A Study on Synthetic Data Generation

Naman Singh Nayal - 2001117

May 3, 2023

Introduction

In this era dominated by machine learning and deep learning algorithms, one specific limitation that everyone faces is the lack of good data to train and test our models with. This can be due to security or privacy reasons and at times the lack of data can be because of a lack of real-world examples. This can lead to issues of high variance or high bias, leading to inaccurate results.

Motivation

While going through several research papers which applied different models on different data sets, a limitation that was common across all of them was the lack of good data. A good data set isn't biased towards a particular subset of data and can provide a good variety. The data set should also be a reflection of the real world and should represent properly all the intricacies so that it can aide in making a proper model. For example, a human skin disease classification model, which required a lot of data, did not have relevant data because people didn't want to share the images of their skin. This led to a small data set that to having pictures taken at a specific angle. Thus models trained on this data set were not able to capture the actual features of the data set leading to poor performance.

Synthetic Data Generation as a solution

To overcome this we look towards synthetic data generation via the use of generative models. Generative models can augment a given image to introduce variance that is required by the data set, eliminating the bias also the images produces are synthetic, thus they don’t have privacy-based issues. This project looks towards three individual synthetic data generation models, that being Variational Auto-encoders, Generative Adversarial Networks, and finally the Denoising Diffusion Probabilistic Models.

Variational Auto-Encoders

Introduction

The class of generative models known as Variational Auto-encoders (VAEs) offers a logical approach to sampling from the model distribution. It is made up of an encoder and a decoder that are represented by neural networks. The encoder takes a picture as input and encodes the image’s mean and variance in the latent space. The decoder then uses this input to create an image by decoding the encoded information in the latent space.

Variational Auto-Encoders

Working

Traditional auto-encoders are used for image reconstruction. They take the input and encode them into a compressed latent space, which is later used to sample data to decode and reform the image. The goal of a variational auto-encoder is to find a distribution $q_\phi(z|x)$ of same latent variables through which we can sample from $z \epsilon q_\phi(z|x)$, to generate new samples $x' \epsilon p_\theta(x|z)$. Rather than forcing to produce a single encoding, variational Auto-encoder forces the encoder to produce a probability distribution function over the encoding. These probability distribution function correspond to a real feature of the object that have not been measured(mainly because there is no metric or technology to measure it with).

Variational Auto-Encoders

Loss Function

The loss function used by the variational auto-encoders is :

$$Loss = E_{z\epsilon q_\phi(z|x)}[p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p_\theta(z)) \tag{1}$$

where $E_{z\epsilon q_\phi(z|x)}[p_\theta(x|z)]$ corresponds to the reconstruction loss while $D_{KL}(q_\phi(z|x)||p_\theta(z))$ is the regularizing term calculated as the divergence of the chosen image from the latent distribution.

Variational Auto-Encoders

Block Diagram

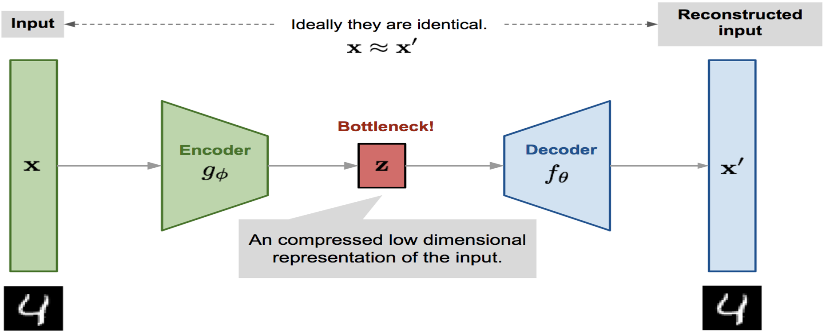
Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
$\mathbf{x}$
Encoder
g_ϕ
Bottleneck!
$\mathbf{z}$
Decoder
f_θ
$\mathbf{x}'$
An compressed low dimensional representation of the input.

