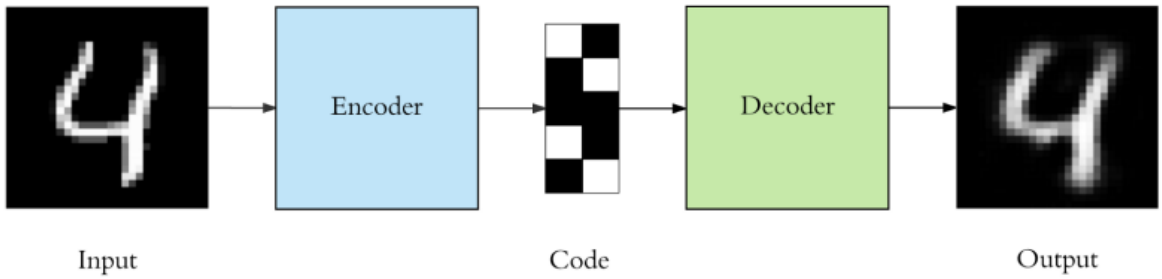
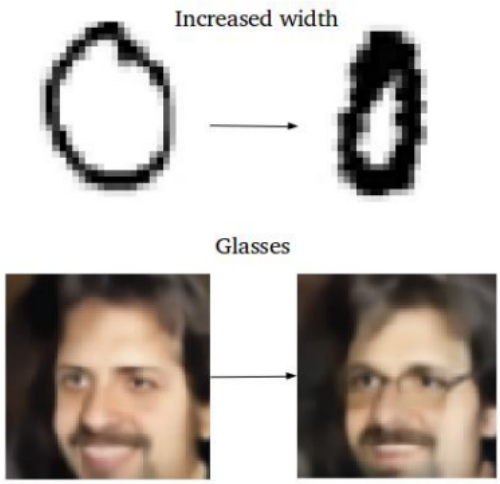
Lecture8 Variational Autoencoders VAE

1. 变分自动编码器 Variational autoencoder

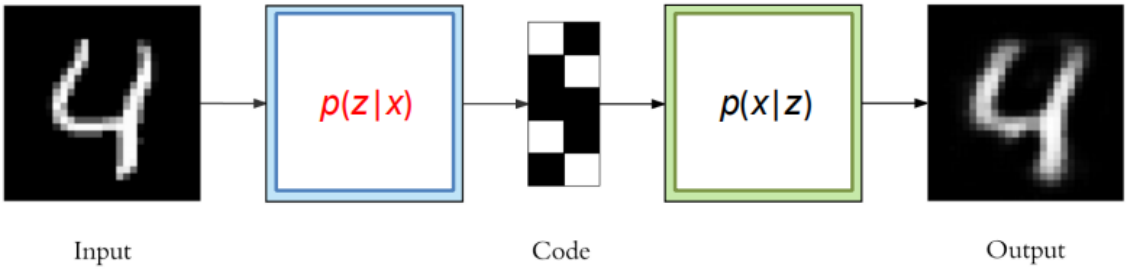
引入



- 我们可以使用一个自动编码器，通过输入的 code 来生成数据吗?
- 可以，但是为什么呢?
  - 探索输入数据的特定变化



- 但是，如何先采样出这个 code z?

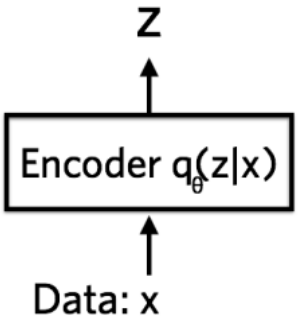


潜在问题

- 如果空间有不连续(例如。簇之间的间隙)，然后你从那里采样/生成一个变体，解码器只会生成一个**不现实**的输出，因为解码器**不知道**如何处理潜在空间的区域，在训练过程中，它**从未看到过**来自那个潜在空间区域的编码向量

