VAE 项目报告

520030910342 柳纪宇

1. 概述

在本项目中，我了解了变分自编码机机 (Variational AutoEncoder, VAE) 的基本原理和实现细节，并在 MNIST 数据集上实现了基于简单 MLP 模型、复杂 MLP 模型、全卷积模型和残差模型的 VAE 训练。

2. VAE 原理

2.1. 基本原理

变分自编码是一种包含隐变量的无监督生成模型，它由编码器和解码器两个部分组成。其中编码器将输入数据映射到一个潜在空间的分布，而解码器从这个分布中采样并重构出与输入数据近似属于同一分布的输出数据。

假设我们有某个包含 N 个样本的独立同分布数据集 $\mathbf{X} = {x^{(i)}}_{i=1}^{N}$，假如我们想从一个隐层向量 z 出发生成与该数据集分布相似的数据，我们需要得知隐层向量 z 的值服从的先验分布 $p_{\theta^*}(z)$ 以及生成样本 x 服从的条件分布 $p_{\theta^*}(x|z)$，接着一个生成模型便可以用 $p_{\theta^*}(z)p_{\theta^*}(x|z)$ 加以表示。然而在实际操作的过程中我们往往无法从训练数据中准确得到使上述分布成立的参数 θ_* 以及 z 的值。

为了解决上述问题，VAE 将隐层向量 z 的分布固定为某一已知分布（如标准高斯分布 $\mathcal{N}(z; 0, I)$），并假设 VAE 的编码器和解码器都遵循对角协方差结构的多元高斯分布，因此 VAE 模型无需拟合隐层向量 z 的复杂分布，而是可以从标准高斯分布中采样数据输入解码器得到生成数据。此外，VAE 将编码器的输出结果定义为 $q_\phi(z|x)$ 的均值和标准差，将解码器的输出定义为满足概率分布 $p_\theta(x|z)$ 的重建结果，二者的概率分布满足下述等式。

$$\log q_\phi(z|x^{(i)}) = \log \mathcal{N}(z; \mu_1, \delta_1^2 I)$$
$$\log p_\theta(x|z) = \log \mathcal{N}(x; \mu_2, \delta_2^2 I)$$

在训练的过程中，编码器参数 ϕ 与解码器参数 θ 将同时得到优化。

2.2. 重参数化技巧

重参数化是 VAE 模型实现的一个重要步骤。由于直接在 $p(z|x)$ 中采样得到 z 值的过程不可微分，因此梯度回传的过程将会在隐层空间处中止，编码器中的参数将得不到有效更新。因此 VAE 模型在此将随机变量 z 表示成一个确定的变量 $g_\phi(\epsilon, x) = \mu + \delta\epsilon$ $(\epsilon \in \mathcal{N}(0, 1))$。由 $\int q_\phi(z|x)f(z)dz = \int p(\epsilon)f(z)d\epsilon = \int p(\epsilon)f(g_\phi(\epsilon, x))d\epsilon$ 可得 $\int q_\phi(z|x)f(z)dz \simeq \frac{1}{L}\sum_{l=1}^{L} f(g_\phi(x, \epsilon^{(l)})$。从而我们便可以顺利对关于 z 的函数的期望值求导了：

$$\begin{aligned}\nabla \mathbb{E}_{\mathcal{N}(z;\mu,\sigma^2)[f(z)]} &= \nabla \mathbb{E}_{\mathcal{N}(z;0,1)}[f(\mu + \sigma\epsilon)] \\ &\simeq \frac{1}{L}\sum_{l=1}^{L}(\mu + \sigma\epsilon^{(l)})\end{aligned}$$

2.3. 误差函数

VAE 的误差函数由两个部分组成，第一部分是重建误差（一般用 BCE 或 MSE 误差来表示），用以衡量重建图像与原图像之间的距离，第二部分是 KL 散度误差，用以衡量编码器得到的分布与标准高斯分布之间的距离。使得编码器输出的 μ 和 σ 与标准高斯分布尽可能地相似。

前者的作用主要是保证生成图像的质量，后者的作用是让编码器得到的分布与我们假定的先验分布尽可能地相似。

2.4. 训练流程

在介绍了 VAE 的基本原理和其中采用的重采样技巧之后，接下来我将分步骤展示 VAE 的训练流程：

1. 输入训练数据 $x^{(i)}$，得到该数据对应的 μ

与 σ 的值。

2. 使用重参数化技巧，得到满足分布 $q_\phi(z|x^{(i)})$ 的隐层向量 $z^{(i,l)} = g_\phi(x^{(i)}, \epsilon^{(l)}) = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(l)}$。

