Diffusion Model

Yinjie Wang

June 10, 2023

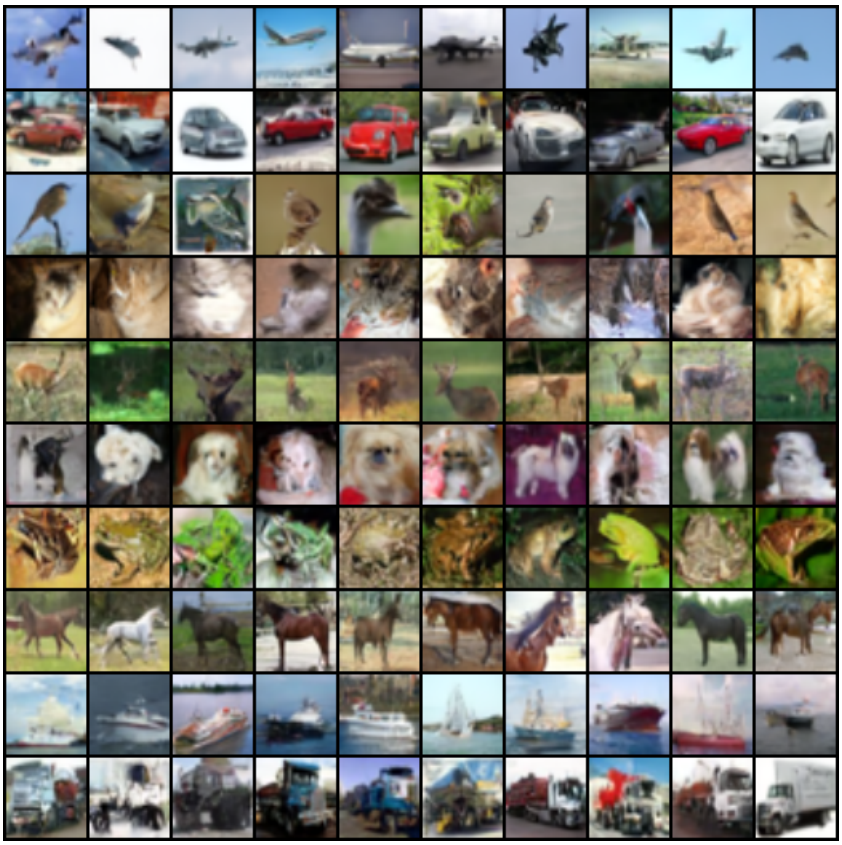


Figure 1: These CIFAR-10 images are generated by classifier-free guidance diffusion model, which requires 81.5 hours of training.

1 Denoising diffusion probabilistic model

1.1 Introduction of DDPM

The diffusion model is well-known for its state-of-the-art sampling ability. In this section, we introduce the first diffusion model that can generate high-quality images: the denoising

diffusion probabilistic model (DDPM). It is also the foundation for all other diffusion models.

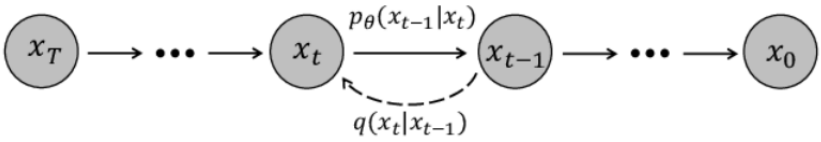


Figure 2: Denoising diffusion probabilistic model

Given an image x_0, DDPM first adds noise to transform x_0 to a Gaussian noise x_T, then learns how to denoise this noise (figure 2). Specifically, the forward process (diffusion process) of DDPM is a Markov chain that gradually adds Gaussian noise to the data x_0:

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1}), \tag{1}$$

where $q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$, and $0 < \beta_t < 1$ $(1 \le t \le T)$ are hyperparameters. It is straightforward to derive from equation 1 that $q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\epsilon)$, where $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$ and $\alpha_t := 1 - \beta_t$. Consequently we can express x_t as $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$, $\epsilon_t \sim \mathcal{N}(0, I)$.

