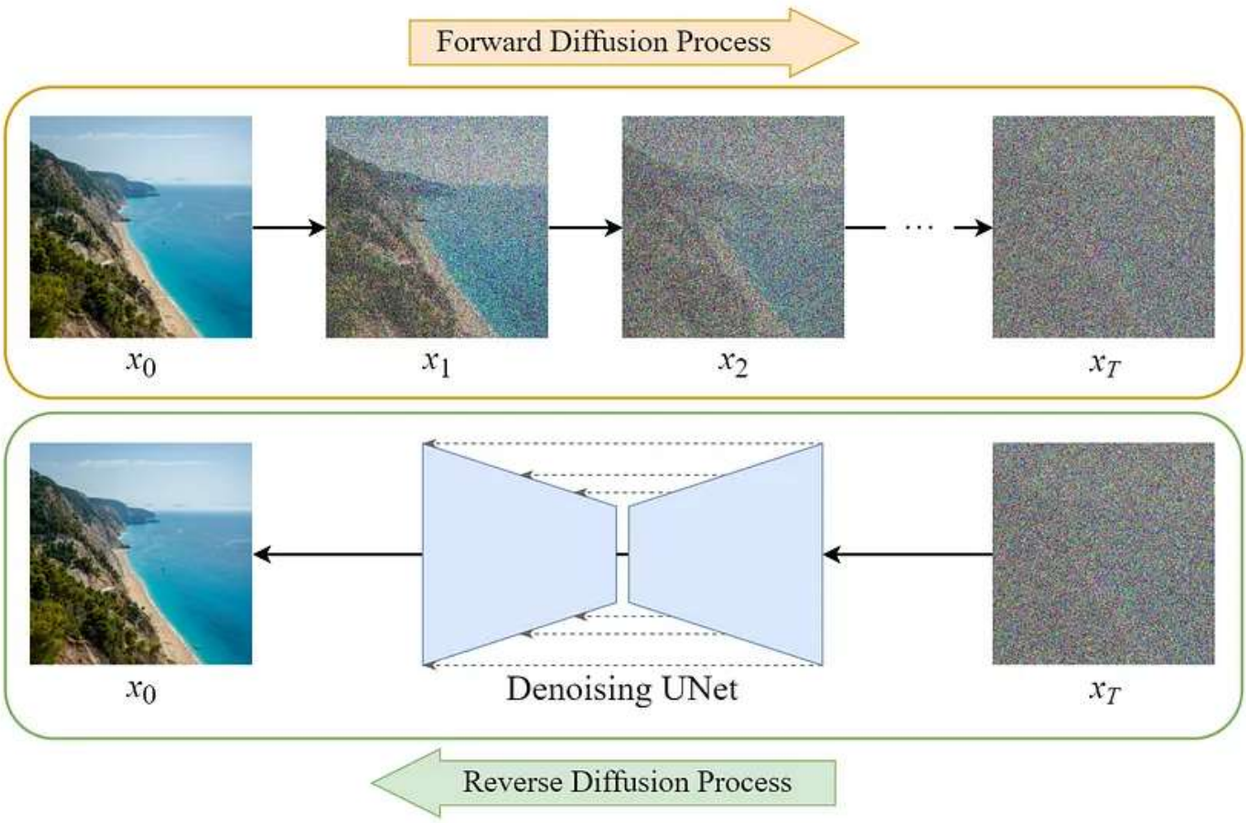
Overview

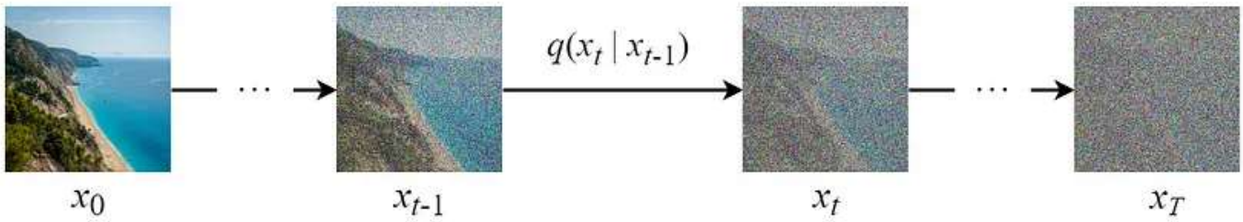


Overview of the Diffusion Model

The training of the Diffusion Model can be divided into two parts:

1. Forward Diffusion Process → add noise to the image.
2. Reverse Diffusion Process → remove noise from the image.

Forward Diffusion Process



Distribution of the noised images — Output — Mean μ_t — Variance Σ_t

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t;\ \sqrt{1-\beta_t}x_{t-1},\ \beta_t I)$$

Notations:

- t : time step (from 0 to T)
- x_0 : a data sampled from the real data distrition $q(x)$ (i.e. $x_0 \sim q(x)$)
- β_t : variance schedule ($0 \le \beta_t \le 1$, and β_0 = small number, β_T = large number)
- I : identity matrix

Forward diffusion process

The forward diffusion process gradually adds Gaussian noise to the input image x_0 step by step, and there will be T steps in total. the process will produce a sequence of noisy image samples $x_1, \ldots, x\_T$.