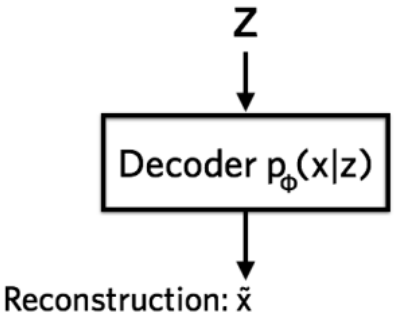
组成

编码器 Encoder



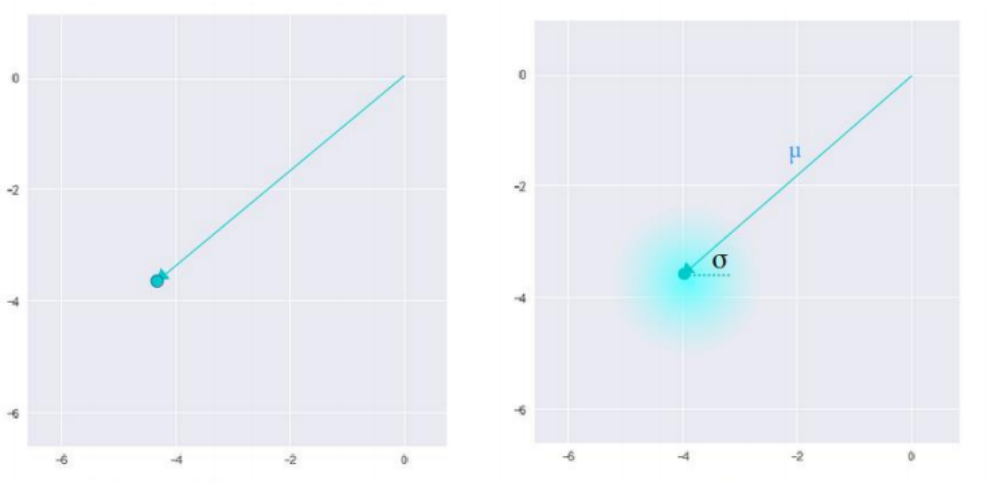
- 编码器是一个神经网络，它在输入中获取一个数据点，输出一个隐藏的表示 z，通常是**低维**的
- 更精确地说，Encoder 输出高斯概率密度 $q_\theta(z \mid x)$ 的参数
- 然后，我们可以从 $q_\theta(z \mid x)$ 中采样一个 code z

解码器 Decoder



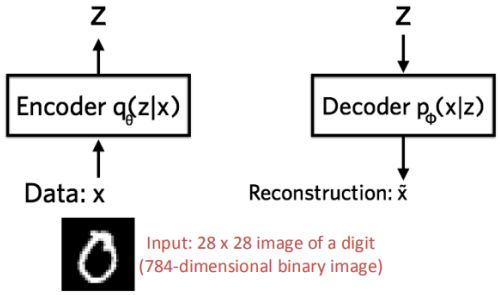
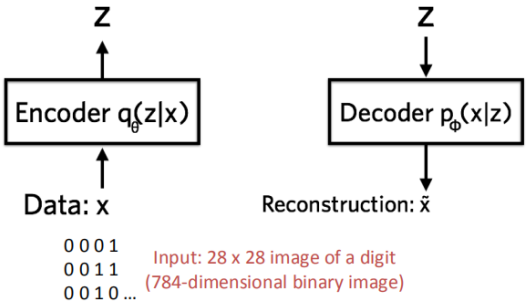
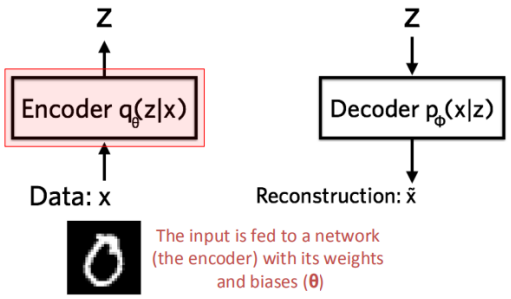
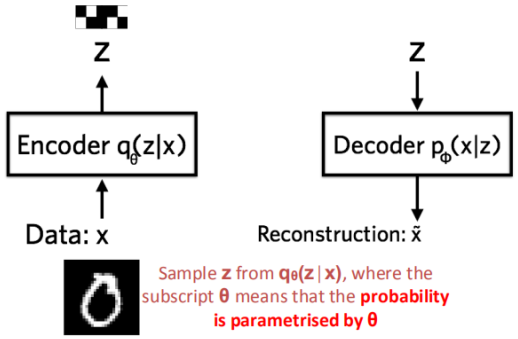
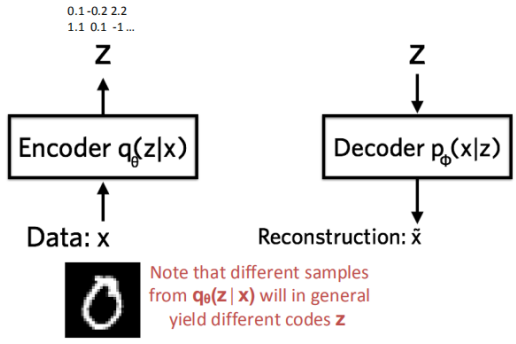
- 解码器是另一个神经网络，它接受输入中的表示 z，并输出 $p_\phi(x \mid z)$ 的参数，这是一个从采样数据计算而来的概率密度

