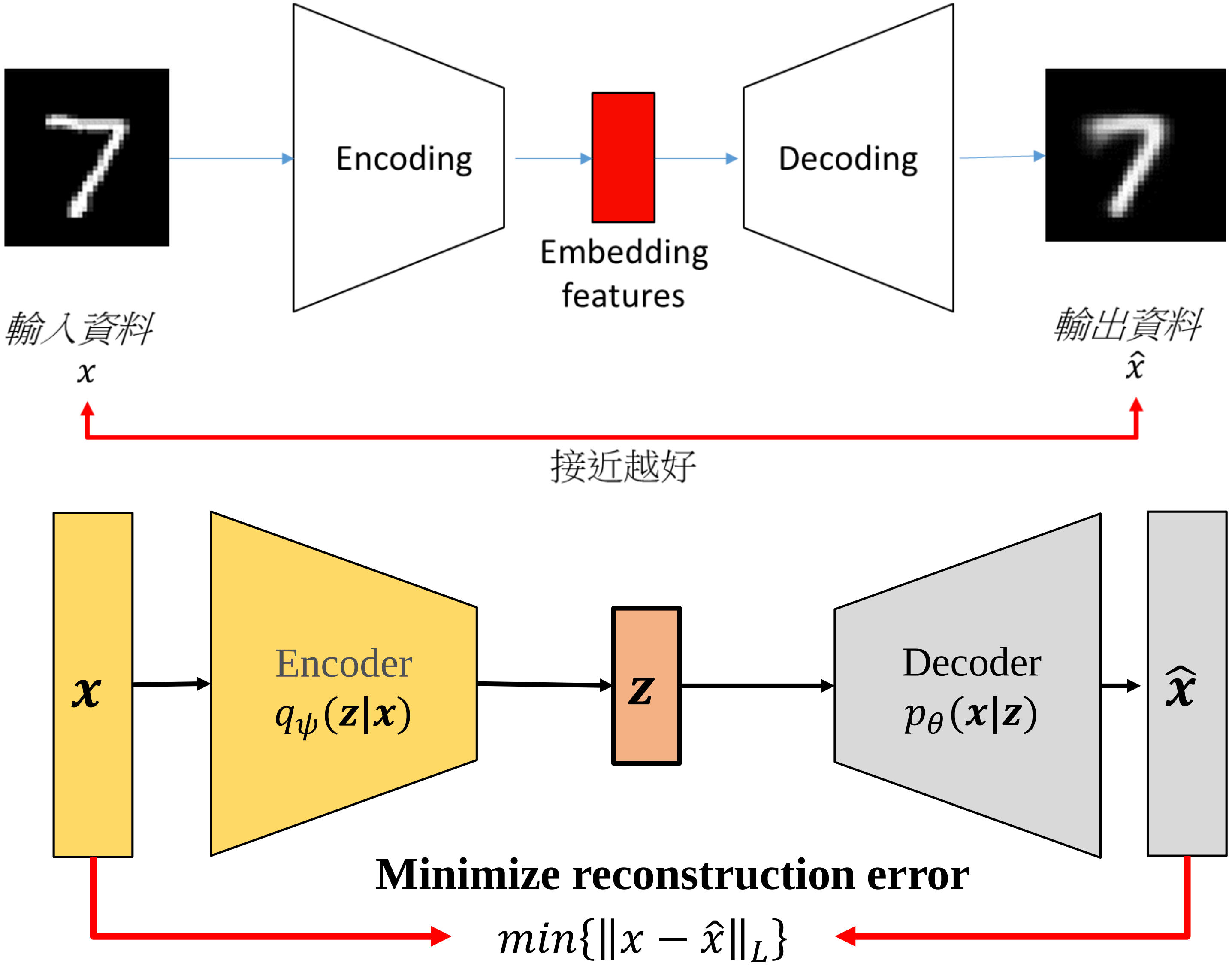
生成式AI:
Variational AutoEncoder & Diffusion model

黃志勝 (Tommy Huang)
義隆電子 人工智慧研發部
國立陽明交通大學 AI學院 合聘助理教授
國立台北科技大學 電資學院合聘助理教授

~~Variational~~ AutoEncoder



Reconstruction loss
叫 decoder 別亂畫圖，畫得像輸入

影像常見用
Binary Cross Entropy（BCE）或 MSE

$$L_{reconstruction} = \mathbb{E}_{q_\psi(\mathbf{z}|\boldsymbol{x})}\left[-log p_\theta(\boldsymbol{x}|\mathbf{z})\right]$$

希望在 sampling 出來的 latent vector z下，decoder 重建出來的 x 的機率越高越好。

AutoEncoder

Autoencoder lower bound: (推導省略)

$$\log(p\ (x)) \geq \boxed{\mathbb{E}_{q_\psi(\boldsymbol{z}|\boldsymbol{x})}\ [logp_\theta(\boldsymbol{x}|\boldsymbol{z})]} - \mathrm{D}_{\mathrm{KL}}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z))$$

當x是二值化輸出(黑白影像)

$$p_\theta(x|z) = Bernoulli(x; \hat{x}) =$$

$\hat{x} = decoder(z)$: 輸出是每個pixel為1的機率

$x \in \{0,1\}$

$Bernoulli$的log-likelihood為(概似函數要最大化)

$$Bernoulli: f(x) = p^x(1-p)^{(1-x)} = \begin{cases} p & if\ x = 1 \\ 1-p & if\ x = 0 \end{cases}$$

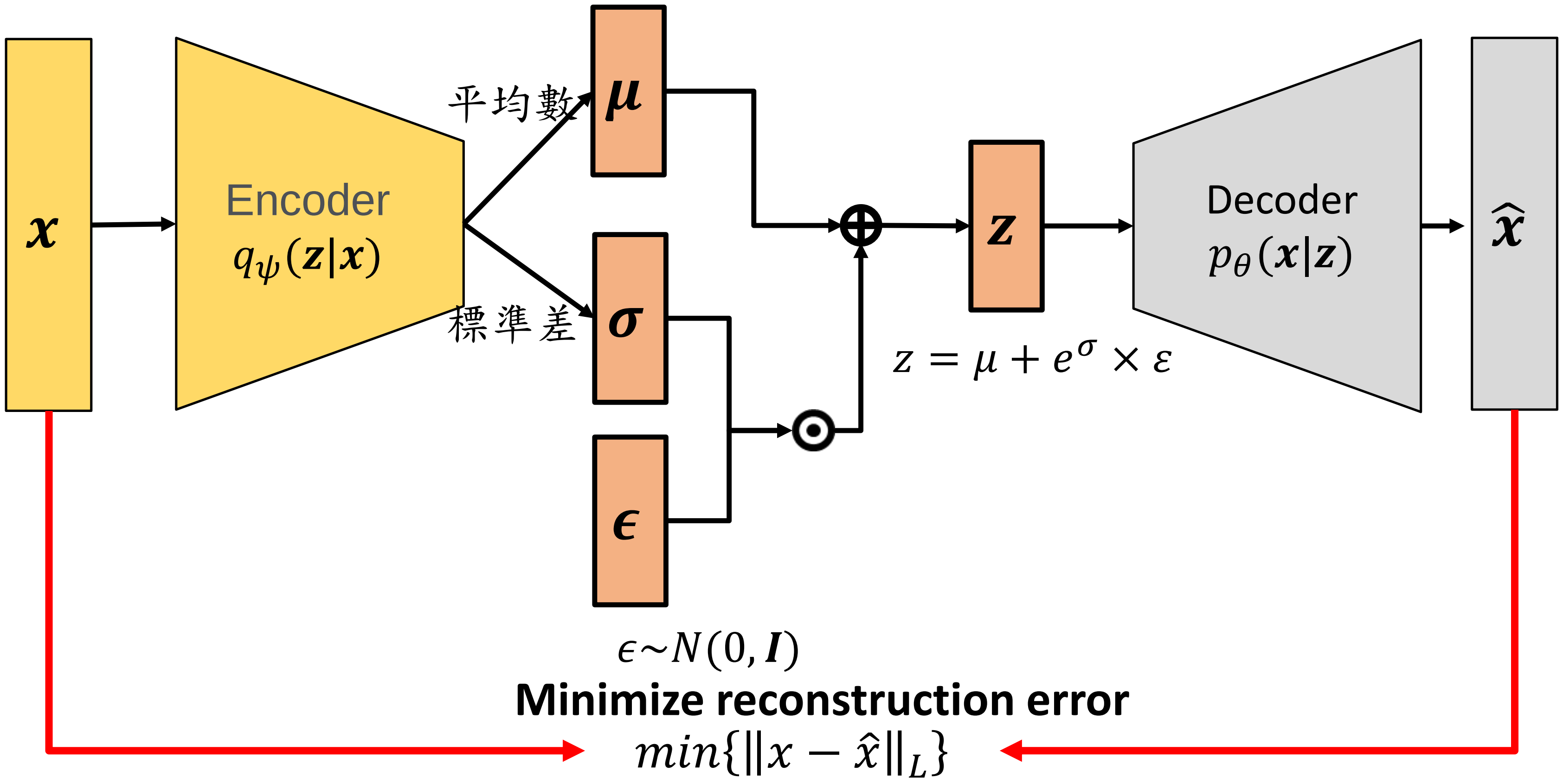
聯合機率分布函數有興趣自行推導
每個樣本機率相乘起來，取log變成相加

$$\log(p_\theta(x|z)) = \sum_i x_i log\hat{x}_i + (1-x_i)\log(1-\hat{x}_i)$$

BCE(目標要最小化，取負號):

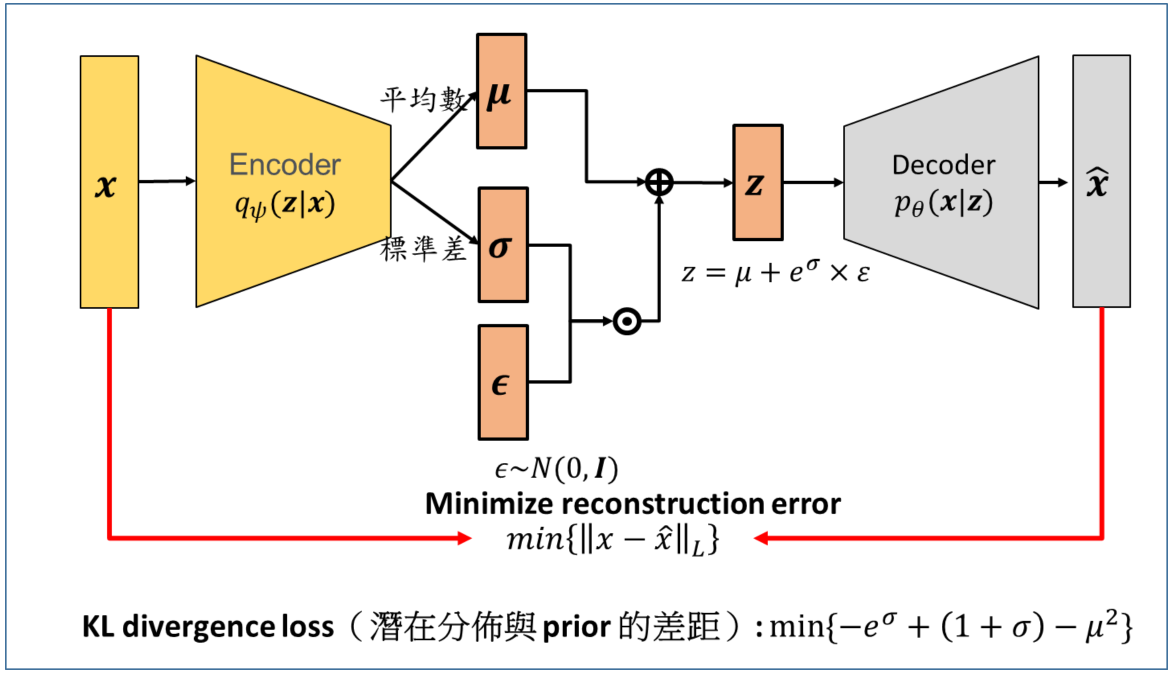
$$L_{reconstruction} = -logp_\theta(x|z)$$

