DCGAN--Image Generation

Technical Report · February 2019

DOI: 10.13140/RG.2.2.23087.79523

1 author:

Ashutosh Chapagain
Kathmandu University

Abstract

The potential of artificial intelligence to emulate human thought processes goes beyond passive tasks and it extends well into creative activities. In this paper, we'll explore the potential of deep learning to generating real like images. We will use Deep Convolutional Generative Adversarial Network (DCGAN) which has proven to be a great success in generating images. We have discussed the theoretical aspect of GAN and also discussed about our methodology to create a DCGAN Model for MNIST Datasets and CelebA Datasets.

Introduction

Learning features of huge unlabelled data and preserving those features to create new set of data has a great scope in fashion, art and machine learning("Understanding Generative Adversarial Networks (GANs)", 2019). Here we present a machine learning model which generates images based on the feature provided by the training images. For our objective adversarial networks can learn good representations of images for supervised learning and generative modeling (Radford, Metz & Chintala, 2016).

Generative Adversarial Networks(GAN) belong to the set of generative models(Goodfellow,et al.,2014). The GAN model consists of two network

- A generative network G(.) that takes in random input z and returns x_g=G(z) that should follow the targeted probability distribution.
- A discriminator network D(.) that takes image vector x_image and classifies whether the generated image is real or generated.

The generator needs to learn how to create data in a way that discriminator isn't able to distinguish as fake. The discriminator network has the task to determine if the image is real or fake. An intuitive way to understand GAN is to imagine a forger trying to create a fake Picasso painting (Chollet, n.d.). At first, the forger(generator) is pretty bad at this task. As times goes on, the forger becomes increasingly competent at imitating the style of Picasso, and the art dealer becomes increasingly expert at spotting fakes.In the end, they have on their hands some excellent fake Picassos. That's what a GAN is: a forger network and an expert network, each being trained to best the other.

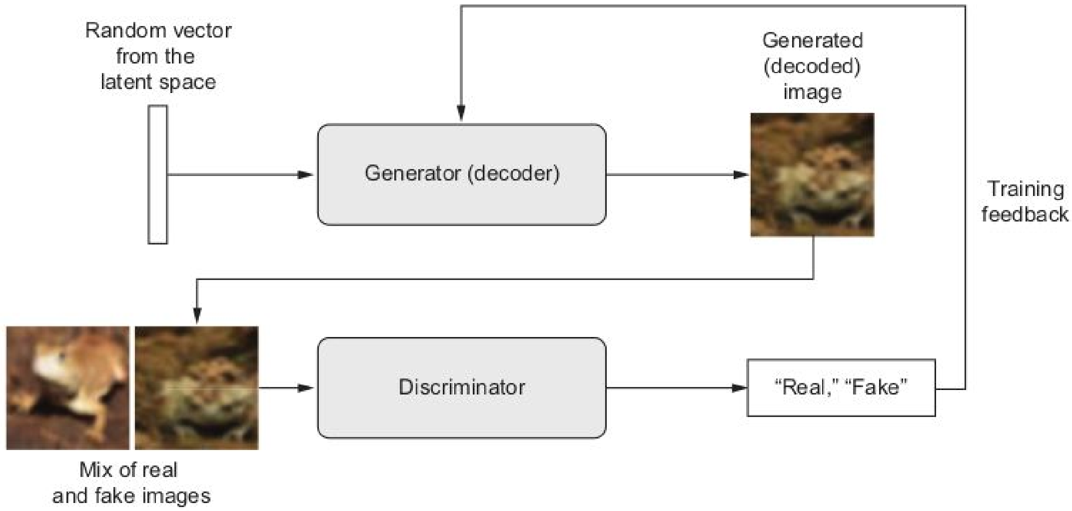


Figure 1: Architecture of GAN Model(Chollet, n.d.)

We will use a Deep Convolutional GAN (DCGAN) which is very similar to GAN, but specifically focuses on using Deep Convolutional networks in place of those fully-connected networks. Convolutional networks in general find areas of correlation within an image.

