深度学习

作者：Calvin
QQ: 179209347
Mail: 179209347@qq.com

介绍

笔记简介：

- 面向对象：深度学习初学者
- 依赖课程：**线性代数，统计概率**，优化理论，图论，离散数学，微积分，信息论

知乎专栏：

https://zhuanlan.zhihu.com/p/693738275

Github & Gitee 地址：

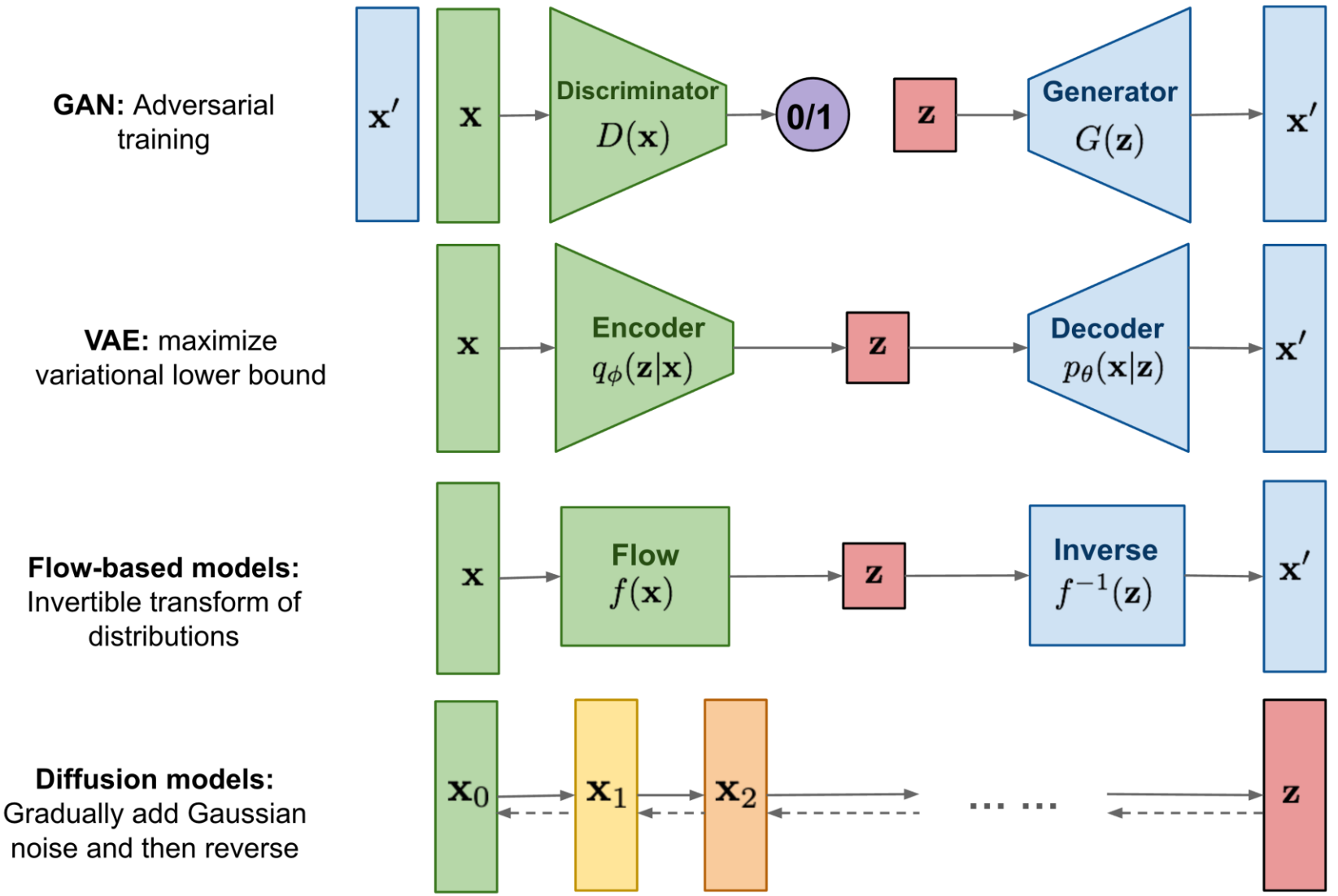
https://github.com/mymagicpower/AIAS/tree/main/deep_learning

https://gitee.com/mymagicpower/AIAS/tree/main/deep_learning

*** 版权声明：**

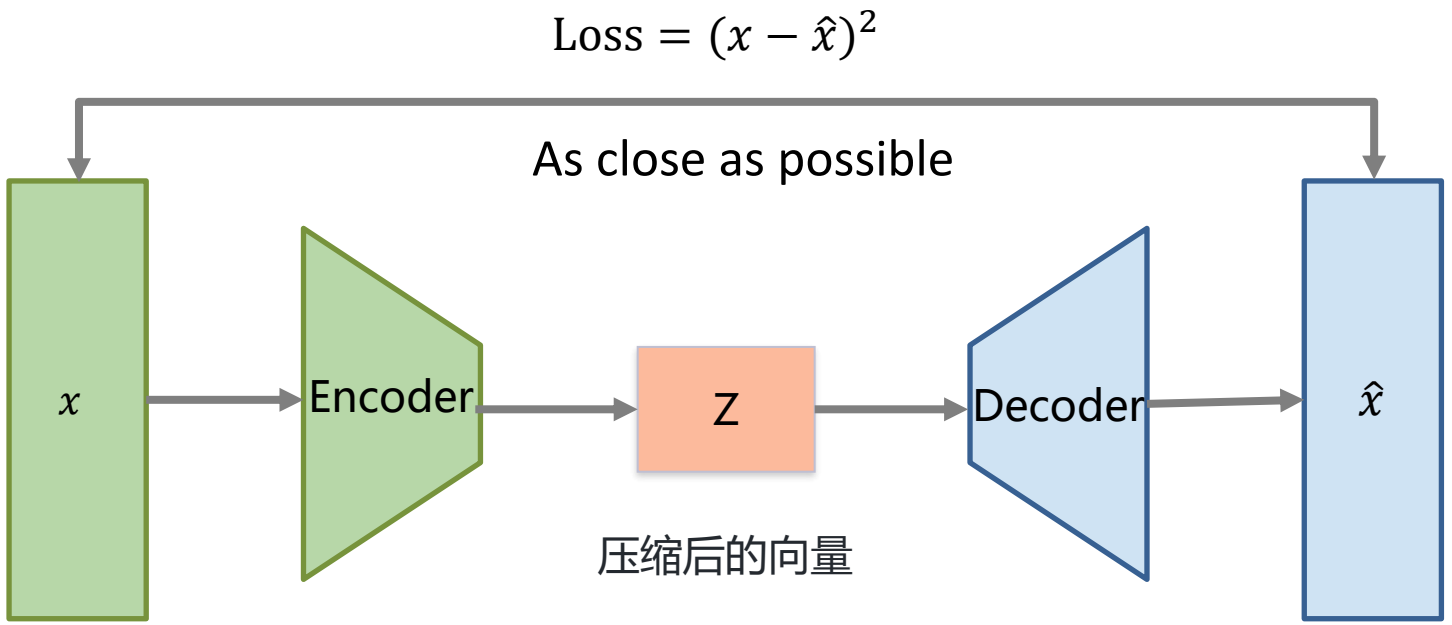
- 仅限用于个人学习
- 禁止用于任何商业用途

生成模型对比



自动编码器

自动编码器（Auto-Encoder）是一种无监督学习的神经网络模型，用于学习数据的有效表示。它由两部分组成：编码器（Encoder）和解码器（Decoder）。编码器将输入数据压缩成潜在空间（latent space）中的编码或隐藏表示，而解码器则将这个编码映射回原始数据空间。



