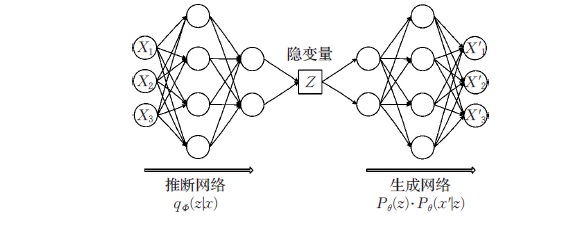
**生成网络的基本原理**

在深度学习之前已经有很多生成模型，但生成模型难以描述难以建模，遇到了很多挑战，而深度学习的出现解决了不少问题。本文介绍基于深度学习思想的生成模型——VAE和GAN，以及GAN的变种模型。

1. **VAE（**Variational autoencoder，变分自编码器）

**1.1 推断网络与生成网络**

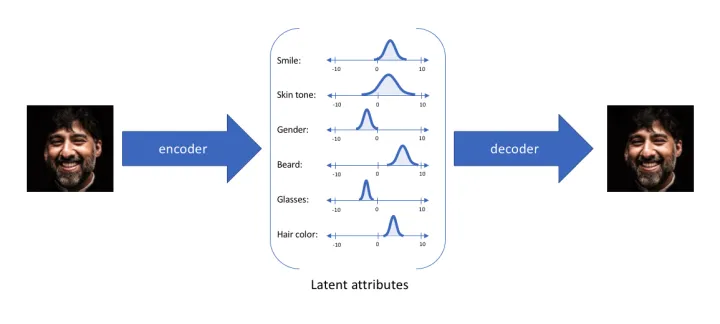
推断网络大多有下面的规律：已知变量，和多任务输出，推断网络对进行建模，它根据输入的观察变量得到出现的可能性。生成网络则是将两者的顺序反过来，它要对进行建模，输入是一些随机值，输出是的概率。



VAE利用两个神经网络建立两个概率密度分布网络：一个用于原始输入数据的变分推断，从产生，一般希望满足高斯分布，这样便于对进行描述，这就说明原来的训练函数的分布并不能满足要求，需要用满足高斯分布的变分函数去近似它；生成网络则根据，通过形成一个分布，以大概率还原。

这很像深度学习模型，从输入到输出，中间的隐藏层称为自编码层，它的值是对于的重新编码。现在是把隐藏层单独拿出来，通过人类提供的编码描述作为输入。

这里可以是一个向量，其中包括识别多种目标，以及目标与目标的关系（位置关系，形状关系等），从到是一个摘要过程，摘要可以是图像，也可以是文本。从到是生成过程，从摘要生成文档或图像，例如GPT做的那样。一个相当于一个描述，对于所生成的图像和文本给予了一种描述。的集合形成了一种新的描述世界的语言，与人类语言不同，有它的语法和语义（token和数值），也具有通用的描述能力，推断网络实现人类语言到的transformer，而生成网络正好相反。



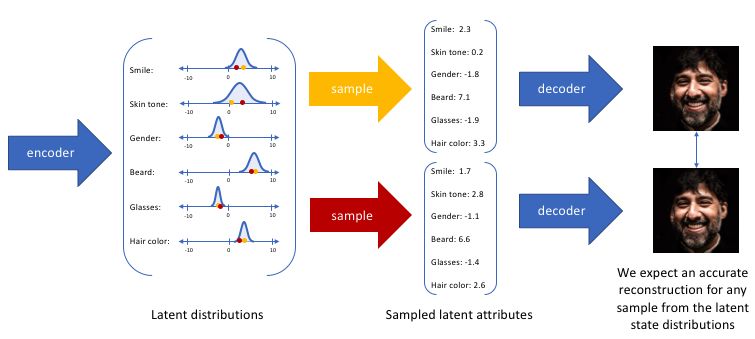


图 通过图像描述生成，这里的有六个维度，根据其中值的不同产生不同的图像

但对于生成满足的数据这样的问题来说，推断网络就会显得力不从心。假设给定，如果用推断网络生成，通过Loss函数和梯度下降，找一个输出近似为的，这样生成数据的效率可能会十分低下。而生成网络解决这个问题就十分简单，首先确定好的取值，然后根据进行计算就行了。

考虑生成网络主要原因如下：

* 这是对我们能够表示和操控高维概率分布的能力的有效检验。
* 我们可以将生成网络结合到强化学习(reinforcement learning)中，例如对于model-based RL可用生成网络来模拟可能发生的未来情况，以便RL算法进行规划(planning)
* 生成网络可以用有损失（部分样本无标记）的数据进行训练，进行半监督学习(semi-supervised learning)，降低了我们获得数据样本的难度。
* 生成网络可以处理多峰值(multi-modal)的输出。对于很多任务，一个输入可能对应多个可能的输出，一些传统的机器学习网络只能学到一种输出而无法学习多种可能的输出。
* 还有一些任务需要产生看起来真实的样本。如由低分辨率图片产生高分辨率图片，图像转换等等。

