Denoising Diffusion Probabilistic

Abstract—This report discusses briefly about the mathematics behind denoising diffusion models and their implementations. LSUN dataset set is used to train the model and the results are discussed by displaying non cherry picked images, cherry picked images, their average perceptual similaries and their interpolations from the nearest neighbour. The limitations of this project is discussed.

1 Introduction

A diffusion probabilistic model (also known as "diffusion model") is variation of latent models that consists of a parameterized Markov chain with learned Gaussian transitions trained using variational inference to produce samples matching the data after finite time[4]. A diffusion model consists of a noise scheduler which is the forward process, a loss function which is used to train the model through back propagation, the reverse process and finally the sampling strategy.

2 Methodology

2.1 Forward Process/ Diffusion process

The forward process(q) or diffusion process, is fixed to a Markov chain that gradually adds Gaussian noise to the data(x) according to a variance schedule β1, . . . , βT. This is defined as posterior function:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

which can also e represented as :

$$\alpha_t := 1-\beta_t \text{ and } \bar{\alpha}_t := \prod_{s=0}^{t} \alpha_s,$$

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

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$$

where ∼ N (0, I). Here, 1 − $\bar{\alpha}$t tells us the variance of the noise for an arbitrary timestep, and this could equivalently be used to define the noise schedule instead of βt. The posterior now can be defined as[4]:

$$\tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}(x_t, x_0), \tilde{\beta}_t\mathbf{I})$$

Training is performed by optimizing the usual variational bound on negative log likelihood. Through variance reduction the negative log likelihood can be rewritten as KL divergence which compared the difference between two gaussian distributions which is represented as follows[4]:

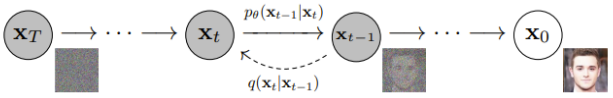
$$L_{\text{vlb}} := L_0 + L_1 + ... + L_{T-1} + L_T$$

$$L_0 := -\log p_\theta(x_0|x_1)$$

$$L_{t-1} := D_{KL}(q(x_{t-1}|x_t, x_0) \,||\, p_\theta(x_{t-1}|x_t))$$

$$L_T := D_{KL}(q(x_T|x_0) \,||\, p(x_T))$$

This equation uses KL divergence to compare forward posteriors, q(xt|xt-1) and p0(xt-1|xt). This is shown in the diagram below.



2.2 Reverse process

The reverse process is the joint distribution of latent variable models where the latent are of the same dimensionality of the data. The reverse process is defined as a Markov chain with learned Gaussian transition starting at when t = T.

$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}):$$

to when 0≤ t <T

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

2.3 Implementation of diffusion model

The diffusion model is implemented by the two algorithms; training and sampling

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

.

2.4 Training Algorithm 1

The dataset used to train these images is LSUN 96 x 96 pixels which consists of 10 categories, airplane, bird, car,

cat, deer, dog, horse, monkey, ship, and truck.
The training process is implemented by through algorithm 1. Line 2 describes obtaining a sample from a dataloader, and adding some Gaussian noise according to the variance schedule B1.....BT based on a certain time-step chosen uniformly at random. This noisy sample is then fed into the UNET architecture and a gradient descent step is taken until it converges. The gradient descent step is taken on line 5 in Algorithm 1. This represents the loss function between the noise generated and noise predicted.
The reverse process can be trained on the mean function approximator μθ to predict μ˜t or by modifying its parameterization, it can train it to predict e[4]. It is found that predicting e worked best, especially when combined with a reweighted loss function which can be represented in a simpler form of:

$$L_{\text{simple}} = E_{t,x_0,\epsilon}\left[||\epsilon - \epsilon_\theta(x_t, t)||^2\right]$$

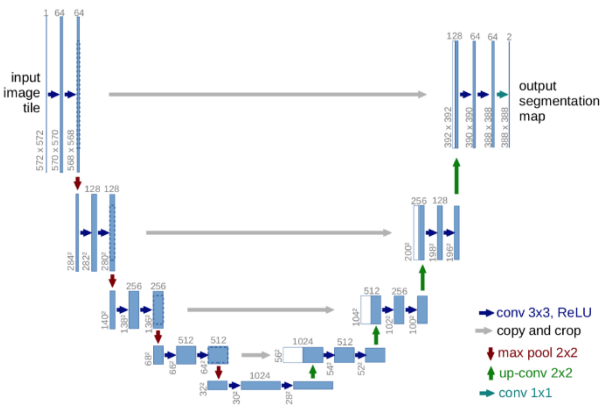
This loss function is derived from the equations in the forward process and can be seen as a reweighted form of Lvlb (without the terms affecting Σθ). The authors found that optimizing Lsimple resulted in much better sample quality than optimizing Lvlb directly.

