VAE&GAN

1. 前言

VAE 和 GAN 这两个模型是近几年深度学习中的大热门，随着监督学习领域的果子被采摘干净，无监督学习越来越受到关注。在早些年，生成28×28的图片都是一件很困难的事，但自从 2013 年 VAE 面世以来，无监督学习获得了飞速发展，业界现在已经能生成 2K 超清图片了。

下图是英伟达生成的人脸图片，足以以假乱真。

2. 生成模型

生成模型是指能够随机生成观测数据的模型，尤其是在给定某些隐含参数的条件下。它给观测值和标注数据序列指定一个联合概率分布。在机器学习中，生成模型可以用来直接对数据建模（例如根据某个变量的概率密度函数进行数据采样），也可以用来建立变量间的条件概率分布。条件概率分布可以由生成模型根据贝叶斯定理形成。

生成模型的根本目的是根据给出的样本$\{X_1, X_2, ..., X_n\}$，求出这个样本的总体分布$p(X)$。如果能求出的话，那就可以直接根据$p(X)$来采样，得到所有可能的X了。但是现实中直接对数据样本建模求出数据的分布往往是十分困难的，需要用一些技巧，间接地来求解$p(X)$。VAE 和 GAN 就是通过构造一个从正态分布到数据集原始分布的映射来解决这个问题，但两者又有不同，下面来详细了解下这两个模型。

3. 变分自编码器 VAE

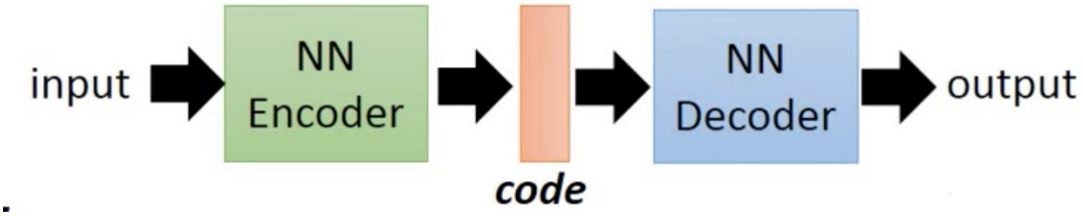
变分自编码器（Variational Autoencoder, VAE）由 Diederik P Kingma 和 Max Welling 在 2013 年发表的论文 Auto-encoding Variational Bayes 中首次提出，这篇论文数学性很强，但证明过程省略了很多中间步骤，比较晦涩，不过作者从原始数据集的分布$p(X)$开始推导到自编码器的构造一气呵成，非常的神奇，建议对这个模型有一定了解后去看。2016 年 Carl Doersch 写了一篇 VAEs 的 tutorial，对 VAEs 做了更详细的介绍，且比原始论文更加易懂。

VAE 是一种深度生成模型，其思想是利用神经网络来建立一个把

可观测变量映射为隐变量的编码器和把隐变量映射为可观测变量的解码器，这种编码-解码的结构和自编码器比较类似，不同点在于变分自编码器中的编码器的输出是分布。

3.1 自编码器

要讲变分自编码器，自然要先讲自编码器。自编码器实质上是一种有损压缩算法，结构图如下所示：



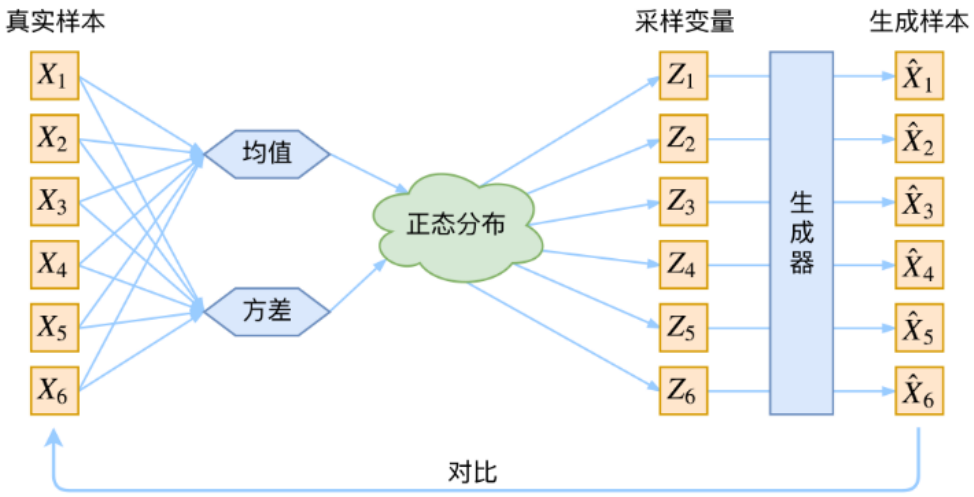
高维的输入数据通过一种变换 f（常用神经网络）变到一组低维的变量（可以称为隐变量），然后再用另一种变换 g（常用神经网络）还原到原来的数据，即 $X_i \xrightarrow{f} Z_i \xrightarrow{g} \hat{X}_i$，这个过程称为编码-解码。然后设计一种度量 $D(\hat{X}_i, X_i)$ 描述还原后的数据和原始数据的相似度，就可以最小化 $D(\hat{X}_i, X_i)$ 来训练模型了。训练好之后，只要提供变换之后的变量和还原算法，就可以生成原始数据了，所以这个算法称为自编码器。

