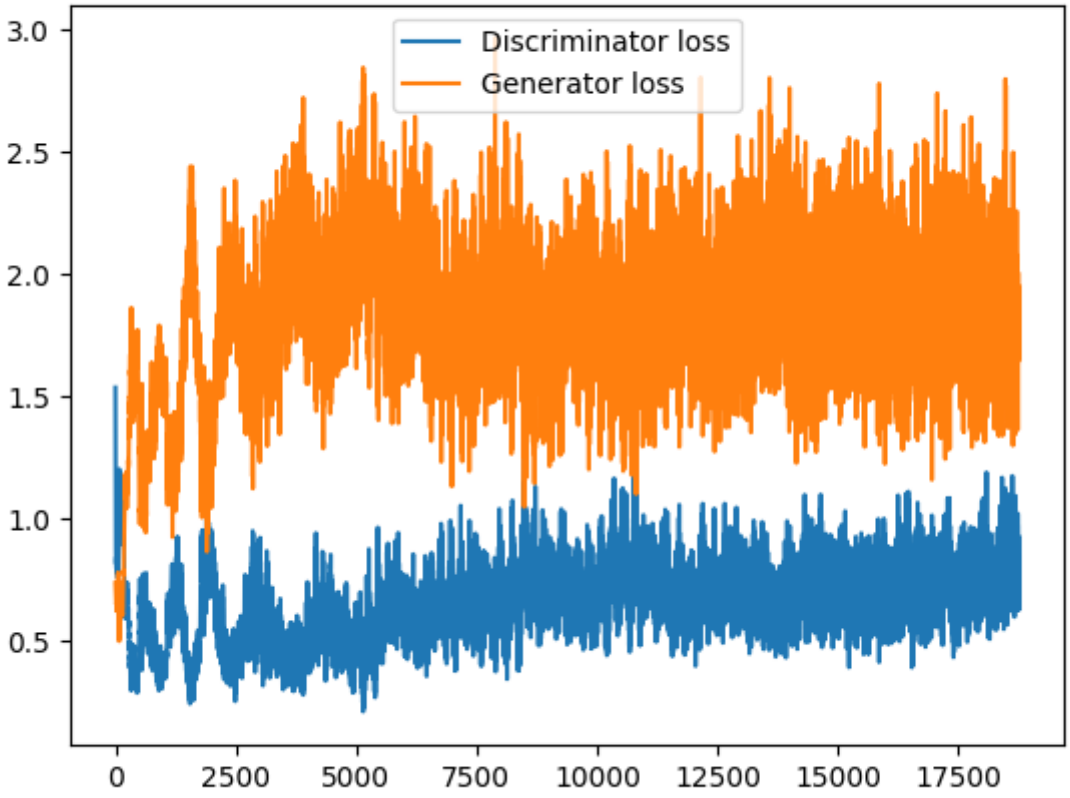
生成对抗网络实验报告

姓名：崔江浩
学号：2011915
专业：计算机科学与技术

1 原始GAN

1.1 loss曲线



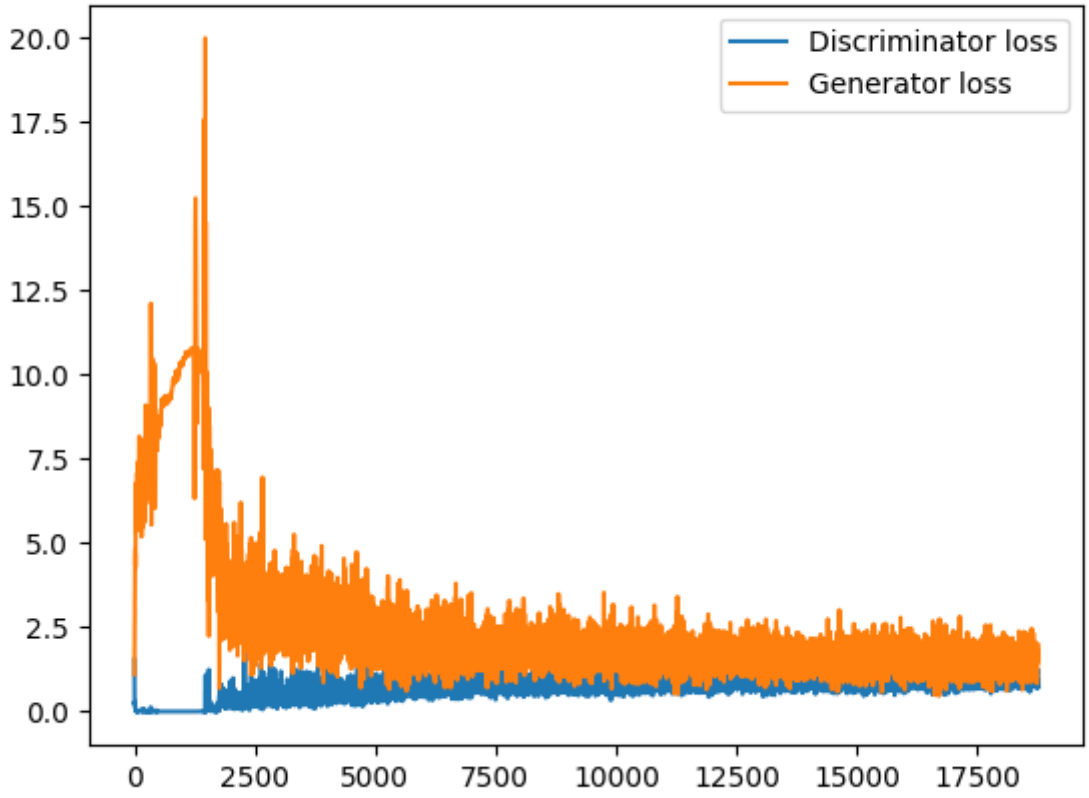
GAN（生成对抗网络）的训练过程是一场双方不断调整和优化的minimax博弈。在这场博弈中，生成器和判别器一直在对抗中学习和进化。因此，GAN的loss曲线通常不会像其他监督学习模型那样平滑递减，而是会出现频繁的起伏，即损失值可能时而上升，时而下降。这就是GAN训练难度较高的一个原因，因为我们无法仅通过观察loss值的变化来准确评估GAN的训练效果。

1.2 生成器和判别器的模型结构

```
Discriminator(
  (fc1): Linear(in_features=784, out_features=128, bias=True)
  (nonlin1): LeakyReLU(negative_slope=0.2)
  (fc2): Linear(in_features=128, out_features=1, bias=True)
)
Generator(
  (fc1): Linear(in_features=100, out_features=128, bias=True)
  (nonlin1): LeakyReLU(negative_slope=0.2)