3. 将该隐层向量输入解码器，得到重建图像。

4. 计算相应的重建误差与 KL 散度误差，进行梯度回传更新编码器参数 ϕ 与解码器参数 θ。VAE 模型的基本 pipeline 可见图1。

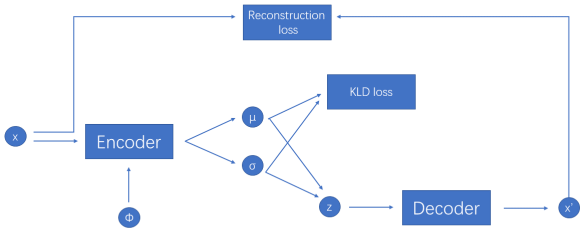


图 1: *VAE* 模型 *pipeline* 结构图

2.5. VAE 与 GAN

VAE 和 GAN 是两种常见的生成模型，尽管二者的设计目的都是图像生成场景，但二者的原理和实现方法之间有许多不同：

1.VAE 采用无监督学习的训练方式，而 GAN 一般使用有监督学习的方式。

2.VAE 对样本分布的拟合通过编码器-解码器结构进行；GAN 并不直接拟合分布，而是使用对抗训练的方式使生成器和判别器相互竞争，从而使得生成器间接学习到样本分布。

3.VAE 的损失函数由隐层分布与正态分布的 KL 散度误差和重建误差两部分组成，而 GAN 的误差则是由生成器误差和判别器误差两部分组成。

由于 GAE 的训练需要在生成器和判别器之间达到平衡状态，故其训练过程相较 VAE 更为复杂。但一旦其达到平衡状态，它也就能够生成比 VAE 更为复杂可信的图像。

VAE 和 GAN 也有着不同的应用场景：由于 VAE 能够学习到数据样本的复杂模式，因此它能够很好地区分出样本中潜在的异常值；由于 GAE 有着较好的生成效果，因此它一般被用于高分辨率、高可信度的图像生成任务。

3. 实验过程

3.1. 多层感知机模型

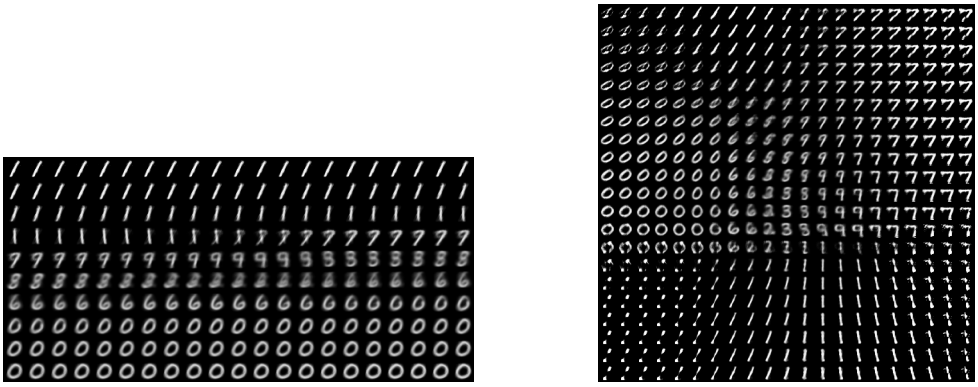
首先，我搭建了简单的多层感知机模型（见./model/VAE_SIMPLE.py 中的 VAE_SIMPLE 类）进行本次实验。该模型的编码器由一个线性输入层、两个线性输出层（分别输出 μ 和 σ）构成，编码器的输出结果在经过重参数化后输入由两个线性层组成的解码器，最终得到重建图像。该模型的损失函数由重建误差（二元交叉熵误差）和编码器输出与标准高斯分布之间的 KL 散度组成。

在模型实现细节上，我在解码器的最后一层使用了 sigmoid 作为激活函数，而在其他位置使用了 relu 作为激活函数。这么做的目的一方面是为了将输出限制在 0 到 1 之间，确保输出与标签之间具有可比性并使得我们可以使用 BCE 作为重建损失；另一方面，在其他位置使用 relu 使得我们能够避免梯度消失等问题。

