故事线：

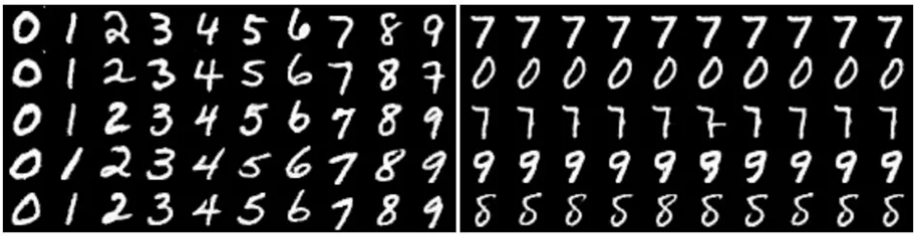
背景——原理概括——心路历程——总结反思——效果汇总——项目构成及运行指令——引用文献

infoGAN & ACGAN

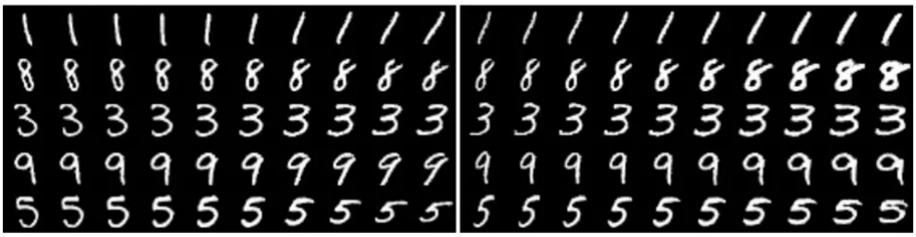
@$author$ 崔轩宁($CeoxNim$)

背景

标准的GAN网络中，生成数据是不可控的，无法通过控制噪声z的某个维度来控制生成数据的语义特征。$infoGAN$的提出便是希望通过$latent\ code$的设置，使网络可以学到能解释的特征表示，并且这种特征是未标签化的，即无需人为设置。



(a) Varying c_1 on InfoGAN (Digit type)　(b) Varying c_1 on regular GAN (No clear meaning)

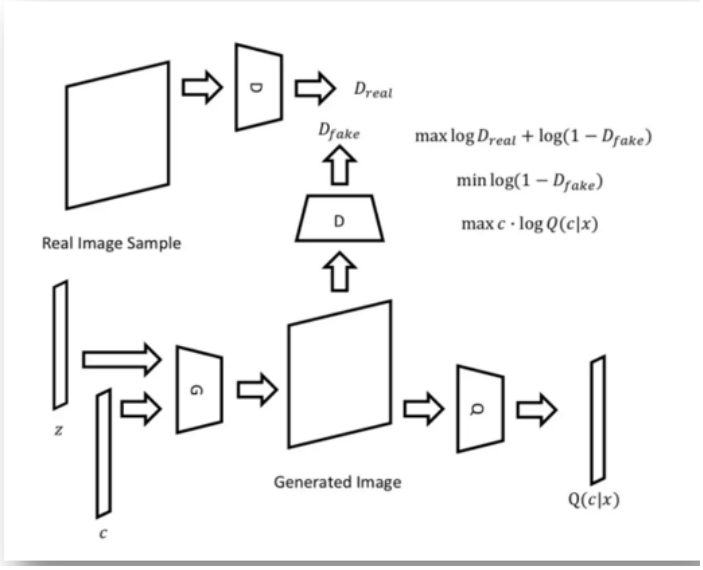


(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation)　(d) Varying c_3 from -2 to 2 on InfoGAN (Width)

PS: 出于训练的可操性，我们在$MNIST$数据集上进行训练及相关的调试工作，但当然，算法的核心思想是通用的，将其迁移到其它数据集在算力足够的情况下不是难事。

原理概括

将原始的噪声输入分为两部分，一部分仍为原来的噪声，另一部分为由若干个$latent\ variables$拼接而成的$latent\ code$，记作c，这部分用来标识生成图片的不同的特征维度。如果我们使得生成图片与c之间具有很强的相关性，我们就可以近似理解为生成的图片可以根据c的调整而调整，即学到了相应特征。图解如下：



