Myelin Foundry Challenge

Dhruv Srikanth

Contents

Section 1

Language Used

Python 3.x

Platform Used

Google Colaboratory - https://colab.research.google.com/notebooks/intro.ipynb

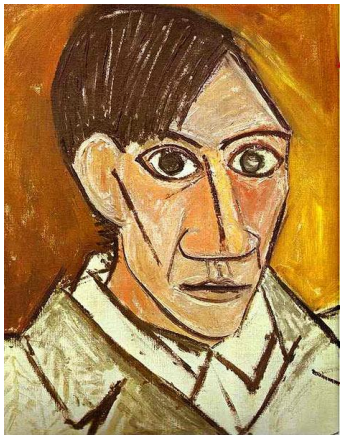
Non-Standard Python Modules Used –

- Tensorflow - https://www.tensorflow.org/
- Pillow - https://pillow.readthedocs.io/en/stable/
- Matplotlib - https://matplotlib.org/#

Inputs – (provided in Github repository)

Content Image

Style Image



Section 2

Output – (provided in Github repository)

Citations

Gatys, Leon & Ecker, Alexander & Bethge, Matthias. (2015). A Neural Algorithm of Artistic Style. arXiv. 10.1167/16.12.326.

Website Looked Through

https://www.datacamp.com/community/tutorials/implementing-neural-style-transfer-using-tensorflow

https://www.tensorflow.org/tutorials/generative/style_transfer

Process Summary

- Input images – content and style images – are loaded.
- Following this, a pretrained model – VGG19 (deep CNN network) – is loaded.
- This model is used to extracts the macro structure from the content image and texture details from the style image. This is done by selecting the output of block5_conv2 in the network for the content image and a convolutional layer in multiple blocks for the style image (mentioned below).
- After this, a custom model is defined that computes the gram matrix as well (required for the style image). After this, a style loss (defined as the MSE between generated image and target image), content loss (defined as the MSE between generated image and target image) and variation loss (used to reduce the amount of noise present in the output image).
- Define optimizer used during training – Adam.
- Train model optimizing after every iteration to produce desired stylized output image.

Learnings

Gram matrix

I was unaware of the gram matrix, its calculation, significance and implementation prior to this challenge. This challenge drove me to discover the gram matrix and learn of its use and underlying computations.

Section 3 - Model

Model Details

Pretrained model on ImageNet data, VGG19 is used. The fully connected layer is not included in model used. VGG19 model contains these blocks –

block1_conv1

block1_conv2

block1_pool

block2_conv1

block2_conv2

block2_pool

block3_conv1

block3_conv2

block3_conv3

block3_conv4

block3_pool

block4_conv1

block4_conv2

block4_conv3

block4_conv4

block4_pool

block5_conv1

block5_conv2

block5_conv3

block5_conv4

block5_pool

layers for extracting structure from content image are –

block5_conv2

layers for extracting textures from style image are –

block1_conv1

block2_conv1

block3_conv1

block4_conv1

block5_conv1

Model Metrics

- Mean Square Error (MSE) is used to compute content loss and style loss (with Gram matrix).
- Final loss computed (includes variation loss, content loss and style loss) = 5074096.5
- Loss weights –
    - Content loss weight=1e4
    - Style loss weight=1e-2
    - Variation loss weight=30

Training

- Learning rate = 0.02
- Beta = 0.99
- Epsilon = 1e-1
- No. of epochs = 10
- No. steps in each epoch = 100