When T → ∞, the final result will become a completely noisy image as if it is sampled from an isotropic Gaussian distribution.

But instead of designing an algorithm to iteratively add noise to the image, we can use a closed-form formula to directly sample a noisy image at a specific time step t.

Closed-Form Formula

The closed-form sampling formula can be derived using the Reparameterization Trick.

$$\text{If } z \sim \mathcal{N}(\mu, \sigma^2) \text{ then}$$
$$z = \mu + \sigma\varepsilon \quad \text{where } \varepsilon \sim \mathcal{N}(0,1)$$

Reparameterization trick

With this trick, we can express the sampled image x_t as follows:

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon_{t-1}$$

Express $\mathbf{x}_t$ using the reparameterization trick

Then we can expand it recursively to get the closed-form formula:

$$
\begin{aligned}
x_t &= \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon_{t-1} \\
&= \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon_{t-1} \\
&= \sqrt{\alpha_t}\, \left(\sqrt{\alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}}\, \varepsilon_{t-2} \right) + \sqrt{1-\alpha_t}\, \varepsilon_{t-1} \\
&= \sqrt{\alpha_t\, \alpha_{t-1}}\, x_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\, \varepsilon_{t-2} + \sqrt{1-\alpha_t}\, \varepsilon_{t-1} \\
&= \sqrt{\alpha_t\, \alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_t\, \alpha_{t-1}}\, \bar{\varepsilon}_{t-2} \\
&\vdots \\
&= \sqrt{\alpha_t\, \alpha_{t-1}\, \dots\, \alpha_1}\, x_0 + \sqrt{1-\alpha_t\, \alpha_{t-1}\, \dots\, \alpha_1}\, \varepsilon \\
&= \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon
\end{aligned}
$$

How?

$$
\begin{aligned}
&\varepsilon_0, \dots, \varepsilon_{t-2}, \varepsilon_{t-1} \sim \mathcal{N}(0, I) \\
&\bar{\varepsilon}_0, \dots, \bar{\varepsilon}_{t-2}, \bar{\varepsilon}_{t-1} \sim \mathcal{N}(0, I) \\
&\varepsilon \sim \mathcal{N}(0, I) \\
&\alpha_t = 1 - \beta_t \\
&\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i
\end{aligned}
$$

Derivation of the closed-form formula

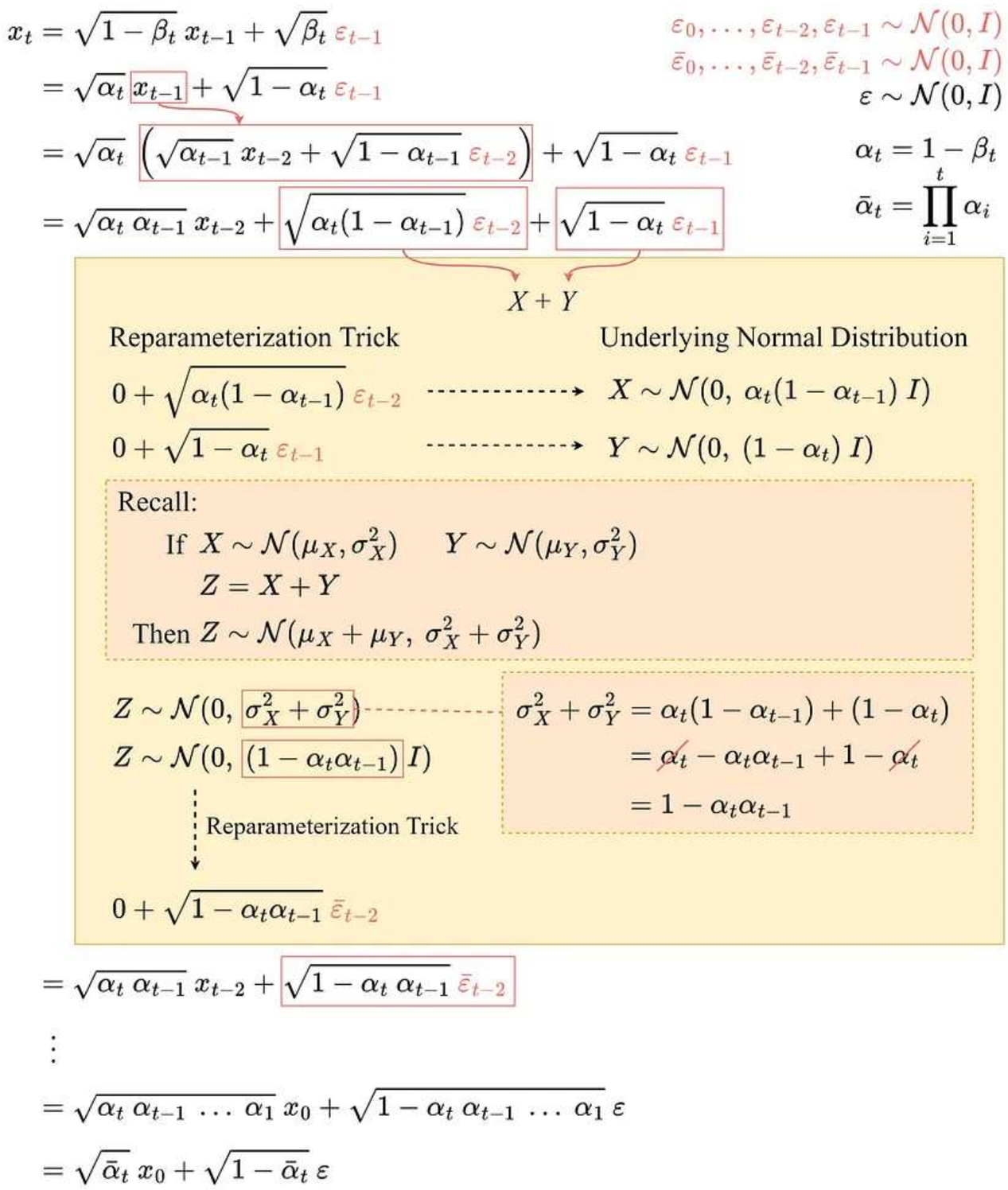
> *Note:*
> *all the ε are i.i.d. (independent and identically distributed) standard normal random variables.*
>
> *It is important to distinguish them using different symbols and subscripts because they are independent and their values could be different after sampling.*

