IMAGE STYLE TRANSFER USING MACHINE LEARNING

Dr Bagam Laxmaiah [1], Jonnadula Narasimharao [2], Abhigna Maringanti [3],

SriTeja Govindula [4], Suresh Dharamsoth [5]

[1]Associate Professor, CMR Technical Campus, Hyderabad,Telangana,India,

[2]Associate Professor, CMR Technical Campus, Hyderabad,Telangana,India

[3]U.G Student, CMR Technical Campus, Hyderabad,Telangana,India

[4]U.G Student, CMR Technical Campus, Hyderabad, Telangana,India

[5]U.G Student, CMR Technical Campus, Hyderabad, Telangana,India

Abstract

The principle of Image style transfer is todefine two distance functions, one that describes the content image and the other that describes the style Image.By using these content and Style Images[5][6] as inputs we will be getting the desired output which has the content image merged with style image.The ouput will be in the graphical model of the content image.

In summary, we'll take the base input image, a contentimage that we want to match, the style image that we want to match by undergoing the process of convolutional neural network [7] firstly the content image is undergoing the process of content loss and style image as style loss after content loss and style lossit will undergo the process of gram matrix and the final image will be formed.

Keywords: *Content Image, Style Image, Content Loss, Style Loss, Convolutional Neural Networks, Gram Matrix Deep Lab Semantic Segmentation.*

1. Introduction

Dealing With the Image Formation,The Image style transfer is an approach to convert one image form into another image form by using convolutional neural network. In this we have to take two images 1) content- image, 2)style-image. The content-image is nothing but it is a main basic image like a dog or human being etc,which exist in real world. The style image is a graphical image like tree graphical image and etc..which is in 3d fromat.By using Some machine learning functionalities and modules like we need to make content loss and style loss.After these we will be getting the desired output image.

2. Literature Survey

In this project,we have taken a content image and thestyle image we will save the content image[6] and thestyle images[5] from any website and we need to use them by their location in the system.Here we use manymachine learning modules,Algorithms.Convolutional neural network algorithm[7] is used here for formation of the images.OpenCv[8] ,TensorFlow [9] are the functionalities used to process the images.By using these we can get output image in the merged form of both content image and the style image.We need to calculate content loss and the style loss Then undergoes some modules in machine learning like deep lab semantic segmentation[10],open cv[8],tensor flow[9] making the images forming the output desired image.Content loss[11] is nothing but the difference ofcontent of the content image to the content of the output image is the content loss and the content loss iscalculated by Euclidean distance and the content loss can be calculated with the content image only.

The content loss is: content of content image –content of output image.
The Euclidean distance is:

$$d(x,\ y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

The Style loss[12] is defined as the difference between the output image and the style image.Whichinturn gives the style loss. Here we cannot compare aintermediate features of style loss and content loss so, that is why we introduce gram matrix

In the gram matrix[13] we have to equal the both pixel sizes. Let us consider a content image pixel size is [3280*4250] and the style image pixel size is [4290*5690] so the both pixel sizes is different by using the gram matrix we have to equal the both pixel sizes of content image and style image, and we have to calculate style image by using gram matrix. Ingram matrix we have to calculate each layer style information the formulae is:

$$L_{style}(a, x) = \sum_{l \in L} w_l E_l$$

3. Proposed System

We have proposed Image style transfer using machine learning in this there are two images 1) content-image 2) style-image so, we have to take these two images as an input image to training modelHere in proposed system we will create a data sets ofimages for target Image. We give an input as any image and using this data sets the image is convertedto our target image. Firstly the content image is givenand then it undergoes deeplab semantic segmentation which makes the content image as the mask of the content image then the formed mask of the content image and the style image goes image style transfer which in undergoes machine learning modules like opencv and tensorflow.And finally the desired merged form of the image is formed.The below Image shows the Architecture of the image style transfer using machine learning.

