Statistical Computing
Chap. 5.1: Variational Autoencoder

LIU, Ran

April 6, 2025

Autoencoder

- Autoencoder 是一种无监督学习方法，用于数据压缩与重构。
- 包含两个部分:
  - 编码器 (encoder)： $f_\phi(x) \to z$
  - 解码器 (decoder)： $g_\theta(z) \to \hat{x}$
- 损失函数为重构误差:

$$\mathcal{L}_{\text{AE}} = \|x - \hat{x}\|^2$$

- Variational Autoencoder (VAE) 是其概率版本，引入了潜变量的先验分布，并使用变分推断近似后验。

生成模型设定

我们构建如下生成模型：

$$p(z) = \mathcal{N}(z; 0, I)$$
$$p_\theta(x \mid z) = \mathcal{N}(x; \mu_\theta(z), \sigma_\theta^2(z) I)$$

其中 θ 是生成模型参数，μ_θ 和 σ_θ 可包含复杂的映射，例如神经网络。

我们假设 θ 是未知但确定的，隐变量 z 是未知的随机变量。那么类似高斯混合模型，此时为了处理隐变量未知情况下最大化观测变量的对数似然函数，我们可以使用 EM 算法，但这里

$$p_\theta(z \mid x) \propto p_\theta(x \mid z) p(z)$$

后验分布的归一化常数难以求解，我们使用 VEM，用变分分布 q 近似 $p_\theta(z \mid x)$。

在使用 VEM 时，我们建立参数化的变分分布 $q_\phi(z)$，在 VAE 中，还用到了 x 的信息，也就是 $q_\phi(z \mid x)$，其中 ϕ 包含神经网络的参数。这里的变分分布其实充当了 encoder 的部分。

$$q_\phi(z \mid x) = \mathcal{N}(z; \mu_\phi(x), \sigma_\phi^2(x) I)$$

事实上，我们也可直接用 Monte Carlo EM，即不需要建模 encoder，毕竟只是缺少后验的归一化常数，仍可以采样 $z^{(i)}$，然后计算 Q 函数，再更新 θ。不过这样会导致，给定数据 x，不能快速采样得到对应的隐变量 z。在 VAE 的原始论文中，也有比较两种方法。

高维隐变量空间推荐建模 encoder 模型，不然采样可能会非常慢。

VAE 在 VEM 框架中的解释

- x：观测数据
- z：潜变量（随机变量）
- θ：生成模型参数（Decoder）
- ϕ：变分分布参数（Encoder）
- 目标：最大化观测似然（边际似然）$p_\theta(x)$
- 解决方法：等价于最大化 ELBO

$$\mathcal{L}(\phi, \theta) = \arg\max_{\phi} \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x, z) - \log q_\phi(z \mid x)\right]$$

回顾 VEM 两步（为什么能看出等价于最大化 ELBO）：

1. 变分 E 步（在 θ 固定下，优化 $q_\phi(z)$，使用变分推断）：

$$\phi^{(t+1)} = \arg\max_{\phi} \mathcal{L}(\phi, \theta^{(t)})$$

2. M 步（在 $q_\phi(z)$ 固定下，优化 θ）：

$$\theta^{(t+1)} = \arg\max_{\theta} \left(\mathbb{E}_{q_{\phi^{(t+1)}}(z)} [\log p(x, z \mid \theta)] \right) = \arg\max_{\theta} \mathcal{L}(\phi^{(t+1)}, \theta)$$

前一项是近似 Q 函数，其实每轮更新都可以看成针对 ELBO 的优化。

VAE 最终目标函数

此时的 ELBO 为：

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x, z) - \log q_\phi(z \mid x) \right]$$

展开联合概率 $p_\theta(x, z) = p_\theta(x \mid z) p(z)$：

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x \mid z) \right] - \mathrm{KL}(q_\phi(z \mid x) \parallel p(z))$$

其中：

- 第一项鼓励重构质量
- 第二项促使编码器输出接近先验

变分分布设定

在变分自编码器（VAE）中，设变分分布为：

$$q_\phi(z \mid x) = \mathcal{N}(z; \boldsymbol{\mu}_\phi(x), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(x)))$$

先验分布为标准正态分布 $p(z) = \mathcal{N}(0, I)$，则 KL 散度为：

$$\mathrm{KL}(q_\phi(z \mid x) \,\|\, p(z)) = \frac{1}{2} \sum_{i=1}^{d} \left(\log \frac{1}{\sigma^2_{\phi,i}} + \sigma^2_{\phi,i} + \mu^2_{\phi,i} - 1 \right)$$

$\mu_\phi(x)$ 和 $\sigma^2_\phi(x)$ 是编码器网络输出的均值和方差。该项可以直接计算，用于损失函数中。

多元正态分布之间的 KL 散度

设有两个 d 维多元正态分布：

$$P = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \quad Q = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$

KL 散度定义为：

$$\mathrm{KL}(P \parallel Q) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} \, d\mathbf{x}$$

