CONTENTS

ABSTRACT

Image style transfer is a deep learning technique for blending the content of one image with the style of another. The output is a new image with the same content as the original content image, but with the style of the style image. This can be achieved by using neural networks to separate and recombine the content and style features of the images.
Image style transfer is an important research content related to image processing in computer vision. Compared with traditional artificial computing methods, deep learning-based convolutional neural networks have powerful advantages. This new method has high computational efficiency and a good style transfer effect. Pre-trained VGG-19 (Visual Geometry Group) neural network models are used to achieve image style transfer.

The main applications of this project are:

1. **Artistic expression**: creating unique and visually appealing images by combining the content of one image with the style of a painting or a famous artist's work.
2. **Graphic design:** using style transfer to generate various design variations and logos.
3. **Photography:** editing photos to give them a desired aesthetic or mood.
4. **Film and video production:** transferring style from reference images to create a consistent visual style for a film or video project.
5. **Virtual Reality and Augmented Reality:** adding artistic styles to VR and AR experiences to enhance their visual appeal.
6. **Printing and publishing:** using style transfer to automatically generate visually appealing layouts and designs for books, magazines, and other printed materials.

The main features of this project are:

1. Use of VGG-19 as a pre-trained convolutional neural network
2. Transfer of style from a reference image to a content image
3. Use of a loss function to optimize image generation
4. Combination of content and style losses for overall transfer
5. Generation of a unique, stylized image that preserves content features.

CHAPTER 1

INTRODUCTION

1.1 GENERAL INTRODUCTION

Image style transfer is a technique that enables you to blend the content of one image with the style of another, resulting in a new image with the visual traits of both. The VGG-19 convolutional neural network (CNN), which was pre-trained on a sizable dataset of images for image identification tasks, is one well-liked technique for carrying out visual style transfer.

The fundamental idea underlying style transfer with VGG-19 is to use the network's feature maps to gather information about the content and style of the input images. Although the lower-level feature maps that capture the texture and color information of the style picture provide the style information, the higher-level feature maps that capture the semantic information of the input image provide the content information.

The neural style transfer method, which involves maximizing an objective function that balances the content and style of the input images, can be used to execute style transfer using VGG-19. This entails producing a new image that minimizes the separation between the input image's content features and the generated image as well as the style image's features and the generated image.

Create interactive web apps for data science and machine learning tasks using the Python package Streamlit. A user-friendly tool for doing image style transfer on their photographs can be made by combining a Streamlit GUI with the VGG-19 style transfer algorithm. Features like choosing the content and style images, modifying the style transfer parameters, and instantly previewing the resulting image are examples of this. When used together, VGG-19 and Streamlit can offer an easy-to-use method of transferring image style.

1.2 GOAL OF THE PROJECT

This project's objective is to provide a user-friendly solution for image style transfer that makes use of the VGG-19 convolutional neural network and a Streamlit Interface. The application is made to enable users to blend the appearance of one image with the content of another, offering a flexible and user-friendly method for modifying images, expressing one's creativity, and creating designs. Users can use this tool to produce one-of-a-kind, customized photographs that reflect their aesthetic tastes or company identity as well as to improve the visual attractiveness of their already existing images. The goal of this project is to offer a workable and approachable solution for a range of image-related problems by combining the strength of VGG-19 and the ease of Streamlit.

CHAPTER 2

LITERATURE SURVEY

2.1 STUDY OF SIMILAR WORK

Image style transfer is a method that enables the blending of one image's visual aesthetic with its content. Due to the vast range of uses for this approach, including artistic expression, design, and image manipulation, its popularity has grown. For image style transfer, several strategies have been put forth, each with advantages and disadvantages. The VGG-19, DCGAN, and CAST frameworks are three different visual style transfer algorithms that we evaluate and examine in this study. Due to its capacity to record both content and style information, the CNN-based VGG-19 technique has been frequently employed for image style transfer.

A generative model called DCGAN can create images that are stylistically similar to a reference image while maintaining the integrity of the input image. The CAST framework integrates object awareness with both content and style transfer. It is a framework for content-aware style transfer. By contrasting and evaluating different approaches, we hope to shed light on their benefits and drawbacks as well as suggest possible lines of inquiry for further study on picture style transmission.

2.1.1 EXISTING SYSTEM

The Existing System now uses a variety of methods and strategies for image style transfer. Using pre-trained deep neural networks, like VGG-19, to extract content and style information from the input photos is a typical strategy. These networks serve as feature extractors in the image style transfer process and are trained on big datasets of images for object recognition tasks.

Another strategy is to create new photos with the desired style while keeping the original image's content using generative models like GANs (Generative Adversarial Networks) and VAEs (Variational Autoencoders). These models can either learn to create new styles based on a given reference image or can be trained on datasets of photos with a particular style.

Another strategy is to employ a deep learning framework. The CAST Framework is a multi-stage deep learning strategy for using contrastive learning to transfer the style of one image to another. It can maintain fine features in the image and manage a variety of styles and domains. It improves the capacity to manage several different styles at once and preserve semantic coherence in the transmitted image.

2.1.2 DRAWBACK OF EXISTING SYSTEM

- **Limited diversity in the results of the style transfer:** One of the primary shortcomings of the VGG-19 and DCGAN-based approaches is that they can only produce a limited diversity in the results of the style transfer. This can make the procedure less engaging and creative because the copied photos may end up looking extremely similar to one another.

- **Costly to compute**: This is applicable for all algorithms, can be costly to compute, especially when working with high-resolution images or intricate designs. The process of transferring styles may become time- and resource-consuming as a result.

- **Absence of fine details:** This is also applicable for all algorithms, occasionally provide transferred images that are hazy or lack fine features, which might lower the output's overall quality.

- **Limited control over the style transfer process:** This is present in both VGG-19 and DCGAN-based techniques, which makes it possible that the user won't get the precise outcome they were hoping for.

- **Absence of user-friendly GUI:** Some current systems might not have one, which can make it challenging for users to interact with the system and produce the needed outcomes.

- **Prototype models only:** Since some of the current systems are only prototypes, they might not be completely optimised or scaled for usage in production. This may restrict their scalability and range of use.

CHAPTER 3

OVERALL DESCRIPTION

3.1 PROPOSED SYSTEM

The proposed system aims to leverage the advantages of VGG-19 over VGG16 neural network models for image style transfer. The research paper entitled "**Image Style Transfer Based on VGG Neural Network Model"** published at the IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA) in 2022 compared the two models and concluded that VGG16 has a simpler architecture, lower computational complexity, comparable performance, and faster training. However, VGG-19 offers a deeper architecture, higher accuracy, increased model capacity, and flexibility for transfer learning, making it a more future-proof choice for the proposed application. Additionally, the proposed system will include a user-friendly GUI and user support materials, making it commercially viable and accessible to a wide range of users.

3.2 FEATURES OF PROPOSED SYSTEM

The proposed system for image style transfer using VGG-19 neural network model will have several key features, including Image Style Transfer, VGG-19 Neural Network Model, User-friendly GUI, Real-time Visualization, Customizable Parameters, Comprehensive Documentation and User Support, and Commercialization Potential. The proposed system will utilize the VGG-19 neural network model, which has a deeper architecture, higher accuracy, increased model capacity, and flexibility for transfer learning. It will also have a user-friendly GUI, customizable parameters, comprehensive documentation and user support materials, and commercialization potential.

3.3 FUNCTIONS OF PROPOSED SYSTEM

The proposed system for image style transfer using VGG-19 neural network model will perform the following functions:

1. **Image Input:** The system will allow users to input a source image and a style image. These images will serve as the input for the style transfer process.

2. **Style Transfer:** The system will apply the style of the style image onto the source image using the VGG-19 neural network model. The style transfer process will involve extracting style features from the style image and transferring them onto the source image, while preserving the content of the source image.

3. **Parameter Customization:** The system will provide users with the ability to customize various parameters, such as the scale of the style transfer, to achieve the desired level of style blending. Users will be able to adjust these parameters in real-time and see the effect on the output image.

4. **Output Image Generation:** The system will generate the style-transferred output image based on the source image, style image, and the adjusted parameters. The output image will reflect the desired style blending between the source and style images.

5. **Multiple Device Compatibility:** The system should be compatible with different devices, including desktop computers, laptops, tablets, and smartphones, allowing users to access and use the application across various platforms.

6. **User-friendly GUI:** The system will have a graphical user interface (GUI) that is intuitive and easy to use. Users will be able to interact with the system through the GUI, input images, adjust parameters, and visualize the output image in real-time.

7. **Download Button for PNG Image:** The system will provide users with a "Download" button in the output interface, which will allow them to save the style-transferred output image in PNG file format. This will provide users with an easy and convenient way to save the generated image for further use.

8. **Documentation and User Support:** The system will be accompanied by comprehensive documentation and user support materials, including user manuals, tutorials, and troubleshooting guides. This will assist users in effectively utilizing the system, understanding its functions, and resolving any issues they may encounter.

9. **Commercialization Potential:** The system will aim to have commercialization potential, with a user-friendly GUI, customizable parameters, and comprehensive user support materials. This will make the system attractive for various applications and industries, providing potential for commercialization and adoption in real-world scenarios.

3.4 REQUIREMENTS SPECIFICATION

1. **High Accuracy:**
   The system should accurately perform image style transfer using VGG-19 model, producing visually appealing and faithful style-transferred output images.

2. **Good Speed:**
   The system should process style transfer in a timely manner, without significant delays or lags.

3. **User-friendly Interface:**
   The system should have an intuitive and visually appealing GUI that allows users to easily input images, adjust parameters, visualize results, and download output images.

4. **Customizable Parameters:**
   The system should provide users with flexibility to customize parameters for style blending.

5. **Multiple Device Compatibility:**
   The system should be compatible with various devices and operating systems, such as Windows, macOS, iOS, and Android, to allow users to access and use the application across different platforms.

6. **Robustness and Reliability:**
   The system should be able to handle different image types, sizes, and styles without crashing or producing erroneous results.

7. **Documentation and User Support:**
   The system should be accompanied by comprehensive documentation and user support materials, including manuals, tutorials, and troubleshooting guides.

8. **Commercialization Potential:**
   The system should have potential for commercialization, with scalability, user appeal, and value in various applications and industries.

3.5 FEASIBILITY STUDY

Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system. Feasibility study is one of stage among important four stages of Software Project Management Process. As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyse whether software product will be right in terms of development, implantation, contribution of project to the organization etc.
In our proposed system the product is feasibility can be achieved in all four aspects Technical Operational, Economical and Behavioural.

3.5.1 TECHNICAL FEASIBILITY

In Technical Feasibility current resources both hardware software along with required technology are analysed/assessed to develop project. This technical feasibility study gives report whether there exists correct required resources and technologies which will be used for project development. Along with this, feasibility study also analyses technical skills and capabilities of technical team, existing technology can be used or not, maintenance and up-gradation is easy or not for chosen technology etc. In this proposed system technical feasibility is achieved according to above criteria.

3.5.2 OPERATIONAL FEASIBILITY

In Operational Feasibility degree of providing service to requirements is analysed along with how much easy product will be to operate and maintenance after deployment. Along with these other operational scopes are determining usability of product, determining suggested solution by software development team is acceptable or not etc. The Operational feasibility can be ensured by the proposed system.

3.5.3 ECONOMICAL FEASIBILITY

Economic feasibility the most important and frequently used method for evaluating the effectiveness of the proposed system. It is very essential because the main goal of the proposed system is to have economically better results along with increased efficiency. Cost benefit analysis is usually performed for the expected from the proposed system. Since the organization is well equipped with the required hardware, the project was found to be economically feasible and the users who possess a device supports Windows operating system can easily use it.

