Generating Images with Denoising diffusion probabilistic model

Anonymous author

Abstract

Denoising Diffusion Probabilistic Models have been widely applied in various applications. This paper focuses on using the model to learn and generate 48x48 pixels church images based on the LSUN dataset. The forward and backward processes are described in detail through a set of equations and diagrams. The results section showcases the high-quality and diverse images generated by the trained model, which are then evaluated using the lpips matrix to select the best samples. The limitations of the model are also discussed, covering aspects such as hardware requirements and comparisons with other models.

1 Methodology

The Denoising Diffusion Probabilistic Models[2] are used to generate the image in this paper. In the model training, there are two sections, including the forward process and backward process, shown as the figure1. The forward process represents the noise addition for the image. The gaussian noise is added to the original image repeatedly until it transforms into a noisy image during this process.

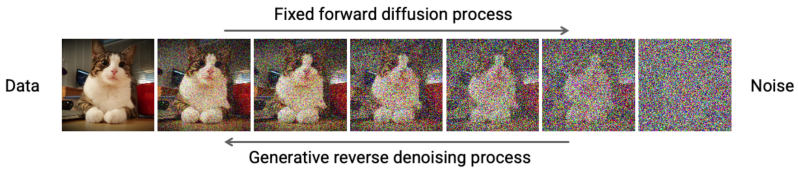

Figure 1: Diffusion model[3]

The equation 1 - equation 3 describe the forward process. The ϵ represents noise, the α_t, $1-\alpha_t(\beta)$ represent the weight of the image and the noise. The equation 1 correspond the every steps of noise addition, where x_t is the noise-added image after t-step noise addition. By repeating the equation 1, the original image x_0 will gradually become to the noise-added image x_t. Because the whole process is a cumulative multiplication, it can be reduced to the equation 2 so that it allows the image x_t at time t can be calculated based on the image x_0 directly. Therefore, the main formula of the forward process is shown as the equation 3.

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon_1 \tag{1}$$

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t \tag{2}$$

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{a}_t}x_0, (1-\bar{a}_t)I) \tag{3}$$

To implement the reversion of the noise image, the noise distribution of every step is the key. Assume the noise distribution of step t is known, and the condition is x_0. The distribution in x_{t-1} can be expressed as the left part of equation 4. Then, use Bayes's rule to get the right part of equation 4.

$$q(x_{t-1}|x_t, x_0) = q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} \tag{4}$$

Hence, there are three new expressions, $q(x_t|x_{t-1})$, $q(x_{t-1}|x_0)$ and $q(x_t|x_0)$. According to the equation 3, these three expressions can be expressed as the equation 5 - equation 7.

$$q(x_t|x_{t-1}, x_0) \to \sqrt{a_t}x_{t-1} + \sqrt{1-a_t}\epsilon \sim \mathcal{N}(\sqrt{a_t}x_{t-1}, 1-a_t) \tag{5}$$

$$q(x_{t-1}|x_0) \to \sqrt{\bar{a}_{t-1}}x_0 + \sqrt{1-\bar{a}_{t-1}}\epsilon \sim \mathcal{N}(\sqrt{\bar{a}_{t-1}}x_0, 1-\bar{a}_{t-1}) \tag{6}$$

$$q(x_t|x_0) \to \sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon \sim \mathcal{N}(\sqrt{\bar{a}_t}x_0, 1-\bar{a}_t) \tag{7}$$

These three expressions are then substituted into the equation 8 separately and can be summarised to the equation 9 according to equation 4. By simplifying, this leads to equation 10. Then mean μ and variance σ^2 can extract from the equation 10 according to the expansion of equation 8.

$$\mathcal{N}(\mu, \sigma^2) \propto exp(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}) \tag{8}$$

$$\propto exp(-\frac{1}{2}(\frac{(x_t - \sqrt{a_t}x_{t-1})^2}{1-a_t} + \frac{(x_{t-1} - \sqrt{\bar{a}_{t-1}}x_0)^2}{1-\bar{a}_{t-1}} - \frac{(x_t - \sqrt{\bar{a}_t}x_0)^2}{1-\bar{a}_t})) \tag{9}$$

$$= exp(-\frac{1}{2}((\frac{a_t}{1-a_t} + \frac{1}{1-\bar{a}_{t-1}})x_{t-1}^2 - (\frac{2\sqrt{a_t}}{1-a}x_t + \frac{2\sqrt{\bar{a}_{t-1}}}{1-\bar{a}_{t-1}}x_0)x_{t-1} + C(x_t, x_0))) \tag{10}$$

