Faces combination and Faces Morphing with Progressive Growing of GAN

Yiran Guo
School of Computer Science
Simon Fraser University
yga58@sfu.ca

Betty Huang
School of Computer Science
Simon Fraser University
wha47@sfu.ca

Haokun Liu
School of Computer Science
Simon Fraser University
haokunl@sfu.ca

Cocoa Ding
School of Computer Science
Simon Fraser University
kexuand@sfu.ca

Michael Gergely
School of Computer Science
Simon Fraser University
mgergely@sfu.ca

Abstract

Deep Convolutional Generative Adversarial Networks (DCGAN) maps latent vectors to high dimensional images and has the capability of merging multiple images together to produce a new meaningful one by doing latent vectors arithmetic. The limitation of the latent vectors arithmetic in DCGAN is that the generated images are all mapped from a random noise. It is not straightforward to find an appropriate latent representation for any given image. In this project, we try to combine two real portraits to get a new one that contains facial features from both, also apply linear interpolation to morph. We use an existed method to find an approximated latent vector for any real facial image and apply arithmetic based on these two approximations in order to produce a combined result. Moreover, we use progressive growing of GAN to generate higher resolution images.

1 Introduction

Generative models used to create realistic images have developed rapidly during these years due to the proposal of generative adversarial networks (GAN)[1]. DCGAN[2] adapts Convolutional Neural Network(CNN) into GAN model which becomes a common and standard way for training GANs, but the training becomes unstable and produce images with high variation when generating higher resolution images. The progressive growing of GAN[3], which is recently proposed by Karras, T et al, has overcome this issue. It uses an advanced training methodology called progressive growing that gradually adds layers during the training process. By applying this method, GAN can generate more realistic images with higher quality(1024x1024 pixel)(Figure 1).

Typically, GAN consists of a generator and a discriminator. For the generator, it starts with a random noise as an input, then feeds into the neural network and produces a fake image. Originally, the initial input noise is drawn from the uniform distribution but recent research has suggested that sampling from a Gaussian distribution[4] is a better approach. For the discriminator, it takes both real and fake images as inputs and tries to classify them as real or fake. The generator tries to fool the discriminator. Two neural networks continuously compete with each other until the discriminator cannot distinguish between the real and the fake one.

One obvious limitation in GANs is that all generated images are mapped from a random noise in the latent space. We have no control over what images to be generated. Unlike variational autoencoder (VAE) which consists of an encoder and a decoder, there is no encoding procedure in GAN. In VAE, an image is encoded into a latent vector by the encoder and then the decoder reconstructs the latent

vector to the desired image. The generator in GAN is similar to the decoder in VAE but it starts with a random noise instead of a specific latent variable. VAE tends to produce blurry images while GAN can produce sharp ones.

One interesting experiment introduced in original DCGAN paper is that we can do arithmetic among different latent vectors and get a new image from generator. The new image contains information from all these latent vectors. For example, we can combine different faces together using GAN that is trained using CelebA dataset. But the procedure is manual if we want to combine two specific images. For example, to combine a man with glasses and a smiling woman, we need to find their latent vectors and then do the arithmetic. Because there is no encoder in GAN, the backward mapping is not straightforward. Latent vector arithmetic is limited to generated images since for each them has one exact corresponding representation in latent space.

The goal of this project is to merge two real face images into one that combines all features using latent vectors arithmetic of GAN. As illustrated above, we are seeking for the most appropriate method to transfer two input images into latent vectors. Lipton et al.(2017) proposed a method[5] which could precisely reconstruct the latent vectors for any generated images by performing gradient descent over the components of latent representations. With the help of this method, we can get a approximated latent vector for any given real image in order to apply the arithmetic. The original paper uses DCGAN to do the recovery. In our project, we test this method on a pre-trained PG-GAN in order to obtain a better synthetic images quality.

The focus of this project is to find best latent representations for any give images. The assumption of our work is that there exist a latent representation for any real images that can be interpreted by the generator of PG-GAN.

