An Introduction to Variational Autoencoders

Notes to get started

SummerSemester 2024
05.08.2024

University of Chinese Academy of Sciences

ZTS
plote5024@mail.com
Beijing

Inhaltsverzeichnis

1. Introduction

1.1. Motivation

机器学习的一个主要分支是生成模型和判别模型。在判别建模中，目标是学习给定观察值的预测器，而在生成建模中，目标是解决学习所有变量 **All variables** 的联合分布。

生成模型模拟数据在现实世界中的生成方式。在几乎每一门科学中，“建模”都被理解为通过假设理论和通过观察测试这些理论来揭示这个生成过程。例如，当气象学家为天气建模时，他们使用高度复杂的偏微分方程来表达天气的潜在物理特性。生物学家、化学家、经济学家等等也是如此。科学中的建模实际上几乎全是生成模型。

试图理解数据生成过程的另一个原因是，它自然地表达了世界的因果关系。因果关系的优点，就是它们比单纯的相关性更能概括新情况。

为了将生成模型转化为 Discriminator，我们需要使用贝叶斯规则。即我们可以将生成模型的输出转换为分类任务中需要的条件概率，从而实现判别功能。

在判别方法中，我们直接学习一个与未来预测方向一致的映射。这与生成模型的方向相反。例如，可以认为一张图像在现实世界中是通过先识别物体，然后生成三维物体，再将其投影到像素网格上来生成的。

而判别模型则直接将这些像素值作为输入，并将其映射到标签上。虽然生成模型能够有效地从数据中学习，但它们往往比纯粹的判别模型对数据做出更强的假设，这通常会在模型出错时导致更高的渐近偏差（**Asymptotic bias**）。

因此，如果模型出错（事实上几乎总是有一定程度的误差），如果我们只关注于学习如何区分，并且我们有足够多的数据，那么纯粹的判别模型在判别任务中通常会导致更少的错误。然而，取决于数据量的多少，研究数据生成过程可能有助于指导判别器（如分类器）的训练。例如，在半监督学习的情况下，我们可能只有少量的标记样本，但有更多的未标记样本。在这种情况下，可以利用数据的生成模型来改进分类。

这种方法可以帮助我们构建有用的世界抽象，这些抽象可以用于多个后续的预测任务。这种追求数据中解缠、语义上有意义、统计上独立和因果关系的变化因素的过程，通常被称为无监督表示学习，而变分自编码器（VAE）已经广泛应用于此目的。

或者，可以将其视为一种隐式正则化形式：通过强制表示对数据生成过程有意义，我们将从输入到表示的映射过程约束在某种特定的模式中。通过预测世界这一辅助任务，我们可以在抽象层面上更好地理解世界，从而更好地进行后续的预测。

变分自编码器（VAE）可以看作是两个耦合但独立参数化的模型：编码器或识别模型**(Encoder or Recognition Model)**，以及解码器或生成模型**(Decoder or Generative Model)**。这两个模型相互支持。识别模型向生成模型提供其后验分布的近似值，后者需要这些近似值在“期望最大化”学习的迭代过程中更新其参数。反过来，生成模型为识别模型提供了一个框架，使其能够学习数据的有意义表示，包括可能的类别标签。根据贝叶斯规则，识别模型是生成模型的近似逆。

与普通的变分推断（VI）相比，VAE 框架的一个优势在于识别模型（也称为推断模型）现在是输入变量的一个（随机）函数。这与 VI 不同，后者对每个数据实例都有一个独立的变分分布，这对于大数据集来说效率低下。识别模型使用一组参数来建模输入与潜变量之间的关系，因此被称为“摊销推断”。这种识别模型可以是任意复杂的，但由于其构造方式，只需一次从输入到潜变量的前馈传递即可完成，因此仍然相对快速。然而，我们付出的代价是，这种采样会在学习所需的梯度中引入采样噪声。

或许 VAE 框架最大的贡献是认识到我们可以使用现在被称为“重参数化技巧”的简单方法来重新组织我们的梯度计算，从而减少梯度中的方差。

1.2. Aim

provides a principled method for jointly learning *deep latent-variable models* and corresponding inference models using *stochastic gradient descent*.

该框架在生成建模、半监督学习和表示学习等方面有广泛的应用。

1.3. Probabilistic Models and Variational Inference

由于概率模型包含未知数且数据很少能完整地描述这些未知参数，我们通常需要对模型的某些方面假设一定程度的不确定性。这种不确定性的程度和性质通过（条件）概率分布来描述。在某种意义上，最完整的概率模型形式通过这些变量的联合概率分布来指定模型中所有变量之间的相关性和高阶依赖关系。

我们用 $\boldsymbol{x}$ 表示所有观测变量的向量，其联合分布是我们希望建模的。（使用小写粗体来表示底层的观测随机变量集，即将其展平和拼接，使该集合表示为一个单一的向量。）