下面就来看看生成网络的建模方法。

**1.2 变分近似及其下界**

虽然生成网络和推断网络的形式不同，但两者建模的方法总体来说相近，生成网络一般也通过最大化后验概率的形式进行建模优化。也就是利用贝叶斯公式：

这个公式在复杂的模型和大规模数据面前极难求解。为了解决这个问题，这里将继续采用变分的方法用一个变分函数代替。这里选择KL散度衡量两者的相近程度。根据KL公式就有：

KL散度是两个函数和之间相近程度的度量，

(设为[0,1]均匀分布)

根据贝叶斯公式进行变换，就得到了：

将等式左右交换，就得到了下面的公式并乘以：

其中（。这样一来，由于是一个固定值，因此的最小问题，变成左边最大问题。

首先看等号左边，虽然的概率分布不容易求出，但在训练过程中当已经给定，已经是个固定值,这个界定了变分的下界，决定了我们能够以多大近似度模拟。

如果训练的目标是希望尽可能小，就相当于让等号右边的那部分尽可能变大。等号右边的第一项实际上是基于概率的对数似然期望，第二项又是一个负的KL散度，所以我们可以认为，为了找到一个好的满足高斯分布的（实际上是），使得它和尽可能相近，实现最终的优化目标，优化的目标将变为：

右边第一项的log似然的期望最大化：

右边第二项的KL散度最小化：

右边两个项目的优化难度相对变小了一些，在计算第一项时，自然满足了生成网络最大化生成的要求。下面就来看看如何基于它们做进一步的计算。

**1.3 重参数化技巧**

上面提到了是的变分函数，由于我们希望的输出满足高斯分布，其中表示网络，表示网络参数。不妨假设,是输出的期望向量，,它是确定的；向量是一个均值为0的高斯分布，它表示了随机的部分，基于这样的表示方法，条件概率中的随机性将主要来自这里。这就把看作了期望为的高斯分布，这与原始的可能并不一样，但是原始的并不知道，因此作此假设，这时去近似一个高斯分布函数。抽取是使得均值为0，减少计算难度，并不是必要的。经过变换，条件概率值完全取决于的概率。也就是说，如果的第个分量，令，那么。上面关于变分推导的公式就变成了下面的公式：

这个变分函数的建模就完成了。现在利用KL散度的近似逼近,去找一个满足是高斯分布的变分函数，找一个实际上就是修改一个得到一个新的推断网络参数。

假设近似一个高斯分布，有两个好处，一个是高斯分布比较熟悉，易于用来描述事物（某个标准模式处于中心），另一个是易于用Sigmoid函数逼近。

当每一个分量都是高斯分布，维向量的联合概率分布密度函数是：

（）分别是和的联合分布的协方差矩阵，是维度。

当我们用一个高斯函数去近似另一个高斯函数时，KL散度公式为：

**1.4 Encoder和Decoder的计算公式**

回顾一下两个优化目标，下面就来想办法求解这两个目标。首先来看看第一个优化目标，也就是让公式右边第二项最小化。被假设成一个多维且各维度独立的高斯分布，这里可以给出一个更强的假设，那就是这个高斯分布各维度的均值为0，协方差为单位矩阵（即服从各维度独立），则进一步简化成为：

前面提到了一个用深层网络实现的网络，它的输入是一批随机采样的，输出是，将每一批的计算它们的均值和方差。这样利用上面的公式，利用梯度下降（或者其他方法），使得右边最小化。实际计算过程中不需要将协方差表示成矩阵的形状，只需要一个向量来表示协方差矩阵的主对角线即可（即继续假设的输出相互之间独立），公式将被进一步简化：

，

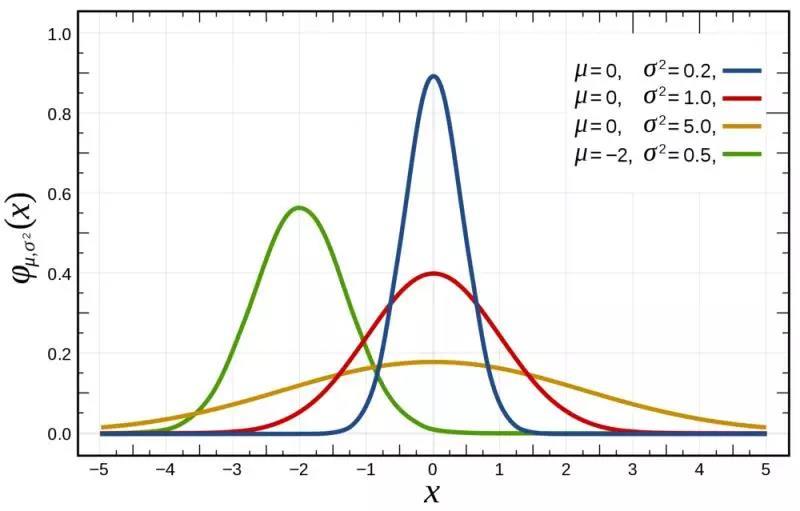
通过最小化，确定向量，这一步通过网络的参数调整实现了对于的近似度量，实际的是，于是令，就近似逼近于。该过程称为编码（Encoder），模型称为推断网络。

