ML2022-2023 Spring HW06 Report

Report Questions

2022 HW06

Q1

Describe the difference between WGAN* and GAN**, list at least two differences.

Answer:
Here are two key differences between WGAN (Wasserstein GAN) and GAN (Generative Adversarial Network):

Loss Function:
The original GAN uses a binary cross-entropy loss. The discriminator is trained to distinguish between real and fake images by minimizing the binary cross-entropy loss, while the generator is trained to maximize the discriminator's loss for fake images. The loss functions for the discriminator and the generator in a standard GAN are:

$$\mathcal{L}_D = - E[logD(x)] - E[log(1 - D(G(z)))]$$
$$\mathcal{L}_G = - E[log(D(G(z)))]$$

WGAN replaces the binary cross-entropy loss with a Wasserstein distance (Earth-Mover distance) loss. This distance measures how much mass needs to be transported to transform one distribution into another. The loss functions for the discriminator and the generator in WGAN are:

$$\mathcal{L}_D = - E[D(x)] + E[D(G(z))]$$
$$\mathcal{L}_G = - E[D(G(z))]$$

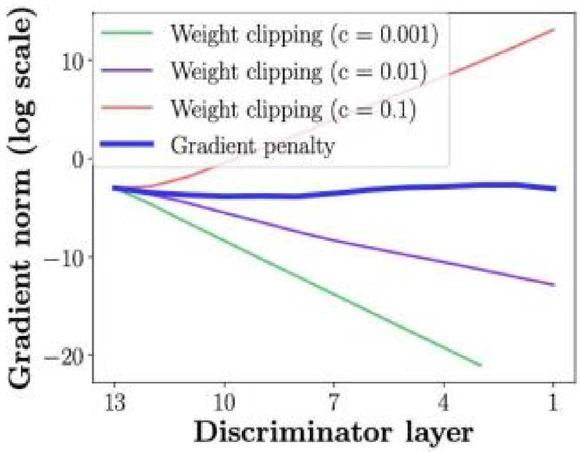
This modification aims to provide better gradients for the generator, leading to more stable training.

Gradient Penalty and Weight Clipping:
In the original GAN formulation, there is no explicit mechanism to ensure the discriminator's gradients are well-behaved, which can lead to training instability.
The discriminator can easily saturate (i.e., predict values close to 0 or 1 with high confidence), causing vanishing gradients for the generator.
WGAN introduces weight clipping to enforce a Lipschitz constraint on the discriminator.
This constraint ensures that the gradients are not excessively large, promoting more stable training.

Later improvements in WGAN-GP (Wasserstein GAN with Gradient Penalty) replace weight clipping with a gradient penalty, which penalizes the norm of the gradient of the discriminator's output with respect to its input. This approach more effectively enforces the Lipschitz constraint.

Q2

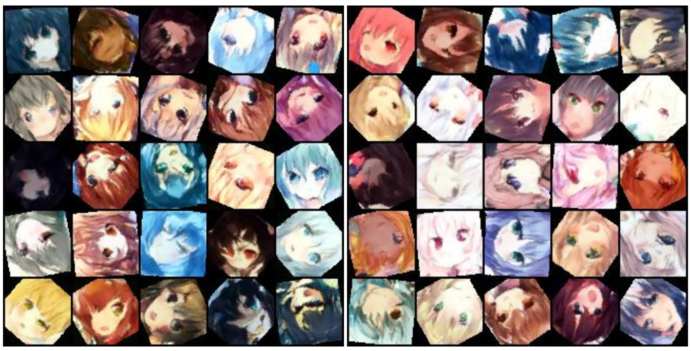
Please plot the “Gradient norm” result.
a. Use training dataset, set the number of discriminator layer to 4 (minimum requirement)
b. Plot two setting:
i. weight clipping
ii. gradient penalty
c. Y-axis: gradient norm(log scale), X-axis: discriminator layer number (from low to high)

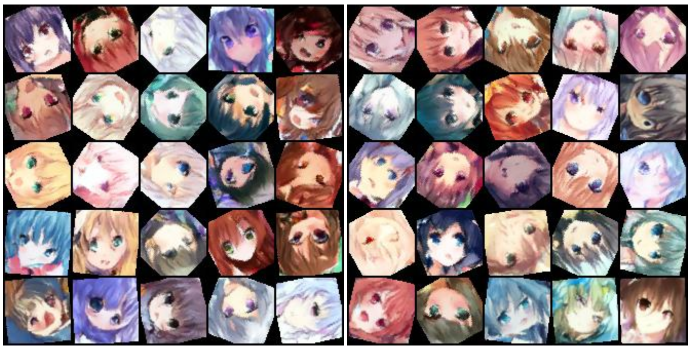


2023 HW06

Q1

Sample 5 images and show the progressive generation. Then, briefly describe their differences in different time steps.
Answer:





As the time steps increase, the generated images are getting better.

Q2

Canonical diffusion model (DDPM) is slow during inference, Denoising Diffusion Implicit Model (DDIM) is at least 10 times faster than DDPM during inference, and preserve the qualities. Please describe the differences of training, inference process, and the generated images of the two models respectively. Briefly explain why DDIM is faster.

1. Training Process
DDPM:

Noise Schedule: DDPMs are trained with a predefined noise schedule over many timesteps. During training, noise is gradually added to the data, and the model learns to reverse this process by denoising.

Gradual Denoising: The model is trained to perform gradual denoising over a large number of steps, typically several hundred to a thousand.

DDIM:

Noise Schedule: DDIMs use a similar noise schedule to DDPMs during training. However, the model is designed with inference efficiency in mind.

Shared Training Process: The training process of DDIM is almost identical to that of DDPM. The main differences come into play during the inference phase.

2. Inference Process

DDPM:

Sampling: Inference in DDPM involves starting from pure noise and iteratively denoising over many steps, typically several hundred to a thousand. Each step involves a stochastic process, where the model predicts noise, and a small amount of randomness is added back to the sample at each step.

Slow Process: This iterative denoising process, with its many steps and stochastic nature, makes the inference process slow.

DDIM:

Sampling: DDIM employs a non-Markovian process that allows for larger steps in the denoising process.

Reduced Steps: By using a deterministic approach and skipping intermediate noise addition steps, DDIM can achieve high-quality samples in significantly fewer steps, typically around 10-20 steps.

Efficiency: This results in a much faster inference process while still maintaining the quality of the generated images.

3. Generated Images
DDPM:

Quality: DDPMs are known for generating high-quality images because of their gradual and fine-tuned denoising process.

Consistency: The generated images are consistent with the training data distribution, thanks to the many small denoising steps.

DDIM:

Quality: Despite the reduced number of denoising steps, DDIMs can generate images of comparable quality to DDPMs. The deterministic nature of DDIM helps maintain the image quality.

Efficiency: The main advantage of DDIM lies in its ability to produce high-quality images much faster than DDPM.

4. Why DDIM is Faster
Deterministic Sampling: DDIM's deterministic approach eliminates the need for adding small amounts of noise back into the sample at each step. This allows the model to take larger steps towards denoising, reducing the total number of steps required.

Non-Markovian Process: DDIM employs a non-Markovian process that allows the model to "skip" intermediate steps without losing the ability to accurately denoise the sample.

Reduced Steps: By reducing the number of denoising steps from several hundred to just around 10-20, DDIM achieves a significant speedup in the inference process.

In summary, while DDPM focuses on a gradual and stochastic denoising process that ensures high-quality image generation, DDIM optimizes the inference phase to be deterministic and efficient, significantly speeding up the generation process while maintaining image quality.