# 生成对抗网络

2018.11.13

GANs中包含了两个模型，一个是生成模型$G$，另一个是判别模型$D$，下面通过一个生成图片的例子来解释两个模型的作用：

- 生成模型$G$：不断学习训练集中真实数据的概率分布，目标是将输入的随机噪声转化为可以以假乱真的图片（生成的图片与训练集中的图片越相似越好）
- 判别模型$D$：判断一个图片是否是真实的图片，目标是将生成模型$G$产生的“假”图片与训练集中的“真”图片分辨开。

GANs的实现方法是让$D$和$G$进行博弈，训练过程中通过相互竞争让这两个模型同时得到增强。由于判别模型$D$的存在，使得 $G$ 在没有大量先验知识以及先验分布的前提下也能很好的去学习逼近真实数据，并最终让模型生成的数据达到以假乱真的效果（即$D$无法区分$G$生成的图片与真实图片，从而$G$和$D$达到某种纳什均衡）。

# 目标函数

GANs中生成模型和判别模型的选择没有强制限制，在Ian的论文中，判别模型$D$和生成模型$G$均采用多层感知机。GANs定义了一个噪声$p_z(x)$ 作为先验，用于学习生成模型$G$ 在训练数据$x$上的概率分布$p_g$，$G(z)$表示将输入的噪声$z$映射成数据（例如生成图片）。$D(x)$代表$x$ 来自于真实数据分布$p_{data}$而不是$p_g$的概率。据此，优化的目标函数定义如下minmax的形式：

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$



原论文在参数的更新过程，是对$D$更新$k$次后，才对$G$更新1次。上式中的minmax可理解为当更新$D$时，需要最大化上式，而当更新$G$时，需要最小化上式，详细解释如下：

- 在对判别模型$D$的参数进行更新时：对于来自真实分布$p_{data}$的样本$x$而言，我们希望$D(x)$的输出越接近于1越好，即$logD(x)$越大越好；对于通过噪声$z$生成的数据$G(z)$而言，我们希望$D(G(z))$尽量接近于0（即$D$能够区分出真假数据），因此$log(1-D(G(z)))$也是越大越好，所以需要$maxD$。

$$\max_D V(D,G) = E_{x\sim p_{data}\ (x)}[log(D(x))] + E_{z\sim p_z\,(z)}[log(1-D(G(z)))]$$

- 在对生成模型$G$的参数进行更新时：我们希望$G(z)$尽可能和真实数据一样，即$p_g = p_{data}$。因此我们希望$D(G(z))$尽量接近于1，即$log(1-D(G(z)))$越小越好，所以需要$minG$。需要说明的是，$logD(x)$是与$G(z)$无关的项，在求导时直接为0。

$$\min_G V(D,G) = E_{z\sim p_z\,(z)}[log(1-D(G(z)))]$$

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$



- 整个式子由两项构成。x表示真实图片，z表示输入G网络的噪声，而G(z)表示G网络生成的图片。
- D(x)表示D网络判断**真实图片是否真实**的概率（因为x就是真实的，所以对于D来说，这个值越接近1越好）。而D(G(z))是**D网络判断G生成的图片的是否真实的概率。**
- G的目的：上面提到过，D(G(z))是**D网络判断G生成的图片是否真实的概率**，G应该希望自己生成的图片“越接近真实越好”。也就是说，G希望D(G(z))尽可能得大，这时V(D, G)会变小。因此我们看到式子的最前面的记号是min_G。
- D的目的：D的能力越强，D(x)应该越大，D(G(x))应该越小。这时V(D,G)会变大。因此式子对于D来说是求最大(max_D)

**算法 13.1:** 生成对抗网络的训练过程

**输入:** 训练集 $\mathcal{D}$，对抗训练迭代次数 $T$，每次判别网络的训练迭代次数 $K$，小批量样本数量 $M$

随机初始化 $\theta, \phi$;

**for** $t \leftarrow 1$ **to** $T$ **do**

  // 训练判别网络 $D(\mathbf{x}, \phi)$

  **for** $k \leftarrow 1$ **to** $K$ **do**

    // 采集小批量训练样本

    从训练集 $\mathcal{D}$ 中采集 $M$ 个样本 $\{\mathbf{x}^{(m)}\}, 1 \leq m \leq M$;

    从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 $M$ 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

    使用随机梯度上升更新 $\phi$，梯度为

$$\frac{\partial}{\partial \phi}\left[\frac{1}{M}\sum_{m=1}^{M}\left(\log D(\mathbf{x}^{(m)}, \phi) + \log\left(1 - D\big(G(\mathbf{z}^{(m)}, \theta), \phi\big)\right)\right)\right];$$

  **end**

  // 训练生成网络 $G(\mathbf{z}, \theta)$

  从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 $M$ 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