Variational AutoEncoder



KL divergence loss（潛在分佈與 prior 的差距）: $\min\{-e^{\sigma} + (1+\sigma) - \mu^2\}$

Variational AutoEncoder



Autoencoder lower bound: (推導省略)

$$\log(p(x)) \geq \mathbb{E}_{q_\psi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \boxed{\mathrm{D_{KL}}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z))}$$

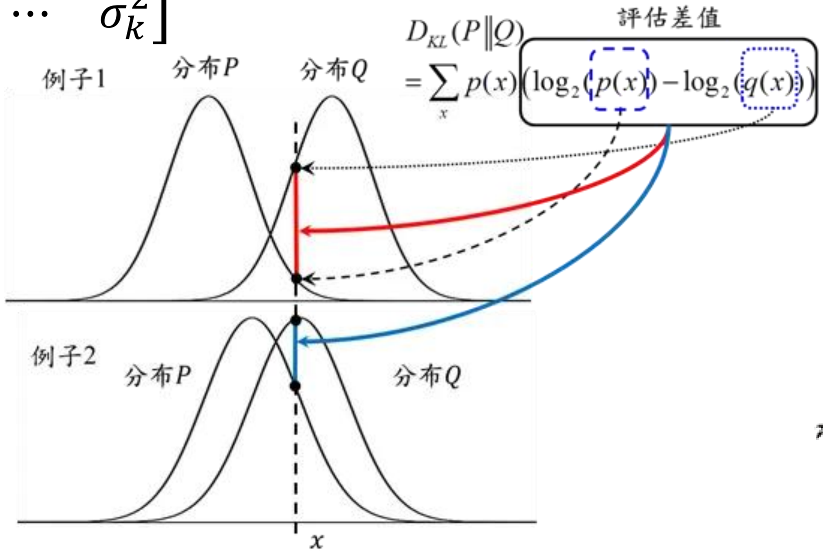
$$L_{\mathrm{KL}} = \mathrm{D_{KL}}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z))$$

$q_\psi(\boldsymbol{z}|\boldsymbol{x}) = \boldsymbol{N}(\mu, diag(\sigma^2))$: encoder latent distribution

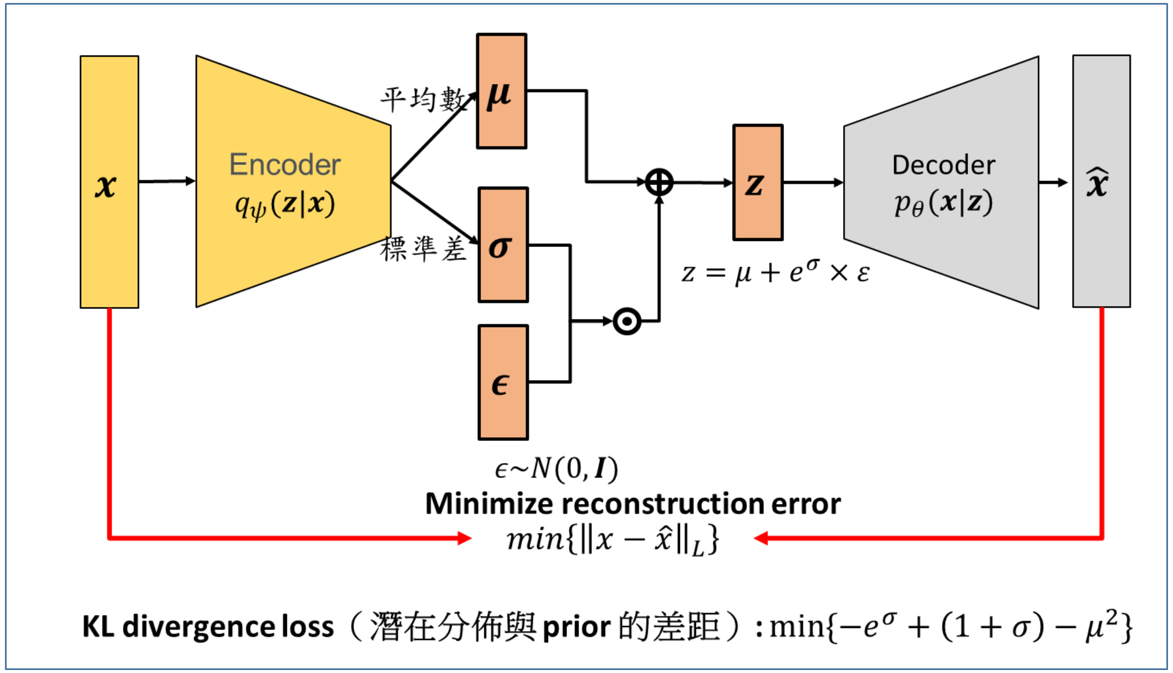
$p(z) = N(0, \boldsymbol{I})$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k^2 \end{bmatrix}$$

- 讓 encoder 學出來的 latent distribution $q_\psi(\boldsymbol{z}|\boldsymbol{x})$不要亂跑，而是$\boldsymbol{Z}$盡量靠近一個標準常態分布 $Z \sim N(0, \boldsymbol{I})$
- 減少 overfitting
- 可以保證潛在空間是連續且有結構的 → 可做樣本生成、插值



Variational AutoEncoder



- 讓 encoder 學出來的 latent distribution $q_\psi(\boldsymbol{z}|\boldsymbol{x})$不要亂跑，而是$\boldsymbol{Z}$盡量靠近一個標準常態分布 $Z \sim N(0, \boldsymbol{I})$
- 減少 overfitting
- 可以保證潛在空間是連續且有結構的 → 可做 樣本生成、插值

Autoencoder lower bound: (推導省略)

$$\log(p(x)) \geq \mathbb{E}_{q_\psi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \boxed{D_{KL}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z))}$$

$$L_{KL} = D_{KL}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z))$$

$q_\psi(\boldsymbol{z}|\boldsymbol{x}) = \boldsymbol{N}(\mu, diag(\sigma^2))$: encoder latent distribution

$p(z) = N(0, \boldsymbol{I})$

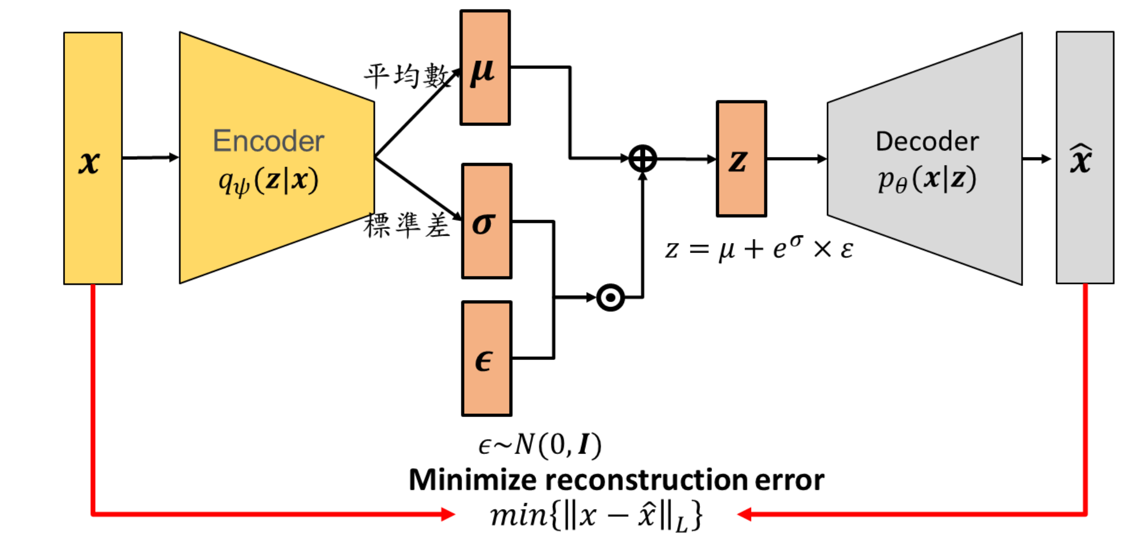
$$\Sigma = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k^2 \end{bmatrix}$$

$$D_{KL}(q_\psi(\boldsymbol{z}|\boldsymbol{x})||p(z)) = \frac{1}{2}(trace(\Sigma) + \mu^T\mu - k - \log(det(\Sigma)))$$

$$= \frac{1}{2}(\sum_i \sigma_i^2) + \mu^T\mu - k - \sum_i \log\sigma_i^2)$$

$$= \frac{1}{2}\sum_i(\sigma_i^2 + \mu_i^2 - 1 - \log\sigma_i^2)$$

Variational AutoEncoder



KL divergence loss（潛在分佈與**prior**的差距）: $\min\{-e^{\sigma} + (1+\sigma) - \mu^2\}$

KL divergence loss（潛在分佈與 **prior** 的差距）:

$$\min\{\boxed{-e^{\sigma}} + (1 + \boxed{\sigma}) - \mu^2\}$$

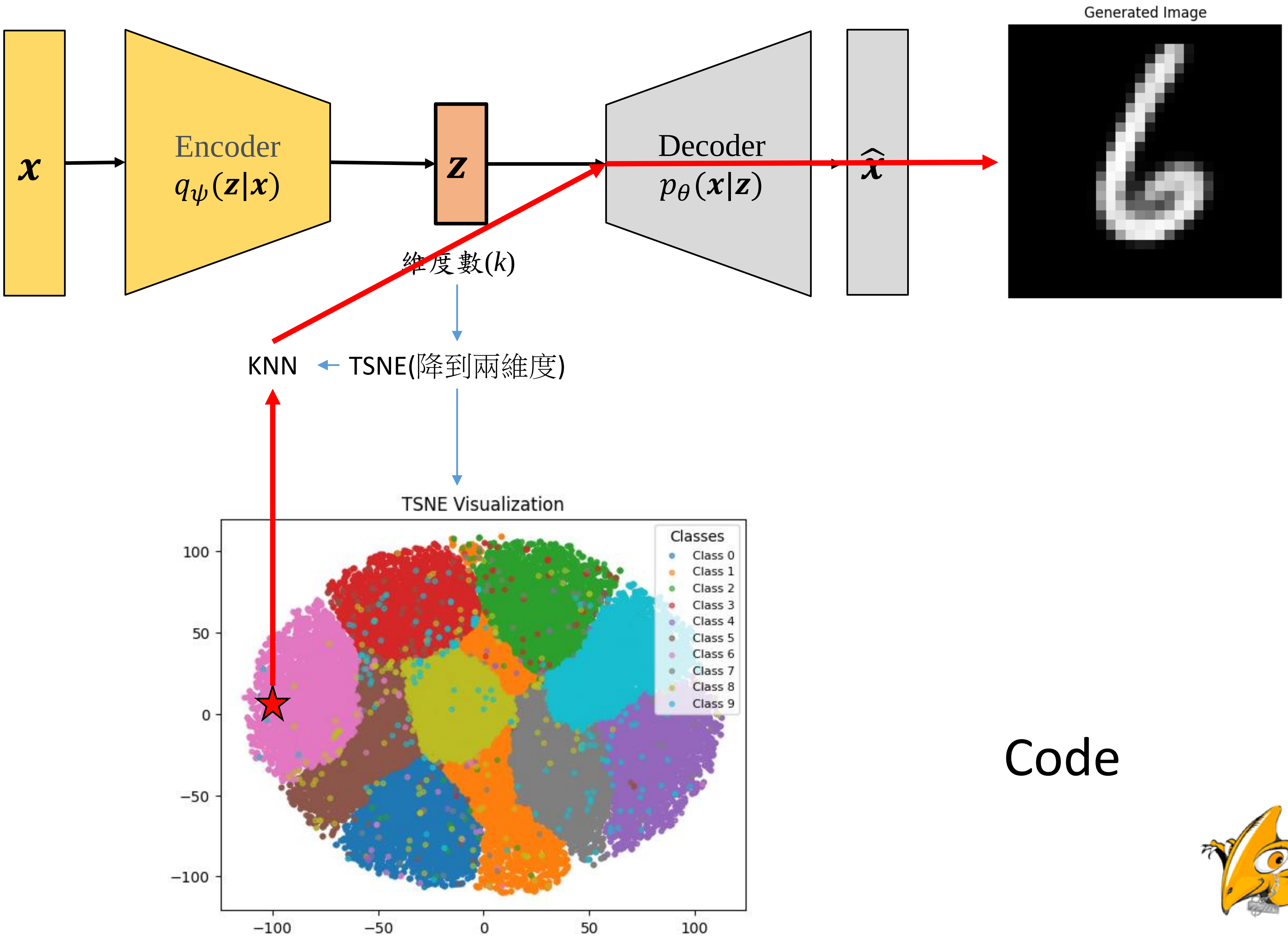
$$-\mathrm{D_{KL}}(q_\psi(\mathbf{z}|\mathbf{x})||p(z = \frac{1}{2}\sum_i (\boxed{-\sigma_i^2} - \mu_i^2 + 1 + \boxed{log\sigma_i^2})$$

```
kl_loss = -0.5 * torch.sum(1 + log_var - mu.pow(2) - log_var.exp())
```