The key aspect of DDPM is denoising, which reverses the noise x_T back to x_0. The denoising process (reverse process) is also a Markov chain:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t), \tag{2}$$

where $p(x_T) = \mathcal{N}(x_T; 0, I)$. We use $q(x_{t-1} \mid x_t, x_0)$ to estimate $p_\theta(x_{t-1} \mid x_t)$. By Bayes' rule, we have $q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I)$, where

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \tag{3}$$

$$\tilde{\beta}_t := \sigma_t^2 := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t, \tag{4}$$

as we will show in the derivations provided in appendix. In order to estimate $p_\theta(x_{t-1} \mid x_t)$ using $q(x_{t-1} \mid x_t, x_0)$, we need to estimate x_0 from x_t. Based on $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$, we only need to estimate ϵ_t using neural network $\epsilon_\theta(x_t, t)$. Therefore we have $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}[x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)]$, and

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}[x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)], \sigma_t^2 I). \tag{5}$$

Similar to VAE, our objective here is to minimize the ELBO loss:

$$\begin{aligned} Loss_{ELBO} &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right] \\ &\equiv \sum_{t=1}^{T} \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}||\epsilon_t - \epsilon(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)||^2, \end{aligned} \tag{6}$$

where "$\equiv$" indicates that the expressions are equivalent when optimizing $\epsilon_\theta(x_t, t)$. We will show the derivations in appendix.

I believe there is a potential reason why VAEs don't have comparable sampling ability to diffusion model. Recall that in VAE's loss function, we have the KL loss $\mathrm{KL}\left[q(z \mid x; \phi)||p(z)\right]$. For different x, $q(z \mid x; \phi)$ varies, especially when the reconstruction loss is dominates(causing $q(z \mid x; \phi) \to 1$ only when z is the latent variable corresponding to x). Consequently, it's challenging to adequately minimize $\mathrm{KL}\left[q(z \mid x; \phi)||p(z)\right]$ for every x in a VAE. However, in diffusion models, there is no such problem. The corresponding term in loss function, $\mathrm{KL}\left[q(x_T \mid x_0)||p(x_T)\right]$, can be adequately minimized since $x_T = \sqrt{\bar{\alpha}_T}x_0 + \sqrt{1-\bar{\alpha}_T}\epsilon_T \to \mathcal{N}(0, I)$ as $T \to \infty$.

It has been found that using the following simple loss function can generate better images than the ELBO loss in eqaution 6.

$$Loss_{simple} = \mathbb{E}_{t,x_0,\epsilon}\left[||\epsilon - \epsilon_\theta(x_t, t)||^2\right] \tag{7}$$

This simple loss function just ignores the weight $\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}$ in equation 6. However, doing so will make the log-likelihood not comparable to other log-likelihood-based models. The improved DDPM proposed to use $Loss_{simple} + \lambda Loss_{ELBO}$ as the loss function to solve this problem.

1.2 Some implementation details

Now we present the training and sampling algorithms in figure 3. Based on the original paper, we set T to 1000, which means we need to train for at least 1000 epochs. In practice, we train for 2500 epochs in total, which takes several days with A100. The parameters β_t are all chosen based on original paper.

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

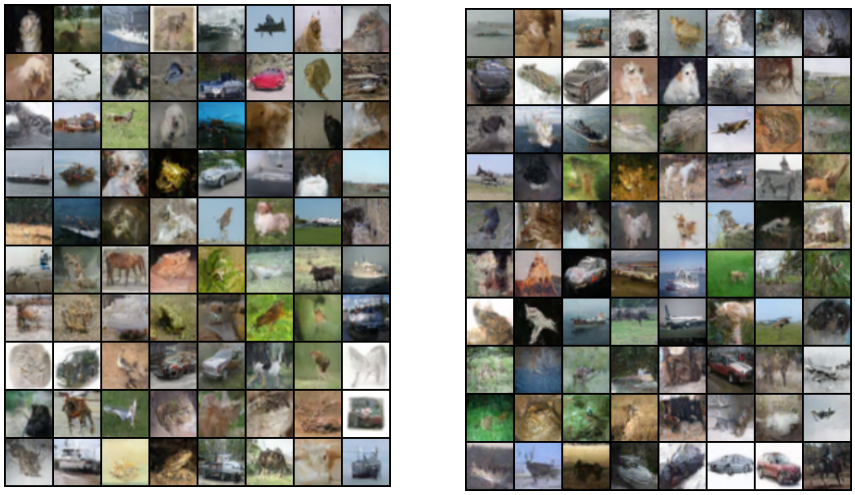
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Figure 3: training and sampling algorithms

The network structure used for $\epsilon_\theta(x_t, t)$ is U-Net, a type of network commonly used in denoising tasks. We use 2 residual blocks in each layer and perform downsampling (upsampling) 4 times in U-Net. As for the timestamp t, we generate embeddings for each of them and add them into the residual blocks.

