

A

Major Project Report

On

# **IMAGE STYLE TRANSFER USING MACHINE LEARNING**

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

**Maringanti Abhigna (187R1A0594)**

**Govindula Sriteja (187R1A05A5)**

**Dharamsoth Suresh (187R1A0576)**

Under the Guidance of

**Dr. Laxmaiah**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGC Act. 1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

**2018-22**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



## CERTIFICATE

This is to certify that the project entitled “**Image Style Transfer Using Machine Learning**” being submitted by **Maringanti Abhigna (187R1A0594), Govindula Sriteja (187R1A05A5), Dharamsoth Suresh (187R1A0576)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**Dr. B. Laxmaiah**  
(Associate Professor)  
INTERNAL GUIDE

**Dr. A. RAJI REDDY**  
DIRECTOR

**Dr. K. SRUJAN RAJU**  
HOD

**EXTERNAL EXAMINER**

Submitted for viva voice Examination held on \_\_\_\_\_

## ACKNOWLEDGMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Dr. B. Laxmaiah**, Associate Professor for her exemplary guidance, monitoring and constant encouragement through out the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. A. Uday Kiran, Mr. J. Narasimha Rao, Mrs. G. Latha, Dr. T. S. Mastan Rao, Mr. A. Kiran Kumar**, for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy**, Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy**, Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

**Maringanti Abhigna** (187R1A0594)

**Govindula Sriteja** (187R1A05A5)

**Dharamsoth Suresh** (187R1A0576)

## **ABSTRACT**

The principle of Image style transfer is to define two distance functions, one that describes the content image and the other that describes the style Image. By using these content and Style Images as inputs we will be getting the desired output which has the content image merged with style image. The output will be in the graphical model of the content image. The output we get here will be having more accuracy and it is an easy process which takes less time to convert the image.

In summary, we'll take the base input image, a content image that we want to match, and the style image that we want to match. After Transforming using image style transfer and some machine Learning Modules to get the desired image.

## LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 2.1	Example for a content Image	4
Figure 2.2	Example for a Style Image	5
Figure 2.3	Data Preparation Process	7
Figure 2.4	Sample for Model Building	8
Figure 2.5	Sample Example for Image Style Transfer using Machine Learning	9
Figure 3.1	Architecture For Image Style Transfer using Machine Learning	12
Figure 3.2	Style Loss	17
Figure 3.3	Gram Matrix	18
Figure 3.4	Foreground and Background for Deep Lab Semantic Segmentation	20
Figure 3.5	Example for Deep Lab Semantic Segmentation	21
Figure 3.6	Example for Deep Lab Semantic Segmentation	22

## LIST OF FIGURES

Figure 3.7	Use Case Diagram For Image Style Transfer using Machine Learning	23
Figure 3.8	Class diagram for For Image Style Transfer using Machine Learning	24
Figure 3.9	Sequence Diagram For Image Style Transfer using Machine Learning	25
Figure 3.10	Activity Diagram For Image Style Transfer using Machine Learning	26
Figure 5.1	Content Image	31
Figure 5.2	Style Image	31

## LIST OF SCREENSHOTS

<b>SCREENSHOT NO.</b>	<b>SCREENSHOT NAME</b>	<b>PAGE NO</b>
Screenshot 5.1-5.2	Code Execution	32
Screenshot 5.3	Result of the Final Image	34

# TABLE OF CONTENTS

	PAGE NO
<b>ABSTRACT</b>	i
<b>LIST OF FIGURES</b>	ii
<b>LIST OF SCREENSHOT</b>	iii
<b>1. INTRODUCTION</b>	
1.1 PROJECT SCOPE	1
1.2 PROJECT PURPOSE	1
1.3 PROJECT FEATURES	1
<b>2. SYSTEM ANALYSIS</b>	
2.1 INTRODUCTION	2
2.2 PROBLEM STATEMENT	2
2.3 EXISTING SYSTEM	6
2.3.1 LIMITATIONS OF EXISTING SYSTEM	6
2.4 PROPOSED SYSTEM	7
2.4.1 ADVANTAGES OF PROPOSED SYSTEM	8
2.5 FEASIBILITY STUDY	10
2.5.1 ECONOMIC FESIBILITY	10
2.5.2 TECHNICAL FEASIBILITY	10
2.5.3 SOCIAL FEASIBILITY	10
2.6 HARDWARE & SOFTWARE REQUIREMENTS	11
2.6.1 HARDWARE REQUIREMENTS	11
2.6.2 SOFTWARE REQUIREMENTS	11
<b>3. ARCHITECTURE</b>	
3.1 PROJECT ARCHITECTURE	12
3.2 DESCRIPTION	20
3.3 USECASE DIAGRAM	23
3.4 CLASS DIAGRAM	24
3.5 SEQUENCE DIAGRAM	25
3.6 ACTIVITY DIAGRAM	26



<b>4. IMPLEMENTATION</b>	
4.1 SAMPLE CODE	27
<b>5. SCREENSHOT</b>	
5.1 CODE EXECUTION PART-1	32
5.2 CODE EXECUTION PART-2	33
5.3 RESULT OF FINAL IMAGE	34
<b>6. TESTING</b>	
6.1 INTRODUCTION TO TESTING	35
6.2 TYPES OF TESTING	35
6.2.1 UNIT TESTING	35
6.2.2 INTEGRATION TESTING	35
6.2.3 FUNCTIONAL TESTING	36
<b>7. CONCLUSION &amp; FUTURE SCOPE</b>	
7.1 PROJECT CONCLUSION	37
7.2 FUTURE SCOPE	37
<b>8. BIBLIOGRAPHY</b>	
8.1 REFERENCE	38
8.2 WEBSITES	38
8.3 GITHUB LINK	39

# **1.INTRODUCTION**

## **1.INTRODUCTION**

### **1.1 PROJECT PURPOSE**

This project is titled as “Image Style Transfer using Machine Learning”. This Provides an easy method to convert the images into desired output image which will be in the merged form.This Merged Form will be formed from content image and the style Image.

### **1.2 PROJECT PURPOSE**

This has been developed to facilitate the process of creating a image in the 3d format or in the form of merged images.As we use two images which are content image and the style image which in turn converts into the final desired image using few machine learning modules.This reduces the process time without Re-Creating the image.

### **1.3 PROJECT FEATURES**

The main feature of this project is forming an image which is the merged form or the 3d structured form of the image using two main images like the content image and the style image..We have an existing system of Re-Creating the image which we need directly and in this process of recreation it takes much time to form the image .

## **2.SYSTEM ANALYSIS**

## 2. SYSTEM ANALYSIS

### 2.1 INTRODUCTION

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

### 2.2 PROBLEM STATEMENT

The Problem Statement is we need to convert the input images which are content image and the style image into the merged form of the image which will look as the 3d structured image. Basically for creating an image we need to draw it or recreate it using photoshops etc, This recreation takes much time as we need to create every part of the image. There is also no much accuracy for this recreation process which already exists, Because in creation of the image we may get mistakes in the size, shape, structure etc. So using this project of Image Style Transferring we will get the image as we required.

We have taken a content image and the style image we will save the content image and the style images from any website and we need to use them by their location in the system. Here we use many machine learning modules, Algorithms. Convolutional neural network algorithm is used here for formation of the images. Open Cv, Tensor Flow 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, open cv, tensor flow making the images forming the output desired image. Content loss is nothing but the difference of content

of the content image to the content of the output image is the content loss and the content loss is calculated by Euclidean distance and the content loss can be calculated with the content image only.

content loss = 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 is defined as the difference between the output image and the style image. Which in turn gives the style loss. Here we cannot compare a intermediate features of style loss and content loss so, that is why we introduce gram matrix. In the gram matrix 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. In gram matrix we have to calculate each layer style information

The formulae is:

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

**Content Image:**

It is described as real world object or image. It may be a person or any physically existing thing.



Figure 2.1 Example for a content image

**Style Image:**

It is in the form of a 3d picture.



Figure 2.2 Example for a style image



## **2.3 EXISTING SYSTEM**

The project is based on transferring the image of one distinct model to another model. As this project is not there directly, we have to do transfer the image only by recreating it like in photoshops like paint, etc. This process of converting the image using recreation takes much time also.

So, exactly perfect model cannot be designed in this existing system. There is no much accuracy for this existing model.

### **2.3.1 LIMITATIONS OF EXISTING SYSTEM**

- 1) Time consumption is very high
- 2) There is a chance of mistakes in the image (like size difference, model difference).
- 3) No much Accuracy.

## 2.4 PROPOSED SYSTEM

To avoid the problem of our existing 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 model. We use some machine learning modules which helps in conversion of the two images. We give Style image and content image as an input and using this data sets the image is converted to our target image. Here we will use Deep Lab Semantic Segmentation to transfer the content image into the shape of the image. Here Deep lab semantic segmentation is the process of converting the content image into its mask of the content image. Here we will have 3 modules namely, Data Preparation, Model Building and Accuracy

- **DATA PREPARATION**

In this module we have to take two images, content-image and style-image and we have to load this two images into the training model. So here the data is getting prepared. The data we use here are content image and style image. So these images are set for the process.

$$\text{content image} + \text{style image} = \text{output}$$



Figure 2.3 Data Preparation process

- **MODEL BUILDING**

In this model building module we have to train the model. After training the model the content image and style image undergo transferring using “Opencv”, “Tensor flow” modules. After this the resultant output which is in a merged format will be formed.

```
import tensorflow as tf
from matplotlib import pyplot as plt
import numpy as np
import cv2
```

Figure 2.4 Sample for Model Building

- **ESTIMATING ACCURACY OF THE MODEL**

In the previous module we have train the model to get the output, in this module we can check how accuracy of the application producing an output. If the output has not much accuracy we need to do the process using backpropagation algorithm we need to train the model to produce best accuracy model.

