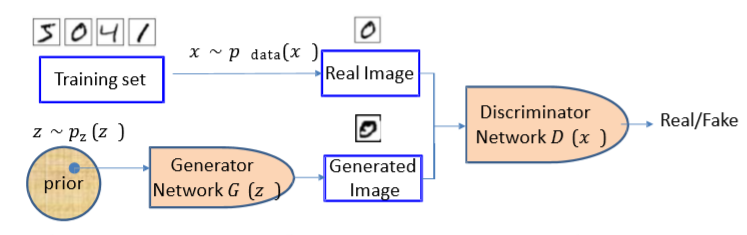
**实验6：Mnist数据集生成对抗网路**

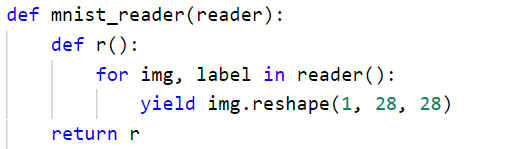
张靖祥 2017304010413 计算机172

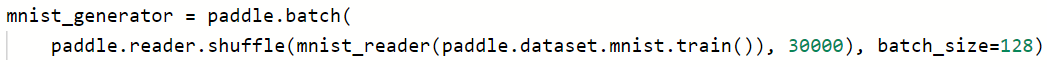
# 1. 实验代码分析

本实验仍然使用的是百度的paddlepaddle平台，因为此平台可以支持动态网络的训练，不同于tensorflow的是可以动态的定义网络的结构，以进行更为复杂的网络模型的训练。对抗性网络的模型如下。