假设观测变量 $\boldsymbol{x}$ 是来自未知底层过程的随机样本，其真实的概率分布 $p^*(\boldsymbol{x})$ 是未知的。我们尝试用一个选定的模型 $p_\theta\boldsymbol{x}$ 来近似这个底层过程，其中参数为θ：

$$\boldsymbol{x} \sim p_\theta(\boldsymbol{x})$$

学习最常见的是一个搜索参数 θ 的过程，使得由模型给出的概率分布函数 $p_\theta(\boldsymbol{x})$ 近似于数据的真实分布 $p^*(\boldsymbol{x})$，即对于任何观测到的 $\boldsymbol{x}$，两者尽可能接近：

$$p_\theta(\boldsymbol{x}) \approx p^*(\boldsymbol{x})$$

1.3.1. Conditional Models

在分类或者回归问题上面，我们不关心无条件模型 $p_\theta(\boldsymbol{x})$，更倾向于条件模型 $p_\theta(y|x)$，它近似于底层的条件分布 $p^*(y|x)$：即在观测变量$\boldsymbol{x}$的值上，对变量y的值进行分布。在这种情况下，bold(x)通常被称为模型的输入。与无条件情况类似，我们选择并优化一个模型 $p_\theta(y|x)$，使其接近未知的底层分布，即对于任何 $\boldsymbol{x}$ 和 y:

$$p_\theta(y|\boldsymbol{x}) \approx p^*(y|\boldsymbol{x})$$

1.4. Directed Graphical Models and Neural Networks

我们使用有向概率图模型(Directed probability graph model)，或贝叶斯网络。有向图模型是一种概率模型，其中所有的变量被拓扑组织成一个有向无环图。这些模型的变量的联合分布被分解为先验分布和条件分布的乘积:

$$p_\theta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_M) = \prod_{j=1}^{M} p_\theta\left(\boldsymbol{X}_j \mid P_a\left(\boldsymbol{X}_j\right)\right)$$

$P_a\left(\boldsymbol{x}_j\right)$ 是有向图中节点 j 的父向量的集合。

1.5. Dataset

我们收集$N \geq 1$个数据点组成的数据集 $\mathcal{D}$:

$$\mathcal{D} = \left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}\right\} \equiv \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N} \equiv \boldsymbol{x}^{(1:N)}$$

数据集被认为由同一（不变）系统的独立测量值组成。在这种情况下，观测数据 $\mathcal{D} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$ 被称为独立同分布(i.i.d.)。在独立同分布的假设下，给定参数的情况下，数据点的概率因子化为各个数据点概率的乘积。因此，模型分配给数据的对数概率表示为：

$$\log p_{\theta(\mathcal{D})} = \sum_{x \in \mathcal{D}} \log p_{\theta(x)}$$

1.6. Maximum Likelihood and Minibatch SGD

概率模型最常见的标准是最大对数似然(ML). **Maximization of the log-likelihood criterion is equivalent to minimization of a Kullback Leibler divergence between the data and model distributions.**

优化方式比较好用的方法是 ***stochastic gradient descent***

考虑一个数据集$\mathcal{D}$包含 N 个数据点,我们无法每次都使用整个数据集来计算梯度，因为这样计算代价太高。因此，我们随机抽取一个小批量数据 $\mathcal{M}$（例如，大小为$N_{\mathcal{M}}$），并计算这个小批量数据上的梯度。我们希望这个小批量数据上的梯度能够在期望值上等于整个数据集上的梯度。

在随机梯度下降中，这意味着（公式推导放在 1.6.1，不占用篇幅继续写结论）：

$$\mathbb{E}\left[\nabla_\theta \log p_\theta(\mathcal{M})\right] = \nabla_\theta \log p_\theta(\mathcal{D})$$

$\mathcal{M}$是从$\mathcal{D}$中随机抽出来的小批量数据，$\log p_\theta(\mathcal{M})$是这个小批量的对数似然，$\log p_\theta(\mathcal{D})$是整个数据集的对数似然。

对数似然无偏估计：

对于整个数据集 $\mathcal{D}$，其对数似然是：

$$\log p_\theta(\mathcal{D}) = \sum_{x \in \mathcal{D}} \log p_\theta(x)$$

对于小批量数据 $\mathcal{M}$，其对数似然是：

$$\log p_\theta(\mathcal{M}) = \sum_{x \in \mathcal{M}} \log p_\theta(x)$$

在期望上，小批量数据的对数似然可以近似整个数据集的对数似然：

$$\frac{1}{N_{\mathcal{D}}} \log p_\theta(\mathcal{D}) \simeq \frac{1}{N_{\mathcal{M}}} \log p_\theta(\mathcal{M})$$

通过这样的迷你批次 ***minibatches***，我们可以形成最大似然准则的无偏估计：

$$\frac{1}{N_{\mathcal{D}}} \log p_\theta(\mathcal{D}) \simeq \frac{1}{N_{\mathcal{M}}} \log p_\theta(\mathcal{M}) = \frac{1}{N_{\mathcal{M}}} \sum_{x \in \mathcal{M}} \log p_\theta(x)$$

$$\frac{1}{N_{\mathcal{D}}} \nabla_\theta \log p_\theta(\mathcal{D}) \simeq \frac{1}{N_{\mathcal{M}}} \nabla_\theta \log p_\theta(\mathcal{M}) = \frac{1}{N_{\mathcal{M}}} \sum_{x \in \mathcal{M}} \nabla_\theta \log p_\theta(x)$$

1.6.1. 小批量数据的梯度期望值等于整个数据集的梯度

假设数据集$\mathcal{D}$有N个数据集，小批量$\mathcal{M}$之中有$N_{\mathcal{M}}$个数据集，并且小批量数据是从数据集中随机抽取的。

整个数据集的对数似然梯度:

$$\nabla_\theta \log p_\theta(\mathcal{D}) = \nabla_\theta \sum_{i=1}^{N} \log p_\theta(x_i) = \sum_{i=1}^{N} \nabla_\theta \log p_\theta(x_i)$$

小批量数据的对数似然梯度:

$$\nabla_\theta \log p_\theta(\mathcal{M}) = \frac{N}{N_\mathcal{M}} \sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_{\theta(x_j)}$$

这里乘以$\frac{N}{N_\mathcal{M}}$证明写在 1.6.1.1 了，这里就不占篇幅继续向下推导了。

假设我们抽取了$N_\mathcal{M}$个样本，样本是独立同分布的，我们可以用数学期望来表示这种抽样过程:

$$\mathbb{E}\left[\nabla_\theta \log p_\theta(\mathcal{M})\right] = \mathbb{E}\left[\frac{N}{N_\mathcal{M}} \sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_\theta\left(x_j\right)\right]$$