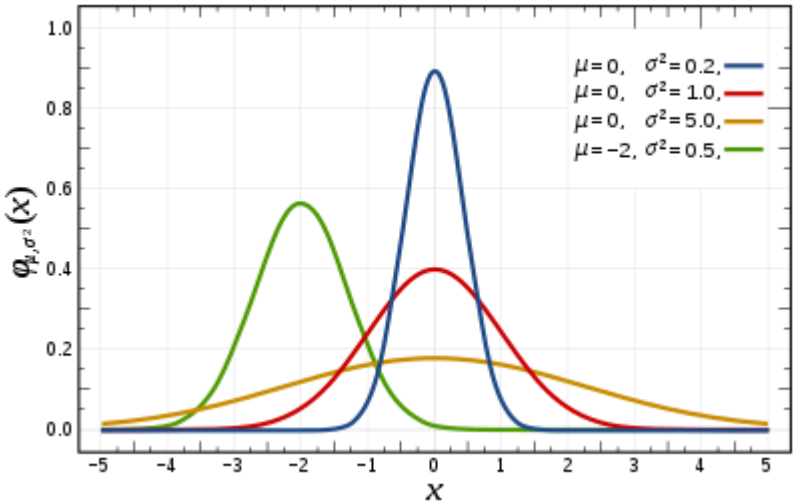
正态分布 – 一元正态分布

正态分布或**高斯分布**是实值随机变量的一种连续概率分布。其概率密度函数的一般形式为：

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

参数 μ 是分布的均值或期望，参数 σ 是标准差。方差是 σ^2 。

符号：$X \sim \mathcal{N}(\mu, \sigma^2)$



标准正态分布

最简单的正态分布称为标准正态分布或单位正态分布。这是特殊情况 μ=0 和 σ=1，它由概率密度函数（或密度） 描述：

$$\varphi(z) = \frac{e^{-z^2/2}}{\sqrt{2\pi}}$$

符号：$X \sim \mathcal{N}(0,1)$

正态分布 – 独立多元正态分布

假设n个变量互不相关，服从正态分布（维度不相关多元正态分布），根据联合概率密度公式有：

$$f(x) = p(x_1, x_2 \ldots x_n) = p(x_1)p(x_2) \ldots p(x_n) = \frac{1}{(\sqrt{2\pi})^n \sigma_1 \sigma_2 \cdots \sigma_n} e^{-\frac{(x_1-\mu_1)^2}{2\sigma_1^2} - \frac{(x_2-\mu_2)^2}{2\sigma_2^2} \cdots - \frac{(x_n-\mu_n)^2}{2\sigma_n^2}}$$

令：$z^2 = \frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2} \cdots + \frac{(x_n-\mu_n)^2}{\sigma_n^2}, \sigma_z = \sigma_1 \sigma_2 \cdots \sigma_n$

则有：

$$f(z) = \frac{1}{(\sqrt{2\pi})^n \sigma_z} e^{-\frac{z^2}{2}}$$

转换成矩阵形式：

$$z^2 = z^{\mathrm{T}} z = [x_1 - \mu_1, x_2 - \mu_2, \cdots, x_n - \mu_n] \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \cdots & 0 \\ \vdots & \cdots & \cdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sigma_n^2} \end{bmatrix} [x_1 - \mu_1, x_2 - \mu_2, \cdots, x_n - \mu_n]^{\mathrm{T}}$$

正态分布 – 独立多元正态分布

各个维度的变量： $x = [x_1, x_2, \cdots, x_n]^{\mathrm{T}}$
各个维度的均值： $E(x) = \mu_x = [\mu_1, \mu_2, \cdots, \mu_n]^{\mathrm{T}}$,
各个维度的方差： $\sigma(x) = [\sigma_1, \sigma_2, \cdots, \sigma_n]^{\mathrm{T}}$

则有：

$$x - \mu_x = [x_1 - \mu_1, x_2 - \mu_2, \cdots, x_n - \mu_n]^{\mathrm{T}}$$

Σ 代表变量 X 的协方差矩阵， i行j列的元素值表示 x_i 与 x_j 的协方差。

$$\sum = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \cdots & \cdots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix}$$

由于各变量之间是相互独立的，所以只有对角线上 (i = j) 存在元素，其他都为 0，即 **Σ** 是一个对角阵，且 x_i 与它本身的协方差就等于方差。根据对角矩阵的性质，它的逆矩阵：

$$\Sigma^{-1} = \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \cdots & 0 \\ & \cdots & \cdots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sigma_n^2} \end{bmatrix}$$

正态分布 – 独立多元正态分布

由于对角矩阵的行列式等于对角元素的乘积，所以有：

$$\sigma_z = |\sum|^{\frac{1}{2}} = \sigma_1\sigma_2.\ldots\ldots\sigma_n$$

$$z^{\mathrm{T}}z = (x-\mu_x)^{\mathrm{T}}\sum{}^{-1}(x-\mu_x)$$

多元高斯正态分布形式为：

$$f(z) = \mathcal{N}(\mathbf{x}|\mu,\mathbf{\Sigma}) = \frac{1}{(\sqrt{2\pi})^n\sigma_z}e^{-\frac{z^2}{2}} = \frac{1}{(\sqrt{2\pi})^n|\sum|^{\frac{1}{2}}}e^{-\frac{(x-\mu_x)^{\mathrm{T}}(\sum)^{-1}(x-\mu_x)}{2}}$$

其中 μ 是一个 n 维均值向量， $\mathbf{\Sigma}$ 是 n x n 的协方差矩阵，并且 $|\mathbf{\Sigma}|$ 表示 $\mathbf{\Sigma}$ 的行列式。

符号：$X \sim \mathcal{N}(\mu,\mathbf{\Sigma})$

多元标准正态分布：

$$f(z) = \mathcal{N}(\mathbf{x}|\mu,\mathbf{\Sigma}) = \frac{1}{(\sqrt{2\pi})^n}e^{-\frac{x^{\mathrm{T}}x}{2}}$$

符号：$X \sim \mathcal{N}(\mathbf{0},\mathbf{I})$

重参数(reparameterization trick)

重参数化技巧（Reparameterization Trick）是生成模型中一种关键的方法，它允许我们在优化过程中将样本的随机性转移到一个确定性的模型中，使得我们可以通过标准的梯度下降方法来训练模型。
正态高斯分布的线性变换仍然是正态高斯分布：核心思想是将随机变量的采样过程重新参数化为一个确定性的操作，使得采样过程变得可导。以正态分布为例，我们可以**将从正态分布中采样的操作重新参数化为从一个固定的标准正态分布中采样，然后通过线性变换和平移**来获得我们所需的正态分布采样值。这样，整个采样过程就变成了可导的，可以直接应用于神经网络的反向传播算法中。

数学表示
如果从高斯分布 $z \sim \mathcal{N}(\mu, \sigma^2)$采样一个 z，我们可以将其重参数化为：

$$z = \mu + \sigma \odot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, 1)$$

其中，μ 和 σ 是学习到的参数，ϵ 是从标准正态分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 采样的噪声，$\odot$ 表示逐元素乘法。

