Deep Learning: Graphics
Final Project - Sketch to Painting Synthesis
Written Report

Dallas Scott (ds4015)

Abstract

The aim of this report is to investigate artwork synthesis and colorization given a rough outline or sketch of the desired artwork components and colors. The model we are going to look at is a generative adversarial network which was trained on a series of existing artworks using contour outlines and color palettes extracted from the works, which is then used for inference on an unseen dataset of sketches to stylize and colorize them given a custom color palette selected by the user.

Introduction

Much of the work being done in image synthesis using machine learning tends to revolve around stylizing photographs into a particular art style or generating novel works via a text prompt. These approaches have

their uses and are certainly interesting, but there is less emphasis on visualizing what a given sketch might look like as a completed work. This usage is of particular interest to training artists who may not be at a level in painting ability to be able to turn their sketches into convincing colorized media but who might wish to have some sort of visualization for inspiration.

The aim of this project is to take a given outline or rough ink/pencil sketch of certain objects or figures, a set of individual colors making up a color scheme, and generate a novel synthesis of a painting which uses the outlines in the sketch and the provided color palette as the basis for its painting. The result is a fully fleshed out painting that builds upon what an artist has already provided in rough outline form and a desired set of colors for that drawing.

Previous Work

The concept of an adversarial model for image synthesis appears to be a relatively recent phenomenon. The process is described in a 2014 paper by Goodfellow et al. (https://www.researchgate.net/publication/263012109_Generative_Adversarial_Networks) wherein a framework is conceived as a two-player adversarial game where the goal of one

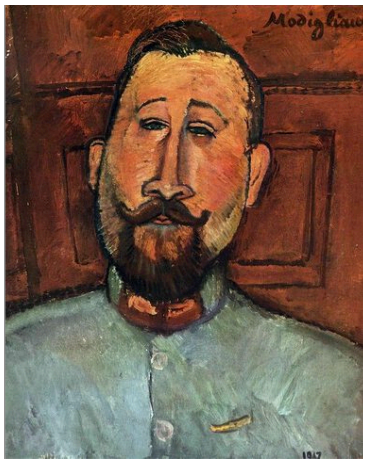
player is to trick the other. The players in this case are a Generator and Discriminator (G and D, respectively) defined by multilayer perceptrons (MLP). The Generator in this case is set up to synthesize images in an attempt to trick the Discriminator into thinking they are real. The Discriminator, by contrast, has as its aim the accurate prediction of whether an image was generated by G or is an original. Using back propagation, the GAN model learns to accurately synthesize images.

A 2017 paper by Isola et al. (https://arxiv.org/abs/1611.07004) describes the Pix2Pix image-to-image problem and how adversarial networks can be used for, among other tasks, colorizing images and reconstructing objects from contours.

Overview

Preprocessing

The first step in this process involves obtaining a dataset of individual artworks. With these, the overall structure of the objects in the artwork are extracted in the form of edge contours using OpenCV's Canny.



Original Artwork

Contour Detection

Contours generated with Canny with Gaussian Blur - geometrical/abstract synthesis

The initial contours were selected so as not to be exact copies of the original painting contours but to have something of a geometrical quality, which would make for abstract stylizations during the image synthesis process. The model was then retrained using contours which more accurately follow the contours of the original shapes and objects in the artwork and which have much more detail in some cases.

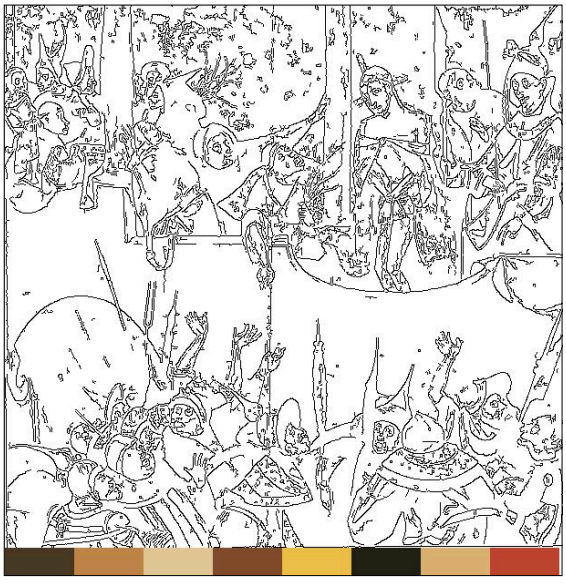
Contours generated with Canny with low/high thresholds - faithful synthesis

These contours are used as input to the GAN to serve as guidelines for object placement and accurate composition.