## 1.1 真实图片的输入：

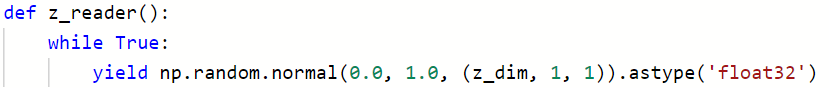




采用了和paddle自带的dataset中一样的读取数据的方法，都是使用了yield进行产生数据

## 1.2 噪声的产生



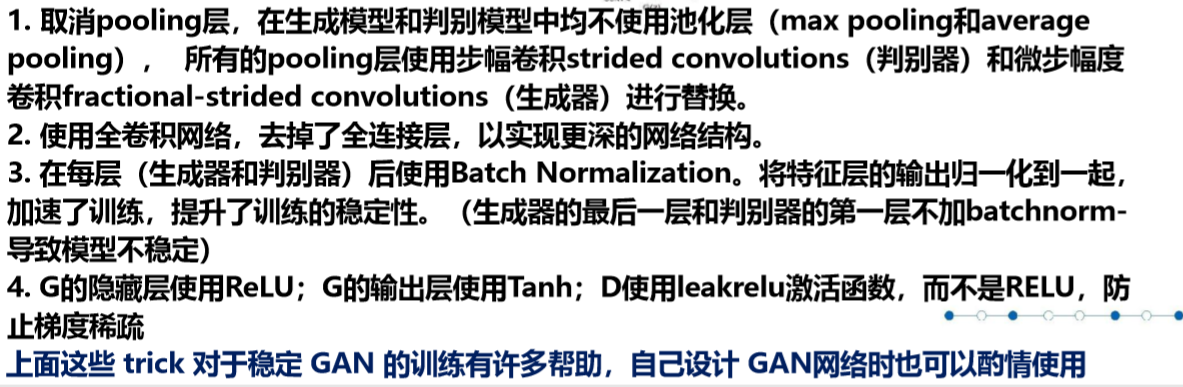




可见，产生的噪声数据的尺寸为128\*100 （128为batch size）

## 1.3 Generator Network（G网络）

PPT45页的内容为



可以总结为4点：

不用池化层降采样

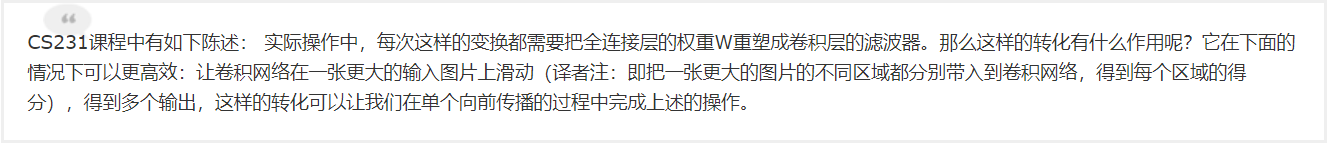
去掉全连接层，改为全卷积

每层之间都用Batch Norm

使用不同的激活函数

**全卷积VS.全连接**

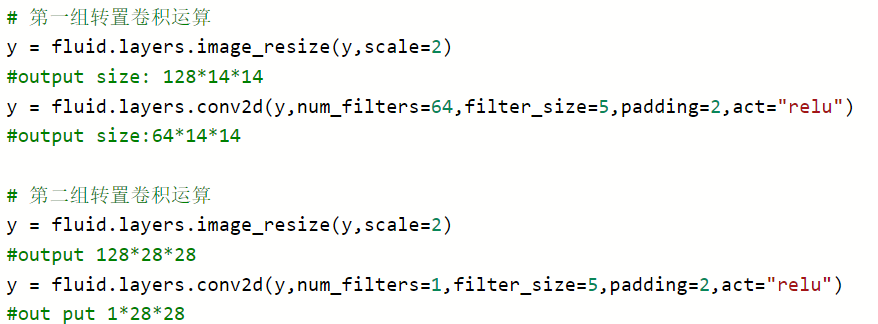
参考：<https://www.cnblogs.com/liuzhan709/p/9356960.html>



