南開大學
Nankai University

计算机学院

深度学习与应用实验报告

GAN 实验报告

学号：2112529

姓名：赵廷枫

2024 年 6 月 21 日

目录

1 实验要求

1. 掌握 GAN 原理
2. 学会使用 PyTorch 搭建 GAN 网络来训练 FashionMNIST 数据集
3. 用卷积实现生成器和判别器：学习 DCGAN 网络结构并使用 FashionMNIST 来训练

2 实验内容

1. 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，成器和判别器的模型结构（print(G)、print(D)）
2. 自定义一组随机数，生成 8 张图
3. 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。
4. 解释不同随机数调整对生成结果的影响（重点部分）
5. 学习 DCGAN 网络原理并搭建 DCGAN 网络结构
6. 在 FashionMNIST 数据集上训练 DCGAN

3 GAN

生成对抗网络（GAN[1]）是一种由两个神经网络组成的机器学习模型，一个生成器（Generator）和一个判别器（Discriminator），通过相互对抗的方式进行训练。生成器试图生成逼真的数据样本，而判别器则试图区分这些生成的样本与真实数据。生成器的目标是最大程度地欺骗判别器，使其无法区分生成的数据和真实数据，而判别器的目标是提高其识别真实数据和生成数据的准确性。通过这种博弈过程，生成器不断改进，最终生成的样本越来越逼真。

3.1 网络结构

根据论文中对于模型的原理阐述以及老师给出的实验指导，我设计的 GAN 网络结构如下：

```
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)
)
```

