CS230

Art Nouveau Style Transfer with Face Alignment

Elena Tuzhilina*
Department of Statistics
Stanford University
elenatuz@stanford.edu

Abstract

Flourished throughout Europe and the United States at the turn of 19th and 20th centuries, Art Nouveau still remains one of the most beautiful decorative art movements. Promulgating the idea of art and design as part of everyday life and inspired by natural forms and patterns of plants and flowers, it has influenced different aspects of art and architecture, such as interior, furnishings and glass design, as well as graphic work, posters, and illustration. Furthermore, this arts-and-architecture movement blurred the line between fine art and mass production and completely transformed the approach to advertising making the Art Nouveau aesthetic even more accessible to the public.This project inspired by Henri de Toulouse-Lautrec and Alphonse Mucha works of art is aimed to develop a deep learning tool transforming already boring advertisements into a bright and bold Art Nouveau fine art posters. Even though this was the initial motivation of the project, the method proposed is not limited to application to advertisement only, but also can be used as a photo filter.

1 Introduction

To achieve this task the Neural Style Transfer was chosen as a baseline method. Although this Neural Network technique can produce an efficient and visually appealing style transformation for landscapes, it is prone to face distortion in the case of portrait images. This was the main challenge of this project as most of the images in the dataset under consideration, i.e. Art Nouveau posters, depict young women advertising some products. To address this challenge I have introduced an additional Face Detection step to the Neural Style Transfer approach that was, subsequently, used for a preliminary images alignment. The resulting method takes a pair of input images *(content, style)* and aligns them according to the detected faces; the images $(content_{aligned}, style_{aligned})$ are further passed through NST network, which produces the output representing the resulting Art Nouveau stylized image (see Section 4).

2 Related work

The original and pioneering work on Neural Style Transfer by Gatys et al. (see (17)) has introduced the idea to use VGG-16 neural network to extract features for style and concept images and to create the content and style losses function based on these features by themselves and the Gram matrix of features, respectively.

One downside to the approach proposed by Gatys et al. is that it produces a lot of high frequency artifacts. One can use an explicit regularization to decrease the presence of high frequency components in the resulting image. For example, is is common to introduce a total variation loss into the scheme. Moreover, some authors propose to add a per-pixel loss between the output and input images to further improve the resulting stylized image (see, for example (18)). It is also common to add regularization to the baseline method to improve the neural network performance and convergence speed.

Another point to highlight here is that, depending on the data and the task, one can consider not only adding a regularization, but also to modify the style and content losses by themselves. For example, propose to use Markov random field models to construct the loss function for both photographic and non-photo-realistic (artwork) synthesis tasks.

*https://web.stanford.edu/ elenatuz

CS230: Deep Learning, Fall 2019, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

There are also many great tutorials on Neural Style Transfer method and its modifications that can be useful exploring this topic (see, for example, (13; 14; 15; 16)).

3 Dataset and Features

3.1 Art Nouveau data

This project requires two datasets: an Art Nouveau data set and the target images data set that we will transfer to fine art posters. Initially, I was planning to download the first part of the data from "Painter by numbers" Kaggle competition (9). This data contains 80Gb of images in .png format downloaded from the (10) representing the paintings of about 2000 various artists. Using the data description available on Kaggle website, it is possible to filter Art Nouveau style paintings resulting in about 5000 paintings for more than 50 different artists. However, after a more detailed expertise of the downloaded data, I have discovered that the Kaggle's definition of "Art Nouveau" is very broad, and that most of the authors included in this dataset, actually, are far from the classical Art Nouveau style. So I have extracted Alphonse Mucha paintings only to make the style as homogeneous as possible. The Mucha dataset from Kaggle contained 200 images.

3.2 Target data

For the second part of the data I have used Flickr Images. All of the images by Alphonse Mucha in our dataset depict woman and, more specifically, woman in long dresses. Therefore, the challenge was to find a good tag that will produce suitable Flickr data. I have tried various tags combinations: "advertisement" and "ad"; more specific "advertisement, perfume" and "advertisement, makeup"; tags narrowing the result to women only, like "advertisement, women" or "advertisement, dress", etc. I have ended up with the tag "women, vintage dress" that produces more or less appropriate content.

Second challenge was to download the data from Flickr according to the chosen tag. I have used the methodology described in (12), writing two separate scripts, one to create the list containing corresponding filtered Flickr image URLs, another to download the images from the saved URLs (see `Flickr code//flickr_GetUrl.py` and `Flickr code//get_images.py` in the repository). The resulting data set contains about 2000 "content" images.

Comment: All of the images in both style and content datasets are *.jpg* images of different resolution. Since the Neural Style Transfer approach requires the input style and content images to be of the same size the data requires some additional prepossessing. This preprocessing will be done later at the alignment step of the method proposed (see 4.2.1).