Related Work

Variational Encoders: They are a kind of generative model that's appropriate for the task of image editing via concept vectors(Rezende,et al.,2014). Variational Encoders turns the image into parameters of statistical distribution: a mean and a variance. The VAE then uses the mean and variance parameters to randomly sample one element of distribution, and decodes that element back to the original input (Kingma,et al.,2013). The stochasticity of this process improves the robustness and forces the latent space to encode meaningful representations everywhere: every point sampled in the latent space is decoded to valid output.

VAEs result in highly structured, continuous latent representations. VAE has a tendency to approximate roughly which is over simplified compared to the true complex distribution of the images. GANs consider the complexity of the distribution. Once training is over, the GANs are capable of turning any point in its input space into a believable image[Chollet, n.d.].

Methodology

Datasets

1. MNIST dataset is a curated list of all handwritten digits. We have used dataset for quick validation of our model. All images are scaled to 28X28.
2. The CelebA dataset consists of over 10k identities and 200k total images. All images are originally of size 160X160 pixels. They are rescaled to 28X28 pixels.

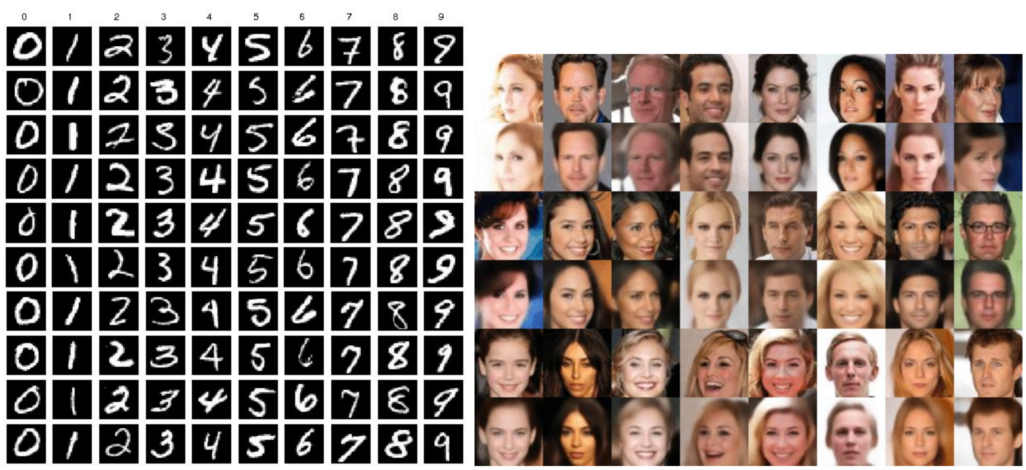


Figure 2: Datasets used MNIST (left) and CelebA (right)

Discriminative Model Implementation

For feature extraction 64 filters of size 3X3 were applied on the original image. Average pooling (2X2) and batch normalization was performed on the layers to reduce noise and to generalize the features.

Again the resulting layers were stacked on top of each other and 128 filters and 256 filters of size 3X3 each were applied. Each convolution layer was followed by average pooling and batch normalization.

The layer was flattened and dropout with probability 0.4 was applied. A Dense network was stacked on top of the convolutional network with an output of 1 which determined whether the image fed into discriminator was real or fake. The discriminator model was the classification model which classified the images as *real or fake*.

Generative Model Implementation

A generator network maps vectors of shape (latent_dim,) to images of shape (32, 32, 3) . The features of generative models are same as the discriminator except that it applies convolution with a fractional stride (convolution transpose) (Chollet, n.d.).

Optimizing the Model

Weights are updated as to maximize the probability that any real data input x is classified as belonging to the real dataset, while minimizing the probability that any fake image is classified as belonging to the real dataset. In more technical terms, the loss/error function used *maximizes the function D(x), and it also minimizes D(G(z))*.

Furthermore, the generator function *maximizes D(G(z))*.