The mean μ_t expression after simplification is shown as equation 11. It indicates that the mean μ_t is related to the x_t and x_0. After that, use equation 2 to replace the x_0 and the new expression of mean μ_t is shown as equation 12.

$$\mu_t = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})}{1-\bar{a}_t}x_t + \frac{\sqrt{\bar{a}_{t-1}}(1-a_t)}{1-\bar{a}_t}x_0 \tag{11}$$

$$\mu_t = \frac{1}{\sqrt{a_t}}(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}}\epsilon_t) \tag{12}$$

To predict the mean μ_t, the neural network is introduced to learn the generation of the noise matching μ_t. And the mean of the noise can be expressed as the equation 13.

$$\mu_\theta = \frac{1}{\sqrt{a_t}}(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}}\epsilon_\theta) \tag{13}$$

In the training process, the role of the network is learned to minimize the difference between μ_t and μ_θ to generate more realistic noise. Therefore, the simplified loss function can be expressed as equation 14.

$$L_t = \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}[||\epsilon_t - \epsilon_\theta(x_t, t)||^2] \tag{14}$$

where ϵ_t and $\epsilon_\theta(x_t, t)$ are the actual noise and predicted noise.
The figure 2 describes the training process of the model θ in order to learn to generate the approximate noise. Once the model training is completed, apply the trained model to sample the image from random noise. In the sampling process, the predicted noise will be subtracted from the noise image in every step to implement the denoising of the image, that is, the generation of the image (Figure 2).

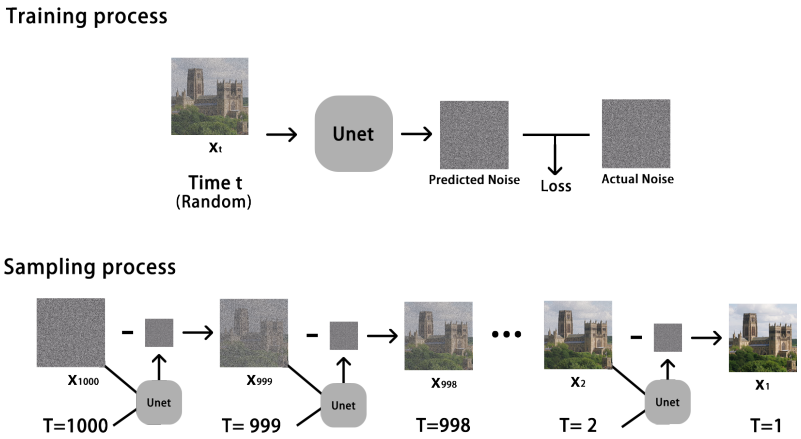

Figure 2: Training and sampling processes

2 Results

The model[8] is trained on the church class of the LSUN dataset[6] at 48*48 pixels. The results of the model training 150 epochs are shown in figure 3. The samples show churches in different times and weather, which indicates well diversified. In addition, apply linear interpolation to the input noise and perform sampling. All midpoints (Figure 4) have realistic shape and correct feature of the church. These both represent a great performance of the model on the image generation for the church class. For selecting the best samples from similar outputs, the lpips package[7] is applied to calculate the perceptual loss between 64 samples (Figure 3) and the nearest neighbours. And six samples with the lowest perceptual loss are shown in the figure 5, which represents the best set of the samples. Their perceptual losses are between 0.29 and 0.32, which approximate the perceptual losses between the different batch datasets. It indicates that the generated images are comparable to that of the original dataset and are diverse.

