


Content (C) Style (S)

Generated image (G)

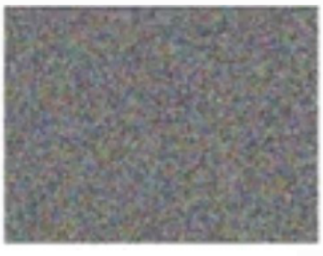
Cost function:

measures similarity

$$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$$

1. Initiate G randomly

G: 100 × 100 × 3 (RGB)



2. Use gradient descent to minimize $J(G)$

$$G := G - \frac{d}{dG} J(G)$$

C

S

=

G

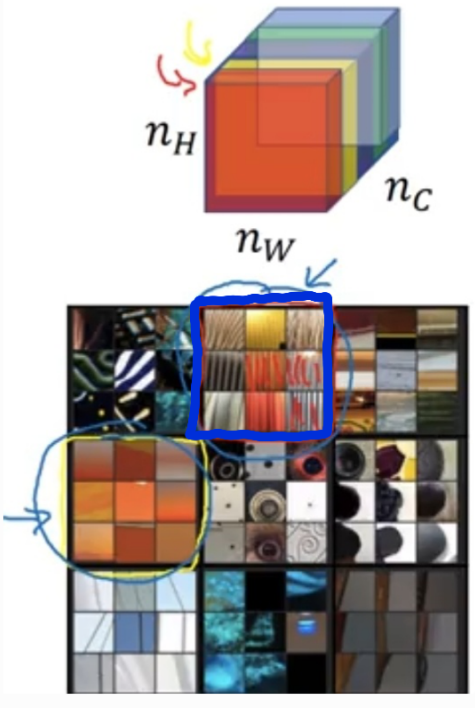
- Say you use hidden layer l to compute content cost.
- Use pre-trained ConvNet. (E.g., VGG network)
- Let $a^{[l](C)}$ and $a^{[l](G)}$ be the activation of layer l on the images
- If $a^{[l](C)}$ and $a^{[l](G)}$ are similar, both images have similar content

$$J_{content}(C,G) = \frac{1}{2}\|a^{[l](C)} - a^{[l](G)}\|^2$$

↳ Element-wise sum of squares differences

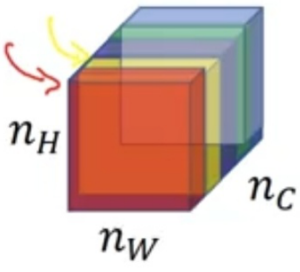
But, what do we really mean by style? ⇒ CORRELATION

Style image



Correlated?
Uncorrelated

Generated Image



A style is a pattern present in the whole picture. So, if the style is that of a vertical lines-like painting, all layers output will be correlated with the top-center channel output (squared in blue)

That is formalised as the style matrix.

Style matrix

H W C

Let $a^{[l]}_{i,j,k}$ = activation at (i,j,k). $G^{[l]}$ is $n_c^{[l]} \times n_c^{[l]}$;

$$G^{[l](G)}_{kk'} = \sum_{i=1}^{n_H}\sum_{j=1}^{n_W} a^{[l](G)}_{i,j,k}\, a^{[l](G)}_{i,j,k'} \quad ;$$

$$J^{[l]}_{style}(S,G) = \frac{1}{(2n_H^{[l]}n_W^{[l]}n_C^{[l]})^2}\sum_k\sum_{k'}\left(G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'}\right)^2$$

In the deeper layers of a ConvNet, each channel corresponds to a different feature detector. The style matrix $G^{[l]}$ measures the degree to which the activations of different feature detectors in layer l vary (or correlate) together with each other.

What you should remember:

- The style of an image can be represented using the Gram matrix of a hidden layer's activations.
- You get even better results by combining this representation from multiple different layers.
- This is in contrast to the content representation, where usually using just a single hidden layer is sufficient.
- Minimizing the style cost will cause the image G to follow the style of the image S.

What you should remember:

- The total cost is a linear combination of the content cost $J_{content}(C, G)$ and the style cost $J_{style}(S, G)$.
- α and β are hyperparameters that control the relative weighting between content and style.

4 - Neural Style Transfer (NST)

Next, you will be building the Neural Style Transfer (NST) algorithm in three steps:

- First, you will build the content cost function $J_{content}(C, G)$
- Second, you will build the style cost function $J_{style}(S, G)$
- Finally, you'll put it all together to get $J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$. Exciting!

4.1 - Computing the Content Cost

4.1.1 - Make Generated Image G Match the Content of Image C

One goal you should aim for when performing NST is for the content in generated image G to match the content of image C. To do so, you'll need an understanding of **shallow versus deep layers** :

- The shallower layers of a ConvNet tend to detect lower-level features such as *edges and simple textures.*
- The deeper layers tend to detect higher-level features such as more *complex textures and object classes.*

To choose a "middle" activation layer $a^{[l]}$:

You need the "generated" image G to have similar content as the input image C. Suppose you have chosen some layer's activations to represent the content of an image.

