A Neural Algorithm of Artistic Style

1 Citation

Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." arXiv preprint arXiv:1508.06576 (2015).

`https://arxiv.org/pdf/1508.06576.pdf`

2 Abstract

We present a deep learning system that can take the content of one image and render it in the artistic style of another image.

3 Introduction

We can reconstruct the content of image from feature maps of the CNN. The lower-level (i.e. closer to input) feature maps reconstruct input exactly, while higher-level feature maps are more abstract.

To reconstruct style, we need to analyze correlation of filter responses. If we compute these correlations in multiple layers, we get a good representation of image texture (i.e. style). For the style representation, higher-level layers of the network do better for reconstruction. Also, the notion of using correlations between filter responses has an analog in biological systems.

With this approach, we can take some white noise and push it to match the content of one image and style of another. This means you can take a photograph and render it in the style of Van Gogh's "Starry Night", for example.

4 Methods

We use the VGG-19 neural network, pretrained on ImageNet. Specifically, we use the 19 conv and 5 pooling layers. We replace max-pooling with average pooling because that allows better gradient flow.

Now, let us call our original image $\vec{p}$ and our reconstruction $\vec{x}$. Suppose layer l has N_l feature maps each with size (i.e. width $\times$ height) M_l. We store the filter responses in a matrix $F^l \in \mathbb{R}^{N_l \times M_l}$, where F^l_{ij} is the activation of the i^{th} filter in position j in layer l. For content reconstruction, we define this loss function that measures how different the reconstructed activations are from the original:

$$Ł_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} (F^l_{ij} - P^l_{ij})^2$$

The derivative is:

$$\frac{\partial L_{content}}{\partial F_{ij}^l} = \mathbb{I}[F_{ij}^l > 0](F^l - P^l)_{ij}$$

We can backpropagate this to make the reconstruction match the content of the original.

To represent style, we the Gram matrix $G^l \in \mathbb{R}^{N_l \times N_l}$, where

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l$$

Now, suppose that our original image is $\vec{a}$ (with style G^l) and the reconstruction is $\vec{x}$ (with style A^l). Then we define the loss as

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

However, we do style reconstruction using multiple layers, rather than just one as in content reconstruction (this gives better results), so we sum the above term over all the layers that we consider.

$$L_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

for weights w_l. Our derivative is then:

$$\frac{\partial E_l}{\partial F_{ij}^l} = \mathbb{I}[F_{ij}^l > 0] \frac{1}{N_l^2 M_l^2} ((F^l)^T (G^l - A^l))_{ji}$$

To combine the two loss functions, we do:

$$L_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha L_{content}(\vec{p}, \vec{x}) + \beta L_{style}(\vec{a}, \vec{x})$$

For content representation, we use layer conv4_2. For style representation, we use layers conv1_1, conv2_1, conv3_1, conv4_1, and conv5_1 (with $w_l = \frac{1}{5}$ for all of them). We set $\frac{\alpha}{\beta} = 10^{-3}$ or 10^{-4}.