#### **2.4.1 ADVANTAGES OF PROPOSED SYSTEM**

1. Process is very easy.
2. It takes less time to form the final image.
3. Accuracy is more.

These are some of the sample images in which we can see the content image, style image and the output image

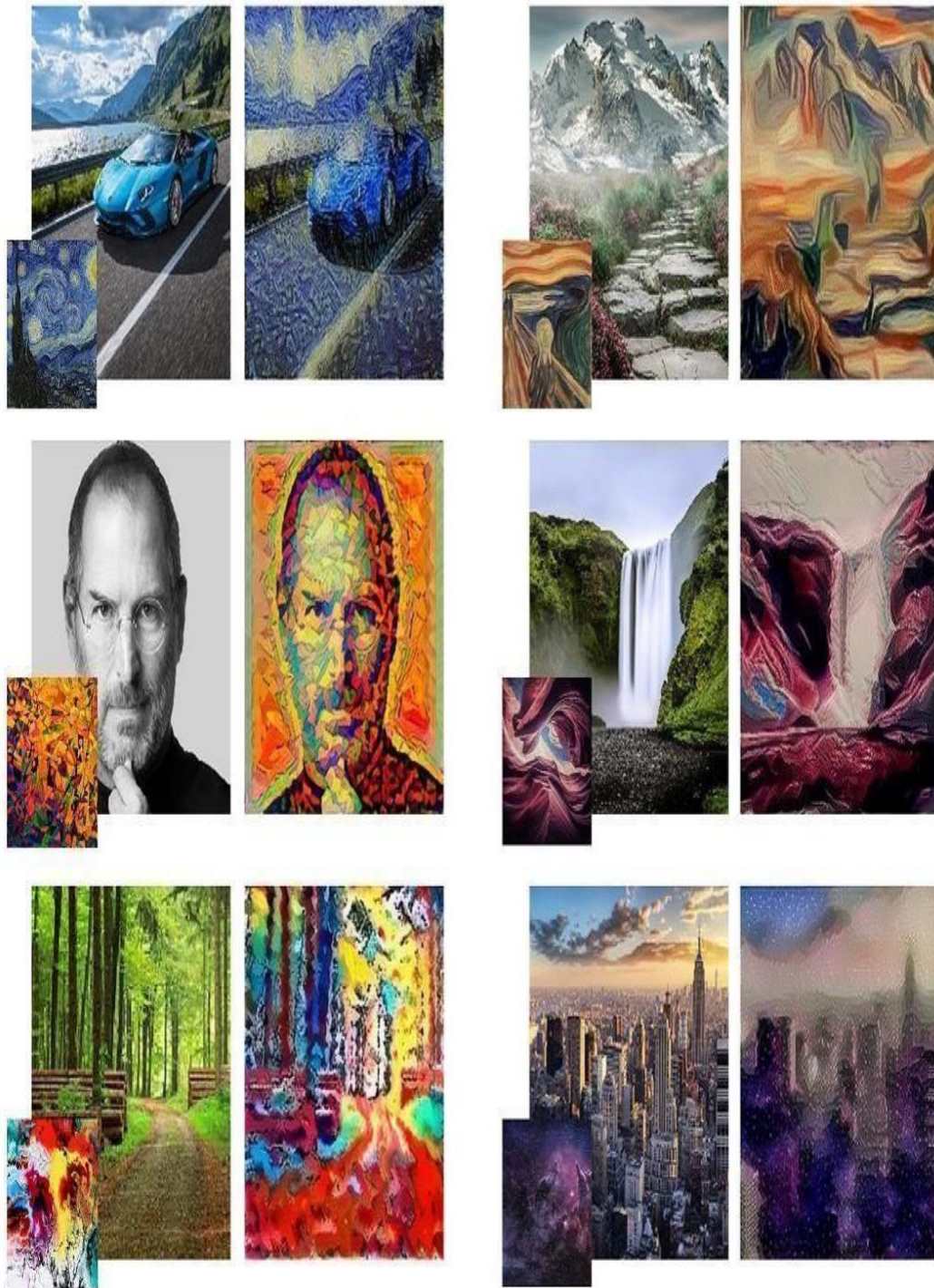


Figure 2.5 Sample example of images for Image Style Transfer Using Machine Learning

## 2.5 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis are

Economic Feasibility

Technical Feasibility

Social Feasibility

### 2.5.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### 2.5.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### 2.5.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## 2.6 HARDWARE & SOFTWARE REQUIREMENTS

### 2.6.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

Processor	:	Intelp2 or later
Hard disk	:	40 GB Or More
RAM	:	512 MB Or More
Input devices	:	Keyboard, monitor.

### 2.6.2 SOFTWARE REQUIREMENTS:

Software Requirements are the requirements which we use as a platform and the operating system on which we are working and the tools which we use. These all comes under software requirements.

The following are some software requirements.

Technologies	:	Python, Machine Learning.
Software Tools	:	Spyder(Idle)
Operating System	:	Windows 10

# **3.ARCHITECTURE**

### 3.ARCHITECTURE

#### 3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for Image Style Transfer Using Machine Learning. In this project Architecture describes the process of the project that means it shows every step of the project in the form of an image.

Here in this Image style Transfer using machine learning firstly, the content image which we take will undergo deep lab semantic segmentation which undergoes the process and mask of the image is formed this mask of the content image and the style image both undergo image transfer will then forms the output final image we required. So Firstly we need to import content image and the style image. Content image undergoes semantic segmemntation as shown and then the process of merging the images takes place. Below is the Architecture Diagram for Image Style Transfer Using Machine Learning.

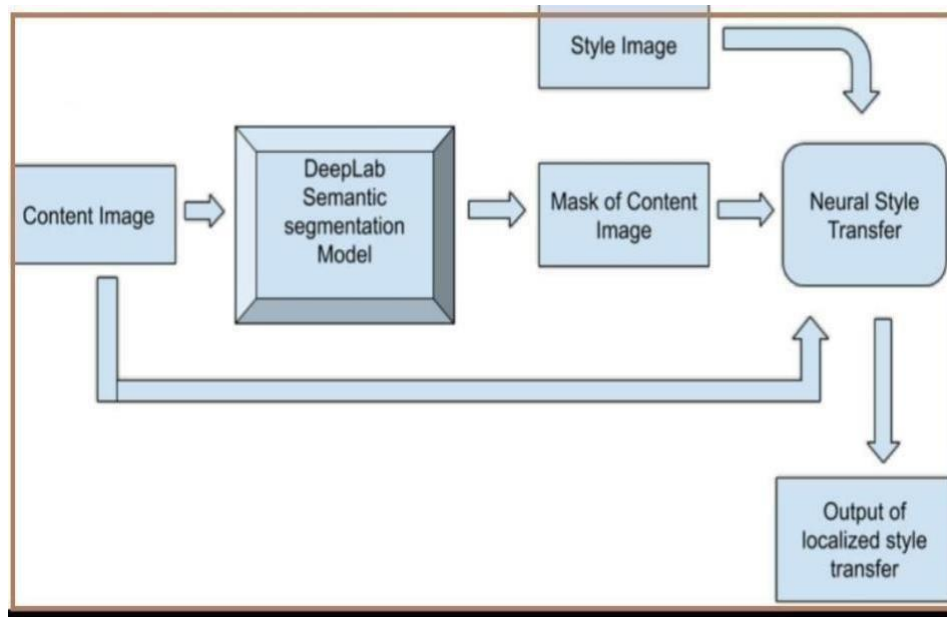


Figure 3.1 Architecture for Image Style Transfer Using machine learning.



- **INTERMEDIATE LAYERS OF STYLE AND CONTENT**

So why do these intermediate outputs within our pretrained image classification network allow us to define style and content representations? at a high level, in order for a network to perform image classification (which this network has been trained to do), it must understand the image. This requires taking the raw image as input pixels and building an internal representation that converts the raw image pixels into a complex understanding of the features present within the image.

This is also a reason why convolutional neural networks are able to generalize well: they're able to capture the invariances and defining features within classes (e.g. cats vs. dogs) that are agnostic to background noise and other nuisances. Thus, somewhere between where the raw image is fed into the model and the output classification label, the model serves as a complex feature extractor. By accessing intermediate layers of the model, you're able to describe the content and style of input images.

- **BUILD THE MODEL**

The networks in [tf.keras.applications](https://keras.io/applications/) are designed so you can easily extract the intermediate layer values using the Keras functional API.

To define a model using the functional API, specify the inputs and outputs:

```
model = Model(inputs, outputs)
```

The following function builds a VGG19 model that returns a list of intermediate layer outputs:

```
def vgg_layers(layer_names):
    vgg = tf.keras.applications.VGG19(include_top=False,
                                     weights='imagenet')
    vgg.trainable = False
```

```

outputs = [vgg.get_layer(name).output for
name in layer_names]
model = tf.keras.Model([vgg.input], outputs)
return model

```

- **CALCULATE STYLE**

The content of an image is represented by the values of the intermediate feature maps. It turns out, the style of an image can be described by the means and correlations across the different feature maps. Calculate a Gram matrix that includes this information by taking the outer product of the feature vector with itself at each location, and averaging that outer product over all locations. This Gram matrix can be calculated.

- **RUN GRADIENT DESCENT**

With this style and content extractor, you can now implement the style transfer algorithm. Do this by calculating the mean square error for your image's output relative to each target, then take the weighted sum of these losses.

- **TOTAL VARIATION LOSS**

One downside to this basic implementation is that it produces a lot of high frequency artifacts. Decrease these using an explicit regularization term on the high frequency components of the image. In style transfer, this is often called the total variation loss.

- **DEFINING FUNCTIONS TO BUILD THE STYLE TRANSFER NETWORK**

Here you define several functions that will help you later to fully define the computational graph of the CNN given an input.