但是自编码器有一个很大的缺点，它是样本相关的，也就是说，有一个新的样本，它没有进入过自编码器的训练过程，如果将它放到编码器中编码得到隐变量，然后用解码器对这个隐变量进行解码，得到的数据很有可能跟原数据差别很大。这说明自编码器的泛化能力很差。毕竟训练样本是有限的，而样本空间是无限的，尤其是一个很复

杂的分布，很难用有限的离散数据去描述。变分自编码器则通过假设隐变量服从某个简单的分布（基本上都用正态分布），来实现无限到无限的映射。

3.2 一种新的编码解码技巧

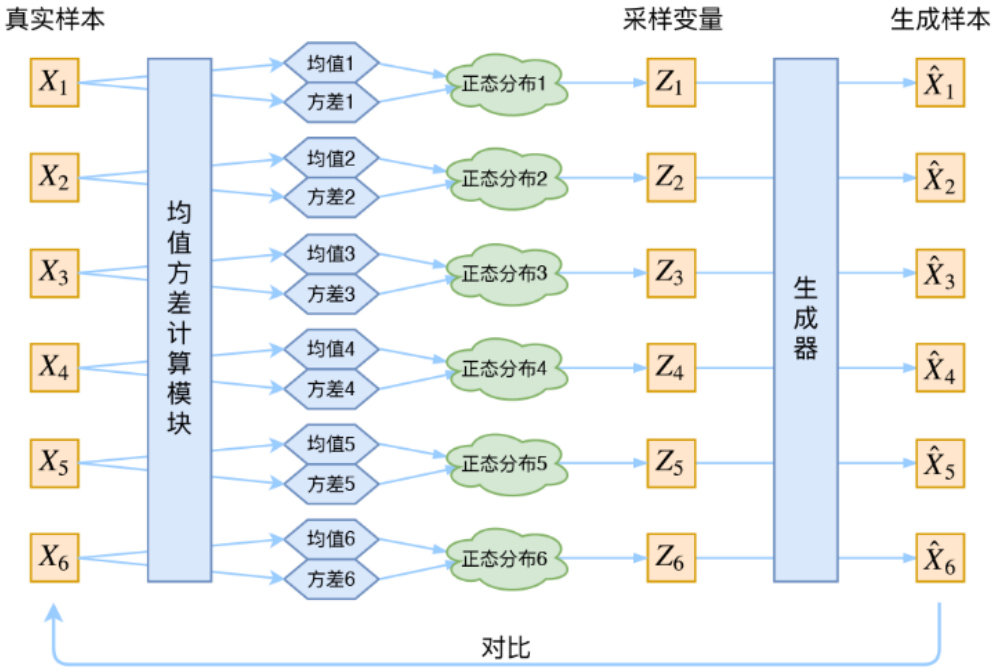
上节讲到假设隐变量服从正态分布，而正态分布只需要知道均值和方差，就能确定了，所以就构造一个 $X \xrightarrow{f} Z \sim N(\mu, \sigma^2 I) \xrightarrow{g} \hat{X}$ 的自编码器：



设计一个编码器，输入为原始数据，输出一组均值和方差来表示各分量独立的多维正态分布。这里用多维正态分布而不用一维正态分布是因为一个变量包含的信息太少了，k 维正态分布一次采样能有 k 个变量，包含更多信息，而且这个模型主要是生成数据不是压缩数据，所以维度再高，只要能采样，都不影响结果。然后从这个多维正态分布中进行采样得到隐变量 Z，输入到解码器，就可以输出结果了。

上面整个流程看似已经解决了自编码器的新数据生成问题，但实际上是不可能实现的，因为 $X_1, X_2, ..., X_n$ 如果变换到同一个分布 $p(Z)$，

我们无法判断从中采样出来的Z_i是不是对应的还是X_i，那么最小化$D(\hat{X}_i,X_i)$是有问题的。所以还需要改进：给每一个X_i指定一个Z_i，即编码器求的是$Z_i|X_i \sim N(\mu_i,\sigma_i^2 I)$而不是$Z \sim N(\mu,\sigma^2 I)$。这样$Z_i$经过解码器得到的$\hat{X}_i$就跟$X_i$对应了。请看下图：



接下来就可以用一个神经网络来构建编码器$\mu_i,\sigma_i = f(X_i)$，然后从$N(\mu_i,\sigma_i^2)$中进行抽样得到Z_i，最后用另一个神经网络拟合$X_i = g(Z_i)$就可以了。

3.3 重参数技巧

这是模型实现的一个技巧，论文中叫做 reparameterization trick。上面讲到要从中$N(\mu_i,\sigma_i^2)$采样一个Z_i，Z_i是这么算出来的，先从标准正态分布中抽样一个$\varepsilon_i \sim N(0,I)$（可以称为高斯噪声），再另$Z_i = \mu_i + \sigma_i \odot \varepsilon_i$，这个$Z_i$服从$N(\mu_i,\sigma_i^2)$。这么做看似是多此一举，实际编程中是必须的。因为虽然很多编程语言中正态分布是用标准正态分