3.5.4 BEHAVIORAL FEASIBILITY

The proposed system satisfies behavioral feasibility because the system is providing with good and minimalistic GUI which can easily be understand for any end users and it encapsulates the conversion procedure from the users. Hence, it’s easier to operate the system with ease.

CHAPTER 4

OPERATING ENVIRONMENT

4.1 HARDWARE REQUIREMENTS

1. Processor: Intel i3 7th Gen 3.0 GHz or higher; AMD Ryzen 3 3200G or higher
2. Hard disk: 500 GB
3. RAM: 6GB or higher
4. Monitor: 17" Color Monitor or higher
5. Mouse: Microsoft
6. Keyboard: Microsoft multimedia keyboard

4.2 SOFTWARE REQUIREMENTS

1. Operating System: Windows 10 or higher; Mac OS X 10.11 or higher
2. Framework: Flask Framework
3. Environment: PyCharm Community Edition 2022.2
4. Language: Python 3.8.10, Streamlit 1.18.1, Numpy 1.20.3 and Tensor Flow 2.3.0
5. Documentation: Microsoft Word 2010 or higher

4.3 TOOLS AND PLATFORMS

4.3.1 PYCHARM

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

4.3.2 PYTHON 3.8

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

4.3.3 STREAMLIT

Streamlit is an open-source Python library that allows data scientists and developers to create interactive web applications for data visualization and machine learning (ML) purposes. It provides a simple and intuitive way to build web applications directly from Python scripts, without requiring extensive web development experience.
With Streamlit, you can easily create interactive user interfaces (UIs) for data-driven applications by writing Python code. Streamlit automatically handles the rendering of the UI components, such as buttons, sliders, and charts, making it easy to create interactive and responsive web applications with just a few lines of code.

Under the hood, Streamlit uses the Flask web framework, which is a lightweight and flexible web framework for Python. Flask provides the underlying web server functionality for Streamlit to handle incoming requests and serve the web application to users. However, as a Streamlit user, you do not need to directly interact with Flask or have knowledge of Flask's intricacies, as Streamlit abstracts the complexity of web development and provides a high-level interface for building web applications using familiar Python syntax.
Streamlit also provides seamless integration with popular Python libraries, such as Pandas, Matplotlib, and TensorFlow, allowing you to easily incorporate data analysis, visualization, and ML functionalities into your web applications.
Streamlit is commonly used in data science and ML workflows for creating prototypes, building dashboards, and sharing data-driven applications with others. It is widely used in industries such as finance, healthcare, and retail, as well as in research and academia, for creating interactive and user-friendly web applications for data analysis and visualization.

4.3.4 DEEP LEARNING

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.
In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

4.3.5 CNN

In deep learning, a **convolutional neural network** (**CNN/ConvNet**) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics **convolution** is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.

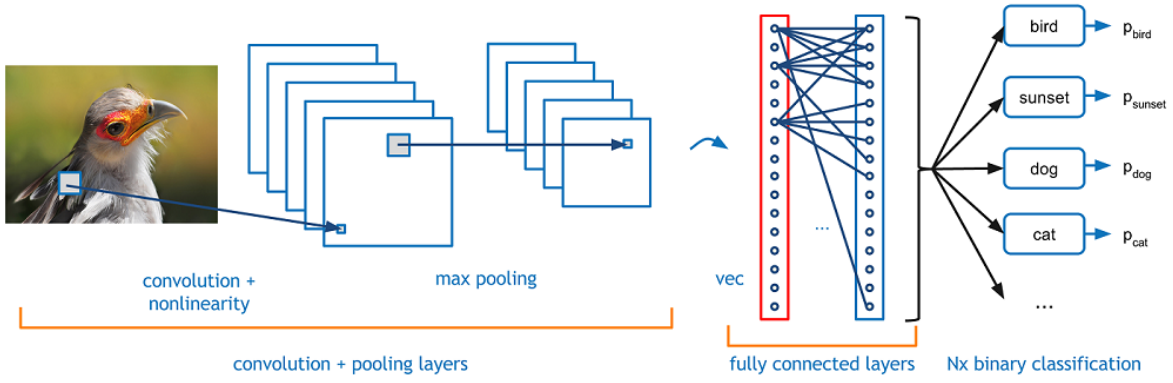


Figure 1: CNN

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

4.3.6 VGG-19

VGG-19, also known as **Visual Geometry Group-19**, is a deep convolutional neural network (CNN) architecture for image recognition and computer vision tasks. It was developed by the Visual Geometry Group at the University of Oxford, and it is a variant of the original VGG network, which was introduced by Simonyan and Zisserman in 2014.
VGG-19 consists of 19 weight layers, including 16 convolutional layers and 3 fully connected layers. The convolutional layers are designed to learn local features from input images, while the fully connected layers are used for high-level feature extraction and classification. VGG-19 has a uniform architecture with small convolutional filters (3x3) and max pooling layers (2x2) applied sequentially, followed by fully connected layers. It is known for its deep architecture with a large number of parameters, which makes it suitable for tasks that require capturing fine-grained features in images.
One of the notable contributions of VGG-19 is its simplicity and the ability to train deep CNNs with relatively small filter sizes. It has been widely used as a benchmark architecture for image classification and feature extraction tasks in computer vision research and applications. VGG-19 has achieved top performance on various image recognition tasks, such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) and is often used as a base architecture for transfer learning, where pre-trained VGG-19 models are used as a starting point for fine-tuning on specific tasks with smaller datasets.

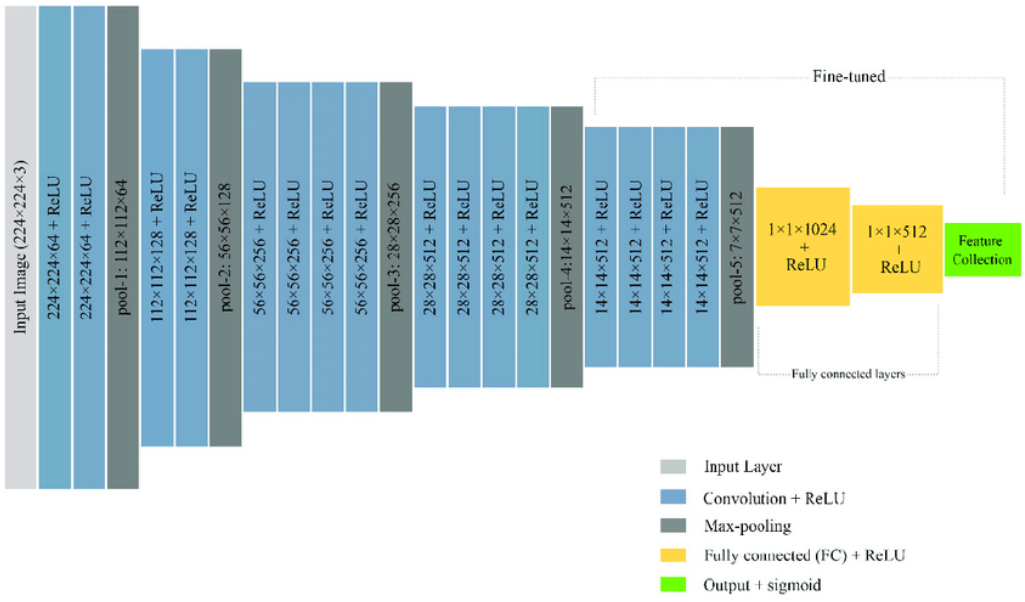


Figure 2: VGG-19 Architecture

4.3.7 GRADIENT DESCENT

Gradient descent is an optimization algorithm used to find the minimum of a function by iteratively adjusting the parameters in the direction of the negative gradient. In other words, it tries to find the lowest point of a function by moving in the direction of steepest descent. The algorithm updates the parameters with a learning rate that controls the size of the steps taken at each iteration. While gradient descent can be effective in finding the minimum of a function, it can be slow and computationally expensive, especially for large datasets.

Stochastic gradient descent (SGD) is a variation of gradient descent that addresses some of its limitations. Instead of using the entire dataset to compute the gradient at each iteration, SGD randomly selects a subset of the data (a mini-batch) and computes the gradient using only that subset. This reduces the computational cost of each iteration and can speed up the optimization process. Moreover, exponential coverage allows to sample more examples that are more important to the optimization problem, which can lead to better convergence rates.

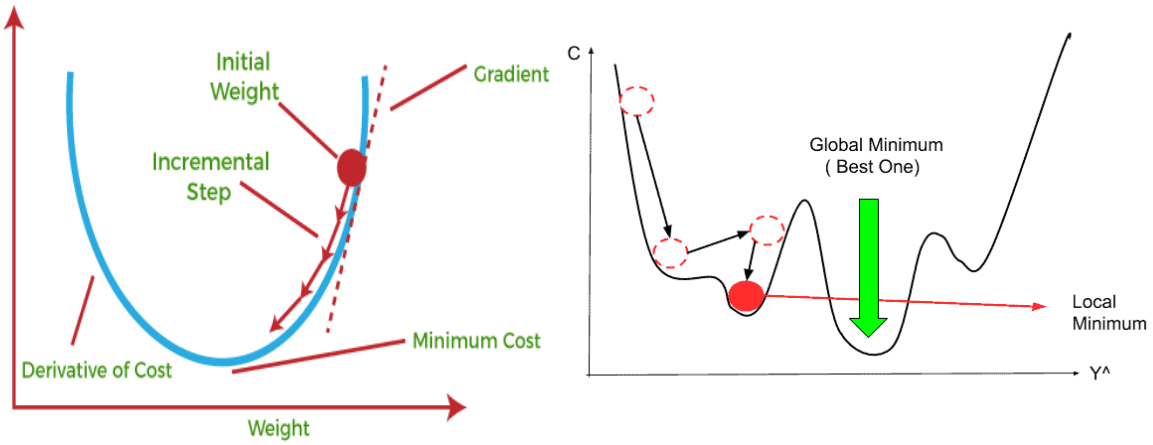


Figure 3: Gradient Descent (left) and Stochastic Gradient Descent (right)

4.3.8 TENSORFLOW 2 AND KERAS

TensorFlow 2 is an end-to-end, open-source machine learning platform. You can think of it as an infrastructure layer for differentiable programming. It combines four key abilities:

- Efficiently executing low-level tensor operations on CPU, GPU, or TPU.
- Computing the gradient of arbitrary differentiable expressions.
- Scaling computation to many devices, such as clusters of hundreds of GPUs.
- Exporting programs ("graphs") to external runtimes such as servers, browsers, mobile and embedded devices.

Keras is the high-level API of TensorFlow 2: an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

Keras empowers engineers and researchers to take full advantage of the scalability and cross-platform capabilities of TensorFlow 2: you can run Keras on TPU or on large clusters of GPUs, and you can export your Keras models to run in the browser or on a mobile device.

4.3.9 JUPYTER NOTEBOOK

Jupyter Notebook (formerly IPython Notebook) is a web-based interactive computational environment for creating notebook documents. Jupyter Notebook is built using several open-source libraries, including IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax. A Jupyter Notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a JSON document, following a versioned schema, usually ending with the ".ipynb" extension.