**CREATING TENSORFLOW VARIABLES:**

1. content image (`tf.placeholder`)
2. style image (`tf.placeholder`)
3. generated image (`tf.Variable` and `trainable=True`)
4. pretrained weights and biases (`tf.Variable` and `trainable=False`)

- **COMPUTING THE VGG NET OUTPUT**

Here you are computing the VGG net output by means of convolution and pooling operations. Note that you are replacing the `tf.nn.max_pool` with `tf.nn.avg_pool` operation, as `tf.nn.avg_pool` gives better visually pleasing results during style transfer. Feel free to experiment with `tf.nn.max_pool` by changing the operation in the function below.

- **LOSS FUNCTIONS**

In this section we define two loss functions; the *content loss* function and the *style loss* function. The content loss function ensures that the activations of the higher layers are similar between the content image and the generated image. The style loss function makes sure that the correlation of activations in *all* the layers are similar between the style image and the generated image.

- **CONTENT COST FUNCTION**

The content cost function is making sure that the content present in the content image is captured in the generated image. It has been found that CNNs capture information about content in the higher levels, where the lower levels are more focused on individual pixel values. Therefore we

use the top-most CNN layer to define the content loss function. Let,

$A^l_{ij}(I)$  be the activation of the  $l$  th layer,  $i$  th feature map and  $j$  th position obtained using the image  $I$ .

$$L_{content} = \frac{1}{2} \sum_{i,j} (A^l_{ij}(g) - A^l_{ij}(c))^2$$

Essentially  $L_{content}$  captures the root mean squared error between the activations produced by the generated image and the content image. But why does minimising the difference between the activations of higher layers ensure the content of the content image is preserved.

- **INTUITION BEHIND CONTENT LOSS**

If you visualise what is learnt by a neural network, there's [evidence](#) that suggests that different feature maps in higher layers are activated in the presence of different objects. So if two images to have the same content, they should have similar activations in the higher layers.

- **STYLE LOSS FUNCTION**

Defining the style loss function requires more work. To extract the style information from the VGG network, we use all the layers of the CNN. Furthermore, style information is measured as the amount of correlation present between feature maps in a given layer. Next, a loss is defined as the difference of correlation present between the feature

- **INTUATION BEHIND THE STYLE LOSS**

Though the above equation system is a mouthful, the idea is relatively simple. The goal is to compute a style matrix (visualised below) for the generated image and the style image. Then the style loss is defined as the root mean square difference between the two style

Below you can see an illustration of how the style matrix is computed. The style matrix is essentially a Gram matrix, where the  $(i,j)$ th element of the style matrix is computed by computing the element wise multiplication of the  $i$ th and  $j$ th feature maps and summing across both width and height. In the figure, red cross denotes element wise Multiplication and the red plus sign denotes summing across both width height of the feature maps.

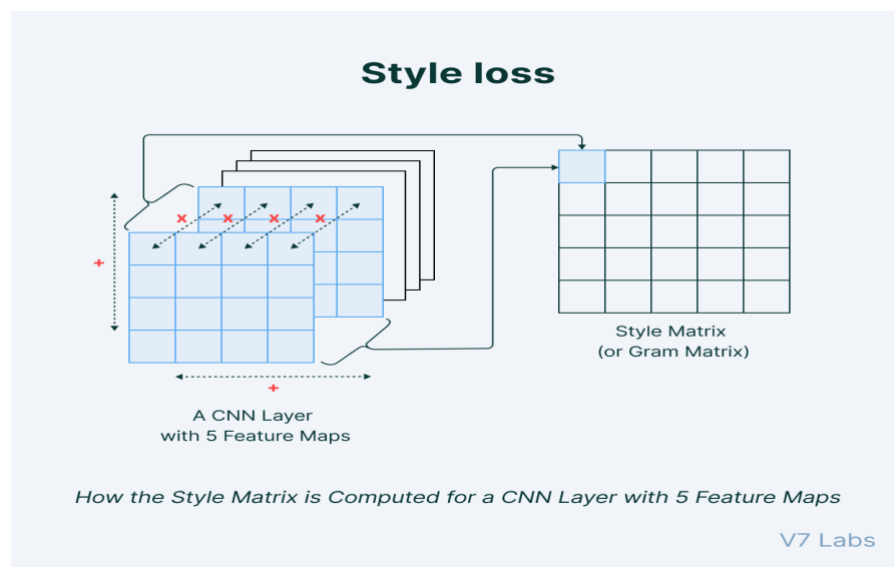


Figure 3.2 Style Loss

- **GRAM MATRIX**

It's great that we know how to compute the style loss. But you still haven't been shown "why the style loss is computed using the Gram matrix". The Gram matrix essentially captures the "distribution of features" of a set of feature maps in a given layer. By trying to minimize the style loss between two images, you are essentially matching the distribution of features between the two images [3, 4].

So let me take a shot at explaining this a bit more intuitively. Say you have the following feature maps For Simplicity I Assume only 3

feature maps, and two of them are completely inactive. You have one feature map set where the first feature map looks like a dog, and in the second feature map set, the first feature map looks like a dog upside down. Then if you try to manually compute content and style losses, you will get these values. This means that we haven't lost style information between two feature map sets. However, the content is quite different.

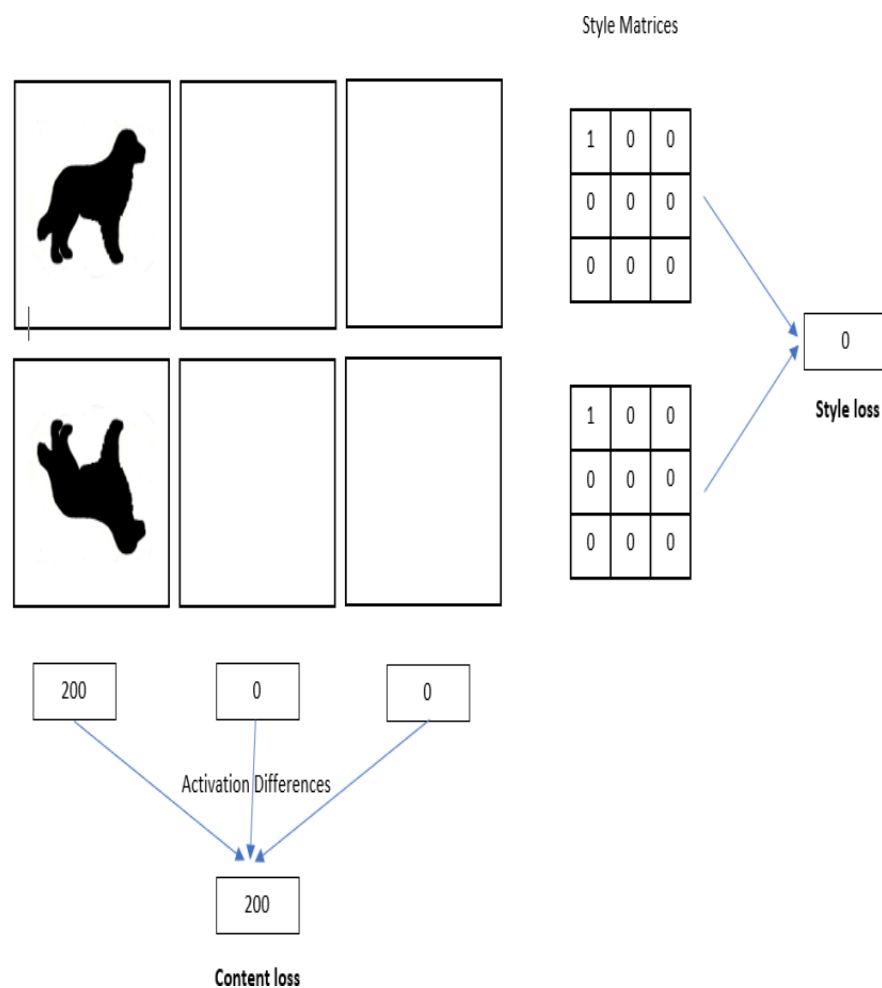


Figure 3.3 Gram Matrix

- **FINAL LOSS**

The final loss is defined as,

$$L = \alpha L_{content} + \beta L_{style} ,$$

where  $\alpha$  and  $\beta$  are user-defined hyper parameters. Here  $\beta$  has absorbed the  $M^l$  normalisation factor defined earlier. By controlling  $\alpha$  and  $\beta$  you can control the amount of content and style injected to the generated image. You can also see a nice visualization of different effects of different  $\alpha$  and  $\beta$  values in the paper.

- **DEFINING THE INPUT PIPELINE**

Here you define the full input pipeline, `tf.data` provides a very easy to use and intuitive interface to implementing input pipelines. For most of the image manipulation tasks you can use the `tf.image` API, however the ability of `tf.image` to handle dynamic sized images is very limited. For example, if you want to dynamically crop and resize images it is better to do in the form of a generator as implemented below.

You have defined two input pipelines; one for content and one for style. The content input pipeline looks for `jpg` images that start with the word `content`, where the style pipeline looks for images starting with `style`.

### 3.2 DESCRIPTION

Here we will be taking the content image and this content image undergoes deep lab semantic segmentation and then mask of the content image is formed and then this mask of the content image and the style image undergoes through some modules of machine learning like open cv and tensor flow and finally we will get the desired image as the merged form. This process makes the image formed with more accuracy and the machine learning algorithms used will help to form the perfect image without errors.

The content image is nothing but main image will undergo the process of deeplab semantic segmentation in deep lab semantic segmentation it will take shadow of the content image the result of the DLSS (Deep lab semantic segmentation) is mask of content and style image, mask of content undergo the process of neural style transfer to produce resultant output. Here we will go through three modules which will process and leads to the final output image.