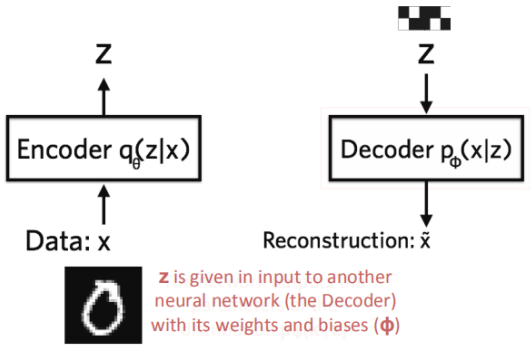
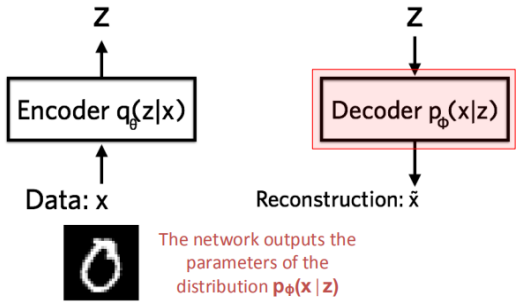
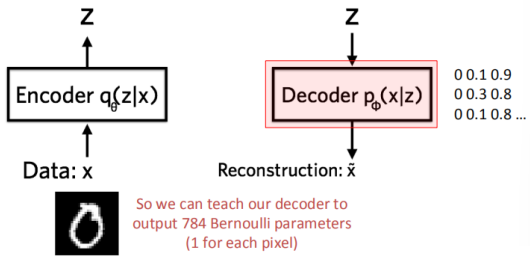
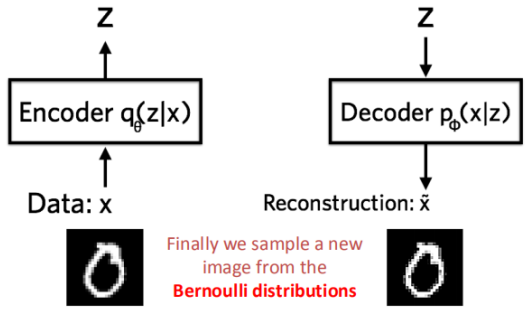
AE 和 VAE 的区别



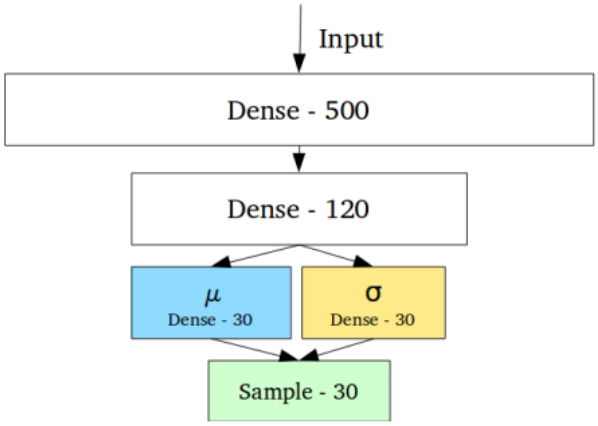
- 标准自动编码器
  - 直接编码坐标
- 变分自动编码器
  - 使用 μ 和 σ 初始化一个概率分布

