Graphic Generation with DifGAN

Group number: 2 Gui Yuxi Hu Yue Lyu Xinyi Xu Haorui

Project Number: 7 Instructor: Hehe Fan

July 21, 2023

Abstract

In this paper, we introduce DifGAN, a novel model that addresses the challenge of generating high-dimensional inputs. While vanilla GANs work well for low-dimensional noises and images, simply extending them to higher-dimensional inputs can be inefficient. To overcome this limitation, we explore the use of Diffusion-GAN, which compares noisy versions of real and generated images obtained through a Gaussian mixture distribution over diffusion steps. By extending both the discriminator and generator to multiple steps, DifGAN aims to provide more stable, diverse, and faster sampling capabilities. We conduct experiments on the Mnist dataset, comparing DifGAN with simple GAN, high-dimensional GAN, and Diffusion-GAN. Results show that Diffusion-GAN outperforms simple GAN and high-dimensional GAN, while DifGAN exhibits oscillations and pattern collapse issues. Further optimization and neural network structure adjustments are suggested to improve DifGAN's performance in generating high-dimensional images.

1 INTRODUCTION

We know that vanilla GANs can map low-dimensional noises into new images with the same statistics as the training set. But if it is changed into higher-dimensional inputs, simply using the original GAN model may still work, but are there other better models?

After consulting literature, we found an efficient method, Diffusion-GAN, to solve this problem. Unlike vanilla GANs, which compare the real and generated images directly, Diffusion-GAN compares the noisy versions of them, which are obtained by sampling from the Gaussian mixture distribution over the diffusion steps. By using each pair of noisy versions of them as the two inputs of the discriminator, we can get the output, which is the discriminator's judgement at this timestep. After calculating the weighted sum of all the outputs, we can get the "final output" of the discriminator, which can then be used to train discrimator and generator the same way as vanilla GANs.

After reimplementing the above Diffusion-GAN, we came up with another mothod to deal with high-dimensional inputs, which we called DifGAN. The difference with Diffusioin-GAN is that Diffusioin-GAN only extends the discriminator to T steps, while our DifGAN extends both the discriminator and the generator to T steps. The detailed principle is described in Part 2.

2 METHOD

Algorithm 1: DDPM Training

repeat
  $x_0 \sim q(x_0)$;
  $t \sim Uniform(\{1, \dots, T\})$;
  $\varepsilon \sim N(0, I)$;
  Take gradient descent step on $\nabla_\theta ||\varepsilon - \mathbf{z}_\theta(\sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\varepsilon, t)||^2$;
until *converged*;

Algorithm 2: DDPM Sampling

$x_T \sim N(0, I)$;
for $t = T, \dots, 1$ **do**
  $\mathbf{z} \sim N(0, I)$ *if* $t > 1$, *else* $\mathbf{z} = 0$;
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\mathbf{z}_\theta(\mathbf{x}_t, t)) + \sigma\mathbf{z}$;
end
return $\mathbf{x}_0$

Above is the pseudocode of diffusion. The trick of diffusion is to use N-step generation instead of one-step generation. We apply this idea to our model.

Algorithm 3: GAN

for *number of training iterations* **do**
  for *k steps* **do**
    Sample minibatch of m noise samples $\{z^{(1)}, z^{(2)}, \dots, z^{(m)}\}$ from $p_g(z)$;
    Sample minibatch of m examples $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ from data generating distribution $p_{data}(x)$;
    Update the discriminator: $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [logD(x^{(i)}) + \log(1 - D(G(z^{(i)})))]$;
  end
  Sample minibatch of m noise samples $\{z^{(1)}, z^{(2)}, \dots, z^{(m)}\}$ from $p_g(z)$;
  Update the generator: $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)})))$;
end

Above is the pseudocode of GAN. In every iteration, we first update the discriminator k times, then update the generator 1 time. As for the loss formula $logD(x^{(i)})+\log(1-D(G(z^{(i)})))$, if the discriminator is a perfect one, it will always tell real world image to be a real world image, i.e. $D(x^{(i)})$ will always be 1, and tell the generated image to be a fake image, i.e. $D(G(z^{(i)}))$ will always be 0. So for a perfect discriminator, the loss will always be 0. Additionally, the worse the discriminator is, the smaller the loss is. The loss formula will be applied to our model DifGAN.

