Image Generation with Generative Adversarial Networks

Arsum Chaudhary
Pomona College
anna2021@mymail.pomona.edu

Ben Colbeck
Harvey Mudd College
bcolbeck@g.hmc.edu

Gibson Friedman
Claremont McKenna College
gfriedman25@cmc.edu

Tejas Hegde
Harvey Mudd College
thegde@g.hmc.edu

Abstract

Generative Adversarial Networks (GANs) have emerged as a revolutionary paradigm in the field of machine learning, offering a powerful framework for generating realistic data samples. This paper provides a review of our exploration with GANs, including a discussion of the dataset and methods we used. Furthermore, we discuss our results and experiments we conducted. Additionally, we explore the challenges and limitations associated with GANs, including ethical considerations. Finally, we discuss avenues for future research, emphasizing the potential of GANs to revolutionize numerous fields, from creative design to healthcare and beyond.

1 Introduction and related work

The primary goal of this project on "Image Generation with Generative Adversarial Networks (GANs)" is to harness the power of GANs to create highly realistic and diverse images, with a focus on handwritten digits. We followed a PyTorch tutorial [Inkawhich, 2024] that works with human faces as its dataset. We modified the code to fit our purposes and chosen dataset, the MNIST dataset of handwritten digits. This technology has the potential to revolutionize industries such as entertainment, gaming, and fashion by enabling the creation of photorealistic character models without the need for extensive photoshoots or manual graphic design. Moreover, it can aid in data augmentation for training machine learning models where obtaining real-world data is impractical or poses privacy concerns. Such advancements could significantly reduce costs and time involved in content creation, while also opening new avenues for personalized media and virtual reality experiences. This topic fascinates our team due to its blend of deep technical challenges and profound ethical considerations. It represents a cutting-edge frontier in artificial intelligence that not only pushes the envelope for what is achievable in image generation but also necessitates a thoughtful exploration of its implications on society, privacy, and authenticity in the digital age.

2 Datasets

For this project we used the MNIST dataset [Deng, 2012]. The MNIST dataset consists of 70,000 handwritten digits with 10,000 being used for a testing split and the other 60,000 images used for a training split. The features in this dataset are 28x28 images of the numbers 0-9 with corresponding labels to the number that appears in the image. We decided to use the MNIST dataset for our project as the dataset provides usable image data in a smaller image format that will make model training

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

faster than it might be with other datasets with higher quality images due to the simple data structure of MNIST.

3 Methods

GANs use a two model architecture for a minimax algorithm. One model is called the generator; the other is called the discriminator. What makes the algorithm a form of minimax is the fact that the discriminator attempts to minimize loss while the generator tries to maximize it. The discriminator trains to correctly differentiate real and fake images. The generator trains to create fake images from a random input that trick the discriminator.

As recommended in Nathan Inkawhich's tutorial for GANs, we used Binary Cross Entropy loss to evaluate the models at each step. BCE loss is a standard loss function for one-hot encodings.

$$\text{Log loss} = \frac{1}{N} \sum_{i=1}^{N} -(y_i * \log(p_i) + (1-y_i) * \log(1-p_i))$$

Figure 1: BCE loss

We also used a similar neural network architecture as Nathan Inkawhich's tutorial, adjusted for the dimensions of the images from the MNIST dataset.

4 Results and experiments

In order to measure our results, we visualized the generator and discriminator loss over iterations during the training of the GAN. Initially, the generator loss was high, and after a couple thousand iterations, it leveled out, which was what we expected. The discriminator loss was about the same throughout training.