使用上述模型在 FashionMNIST 数据集上进行训练，共训练了 10 个 epoch，训练 1、5、10 个 epoch 之后的一些结果如下：

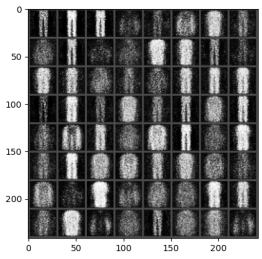

图 3.1: 第一个 epoch

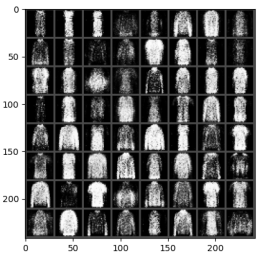

图 3.2: 第五个 epoch

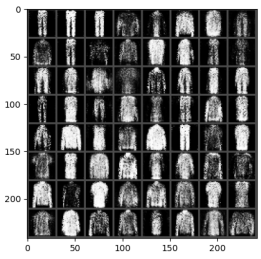

图 3.3: 第十个 epoch

3.2 损失曲线

全部训练过程中生成器和判别器的损失曲线如下

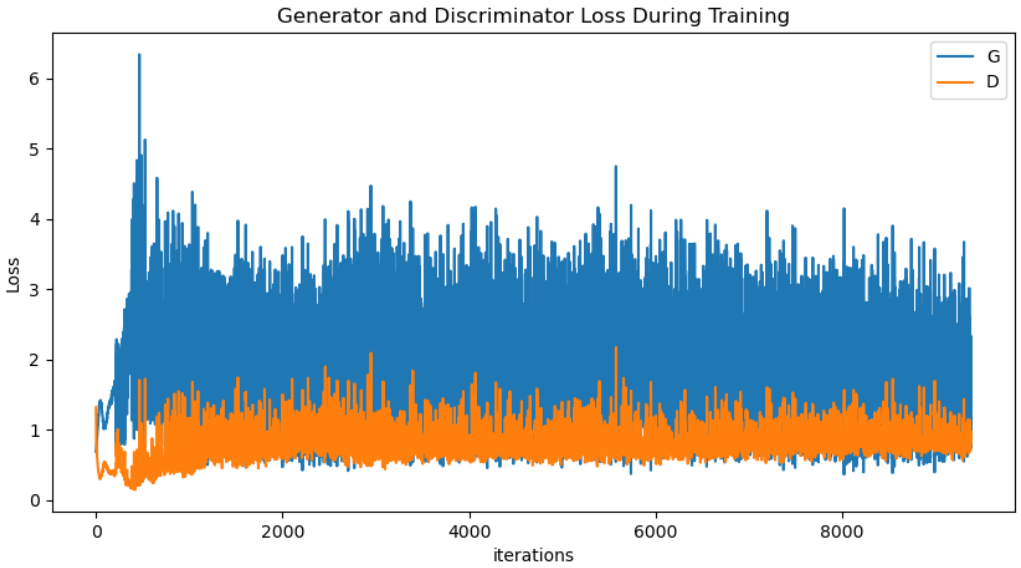

图 3.4: 初始版本 GAN 训练过程中生成器和判别器的损失曲线

可以看出生成器损失一开始先上升后下降之后便开始不断波动，判别器损失先下降后上升之后开始不断波动，这符合生成器和判别器两者不断对抗过程中应有的损失变化。

4 随机数实验

4.1 自定义随机数

我们首先是自定义了 100 个随机数，然后使用这 100 个随机数生成八张图片如下：

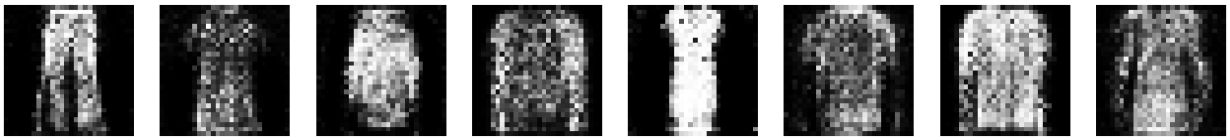

图 4.5: 自定义随机数生成八张图片

4.2 调整随机数

之后我们对其中的五个随机数进行调整，每次进行三次调整，分别是乘以 2、置 0、取相反数，如此得到的 15 组八张图片如下：

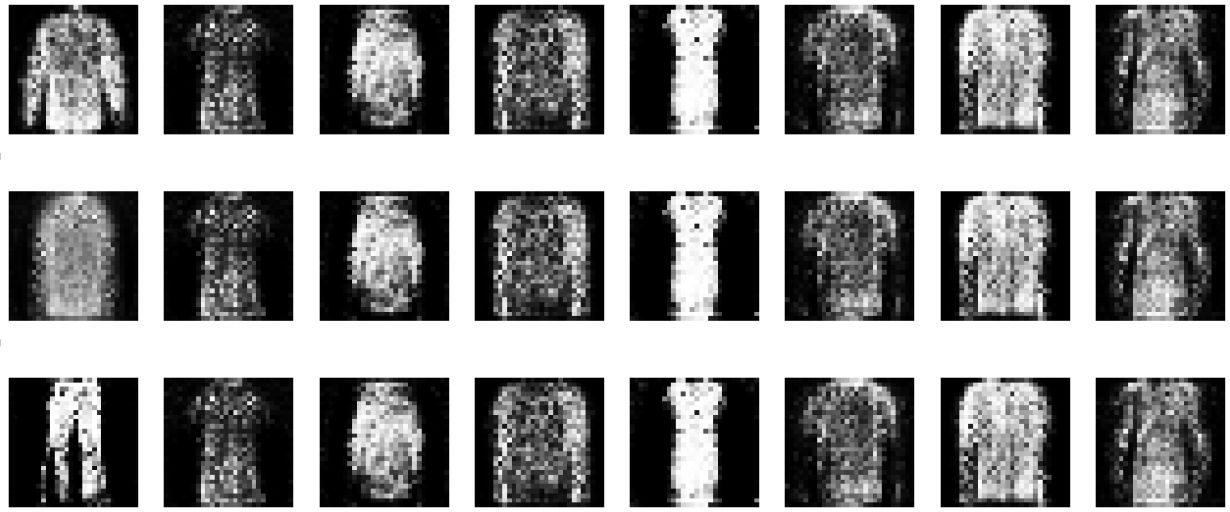

图 4.6: 对第一个随机数进行三次调整

5 随机数分析

随机数通常用作生成器（Generator）的输入，这些随机数通常被称为“噪声”或“潜在向量”（latent vectors）。通过调整这些潜在向量，生成器可以产生多样化的输出，从而使得生成的数据集尽可能地覆盖真实数据集的分布。

噪声向量的随机性直接决定了生成器输出的多样性。不同的随机输入应该理想地导致不同的输出，这有助于模型学习到从潜在空间到数据空间的广泛映射。如果噪声向量缺乏足够的随机性或变化，生成的样本可能会表现出过于相似的特征，即所谓的“模式崩溃”（mode collapse）。