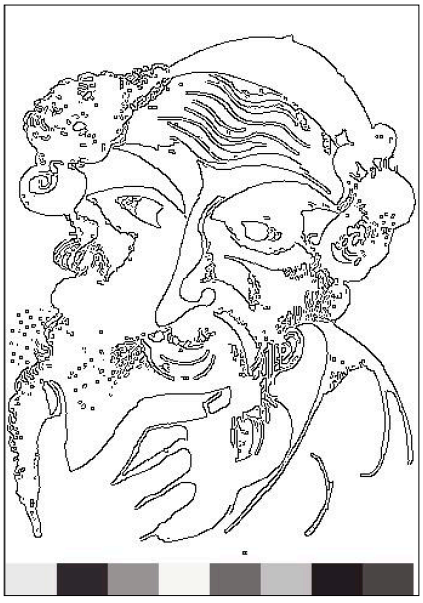
The second step involves extracting the color palette of the original artwork. This is done using KMeans clustering with k = number of colors in the palette. Here, we used k=8 to extract a band of the 8 most predominant colors in the artwork and then superimposed this palette as a bar at the bottom of each contour extraction (after resizing to make empty space at the bottom for placement of the palette) that is the full width of the contour image.

Sketch + Palette

Sketch + Palette



Sketch + Palette



Sketch + Palette

This combined sketch + palette dataset is what will be used to train the GAN, in conjunction with the original artworks the contours and colors were drawn from as targets.

Technical Description

The model used is a generative adversarial network which makes use of a UNet and PatchGAN discriminator to train on the combined sketch/palette images as inputs and original artworks as targets. The dataset is split into 80% training, 20% validation (both sketch/palette and corresponding original artworks) and the training sets (s/p + a) are loaded into the GAN model.

Each sketch/palette image is cropped to separate the contour image on top from the palette bar on the bottom and the palette separated into its component colors, giving three separate components: sketch, palette and original artwork.

The UNet consists of an encoder and decoder with a series of Conv2d + batch normalization + ReLU layers in the encoder to shrink the image down to size and Conv2d + batch normalization + ReLu and a final tanh layer to re-expand to the original dimensions.

The UNet makes use of skip connections to enable close adherence to the outlines in the sketch when generating the new artwork.

The PatchGAN discriminator uses a series of convolutional layers, batch normalization and LeakyReLU activation, looking at individual patches of an artwork and making a determination as to whether they are real or fake.

In the actual training loop, the generator makes an image, the discriminator makes an assessment as to its authenticity and the losses for each are calculated. The model is trained with a batch size of 4 images for 100 epochs, using Adam for optimization and Torch's BCEWithLogitsLoss for the loss function. Bilinear interpolation is applied to the palette to rectify vertical banding in the validation images.

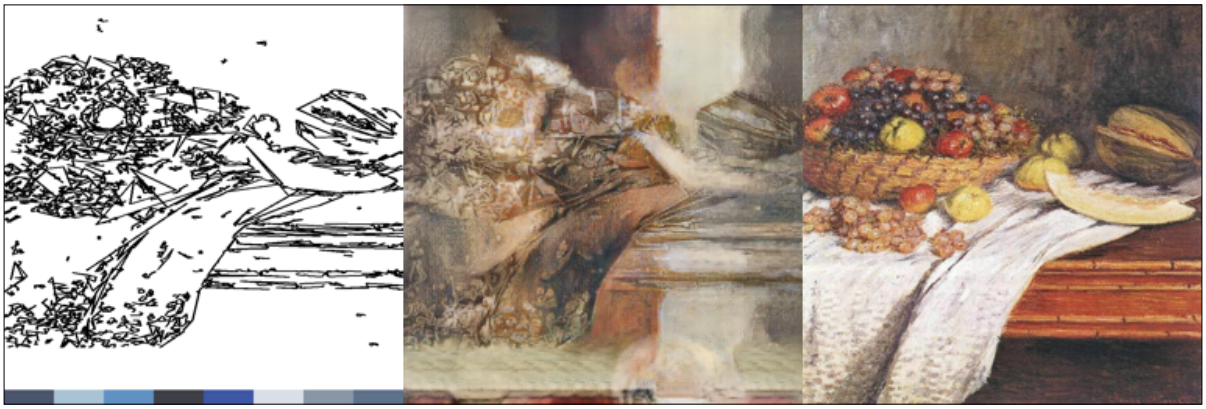
For testing, we load the validation set (sketch + palette combination), convert to tensors, pass through the trained model to generate a new colorized artwork and save the output as 3 files: the newly synthesized artwork, the original artwork from the original dataset the contour and palette were taken from, and the contour image itself (the latter two for comparison purposes only to see how accurate the synthesis is).

Results and Analysis

During training, the discrimination and generation losses remain fairly well balanced, indicating adequate learning without overfitting.