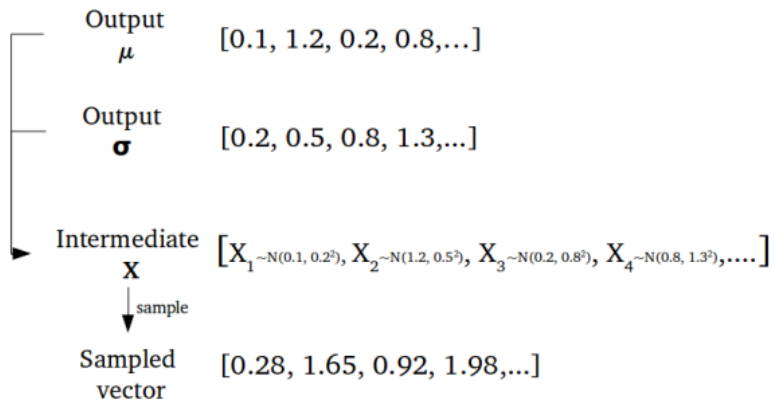
流程

图示	解释
	输入：一个数字的28x28图像（784维二进制图像）
	输入图像的实际表达：二维 0-1 矩阵
	输入被输入到一个具有权重和偏差（θ）的网络（编码器） 网络输出 $q_\theta(z \mid x)$ 的**均值和方差**
	采样出来自 $q_\theta(z \mid x)$ 的样本 z，其中下标 θ 表示概率由 θ 参数化
	Code z 的实际表示 请注意，来自 $q_\theta(z \mid x)$ 的不同样本通常会产生不同的代码 z

图示	解释
	z 作为输入提供给另一个神经网络（Decoder）及其权重和偏差（ϕ）
	网络输出分布 $p_\phi(x \mid z)$ 的参数
	我们可以教解码器输出 784 个伯努利参数（每个像素1个）
	最后，我们从伯努利分布中抽取一个新的图像

VAE 编码器





损失函数

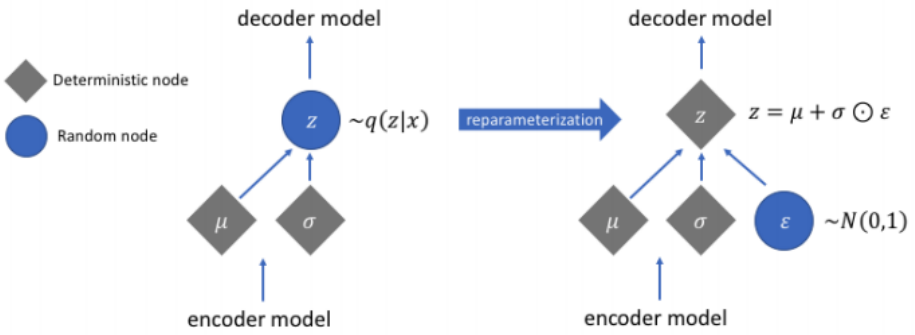
- 从低维表示 z 到高维重建 x 丢失了多少信息?
  - 这可以用重构**对数似然对数** $p_\phi(x \mid z)$ 来测量，它告诉我们解码器是如何有效地学会重构 x 对于给定 z
  - 听起来不错，但是如果我们想做反向传播并学习两个网络（编码器和解码器）的最优参数，我们就需要一个 loss 函数

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)} \left[\log p_\phi \left(x_i \mid z\right)\right] + \mathbb{KL}\left(q_\theta \left(z \mid x_i\right) \| p(z)\right)$$

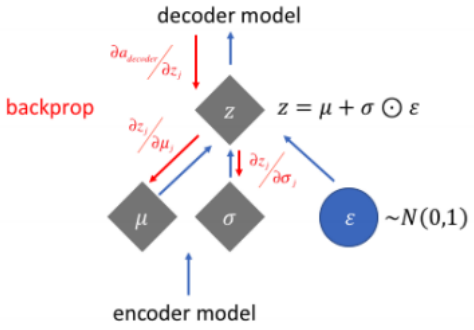
- 第 i 个数据点的损失函数，总损失是所有 l_i 的总和
- 请记住，对于一个数据点，我们可以采样许多代码 z
- 这就是为什么这里我们有预期的对数似然（负的，我们想要最小化）$-\mathbb{E}_{z \sim q_\theta(z|x_i)} \left[\log p_\phi \left(x_i \mid z\right)\right]$
  - 这一项鼓励解码器学习重构数据
  - 坏的编码器（例如，当本来图片应该是白色的块块被计算成高概率是黑色的时候）= 大成本
- 正则化项：衡量 q 与 p 的接近程度 $\mathbb{KL}\left(q_\theta \left(z \mid x_i\right) \| p(z)\right)$
  - 在 VAE 中，我们让 $p(z) = Gaussian(0, 1)$
    - 添加它是为了确保编码器不会作弊，并将每个数据点映射到不同区域的空间中
    - 这是不好的，因为我们希望相同数量的不同图像紧密地隐藏在嵌入空间中
    - 空间必须是“有意义的”，所以我们会惩罚这种行为

