第十三章、变分自动编码器与生成对抗网络

赵涵

2023 年 5 月 11 日

1 变分自动编码器

变分自动编码器（Variational Auto-Encoder）被广泛应用于机器学习中复杂模型的推断，是机器学习中的一个十分重要的工具。利用变分自动编码器我们可以在学习训练图片之后，自动产生相似的图片。在介绍VAE 之前，我们先做一个铺垫，介绍一下什么是自动编码器，然后在此基础上，在引入目前深度生成模型的两大支柱——VAE （生成模型在机器学习领域，现在大放异彩，我们这里介绍的两个模型属于热度已经下去了的模型，现在在自然语言处理，有GPT3，BERT等大型深度模型）。

1.1 自动编码器

自动编码器（Auto Encoder）是最简易的深度模型，输入一个数据，通过神经网络进行编码，得到一个向量，然后在通过一个神经网络，进行译码，复原数据点，把然后两个过程拼接在一起，就构成了自动编码器。对于自动编码器来说，编码器和译码器对应的神经网络，一般是对称的，换句话说，译码是解码的逆过程。损失函数就是输入数据与输出数据，两者之间的差别，衡量两者的差别，可以选择均方误差，也可以选择交叉熵。对于图片的训练，一般选择使用GPU加速，例如在python 环境下，采用Tensorflow 或者Pytorch 深度学习库，然后借助计算机的GPU 共同实现。下面我们给出一个简易版的自动编码器的结构：

- (encoder): Sequential:
  1. Linear(in_features=784, out_features=256, bias=True)
  2. ReLU()
  3. Linear(in_features=256, out_features=20, bias=True)
  4. ReLU()
- (decoder): Sequential:
  1. Linear(in_features=20, out_features=256, bias=True)
  2. ReLU()
  3. Linear(in_features=256, out_features=784, bias=True)
  4. Sigmoid()

这里我们采用mini-batch梯度下降，看一下训练AE 的损失函数的变化（1）： AE的缺点是很明显的，只具备编码功能，不具备生成功能，然后作为一个生成模型，不具备生成功能，所以只能作为一个玩具模型，为了引入变分自动编码器。并且由于神经网络的结构是任意的，所以调参是非常困难的。

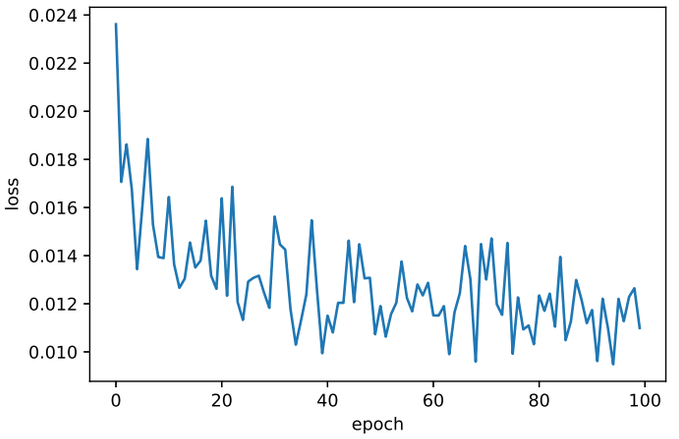

Figure 1: 自动编码器在小批量梯度下降时对应的损失函数变化。

1.2 变分自动编码器

我们先介绍变分自动编码器的数学基础。假定X 为样本集合；θ是模型参数；Z是隐变量。

$$P(X|\theta) = \int_Z P(X, Z|\theta)dZ \tag{1}$$

根据贝叶斯公式：

$$P(Z|X, \theta) = \frac{P(Z, X|\theta)}{P(X|\theta)} \tag{2}$$

我们对两边同时取对数：

$$\log P(Z|X, \theta) = \log P(Z, X|\theta) - \log P(X|\theta) \tag{3}$$

我们有：

$$\log P(X|\theta) = \log P(Z, X|\theta) - \log P(Z|X, \theta) \tag{4}$$

隐变量是我们人为加入的，所以我们可以对隐变量引入一个先验分布$q(Z)$（$q(Z) \neq 0$）。

$$\log P(X|\theta) = \log \frac{P(Z, X|\theta)}{q(Z)} - \log \frac{P(Z|X, \theta)}{q(Z)} \tag{5}$$