由于 $\mathcal{M}$ 中的样本是独立同分布的，所以:

$$\mathbb{E}\left[\sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_\theta\left(x_j\right)\right] = N_\mathcal{M} \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right]$$

其中x是从$\mathcal{D}$中随机抽取的一个样本。因此:

$$\begin{aligned}\mathbb{E}\left[\nabla_\theta \log p_\theta(\mathcal{M})\right] &= \mathbb{E}\left[\frac{N}{N_\mathcal{M}} \sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_\theta\left(x_j\right)\right] \\ &= \frac{N}{N_\mathcal{M}} N_\mathcal{M} \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right] \\ &= N \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right]\end{aligned}$$

由于数据点 x 是从整个数据集中抽取的，所以 $\mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right]$ 实际上就是总体梯度的平均值:

$$N \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right] = \sum_{i=1}^{N} \nabla_\theta \log p_\theta\left(x_i\right) = \nabla_\theta \log p_\theta(\mathcal{D})$$

1.6.1.1. Coefficient proof

是为了补偿小批量数据与整个数据集的大小差异，使得梯度估计在期望值上保持一致。这里乘系数的原因是:

1. 比例调整：由于小批量数据 $\mathcal{M}$ 只是整个数据集 $\mathcal{D}$ 的一个子集，其样本数量 $N_\mathcal{M}$ 小于 N。乘以比例系数可以调整小批量数据的梯度估计，使其在期望上与整个数据集的梯度一致。
2. 无偏估计：这种调整确保了小批量数据梯度估计的期望值等于整体数据集的梯度，从而在期望上保持无偏。

定义整个数据集的对数似然梯度:

$$\nabla_\theta \log p_\theta(\mathcal{D}) = \nabla_\theta \sum_{i=1}^{N} \log p_\theta(x_i) = \sum_{i=1}^{N} \nabla_\theta \log p_\theta(x_i)$$

定义小批量数据的对数似然梯度：

$$\nabla_\theta \log p_\theta(\mathcal{M}) = \sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_\theta(x_j)$$

现在假设我们从数据集中随机抽取小批量数据 $\mathcal{M}$,由于 $\mathcal{M}$ 中的数据点是独立同分布的，我们有：

$$\mathbb{E}\left[\nabla_\theta \log p_\theta(\mathcal{M})\right] = \mathbb{E}\left[\sum_{j=1}^{N_\mathcal{M}} \nabla_\theta \log p_\theta(x_j)\right] = N_\mathcal{M} \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right]$$

为了使小批量数据的梯度在期望上等于整个数据集的梯度，我们需要将小批量数据的梯度进行比例调整。我们定义调整后的梯度估计为：

$$\tilde{\nabla}_\theta \log p_\theta(\mathcal{M}) = \frac{N}{N_M} \sum_{j=1}^{N_M} \nabla_\theta \log p_\theta(x_j)$$

我们现在计算这个调整后的梯度的期望值：

$$\begin{aligned}
\mathbb{E}\left[\tilde{\nabla}_\theta \log p_{\theta(\mathcal{M})}\right] &= \mathbb{E}\left[\frac{N}{N_M} \sum_{j=1}^{N_M} \nabla_\theta \log p_\theta(x_j)\right] \\
&= \frac{N}{N_M} \mathbb{E}\left[\sum_{j=1}^{N_M} \nabla_\theta \log p_\theta(x_j)\right] \\
&= \frac{N}{N_M} N_M \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right] \\
&= N \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right]
\end{aligned}$$

由于x是从整个数据集中随机抽取的样本，我们有：

$$N \mathbb{E}\left[\nabla_\theta \log p_\theta(x)\right] = \sum_{i=1}^{N} \nabla_\theta \log p_\theta(x_i) = \nabla_\theta \log p_\theta(\mathcal{D})$$

1.7. Learning and Inference in Deep Latent Variable Models

1.7.1. Latent Variables

我们可以将前一节讨论的完全观测有向模型扩展到包含潜变量的有向模型。潜变量是模型的一部分，但我们并不观测到它们，因此它们不属于数据集的一部分。我们通常用 $\boldsymbol{z}$ 来表示这样的潜变量。在无条件建模观测变量 $\boldsymbol{x}$ 的情况下，有向图模型将表示一个观测变量 $\boldsymbol{x}$ 和潜变量 $\boldsymbol{z}$ 的联合分布 $p_\theta(\boldsymbol{x}|\boldsymbol{z})$。观测变量 $\boldsymbol{x}$ 的边缘分布由以下公式给出：

$$p_\theta = \int p_\theta(\boldsymbol{x}, \boldsymbol{z}) \, \mathrm{d}\boldsymbol{z}$$

这也被称为（单个数据点的）边缘似然或模型证据，当它作为 θ 的函数时。

这种关于 $\boldsymbol{x}$ 的隐式分布可能是非常灵活的。如果 $\boldsymbol{z}$ 是离散的，并且 $p_\theta = (\boldsymbol{x}|\boldsymbol{z})$ 是一个高斯分布，那么 $p_\theta(\boldsymbol{x})$ 就是一个高斯混合分布。对于连续的 $\boldsymbol{z}$, $p_\theta(\boldsymbol{x})$ 可以看作是一个无限混合，这种混合潜在地比离散混合更强大。这样的边缘分布也称为复合概率分布。