我们可以看到，如果不考虑左下角的潜在特征向量c及右下角的神经网络Q，即为原始的GAN模型。神经网络Q起到特征提取的作用，通过Q的处理得到一个和c同样规格的向量，如果处理后得到的向量与c有很强的相关性，则可以说明所生成的图片很好地体现了$latent\ code$所代表的特征。

接下来我们通过公式来深入理解一下：

原始的GAN的目标函数：

$$min_G max_D V(D,G) = E_{x\sim P_{data}}[logD(x)] + E_{z\sim noise}[log(1-D(G(z)))]$$

当我们引入$latent\ code$后：

$$min_G max_D V(D,G) = E_{x\sim P_{data}}[logD(x)] + E_{z\sim noise}[log(1-D(G(z,c)))]$$

我们希望所生成图片与$latent\ code$的互信息尽可能大（这里用互信息度量二者之间的相关性）：

$$min_G max_D V_1(D,G) = V(D,G) - \lambda I(c;G(z,c))$$

论文中提到互信息直接求并不方便，并在别处证明了其一下界：

$$I(c;G(z,c)) \geq E_{x\sim G(z,c)}[E_{c'\sim P(c|x)}[logP(c'|x)]] + H(c) \triangleq L_I(G,Q)$$

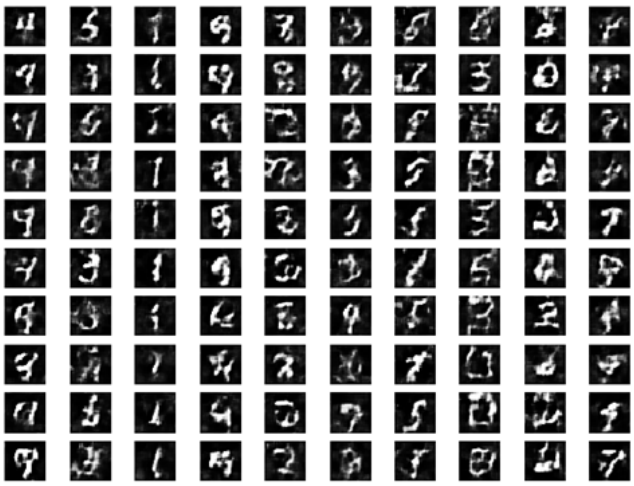
故最终目标函数即为:

