

Exploration of machine-learning-based visual art creation

2020 Winter Research Report

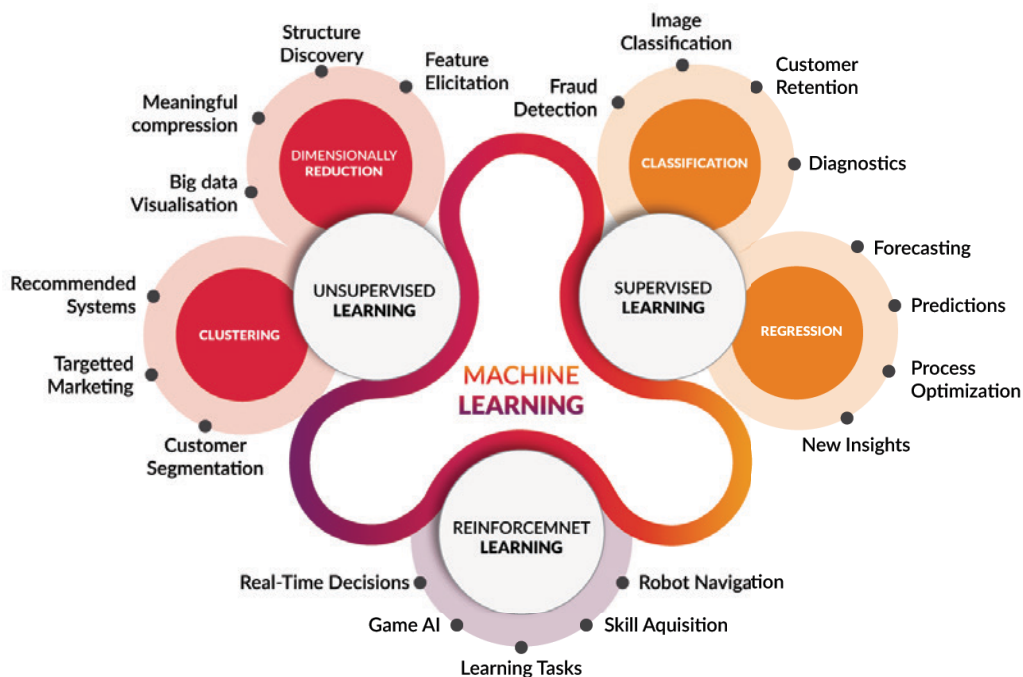
Weihaio Qiu

In this quarter, I have explored broadly machine learning by taking online classes, reviewing related papers, and performing hands-on experiments. In this report, I will list all my explorational experiences and the conclusions deriving from them.

Part1 Coursera Course

I believe progressing in machine-learning-based art creation requires deep understanding of machine learning, especially neural networks, so I took related courses on Coursera. I have completed three courses. The durations of the courses are estimated by the number of videos and homework it contains, though it usually take much shorter time for me to complete them.

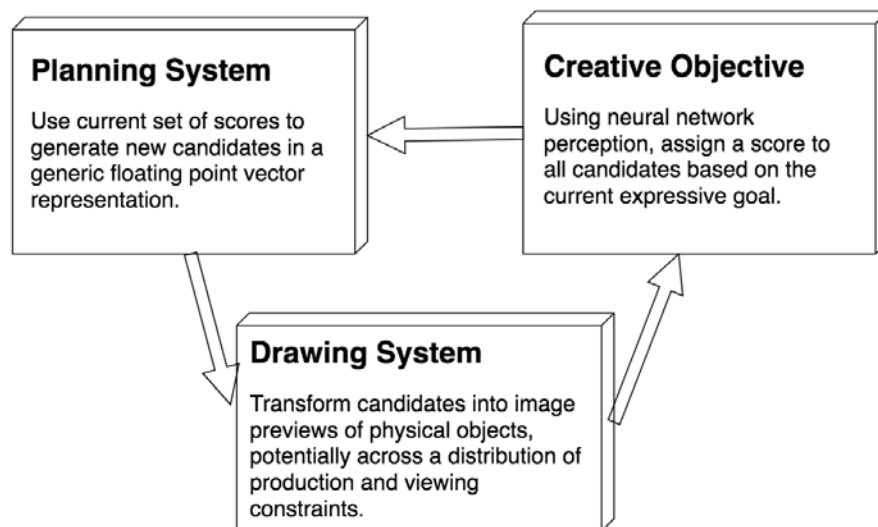
- **11-week Machine Learning course**
 - *overview of the traditional machine learning*
 - *Supervised learning: linear regression, logistic regression (i.e. classification)*
 - *Unsupervised learning algorithms: anomaly detection, dimension reduction, clustering*
 - *Gradient-descent*
- **4-week Neural Network and Deep Learning course**
 - *Forward propagation & backward propagation*
 - *Loss function*
 - *Activation function*
 - *Gradient Calculation*
- **4-week Convolutional Neural Network course**
 - *3D convolution, padding, strides*
 - *pooling*
 - *Different CNNs, VGG, AlexNet, ResNet, Inception Network*
 - *Image Classification*
 - *Neural Style Transfer*
 - *Feature Visualization*