1.7.2. Deep Latent Variable Models

deep latent variable model **(DLVM)**

表示那些分布由神经网络参数化的潜变量模型 $p_\theta(\boldsymbol{x},\boldsymbol{z})$。这样的模型可以基于某些上下文条件，如 $p_\theta(\boldsymbol{x},\boldsymbol{z} \mid \boldsymbol{y})$。DLVM 的一个重要优势是，即使有向模型中的每个因子（先验或条件分布）相对简单（例如条件高斯分布），其边缘分布 $p_\theta(\boldsymbol{x})$ 也可以非常复杂，即包含几乎任意的依赖关系。这种表达能力使得深度潜变量模型在近似复杂的底层分布 $p^*(\boldsymbol{x})$ 时非常有吸引力。

或许最简单也是最常见的 DLVM 是通过以下结构指定的分解模型：

$$p_\theta(\boldsymbol{x},\boldsymbol{z}) = p_\theta(\boldsymbol{z})p_\theta(\boldsymbol{x}|\boldsymbol{z})$$

其中，$p_\theta(\boldsymbol{z})$ 和 $p_\theta(\boldsymbol{x}|\boldsymbol{z}$ 是已知的。分布 $p(\boldsymbol{z})$ 通常被称为潜变量 $\boldsymbol{z}$ 的先验分布，因为它不以任何观测数据为条件。

1.8. Intractabilities

dlvm 中最大似然学习的主要困难是 **the marginal probability of data under the model** 通常难以处理。

由于计算边际似然(或模型证据)的方程中的积分，$p_\theta(x) = \int p_\theta(x,z)\,\mathrm{d}z$，**not having an analytic solution or efficient estimator.**。由于这种不可解性，我们不能对其参数进行微分并优化它，就像我们对完全观测模型所做的那样。计算边缘似然 **Need to integrate high-dimensional space**，这在实践中是非常困难的。所以在 MLE 中，我们需要最大化数据在模型下的概率。对于 DLVMs，这涉及到计算边缘似然 $p_\theta(x)$，但由于其不可解性，直接进行 MLE 是不现实的。由于直接计算边缘似然是不可行的，我们通常使用变分推断（**Variational Inference**）或马尔可夫链蒙特卡罗（**MCMC**）方法来近似后验分布。这些方法通过引入可解的近似来处理不可解的积分问题。同样，神经网络参数化的(有向模型) $p(\theta|\mathcal{D})$ 的后验通常难以精确计算，需要近似推理技术。

2. Variational Autoencoders

2.1. Encoder or Approximate Posterior

dlvm 估计这种模型中的对数似然分布和后验分布的问题。变分自编码器(VAEs)框架提供了一种计算效率高的方法来优化 dlvm，并结合相应的推理模型使用 SGD 进行优化。为了将 DLVM 的后验推理和学习问题转化为可处理的问题，我们引入了一个参数推理模型 $q_\phi(z|x)$。这个模型也被称为编码器或识别模型。用 ϕ 表示该推理模型的参数，也称为变分参数。我们优化变分参数 ϕ:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$

像 DLVM 一样，推理模型可以是(几乎)任何有向图形模型:

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = q_\phi(\boldsymbol{z}_1,\ldots,\boldsymbol{z}_M\ |\boldsymbol{x}) = \prod_{j=1}^{M} q_\phi\left(\boldsymbol{z}_j \mid P_a\left(\boldsymbol{z}_j\right),\boldsymbol{x}\right)$$

$P_a(\mathbf{z}_j)$ 是变量 $\mathbf{z}_j$ 在有向图中的父变量集合。与 DLVM 类似，分布 $q_\phi(\mathbf{z}|\mathbf{x})$ 可以使用深度神经网络参数化。

$$(\mu, \log\sigma) = \text{EncoderNeuralNet}_\phi(x)$$

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(z; \mu, \text{diag}(\sigma))$$

我们使用单个编码器神经网络对数据集中的所有数据点执行后验推理。这可以与更传统的变分推理方法形成对比，其中变分参数不是共享的，而是每个数据点单独迭代优化的。通过平摊推理，我们可以避免每个数据点的优化循环，并利用 SGD 的效率。

2.2. Evidence Lower Bound (ELBO)

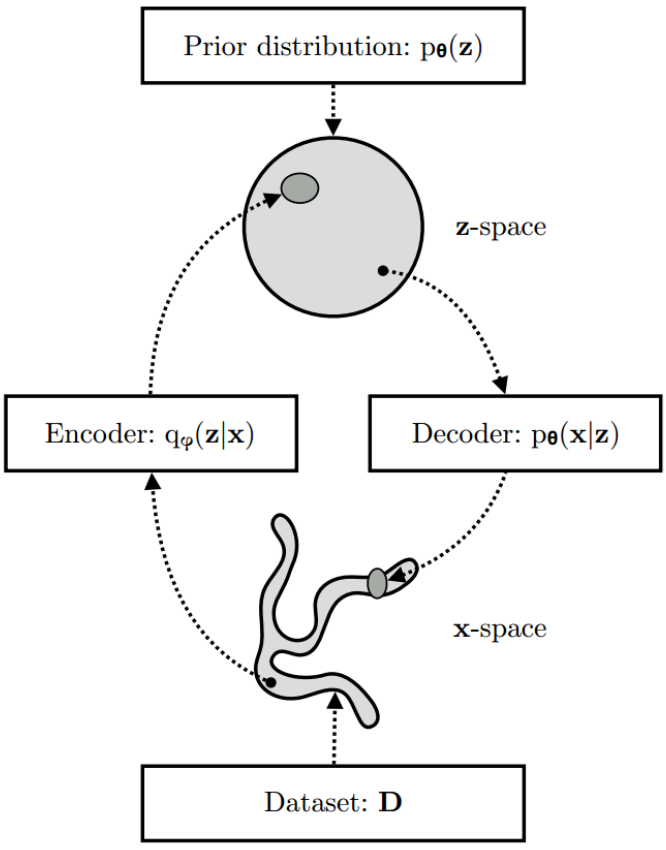

Abbildung 1: It shows how to learn and generate new data through mapping between latent variable space (z-space) and observed data space (x-space).