$$min_G max_D V_1(D,G) = V(D,G) - \lambda L_I(G,Q)$$

而我们发现$L_I(G,Q)$实际上是期望的求取，若其为离散，我们易直接求得；若其为连续，我们可以用蒙特卡洛等随机算法对其进行离散近似，从而得到其期望值。

心路历程

在自以为理解上述原理流程后进行代码编写，效果如下：

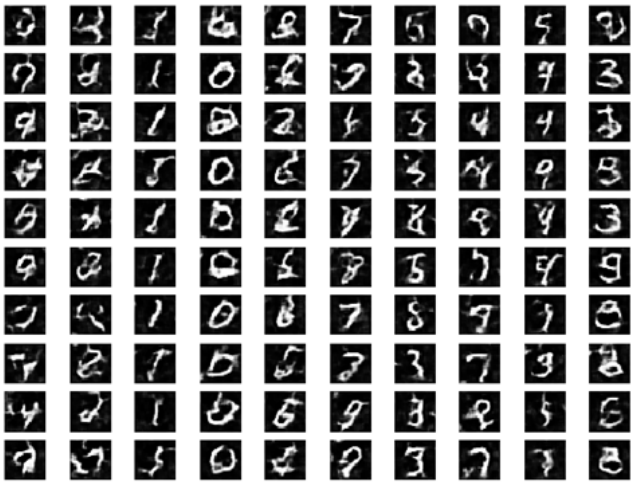


可以看到主要存在两个问题：

①效果太差，可辨识度较低，并没有达到理想中的“以假乱真”效果
②没有按照预期的顺序生成，即尽管每一列生成了形状近似的图片，但并未按照预期$0-9$顺序生成

针对问题一

首先对其进行调参，主要包括更改神经网络架构设置及其中参数设置，以及目标函数中λ的调整，效果如下：



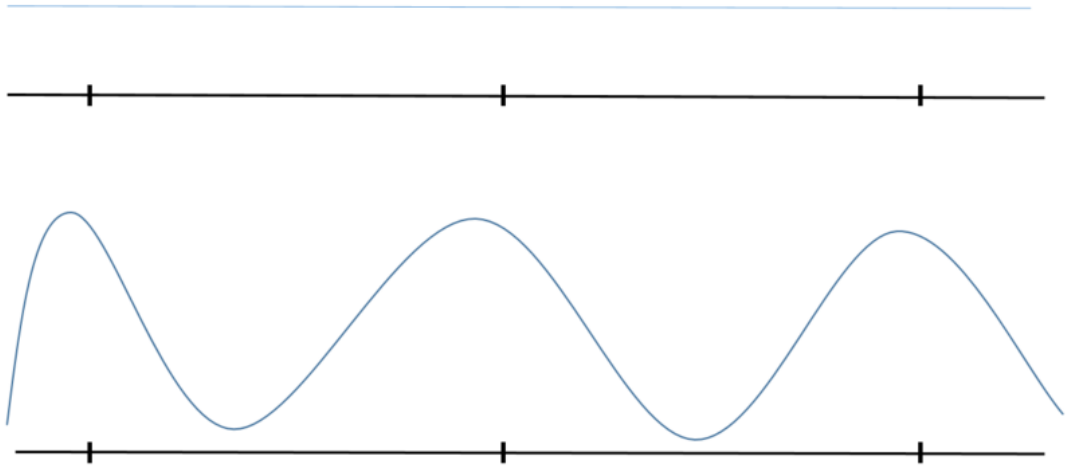
可以看到，相比于之前略有好转，但是仍未达到理想中的预期。私以为是因为自己对神经网络的架构理解得并不透彻，导致调参效果不尽人意，考虑到每次调整后都要重新训练模型，耗时太长，且确实不懂该如何调节性价比更优，便参考了网上的代码，对神经网络层面的架构作出修改，以及对损失函数做出了一定调整（这是因为考虑到数字识别是一个离散的变量，如果采用$one-hot$编码，则其非0即1，从而直接$softmax$后利用交叉熵来优化要更好一些），效果如下：