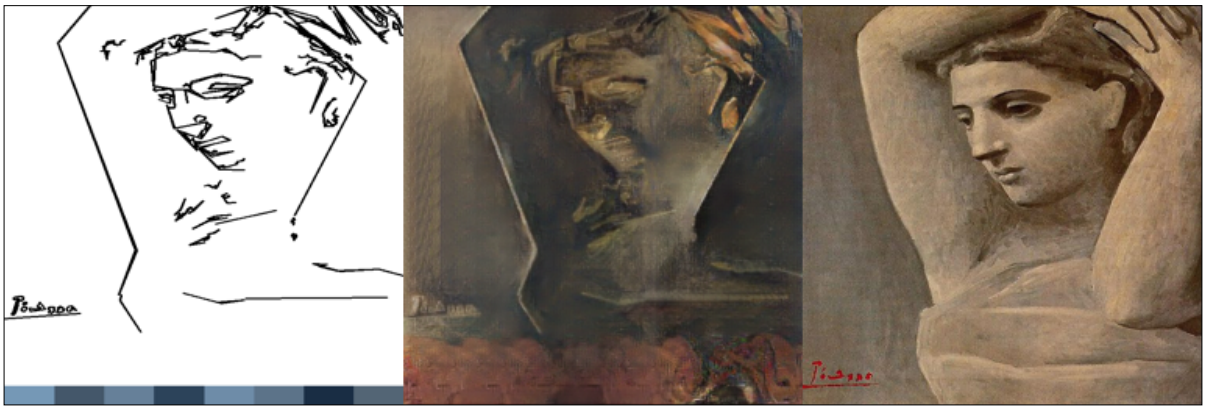
Epoch 0: D=0.5239, G=7.0728
Epoch 1: D=0.1478, G=8.4078
Epoch 2: D=0.1191, G=7.3256
Epoch 3: D=0.3254, G=7.3745
Epoch 4: D=1.2673, G=6.1012
Epoch 5: D=0.5562, G=6.9392
Epoch 6: D=0.9015, G=8.4213
Epoch 7: D=0.3393, G=7.2014
Epoch 8: D=0.5730, G=7.7263
Epoch 9: D=0.5420, G=9.2864
Epoch 10: D=0.3471, G=10.0474
Epoch 11: D=0.4883, G=8.3173
Epoch 12: D=0.6978, G=4.0860
Epoch 13: D=0.5689, G=7.4231
Epoch 14: D=0.3548, G=8.5401
Epoch 15: D=0.3396, G=8.6927
Epoch 16: D=0.4600, G=8.7576
Epoch 17: D=0.5930, G=6.1039
Epoch 18: D=0.5391, G=9.1202
Epoch 19: D=0.6155, G=7.6032
Epoch 20: D=0.5824, G=9.8805
Epoch 21: D=0.4980, G=7.6892
Epoch 22: D=0.3530, G=7.4523
Epoch 23: D=0.5013, G=7.1270
Epoch 24: D=0.2527, G=8.6542
Epoch 25: D=0.4054, G=9.2544
Epoch 26: D=0.3531, G=9.3148
Epoch 27: D=0.7091, G=5.9413
Epoch 28: D=0.1823, G=9.2922
Epoch 29: D=0.5549, G=6.6125
Epoch 30: D=0.8576, G=7.2841
Epoch 31: D=0.5762, G=7.0695
Epoch 32: D=0.5702, G=7.7625
Epoch 33: D=0.5982, G=6.0779
Epoch 34: D=0.5337, G=6.9687
Epoch 35: D=0.5987, G=6.1778
Epoch 36: D=0.5005, G=8.8672
Epoch 37: D=0.7025, G=7.4563
Epoch 38: D=0.7054, G=6.6494
Epoch 39: D=0.8542, G=2.5761
Epoch 40: D=0.7354, G=4.6232
Epoch 41: D=0.4811, G=7.9002
Epoch 42: D=0.3667, G=9.3928
Epoch 43: D=0.6427, G=8.6703
Epoch 44: D=0.6873, G=7.1524
Epoch 45: D=0.5781, G=6.6971
Epoch 46: D=0.6102, G=7.0247
Epoch 47: D=0.6418, G=9.1419
Epoch 48: D=0.5711, G=8.7257
Epoch 49: D=0.3997, G=8.4191
Epoch 50: D=0.5175, G=4.9038
Epoch 51: D=0.5034, G=6.7544
Epoch 52: D=0.3937, G=8.6446
Epoch 53: D=0.5120, G=7.9729
Epoch 54: D=0.7475, G=7.7505
Epoch 55: D=0.7159, G=5.0453
Epoch 56: D=0.5657, G=7.0304
Epoch 57: D=0.4852, G=9.2691
Epoch 58: D=0.4705, G=9.0914
Epoch 59: D=0.7517, G=6.4025
Epoch 60: D=0.5025, G=8.9713
Epoch 61: D=0.5042, G=7.1036
Epoch 62: D=0.9385, G=5.3872
Epoch 63: D=0.5033, G=8.7850
Epoch 64: D=0.7942, G=8.0307
Epoch 65: D=0.6657, G=5.7391
Epoch 66: D=0.5856, G=8.5898
Epoch 67: D=0.5329, G=6.0059
Epoch 68: D=0.4882, G=8.9956
Epoch 69: D=0.6808, G=8.3405
Epoch 70: D=0.3229, G=9.3962
Epoch 71: D=0.4654, G=5.5276
Epoch 72: D=0.3914, G=6.4714
Epoch 73: D=0.7155, G=7.6218
Epoch 74: D=0.4516, G=7.7808
Epoch 75: D=0.7825, G=7.4168
Epoch 76: D=0.6093, G=7.5654
Epoch 77: D=0.7003, G=6.9351
Epoch 78: D=0.5270, G=7.0706
Epoch 79: D=0.3866, G=9.4138
Epoch 80: D=0.2841, G=8.8640
Epoch 81: D=0.7985, G=9.0719
Epoch 82: D=0.4016, G=6.3167
Epoch 83: D=0.3549, G=9.6840
Epoch 84: D=0.2405, G=8.4275
Epoch 85: D=0.5925, G=6.9237
Epoch 86: D=0.7178, G=7.0613
Epoch 87: D=0.6556, G=7.5295
Epoch 88: D=0.4686, G=6.8870
Epoch 89: D=0.5776, G=7.8889
Epoch 90: D=0.5289, G=8.6961
Epoch 91: D=0.4241, G=8.2721
Epoch 92: D=0.7702, G=4.6374
Epoch 93: D=0.6016, G=8.5261
Epoch 94: D=0.7905, G=5.6411
Epoch 95: D=0.4636, G=9.3646
Epoch 96: D=0.5601, G=7.3535
Epoch 97: D=0.5307, G=4.4174
Epoch 98: D=0.3456, G=7.3075