#### Deep Lab Semantic Segmentation

This is a process where we will be converting an image into its mask of the so here the image will be in the form of shadow of the image. And this process here is used for only content image and mask of the content image is used. Here the image consists of foreground and back ground as shown in the below figure. The Shadow image occurs as the background image.

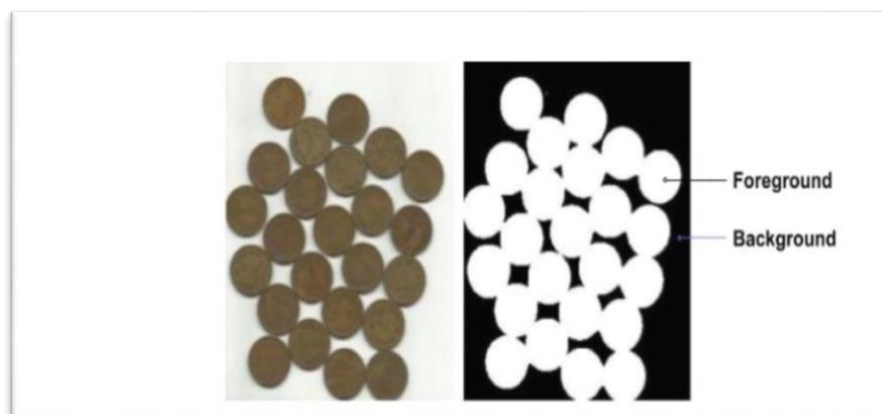


Figure 3.4 Foreground and Background for Deep Lab Semantic Segmentation



Here is a sample for deep lab semantic segmentation. Here a person riding the bicycle is the content image it is formed as the shadow as background.

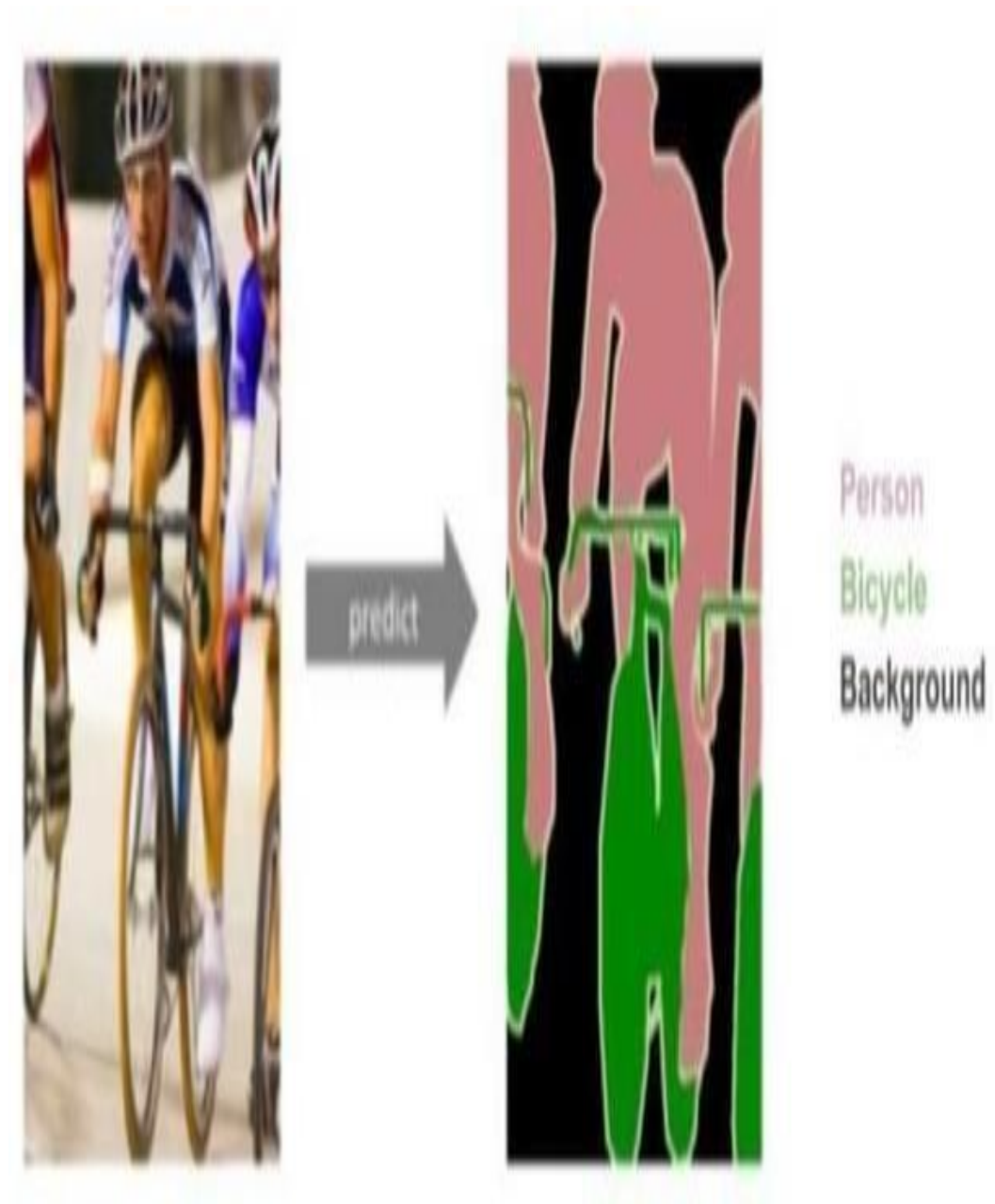


Figure 3.5 Example for Deep Lab Semantic Segmentation



Figure 3.6 Example for Deep Lab Semantic Segmentation

### 3.3 USE CASE DIAGRAM

In the use case diagram we have basically two actors who are the admin and students. The admin has the rights to login, access to resources and to view the details. Whereas the administrator has the login, access to resources of the users and also the right to update and remove the details, and he can also view the user files.

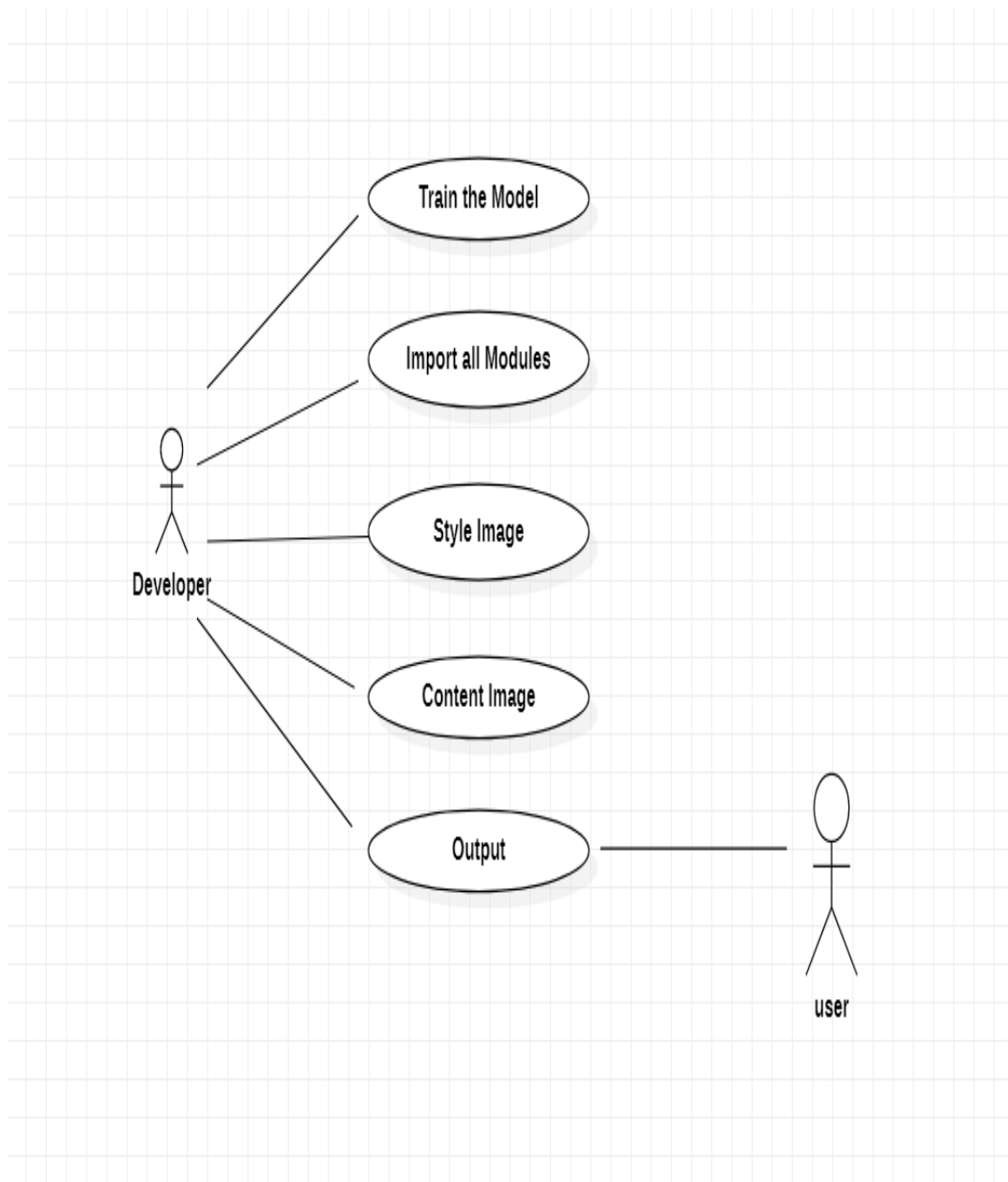


Figure 3.7 Use Case Diagram for Image Style Transfer Using machine learning.