4 Methods

4.1 Neural Style Transfer

As a baseline method in the realization of this project I have used Neural Style Transfer (17) that could be useful if you want to capture the content of one image and combine it with the style of another image. This approach requires two images so-called "style image" and "content image"; the general idea of Neural Style Transfer is as follows. First, use a pretrained VGG-16 network extracting an encoding for both style and content images, then, generate a new image based on the encoding obtained. The generation is based on two cost functions $L_{content}(C, G)$ and $L_{style}(S, G)$ referring to the "similarity" of the generated image with the content and style, respectively. Here, S, C and G denote the content, the style and the generated image. To put these losses in mathematical form denote output of ℓ-th activation layer (usually, L is the total number of layers, but we will consider L to be some value, less or equal to the total number of layers, see Section 5) of VGG-16 net for the corresponding image by $a[\ell](\cdot)$. Further, denote the gram matrix of layer ℓ by $GM[\ell](\cdot)$ measuring the correlation of all the channels of layer ℓ with respect to each other. Thus

$$L_{content} = \frac{1}{2}\|a[L](C) - a[L](G)\|_2^2 \;\; \textit{and} \;\; L_{style} = \sum_{l=1}^{L} \frac{\|GM[\ell](S) - GM[\ell](G)\|_F^2}{\#\textit{elements in } GM[l](\cdot)}.$$

The optimization goal is to minimize the weighted sum of these two losses, i.e.

$$L(G) = \alpha L_{content}(C, G) + \beta L_{style}(S, G).$$

There are several modification of Neural Style Transfer (NST), in particular, I have tried Neural Style Transfer with regularization in the form of Total variation loss (RST). This approach calculates the total variation of the generated image $TV(G)$ and optimizes the combined loss

$$L(G) = \alpha L_{content}(C, G) + \beta L_{style}(S, G) + \gamma TV(G),$$

thereby imposing "smoothness" on the resulting stylized image. Finally, I have applied Fast Neural Style (FST) (7) method to generate the image. After 20 epochs, each 100 steps, the following results were produced (see `Style Transfer demo.ipynb`):

(a) Content (b) Style (c) NST (d) FST (e) RST

Figure 1: Neural Style Transfer results for $\alpha = 10^4$, $\beta = 10^{-2}$ and $\gamma = 30$

4.2 Adding Face detection

One challenge of the Style Transfer approaches under consideration, apparent from the generated images, is the face distortion. To make the generated face appear more realistic, I have decided to implement a "smart" alignment of the style and content images. This alignment is based on the MTCNN face detection algorithm (8). For each face detected on the image MTCNN provides the following output (see `Face detection.ipynb`): a bounding box with its left-top corner coordinates, its height and width; the confidence of face detection; and the coordinates of facial features. An example of the output format is:

```
[{'box': [192, 188, 93, 121], 'confidence': 0.9992227554321289, 'keypoints': {'left_eye':
(218, 234), 'right_eye': (264, 239), 'nose': (237, 265), 'mouth_left': (217, 277),
'mouth_right': (256, 281)}}].
```

In this section I propose two approaches to alignment and subsequent cropping of style and content images based on the information obtained via MTCNN.

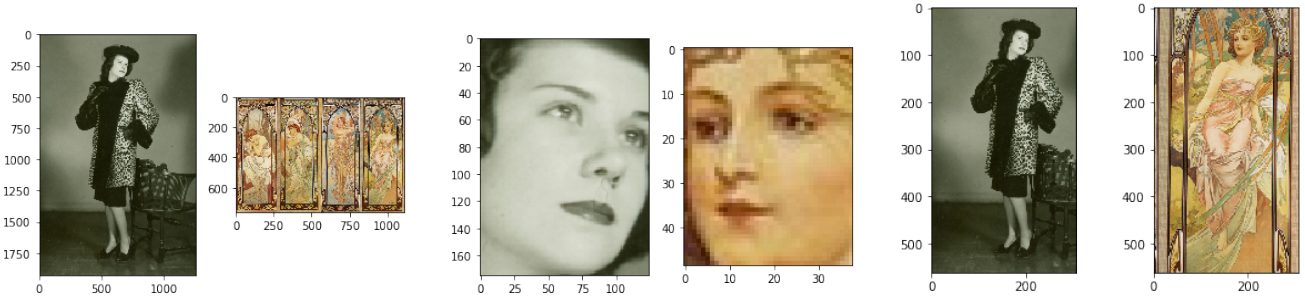
Comment: In addition to content and style faces alignment, MTCNN output provides us with an opportunity to filter the noisy Flickr data via choosing images with a high enough confidence level of face detection (variable `'confidence'`).