  使用随机梯度上升更新 $\theta$，梯度为

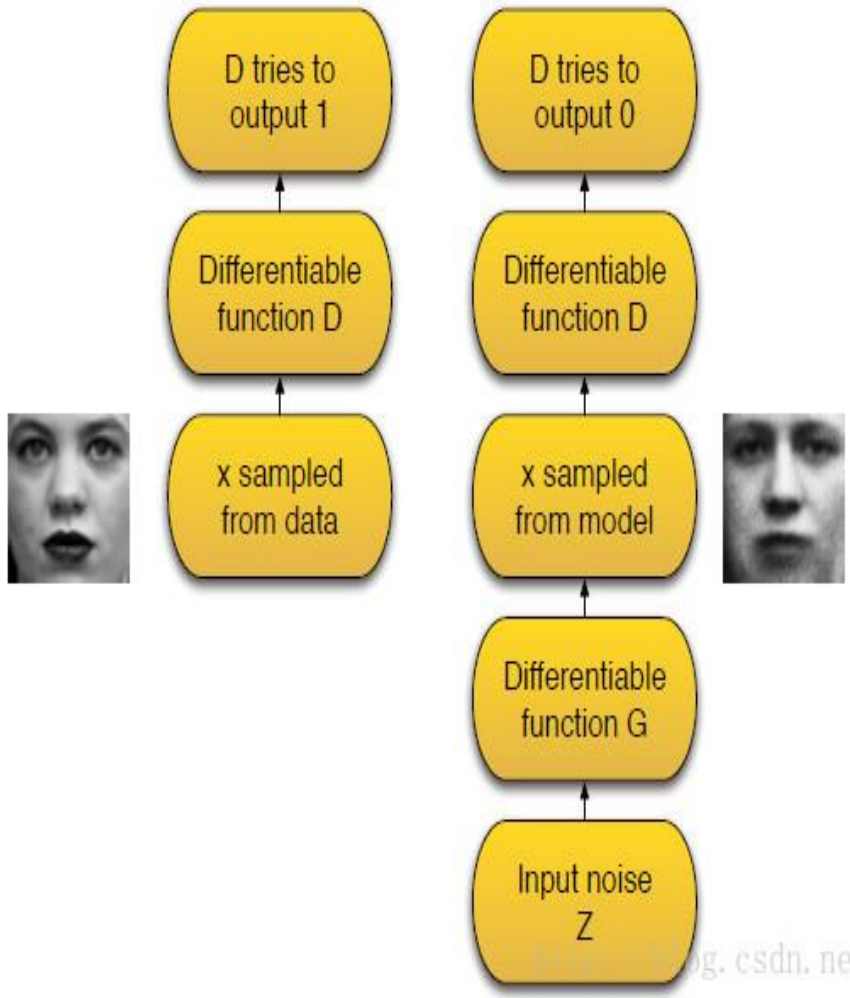
$$\frac{\partial}{\partial \theta}\left[\frac{1}{M}\sum_{m=1}^{M} D\big(G(\mathbf{z}^{(m)}, \theta), \phi\big)\right];$$