因為我們假設Encoder輸出是log variance ($log\sigma_i^2$)

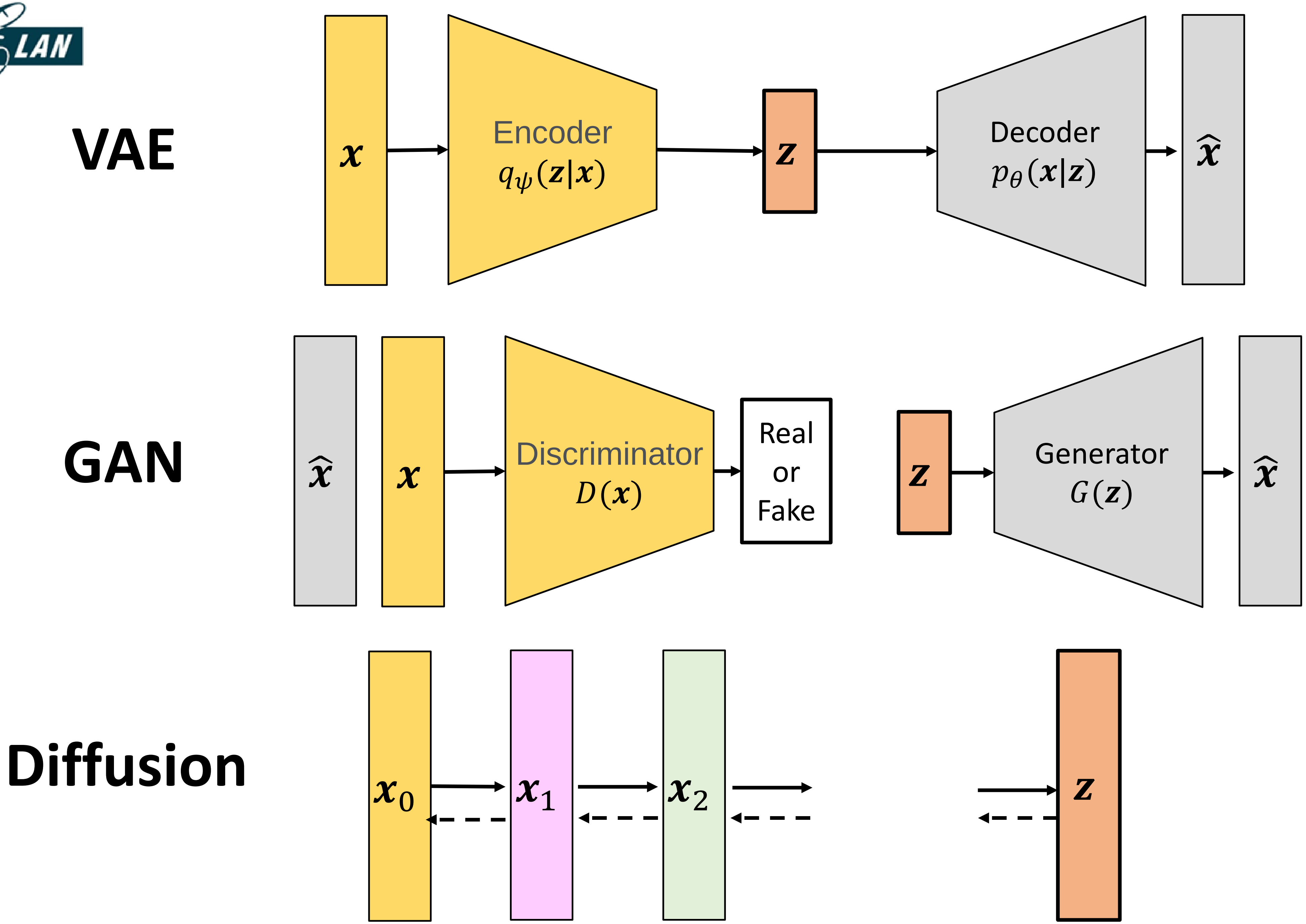
Autoencoder lower bound: (推導省略)

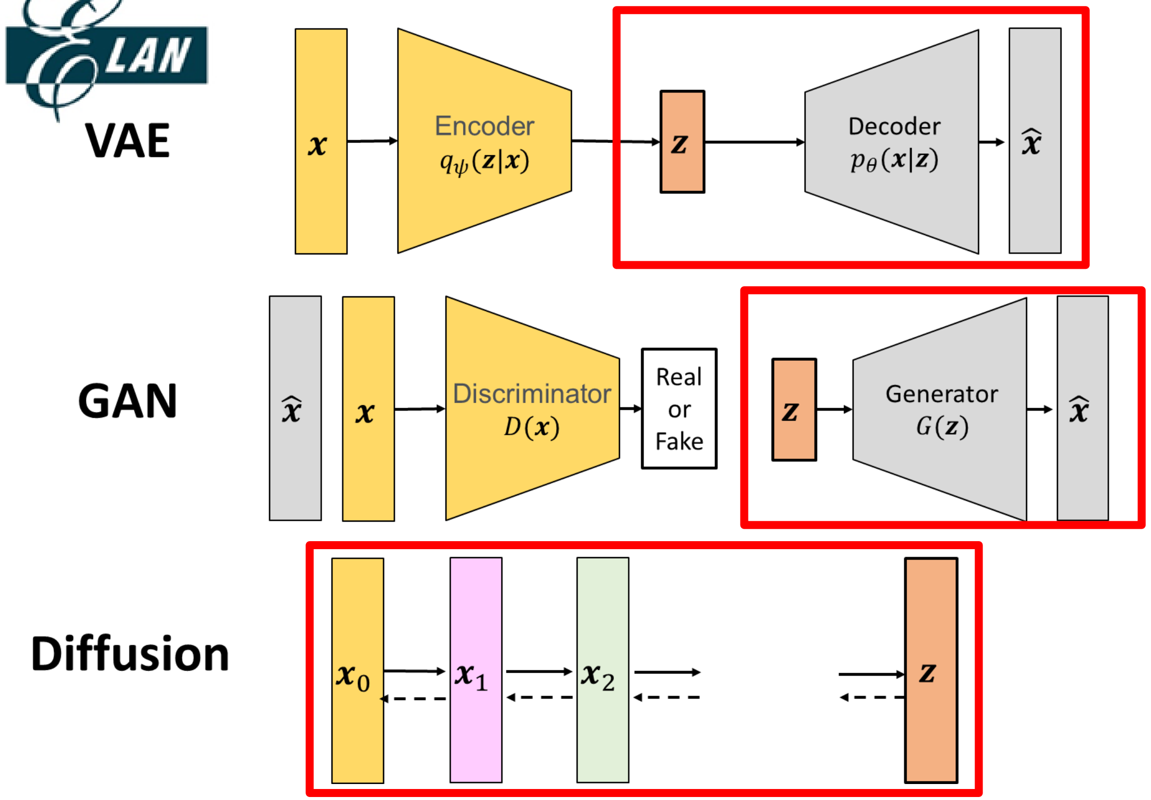
$$\log(p\,(x)) \geq \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x})}\,[log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{D_{KL}}(q_\psi(\mathbf{z}|\mathbf{x})||p(z))$$



Generated Image
x
Encoder
$q_\psi(\mathbf{z}|\mathbf{x})$
z
Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\hat{\mathbf{x}}$
維度數(k)
KNN
TSNE(降到兩維度)
TSNE Visualization
Classes
Class 0
Class 1
Class 2
Class 3
Class 4
Class 5
Class 6
Class 7
Class 8
Class 9
100
50
0
−50
−100
−100
−50
0
50
100

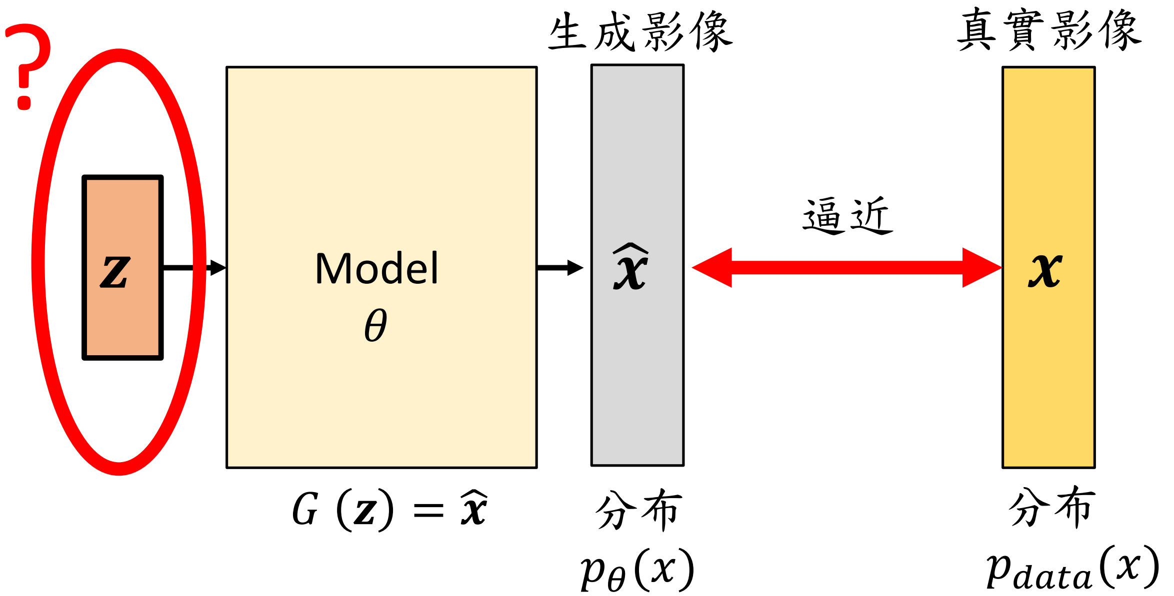
Code


VAE
x
Encoder
$q_\psi(z|x)$
z
Decoder
$p_\theta(x|z)$
$\hat{x}$
GAN
$\hat{x}$
x
Discriminator
$D(x)$
Real
or
Fake
z
Generator
$G(z)$
$\hat{x}$
Diffusion
x_0
x_1
x_2
z


