StyleGAN PlayGround

Georgetown University

ANLY 590: Neural Networks and Deep Learning

Professor James Hickman

Group-1 Members: Dian Zhi | Ruobing Yan | Muwen You | Zihang Weng | Hanshen Jing

Introduction

Our project goal is to understand, implement and train a GAN model to be able to produce realistic fake faces and further explore the possibility of mixing their features to generate new images. StyleGAN can competitively pick apart all the relevant features of human faces and recompose them in a coherent way. From the previous researches and experiments, we can easily find out that the fake face image generated by StyleGAN almost has no difference with the real face from the skin texture, facial features, even hairstyle, and accessories. According to its superiority, we choose to use StyleGAN architecture to generate fake images based on the training datasets from two different styles and then combine the features from both of them to create a mixed image set to see what surprising result we can achieve.

GAN & StyleGAN

Generative Adversarial Networks (GANs) are a deep-learning-based generative model. GANs were first introduced by Ian Goodfellow and other researchers at the University of Montreal. The GAN model architecture involves two sub-models: a generator model for generating new examples and a discriminator model for classifying whether generated examples are real. The generator model takes a fixed-length random vector as input and generates a sample in the domain. In the case of GANs, the generator model applies meaning to points in a chosen latent space, such that new points drawn from the latent space can be provided to the generator model as input and used to generate new and different output examples. The discriminator is a normal classification model. The model takes an example from the domain as input, real and generated, and predicts a binary class label of real or fake.

StyleGAN was developed by a team of researchers at NVIDIA, and it is a type of GAN which is able to generate high-resolution images with fine details. Traditional GAN takes as input a random vector and generates the image corresponding to it. If one wants to change the eye colors, then one needs to try different values for the input vector, which might lead to other face features being changed. StyleGAN solves this limitation by using a hierarchical generative model that can learn the complex distribution of data in the training set. The model has multiple layers, and different range of layers learns different levels of detail in the generated image. For instance, the ending layers learn the finest features, such as eye color. Another application of StyleGAN is style mixing, and we use it in our project heavily. Style mixing allows merging two objects such that the resulting image has features from both sources. Users can obtain the style vector of certain artistic styles or backgrounds and change the style of the generated image.

Data Resource

We selected and adapted two different image datasets for the training set in order to train the StyleGAN model to explore the style mixing functionality. The first one is cartoon style. It's an animated series named Arcane, which is produced by Riot Games. Arcane (S1) is a fantastical tale that explores the backstories of champions and League of Legends mainstays. A lot of the characters and face annotations in this animation are available for us to take screenshots of. In order to create a more representative dataset, we manually chose over 1,000 images from the film's over 7,000 total screenshots that demonstrated the characters' positive traits. The other one is about real celebrity faces. It's called "CelebFaces Attributes Dataset (CelebA)" and we chose it from Kaggle. CelebA is a large-scale face attributes dataset with more than 200K celebrity images. In a similar manner, we randomly chose more than 1000 images to serve as the training set for the two contrasting styles.

Data Processing

In order to facilitate the training of facial features on the selected images, we need to perform pre-processing on the selected images. We divided the process into two steps: first, we identified the frontal faces in the chosen images, and then we cropped the identified facial regions into subimages. We implement face detection using the Haar cascade frontal face classifier in python using the OpenCV library. During cropping, the images were transformed into grayscale images to reduce the dimension of the images. For each face detected, we drew a rectangle around the face and cropped the detected faces based on the combined bounding rectangle points.

In the end, we go through all the images in the two customized datasets and manually select 1000 images from each dataset. As a result, the images in both datasets have been cropped to the same size and focused on the face. By putting all the processed faces into the same folder and trained by StyleGAN, we hypothesized that the model, after training, would learn not only realistic human facial features but also arcane-style human facial features.

Method

Processed arcane style face images and celebrity images were resized to 64x64 pixels. Both image sets were placed in the same training folder as the training data. StyleGAN3 model architecture was used in this project. We used 1 GTX 1080 GPU with a batch size of 32 and a gamma value (the R1 regularization) of 0.025 to run the training loops. For every 1024 images, the model weights and synthetic image examples were generated and saved. Each of these training iterations takes about 30 seconds. After about 1000 training iterations, the generated images stop to improve based on our manual check. The training process ended after 1400 iterations.