But how do we jump from line 4 to line 5?

$$
\begin{aligned}
&\sqrt{\alpha_t(1-\alpha_{t-1})}\, \varepsilon_{t-2} + \sqrt{1-\alpha_t}\, \varepsilon_{t-1} \\
&\sqrt{1-\alpha_t\, \alpha_{t-1}}\, \bar{\varepsilon}_{t-2}
\end{aligned}
$$

How?

Some people find this step difficult to understand. Here I will show you how it works:



Detailed derivation from line 4 to line 5

Let's denote these two terms using X and Y. They can be regarded as samples from two different normal distributions. i.e. $X \sim N(0, \alpha_t(1-\alpha_{t-1})I)$ and $Y \sim N(0, (1-\alpha_t)I)$.

Recall that the sum of two normally distributed (independent) random variables is also normally distributed. i.e. if $Z = X + Y$, then $Z \sim N(0, \sigma^2_X+\sigma^2_Y)$.

Therefore we can merge them together and express the merged normal distribution in the reparameterized form. This is how we combine the two terms.

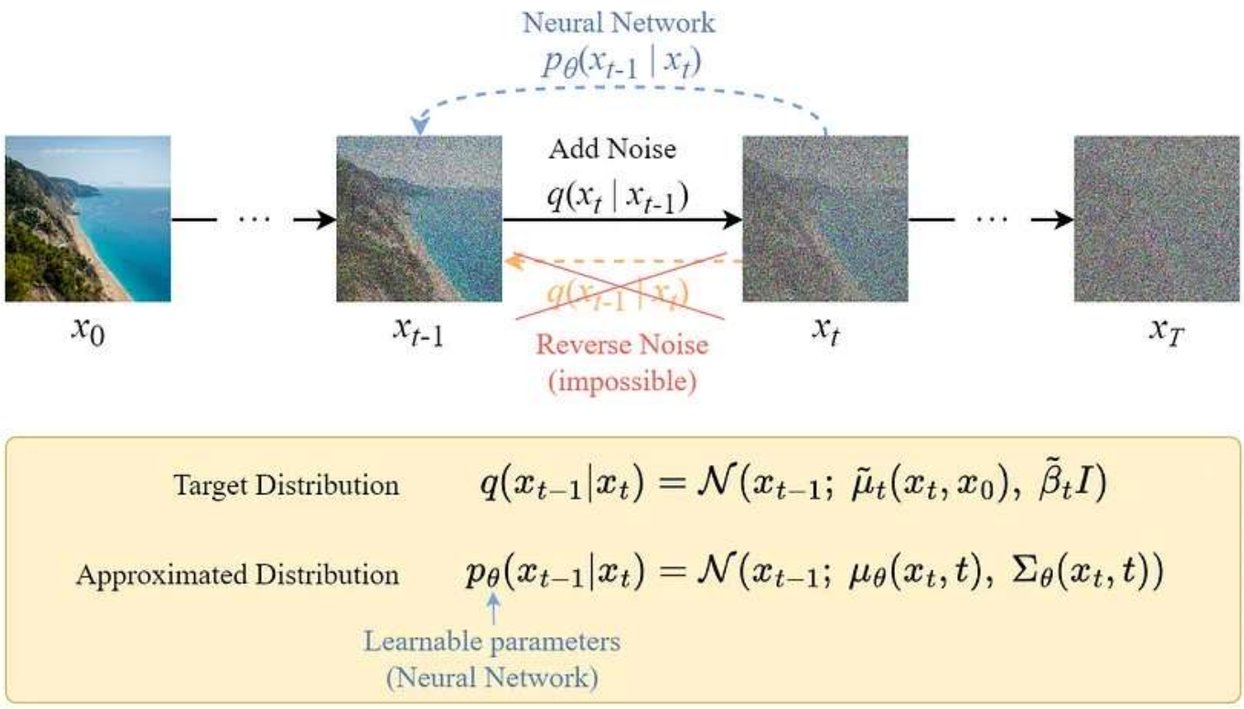
Repeating these steps will give us the following formula which depends only on the input image x_0:

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

The closed-form formula

Now we can directly sample x_t at any time step using this formula, and this makes the forward process much faster.

Reverse Diffusion Process



Reverse diffusion process

Unlike the forward process, we cannot use $q(x_{t-1}|x_t)$ to reverse the noise since it is intractable (uncomputable).

Thus we need to train a neural network $p\theta(x_{t-1}|x_t)$ to approximate $q(x_{t-1}|x_t)$. The approximation $p\theta(x_{t-1}|x_t)$ follows a normal distribution and its mean and variance are set as follows:

$$\begin{cases} \mu_\theta(x_t, t) & := \tilde{\mu}_t(x_t, x_0) \\ \Sigma_\theta(x_t, t) & := \tilde{\beta}_t I \end{cases}$$

mean and variance of **pθ**

Loss Function

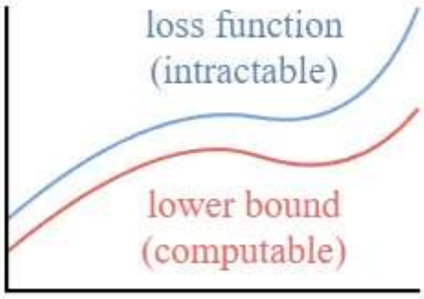
We can define our loss as a Negative Log-Likelihood:

$$\text{Loss} = -\log(p_\theta(x_0))$$

Depends on $x_1, x_2, \ldots, x_T$
Therefore it is intractable!

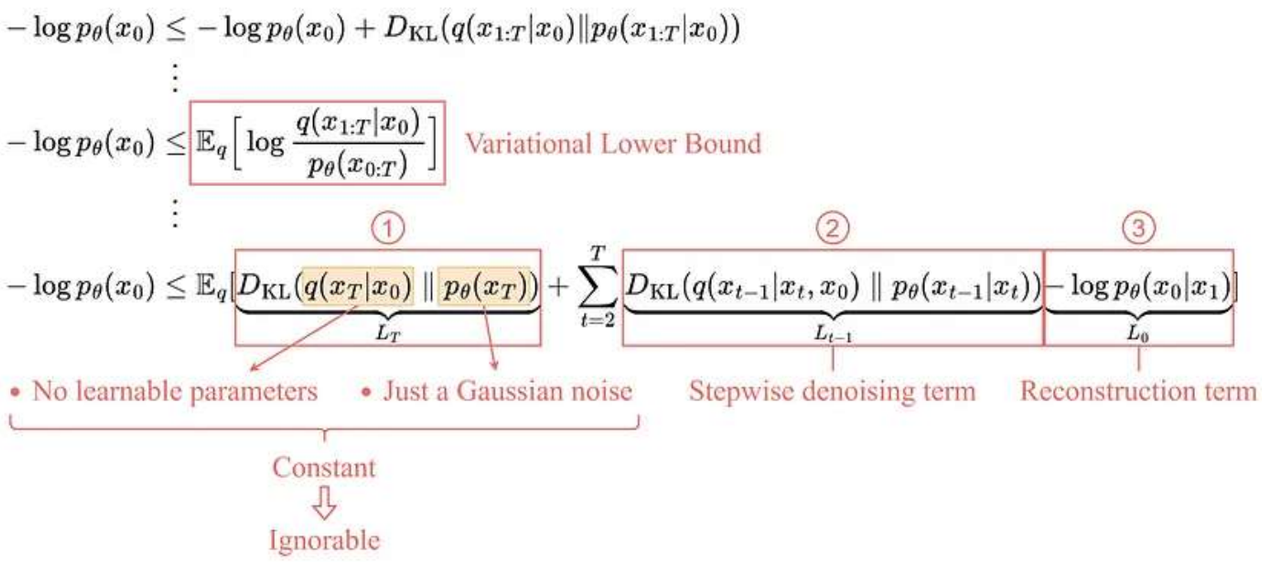
Negative log-likelihood

This setup is very similar to the one in VAE. instead of optimizing the intractable loss function itself, we can optimize the Variational Lower Bound.



Variational Lower Bound

By optimizing a computable lower bound, we can indirectly optimize the intractable loss function.



Derivation and expansion of the Variational Lower Bound

By expanding the variational lower bound, we found that it can be represented with the following three terms:

1. *L_T*: Constant term

Since *q* has no learnable parameters and *p* is just a Gaussian noise probability, this term will be a constant during training and thus can be ignored.