Contour+Palette / Synthesized / Original

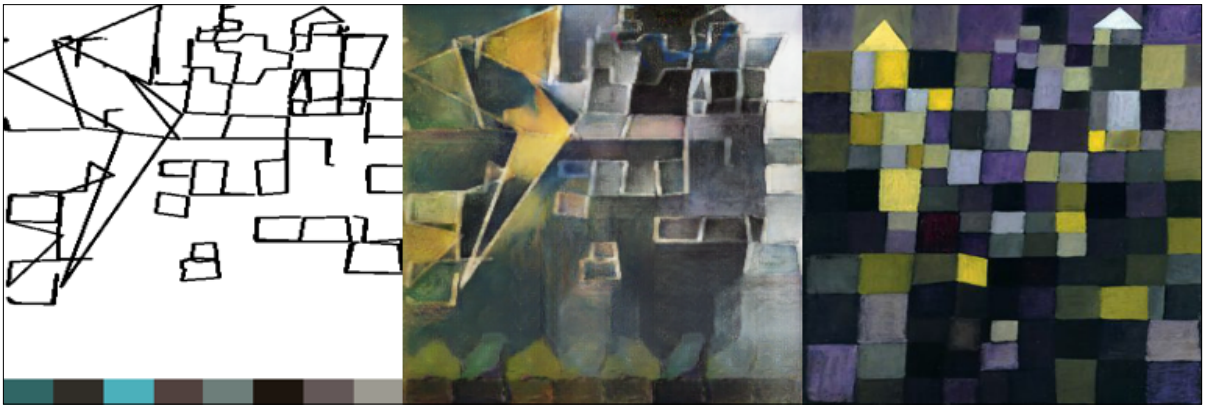


Contour+Palette / Synthesized / Original

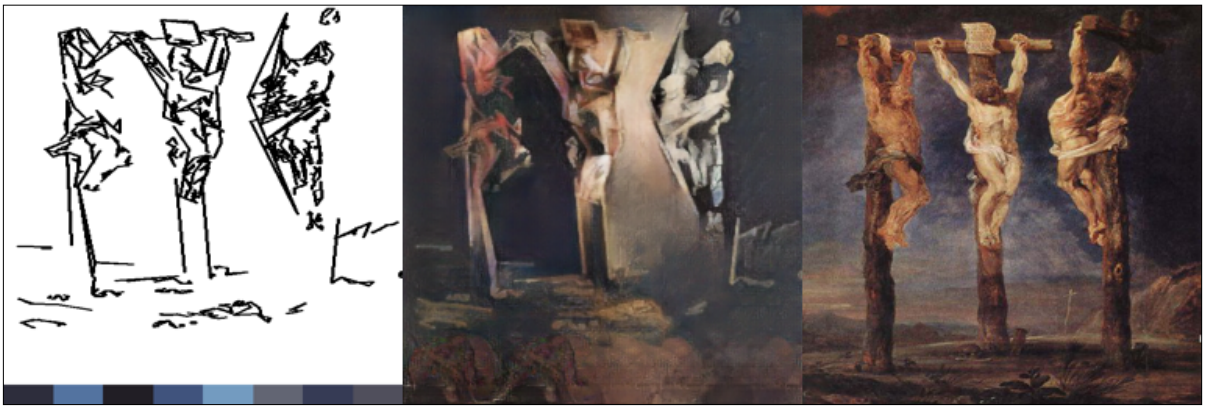
Contour+Palette / Synthesized / Original



Contour+Palette / Synthesized / Original



