CSCI 1430 Final Project Report:
Album Cover GAN

Team ”MS Paint”: Hunter Karas, Jack Riley, Michael Xu
Brown University

Abstract

Generative adversarial networks are already commonly used to generate any kind of image, given some text input. Our project implements 2 GANs: one using transfer learning using a pretrained StyleGAN2 model, and one using our own GAN implementation. Our results show that defining characteristics in album art are rare, but potentially present. Nevertheless, using GANs to generate album art is a difficult task given the great variety in album art designs.

1. Introduction

With this final project, our group aspires to utilize transfer learning and the concept of generative adversarial networks (GANs) to create a GAN capable of generating album covers using a dataset of existing album art. The most famous album covers, such as Pink Floyd’s *Prism*, can become popular works of art. As a group, we are curious if we are able to create tasteful album covers that can also be considered as art pieces.

We understand the difficulty of designing a GAN that can produce acceptable album covers. Training and dealing with GANs can be finicky, and deciding what art is acceptable is a subjective task. Nevertheless, we are excited to present the results of our GAN-procured album art, which potentially shows some patterns present in album art designs.

2. Method

(Repo available here.)

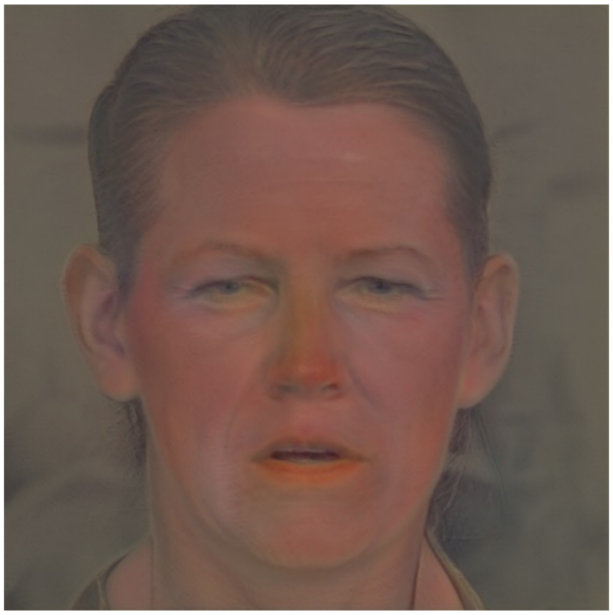
Our first step was collecting a large dataset of album covers. We decided on scraping from Spotify for its large amount of available songs (around 60 million) and easily-accessible API. The Spotify API has endpoints for collecting tracks from a given playlist [2]. So, to gather our dataset, we found playlists with a large number of tracks (in the thousands), and ran a program that iterated through all of the playlists to download the album cover of each song on the playlist. The scripts we used are available in the repo linked above. Spotify only allows access for 100 songs at a time, so we used an offset variable to navigate our playlists, which each run about 10,000 songs long.

Along the way, we resized the images from 640x640 to 512x512 and ensured that all grayscale images were converted to RGB so that they could be used in our GAN model. Our original dataset was comprised of about 11 thousand images, but after seeing our first GAN results, we increased the dataset to about 30 thousand images in hopes of producing better results, allowing the discriminator to pick up patterns typical to album covers (if there were any).

Collecting the images was quick enough. With good internet, 30 thousand covers were retrieved in about 45 minutes. However, resizing and converting grayscale images to RGB took around 340 minutes.

Our original concept involved using transfer learning for our GAN. We took the model for StyleGAN2, [4], NVIDIA’s proprietary GAN. It began in 2018 as a project with the goal of generating convincing human faces, but is now a popular option for retraining on different kinds of image datasets [3], which is why we selected it.

After retraining the last few layers of StyleGAN on a very small image dataset (100 images), we received this result from our GAN after 200 epochs.



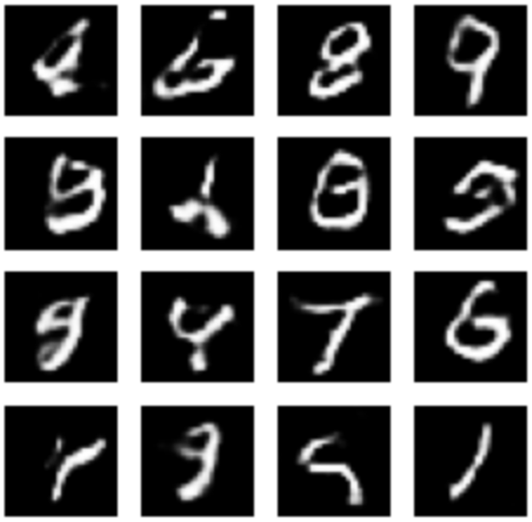
Nice, but not exactly what we were looking for. Running the program multiple times shows similar faces on slightly different colored backgrounds.