**网络的配置与各层之间数据的尺寸**

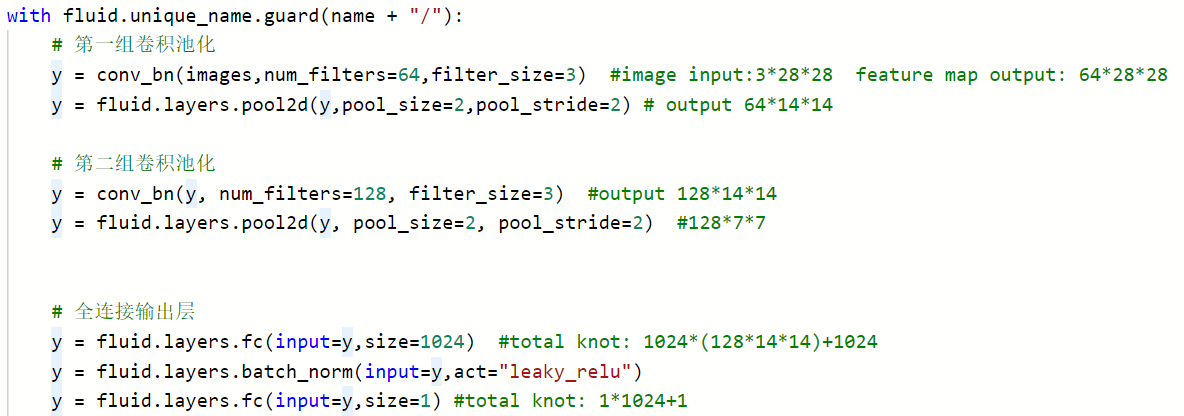


由于是数据的刚开始，因此使用了全连接层



这里精妙的使用了反卷积，扩采样，将图片的尺寸从fc层输出的128\*7\*7转化为了128\*28\*28，进而又转化为了1\*28\*28的图像。

## 1.4 Discriminator（G网络）



（图片仅为主要部分的标注）

简单的判别网络，两个卷积层，两个池化层，两个全连接层。

## 1.5 训练过程

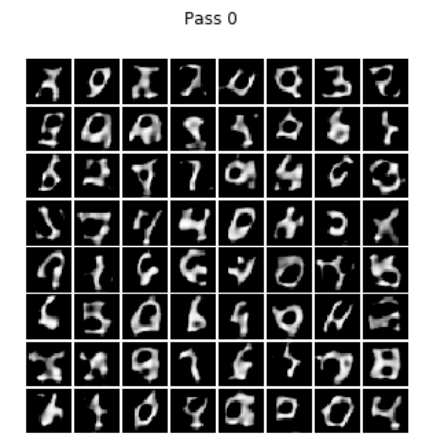
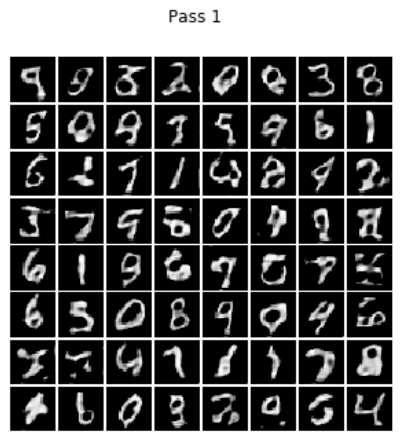
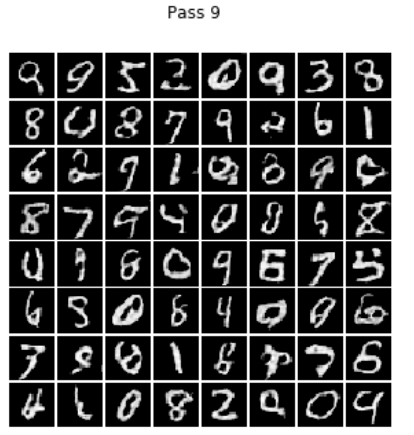
训练的顺序依次为：训练D识别假图片、训练D识别真图片、训练G生成图片



