## 生成对抗网络实验报告

姓名：张三 学号：xxxxxxxxx

实验要求：

* 掌握GAN原理
* 学会使用PyTorch搭建GAN网络来训练FashionMNIST数据集

报告内容：

* 老师提供的原始版本GAN网络结构（也可以自由调整网络）在FashionMNIST上的训练loss曲线，生成器和判别器的模型结构（print(G)、print(D)）
* 自定义一组随机数，生成8张图
* 针对自定义的100个随机数，自由挑选5个随机数，查看调整每个随机数时，生成图像的变化（每个随机数调整3次，共生成15x8张图），总结调整每个随机数时，生成图像发生的变化。
* 解释不同随机数调整对生成结果的影响（重点部分）
* 格式不限

作业提交：

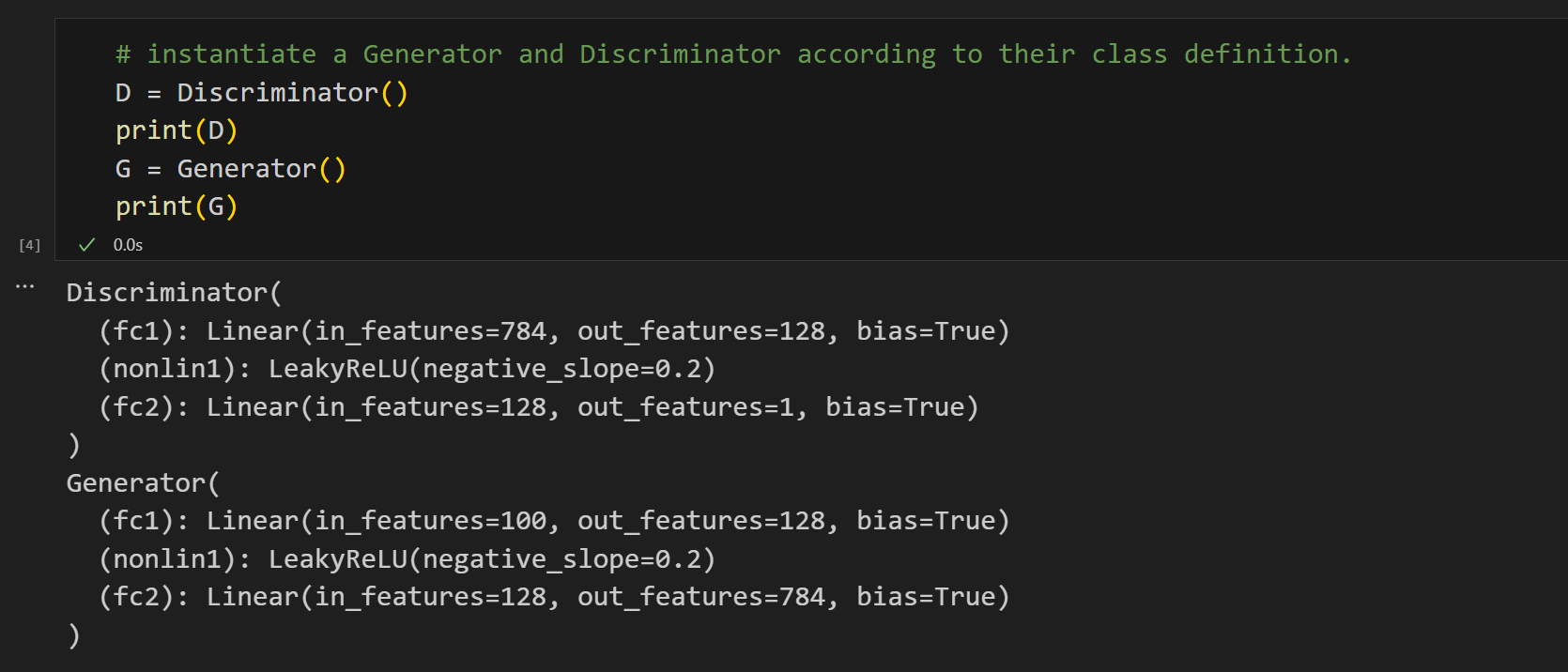
* 期末前将报告和代码（可将jupyter notebook里代码复制到一个xxx.py文件中）打包（学号+姓名.zip），提交方式另行通知
* 实验报告内容应工整
* 加分项：用卷积实现生成器和判别器。