N维数学表示
如果从高斯分布 $z \sim \mathcal{N}(\mu, \mathbf{\Sigma})$采样一个 z，我们可以将其重参数化为：

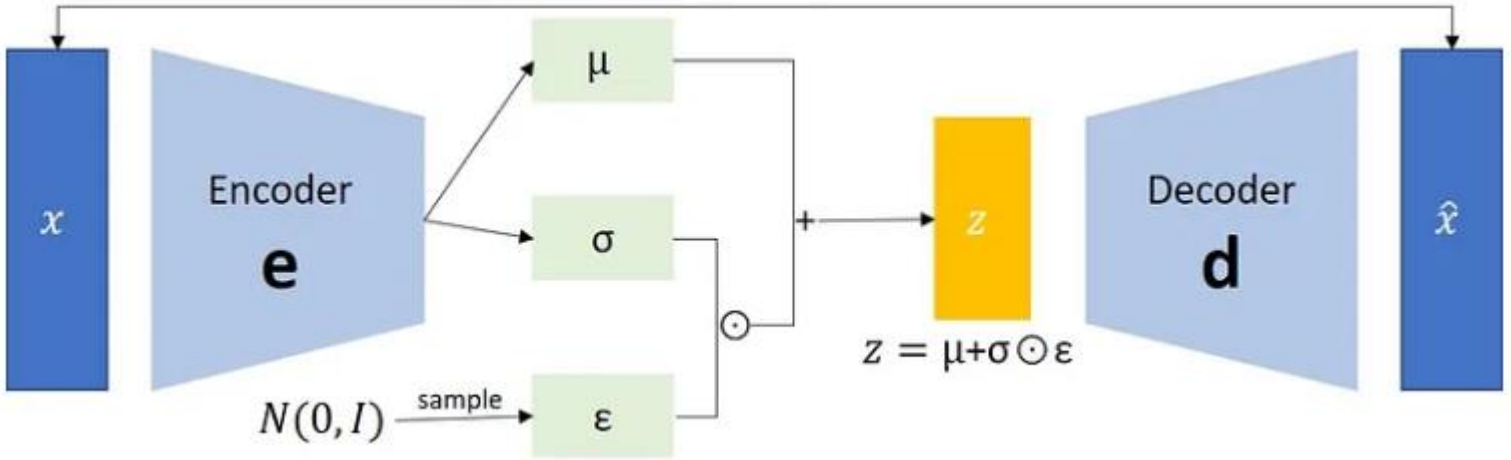
$$z = \mu + \sum^{\frac{1}{2}} \odot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

简写为：$z = \mu + \sigma \odot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

变分自编码器（Variational Autoencoder，VAE）

变分自编码器（Variational Autoencoder，VAE）是一种生成模型，结合了自动编码器（Auto-Encoder）和概率潜变量模型的思想。它可以用于学习数据的潜在表示，并且可以生成与原始数据类似的新样本。

最小化损失1： $\text{Loss} = (x - \hat{x})^2$

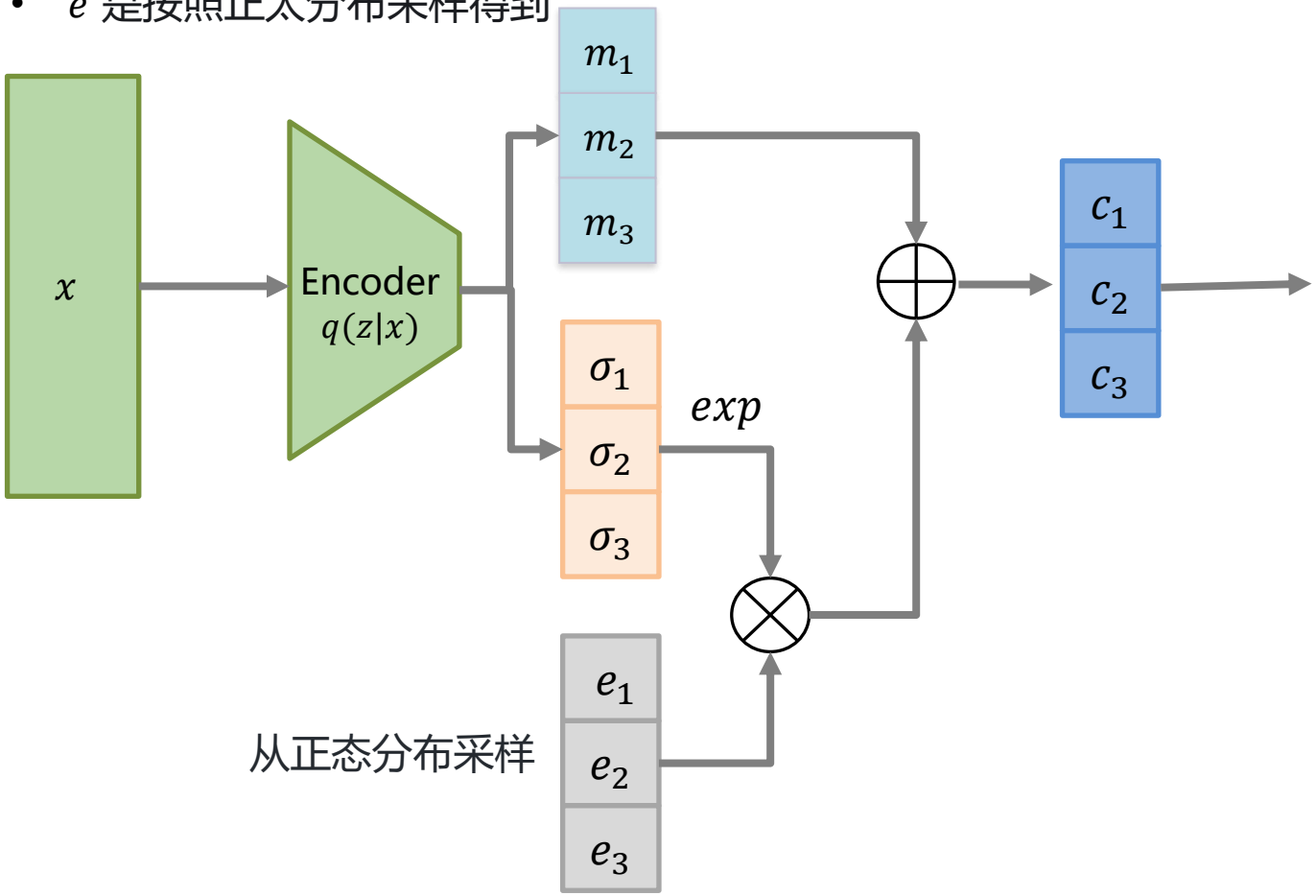


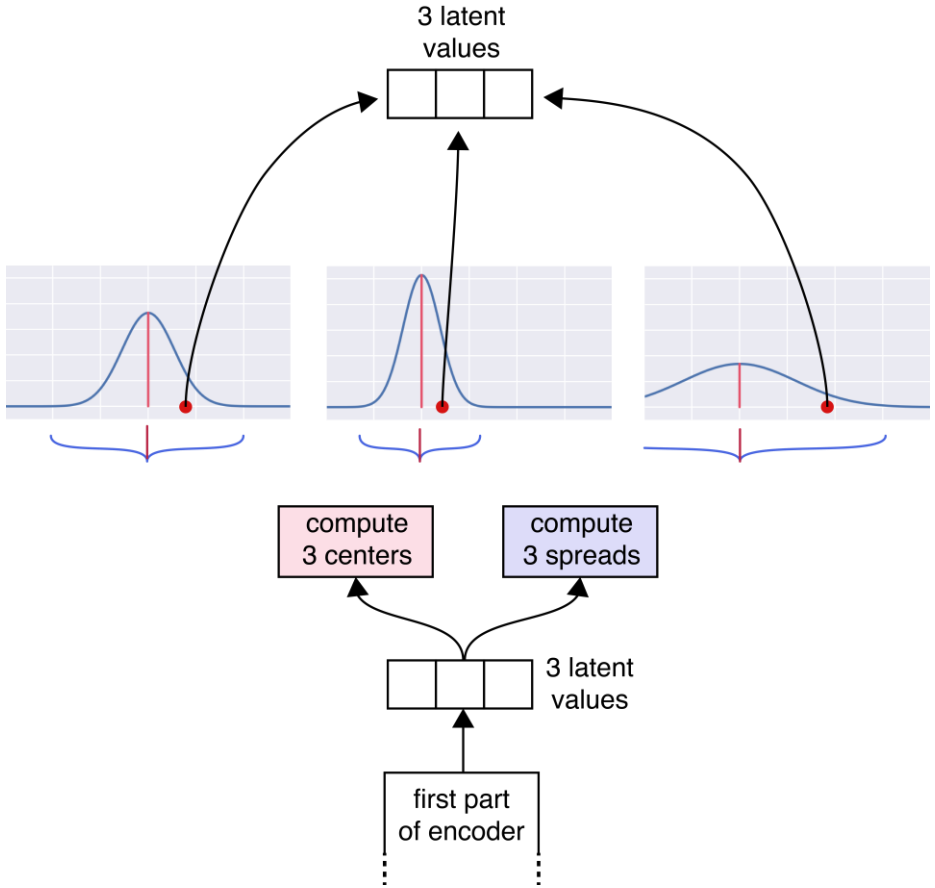
最小化损失2： $\frac{1}{2}\sum_{i=1}^{N}(\exp(\sigma_i) - (1 + \sigma_i) + {\mu_i}^2)$

