Neural Style Transfer with VGG19

Overview

Neural Style Transfer (NST) is a deep learning technique that applies the artistic style of one image to another while retaining its content. This project implements NST using the VGG19 model in TensorFlow, enabling the generation of stylized images by blending content and style features.

Requirements

- **Python 3.x**: The primary programming language used.
- **TensorFlow**: A deep learning framework for handling neural networks.
- **NumPy**: For numerical operations and array manipulations.
- **Matplotlib**: To visualize and display images.
- **Pillow (PIL)**: For image processing and manipulation.

Project Files

The project consists of several key files:

- **Neural_style_transfer.py**: The main Python script that performs the style transfer.
- **wallpaper.png**: The content image (background image that retains its structure).

- **img.png**: The style image (artistic image that provides the style features).

- **output_image.jpg**: The final output image after applying Neural Style Transfer.

How Neural Style Transfer Works

Neural Style Transfer works by combining the features of two images: the **content image** and the **style image**. It does this using a deep convolutional neural network, specifically the **VGG19 model**, pre-trained on the ImageNet dataset.

Step 1: Load and Preprocess Images

- Convert images to **RGB format**.
- Resize images to **224x224 pixels** for model compatibility.
- Normalize pixel values to improve neural network performance.

Step 2: Extract Features Using VGG19

The **VGG19 model** is a deep convolutional network trained on millions of images. It captures both **content** and **style** features at different layers:

- **Content is extracted from** block5_conv2.
- **Style is extracted from** multiple layers (block1_conv1, block2_conv1, etc.).

Step 3: Compute the Gram Matrix for Style Loss

The **Gram Matrix** measures correlations between feature maps to capture the style of an image.

Mathematical Representation:

- **Gram Matrix (G) Calculation:**

where **A** is the feature map matrix.

Step 4: Compute Loss Function

NST uses two primary losses:

1. **Content Loss**: Ensures that the generated image maintains the original content.
2. **Style Loss**: Ensures that the generated image captures the artistic style of the reference image.

The **total loss** is a weighted sum of the two losses:

where and are weight factors.

Step 5: Optimize the Image

To generate the final image, **gradient descent** is used to minimize the total loss.

The final output image is a blend of the content from **wallpaper.png** and the artistic style of **Image.png**.