GAN的全称为生成对抗网络，是一种生成模型，最早可以追溯到2014年；在这篇论文中，作者提出了一种通过对抗过程估计生成模型的框架，其中包含两个模型：一个生成模型 G 和一个判别模型 D。生成模型 G 用来生成伪造数据，而判别模型 D 用来评估一个数据样本是真实数据还是伪造数据。这两个模型通过对抗过程相互训练，最终得到一个能够生成类似于真实数据的生成模型。

GAN 是一种非常有效的生成模型，已经在图像生成、语音合成、自然语言处理等领域取得了广泛应用。这篇论文对于深度学习和生成模型的发展具有重要意义，并且一直受到广泛的研究和引用。在此前的生成模型领域，都存在一些问题，例如生成效果不好或过度平滑等；GAN 的诞生开创了生成模型的一个全新的世界。其主要利用了博弈论的原理：训练两个神经网络分别是 D 和 G，分别是判别网络 D 和生成模型 G，生成模型是学习给定样本的数据分布，并尽可能的生成出符合给定样本数据分布的全新数据；判别器的作用是判断给定的样本是生成器生成出来的还是原始数据。我们期望于判别模型尽可能的能够分清给定数据到底是生成器生成的，还是原始数据分布；并且期望于生成器能够尽可能的逼近原始数据，做到以假乱真的效果。

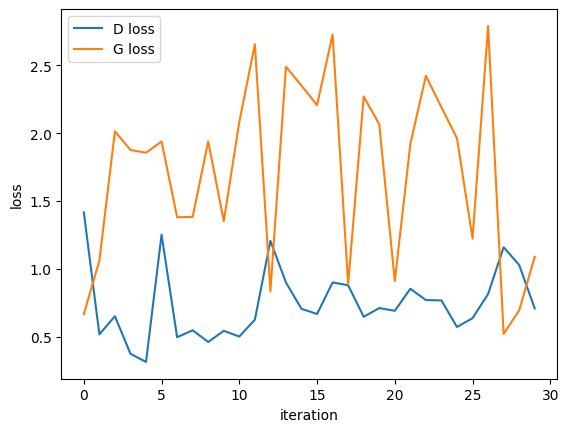
最终我们可以达到纳什平衡：判别器对于给定的数据有 50% 的概率认为是原始数据，有 50% 的概率认为是生成器生成的数据。这样就达到了我们 GAN 的收敛效果。事实上，可以用验钞机和罪犯的例子来理解：印假钞的罪犯期望于能够做出以假乱真的假币，在外人眼里与真钞毫无区别；而我们的验钞机则想要区分出一张钞票的真伪。

在本次实验中，我们使用老师给定的代码作为Baseline，并通过卷积实现加分项。

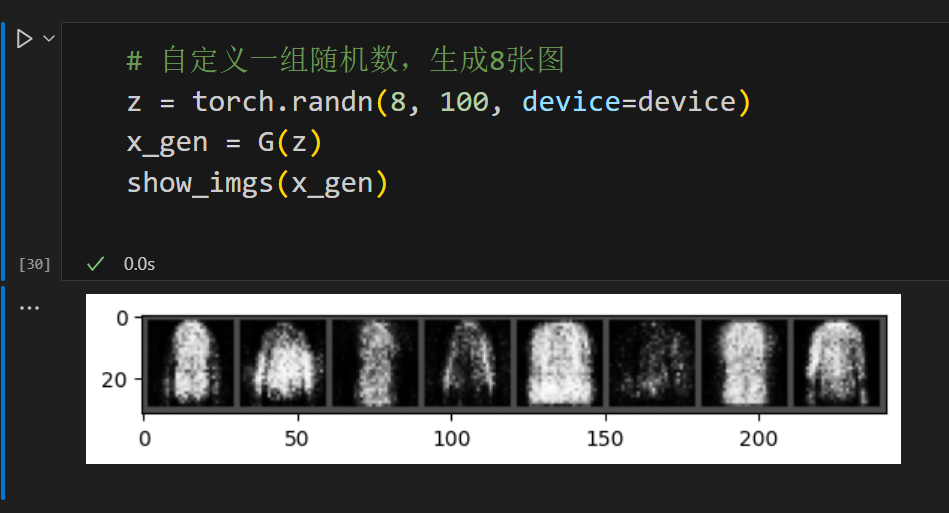


上面为老师给出的判别器和生成器，我们可以看到，生成器最终是要输出一个784维的向量，也即28\*28的一个灰度图，作为我们生成的模型；同时判别器只需要输出一个数值，作为判别器对于输入图像的判断(是否是生成器生成的)。