2.5 Sampling Algorithm 2

Algorithm 2, resembles Langevin dynamics with eθ as a learned gradient of the data density. Langevin algorithms are gradient descent methods augmented with additive noise, and are widely used in Markov Chain Monte Carlo (MCMC) sampling, optimization, and learning[5]..

3 UNET Architecture

The architecture implemented is the UNET architecture. UNET is a U-shaped encoder-decoder network architecture, which consists of four encoder blocks and four decoder blocks that are connected via a bridge. The encoder network (contracting path) half the spatial dimensions and double the number of filters (feature channels) at each encoder block. Likewise, the decoder network doubles the spatial dimensions and half the number of feature channels.



The encoder network extracts feature and learn abstract representation of the input images. Each encoder block consists of two 3x3 convolutions, where each convolution is followed by a ReLU (Rectified Linear Unit) activation function. The output of the ReLU acts as a skip connection for the corresponding decoder block. These skip connections provide additional information that helps the decoder to generate better semantic features. They also act as a shortcut connection that helps the indirect flow of gradients to the earlier layers without any degradation. The bridge connects the encoder and the decoder network and completes the flow of information. The decoder network is used to take the abstract representation and generate a semantic segmentation mask. The decoder block starts with a 2x2 transpose convolution. Next, it is concatenated with the corresponding skip connection feature map from the encoder block. These skip connections provide features from earlier layers that are sometimes lost due to the depth of the network. After that, two 3x3 convolutions are used, where each convolution is followed by a ReLU activation function. The output of the last decoder passes through a 1x1 convolution with sigmoid activation. The sigmoid activation function gives the segmentation mask representing the pixel-wise classification[6].

At different timestep, in the forward process, the intensity of noise generated changes. The larger the timestep, the higher the intensity. Therefore, sinusoidal positional embedding are implemented in the UNET, so the UNET can predict the right intensity of noise. This is done by encoding a timestep to each block in the UNET[7].

4 Results

4.1 Perceptual Similarity

Example scripts to take the distance of all pairs of images within a directory. Two directory of images are evaluated, non cherry picked images and cherry picked images, and their perceptual similarity values are seen in the table below. Perceptual similarity measures valuate the distance between image patches. The higher the value means further or more different the images are in comparison to each other. Lower the value means more similar the images are with each other.

Directory	Average Perceptual Similarity
Non cherry picked 64 images	0.536516 +/- 0.015696
Cherry picked 20 images	0.67859 +/- 0.03687

4.2 Non cherry picked image

See appendix 1

4.3 Cherry picked images

See appendix 2

4.3 Interpolation

5 Limitation

The limitations are that the model would have performed better if attention blocks were used to be able to extract better features of the images. The images produced are most greyish in colour, the contrast between the colours are not very apparent. The images produces are also blurry. The cherry picked images are somewhat similar to the trained dataset. In the future I would like to expand into VQGans. VQGans is a GAN architecture which can be used to learn and generate novel images based on previously seen data. These models are faster to train in comparison to Denoising Diffusion Models[8].

6 Bonus

Bonus = 2 + 2 + 2 = 6 for using LSUN 96 x 96 pixels

References

[1] Nichol, A. and Dhariwal, P. (2021). Improved Denoising Diffusion Probabilistic Models. arXiv:2102.09672 [cs, stat]. [online] Available at: https://arxiv.org/abs/2102.09672#:~:text=Improved%20Denoising%20Diffusion-sion%20Probabilistic%20Models%20Alex%20Nichol%2C%20Prafulla [Accessed 30 Jan. 2023].W.-K. Chen, *Linear Networks and Systems.* Belmont, Calif.: Wadsworth, pp. 123-135, 1993. (Book style)

[2] Dhariwal, P., Nichol, A. and Openai (n.d.). Diffusion Models Beat GANs on Image Synthesis. [online] Available at: https://arxiv.org/pdf/2105.05233.pdf.K. Elissa, “An Overview of Decision Theory," unpublished. (Unpublished manuscript)

[3] Song, J., Meng, C. and Ermon, S. (2022). Denoising Diffusion Implicit Models. arXiv:2010.02502 [cs]. [online] Available at: https://arxiv.org/abs/2010.02502.C. J. Kaufman, Rocky Mountain Research Laboratories, Boulder, Colo., personal communication, 1992. (Personal communication)

