

Find a "style transfer" Python code based on CNN architectures. You can find other codes aside from the sample. You can select one of the two photos below:



You may choose two from the list of styles:

Vincent Van Gogh

Ukiyo-e

Pablo Picasso

Bencab

Paul Gauguin

Fernando Amorsolo

Leonardo da Vinci

Juan Luna

Our task is to create a style transfer algorithm that will merge the style of two images is one image.

This is the snippet of the code that I used in accomplishing the task for this assignment.

```
In [2]: import os
import tensorflow as tf
# Load compressed models from tensorflow_hub
os.environ['TFHUB_MODEL_LOAD_FORMAT'] = 'COMPRESSED'
```

```
In [4]: import IPython.display as display

import matplotlib.pyplot as plt
import matplotlib as mpl
mpl.rcParams['figure.figsize'] = (12, 12)
mpl.rcParams['axes.grid'] = False

import numpy as np
import PIL.Image
import time
import functools
```

```
In [5]: def tensor_to_image(tensor):
    tensor = tensor*255
    tensor = np.array(tensor, dtype=np.uint8)
    if np.ndim(tensor)>3:
        assert tensor.shape[0] == 1
        tensor = tensor[0]
    return PIL.Image.fromarray(tensor)
```

```
In [32]: content2_path = "Technological_Institute_of_the_Philippines_Quezon_City.jpg"
style_path = "/Users/asus/Desktop/The Starry Night.jpg"
style2_path = "/Users/asus/Desktop/Minotauromachy.jpg"
```

```
In [33]: def load_img(path_to_img):
max_dim = 512
img = tf.io.read_file(path_to_img)
img = tf.image.decode_image(img, channels=3)
img = tf.image.convert_image_dtype(img, tf.float32)

shape = tf.cast(tf.shape(img)[:1], tf.float32)
long_dim = max(shape)
scale = max_dim / long_dim

new_shape = tf.cast(shape * scale, tf.int32)

img = tf.image.resize(img, new_shape)
img = img[tf.newaxis, :]
return img
```

```
In [34]: def show_image(image, title=None):
if len(image.shape)>3:
    image=tf.squeeze(image,axis=0)
plt.imshow(image)
if title:
    plt.title=title
```

```
In [35]: def imshow(image, title=None):
if len(image.shape) > 3:
    image = tf.squeeze(image, axis=0)

plt.imshow(image)
if title:
    plt.title(title)
```

```
In [36]: content_image = load_img(content2_path)
style_image = load_img(style_path)

plt.subplot(1, 2, 1)
imshow(content_image, 'Content Image')

plt.subplot(1, 2, 2)
imshow(style_image, 'Style Image')
```

```
In [37]: content2_image = load_img(content2_path)
style2_image = load_img(style2_path)

plt.subplot(1, 2, 1)
imshow(content2_image, 'Content Image')

plt.subplot(1, 2, 2)
imshow(style2_image, 'Style Image')
```

```
In [41]: import tensorflow_hub as hub
hub_model = hub.load('https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2')
stylized_image = hub_model(tf.constant(content_image), tf.constant(style_image))[0]
tensor_to_image(stylized_image)
```

```
In [42]: import tensorflow_hub as hub
hub_model = hub.load('https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2')
stylized2_image = hub_model(tf.constant(content2_image), tf.constant(style2_image))[0]
tensor_to_image(stylized2_image)
```

```
In [44]: x = tf.keras.applications.vgg19.preprocess_input(content_image*255)
x = tf.image.resize(x, (224, 224))
vgg = tf.keras.applications.VGG19(include_top=True, weights='imagenet')
prediction_probabilities = vgg(x)
prediction_probabilities.shape
```

A local file was found, but it seems to be incomplete or outdated because the auto file hash does not match the original value of cbe5617147190e668d6c5d5026f83318 so we will re-download the data.
 Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weights_tf_dim_ordering_tf_kernels.h5
 574710816/574710816 [=====] - 601s 1us/step

Out[44]: TensorShape([1, 1000])

```
In [45]: predicted_top_5 = tf.keras.applications.vgg19.decode_predictions(prediction_probabilities.numpy())[0]
[(class_name, prob) for (number, class_name, prob) in predicted_top_5]
```

Downloading data from https://storage.googleapis.com/download.tensorflow.org/data/imagenet_class_index.json
 35363/35363 [=====] - 0s 2us/step

```
Out[45]: [('dock', 0.10787788),
('dam', 0.09949343),
('pier', 0.07090867),
('garbage_truck', 0.046317216),
('crane', 0.038242985)]
```

```
In [46]: vgg = tf.keras.applications.VGG19(include_top=False, weights='imagenet')

print()
for layer in vgg.layers:
    print(layer.name)
```

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5
 80134624/80134624 [=====] - 71s 1us/step

```
input_3
block1_conv1
block1_conv2
block1_pool
block2_conv1
block2_conv2
block2_pool
block3_conv1
block3_conv2
...
```

```
In [47]: content_layers = ['block5_conv2']
content2_layers = ['block5_conv2']

style_layers = ['block1_conv1',
               'block2_conv1',
               'block3_conv1',
               'block4_conv1',
               'block5_conv1']

style2_layers = ['block1_conv1',
                'block2_conv1',
                'block3_conv1',
                'block4_conv1',
                'block5_conv1']

num_content_layers = len(content_layers)
num_style_layers = len(style_layers)

num_content2_layers = len(content2_layers)
num_style2_layers = len(style2_layers)
```

Upon completing the task, the code has three output images.

The first one is the style transfer between the base photo and the artwork of Vincent Van Gogh “The Starry Night”.



- The combination of the two is not quite nice, because the artwork didn't highlight the corners of the building in the base image. Also, Van Gogh style in The Starry Night is full of circular structure so that it doesn't compliment the base photo.

The second output is the style transfer between the base photo and Pablo Picasso's artwork “Minatauromachy”.



- The artwork compliments the base photo very nicely, because the artwork has sharp edges that highlight the base photo's building corners and the black and white motif is very nice.

Lastly, the last output is the style transfer between the first and second output.



- My thoughts about this output are that the algorithm combine the first two output and because of that the base photo became more detailed than the first two outputs.

To conclude this assignment, I feel accomplished after doing this because I perfectly simulate this program even though my laptop is crashing at first. The one thing that challenge me is the creating that path for the images because I only know one way to path a file in the code.

Reference for the Style Images:



https://en.wikipedia.org/wiki/The_Starry_Night



<https://www.wikiart.org/en/pablo-picasso/the-minotauromachie-1935>