计算机科学与技术学院神经网络与深度学习课程实验报告

实验题目：cGAN		学号：201918130222
日期：2021/11/28	班级： 智能	姓名： 魏江峰
Email：2257263015@qq.com		

实验目的：

完成 GAN，探索并开发一个 GAN 用于服装的生成。

实验软件和硬件环境：

硬件环境：

处理器：Intel core i7 9750-H

电脑：神州 z7m-ct7nk

软件环境：

Pycharm 与 jupyter notebook

实验原理和方法：

生成式对抗网络（GAN, Generative Adversarial Networks ）是一种深度学习模型，是近年来复杂分布上无监督学习最具前景的方法之一。模型通过框架中（至少）两个模块：生成模型（Generative Model）和判别模型（Discriminative Model）的互相博弈学习产生相当好的输出。原始 GAN 理论中，并不要求 G 和 D 都是神经网络，只需要是能拟合相应生成和判别的函数即可。但实用中一般均使用深度神经网络作为 G 和 D 。一个优秀的 GAN 应用需要有良好的训练方法，否则可能由于神经网络模型的

自由性而导致输出不理想。

实验步骤：（不要求罗列完整源代码）

1，导入库并加载数据集

2，完成 Generator 类

__Init__:

```
self.model_neural = nn.Sequential(
        nn.Linear(opt.latent_dim + opt.label_dim, 128),
        # nn.ReLU(),
        nn.LeakyReLU(0.2),
        nn.Linear(128, 256),
        # nn.ReLU(),
        nn.LeakyReLU(0.2),
        nn.Linear(256, 512),
        # nn.ReLU(),
        nn.LeakyReLU(0.2),
        nn.Linear(512, 1024),
        # nn.ReLU(),
        nn.LeakyReLU(0.2),
```

```
            nn.Linear(1024, np.prod(img_shape)),
            nn.Tanh()


            )
 def forward(self, noise, labels):
        # START CODE HERE
        # your code here
        # concatenate noise and label
        # get_input = torch.cat((noise, self.label_embedding(labels)), 1)
        c = self.label_embedding(labels)
        gen_input = torch.cat([noise,c],1)
        img = self.model_neural(gen_input)
        # END CODE HERE
        return img
```

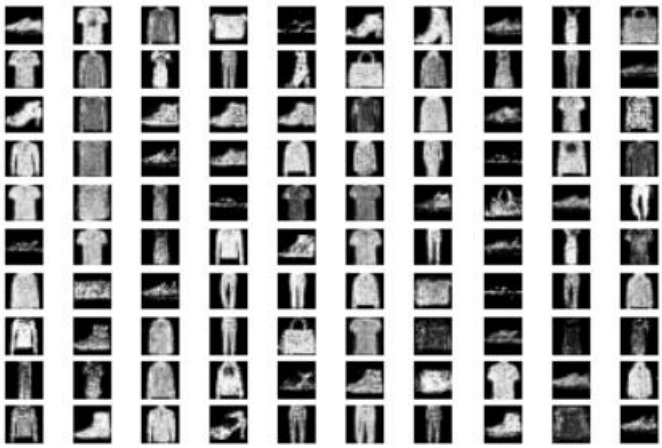
3，完成 discriminator 类：

```
__init__:
 self.model_neural = nn.Sequential(
            nn.Linear(opt.label_dim + 28*28, 512),
            # nn.ReLU(),
            nn.LeakyReLU(0.2),
```

```
            nn.Linear(512, 512),
            # nn.ReLU(),
            nn.LeakyReLU(0.2),

            nn.Linear(512, 512),
            # nn.ReLU(),
            nn.LeakyReLU(0.2),

            nn.Linear(512, 1),
            nn.Sigmoid()# 之前训练的时候忘了使用这个
        )
    def forward(self, img, labels):

        # START CODE HERE
        # your code here
        # concatenate image and label

        img = nn.Flatten()(img)

        dis_input = torch.cat((img, self.label_embedding(labels)), 1)
        validity=self.model_neural(dis_input)
        # END CODE HERE
```

```
        return validity


4,训练 GAN
random_vector = torch.randn(batch_size,100)
        if cuda:
            random_vector = random_vector.cuda()
        fake_imgs = generator(random_vector, labels)
optimizer_G.zero_grad()
        g_loss.backward(retain_graph=True)
        optimizer_G.step()
    d_loss_real = adversarial_loss(discriminator(
            real_imgs[0:half_batch], labels[0:half_batch]),
valid[0:half_batch])
        d_loss_fake = adversarial_loss(discriminator(
            fake_imgs.detach()[0:half_batch], labels[0:half_batch]),
fake[0:half_batch])
        d_loss = d_loss_real + d_loss_fake
   optimizer_D.zero_grad()
        d_loss.backward(retain_graph=True)
        optimizer_D.step()
5，加载生成器并生成图像
```

```
z = torch.normal(
        0., 1., size=(n_samples, latent_dim))
    labels = torch.tensor(randint(0, n_classes, size=n_samples))
    if cuda:
        z.cuda()
        labels.cuda()
```



结论分析：

生成对抗网络 GAN 能够通过训练学习到数据分布，进而生成新的样本。可是 GAN 的缺点是生成的图像是随机的，不能控制生成图像属于何种类别。比如数据集包含飞机、汽车和房屋等类别，原始 GAN 并不能在测试阶段控制输出属于哪一类。为了解决这个问题，人们提出了 conditional Generative Adversarial Network，cGAN 的图像生成过程是可控的。

就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟 1 - 3 道问答题：