在我们具体的实验中我们发现，只修改某一个随机数只会影响某一个图片的生成，其他的图片不受影响，这也印证了我们上面对于原理的解释。

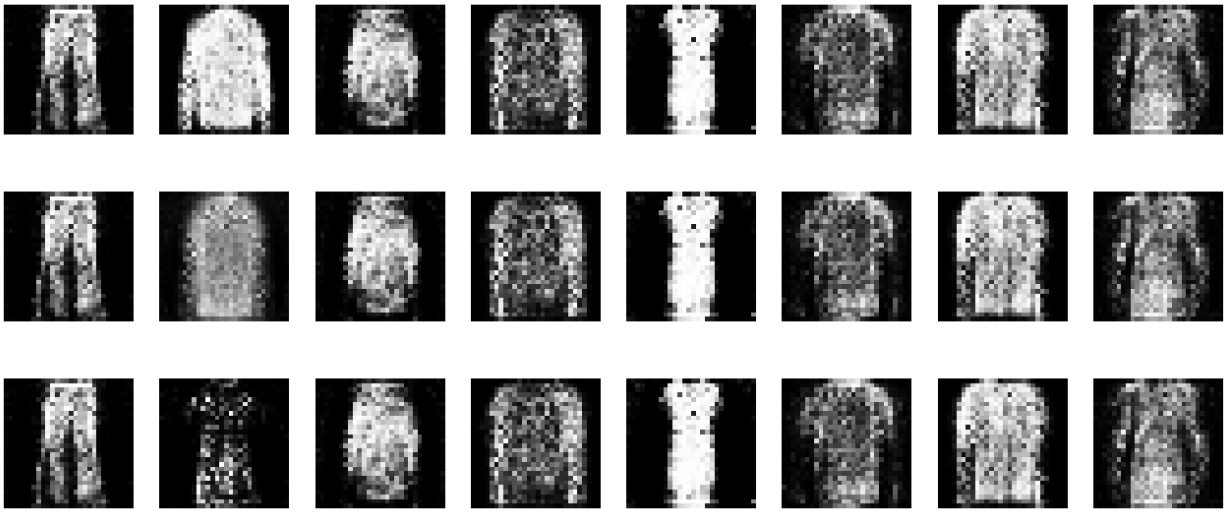

图 4.7: 对第二个随机数进行三次调整

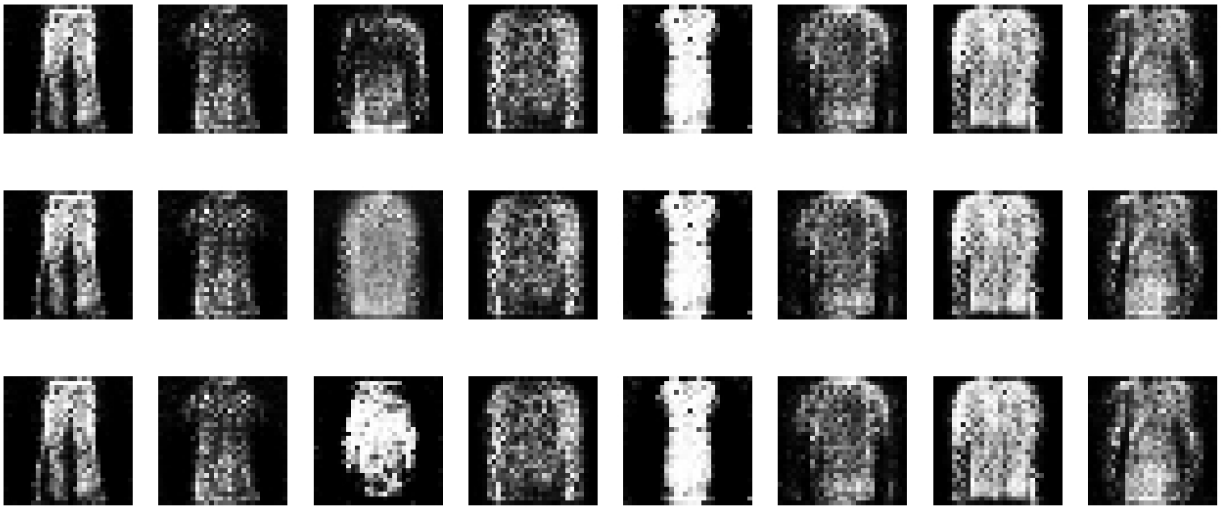

图 4.8: 对第三个随机数进行三次调整

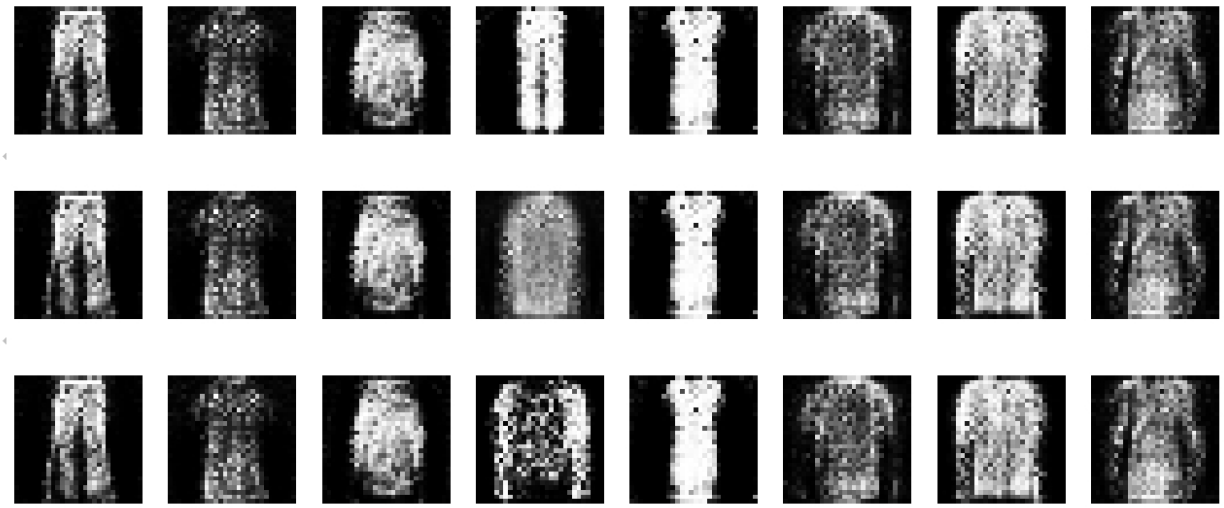

图 4.9: 对第四个随机数进行三次调整

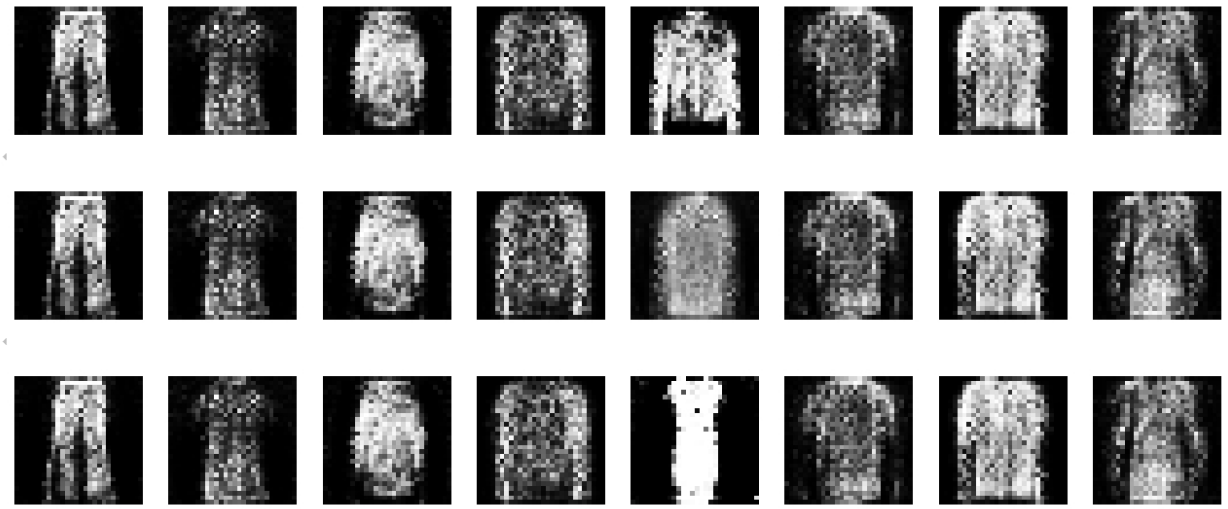

图 4.10: 对第五个随机数进行三次调整

6 DCGAN

深度卷积生成对抗网络（DCGAN[2]）是生成对抗网络（GAN）的一个变体，利用卷积神经网络（CNN）来改进图像生成任务。DCGAN 由一个生成器（Generator）和一个判别器（Discriminator）组成，生成器负责生成逼真的图像，判别器则负责区分真实图像和生成图像。生成器使用卷积转置层逐步将低维噪声向量转换为高维图像，而判别器使用卷积层提取图像特征进行分类。通过生成器和判别器之间的对抗训练，生成器逐渐学会生成更加真实的图像，从而提升生成质量。DCGAN 的关键在于其卷积结构，使得生成的图像具有更好的局部特征和整体质量。