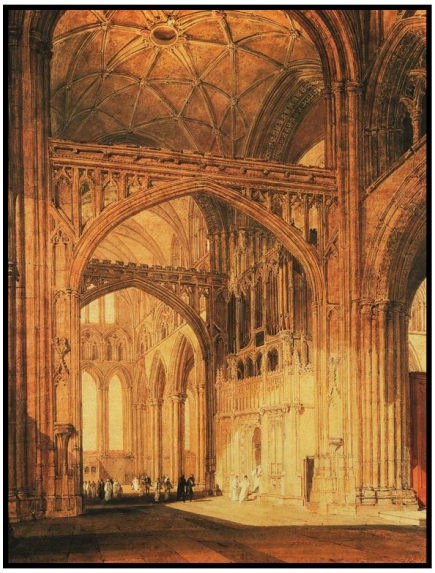
Contour+Palette / Synthesized / Original



Contour+Palette / Synthesized / Original

Contour+Palette / Synthesized / Original

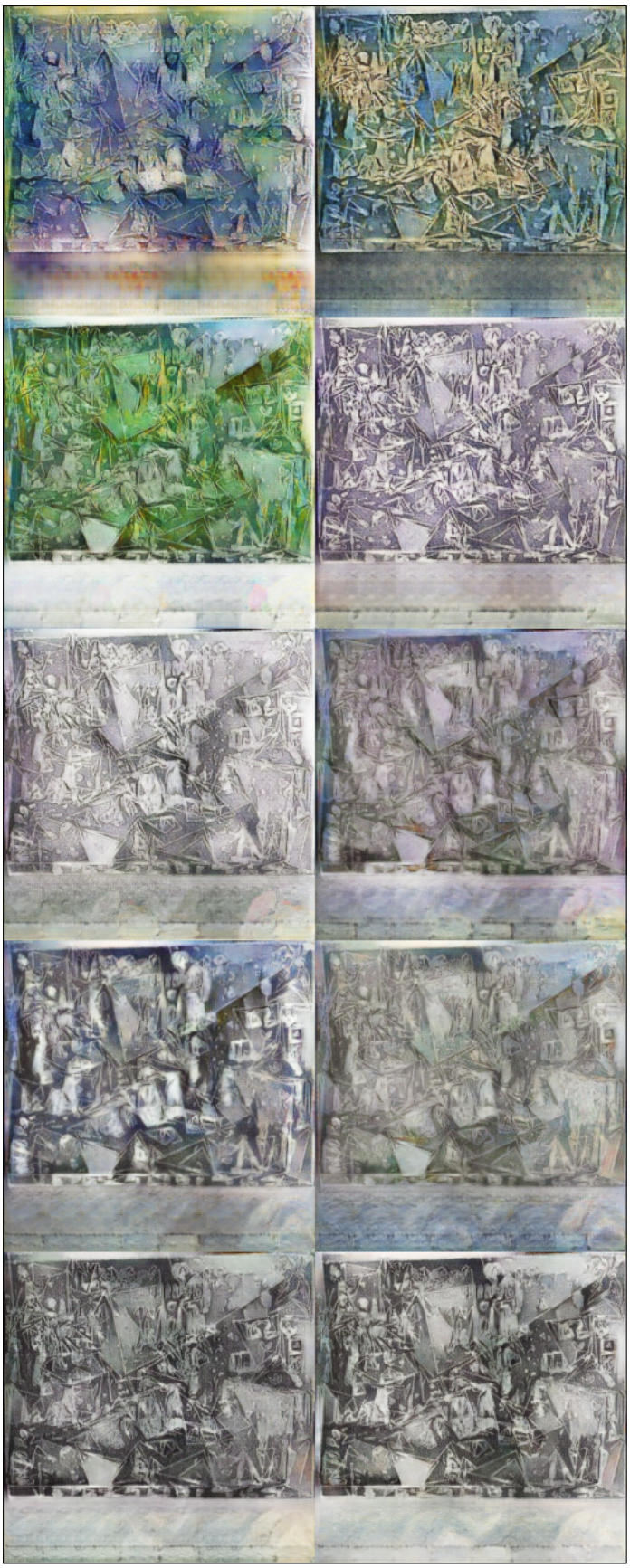
Contour+Palette / Synthesized / Original