对左右两边的隐变量同时积分，目的是去掉我们人为引入的隐变量$q(Z)$。

$$left = \int_Z q(Z) \log P(X|\theta)dZ = \log P(X|\theta) \int_Z q(Z)dZ = \log P(X|\theta) \tag{6}$$

$$right = \int_Z q(Z) \log \frac{P(Z, X|\theta)}{q(Z)}dZ - \int_Z q(Z) \log \frac{P(Z|X, \theta)}{q(Z)}dZ \tag{7}$$

第一项是ELBO（evidence lower bound），第二项是KL 散度，在之前讲解EM 算法和变分推断时，都已经详细介绍了，这里是第三次出现了，我们不在赘述。所以我们有：

$$\log P(X|\theta) = ELBO + KL(q(Z)||P(Z|X, \theta)) \tag{8}$$

VAE的目的就是使用一个先验分布q，去近似真实分布p，方法是借助神经网络，这就是VAE的基本内容，关于先验的分布的选择，多种多样，由此诞生了各种VAE，我们接下来介绍一个最基本的，也是最常用的VAE 模型，把$q(Z)$ 选定为一个多维高斯分布。

对于一个模型，首先就是讨论一下它的损失函数，我们在之前已经知道，最大化ELBO和最小化KL散度是等价的，所以我们这里选择最小化KL散度作为损失函数的其中一项，另一项取成重构误差，即从自动编码器而来，输入的数据与输出的数据，要尽可能地保持一致，这样才符合编码器的意义。总的来说，损失函数为：

- 重构误差（the reconstruction loss）；
- KL散度（the KL divergence），我们这里认为先验分布为多为高斯分布，需要注意，这里的先验分布不是唯一的，高斯分布只是作为一个例子。

我们从最小化隐变量的先验分布与后验分布的KL 散度出发，通过化简，得到我们上述的损失函数的两个部分：

$$\langle \phi^*, \theta^* \rangle = \arg\min_{\theta} KL(q_\phi(Z)||P(Z|X,\theta)) \tag{9}$$

我们接下来做如下的化简：

$$KL(q_\phi(Z)||P(Z|X,\theta)) = \int_{q(Z)} q(Z)\log\frac{q(Z)}{P(Z|X,\theta)}dZ \tag{10}$$

$$= \int_{q(Z)} q(Z)\log q(Z)dZ - \int_{q(Z)} q(Z)\log P(Z|X,\theta)dZ \tag{11}$$

$$= \int_{q(Z)} q(Z)\log q(Z)dZ - \int_{q(Z)} q(Z)\log\frac{P(X|Z,\theta)P(Z,\theta)}{P(X|\theta)}dZ \tag{12}$$

$$= \int_{q(Z)} q(Z)\log q(Z)dZ - \int_{q(Z)} q(Z)\log P(X|Z,\theta)dZ - \int_{q(Z)} q(Z)\log P(Z,\theta)dZ + \int_{q(Z)} q(Z)\log P(X|\theta)dZ \tag{13}$$

$$= \int_{q(Z)} q(Z)\log q(Z)dZ - \int_{q(Z)} q(Z)\log P(X|Z,\theta)dZ - \int_{q(Z)} q(Z)\log P(Z,\theta)dZ + \log P(X|\theta) \tag{14}$$

$$= \int_{q(Z)} q(Z)\log\frac{q(Z)}{P(Z,\theta)}dZ - \int_{q(Z)} q(Z)\log P(X|Z,\theta)dZ + \log P(X|\theta) \tag{15}$$

$$= KL(q(Z)||P(Z|\theta)) - E_{q(Z)}[\log P(X|Z,\theta)] + \log P(X|\theta) \tag{16}$$

