Generating Fashion Through Neural Style Transfer

Luis M. Alcaraz
alcaraz@stanford.edu

Robert Hu
roberthu@stanford.edu

Aayush Agrawal
aayush2k@stanford.edu

1 Introduction

Machine learning algorithms have been lauded for how quickly they have been able to surpass human benchmarks. It has already penetrated and revolutionized industries that revolve around data; however, machine learning can also be used to elevate industries in which human creativity has dominated - like fashion. Traditionally, designers have relied on the tried and tested process involving research, idea development, concept, and pattern making.

In this project, we propose a new fashion-designing framework: a system that is able to extrapolate the style of designs and artworks to existing shirts/tops. As a result, we were able to generate new ideas and designs for shirts/tops. Additionally, the system outputs this transformation from images of shirts/tops - without being provided input-output pairs.

2 Related Works

Prior research conducted by Gatys allowed for image representations to be encoded through VGG model layers, which separated the content and style of images before recombining them to form new images [5]. Gatys' work involved extracting style as a weighted set of gram matrices and feature maps of content taken from higher layers of the network.

Meanwhile, Johnson has created an image transformation network, which transforms images with different styles [7]. Johnson's work trained a feed-forward transformation network using perceptual loss functions, which worked in real-time. They used a pre-trained loss network to reduce the time spent training as opposed to using a per-pixel loss function approach.

3 Data

Since our project uses principles of style transfer, we worked with two different data sets. The first consists of 44,441 unique images of articles of clothing. However, we did not use all 44,000 images, as we wanted to focus solely on shirt and top design manipulation; thus, our first data set consists of around 15,000 unique images of clothing. A current limitation of this data set is that a small percentage of the images contain humans wearing the shirt. We only want images that solely contain the fashion accessory for the purpose of style transfer, and we filtered out pictures with humans. We obtained the data set from Kaggle: https://www.kaggle.com/paramaggarwal/fashion-product-images-dataset

The second data set contains the various styles that we wanted to extrapolate onto the images of shirts and tops. As a result, we chose a data set that contains artwork from 50 influential artists including Vincent Van Gogh, Jean-Michel Basquiat, and Frida Kahlo. There are roughly 16,800 different artworks, which are all viable candidates for the style transfer. A limitation of this data set is that the artworks are all paintings, which we realize is limited in terms of scope. We obtained the data set from Kaggle: https://www.kaggle.com/ikarus777/best-artworks-of-all-time

3.1 Data Processing

The images from the Fashion Dataset have an input size 2400x1800x3 (3 pertaining to the RGB values). Since our algorithm takes in sizes of 400x300, we resized the input images to that size. We also normalized pixel values for faster training.

4 Approach

Our method involves the use of Neural Style Transfer. This technique consist of merging two images, one "content" image and one "style" image, to generate a new image that contains the style from the style image but infused into the content image. For future reference, we will denote the content image as (C), the style image as S, and the generated image as (G). To accomplish this we utilize the following formula:

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

where $J_{content}$ is written like:

$$J_{content}(C,G) = \frac{1}{4 * n_H * n_W * n_C} * \sum_{all-entries} (a^{(C)} - a^{(G)})^2.$$

where J_{style} is written like:

$$J_{style}(S,G) = \frac{1}{4 * (n_C)^2 * (n_H * n_W)^2} * \sum_{i=0}^{n_C} \sum_{j=0}^{n_C} (G^{(S)}_{(gram)ij} - G^{(G)}_{(gram)ij})^2.$$

The total style loss across each layer will then be:

$$L_{style}(S,G) = \sum_{l \in L} w_l J_{style}(S,G).$$

We will weight each layer equally and thus:

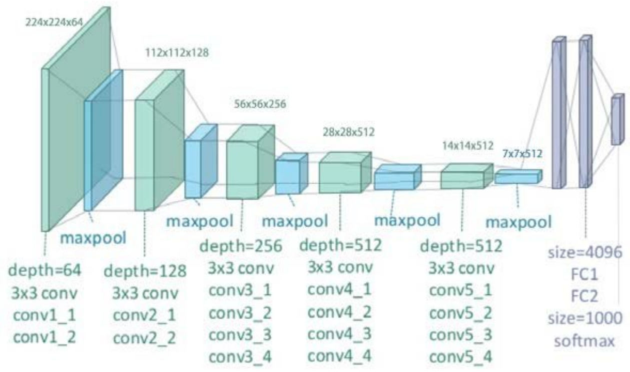
$$w_l = \frac{1}{||L||}.$$

In these formulas we can see the use of hyperparamenters α, β, and the learning rate. Lanston Chu explores the selection of hyperparameters in the context of style transfer [3]. He concludes that when the learning rate is small, the optimizer takes longer in finding the local optima, infusing the content image with little 'taste' from the style image. However, when the learning rate is high, the optimizer fails to converge, generating an image that no longer resembles the content image. Furthermore, Chu highlights the reason $\alpha = 10$, $\beta = 40$ is so commonly used in the field. A drastic change in either α or β will result in a generated image that closely resembles either the content or style image, respectively. In addition, we also found that a change in the learning rate achieves these same conclusions (no/little style transferred or content overwritten). From these discoveries, we have chosen to set the hyperparameters to the industry standard. However, to achieve a more inconspicuous generated fashion piece (or a more lively piece) tuning of these hyperparameters will achieve that.

In terms of the style image, cost consists of the use of Gram matrices which represents how similar the row of the matrix is to the column of the matrix while the content image is comprised of the hidden layer activations in the current layer. These equations are utilized to compute the cost after running the respective images through a VGG19 model. Furthermore, we will utilize the Adam optimizer to minimize J(G), slowly tuning the pixel values for the desired outcome.

4.1 Architecture

In the implementation of our algorithm, we chose to utilize a pre-trained model, VGG19, in order to generate a version of the content image with style derived from the style image. Using a pre-trained model allowed us to focus on achieving style transfer without having to worry about creating a CNN model from scratch [2]. This concept is known as transfer learning. The VGG19 model, a successor to the AlexNet model [1], is a neural network that consist of 19 layers. Its architecture can be better represented in the following image:



We have utilized this model due to the fact that it can recognize low level features at the shallower layers and high level features at the deeper layers. (Coursera) Furthermore, this same model was also used in Gatys' paper that introduced style transfer. Although there are other models like ResNet and Inception, we believe that the VGG model worked better with our situation. As was explored by Reiichiro Nakano in his paper Adversarially Robust Neural Style Transfer, where he discovers that non-VGG models fail to work when applying the models to achieve style-transfer, due to the fact that VGG models are slightly more robust than ResNet models.

5 Results and Discussion

After inputting various shirts and tops and combining them with random artworks from our dataset, we were able to generate new fashion styles.

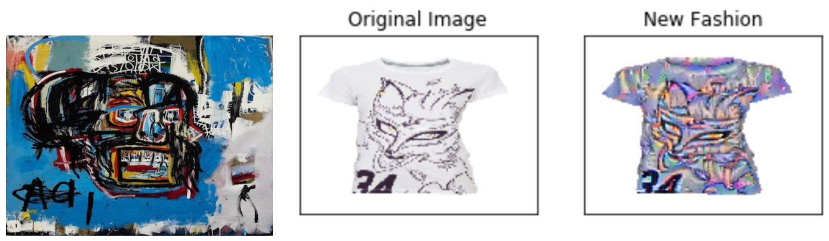


Figure 1: Left: Jean-Michel Basquiat Style, Right: Jean-Michel Basquiat Fashion Piece