变分自编码器（VAE）- 编码器

变分自编码器生成的不再是简单的特征，而是一个分布，包括：

- m 是均值
- σ 是方差
- e 是按照正太分布采样得到

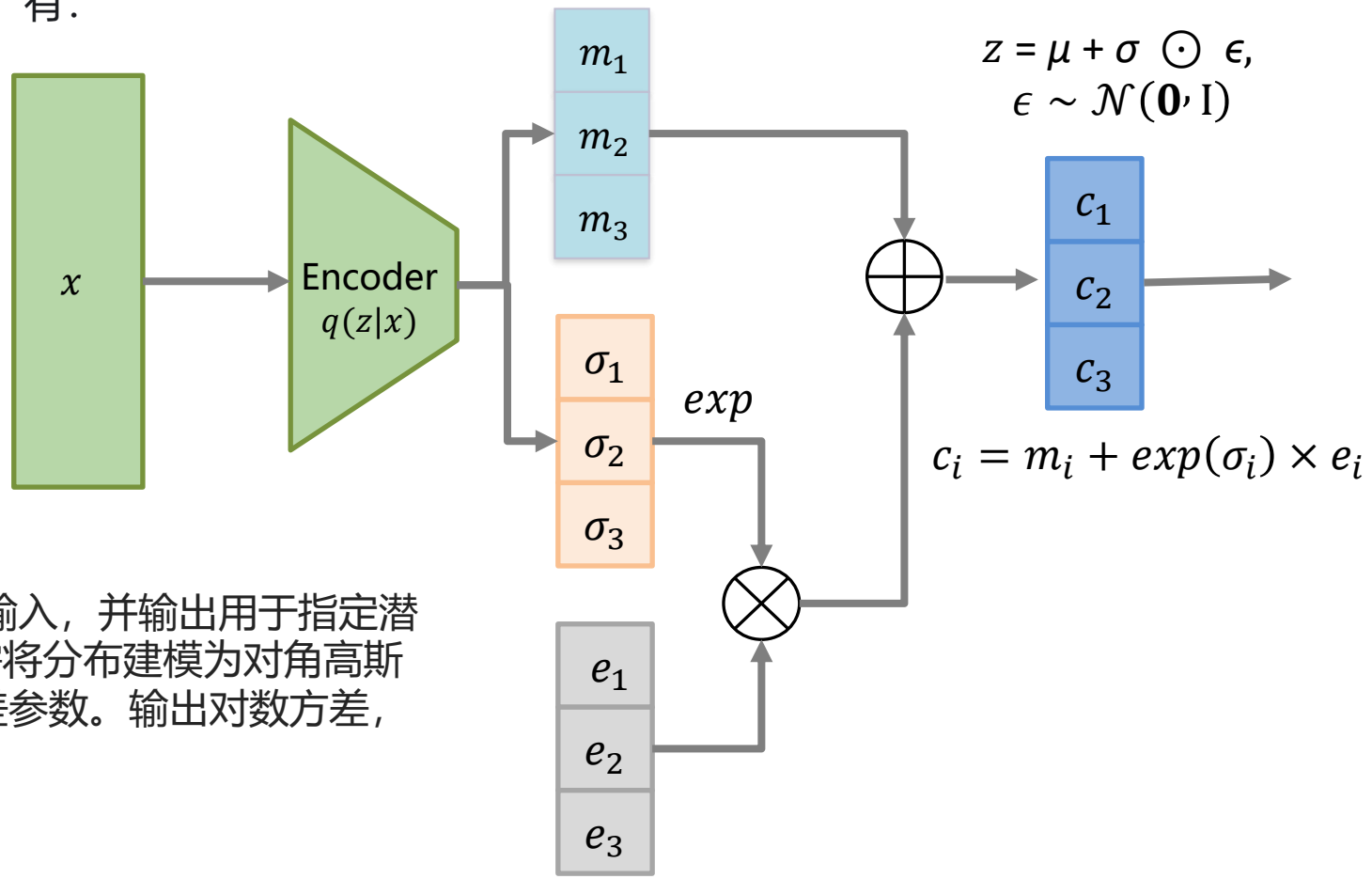




变分自编码器（VAE）- 为解码器生成样本 z

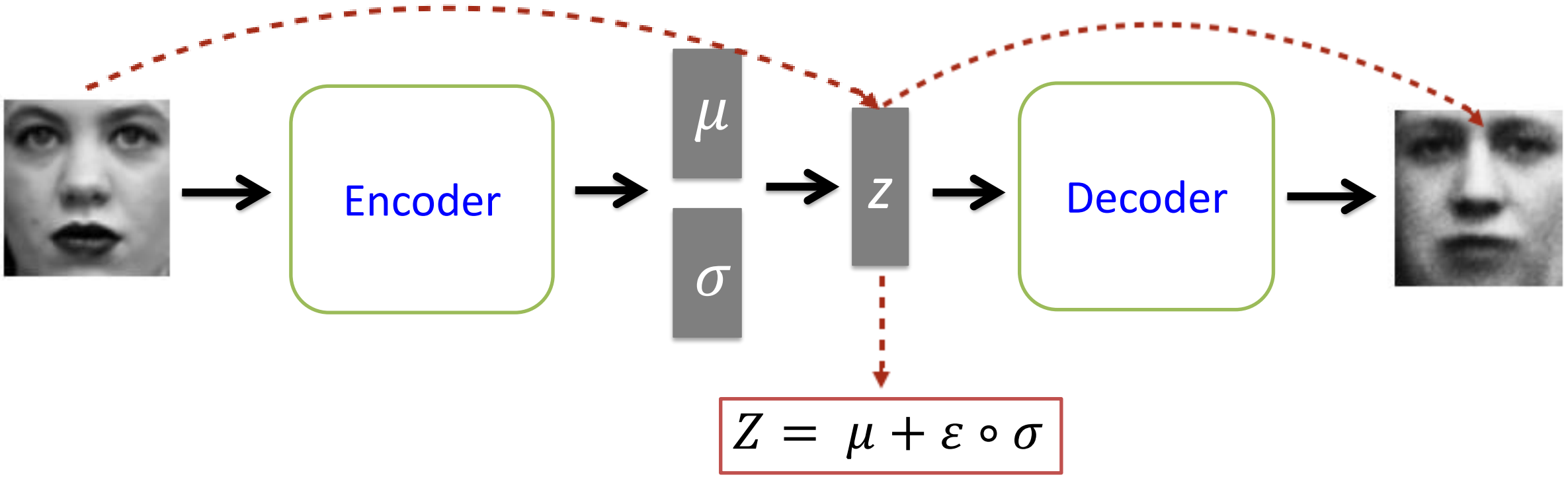
生成的特征，根据公式 $z = \mu + \sigma \odot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathrm{I})$，有：