2. L_{t-1}: Stepwise denoising term

This term compares the target denoising step *q* and the approximated denoising step *pθ*.

> *Note that by conditioning on x_0, the $q(x_{t-1}|x_t, x_0)$ becomes tractable.*

$$\overbrace{D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t))}^{L_{t-1}}$$

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\boldsymbol{\mu}}(x_t, x_0), \tilde{\beta}_t I) \qquad p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_\theta(x_t, t), \beta_t I)$$

Neural Network

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

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

$$\vdots$$

$$\tilde{\boldsymbol{\mu}}_t(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_t\right)$$

Details of the stepwise denoising term

After a series of derivations, the mean $\tilde{\mu_t}$ of $q(x_{t-1}|x_t, x_0)$ is shown above.

For those who want to see the step-by-step derivation of the mean $\tilde{\mu_t}$ (yellow box), please see the Appendix.

To approximate the target denoising step q, we only need to approximate its mean using a neural network. So we set the approximated mean $\mu\theta$ to be in the same form as the target mean $\tilde{\mu_t}$ (with a learnable neural network $\varepsilon\theta$):

$$\tilde{\boldsymbol{\mu}}_t(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_t\right)$$

$$\text{Set} \quad \boldsymbol{\mu}_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_\theta(x_t, t)\right)$$

Approximated mean

The comparison between the target mean and the approximated mean can be done using a mean squared error (MSE):

$$\begin{aligned}
L_t &= \mathbb{E}_{x_0,\boldsymbol{\varepsilon}}\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(x_t) - \boldsymbol{\mu}_\theta(x_t, t)\|^2\right] \\
&= \mathbb{E}_{x_0,\boldsymbol{\varepsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_t\right) - \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\varepsilon}_\theta(x_t, t)\right)\right\|^2\right] \\
&= \mathbb{E}_{x_0,\boldsymbol{\varepsilon}}\left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\sigma_t^2}\|\boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_\theta(x_t, t)\|^2\right]
\end{aligned}$$

ignorable

$$L_t^{\text{simple}} = \mathbb{E}_{t\sim[1,T],x_0,\boldsymbol{\varepsilon}_t}\left[\|\boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_\theta(x_t, t)\|^2\right]$$

Derivation of the simplified stepwise denoising loss

Experimentally, better results can be achieved by ignoring the weighting term and simply comparing the target and predicted noises with MSE.

So, it turns out that to approximate the desired denoising step q, we just need to approximate the noise ε_t using a neural network $\varepsilon\theta$.

3. L_0: Reconstruction term

This is the reconstruction loss of the last denoising step and it can be ignored during training for the following reasons:

- It can be approximated using the same neural network in L_{t-1}.
- Ignoring it makes the sample quality better and makes it simpler to implement.

Simplified Loss

So the final simplified training objective is as follows:

$$x_t = \sqrt{\bar{a}_t}\, x_0 + \sqrt{1 - \bar{a}_t}\, \varepsilon$$

$$L_{\text{simple}} = \mathbb{E}_{t, x_0, \varepsilon}\left[\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2 \right]$$