Original

Contour

Synthesized: Epochs 5-99



Weighting coefficients used:

L1: Pixel value difference between true/synthesized work
Edge weight: Compares Sobel edges with original contours
Total variation (TV): Sum of differences between neighbor pixels

The low L1 weight of 1.0 accounts for the abstract quality of synthesized images. An increase in this would have likely produced more faithful reproductions of the artworks at the expense of stylization.

An edge weight of 5.0 made for fairly good anchoring to the existing contours in the synthesized images.

Total variation was used to smooth out the synthesis and remove dot-like artifacts.

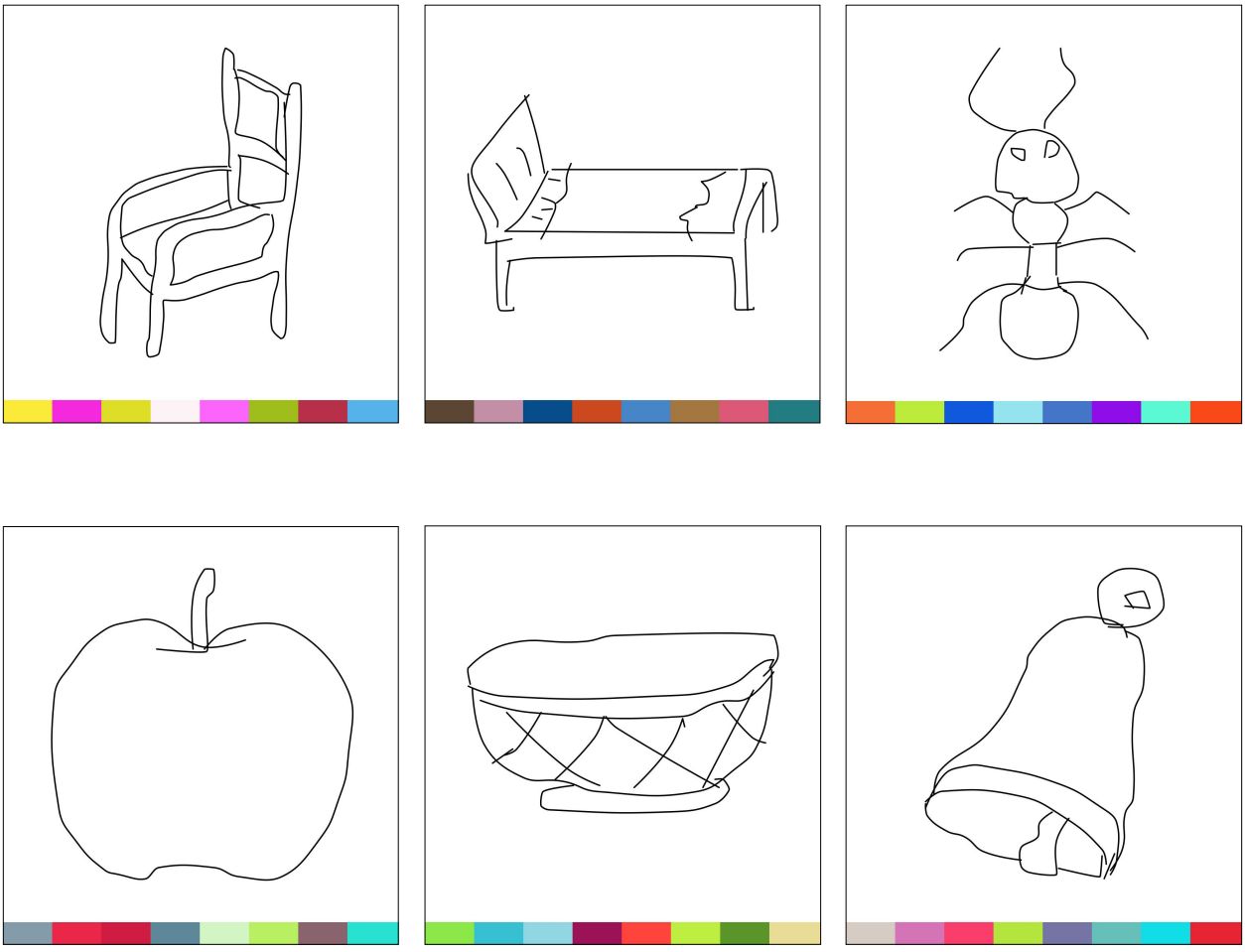
The resulting images in most cases provide nice stylizations of the originals, though there remain in many some tiling and blob artifacts. Perhaps increasing the L1 weight during training or training for more epochs would have mitigated some of this.