4.3.10 NVIDIA CUDA

CUDA (or **Compute Unified Device Architecture**) is a parallel computing platform and application programming interface (API) that allows software to use certain types of graphics processing units (GPUs) for general purpose processing, an approach called general-purpose computing on GPUs (GPGPU). CUDA is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements, for the execution of compute kernels.
CUDA is designed to work with programming languages such as C, C++, and Fortran. This accessibility makes it easier for specialists in parallel programming to use GPU resources, in contrast to prior APIs like Direct3D and OpenGL, which required advanced skills in graphics programming. CUDA-powered GPUs also support programming frameworks such as OpenMP, OpenACC and OpenCL and HIP by compiling such code to CUDA.

4.3.11 CANVA

Canva is an Australian graphic design platform, used to create social media graphics, presentations, posters, documents and other visual content The app includes templates for users to use. The platform is free to use and offers paid subscriptions such as Canva Pro and Canva for Enterprise for additional functionality. In 2021, Canva launched a video editing tool. Users can also pay for physical products to be printed and shipped. It has announced it intends to compete with Google and Microsoft in the office software category, with website and whiteboard products.

Figure 4: The logo for the Stylegen application was created using Canva

CHAPTER 5

DESIGN

5.1 SYSTEM DESIGN

System design is the process of defining the architecture, modules, and data for a system to satisfy specified requirements. It is the phase where the SRS document is converted into a format that can be implemented and decides how the system will operate. The purpose of design phase is to plan a solution for problem specified by the requirements. System design aims to identify the modules that should be in the system, the specification of those modules and how they interact with each other to produce the result. The goal of the design process is to produce a model for or representation of a system can be used later to build. The produced model is called design of the system.

5.2 PROGRAM DESIGN

This program design consists of five steps/modules that work together to achieve image style transfer:

Step 1: Image Loading and Pre-processing Module
In this step, the module will load the content and style images, pre-process them, and convert them into tensor format suitable for the VGG-19 network. It will also take the number of epochs from the end user.

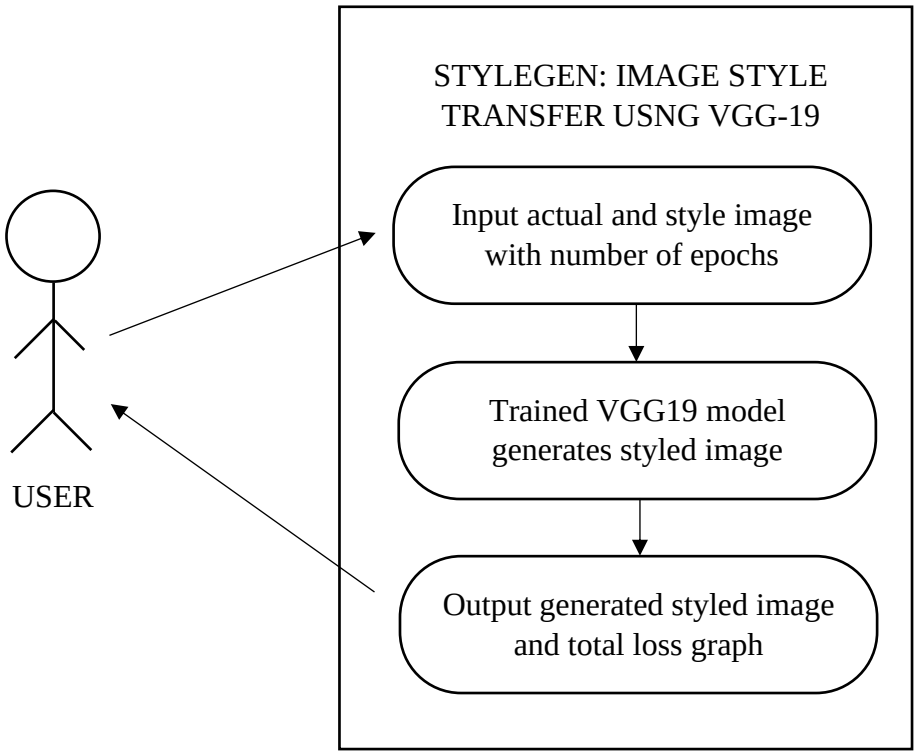
Step 2: VGG-19 Feature Extraction Module
The purpose of this module is to extract features from both the content and style images using the VGG-19 network. The module will separate the layers for style and content loss calculation and perform the feature extraction process.

Step 3: Loss Calculation Module
This module will calculate the total loss for the image style transfer. It will consist of a content loss term and a style loss term, which will be computed using the features extracted by the VGG-19 network. The module will involve the calculation of the Gram matrix, style loss, content loss, total validation loss, and gradient loss.

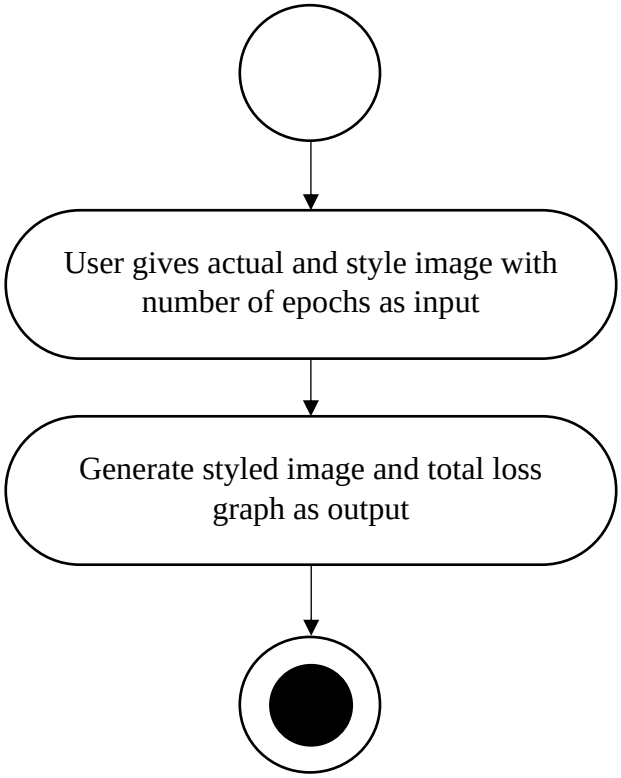
Step 4: Optimization Module
In this step, an optimization algorithm, such as stochastic gradient descent, will be utilized to minimize the total loss and generate the stylized output image. The module will generate the styled image iteratively based on the calculated gradients and total loss in each iteration.

Step 5: De-processing Module
The de-processing module will perform any necessary post-processing on the output image. This may include converting the tensor output back to an image format, rescaling it to the original size, and saving it to disk or displaying it. Additionally, the module will create a graph to visualize the total loss values over time. The output image can be downloaded and the total loss graph can be viewed.

5.3 USE CASE DIAGRAM

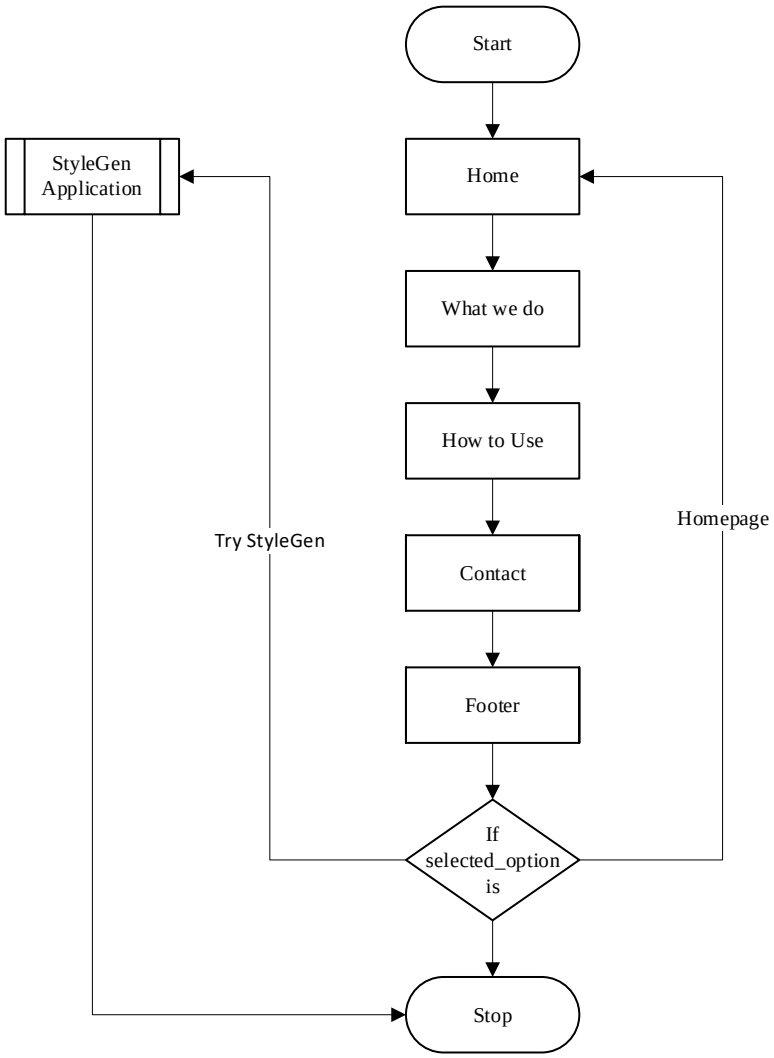


5.4 ACTIVITY DIAGRAM

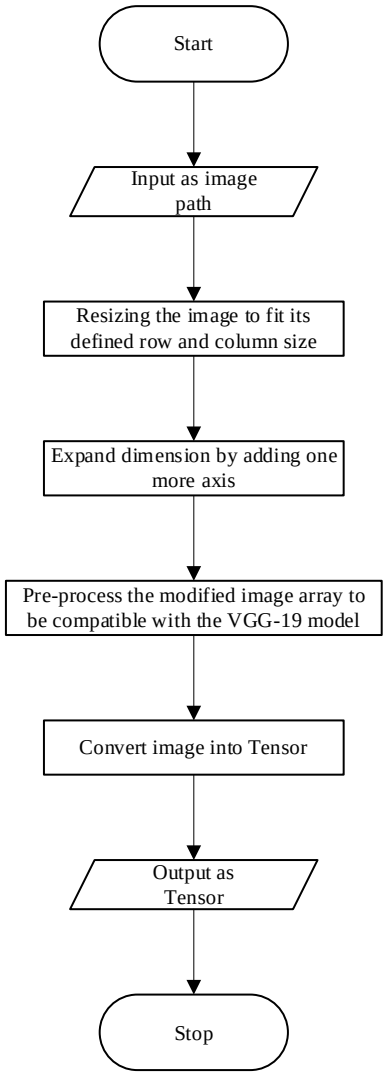


5.5 PROPOSED PROJECT PIPELINE

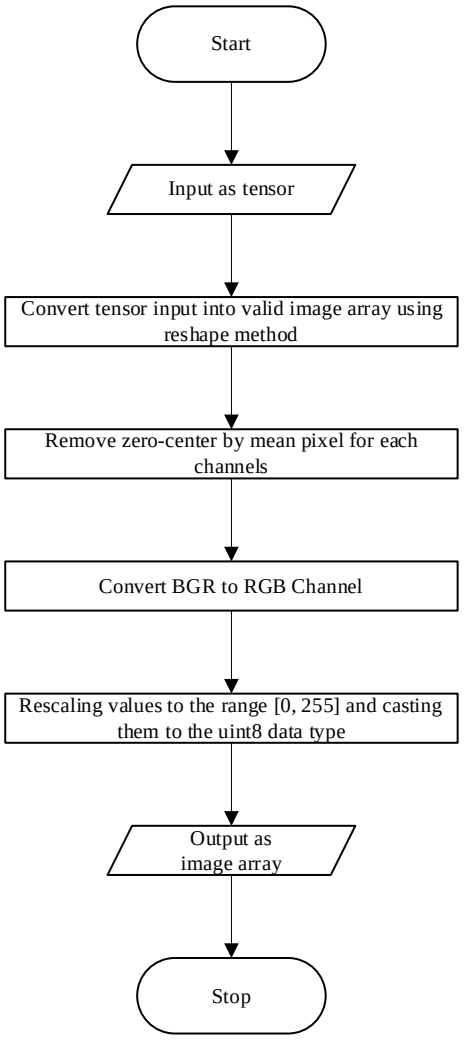
5.5.1 PIPELINE FOR PROJECT DESIGN OVERVIEW



5.5.2 PIPELINE FOR PRE-PROCESSING



5.5.3 PIPELINE FOR DE-PROCESSING



5.5.4 PROCESS OF COMPUTE LOSS AND GRADIENTS

The computation of loss and gradients consists of four major steps. These are as follows:

1. **Calculate Content Loss:** Compute the content loss by comparing the feature representation of the target content image with the feature representation of the generated image.