6.1 DCGAN 网络结构

将论文中的模型结构总结如下，左边为判别器结构，右边为生成器结构：

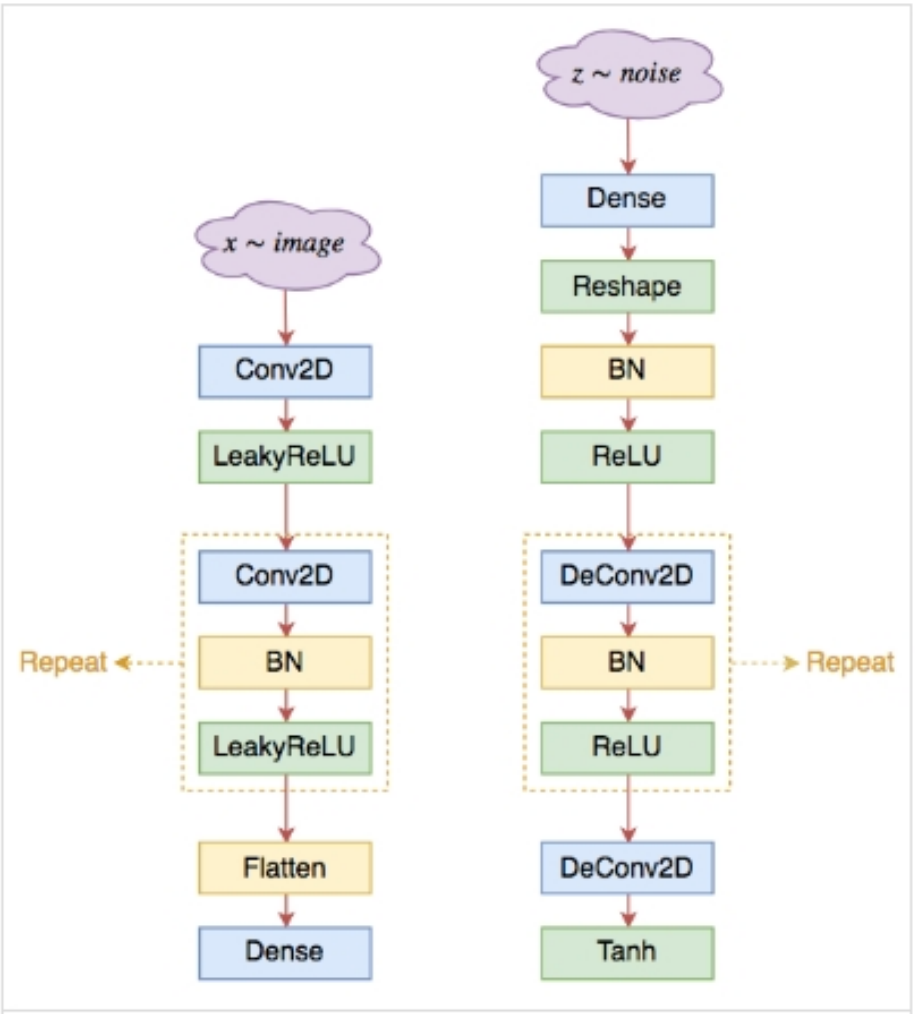

图 6.11: DCGAN 模型结构，图片来自：https://kexue.fm/archives/6549

参考上述论文以及开源仓库https://github.com/s-chh/Pytorch-DCGAN32.git中对于原理的讲解以及具体实现的细节，我设计的 DCGAN 网络结构如下：

使用上述模型在 FashionMNIST 数据集上进行训练，共训练了 10 个 epoch，训练 1、5、10 个 epoch 之后的一些结果如下：

```
Discriminator(
  (conv1): Conv2d(1, 16, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2))
  (conv2): Conv2d(16, 32, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2))
  (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc): Linear(in_features=3136, out_features=1, bias=True)
)
```

图 6.12: 判别器模型结构

```
Generator(
  (fc1): Linear(in_features=100, out_features=3136, bias=True)
  (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (deconv2): ConvTranspose2d(64, 32, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2), output_padding=(1, 1))
  (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (deconv3): ConvTranspose2d(32, 16, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2), output_padding=(1, 1))
  (bn3): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (deconv4): ConvTranspose2d(16, 784, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
)
```