因为$P(X|\theta)$是一个常数，所以我们对KL 散度进一步化简，得到如下的表达式：

$$loss = -E_{q(Z)}[\log P(X|Z,\theta)] + KL(q(Z)||P(Z|\theta)) \tag{17}$$

对于先验分布，我们有一个事实，就是$q(Z) = q(Z|X)$，所以有：

$$loss = -E_{q(Z|X)}[\log P(X|Z,\theta)] + KL(q(Z|X)||P(Z|\theta)) \tag{18}$$

第一项是重构误差，很明显可以看出是编码后的分布，和解码后的分布，进行比较，这里的选择也是不唯一的，一般选择交叉熵函数，偶尔也选择最小均方误差作为重构误差，主要还是根据处理问题的形式。通过数学表述，我们可以清晰地看到，可以通过对X的采样得到隐变量Z，反过来，又可以通过采样Z，去得到新的X，这就是第一项表达式的意义。当然，我们期待着第一项的值越大越好，换句话说，重构误差为0，输入模型的数据，与输出的完全相同。

我们把选取的先验高斯分布代入，用z 表示隐藏变量的一个分量，x表示数据集X 的任意一个样本，我们有：

$$P(z|\mu_P,\sigma_P) = \frac{1}{\sqrt{2\pi\sigma_P^2}}\exp(-\frac{(z-\mu_P)^2}{2\sigma_P^2}) \tag{19}$$

$$q(z|x,\mu_q,\sigma_q) = \frac{1}{\sqrt{2\pi\sigma_q^2}}\exp(-\frac{(z-\mu_q)^2}{2\sigma_q^2}) \tag{20}$$

这里μ_P, σ_P是先验分布的参数，因为是人为选择的，可以视为常数。而μ_q, σ_q 是通过编码器计算得到的，对于单个样本来说，也可以认为是一个常数。把这两个表达式代入到损失函数的第二项化简，得到如下的表达式：

$$KL(q(z|x,\mu_q,\sigma_q)||P(z|\mu_P,\sigma_P)) = \int_{q(z)} q(z)\log\frac{q(z)}{P(z,\theta)}dz \tag{21}$$

$$= \int \frac{1}{\sqrt{2\pi\sigma_q^2}}\exp(-\frac{(z-\mu_q)^2}{2\sigma_q^2})\log\frac{\frac{1}{\sqrt{2\pi\sigma_q^2}}\exp(-\frac{(z-\mu_q)^2}{2\sigma_q^2})}{\frac{1}{\sqrt{2\pi\sigma_P^2}}\exp(-\frac{(z-\mu_P)^2}{2\sigma_P^2})}dz \tag{22}$$

$$= \int \frac{1}{\sqrt{2\pi\sigma_q^2}}\exp(-\frac{(z-\mu_q)^2}{2\sigma_q^2})\times\{-\frac{1}{2}\log(2\pi)-\log(\sigma_q)-\frac{(z-\mu_q)^2}{2\sigma_q^2}+\frac{1}{2}\log(2\pi) \tag{23}$$

$$+\log(\sigma_P)+\frac{(z-\mu_P)^2}{2\sigma_P^2}\}dz \tag{24}$$

$$= \int \frac{1}{\sqrt{2\pi\sigma_q^2}}\exp(-\frac{(z-\mu_q)^2}{2\sigma_q^2})\times\{-\frac{(z-\mu_q)^2}{2\sigma_q^2}+\log(\frac{\sigma_P}{\sigma_q})+\frac{(z-\mu_P)^2}{2\sigma_P^2}\}dz \tag{25}$$

$$= E_{q(z)}\{-\frac{(z-\mu_q)^2}{2\sigma_q^2}+\log(\frac{\sigma_P}{\sigma_q})+\frac{(z-\mu_P)^2}{2\sigma_P^2}\} \tag{26}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2\sigma_q^2}E_{q(z)}\{(z-\mu_q)^2\}+\frac{1}{2\sigma_P^2}E_{q(z)}\{(z-\mu_P)^2\} \tag{27}$$

