Abstract

Nowadays, there are many methods for neural style transfer (NST) task. In our project, we chose two model-optimization-based method – Per-style-per-model and arbitrary-style-per-model. The first one need to train different models for different style pictures. The second method only trains one model to fit all the styles. We compared these two models and evaluate them in the situation of limited dataset and hardware. And we found that these models can work well on different styles.

Introduction

Neural style transfer (NST) is a neat idea to demonstrate that artificial intelligence can play a part in producing images with artistic attributes. NST builds on the key idea which extract style information form artistic images and recombine them with the content extracted from other natural images to form a new image that both resembles the content image as well as seemingly been "artistically produced" by a professional artist.

Related Works:

Gatsy and his colleagues lay the foundation of neural style transfer, however, his approach to jointly minimizing the feature reconstruction loss of content and style in order to optimize the output image quality. The images they produced are high in resolution, yet also requires high levels of computational power.

Johnson et al, however, trained a feed-forward network in response to this burden in computation in order to reach a faster and more efficient outcome in optimizing styled images. In his Fast Style Transfer Architecture, Johnson et al (2016) used a pre-trained VGG-16 as Loss Network to define Perceptual Losses. He also defined two perceptual loss functions to measure high level perceptual and semantic difference between images. The first function is the Feature Reconstruction Loss Function, where φ_j is the output at the jth layer of the network φ for the input x.

$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$$

The second function is the Style Reconstruction Loss Function, where G_j is the gram matrix of the jth layer.

$$\ell_{style}^{\phi,j}(\hat{y}, y) = \|G_j^\phi(\hat{y}) - G_j^\phi(y)\|_F^2.$$

In a more recent research, T. Q. Chen and M. Schmidt (2016) defines a new optimization objective for style transfer while only depends on one layer of the convolutional neural network. In their optimization function: C and S represent the RGB representations of the content and style images, respectively.

$$\phi_i^{ss}(C, S) := \underset{\phi_j(S),\, j \in N_s}{\arg\max} \frac{\langle \phi_i(C), \phi_j(S) \rangle}{||\phi_i(C)|| \cdot ||\phi_j(S)||}.$$

And their stylized image computation is achieved though placing a loss function on the activation space with target activations $\Phi_{ss}(C, S)$.

$$I_{stylized}(C,S) = \underset{I \in \mathbb{R}^{3\times H\times W}}{\arg\min} ||\Phi(I) - \Phi^{ss}(C,S)||^2 + \lambda \ell_{TV}(I)$$

The use of an 'Inverse Network': T. Q. Chen and M. Schmidt also proposed another way of optimization, which is to train an inverse network to approximate the optimum of the loss function.

$$\min_{\widehat{\Phi^{-1}}} \frac{1}{n} \sum_{j=1}^{n} ||\Phi(\widehat{\Phi^{-1}}(\Phi_j)) - \Phi_j||^2 + \lambda \ell_{TV}\left(\widehat{\Phi^{-1}}(\Phi_j)\right).$$

Dataset

For our content dataset, we use the AISegment.com - Matting Human Dataset. It is a high resolution extraction of humans from images. This dataset, developed by AISegment aims to help by providing a solid quality dataset of images and masks. And for our style dataset: A Collection of digital illustrations extracted from Pixiv.net. All the image credit goes to Pixiv artists: Sakimori, Hinata, and echowater.

Methods:

In this project, we are looking into two different methods of NST, one is the per-style-per-model, and the other is the arbitrary-style-per-model. We aim to reproduce the image-styling process of these two models. And the result will be evaluated both quantitatively and qualitatively. For the quantitative evaluation, we are looking at the training time as well as the "speed" of producing output images. And for the qualitative evaluation, we will manually compare the end quality of the result images to find out which method produces the optimal image reconstruction quality.