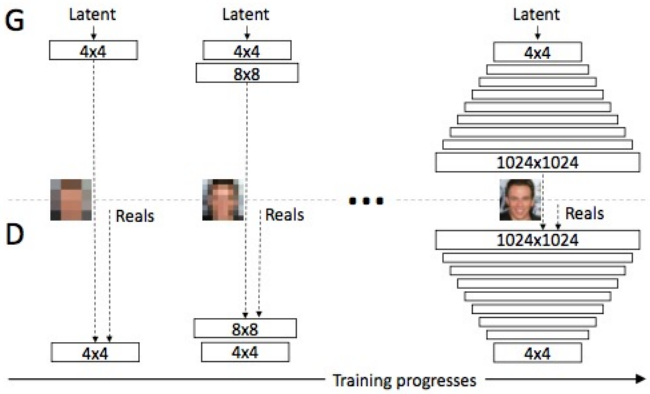

Figure 1: PG-GAN Architecture

2 Approach

To do backward mapping for any generated images, Lipton et al.(2017) applies the following idea: for a latent vector z, the mapping image by generator G is $G(z)$. The task is for the given $G(z)$ we need to do backward mapping to recover the corresponding latent vector z. Firstly, we initiate a random noise z' and its corresponding mapping image $G(z')$, then we have to find a z' value that is closest to the original z by minimizing the L2 loss

$$\min_{z'} ||G(z) - G(z')||_2^2$$

We keep updating z' in order to decrease the difference between our $G(z')$ and given $G(z)$ to find the best z' value. For the optimization over z', we apply gradient descent to update

$$z' \leftarrow z' - \eta \nabla_{z'} ||G(z) - G(z')||_2^2$$

As the original paper points out, the loss function is a non-convex shape which has multiple minimas. Hence the optimization does not guarantee to precisely recover the original latent vector z from z'.

But for the image that is created by the generator, its recovered latent vectors is robust enough for generator to produce an indistinguishable image from the original one.

The latent vector is originally drawn from a bounded domain. For training DCGAN, noise is sampled from a uniform distribution in the range of [-1,1]. For PG-GAN, the latent vector is sampled from a Gaussian distribution which central at 0, with a variance value of 1. When updating z', it is possible that some components may jump out of the range. To keep all the components inside of this domain, authors of original paper propose a new clipping method called stochastic clipping. This method reassigns the out-bounded component of z' a new random value from the same distribution. As the paper mentioned "While this cant guard against an interior local minima, it helps if the only local minima contain components stuck at the boundary". After applying the new technique, optimization is modified as:

$$z' \leftarrow clip(z' - \eta \nabla_{z'} ||G(z) - G(z')||_2^2)$$

We apply the recovery method, that is tested on DCGAN model in the original paper, on the GAN that has trained by progressive growing. The method works well for any given images but need to be carefully specify the initial random noise z' and the learning rate η.

Figure 2: dcgan - reconstruct

3 Experiment

3.1 Dataset

The pre-trained PG-GAN[6] is trained on CelebA-HQ dataset which is contributed by authors of PG-GAN paper. They manipulate the original CelebA dataset and produce a CelebA-HQ dataset which contains 30k reconstructed image. The reconstruction steps are demonstrated in the PG-GAN paper with details.

3.2 Latent Vector Approximation

When dealing with unseen images, that is, images from real world instead of those been generated, we treat the real image as $G(z)$ where we assume there exists a specific latent vector z for the given unseen image. We use this method as an encoder to encode a given image to latent space. Although it is almost impossible to get a precise latent vector that can be interpreted by the generator, we can still get a reasonable approximation by using the optimization shown above.

We first run the recovery method with DCGAN model described by Radford et al. (2015) which is trained on CelebA dataset. As the result described by Lipton et al.(2017), latent vectors can be recovered precisely for generated images. We apply this method to a real image, which the generator has never seen before, and run it for over 10k iterations as the paper suggests. Figure 2 shows that it is difficult to encode an latent vector for DCGAN to capture facial features and the output is really blurry since the image quality is an important factor for future face combination.

The dimensions of latent vectors are variate for different GAN models. DCGAN uses 100 dimensional latent vector as input, while progressive growing of GAN uses 512 dimensional latent vector instead for higher image resolution propose. The dimension of latent vectors is one of the key factors that influence the resolution of the output image. Because PG-GAN is trained to interpret larger latent vectors, it is easy for generator to produce a clearer face.

We noticed that when minimizing the L2 loss, small mean square error does not lead to a better latent representation, z can be easily stuck in a local minima. As the paper mentioned, the stocastic clipping can not guard against local minima. So the initial in this method is important for encoding real images.

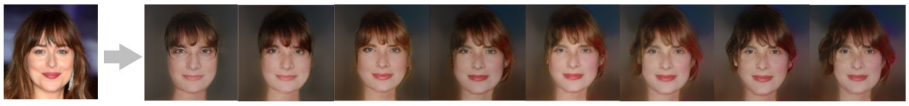

Figure 3: $G(z')$ for every 100 iterations. This shows the process of reconstructed images $G(z')$ while updating z'. The initial random noise z' and the learning rate η need to be carefully specified to produce a similar-looking face image.

