LOVELY PROFESSIONAL UNIVERSITY

Report On

Painting style transfer to other pictures and selfies(Neural style transfer)

Subject: INT247 (Machine Learning Foundation)

Team:

Roll. No	Reg. No	Name of student
07	11705063	Uppalapati Naga Varma
08	11705095	Vunnam Lakshminarayana

Submitted to:

Dr. Aditya Khamparia (Head of the Dept. Machine Learning)

Introduction:

We are trying to transfer painting style from one picture to other normal picture in real time which is called artistic style transfer or neural style transfer. This is achieved by convolutional neural networks, we need not to bare the headache to train the network as we are going to take a pretrained model to achieve our goal.

So, finally what are we trying to achieve?

Here are some example pictures to have a rough idea.

Content Image

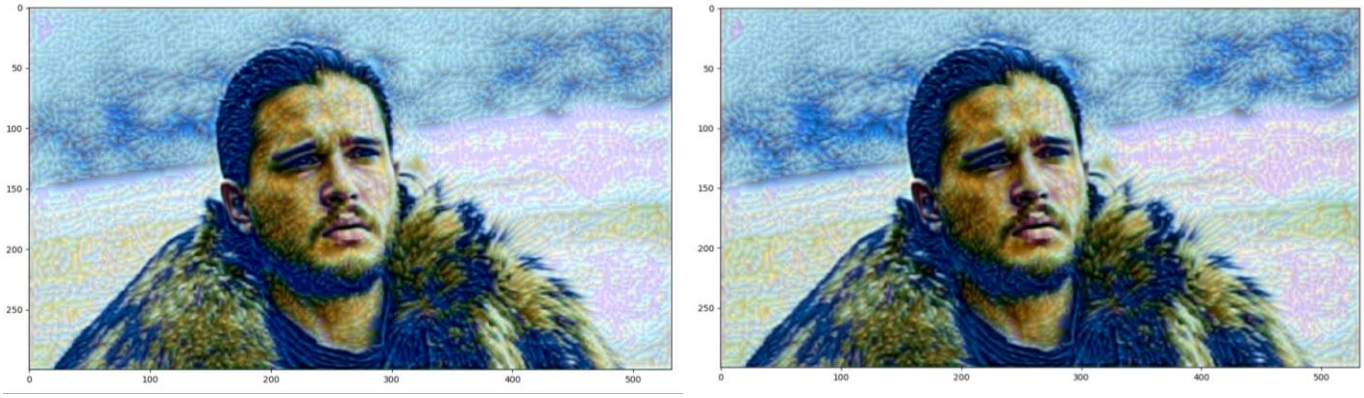
Style Image

Output:

Content image

Style image

output:



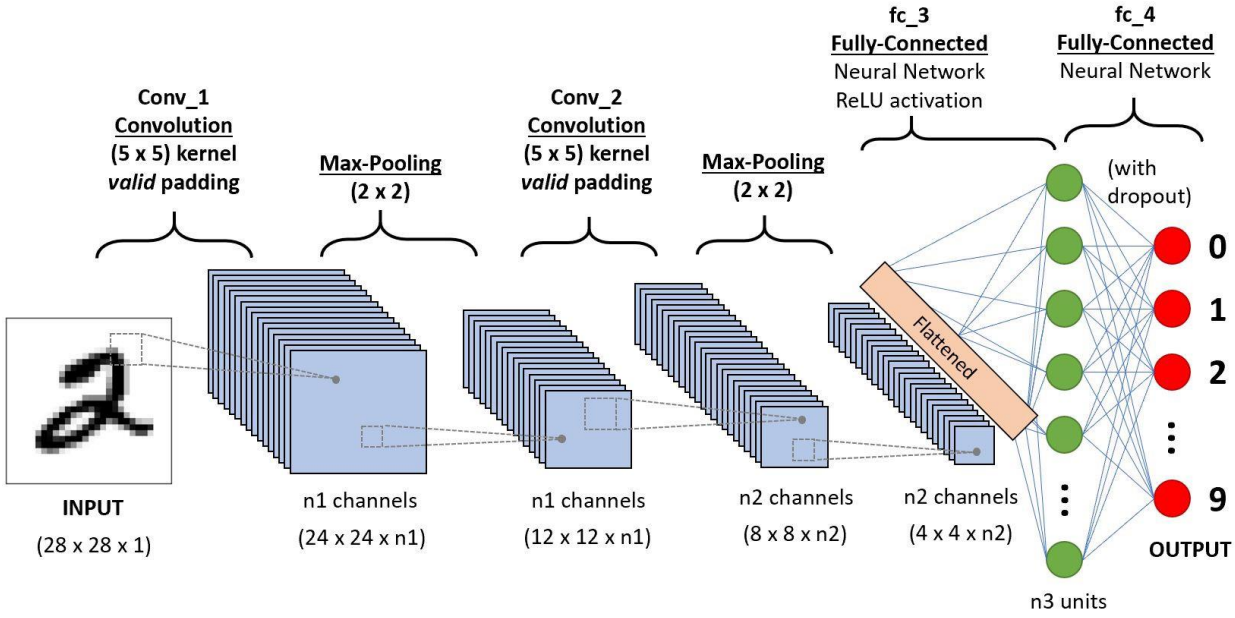
So, we got an idea that we wanted to merge or blend two images into one and to achieve this we use convolutional neural networks. But what is convolutional neural networks.