PS：这里更换了图片输出方式，觉得一张图上好看些而且方便比较233

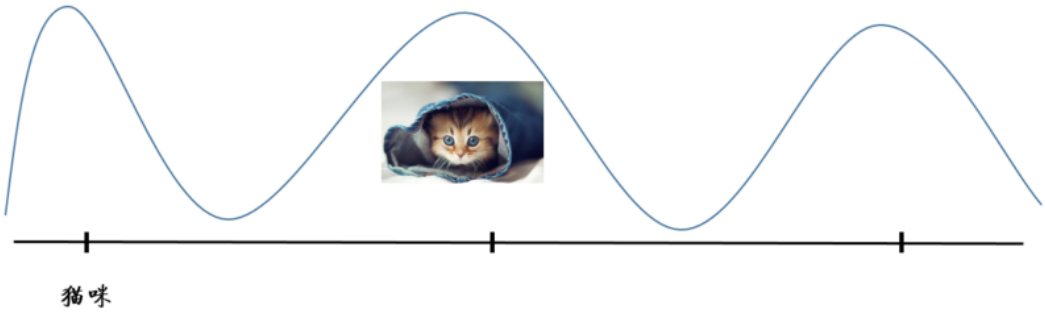
到这里，生成图片的效果已经基本上可以以假乱真了，问题一得到基本解决

针对问题二：

考虑图片不能正确按顺序生成的原因，相同的特征被识别出来并被聚集到一起，这一点没有问题

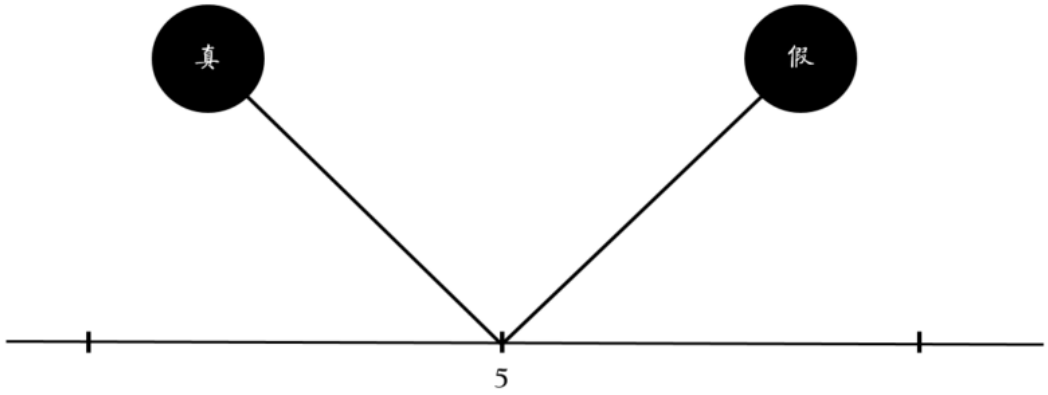


问题在于机器并不知道所聚集的特征在人类的语境下被称作什么，举例言之如下

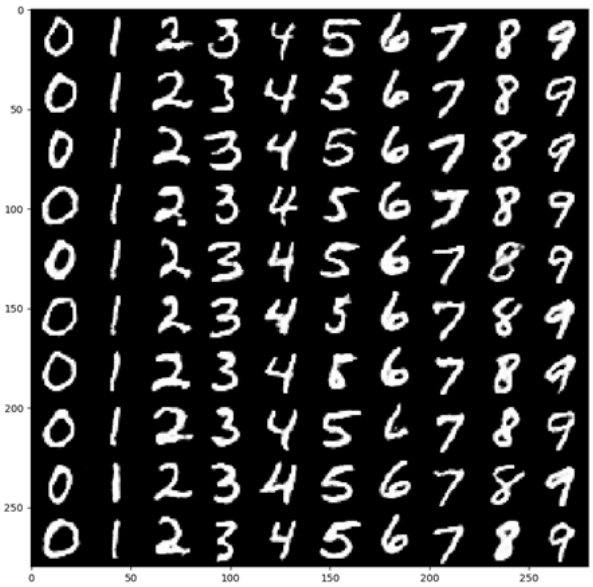


猫咪

于是乎问题的解决就需要让网络在猫咪图片和猫咪标签之间建立联系，该如何建立呢？我们没有办法直接告诉网络这个东西叫做猫咪，因为文字本就是一类抽象的符号，他需要与实体建立对应联系才可。因此为了建立这种联系，我们可以先将正确的图片与标签对应，这就为其明确了每个标签的含义，在后续的学习过程中也就会按照我们预期的顺序进行