Part2 Literature Review

1. **Perception Engines by Tom White**
2. Synthetic Abstractions by Tom White
3. SVG VAE: Generating Scalable Vector Graphics Typography by Raphael Gontijo Lopes et.al.
4. SvgAI – Training Methods Analysis of Artificial Intelligent Agent to use SVG Editor by Anh H. Dang et.al.
5. Variational Autoencoder
 - <https://github.com/L1aoXingyu/pytorch-beginner/tree/master/08-AutoEncoder>
6. **An Introduction to Image Synthesis with Generative Adversarial Nets** by He Huang et.al.
 - Overview of different kinds of GANs, worth reading.
7. **Differentiable Image Parameterizations**
 - Formal description of “Parameter-optimization”
8. **CPPN Compositional pattern producing networks: A novel abstraction of development** by K.O. Stanley.
 - A differentiable neural network that can produce interesting images. This kind of system can be used for “Parameter-optimization” pipeline
 - <http://blog.otoro.net/2016/03/25/generating-abstract-patterns-with-tensorflow/>
 - <http://blog.otoro.net/2016/04/06/the-frog-of-cifar-10/>
 - <http://blog.otoro.net/2016/06/02/generating-large-images-from-latent-vectors-part-two/>
9. **Fourier-CPPNs for Image Synthesis** by Matthew Tesfaldet
 - Improved version of the original CPPN in terms of image details
 - demo: <http://www.aiartonline.com/highlights/mattie-tesfaldet/>
10. <http://www.aiartonline.com>
 - a good archive of neural art projects in NIPS.

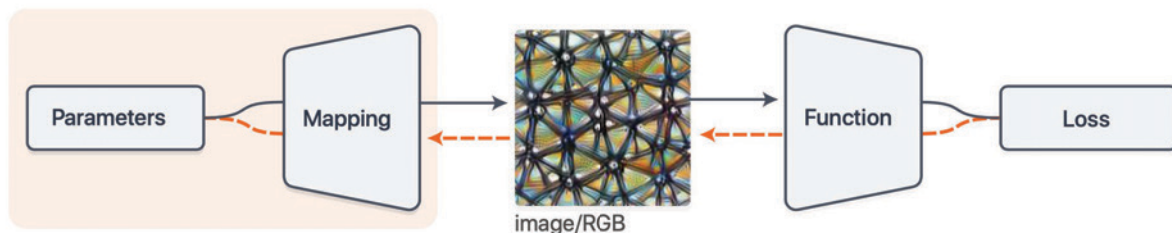
Part3 Architecture design



Source: *Perception Engines* by Tom White

Inspired by how Tom White differs from other neural-art artists, I would like to improve the current mainstream neural-art approach by adding an image parameterization component to it. To be clarified, image parameterization is the process of generating a RGB image from a set of parameters. It includes all kinds of image-producing algorithm in generative art, which uses parameters to control the final look of the image. The artists such as Mario Klingemann, Sofia Crespo did not use image parameterization in their past creation. Instead, they all used “dreaming” methods, which is to create a new image by updating the pixel values in an initial image until it generates certain activations in a pre-trained CNN. The desired activation is set by the artist, and by selecting proper activation, the artist can finish different tasks in this pixel-optimization process, and create different images. On the other hand, Tom White’s method start with an image parameterization drawing system that can generate the images from the parameters. By randomly changing the parameters, the system can find the most suitable parameters which can generate an image activates the pre-trained CNN most correctly. Tom White’s extra image-producing system enables the customization of final image looks.