VAE
x
Encoder
$q_\psi(z|x)$
z
Decoder
$p_\theta(x|z)$
$\hat{x}$
GAN
$\hat{x}$
x
Discriminator
$D(x)$
Real or Fake
z
Generator
$G(z)$
$\hat{x}$
Diffusion
x_0
x_1
x_2
z

生成影像

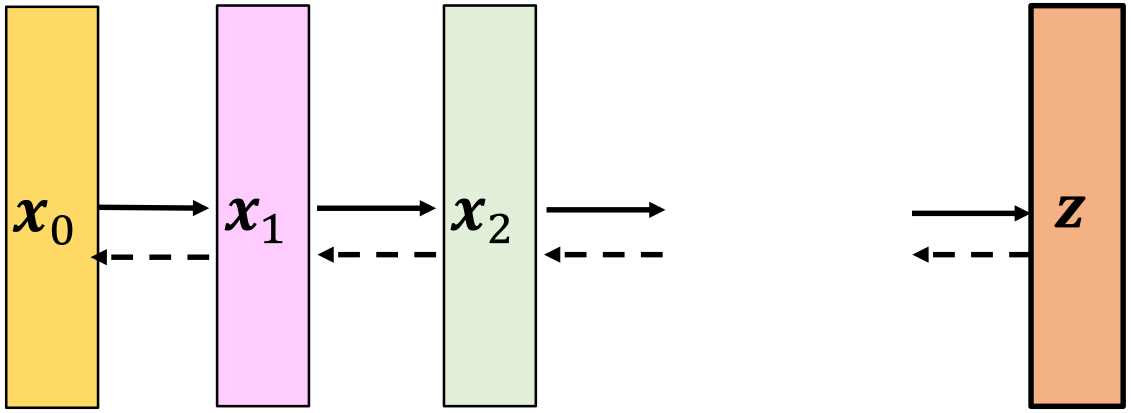
真實影像


?
z
Model
θ
生成影像
$\hat{x}$
逼近
真實影像
x
$G\ (z) = \hat{x}$
分布
$p_\theta(x)$
分布
$p_{data}(x)$

Diffusion Model (DDPM)

$\boldsymbol{x}_0$:原始圖片

$\boldsymbol{z}$:雜訊



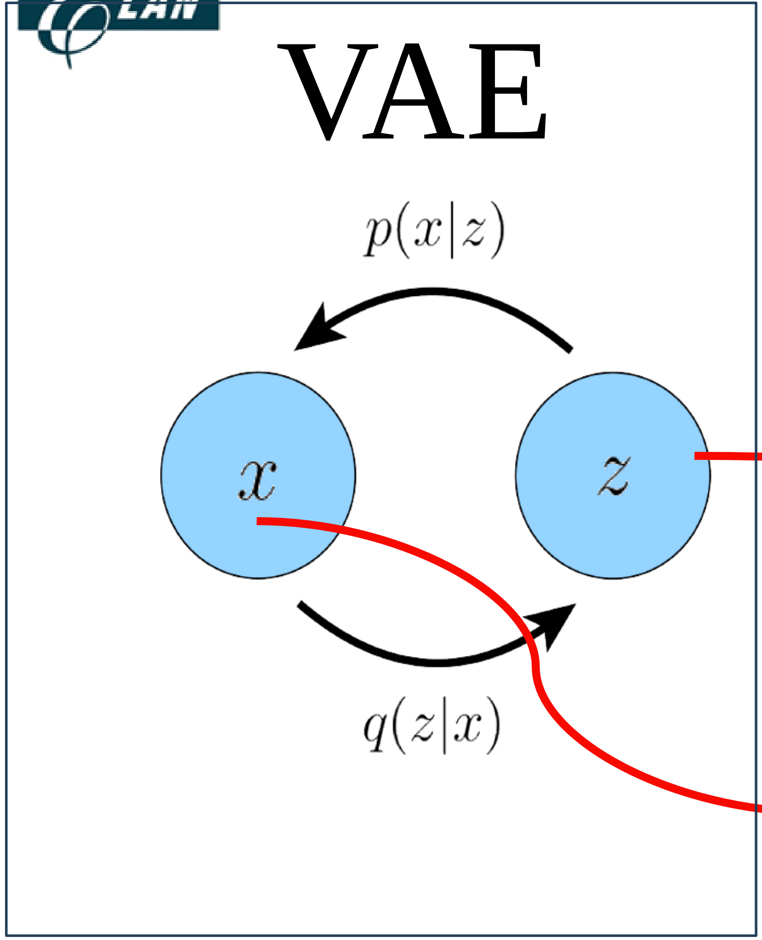
Gradually add Gaussian noise and then reverse

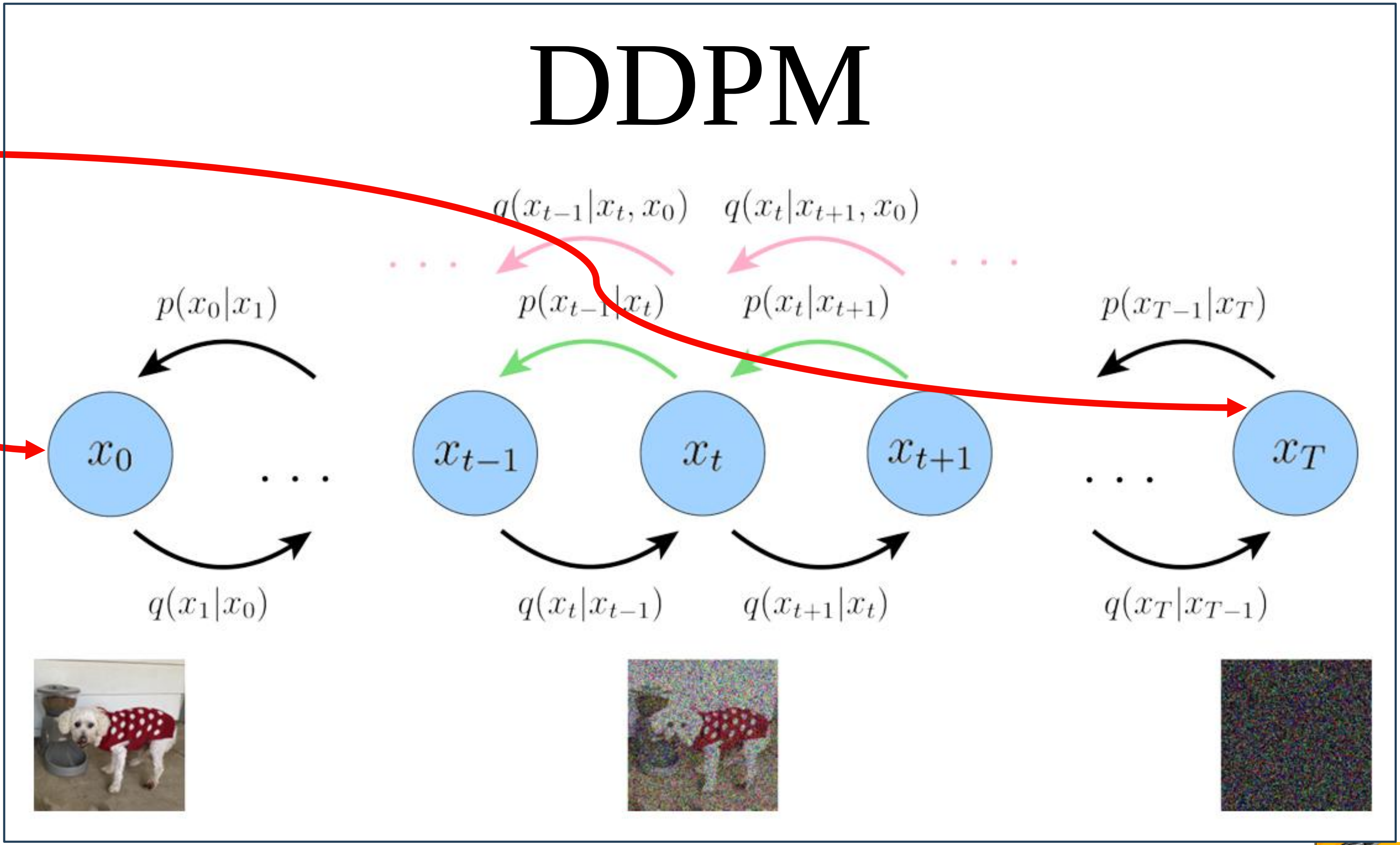
Markov chain of diffusion steps to slowly add random noise to data and then learn to reverse the diffusion process to construct desired data samples from the noise

Denoising diffusion probabilistic models (**DDPM**; Ho et al. 2020).

$$p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \dots \boldsymbol{x}_{t-p}) = p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \dots \boldsymbol{x}_{t-p}, \boldsymbol{x}_{t-p-1}, \dots \boldsymbol{x}_0)$$

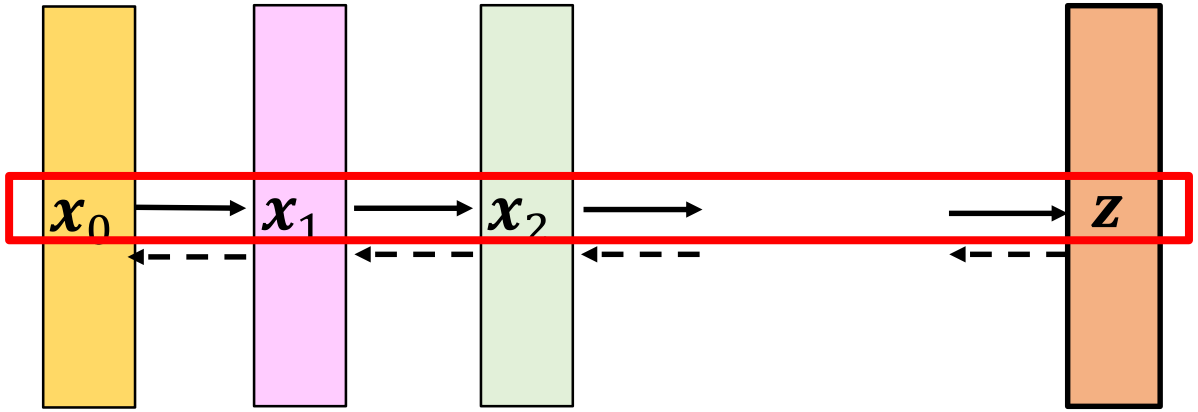
https://lilianweng.github.io/posts/2021-07-11-diffusion-models/