  (fc2): Linear(in_features=128, out_features=784, bias=True)
)
```

2 用卷积实现生成器和判别器的GAN

1.1 loss曲线



1.2 生成器和判别器的模型结构

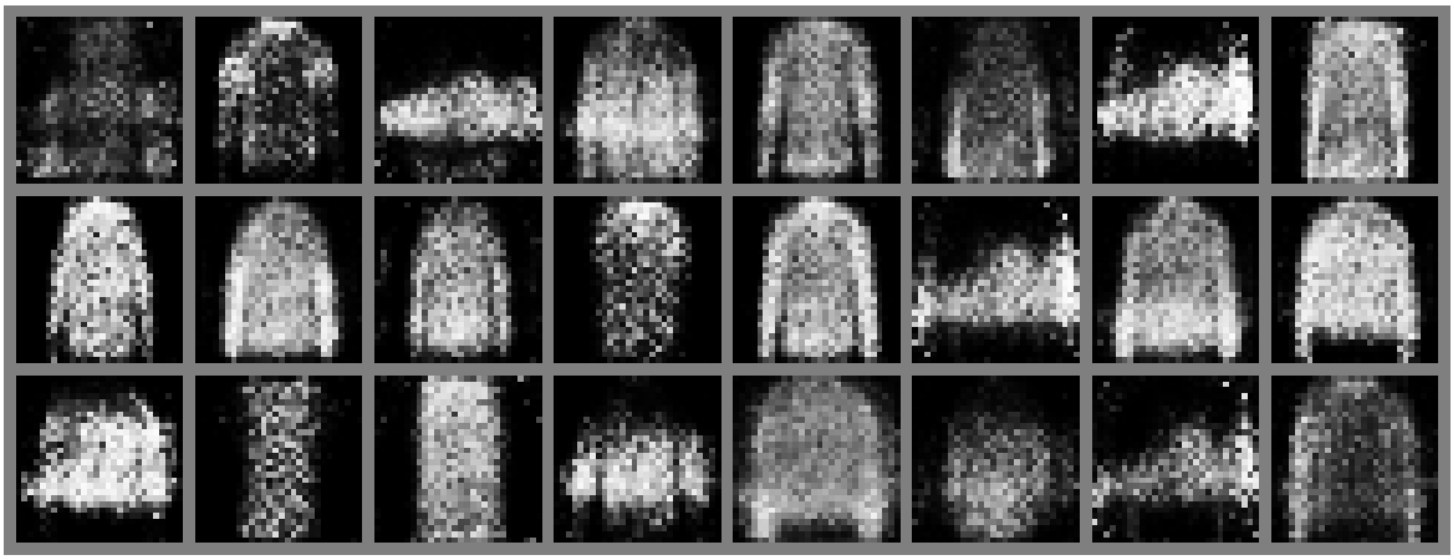
```
Discriminator(
  (main): Sequential(
    (0): Conv2d(1, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): Conv2d(256, 1, kernel_size=(7, 7), stride=(1, 1), bias=False)
    (6): Sigmoid()
  )
)
Generator structure:
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(100, 256, kernel_size=(7, 7), stride=(1, 1), bias=False)
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1),
bias=False)
    (4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(128, 1, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): Tanh()
  )
)
```