However, there are some reasons that why this type of neural-art architecture was not widely accepted by other artists. One of them is the inability to back-propagate the loss to the parameters. In other word, the parameters are not adjusted based on the difference between the current CNN activations and the target activation. They are changed randomly for certain iterations, and the best set of parameters among all variations is chosen as the final. On the one hand, the parameter-optimization process is not efficient. It takes long time to converge to the optimal. **On the other hand, it diminish the credibility of the artists’ argument about the result image, which are claimed to be the one that activates the CNN closest to the way he desired, for example, being recognized as an electric fan. This is not true because the final parameters are the local optimal rather than the global optimal. There might be another image that can perform better than the generated image, and recognized by the CNN as an electric fan with higher score.**

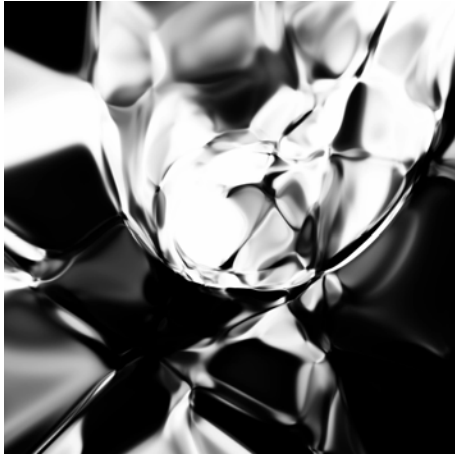


source: *Differentiable Image Parameterizations*

To overcome these weakness, a differentiable image parameterization system should be used to generate RGB images from a set of parameters. In this way, the loss from CNN can be back-propagated to the parameters and used for to calculate the gradients, which will facilitate parameters adjustment. Most traditional generative art algorithms are not differentiable, because the parameters have gone through many non-differentiable computations before they generate the final image. However, keeping all computation differentiable greatly limits the expressiveness of the algorithm. It is challenging to build a differentiable image parameterization system that can produce image with complexity that comparable to traditional generative art algorithm.

As far as I am concerned, this is a research area that has not been studied very well. The most recent impactful paper is Compositional pattern-producing networks (CPPN) by Kenneth O. Stanley 2007. In general, this is a neural-net-based image parameterization system that can

produces images with relatively high complexity and varieties. David Ha, a researcher in Google Brain, has explored its usage in several generative art projects, *Generating Large Image from Latent Vectors*, *Generating Abstract Pattern with TensorFlow*. A obvious weakness of the original CPPN is the lack of fine details, which is then slightly improved by F-CPPN, came up by Matthew Tesfaldet in 2019. However, I think there are still a lot more to be researched in the neural-nets-based image parameterization, especially for the purpose of artistic creation.



Overall, I think I am going to extend the traditional “dreaming” architecture by adding the differentiable image parameterization component to improve the visual variety of produced images. My main contribution will be building a differentiable image parameterization system that outperforms the current CPPN or L-CPPN when used in art creation. For now, it is still very general, but hopefully I will come up with some metrics to measure the performance soon.

Part4 Little Experiment with Neural Style Transfer

I generated a video from my experiments with neural style transfer. In neural style transfer, three images are used to generate a new image, a content image, a style image, and an input image, which initially is just a replica of the content image but will be updated in optimization process. The video presented the process how the neural network transfer the input image into a new image with the “style” of the style image and the “content” of the content image.

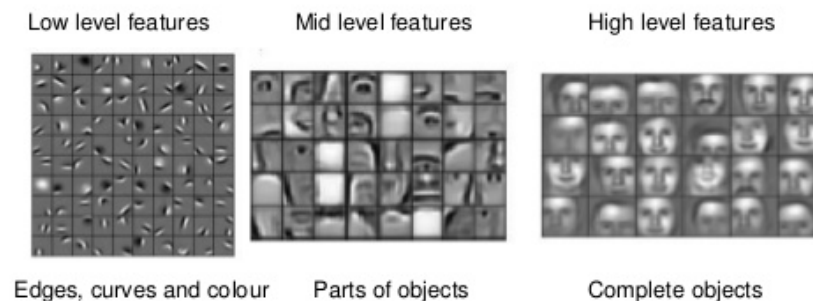
Obviously, it is not straightforward to describe the “style” and the “content” of an image to computers, not to mention programming the computers to draw an image that combines one image’s style and the other image’s content. Think about why we can identify a typical painting of Van Gogh instantly by its “style”: we can see the curly strokes which blend in more than one



color are used to depict a pure color background, such as night sky and a portrait backdrop. Similar to face detection, image features must be extracted so that the machine can recognize Van Gogh's painting by detect if certain features are present in certain way. People have come up with good features to represent face, so that it is not impossible to find features representing Van Gogh's paintings' characteristics. The real challenge is that describing the paintings of different artists requires very different features, which takes great effort.