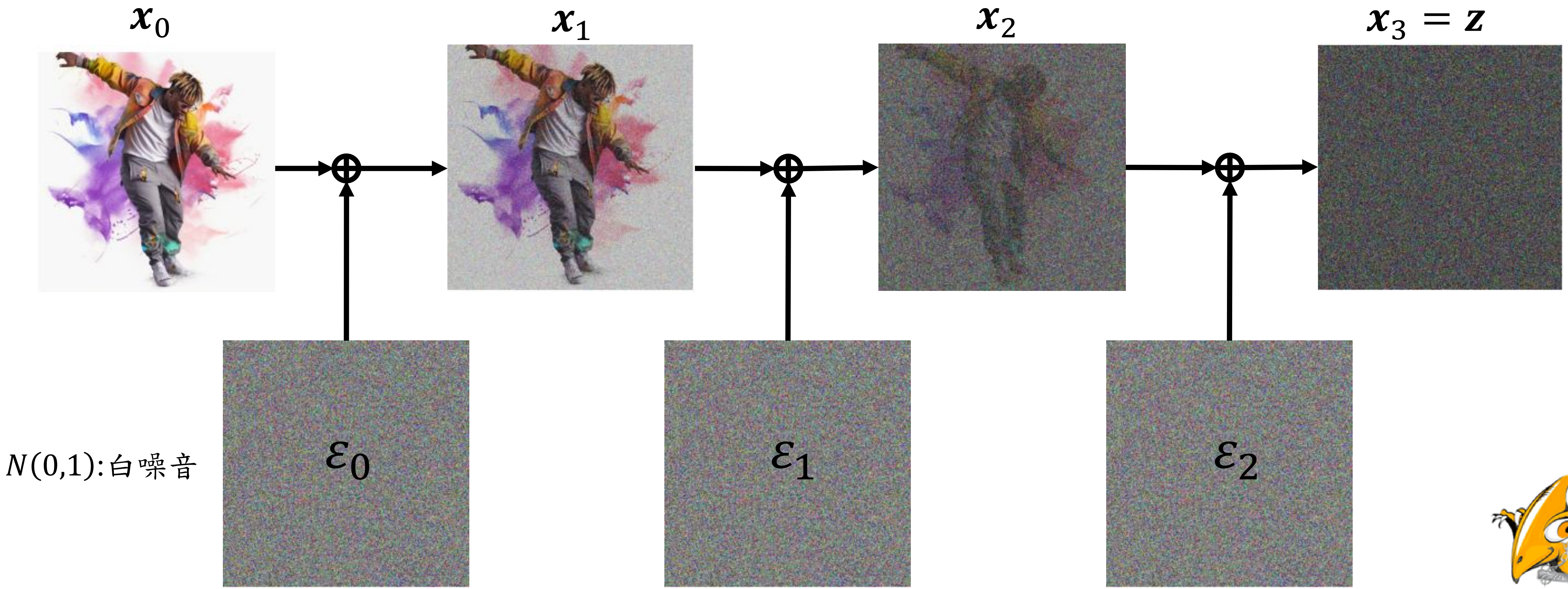


https://arxiv.org/pdf/2208.11970

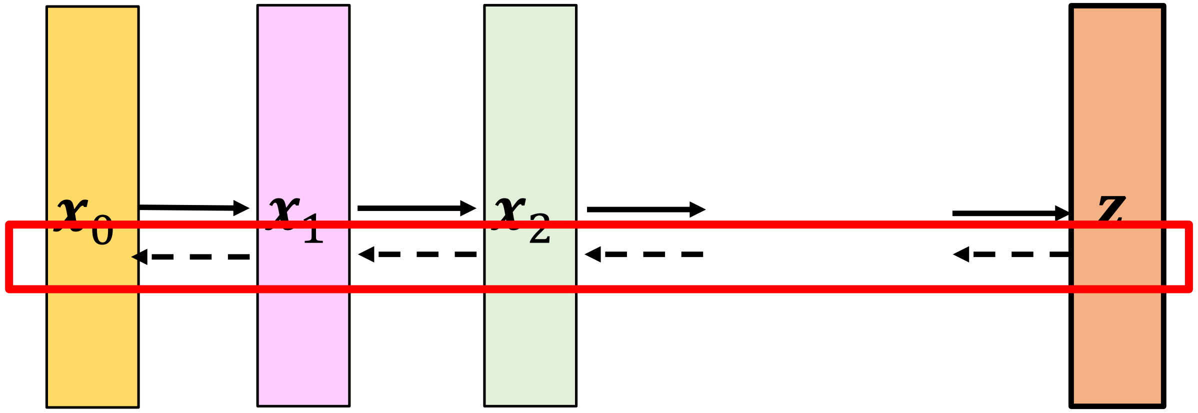
Forward diffusion process
(模糊化程序)



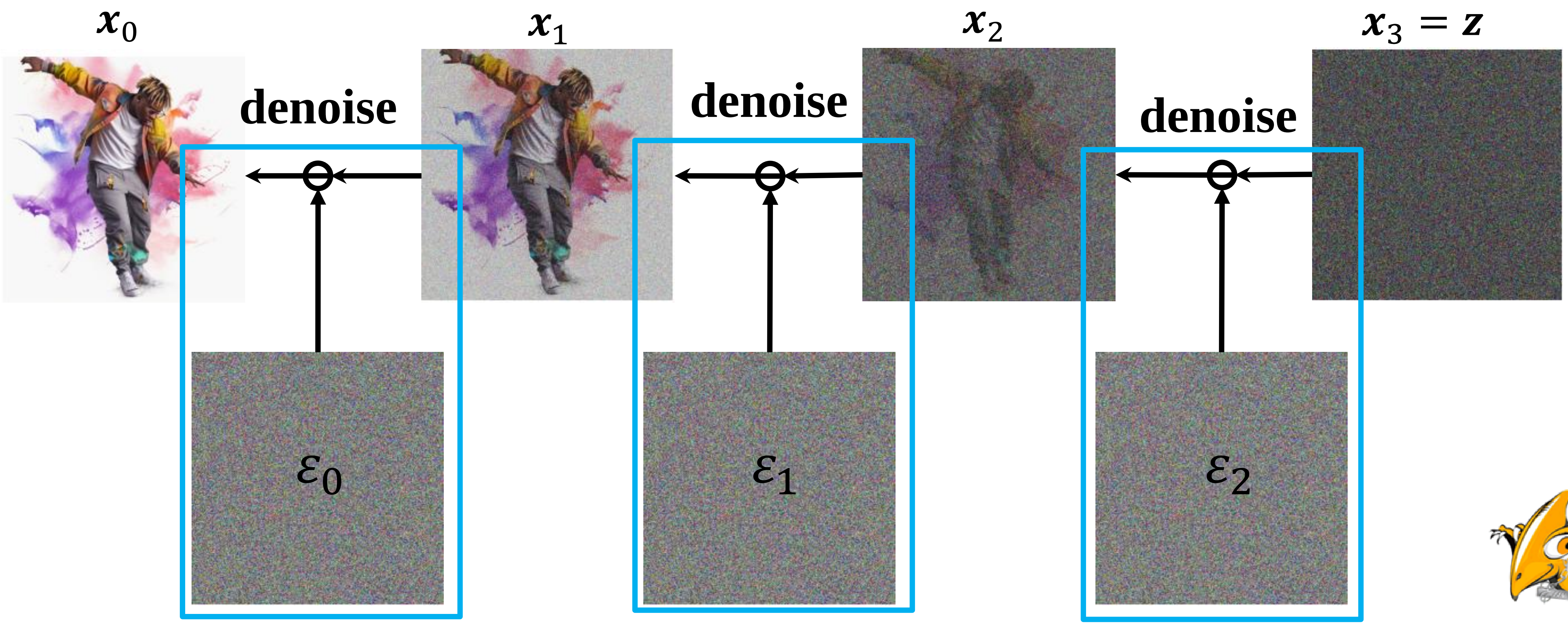
Gradually add Gaussian noise and then reverse



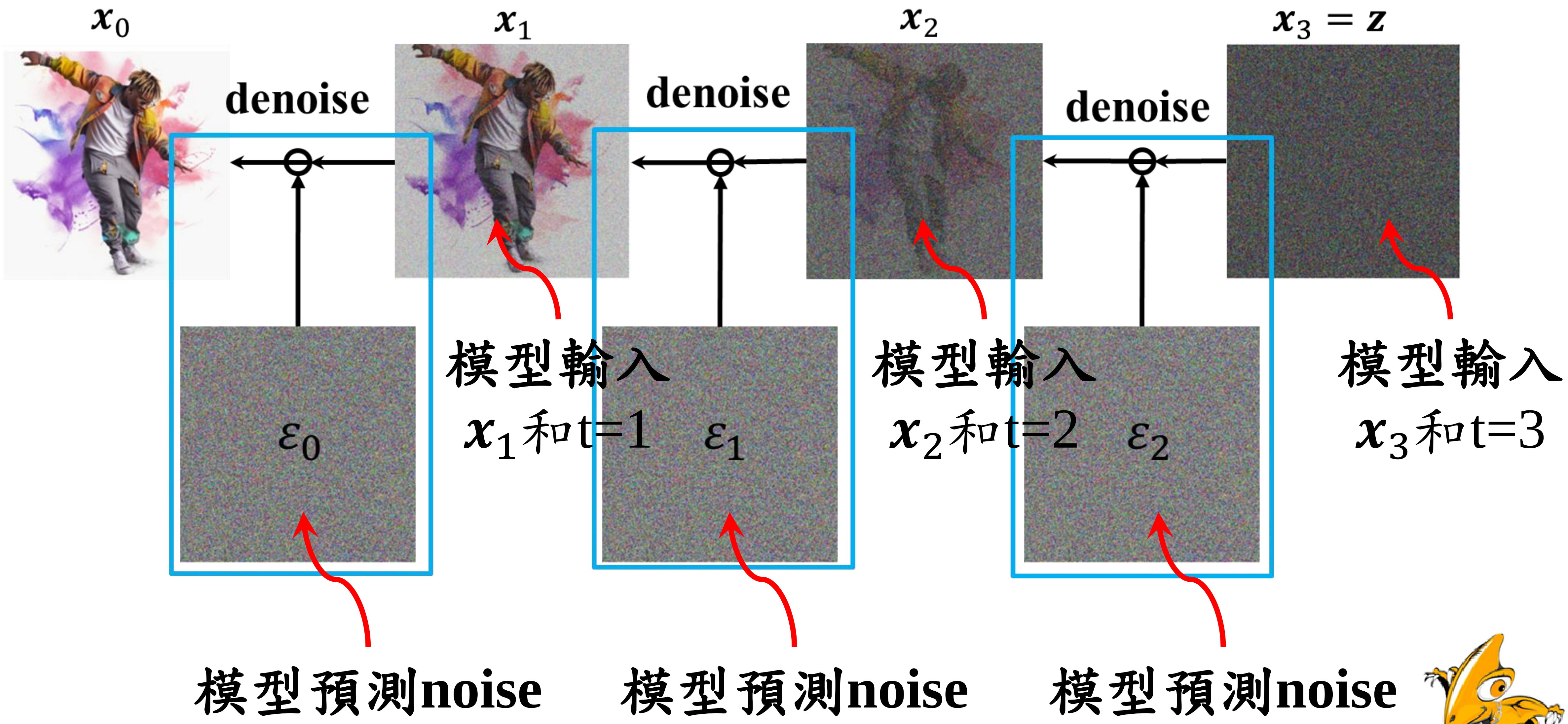
圖片生成
Reverse diffusion process
(逆模糊化程序)