### 3.4 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

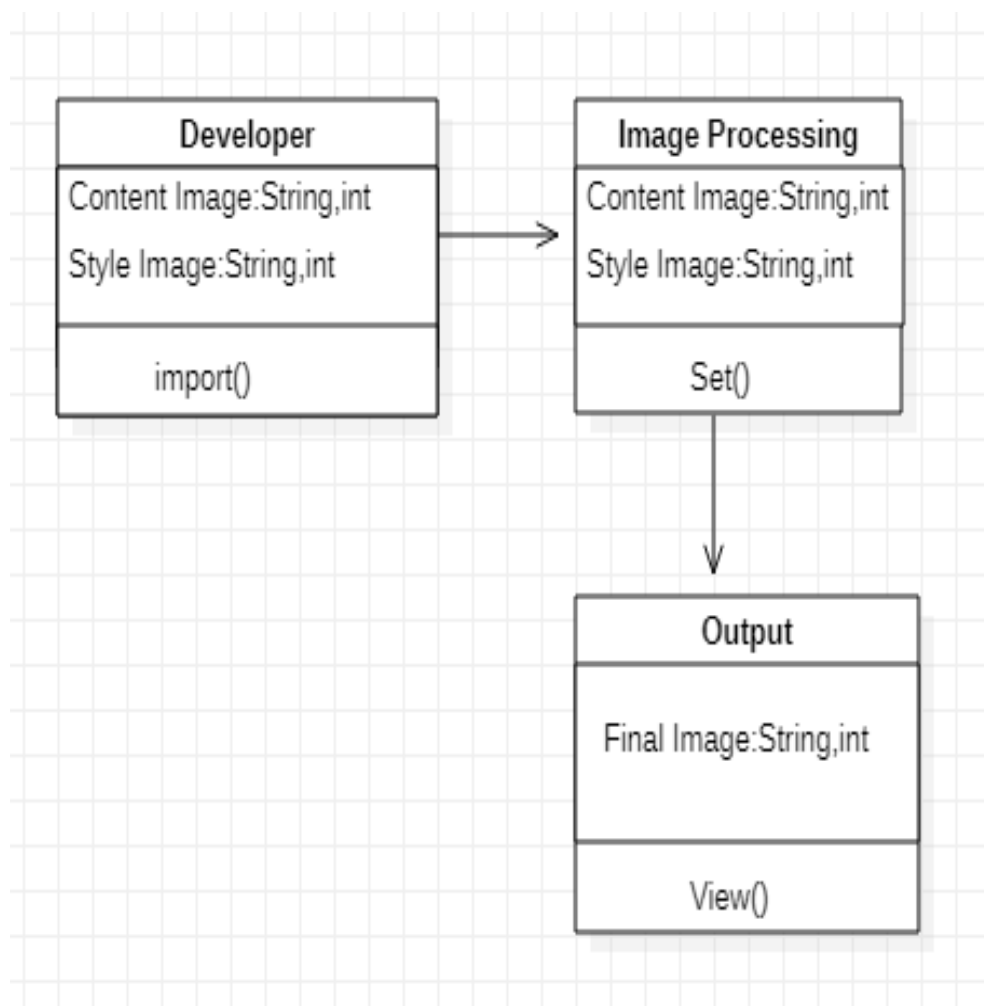


Figure 3.8 Class Diagram for Image Style Transfer Using machine learning.

### 3.5 SEQUENCE DIAGRAM

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more.

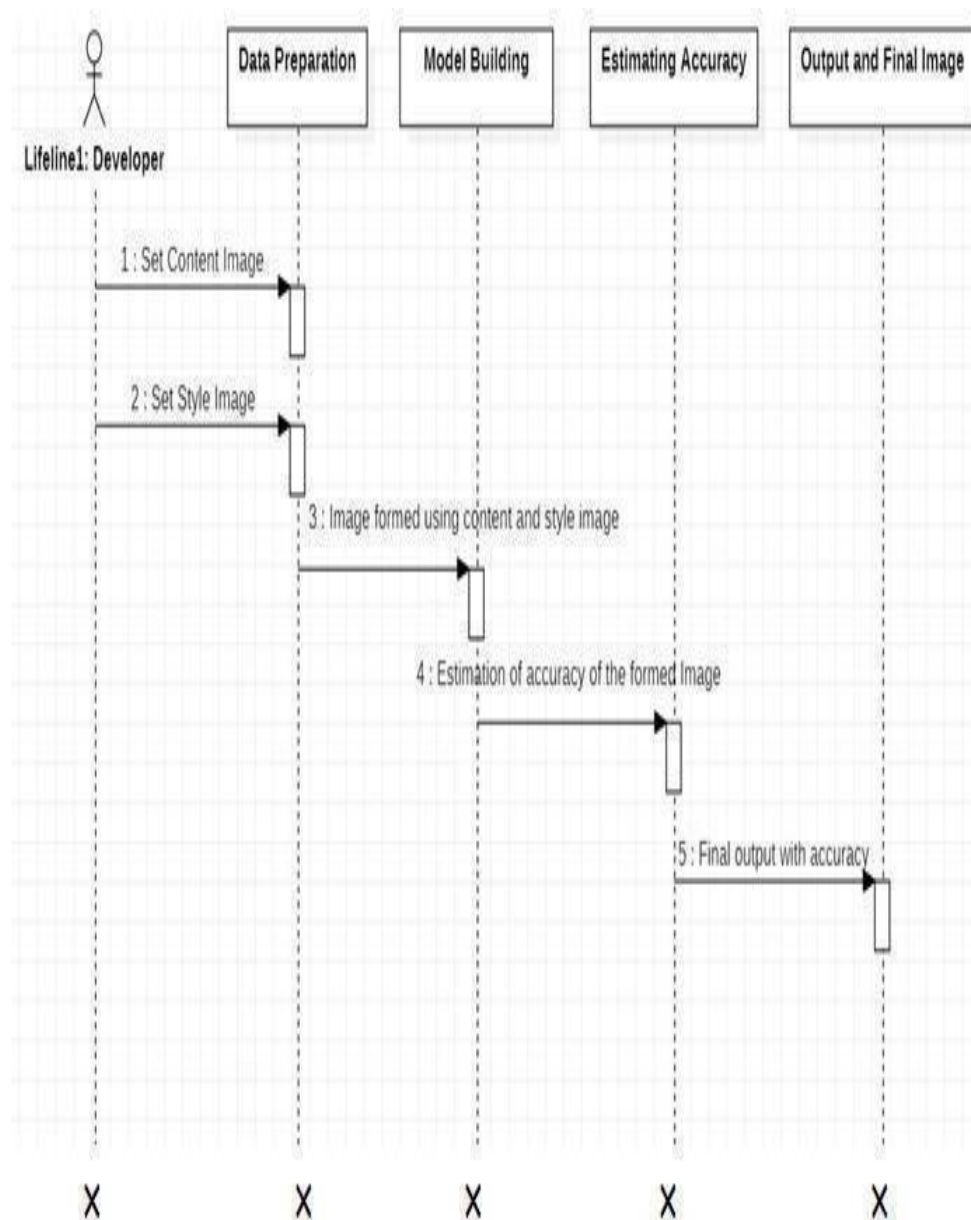


Figure 3.9 Sequence Diagram for imageStyle Transfer Using machine learning.

### 3.6 ACTIVITY DIAGRAM

It describes about flow of activity states. Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

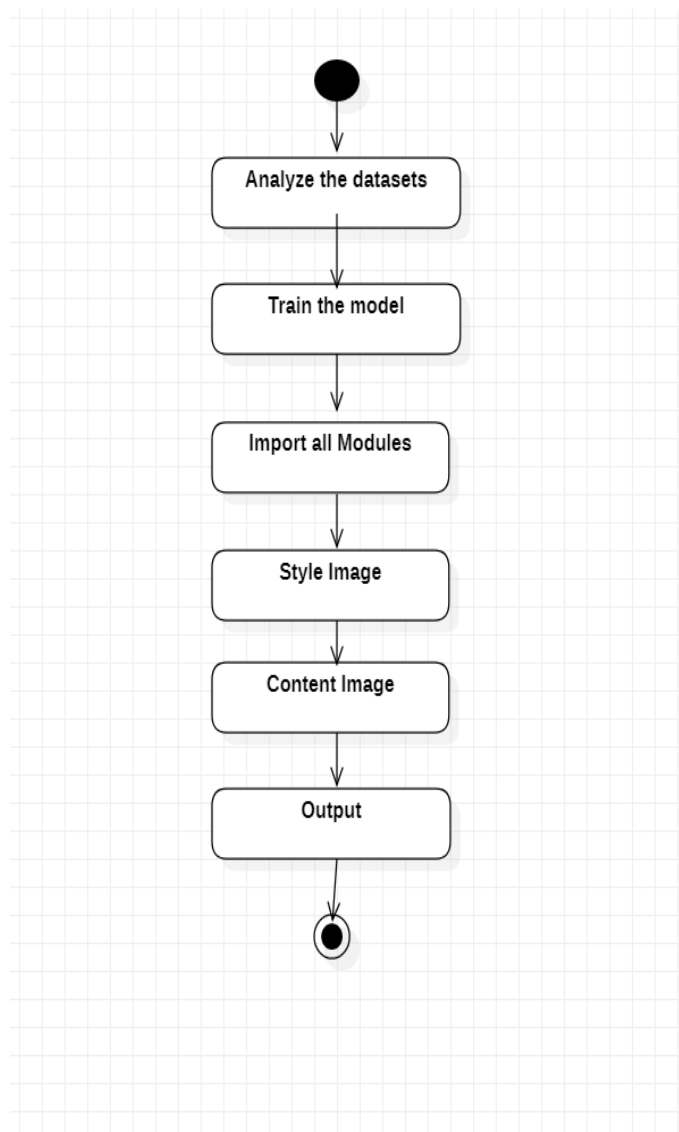


Figure 3.10 Activity Diagram for image Style Transfer Using machine learning.