接下来是第二个优化目标，也就是让公式右边第一项的似然期望最大化。即

这是对于生成网络而做的。这个计算仍然复杂，假设对于的采样是独立随机的，因此各个是相等的，因此将上式写作：

其中是样本的个数。为了使期望似然得到了最大化，需要使得尽可能大，即对于每个由产生的,生成网络尽可能还原。由于前面的Encoder可以计算出了对应的（已经假设满足一个正态分布），这里就可以根据似然再建立一个深层网络，先随机输入得到一个，再输入，输出为变量，并通过Loss函数优化生成网络的参数，使得最大化。这个方法类似于蒙特卡洛算法MC，该过程被称为解码（Decoder），模型称为生成网络。

但是期望似然最大，并不意味着概率曲线最陡峭（即方差小），我们希望的输出集中于，其他输出的概率急剧下降，期望似然不能保证这一点，这时VAE的缺点。如下图，同样期望下蓝线比红线更陡。



下面就来看看VAE实现的代码，同时来看看VAE模型生成的图像是什么样子。

**1.5 实现**

本节要介绍VAE模型的一个比较不错的实现——GitHub - cdoersch/vae\_tutorial: Caffe code to accompany my Tutorial on Variational Autoencoders，这个工程还配有一个介绍VAE的文章。这个实现使用的目标数据集依然是MNIST，模型的架构如图所示。为了更好地了解模型的架构，这里将模型中的一些细节隐去，只留下核心的数据流动和Loss计算部分。

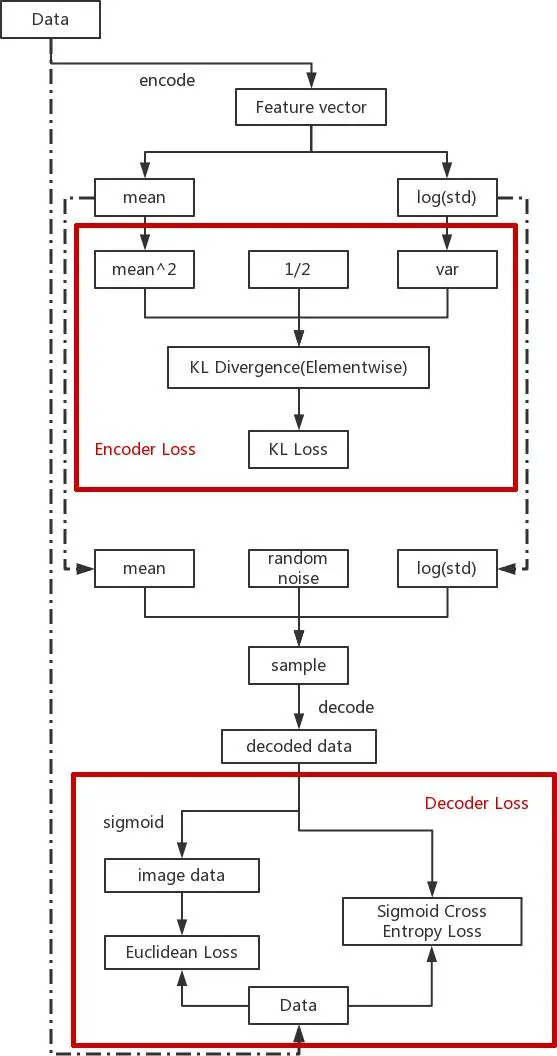


图 VAE模型结构图

图中粗框表示求解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，除了Encoder和Decoder，图中还有三个主要部分。

Encoder的Loss计算：KL散度。

的重采样生成。

Decoder的Loss计算：最大似然。

这其中最复杂的就是第一项，Encoder的Loss计算。由于Caffe在实际计算过程中只能采用向量的计算方式，没有广播计算的机制，所以前面的公式需要进行一定的变换：

在完成了前面的向量级别计算后，最后一步就是完成汇总加和的过程。这样Loss计算就顺利完成了。

上面对VAE理论和实验的介绍，经过训练后VAE的解码器在MNIST数据库上生成的字符如下图所示。

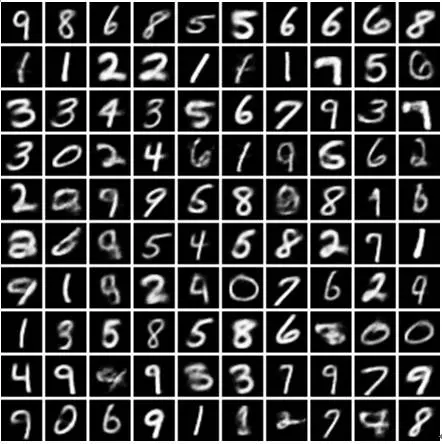


图 VAE生成的数字图

**2．GAN**

前面我们介绍了VAE，下面来看看GAN（Generative Adversarial Network），这个网络通过对抗博弈的角度展现生成网络和推断网络各自的威力的，在效果上比VAE还要好些。