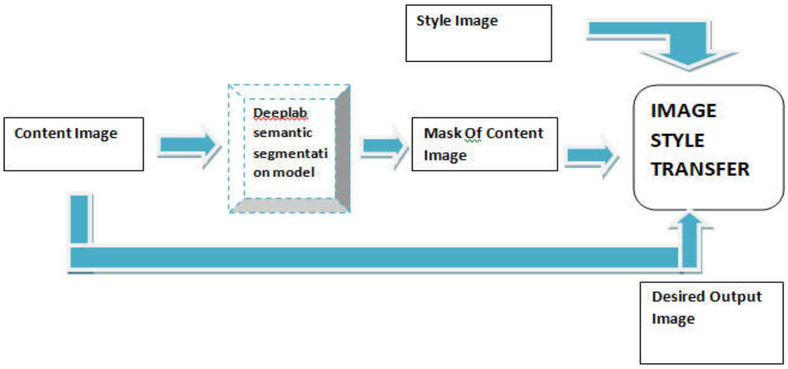
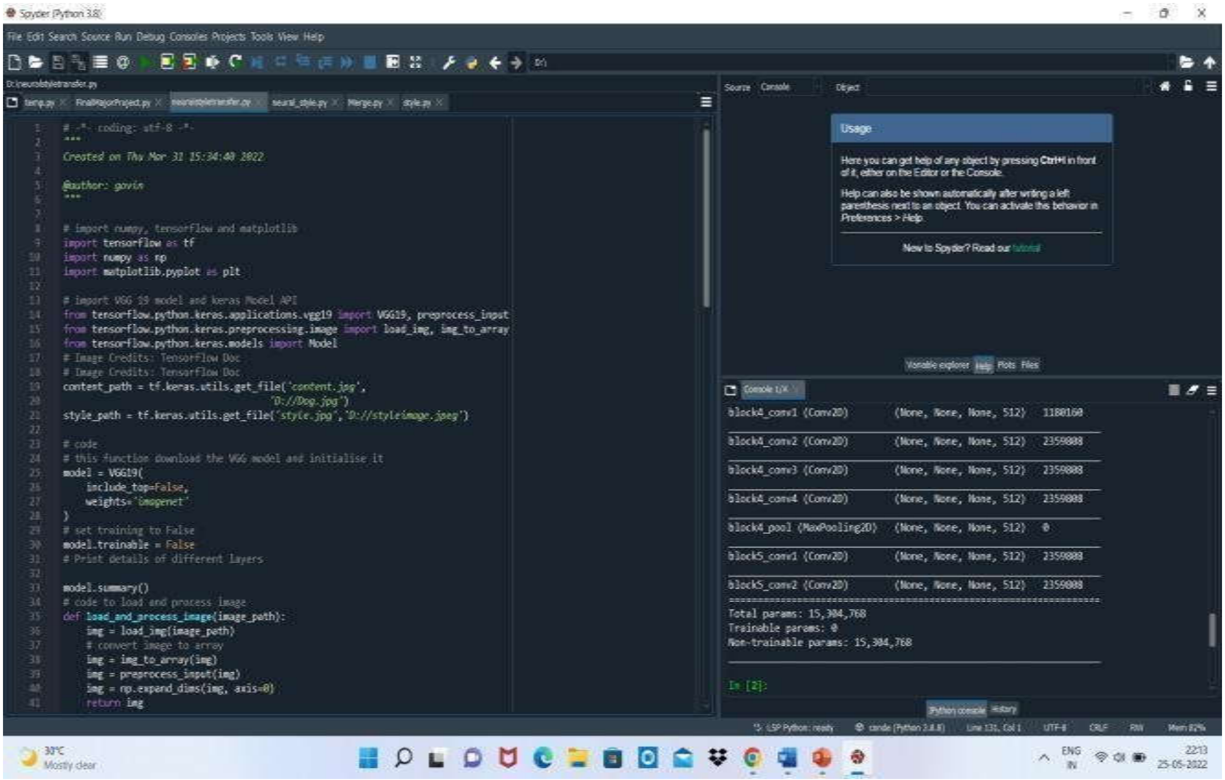


Figure 3.1: Architecture of the Model.

4.Result

Here for a sample we took a dog as a content image and the graphical image as the Input. We need to give input as the location name in the code i.e,where the image is saved in the system.So before the process we need to save the input images which are content image and the style image in the files on the system.While giving input we need togive the location of the image.


