NEURAL STYLE TRANSFER: VAN GOGH - PICASSO - KLIMT STYLE

report

Andrea Nappi andi.nappi@gmail.com
Alex Orlandi alex.orlandi@icloud.com
02-03-2024

INTRODUCTION TO NEURAL STYLE TRANSFER:

Neural style transfer is a technique that belongs to the world of generative AI. It allows to generate new images that try to bond together the content of an image (content reference) and the style of another (style reference), therefore transferring the style from one image to another, using neural networks. This process not only showcases the potential of neural networks in generating visually appealing outputs but also opens doors to innovative applications in fields such as digital art and image manipulation.

THE SCOPE OF THIS ELABORATE:

The scope of this elaborate is, building upon the work by Alexis Jacq in 'Neural Transfer Using PyTorch', to create a user-friendly tool enabling the transformation of any image's style to either Van Gogh, Picasso, or Klimt style. This expansion involves modifying the code enabling it to digest multiple style references, thereby improving the model's ability to capture the essence of the chosen artist. Users will also have the option to customize their output by adjusting carefully chosen hyperparameters, yet without breaking the delicate equilibrium which is key to get a well-crafted style transfer instead of a very nice mess. The chosen artworks, selected for their embodiment of each artist's style, are:

- **Van Gogh**: “The starry night”, “Starry night over the Rhone”, “Wheat field with cypresses”
- **Picasso:** “Woman with watch”, “Girl before a mirror”, “The dream”
- **Klimt**: “The tree of life”, “The Kiss”, “Portrait of Adele Bloch-Bauer”

THE MULTI IMAGES NEURAL STYLE TRANSFER:

The developed multi-image neural style transfer possesses the capability to process any number of style images and amalgamate their most distinctive features, producing a unique artwork that preserves the essence of the original content reference while incorporating elements from multiple style references.
This section encompasses the pivotal steps that allow the “magic” to happen.
Particular emphasis is put on the tailored implementations designed to allow the extraction and integration of styles from multiple sources.

The first step consists of defining the content loss and the style loss, essential tools for evaluating the similarity between the output and the references, both content-wise and style-wise.

- **Content loss**: The content loss is computed by comparing the feature maps of the output image and the content reference using Mean Squared Error (MSE).
- **Style loss**: The style loss is computed by comparing the Gram matrices of the output image and the style reference. A Gram matrix is a square matrix derived from the inner products of feature vectors extracted from an image, effectively capturing its style. This comparison is also conducted using Mean Squared Error (MSE).

After defining the losses, it is essential to incorporate them within a sequential model, giving birth to the actual style transfer model. Here, a pre-trained CNN serves as the architectural foundation, with content and style loss layers integrated after carefully selected convolutional layers. Whenever a convolutional layer that has been selected as a style loss layer predecessor is encountered, a style loss layer gets inserted for each of the style references. Based on the chosen convolutional layers’ features, the losses are computed, and then during the appropriately displayed loss layers they are assigned to an item that keeps track of them

The final major step involves defining the gradient descent process. At each step, the model computes the total loss between the current output and all the references by summing the current content score and the current style score.
The content score at each step is determined as the sum of the content losses computed during the whole step, scaled by the number of times the content loss is computed, and ultimately weighted by a “content weight”, which is the hyperparameter that defines the importance given to the content presence in the final output.
The style score at each step is calculated as the sum of all the style losses computed during the step. Before the aggregation, each single style loss is scaled by the style weight associated with its reference (which is the hyperparameter that defines the importance given to that style's presence in the final output) and by the number of times the style loss is computed (for a single image).
Scaling by the number of times the losses are computed is not necessary, as the effect it generates can be compensated by the content/style weight hyperparameter, but including this factor can still be beneficial as it allows the weights to be chosen in a range of values that grants more control over the scores, resulting in more stability and less chances to end up in a vanishing/exploding gradient scenario.