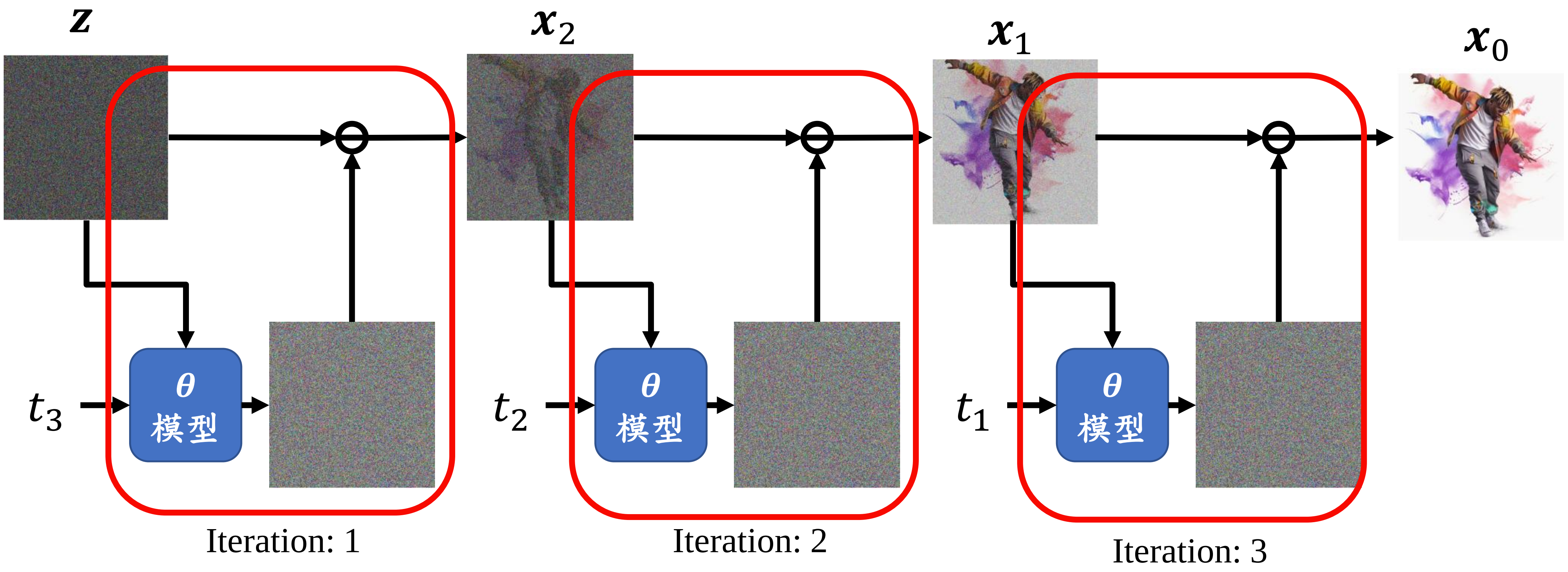
Gradually add Gaussian noise and then reverse



圖片生成:
Reverse diffusion process (逆模糊化程序)

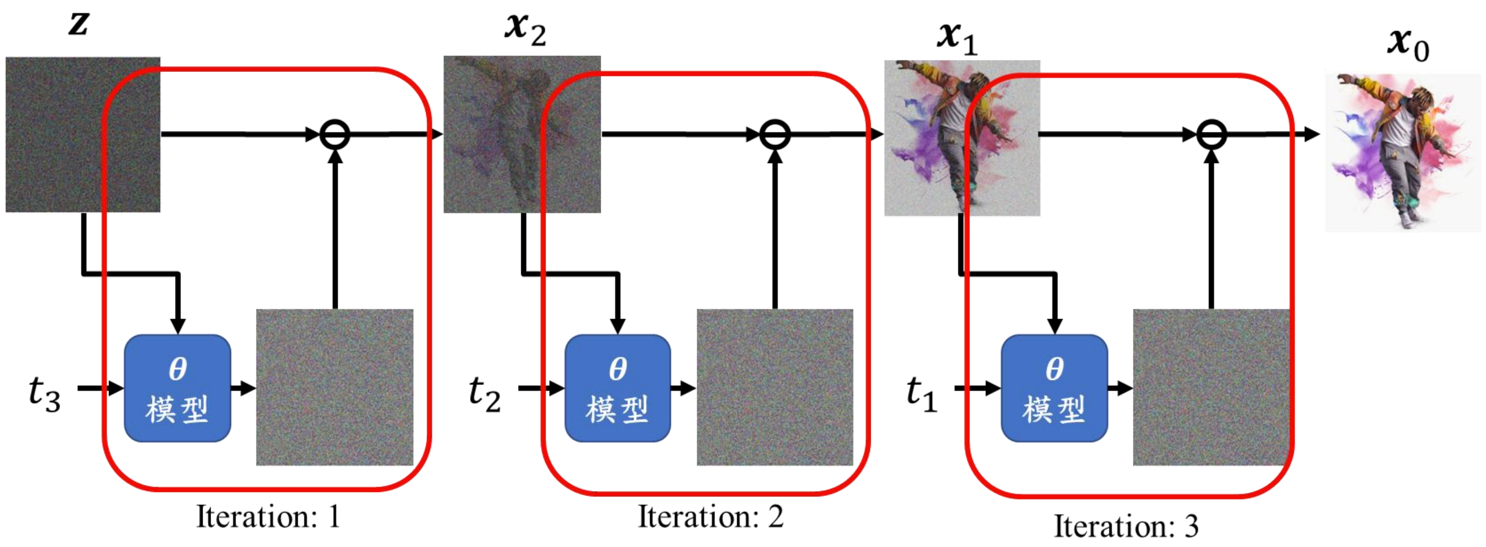


圖片生成:
Reverse diffusion process (逆模糊化程序)



每一步模型(θ) → 預測noise

圖片生成:
Reverse diffusion process (逆模糊化程序)



θ: 模型 → 預測noise

Iteration: 1

$$\hat{\varepsilon}_2 = \theta(x_3 = z, t = 3)$$

$$x_2 = x_3 - \beta_2 \hat{\varepsilon}_2$$

$$\Downarrow$$

Iteration: 2

$$\hat{\varepsilon}_1 = \theta(x_2, t = 2)$$

$$x_1 = x_2 - \beta_1 \hat{\varepsilon}_1$$

$$\Downarrow$$

Iteration: 3

$$\hat{\varepsilon}_0 = \theta(x_2, t = 2)$$

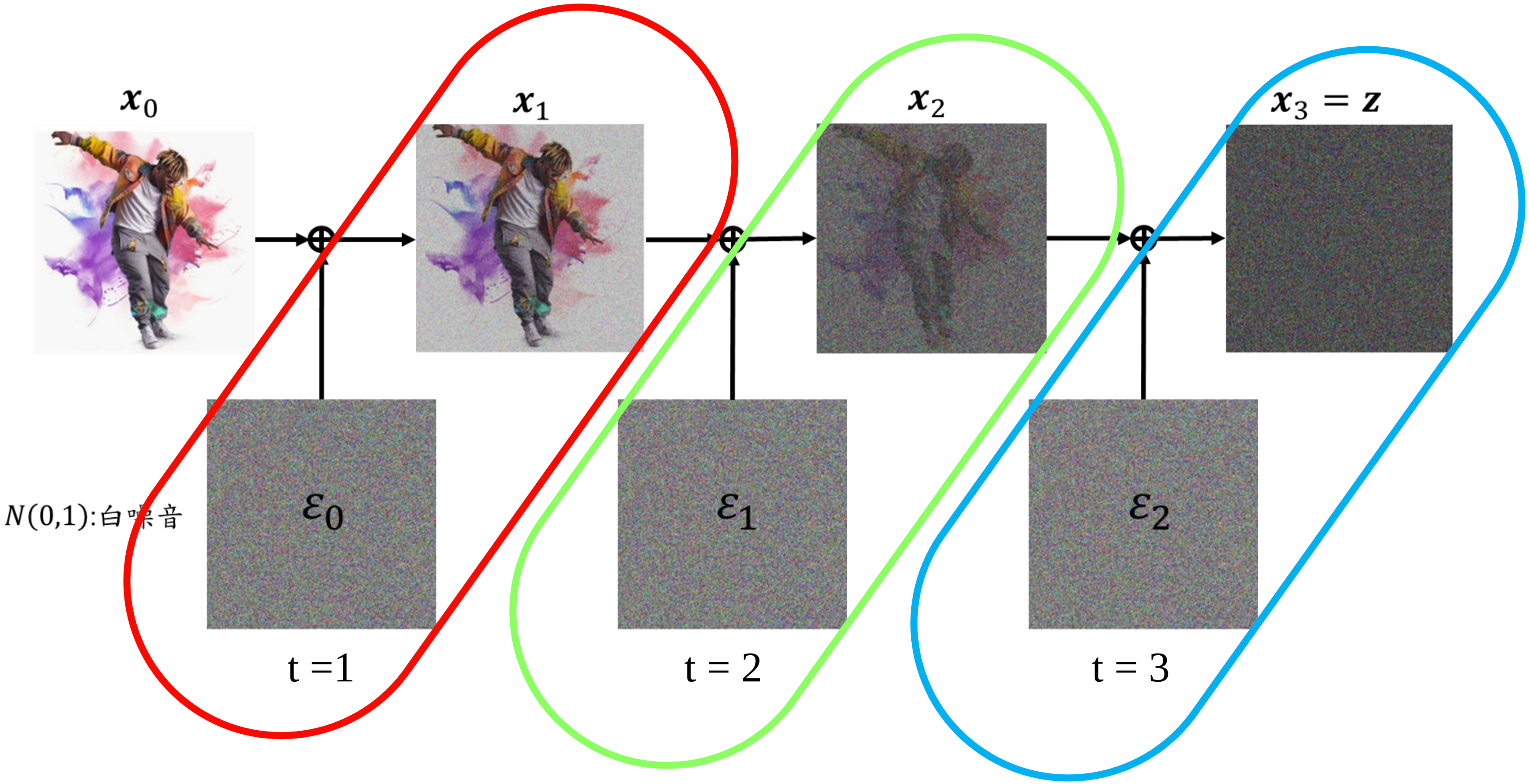
$$x_0 = x_1 - \beta_0 \hat{\varepsilon}_1$$

$\beta_0, \beta_1, \beta_2$: noise係數，訓練前就給定。

如何取得訓練資料訓練模型

如何訓練模型(θ) → 預測noise

Unsupervised learning



訓練資料產生

X→Y

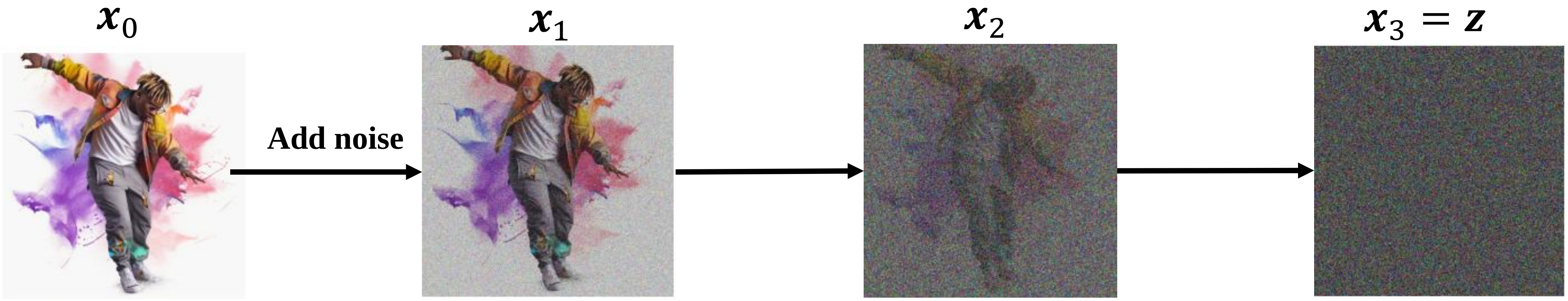
$(x_1, t = 1) \rightarrow \varepsilon_0$