1.3 results

In VAE experiments, we were unable to generate recognizable images of CIFAR-10 dataset using an 8-layer network structure. Therefore, we shifted our focus to CIFAR-10 images to evaluate the performance of diffusion models. After training 2500 epochs (takes 70.6 hours), we get the following generated samples (figure 4):



$\sigma_t^2 = \tilde{\beta}_t$ $\sigma_t^2 = \beta_t$

Figure 4: Generated samples by DDPM

We found a significant improvement compared to the previous VQVAE results (figure 10). The DDPM proposes two options for the variance: $\sigma_t^2 = \tilde{\beta}_t$ and $\sigma_t^2 = \beta_t$. Our experiments here indicates there is no obvious difference in their performance.

However, the recognition of the generated images is not good enough. Therefore, we employ conditioning to improve sampling quality in the following experiments.

2 Conditional diffusion model

As we have observed in VAE's experiments, adding conditions can improve the sampling quality. Therefore we use the conditional diffusion model to aid us in generating clearer images. In the following context, we will introduce two conditional diffusion models we have implemented: the conditional diffusion model and classifier-free guidance diffusion model. There are some other conditional techniques have proven to be helpful. For instance, the cascaded diffusion model utilizes the generated low-resolution images as conditions to generate high-resolution images.

2.1 Incorporate labels

We have labels c for images of CIFAR-10, and only need to modify $\epsilon_\theta(x_t, t)$ to $\epsilon_\theta(x_t, t, c)$ to achieve conditional models. For the ten different labels in CIFAR-10, we use the same trick of timestamp: generate embeddings for each label and adding them in the residual blocks.

2.2 Classifier-free guidance diffusion model

We first introduce the classifier guidance model:

$$\tilde{p}_\theta(x_t \mid c) \propto p_\theta(x_t \mid c) p_\theta(c \mid x_t)^w, \tag{8}$$

where $w > 0$ and $p_\theta(c \mid x_t)$ is a pre-trained classifier. The objective of this is to up-weight probability of data that can be classified well, which will result in a higher inception score. However, this comes at the expense of decreased diversity.

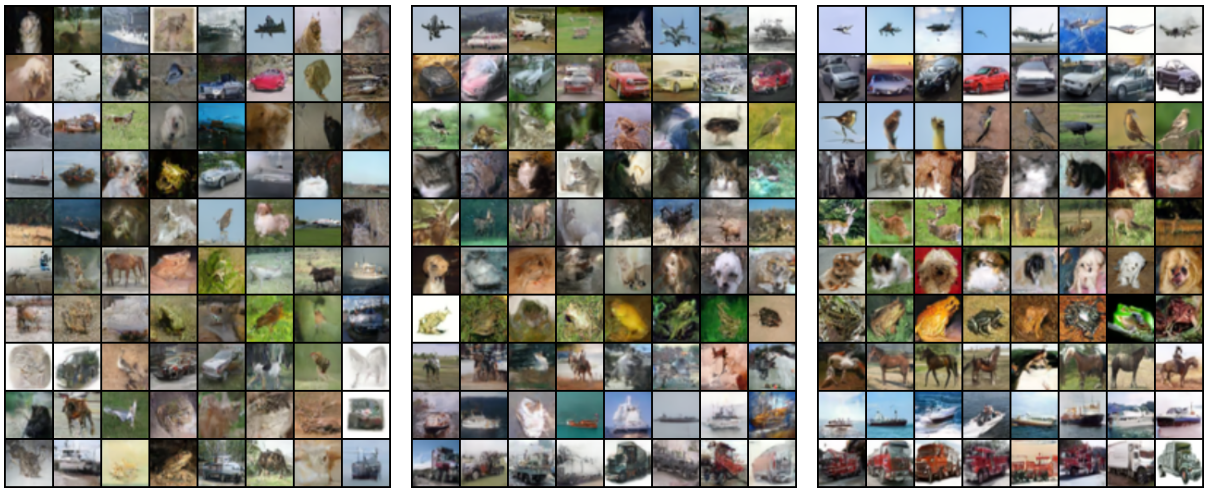
Additionally, we need to pre-train a classifier and obtain it's gradients during training, which is time-consuming. Therefore the classifier-free guidance model is proposed. Here is a brief derivation for it. Note that $p_\theta(c \mid x_t) \propto p_\theta(x_t \mid c)/p_\theta(x_t)$ and $\nabla_{x_t} \log p_\theta(c \mid x_t) = -\frac{1}{\sigma_t}\left[\epsilon_\theta(x_t, c) - \epsilon_\theta(x_t)\right]$. Thus we have our score estimator:

$$\tilde{\epsilon}_\theta(x_t, c) = (1 + w)\epsilon_\theta(x_t, c) - w\epsilon_\theta(x_t). \tag{9}$$

In the implementations, we follow the original paper by training $\epsilon_\theta(x_t)$ and $\epsilon_\theta(x_t, c)$ simultaneously, providing $c = 0$ with a probability of 0.1.