又因为方差$\sigma_q^2 = E_{q(z)}\{(z-\mu_q)^2\}$，所以有：

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{\sigma_q^2}{2\sigma_q^2}+\frac{1}{2\sigma_P^2}E_{q(z)}\{(z-\mu_P)^2\} \tag{28}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{1}{2\sigma_P^2}E_{q(z)}\{(z-\mu_q+\mu_q-\mu_P)^2\} \tag{29}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{1}{2\sigma_P^2}E_{q(z)}\{(z-\mu_q)^2+(\mu_q-\mu_P)^2+2(z-\mu_q)(\mu_q-\mu_P)\} \tag{30}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{1}{2\sigma_P^2}E_{q(z)}\{(z-\mu_q)^2\}+E_{q(z)}\{(\mu_q-\mu_P)^2\}+2E_{q(z)}\{(z-\mu_q)(\mu_q-\mu_P)\} \tag{31}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{1}{2\sigma_P^2}\{\sigma_q^2+(\mu_q-\mu_P)^2+2*0*(\mu_q-\mu_P)\} \tag{32}$$

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{\sigma_q^2+(\mu_q-\mu_P)^2}{2\sigma_P^2} \tag{33}$$

令$\mu_P = 0, \sigma_P = 1$，最终得到第二项的表达式：

$$= \log(\frac{\sigma_P}{\sigma_q})-\frac{1}{2}+\frac{\sigma_q^2+(\mu_q-\mu_P)^2}{2\sigma_P^2} \tag{34}$$

$$= \log(\frac{1}{\sigma_q})-\frac{1}{2}+\frac{\sigma_q^2+\mu_q^2}{2} \tag{35}$$

$$= -\frac{1}{2}[\log(\sigma_q^2)+1-\sigma_q^2-\mu_q^2] \tag{36}$$

一个简单的模型构架，如下图（2）： 变分自动编码器在编码采样的过程中，存在一个重参数化技巧，我们在此直接给出具体做法（3）： 之所以这样做，目的是为了做梯度反向传播时，不会中断，传播过程如下图（4）： 最终，我们通过更新，为了让先验分布与后验的真实分布，趋于一致，如下图（6）：

