Name: **Sai Aravind Reddy Katteragandla**
Email: skatteragandla@csuchico.edu

CSCI 611 - 01 Applied Machine Learning Spring 2025

Assignment 4: Style Transfer

Introduction

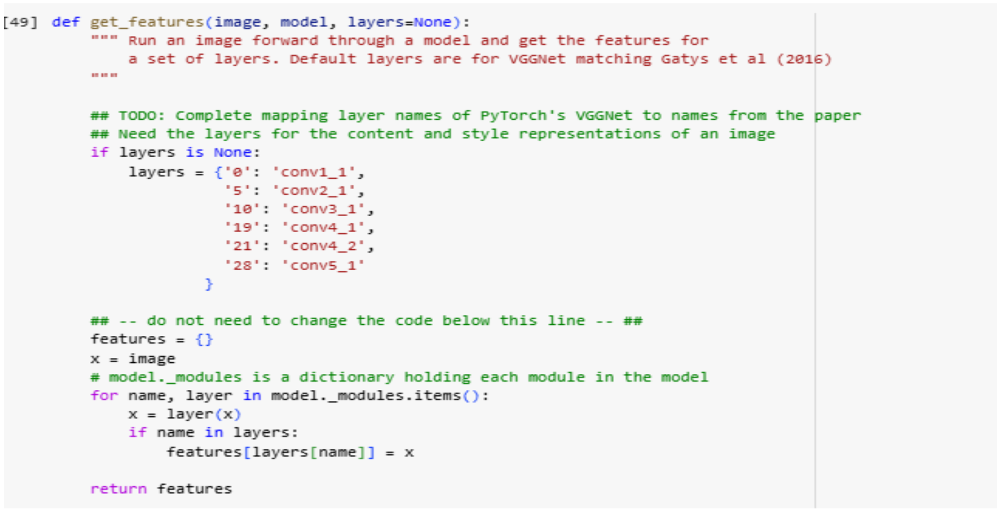
Style transfer is a deep learning technique for transferring the style of one image (e.g., artwork) to another's content (e.g., photograph). This project aims to apply using a pretrained VGG19 network to combine the content of one image with another's artistic style.

Reference Work

The implementation here follows the NST approach in the paper "A Neural Algorithm of Artistic Style" by Gatys et al. (2015). The approach takes content and style representations from a convolutional neural network and optimizes a target image to be close to them.

Implementation Summary

- Model: VGG19 pretrained on ImageNet
- Framework: PyTorch
- Key Layers Used:
- Content layer: conv4_2
- Style layers: conv1_1, conv2_1, conv3_1, conv4_1, conv5_1

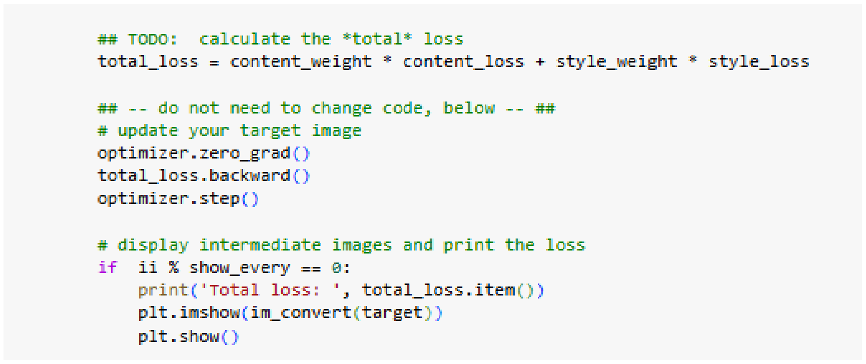


```
[49] def get_features(image, model, layers=None):
         """ Run an image forward through a model and get the features for
             a set of layers. Default layers are for VGGNet matching Gatys et al (2016)
         """

         ## TODO: Complete mapping layer names of PyTorch's VGGNet to names from the paper
         ## Need the layers for the content and style representations of an image
         if layers is None:
             layers = {'0': 'conv1_1',
                       '5': 'conv2_1',
                       '10': 'conv3_1',
                       '19': 'conv4_1',
                       '21': 'conv4_2',
                       '28': 'conv5_1'
                     }

         ## -- do not need to change the code below this line -- ##
         features = {}
         x = image
         # model._modules is a dictionary holding each module in the model
         for name, layer in model._modules.items():
             x = layer(x)
             if name in layers:
                 features[layers[name]] = x

         return features
```

Loss Functions Used:

- **Content Loss:** Mean squared error between conv4_2 features of content and target
- **Style Loss:** MSE between Gram matrices of style and target features for each style layer

- **Total Loss:**



```
    ## TODO:  calculate the *total* loss
    total_loss = content_weight * content_loss + style_weight * style_loss

    ## -- do not need to change code, below -- ##
    # update your target image
    optimizer.zero_grad()
    total_loss.backward()
    optimizer.step()

    # display intermediate images and print the loss
    if  ii % show_every == 0:
        print('Total loss: ', total_loss.item())
        plt.imshow(im_convert(target))
        plt.show()
```

Part A

Content Image: Picasso Painting

Style Image: Ballerina Photograph

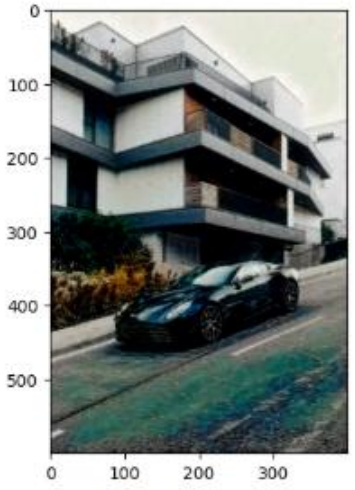
- Steps: 2000
- Style Weight: 1×10^6
- Observations: The image gradually transformed, blending strong abstract brush strokes from the style image. Loss decreased over time showing convergence.