What the transfer style model tries to do is to lower the final loss value by adjusting its parameters, so, practically speaking, generating a new picture that is more similar in content+style to the references.

Ultimately, to run the style transfer, the following hyperparameters are needed:

- Initial CNN
- Normalization mean
- Normalization standard deviation
- Content image
- Style image(s)
- Input image
- Style weight(s)
- Content weight
- Number of steps
- Content layers
- Style layers
- Optimizer

Among these hyperparameters, some are suggested to keep pre-defined and tested values (and in some cases a change in them should be matched with minor changes in the style transfer architecture), including:

- The initial CNN, which is necessary to build the style transfer model, the proposed one is VGG19.
- The normalization mean and the normalization standard deviation, used to preprocess and standardize the input image before passing it to the model, the used ones are respectively [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225].
- The number of steps, which is the number of times the model will try to improve the outcome before presenting the ultimate version, the proposed value is 300.
- The content layers and the style layers, which are the convolutional layers after which it is intended to include loss layers, the suggested values are the 4^{th} convolutional layer for the content loss and the 1^{st}, 2^{nd}, 3^{rd}, 4^{th} and 5^{th} convolutional layers for the style loss, it is to be noted that this choice is tight with the initial CNN choice.

For the remaining hyperparameters, an ad hoc function has been designed, which asks to be given:

- The painter whose style is desired (among van Gogh, Picasso, and Klimt).
- The image from which the content has to be kept.
- The image from which the model should start doing the work.
- Eventually, a content faithfulness hyperparameter that can marginally impact the presence of the original content in the final product.

Such function enables the user to choose the style he wants to recreate and have some additional influence on the final output, but confined by carefully designed borders, so that a satisfactory result is guaranteed. Based on this information, the function will provide:

- The style references, which are the pictures from which the style will be captured; they are strictly related to the choice of the painter.
- The content reference, which is the picture from which the content will be taken; it is given directly by the user.
- The input image, which is the picture from which the process will start operating (it is highly suggested, due to the parametrization of the weights, that the input image matches the content image); it is given directly by the user.
- The style weight for each style reference, which is the importance given to each picture's style and to the overall presence of style in the ultimate result; they are strictly related to the choice of the painter.
- The content weight, which is the importance given to the presence of the content in the ultimate result; it is related to the choice of the artist and, if present, to the content faithfulness parameter, which can alter the content weight's value by up to 20%.
- The optimizer, which is always the Limited-memory Broyden–Fletcher–Goldfarb–Shanno optimizer; it depends on the input image.

THE OUTCOMES:

Here, some of the obtained results are presented, together with the hyperparameters that generated them:

1. VAN GOGH STYLE: THE SCREAM (MUNCH):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “The Scream” (Munch)
- **Style image(s):** “The starry night”, “Starry night over the Rhone” and “Wheat field with cypresses”
- **Input image**: “The Scream” (Munch)
- **Style weight(s)**: respectively 200, 150, 150
- **Content weight**: 0.001
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Van Gogh):