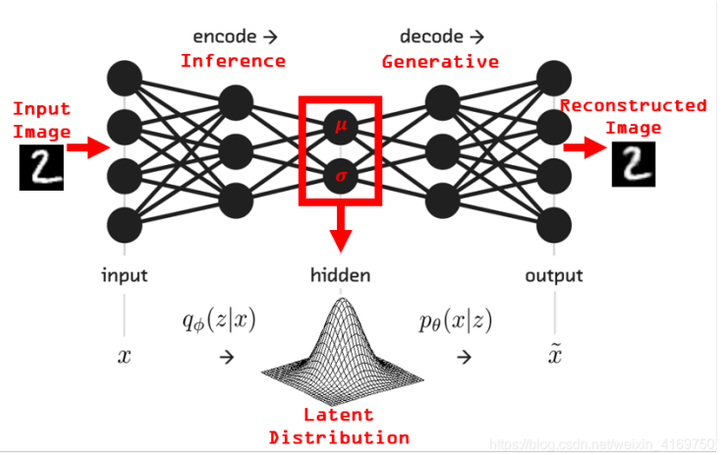

Figure 2: 变分自动编码器的模型架构。

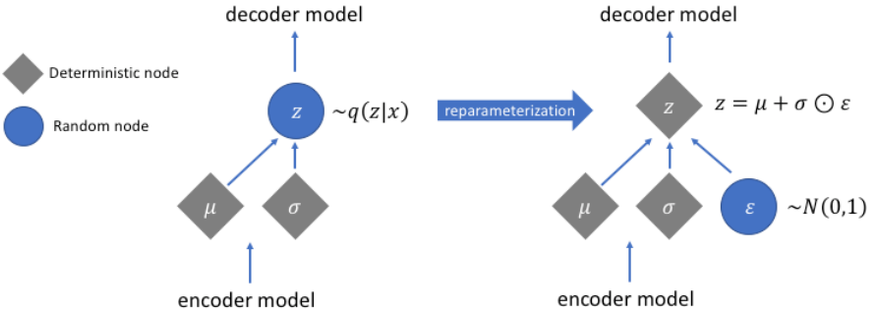

Figure 3: 重参数化技巧。

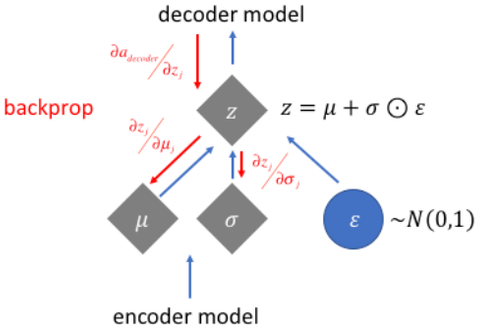

Figure 4: 重参数化方法在反向传播过程中的体现。

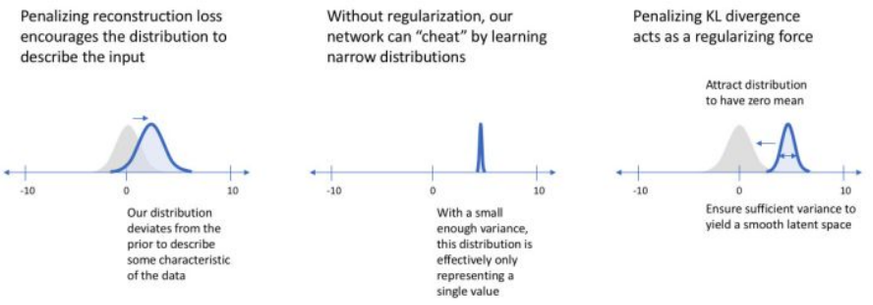

Figure 5: 先验分布与后验分布的逐步趋于一致。

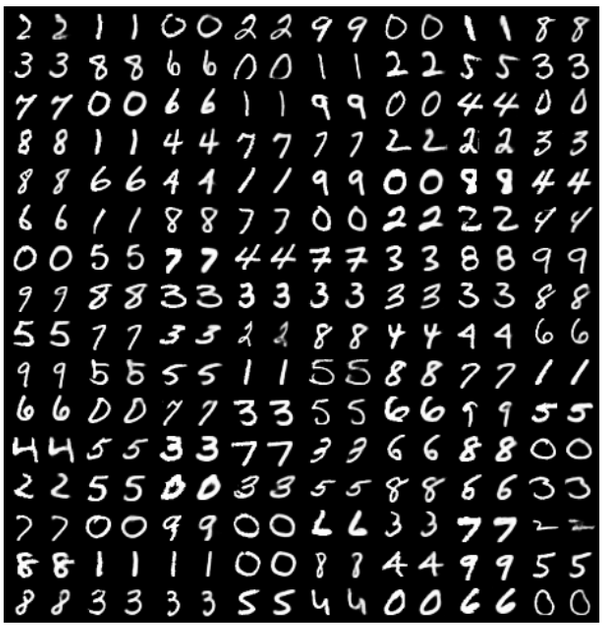

Figure 6: 卷积VAE：输入手写体，重构的图片。

以上我们介绍了VAE的主要内容。在一般的全连接层基础上，可以引入卷积操作，可以对图片处理的更加准确与高效。我们对卷积操作已经在深度学习时，进行了简单的介绍，我们这里直接给出卷积VAE 在图片重构与生成的结果：

2 生成对抗网络

五年前，Generative Adversarial Networks（GANs）在深度学习领域掀起了一场革命。这场革命产生了一些重大的技术突破。Ian Goodfellow 等人在“Generative Adversarial Networks”中提出了生成对抗网络。学术界和工业界都开始接受并欢迎GAN的到来。GAN的崛起不可避免。

首先，GAN最厉害的地方是它的学习性质是无监督的。GAN也不需要标记数据，这使GAN功能强大，因为数据标记的工作非常枯燥。

其次，GAN的潜在用例使它成为交谈的中心。它可以生成高质量的图像，图片增强，从文本生成图像，将图像从一个域转换为另一个域，随年龄增长改变脸部外观等等。这个名单是远远不够的。我们将在本文中介绍一些流行的GAN架构。