## **4.IMPLEMENTATION**

## 4.1 SAMPLE CODE

```
# 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

# code
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', 'block3_conv1', 'block5_conv1']
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)) style_img =
cv2.resize(style_img,(512,512)) final_img =
cv2.add(content_img,style_img) cv2.imshow("neural style
transfer",final_img) cv2.waitKey(0)
cv2.destroyAllWindows()

```

## **5.SCREENSHOTS**

## 5.SCREENSHOTS

**Content-Image:**



Figure 5.1 Content Image

**Style-Image:**



Figure 5.2 Style image

## Execution 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 to give the location of the image.

### 5.1 CODE EXECUTION PART-1

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar 31 15:34:40 2022
4
5 @author: govin
6 """
7
8 # import numpy, tensorflow and matplotlib
9 import tensorflow as tf
10 import numpy as np
11 import matplotlib.pyplot as plt
12
13 # import VGG 19 model and keras Model API
14 from tensorflow.python.keras.applications.vgg19 import VGG19, preprocess_input
15 from tensorflow.python.keras.preprocessing.image import load_img, img_to_array
16 from tensorflow.python.keras.models import Model
17 # Image Credits: TensorFlow Doc
18 # Image Credits: TensorFlow Doc
19 content_path = tf.keras.utils.get_file('content.jpg',
20                                     'D://Dog.jpg')
21 style_path = tf.keras.utils.get_file('style.jpg', 'D://styleImage.jpeg')
22
23 # code
24 # this function download the VGG model and initialise it
25 model = VGG19(
26     include_top=False,
27     weights='imagenet'
28 )
29 # set training to False
30 model.trainable = False
31 # Print details of different layers
32
33 model.summary()
34 # code to load and process image
35 def load_and_process_image(image_path):
36     img = load_img(image_path)
37     # convert image to array
38     img = img_to_array(img)
39     img = preprocess_input(img)
40     img = np.expand_dims(img, axis=0)
41     return img

```

Console Output:

```

block4_conv1 (Conv2D) (None, None, None, 512) 1180160
block4_conv2 (Conv2D) (None, None, None, 512) 2350080
block4_conv3 (Conv2D) (None, None, None, 512) 2350080
block4_conv4 (Conv2D) (None, None, None, 512) 2350080
block4_pool (MaxPooling2D) (None, None, None, 512) 0
block5_conv1 (Conv2D) (None, None, None, 512) 2350080
block5_conv2 (Conv2D) (None, None, None, 512) 2350080
Total params: 15,304,768
Trainable params: 0
Non-trainable params: 15,304,768
In [2]:

```

Screenshot 5.1 Code Execution

## 5.2 CODE EXECUTION PART-2

```

43 def deprocess(img):
44     # perform the inverse of the pre-processing step
45     img[:, :, 0] += 103.939
46     img[:, :, 1] += 116.779
47     img[:, :, 2] += 123.68
48     # convert RGB to BGR
49     img = img[:, :, ::-1]
50
51     img = np.clip(img, 0, 255).astype('uint8')
52     return img
53
54
55
56 def display_image(image):
57     # remove one dimension if image has 4 dimension
58     if len(image.shape) == 4:
59         img = np.squeeze(image, axis=0)
60
61     img = deprocess(img)
62
63     plt.grid(False)
64     plt.xticks([])
65     plt.yticks([])
66     plt.imshow(img)
67     return
68
69 # Load content image
70 content_img = load_and_process_image(content_path)
71
72 # Load style image
73 style_img = load_and_process_image(style_path)
74
75 # Define content model
76 content_layer = 'block5_conv2'
77 content_model = Model(
78     inputs=model.input,
79     outputs=model.get_layer(content_layer).output
80 )
81 content_model.summary()
82 # Define style model
83 style_layers = [
84     'block1_conv1',
85     'block1_conv2',
86     'block1_conv3',
87     'block1_conv4',
88     'block2_conv1',
89     'block2_conv2',
90     'block2_conv3',
91     'block2_conv4',
92     'block3_conv1',
93     'block3_conv2',
94     'block3_conv3',
95     'block3_conv4',
96     'block4_conv1',
97     'block4_conv2',
98     'block4_conv3',
99     'block4_conv4',
100     'block5_conv1',
101     'block5_conv2',
102     'block5_conv3',
103     'block5_conv4'
104 ]

```

Usage

Here you can get help of any object by pressing **Ctrl+H** 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**.

Now to Spyder? Read our [tutorial](#)

Variable explorer | Help | Plot | Files

Console (Python 3.8)