布算出来的，但这个操作是隐藏的，也就说对整个模型来说只能看到采样出来的值，而不是表达式，也就不能进行求导操作，无法计算梯度，那就不能用反向传播算法来优化模型了。

3.4 分布标准化

我们来回顾一下这个算法的流程。首先，对一个数据X_i，进入到编码器求出一个分布（均值μ_i和方差σ_i）。然后从这个分布进行一次采样（$Z_i = \mu_i + \sigma_i \odot \varepsilon_i, \varepsilon_i \sim N(0, I)$）。接下来对$Z_i$进行解码，得到$\hat{X}_i$。最后计算$D(\hat{X}_i, X_i)$。很明显，最小化$D(\hat{X}_i, X_i)$这个过程是受到噪声$\varepsilon_i$影响的。不过这个噪声的强度$\sigma_i$（也就是方差）是用一个神经网络算出来的，所以最终模型用传统的优化算法来最小化$D(\hat{X}_i, X_i)$，一定会尽可能地让方差为 0，如果方差为 0，也就没有随机性了，不管怎么采样都得到一个确定的结果（也就是均值），那么拟合一个当然比拟合多个要简单，而且损失函数更小。下面给出证明：

假设只有一个样本X，

假设$\sigma = 0$，那么经过编码器后得到$\mu = f(X)$，采样然后做估计得到$\hat{X} = g(\mu)$，如果用 MSE 做损失函数，对上述过程，只需要$g = f^{-1}$就可以使$MSE\left(\hat{X}, X\right) = 0$。

假设$\sigma \neq 0$，经过编码器后得到$\mu, \sigma = f(X)$，采样后进入解码器得到$\hat{X} = g(\mu + \sigma \odot \varepsilon)$，也用 MSE 做损失函数。很明显，对$\forall \varepsilon$，$MSE\left(\hat{X}, X\right) = 0$仅当$g' = 0$，或$g = C$，$C$是常数。

但解码器不可能是常数函数，所以这个模型的最优解肯定是$\sigma = 0$，

这样模型也就退化成了自编码器，泛化能力很差。实际应用中为了解决模型泛化能力差，经常在损失函数中加一个正则项。VAE 模型的泛化能力差的能力差是因为$\sigma=0$，那么不让$\sigma=0$就可以了，即假设$Z|X\sim N(0,I)$，等价于在损失函数上加上一个正则项$KL(N(\mu,\sigma^2 I)||N(0,I))$（KL 散度用来描述两个分布的差距，下一节会介绍）。VAE 认为所有的$P(Z|X)$都向标准正态分布看齐，这样就防止了噪声为 0，模型失去泛化能力。

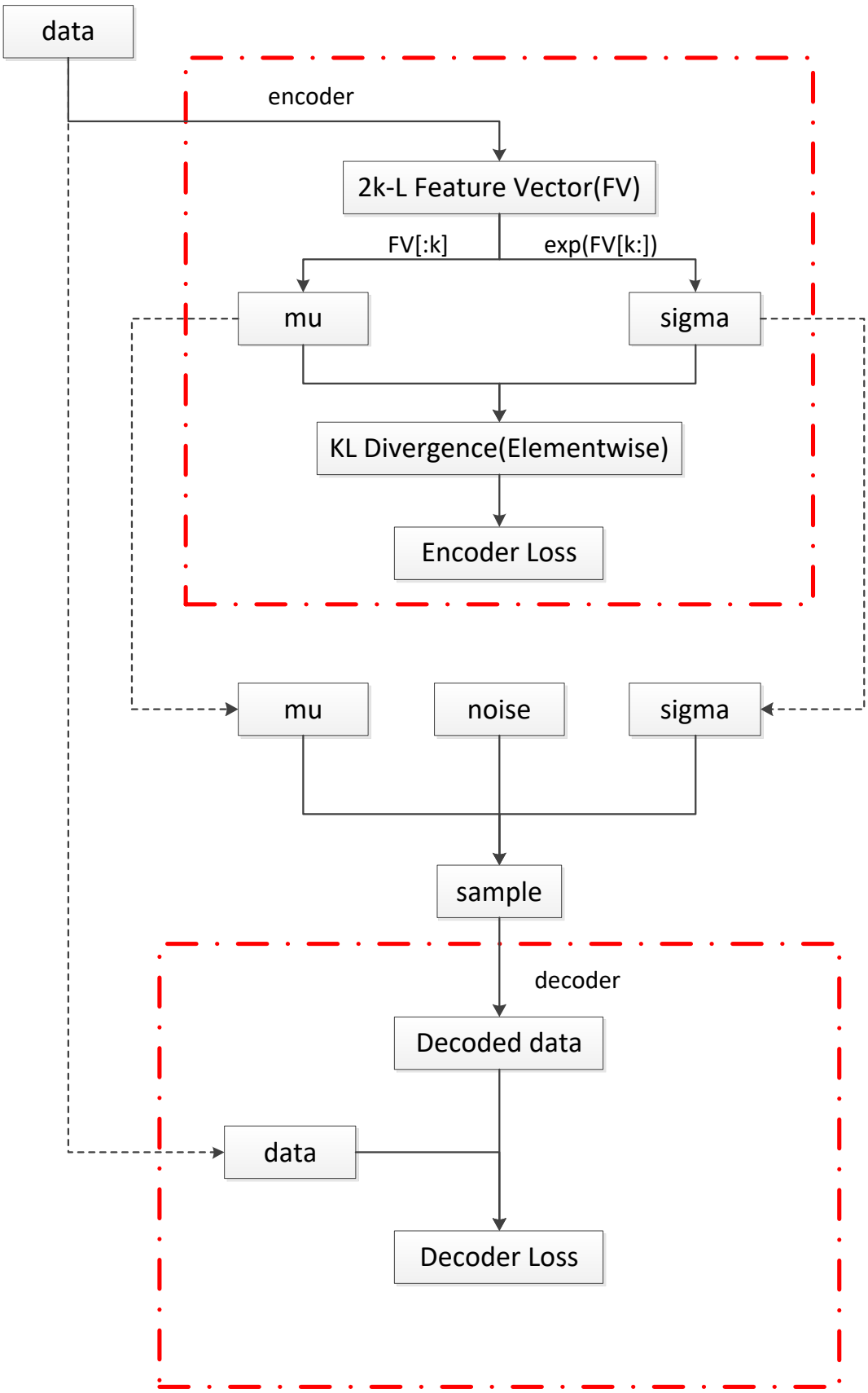
假设$Z|X\sim N(0,I)$还有一个至关重要的作用，那就是保证模型有“生成”能力。在 3.5.2.2 节中，介绍了生成模型和分布变换，讲到要建立一个简单分布到原始数据集分布的映射。但是 VAE 建立了 n 个简单分布到原始数据集分步的映射，那么在应用时就不知道在哪个分布中进行采样，然后解码。事实上，按上面步骤构建 VAE 模型并训练完成后，只需要在标准正态分布中进行采样，经解码后就能得到属于原始数据集分布的数据。