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\neuralstyletransfer.py
temp.py FinalMajorProject.py neuralstyletransfer.py neural_style.py Merge.py style.py
Source Console Object
Variable explorer Help Plots Files
Console 1/A
Python console History
LSP Python: ready conda (Python 3.8.8) Line 131, Col 1 UTF-8 CRLF RW Mem 82%
30°C Mostly clear
ENG IN 22:13 25-05-2022

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 31 15:34:40 2022

@author: govin
"""

# import numpy, tensorflow and matplotlib
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

# import VGG 19 model and keras Model API
from tensorflow.python.keras.applications.vgg19 import VGG19, preprocess_input
from tensorflow.python.keras.preprocessing.image import load_img, img_to_array
from tensorflow.python.keras.models import Model
# Image Credits: Tensorflow Doc
# Image Credits: Tensorflow Doc
content_path = tf.keras.utils.get_file('content.jpg',
                                       'D://Dog.jpg')
style_path = tf.keras.utils.get_file('style.jpg','D://styleimage.jpeg')

# code
# this function download the VGG model and initialise it
model = VGG19(
    include_top=False,
    weights='imagenet'
)
# set training to False
model.trainable = False
# Print details of different layers

model.summary()
# code to load and process image
def load_and_process_image(image_path):
    img = load_img(image_path)
    # convert image to array
    img = img_to_array(img)
    img = preprocess_input(img)
    img = np.expand_dims(img, axis=0)
    return img
```

Usage

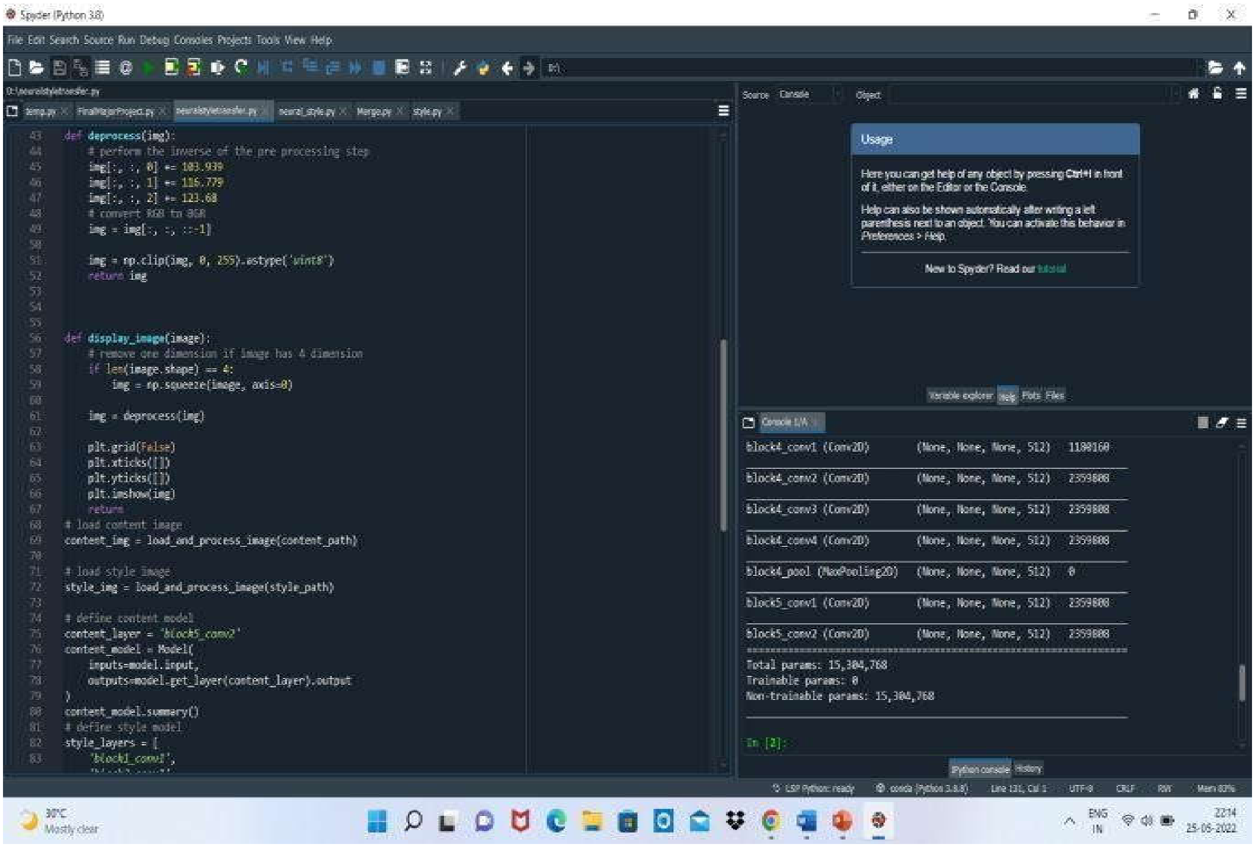
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial

```
block4_conv1 (Conv2D)        (None, None, None, 512)   1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, None, None, 512)   0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, None, None, 512)   2359808
=================================================================
Total params: 15,304,768
Trainable params: 0
Non-trainable params: 15,304,768
_________________________________________________________________

In [2]:
```



```python
def deprocess(img):
    # perform the inverse of the pre processing step
    img[:, :, 0] += 103.939
    img[:, :, 1] += 116.779
    img[:, :, 2] += 123.68
    # convert RGB to BGR
    img = img[:, :, ::-1]

    img = np.clip(img, 0, 255).astype('uint8')
    return img



def display_image(image):
    # remove one dimension if image has 4 dimension
    if len(image.shape) == 4:
        img = np.squeeze(image, axis=0)

    img = deprocess(img)

    plt.grid(False)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(img)
    return
# load content image
content_img = load_and_process_image(content_path)

# load style image
style_img = load_and_process_image(style_path)

# define content model
content_layer = 'block5_conv2'
content_model = Model(
    inputs=model.input,
    outputs=model.get_layer(content_layer).output
)
content_model.summary()
# define style model
style_layers = [
    'block1_conv1',
```

Usage

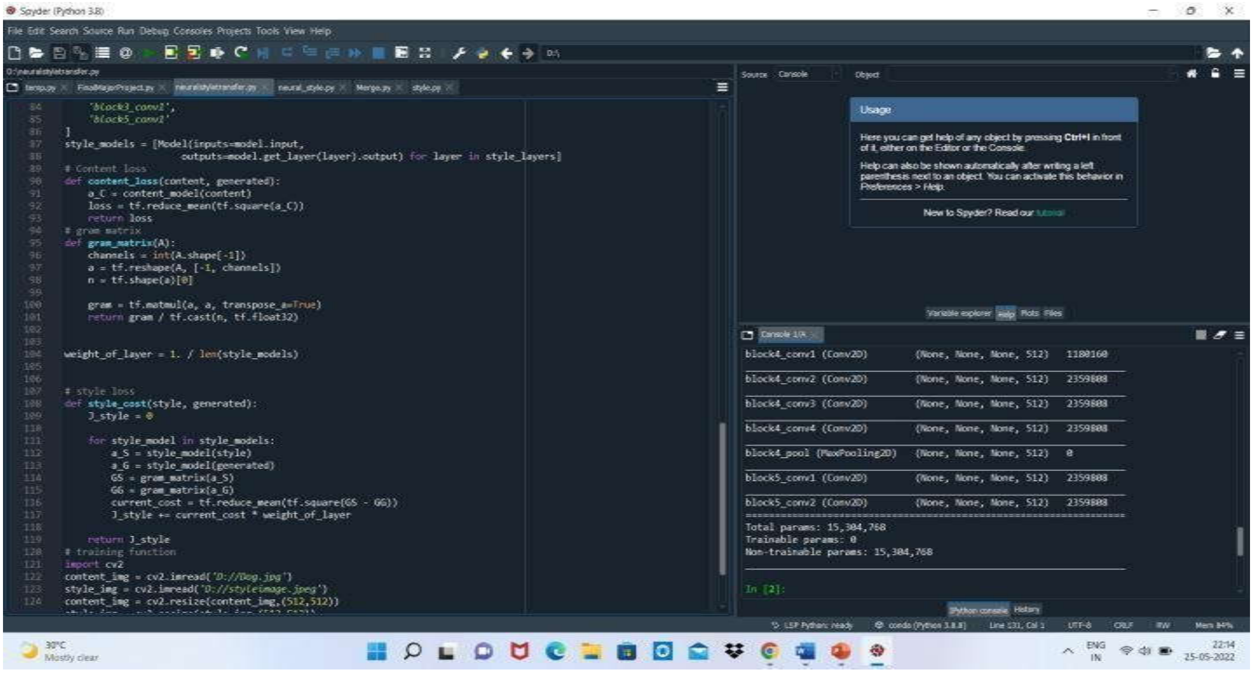
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial

```
block4_conv1 (Conv2D)        (None, None, None, 512)   1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, None, None, 512)   0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, None, None, 512)   2359808
=================================================================
Total params: 15,304,768
Trainable params: 0
Non-trainable params: 15,304,768
_________________________________________________________________