   Formula
   $ContentLoss = 0.5 * sum((F\_target - F\_generated)^2)$

   Where, F_target and F_generated are the feature maps of the selected layer for the target content image and the generated image, respectively.

2. **Calculate Style Loss:** Compute the style loss by comparing the Gram matrices of the feature representations of the style image and the generated image at the selected layers.

   Formula
   $StyleLoss = sum(w\_l * sum((G\_l\_target - G\_l\_generated)^2))$

   Where, G_l_target and G_l_generated are the Gram matrices of the feature maps of the selected layer for the style image and the generated image, respectively, and w_l is a weighting factor for the corresponding layer.

   NOTE

   Calculate Gram Matrix:

   Given a feature map F of size C x H x W, where C is the number of channels and H, W are the height and width of the feature map respectively, the Gram matrix G of F is a C x C matrix defined as:

   $G = F * F^T / (C * H * W)$

   *Where, * denotes matrix multiplication.*

3. **Calculate Total Loss**: Combine the content loss and the style loss to obtain the total loss.

   Formula
   $TotalLoss = \alpha * ContentLoss + \beta * StyleLoss$

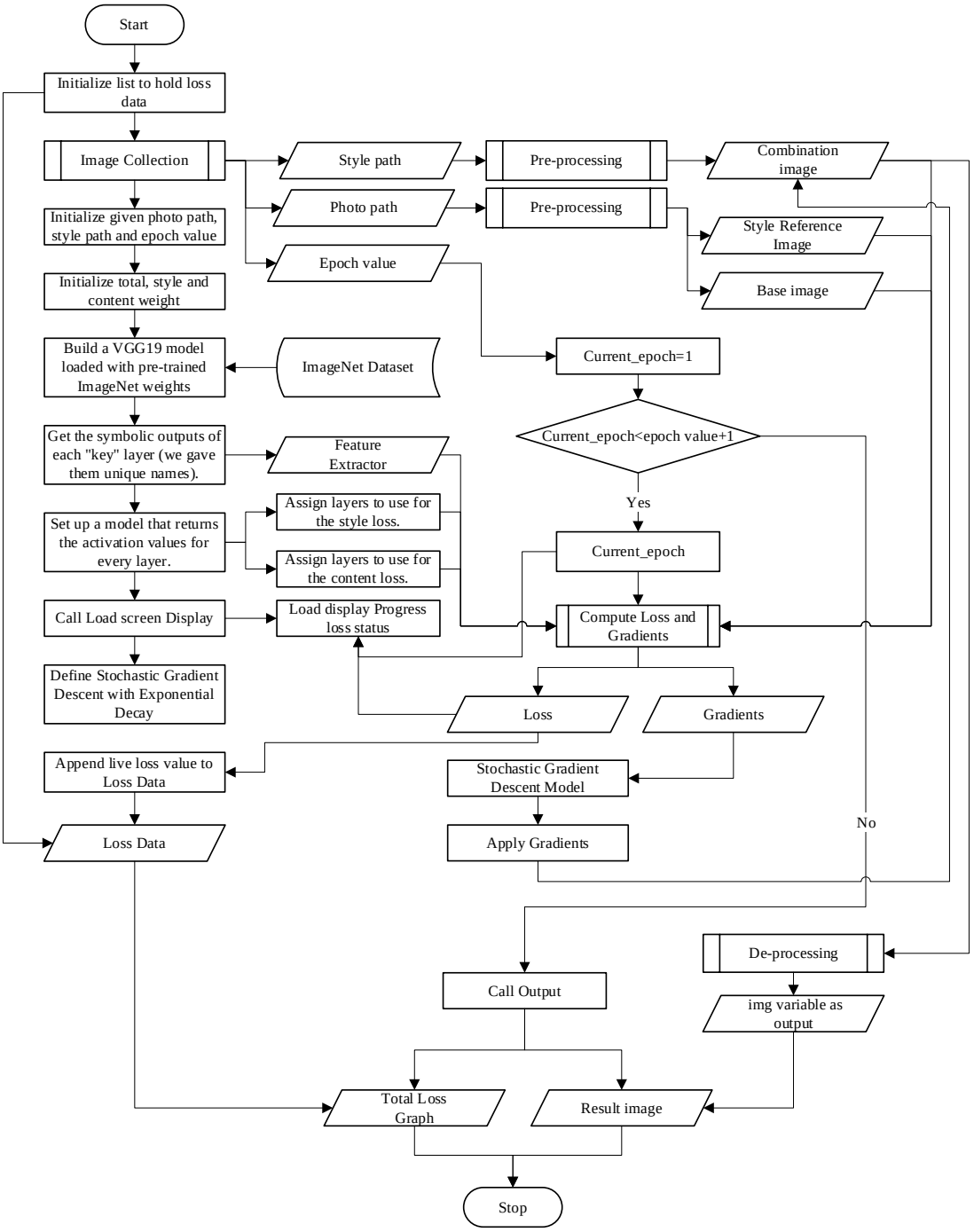
   Where, α (alpha) and β (beta) are weighting factors for the content and style loss, respectively.

4. **Compute Gradients:** Compute the gradients of the total loss with respect to the pixel values of the generated image.

   Formula
   Gradients = d(TotalLoss) / d(GeneratedImage)

   Where, GeneratedImage is the image being generated and the differentiation of TotalLoss with respect to GeneratedImage i.e., d(TotalLoss) / d(GeneratedImage) is the gradient of the total loss with respect to the pixel values of the generated image.

5.5.5 PIPELINE FOR PROPOSED SYSTEM (STYLEGEN APPLICATION)



5.6 VGG-19 MODEL ARCHITECTURE

5.6.1 LAYERED VIEW

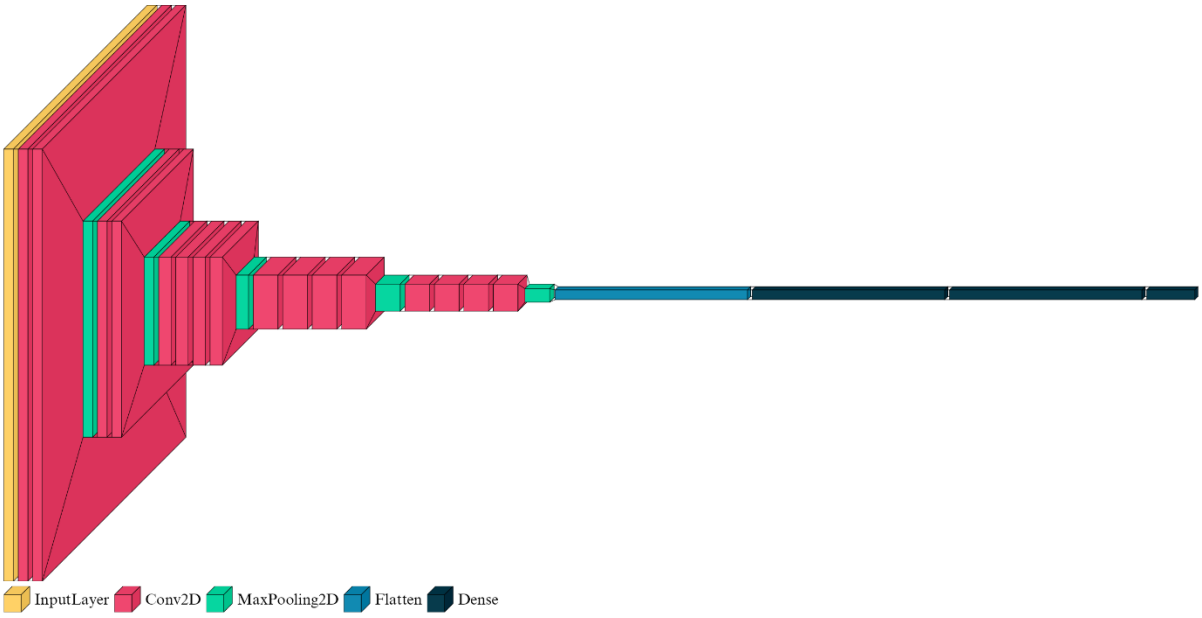


Figure 5: VGG-19 Neural Network Layered View

5.6.2 STYLE AND CONTENT LAYERED VIEW

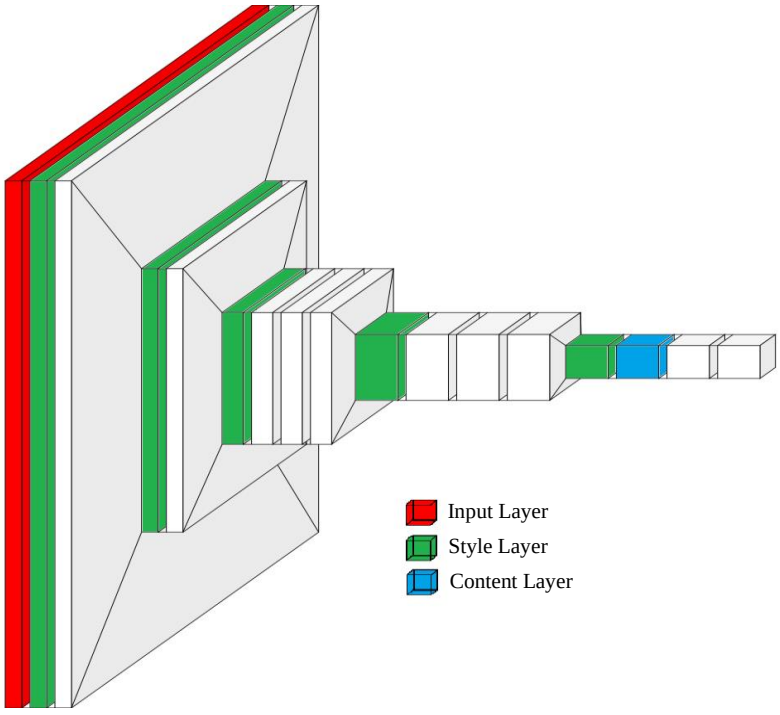


Figure 6: VGG-19 Layered View separated for processing

5.6.3 MODEL SUMMARY

```
Model: "vgg19"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 224, 224, 3)]     0

Style Layers

block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792

block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856

block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168

block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160

block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808

Content Layer

block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808


=================================================================
Total params: 6,270,592
Trainable params: 6,270,592
```

