Title: Neural style transfer

URL: https://en.wikipedia.org/wiki/Neural_style_transfer

PageID: 59892172

Categories: Category:Algorithms, Category:Deep learning



---

Neural style transfer ( NST ) refers to a class of software algorithms that manipulate digital images, or videos, in order to adopt the appearance or visual style of another image. NST algorithms are characterized by their use of deep neural networks for the sake of image transformation. Common uses for NST are the creation of artificial artwork from photographs, for example by transferring the appearance of famous paintings to user-supplied photographs. Several notable mobile apps use NST techniques for this purpose, including DeepArt and Prisma . This method has been used by artists and designers around the globe to develop new artwork based on existent style(s).

## History

NST is an example of image stylization , a problem studied for over two decades within the field of non-photorealistic rendering . The first two example-based style transfer algorithms were image analogies [ 1 ] and image quilting. [ 2 ] Both of these methods were based on patch-based texture synthesis algorithms.

Given a training pair of images–a photo and an artwork depicting that photo–a transformation could be learned and then applied to create new artwork from a new photo, by analogy. If no training photo was available, it would need to be produced by processing the input artwork; image quilting did not require this processing step, though it was demonstrated on only one style.

NST was first published in the paper "A Neural Algorithm of Artistic Style" by Leon Gatys et al., originally released to ArXiv 2015, [ 3 ] and subsequently accepted by the peer-reviewed CVPR conference in 2016. [ 4 ] The original paper used a VGG-19 architecture [ 5 ] that has been pre-trained to perform object recognition using the ImageNet dataset.

In 2017, Google AI introduced a method [ 6 ] that allows a single deep convolutional style transfer network to learn multiple styles at the same time. This algorithm permits style interpolation in real-time, even when done on video media.

## Mathematics

This section closely follows the original paper. [ 4 ]

### Overview

The idea of Neural Style Transfer (NST) is to take two images—a content image $p \rightarrow$ {\displaystyle {\vec {p}}} and a style image $a \rightarrow$ {\displaystyle {\vec {a}}} —and generate a third image $x \rightarrow$ {\displaystyle {\vec {x}}} that minimizes a weighted combination of two loss functions: a content loss $L_{\text{content}}(p \rightarrow, x \rightarrow)$ {\displaystyle {\mathcal {L}}_{\text{content }}({\vec {p}},{\vec {x}})} and a style loss $L_{\text{style}}(a \rightarrow, x \rightarrow)$ {\displaystyle {\mathcal {L}}_{\text{style }}({\vec {a}},{\vec {x}})} .

The total loss is a linear sum of the two: $L_{\text{NST}}(p \rightarrow, a \rightarrow, x \rightarrow) = \alpha L_{\text{content}}(p \rightarrow, x \rightarrow) + \beta L_{\text{style}}(a \rightarrow, x \rightarrow)$ {\displaystyle {\mathcal {L}}_{\text{NST}}({\vec {p}},{\vec {a}},{\vec {x}})=\alpha {\mathcal {L}}_{\text{content}}({\vec {p}},{\vec {x}})+\beta {\mathcal {L}}_{\text{style}}({\vec {a}},{\vec {x}})} By jointly minimizing the content and style losses, NST generates an image that blends the content of the content image with the style of the style image.

Both the content loss and the style loss measures the similarity of two images. The content similarity is the weighted sum of squared-differences between the neural activations of a single convolutional neural network (CNN) on two images. The style similarity is the weighted sum of Gram matrices within each layer (see below for details).

The original paper used a VGG-19 CNN, but the method works for any CNN.

Symbols

Let x → {\textstyle {\vec {x}}} be an image input to a CNN.

Let F l ∈ R N l × M l {\textstyle F^{l}\in \mathbb {R} ^{N_{l}\times M_{l}}} be the matrix of filter responses in layer l {\textstyle l} to the image x → {\textstyle {\vec {x}}} , where:

N l {\textstyle N_{l}} is the number of filters in layer l {\textstyle l} ;

M l {\textstyle M_{l}} is the height times the width (i.e. number of pixels) of each filter in layer l {\textstyle l} ;

F i j l ( x → ) {\textstyle F_{ij}^{l}({\vec {x}})} is the activation of the i th {\textstyle i^{\text{th}}} filter at position j {\textstyle j} in layer l {\textstyle l} .

A given input image x → {\textstyle {\vec {x}}} is encoded in each layer of the CNN by the filter responses to that image, with higher layers encoding more global features, but losing details on local features.

Content loss

Let p → {\textstyle {\vec {p}}} be an original image. Let x → {\textstyle {\vec {x}}} be an image that is generated to match the content of p → {\textstyle {\vec {p}}} . Let P l {\textstyle P^{l}} be the matrix of filter responses in layer l {\textstyle l} to the image p → {\textstyle {\vec {p}}} .