2.3 Benchmark for these models

In this subsection, we benchmark the performance of the three different diffusion models we have introduced: DDPM, conditional DDPM, and the classifier-free guidance diffusion model (figure 5).



DDPM Conditional DDPM Classifier-free guidance

Figure 5: Generated samples of different models

We found that the classifier-free guidance diffusion model can sample the clearest and most recognizable images among the three models.

We also evaluated their performance based on Frechet Inception Distance (FID) and Inception Score (IS). Specifically, we generated 30,000 images in each setting, which took approximately 20 hours to sample. Table 1 indicates that classifier-free guidance model achieved the best performance among all.

	DDPM	Conditional DDPM	Classifier-free guidance
FID	44.81	25.45	**14.01**
IS	7.08	7.27	**8.83**

Table 1: Performance of different models (evaluated by FID and IS)

3 Denoising diffusion implicit model

3.1 Introduction and derivation

The forward and reverse processes of proposed Denoising diffusion implicit model (DDIM) have been modified to enable training the model in the same manner as DDPM, while allowing for the selection of different variances during sampling.

Note that we only utilized $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ and $q(x_{t-1} \mid x_t, x_0)$ (estimate $p_\theta(x_{t-1} \mid x_t)$) derived from forward process, instead of the exact form of $q(x_{1:T} \mid x_0)$. Thus, DDIM sets the forward process to be:

$$q_\sigma(x_{1:T} \mid x_0) = q_\sigma(x_T \mid x_0) \prod_{t=2}^{T} q_\sigma(x_{t-1} \mid x_t, x_0), \tag{10}$$

$$\text{where} \quad q_\sigma(x_T \mid x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_T}x_0, (1-\bar{\alpha}_T)I), \tag{11}$$

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}}, \sigma_t^2 I). \tag{12}$$

We can employ mathematical induction method to prove that the forward process defined by equations 10, 11 and 12 can ensure that $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ holds for $1 \leq 1 \leq T$. But this is the proof after the method is derived, now let's provide a short derivation for $q_\sigma(x_{t-1} \mid x_t, x_0)$: Setting $x_{t-1} = ax_0 + bx_t + \sigma_t^2\epsilon$, we can solve for a and b given $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ and $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_{t-1}$:

$$a = \sqrt{\bar{\alpha}_{t-1}} - \sqrt{\bar{\alpha}_t}\sqrt{\frac{1-\bar{\alpha}_{t-1}-\sigma_t^2}{1-\bar{\alpha}_t}}, \quad b = \sqrt{\frac{1-\bar{\alpha}_{t-1}-\sigma_t^2}{1-\bar{\alpha}_t}}. \tag{13}$$

The generative process is estimated using the forward process:

$$p_\theta(x_{0:t}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t), \tag{14}$$

$$\text{where} \quad p_\theta(x_{t-1} \mid x_t) = q_\sigma(x_{t-1} \mid x_t, \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t))). \tag{15}$$

We state that the simple loss function $Loss_{simple}$ does not change. Because $\mathrm{KL}(q_\sigma(x_{t-1} \mid x_t, x_0)||p_\theta(x_{t-1} \mid x_t)) = C\,\mathbb{E}_{\epsilon_t \sim \mathcal{N}(0,I)}\left[||\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)||^2\right]$, and C is ignored in the simple loss function. Under this framework, we can train DDIM in the same way as DDPM, while sampling images using different chosen variances.

However, we should note that C contains term $\frac{1}{\sigma_t^2}$. When $\sigma_t = 0$ (DDIM) , the ELBO loss is not well-defined.

3.2 Accelerate sampling

One of the most troublesome problems of diffusion model is its sampling time. We need to sample T times to generate the images, which is very time-consuming. Therefore, DDIM proposes an accelerated sampling method.