In [2]:
```



```python
    'block3_conv2',
    'block5_conv2'
]
style_models = [Model(inputs=model.input,
                      outputs=model.get_layer(layer).output) for layer in style_layers]
# Content loss
def content_loss(content, generated):
    a_C = content_model(content)
    loss = tf.reduce_mean(tf.square(a_C))
    return loss
# gram matrix
def gram_matrix(A):
    channels = int(A.shape[-1])
    a = tf.reshape(A, [-1, channels])
    n = tf.shape(a)[0]

    gram = tf.matmul(a, a, transpose_a=True)
    return gram / tf.cast(n, tf.float32)


weight_of_layer = 1. / len(style_models)


# style loss
def style_cost(style, generated):
    J_style = 0

    for style_model in style_models:
        a_S = style_model(style)
        a_G = style_model(generated)
        GS = gram_matrix(a_S)
        GG = gram_matrix(a_G)
        current_cost = tf.reduce_mean(tf.square(GS - GG))
        J_style += current_cost * weight_of_layer

    return J_style
# training function
import cv2
content_img = cv2.imread('D://Dog.jpg')
style_img = cv2.imread('D://styleimage.jpeg')
content_img = cv2.resize(content_img,(512,512))
```

Usage

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

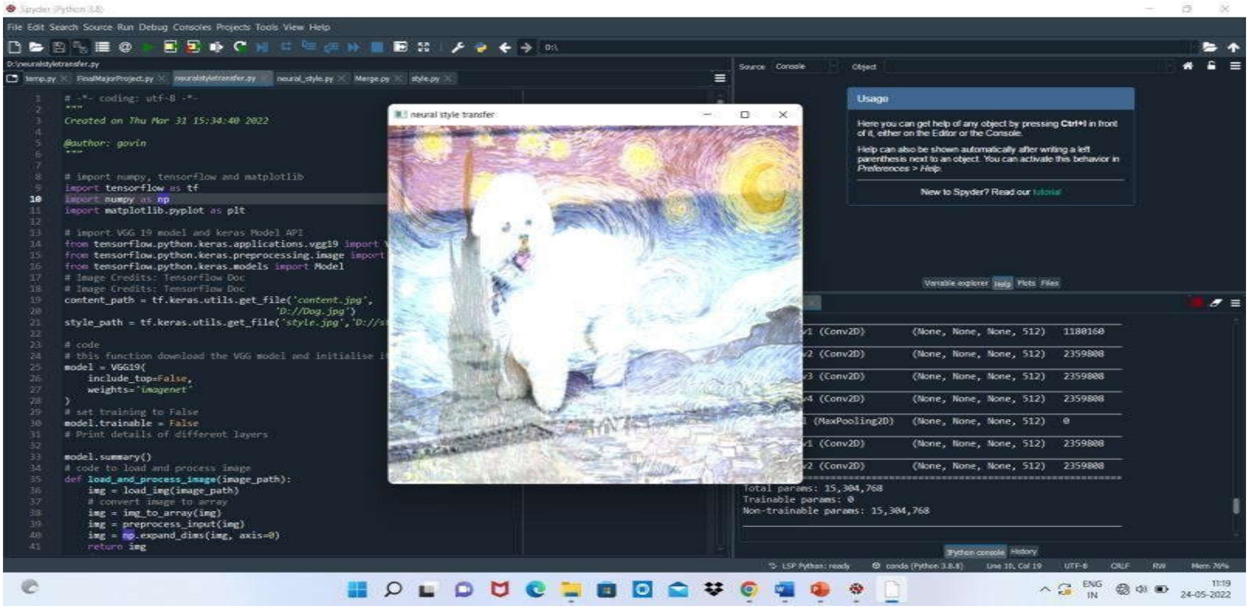
New to Spyder? Read our tutorial

```
block4_conv1 (Conv2D)        (None, None, None, 512)   1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, None, None, 512)   0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, None, None, 512)   2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, None, None, 512)   2359808
=================================================================
Total params: 15,304,768
Trainable params: 0
Non-trainable params: 15,304,768
_________________________________________________________________

In [2]:
```

Figure 4.1-4.3-Code Execution

Then the images undergo processing firstly the content image undergo DeepLab Semantic Segmentation and we will be getting mask of the content image and then we will be giving style image this style image and the mask of the content image undergoes opencv and tensorflow functions. Tensorflow is used for mathematical computations and opencv is used for image processing. Finally we will get the output image as follows