We had seen StyleGAN retrained on a dataset of illustrated bikes, and had hoped for a more specific ”album cover-esque” result. One possible explanation for this is the wide range of images that could be classified as ”album covers.” We had hoped to see that certain commonly replicated features of album art (perhaps the remains of a Parental Advisory sticker or colored border) might be represented in the generated image, but these features likely were not as frequently occurring as they needed to be to solidify themselves in the model. In the linked bike example, all of the images were illustrated from the same perspective, with the same size and style of bike. Even as we used the same ”amount” of transfer learning (number and kind of retrained CNN layers), our dataset of albums did not have significant repeated features, so StyleGAN's face generation persisted through the last few layers. The result is more of an ”album-cover-stylized” version of one of StyleGAN's faces.

Understanding this, we had a few options for modification. For one, we could divide and organize our album cover dataset by artist or genre (Spotify's API has fairly specific ”genre tags” for each song), and ask for user input specifying the artist or genre, in hopes that the smaller and more related album arts would have more common features or styles. Our second option was to abandon the album art concept as a whole, choosing a different kind of image to generate—one that has more defining/repeated characteristics. A third option was to create our own GAN so that faces would not be generated at all, and instead shift our focus to mapping patterns of design that are present in album art, rather than creating a viable album art product.

In the spirit of music and learning, we chose the third option in hopes that some defining album art characteristics might be seen if we eliminate StyleGAN's pre-trained layers.

We followed Tensorflow's GAN tutorial [1] to implement our own GAN from scratch. This tutorial works through the process of building and training a GAN to generate the MNIST dataset. The model architectures (discriminator and generator) are visible near the bottom of our repo's notebook file. As a starting point, our team followed the tutorial to completion and was able to generate images which closely resembled the MNIST dataset.



We then set out to adjust this model to take in our custom dataset and produce album art. To do so, we needed to modify the tutorials data loading scripts, model scaling methods, and the visualization code. First, we did some research on how to properly load the dataset into the model and decided to use a ’tf.data.Dataset’ object, which is a data processing object provided by tensorflow. This object allowed us to efficiently load the image data in batches to seamlessly integrate with the model. Next, we needed to adjust the scaling process used in the tutorial GAN. Initially, the generator model was programmed to take in a tensor of noise, with dimension (1, 100), and scale this up into a size of (28, 28). However, since our input images were of size (504, 504), we needed to modify the generator to upscale the noise tensor to this dimension. Lastly, we added some quick adjustments to the visualization section to show our single outputs, rather than the grid output of MNIST. The image below is an example of this model's output after training on 11,000 example images for 172 epochs.

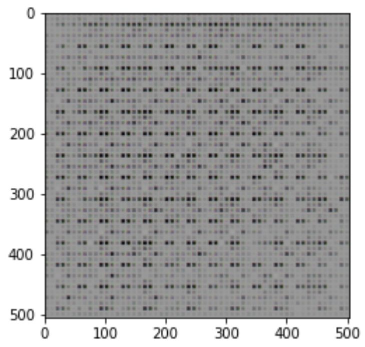


Figure 1. Epoch 172 on 11k album cover dataset

It is clear that this output does not resemble a modern album cover. However, we were not too surprised by this result. When we first proposed our project idea, we knew that training GANs is a very difficult task. As a result, the output of our "from scratch" model was within our expectations. Furthermore, we were only able to train this model on 11,000 images, rather than 30,000, due to RAM limitations of Google Colab. We did make multiple attempts to train this model on GCP, but failed to get substantial results. Additionally, these results are likely due to significant variation in our training data. Due to the subjective nature of album art, our dataset contained drastically different image examples. It is possible that this contributed to the poor results.

3. Results

Figure 1 shows the result of our own GAN after 172 epochs. Below, Figure 2 shows the result of epoch 100 on the 11,000 image dataset. There is not much variation to see. Because we are using a GAN, there is no objective way to assess our results—we must do so visually. It is easy to assess the fact that the output looks nothing like most album covers in our dataset, but we would like to know whether that failure is a result of the GAN or simply the nature of album covers. Most likely, it is the latter. Browsing through our album cover dataset shows a viewer the sheer variety in style, color, density, and content present in album art.

However, we do note that there is a noticable brightness around the edges, suggesting the frequency of brighter borders in album art, as well as a dark line at the top, potentially remnant of text lying at the top of covers naming the release. Rather than creating an album art product, this image seems to show album art design tendencies to include these features. We also note that we produced the image output in RGB colorspace, not gray, and the gray output is likely a result of the summing variety in album art colors, rather than an overwhelming presence of grayscale album art.