We choose s+1 timestampe: $T = \tau_s > \tau_{s-1} > \cdots > \tau_0 = 0$ for sampling. Specifically,

$$x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}}{\bar{\alpha}_{\tau_i}}} x_{\tau_i} + \left(\sqrt{1 - \bar{\alpha}_{\tau_{i-1}} - \sigma_{\tau_i}^2} - \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}(1 - \bar{\alpha}_{\tau_i})}{\bar{\alpha}_{\tau_i}}} \epsilon_\theta(x_{\tau_i}, \tau_i) \right) + \sigma_{\tau_i} \epsilon_{\tau_i}. \tag{16}$$

This model still shares the same simple loss function and training process. The corresponding forward and reverse processes, along with the derivations, are in the appendix. In this way, we only need to sample s times to get the images.

However, when s is small, the generated images tend to be blurry. DDIM addresses this problem by setting $\sigma_t = 0$, which eliminates the variance and results in better sample quality. We will evaluate the effect of variance σ_t^2 and sampling step s on sampling quality afterward.

There are other methods to accelerate the sampling process.. For example, progressive distillation teaches pre-trained DDIM to replace 2 steps by only 1 step during the sampling process, making DDIM sampling faster.

3.3 Effect of sampling step in DIMM

We evaluate the effect of different sampling step s here. In this experiment, we use the same x_T to sample with different number of steps. Note that when $\sigma_t = 0$, the sampling precedure is deterministic, therefore, using the same x_T should make the generated images similar and easier to compare.

$s = 10$ (takes 3 seconds) $s = 100$ (takes 25 seconds) $s = 1000$ (takes 243 seconds)

Figure 6: Sampling with different number of steps

We found that there is almost no difference in performance between $s = 100$ and $s = 1000$, and although $s = 10$ results in slightly more blurry images, they are still recognizable. However, since the sampling time and number of steps are directly proportional, $s = 100$ becomes very attractive for real-world applications.

3.4 Effect of variance

We evaluate the effect of variance σ_t^2 and explain why DDIM use $\sigma_t^2 = 0$ ultimately. We have the option $\sigma_{\tau_i}^2 = \eta \tilde{\beta}_{\tau_i} = \eta \frac{1 - \bar{\alpha}_{\tau_{i-1}}}{1 - \bar{\alpha}_{\tau_i}} (1 - \frac{\alpha_{\tau_i}}{\alpha_{\tau_{i-1}}})$, where $0 < \eta < 1$, and $\sigma_{\tau_i}^2 = 1 - \frac{\alpha_{\tau_i}}{\alpha_{\tau_{i-1}}}$ (denoted

as simple variance $\hat{\sigma}$). When opting for the simple variance, $1-\bar{\alpha}_{\tau_{i-1}}-\sigma_{\tau_i}^2$ may be less than 0, therefore we set this $\sigma_{\tau_i}^2$ to $\frac{1-\bar{\alpha}_{\tau_{i-1}}}{1-\bar{\alpha}_{\tau_i}}(1-\frac{\alpha_{\tau_i}}{\alpha_{\tau_{i-1}}})$.

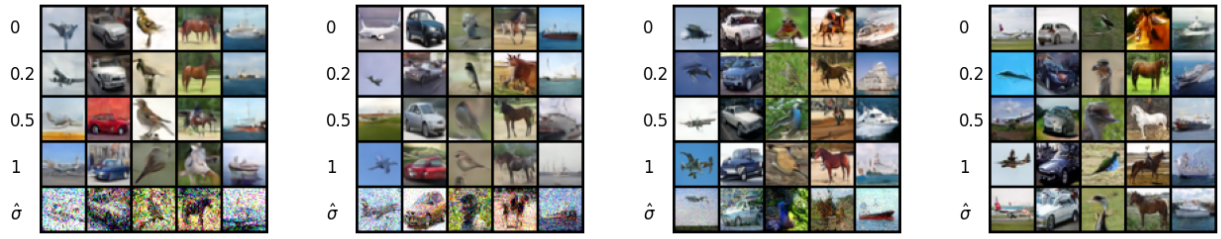
We calculated the FID of generated samples under different choices of σ_t^2 and s (Table 2). The results are, in some ways, consistent with the DIMM paper: when s increases, FID drops; when η increases, FID increases ($s = 10$ and $s = 20$); and when s is small, choosing simple variance $\hat{\sigma}$ leads to extremely poor performance.

However, there are some differences compared to the original paper. Specifically, when $s = 1000$, the simple variance $\hat{\sigma}$ was reported to achieve the best performance, and smaller η values showed better performance. In our results, however, $\hat{\sigma}$ does not outperform $\eta = 0$, and larger η values show better performance.