$$c_i = m_i + exp(\sigma_i) \times e_i$$

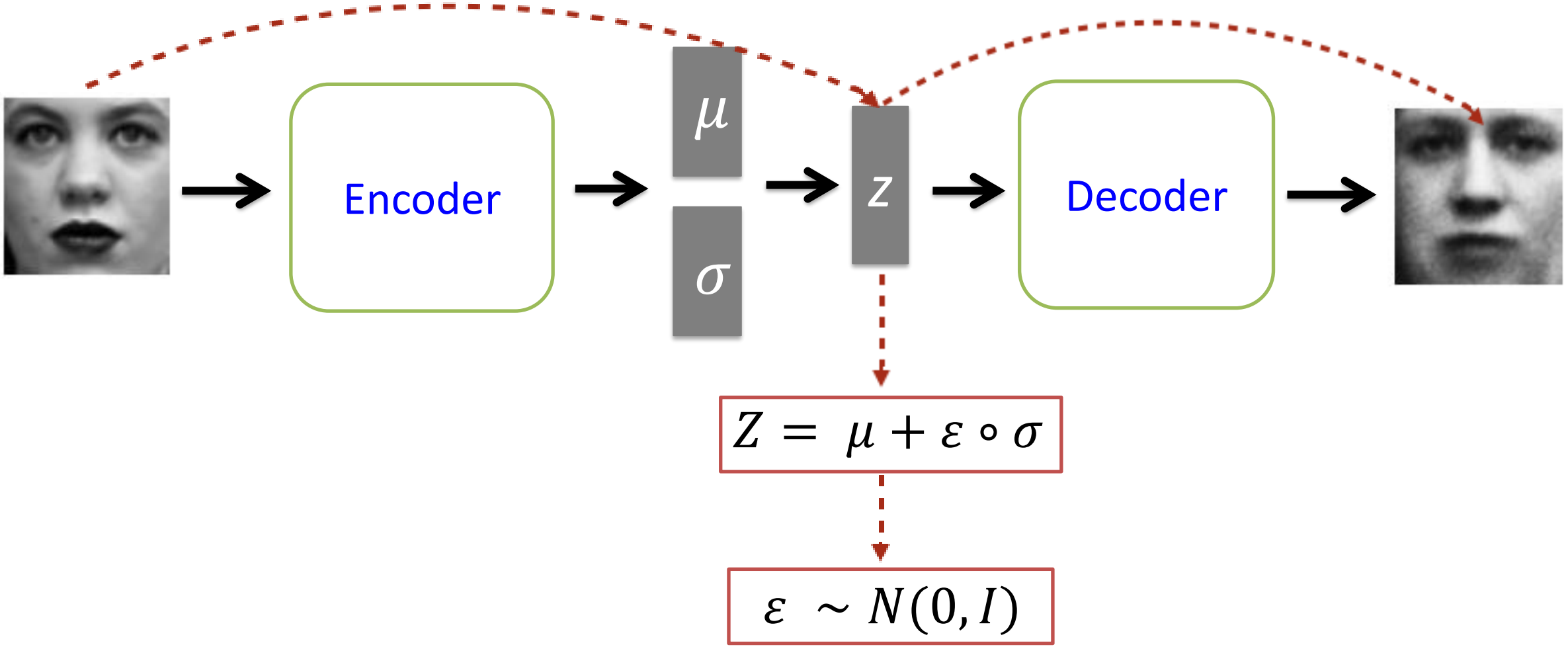


这定义了近似后验分布 $q(z|x)$，其将观测值作为输入，并输出用于指定潜在表示 z 的条件分布的一组参数。在本例中，只需将分布建模为对角高斯分布，网络输出因子化高斯分布的均值和对数方差参数。输出对数方差，而不是直接输出方差，以确保数值稳定性。