```

block4_conv1 (Conv2D) (None, None, None, 512) 1189160
block4_conv2 (Conv2D) (None, None, None, 512) 2359088
block4_conv3 (Conv2D) (None, None, None, 512) 2359088
block4_conv4 (Conv2D) (None, None, None, 512) 2359088
block4_pool (MaxPooling2D) (None, None, None, 512) 0
block5_conv1 (Conv2D) (None, None, None, 512) 2359088
block5_conv2 (Conv2D) (None, None, None, 512) 2359088
Total params: 15,384,768
Trainable params: 0
Non-trainable params: 15,384,768
In [2]:

```

Python console history

LF Python: ready | conda (Python 3.8.6) | Line 131, Col 1 | UTF-8 | CRLF | RW | Mem 82%

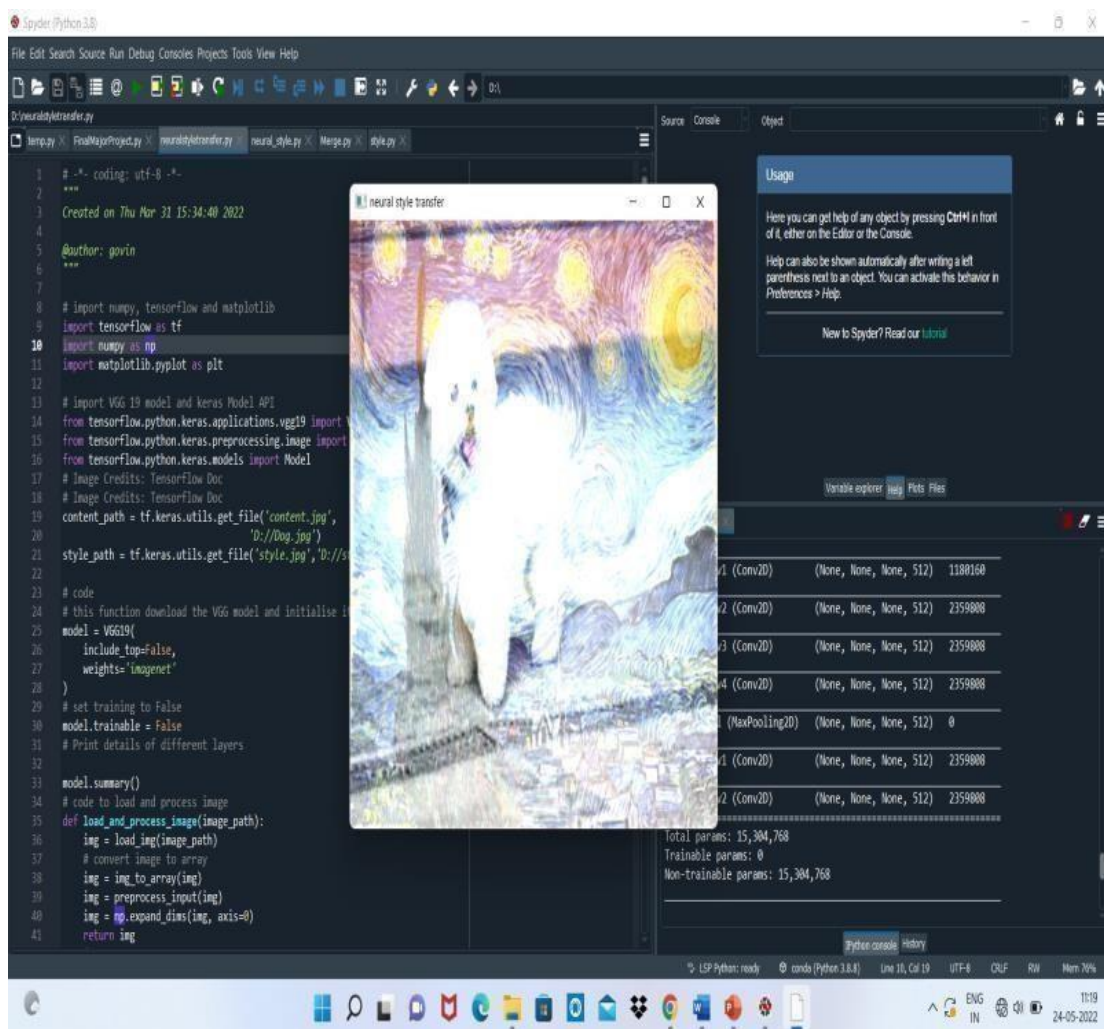
39°C Mostly clear

22:14 25-05-2022

Screenshot 5.2 Code Execution

### 5.3 RESULT OF FINAL IMAGE

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



Screenshot 5.3 Result of the final image



## **6.TESTING**

## 6.TESTING

### 6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover very conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### 6.2 TYPES OF TESTING

#### 6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

#### 6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination

components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### 6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures : Interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

## **7.CONCLUSION**

## **7.CONCLUSION & FUTURE SCOPE**

### **7.1 PROJECT CONCLUSION**

By using this principle of Image Style Transfer using Machine Learning we can get our desired image without recreation. The Accuracy will also be more for this method and it takes less time because there is no recreation of the images. 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 get the image with high Accuracy and it takes less time because there is no recreation of the images, only conversion of images directly.

### **7.2 FUTURE SCOPE**

This Project Can be used to Process the images without big process like re creation of the image we Can easily use this method for converting image into a 3d format by directly giving the images which we want as input.

We can add a feature like voice recognition, we need to recognize the content image and style image using voice by this output image will be formed, here there is no importing of content image and style image.

## **8.BIBLIOGRAPHY**

## 8.BIBLIOGRAPHY

### 8.1 REFERENCES

- [1] N. Ashikhmin. Fast texture transfer. *IEEE Computer Graphics and Applications*, 23(4):38–43, July 2003. [1](#)
- [2] M. Berning, K. M. Boergens, and M. Helmstaedter. SegEM: Efficient Image Analysis for High-Resolution Connections. *Neuron*, 87(6):1193–1206, Sept. 2015. [2](#)
- [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 Neural Networks Rival the Representation of Primate IT Cortex for Core Visual Object Recognition. *PLoS Comput Biol*, 10(12):e1003963, Dec. 2014. [8](#)
- [4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. *arXiv:1412.7062 [cs]*, Dec. 2014. arXiv: 1412.7062. [2](#)
- [5] M. Cimpoi, S. Maji, and A. Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3828–3836, 2015. [2](#)

### 8.2 WEBSITES

- [1] [www.google.com](http://www.google.com).
- [2] CNN-Convolutional neural Network  
Algorithm <https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/>
- [3] Opencv Module in Machine Learning <https://pypi.org/project/opencv-python/>
- [4] Tensor flow module in machine learning  
[https://www.tensorflow.org/hub/api\\_docs/python/hub/Module](https://www.tensorflow.org/hub/api_docs/python/hub/Module)
- [5] Deep Lab Semantic Segmentation for content Image  
<https://towardsdatascience.com/the-evolution-of-deeplab-for-semantic-segmentation-95082b025571>

[6] Content loss and the style loss for the images.

<https://nnart.org/what-is-content-loss/> <https://www.coursera.org/lecture/generative-deep-learning-with-tensorflow/style-loss-l9bTa>

[7] Image processing Process

<https://towardsdatascience.com/convolution-neural-network-for-image-processing-using-keras-dc3429056306>

[8] Calculation of gram matrix

<https://towardsdatascience.com/neural-networks-intuitions-2-dot-product-gram-matrix-and-neural-style-transfer-5d39653e7916>

[9] Keras in python <https://www.javatpoint.com/keras>

### 8.3 GITHUB LINK

<https://github.com/MaringantiAbhigna/IMAGE-STYLE-TRANSFER-USING-MACHINE-LEARNING.git>





# **iJRASET**

International Journal For Research in  
Applied Science and Engineering Technology



---

# **INTERNATIONAL JOURNAL FOR RESEARCH**

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

---

**Volume: 10    Issue: VI    Month of publication: June 2022**

**DOI:**

**[www.ijraset.com](http://www.ijraset.com)**

**Call:  08813907089**

**E-mail ID: [ijraset@gmail.com](mailto:ijraset@gmail.com)**

# Image Style Transfer Using Machine Learning

Dr. Bagam Laxmaiah<sup>1</sup>, Jonnadula Narasimharao<sup>2</sup>, Abhigna Maringanti<sup>3</sup>, SriTeja Govindula<sup>4</sup>, Suresh Dharamsoth<sup>5</sup>

<sup>1, 2, 3, 4, 5</sup>Dept. Of CSE, CMR Technical Campus

**Abstract:** The principle of Image style transfer is to define 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 output will be in the graphical model of the content image.

In summary, we'll take the base input image, a content image 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 loss it 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.

## I. 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 format. 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.

## II. LITERATURE SURVEY

In this project, we have taken a content image and the style image we will save the content image[6] and the style images[5] from any website and we need to use them by their location in the system. Here we use many machine 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 of content of the content image to the content of the output image is the content loss and the content loss is calculated 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:

The Style loss[12] is defined as the difference between the output image and the style image. Which in turn gives the style loss. Here

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

we cannot compare intermediate 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 are 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. In gram matrix we have to calculate each layer style information the formulae is:

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

### III. 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 model. Here in proposed system we will create a data sets of images for target Image. We give an input as any image and using this data sets the image is converted to our target image. Firstly the content image is given and 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.

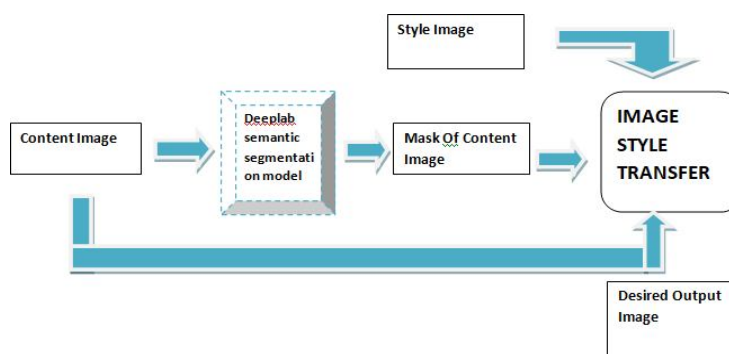
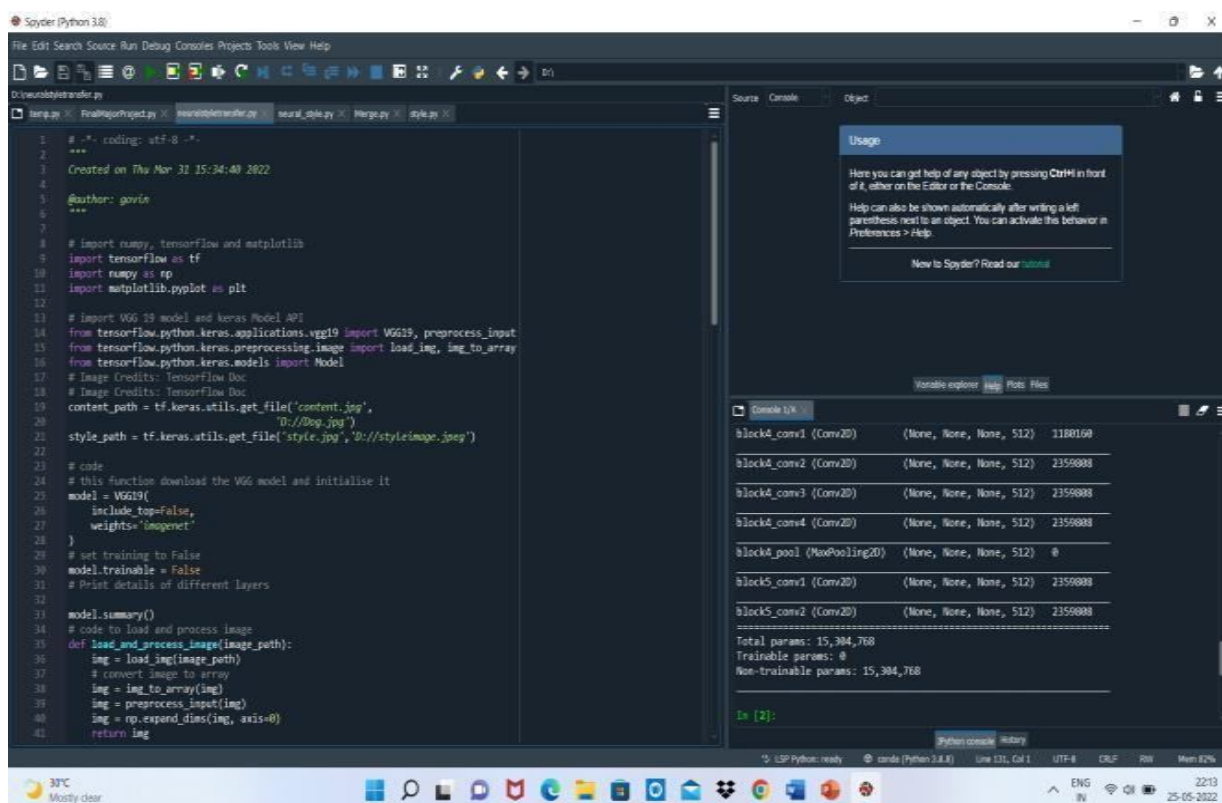


Figure 3.1: Architecture of the Model.

### IV. 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 to give the location of the image.