Figure 1: Block Diagram

Variational Auto-Encoders

Resulting Data

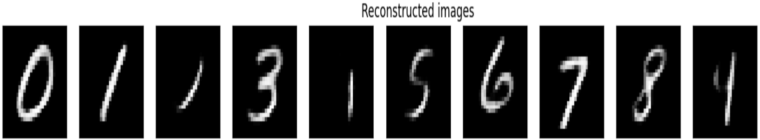

Figure 2: Epoch 99

Variational Auto-Encoders

Graph of loss vs Epoch

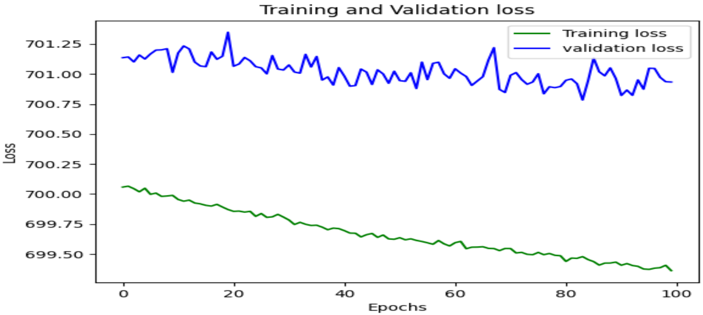

Figure 3: VAE loss vs Epoch

Variational Auto-Encoders

Uses

- The best use case for VAE is in text generation as text data does not rely upon quality in regards to pixels.
- Another use case is when time and variation in data are of the essence as VAE is the fastest model along with being the one that produces varying outputs.

Variational Auto-Encoders

Limitations

- The biggest limitation of VAE is that the generated output tends to be blurry and lacks sharp contrast between the background and the output.
- This reduces the quality of the generated image making it less viable to be used for data sets that rely on the high quality of the image to work on.
- This lack of quality makes the model unusable in a lot of cases.

Generative Adversarial Networks

Introduction

Deep neural net topology known as ”Generative Adversarial Models” (GANs) feature two neural networks that are in direct competition with one another. Generative refers to the process of creating a probability distribution function that closely resembles the distribution function of the original data. The conflict between the discriminator and generator networks is referred to as adversarial. As a benchmark for any new synthetic image-generating model that emerges, GANs have shown to be particularly beneficial in the synthetic data production department.

Working

The fundamental premise of GANs is an intriguingly straightforward one: let us pit two neural networks against one another in the hopes that the rivalry would spur them on to domination. The main objective is to produce material, such as photos, that is identical to the content found in the training data. Two distinct models are required to accomplish it:

1 A discriminator's job is to categorize images as real or fake based on their input, which might be actual or generated by a generator.

2 A generator that outputs graphics based on the input of random noise.

Generative Adversarial Networks

Loss Function

The loss function used by the generative adversarial networks is :

$$\min_G \max_D V(D,G) = E_{x\epsilon \text{þ}_{data}(x)}[\log(D(x))] + E_{z\epsilon \text{þ}_z(z)}[\log(1 - D(G(z)))] \tag{2}$$

where $E_{x\epsilon \text{þ}_{data}(x)}[\log(D(x))]$ corresponds to the real image sampled from the real set, and $E_{z\epsilon \text{þ}_z(z)}[\log(1 - D(G(z)))]$ corresponds to the fake image produced by the generator. The task of the discriminator is to maximize V(D,G) while task of the generator is the minimize the same.