鉴于这种架构，我们可以使用**反向传播**来计算网络参数的损失的梯度，然后使用任何梯度变体的下降进行优化

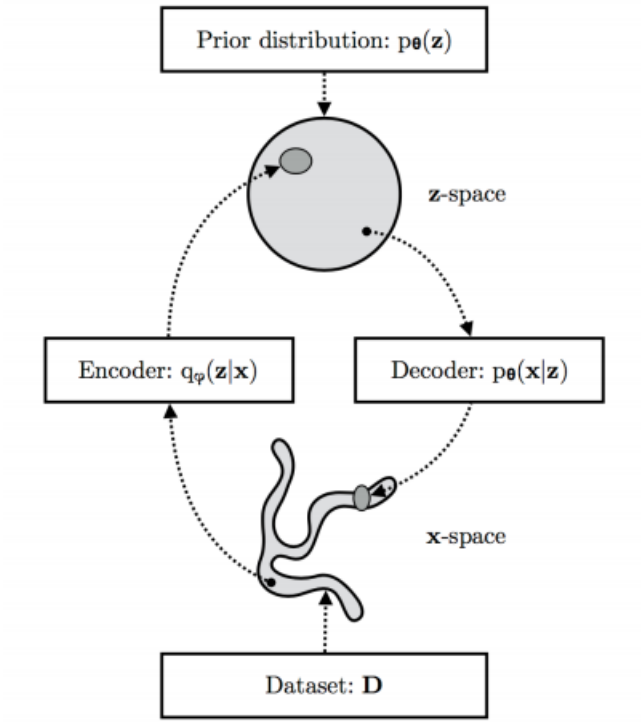
重新参数化技巧

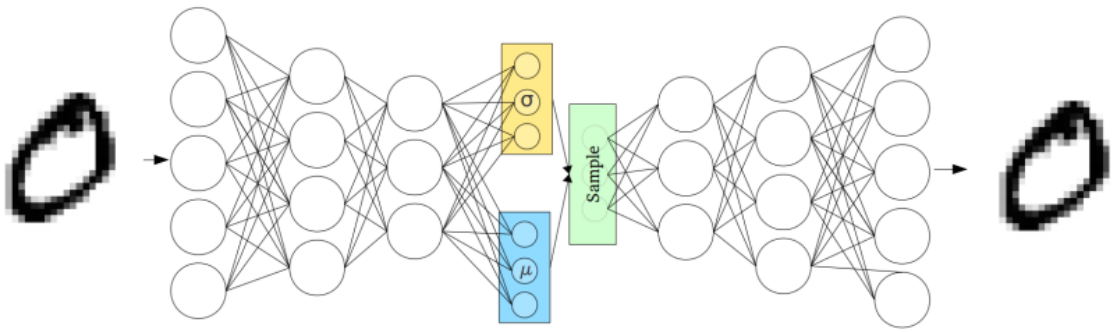


- 我们从一个单位高斯函数中随机采样 ε，然后用潜分布的均值 μ 对其进行平移，再用潜分布的方差 σ 对其缩放
- 通过这种重新参数化，我们现在可以优化分布的参数，同时仍然保持从该分布随机抽样的能力

