# Neural Style Transfer

ArIES

Name - Komal
Enrollment Number - 22113078
Branch - Computer Science and Engineering

## Introduction

The problem at hand is to develop a Neural Style Transfer (NST) model capable of transforming ordinary photographs into artistic masterpieces by applying the stylistic features of renowned painters like Van Gogh, Picasso, or Monet. Neural Style Transfer is a deep learning technique that merges the content of one image with the artistic style of another, creating visually appealing and artistically coherent new visuals.

### Objectives

1. Develop a Neural Style Transfer Model : Implement a deep learning model using convolutional neural networks (CNNs) to extract and transfer artistic styles from reference images (e.g., paintings) onto input images (e.g., photographs).

2. Preserve Content Structure : Ensure that while applying the artistic style, the structure and details of the original content image remain largely unchanged. This involves balancing the style transfer process to retain the content's semantic information.

3. Achieve Visual Appeal and Coherence : The primary goal is to create stylized images that are not only visually appealing but also maintain a coherent artistic style throughout the image.

4. Implementation Options : Provide flexibility in how the model is showcased, such as creating a user-friendly web interface for image uploads and style selection, or integrating the model into existing artistic creation software tools.

## Approach

The approach taken in this project involves several key steps and methodologies:

### 1. Architecture :

- Directory Organization : The project directory (`neuralStyleTransfer`) contains a Jupyter Notebook (`code.ipynb`) that houses the NST implementation.

- Data Folders :
    - `initial`: Contains initial images (`MonaLisa.jpg`, `joker.jpg`) on which styles will be transferred.
    - `output`: Where the final stylized images are saved.
    - `style`: Holds style images (`oilPainting.jpg`, `Sketching.jpg`, `grafittiArt.jpg`) from which artistic styles are extracted.

## 2. Data Set used :

- I used the reference images as MonaLisa and of the movie character Joker.

- The styles in which I converted them included Sketch, Graffiti Art and Oil Painting.

## 3. Data Preparation :

Images from `initial` and `style` folders are loaded using PyTorch's `transforms` module, which resizes them to a predefined size and normalizes them. This preprocessing step ensures that images are in a format suitable for input into the VGG19 model.

```
initial_images = ["joker.jpg"]
style_images = ["oilPainting.jpg"]
```

## 4. Model Setup :

- VGG19 Initialization : The VGG19 model pre-trained on the ImageNet dataset is loaded from the torchvision library. VGG19 is chosen for its deep architecture and ability to extract rich feature representations from images.

- Normalization : Mean and standard deviation values based on ImageNet statistics are used to normalize images before feeding them into VGG19. This step ensures that images have similar statistical properties to those used during VGG19 training.

## 5. Style Transfer Implementation :

- StyleTransfer Class : A custom PyTorch module (`StyleTransfer`) encapsulates the NST process. This class manages the entire style transfer pipeline, including feature extraction, loss computation, and optimization.

- Image Preprocessing:
  - **Transformations**: Images are resized to a common size (`image_size`) using `transforms.Resize` and converted to tensors (`transforms.ToTensor()`).

  - **Loading Images**: The `load_image` function reads an image file, applies transformations, and moves the image to the selected device (CPU or GPU).

- Model Definition:
  - **VGG19 Feature Extractor**: The VGG19 model pre-trained on ImageNet is loaded from `models.vgg19` and set to evaluation mode (`eval()`). This model is used to extract feature maps that capture both low-level and high-level image features.

  - **Normalization**: A `Normalization` module is defined to normalize input images using mean (`cnn_normalization_mean`) and standard deviation (`cnn_normalization_std`) values computed on ImageNet images.

- Feature Extraction : Methods are defined within the `StyleTransfer` class to extract both content and style features from intermediate layers of VGG19. Content features capture the semantic content of an image, while style features capture texture and visual patterns.