我使用了学习率 0.01 的 Adam 优化器进行优化，在隐层向量 z 维数为 1 和 2 的情况下分别训练了 50 个 epoch，最终得到的可视化结果如图2所示。可以看出在图（a）的中间部分和图（b）的左上、左下部分，该模型的重建效果并没有达到预期的效果。

此外，我还在原有 MLP 模型的基础上为编码器和解码器添加了更多、更复杂的线性层和激活函数层（见./model/Res_VAE.py 中的 MLP_VAE 类）。其中编码器的隐层向量维度分别为 512，256，128，而解码器的隐层向量维度分别为 128，256，512。

我使用同样的超参数在 z 维数为 1 和 2 的情况下分别训练了 50 个 epoch，最终得到的可视化结果如图3所示。通过该可视化结果我们可以发现，在 z 向量维度为 1 时无论 z 值如何变化重建出的结果都是相同的，而在 z 向量维度为 2 时模型重建结果仍然比较单一。通过观察训练过程中的损



(a) Dim-1 Gaussian (b) Dim-2 Gaussian

图 2: 简单多层感知机模型结果可视化

失变化，我发现在 z 向量维度为 1 时 KL 散度误差降到了 0，而重建误差却仍然很高，这说明该模型的编码器很好地拟合了我们假定的先验分布，但解码器在重建时没能很好地还原原样本空间的分布；而在 z 向量维度为 2 时，训练结束时的重建误差和 KL 散度误差都较高。推测导致该结果的可能原因是，对于 MNIST 数据集来说该模型的参数太过复杂导致过拟合现象，因此无法有效学习样本的真实分布信息。

3.2. 全卷积模型

受经典编码器-解码器模型结构影响，我使用全卷积网络编写了 VAE 的编码器和解码器（见./model/Conv_VAE.py 中的 Conv_VAE 类）。该网络的编码器由三个卷积层和若干全连接层组成，解码器由相应的三个反卷积层和全连接层组成。在此我仍然保留了原先的激活函数、损失函数和超参数，使用该模型得到的可视化结果见图4，对比该结果与之前 MLP 模型的结果可以看出：全卷积模型得到的结果中模糊、破碎、无法辨识的数字所占比例相对较少，生成图像的质量也相对较高，但仍有一些图像没有达到预期的效果。

在分析误差结果时，我注意到使用全卷积网络在隐层向量维数为 1 是在 KLD 误差上表现较好，但在重建误差上表现较为一般；而在隐层向量为 2 时全卷积网络获得了较好的重建误差，但 KLD 误差相较 MLP 网络较高。且在训练后期，模型的重建误差下降的同时 KLD 误差却在上升，我猜测这是因为在平均权重的前提下重建误差有着较高的比重，故在二者形成 trade-off 时网络会优先学习能够降低重建误差的参数。

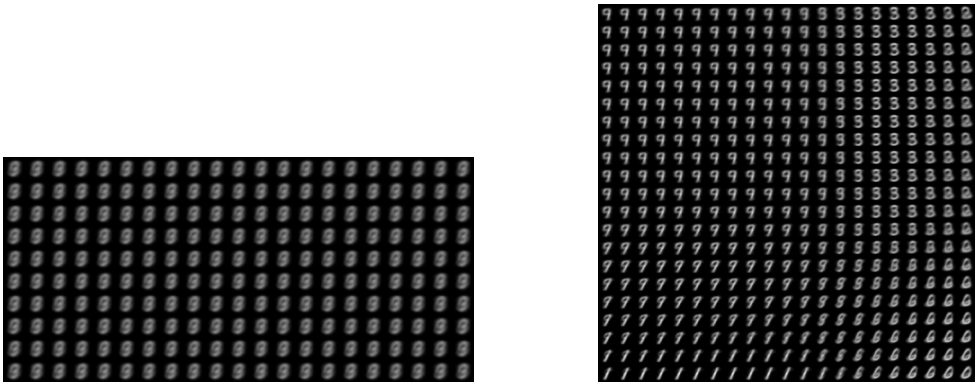
3.3. 残差网络模型

此外，我还尝试了使用残差网络作为模型的编码器与解码器（见./model/Res_VAE.py 中的 Res_VAE 类），通过使用残差块以更多地保留编码解码过程中先前的信息以达到更好的生成效果。该模型的编码器由两个残差卷积层和两个全连接层（用于输出 μ 和 σ）组成，解码器由两个反卷积层、一个残差卷积层和一个全连接层组成。