FID	$s = 10$	$s = 20$	$s = 1000$
$\eta = 0$	**29.10**	**20.17**	18.20
$\eta = 0.5$	29.23	21.02	15.36
$\eta = 1$	31.78	21.93	**14.01**
$\hat{\sigma}$	349.74	232.78	21.36

Table 2: Effect of variance under different s (evaluated by FID)

From the perspective of the generated images (Figure 7), we can derive similar results. When $s = 10$ and $s = 20$, a smaller η leads to better performance, and the generated images with simple variance $\hat{\sigma}$ are extremely blurry. However, when $s = 1000$, there is no significant difference between the different choices of variance.



$s = 10$ $s = 20$ $s = 100$ $s = 1000$

Figure 7: Effect of variance under different s; The y-axis label represents $\eta = 0$, $\eta = 0.2$, $\eta = 0.5$, $\eta = 1$, and simple variance $\hat{\sigma}$, respectively.

However, even though $\sigma_t^2 = 0$ yields great performance, it lacks theoretical guarantees (the weights in the ELBO loss contain $1/\sigma_t^2$), at least in the original DDIM paper.

4 Additional experiments and explorations

4.1 Denoising process

Figure 8: Denoising process

The timestamps chosen here are x_{900}, x_{300}, x_{200}, x_{100}, x_{90}, x_{80}, x_{70}, x_{60}, x_{50}, x_{40}, x_{30}, x_{20}, x_{10}, and x_0, respectively. It is interesting that when $t \geq 90$, the images are not recognizable, and when $t < 90$, the contents of the images are already settled and won't change, only becoming clearer. This raises the question: which timestamp should we choose to achieve the best performance when accelerating sampling?

4.2 Could DIMM replace VAE?

One key difference between the diffusion model and VAE is that in VAE, every data point x has a corresponding latent variable z, whereas in the diffusion model, this is not the case. This represents a special advantage of VAE, as having a specific latent variable for each data point can lead to greater interpretability. With VAE, we can model structured data and explore the effects of latent variables.

However, when $\sigma_t = 0$, the sampling procedure of DIMM is deterministic, which implies that each generated x_0 has a corresponding latent variable x_T. This raises the question: could DIMM replace VAE? I believe that at this time, the answer is no.

In VAE's training process, we have already learned a mapping from x to z denoted by $z = \mu_E(x)$, where the posterior distribution is $z \mid x \sim \mathcal{N}(\mu_E(x), \sigma_E^2(x))$. However, in DIMM, we need to learn the inverse mapping (from x_0 to x_T) after training. This is because the posterior distribution $x_T \mid x_0 \sim \mathcal{N}(\sqrt{\bar{\alpha}_T}x_0, (1-\bar{\alpha}_T)I)$ provides no information about the mapping from x_0 to x_T ($\bar{\alpha}_T \approx 10^{-7}$ when $T = 1000$). Consequently, this limitation renders DIMM inapplicable for inference tasks. Secondly, DIMM lacks theoretical guarantees in its original paper because the ELBO loss is not valid when $\sigma_t = 0$. The weight of the ELBO loss contains $1/\sigma_t^2$, which is omitted, similar to what they have done in the simple loss function.

Anyway, we acknowledge that DIMM can map x_0 to x_T and provide insights into the effect of latent variables. In our experiments, we shifted values in x_T by 0.01 each time and observed the corresponding changes in x_0 (Figure 9). We found that the generated images are not very sensitive to the shift in x_T, indicating the possibility of learning a reverse mapping afterwards.

