray(x):
        return resize(x, (299, 299, 3), mode='constant')
    grayscaled_rgb_resized = np.array([resize_gray(x) for x in grayscaled_rgb])
    grayscaled_rgb_resized = preprocess_input(grayscaled_rgb_resized)
    with inception.graph.as_default():
        embed = inception.predict(grayscaled_rgb_resized)
    return embed
#Generate training data
def image_a_b_gen(dataset=X_train, batch_size = 20):
    for batch in datagen.flow(dataset, batch_size=batch_size):
        X_batch = rgb2gray(batch)
        grayscaled_rgb = gray2rgb(X_batch)
        lab_batch = rgb2lab(batch)
        X_batch = lab_batch[:,:,:,0]
        X_batch = X_batch.reshape(X_batch.shape+(1,))
        Y_batch = lab_batch[:,:,:,1:] / 128
        yield [X_batch, create_inception_embedding(grayscaled_rgb)], Y_batch
```

```
CPU times: user 235 µs, sys: 0 ns, total: 235 µs
Wall time: 239 µs
```

In [9]:

```
# Set a learning rate annealer
learning_rate_reduction = ReduceLROnPlateau(monitor='loss',
                                            patience=3,
```

```
                                            verbose=1,
                                            factor=0.5,
                                            min_lr=0.00001)
filepath = "Art_Colorization_Model.h5"
checkpoint = ModelCheckpoint(filepath,
                             save_best_only=True,
                             monitor='loss',
                             mode='min')
model_callbacks = [learning_rate_reduction,checkpoint]
```

# Train the Model

In [10]:

```
%%time
BATCH_SIZE = 20
model.fit_generator(image_a_b_gen(X_train,BATCH_SIZE),
            epochs=30,
            verbose=1,
            steps_per_epoch=X_train.shape[0]/BATCH_SIZE,
            callbacks=model_callbacks
                  )
```

```
Epoch 1/30
102/101 [==============================] - 174s 2s/step - loss: 0.0107
Epoch 2/30
102/101 [==============================] - 159s 2s/step - loss: 0.0051
Epoch 3/30
102/101 [==============================] - 164s 2s/step - loss: 0.0048
Epoch 4/30
102/101 [==============================] - 164s 2s/step - loss: 0.0047
Epoch 5/30
102/101 [==============================] - 164s 2s/step - loss: 0.0046
Epoch 6/30
102/101 [==============================] - 166s 2s/step - loss: 0.0047
Epoch 7/30
102/101 [==============================] - 163s 2s/step - loss: 0.0045
Epoch 8/30
102/101 [==============================] - 164s 2s/step - loss: 0.0044
Epoch 9/30
102/101 [==============================] - 164s 2s/step - loss: 0.0043
Epoch 10/30
102/101 [==============================] - 163s 2s/step - loss: 0.0042
Epoch 11/30
102/101 [==============================] - 162s 2s/step - loss: 0.0042
Epoch 12/30
102/101 [==============================] - 162s 2s/step - loss: 0.0042
Epoch 13/30
102/101 [==============================] - 161s 2s/step - loss: 0.0041
Epoch 14/30
102/101 [==============================] - 163s 2s/step - loss: 0.0041
Epoch 15/30
102/101 [==============================] - 162s 2s/step - loss: 0.0040
Epoch 16/30
102/101 [==============================] - 162s 2s/step - loss: 0.0040
Epoch 17/30
102/101 [==============================] - 162s 2s/step - loss: 0.0040
Epoch 18/30
102/101 [==============================] - 162s 2s/step - loss: 0.0039
Epoch 19/30
102/101 [==============================] - 162s 2s/step - loss: 0.0039
Epoch 20/30
102/101 [==============================] - 162s 2s/step - loss: 0.0039
Epoch 21/30
102/101 [==============================] - 161s 2s/step - loss: 0.0039

Epoch 00021: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.
Epoch 22/30
102/101 [==============================] - 161s 2s/step - loss: 0.0037
Epoch 23/30
```

```
102/101 [==============================] - 162s 2s/step - loss: 0.0036
Epoch 24/30
102/101 [==============================] - 162s 2s/step - loss: 0.0036
Epoch 25/30
102/101 [==============================] - 162s 2s/step - loss: 0.0036
Epoch 26/30
102/101 [==============================] - 162s 2s/step - loss: 0.0035
Epoch 27/30
102/101 [==============================] - 164s 2s/step - loss: 0.0035
Epoch 28/30
102/101 [==============================] - 163s 2s/step - loss: 0.0034
Epoch 29/30
102/101 [==============================] - 162s 2s/step - loss: 0.0033
Epoch 30/30
102/101 [==============================] - 163s 2s/step - loss: 0.0035

Epoch 00030: ReduceLROnPlateau reducing learning rate to 0.0002500000118743628.
CPU times: user 1h 36min 6s, sys: 15min 53s, total: 1h 51min 59s
Wall time: 1h 21min 42s
```

In [11]:

```
model.save(filepath)
model.save_weights("Art_Colorization_Weights.h5")
```

# Sample the Results

In [12]:

```
sample = X_test
color_me = gray2rgb(rgb2gray(sample))
color_me_embed = create_inception_embedding(color_me)
color_me = rgb2lab(color_me)[:,:,:,0]
color_me = color_me.reshape(color_me.shape+(1,))
output = model.predict([color_me, color_me_embed])
output = output * 128
decoded_imgs = np.zeros((len(output),256, 256, 3))
for i in range(len(output)):
    cur = np.zeros((256, 256, 3))
    cur[:,:,0] = color_me[i][:,:,0]
    cur[:,:,1:] = output[i]
    decoded_imgs[i] = lab2rgb(cur)
    cv2.imwrite("img_"+str(i)+".jpg", lab2rgb(cur))
```

In [13]:

```
plt.figure(figsize=(20, 6))
for i in range(10):
    # grayscale
    plt.subplot(3, 10, i + 1)
    plt.imshow(rgb2gray(X_test)[i].reshape(256, 256))
    plt.gray()
    plt.axis('off')
    # recolorization
    plt.subplot(3, 10, i + 1 +10)
    plt.imshow(decoded_imgs[i].reshape(256, 256,3))
    plt.axis('off')
    # original
    plt.subplot(3, 10, i + 1 + 20)
    plt.imshow(X_test[i].reshape(256, 256,3))
    plt.axis('off')
plt.tight_layout()
plt.show()
```