1.Theoretical Justification

My proposed solution is based on the idea outlined in "Example 1: Accelerating DDPM with DIP-based Initial Priors".

DDPM is known for its remarkable ability to generate high-quality images by iteratively diffusing noise levels in a reverse process, which, unfortunately, can be computationally intensive. Conversely, DIP offers a fast forward process by leveraging the architecture of a CNN (Convolutional Neural Network) without requiring additional training data.

The proposed approach combines these strengths by first training a DIP model using the image to be restored as training data. By selecting an appropriate stopping point during training, we ensure that the DIP model captures the high-level structures and patterns present in the image without overfitting to the noise or spending excessive computational resources.

Subsequently, the trained DIP model's output serves as the starting point for the DDPM reverse diffusion process. By initializing DDPM with trained DIP model's output, we significantly reduce the number of steps needed for the reverse process. This is because the initialized image is already close to the target distribution, allowing DDPM to refine it efficiently.

Potential Benefits and Limitations:

Benefits:

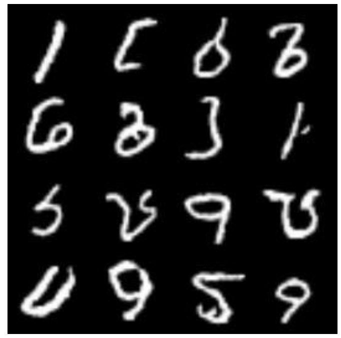
1.Improved Efficiency: By DIP-based initial prior, we expedite the reverse process of DDPM, significantly reducing computational overhead compared to starting from pure noise.

Limitations:

1.Dependency on Pre-trained CNN Quality: The effectiveness of the proposed approach relies heavily on the quality of the pre-trained CNN. If the CNN fails to capture relevant image features accurately, it may lead to suboptimal results in the DDPM reverse process.

2.Reduced Diversity in DDPM Reverse Process: The diversity in the DDPM reverse process is reduced because it starts from the trained DIP model’s output instead of pure noise.

2.Experimental Verification



1.Diversity: The left image shows the result obtained using DDPM+DIP, while the right image shows the result obtained using only DDPM.

DDPM+DIP: Because the trained DIP model's output serves as the starting point for the DDPM reverse diffusion process, the diversity is reduced, resulting in all pictures being number 3.

DDPM: Starting from pure noise allows for obtaining images of numbers 0-9.

2.Quality: From the images, both methods yield good quality results.

```
Starting optimization with ADAM
Sampling: 100%|
Sampling with DIP (Elapsed Time: 0.51 seconds)
Sampling: 100%|
Sampling without DIP (Elapsed Time: 15.03 seconds)
```

3. Generation Speed: From the top image, it can be seen that using the DDPM+DIP method takes approximately 0.5 seconds(the number of diffusion steps:20), while using only the DDPM method takes about 15 seconds(the number of diffusion step:1000), which is roughly 30 times longer.

3.Ablation Studies and Analysis

1.CNN Architecture: I experimented with various CNN architectures, aiming for the model to quickly capture sufficient image features while avoiding overfitting to noise.

The architecture I initially used was :

```
net1 = skip(
            input_depth, 3,
            num_channels_down = [8, 16, 32, 64, 128],
            num_channels_up   = [8, 16, 32, 64, 128],
            num_channels_skip = [0, 0, 0, 4, 4],
            upsample_mode='bilinear',
            need_sigmoid=True, need_bias=True, pad=pad, act_fun='LeakyReLU')
```

,but later I found that using a simpler architecture would be better.

```
net2 = skip(
            input_depth, 3,
            num_channels_down = [8, 16, 32],
            num_channels_up   = [8, 16, 32],
            num_channels_skip = [0, 0, 0],
            upsample_mode='bilinear',
            need_sigmoid=True, need_bias=True, pad=pad, act_fun='LeakyReLU')
```

This is because the images I chose are very simple (with fewer high-frequency components), so a complex CNN is not necessary. Using a simpler CNN can speed up convergence and more easily avoid overfitting.

Picture 1 Picture 2 Picture 3 Picture 4

Picture 1 shows an image restored by net1 after 50 epochs. Picture 2 shows the result of applying the DDPM+DIP method to this image.

Picture 3 shows an image restored by net2 after 50 epochs. Picture 4 shows the result of applying the DDPM+DIP method to this image.

It can be observed that net1 likely needs more epochs to converge, while net2 has successfully captured the high-level structures and patterns present in the image. Therefore, using net2 can be more efficient.

2.Noise Schedule

In the context of Improved Denoising Diffusion Probabilistic Models (reference: https://arxiv.org/abs/2102.09672), it is noted that the Linear Schedule performs poorly with low-resolution images compared to the Cosine Schedule. My own testing on the MNIST dataset confirms this claim. Therefore, I have ultimately chosen to use the Cosine Schedule.

3. Noise Levels

Reconstruction Quality:

Low Noise Level: DIP typically produces higher-quality images when the input noise level is low. Less noise means the network can concentrate on reconstructing the image details more accurately.

High Noise Level: At higher noise levels, the quality of the images reconstructed by DIP may decrease. The noise can obscure the original image details, making it challenging for the network to distinguish between the true signal and the noise, thus affecting the reconstruction quality.

Picture 5

Picture 6

Picture 7

Picture 8

Picture 5 shows an image with noise added using np.random.normal(scale=25/255, size=img_np.shape). The result of applying DDPM+DIP method to this image is shown in Picture 6.

Picture 7 shows an image with noise added using np.random.normal(scale=100/255, size=img_np.shape). The result of applying DDPM+DIP method to this image is shown in Picture 8.

It can be observed that when the input image has a lower noise level, the DDPM+DIP method produces better results.