Convolutional neural networks (CNN)

As per the internet knowledge it goes as, A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a CNN is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, CNNs have the ability to learn these filters/characteristics.

A CNN is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters

involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.



Convolutional Neural networks consists of layers of small computational units that process visual information hierarchically in a feed forward manner. Each layer of units can be understood as a collection of images filters, each of which extracts a certain feature from the input image. Thus, the output of a given layer consists of so-called feature maps: differently filtered versions of the input image.

Extraction of Content from an Image:

Along the processing hierarchy of the network, the input image is transformed into representations that increasingly care about the actual content of the image compared to its detailed pixel values. We therefore refer to the feature responses in higher layers of the network as the content representation.

The second part of the fourth convolutional layer (4_2) of pretrained VGG19 network is used as content extractor

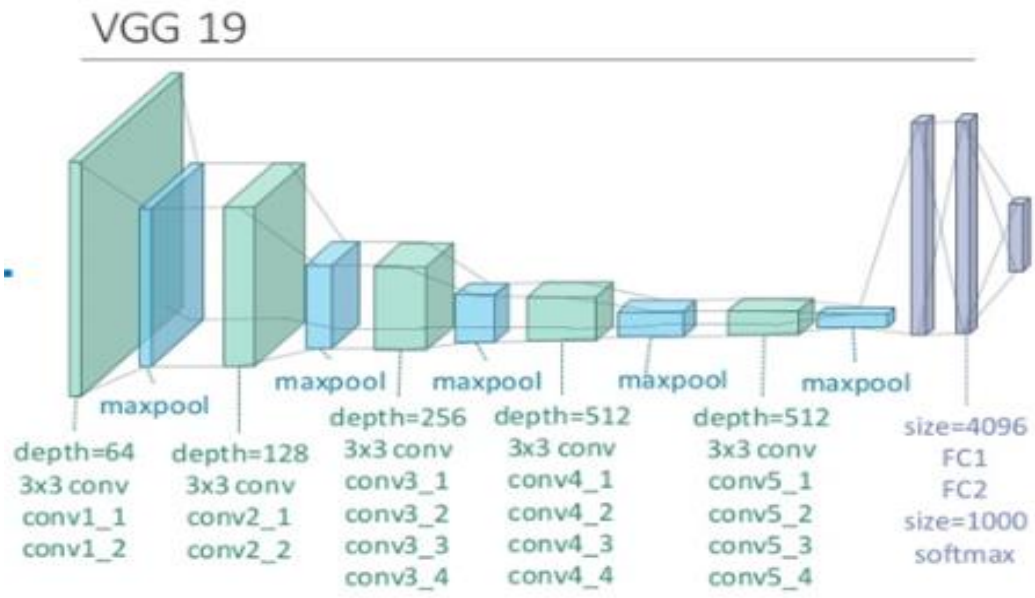
Extraction of Style from an Image:

We just do the opposite to above. To obtain a representation of the style of an input image, we use correlations between the different filter responses over the spatial

extent of the feature maps. We obtain a stationary, multi-scale representation of the input image, which captures its texture information but not the global arrangement.