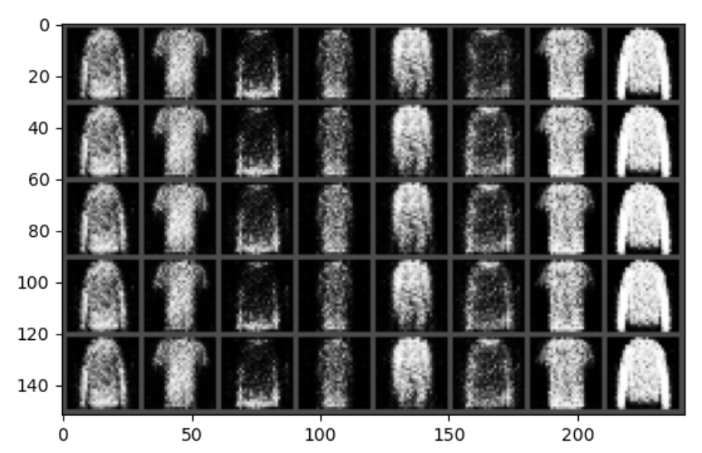
我们在FashionMNIST上进行训练，训练过程中loss如下所示：



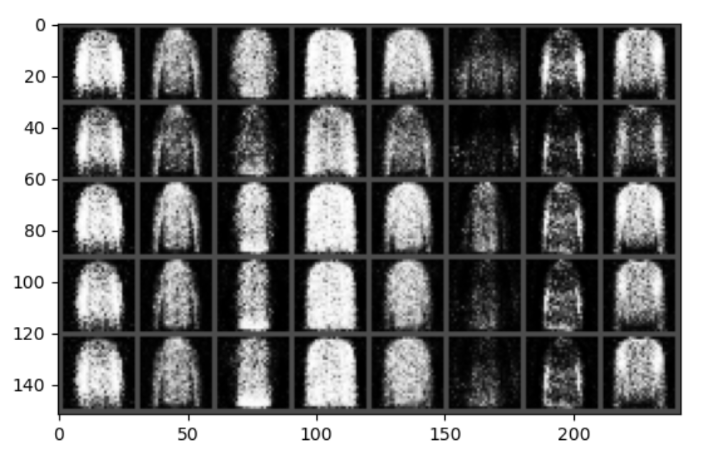
接下来我们自行定义随机数，生成八张图，如下所示：



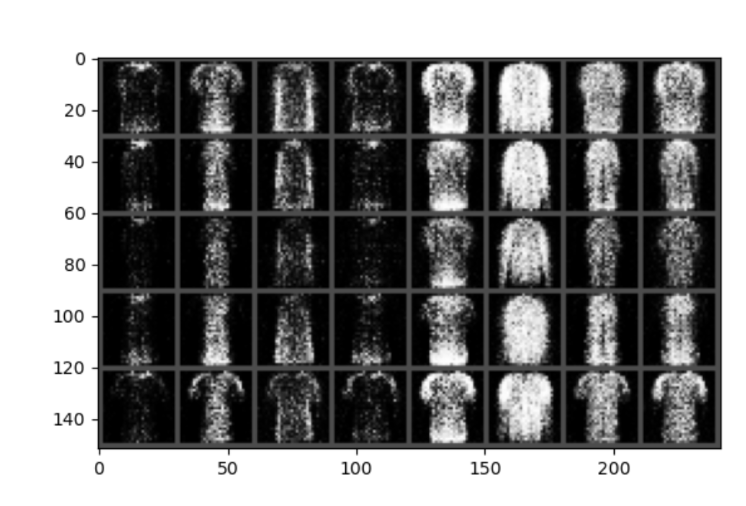
接下来我们随机调整五个维度，并且维度上输出原图，+0.5和-0.5三种：



上图为随机数为0.5

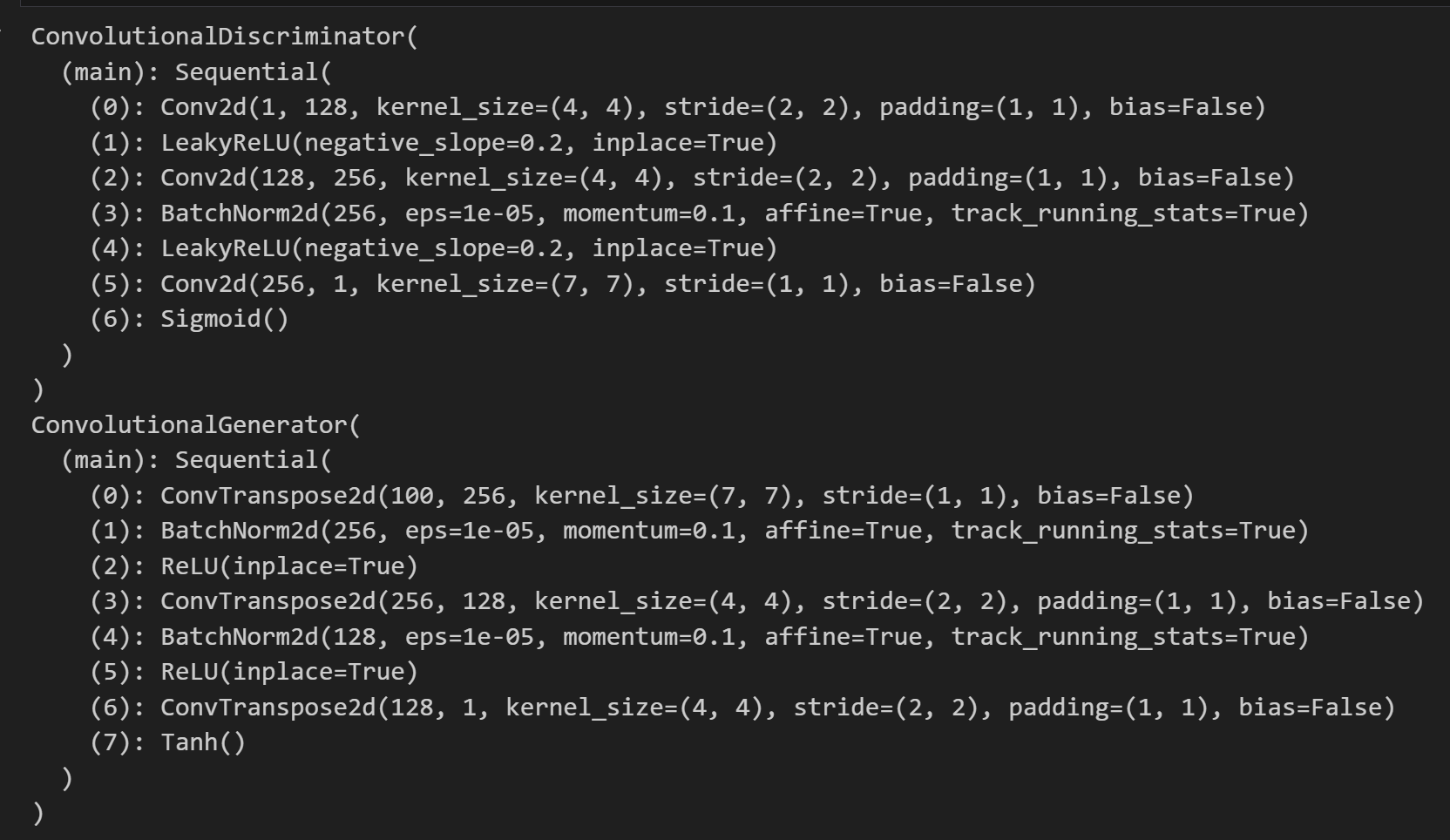


上图为随机数为3

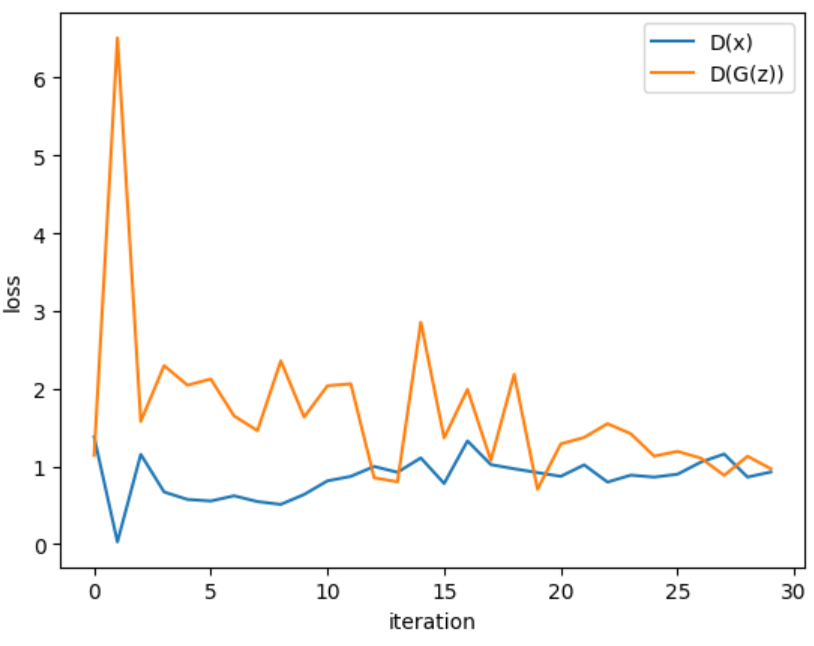