Generative Adversarial Networks

Block Diagram

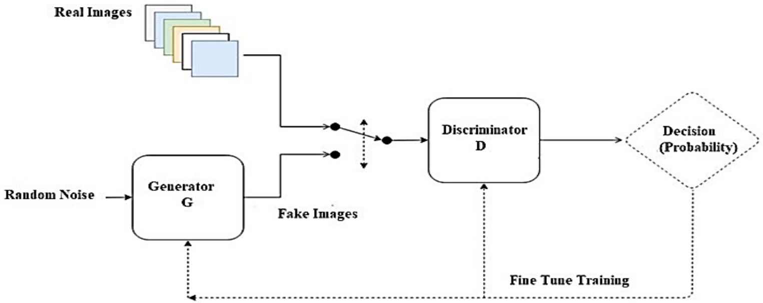

Figure 4: Block Diagram

Generative Adversarial Networks

Resulting Data

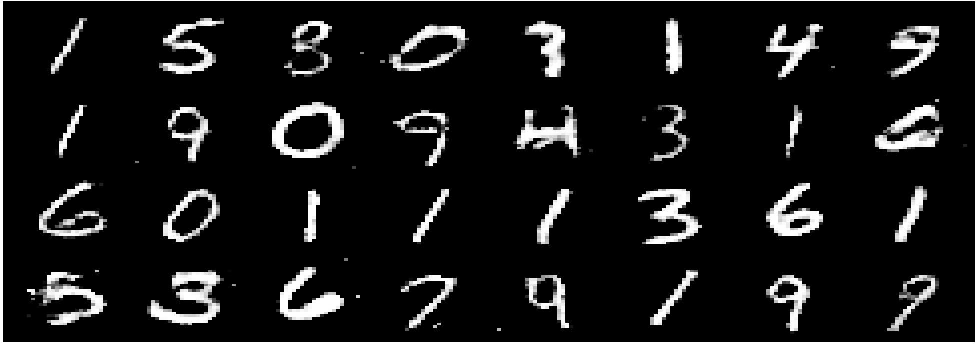

Figure 5: Epoch 99

Graph of loss vs Epoch

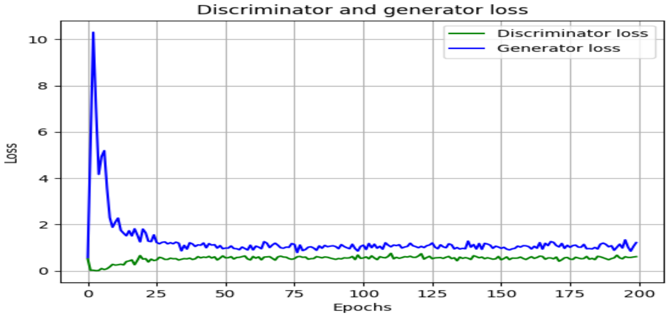

Figure 6: GAN loss vs Epoch

Generative Adversarial Networks

Uses

- The best use case for GANs is when number of classes to be generated by a model is low and there is a requirement for high-quality images
- ChatGPT uses GAN's to produce synthetic text to create authentic and engaging conversations.

Generative Adversarial Networks

Limitations

- Mode collapse: During the training, the generator may collapse to a setting where it always produces the same output. This is called mode collapse.
- Hard to achieve Nash Equilibrium making the model unstable.

Denoising Diffusion Probabilistic Model

Introduction

The way Denoising Diffusion Probabilistic models (DDPMs) function is by first destroying the picture in the forward phase and then creating it in the backward step of denoising. This is also referred to as a Markov Chain. In diffusion models, the latent space is of the same dimensionality as the input. The model's job is to forecast the noise that was introduced to each image. The diffusion model can take a random noise sample and produce an image that is highly similar to the collection of input photos by analyzing the noise added and eliminating it.

Denoising Diffusion Probabilistic Model

Working

It is challenging to produce a natural image from noise in a single step. Imagine creating an image in several smaller steps, sort of like how old-style photography would let an image develop on a Kodak film. As a result, each step's input and output should be easier for the neural network to process than going straight from pure noise to a final natural image. But how should it be expressed mathematically? Two processes are required:

1 A forward process, which turns a natural image into noise.

2 A backward process, which turns noise into a natural image.

Denoising Diffusion Probabilistic Model

Loss Function

The loss function can be expressed as:

$$L_t = E_{x_0,t,\epsilon}[||\epsilon - \epsilon_\theta(\sqrt{\hat{\alpha}_t}x_0 + \sqrt{1 - \hat{\alpha}_t}\epsilon, t)||^2] \quad (3)$$

where ϵ is the pure noise sampled at time t, x_0 is the original image, and $\epsilon_\theta(\sqrt{\hat{\alpha}_t}x_0 + \sqrt{1 - \hat{\alpha}_t}\epsilon, t)$ represents our neural network where $\sqrt{\hat{\alpha}_t}x_0 + \sqrt{1 - \hat{\alpha}_t}$ is the image produced by adding noise at timestamp t.