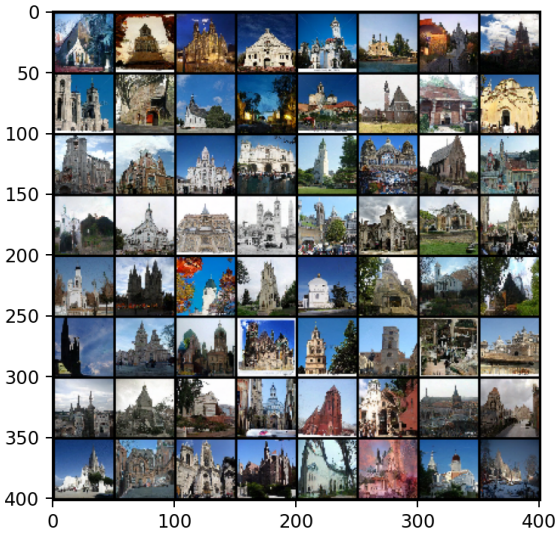

Figure 3: Generated samples

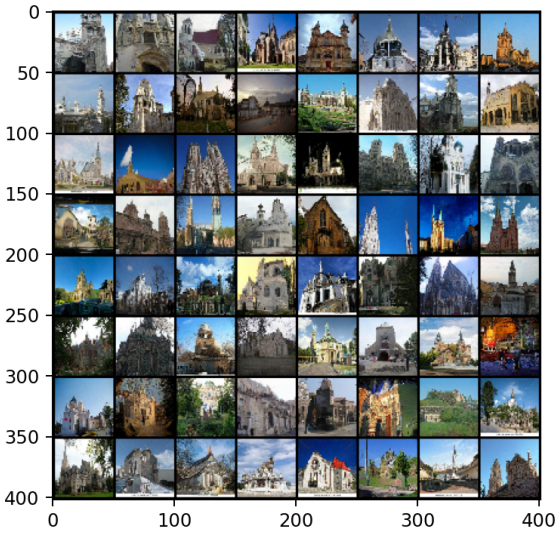

Figure 4: Interpolation samples

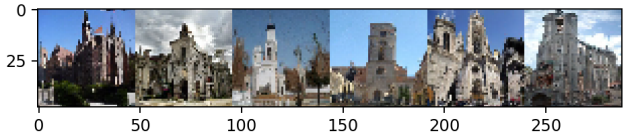

Figure 5: Selected samples

3 Limitations

The quality of the image is quite close to the actual image, and the features of the image are visible. However, training for higher-resolution images requires more GPU memory. The limitation of the configuration results in the generated image size being 48*48 pixels, much blurrier than the normal image. Moreover, the sample time cost for every 64 samples is about 8 mins in the RTX3070 laptop, which is relatively longer than the time cost of the GAN[1].
In the future, the source code will be tested on higher configuration computers to generate the image in higher resolution, such as 64*64 or 128*128. For increasing the slow sample speed, improved diffusion models will be applied such as stable diffusion model[4] and denoising diffusion implicit model[5]. Furthermore, the training in the paper is based on a single-class dataset, which makes the model only generate the image in one class. Nextly, the muti-class dataset will be used to test the DDPM model and compare the result with the single-class dataset.

Bonuses

This submission has a total bonus of +2 marks, as it is trained with LSUN dataset resized to 48x48 pixels.

References

[1] Ian Goodfellow et al. "Generative Adversarial Networks". In: *Commun. ACM* 63.11 (Oct. 2020), pp. 139–144. ISSN: 0001-0782. DOI: 10.1145/3422622. URL: https://doi.org/10.1145/3422622.

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[3] NVIDIA. *Improving Diffusion Models as an Alternative to GANs, Part 1*. 2021. URL: https://developer.nvidia.com/blog/improving-diffusion-models-as-an-alternative-to-gans-part-1/ (visited on 02/13/2023).

[4] Robin Rombach et al. "High-Resolution Image Synthesis with Latent Diffusion Models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022. URL: https://github.com/CompVis/latent-diffusionhttps://arxiv.org/abs/2112.10752.

[5] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models". In: *arXiv preprint arXiv:2010.02502* (2020).

[6] Fisher Yu et al. "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop". In: *arXiv preprint arXiv:1506.03365* (2015).

[7] Richard Zhang et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *CVPR*. 2018.

[8] zoubohao. *DenoisingDiffusionProbabilityModel-ddpm-*. 2022. URL: https://github.com/zoubohao/DenoisingDiffusionProbabilityModel-ddpm- (visited on 02/13/2023).