Generalization

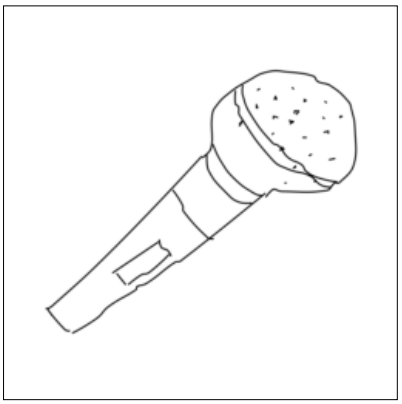
To test how the trained GAN performs on a set of non-classical artworks, I performed inference on a set of 20,000 unrelated simple sketches sourced from the following:

https://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/

To give some idea of these sketches, here are examples along with randomized color palettes of 8 colors:



The colorized results show each component of the sketch given a unique color from the provided palette. Banding artifacts remain owing to the simplicity of the sketches compared to the contours of the trained dataset, but the results generalize fairly decently.



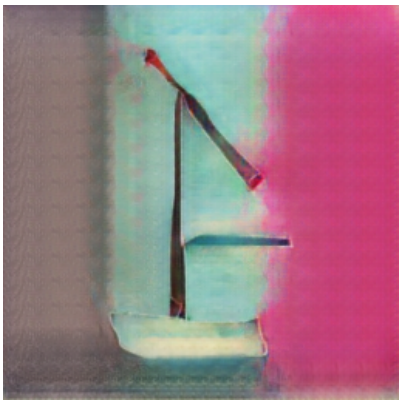
Original

Synthesized

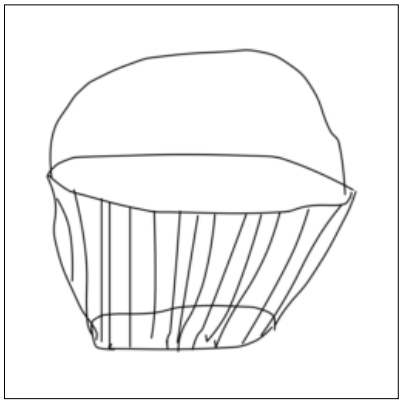
Original

Synthesized

Original

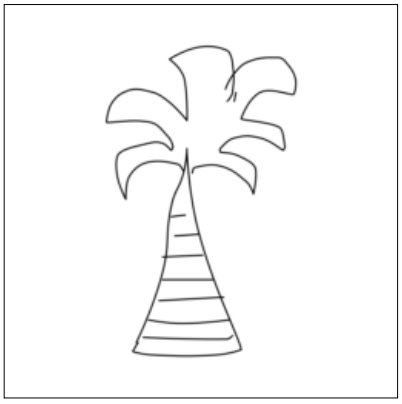


Synthesized



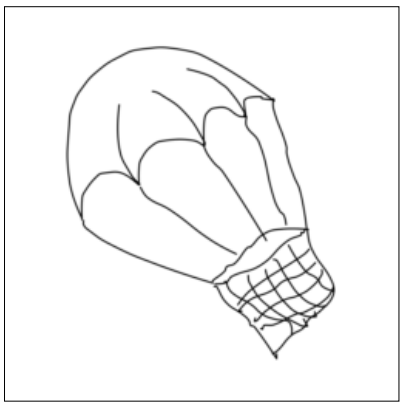
Original

Synthesized



Original

Synthesized



Original

Synthesized

Applications

As discussed at the outset, the primarily application of this model I view as being a visualization aid for aspiring art students who might wish to make the transition from sketch work to painting but aren't familiar with color theory or how the various pieces might go together. By synthesizing a painted image from a given sketch in any given color palette, the artist can quickly and easily see what his/her works might look like in a different medium.

Other applications might be graphic design for magazines or in the the advertising industry, where a rough sketch can be mocked up into a colorful visualization either for use on its own or as an intermediary step toward something more tangible.

Technical artists (e.g., architects) might also benefit from this technique in turning their designs into colorful and vibrant visual models or as a tool for altering design concepts on the basis of various color schemes, which the model enables testing with little effort.