带入多元正态分布的密度函数，推导结果为：

$$\mathrm{KL}(P \parallel Q) = \frac{1}{2} \left[\log \frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_0|} - d + \mathrm{Tr}(\boldsymbol{\Sigma}_1^{-1} \boldsymbol{\Sigma}_0) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^\top \boldsymbol{\Sigma}_1^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \right]$$

重参数化技巧

因为 ϕ 参数存在求期望的分布的中，不好求导，我们使用重参数化的方式，将它提出来，方便对 ELBO 求导。

设 $z = g_\phi(\epsilon, x)$，其中 $\epsilon \sim p(\epsilon)$，则

$$\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] = \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[\log p_\theta(x \mid g_\phi(\epsilon, x))\right]$$

此时对 ϕ 求偏导为：

$$\nabla_\phi \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] = \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[\nabla_\phi \log p_\theta(x \mid g_\phi(\epsilon, x))\right]$$

在 VAE 中，我们使用

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

重参数化技巧（Reparameterization Trick）

在变分自编码器（VAE）中，变分分布 $q_\phi(z \mid x)$ 依赖于参数 ϕ，使得以下期望不易直接对 ϕ 求导：

$$\mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x \mid z) \right]$$

为了解决这个问题，引入重参数化技巧，将随机变量 z 表达为一个确定性函数 $g_\phi(\epsilon, x)$，其中噪声 ϵ 与 ϕ 无关：

$$z = g_\phi(\epsilon, x), \quad \epsilon \sim p(\epsilon)$$

此时期望可以写为：

$$\mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x \mid z) \right] = \mathbb{E}_{\epsilon \sim p(\epsilon)} \left[\log p_\theta(x \mid g_\phi(\epsilon, x)) \right]$$

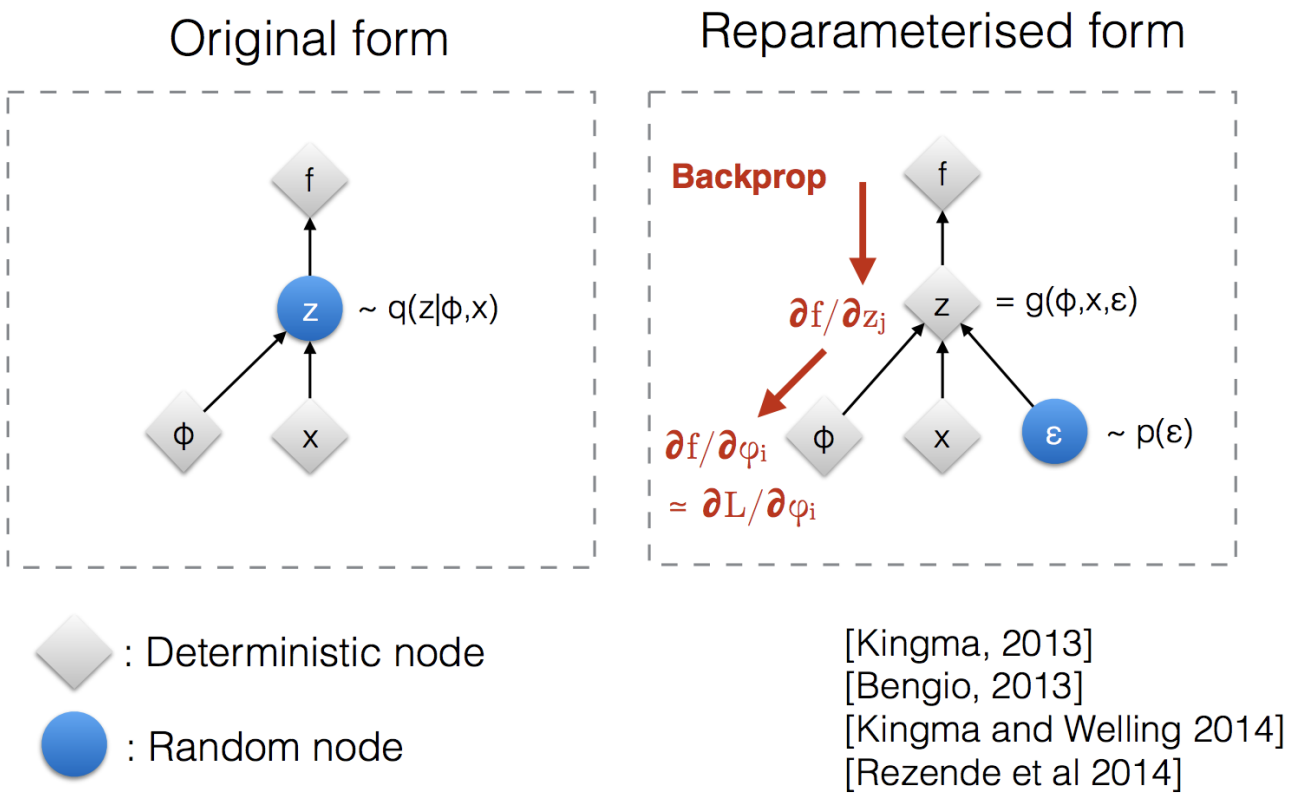
从而可以对 ϕ 求导：

$$\nabla_\phi \, \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x \mid z) \right] = \mathbb{E}_{\epsilon \sim p(\epsilon)} \left[\nabla_\phi \log p_\theta(x \mid g_\phi(\epsilon, x)) \right]$$