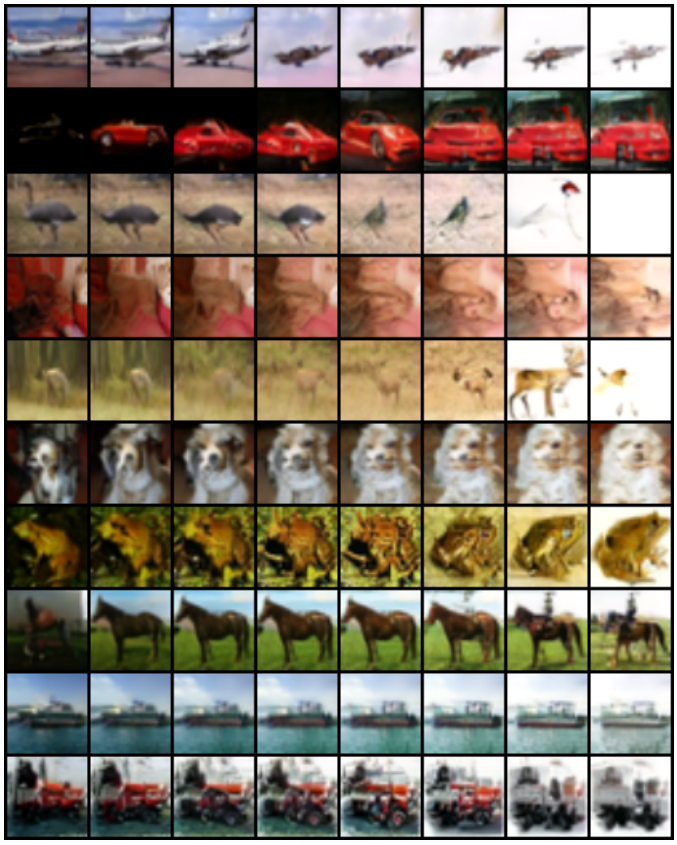


Figure 9: Effect of latent variable (Changes caused by shifts in the latent variable x_T values)

5 Appendix

5.1 Derivation for $q(x_{t-1} \mid x_t, x_0)$

Utilizing Bayes' rule, we have

$$q(x_{t-1} \mid x_t, x_0) = q(x_t \mid x_{t-1}, x_0)\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} = q(x_t \mid x_{t-1})\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)}. \tag{17}$$

Given $q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$ and $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$, we have

$$\begin{aligned} q(x_{t-1} \mid x_t, x_0) =& \frac{1}{\beta_t\sqrt{2\pi}} e^{-\frac{1}{2\beta_t}(x_t - \sqrt{1-\beta_t}x_{t-1})^2} \frac{1}{(1-\bar{\alpha}_{t-1}\sqrt{2\pi})} e^{-\frac{1}{2(1-\bar{\alpha}_{t-1})}(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2} \cdot \\ & (1-\bar{\alpha}_t)\sqrt{2\pi} e^{\frac{(x_t - \sqrt{\bar{\alpha}_t}x_0)^2}{2(1-\bar{\alpha}_t)}} \\ =& \frac{1-\bar{\alpha}_t}{\beta_t(1-\bar{\alpha}_{t-1})\sqrt{2\pi}} e^{-\frac{(x_{t-1} - \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t - \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0)^2}{2\beta_t\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}}}. \end{aligned} \tag{18}$$

Thus, we have $q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I)$, where

$$\tilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \tag{19}$$

$$\tilde{\beta}_t := \sigma_t^2 := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \tag{20}$$

5.2 Derivation for ELBO loss

Now we outline the derivation of the ELBO loss:

$$\begin{aligned} Loss_{ELBO} &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log\frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right] \\ &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p(x_T) - \sum_{t=1}^{T}\log\frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\right] \\ &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p(x_T) - \sum_{t=1}^{T}\log\left(\frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)}\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)}\right)\right] \\ &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log\frac{p(x_T)}{q(x_T \mid x_0)} - \sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} - \log p_\theta(x_0 \mid x_1)\right] \\ &= \mathrm{KL}\left[q(x_T \mid x_0)||p(x_T)\right] + \sum_{t=2}^{T}L_{t-1} + L_0. \end{aligned} \tag{21}$$

$\mathrm{KL}\left[q(x_T \mid x_0)||p(x_T)\right]$ can be omitted from consideration since it does not depend on θ. For L_{t-1}, we have

$$
\begin{aligned}
L_{t-1} &= -\,\mathbb{E}_{q(x_{1:T}|x_0)}\left[\log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)}\right] \\
&= -\,\mathbb{E}_{q(x_t|x_0)q(x_{t-1}|x_t,x_0)q(x_{1:T/\{t-1,t\}}|x_{t-1},x_t,x_0)}\left[\log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)}\right] \\
&= \mathbb{E}_{q(x_t|x_0)}\left[\mathrm{KL}(q(x_{t-1} \mid x_t, x_0)||p_\theta(x_{t-1} \mid x_t))\right] \\
&= \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\,\mathbb{E}_{\epsilon_t\sim\mathcal{N}(0,I)}\left[\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)\right].
\end{aligned} \tag{22}
$$

Given that x_0 is deterministic given x_1 (see algorithm 2 in figure 3), the term $\log p_\theta(x_0 \mid x_1)$ is technically not applicable. Many papers include L_0 in their loss function without explicitly calculating or deriving $\log p_\theta(x_0 \mid x_1)$. Instead, they use $\mathbb{E}_{\epsilon_1}\left[||\epsilon_1 - \epsilon_\theta(x_1, 1)||^2\right]$ as L_0.