# 2. 训练结果

## 2.1 直接训练

第一个周期时，生成的图片还难以辨认；第二个周期后，生成的图片已经初见雏形了；第十个周期后，已经和手写数字无异了。

## 2.2修改网络后的各种bug

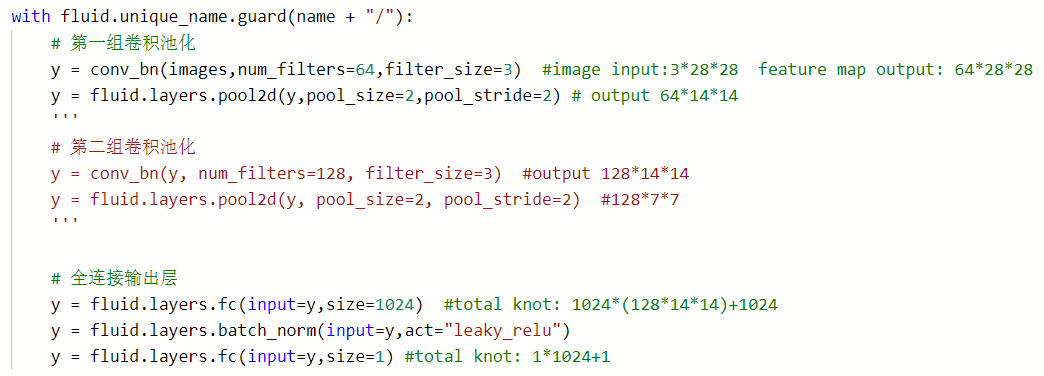
本实验中，可修改的部分无非就是生成器网络模型

### 2.2.1降低网络的复杂度遇到的bug

G网络配置成如下结构，删除第一个全连接层

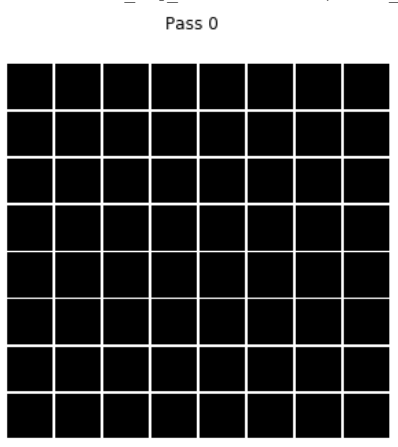


D网络去掉第二个卷积池化层

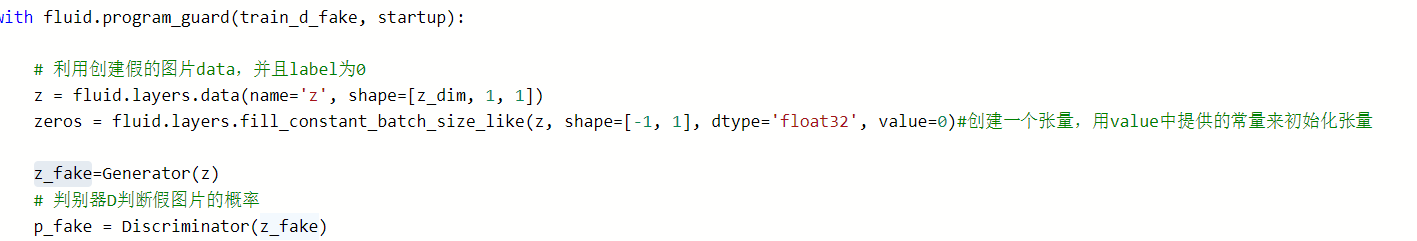


注意：1.每次运行都要重启

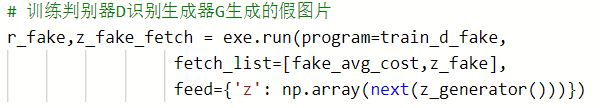
出现了如下问题



测试如下：



我在train fake这里添加了一个z\_fake变量，记录生成器生成的图片



然后我将其取出，查看其值



**全零矩阵**

于是看到PPT上



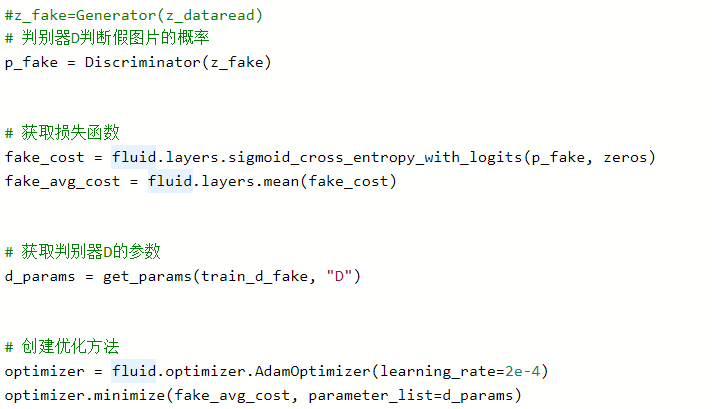
我怀疑全连接层使用relu作为激活函数，导致一半结点随机失活变为0.