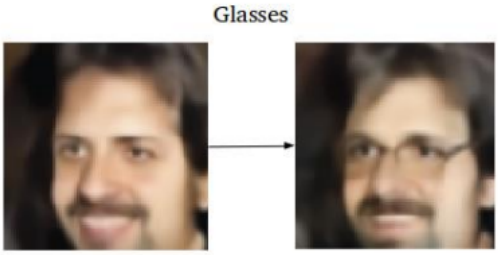


VAE 架构

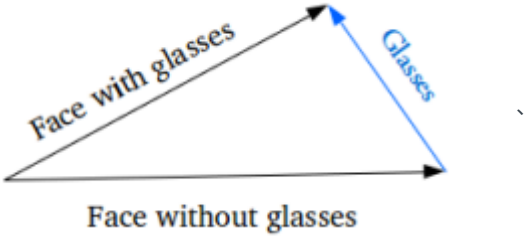




VAE 潜在空间探索



- 需要：在“带眼镜的脸”和“没有眼镜的脸”之间找到图片
- 在潜在空间中，通过嵌入两张没有眼镜和有眼镜的脸，计算出从“没有眼镜的脸”到“有眼镜的脸”的转换向量



- 将转换向量添加到潜在表示中，然后解码该表示，以将其映射回图像空间
- 我们不是将输入编码为单个点，而是将其编码为**潜在空间上的分布**
    - 首先，输入被编码为潜在空间上的分布
    - 其次，从该分布中采样出潜在空间中的一个点
    - 第三，对采样点进行解码，并计算出重构误差
    - 最后，将重构误差通过网络进行反向传播

从概率视角来看 VAE

- 为什么这被称为“变分”自动编码器?
- 为了理解原因，我们需要从一个不同的角度来看变分自编码器，即，从一个**概率模型的角度**出发
- 设数据 x 和潜在变量 z 有一个生成模型，联合概率 $p(x,z)=p(x|z)p(z)$
    - 首先我们从先验 $p(z)$ 中抽取 z
    - 然后我们从概率 $p(x|z)$ 中抽取 x
    - 在这种情况下，神经网络的学习被称为**推理 inference**
- 我们想要推断出 $p(x)$ 的最优参数
    - 换句话说，我们想要最大化 $p(x)$
    - 确切地讲，$logp(x)=logp(x,z)/p(z|x)$
    - 但 $p(z|x)=p(x,z)/p(x)$，$p(x)$ 的计算需要指数时间，因为 $p(x)=\int p(x|z)p(z)dz$
    - $p(z|x)$ 被称为后验，在这类问题中，它通常难以处理
    - 这就是变分元素发挥作用的地方
    - 变分推断将后验 $p(z|x)$ 近似为分布 $q_\lambda(z|x)$
    - 当然我们希望我们的近似值是好的，也就是说，接近真实的后验
    - $$\begin{aligned}&\mathbb{KL}\left(q_\lambda(z\mid x)\|p(z\mid x)\right)=\\&\mathbf{E}_q\left[\log q_\lambda(z\mid x)\right]-\mathbf{E}_q[\log p(x,z)]+\log p(x)\end{aligned}$$

- 我们希望它很小
- $$q_\lambda^*(z \mid x) = \arg\min_\lambda \mathbb{KL}\left(q_\lambda(z \mid x) \| p(z \mid x)\right)$$
- 这就是我们要找的，但是计算 $p(z|x)$ 又回到了原点，我们来将里面的部分引入一个表达
- $$ELBO(\lambda) = \mathbf{E}_q[\log p(x, z)] - \mathbf{E}_q\left[\log q_\lambda(z \mid x)\right]$$
- 然后我们可以写这个样子
- $$\log p(x) = ELBO(\lambda) + \mathbb{KL}\left(q_\lambda(z \mid x) \| p(z \mid x)\right)$$
- $KL \geq 0$，所以为了最小化 KL，我们可以最大化 $ELBO$
- 最大化 $ELBO$ 意味着
  - q 接近 p
  - 更高的 p（更好的生成器）
- $$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta\left(z \mid x_i\right)\left[\log p_\phi\left(x_i \mid z\right)\right] - \mathbb{KL}\left(q_\theta\left(z \mid x_i\right) \| p(z)\right)$$
- 通过解释 q 和 p 的参数，我们将网络和概率观点联系起来，并注意到上述是变分自编码器的(负)损失函数

更多的资料

- 理解 VAE：Understanding Variational Autoencoders (VAEs) | by Joseph Rocca | Towards Data Science
- 各种可视化：https://jaan.io/what-is-variational-autoencoder-vae-tutorial/
- 交互 VAE：https://www.siarez.com/projects/variational-autoencoder
- 变形脸：http://vdumoulin.github.io/morphing_faces/online_demo.html
- MNIST demo：http://dpkingma.com/sgvb_mnist_demo/demo.html