Simplified training objective

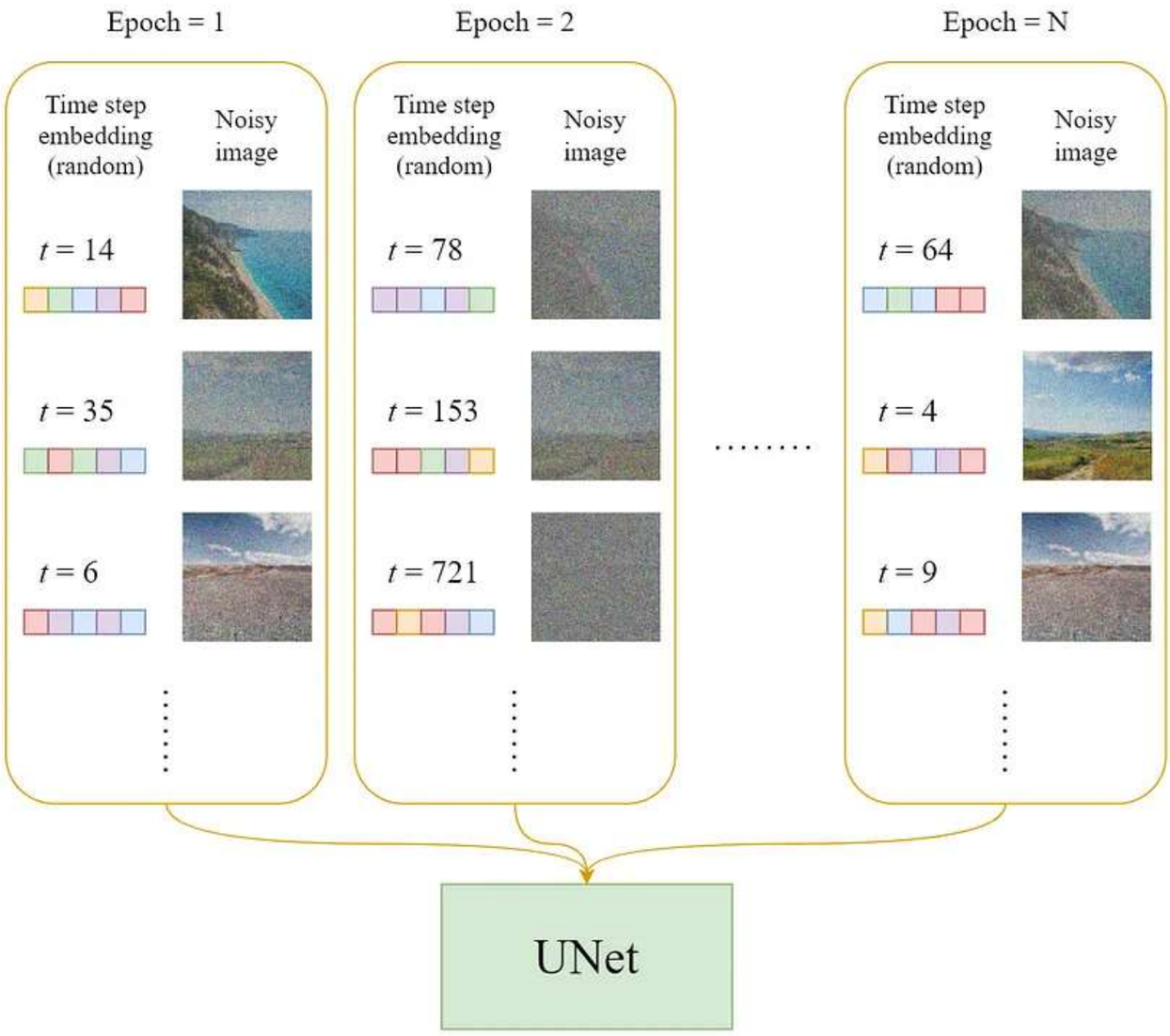
> *We find that training our models on the true variational bound yields better codelengths than training on the simplified objective, as expected, but the latter yields the best sample quality. [2]*

The U-Net Model

Dataset

In each epoch:

1. A random time step t will be selected for each training sample (image).
2. Apply the Gaussian noise (corresponding to t) to each image.
3. Convert the time steps to embeddings (vectors).