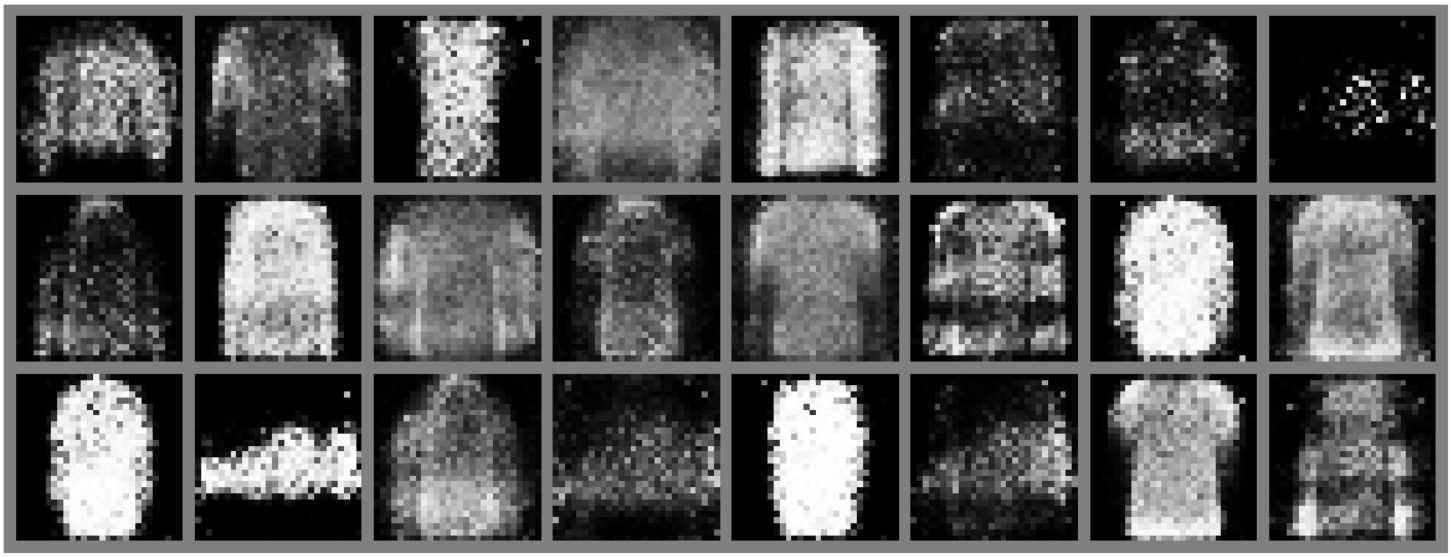
3 自定义一组随机数，生成8张图

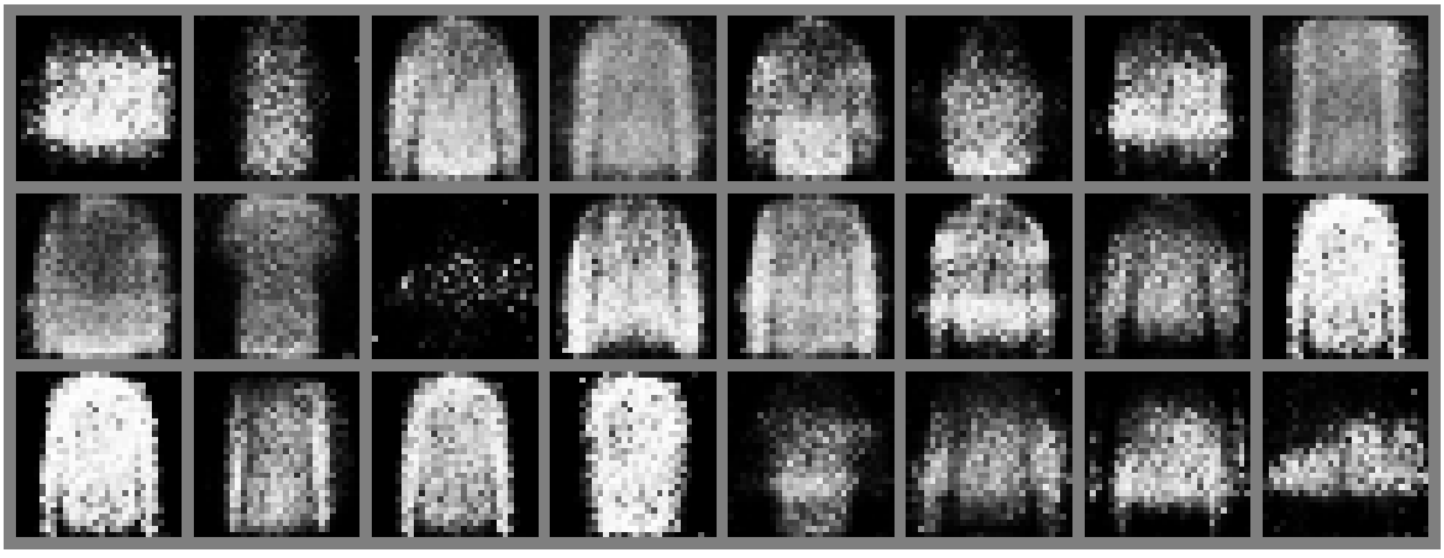
4 针对自定义的100个随机数，自由挑选5个随机数

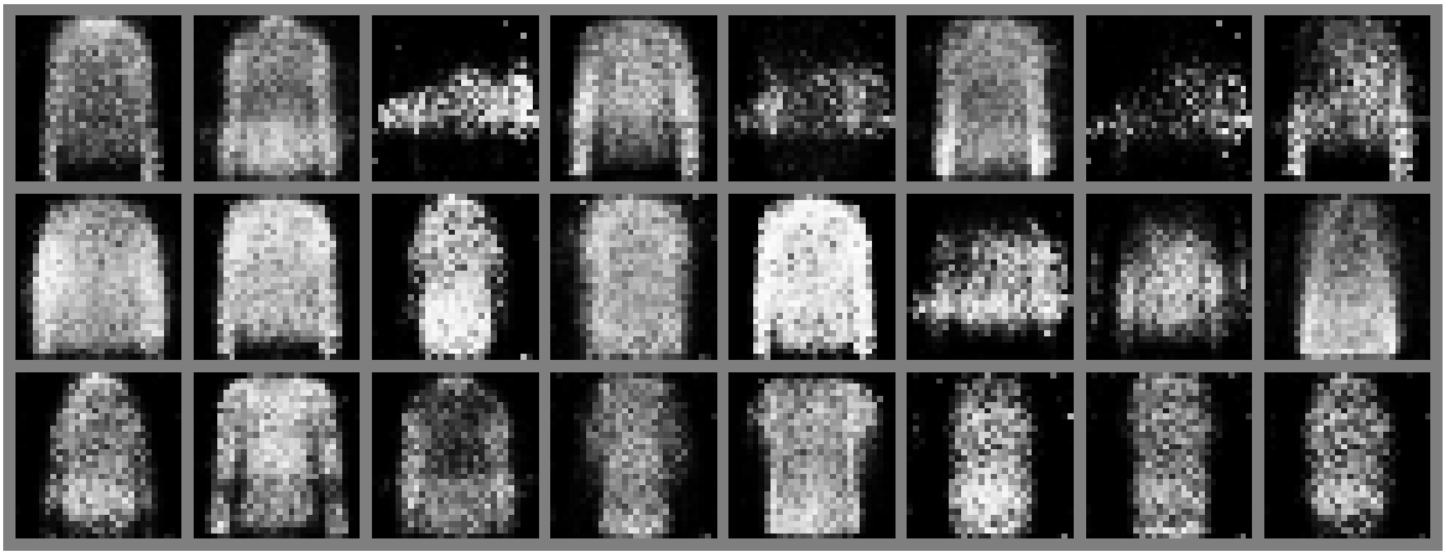
我将导入模型并为100个随机向量生成图像。然后，选择5个随机向量并稍微调整它们，观察生成的图像如何变化。然后分别对这5个随机向量进行3次小幅度调整，观察图像如何随着这些调整而变化。总的来说，将生成15 (随机向量) x 8 (图像/向量) = 120张图像。