Visual Progress:

```
Total loss:  12603714.0
```

```
Total loss:  6971653.5
```

```
Total loss:  4394505.5
```



```
Total loss:  2991445.5
```

```
Total loss:  2114846.0
```

Figure 3: Sample outputs after 500, 1000, 1500, and 2000 steps. Total loss consistently reduced as shown below:

- Step 500/2000: Loss ≈ 12,683,714.0
- Step 1000/2000: Loss ≈ 6,971,653.5
- Step 1500/2000: Loss ≈ 4,394,505.5
- Step 2000/2000: Loss ≈ 2,991,445.5

Final output showed significant stylization with evident texture from the cave overlaid onto the car/building image (Figure 4).

Part B

New Content: Car and Building Photo

New Style: Ice Cave Image

We ran 3 experiments:

Experiment	Style Weight	Steps	Final Loss	Comments
Exp 1	1×10^5 times	1000	~546,861	Lighter texture transfer
Exp 2	1×10^7 times	2000	~21,155,254	Heavy texture, distorted content
Exp 3	1×10^6 times	3000	~942,287	Balanced result, clear stylization

Findings

- Higher style weights increase artistic texture but can distort content.
- More steps improve convergence and detail.
- Visual outputs show the best results with a balanced style weight and enough steps (Exp 3).

Conclusion

For this task, I employed neural style transfer using a pre-trained VGG19 network to effectively combine the content of one image (a car in front of a building) with the style of another (an ice cave texture). This was achieved through content and style feature extraction from specific convolutional layers, Gram matrix calculation for style representation, and iterative optimization of a target image to minimize both content and style losses.

I experimented with different style weights and number of steps to observe their effects on the stylization results. Lower style weights resulted in more subtle textures with more content preserved, while higher style weights resulted in more extreme stylization at the cost of finer content details. The number of steps also played a significant role longer training resulted in smoother and more coherent results.

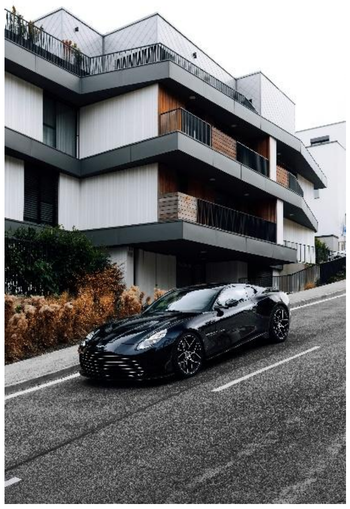
Among the experiments, Experiment 3 (style_weight = 1e6, steps = 3000) gave the best balance and resulted in a visually striking blend of artistic texture and discernible structure. The stylized output preserved the architectural elements of the car and building while taking on the characteristic texture and color palette of the ice cave.

This project helped me fully understand how style and content could be separated and recombined using convolutional neural networks. I also gained first-hand experience with feature extraction, optimization techniques, and the impact of hyperparameters on artistic output.

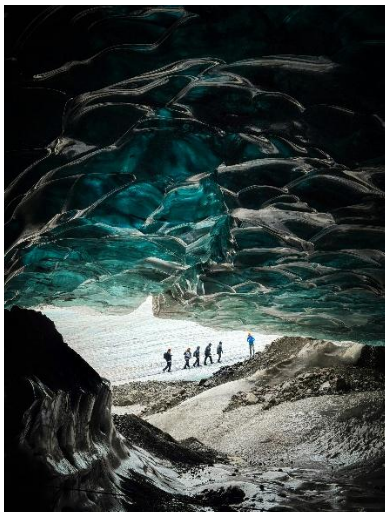
In summary, neural style transfer is such a fascinating instance of deep learning's creative use. The results of this project demonstrate not only the technical success of the implementation but also the flexibility and expressiveness that style transfer allows.

Results

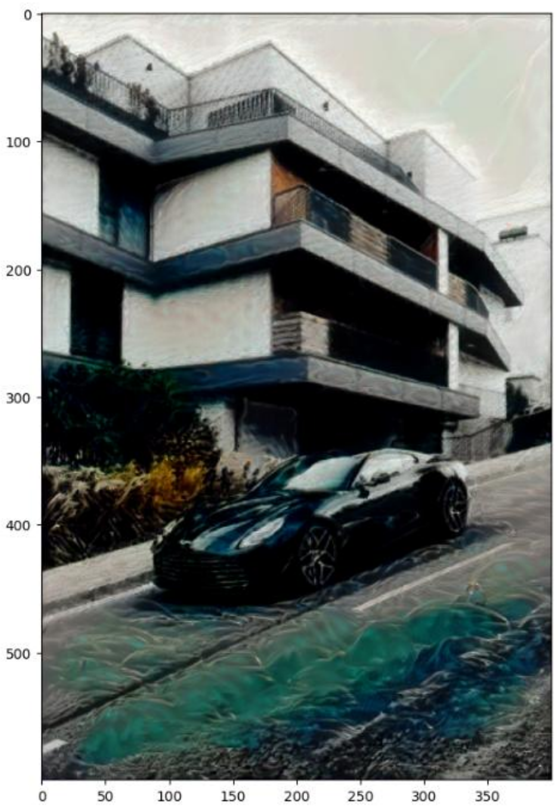
1. **Content Image**



2. Style Image



3. Final Output Image


0
100
200
300
400
500
0
50
100
150
200
250
300
350