证明：

$$p(Z)=\sum_X p(Z|X)p(X)=\sum_X N(0,I)p(X)=N(0,I)\sum_X p(X)=N(0,I)$$

这样就满足了最开始的假设$Z\sim N(0,I)$，也满足$p(X)=\int_Z p(X|Z)p(Z)dZ$，所以从标准正态分布中采样来生成数据是合理的。

3.5VAE 结构图


data
encoder
2k-L Feature Vector(FV)
FV[:k]
exp(FV[k:])
mu
sigma
KL Divergence(Elementwise)
Encoder Loss
mu
noise
sigma
sample
decoder
Decoded data
data
Decoder Loss

图中红色虚框表示求解 Loss 的部分。虚线展现了两个模块之间数据共享的情况。可以看出图的上半部分是优化 Encoder 的部分，下面是优化 Decoder 的部分。

data 先进入 Encoder，输出一个长度为 2k 的向量，这个向量前 k 个元素认为是均值 mu，后 k 个元素认为是方差（exp 是为了保证方差非负）sigma，这样就生成了一个 k 维正态分布。用这个正态分布和标准正态分布求 KL 散度（相对熵），记为 Encoder Loss。

KL 散度的定义：

假设 $P(x),Q(x)$ 是随机变量 X 上的两个概率分布，它们的 KL 散度是

$$KL(P\|Q)=\sum_x P(x)\log(P(x)/Q(x))$$

在 VAE 模型中，事先假设了 Q 是标准正态分布，P 各分量独立的正态分布，所以可以化简为：

$$KL(N(\mu,\sigma^2 I)\|N(0,I))=-0.5*\sum_i(1+\log(\sigma_i^2)-\mu_i^2-\sigma_i^2)$$

在求出的正态分布中进行采样，采样算法是 $sampel=mu+sigma\odot noise, noise\sim N(0,I)$，sample 进入 Decoder，输出对原始数据的一个估计，并计算与原始数据的差距 Decoder Loss，损失函数可根据情况自己定义。

总体损失 $Loss=EncoderLoss+DecoderLoss$，用优化算法（常用 Adam）优化 Loss。

4. 生成式对抗网络 GAN

自从 Ian Goodfellow 在 14 年发表了论文 Generative Adversaial Networks 以来，生成式对抗网络 GAN 广受关注，尤其是人工智能三巨头之一的 Yann Lecun 在 Quora 曾说，他最激动的深度学习进展是生成式对抗网络。

GAN 的几篇基础论文：

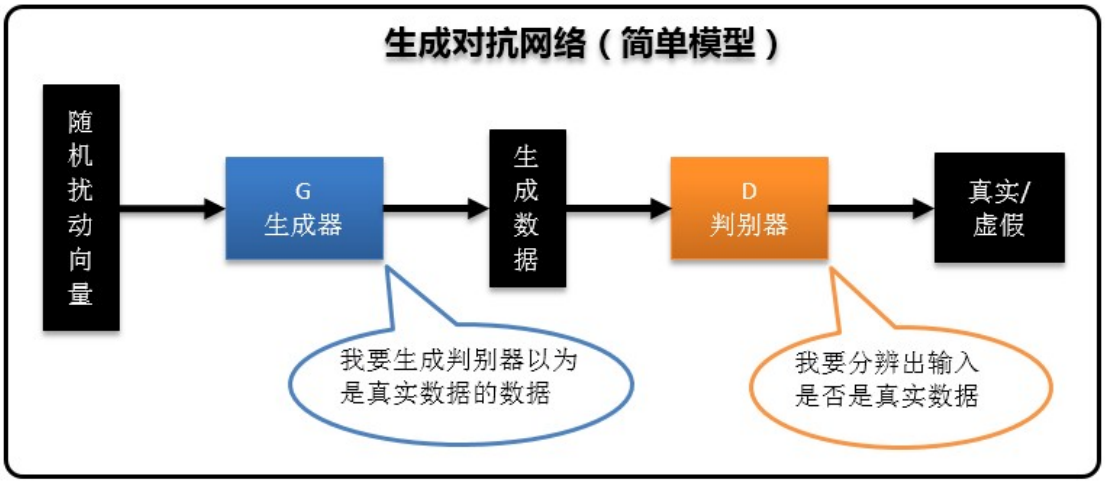
GAN: Goodfellow. "Generative adversarial nets". GAN 原始论文，重要性不用多说；