变分自编码器（VAE）- 重参化

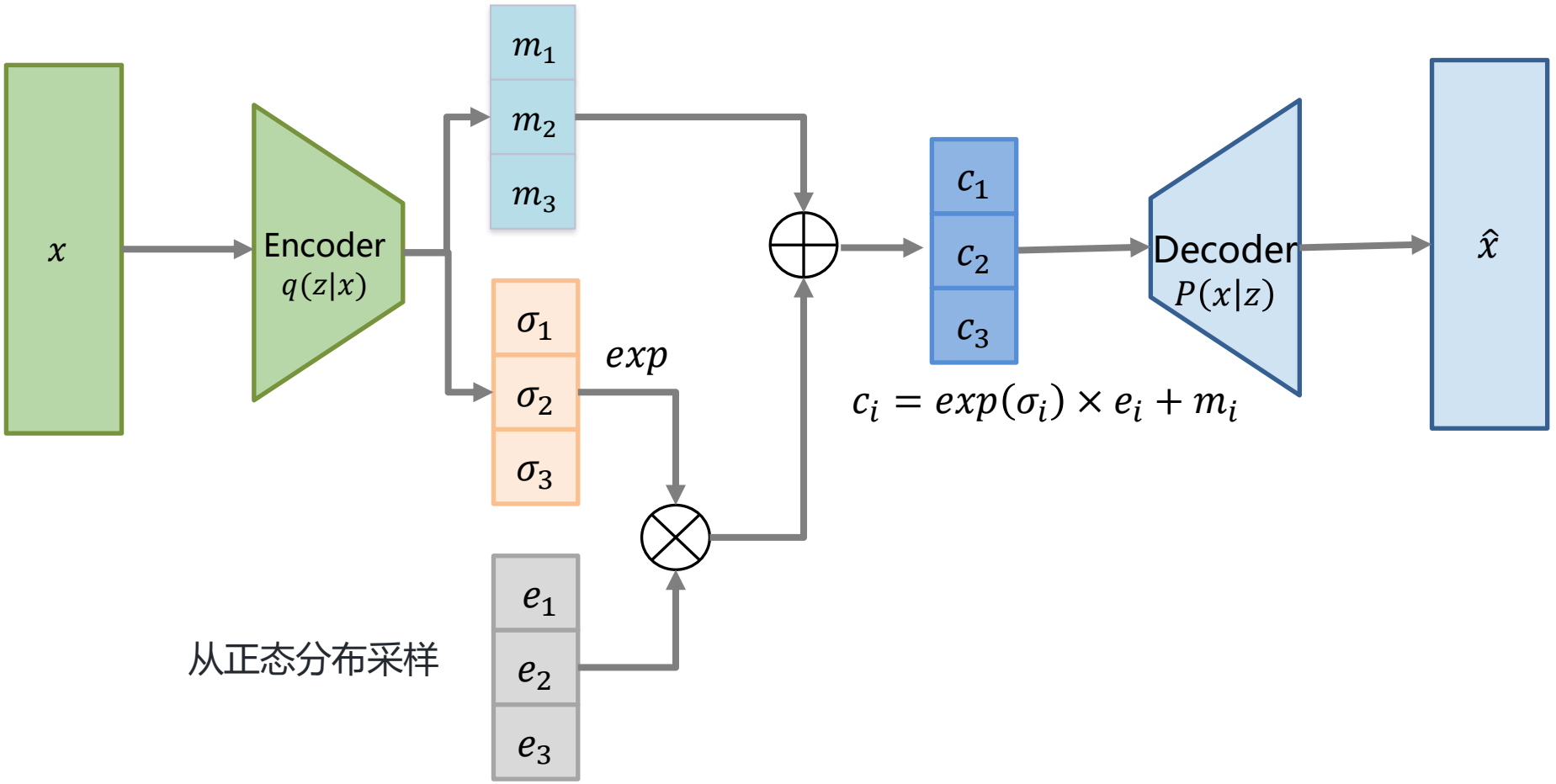


变分自编码器（VAE）- 重参化



变分自编码器（VAE）- 解码器

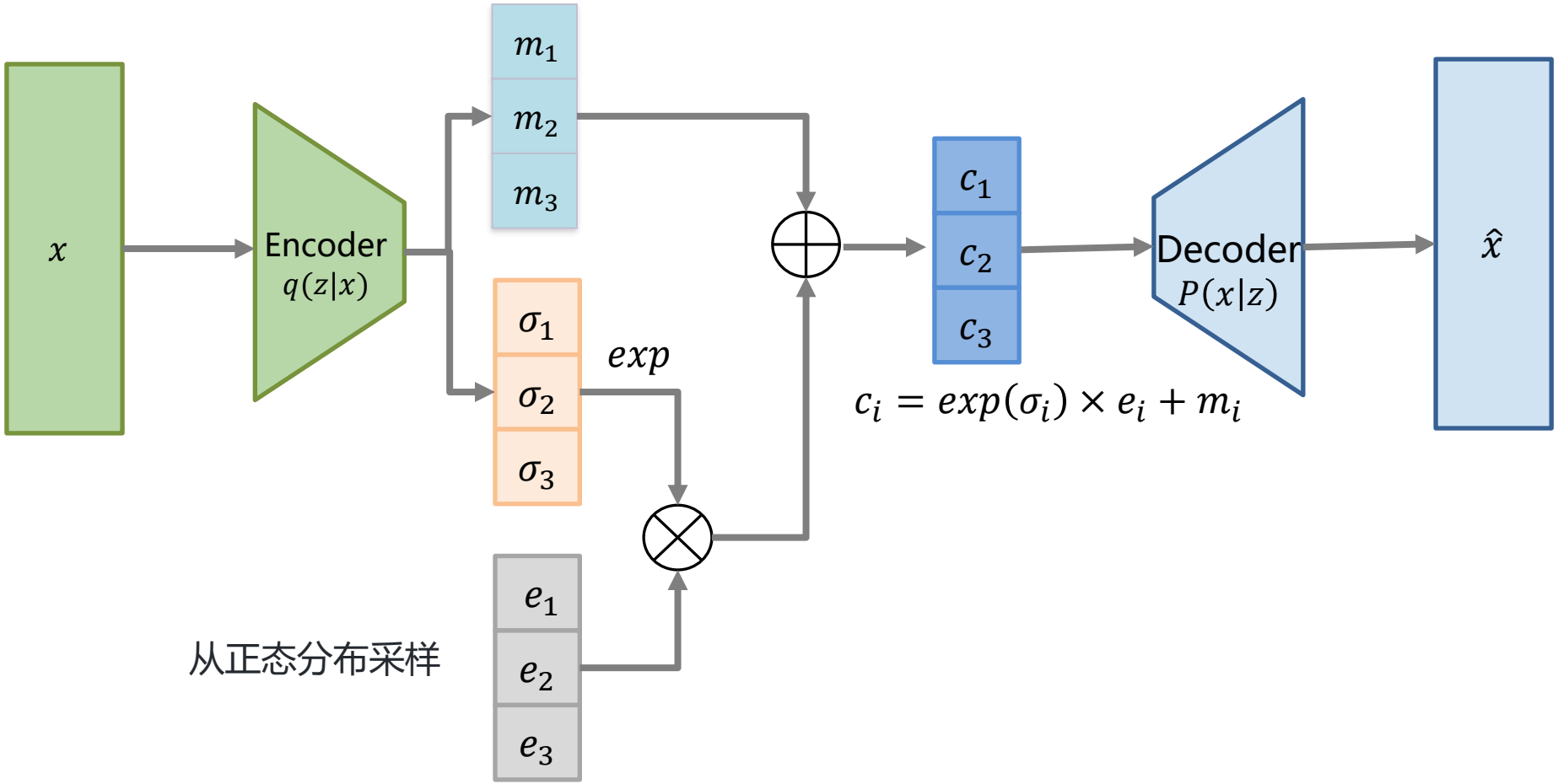
解码器定义了 $q(x|z)$，其将潜在样本作为输入并输出用于观测的条件分布的参数。将潜在分布先验 $P(z)$ 建模为单位高斯。



变分自编码器（VAE）- 损失函数

重构出来的特征和输入特征除了L2损失以外，还要最小化：

$$\sum_{i=1}^{3}(exp(\sigma_i)-(1+\sigma_i)+(m_i)^2)$$



VAE 工作原理

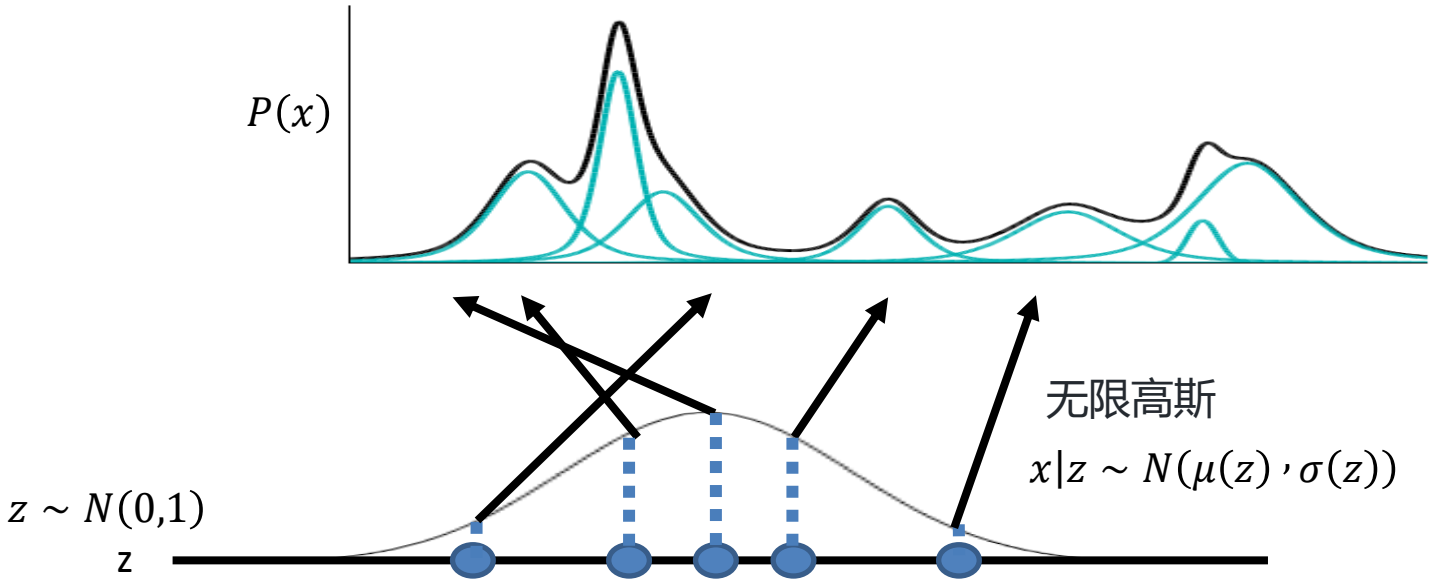
VAE 的理论基础是高斯混合模型（GMM）,**离散**全概率公式:

$$p(x) = \sum_z p(z)p(x|z)$$

不同之处在于，我们的代码被**连续**变量 z 取代，z 遵循标准正态分布 $\mathcal{N}(0,1)$ 。
对于每个样本 z，将有两个变量 μ 和 σ，它们分别确定 z 对应的高斯分布的均值和标准差，然后积分域中所有高斯分布的累加就成为原始分布$P(x)$：

$$P(x) = \int_z P(z)P(x|z)dz$$

我们假设 z 服从标准高斯分布，先验分布 $P(x|z)$ 是高斯分布，$x|z \sim N(\mu(z), \sigma(z))$ 。$\mu(z)$ 、$\sigma(z)$是两个函数， 分别是 z 对应的高斯分布的均值和方差，则 $P(x)$ 就是在积分域上所有高斯分布的累加。