**再之后重启后，使用原版网络，仍然出现全都是0的情况**

### 2.2.2输出中间过程的bug

我展开了train\_d\_fake函数中Generator部分的函数，定义如下



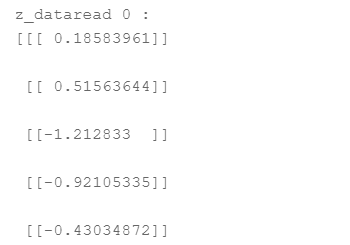


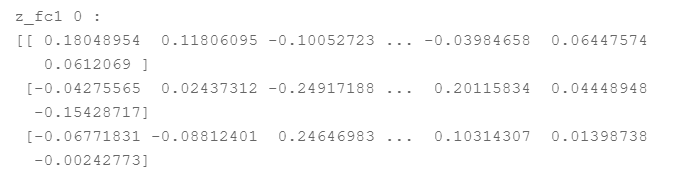
定义了训练函数如下：

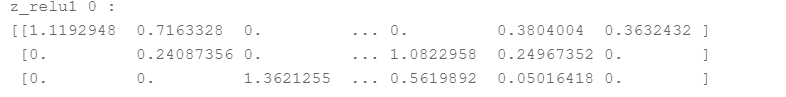


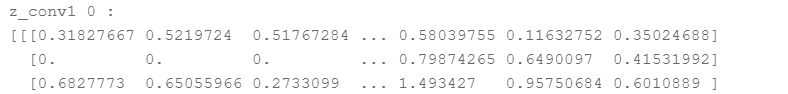


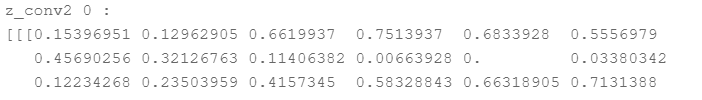
其输出结果见下

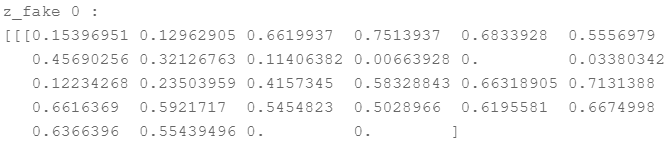






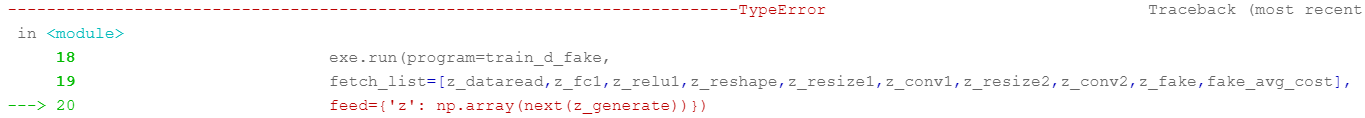




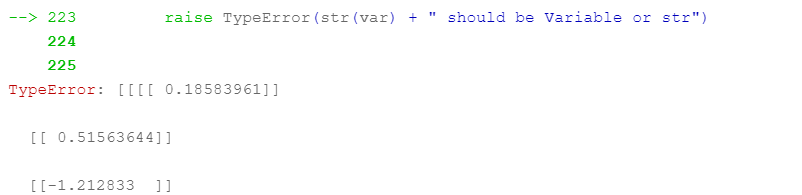


**可见，单部输出结果正常，但是一旦一起输出，结果出现异常**

紧接着出现generator数据产生错误

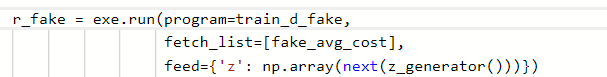


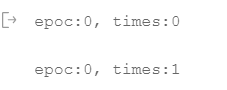
。。。



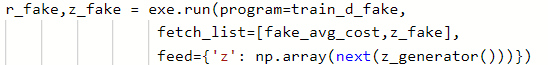
接踵而至的是各种bug

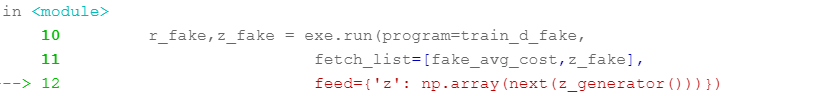
如下方法输出为正常：





如下方法输出异常





错误原因：输出中间过程

### 2.2.3 Bug总结

总结：paddlepaddle出现两个bug

1：直接输出结果，结果为0；分步骤进行输出，结果不为0

2：直接输出结果，不会报错；输出中间的任意一个状态，报错：输入的数据类型有误。然而网络执行过程的与输入数据没有任何关系。

--------------------------------------------------------------分割线----------------------------------------------------------------------------