Initial Project Work

Prior to the current project's inception, the goal was to identify objects within artworks and synthesize new artworks with those objects in a rearranged composition. The initial idea was to perform object detection and segmentation on the original artworks, generating labels for objects within the works, extracting a cropped mask of those artworks and then using the crops as input to a GAN to generate novel pieces.

The COCO val2017 common objects dataset was used for object detection. All of these images were passed through the fast-neural-style transformation network to create 4 different art stylizations of the COCO photographs.

Fns stylization of COCO val2017

Fns stylization of COCO val2017

After training with both the original

COCO dataset combined with the stylized versions of the COCO images, inference was performed on the artwork dataset, though the results were subpar. Given the limited number of COCO classes and object labels of many modern objects (cars, appliances, etc.), there were not enough object detections in the artwork dataset.

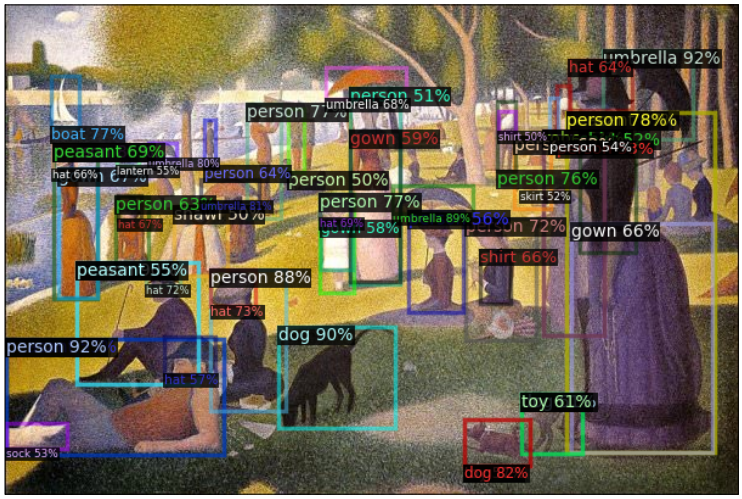
At this point, the decision was made to use a pretrained Detectron2 model for object recognition. This model fared better at detecting objects in the artwork dataset, though the number of objects being detected was still fairly small and there were many repeated categories (e.g., person, hat, horse, dog).

In order to have a more robust object detection to be able to pass into a GAN, a pretrained Detic model (built on Detectron2) was used, which has the potential of identifying some 21,000 classes of objects. The key advantage to the Detic model is the ability to use a custom vocabulary to be able to find and detect objects suitable to the dataset the inference is being performed upon.

Now the task was to come up with a custom dictionary of object words that would most likely be found in classical artworks. Initially this was built up by hand observing a series of artworks just to test the model. The classes in the custom vocabulary would accurately be predicted in those artworks they were observed in, but with over 5000 training

artworks, building a full list of terms was not feasible. Thus, I pulled from WordNet a list of 1500 generic words from the ‘people’, ‘nature’, man-made objects, and ‘animal’ Synsets to generate a list of common terms that might be found in classical artworks.

Running the model with this custom dictionary proved much more successful at identifying objects. Bounding boxes with the object labels are drawn on the artwork images and using the Segment Anything Model (SAM), masks of the objects are cropped from the image and saved separately for use as input into a GAN.

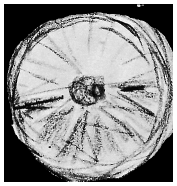


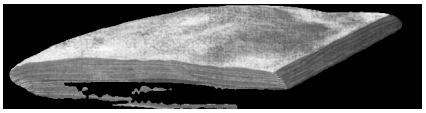
Detic with Custom Vocabulary

Not Always Accurate but Close

Examples of SAM mask crops








There were two main issues with this initial project's progression. First, the resulting image crops when used as inputs to a GAN resulted in subpar results. Second, there was an over reliance on pretrained models for the object identification and image segmentation portion of the project. For these reasons, it was reimagined into the second project detailed in the sections above where there was no reliance on pretrained models and the resulting image synthesis was able to produce much better defined results.

Conclusion

Image synthesis is a novel field in Machine Learning with a variety of potential applications. This GAN model aims to convert rough sketch work into finished painting in what is essentially a medium transfer without the need for actual transfer of media. It is aimed primarily at artists but its practical applications go well beyond that. By giving the user the ability to supply any sort of color scheme they desire along with a sketch or contour drawing, the possibilities for new syntheses and conceptualizations are without bound.

Future considerations for this project would be the introduction of a per-pixel color matching during training to ensure that not only are colors

from the supplied palette being used and no others but also that they are being placed in roughly the appropriate places. Further validation on sketch datasets that are intermediary between simplistic and complex (i.e., those of a medium level of complexity which was not tested during this project) will further confirm or refute generalizability of the model.