VAE 通过编码器和解码器网络，利用先验分布、后验近似和重建分布，实现对复杂数据分布的近似建模和生成。

变分参数ϕ:

$$\begin{aligned}
\log p_\theta(\boldsymbol{x}) &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\Big[\log p_\theta(x)\Big] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}\right]\right] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}\right]\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left[\frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]\right]}_{=\mathcal{L}_{\theta,\phi}(x)\ \text{(ELBO)}} + \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left[\frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})}\right]\right]}_{=D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})\|p_\theta(\boldsymbol{z}|\boldsymbol{x})\right)}
\end{aligned}$$

第二项是$q_\phi(z|x)$与$p_\theta(z|x)$之间的 **Kullback-Leibler (KL)**散度是非负的,当等于 0 时,$q_\phi(z|x)$ 等于真实后验分布:

$$D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})\|p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) \geq 0$$

第一项是变分下界，也称为证据下界(ELBO):

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

由于 KL 散度的非负性，ELBO 是数据的对数似然的下界:

$$\begin{aligned}
\mathcal{L}_{\theta,\phi}(\boldsymbol{x}) &= \log p_\theta(\boldsymbol{x}) - D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})\|p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) \\
&\leq \log p_\theta(\boldsymbol{x})
\end{aligned}$$

2.3. Stochastic Gradient-Based Optimization of the ELBO

ELBO 的一个重要性质是，它允许使用随机梯度下降(SGD)对所有参数(ϕ 和 θ)进行联合优化。

我们可以从ϕ和θ的随机初始值开始，随机优化它们的值，直到收敛。

$$\mathcal{L}_{\theta,\phi}(\mathcal{D}) = \sum_{\mathbf{x}\in\mathcal{D}} \mathcal{L}_{\theta,\phi}(\mathbf{x})$$

一般来说，单个数据点 ELBO 及其梯度 $\nabla_{\theta,\phi}\mathcal{L}_{\theta,\phi}(x)$ 是难以处理的。然而，正如我们将展示的那样，存在良好的无偏估计量$\tilde{\nabla}_{\theta,\phi}\mathcal{L}_{\theta,\phi}(x)$，这样我们仍然可以执行小批量 SGD。生成模型参数θ下 ELBO 的无偏梯度很容易得到:

$$\begin{aligned}
\nabla_\theta\mathcal{L}_\theta\mathcal{L}_{\theta,\phi}(\mathbf{x}) &= \nabla_\theta\mathbb{E}_{q_\phi(\mathbf{z})}\Big[\log p_\theta(\mathbf{z},\mathbf{z})\Big] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{z})}\Big[\nabla_\theta\Big(\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{z})\Big)\Big] \\
&\text{Monte Carlo} \\
&\simeq \nabla_\theta\Big(\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{x})\Big) \\
&= \nabla_\theta(\log p_\theta(\mathbf{x},\mathbf{z})
\end{aligned}$$

2.4. Reparameterization Trick

对于连续潜变量和可微编码器和生成模型，可以通过变量的变化直接对 ELBO 进行ϕ和θ的微分，也称为重参数化技巧。

2.4.1. Change of variables

首先，我们将随机变量 $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})$表示为另一个随机变量 ϵ 的可微(可逆)变换，给定z和ϕ:

$$\mathbf{z} = g(\epsilon, \phi, \mathbf{x})$$

其中，随机变量 ϵ 的分布与 x 或 ϕ 无关。这一步的目的是将随机变量 z 表示为一种可微分的方式，从而使得梯度可以有效地计算。

重新参数化技巧的主要优势在于它将不可微分的采样过程转化为可微分的参数化过程，从而使得梯度下降优化成为可能。

Algorithm 1: ELBO 的随机优化。由于噪声来源于小批量采样和$p(\varepsilon)$的采样，因此这是一个双重随机优化过程。我们也把这个过程称为自动编码变分贝叶斯(AEVB)*the Auto-Encoding Variational Bayes* 算法。

Data:

- $\mathcal{D}$: Dataset
- $q_\phi(z|x)$: Inference model
- $p_\theta(x, z)$: Generative model

Result:

- θ, ϕ: Learned parameters

$(\theta, \phi) \leftarrow$ Initialize parameters
while SGD not converged **do**
  $\mathcal{M} \sim \mathcal{D}$ (Random minibatch of data)
  $\varepsilon \sim p(\varepsilon)$ (Random noise for every datapoint in $\mathcal{M}$)
  Compute $\tilde{\mathcal{L}}_{\theta,\phi}(\mathcal{M}, \varepsilon)$ and its gradients $\nabla_{\theta,\phi}\tilde{\mathcal{L}}_{\theta,\phi}(\mathcal{M}, \varepsilon)$
  Update θ and ϕ using SGD optimizer
end

2.4.2. Gradient of expectation under change of variable

在变量变换的基础上，我们可以将期望用新的随机变量 ε 表示：

$$\mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}[f(\mathbf{z})]$$

其中，$\mathbf{z} = g(\varepsilon, \phi, \mathbf{x})$。通过这种变换，期望运算符和梯度运算符变得可交换，我们可以进行简单的蒙特卡罗估计:

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})(\mathbf{z})}[f(\mathbf{z})] = \nabla_\phi \mathbb{E}_{p(\epsilon)}[f(\mathbf{z})]$$