CGAN: Mehdi Mirza, Simon Osindero. "Conditional Generative Adversarial Nets".给 GAN 加上条件，使生成的样本符合预期；

DCGAN: Alec Radford, Luke Metz, Soumith Chintala."Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks".优化了网络结构，使网络更容易训练。

生成式对抗网络启发自博弈论中的二人零和博弈（two-player game），GAN 模型中的两位博弈方分别由生成式模型（generative model）和判别式模型（discriminative model）充当。生成模型 G 捕捉样本数据的分布，用服从某一分布（正态分布等）的噪声 Z 生成一个类似真实训练数据的样本，追求效果是越像真实样本越好；判别模型 D 是一个二分类器，估计一个样本来自于训练数据（而非生成数据）的概率，如果样本来自真实的训练数据，D 输出大概率，否则 D 输出小概率。可以做如下类比：生成模型 G 是假币制造团伙，专门制造假币，判别网络 D 好比警察，专门检测使用的货币是真币还是假币。G 的目标是想办法生成和真币一样的货币，使得 D 判断不出来，D 的目

标是想反复检测出来G生成的是假币。如图所示：



4.1 GAN 的训练过程

在理想情况下，最终博弈的结果是：生成器G可以生成足以以假乱真的数据$G(z)$，而判别器D难以判定生成器生成的数据是不是真实的。所以优化目标是：

$$\min_G \max_D V(D,G) = E_{x\sim pdata(x)}\left[\log D(x)\right] + E_{z\sim pz(z)}\left[\log(1-D(G(z)))\right]$$

x表示真实数据，z表示输入生成器G的噪声，而$G(z)$表示G生成的数据。

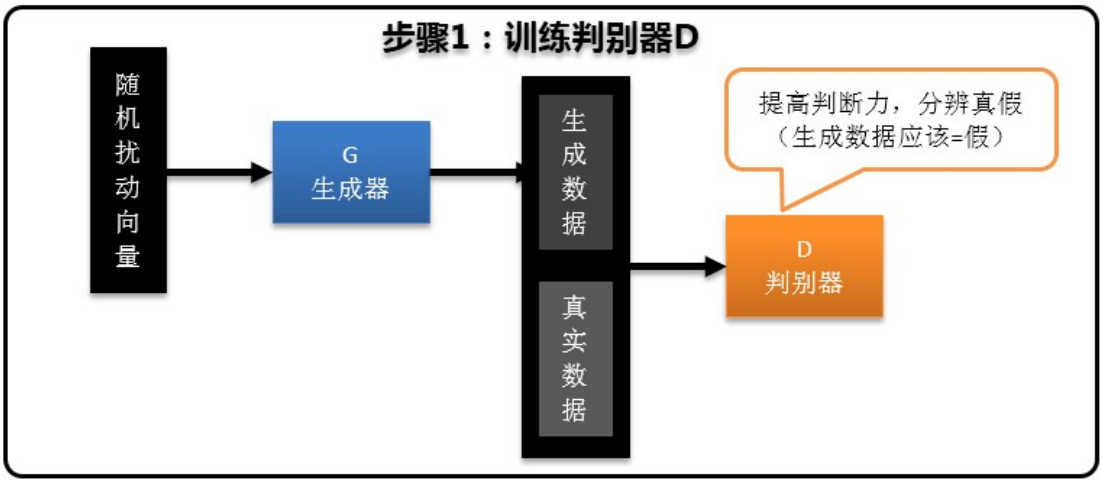
$D(x)$表示判别器D判断真实数据真实的概率，$D(G(z))$表示判别器判断生成数据是否真实的概率。

G的目的：生成希望判别器认为自己生成的数据是真的，也就是希望$D(G(z))$越大越好，所以生成器的目标是最小化$V(D,G)$。

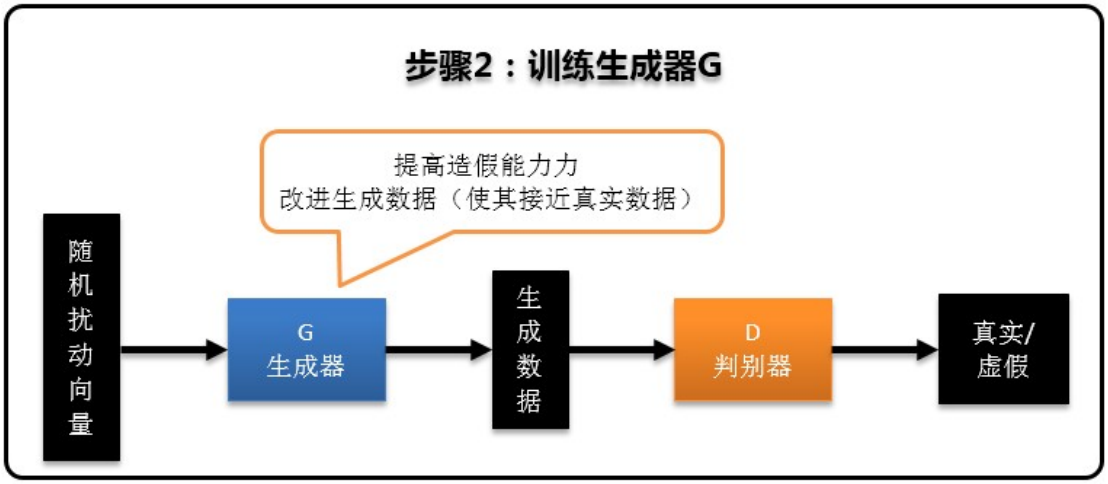
D的目的：判别器希望能正确判别输入数据的真假，所以希望$D(x)$越大越好，$D(G(z))$越小越好，所以判别器的目标是最大化$V(D,G)$。