On performance: though our original proposed problem

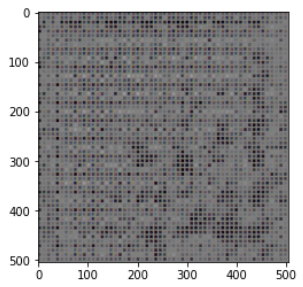


Figure 2. Epoch 100 on 11k album cover dataset

may have been contrived from the start, as discussed above, inefficient performance did limit our ability to achieve results on a larger dataset of 30k images using Google CoLaboratory, which has RAM and GPU RAM performance caps. We did attempt to run the transfer learning model on GCP with the 30k image dataset, but ran into NumPy, Tensorflow, Python version, and GPU compatibility errors when doing so, as StyleGAN2 requires NumPy 1.19 and Tensorflow 1.x, which in turn required a Python version downgrade. If we were to re-train the StyleGAN2 model on the larger dataset, however, it is still unlikely that we would have left the image of a face behind, as our dataset of album covers would still lack defining, universal characteristics.

3.1. Technical Discussion

One thing our group is wondering is how we could have improved our generated album covers. We mentioned how StyleGAN2 was used on a dataset of bikes and we hoped the successful application of StyleGAN2 on the bike dataset would yield success for own dataset. The stylish photo we generated with the 100-image dataset (see pg. 2) could be an album cover, but StyleGAN2's specialization with face images shows a little too much in our generated photo.
In our proposal, we mentioned how determining the success of our project would be tough because art is subjective. Maybe someone would consider our generated images for their next album cover. Another signal of success is whether our produced images look similar to the album art we used to train our GAN, but we gave a wide variety of album art so the expectation is that our generated art is a collective representation of the dataset.

3.2. Societal Discussion

- *Ethical concern 1: Graphic designers hired by songwriting teams to create album artwork are threatened by the models. Designers will be forced to learn ML, or else become employed.*

  While there are many graphic designers who specialize in album art, it is not true that their only employment opportunities are within album art commissions. Their skills can be applied to many other media as well, and it is unlikely that mass employment will ensue.

- *Ethical concern 2: The graphic designers in the dataset will not be credited or compensated for their input to GAN album covers.*

  In our implementation, a single album artwork is one of 11 thousand others, and in a single given GAN result, its ”creative direction” is not derivative of a given artist or cover. Thus, the creative property of these designers is maintained.

- *Ethical concern 3: The dataset is Western-centric, and thus the GAN ignores non-western album art styles in its generation.*

  This is a fair point regarding our dataset, which uses songs gathered on playlists created by Western-based users. All that needs to change to fix this is to incorporate non-Western playlists when scraping Spotify.

- *Ethical concern 4: The album art, which will be displayed to the public, could potentially contain explicit or otherwise sensitive material.*

  Of course, this is true. However, no song today is ever released without the critical eyes of some person (manager, artist, etc.), who will also look over album art at some point before distribution and judge its material.

- *Ethical concern 5: How is intellectual property determined? If two artists use the same model but get different unique results, is there copyright infringement?*

  Another fair concern—a solution is to specify in the terms of the model's use that ownership cannot be claimed over another person's use of the same model to avoid frivolous lawsuits.

4. Conclusion

Album covers are not consistent enough in their features to generate using a generative adversarial network. Datasets that may have defining characteristics (like anything found on Google via string input, MNIST, or human faces) are better problems to solve using GANs, and retraining a face-geared GAN like StyleGAN2 with just any dataset does not guarantee results that match even match the ”style” of that dataset... there *is no* overarching ”style” with our dataset.

While the results were visually disappointing, we did not expect them to produce an output that could be placed as an album cover itself. We were curious, however, if there were any visual tendencies in album covers that might appear in the images. Indeed, the brighter ”ring” about the edges, and a potential hint of text at the top of epoch 100 shown in Figure 2 may be results of album cover tendencies to have a border and text that the top.

If done again, we might consider taking text as input for an artist name or genre tag, or otherwise locating a more specific dataset whose content has some permeating visual theme throughout.

References

[1] Deep convolutional generative adversarial network. https://www.tensorflow.org/tutorials/generative/dcgan. 2

[2] Spotify playlist api endpoints. https://developer.spotify.com/console/playlists/. 1

[3] Fathy Rashad. How to train stylegan2-ada with custom dataset. https://towardsdatascience.com/how-to-train-stylegan2-ada-with-custom-dataset-dc268f 2017. 1

[4] Janne Hellsten Tero Karras. Stylegan2 - official tensorflow implementation. https://github.com/NVlabs/stylegan2, 2021. 1

Appendix

Team contributions

Hunter GAN research, custom GAN implementation, transfer learning implementation with StyleGAN2. Report Method section.

Jack Collected album cover dataset via Spotify API, custom GAN implementation + GCP debugging. Report Method/Societal Discussion sections.

Michael: GAN research and transfer learning implementation with StyleGAN2. Designed and printed poster. Remaining report sections.