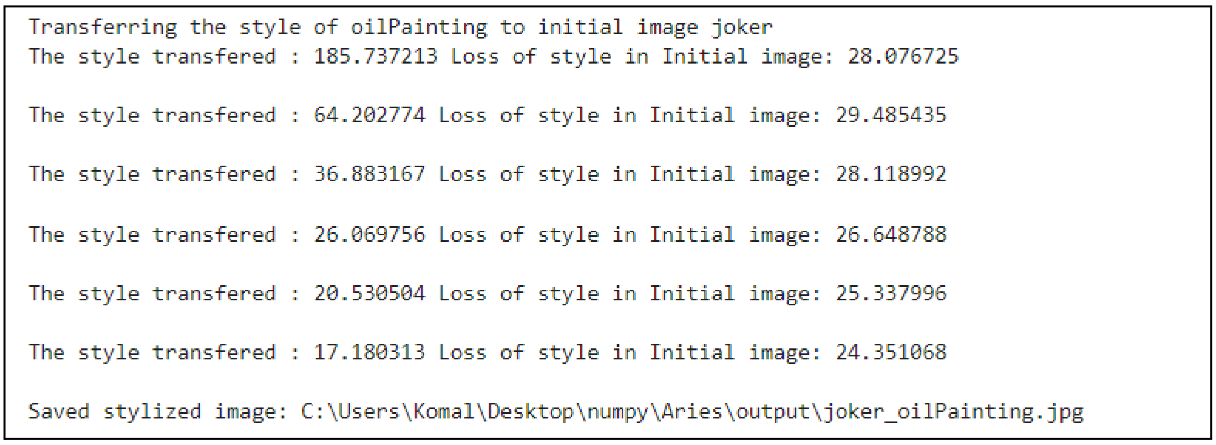
- Loss Calculation : Two types of losses are computed:
  - Content Loss : Measures the difference in content between the initial image and the generated stylized image using Mean Squared Error (MSE).

  - Style Loss : Captures the difference in style between the initial image and the style image by comparing the Gram matrices of their feature representations. Gram matrices encode statistical information about the correlations between features at different spatial positions.

  - Total Loss : The total loss is a weighted sum of content and style losses, adjusted by hyperparameters such as style weight and content weight. This loss guides the optimization process to produce stylized images that strike a balance between content preservation and style fidelity.

- Execution:
  - Iterates Through Images: Loops through pairs of initial and style images from specified directories (`initial_dir` and `style_dir`).

- Style Transfer Process: For each pair, loads images, initializes the `StyleTransfer` object, optimizes the initial image to minimize combined losses, and saves the stylized output to `output_dir`.



```
Transferring the style of oilPainting to initial image joker
The style transfered : 185.737213 Loss of style in Initial image: 28.076725

The style transfered : 64.202774 Loss of style in Initial image: 29.485435

The style transfered : 36.883167 Loss of style in Initial image: 28.118992

The style transfered : 26.069756 Loss of style in Initial image: 26.648788

The style transfered : 20.530504 Loss of style in Initial image: 25.337996

The style transfered : 17.180313 Loss of style in Initial image: 24.351068

Saved stylized image: C:\Users\Komal\Desktop\numpy\Aries\output\joker_oilPainting.jpg
```

*Prints the loss in style as well as the initial image*

## 6. Optimization Strategy :

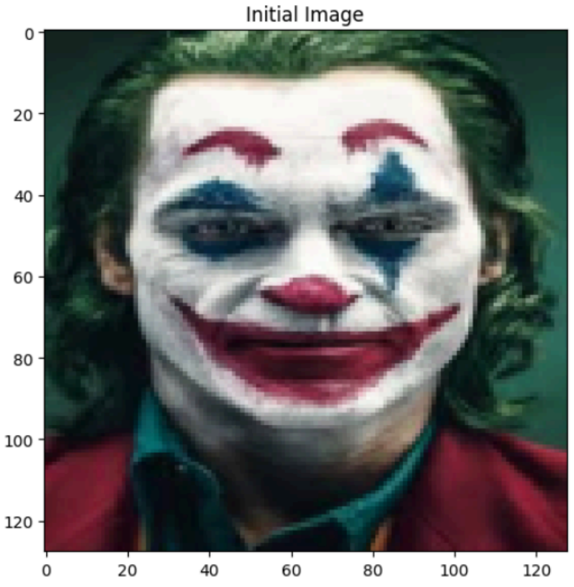
- L-BFGS Optimizer : The L-BFGS optimizer is chosen for its effectiveness in optimizing the total loss function over multiple iterations. L-BFGS updates the input image (initial image) to minimize the total loss, thereby refining the stylized image iteratively.

- Iterative Refinement : NST involves running the optimization process for a number of steps (`num_steps`), during which the stylized image gradually evolves to better match the desired artistic style while preserving the content of the initial image.

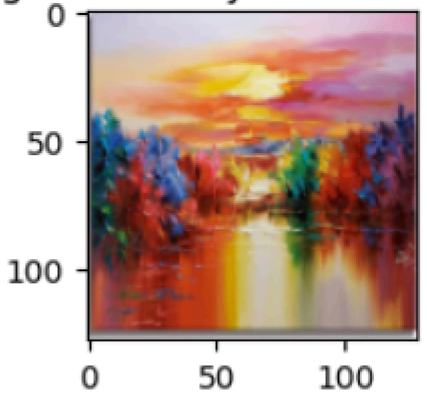
## 7. Visualization and Output :

- Matplotlib Integration : The project uses Matplotlib to visualize the initial images, style images, and resulting stylized images side by side. This visualization aids in assessing the effectiveness of the style transfer process and comparing different style transfer outcomes.