训练过程如下：

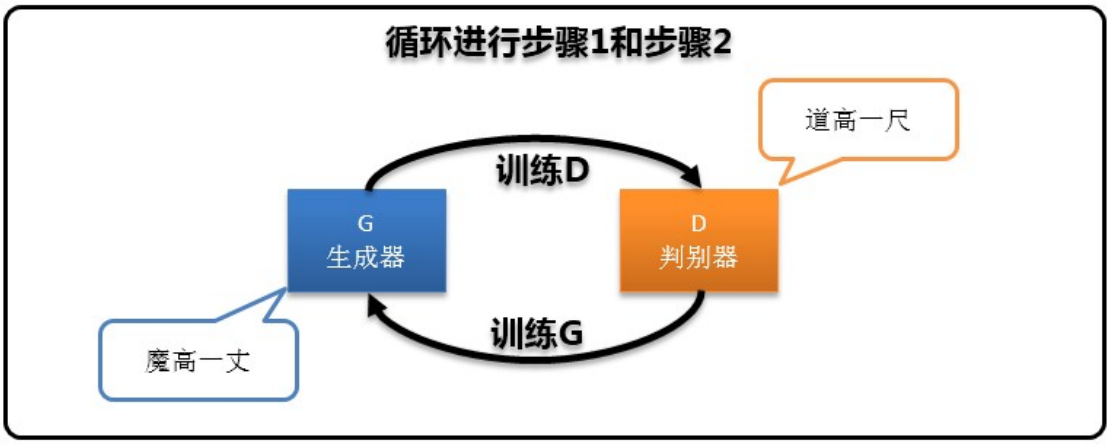
1. 随机生成一个扰动噪声，进入生成器生成数据，和真实数据一起进入判别器，让判别器判别真假