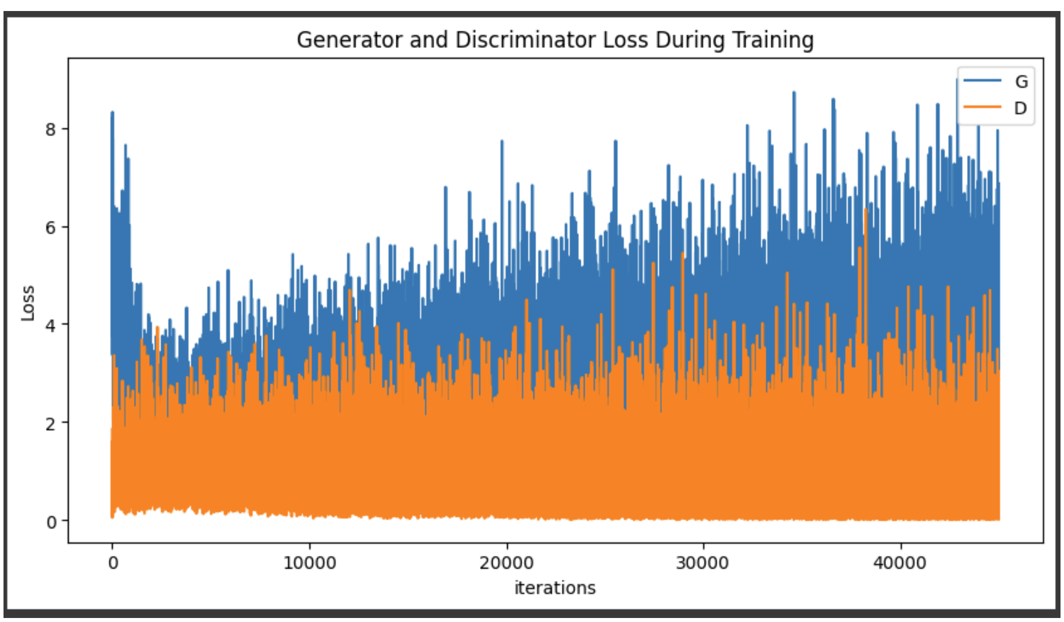

Figure 2: Loss vs iterations

In terms of the actual images generated, qualitatively they appeared to largely resemble handwritten digits. We included code that visualizes the images over the iterations.

Figure 3: Images generated by our GAN

On the whole, our GAN performed well, generating images that could pass as handwritten digits. Also, the loss was acceptably low as compared to other GANS and the tutorial.

5 Conclusion and future work

Based on our results in this project, we realized that while GANs may not be the most efficient or best way to generate images, the ease of training as well as compatibility across a wide range of datasets from simple ones such as MNIST to more complex datasets such as faces makes GANs a very good choice for image generation. To extend on this project in one way, we could try to train our GAN to generate more complex images such as faces or animals rather than just numbers as well as fine tuning our code to maximize the quality of the output. One problem that GANs could help to solve is in the area of data shortage. With data shortages quickly becoming a problem for companies looking to train future versions of their models, GANs can be used to create synthetic data to ensure that there is more than enough data to train other types of AI models.

6 Broader impacts

Generative Adversarial Networks present many opportunities to provide major benefits to society. For one, they can be used in creative industries such as fashion or art to help designers create mock-ups of their projects quickly and without wasting expensive resources such as fabrics or paints. GANs such as the DCGAN that we implemented have been used in medical image analysis helping to train models to diagnose patients without violating patient confidentiality laws by generating data. While GANs come with a lot of benefits they also can be very easily misued. For one, while GANs can help people in creative industries with their work, the rate at which GAN-generated artwork can be produced could lead to art markets being flooded with generated artwork potentially causing human-created work to be devalued and hurting people's jobs. Another way that GANs could be misused is in the creation of deepfakes. GAN-generated deepfakes can and have been used to spread misinformation and impersonate individuals. On a smaller scale this could lead to GAN-generated blackmail being created as well as people's privacy being violated and on a larger scale this could lead to public opinion and elections being influenced.

For training our GAN we used the MNIST dataset which is an unbiased dataset as it only contains numbers. In the reality of training a GAN, most datasets used can contain biases that can be inherited and sometimes even amplified by GANs. For example, if a dataset of faces was being used to train a GAN, certain cultures or demographics could be over or underrepresented which could cause a GAN to not accurately depict people of certain cultures. Datasets that contain biased or harmful images could also lead a GAN to generate stereotypical or harmful representations of people from different cultures.

Along with these risks of bias in datasets, the environmental impact of GANs can be very harmful as GANs require high performance GPUs and anywhere from a few hours to several weeks to train leading to a large carbon footprint when training a large model. In training our model, we utilized the T4 GPUs in Google Colab and training our model took roughly 5-6 minutes. While this might not have required a significant amount of energy usage, if we had trained our model on a larger or more complicated dataset it could have taken multiple hours and been much more energy intensive.

7 Code

Here is a link to our GitHub repository: `https://github.com/gibsonfriedman/GAN_Network`

References

Li Deng. The mnist database of handwritten digit images for machine learning research, 2012.

Nathan Inkawhich. Dcgan tutorial. `https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html`, 2024. Accessed: 2024-05-01.