**2.1 GAN的概念**

同VAE模型类似，GAN模型也包含了一对子模型。GAN的名字中包含一个对抗的概念，为了体现对抗这个概念，除了生成网络，其中还有另外一个模型帮助生成网络更好地学习观测数据的条件分布。这个模型可以称作分类网络，它的输入是数据空间内的任意一张图像，输出是一个概率值，表示这张图像属于真实数据的概率。对于生成网络来说，它的输入是一个随机变量，服从某种分布，输出是一张图像，如果它生成的图像经过网络后的概率值很高，就说明生成网络已经比较好地掌握了数据的分布模式，可以产生符合要求的样本；反之则没有达到要求，还需要继续训练。

两个网络的目标如下所示：

分类网络的目标是最大化，也就是甄别出哪些图是真实数据分布中的。

生成网络的目标是最大化，也就是让自己生成的图被分类网络判断为来自真实数据分布。

看上去两个网络目标联系并不大，下面就要增加两个网络的联系，如果生成网络生成的图像和真实的图像有区别，分类网络要给它判定比较低的概率。

基于上面的例子，两个网络的目标就可以写成一个目标函数。

上面这个公式对应的网络架构如下图所示。

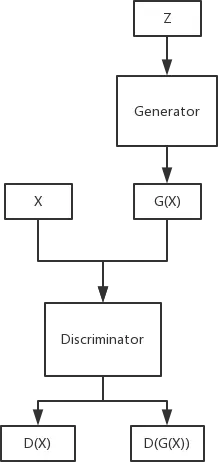


图 GAN的基本形式

对应的模型学习算法伪代码如下所示：

:

表示生成网络

表示分类网络

表示训练数据

:

:

上面的代码只是从宏观的层面介绍了模型的优化方法，其中表示了分类网络的迭代次数，一般大于等于1。从上面的公式可以看出，两个网络的目标是对立的。生成网络希望最大化自己生成图像的似然，分类网络希望最大化原始数据的似然的同时，能够最小化生成的图像的似然。既然是对立的，那么两个网络经过训练产生的能力就可能有很多种情况。它们既可能上演“魔高一尺，道高一尺”，“道高一丈，魔高十丈”的竞争戏码，在竞争中共同成长，最终产生两个强大的网络；也可能产生一个强大的网络，将另一方完全压倒。

如果分类网络太过强大，那么生成网络会产生两种情况：一种情况是发现自己完全被针对，网络参数无法优化；另外一种情况是发现分类网络的一些漏洞后，它的网络将退化，不管输入是什么样子，输出统一变成之前突破了分类网络防线的那几类结果。这种情况被称为“Mode Collapse”，有点像一个复杂强大的模型崩塌成一个简单弱小的模型，这样的模型即使优化结果很好，也不能拿去使用。

如果分类网络不够强大，它的分类不够精准，而生成网络又是按照它的推断结果生产，那么生产出的产品不会很稳定，这同样不是我们想看到的结果。

总而言之，对抗是GAN这个模型要面对的一个大问题，已经证明，这个过程会收敛到纳什均衡

**2.2 GAN的训练分析**

关于GAN训练求解的过程，我们首先来证明第一步：当生成网络固定时，分类网络的最优形式。

首先将目标函数做变换：

由于两部分求和的区域不同，会对后面的计算造成困难，我们首先将两个积分区域统一。我们将生成图像的分布与真实图像的分布做一个投射，只要能够在真实数据出现的地方保证判别正确最大化即可，于是公式就变成了：

只要让求和公式内部最大化，整个公式就可以实现最大化。这样问题就转变为最大化下面的公式：

对它进行求导取极值，可以得到：

令上面的式子为0，我们可以得到结果：

对于每一个，都满足上式，则整个

最大，这就是理论上优化结果，如果一张图像在真实分布中出现的概率大而在生成分布中出现的概率小，那么最优的分类网络会认为它是真实图像，反之则认为不是真实图像。如果生成网络已经达到了完美的状态，也就是说对每一幅图像都有：

，那么

接下来就可以利用上面的结果，计算当生成网络达到完美状态时，损失函数的值。我们将（纳什均衡）的结果代入，可以得到：

也就是说生成网络损失函数的理论最小值为-2log2。那么，一般情况下它的损失函数是什么样子呢？我们假设在某一时刻判别式经过优化已经达到最优，所以

我们将这个公式代入之前的公式，可以得到：

后面的两个KL散度的计算公式可以转化为Jenson-Shannon散度，也就是：

这其实是生成网络真正的优化目标函数。在介绍VAE时， JS散度又是什么？它又有什么特性和优势？从最直观的角度，读者可以发现一个KL散度不具备的性质——JS散度是对称的：

对称又能带来什么好处呢？它能让散度度量更加符合实际，因为用一个函数逼近另一个函数，总是希望另一个函数也近似这个函数。接下来将用一段代码展示这其中的道理。首先给出两个离散随机变量的KL散度和JS散度的计算方法：

为两个list，里面存着对应的取值的概率，整个list相加为1

:

下面将用3组实验看看两个散度的计算结果。首先选定一个简单的离散分布，然后求出它的KL散度和JS散度。在此基础上，把两个分布分别做一定的调整。首先是基础的分布：

#以下为运行结果显示

接下来把公式中第二个分布做修改，假设这个分布中有某个值的取值非常小，就有可能增加两个分布的散度值，它的代码如下所示：

#以下为运行结果显示

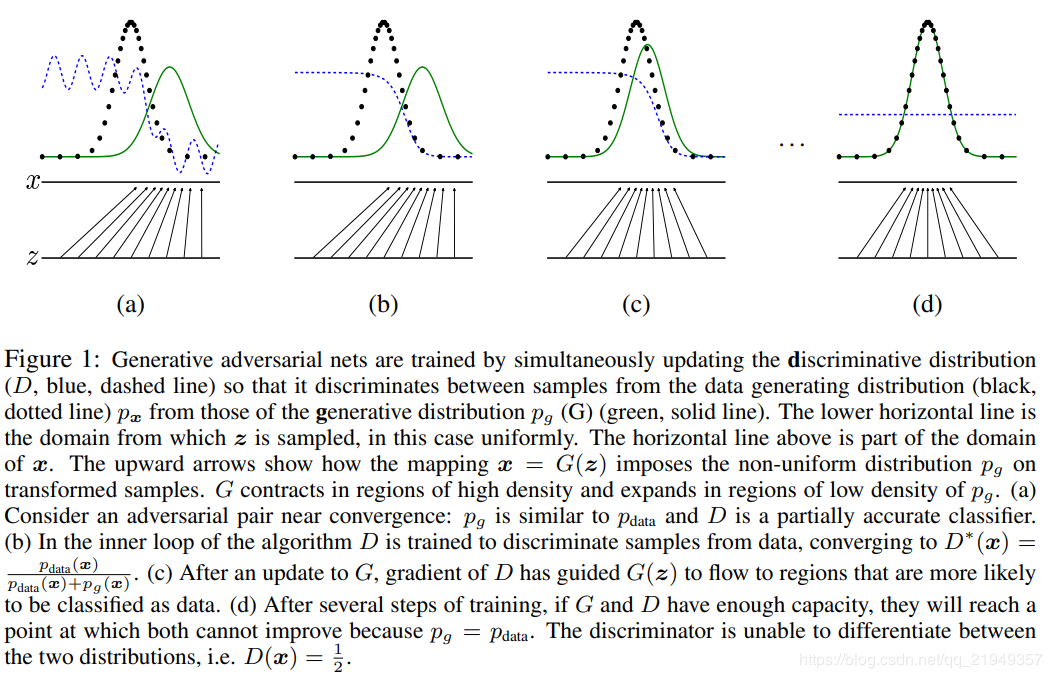
可以看出KL散度的波动比较大，而JS的波动相对小。

最后修改前面的分布，代码如下所示：

这回得到的结果是这样的：

如果将第二个实验和第三个实验作对比，就可以发现KL散度在衡量两个分布的差异时具有很大的不对称性。如果后面的分布在某一个值上缺失，就会得到很大的散度值；但是如果前面的分布在某一个值上缺失，最终的KL散度并没有太大的波动。这个例子可以很清楚地看出KL不对称性带来的一些小问题。而JS具有对称性，所以第二个实验和第三个实验的JS散度实际上是距离相等的分布组。

从这个小例子我们可以看出，有时KL散度下降的程度和两个分布靠近的程度不成比例，而JS散度靠近的程度更令人满意，这也是GAN模型的一大优势。



**2．3 GAN实战**

看完了前面关于GAN的理论分析，下面我们开始实战。在实战之前目标函数还要做一点改动。从前面的公式中可以看出这个模型和VAE一样都是有嵌套关系的模型，对于分类网络，采用下面的Loss函数：

对于生成网络，采用下面的Loss函数：

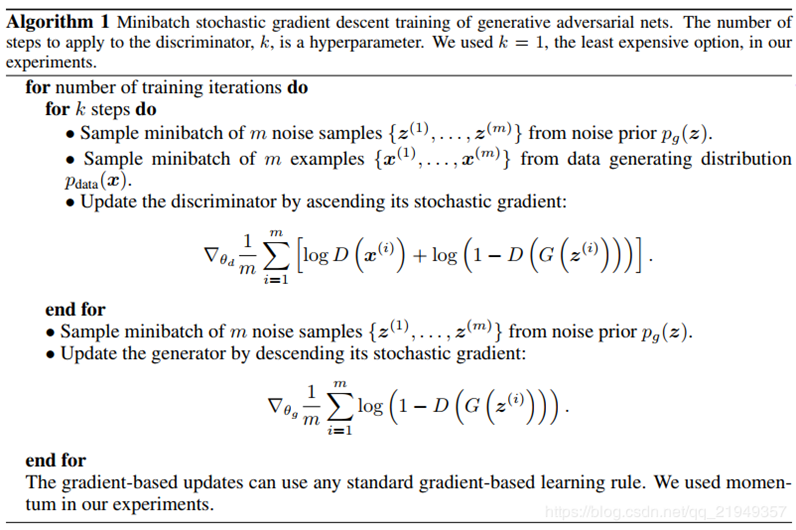


图 GAN的基本算法

一般采取次训练分类网络，再训练一次生成网络的方法。

下面来看一个具体的基于深层模型的实现——DC-GAN。全称是Deep Convolution GAN。也就是用深度卷积网络进行对抗生成网络的建模。在此之前，也有一些基于卷积神经网络的GAN实现，但是相对来说，DC-GAN的最终表现与同期的模型相比更优秀，在介绍它的论文中，作者也详细介绍了模型的一些改进细节。