上图为随机数为10。首先我们观察可以看出，当随机数设定的较小的时候(如03)，在不同位置改变一个这样的小随机数，收获的效果是比较小的，甚至可以说基本没有什么肉眼可以观察出来的变化;但是当随机数设置的较大的时候(如10)，如果我们在不同位置改变一个这样的小随机数，收获的效果还是比较明显的，可以很清晰的从最后一张图上看出，相对于位置60变为10，位置80的改变直接让裤子变成了两件衣服。当我们在观察相同位置的随机数改变时，我们发现，过小的随机数(0.3)或者过大的随机数(10)都会导致生成的图像的亮度较低，甚至有些图片接近于全黑，而当我们使用一个适中的随机数的时候，我们的模型生成的效果相对来说较为明亮，且生成的图像的效果较好。

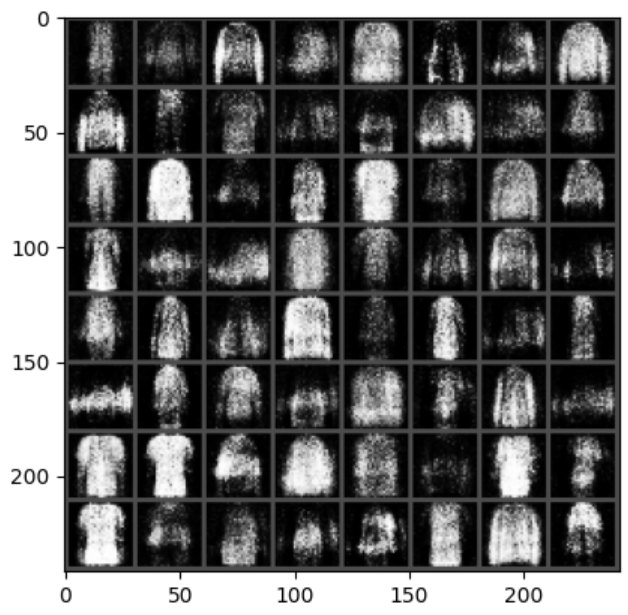