Denoising Diffusion Probabilistic Model

Block Diagram

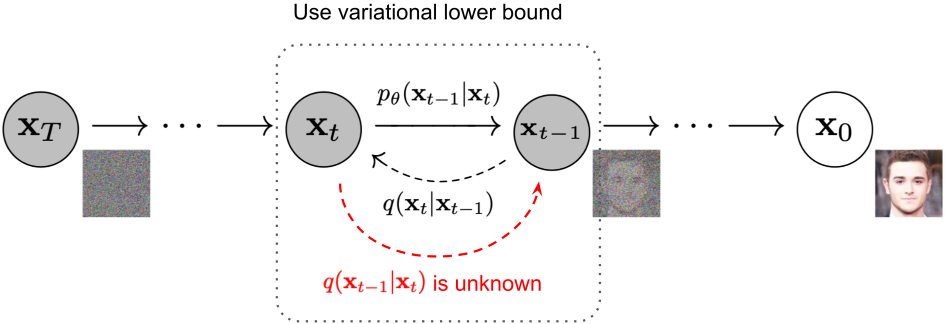
Use variational lower bound
$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is unknown

Figure 7: Block Diagram

Denoising Diffusion Probabilistic Model

Resulting Data

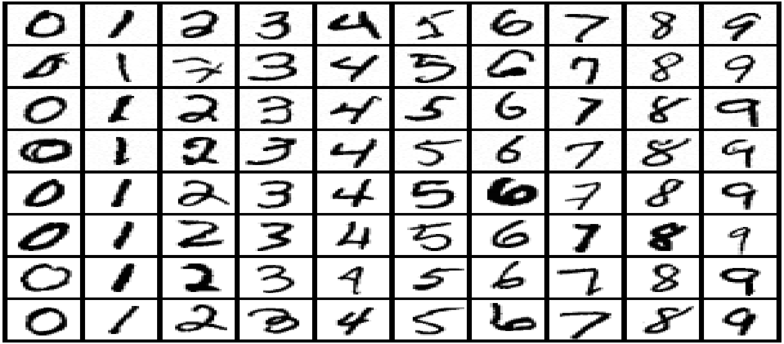

Figure 8: Epoch 99

Denoising Diffusion Probabilistic Model

Graph of loss vs Epoch

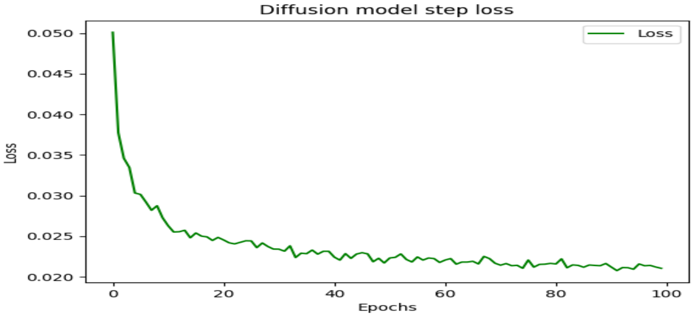

Figure 9: Graph

Denoising Diffusion Probabilistic Model

Uses

- DDPMs are relatively recent models and have been used extensively for image generation as well as text-to-image generation and so on.
- Imagen, Hugging Face and Dall-E are examples of websites that use DDPMs for text-to-image and image-to-image generation.
- They also find their uses in audio generation as audio signals can be noised and denoised in a similar fashion.

Denoising Diffusion Probabilistic Model

Limitations

- The major limitation of DDPMs is the slow generation process, as at every step it deals with the data at the same size as the original input.
- Another limitation is the inability for dimension reduction, which both VAE and GAN have shown are capable of.

Conclusion

The above three models shown performed decently when it came to generating samples for the MNIST data sets in 100 epochs. While the quality of output and the training time differed, the samples produced were reasonably good. This showcases how these synthetic data generation models can be used to create a good data set. The models aren't limited to generating image-based data only, they can also generate text-based as well as audio-based data. Thus, generative models can provide a solution to the problem of lack of data and synthetic data seems to be the step toward the future development of ML, AI, and data science fields.

Future Works

Synthetic data generation is a work in progress as there are no definitive models superior over one another. There are class of hybrid models, like some which use GANs along with DDPMs and also modifications of the models explored by using different types of neural networks or using different hyper-parameters. While these have improved the models, they haven't been able to overcome the limitations of the models as those limitations relate to the essential architecture. There is also a possibility of developing synthetic video generation, which will further blur the line between fake and real. However, synthetic data has the potential to not only replace, but surpass real data in terms of providing a proper data-set, though it still has a long way to go.