The content loss is defined as the squared-error loss between the feature representations of the generated image and the content image at a chosen layer l {\displaystyle l} of a CNN: L content ( p → , x → , l ) = 1 2 ∑ i , j ( A i j l ( x → ) − A i j l ( p → ) ) 2 {\displaystyle {\mathcal {L}}_{\text{content }}({\vec {p}},{\vec {x}},l)={\frac {1}{2}}\sum _{i,j}\left(A_{ij}^{l}({\vec {x}})-A_{ij}^{l}({\vec {p}})\right)^{2}} where A i j l ( x → ) {\displaystyle A_{ij}^{l}({\vec {x}})} and A i j l ( p → ) {\displaystyle A_{ij}^{l}({\vec {p}})} are the activations of the i th {\displaystyle i^{\text{th}}} filter at position j {\displaystyle j} in layer l {\displaystyle l} for the generated and content images, respectively. Minimizing this loss encourages the generated image to have similar content to the content image, as captured by the feature activations in the chosen layer.

The total content loss is a linear sum of the content losses of each layer: L content ( p → , x → ) = ∑ l v l L content ( p → , x → , l ) {\displaystyle {\mathcal {L}}_{\text{content }}({\vec {p}},{\vec {x}})=\sum _{l}v_{l}{\mathcal {L}}_{\text{content }}({\vec {p}},{\vec {x}},l)} , where the v l {\displaystyle v_{l}} are positive real numbers chosen as hyperparameters.

Style loss

The style loss is based on the Gram matrices of the generated and style images, which capture the correlations between different filter responses at different layers of the CNN: L style ( a → , x → ) = ∑ l = 0 L w l E l , {\displaystyle {\mathcal {L}}_{\text{style }}({\vec {a}},{\vec {x}})=\sum _{l=0}^{L}w_{l}E_{l},} where E l = 1 4 N l 2 M l 2 ∑ i , j ( G i j l ( x → ) − G i j l ( a → ) ) 2 . {\displaystyle E_{l}={\frac {1}{4N_{l}^{2}M_{l}^{2}}}\sum _{i,j}\left(G_{ij}^{l}({\vec {x}})-G_{ij}^{l}({\vec {a}})\right)^{2}.} Here, G i j l ( x → ) {\displaystyle G_{ij}^{l}({\vec {x}})} and G i j l ( a → ) {\displaystyle G_{ij}^{l}({\vec {a}})} are the entries of the Gram matrices for the generated and style images at layer l {\displaystyle l} . Explicitly, G i j l ( x → ) = ∑ k F i k l ( x → ) F j k l ( x → ) {\displaystyle G_{ij}^{l}({\vec {x}})=\sum _{k}F_{ik}^{l}({\vec {x}})F_{jk}^{l}({\vec {x}})}

Minimizing this loss encourages the generated image to have similar style characteristics to the style image, as captured by the correlations between feature responses in each layer. The idea is that activation pattern correlations between filters in a single layer captures the "style" on the order of the receptive fields at that layer.

Similarly to the previous case, the w l {\displaystyle w_{l}} are positive real numbers chosen as hyperparameters.

Hyperparameters

In the original paper, they used a particular choice of hyperparameters.

The style loss is computed by w l = 0.2 {\displaystyle w_{l}=0.2} for the outputs of layers conv1_1 , conv2_1 , conv3_1 , conv4_1 , conv5_1 in the VGG-19 network, and zero otherwise. The content loss is computed by w l = 1 {\displaystyle w_{l}=1} for conv4_2 , and zero otherwise.

The ratio α / β ∈ [ 5 , 50 ] × 10 − 4 {\displaystyle \alpha /\beta \in [5,50]\times 10^{-4}} .

Training

Image x → {\displaystyle {\vec {x}}} is initially approximated by adding a small amount of white noise to input image p → {\displaystyle {\vec {p}}} and feeding it through the CNN. Then we successively backpropagate this loss through the network with the CNN weights fixed in order to update the pixels of x → {\displaystyle {\vec {x}}} . After several thousand epochs of training, an x → {\displaystyle {\vec {x}}} (hopefully) emerges that matches the style of a → {\displaystyle {\vec {a}}} and the content of p → {\displaystyle {\vec {p}}} .

As of 2017 [update] , when implemented on a GPU , it takes a few minutes to converge. [ 8 ]

Extensions

In some practical implementations, it is noted that the resulting image has too much high-frequency artifact, which can be suppressed by adding the total variation to the total loss. [ 9 ]

Compared to VGGNet, AlexNet does not work well for neural style transfer. [ 10 ]

NST has also been extended to videos. [ 11 ]

Subsequent work improved the speed of NST for images by using special-purpose normalizations . [ 12 ] [ 8 ]

In a paper by Fei-Fei Li et al. adopted a different regularized loss metric and accelerated method for training to produce results in real-time (three orders of magnitude faster than Gatys). [ 13 ] Their idea was to use not the pixel-based loss defined above but rather a 'perceptual loss' measuring the differences between higher-level layers within the CNN. They used a symmetric convolution-deconvolution CNN. Training uses a similar loss function to the basic NST method but also regularizes the output for smoothness using a total variation (TV) loss. Once trained, the network may be used to transform an image into the style used during training, using a single feed-forward pass of the network. However the network is restricted to the single style in which it has been trained. [ 13 ]

In a work by Chen Dongdong et al. they explored the fusion of optical flow information into feedforward networks in order to improve the temporal coherence of the output. [ 14 ]

Most recently, feature transform based NST methods have been explored for fast stylization that are not coupled to single specific style and enable user-controllable blending of styles, for example the whitening and coloring transform (WCT). [ 15 ]

References