Figure 7: 卷积VAE：通过采样得到的新样本。

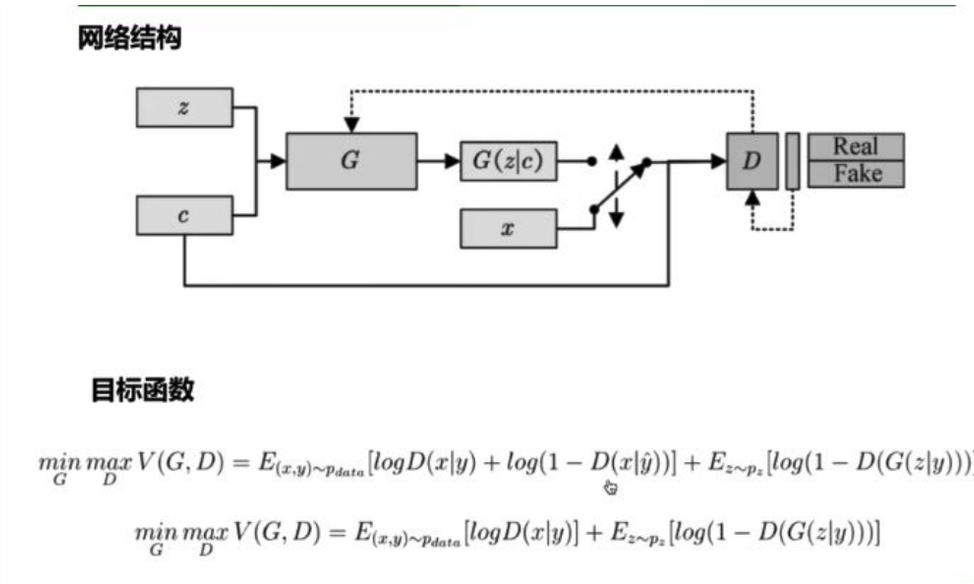

Figure 8: GAN的模型架构。

第三，围绕GAN不断的研究是如此令人着迷，以至于它吸引了其他所有行业的注意力。我们将在本文后面部分讨论重大技术突破。

2.1 诞生

生成对抗网络（GAN）具有两个网络，生成器网络和鉴别器网络。这两个网络可以是神经网络，从卷积神经网络，递归神经网络到自动编码器。在这种配置中，两个网络参与竞争游戏并试图相互超越，同时帮助他们完成自己的任务。经过数千次迭代后，如果一切顺利，生成器网络可以完美生成逼真的虚假图像，并且鉴别器网络可以很好地判断的图像是真实的还是虚假的。换句话说，生成器网络将来自潜在空间的随机噪声矢量（不是来自潜在空间的所有GAN样本）变换为真实数据集的样本。GAN的训练是一个非常直观的过程。

GAN具有大量的实际用例，如图像生成，艺术品生成，音乐生成和视频生成。此外，它还可以提高图像质量，图像风格化或着色，面部生成以及其他更多有趣的任务。 上图（8）表示了一般的GAN网络的架构。首先，从潜在空间采样D维的噪声矢量并发送到生成器网络。生成器网络将该噪声矢量转换为图像。然后将生成的图像发送到鉴别器网络以进行分类。鉴别器网络不断地从真实数据集和由发生器网络生成的图像获得图像。它的工作是区分真实和虚假的图像。所有GAN架构都遵循这样的设计。

2.2 青春期

在青春期，GAN产生了许多流行的架构，如DCGAN，StyleGAN，BigGAN，StackGAN，Pix2pix，Age-cGAN，CycleGAN等。这些结构的结果都非常令人满意。下面详细讨论这些GAN架构。

2.2.1 DCGAN

这是第一次在GAN中使用卷积神经网络并取得了非常好的结果。之前，CNN在计算机视觉方面取得了前所未有的成果。但在GAN中还没有开始应用CNNs。Alec Radford，Luke Metz，Soumith Chintala等人“Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”提出了DCGAN。这是GAN研究的一个重要里程碑，因为它提出了一个重要的架构变化来