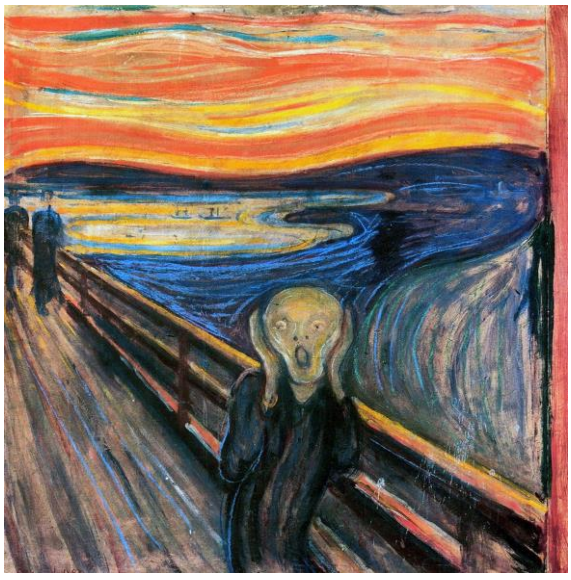
The starry night

Starry night over the Rhone

Wheat field with cypresses

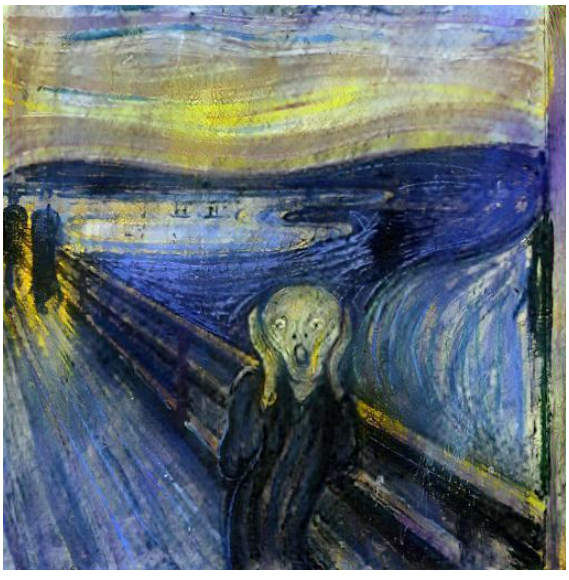
Content reference (Munch):

The Scream



Outcome:

The Scream – Van Gogh style



2. PICASSO STYLE: THE SCREAM (MUNCH):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “The Scream” (Munch)
- **Style image(s):** “Woman with watch”, “Girl before a mirror” and “The dream”
- **Input image**: “The Scream” (Munch)
- **Style weight(s)**: respectively 200, 150, 150
- **Content weight**: 0.002
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Picasso):

Woman with watch

Girl before a mirror

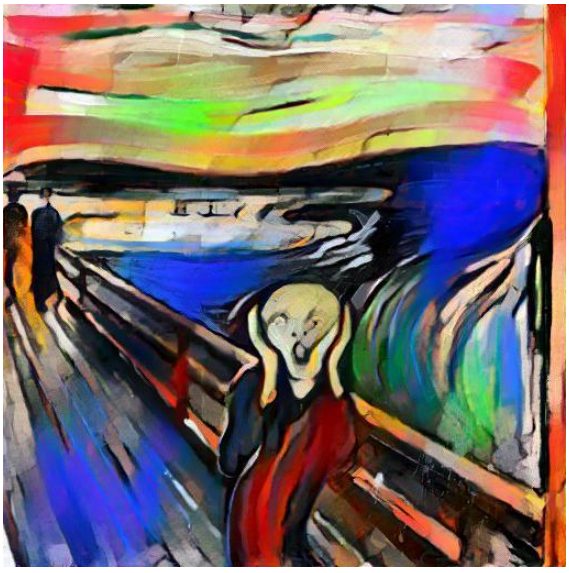
The dream

Content reference (Munch):

The Scream

Outcome:

The Scream – Picasso style



3. KLIMT STYLE: THE SCREAM (MUNCH):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “The Scream” (Munch)
- **Style image(s):** “The tree of life”, “The Kiss” and “Portrait of Adele Bloch-Bauer”
- **Input image**: “The Scream” (Munch)
- **Style weight(s)**: respectively150, 150, 200
- **Content weight**: 0.0003
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Klimt):