```python
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 31 15:34:40 2022

@author: govin
"""

# import numpy, tensorflow and matplotlib
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

# import VGG 19 model and keras Model API
from tensorflow.python.keras.applications.vgg19 import V
from tensorflow.python.keras.preprocessing.image import
from tensorflow.python.keras.models import Model
# Image Credits: Tensorflow Doc
# Image Credits: Tensorflow Doc
content_path = tf.keras.utils.get_file('content.jpg',
                                        'D://Dog.jpg')
style_path = tf.keras.utils.get_file('style.jpg','D://s

# code
# this function download the VGG model and initialise i
model = VGG19(
    include_top=False,
    weights='imagenet'
)
# set training to False
model.trainable = False
# Print details of different layers

model.summary()
# code to load and process image
def load_and_process_image(image_path):
    img = load_img(image_path)
    # convert image to array
    img = img_to_array(img)
    img = preprocess_input(img)
    img = np.expand_dims(img, axis=0)
    return img
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
v1 (Conv2D)          (None, None, None, 512)   1180160
_____________________________________________________
v2 (Conv2D)          (None, None, None, 512)   2359808
_____________________________________________________
v3 (Conv2D)          (None, None, None, 512)   2359808
_____________________________________________________
v4 (Conv2D)          (None, None, None, 512)   2359808
_____________________________________________________
1 (MaxPooling2D)     (None, None, None, 512)   0
_____________________________________________________
v1 (Conv2D)          (None, None, None, 512)   2359808
_____________________________________________________
v2 (Conv2D)          (None, None, None, 512)   2359808
=====================================================
Total params: 15,304,768
Trainable params: 0
Non-trainable params: 15,304,768
_____________________________________________________
```

Figure 4.4:Final Result Image

5.Conclusion

By using this principle of Image Style Transfer using Machine Learning we can get our desired image without recreation by using an algorithm called CNN,We use some of the modules of machine learning like opencv and tensorflow and we will getthe image with high Accuracy and it takes less time because there is no recreation of the images,only conversion of images directly.

6.References

[1] N.Ashikhmin.Fasttexturetransfer.IEEEComputerGraph- ics and Applications, 23(4):38–43, July2003.

[2] M. Berning, K. M. Boergens, and M. Helmstaedter.SegEM: Efficient Image Analysis for High-Resolution Connec-tomics. Neuron, 87(6):1193–1206, Sept. 2015.

[3] C.F.Cadieu,H.Hong,D.L.K.Yamins,N.Pinto,D.Ardila,

E. A. Solomon, N. J. Majaj, and J. J. DiCarlo. Deep Neu-

[ralNetworksRivaltheRepresentationofPrimateITCor- tex for Core

Visual Object Recognition. PLoSComputBiol, 10(12):e1003963, Dec. 2014.

[4] L.-C. Chen, G. Papandreou, I.Kokkinos, K.Murphy, and A. L. Yuille. Semantic Image Segmentationwith Deep Convolutional Nets and Fully Connected CRFs. arXiv:1412.7062 [cs], Dec. 2014. arXiv: 1412.7062.

[5] Different Style images from the web.

[6] Differentcontent images fromtheweb

[7] CNN-Convolutional neural Network Algorithm https://machinelearningmastery.com/convolutional-layers-for-deep-learning- neural-networks/

[8] Opencv Module in Machine Learning https://pypi.org/project/opencv-python/

[9] Tensorflowmodule in machinelearning https://www.tensorflow.org/hub/api_docs/python/hub/Module

[10] Deep Lab Sematic Segmentation forcontent Image https://towardsdatascience.com/the-evolution-of-deeplab-for-semantic- segmentation-95082b025571

[11] Content loss and the style loss for theimages. https://nnart.org/what-is-content-loss/ https://www.coursera.org/lecture/generative- deep-learning-with-tensorflow/style-loss-l9bTa

[12] Image processing Process https://towardsdatascience.com/convolution-neural- network-for-image- processing-using-keras-dc3429056306

[13] Calculation of gram matrix https://towardsdatascience.com/neural-networks-intuitions-2-dot-product-gram- matrix-and-neural-style-transfer-5d39653e7916

[14] Keras in python https://www.javatpoint.com/keras