Algorithm 4: DifGAN

for *number of training iterations* **do**
  sample m noise $\{z_T^{(1)}, z_T^{(2)}, \dots, z_T^{(m)}\}$;
  sample m image from real world $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$;
  for *t=T ... 1* **do**
    for *i=1 ... m* **do**
      $z_{t-1}^{(i)} = G(z_t^{(i)}, t)$
    end
  end
  $\nabla_{\theta_d} \frac{1}{m} \sum_{j=1}^{m} \sum_{k=0}^{T-1} \eta_k^d [logD(x^{(j)}) + \log(1 - D(G(z_k^{(j)})))]$;
  $\nabla_{\theta_g} \frac{1}{m} \sum_{j=1}^{m} \sum_{k=0}^{T-1} \eta_k^g \log(1 - D(G(z_k^{(j)})))$;
end

In every iteration, we sample m noise from Gaussian distribution and m image from real world. Then for every noise $z_T^{(i)}$, we input it into the generator G with parameter T, and the generator will output a latent product $z_{T-1}^{(i)}$, which is the input of generator in the next iteration. And we repeat the action T times, finally get the generated image $z_0^{(i)}$. As for optimization phase, it's quite similar to the one of GAN. The difference is we use the weighted sum of the losses in every iteration as the final loss.

3 EXPERIMENTS

To evaluate the performance of our DifGAN, we train on dataset Mnist and comparing our DifGAN with 3 other models, GAN, high-dimension GAN and Diffusion-GAN.

3.1

The results of GAN1, high-dimension GAN2, Diffusion-GAN3 and DifGAN4 on Mnist datasets of different batch are as follows:

For more accurate evaluation, we introduce the FID(Fréchet Inception Distance) method to quantify the image quality generated by these four models as the training epoch increases5.

800 10000 20000 40000 60000 80000 120000 160000 184800

Figure 1: The result of GAN on Mnist dataset, showing 9 images of corresponding batch

1000 10000 20000 40000 60000 80000 120000 160000 187000

Figure 2: The result of high-dimension GAN on Mnist dataset, showing 9 images of corresponding batch

800 10000 20000 40000 60000 80000 120000 160000 184800

Figure 3: The result of Diffusion-GAN on Mnist dataset, showing 9 images of corresponding batch

1000 10000 20000 25000 30000 32500 35000 40000 45000

Figure 4: The result of DifGAN on Mnist dataset, showing 9 images of corresponding batch

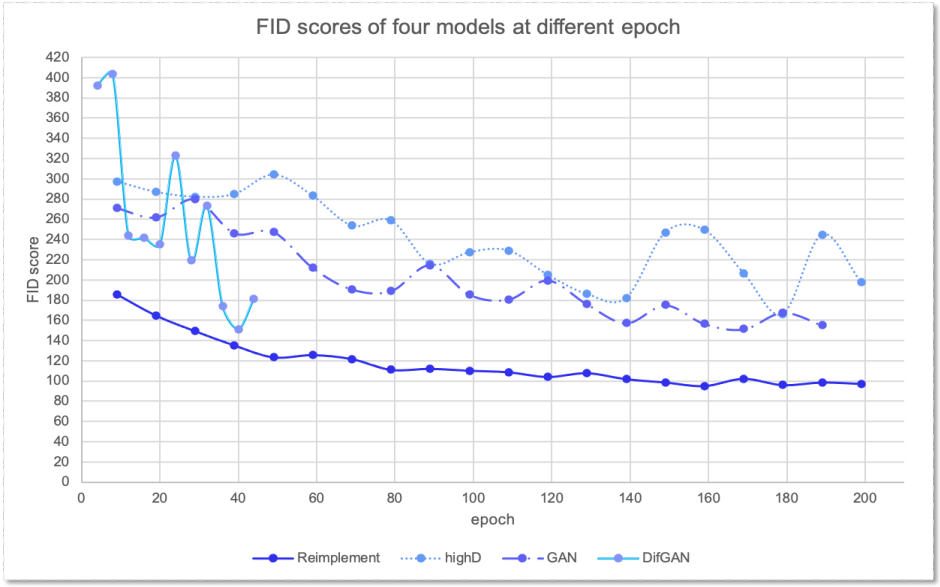

Figure 5: The FID scores of four models in Mnist dataset at different epoch

Combined with the above picture, it can be seen that high-dimension GAN is worse than GAN, in other words, after simply increasing the dimension, the effect of the model has deteriorated. And the result of Diffusion-GAN which we reimplement is much better than simple GAN and high-dimension GAN. The result of our DifGAN has very clear oscillations, but the trend is downward.

But when the training epoch exceeds a certain limit, the generated images of DifGAN go wrong.4 Our initial speculation is that there has been a pattern collapse. Maybe it can be avoided by optimizing neural network structure, adjusting the parameter, improving training strategies.