- In practice, you'll get the most visually pleasing results if you choose a layer in the **middle** of the network--neither too shallow nor too deep. This ensures that the network detects both higher-level and lower-level features.
- After you have finished this exercise, feel free to come back and experiment with using different layers to see how the results vary!

To forward propagate image "C:"

- Set the image C as the input to the pretrained VGG network, and run forward propagation.
- Let $a^{(C)}$ be the hidden layer activations in the layer you had chosen. (In lecture, this was written as $a^{[l](C)}$, but here the superscript $[l]$ is dropped to simplify the notation.) This will be an $n_H \times n_W \times n_C$ tensor.

To forward propagate image "G":

- Repeat this process with the image G: Set G as the input, and run forward progation.
- Let $a^{(G)}$ be the corresponding hidden layer activation.

4.2.1 - Style Matrix

Gram matrix

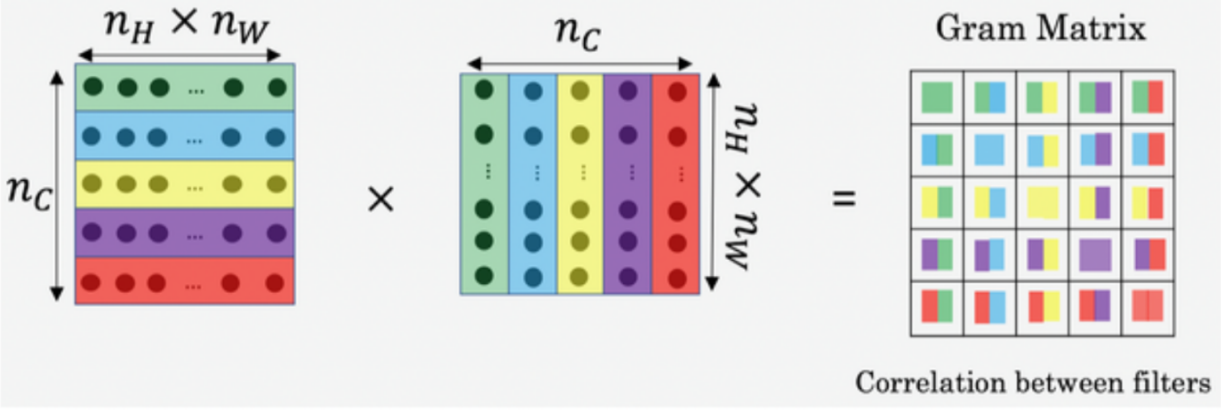
- The style matrix is also called a "Gram matrix."
- In linear algebra, the Gram matrix G of a set of vectors $(v_1, \dots, v_n)$ is the matrix of dot products, whose entries are $G_{ij} = v_i^T v_j = np.dot(v_i, v_j)$.
- In other words, G_{ij} compares how similar v_i is to v_j: If they are highly similar, you would expect them to have a large dot product, and thus for G_{ij} to be large.

Two meanings of the variable G

- Note that there is an unfortunate collision in the variable names used here. Following the common terminology used in the literature:
  - G is used to denote the Style matrix (or Gram matrix)
  - G also denotes the generated image.
- For the sake of clarity, in this assignment G_{gram} will be used to refer to the Gram matrix, and G to denote the generated image.

Compute Gram matrix G_{gram}

You will compute the Style matrix by multiplying the "unrolled" filter matrix with its transpose:



$$\mathbf{G}_{gram} = \mathbf{A}_{unrolled}\mathbf{A}_{unrolled}^T$$

$G_{(gram)ij}$: correlation

The result is a matrix of dimension (n_C, n_C) where n_C is the number of filters (channels). The value $G_{(gram)i,j}$ measures how similar the activations of filter i are to the activations of filter j.

$G_{(gram)ii}$: prevalence of patterns or textures

- The diagonal elements $G_{(gram)ii}$ measure how "active" a filter i is.
- For example, suppose filter i is detecting vertical textures in the image. Then $G_{(gram)ii}$ measures how common vertical textures are in the image as a whole.
- If $G_{(gram)ii}$ is large, this means that the image has a lot of vertical texture.