5.6.4 MODEL ACCURACY

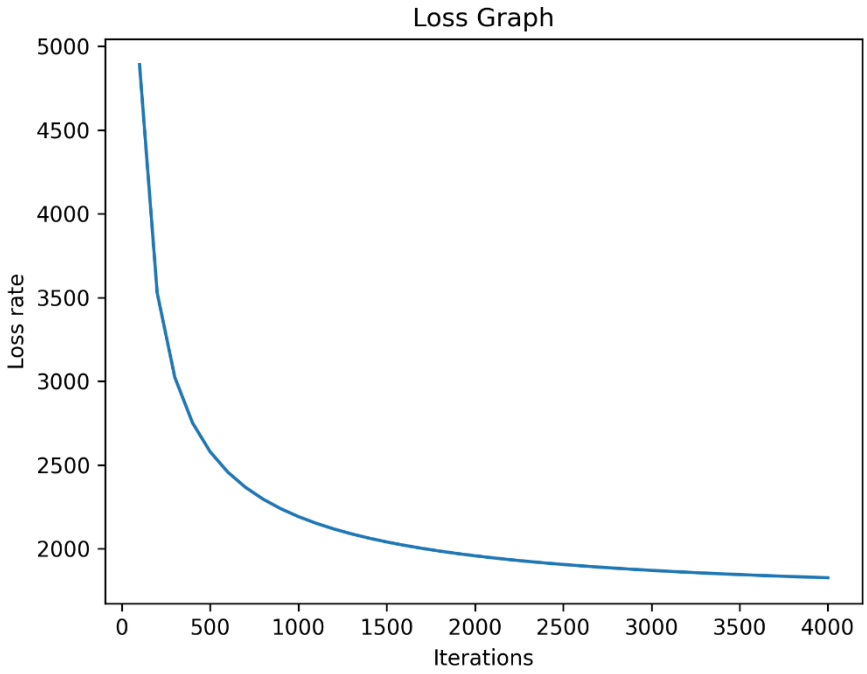


Figure 7: Total Loss Graph example for given input

5.7 INPUT DESIGN

Input is the process of converting user inputs computer-based format. The project requires a set of information from the user to prepare a report. In the order, when organized input data are needed. Input data is collected and organized into groups of similar data. The goal behind designing input data is to make the data entry easy and make it free from logical error. So, the input screens in the system should be really flexible and faster to use. The user need not to input any data manually, we are using the video data directly from the web camera.

Objectives: -

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understandable.
- To make clutter free screens.
- The prevention of irrelevant data entry.
- To make a user-friendly input screen.

Here in our system, 'StyleGen: Image Style Transfer using VGG-19', input screens ensure the reliability and accuracy of the system. The input design determines whether the user can interact directly with the computer. With input design, we can say that it is more user friendly as compared to the existing manual system containing paper operations.

5.8 OUTPUT DESIGN

Outputs are the most important direct source of information to the user. Efficient and eligible output design should improve the system's relationship with the user, Output design generally deals with the results generated by the system i.e., the styled image output which can be downloaded and the total loss graph for analysing the performance of the model. The end users will not actually operate the interior model of system, but they will use the output from the system.

CHAPTER 6

FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

6.1 FUNCTIONAL REQUIREMENT

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as set of inputs, the behaviour, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Generally functional requirements are expressed in the form of "system must do requirement".

6.2 NON-FUNCTIONAL REQUIREMENT

A non-functional requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. Some of the non-functional requirements are mentioned below.

Performance: The system shall provide an ability to perform its tasks efficiently and quickly. It includes metrics like response time, throughput, and resource utilization.

Usability: The system shall have a clean interface with only needed features, clear terminology and tools tips where necessary. Letters for corresponding sign shall be specified in clear way.

Security: The system shall provide an ability to protect sensitive data and prevent unauthorized access. It includes metrics like data security and access controls.

Efficiency: The system shall respond to different sign languages with ease and effective manner.

Portability: The system shall be independent of the specific technological platform used to implement it.

Reliability: Reliability defined as a measure of the time between failures occurring in system, so that the system shall operate without any failures for a particular period of time.

Maintainability: The system shall provide an ability to be updated, repaired, and enhanced over time. It includes metrics like code quality, modularity, and documentation.

Availability: Availability measures the percentage of time the system is in its operational state so that the system be available for use 24 hours per day and 365 days per year.

CHAPTER 7

TESTING

7.1 TESTING STRATEGIES

Software Testing is the process of executing a program or system with the intent of finding errors. Testing involves any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. The scope of software testing includes examination of code as well as execution of that code in various environments and conditions as well as examining the quality aspects of code: does it do what it is supposed to Do and do what it needs to do. Testing helps not only to uncover errors introduced during coding, but also locates errors committed during the previous phases.

Testing Objectives Include:
- Testing is a process of executing a program with the intent offending an error.
- A good test case is one that has a probability of finding an as yet undiscovered error.

Testing Principles:
- All tests should be traceable to end user requirements
- Tests should be planned long before testing begin
- Testing should begin on a small scale and progress towards testing in large
- Exhaustive testing is not possible.
- To be most effective testing should be conducted by an independent third party.

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage the main workload, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion.

Testing helps not only to uncover errors introduced during coding, but also locates errors committed during the previous phases. Thus, the aim of testing is to uncover requirements, design or coding errors in the program. Software Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding, Testing present interesting anomaly for the software engineer.

7.2 UNIT TESTING

This is the first of testing. In this different module are tested against the specification produces during the design of the modules. It refers to the verification of single program module in an isolated environment. Unit testing focuses on the modules independently of one another to locate errors.

In our project we test each module and each forms individually. Each form has tested using appropriate values. The input screens need to be designed very carefully and logically. While entering data in the input forms, proper validation checks are done.

7.3 INTEGRATION TESTING

Integration testing (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

In our project we used incremental top-down approach in which, testing is done by integrating two or more modules that are logically related to each other and then tested for proper functioning of the application. Then the other related modules are integrated incrementally and the process continues until all the logically related modules are integrated and tested successfully i.e., from testing is done from prediction of model to image capturing by integrating on each iteration.

7.4 SYSTEM TESTING

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.

System testing takes, as its input, all of the integrated components that have passed integration testing. The purpose of integration testing is to detect any inconsistencies between the units that are integrated together. System testing seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. The actual result is the behaviour produced or observed when a component or system is tested.

System testing is performed on the entire system in the context of either functional requirement specifications (FRS) or system requirement specification (SRS), or both. System testing tests not only the design, but also the behaviour and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software or hardware requirements specifications.

7.5 TESTING RESULTS

TEST CASE ID	TEST CASE NAME	TEST CASE STEP	EXPECTED RESULT	STATUS	DEFECTS
1	Image loading and pre-processing module	This module will load the content and style images, pre-process them to prepare them for input into the VGG-19 network, and convert them to tensor format.	Convert content image and style image into tensor and make them fit for VGG model inputs and takes number of epochs from end user.	PASS	NIL
2	VGG-19 feature extraction module	This module will use the VGG-19 network to extract features from both the content and style images.	Layer separation for style and content loss calculation and working of feature extraction module.	PASS	NIL
3	Loss calculation module	This module will calculate the total loss for the image style transfer. The total loss will be composed of a content loss term and a style loss term, both of which will be computed using the features extracted by the VGG-19 network.	Gram matrix, Style loss, Content Loss, Total validation loss and Gradient calculation is same as per requirements.	PASS	NIL
4	Optimization module	This module will use an optimization algorithm (such as stochastic gradient descent) to minimize the total loss and generate the stylized output image.	Generation of styled image from calculated gradients and total loss in each iteration.	PASS	NIL

5	De-processing module	This module will perform any necessary postprocessing on the output image, such as converting it back to image format and rescaling it to the original size. It may also display the output image or save it to disk. And Draw the Total Loss graph	Storing total loss value in a list, working of de-processing module i.e., convert tensor output to image, output image needs to be downloaded and view total loss graph.	PASS	NIL

CHAPTER 8

RESULTS AND DISCUSSION

8.1 RESULTS (SALIENT FEATURES)

The objective of the proposed system is to generate a styled image from content image and style image using VGG-19 neural network model. We make this possible with the help of Python libraries like Streamlit, Tensor Flow, Keras, etc... At first, we need to properly intake content image, style image and epoch range from end user and need to pre-process properly.

- User interface is designed such that they are very user friendly and the user is not required to input data manually.
- Not much training required.
- Easy analysis of data and statistical view.
- There are no needs of experts.
- The new system is more user friendly.
- This system is much faster and efficient than the old system.
- Lower points failure than the old system.
- It is cheaper than the existing system.
- Easy to implement on any system.
- Improving the performance configuration of a system can decrease the time required to generate output.