VAE 工作原理

$$P(x) = \int_z P(z)P(x|z)dz$$

由于 $p(z)$ 已知，$p(x|z)$ 未知，且 $x|z \sim N(\mu(z), \sigma(z))$。我们真正需要解决的是 $\mu(z)$ 和 $\sigma(z)$ 的表达式，但 $p(x)$ 太复杂了，μ 和 σ 很难计算，我们需要引入两个神经网络来帮助我们解决它。

我们假设 $p(x)$ 越大越好，那么：

$$\text{Maximum } L = \sum_x \log P(x)$$

$$\log P(x) = \int_z q(z|x)\log P(x)dz$$

$$= \int_z q(z|x)\log\left(\frac{P(z,x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)\log\left(\frac{P(z,x)}{q(z|x)}\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)\log\left(\frac{P(z,x)}{q(z|x)}\right)dz + \int_z q(z|x)\log\left(\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)\log\left(\frac{P(z,x)}{q(z|x)}\right)dz + KL(q(z|x)||P(z|x))$$

VAE 工作原理 - 变分下界（Variational Lower bound）/变分推理

$$\log P(x) = \int_z q(z|x)\log\left(\frac{P(z,x)}{q(z|x)}\right)dz + KL(q(z|x)||P(z|x))$$

上面公式的第二项 $\boldsymbol{KL(q(z|x)||P(z|x))}$ 是一个大于或等于 0 的值，所以我们找到了 $\log P(x)$ 的下界。

$$\log P(x) \geq \int_z q(z|x)\log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz$$

我们将该下限表示为 ELBO：

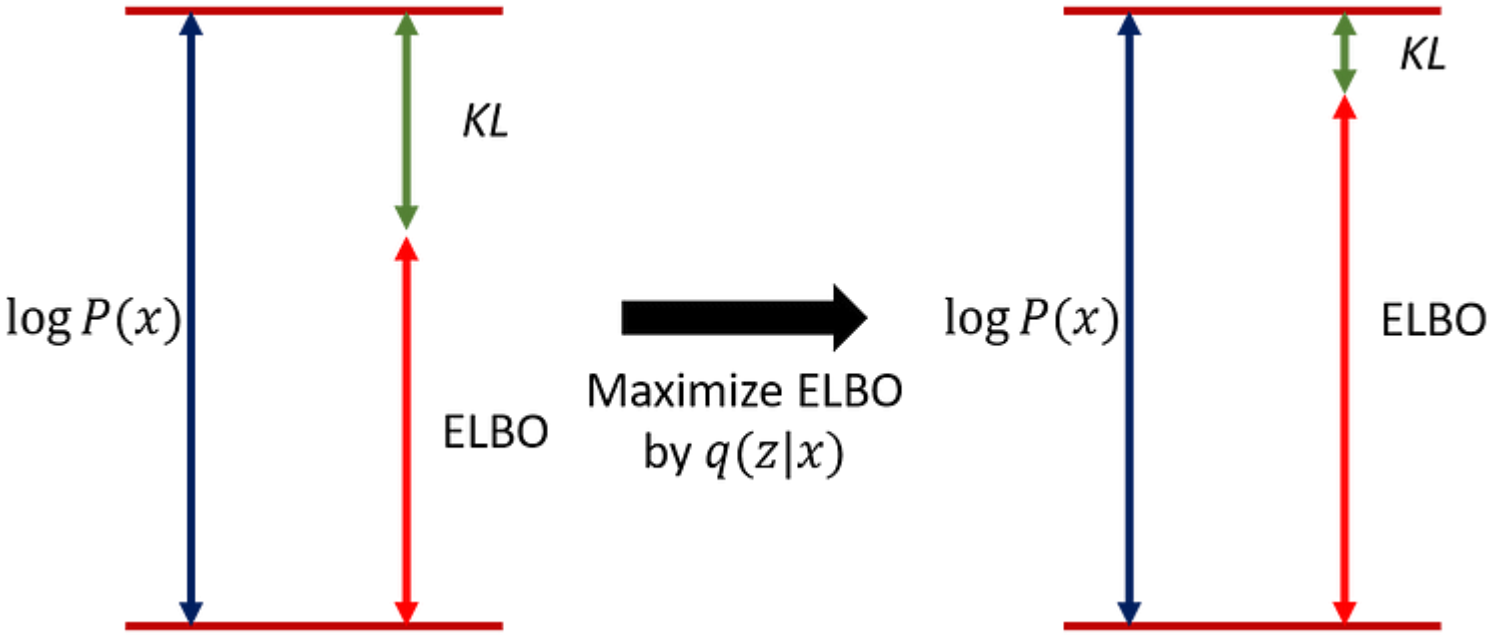
$$\log P(x) \geq \text{ELBO} = \int_z q(z|x)\log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz = \mathrm{E}_{q(z|x)}\left[\log\frac{p(x,z)}{q(z|x)}\right]$$

因此，我们可以将原始形式修改为：

$$\log P(x) = \text{ELBO} + KL(q(z|x)||P(z|x))$$

VAE 工作原理 - 变分下界（Variational Lower bound）/变分推理

$$\log P(x) = \text{ELBO} + KL(q(z|x)||P(z|x))$$



通过调整 $q(z|x)$ 使 ELBO 越来越高，KL散度越来越小。
当我们调整 $q(z|x)$ 使 $q(z|x)$ 和 $P(z|x)$ 相同时，KL散度消失为 0，ELBO 与 $\log P(x)$ 完全一致。
可以得出结论，我们总是可以将 ELBO 调整为等于$\log P(x)$，并且由于 ELBO 是 $\log P(x)$的下限，因此求解最大 $\log P(x)$ 等价于求解最大ELBO。

VAE 工作原理 - 最大化期望

调整 $P(z|x)$ 是调整**解码器**，并且调整 $q(z|x)$ 是在调整**编码器**。每次解码器前进时，编码器都会被调整以与之保持一致，因此解码器只会在下一个训练时期之后变得更好。

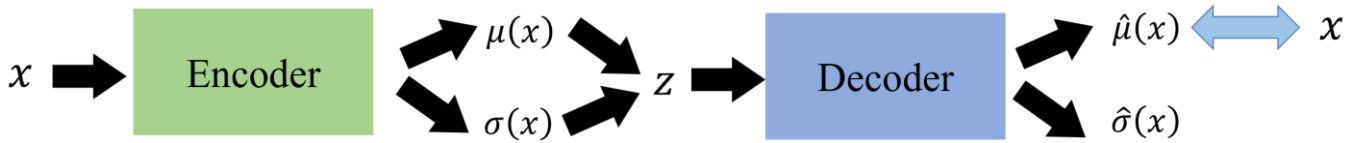
$$\begin{aligned}\text{ELBO} &= \int_z q(z|x)\log\left(\frac{P(z,x)}{q(z|x)}\right)dz \\ &= \int_z q(z|x)\log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz \\ &= \int_z q(z|x)\log\left(\frac{P(z)}{q(z|x)}\right)dz + \int_z q(z|x)\log P(x|z)dz \\ &= -KL(q(z|x)||P(z)) + \int_z q(z|x)\log P(x|z)dz\end{aligned}$$

因此，最大化 ELBO 等效于最小化 $KL(q(z|x)||P(z))$，并使**第二项**中的积分方程最大化。

$$\text{Maximum}\int_z q(z|x)\log P(x|z)dz = \text{Maximum}\,\mathrm{E}_{q(z|x)}[\log P(x|z)]$$