**end**

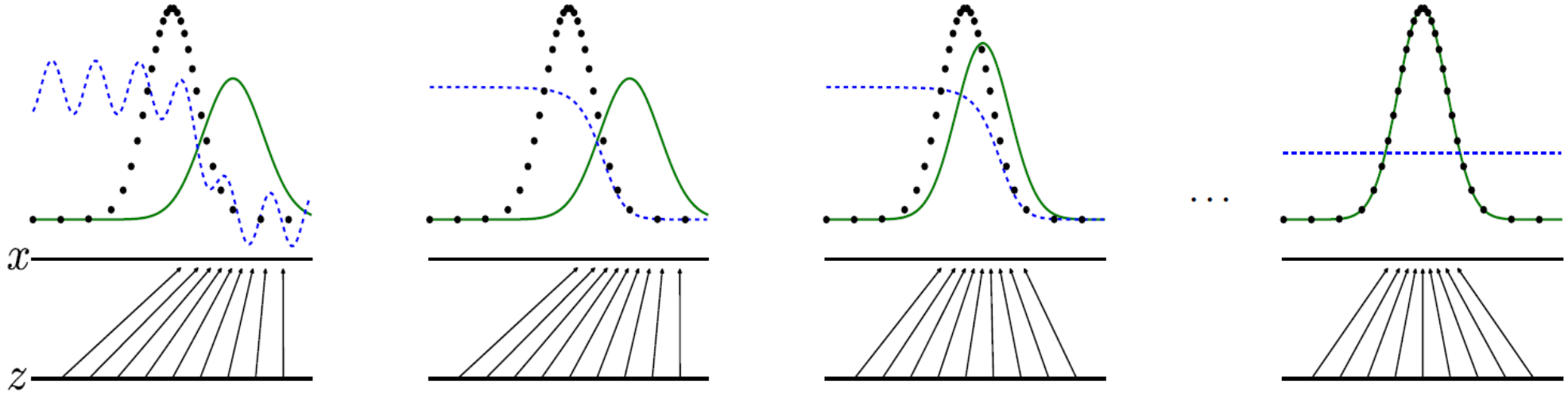
**输出:** 生成网络 $G(\mathbf{z}, \theta)$

原论文中对GANs理论上的有效性进行了分析，即当固定$G$更新$D$时，最优解为$D^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$；而在更新$G$时，目标函数取到全局最小值当且仅当$p_g = p_{data}$。最后两个模型博弈的结果是$G$可以生成以假乱真的数据$G(z)$。而$D$难以判定$G$生成的数据是否真实，即$D(G(z)) = 0.5$。

# Adversarial Nets Framework




第一阶段只有判别模型$D$参与。将训练集中的样本$x$作为$D$的输入，输出0-1之间的某个值，数值越大意味着样本$x$为真实数据的可能性越大。在这个过程中，我们希望$D$尽可能使输出的值逼近1。第二阶段中，判别模型$D$和生成模型$G$都参与，首先将噪声$z$输入$G$，$G$从真实数据集里学习概率分布并产生假的样本，然后将假的样本输入判别模型$D$，这一次$D$将尽可能输入数值0。所以在这个过程中，判别模型$D$相当于一个监督情况下的二分类器，数据要么归为1，要么归为0。





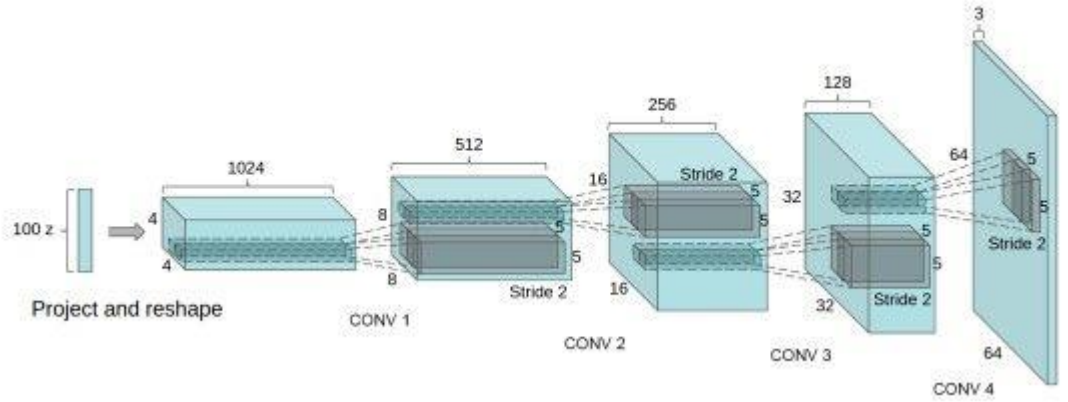

其中，蓝色虚线为判别模型$D$的分布，黑色虚线为真实数据的分布$p_{data}$，绿色实线为生成模型$G$学习的分布$p_g$。下方的水平线为均匀采样噪声$z$的区域，上方的水平线为数据$x$的区域。朝上的箭头表示将随机噪声转化成数据，即$x = G(z)$。从图（a）到图（b）给出了一个GANs的收敛过程。图(a)中$p_g$与$p_{data}$存在相似性，但还未完全收敛，$D$是个部分准确的分类器。图(b)中，固定$G$更新$D$，收敛到$D^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$。图(c)中对$G$进行了1次更新，$D$的梯度引导$G(z)$移向更可能分类为真实数据的区域。图(d)中，训练若干步后，若$G$和$D$均有足够的capacity，它们接近某个稳定点，此时$p_g = p_{data}$。判别模型将无法区分真实数据分布和生成数据分布，即$D(x) = 0.5$。

# Deep Convolutional Generative Adversarial Networks

DCGAN的原理和GAN是一样的，这里就不在赘述。它只是把上述的G和D换成了两个卷积神经网络（CNN）。但不是直接换就可以了，DCGAN对卷积神经网络的结构做了一些改变，以提高样本的质量和收敛的速度，这些改变有：

- 取消所有pooling层。G网络中使用转置卷积（transposed convolutional layer）进行上采样，D网络中用加入stride的卷积代替pooling。
- 在D和G中均使用batch normalization
- 去掉FC层，使网络变为全卷积网络
- G网络中使用ReLU作为激活函数，最后一层使用tanh
- D网络中使用LeakyReLU作为激活函数

DCGAN中的G网络示意：



GAN网络汇总
https://github.com/zhangqianhui/AdversarialNetsPapers