8.2 SCREENSHOTS
8.2.1 DESKTOP VIEW

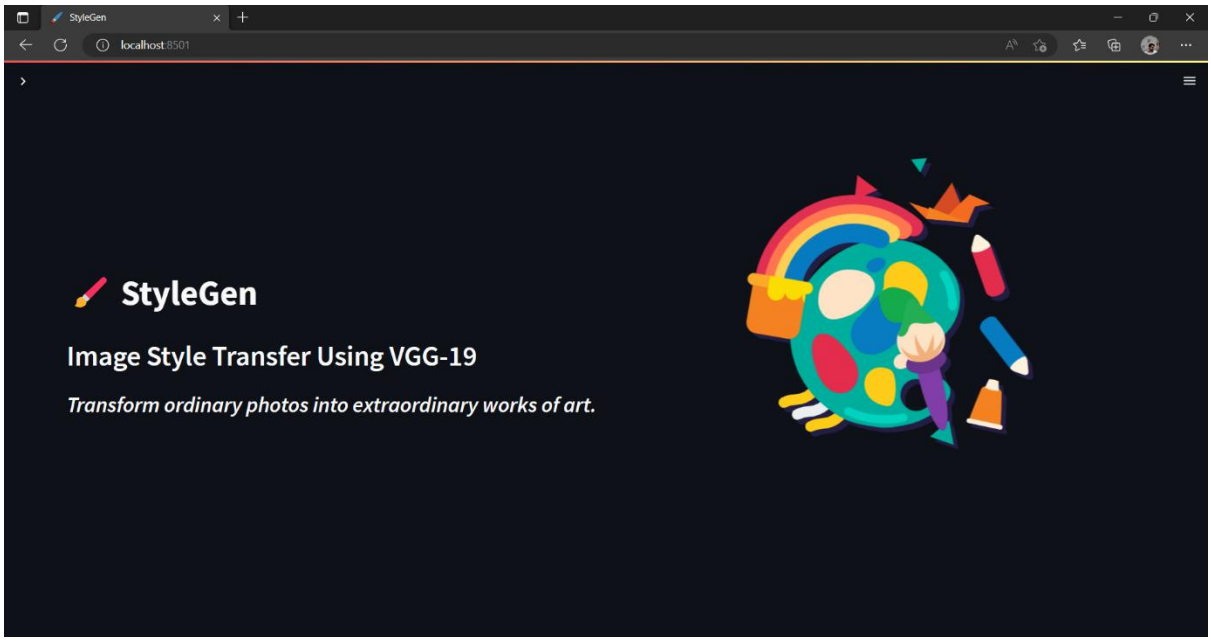


Figure 8: Home page of StyleGen

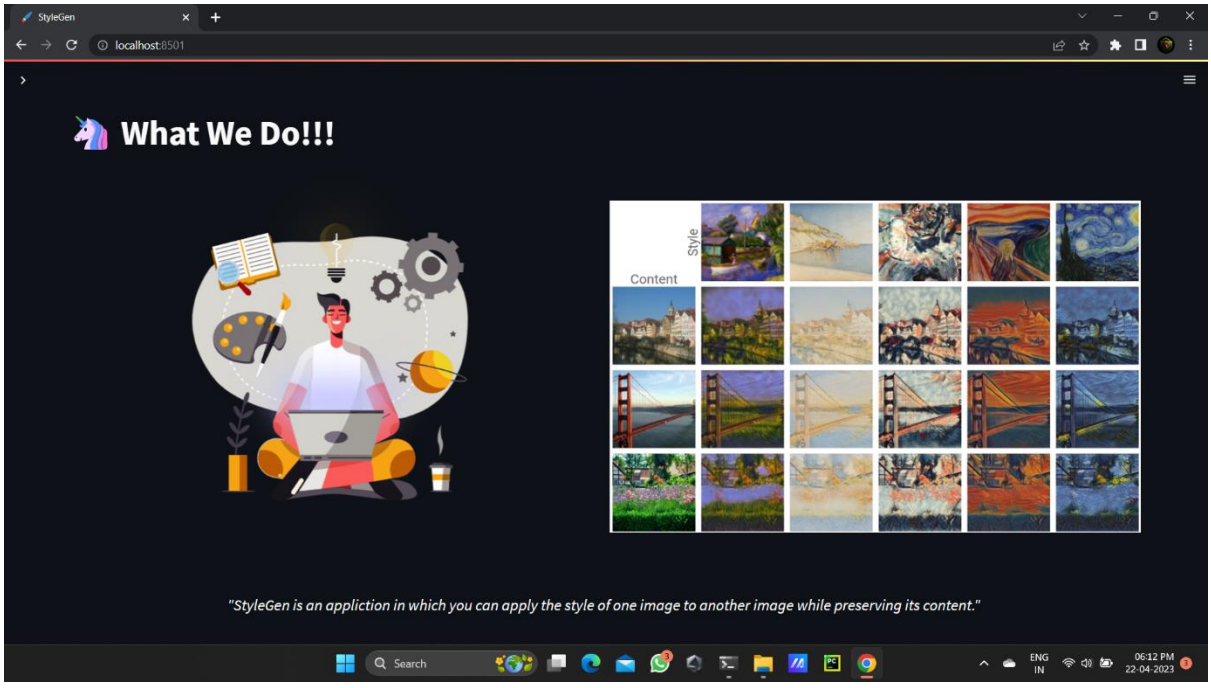


Figure 9: Home page of StyleGen ("What We Do" Section)

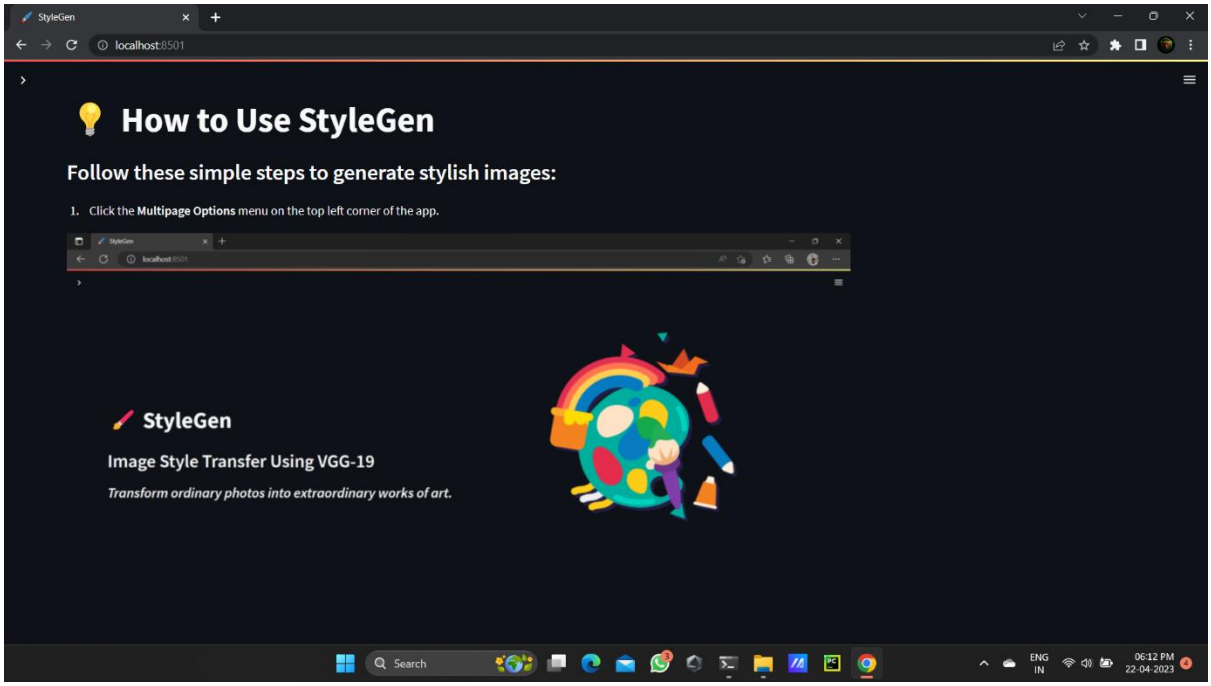


Figure 10: Home page of StyleGen ("How to Use StyleGen" Section)

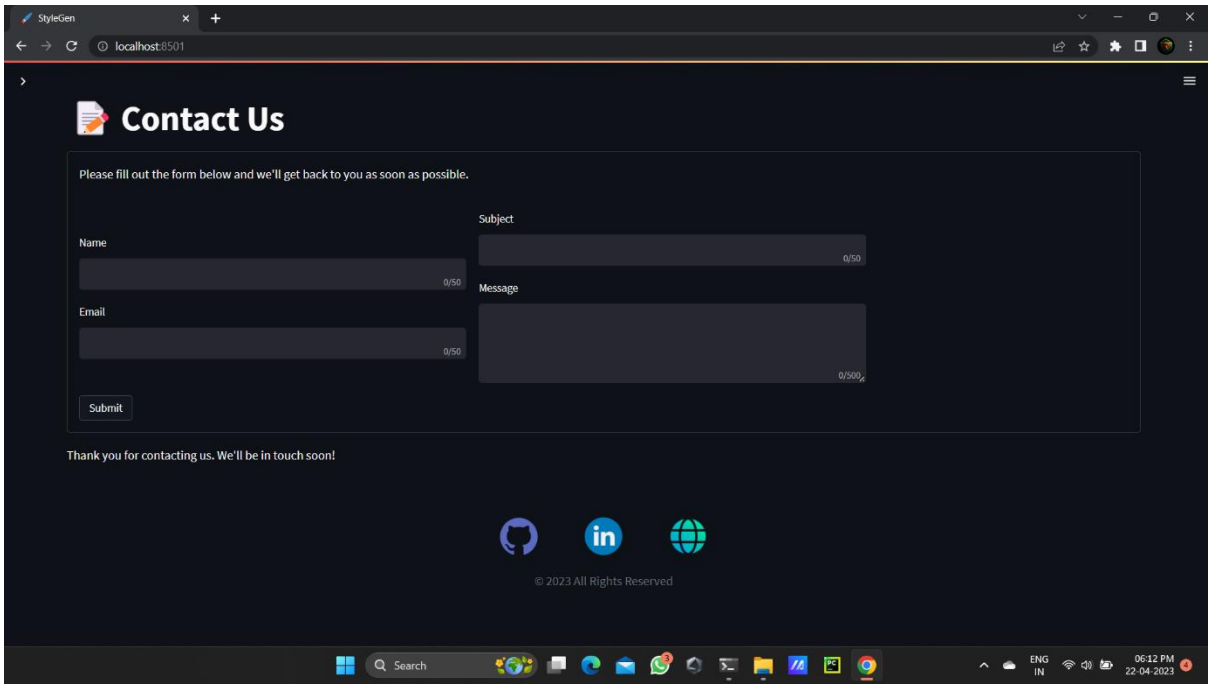


Figure 11: Home page of StyleGen ("Contact Us" Section and "Footer" Section)

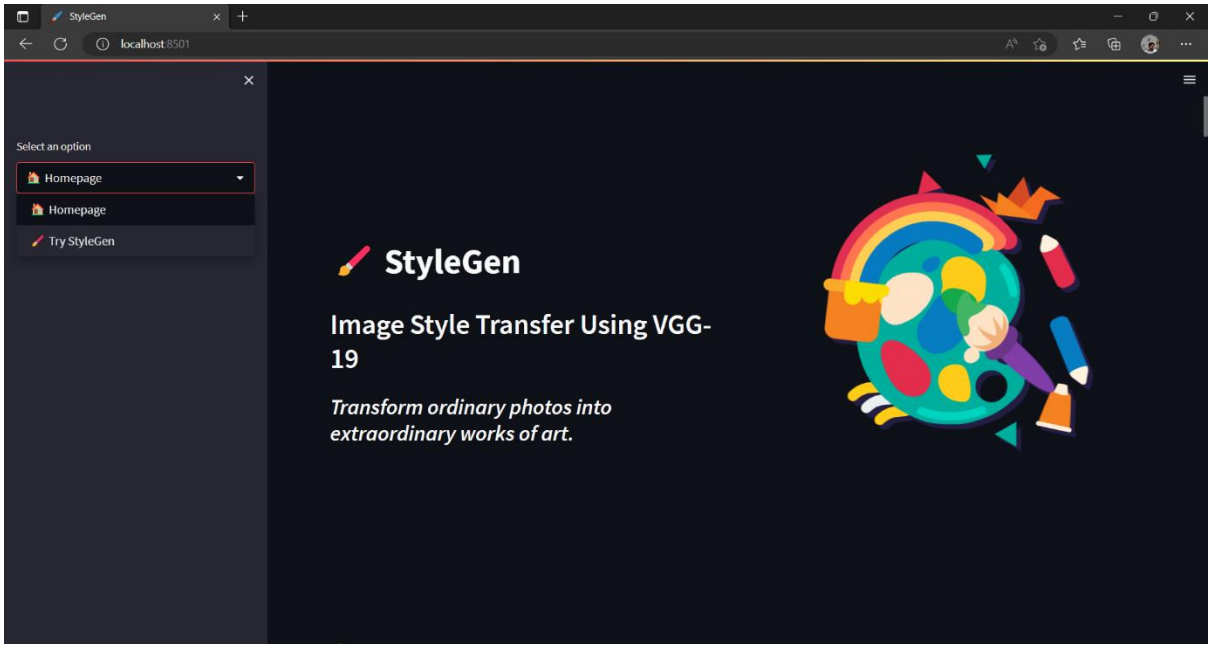


Figure 12: StyleGen Application (Multi-page Menu)