将Pooling层替换成带有stride的卷积层

使用Batch Normalization

放弃使用全连接层

将卷积层的非线性部分换成ReLU或者Leaky ReLU

下面将使用DC-GAN的模型进行实验，这个实验使用的数据集还是MNIST。由于Caffe并不是十分适合构建GAN这样的模型，因此这里使用另外一个十分流行且简单易懂的框架——Keras来展示DC-GAN的一些细节。代码来自https://github.com/jacobgil/keras-dcgan。由于Keras的代码十分直观，这里就直接给出源码。首先是生成模型：

这里需要说明的一点是，这个实现和论文中的描述有些不同，不过对于MNIST这样的小数据集，这样的模型差异不影响效果。

分类网络的结构如下所示，仔细地读一遍就可以理解，这里不再赘述。

完成训练后，生成网络生成的手写数字如下图所示。

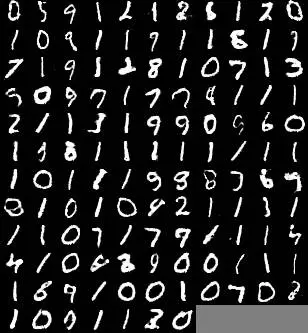
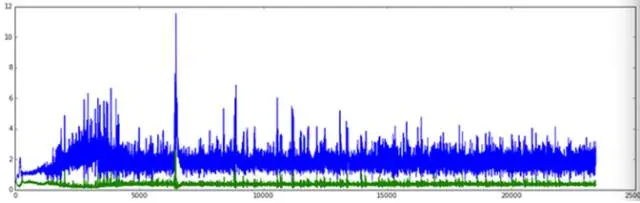


图 GAN生成的图像

除了个别数字外，大多数数字生成得和真实数据很像。将前面的图与此图比较，我们可以发现，GAN模型生成的数字相对而言更为“清晰”，而VAE模型的数字略显模糊，这和两个模型的目标函数有很大的关系。另外，两个模型在训练过程中的Loss曲线如下图所示。



d

图 GAN中生成网络和推断网络的损失函数

其中上面的曲线表示生成模型的Loss，下面的曲线是分类网络的Loss，虽然这两个Loss的绝对数值看上去不能说明什么问题，但是相信读者还是可以看出两个网络的Loss存在着强相关的关系，这也算是对抗过程中的此消彼长。

最终生成的数据还算令人满意，我们还很好奇，在网络优化过程中生成网络生成的图像都是什么样的呢？接下来就来观察生成图像的演变过程。在优化开始时，随机生成的图像如下图所示。

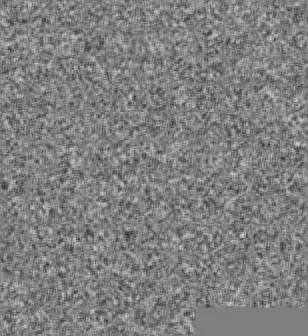


图 GAN生成网络的初始图像

其实就是噪声图片，一点都不像数字。经过400轮的迭代，生成网络可以生成的图像如下图所示。

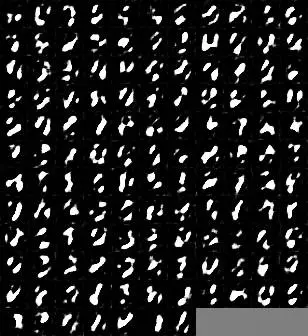


图 GAN生成网络400轮迭代训练后的图像

可以看出数字的大体结构已经形成，但是能够表征数字细节的特征还没有出现。

经过10个Epoch后，生成的作品如下图所示。



图 GAN生成网络经过10个Epoch迭代训练后的图像

这时有些数字已经成形，但是还有一些数字仍然存在欠缺。

20轮Epoch后的结果如下图所示。

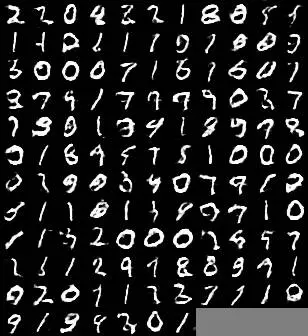


图 GAN生成网络经过20个Epoch迭代训练后的图像

这时的数字已经具有很强的辨识度，但与此同时，我们发现生成的数字中有大量的“1”。

当完成了所有的训练，取出生成网络在最后一轮生成的图像，如下图所示。

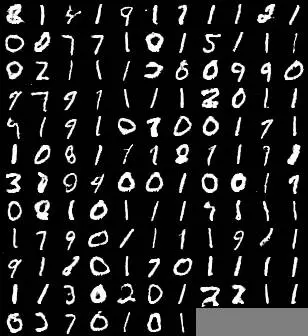


图 GAN生成网络最终生成的图像

可以看出这里面的数字质量更高一些，但是里面的“1”更多了。

从网络的训练过程中可以看出，一开始生成的数字质量都很差，但生成数字的多样性比较好，后来的数字质量比较高但数字的多样性逐渐变差，网络的特性在不断发生变化。这个现象和两个网络的对抗有关系，也和增强学习中的“探索—利用”困境很类似。