3.3 Latent Vectors Arithmetic

As the original DCGAN paper points out, latent vector arithmetic can produce a combined synthetic image, we apply the same idea on PG-GAN. We initially test on multiple random noises rather than encoded approximated latent vector. The original DCGAN paper mentions that before doing latent vectors arithmetic, it is better to average three latent vectors with similar characteristic in pixel space for each operand in order to make the arithmetic more stable. We find that PG-GAN does not suffer from such unstablility. It has a strong capability of producing new reasonable synthetic images even if we directly do arithmetic over two original latent vectors (Figure 4).

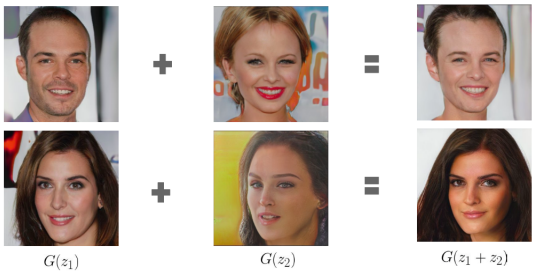

Figure 4: Face Combination

We then perform arithmetic over two approximated latent vectors that we calculate from section 3.1. The desired latent vectors that combine information from both mapping images can then be produced. The overall procedure is shown as Figure 5. Because stochastic clipping is applied when updating z', we got slightly different latent vector each time which can be decoded into different images.

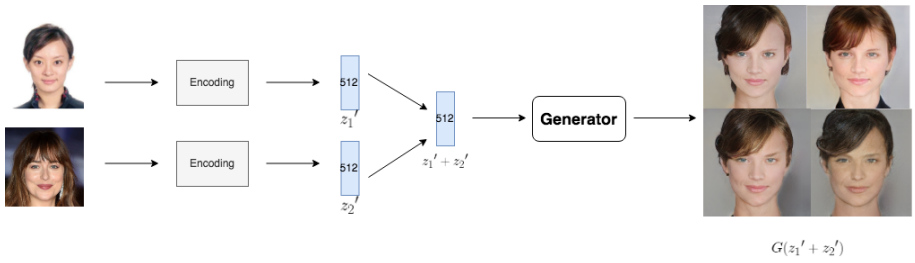

Figure 5: Architecture

3.4 Faces Morphing

To do faces morphing, we need to find a method that can transit one image to the other smoothly. Fortunately, each latent vector is drawn from a continuous distribution, hence the latent space is also continuous. Then we can use interpolation to traverse between two given latent vectors in latent space. We can see each latent vector as a point in high dimensional latent space, then the interpolation between two points is straightforward. Figure 6 shows the interpolation process of reconstructed images at each step.

We found that faces morphing using PG-GAN model between two realistic images is more smoother than using computer graphic technique. The details from the original images can be transit smoothly from one to another during interpolation. But a successful morphing between two real images requires good latent vector approximations. An appropriate latent vector approximation can produce a reconstructed output that is similar to the original real image. It can eliminate the gap between the real image and the first reconstructed mapping image.

Figure 6: Interpolation

4 Conclusion and Future Work

It is better to train a face encoder using generated images to predict its corresponding latent vector, since training data can predict latent vector for any given face image. There also exist some other techniques (e.g. identity-preserved) to optimize the encoded latent variable to make the reconstructed image more similar to the input image (Antipov et al. 2017)[7]. A simple neural network can do the work of an encoder, but the training for a high resolution is expensive. Applying latent vector optimization techniques over our approximation may also produce a better latent representation for the input image.

5 Contribution

Yiran Guo: Manipulate and run code for face combination and interpolation experiments, and write report and poster
Betty Huang: Did latent vector recovery experiments using DCGAN, and write report and poster
Haokun Liu: Try different methods of optimization of latent vectors, and write report and poster
Cocoa Ding: Design poster, and write report and poster
Michael Gergely: Write report and poster

6 References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

[2] Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

[3] Karras, T., Aila, T., Laine, S., Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.

[4] White, T. (2016). Sampling generative networks. arXiv preprint arXiv:1609.04468.

[5] Lipton, Z. C., Tripathi, S. (2017). Precise recovery of latent vectors from generative adversarial networks. arXiv preprint arXiv:1702.04782.

[6] Ptrblck. (2017, November 04). Ptrblck/prog$_g ans_p ytorch_i nference. Retrieved from https$: $//github.com/ptrblck/prog_g ans_p ytorch_i nference$

[7] Antipov, G., Baccouche, M., Dugelay, J. L. (2017, September). Face aging with conditional generative adversarial networks. In Image Processing (ICIP), 2017 IEEE International Conference on (pp. 2089-2093). IEEE.