由于 z 是 ε, φ, x 的可微函数，我们可以交换梯度运算符和期望运算符,然后通过从 $p(\varepsilon)$ 中采样 ε，我们可以得到对梯度的近似估计：

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}\left[\nabla_\phi f(\mathbf{z})\right]$$
$$\simeq \nabla_\phi f(\mathbf{z})$$

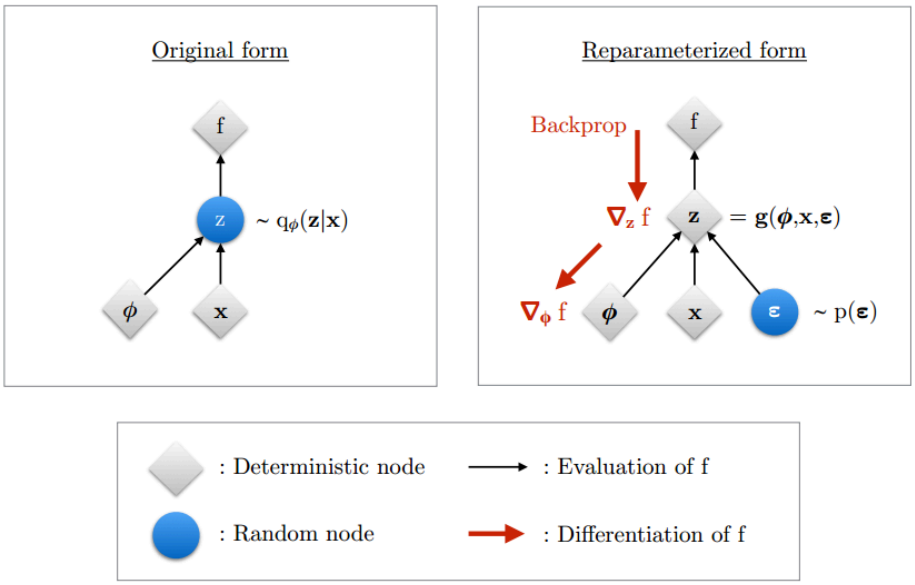

Abbildung 2: This image illustrates the reparameterization trick, a critical technique in variational autoencoders (VAEs) for efficient gradient-based optimization. The trick allows us to rewrite the sampling process of latent variables in a differentiable manner. .

图 2 展示了重新参数化技巧的工作原理，主要包括以下两个部分：原始形式和重新参数化形式。

左图：原始形式（Original form）

1. 节点表示：
   - 灰色节点（**Deterministic node**）：表示确定性节点，如目标函数 f。
   - 蓝色节点（**Random node**）：表示随机节点，如潜变量 z。
2. 图示描述：
   - 随机变量 z 从推断模型 $q_\phi(z|x)$ 中采样。
   - 目标函数 f 依赖于 z 和参数 ϕ。
2. 问题：
   - 我们希望对目标函数 f 求梯度以优化参数 ϕ。
   - 由于 z 是从 $q_\phi(z|x)$ 中采样的，无法直接对 z 进行反向传播，从而无法对 φ 求导。

右图：重新参数化形式（Reparameterized form）

1. 节点表示：
   → 灰色节点（**Deterministic node**）：表示确定性节点，如目标函数 f。
   → 蓝色节点（**Random node**）：表示随机节点，如噪声 ε。
2. 图示描述：
   → 将随机变量 z 表示为 ε, ϕ, x 的可微函数：$z = g(\phi, x, \varepsilon)$。
   → ε 是从简单的分布 $p(\varepsilon)$ 中采样的随机噪声。
3. 梯度计算：
   → 由于 z 现在是 ε, ϕ, x 的可微函数，我们可以对 z 进行反向传播。
   → 这使得我们可以对参数 ϕ 求导并优化目标函数 f。

重新参数化技巧的步骤

1. 变量变换：将潜变量 z 表示为噪声 ε 和参数 ϕ、观测数据 x 的可微函数：$z = g(\varepsilon, \phi, x)$。
2. 期望重写：利用变量变换，将期望用新的随机变量 ε 表示：$\mathbb{E}_{q_{\phi(z|x)}}[f(z)] = \mathbb{E}_{p(\varepsilon)}[f(z)]$。

3. 梯度计算：交换梯度运算符和期望运算符，使用蒙特卡罗采样近似期望，从而计算梯度。

重新参数化技巧的优势

- 使梯度计算可行：通过将采样过程外部化，使得梯度可以通过反向传播进行计算。
- 提高计算效率：简化梯度计算过程，使用蒙特卡罗采样进行近似估计。

2.4.3. Gradient of ELBO

在重新参数化的情况下，我们可以替换期望 $q_\phi(z|x)$与一个$p(\varepsilon)$。ELBO 可以重写为：

$$\begin{aligned}\mathcal{L}_{\theta,\phi}(x) &= \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{x})}\left[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{z})\right] \\ &= \mathbb{E}_{p(\epsilon)}\left[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{x})\right]\end{aligned}$$

因此，我们可以形成单个数据点 ELBO 的简单蒙特卡罗估计量$\tilde{L}_{\theta,\phi}(x)$，其中我们使用来自$p(\varepsilon)$的单个噪声样本$\varepsilon$:

$$\begin{aligned}\epsilon &\sim p(\epsilon) \\ z &= g(\phi, x, \epsilon) \\ \tilde{L}_{\theta,\phi}(x) &= \log p_\theta(x,z) - \log q_\phi(\mathbf{z}\mid\mathbf{x})\end{aligned}$$

这一系列操作可以在 TensorFlow 等软件中表示为符号图，并毫不费力地微分参数 θ 和 φ.