Since during training both the Discriminator and Generator are trying to optimize opposite loss functions, they can be thought of two agents playing a minimax game with value function V(G,D).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

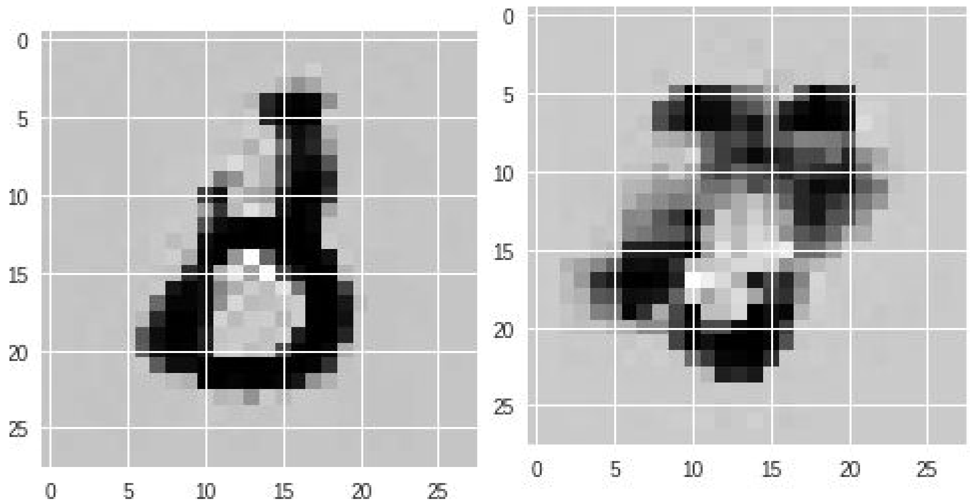
Models were trained for 100 epoches with a batch size of 32 for CelebA dataset.
For MNIST Dataset, the model was trained with an iteration of 50,000 and batch size of 32.

System Specification

Programming Language: Python3
Framework Used : Tensorflow
Development Platform : Google Collaboratory
Training Time : 4 hours for MNIST Dataset and 11 hours for CelebA dataset.

Results

The digits produced image for MNIST dataset were:



A B

Figure3: Image generated from our GAN Model. (A) generated 8. (B) generated 7(inverted 7).

Faces generated from CelebA dataset were:

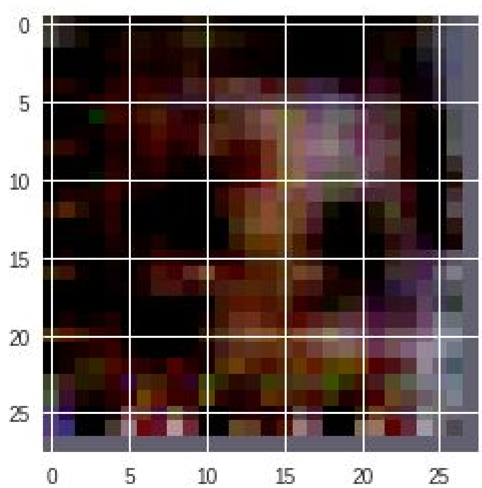


Figure4 : Image Generated from our GAN Model

These were the image generated from our GAN Model. Due to GPU constraints, we were able to train only for 3 epoches out of 100. Hence, we used a pre-trained model from https://tfhub.dev/google/progan-128/1.

The output when we fine tuned the last networks from the pre-trained network with our model were:

Figure: Image generated from our model on CelebA Dataset

Loss in Training

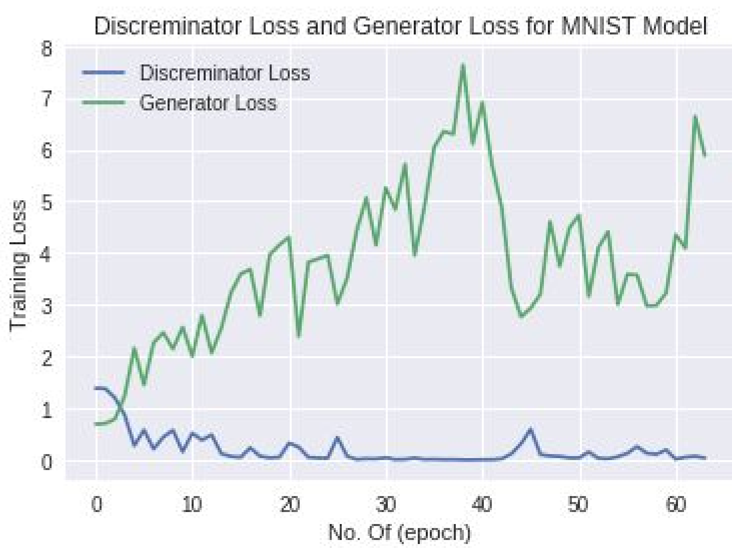


Figure: Training Loss for two models MNIST Dataset

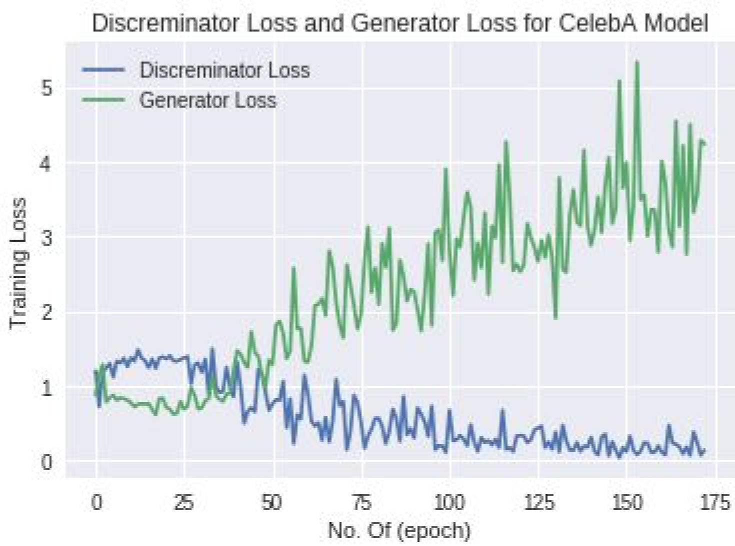


Figure: Training Loss for two models CelebA Dataset

As shown in the figure, in our model, the discriminator model overpowers the generator model.The respective crests and troughs of both the model are inverse to each other, i.e. one model overpowers the other. In ideal case both the models converges to 0 loss after training for a long time.

Difficulties And Shortcomings

1. When training, the generator loss begin to increase considerably, while the discriminative loss tends to zero, hence the discriminator overpowers the generator model. We had to be very careful to tune in the hyper-parameters.
2. Due to the constraints in GPU power, we are not able to generate a perfect image based on previous image.

Conclusion

Sampling from a latent space of images to create entirely new images or edit existing ones is currently the most popular and successful application of creative AI . In this paper, we demonstrated a way to generate images from training the existing similar images. GAN is a dynamic system where the optimization process is seeking not a minimum, but an equilibrium between two forces. It was difficult to train the generated image and they were not as good as the real image. Hence we had to use a pre-trained model for CelebA Dataset.

References

Chollet, F. *Deep learning with Python*.

Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *ICLR 2016*.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra, "Stochastic Backpropagation and Approxi-
mate Inference in Deep Generative Models," arXiv (2014), https://arxiv.org/abs/1401.4082.

Diederik P. Kingma and Max Welling, "Auto-Encoding Variational Bayes, arXiv (2013), https://arxiv.org/
abs/1312.6114.
Understanding Generative Adversarial Networks (GANs). (2019). Retrieved from https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29

ProGAN: How NVIDIA Generated Images of Unprecedented Quality. (2019). Retrieved from https://towardsdatascience.com/progan-how-nvidia-generated-images-of-unprecedented-quality-51c98ec2cbd2