上述期望意味着给定 $q(z|x)$（编码器的输出），$P(x|z)$（解码器的输出）尽可能高。这类似于 AutoEncoder 的损失函数（重建误差）：

$$(x-\hat{x})^2$$



VAE 工作原理 - 最小化KL散度

$$\text{ELBO} = -KL(q(z|x)||P(z)) + \int_z q(z|x)\log P(x|z)dz$$

对于第一项，$-KL(q(z|x)||P(z))$：

$$\int_z q(z|x)\log P(z)dz = \int_z N(z;\mu,\sigma^2)\log N(z;0,I)dz$$

$$= -\frac{J}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{J}\left(\mu_i^2 + \sigma_i^2\right)$$

$$\int_z q(z|x)\log q(z|x)dz = \int_z N(z;\mu,\sigma^2)\log N(z;\mu,\sigma^2)dz$$

$$= -\frac{J}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{J}\left(1 + \log\sigma_i^2\right)$$

于是 $-KL(q(z|x)||P(z))$，可以写为：

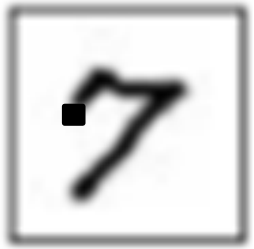
$$-KL(q(z|x)||P(z)) = \int_z q(z|x)\big(\log P(z) - \log q(z|x)\big)dz$$

$$= -\frac{1}{2}\sum_{i=1}^{J}\left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right)$$

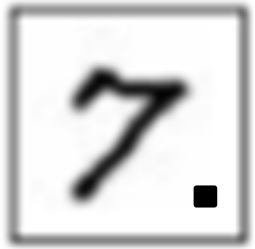
变分自编码器局限性

局限性：

- 人们大多已经从VAE转向使用GAN来生成合成的高维数据
- VAE理论复杂
- 不能很好地泛化
- 在实践中受到限制：重构误差不一定与真实性完全相关



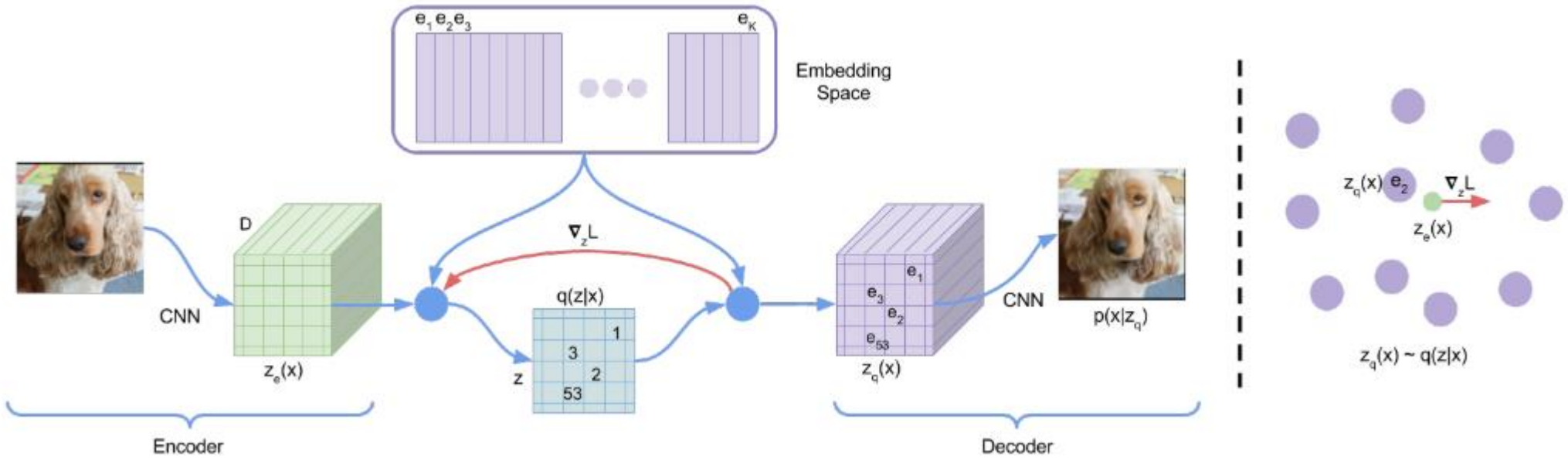
Realistic



Fake

VQ-VAE

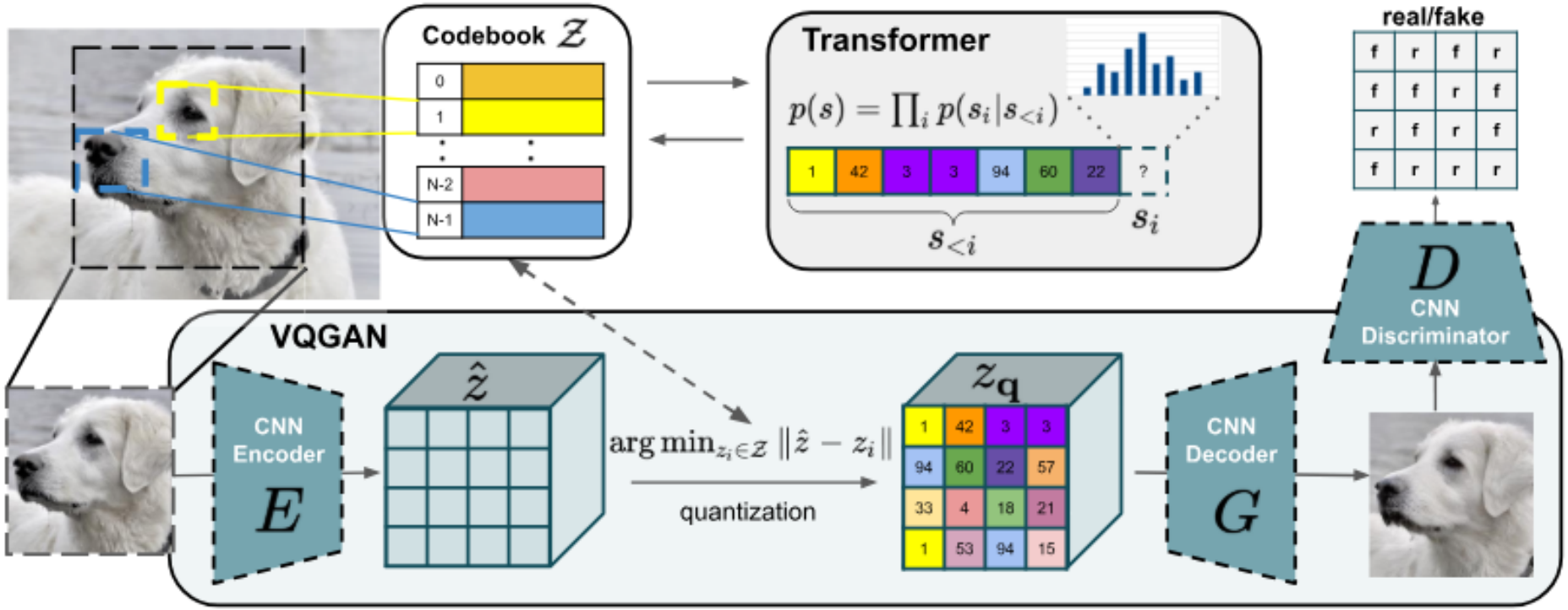
VQ-VAE（Vector Quantized Variational Autoencoder）是一种基于变分自动编码器（VAE）和向量量化（Vector Quantization）的深度学习模型。它结合了自动编码器和向量量化的优点，用于学习高效的数据表示。



DDPM：从一个随机高斯噪声还原图像
LDM：从VQ-VAE得到的离散特征中进行还原

VQ-GAN

VQ-GAN（Vector Quantized Generative Adversarial Network）的核心思想是结合了向量量化（Vector Quantization）和生成对抗网络的概念。
在VQ-GAN中，生成器和判别器之间进行对抗训练，生成器负责生成图像，判别器则负责评估生成的图像是否真实。
与传统的GAN不同，VQ-GAN引入了一个向量量化模块，用于将生成器输出的连续特征映射到一个离散的向量空间中。




Thank
You