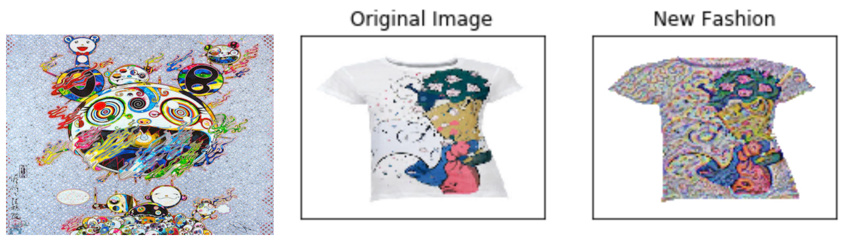


Figure 2: Left: Murakami Style, Right: Murakami Fashion Piece

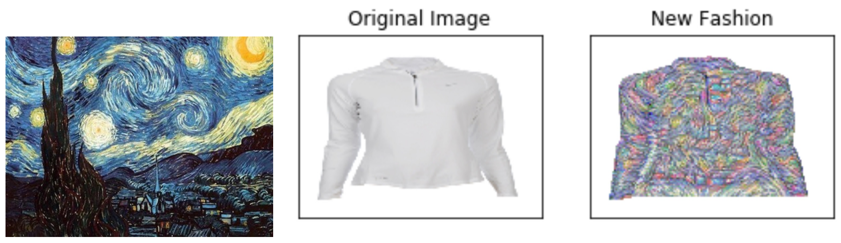


Figure 3: Left: Vincent van Gogh Style, Right: Vincent van Gogh Fashion Piece

5.1 Hyperparameter Discussion

We tested our algorithm with many styles and content images. In evaluating the generated images, we saw a clear distinction between epochs. As the number of generations grew, the actual fashion pieces saw drastic changes. However, we saw that after around 80 epochs, the actual content within the fashion item faded and was no longer the essence of the piece. Since our mission was to create a Neural Style Transfer algorithm that enriched fashion pieces, but not completely overwrite them, we chose to represent our results with the generated image after 60 epochs. Furthermore, we saw that a change in many of the hyperparameters challenged this very idea. As we changed either α, β, the learning rate for the Adam optimizer or even the number of epochs, the generated images either looked too similar to the content piece, or became unrecognizable. Upon evaluating different options, we chose the hyperparameters expressed above. Nevertheless, here are some results that attenuate these discoveries.