站在生成网络的角度思考，一开始生成网络会尽可能地生成各种各样形状的数字，而分类网络会识别出一些形状较差的图像，而放过一些形状较好的图像，随着学习的进程不断推进，分类网络的能力也在不断地加强，生成模型慢慢发现有一些固定的图像模式比较容易通过，而其他的图像模式不那么容易通过，于是它就会尽可能地增大这些正确模式出现的概率，让自己的Loss变小。这样，一个从探索为主的模型变成了一个以利用为主的模型，因此它的数据分布已经不像刚开始那么均匀了。

如果这个模型继续训练下去，生成网络有可能进一步地利用这个模式，这和机器学习中的过拟合也有很相近的地方。

**3.Info-GAN**

本节将要介绍GAN模型的一个变种——InfoGAN，它要解决隐变量可解释性的问题。前面提到GAN的隐变量服从某种分布，但是这个分布背后的含义却不得而知。虽然经过训练的GAN可以生成新的图像，但是它却无法解决一个问题——生成具有某种特征的图像。例如，对于MNIST的数据，生成某个具体数字的图像，生成笔画较粗、方向倾斜的图像等，这时就会发现经典的GAN已经无法解决这样的问题，想要解决就需要想点别的办法。

首先想到的方法就是生成网络建模的方法：挑出几个隐变量，强制指定它们用来表示这些特性的属性，例如数字名称和方向。这样看上去似乎没有解决问题，但这种方法需要提前知道可以建模的隐变量内容，还要为这些隐变量设置好独立的分布假设，实际上有些麻烦又不够灵活。本节的主角——InfoGAN，将从信息论角度，尝试解决GAN隐变量可解释性问题。

**3.1 互信息**

介绍算法前要简单回顾机器学习中的信息论基本知识。这里要介绍的概念叫做互信息，它衡量了随机变量之间的关联关系。假设随机事件的结果已经知道，现在要猜测某个事件的结果，那么知道的取值对猜测有多大帮助？这就是互信息要表达的东西。

我们以掷骰子为例，如果我们知道手中的骰子是不是“韦小宝特制”骰子这件事，那么它会对我们猜测最终投掷的点数有帮助吗？当然有帮助，因为一旦确定这个骰子是“韦小宝特制”，那么骰子点数是几这个信息就变得没有“惊喜”了。同理，“美国第45届总统是谁”这个消息对我们手中骰子投掷出的点数这个事情就没那么多帮助了，所以这两件事情的互信息就低，甚至可以说这两个事件是相互独立的。

了解了上面比较直观的例子，下面就可以给出连续随机变量X,Y互信息的计算公式：

上面的公式可以做如下变换：

就可以发现互信息的进一步解释：它可以变为熵和条件熵的差。同样地，这个公式还可以转变为：

下面随意给出一对随机变量和它们的概率分布，并用上面的代码分析这对变量的互信息：

很明显，下面一组数据的相关性更强，知道其中一个随机变量的取值，就会非常容易猜出同一时刻另外一个随机变量的采样值。如果我们进一步观察第二组数据，会发现任意一组数据的熵都是0.796，也就是说当知道其中一个随机变量的值后，它们的条件熵就变成了0，另一个随机变量变得完全“惊喜”了。虽然条件熵为0这个信息并没有展现在互信息的数值中，但互信息实际上就是在衡量一个相对的信息差距，并不像熵那样衡量信息绝对量。

互信息在这个问题中有什么用？如果说隐变量的确定对确定生成图像的样子有帮助，那么隐变量和最终的图像之间的互信息就应该很大：当知道了隐变量这个信息，图像的信息对变得更确定了。所以InfoGAN这个算法就是要通过约束互信息使隐变量“更有价值”。

**3.2 InfoGAN模型**

那么，InfoGAN模型的具体形式是什么样的呢？如果把互信息定义为损失函数的一部分，这部分损失函数就是InfoGAN中基于经典GAN修改的部分。在前面的两个模型VAE和GAN中，和由生成的之间应该是能够互相提供信息的。但是在前两个网络中，这部分信息未被考虑。Info-GAN在优化参数时考虑了这一部分的信息。前面已经推导出了互信息的公式，那么在具体计算时要使用哪个公式计算呢？

最终的选择是后者，因为图像的分布太难确定，求解它的熵肯定相当困难，所以前者的第一项非常难计算。当然，即使选择了第二项，这个公式也不是很好优化，因为其中还有一个后验项需要求解，这也是个大麻烦，不过这里可以使用本书多次提到的方法——Variational Inference求解这个后验项。