4.2.1 IOU alignment

The *IOU alignment* is aimed to shift and scale the content and style images in such a way that superposing the faces of both input images produces the highest value of Intersect over Union (see `IOU alignment.ipynb`). I proceed as follows:

1. Use MTCNN to detect the faces on the content and style images. Note that, for an image several faces can be detected by MTCNN; one can use the `'confidence'` variable to filter "irrelevant" results. In this particular approach for each image I pick only one face box with the highest confidence value.

2. One detail to take into account is the, possibly, different scales and positions of the two detected faces. I find the linear transformation of the content and style images such that it maximizes the IoU of the transformed bounding boxes. Note that to avoid scaling up of the images and thus losing image quality the scaling is applied to the image with "largest" face box detected (scaling down this image).
3. The style image, especially in the case of Art Nouveau, often contains a lot of stylistic features on its boundary. Therefore, I crop only the "necessary" parts of the aligned pictures, i.e. I crop the aligned content and style images in such a way that leads to the same size whereas minimizing removed information.

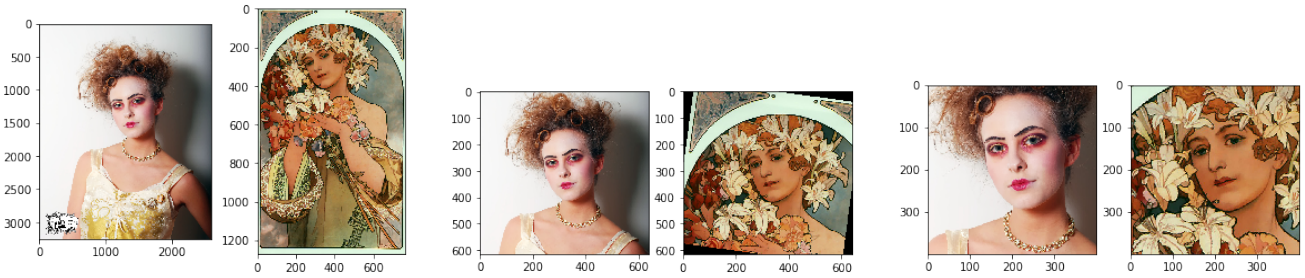


(a) Original (b) Detected faces (c) Align and crop

4.2.2 Procrustes alignment

The MTCNN output contains not only the boxes bounding the face, but also the locations of such face features as eyes, nose and lips (variable `'keypoints'`). The *Procrustes alignment* is aimed to use the locations of these face parts for style and content images to find an affine transformation that aligns them best. Note that this approach allows not only to shift and scale but also rotate the content and style images. The potential benefit of Procrustes alignment is that it may mitigate the distortion of particular face features (see `Procrustes alignment.ipynb`). This approach involves the following steps:

1. Use MTCNN to detect the facial features on the content and style images, again, pick only one detected face with the highest confidence value. Create the matrices $X_{content}, X_{style} \in \mathbb{R}^{5\times 2}$ containing the coordinates for the 5 `'keypoints'` under consideration.
2. Solve the Procrustes optimization problem, i.e. find shifting vector b, scaling constant s and rotation matrix R that minimizes the distance between aligned matrices, in other words, the goal is to optimize the objective $\|X_{content} - s \cdot X_{style}R - b\|_F$.
3. Use b, s and R to scale shift and rotate the content and style images. Take into account the comments about relevant scaling from IOU alignment. Note that since the rotation was introduced into the scheme, to be able to apply this transformation it is necessary to put the face into the center of the aligned figure.
4. Crop the images to the same size minimizing removed information.



(a) Original (b) Detected faces (c) Align and crop

4.3 Combining the NST with Face Detection

Since the content and style images are already aligned and share the face location, we are now able to impose a penalty on the part of image corresponding to face separately from the whole image. I have decided to modified the loss function and restrict the total variation penalty to the face box only. The main motivation was to construct the loss that will keep as many stylistic

features of the background as possible whereas reducing the high frequency stylistic artifacts in the face area. Hence the final loss function is

$$L(G) = \alpha L_{content}(C_{aligned}, G) + \beta L_{style}(S_{aligned}, G) + \gamma TV(G\big|_{face\ box}).$$

5 Experiments

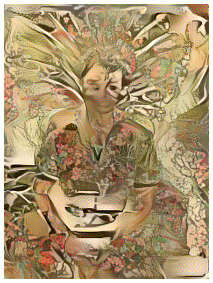
To tune the hyperparameters I have chosen a pair of images and for each set of parameters I have trained the neural network for 20 epochs, each 100 steps. The resulting generated images was stored in the `Results` and was used to evaluate the ability of neural network under consideration to generate a new image (see `Hyperparameter tuning.ipynb`).

Learning rate I have started considering different learning rates such as $lr = 0.002, 0.05, 0.1$ and tracked the convergence speed in each case. According to my data, the "standard" 0.1 value for the learning rate provides with the best results.