该算法最初被称为 *Auto-Encoding Variational Bayes* (AEVB)算法。更一般地说，重新参数化的 ELBO 估计被称为随机梯度变分贝叶斯*(SGVB)*估计。这个估计器也可以用来估计模型参数的后验.

2.4.4. Computation of log qφ(z|x)

ELBO(估计量)的计算需要计算密度对数$q_\phi(z|x)$，给定值x，并给定值z或等价的ε。这个对数密度是一个简单的计算，只要我们选择正确的变换$g()$。

注意，我们通常知道密度$p(\varepsilon)$，因为这是所选噪声分布的密度。只要$g(.)$是可逆函数，则ε和z的密度关系式为:

(通过变量变换和概率密度函数的性质推导出来的,见 2.4.4.1)

$$\log q_\phi(\mathbf{z}\mid\mathbf{x}) = \log p(\epsilon) - \log d_\phi(\mathbf{x},\epsilon)$$

其中第二项是雅可比矩阵$\left(\frac{\partial_z}{\partial_\varepsilon}\right)$的行列式绝对值的对数,包含了从 ε 到 z 变换的所有一阶导数:

$$\log d_\phi(\mathbf{x},\epsilon) = \log\left|\det\left(\frac{\partial \mathbf{z}}{\partial \epsilon}\right)\right|$$

$$\frac{\partial \mathbf{z}}{\partial \epsilon} = \frac{\partial(z_1,\ldots,z_k)}{\partial(\epsilon_1,\ldots,\epsilon_k)} = \begin{pmatrix} \frac{\partial z_1}{\partial \epsilon_1} & \cdots & \frac{\partial z_1}{\partial \epsilon_k} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_k}{\partial \epsilon_1} & \cdots & \frac{\partial z_k}{\partial \epsilon_k} \end{pmatrix}$$

2.4.4.1. deduce

假设我们有一个随机变量 z，它通过一个可微且可逆的变换 g 由随机变量 ϵ 得到，即：

$$\mathbf{z} = g(\epsilon, \phi, \mathbf{x})$$

根据概率密度函数的变换性质，如果 $z = g(\epsilon)$，那么 z 的概率密度函数 $q_\phi(z|x)$ 和 ε 的概率密度函数 $p(\varepsilon)$ 之间有如下关系：

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = p(\epsilon)\left|\det\left(\frac{\partial \epsilon}{\partial \mathbf{z}}\right)\right|$$

但是，因为我们通常是知道 $\frac{\partial z}{\partial \varepsilon}$ 而不是 $\frac{\partial \varepsilon}{\partial z}$，我们需要使用雅可比行列式的逆关系：

$$\left|\det\left(\frac{\partial \epsilon}{\partial z}\right)\right| = \frac{1}{\left|\det\left(\frac{\partial z}{\partial \epsilon}\right)\right|}$$

密度关系表示:

$$q_\phi(\mathbf{z}) = p(\epsilon)|\det((\partial\epsilon)(\partial z))| = p(\epsilon)\left|\frac{1}{\det\left(\frac{\partial \epsilon}{\partial \epsilon}\right)}\right|$$

$$q_\phi(\mathbf{z}) = p(\epsilon)\left|\det\left(\frac{\partial z}{\partial \epsilon}\right)\right|^{-1}$$

2.5. Factorized Gaussian posteriors

一个常见的选择是一个简单的 *factorized Gaussian encoder*

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\left(\mathbf{z}; \mu, \operatorname{diag}\left(\sigma^2\right)\right)$$

其中，μ 和 $\log\sigma$ 通过一个编码器神经网络 $\text{EncoderNeuralNet}_\phi(x)$获得：

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\left(\mathbf{z}; \mu, \operatorname{diag}\left(\sigma^2\right)\right)$$

后验分布因子化为每个潜在变量 z_i 的高斯分布的乘积：

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \prod_i q_\phi\left(z_i \mid \mathbf{x}\right) = \prod_i \mathcal{N}\left(z_i; \mu_i, \sigma_i^2\right)$$

将高斯随机变量 z 表示为标准正态分布 ε 经过线性变换的结果：

$$\epsilon \sim \mathcal{N}(0, \mathbf{I})$$

$$(\mu, \log\sigma) = \mathit{EncoderNeuralNet}_\phi(\mathbf{x})$$

$$\mathbf{z} = \mu + \sigma \odot \epsilon$$

从 ϵ 到 z 的变换的雅可比矩阵是对角矩阵，其对角线元素是 σ:

$$\frac{\partial z}{\partial \varepsilon} = diag(\sigma)$$

对角矩阵的行列式是其对角线元素的乘积，因此其对数行列式是对角线元素的对数之和：

$$\log d_\phi(\mathbf{x}, \epsilon) = \log \left| \det \left(\frac{\partial \mathbf{z}}{\partial \epsilon} \right) \right| = \sum_i \log \sigma_i$$

后验密度为：

$$\begin{aligned} \log q_\phi(\mathbf{z} \mid \mathbf{x}) &= \log p(\epsilon) - \log d_\phi(\mathbf{x}, \epsilon) \\ &= \sum_i \log \mathcal{N}\left(\epsilon_i; 0, 1\right) - \log \sigma_i \end{aligned}$$

2.5.1. Full-covariance Gaussian posterior

因式高斯后验可以推广为具有全协方差的高斯：

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

该分布的重新参数化由下式给出：

$$\begin{aligned} \epsilon &\sim \mathcal{N}(0, \mathbf{I}) \\ \mathbf{z} &= \mu + \mathbf{L}\epsilon \end{aligned}$$