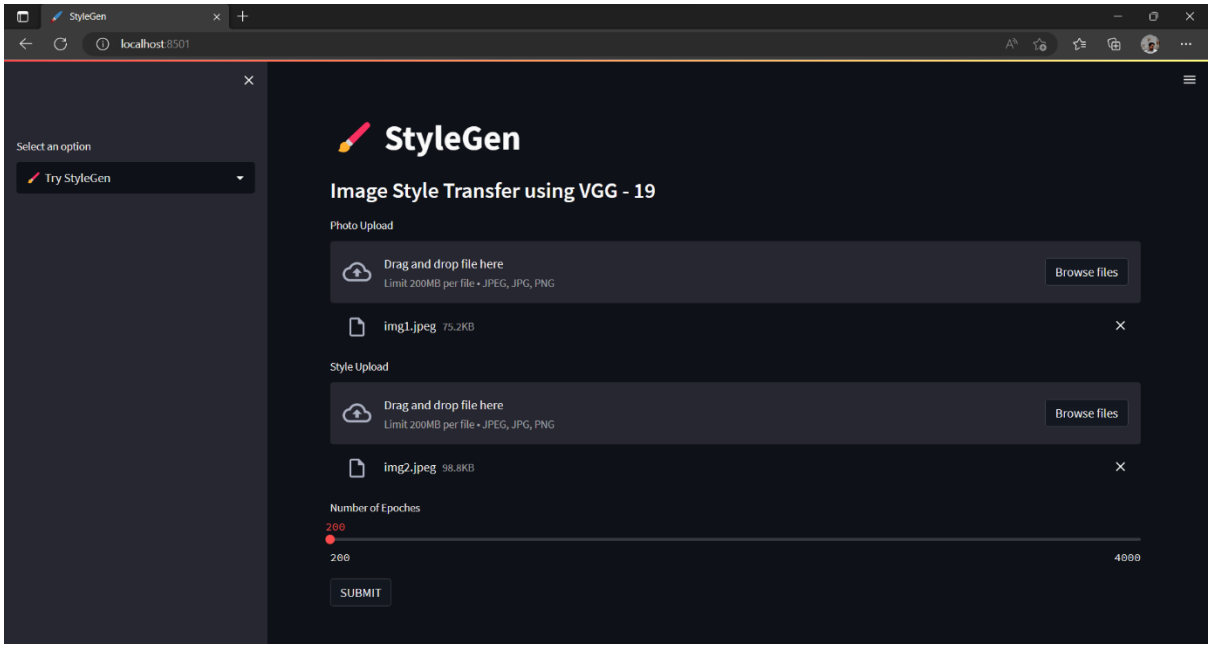


Figure 13: StyleGen Application ("Input" Section)

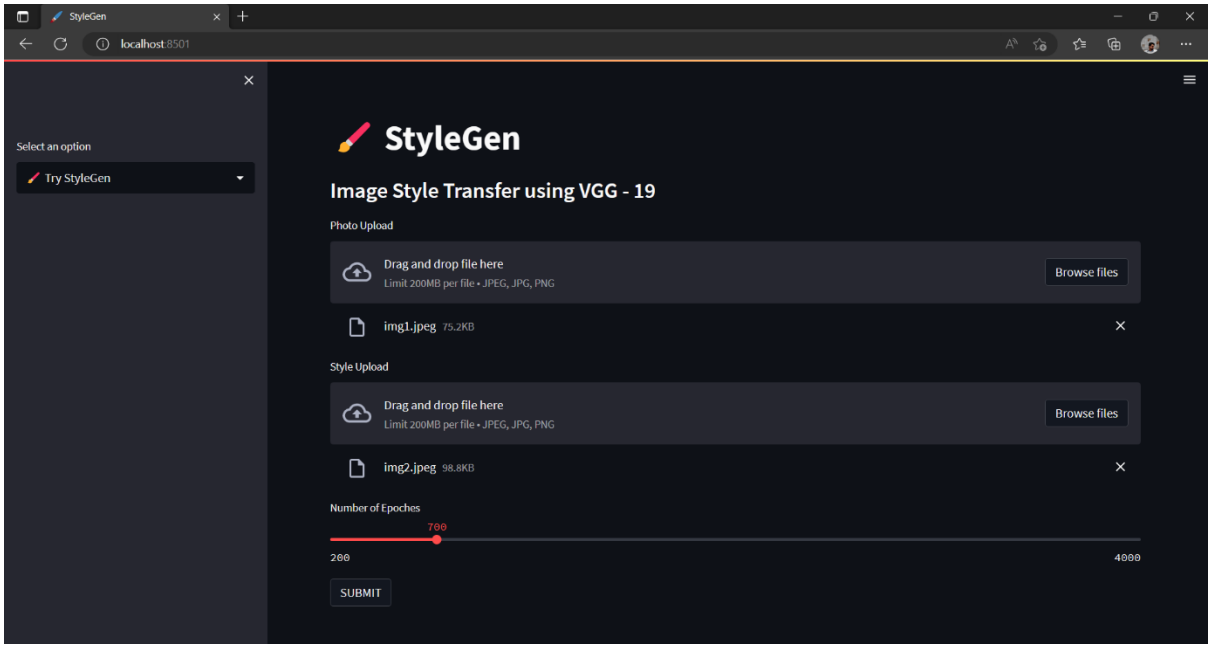


Figure 14: StyleGen Application ("Input" Section setting parameters)

Figure 15: StyleGen Application ("Processing" Section)

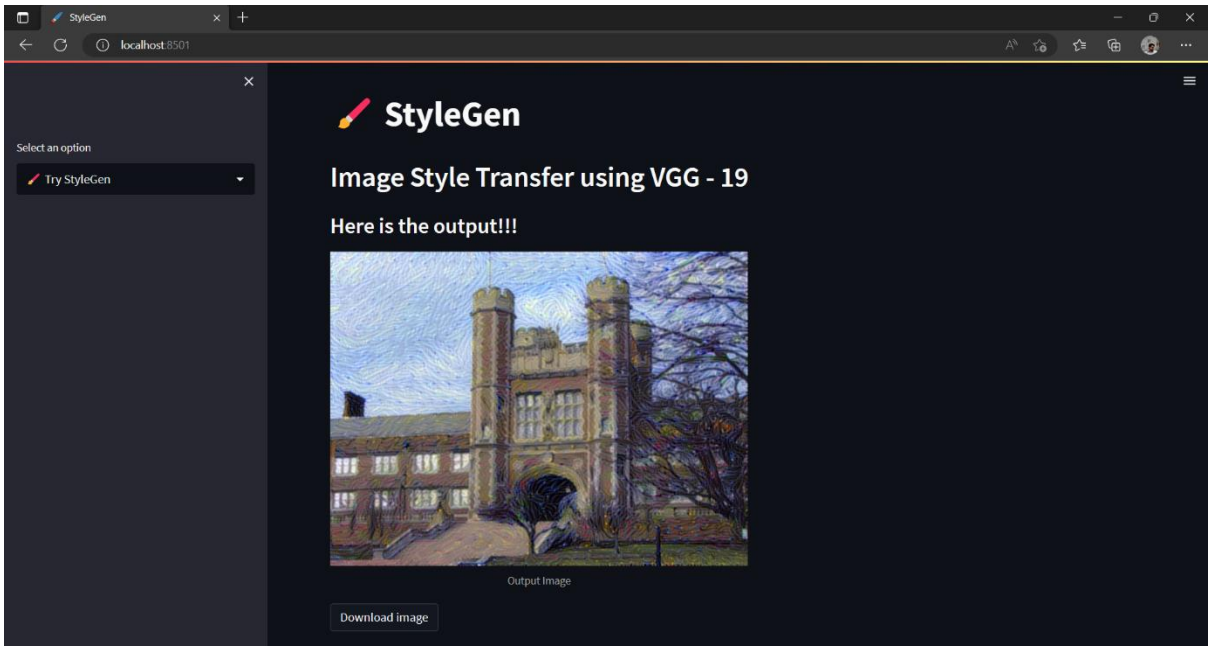


Figure 16: StyleGen Application ("Output" Section)

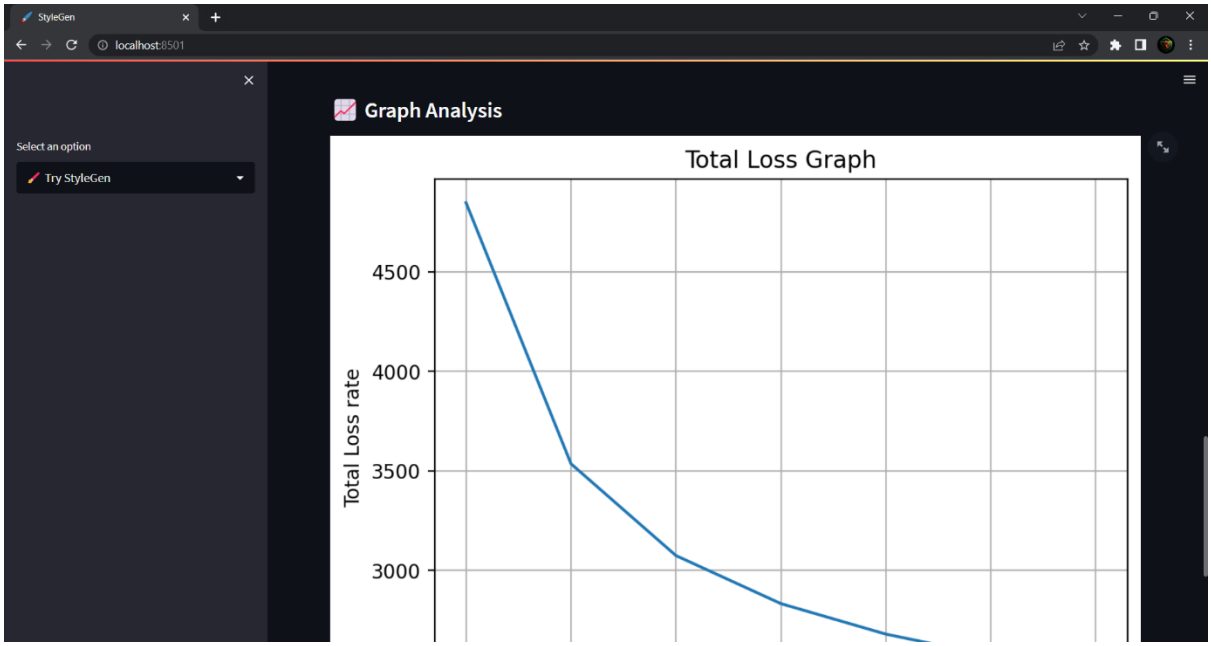


Figure 17: StyleGen Application ("Graph analysis" Section)

8.2.2 MOBILE VIEW

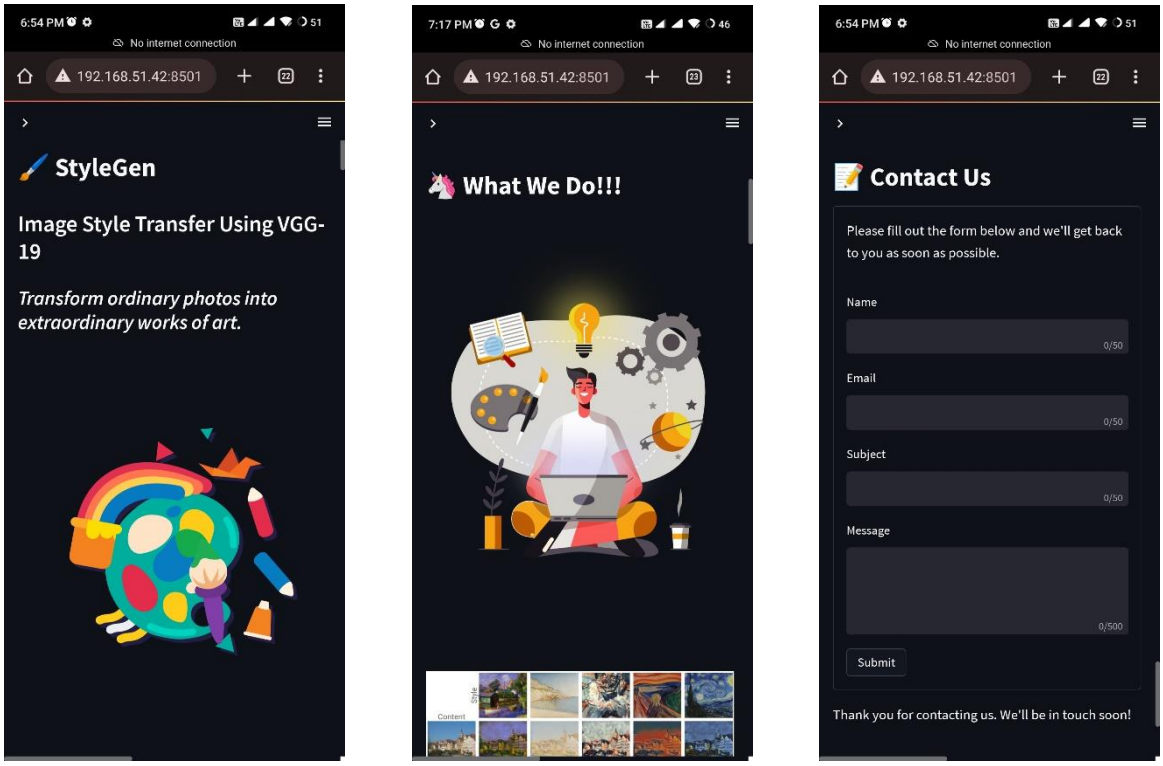


Figure 18: Home page of StyleGen (Mobile View)