3.2

To explore the role of multi-step iteration of the image generation process in DifGAN, we output the intermediate result image.

t=1 t=10 t=20 t=30 t=40 t=50 t=60 t=70 t=100

Figure 6: The intermediate results of DifGAN on Mnist dataset, showing 9 images of different time step

It can be seen that the first picture has begun to take shape, but it is still slightly blurry. As the number of iterations increases, the picture becomes more clear.

4 DISCUSSION

We presented DifGAN - a new model combining diffusion model and GAN model. The idea is inspired by the symmetry among VAE, diffusion and GAN, it come to us that: if diffusion is VAE with higher-dimension internal variables, then there can be a model which assembles GAN and with higher dimension, featuring more stability, diversity and faster sampling, and it might be implemented by combining diffusion and GAN.

When exploring the method of combining diffusion and GAN, we reimplemented the earlier effort by hendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, Mingyuan Zhou of Diffusion-GAN[1], and then we raised the new idea of GAN by extending the generation process of GAN.

In the process of exploring this idea, we also met some problems:

a. We tried several ways of time embedding, but their results doesn't show much difference, so finally we chose the simplest way

b. On some computer, the pytorch-fid cannot run because of the error during the downloading of the Inception model. We solved this problem by using the valid model file downloaded by other teammate

```
    return torch.load(cached_file, map_location=map_location)
  File "D:\Anaconda\lib\site-packages\torch\serialization.py", line 815, in load
    return _legacy_load(opened_file, map_location, pickle_module, **pickle_load_args)
  File "D:\Anaconda\lib\site-packages\torch\serialization.py", line 1051, in _legacy_load
    typed_storage._untyped_storage._set_from_file(
RuntimeError: unexpected EOF, expected 17799 more bytes. The file might be corrupted.
```

c. We also made other effort during exploration, one of them features with more diffusion models: during each generation, take the noised image x_{noised} and a random timestep t as the input of generator. We implemented the idea but during the train the Loss doesn't converge and the results are not good. We speculate that this approach has some flaws in principle, and then we stopped further development of this idea

d. As we didn't implement the adaptive timestep, so there would be the situation that after the timestep exceeds a certain range, the improvement of the learning rate over timestep is not obvious, Especially with MNIST dataset. Considering the size and information carried by the pictures in MNIST is small, so we take small timestep as 100

e. When scoring the four methods with FID, we choose to transform the MNIST dataset into png, while the color depth between generated images and training set images are different, so we wrote simple program to unifies the color depth of both to 24-bit

The experiment showed the FID results of four methods, though the DifGAN models got higher FID score than the GAN methods, there are still many flaws in current results:

a. Training the models cost much more time than GAN

b. The diversity is not as good as expectation

c. The quality of the results fluctuated significantly, we speculate that this is due to the unreasonable setting of parameters and neural networks

To improve current result, there are some methods we can try:

a. Try more kinds of hyperparameters

b. Pre-calculate the best timestep T

c. Optimise the neural network of generator, including use better timestep embedding

5 CONCLUSION

Our study introduced DifGAN, a novel model that combines diffusion model and GAN, aiming to address the challenge of generating high-dimensional inputs. The inspiration behind DifGAN came from the symmetry observed among VAE, diffusion, and GAN models. By extending the generation process of GAN to multiple steps, DifGAN showed potential advantages in terms of stability, diversity, and faster sampling.

The experiments conducted on the Mnist dataset compared DifGAN with three other models: GAN, high-dimensional GAN, and Diffusion-GAN. The results showed that high-dimensional GAN performed worse than the traditional GAN, indicating that simply increasing the dimension did not improve the model's effectiveness. On the other hand, Diffusion-GAN outperformed simple GAN and high-dimensional GAN, demonstrating the effectiveness of its approach.

However, when analyzing the performance of DifGAN, it was observed that the generated images exhibited clear oscillations and started to go wrong as the training epochs progressed, possibly due to pattern collapse. To solve these issues, future work may involve optimizing the neural network structure, adjusting parameters, and refining training strategies.

During the exploration of combining diffusion and GAN, we encountered various challenges, such as issues with time embedding and model downloading. Additionally, another approach involving adaptive timestep did not yield satisfactory results, indicating potential flaws in the principle behind the method.

To further improve the DifGAN model, we may try different hyperparameters, pre-calculate the best timestep T, or optimize the generator's neural network architecture.

In conclusion, DifGAN presents an innovative approach that shows promise in generating high-dimensional inputs, but there is room for improvement in terms of stability, diversity, and overall performance. By addressing the identified challenges and optimizing the model, DifGAN has the potential to become a valuable tool for generating high-dimensional images with superior quality and diversity.

References

[1] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-GAN: Training GANs with Diffusion. *arXiv e-prints*, page arXiv:2206.02262, June 2022.