Thus, we have the last result:

$$
Loss_{ELBO} \equiv \sum_{t=1}^{T} \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}||\epsilon_t - \epsilon(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)||^2, \tag{23}
$$

where "$\equiv$" indicates that the expressions are equivalent when optimizing $\epsilon_\theta(x_t, t)$.

5.3 Model for accelerated sampling

Suppose we choose $s+1$ steps for sampling: $T = \tau_s > \tau_{s-1} > \cdots > \tau_0 = 0$. Denote $\bar{\tau}$ as $\{1, \ldots, T\}/\{\tau_0, \ldots, \tau_s\}$. The inference process is:

$$
q_{\sigma,\tau}(x_{1:T} \mid x_0) = q_{\sigma,\tau}(x_T \mid x_0)\prod_{i=1}^{s} q_{\sigma,\tau}(x_{\tau_{i-1}} \mid x_{\tau_i}, x_0)\prod_{t\in\bar{\tau}} q_{\sigma,\tau}(x_t \mid x_0), \tag{24}
$$

$$
\text{where} \quad q_{\sigma,\tau}(x_t \mid x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, \sqrt{1-\bar{\alpha}_t}I), \quad t \in \bar{\tau} \cup \{T\}, \tag{25}
$$

$$
q_{\sigma,\tau}(x_{\tau_{i-1}} \mid x_{\tau_i}, x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_{\tau_{i-1}}}x_0 + \sqrt{1-\bar{\alpha}_{\tau_{i-1}}-\sigma_t^2}\frac{x_t - \sqrt{\bar{\alpha}_{\tau_i}}x_0}{\sqrt{1-\bar{\alpha}_{\tau_i}}}, \sigma_t^2 I), \quad i \in [S]. \tag{26}
$$

The reverse process is:

$$
p_\theta(x_{0:T}) = p_\theta(x_T)\prod_{i=1}^{s} p_\theta(x_{\tau_{i-1}} \mid x_{\tau_i})\prod_{t\in\bar{\tau}} p_\theta(x_t \mid x_0), \tag{27}
$$

$$
\text{where} \quad p_\theta(x_{\tau_{i-1}} \mid x_{\tau_i}) = q_{\sigma,\tau}\left(x_{\tau_{i-1}} \mid x_{\tau_i}, \frac{1}{\sqrt{\bar{\alpha}_{\tau_i}}}\left[x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}}\epsilon_\theta(x_{\tau_i}, \tau_i)\right]\right), \tag{28}
$$

$$
p_\theta(x_t \mid x_0) = \mathcal{N}(x_0; \frac{1}{\sqrt{\bar{\alpha}_{\tau_i}}}\left[x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}}\epsilon_\theta(x_{\tau_i}, \tau_i)\right], \sigma_t^2 I). \tag{29}
$$

The simple loss function remains unchanged, as $\mathrm{KL}\left[q_{\sigma,\tau}(x_{\tau_{i-1}} \mid x_{\tau_i}, x_0)||p_\theta(x_{\tau_{i-1}} \mid x_{\tau_i})\right] \equiv C\,\mathbb{E}_{\epsilon_{\tau_i}}\left[||\epsilon_{\tau_i} - \epsilon_\theta(x_{\tau_i}, \tau_i)||^2\right]$ and $\mathrm{KL}\left[q_{\sigma,\tau}(x_t \mid x_0)||p_\theta(x_t \mid x_0)\right] \equiv C\,\mathbb{E}_{\epsilon_t}\left[||\epsilon_t - \epsilon_\theta(x_t, t)||^2\right]$.

5.4 Previously generated samples by VQVAE

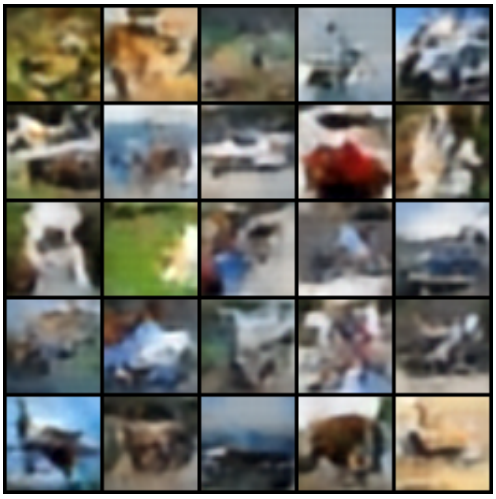


Figure 10: VQVAE's samples

5.5 Resources

Codes: https://github.com/yinjjiew/Diffusion-Model