在 VAE 中，$q_\phi(z \mid x) = \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma^2_\phi(x)))$，通常使用以下重参数化方式：

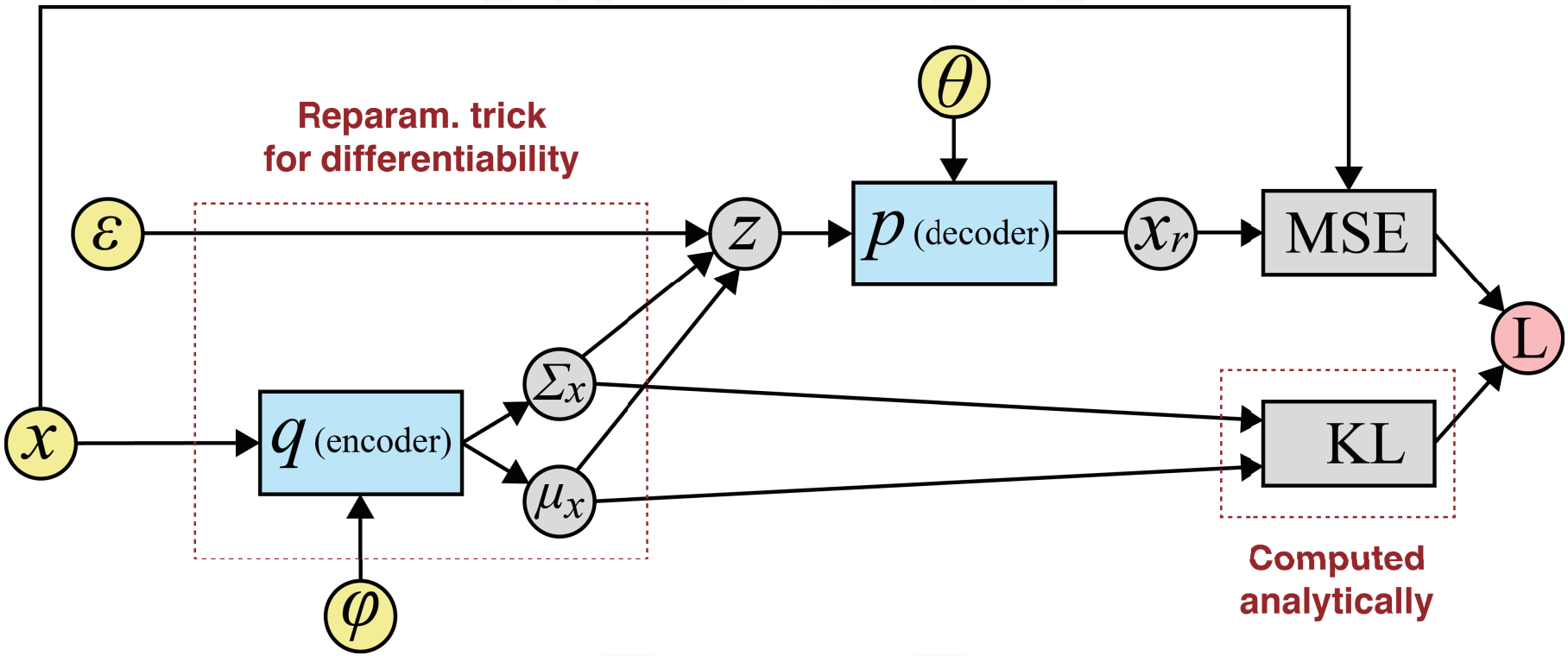
$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

其中 $\odot$ 表示逐元素乘法。此时梯度可反向传播，不会因采样而截断。



[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

整体的 VAE 的框架就为：



VAE 总结

- VAE 在 VEM 框架下，交替优化 ϕ 和 θ
- 使用神经网络建模 $q_\phi(z \mid x)$ 和 $p_\theta(x \mid z)$，其中变分对应的 encoder 不是必须的，可用 MCEM 替代
- 使用重参数化技巧进行高效训练，目标函数是可微的 ELBO
- 离散的隐变量空间可用 VQ-VAE，序列数据可用 TD-VAE

相关资料：

- From Autoencoder to Beta-VAE
  https://lilianweng.github.io/posts/2018-08-12-vae/
- The Reparameterization Trick
  https://gregorygundersen.com/blog/2018/04/29/reparameterization/