[4] Ho, J., Jain, A. and Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. arXiv:2006.11239 [cs, stat]. [online] Available at: https://arxiv.org/abs/2006.11239.S.P. Bingulac, “On the Compatibility of Adaptive Controllers,” *Proc. Fourth Ann. Allerton Conf. Circuits and Systems Theory*, pp. 8-16, 1994. (Conference proceedings)

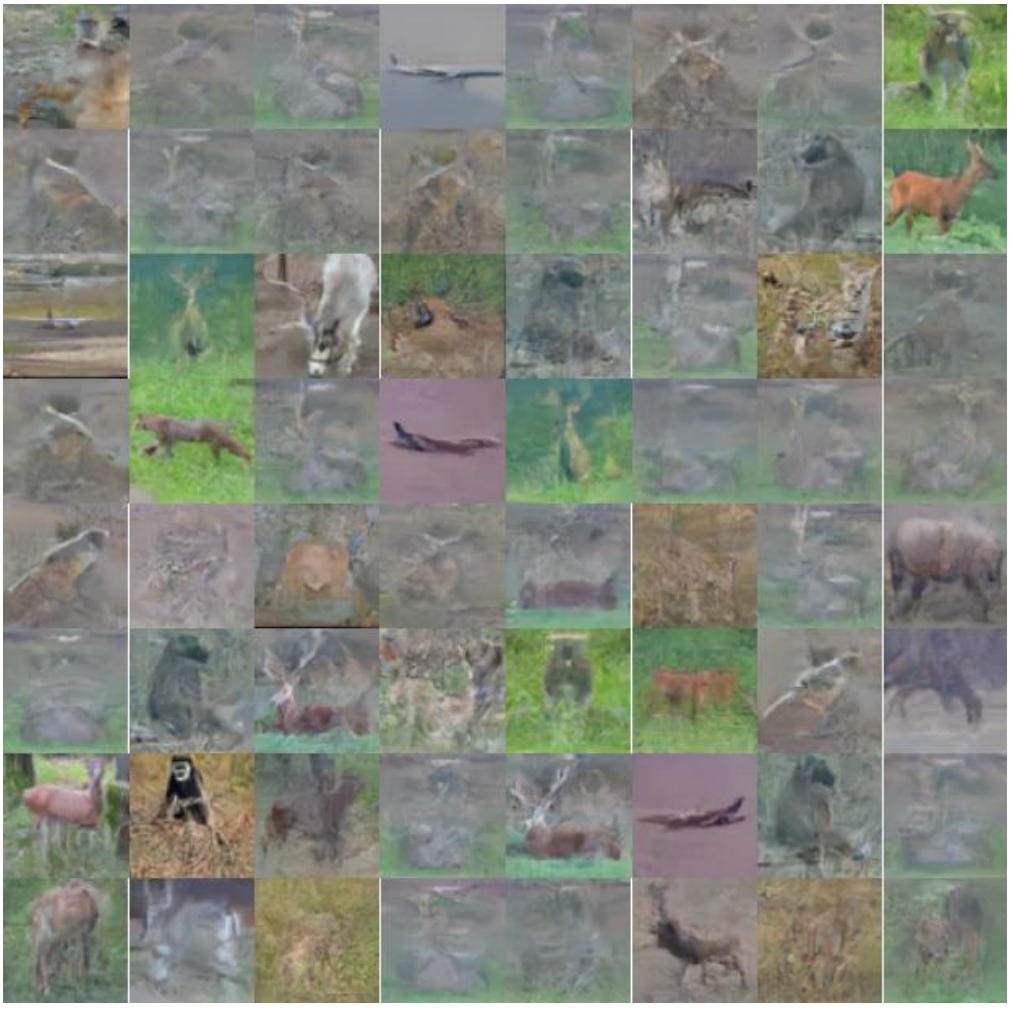
[5] Zheng, Y. and Lamperski, A. (2023). Constrained Langevin Algorithms with L-mixing External Random Variables. arXiv:2205.14192 [cs, math]. [online] Available at: https://arxiv.org/abs/2205.14192#:~:text=Langevin%20algorithms%20are%20gradient%20descent%20methods%20augmented%20with [Accessed 7 Feb. 2023].

[6] Tomar, N. (2021). What is UNET? [online] Analytics Vidhya. Available at: https://medium.com/analytics-vidhya/what-is-unet-157314c87634.

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, Aidan N, Kaiser, L. and Polosukhin, I. (2017). Attention Is All You Need. [online] arXiv.org. Available at: https://arxiv.org/abs/1706.03762.

[8] compvis.github.io. (n.d.). Taming Transformers for High-Resolution Image Synthesis. [online] Available at: https://compvis.github.io/taming-transformers/ [Accessed 15 Feb. 2023].

Appendix 1



Appendix 2