Dataset for training

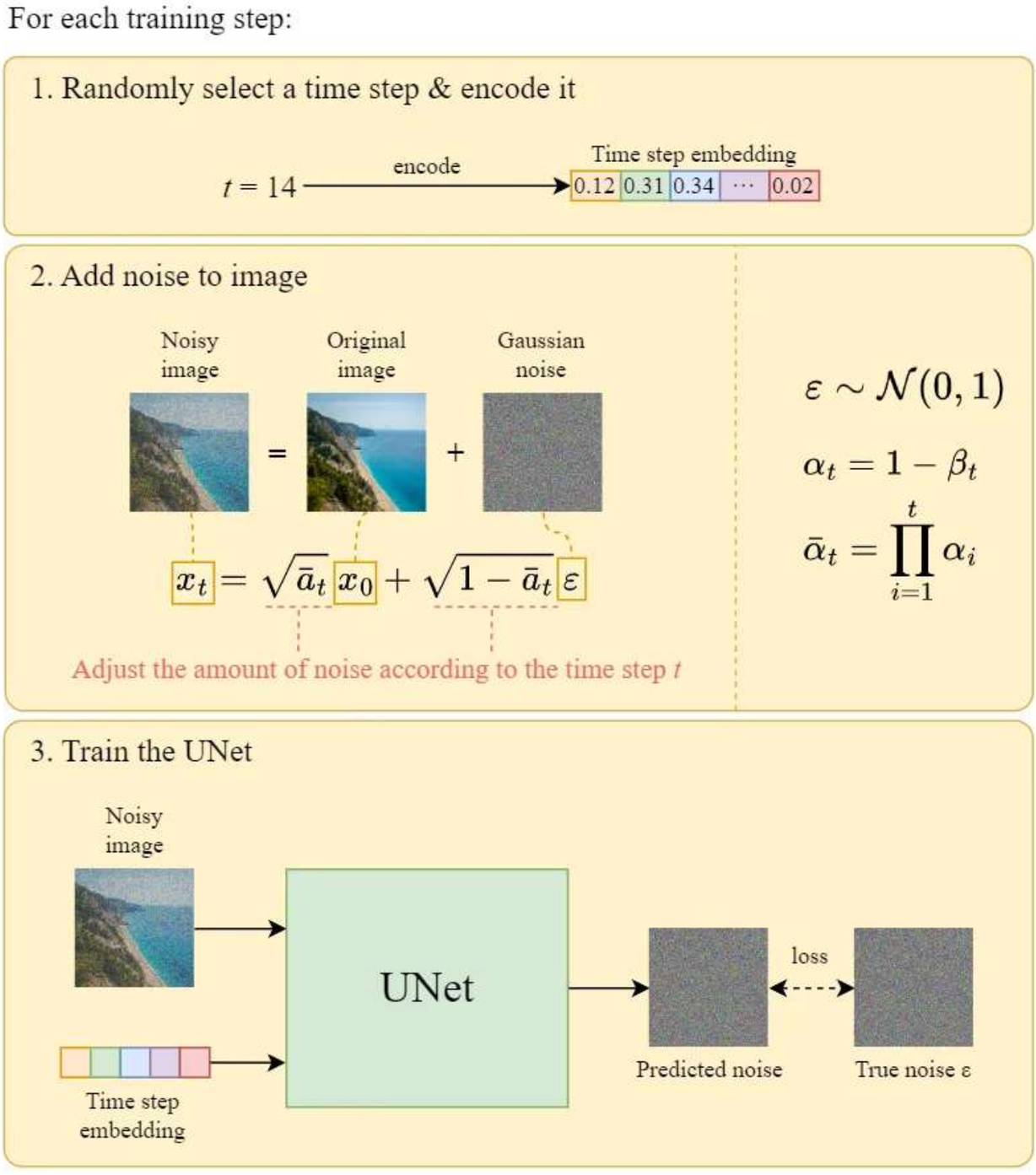
Training

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

Training algorithm [2]

The official training algorithm is as above, and the following diagram is an illustration of how a training step works:



Training step illustration

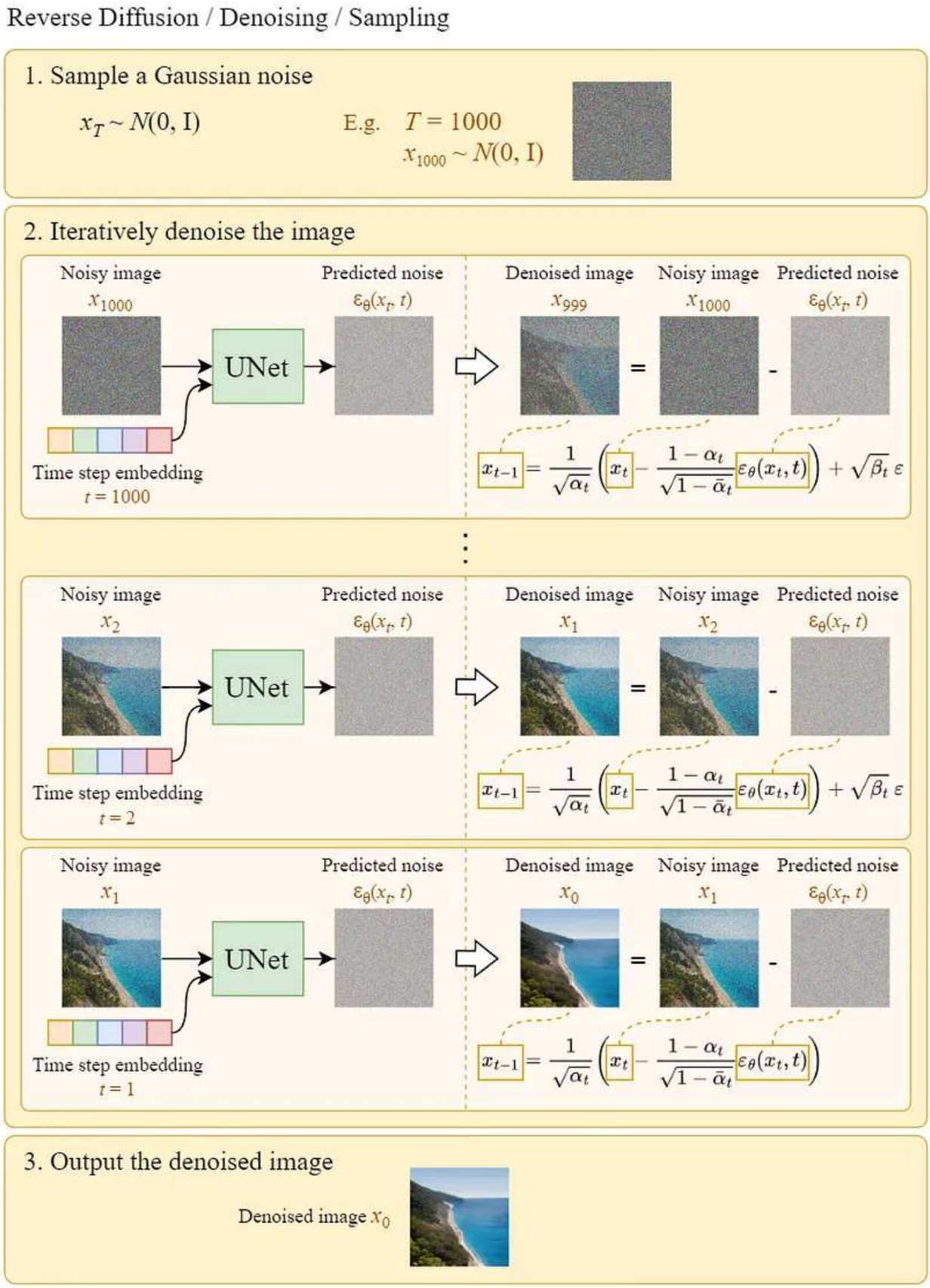
Reverse Diffusion

Algorithm 2 Sampling

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

Sampling algorithm [2]