These correlations between feature maps is known as gram matrix. The layers used for calculation of gram matrix are 1_1, 2_1, 3_1, 4_1, 5_1 with varied style weight constant for each layer. This constant be a hyperparameter used for changing style levels.

Model Overview



Content Loss and Style Loss

The content cost function is making sure that the content present in the content image is captured in the generated image. It has been found that CNNs capture information about content in the higher levels, where the lower levels are more focused on individual pixel values. Therefore, we use the top-most CNN layer to define the content loss function.

$$L_{content} = \frac{1}{2} \sum_{i,j} (A_{ij}^{l}(g) - A_{ij}^{l}(c))^2$$

The content loss

Defining the style loss function requires more work. To extract the style information from the VGG network, we use all the layers of the CNN. Furthermore, style information is measured as the amount of correlation present between features maps in each layer. Next, a loss is defined as the difference of correlation present between the feature maps computed by the generated image and the style image. Mathematically, the style loss is defined as,

$$L_{style} = \sum_l w^l L^l_{style} \text{ where,}$$

$$L^l_{style} = \frac{1}{M^l} \sum_{ij} (G^l_{ij}(s) - G^l_{ij} g)^2 \text{ where,}$$

$$G^l_{ij}(I) = \sum_k A^l_{ik}(I) A^l_{jk}(I).$$

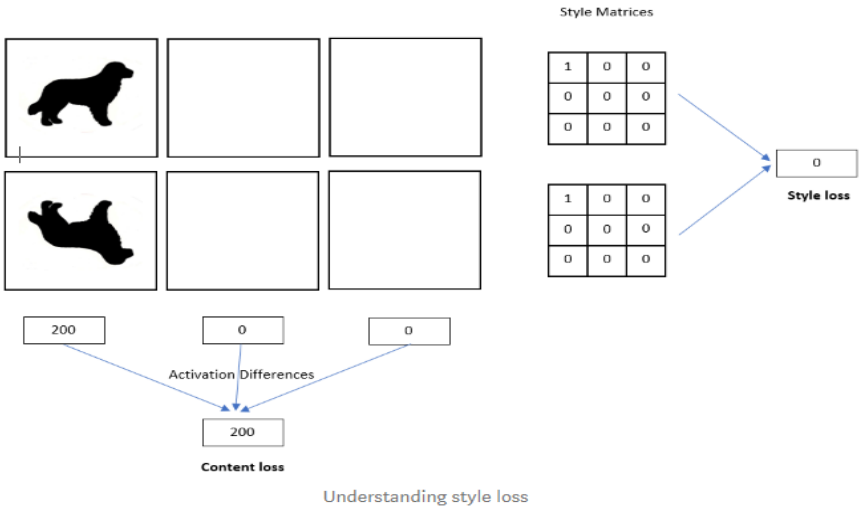
Final Loss:

$$L = \alpha L_{content} + \beta L_{style},$$

where α and β are user-defined hyperparameters. Here β has absorbed the M^l normalization factor defined earlier. By controlling α and β you can control the amount of content and style injected to the generated image.

Why is it that style is captured in the Gram matrix?

It's great that we know how to compute the style loss. But we still haven't been shown "why the style loss is computed using the Gram matrix". The Gram matrix essentially captures the "distribution of features" of a set of feature maps in each layer. By trying to minimize the style loss between two images, you are essentially matching the distribution of features between the two images



Understanding style loss

Finally run the code and save the output as discussed in code in GitHub here: https://github.com/Lovely-Professional-University-CSE/int247-machine-learning-project-2020-kem031-rollno_7_8/blob/master/StyleTransfer.ipynb

And for detailed documentation visit this link: https://sites.google.com/view/lpu2020-int247-78/documentation?authuser=0

References:

[1]. From TensorFlow tutorials https://www.tensorflow.org/tutorials/generative/style_transfer

[2]. From the pytorch documentation https://pytorch.org/tutorials/advanced/neural_style_tutorial.html#

[3]. Notes and explanation of CNN https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

[4]. Guide to neural style transfer https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-neural-style-transfer-ef88e46697ee