解决训练不稳定，模式崩溃和内部协变量转换等问题。从那时起，基于DCGAN 的架构就被应用到了许多GAN架构。

2.2.2 BigGAN

这是GAN中用于图像生成的最新进展。一个谷歌的实习生和谷歌DeepMind部门的两名研究人员发表了一篇“Large Scale GAN Training for High Fidelity Natural Image Synthesis”的论文。本文是来自Heriot-Watt大学的Andrew Brock与来自DeepMind的Jeff Donahue和Karen Simonyan 合作的实习项目。

这些图像都是由BigGAN生成，正如你看到的，图像的质量足以以假乱真。这是GAN首次生成具有高保真度和低品种差距的图像。之前的最高初始得分为52.52，BigGAN的初始得分为166.3，比现有技术（SOTA）好100%。此外，他们将Frechet初始距离（FID）得分从18.65提高到9.6。这些都是非常令人印象深刻的结果。它最重要的改进是对生成器的正交正则化。

2.2.3 StyleGAN

StyleGAN是GAN研究领域的另一项重大突破。StyleGAN由Nvidia在题为“A Style-Based Generator Architecture for Generative Adversarial Network”的论文中介绍。StyleGAN在面部生成任务中创造了新记录。算法的核心是风格转移技术或风格混合。除了生成面部外，它还可以生成高质量的汽车，卧室等图像。这是GANs领域的另一项重大改进，也是深度学习研究人员的灵感来源。

2.2.4 StackGAN

StackJANs由Han Zhang，Tao Xu，Hongsheng Li 还有其他人在题为StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks的论文中提出。他们使用StackGAN来探索文本到图像的合成，得到了非常好的结果。一个StackGAN由一对网络组成，当提供文本描述时，可以生成逼真的图像。

正如上图所看到的，提供文本描述时，StackGAN生成了逼真的鸟类图像。最重要的是生成的图像正类似于所提供的文本。文本到图像合成有许多实际应用，例如从一段文本描述中生成图像，将文本形式的故事转换为漫画，创建文本描述的内部表现。

2.2.5 CycleGAN

CycleGAN有一些非常有趣的用例，例如将照片转换为绘画，将夏季拍摄的照片转换为冬季拍摄的照片，或将马的照片转换为斑马照片，或者相反。CycleGANs 由Jun-Yan Zhu，Taesung Park，Phillip Isola和Alexei A. Efros 在题为“Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”的论文中提出。CycleGAN 用于不同的图像到图像翻译。

2.2.6 Pix2pix

对于图像到图像的翻译任务，pix2pix也显示出了令人印象深刻的结果。无论是将夜间图像转换为白天的图像还是给黑白图像着色，或者将草图转换为逼真的照片等等，Pix2pix在这些例子中都表现非常出色。pix2pix网络由Phillip Isola，Jun-Yan Zhu，Tinghui Zhou和Alexei A. Efros在他们的题为“Image-to-Image Translation with Conditional Adversarial Networks”的论文中提出。

2.2.7 Age-cGAN（Age Conditional Generative Adversarial Networks）

面部老化有许多行业用例，包括跨年龄人脸识别，寻找失踪儿童，或者用于娱乐。Grigory Antipov，Moez Baccouche 和Jean-Luc Dugelay 在他们的题为“Face Aging with Conditional Generative Adversarial Networks”的论文中提出了用条件GAN进行面部老化。

该图显示了Age-cGAN是怎样从原来的年龄转换为目标年龄的。

这些都是非常流行的GAN架构。除了这些，还有数以千计的GAN架构。这取决于哪种架构适合您的需求。

2.3 GAN的损失函数与训练

生成对抗网络是一个博弈的过程，分成两部分，一个是生成器，一个是判别器。这两部分，分别由两个神经网络具体化。它和VAE的区别在于，VAE直接对数据的概率密度函数进行建模，而GAN绕过数据的PDF，通过一个判断真假的判别器去提升生成器的水平，达到生成的图片和真实图片一模一样。所以从初始化模型参数后，GAN的目标就是训练一个准确的判别器和一个高水平的生成器，最后我们把生成器拿过来使用。