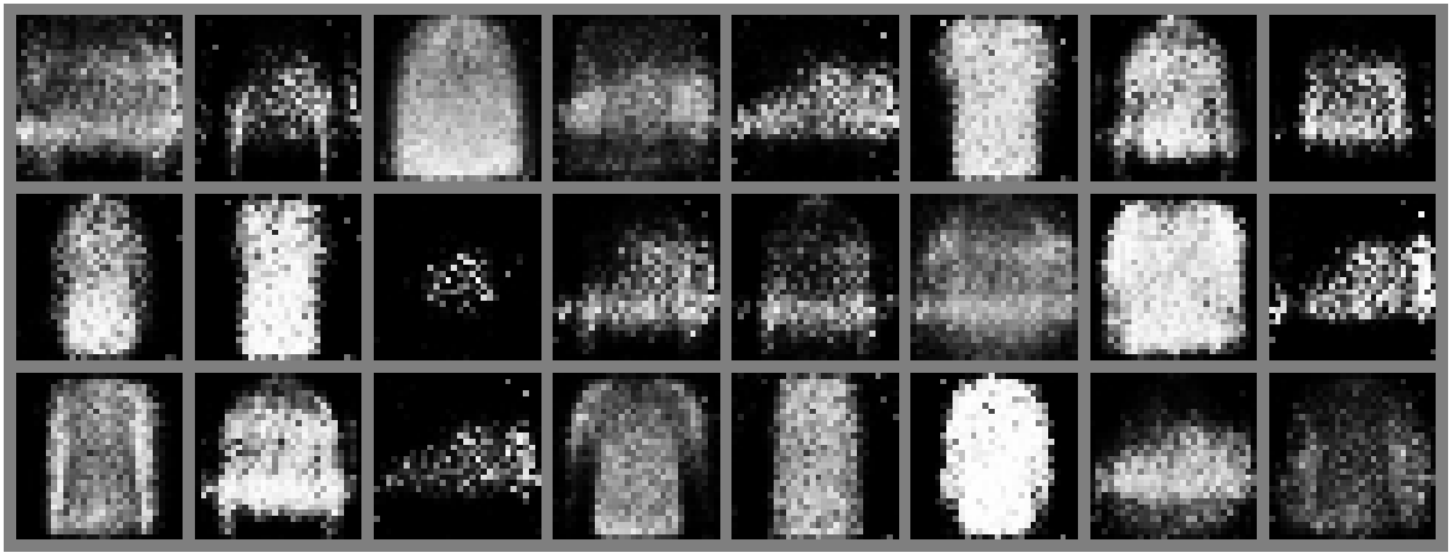
本次实验对每一个选定的随机数（这里选择了索引为1, 20, 40, 60, 80的随机数）做了三次调整。这些调整值在-2到2之间均匀分布，即调整值为-2, 0, 和2。











首先，让我们观察一下实验结果。我们可以发现，当我们设置较小的随机数时，在不同位置改变一个这样的小随机数几乎没有明显的效果，甚至肉眼难以观察到变化。然而，当我们将随机数设定为较大值时，即使在不同位置改变一个小随机数，效果仍然明显。

在观察相同位置的随机数变化时，我们发现过小的随机数或过大的随机数都会导致生成的图像亮度较低，甚至有些图像接近全黑。然而，当我们使用适中的随机数时，生成的图像相对明亮，并且效果较好。

5 解释不同随机数调整对生成结果的影响

在这个特定的GAN模型中，生成器使用一个随机噪声向量作为输入，然后通过一系列的计算来产生一张新的图像。这个随机噪声向量中的每一个数值，都对生成的图像有一定的影响。当我们调整随机噪声向量中的某一个特定数值（例如索引为1, 20, 40, 60, 80的那些）时，我们就改变了生成器接收的输入，因此生成的图像也会发生变化。具体的变化可能包括图像的风格、色彩、形状、纹理等属性，这个取决于生成器模型的内部结构和训练方式。