Result and Conclusion

After the training, the generator of the model was retrieved for evaluation. We first evaluate the generator's ability to generate celebrity faces, and arcane character faces separately. The model generated results were pretty realistic. This suggests that this StyleGAN model, after training, learned the feature space of celebrity faces and arcane character faces well enough to reproduce a realistic image based on a sequence of random seeds as input features. Next, we tested to see if we could get a mixed style good-looking image from the generator by feeding a weighted average of two feature vectors into the model. By adjusting the weight of the weighted average function, the generated result can either be close to the images generated by the first feature vector or the second feature vector. If two vectors have the same weight, then the generated image will tend to possess features of both images. Last but not least, we feed the weighted average feature vector of one celebrity face and an arcane-style character face to the generator. We expect the generated image will not only possess features from celebrity faces but arcane character face features as well. The result matched our expectations.

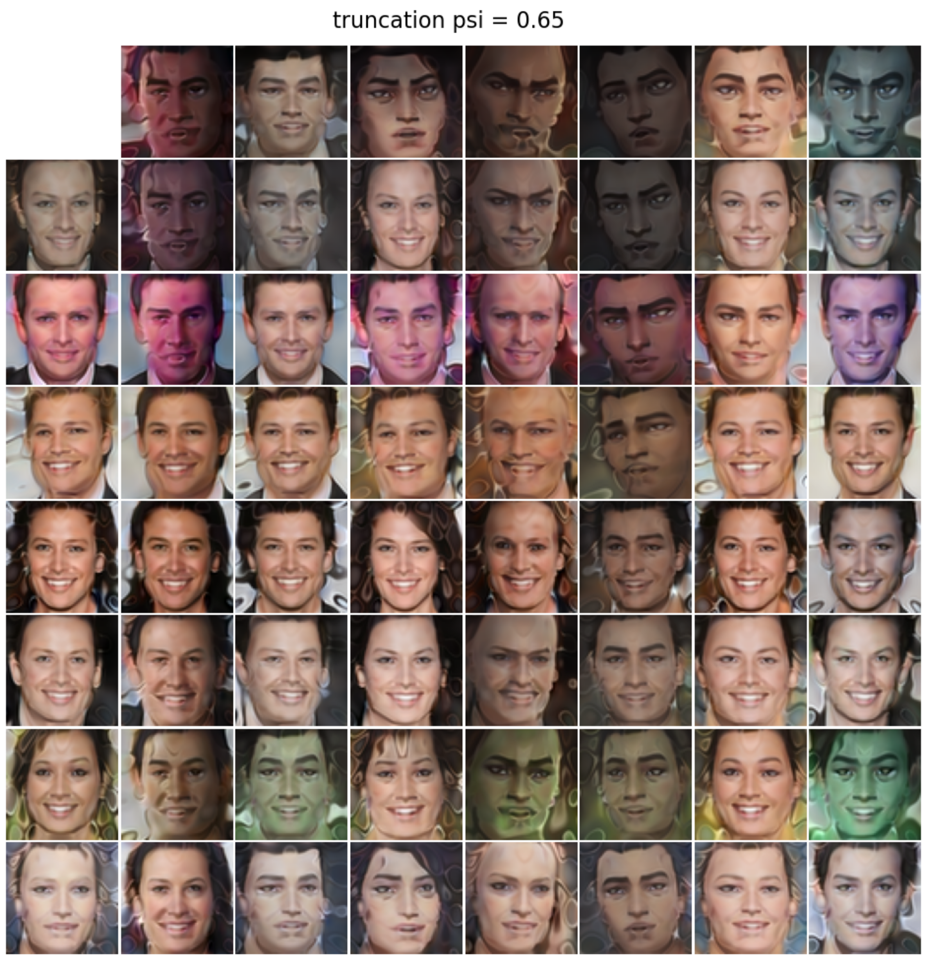

Fig 1. The first column is celebrity faces and the first row is arcane faces.

In conclusion, the trained model was roughly able to reproduce original images from feature vectors, and the model learned the feature space of the training data so well that it could generate images that have combined features from the two training datasets.

Reference

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4401-4410).