Result

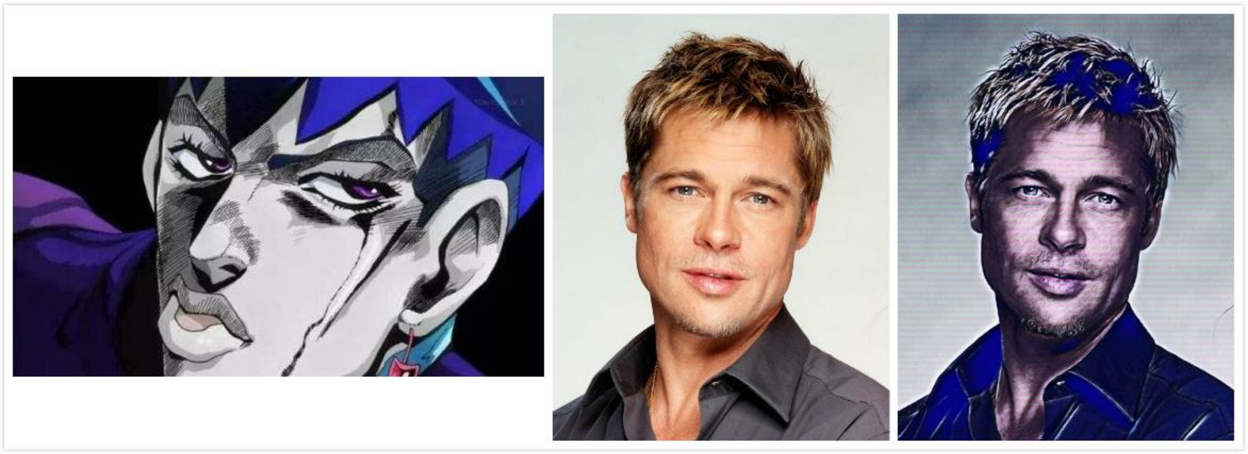
1. Model 1

In this method, the training set of content is 463 pictures of human faces and resize these pictures to 256*256. We train this model with batch_size=4, giving 5 epochs over the training data. Learning rate is 0.002. Training time is about 4 mins and 40secs.

Weights:

	Content_weight	Style_weight	TV_weight
Style 1	1.0	3.0	0.01
Style 2	1.0	15.0	0.01

Because the training set is focusing on faces, so we found the model works well on changing the style of faces. However, it's not so good on other parts of the pictures.



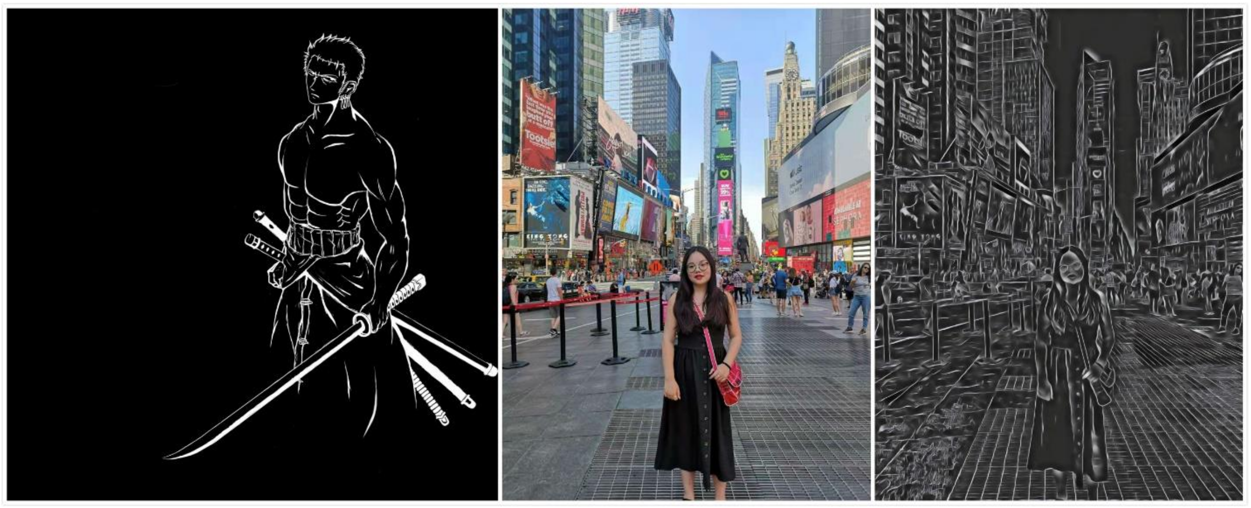
1. Model 2

The training pictures are resized to 256x256. We train this model with batch = 4, giving 5 epochs over the training data. Learning rate: 0.001 TV_weight = 0.0001.

It takes about 1 hour to train the model with training set size equals to 10,000, but once the training finished, it only takes a few seconds to do the style transfer with a GTX1070.

When the dataset is small, it is better to use the Per-style-per-model approach for NST.

The Arbitrary-style-per-model works very well if we want to transfer the picture style to an extremely distinctive style. But the performance is not so good if we need a complex style.



Conclusion:

According to the results shows above, we can find directly that the arbitrary-style-per-model works well on learning the color features in styles, on the other hand, the per-style-per-model has nice performance on the extraction of the texture features in certain styles, it can learn the "brushwork" of the styles and then apply it to the content.

The arbitrary model requires a long training time because it will uses all the style images as the training set, but it only requires one training process. Meanwhile, for each style the per-style-per-model needs to train a separate model, so this model needs several training processes, but each process requires less time than the arbitrary model.

As for which to choose between the two models, it depends on what kind of style transfer problem we want to solve. With a small training set, it is better to choose the per-style-per-model. In contrast, with sufficient amount of training data in a distinctive style, we will get a better performance on the arbitrary-style-per-model.

Reference:

- J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in European Conference on Computer Vision, 2016, pp. 694–711
- T. Q. Chen and M. Schmidt, “Fast patch-based style transfer of arbitrary style,” in Proceedings of the NIPS Workshop on Constructive Machine Learning, 2016
- https://github.com/elleryqueenhomels/fast_neural_style_transfer
- https://github.com/Joanhxp/Style-Swap