5月6日星期三，上完人工智能课，我又对Gan网络产生了新的认识，决定续写实验报告

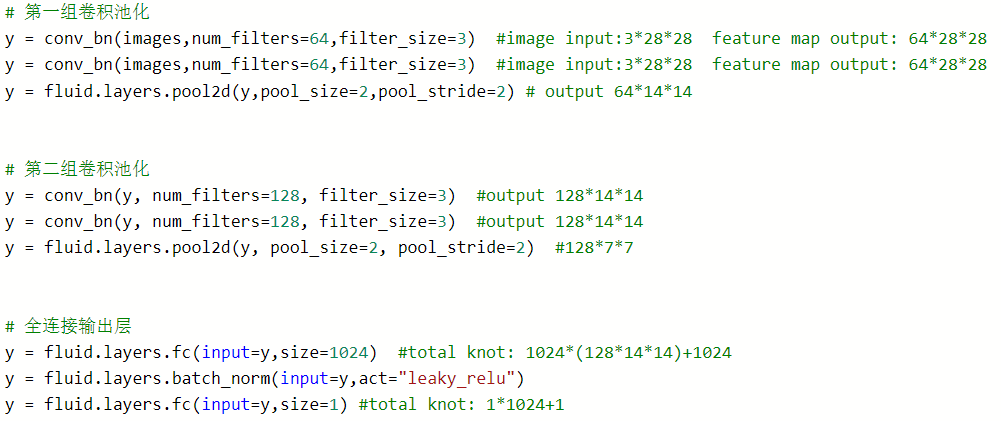
# 3. 采用了RMSProp优化算法后的结果

## 3.1 第一种网络测试

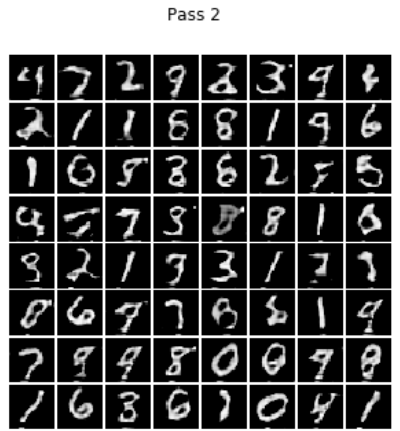
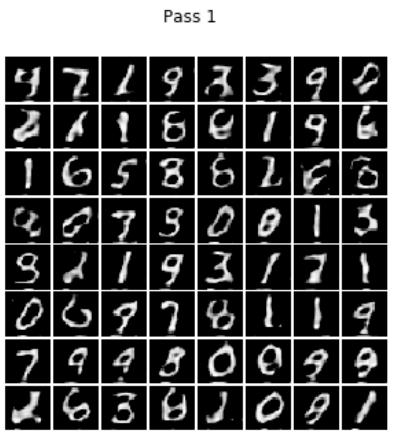
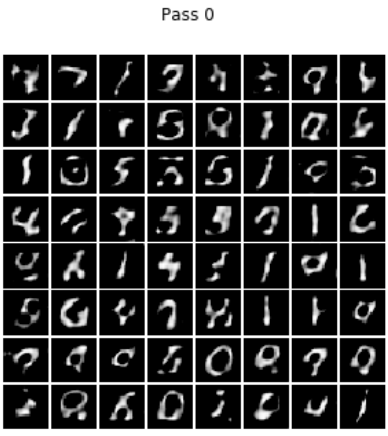
生成器网络：

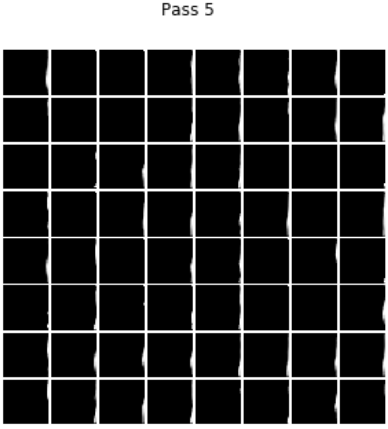
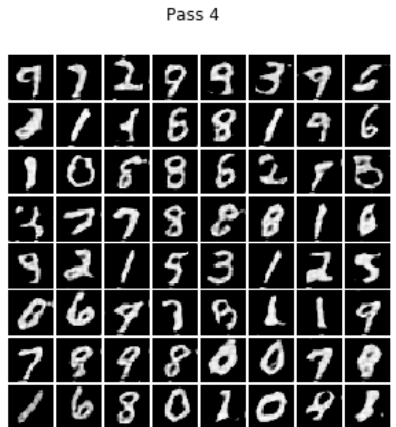