The tree of life

The kiss

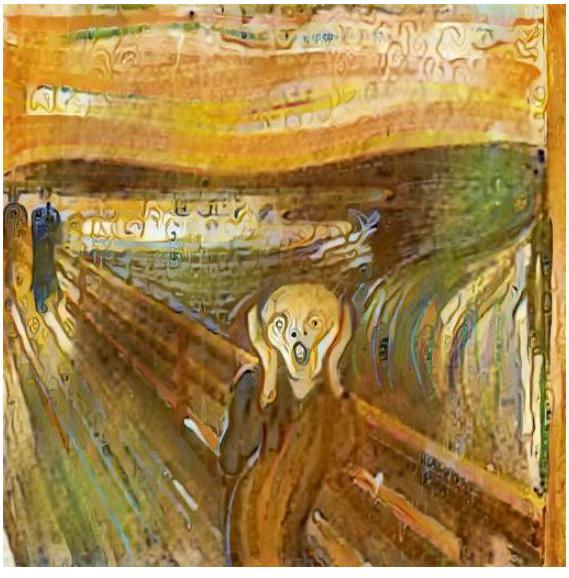
Portrait of Adele Bloch-Bauer

Content reference (Munch):

The Scream

Outcome:

The Scream – Klimt style



4. VAN GOGH STYLE: JAPANESE FOOTBRIDGE (MONET):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “Japanese footbridge” (Monet)
- **Style image(s):** “The starry night”, “Starry night over the Rhone” and “Wheat field with cypresses”
- **Input image**: “Japanese footbridge” (Monet)
- **Style weight(s)**: respectively 200, 150, 150
- **Content weight**: 0.001
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Van Gogh):

The starry night

Starry night over the Rhone

Wheat field with cypresses

Content reference (Monet):

Japanese footbridge

Outcome:

Japanese footbridge – Van Gogh style

5. PICASSO STYLE: JAPANESE FOOTBRIDGE (MONET):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “Japanese footbridge” (Monet)
- **Style image(s):** “Woman with watch”, “Girl before a mirror” and “The dream”
- **Input image**: “Japanese footbridge” (Monet)
- **Style weight(s)**: respectively 200, 150, 150
- **Content weight**: 0.002
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Picasso):

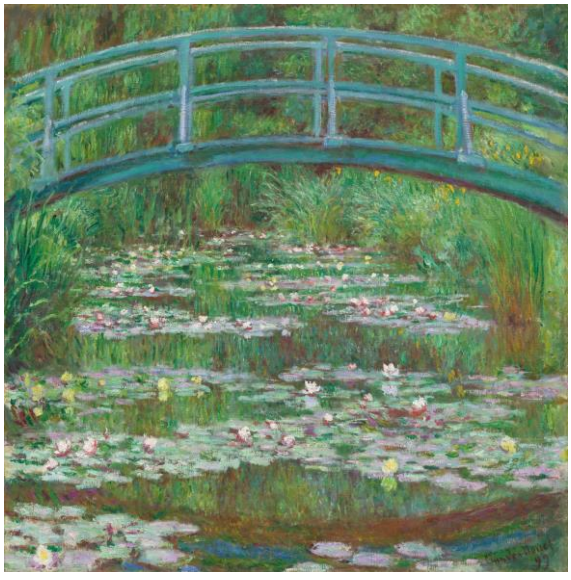
Woman with watch

Girl before a mirror

The dream

Content reference (Monet):

Japanese footbridge



Outcome:

Japanese footbridge – Picasso style

6. KLIMT STYLE: JAPANESE FOOTBRIDGE (MONET):

- **Initial CNN**: VGG19
- **Normalization mean**: [0.485, 0.456, 0.406]
- **Normalization standard deviation**: [0.229, 0.224, 0.225]
- **Content image**: “Japanese footbridge” (Monet)
- **Style image(s):** “The tree of life”, “The Kiss” and “Portrait of Adele Bloch-Bauer”
- **Input image**: “Japanese footbridge” (Monet)
- **Style weight(s)**: respectively150, 150, 200
- **Content weight**: 0.0003
- **Number of steps**: 300
- **Content layers**: 4th Convolutional layer
- **Style layers**: 1st, 2nd, 3rd, 4th and 5th convolutional layers
- **Optimizer**: Limited-memory Broyden–Fletcher–Goldfarb–Shanno

Style references (Klimt):

The tree of life

The kiss

Portrait of Adele Bloch-Bauer

Content reference (Monet):

Japanese footbridge

Outcome:

Japanese footbridge – Klimt style