在介绍VAE时我们曾经运用过重参数化技巧这个方法，这里将再次采用类似的方法。在VAE中，公式是，在Encoder的过程中，输入部分被分解成确定部分和不确定部分，然后利用一个高维非线性模型拟合输入到输出的映射。这里要求输出，和VAE正好相反，需要的是这样的一个公式：

其中表示与图像有相关关系的隐变量，表示与图像无关的隐变量。于是互信息计算公式就变成了：

一般假设的期望为0，于是公式可以做进一步简化：

接下来将期望用蒙特卡罗方法代替，训练时可以通过计算大量样本求平均来代替期望值，于是公式又变成了：

这个方程变简单了很多。当然，我们也看出上面的公式中我们有一个，这个函数可以理解为一个Encoder，这部分模型在经典GAN中并不存在，但是在实际建模过程中，由于Encoder和分类网络的输入相同，且模型目标比较相近，因此二者部分网络结构可以共享。代码的链接在https://github.com/openai/InfoGAN，代码使用的框架为TensorFlow，感兴趣的读者可以自行阅读。模型实现的结构如图10-13所示。

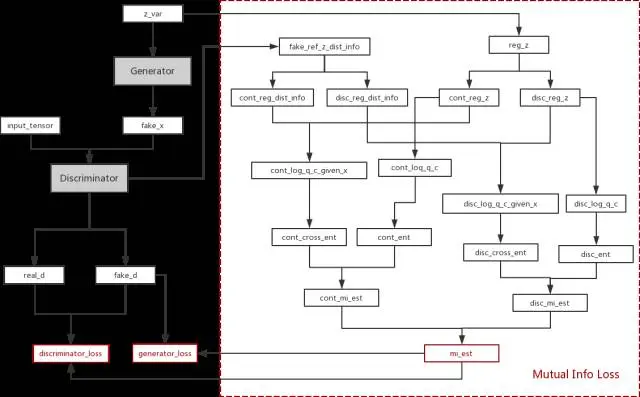


图 InfoGAN模型结构图

虚线部分表示的就是计算互信息的目标函数，这部分内容看似比较复杂，实则不然。由于InfoGAN模型中定义了两种类型的随机变量—服从一般化的分布，用于表示数字内容的离散类型变量，和服从均匀分布用于表示其他连续特征的连续型变量，而两种类型的变量在计算熵的方法不同，因此上面的计算图对它们进行分情况处理。

互信息计算起始于如下两个变量。

表示了模型开始随机生成的隐变量。

：表示了经过Encoder计算后的隐变量分布信息。

接下来，根据连续型和离散型的分类，两个变量分成了以下四个变量。

的连续变量部分

：的连续变量部分

的离散变量部分

的连续变量部分

接下来，四个变量两两组队完成了后验公式的计算：

连续变量的后验

离散变量的后验

同时，输入的隐变量也各自完成先验的计算：

连续变量的先验

离散变量的后验

由于上面的运算全部是元素级的计算，还要把向量求出的内容汇总，得到和。

连续变量的交叉熵

连续变量的熵

离散变量的交叉熵

离散变量的熵

接下来，根据互信息公式两两相减，得到各自的互信息损失。

连续变量的互信息

离散变量的互信息

最后将两者相加就得到了最终的互信息损失。

模型在训练前定义了12个和图像有强烈互信息的随机变量，其中10个变量表示显示的数字，它们组成一个Categorical的离散随机向量；另外2个是服从范围为[-1,1]的连续随机变量。训练完成后，调整离散随机变量输入并生成图像，得到如图10-14所示的数字图像。

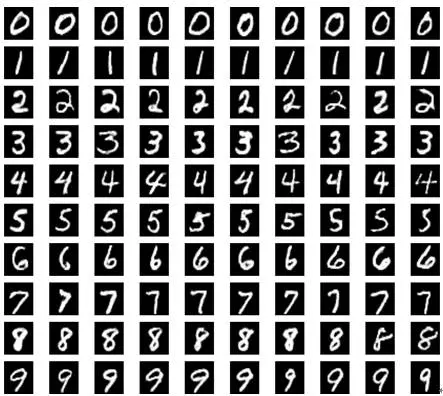


图 10个离散随机变量对生成数字的影响

可以看出模型很好地识别了这些数字。调整另外两个连续随机变量，可以生成如图所示的数字图像。

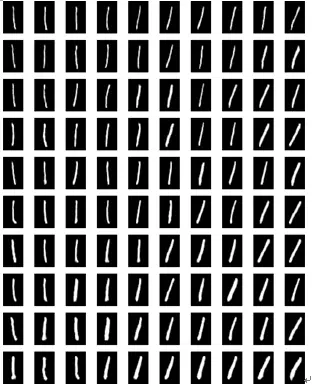


图 2个连续随机变量对生成数字的影响

可以看出，这两个连续随机变量学到了数字粗细和倾斜的特征，而且这是在完全没有暗示的情况下完成的。可见InfoGAN模型的能力。

到此InfoGAN的介绍就结束了。从这个模型可以看出，在经典GAN模型基础上添加更多的内容会产生更多意想不到的效果。