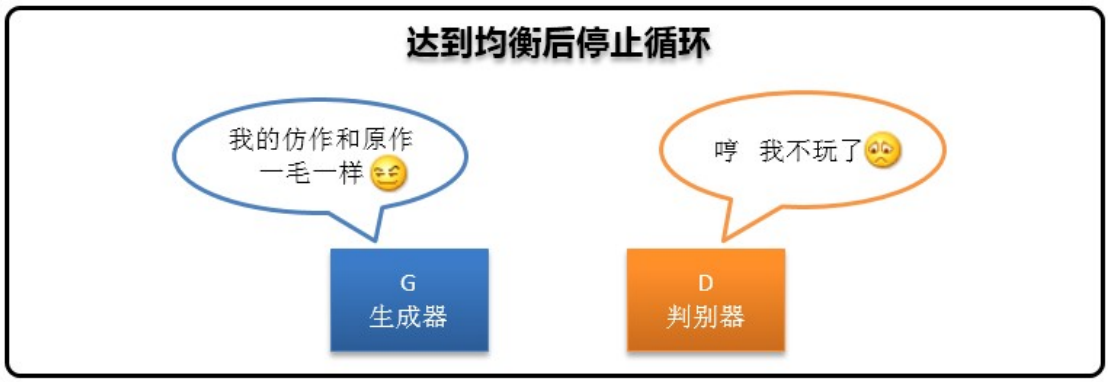
2. 判别器把结果反馈给生成器，生成器想办法进行改进



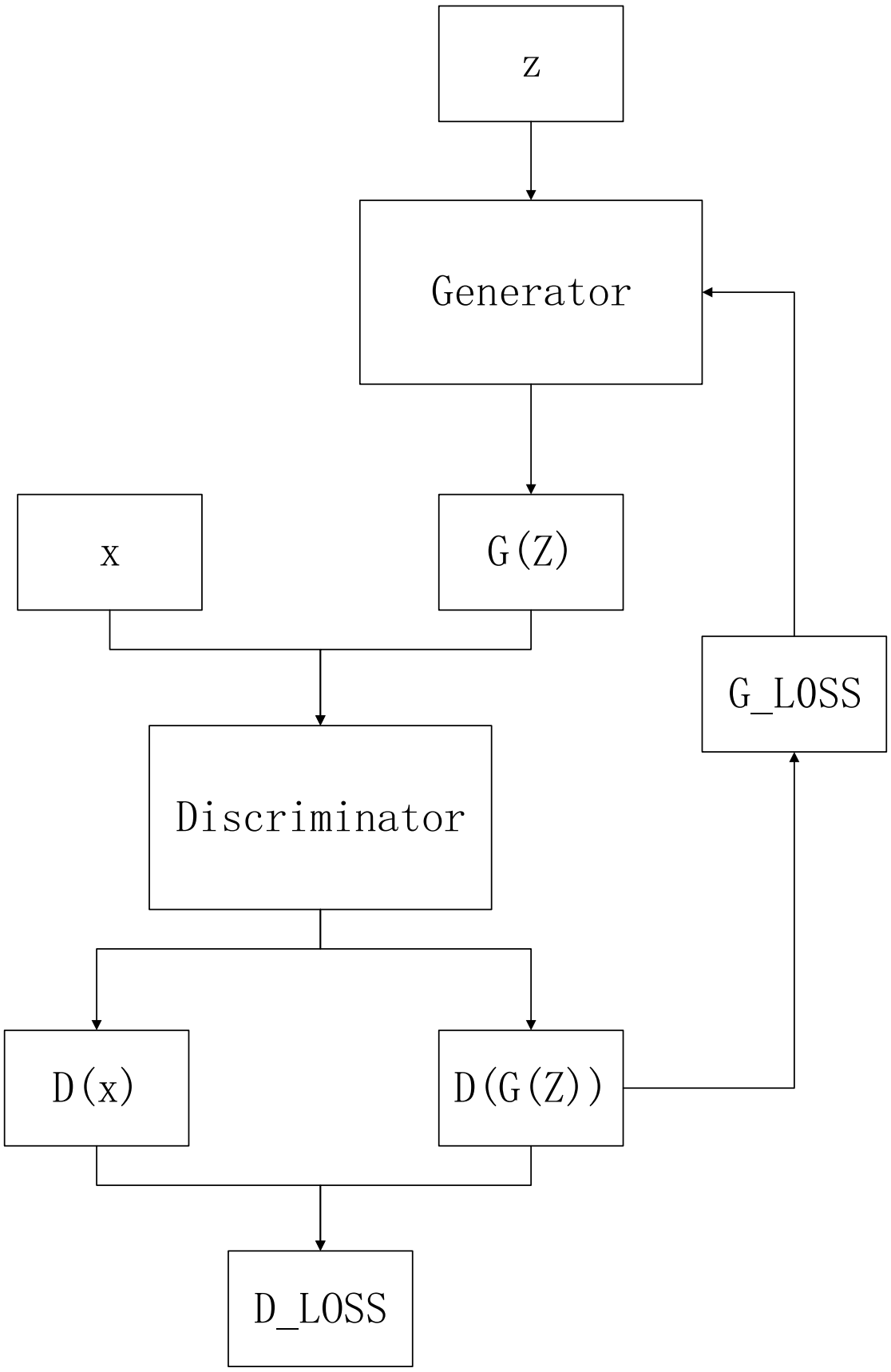
3. 交替训练判别器和生成器



4. 直到生成器胜出，判别器再也判断不了真假

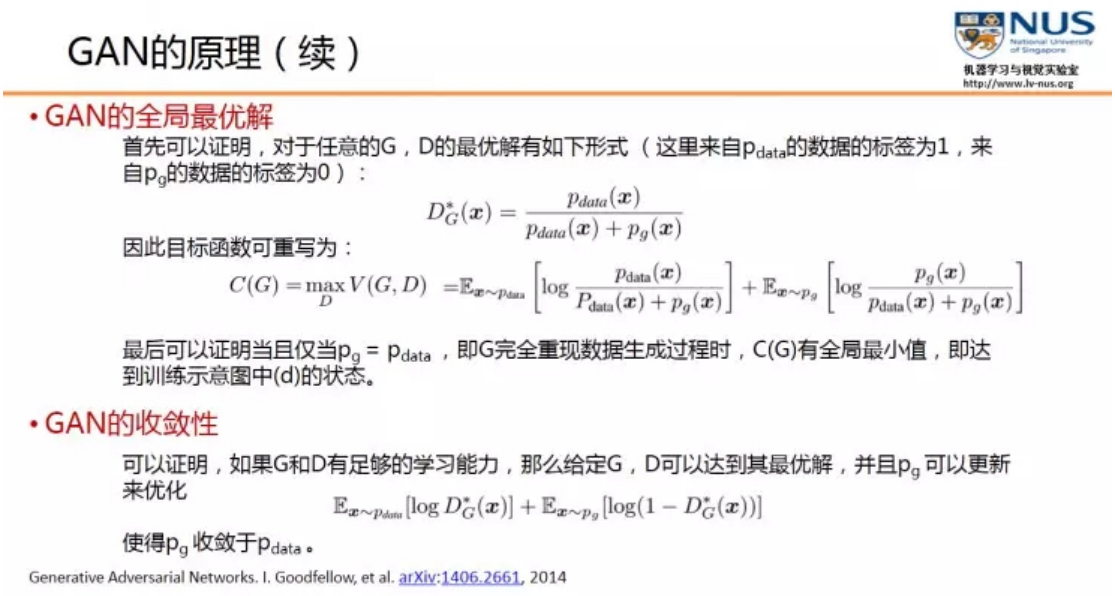

达到均衡后停止循环
我的仿作和原作
一毛一样
G
生成器
哼 我不玩了
D
判别器

4.2 GAN 结构图



4.3 GAN 的收敛性

GAN 是存在全局最优解的。这个全局最优解可以通过一些简单的分析得到。首先，如果固定G，那么D的最优解就是一个贝叶斯分类器。将这个最优解形式带入，可以得到关于G的优化函数。简单的计算可以证明，当产生的数据分布与真实数据分布完全一致时，这个优化函数达到全局最小值。