对于生成器来说，首先要有一个输入，这个输入可以认为是一个随机变量，这个随机变量服从什么分布，取决于不同的问题，一般情况下，会选择标准高斯分布，即$z \sim p_z(z)$。把生成器对应的神经网络用$G(z;\theta_g)$ 来表示，这里θ_g表示生成器神经网络对应的参数。生成的数据用x来表示，即$x = G(z,\theta_g)$。判别器对应的参数为θ_d，输入为真实数据或者生成器生成的所谓的假数据，输出为对真假数据的判断，可以输出$[0,1]$之间的实数，对数据的真实性进行打分，也可以是$[0,+\infty)$的实数，这个并不唯一，根据具体的问题，选择恰当的评判标准，下面我们采用第一种输出的形式，即$[0,1]$。

我们从判别器出发，判别器的目的是要求真实数据输入进来，那么打分要尽可能的高，如果是生成器输出的假数据，那么打分要尽可能的低，所以有：

- x来自于真实数据集，那么要求$\log D(x)$ 尽可能地大；
- x来自于生成器，那么要求$\log(1 - D(G(z)))$ 尽可能地大；

所以综合在一起，得到如下的目标函数：

$$\max_D E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_z}[\log(1 - D(G(z)))] \tag{37}$$

再看生成器，我么要求生成器生成地假数据，要尽可能地骗过判别器，即生成地数据和真实数据非常逼近，甚至一模一样。所以我们有：

$$\min_G E_{z\sim p_z}[\log(1 - D(G(z)))] \tag{38}$$

把这两部分综合在一起，就得到了总的损失函数，即：

$$\min_G \max_D E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_z}[\log(1 - D(G(z)))] \tag{39}$$

理论上说，目标函数的最优解，应该是$p_g = p_{data}$。那么我们就推导一下，看一看是不是和我们的直觉一致。首先记：

$$V(D,G) = E_{x\sim p_{data}}[\log D(x)] + E_{x\sim f_g}[\log(1 - D(x))] \tag{40}$$

我们先把$G(z)$固定，求$\max_D V(D,G)$，我们有：

$$\max_D V(D,G) = \int p_{data} \log D dx + \int p_g \log(1 - D) dx \tag{41}$$

接下来对上面的式子求D的偏导数：

$$\frac{\partial}{\partial}\max_D V(D,G) = \int \frac{\partial}{\partial}\max_D [p_{data}\log D + p_g \log(1-D)]dx \tag{42}$$

$$= \int p_{data}\frac{1}{D} + p_g \frac{-1}{1-D}dx \equiv 0 \tag{43}$$

最后化简，得到：

$$D_G^\star = \frac{p_d}{p_d + p_g} \tag{44}$$

接下来，我们把求到的最优解，$D_G^\star$代入到原方程，有：

$$\min_G \max_D V(D,G) = \min_G V(D_G^\star, G) \tag{45}$$

等价于：

$$\min_G E_{x\sim p_{data}}[\log \frac{p_d}{p_d+p_g}] + E_{x\sim p_g}[\log(\frac{p_g}{p_d+p_g})] \tag{46}$$

这里我们有一个事实，即$p_{data} = p_d$，我们观察，这种期望，有点像KL散度，但是还不是，因为分母是两个概率分布的和，取值为$[0,2]$，所以我们进一步，拼凑出来一个概率分布，有：

$$\min_G E_{x\sim p_{data}}[\log \frac{p_d}{(p_d+p_g)/2}\cdot\frac{1}{2}] + E_{x\sim p_g}[\log(\frac{p_g}{(p_d+p_g)/2}\cdot\frac{1}{2})] \tag{47}$$

那么就可以写成KL散度的形式，即：

$$\min_G KL(p_d, \frac{p_d+p_g}{2}) + KL(p_g, \frac{p_d+p_g}{2}) - \log 4 \geq -\log 4 \tag{48}$$

当$p_d = \frac{p_d+p_g}{2}$时，=成立。那么我们有：

$$p_d^\star = p_d, D_G^\star = \frac{1}{2} \tag{49}$$

换句话说，最后生成器得到的数据和真实数据，对于判别器来说，已经分辨不出来了，这就和直觉保持一致了。

References