Image Style Transfer Using Convolutional Neural Networks

Leon A. Gatys
Centre for Integrative Neuroscience, University of Tubingen, Germany
Bernstein Center for Computational Neuroscience, Tubingen, Germany
Graduate School of Neural Information Processing, University of Tubingen, Germany
leon.gatys@bethgelab.org

Alexander S. Ecker
Centre for Integrative Neuroscience, University of Tubingen, Germany
Bernstein Center for Computational Neuroscience, Tubingen, Germany
Max Planck Institute for Biological Cybernetics, Tubingen, Germany
Baylor College of Medicine, Houston, TX, USA

Matthias Bethge
Centre for Integrative Neuroscience, University of Tubingen, Germany
Bernstein Center for Computational Neuroscience, Tubingen, Germany
Max Planck Institute for Biological Cybernetics, Tubingen, Germany

Summarized by Kshitij Gupta

Abstract

The paper aims to find an image representation such we can separate image content from style. It introduces *A Neural Algorithm of Artisitic Style* that can separate and recombine the image content and style of natural images. It uses pre-trained convolutional neural network built for object detection to capture the style and content embeddings of the given image. Then it uses the style embeddings and the content embeddings of the source and target image respectively to perform style transfer.

Method

It uses a VGG19 convolutional network to extract features of the given image. It is able to capture low-level style/texture information of the given image in the shallow layers and high-level content/semantic information in the deeper layers.

It defines a content loss ($L_{content}$) of the given image and a style loss (L_{style}) and uses a wigthed sum of both the losses to train the model.

It uses the following loss functions to train the model:

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^{l} - P_{ij}^{l}\right)^2$$

where p and x are the source and the generated image respectively, and F and P are their corresponding feature maps obtained from the CNN at layer l.

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

where G is the set of gram matrices in which G^l is the gram matrix calculated using the feature maps of the l^{th} layer. If the l^{th} layer has *C* channels, G^l is a C*C matrix in which G^l_{ij} is the summation of the element-wise product of i^{th} and j^{th} feature maps at layer *l*.

Since we propose that the CNN is able to capture low-level style in the shallower layers of the network, We define a style loss function using those shallower layers of the network as follows:

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

where G^l and A^l the gram matrices computed at the l^{th} layer of the original and the generated image respectively.

We use weighted sum of the particular shallow layers to compute the final style loss as follows:

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

The final loss used to train the model is:

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$$

Now that we have our final loss function, We can compute the style loss by comparing the generated image with the source image and the content loss by comparing the target image with the generated image and carefully set alpha and beta to set the content-style tradeoff and train our model accordingly.

Scope of Improvement

Although this approach gives decent results, we are having to go through several iterations through the model to generate the final image and hence we cannot run this on real-time videos or computationaly weaker devices like mobile phones.