The great advantage of neural networks, especially successful CNNs, is their potential to extract various complex features given enough training samples. With the help of feature visualization, we can observe what features are extracted and detected by each neuron in every layer of the CNN. A neuron will be activated the most if certain pattern similar to its feature visualization is found in the image. We can observe that the complexity of extracted feature increases as the depth of layer increases. Simple features such as lines and circles are detected in the shallow layers. More complicated features such as eyes and wheels are detected in the layers of medium depth. The most complicated features such faces, cars are detected in the very deep layers. Therefore, a deep CNN is able to extract very complicated features.

Stages of feature extraction by CNN



We can expect that Van Gogh's paintings consistently activate a group of neurons the most, because the feature extracted by those neurons represents jointly Van Gogh's paintings' characteristics. In practice, these features are high level complicated feature, representing by neurons in the deep layers.

On the other hand, the "content" of an image, which are the objects depicted in the image, is usually represented by the low level features. The viewers usually recognize an object by the shape of its contours, which consists of lines and curves so that can be detected by the neurons in shallow layers.

An image that combines the "content" of one image and the "style" of another image should activate the same neurons as the content image does in the shallow layers and the style image does in the deep layers.

In the deep layers, the style-transferred image should activate the same neurons as the style image do, so that it has similar high level features to the style image. However, in shallow layers, the style-transferred image should activate the same neurons as the content image do,

so that they share the same low-level features. In this way, the shape contours are preserved, and the objects in the content image are still recognizable in the style-transferred image.

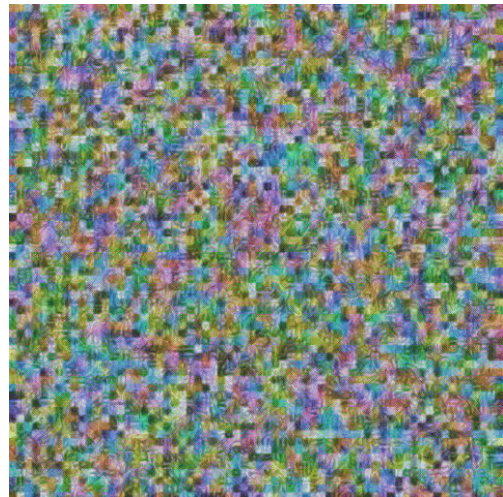
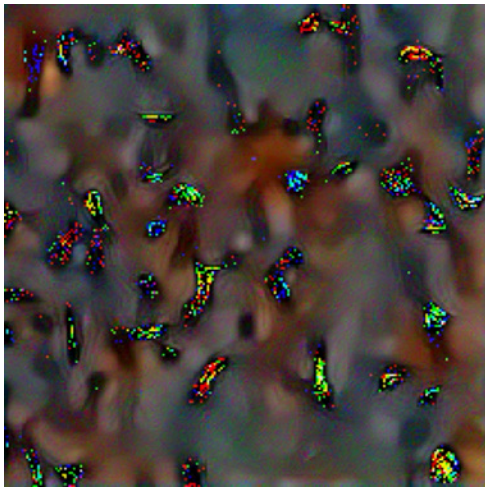
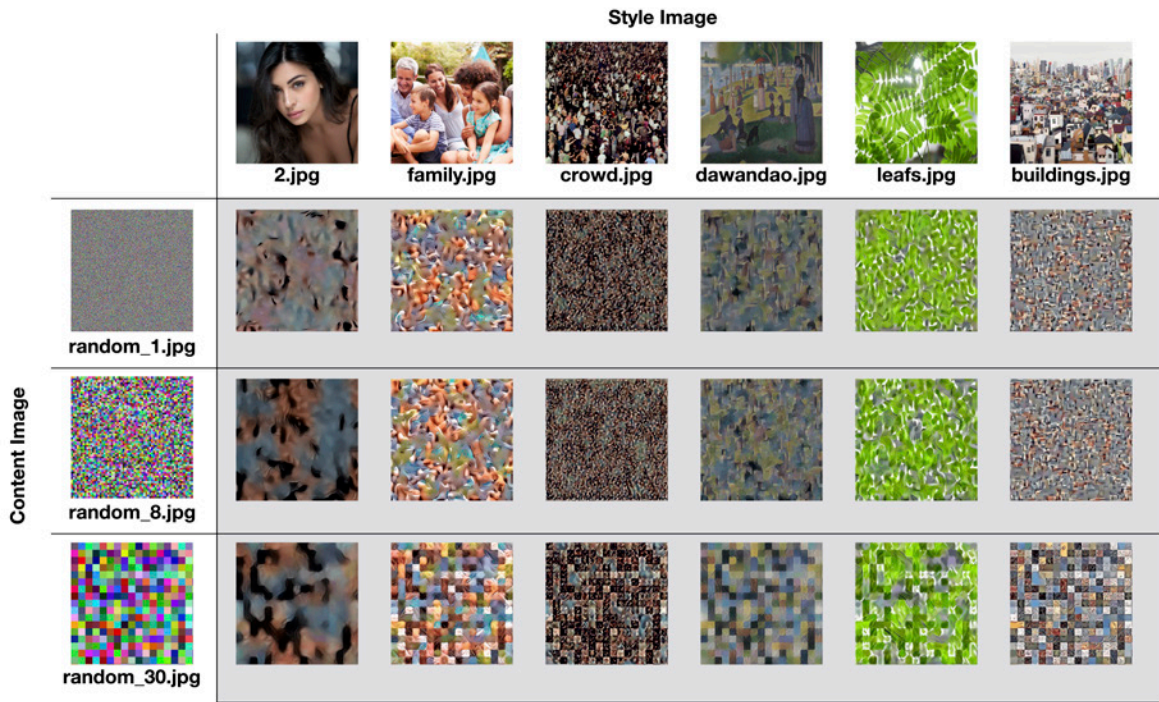
For each style transfer task, the user can decide which layers of neurons are used to describe the style and which layers of neurons are used to describe the content. The activation error of neurons in the “style layers” are summed as the style loss. The activation error of neurons in the “content layers” are summed as the content loss. This is followed by the calculation of total loss, which is the weighted sum of the style loss and the content loss. User can choose two weights for the two loss respectively, depending on whether the user intends to preserve more contents from the content image or obtain more style from the style image. Increasing the weight of the content loss will keep more contour shapes from the content image, so that the objects in the content image are still recognizable in the result image, but the style of the result image might be far from the style image. Whereas increasing the weight of style loss results in the increasing similarity in the visual characteristics to the style image, but may cause the objects in the content image not recognizable.