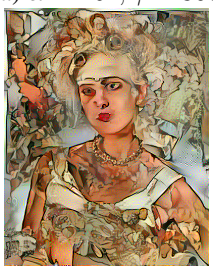
VGG-16 Layers I have done several experiments varying the VGG-16 layers used to create the features for style and content images. In this part I have considered L_c and $L_s \in \{1, \dots, 5\}$ the "depth" of the layers considered for the content and style, respectively. According to the images presented in `Results//Structure` For small value of L_s and L_c there is no significant difference between the resulting image and the original content image. Therefore, the optimal chosen value was $L_s = L_c = L = 5$.

Weights The main hyperparameters in my experiment were the loss weights α, β and γ. Varying these parameters one can control the amount of stylistic and content features in the resulting image as well as the smoothness of the generated face. The style loss weight β was always set to 1 and the log-scale was chosen for α and γ, i.e. $\alpha = 10^3, \dots, 10^6$ and $\gamma = 0, 30, 300, 3000$ where $\gamma = 0$ corresponds to the absence of the total variation penalty. According to the images presented in `Results//Weights` the optimal weights combination is $\alpha = 10^5, \beta = 1, \gamma = 300$ or $\alpha = 10^6, \beta = 1, \gamma = 3000$.

(a) $\alpha = 10^3, \gamma = 300$



(b) $\alpha = 10^4, \gamma = 0$

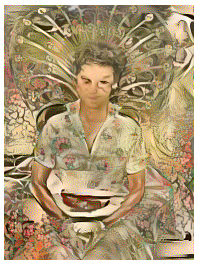


(a) $\alpha = 10^4, \gamma = 300$

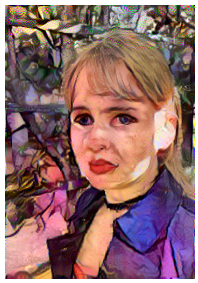
(b) $\alpha = 10^5, \gamma = 0$

5.1 Results

We use the discovered optimal parameter and run NST with facial total variation regularization equipped with IOU and Procrustes alignments. This time the neural network is trained for 50 epochs. The results presented in the table (left image - IOU, right image - Procrustes). For hight resolution images check `Results` in the repository.





6 Conclusion

In this research I consider the problem of Art Nouveau Style transformation. To resolve the face distortion problem common for NST group of approaches in the case of portrait images, I propose two ways to use Face Detection to improve the performance of Neural Style transfer. The *IOU alignment* uses the MTCNN face box to ship and scale the content and style images, whereas *Procrustes alignment* utilizes the facial keypoints and accounts for the rotation dissimilarities as well. The approach under consideration allows to mitigate the face distortion in some cases and according to conducted experiments, the alignment that works better for this task is the Procrustes one. There is still a lot of space for improvement: for example, one interesting direction of further research is to combine the Fast Neural Style Transfer with the suggested technique.

References

Python

[1] *Tensorflow* https://www.tensorflow.org/

[2] *Cv2* https://pypi.org/project/opencv-python/

[3] *Numpy* https://numpy.org/

[4] *Pandas* https://pandas.pydata.org/

[5] *Pillow* https://pillow.readthedocs.io/en/3.1.x/index.html

[6] *SciPy* https://scipy.org/

GitHub repositories

[7] *Fast Neural style* https://github.com/jcjohnson/fast-neural-style

[8] *MTCNN* https://github.com/ipazc/mtcnn

Data

[9] *"Painter by numbers" Kaggle competition*. https://www.kaggle.com/c/painter-by-numbers/data

[10] *Visual art encyclopedia Wikiart*. www.wikiart.org

[11] *Flickr vintage posters dataset* https://www.flickr.com/groups/659012@N23/pool/page2

Tutorials

[12] *How to download Flickr images* https://towardsdatascience.com/how-to-use-flickr-api-to-collect-data-for-deep-learning-experiments

[13] *Neural Style Transfer Tutorial 1* https://www.tensorflow.org/tutorials/generative/style_transfer

[14] *Neural Style Transfer Tutorial 2* https://towardsdatascience.com/neural-style-transfer-tutorial-part-1

[15] *Face swap with Neural Style Transfer* https://blog.paperspace.com/style-transfer-part-2/

[16] *Neural Stylr Transfer with different losses* https://towardsdatascience.com/experiments-on-different-loss-configurations-for-style-transfer

Articles

[17] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, *A Neural Algorithm of Artistic Style* (2015), arXiv, https://arxiv.org/abs/1508.06576

[18] Justin Johnson, Alexandre Alahi, Li Fei-Fei *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*(2016), arXiv, https://arxiv.org/abs/1603.08155

[19] Chuan Li, Michael Wand, *Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis*(2016), arXiv, https://arxiv.org/abs/1601.04589

Miscellaneous

[20] *Procrustes Alignment* https://en.wikipedia.org/wiki/Procrustes_analysis