接下来我们使用卷积实现生成器和判别器，对于生成器，通常涉及到从一个较小的噪声向量开始，逐步上采样生成最终图像；对于判别器，则需要逐步下采样来判断图像是真是假。我们给出最终设计的网络结构如下所示：



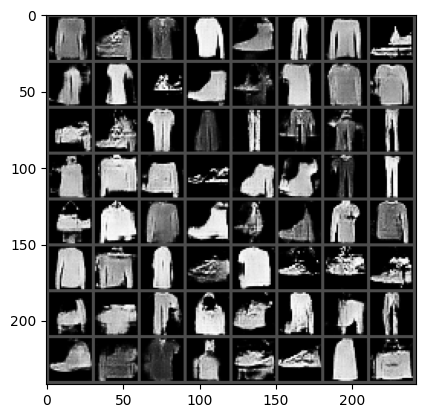
我们同样使用给出代码进行训练，不过调节为Adam优化器，并将学习率设低，训练过程的D(x)和D(G(z))如下所示：



我们可以看到，对于CNN实现的生成器和判别器，确实相比于MLP实现的具有更好的性能，其更加倾向于在纳什平衡点附近收敛。此外对于生成效果，MLP的生成效果如下所示：



而CNN实现的效果如下：



明显可以看到，CNN实现的效果要远远浩宇MLP，其分辨率会更好，看着会更加清晰也更加明亮。