The default neural style transfer uses pretrained VGG-19, which contains 5 convolutional layers of neurons, and uses the layer 4 as the “content layer” and all 5 layers as the “style layers”. The weights for style loss and content loss is set to 1000000 and 1 respectively. Selecting a real photograph as the content image and a modern art painting as the style image, this configuration by default enables transferring the content image into a new image with the style from the painting.

In my experiment, I attempted to understand how machine would extract the style of a real photograph, which is not evident from humans’ perspective. For us, the painting has styles because it depicts objects subjectively so that it looks different than the object actually appears to us. The way the different painters depict are unique. Whereas all photographs are objective so it is difficult to describe the style of certain photographers in aspects of how differently the object looks in the photograph from the real world. To force the machine to summarize the style of a photograph, I used the photograph as the style image, and a random noise image as the content image. I also select the layer 3,4,5 as the “style layers”, and the weights of 0 and 1000000 respectively the content loss and style loss. The neural networks then updates the random noise image gradually over iterations of optimizations, so that the updated image’s activation matches the photograph’s activation in the “style layers”. Finally I created a time-elapse video by assembling the images after different numbers of iterations to show the process when the machine finds a photograph’s style.

I tried with the combinations of the noise images of three different grain size and several photographs. The results are listed below. Please refer to high-resolution pdf file, so that you can zoom in to see the details in the images. Two sample time-elapse videos are also included at the end. The animations of all other results can be found in the zip file.

Overall, I think the neural network changes the image in an unexpected way, by changing the kind of noise in the content images, and playing with different photographs, various abstract images can be generated.



- ▼ buildings
 -  buildings.jpg
 -  result_buildings.jpg_random_8.jpg.gif
 -  result_buildings.jpg_random_30.jpg.gif
 -  result_buildings.jpg_random.jpg.gif
- ▶ crowd
- ▶ dawandao
- ▶ family
- ▶ leaves
- ▶ portrait
- ▶ random_1.jpg
- ▶ random_8.jpg
- ▶ random_30.jpg
- ▶ summary
- ▶ summary
- ▶ summary