STYLE TRANSFER

Guo Jiaming
Harvard
15000@pku.edu.cn

Wang Jianqiao
Yuanpei College
1500017822@pku.edu.cn

ABSTRACT

Neural Style Transfer is a process of using *Convolutional Neural Networks (CNN)* to render a content image in different styles. Our contribution includes designing and training the framework of style transfer (Gatys et al. (2015)) and fast style transfer proposed by Johnson et al. (2016) with pre-trained VGG Networks.

1 INTRODUCTION

1.1 VERY DEEP CONVOLUTIONAL NETWORKS (VGG)

VGG, proposed by Simonyan & Zisserman (2014), is one of the most popular deep neural network classifiers. This network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes are then followed by a softmax classifier. Particularly, the network architecture of VGG16 and VGG19 are presented in Figure 3 in the Appendix. Moreover, pre-trained VGG16 and VGG19 networks [1] we used in this implement of style transfer were successfully trained on ImageNet.

1.2 STYLE TRANSFER

Transferring the style from one image onto another can be considered a problem of texture transfer, whose goal is to synthesize a texture from a source image while constraining the texture synthesis in order to preserve the semantic content of a target image. Texture transfer algorithms use only low-level image features of the target image to inform the texture transfer. Ideally, however, a style transfer algorithm should be able to extract the semantic image content from the target image and then inform a texture transfer procedure to render the semantic content of the target image in the style of the source image. Inspired by the power of *Convolutional Neural Networks (CNN)*, Gatys et al. (2015) proposed a framework to model the content or style of an image through a pre-trained CNN. Moreover, *A Neural Algorithm of Artistic Style* was proposed to separate and recombine the image content and style of natural images, which opened up a new field–Neural Style Transfer. However, the biggest drawback of this algorithm is that it runs too slow. To tackle this, Johnson et al. (2016) proposed a *Fast Style Transfer* algorithm. It introduced an *Image Transform Net*, a feed-forward network that is trained to solve the optimization problem proposed by Gatys et al. (2015) in real-time.

2 MODEL

2.1 A NEURAL ALGORITHM OF ARTISTIC STYLE

A Neural Algorithm of Artistic Style, proposed by Gatys et al. (2015), is generated on the basis of pre-trained VGG19 network, which contains 16 convolutional and 5 pooling layers. Before we introduce the algorithm (shown in Figure 2.1), it is of great importance to notice that what differs this style transfer framework from other deep learning networks is that the different objects they does gradient descent on. In conventional deep learning networks, the network fixes the input and trains the weights by optimizing loss function. However, in this style transfer framework, a pre-trained network is used. In other words, the network trains its input with fixed weight by optimizing its loss

[1] Pre-trained VGG weights:`http://www.vlfeat.org/matconvnet/models/?C=D;O=A`

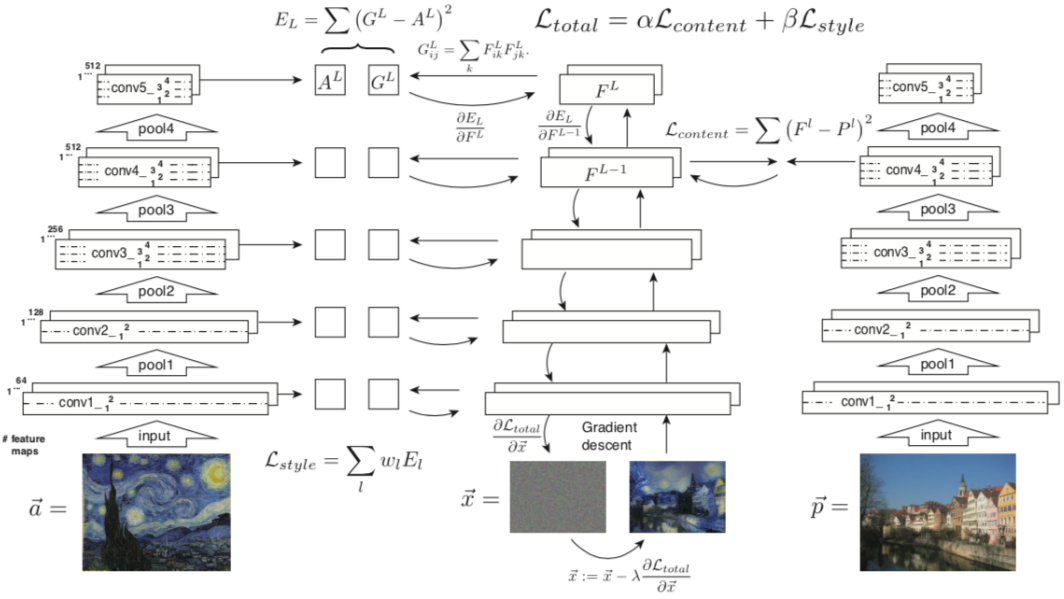

Figure 1: A Neural Algorithm of Artistic Style

function.