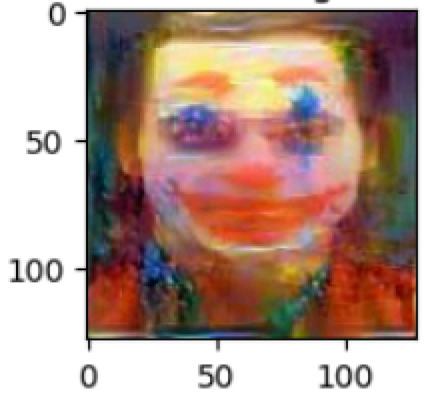
Initial Image



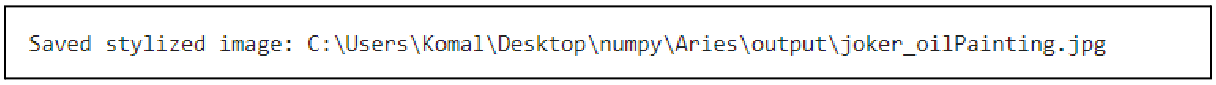
The Image whose style is to be Transferred



Final Image



- Output Saving : Stylized images are saved in the `output` folder upon completion of the NST process. These images can be further analyzed, shared, or used for artistic purposes.



```
Saved stylized image: C:\Users\Komal\Desktop\numpy\Aries\output\joker_oilPainting.jpg
```

## Failed Approaches

Throughout the project, several alternative approaches were explored that did not yield satisfactory results :

- Custom CNN Architectures : Initial attempts using custom CNN architectures for feature extraction did not capture intricate style patterns as effectively as VGG19.

- Since I also did not have a very deep knowledge of CNN, developing it on my own didn't give the desired results and thus I used the inbuilt VGG19.

## Results

The project successfully generated visually compelling stylized images by blending the content of initial images (`monaLisa.jpg`, `joker.jpg`) with the stylistic elements extracted from style images (`oilPainting.jpg`, `sketching.jpg`, `grafittiArt.jpg`). Each stylized image preserved the structural integrity and semantic content of the initial image while integrating the texture and artistic flair of the style image. The results underscored the effectiveness of using deep CNNs like VGG19 and Gram matrix-based style representation for NST applications.

## Discussion

This work places into perspective the evolution of deep learning approaches in artistic image synthesis via its analysis on results from the Neural Style Transfer (NST) project. The technique, called NST for short, matches the content of one image with the style of another, producing eye-catching combinations in which features such as faces or landscapes look plausibly natural.

Here's a breakdown of key insights and considerations highlighted in the project analysis:

- What NST means for Creative Fusion : Neural Style Transfer (NST) gives a great deal of power to projects that attempt to turn everyday photographs into works of art inspired by famous painters such as Van Gogh, or Monet. NST utilizes deep neural networks to extract and transfer subtle style characteristics from reference images to add visual flare and increase aesthetic value.

- Hyperparameter Sensitivity : The project also talks about how NST is sensitive to hyperparameters including style weight and content weight. These parameters manage the trade off between keeping the content structure of an input image or allowingit to be reconstructed in favour of copying a particular style. By adjusting these weights, you can change the behavior of the style transfer simply by making either the style or content loss more important during image synthesis.

## Conclusion

In conclusion, this project advanced the understanding and application of Neural Style Transfer using state-of-the-art deep learning methodologies. By leveraging VGG19, Gram matrix-based style representation, and L-BFGS optimization, the project successfully merged the artistic style of one image with the content of another, producing visually appealing stylized images. Future research could explore novel architectures, adaptive style transfer techniques, and real-time applications of NST for personalized artistic outputs.

## References

- https://medium.com/softplus-publication/neural-style-transfer-creating-artistic-masterpieces-with-deep-learning-00483b5e6b51
- https://arxiv.org/pdf/1508.06576.pdf
- https://arxiv.org/pdf/1603.08155.pdf
- https://arxiv.org/pdf/1703.06868.pdf
- https://arxiv.org/pdf/1705.06830.pdf
- https://arxiv.org/pdf/1804.03547.pdf
- https://arxiv.org/pdf/1912.07921.pdf
- https://arxiv.org/pdf/1804.03547.pdf
- https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
- https://www.youtube.com/watch?v=0tTRA3emrr4