BertonGan: a conditional GAN for performing various tasks

Aaron Schindler, Herbert Wright

December 16, 2022

1 Introduction

1.1 Problem Statement

Generative Adversarial Networks (GANs) [1] tend to require a lot of data. We wish to construct a network that can, given a few images of an unseen face, construct face images/deepfakes that are similar to that one, even if it has not seen that person before. [2] is an example of face swapping, but without the few-shot learning of new faces.

Deep learning is the traditional method of generating and detecting deepfakes [3]. While some computer graphics methods can be used, they lack the same ability of deep learning architectures to capture and learn complex functions efficiently. Because the images generated by deepfake architectures are very realistic, most humans are not able to distinguish between real and fake images. We can use a different (or same in some cases) models to learn the subtle differences between real and fake images, like differences in noise or counts of pixel colors, to effectively decide if an image is real or not.

1.2 Generative Adversarial Networks

Generative Adversarial Networks were introduced by Ian Goodfellow [1].

2 Methods

2.1 BertonGan structure

Our approach is to train an encoder-decoder while also simultaneously training the discriminator network. Both the decoder and discriminator(s) will be conditioned on a latent variable that provides all necessary facial information. The discriminator(s) will give two values corresponding to whether or not the reconstructed image is fake and if the image also corresponds to the latent variable it has been conditioned on. The four network components of our project are outlined below:

1. Face encoder network: $f_F : \mathbb{R}^{n \times W \times H} \to \mathbb{R}^{h_f}$
   (a) Input: n images of the same subjects face
   (b) Output: A latent representation of the subjects face
2. Image encoder network: $f_I : \mathbb{R}^{W \times H} \to \mathbb{R}^{h_I}$
   (a) Input: An image of a subjects face
   (b) Output: A latent representation of the image
3. Image decoder network: $f_G : \mathbb{R}^{h_F + h_I} \to \mathbb{R}^{W \times H}$
   (a) Input: Latent representations of a face and image

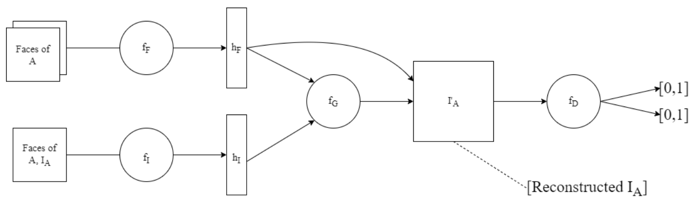

Figure 1: Figure 1 is a visual representation of our total network, comprising each of the four components above

   (b) Output: A reconstructed image decoded from the latent features

4. Discriminator network: $f_D : \mathbb{R}^{W \times H} \times \mathbb{R}^{h_F} \to [0, 1]^2$
   (a) Input: An image and a latent representation of a face
   (b) Output: Two probabilities
      i. Probability of being a fake image
      ii. Probability of being a different person than the faces encoded into the latent vector

A visual encoding of the networks described above is given below:

2.2 Networks

2.3 Training Procedure

We propose using two datasets to train/test. The first is the built in MNIST dataset from the torchvision package. This dataset will allow us to experiment with our networks on a smaller scale, as the MNIST images are 28×28 images of numbers rather than people. The next step is to use the celebA dataset which is also built in to the torchvision package. Our original plan to use the MS-Celeb-1M dataset [4] has been changed because the dataset is no longer publicly available. The celebA dataset comprises over $200,000$ images of faces, that are [**INSERT IMAGE SIZE HERE**] in size.

Given a batch $\beta = (F_A, I_A, I_B)$, where $F_A = n$ faces of person A, $I_A = N$ other faces of person A, and $I_B = N$ faces of people other than person A, we will compute the following quantities:

1. $h_F = f_F(F_A) \in \mathbb{R}^{h_F}$
2. $h_I = f_I(F_B) \in \mathbb{R}^{N \times h_I}$
3. $h_B = f_I(F_B) \in \mathbb{R}^{N \times h_I}$
4. $I'_A = f_G(h_F, h_I) \in \mathbb{R}^{N \times W \times H}$
5. $I'_B = f_G(h_F, h_B) \in \mathbb{R}^{N \times W \times H}$

6. $(R_{A'}, C_{A'}) = f_D(I'_A, h_f) \in [0,1]^{N\times 2}$
7. $(R_A, C_A) = f_D(I_A, h_F) \in [0,1]^{N\times 2}$
8. $(R_{B'}, C_{B'}) = f_D(I'_B, h_F) \in [0,1]^{N\times 2}$
9. $(R_B, C_B) = f_D(I_B, h_f) \in [0,1]^{N\times 2}$
10. $D_A = \|I_A - I'_A\|$
11. $D_B = \|I_B - I'_B\|$

We will optimize f_D by maximizing R_A, R_B, C_A and minimizing $R_{A'}, R_{B'}, C_B$. We also optimize f_F by maximizing $C_A, C_{A'}, C_{B'}$ and minimizing C_B, D_A. Additionally, we optimize f_G, f_I by maximizing $R_{A'}, R_{B'}, C_{A'}, C_{B'}$ and minimizing D_A, D_B. All parameters are optimized by using stochastic gradient descent.

After training, we will be able to use $f_F, f_I,$ and f_G to perform face swaps. We will then use f_D to identify fake/real images generated using face encoding h_F. Additionally, we will use f_F and f_G to generate new images of an already learned face.

3 Experiments

3.1 MNIST Experiments

3.2 CelebA Experiments

4 Conclusion

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[2] B.-S. Lin, D.-W. Hsu, C.-H. Shen, and H.-F. Hsiao, “Using fully connected and convolutional net for gan-based face swapping,” in *2020 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 185–188, IEEE, 2020.

[3] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen, “Deep learning for deepfakes creation and detection: A survey,” *Computer Vision and Image Understanding*, vol. 223, p. 103525, 2022.

[4] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, “Ms-celeb-1m: A dataset and benchmark for large-scale face recognition,” in *European conference on computer vision*, pp. 87–102, Springer, 2016.