# 图形学 PA4 report

计83 何雨泽 2018011351

## 1. 配置环境

windows 10 专业版，使用 Docker 安装，在 jupyter notebook 中打开计图笔记本并进行程序运行。

## 2. 运行代码过程

首先下载 Docker Desktop，设置 CPU 数为 8、内存为 8G，然后在命令行中输入：

```
docker run jittor/jittor python3.7 -m jittor.test.test_example
```

看到正确的训练结果后，继续在命令行中输入：

```
docker run -it -p 8888:8888 jittor/jittor
```

将输出结果中的 http://127.0.0.1:8888/?token=xxxxx 复制到浏览器中打开，新建一个 python3 文件，依次用 jupyter notebook 运行以下代码：

1. 导入包，设置好 hyperparameter：

```
import jittor as jt
from jittor import nn
import numpy as np
import pylab as pl

%matplotlib inline

# 隐空间向量长度
latent_dim = 100
# 类别数量
n_classes = 10
# 图片大小
img_size = 32
# 图片通道数量
channels = 1
# 图片张量的形状
img_shape = (channels, img_size, img_size)
```

2. 定义 CGAN 网络中的生成器 G：

```
class Generator(nn.Module):
    def __init__(self):
        super(Generator, self).__init__()
        self.label_emb = nn.Embedding(n_classes, n_classes)

        def block(in_feat, out_feat, normalize=True):
            layers = [nn.Linear(in_feat, out_feat)]
```

```
            if normalize:
                layers.append(nn.BatchNorm1d(out_feat, 0.8))
            layers.append(nn.LeakyReLU(0.2))
            return layers
        self.model = nn.Sequential(
            *block((latent_dim + n_classes), 128, normalize=False),
            *block(128, 256),
            *block(256, 512),
            *block(512, 1024),
            nn.Linear(1024, int(np.prod(img_shape))),
            nn.Tanh())

    def execute(self, noise, labels):
        gen_input = jt.contrib.concat((self.label_emb(labels), noise),
dim=1)
        img = self.model(gen_input)
        img = img.view((img.shape[0], *img_shape))
        return img
```

3. 定义 CGAN 网络中的判别器 D：

```
class Discriminator(nn.Module):
    def __init__(self):
        super(Discriminator, self).__init__()
        self.label_embedding = nn.Embedding(n_classes, n_classes)
        self.model = nn.Sequential(
            nn.Linear((n_classes + int(np.prod(img_shape))), 512),
            nn.LeakyReLU(0.2),
            nn.Linear(512, 512),
            nn.Dropout(0.4),
            nn.LeakyReLU(0.2),
            nn.Linear(512, 512),
            nn.Dropout(0.4),
            nn.LeakyReLU(0.2),
            nn.Linear(512, 1))

    def execute(self, img, labels):
        d_in = jt.contrib.concat((img.view((img.shape[0], (- 1))),
self.label_embedding(labels)), dim=1)
        validity = self.model(d_in)
        return validity
```

4. 下载预训练模型：

```
!wget https://cg.cs.tsinghua.edu.cn/jittor/assets/build/generator_last.pkl
!wget
https://cg.cs.tsinghua.edu.cn/jittor/assets/build/discriminator_last.pkl
```

5. 生成学号数字串：

```
# 定义模型
generator = Generator()
discriminator = Discriminator()
generator.eval()
discriminator.eval()
```

```
# 加载参数
generator.load('./generator_last.pkl')
discriminator.load('./discriminator_last.pkl')

# 定义一串数字
number = "2018011351"
n_row = len(number)
z = jt.array(np.random.normal(0, 1, (n_row,
latent_dim))).float32().stop_grad()
labels = jt.array(np.array([int(number[num]) for num in
range(n_row)])).float32().stop_grad()
gen_imgs = generator(z,labels)

pl.imshow(gen_imgs.data.transpose((1,2,0,3))[0].reshape((gen_imgs.shape[2],
-1)))
```

## 3. 运行结果