采用如上原理，我们容易将源代码修改得到如下效果



可以看到按照预期$0-9$的顺序生成了图片，问题二得到基本解决。

总结反思

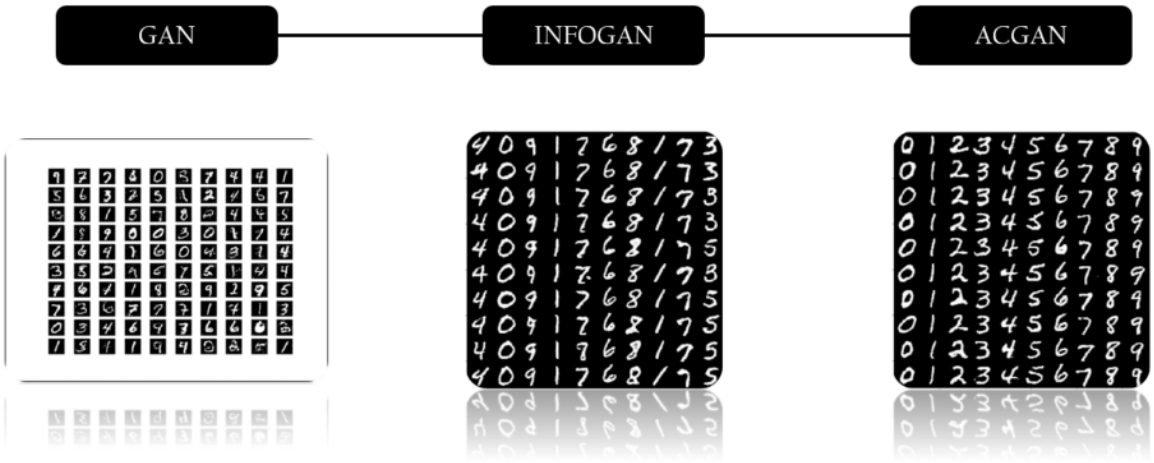
尽管问题二得到了解决，但其实本质已经改变，因为在引入标签后，学习过程就变成了有监督学习，这已经超离了$infogan$的范畴了，上网查询后得知其为另一变体，对应有监督学习的$ACGAN$

所以总的来看，二者都可以学习特征，生成指定类型的图片，只不过一个是无监督学习，可以不预先设定标签，自动提取目标最显著的特征，但缺点是训练过程比较艰难，效果略差；另一个是有监督学习，需要预先设定标签，可以得到更快更好的学习效果，但缺点就是没有办法自动提取其显著特征，且对于连续变化的变量只能近似离散处理（个人理解，如有错误，敬请斧正）

至于为什么$MNIST$训练集中学到的$latent\ variables$是倾斜度和数字粗细，直观理解是其相较于其它特征更加显著，那么为什么不能是宽度甚至山下颠倒呢？这是因为在具体实现中Q与D前期共用同一个网络，这也就保证了最后进行特征提取的前提便是生成图片具有足以以假乱真手写数字的能力。在其确为一个数字的基础上，上下颠倒就说不过去了，过宽和过窄的字也往往偏离真正的手写数字，从而相比较而言，倾斜度与粗细特征更为显著。

但这里要注意，这里的倾斜度和粗细不是固定的，而是当前数据集下的显著特征，倘若换一个数据集，学到的可能就是其他特征了，如光线强弱、脸的转角以及椅子类型等。

效果汇总



PS：这里为了报告的全面性简单写了一个原始的GAN做比，但并未仔细调参，所有效果只能说是差强人意。

项目构成及运行指令

项目构成如下：

名称	修改日期	类型	大小
img	2020/6/12 星期五 1...	文件夹	
MNIST_data	2020/5/31 星期日 2...	文件夹	
Output	2020/5/31 星期日 2...	文件夹	
infoGAN & ACGAN.md	2020/6/12 星期五 1...	Markdown File	8 KB
Acgan.py	2020/6/12 星期五 1...	PY 文件	7 KB
Gan.py	2020/6/12 星期五 1...	PY 文件	6 KB
Infogan.py	2020/6/12 星期五 1...	PY 文件	9 KB

$MNIST_data$为下载到本地的$MNIST$数据包，$Output$为之前训练好的结果，img为存储该报告所需图片

$InfoGAN\ \&\ ACGAN.md$ 即该文档为文字性报告。

运行指令：

在当前界面开启cmd，运行对应python文件即可，如：

$$python\ \ Infogan.py$$

运行效果：

命令行中会有相应的进度显示；

文件夹当前目录下会生成对应名称的文件夹，用以存储相应图片。

引用文献及repo


```
1.InfoGAN: Interpretable Representation Learning by Information Maximizing
Generative Adversarial Nets
2.https://github.com/eriklindernoren/Keras-GAN/blob/master/infogan/infogan.py
3.https://zhuanlan.zhihu.com/p/58261928
4.https://www.jiqizhixin.com/articles/2018-10-29-21
5.https://blog.csdn.net/wspba/article/details/54808833
6.https://zhuanlan.zhihu.com/p/57839730
(应该还有查一些参考资料但以上应该是主要的了~！)
```