判别器网络



结果：

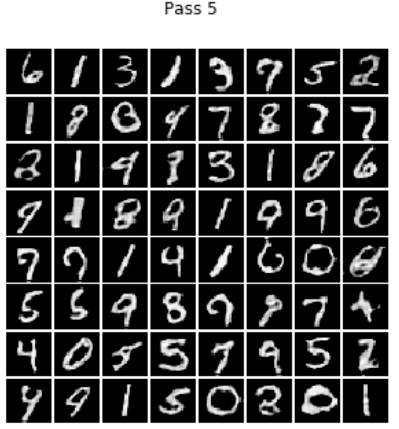
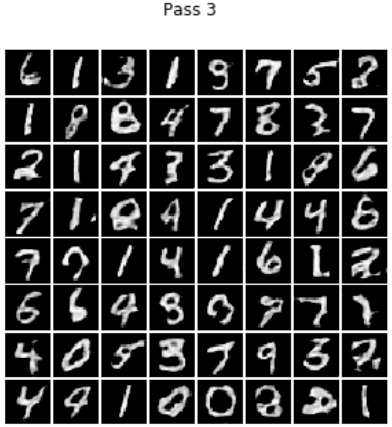
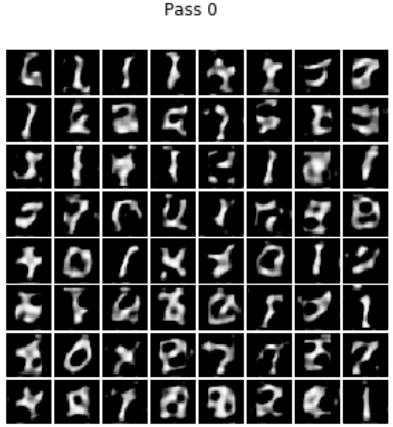


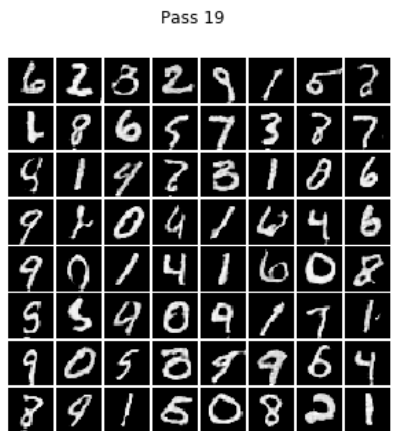
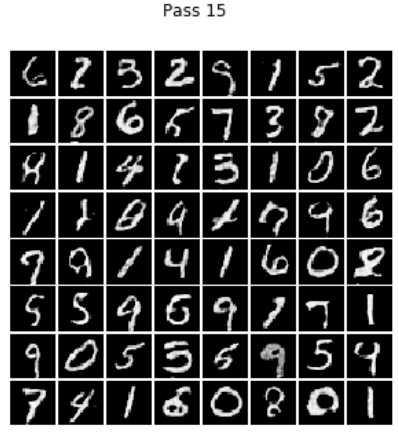
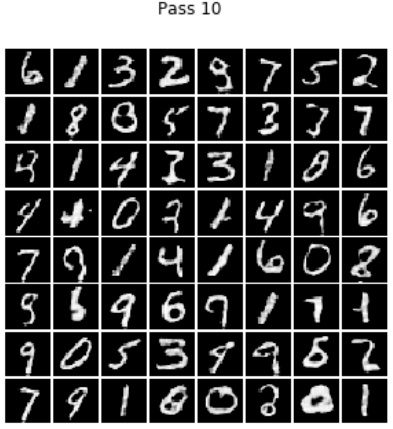


我的猜测是判别器网络太强大，抑制了生成器网络的学习，于是我采用了如下的判别器网络模型



结果如下：





果然效果很好。