保留原先采用的损失函数、超参数、激活函数，在同样训练 50 个 epoch 的前提下达到的结果如图5所示。总体来看，残差网络模型的生成效果与全卷积模型较为相似。而在训练结果上我们可以看出，该模型在 KL 散度误差上胜过了全卷积模型，而在重建误差上则是全卷积模型更胜一筹。

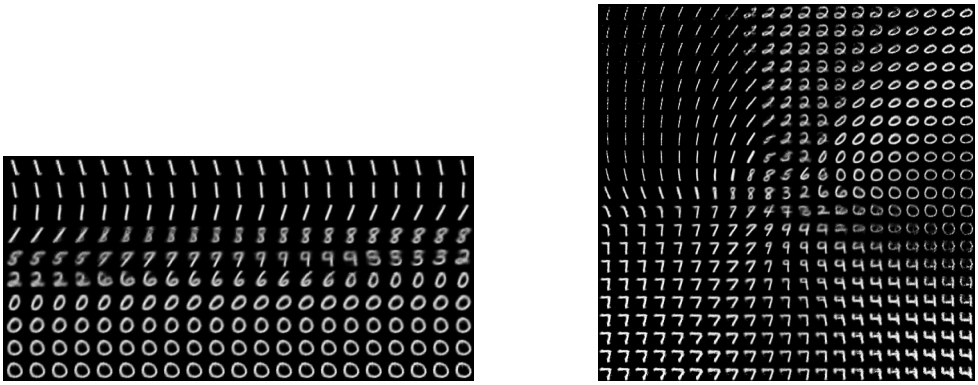
4. 总结

在该实验项目中，我调查并分析了变分自编码机的基本原理、实现技巧和训练流程。在此基础上，我一共使用了简单 MLP、复杂 MLP、全卷积模型、残差网络模型四种网络结构进行模型训练，并分析、比较了它们的可视化结果与误差结果（见



(a) Dim-1 Gaussian (b) Dim-2 Gaussian

图 3: 复杂多层感知机模型结果可视化

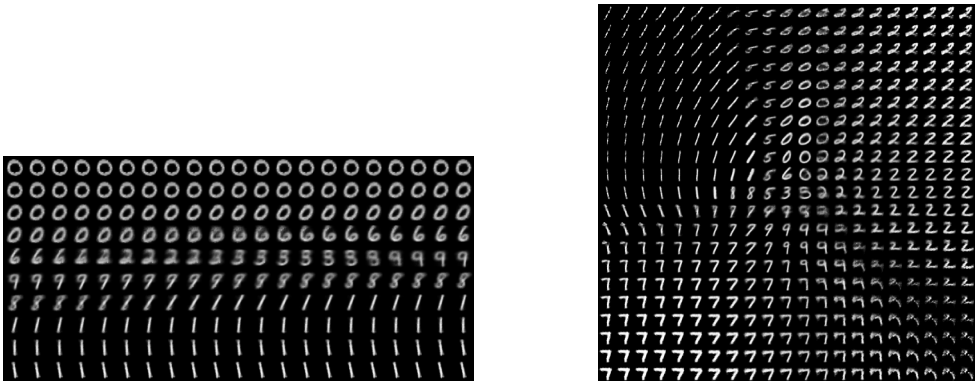


(a) Dim-1 Gaussian (b) Dim-2 Gaussian

图 4: 全卷积模型结果可视化

表1，最佳结果用加粗字体表示）。我在测试集上达到的最小重建误差为 2136.14，使用的模型为隐层向量维数为 2 的全卷积网络模型。

此外，我使用 tensorboard 记录了模型训练过程中的误差变化情况，log 文件保存在项目根目录下的 logdir 文件夹下。



(a) Dim-1 Gaussian (b) Dim-2 Gaussian

图 5: 残差网络模型结果可视化

方法	隐层向量维数	总误差	重建误差	KLD 误差
简单 MLP	1	2552.0	2453.7	98.3
简单 MLP	2	2556.5	2464.7	91.7
复杂 MLP	1	3293.2	3293.2	0.0
复杂 MLP	2	2933.6	2773.8	159.8
全卷积网络	1	2544.6	2476.9	67.6
全卷积网络	**2**	**2252.0**	**2136.14**	**115.9**
残差网络	1	2710.0	2645.5	64.5
残差网络	2	2431.3	2332.3	99.1

表 1: 各网络结构训练结果汇总表