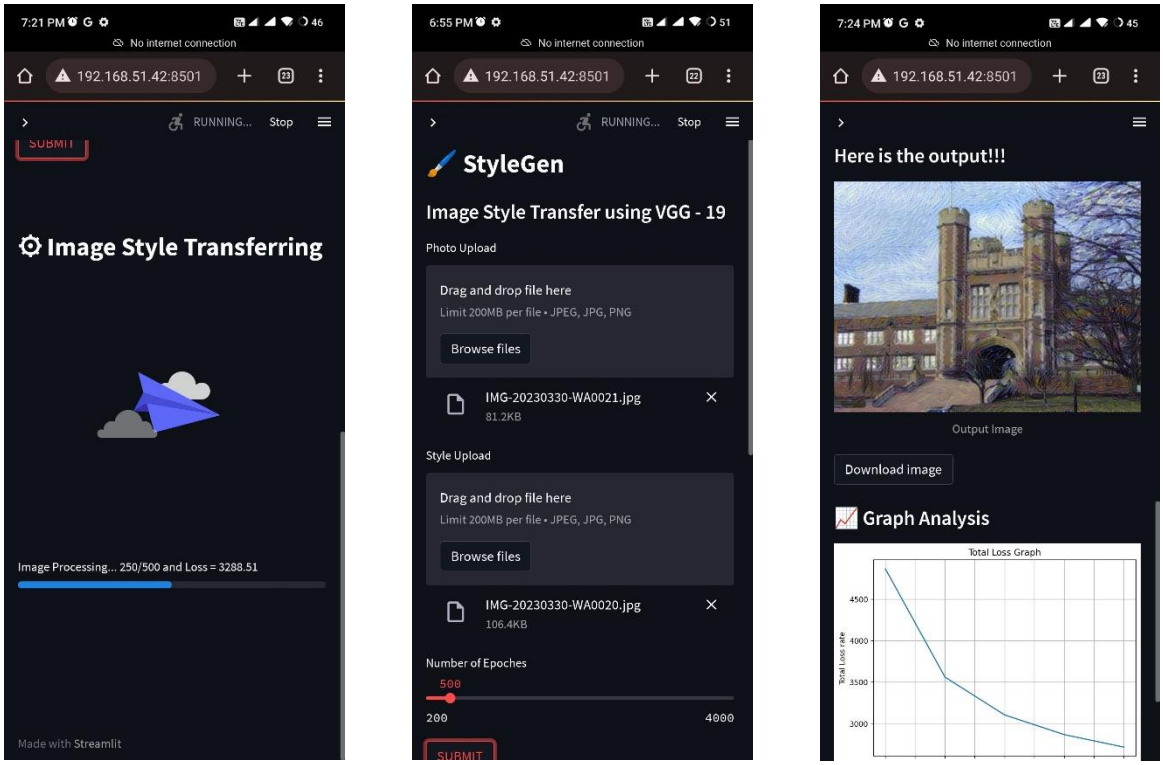


Figure 19: StyleGen Application (Mobile View)

CHAPTER 9

CONCLUSION

9.1 SYSTEM IMPLEMENTATION

After the system has been tested, the implementation type or the changeover technique from the existing system to the new system is a step-by-step process. In the system at first only a module of the system is implemented and checked for suitability and efficiency. When the end user related to the particular module is satisfied with the performance, the next step of implementation is preceded.

Backups are necessary since any time unexpected events may happen. And so, during the program execution, the records are stored in the workspace. This helps to recover the original status of the records from any accidental updating or intentional deletion of records.

An Implementation plan is a management tool for a specific policy measure, or package of measures, designed to assist agencies to manage and monitor implementation effectively. Implementation plans are intended to be scalable and flexible; reflecting the degree of urgency, innovation, complexity and or sensitivity associated with the particular policy measure. Agencies are expected to exercise judgment in this area; however, the level of detail should be sufficient to enable the agency to effectively manage the implementation of a policy measure. At a minimum, plans should reflect the standards outlined in the Guide to Preparing Implementation Plans.

The implementation stage involves following tasks:

- Careful planning
- Investigation of system and constraints
- Design of method to achieve the changeover phase

9.2 CONCLUSION

Deep learning helps computers to derive meaningful links from a plethora of data and make sense of unstructured data. Here, the mathematical algorithms are combined with a lot of data and strong hardware to get qualified information. With this method, information from digital data can be automatically extracted, classified and analysed.

Although deep learning has been around for several years, the trend has only really picked up in the last three to four years. The reason for this was among other things better hardware resources, more sophisticated algorithms and optimized neural networks. Deep learning is not a new approach but a development of the older approach of artificial neural networks.

This project proposes a method that primarily employs a Convolutional Neural Network (VGG-19) instead of basic classification techniques. By utilizing a stochastic gradient descent model to compute loss and gradients, the method effectively achieves StyleGen: Image Style Transfer using VGG-19. This approach can be applied to various editing tasks across different growing industries.

9.3 FUTURE SCOPE

Our proposed system is simple, flexible, and easily customizable, allowing for numerous enhancements. For example:

- Integrating the StyleGen application with the DALL-E 2 architecture could unlock additional features.

- Establishing a structured system with a hierarchy of personnel to oversee and provide necessary services could help maintain harmony within the system, rather than relying on a standalone system as it is now.

- Implementing a profile system to track work history.

- Developing an online community to foster and promote the growth of the application. This could be achieved by creating a website for our application to generate and implement new ideas, ensuring the system remains reliable and enduring.

These enhancements could be introduced as future updates to the system.

BIBLIOGRAPHY

BOOKS

[1] Convolutional neural networks for visual computing (Chapter 4), Ragav Venkatesan and Baoxin Li CRC press

[2] Online book Dive Deep into Machine Learning at https://d2l.ai/

[3] E. Alpayidin, Introduction to Machine Learning, Prentice Hall of India (2005)

[4] Jeeva Jose, "Taming Python by Programming", Khanna Publishers, New Delhi, 2018

WEBSITES

[5] https://www.tensorflow.org/

[6] https://docs.streamlit.io/

[7] https://keras.io/

JOURNAL AND PUBLICATIONS

[8] Image Style Transfer Based on VGG Neural Network Model;2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA); IEEE; 20-21 August 2022; Dalian, China
https://ieeexplore.ieee.org/document/9918891

[9] Style Transfer with Generative Adversarial Networks; Seconda Sessione di Laurea Anno Accademico 2017 – 2018; Prof. Davide Maltoni, Gabriele Graffieti

[10] Zhang, Yuxin, et al. "Domain enhanced arbitrary image style transfer via contrastive learning." *ACM SIGGRAPH 2022 Conference Proceedings*. 2022.

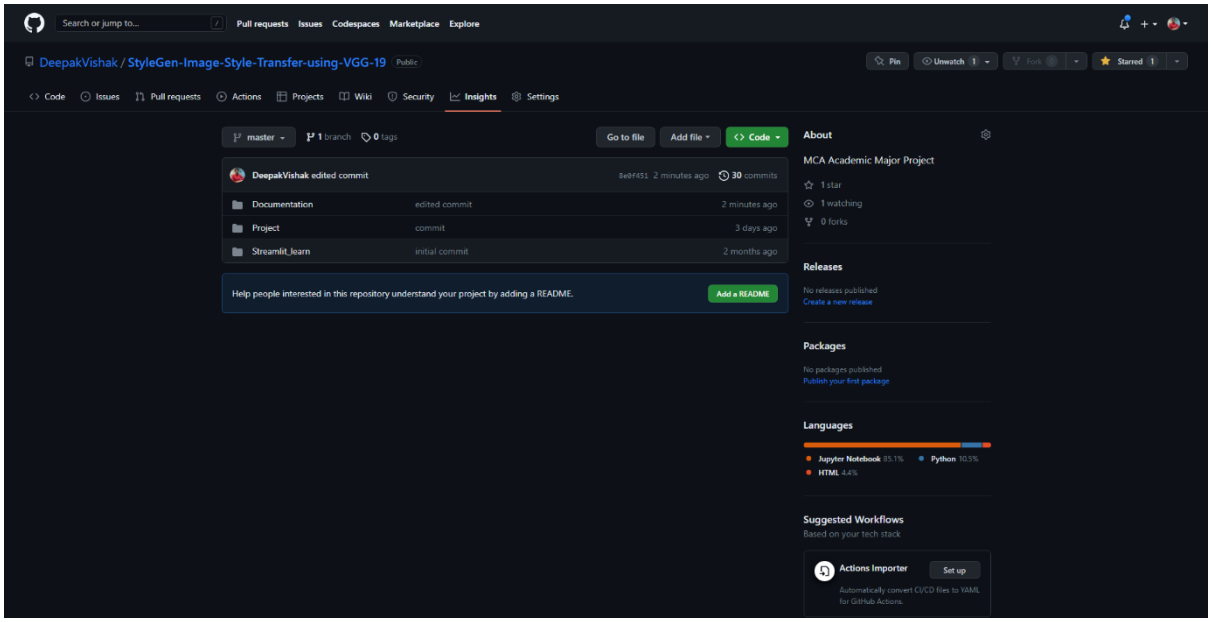
APPENDICES

- **LIST OF TABLES**

- **LIST OF FIGURES**

GIT HISTORY


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
DeepakVishak / StyleGen-Image-Style-Transfer-using-VGG-19
Public
Pin
Unwatch 1
Fork
Starred 1
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Pulse
Contributors
Community
Community Standards
Traffic
Commits
Code frequency
Dependency graph
Network
Forks
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
© 2023 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training
Blog
About


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
DeepakVishak / StyleGen-Image-Style-Transfer-using-VGG-19
Public
Pin
Unwatch 1
Fork
Starred 1
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
master
1 branch
0 tags
Go to file
Add file
Code
DeepakVishak edited commit
30 commits
Documentation
edited commit
2 minutes ago
Project
commit
3 days ago
Streamlit_learn
initial commit
2 months ago
Help people interested in this repository understand your project by adding a README.
Add a README
About
MCA Academic Major Project
1 star
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
Jupyter Notebook 85.1%
Python 10.5%
HTML 4.4%
Suggested Workflows
Based on your tech stack
Actions Importer
Set up
Automatically convert CI/CD files to YAML for GitHub Actions.