$(x_2, t = 2) \rightarrow \varepsilon_1$

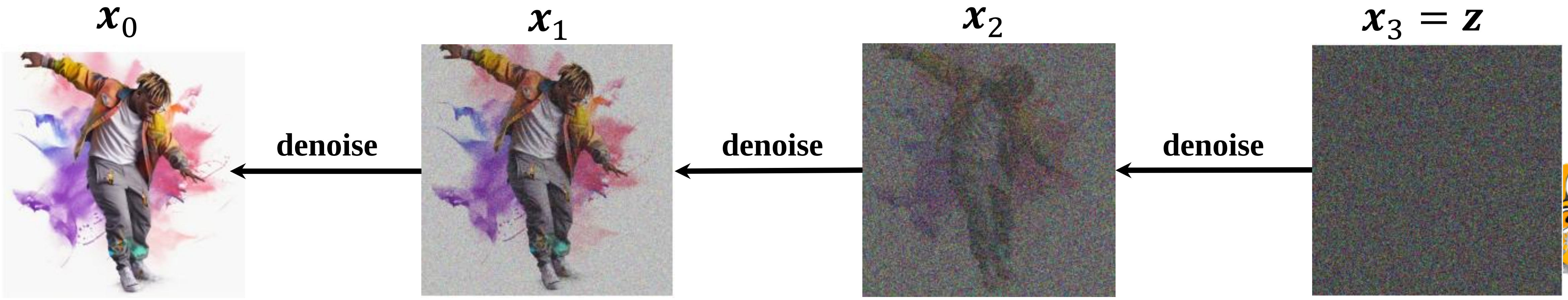
$(x_3, t = 3) \rightarrow \varepsilon_2$

Diffusion

Forward diffusion process



Reverse diffusion process



DDPM

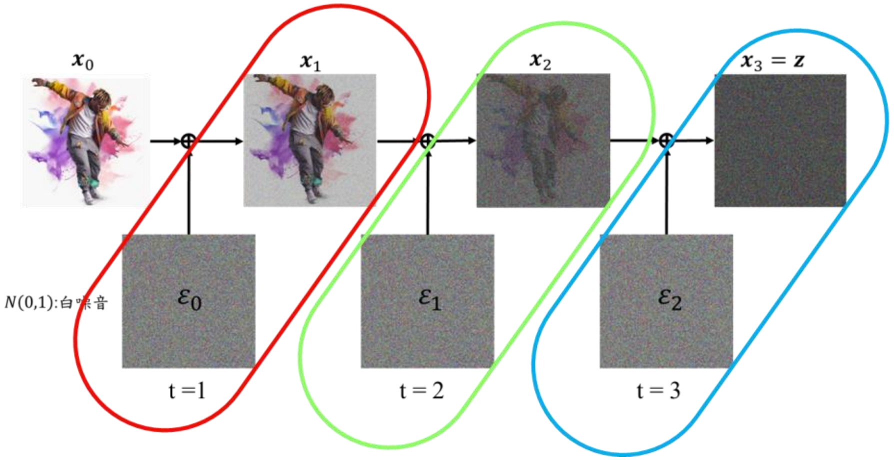
Denoising diffusion probabilistic models (**DDPM**; Ho et al. 2020).

Algorithm 1 Training

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

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **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** $\mathbf{x}_0$



訓練資料產生
X→Y

$\boxed{(x_1, t = 1) \rightarrow \varepsilon_0}$

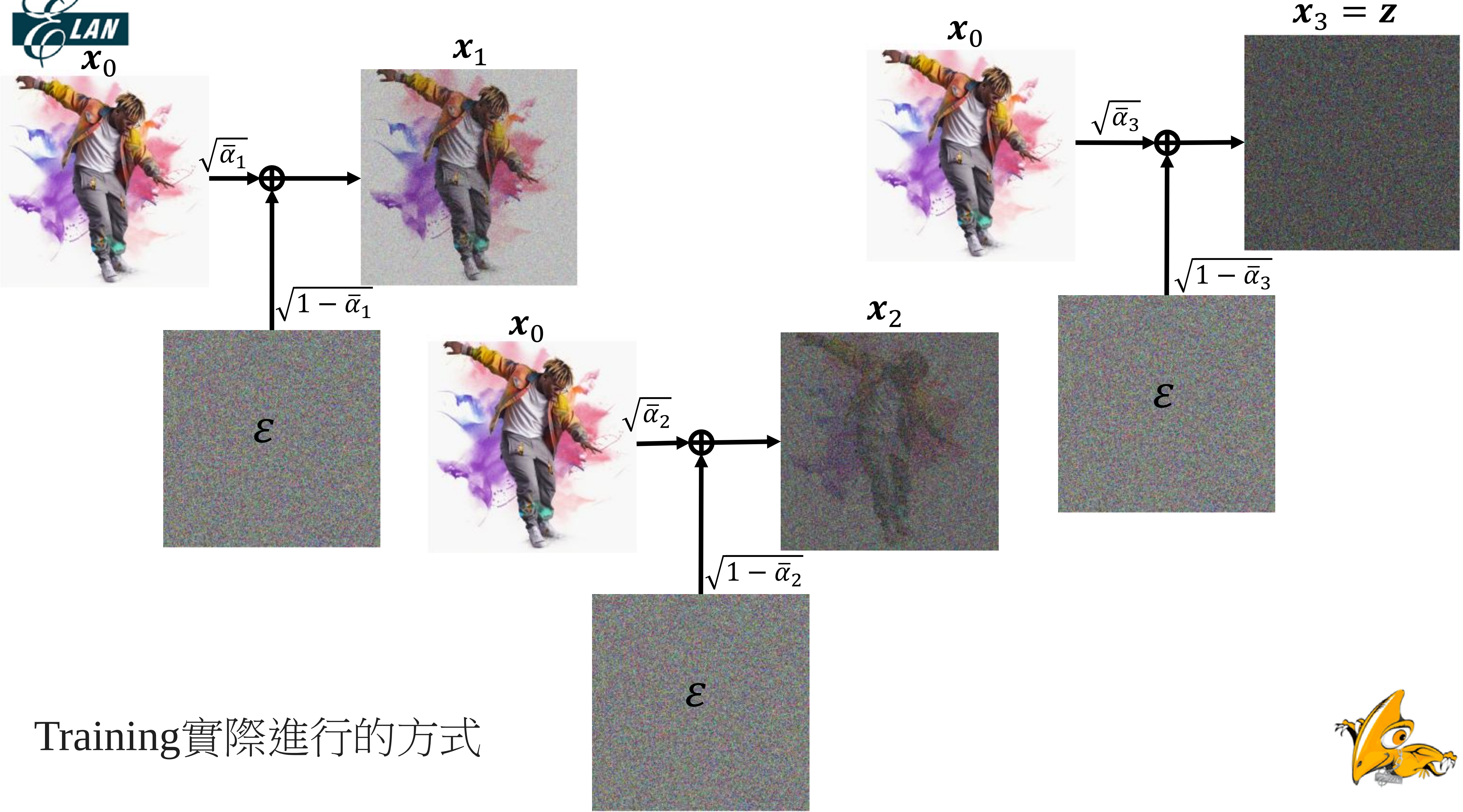
$\boxed{(x_2, t = 2) \rightarrow \varepsilon_1}$

$\boxed{(x_3, t = 3) \rightarrow \varepsilon_2}$

q: 真實資料分布
x_0:真實資料(乾淨的圖片)
ϵ: 白噪音(Target noise)
$\boxed{\epsilon_\theta\text{:模型預測的白噪音}}$

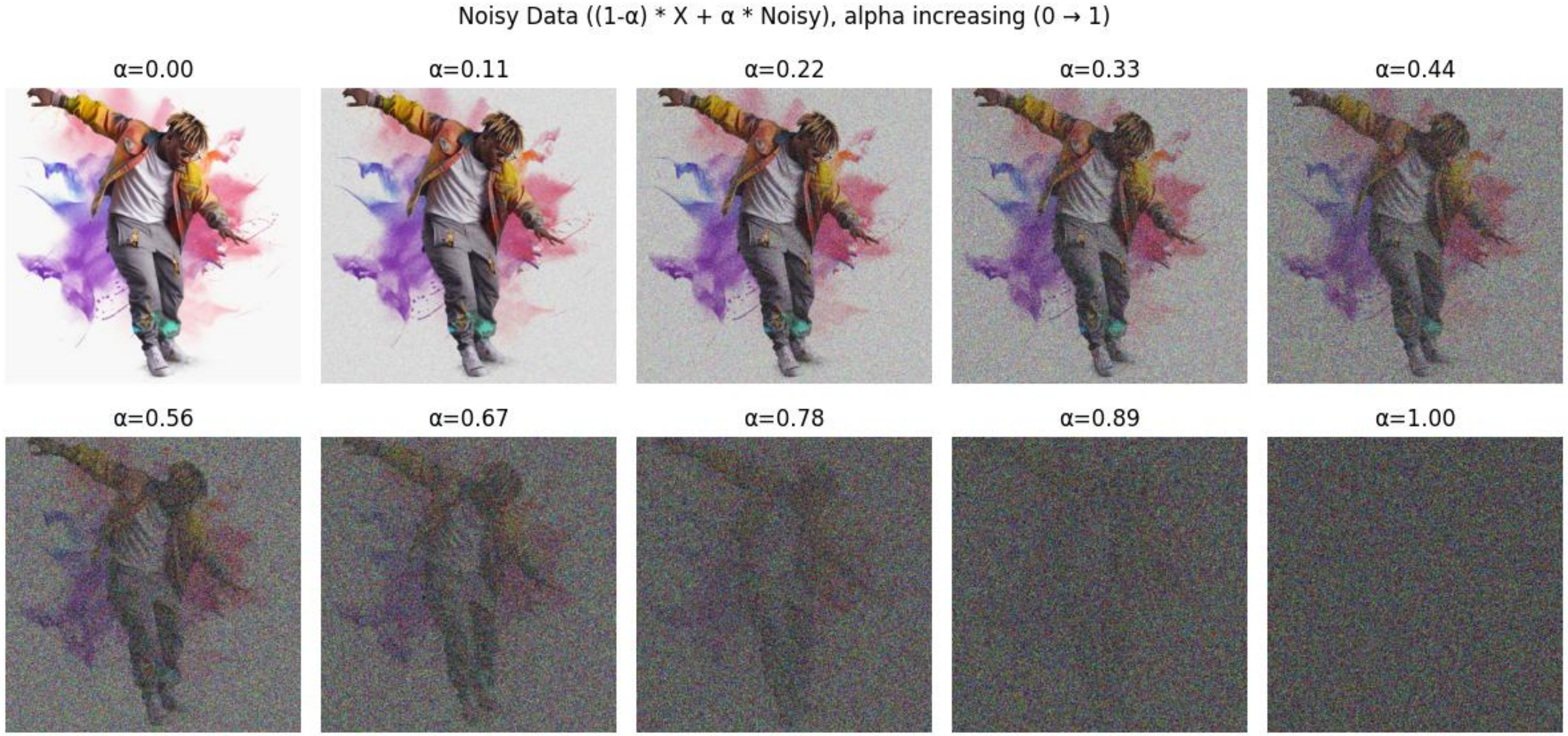
$\bar\alpha_1, \bar\alpha_1, \ldots \bar\alpha_T$: 越來越小

$\boxed{x_t = \sqrt{\bar\alpha_t}x_0 + \sqrt{1-\bar\alpha_t}\epsilon}$: 加了noise後的圖片
(後面在解釋為什麼是x_0)



Training實際進行的方式

Forward diffusion process (模糊化程序)



$$q(x_1|x_0) = N(x_1;\ \sqrt{1-\beta_1}x_0, \beta_1\mathbf{I})$$
$$q(x_2|x_1) = N(x_2;\ \sqrt{1-\beta_2}x_1, \beta_2\mathbf{I})$$
$$\dots$$
$$q(x_T|x_{T-1}) = N(x_T;\ \sqrt{1-\beta_T}x_{T-1}, \beta_T\mathbf{I})$$

β稱為 variance schedule，介於0~1(也可以學來，也可以是固定值(DDPM是固定值))

$$q(x_t|x_{t-1}) = N\left(x_{t-1};\ \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}\right), t = 1,2,\dots,T$$
$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