The process of the algorithm is as follows: (1) let three images pass forward through pre-trained VGG19, including content image, style image, and the the initialized 'image' we want to generate. (2) optimize the loss function and revise the 'image' using a stochastic gradient descent algorithm. In the above process, the content and style images are fixed and the initialized 'image' could be a white noise or the original content (style) image. The loss function contains two parts: the content loss and the style loss.

- **Content Loss**: Let $\vec{p}$ and $\vec{x}$ be the content image and the image that is generated, and P^l and $F^l \in \mathcal{R}^{N_l \times M_l}$ their respective feature representation in layer l, where N_l and M_l denotes the number of filters and the size of each feature map respectively. We then define the squared-error loss between the two feature representations as the content loss on layer l:

  $$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} ||F^l - P^l||_F^2$$

  The total content loss is

  $$\mathcal{L}_{content}(\vec{p}, \vec{x}) = \sum_{l \text{ is a content layer}} \omega_l \mathcal{L}_{content}(\vec{p}, \vec{x}, l)$$

  where ω_l is the weight of layer l.

- **Style Loss**: Let $\vec{a}$ and $\vec{x}$ be the style image and the image that is generated, and A^l and G^l their respective style representation in layer l, which are actually the *Gram matrix* of their feature representation. The contribution of layer l to the total style loss is:

  $$\mathcal{L}_{style}(\vec{a}, \vec{x}, l) = \frac{1}{4N_l^2 M_l^2} ||G^l - A^l||_F^2$$

  and the total style loss is:

  $$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l \text{ is a style layer}} \omega_l \mathcal{L}_{style}(\vec{a}, \vec{x}, l)$$

  where ω_l is the weight of layer l.

- **Total Loss**: The total loss is a combination of content loss and style loss, which is given by

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

where α and β are the weighting factors for content and style reconstruction, respectively. Moreover, the loss ration is given by α/β.

2.2 Fast Style Transfer

Goodfellow et al. (2014) Johnson et al. (2016)

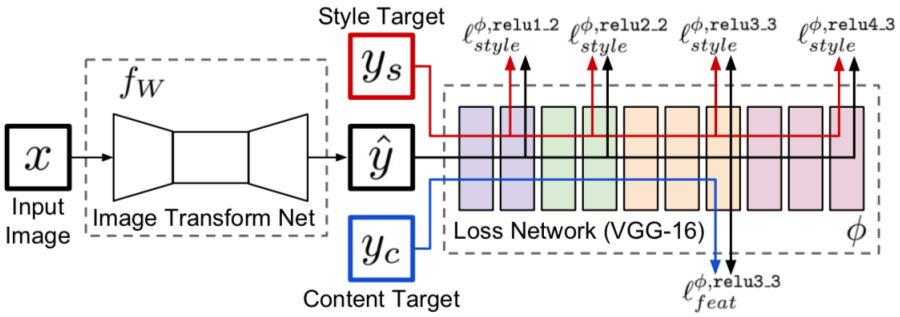

Figure 2: Fast Style Transfer: Image Transformation Network (left) and Loss Network (right).

Table 1: Network architecture used for style transfer networks.

Layer	Activation Size
Input	$3 \times 256 \times 256$
$32 \times 9 \times 9$ conv, stride 1	$32 \times 256 \times 256$
$64 \times 3 \times 3$ conv, stride 2	$64 \times 128 \times 128$
$128 \times 3 \times 3$ conv, stride 2	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
Residual block, 128 filters	$128 \times 64 \times 64$
$64 \times 3 \times 3$ conv, stride 1/2	$64 \times 128 \times 128$
$32 \times 3 \times 3$ conv, stride 1/2	$32 \times 256 \times 256$
$3 \times 9 \times 9$ conv, stride 1	$3 \times 256 \times 256$

3 Experiments

4 Conclusion and Discussion

References

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style, 2015.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

A VGG NETWORKS ARCHITECTURE

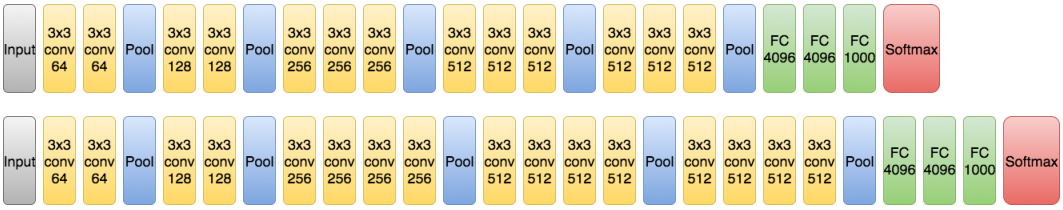

Figure 3: VGG Networks Architecture: VGG16 (above) and VGG19 (bottom).