上述的收敛性证明基于三个条件

1. 训练时间是无限的

2. 生成器和判别器的能力是无限的

3. 直接更新生成器的分布

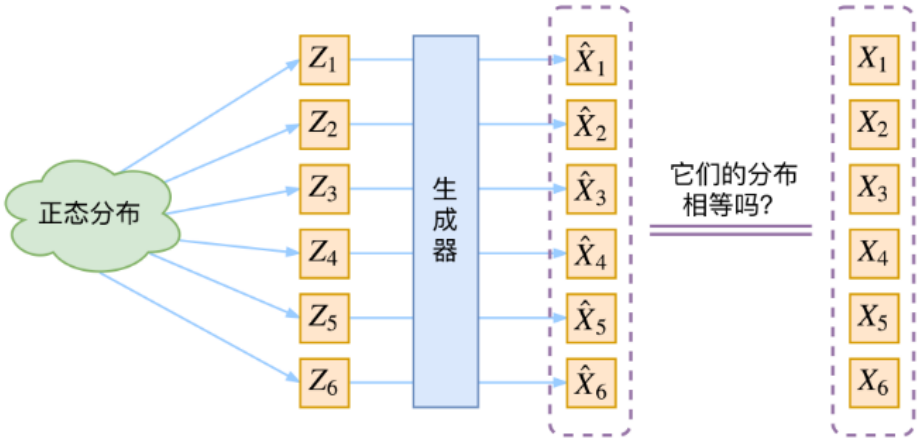
因为假设条件太过理想化，所以如何提升算法性能在实际应用中可能需要优化，这也是当前的研究热点。

5. GAN 和 VAE 的联系和区别

生成模型的本质是求出数据集的分布 $P(X)$，但是 $P(X)$ 不好求，所以现在主流方法是采用间接的方式求 $P(X)$：

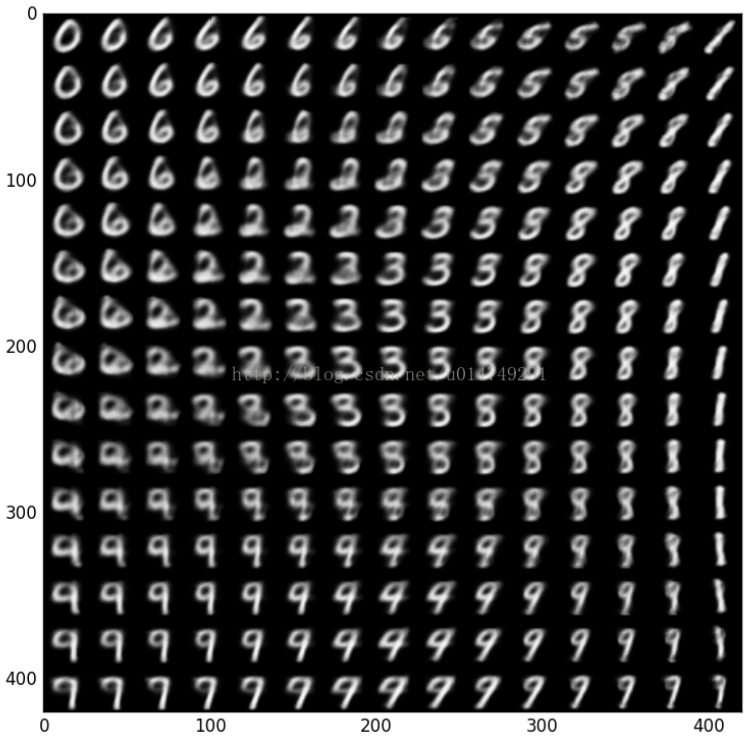
$$p(X)=\sum_{Z} p(X|Z)p(Z)$$

GAN 和 VAE 都是通过构造一个简单分布到复杂分布的映射来实现“生成”能力。



用上述方法构造一个生成器不难，但怎么判断生成数据的分布和样本数据的分布的相似程度呢？VAE 用了一个迂回的方法，先找到简单分布跟样本的关系，再用简单分布映射到样本，这样就可以用传统的损失函数来表述两者的差距了。而 GAN 则很暴力了，既然没有这样的一个度量，就自己训练一个度量出来，所以 GAN 的判别器只输出是真实样本的概率，具体怎么判断则由 GAN 自己来训练。

VAE 是有严格的数学论证的，所以模型稳定性很强，训练速度很快，同时编码器可以用来提取特征。但是更容易生成模糊的数据：



可变性也较差，如编码器的的损失 KL 散度是变分推导的结果，不能修改。且 VAE 优化的是变分下界，是有偏估计。

GAN 的论证都是很基础的那些东西，且基于理想条件，所以 GAN 的稳定性不好，很难训练，控制不好判别器生成器的优化速度容易模式崩塌。但是宽松的假设条件也是有好处的，GAN 的实际效果其实比 VAE 好，而且没有那么多统计上的假设导致 GAN 更能训练出真正逼近真实分布的模型。

VAE 就像是一幢已经造好的房子，只需要装修好就可以入住了，而 GAN 只是打造好了地基，在上面造茅房还是摩天大楼就要看施工队了。VAE 是精致的，数学让 VAE 坚不可摧，但也限制了 VAE 的性能。GAN 则给予了模型很大的自主性，一切都让模型自己训练出来，这种做法也被认为是通向“智能”的大门。

参考文献

[1] [变分自编码器（一）：原来是这么一回事](https://kexue.fm/archives/5253)，苏剑林