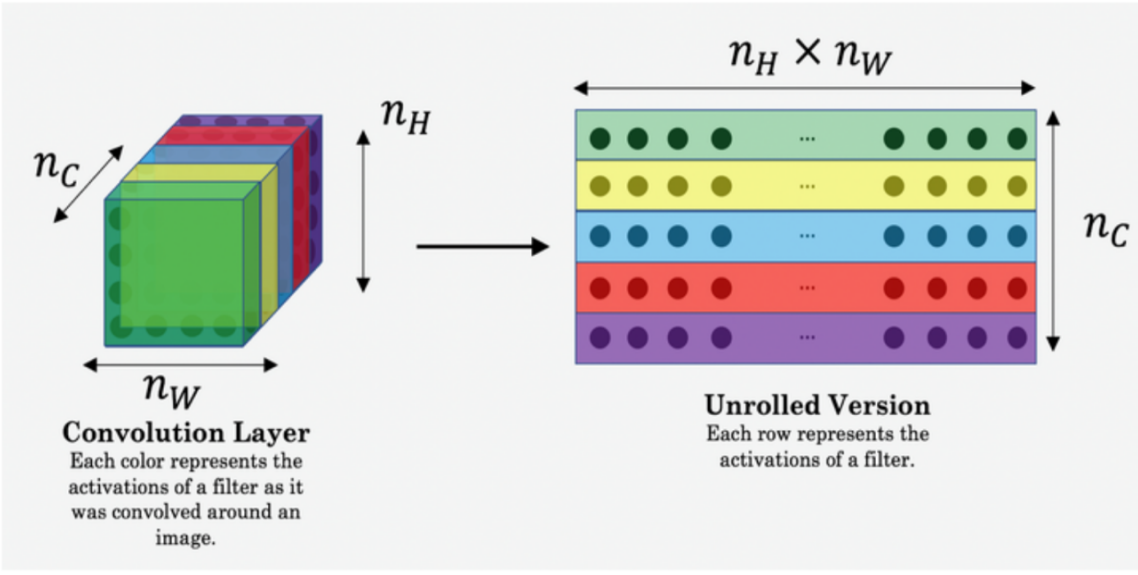
By capturing the prevalence of different types of features ($G_{(gram)ii}$), as well as how much different features occur together ($G_{(gram)ij}$), the Style matrix G_{gram} measures the style of an image.

4.1.2 - Content Cost Function $J_{content}(C, G)$

One goal you should aim for when performing NST is for the content in generated image G to match the content of image C. A method to achieve this is to calculate the content cost function, which will be defined as:

$$J_{content}(C, G) = \frac{1}{4 \times n_H \times n_W \times n_C} \sum_{\text{all entries}} (a^{(C)} - a^{(G)})^2 \tag{1}$$

- Here, n_H, n_W and n_C are the height, width and number of channels of the hidden layer you have chosen, and appear in a normalization term in the cost.
- For clarity, note that $a^{(C)}$ and $a^{(G)}$ are the 3D volumes corresponding to a hidden layer's activations.
- In order to compute the cost $J_{content}(C, G)$, it might also be convenient to unroll these 3D volumes into a 2D matrix, as shown below.
- Technically this unrolling step isn't needed to compute $J_{content}$, but it will be good practice for when you do need to carry out a similar operation later for computing the style cost J_{style}.



4.3 - Defining the Total Cost to Optimize

Finally, you will create a cost function that minimizes both the style and the content cost. The formula is:

$$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$$

4.2.3 Style Weights

- So far you have captured the style from only one layer.
- You'll get better results if you "merge" style costs from several different layers.
- Each layer will be given weights ($\lambda^{[l]}$) that reflect how much each layer will contribute to the style.
- After completing this exercise, feel free to come back and experiment with different weights to see how it changes the generated image G.
- By default, give each layer equal weight, and the weights add up to 1. ($\sum_l^L \lambda^{[l]} = 1$)

Start by listing the layer names:

4.2.2 - Style Cost

You now know how to calculate the Gram matrix. Congrats! Your next goal will be to minimize the distance between the Gram matrix of the "style" image S and the Gram matrix of the "generated" image G.

- For now, you will use only a single hidden layer $a^{[l]}$.
- The corresponding style cost for this layer is defined as:

$$J_{style}^{[l]}(S, G) = \frac{1}{4 \times {n_C}^2 \times (n_H \times n_W)^2} \sum_{i=1}^{n_C} \sum_{j=1}^{n_C} (G_{(gram)i,j}^{(S)} - G_{(gram)i,j}^{(G)})^2 \tag{2}$$

- $G_{gram}^{(S)}$ Gram matrix of the "style" image.
- $G_{gram}^{(G)}$ Gram matrix of the "generated" image.
- Make sure you remember that this cost is computed using the hidden layer activations for a particular hidden layer in the network $a^{[l]}$

4.2.2 - Style Cost

You now know how to calculate the Gram matrix. Congrats! Your next goal will be to minimize the distance between the Gram matrix of the "style" image S and the Gram matrix of the "generated" image G.

- For now, you will use only a single hidden layer $a^{[l]}$.
- The corresponding style cost for this layer is defined as:

$$J_{style}^{[l]}(S,G) = \frac{1}{4 \times {n_C}^2 \times (n_H \times n_W)^2} \sum_{i=1}^{n_C} \sum_{j=1}^{n_C} (G_{(gram)i,j}^{(S)} - G_{(gram)i,j}^{(G)})^2 \tag{2}$$

- $G_{gram}^{(S)}$ Gram matrix of the "style" image.
- $G_{gram}^{(G)}$ Gram matrix of the "generated" image.
- Make sure you remember that this cost is computed using the hidden layer activations for a particular hidden layer in the network $a^{[l]}$