We can generate images from noises using the above algorithm. The following diagram is an illustration of it:



Sampling illustration

Note that in the last step, we simply output the learned mean $\mu\theta(x_1, 1)$ without adding the noise to it.

Summary

Here are some main takeaways from this article:

- The Diffusion model is divided into two parts: forward diffusion and reverse diffusion.
- The forward diffusion can be done using the closed-form formula.

- The backward diffusion can be done using a trained neural network.
- To approximate the desired denoising step q, we just need to approximate the noise ε_t using a neural network $\varepsilon\theta$.
- Training on the simplified loss function yields better sample quality.

Appendix

The following is the detailed derivation of the mean $\tilde{\mu}_t$ of $q(x_{t-1}|x_t, x_0)$ in the **stepwise denoising term** in the **Loss Function section**.

$$
\begin{aligned}
\tilde{\mu}_t(x_t, x_0) &= \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 \\
\tilde{\mu}_t(x_t) &= \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}\cdot\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_t) \\
&= \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}\cdot\frac{1}{\sqrt{\bar{\alpha}_{t-1}\cdot\alpha_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_t) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\cancel{\sqrt{\bar{\alpha}_{t-1}}}\beta_t}{1-\bar{\alpha}_t}\cdot\frac{1}{\cancel{\sqrt{\bar{\alpha}_{t-1}}}\sqrt{\alpha_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_t) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\beta_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}(x_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_t) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\beta_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t - \frac{\beta_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\sqrt{1-\bar{\alpha}_t}\varepsilon_t \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t - \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\sqrt{1-\bar{\alpha}_t}\varepsilon_t \\
&= \frac{\alpha_t}{\sqrt{\alpha_t}}\cdot\frac{(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)}x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t - \frac{1-\alpha_t}{\sqrt{\alpha_t}}\cdot\frac{\sqrt{1-\bar{\alpha}_t}}{(1-\bar{\alpha}_t)}\cdot\varepsilon_t \\
&= \frac{\alpha_t - \bar{\alpha}_{t-1}\cdot\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t - \frac{1-\alpha_t}{\sqrt{\alpha_t}}\cdot\frac{1}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_t \\
&= \frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t - \frac{1}{\sqrt{\alpha_t}}\cdot\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_t \\
&= \frac{\cancel{1-\bar{\alpha}_t}}{\sqrt{\alpha_t}\cancel{(1-\bar{\alpha}_t)}}x_t - \frac{1}{\sqrt{\alpha_t}}\cdot\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_t \\
&= \frac{1}{\sqrt{\alpha_t}}x_t - \frac{1}{\sqrt{\alpha_t}}\cdot\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_t \\
\tilde{\mu}_t(x_t) &= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_t\right)
\end{aligned}
$$

$$
\begin{cases}
\bar{\alpha}_t &= \alpha_1\alpha_2\cdots\alpha_{t-1}\alpha_t \\
\bar{\alpha}_{t-1} &= \alpha_1\alpha_2\cdots\alpha_{t-1}
\end{cases}
\qquad \bar{\alpha}_t = \bar{\alpha}_{t-1}\cdot\alpha_t
$$

$$
\begin{aligned}
\sqrt{\alpha_t} &= \sqrt{\alpha_t}\cdot\frac{\sqrt{\alpha_t}}{\sqrt{\alpha_t}} \\
&= \frac{\alpha_t}{\sqrt{\alpha_t}}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\sqrt{1-\bar{\alpha}_t}}{1-\bar{\alpha}_t} &= \frac{\sqrt{1-\bar{\alpha}_t}}{1-\bar{\alpha}_t}\cdot\frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}} \\
&= \frac{\cancel{1-\bar{\alpha}_t}}{\cancel{(1-\bar{\alpha}_t)}\cdot\sqrt{1-\bar{\alpha}_t}} \\
&= \frac{1}{\sqrt{1-\bar{\alpha}_t}}
\end{aligned}
$$

Derivation of the mean $\tilde{\mu}_t$ of $q(x_{t-1}|x_t, x_0)$

References

[1] K. Roose, "An a.i.-generated picture won an art prize. artists aren't happy.," *The New York Times*, 02-Sep-2022. [Online]. Available: https://www.nytimes.com/2022/09/02/technology/ai-artificial-intelligence-artists.html.

[2] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic models," arXiv.org, 16-Dec-2020. [Online]. Available: https://arxiv.org/abs/2006.11239.