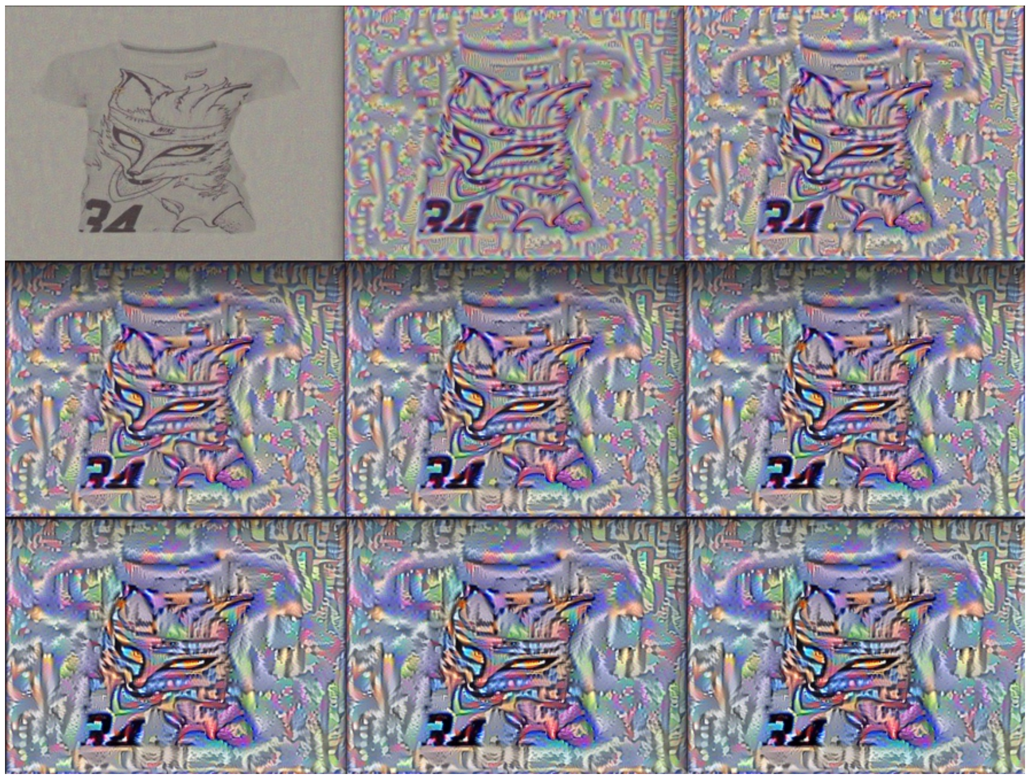


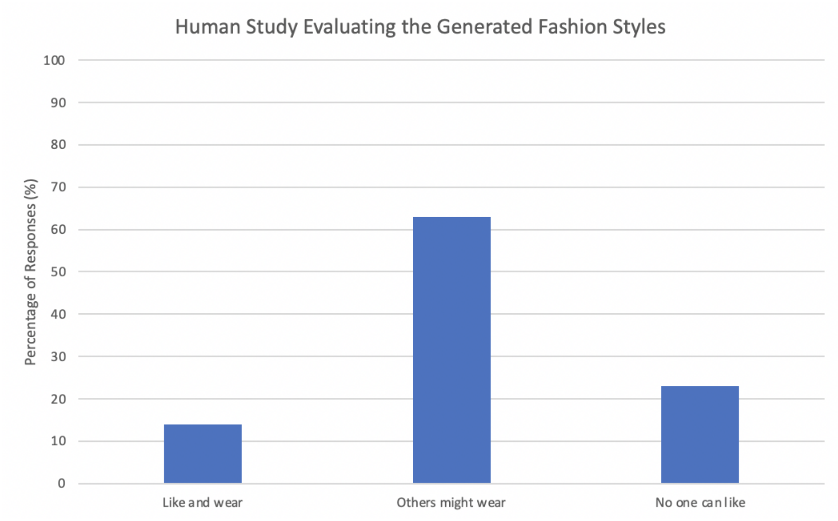
Figure 4: Epoch 0, 20, 40, 60, 80, 100, 120, 140, 160

5.2 Evaluation of Results with Human Study

As mentioned in our introduction, fashion is inherently a subjective process. Thus, we decided to conduct a human study that evaluated our newly-generated fashion styles. We shared a link to a survey via Facebook in which 10 content and style image pairs were randomly selected and the new-fashion generated images were displayed. For each new image, we asked the survey respondents whether they thought it was a viable fashion style. Inspired by common fashion-industry consumer surveys, we gave respondents three options:

1. I like it and I would wear it
2. I can see how others might wear it
3. No one can like it

After 2 weeks of leaving the survey open, we received 215 responses - meaning we got evaluations for 2150 image/style pairs. The bar graph below showcases our results:



Although only 14% of respondents indicated that they would wear the shirt/top, we were still encouraged by the results as our algorithm was randomly matching shirts with artworks. As a result, we see that 14% is the absolute baseline result - one that can be improved after carefully selecting which designs and artworks to match together.

6 Conclusion

We introduce a novel approach to fashion-generation using neural style transfer. We demonstrate that our architecture (that leverages the VGG19 model) is able to transfer the style of various popular artworks onto tops and shirts. From the human study we administered, we also learned that our baseline technique is able to generate fashion styles that a decent number of respondents deemed wearable. Thus, we find that our approach is promising at the least, and suggests that more work must be done into leveraging neural networks in creative industries.

References

[1] Anwar, A. (2020). *Difference between AlexNet, VGGNet, ResNet and Inception*: https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96

[2] Cioloboc, F. (2019, January 04). *Why use a pre-trained model rather than creating your own?* Retrieved November 17, 2020, from https://medium.com/udacity-pytorch-challengers/why-use-a-pre-trained-model-rather-than-creating-your-own-d0e3a17e202f

[3] Chu, L. (2019, March 22). *Convolutional Neural Network – Exploring the Effect of Hyperparameters and Structural Settings for Neural Style Transfer.* Retrieved November 16, 2020, from https://lanstonchu.wordpress.com/2018/09/03/convolutional-neural-network-exploring-the-effect-of-hyperparameters-and-structural-settings-for-neural-style-transfer/

[4] Dumoulin, V. (2016). *A Learned Representation For Artistic Style*: (https://arxiv.org/pdf/1610.07629.pdf)

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

[6] He, K. (2015). *Deep Residual Learning for Image Recognition*: (https://arxiv.org/pdf/1512.03385.pdf)

[7] Johnson, J., Alahi, A., Fei-Fei, Li (2016). *Perceptual Losses for Real-Time Style Transfer and Super-Resolution* (https://arxiv.org/pdf/1603.08155.pdf)

[8] Kingma, Diederik P., and Jimmy Ba. *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980 (2014).

[9] Log0, *TensorFlow Implementation of "A Neural Algorithm of Artistic Style"*: (http://www.chioka.in/tensorflow-implementation-neural-algorithm-of-artistic-style)

[10] MatConvNet: (http://www.vlfeat.org/matconvnet/pretrained/)

[11] Narayanan, H, (2016). *Convolutional neural networks for artistic style transfer*: (https://harishnarayanan.org/writing/artistic-style-transfer/)

[12] Simonyan, K., Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*: (https://arxiv.org/pdf/1409.1556.pdf)