Reverse diffusion process (逆模糊化程序)

逆模糊化我們定義成：

$$p_\theta(x_{t-1}|x_t)$$

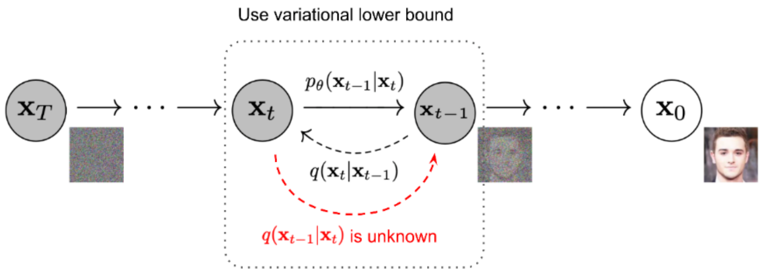
很難去估計$q(x_t|x_{t-1})$，但我們知道

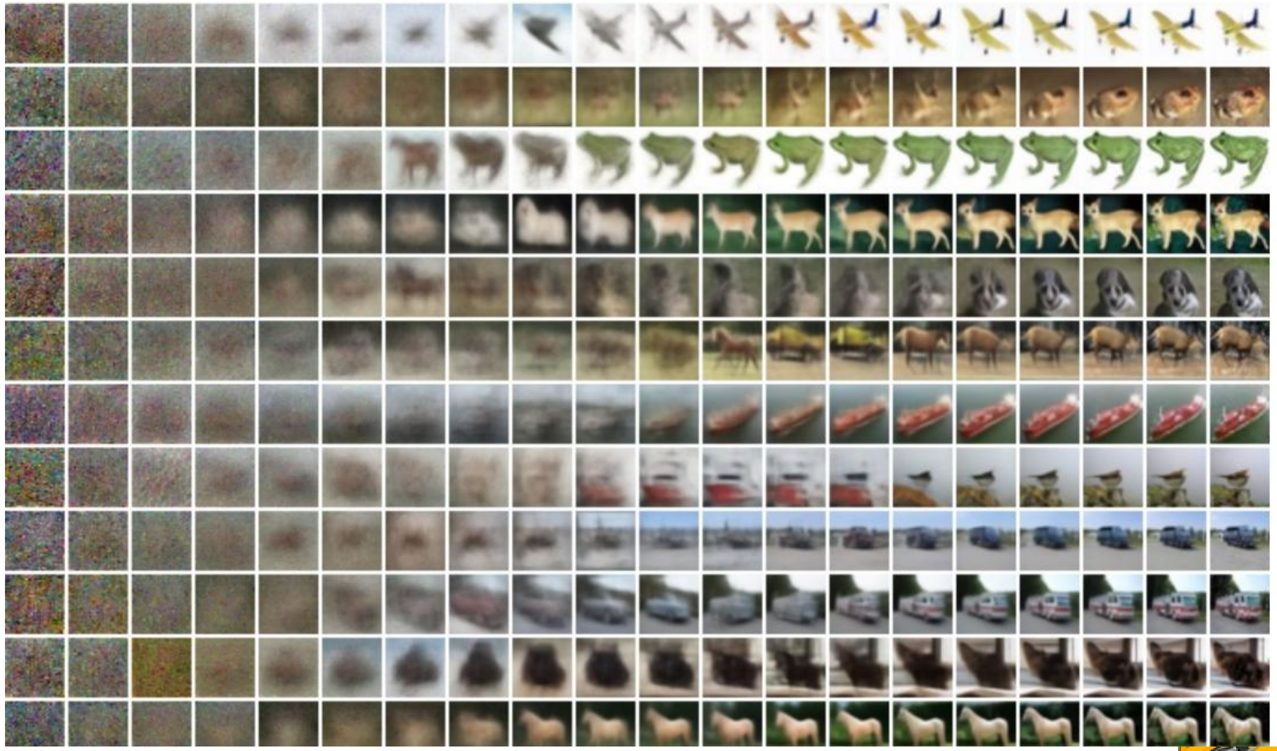
$$q(x_t|x_{t-1}) = N\left(x_{t-1};\ \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}\right)$$

模型p_θ(服從常態分佈)去估計條件機率來表示reverse diffusion process

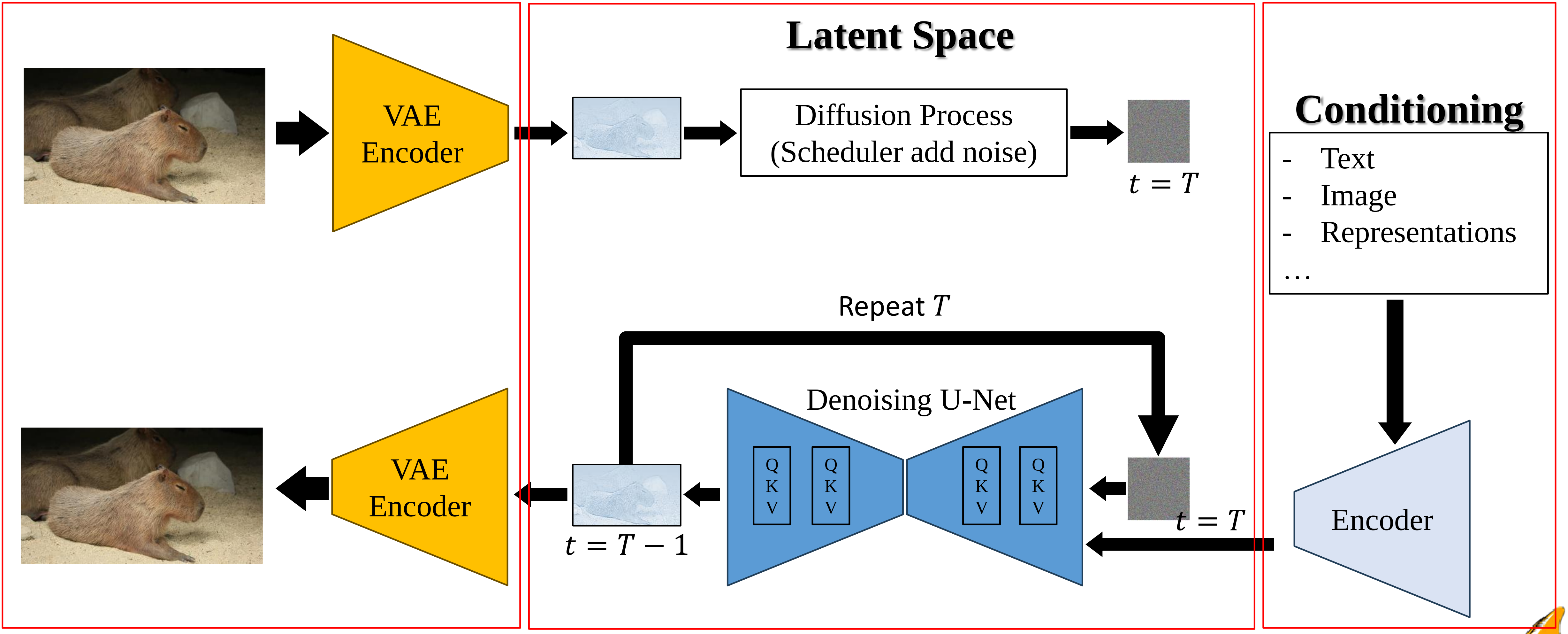
$$p_\theta(x_{t-1}|x_t) \sim N(x_{t-1}; \mu_\theta(x_t,t); \Sigma_\theta(x_t,t))$$

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

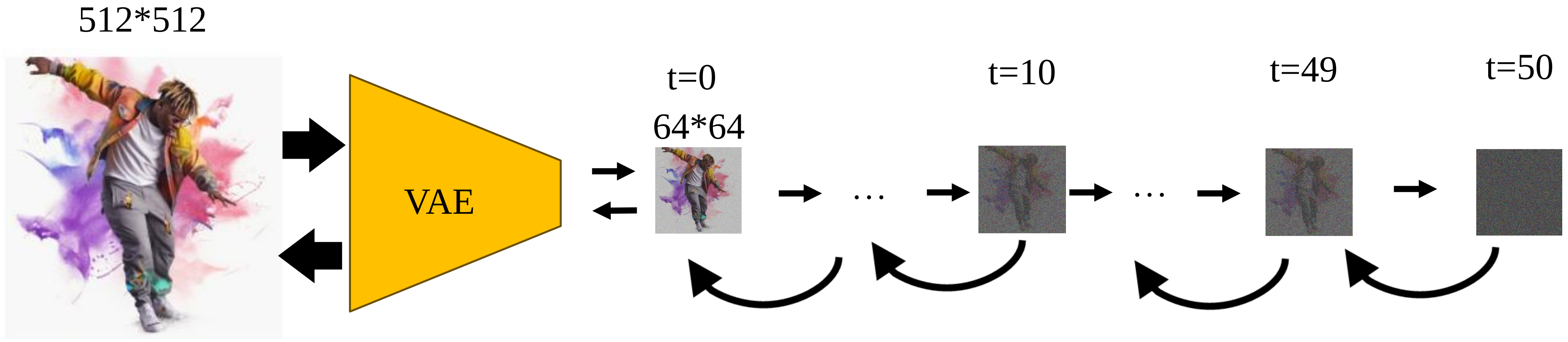




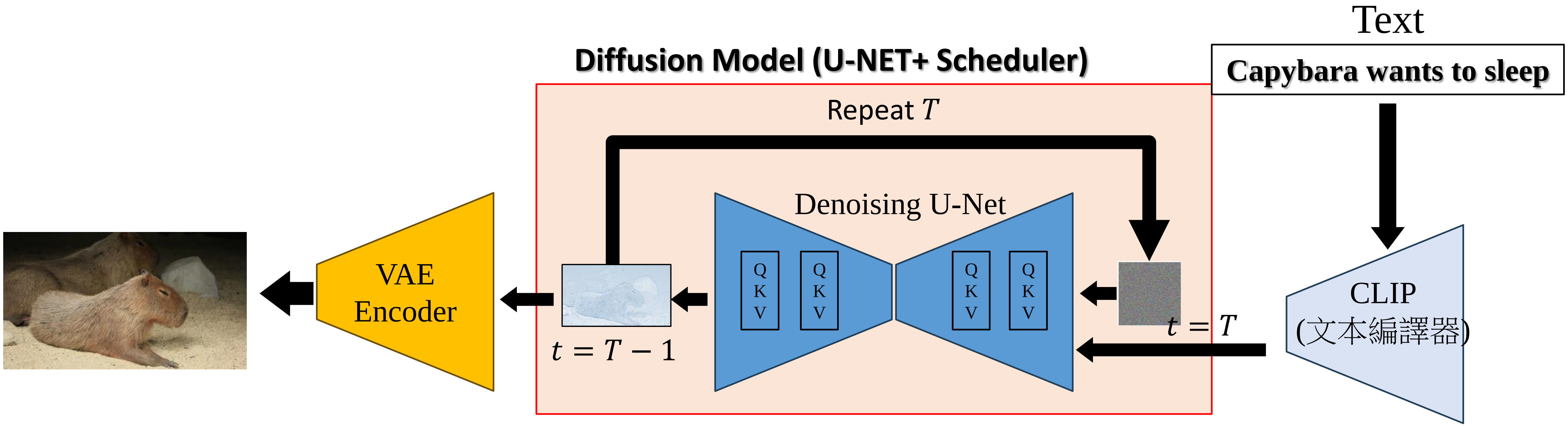
Stable Diffusion



Latent Diffusion Model



文生圖(Text-to-Image, T2I)



1. **Text Encoder**
2. **Diffusion model**
3. **VAE Encoder**

Code example