其中 $\boldsymbol{L}$ 是下(或上)三角矩阵，在对角线上有非零元素。非对角线元素定义了 z 中元素的相关性(协方差)。对于这种全协方差高斯模型，雅可比行列式很简单，因为：

$$\frac{\partial z}{\partial \varepsilon} = \boldsymbol{L}$$

因为 $\boldsymbol{L}$ 是下(或上)三角矩阵，所以行列式是其对角线元素的乘积：

$$\log \left| \det \left(\frac{\partial \mathbf{z}}{\partial \epsilon} \right) \right| = \sum_i \log \left| L_{ii} \right|$$

后验密度的对数计算为：

$$\log q_\phi(\mathbf{z} \mid \mathbf{x}) = \log p(\epsilon) - \sum_i \log \left| L_{ii} \right|$$

协方差矩阵 Σ 可以通过 Cholesky 分解得到：

$$\Sigma = \mathbf{L}\mathbf{L}^T$$

协方差矩阵 $\sum$ 的计算过程为：

$$\Sigma = \mathbb{E}\left[(\mathbf{z} - \mathbb{E}[\mathbf{z}])(\mathbf{z} - \mathbb{E}[\mathbf{z}])^T \right]$$

通过引入 ε，我们有：

$$\Sigma = \mathbb{E}\left[(\mathbf{z} - \mathbb{E}[\mathbf{z}])(\mathbf{z} - \mathbb{E}[\mathbf{z}])^T \right]$$

由于 $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ 有：

$$\mathbb{E}\left[\epsilon \epsilon^T \right] = \mathbf{I}$$

通过神经网络获取参数 μ ,$\log \sigma$和 $\boldsymbol{L}'$ ：

$$(\mu, \log\sigma, \mathbf{L}') \leftarrow EncoderNeurNet_\phi(\mathbf{x})$$

构建$\boldsymbol{L}$矩阵:

$$L \leftarrow \mathbf{L}_{mask} \odot \mathbf{L}^{'} + \text{diag}\,(\sigma)$$

$\boldsymbol{L}_{mask}$是一个掩码矩阵，其对角线下方的元素为 1，对角线和对角线上方的元素为 0。

对于因子化高斯情形，后验密度的对数计算为：

$$\log\left|\det\left(\frac{\partial \mathbf{z}}{\partial \epsilon}\right)\right| = \sum_i \log\sigma_i$$

Algorithm 2: 单数据点 ELBO 无偏估计的计算，用于具有全协方差高斯推理模型和因子化伯努利生成模型的 VAE 示例。L_{mask}是一个掩蔽矩阵，对角线上及以上为零，对角线下为一。

Data:

- x: : a datapoint, and optionally other conditioning information
- ε: a random sample from $p(\varepsilon) = \mathcal{N}(0, I)$
- θ: Generative model parameters
- ϕ: Inference model parameters
- $q_{\phi(z|x)}$: Inference model
- $p_{\theta(x,z)}$: Generative model

Result:

- $\tilde{\mathcal{L}}$: 单数据点 ELBO $\mathcal{L}_{\theta,\phi}(x)$ 的无偏估计

$(\mu, \log\sigma, \boldsymbol{L}') \leftarrow \text{EncoderNeuralNet}_\phi(x)$
$\boldsymbol{L} \leftarrow \boldsymbol{L}_{mask} \odot \boldsymbol{L}' + diag(\sigma)$
$\varepsilon \sim \mathcal{N}(0, I)$
$z \leftarrow \boldsymbol{L}\varepsilon + \mu$
$\tilde{\mathcal{L}}_{\text{logqz}} \leftarrow -\sum_i \left(\frac{1}{2}\left(\varepsilon_i^2 + \log(2\pi) + \log\sigma_i\right)\right)_i$ ▷ $= q_\phi(z|x)$
$\tilde{\mathcal{L}}_{\text{logpz}} \leftarrow -\sum_i \left(\frac{1}{2}\left(z_i^2 + \log(2\pi)\right)\right)$ ▷ $= p_\theta(z)$
$p \leftarrow \text{DecoderNeuralNet}_\theta(z)$
$\tilde{\mathcal{L}}_{\text{logpx}} \leftarrow \sum_i \left(x_i \log p_i + \left(1 - x_i\right)\log\left(1 - p_i\right)\right)$ ▷ $= p_\theta(x|z)$
$\tilde{\mathcal{L}} = \tilde{\mathcal{L}}_{\text{logpx}} + \tilde{\mathcal{L}}_{\text{logpz}} - \tilde{\mathcal{L}}_{\text{logqz}}$

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende schriftliche Hausarbeit (Seminararbeit, Belegarbeit) selbstständig verfasst und keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen der Arbeit, die anderen Werken wörtlich oder sinngemäß entnommen sind, wurden in jedem Fall unter Angabe der Quellen (einschließlich des World Wide Web und anderer elektronischer Text- und Datensammlungen) kenntlich gemacht. Dies gilt auch für beigegebene Zeichnungen, bildliche Darstellungen, Skizzen und dergleichen. Ich versichere weiter, dass die Arbeit in gleicher oder ähnlicher Fassung noch nicht Bestandteil einer Prüfungsleistung oder einer schriftlichen Hausarbeit (Seminararbeit, Belegarbeit) war. Mir ist bewusst, dass jedes Zuwiderhandeln als Täuschungsversuch zu gelten hat, aufgrund dessen das Seminar oder die Übung als nicht bestanden bewertet und die Anerkennung der Hausarbeit als Leistungsnachweis/Modulprüfung (Scheinvergabe) ausgeschlossen wird. Ich bin mir weiter darüber im Klaren, dass das zuständige Lehrerprüfungsamt/Studienbüro über den Betrugsversuch informiert werden kann und Plagiate rechtlich als Straftatbestand gewertet werden.

Ort: ..... **Unterschrift:** .....

Datum: .....