```

1  # -*- coding: utf-8 -*-
2
3  Created on Thu Mar 31 15:34:40 2022
4
5  @author: gavin
6
7
8  # import numpy, tensorflow and matplotlib
9  import tensorflow as tf
10 import numpy as np
11 import matplotlib.pyplot as plt
12
13 # import VGG 19 model and keras Model API
14 from tensorflow.python.keras.applications.vgg19 import VGG19, preprocess_input
15 from tensorflow.python.keras.preprocessing.image import load_img, img_to_array
16 from tensorflow.python.keras.models import Model
17 # Image Credits: TensorFlow Doc
18 # Image Credits: TensorFlow Doc
19 content_path = tf.keras.utils.get_file('content.jpg',
20                                     'D://dog.jpg')
21 style_path = tf.keras.utils.get_file('style.jpg', 'D://styleimage.jpeg')
22
23 # code
24 # this function download the VGG model and initialise it
25 model = VGG19(
26     include_top=False,
27     weights='imagenet'
28 )
29
30 # set training to False
31 model.trainable = False
32 # Print details of different layers
33
34 model.summary()
35 # code to load and process image
36 def load_and_process_image(image_path):
37     img = load_img(image_path)
38     # convert image to array
39     img = img_to_array(img)
40     img = preprocess_input(img)
41     img = np.expand_dims(img, axis=0)
42     return img

```



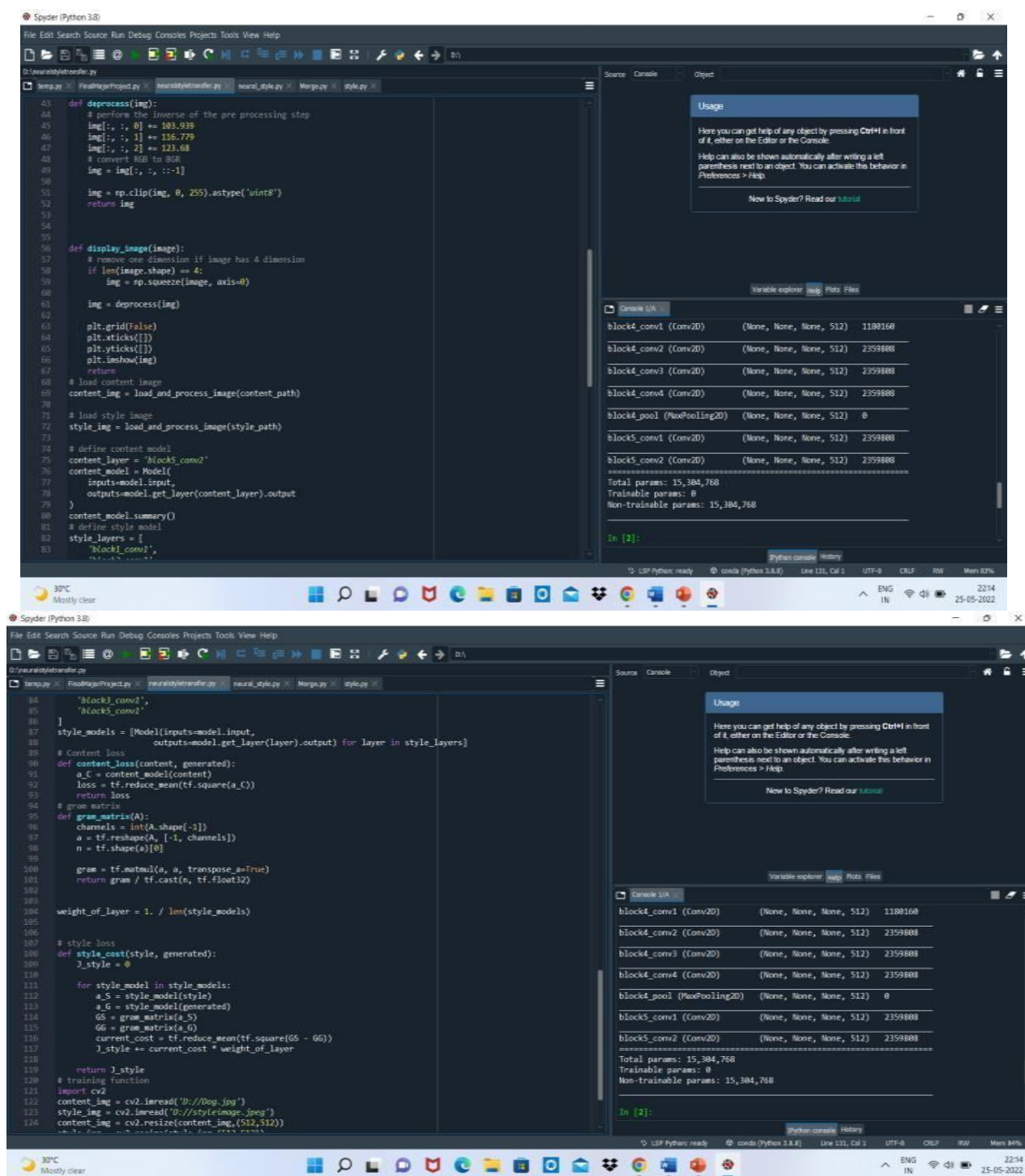


Fig 4.1-4.3: Code Execution

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

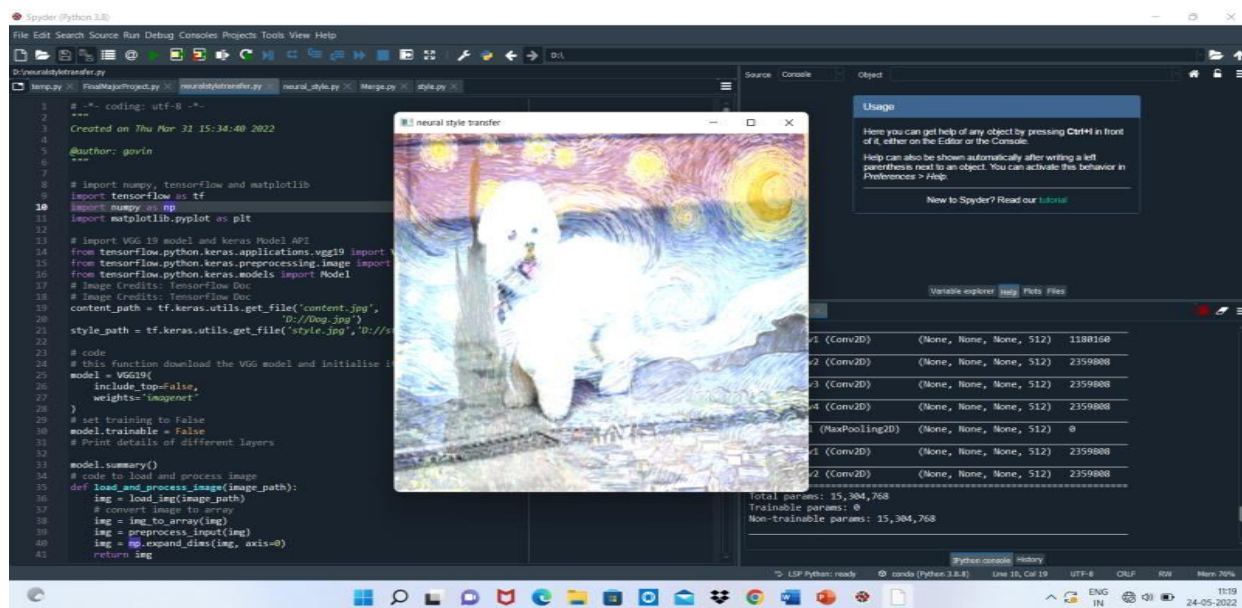


Figure 4.4-Result of the final image

## VI.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 get the image with high Accuracy and it takes less time because there is no recreation of the images, only conversion of images directly.

## 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.Cadiu,H.Hong,D.L.K.Yamins,N.Pinto,D.Ardila, E. A. Solomon, N. J. Majaj, and J. J. DiCarlo. Deep NeuralNetworksRivaltheRepresentationofPrimateITCor-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] Different content 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://pyapi.org/project/opencv-python/>
- [9] Tensorflowmodule in machinelearning [https://www.tensorflow.org/hub/api\\_docs/python/hub/Module](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>





10.22214/IJRASET



45.98



IMPACT FACTOR:  
7.129



IMPACT FACTOR:  
7.429



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24\*7 Support on Whatsapp)



ISSN No. : 2321-9653

# IJRASET

**International Journal for Research in Applied  
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : [www.ijraset.com](http://www.ijraset.com), E-mail : [ijraset@gmail.com](mailto:ijraset@gmail.com)

ISRA  
JIF

ISRA Journal Impact  
Factor: 7.429



45.98  
INDEX COPERNICUS



THOMSON REUTERS  
Researcher ID: N-9581-2016



TOGETHER WE REACH THE GOAL  
SJIF 7.429

## Certificate

*It is here by certified that the paper ID : IJRASET43990, entitled*

*Image Style Transfer Using Machine Learning*

*by*

*Abhigna Maringanti*

*after review is found suitable and has been published in*

*Volume 10, Issue VI, June 2022*

*in*

*International Journal for Research in Applied Science &  
Engineering Technology*

*Good luck for your future endeavors*

By 

Editor in Chief, IJRASET





ISSN No. : 2321-9653

# IJRASET

**International Journal for Research in Applied  
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : [www.ijraset.com](http://www.ijraset.com), E-mail : [ijraset@gmail.com](mailto:ijraset@gmail.com)

## Certificate

*It is here by certified that the paper ID : IJRASET43990, entitled*

*Image Style Transfer Using Machine Learning*

*by*

*SriTeja Govindula*

*after review is found suitable and has been published in*

*Volume 10, Issue VI, June 2022*

*in*

*International Journal for Research in Applied Science &  
Engineering Technology*

*Good luck for your future endeavors*

*By [Signature]*

Editor in Chief, IJRASET

ISRA  
JIF

ISRA Journal Impact  
Factor: 7.429



45.98  
INDEX COPERNICUS



THOMSON REUTERS  
Researcher ID: N-9581-2016



TOGETHER WE REACH THE GOAL  
SJIF 7.429





ISSN No. : 2321-9653

# IJRASET

**International Journal for Research in Applied  
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : [www.ijraset.com](http://www.ijraset.com), E-mail : [ijraset@gmail.com](mailto:ijraset@gmail.com)

## Certificate

*It is here by certified that the paper ID : IJRASET43990, entitled*

*Image Style Transfer Using Machine Learning*

*by*

*Suresh Dharamsoth*

*after review is found suitable and has been published in*

*Volume 10, Issue VI, June 2022*

*in*

*International Journal for Research in Applied Science &  
Engineering Technology*

*Good luck for your future endeavors*

*By [Signature]*

Editor in Chief, IJRASET

ISRA  
JIF

ISRA Journal Impact  
Factor: 7.429



45.98  
INDEX COPERNICUS



THOMSON REUTERS  
Researcher ID: N-9581-2016



TOGETHER WE REACH THE GOAL  
SJIF 7.429