图 6.13: 生成器模型结构

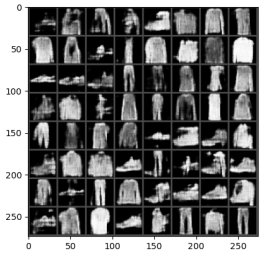

图 6.14: 第一个 epoch

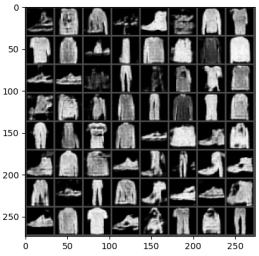

图 6.15: 第五个 epoch

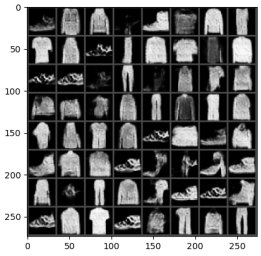

图 6.16: 第十个 epoch

6.2 损失曲线

之后我们在 FashionMNIST 数据集上进行训练得到的生成器和判别器损失曲线如下：

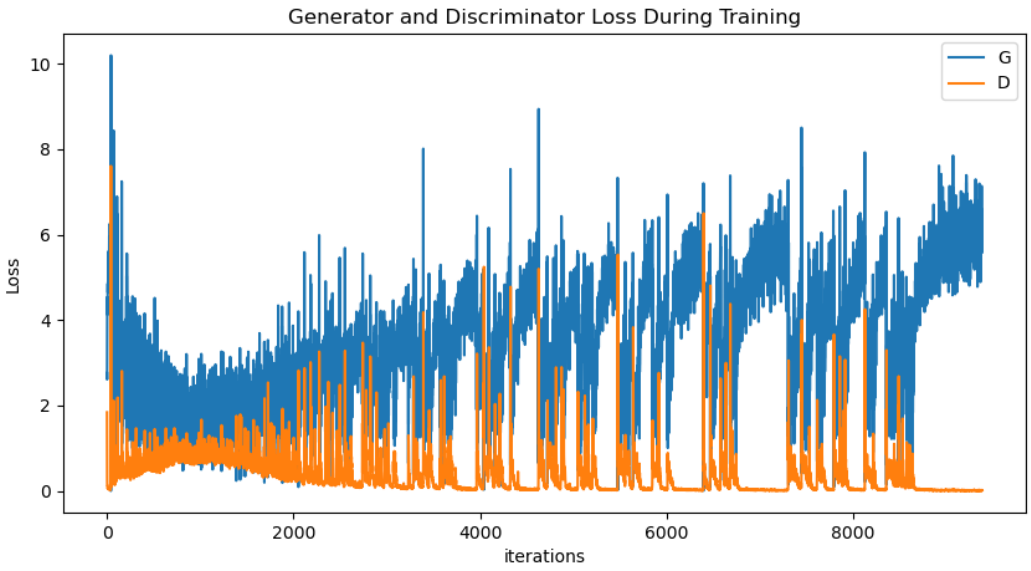

图 6.17: DCGAN 训练过程损失曲线

7 实验总结

在本次实验中，我们首先掌握了生成对抗网络（GAN）的原理，并使用 PyTorch 搭建了基本的 GAN 网络。通过训练 GAN 模型在 FashionMNIST 数据集上的表现，我们记录了训练过程中生成器和判别器的 loss 曲线，并通过调整网络结构和超参数来优化模型性能。训练结果表明，生成器逐步学会生成逼真的 FashionMNIST 图像，判别器的判别能力也随之提高。

接着，我们自定义了一组随机数，生成了多张图像，通过观察不同随机数对生成结果的影响，总结了随机数在潜在空间中的作用。实验结果显示，随机数的不同调整会显著影响生成图像的特征和细节，验证了 GAN 在捕捉数据分布多样性方面的能力。

随后，我们学习了深度卷积生成对抗网络（DCGAN）的原理，并使用卷积神经网络搭建了 DCGAN 结构。在 FashionMNIST 数据集上的训练结果表明，DCGAN 相比于基础 GAN，在生成图像的局部特征和整体质量上有显著提升。通过对比不同模型的生成效果，我们发现 DCGAN 生成的图像更为细腻逼真。

总体而言，本次实验不仅加深了我们对 GAN 和 DCGAN 原理的理解，还通过实战操作掌握了使用 PyTorch 搭建和训练这些网络的技能，为后续的深度学习研究和应用打下了坚实的基础。收获颇丰。

参考文献

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[2] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.