实验报告

单衍喆

2024-9-12

GitHub地址：https://github.com/Venusss1/course.git

目录

1 实验内容

1. **python基础**
2. **pytorch图像处理基础**

2 实验设计

2.1 python基础

2.1.1 python简介

- python是解释型语言：开发过程中没有编译环节
- python是交互式语言：可以再python提示符¿¿¿后直接执行代码
- python是面向对象语言：支持面向对象的风格和代码封装
- python对初学者友好：便于学习且应用广泛

2.1.2 python环境配置

1. 下载安装python

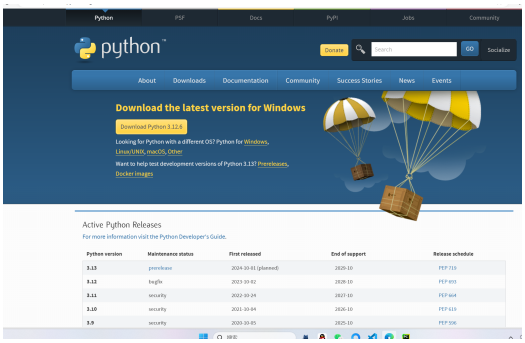

图 1: python官网

2. 配置环境变量
3. 安装pycharm

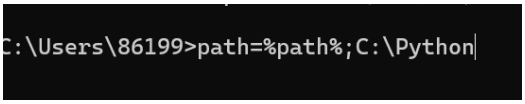

图 2: 环境变量配置

2.1.3 python与c的语法差异

表 1: 语法差异

差异	python	
缩进	用缩进来写模块	使用大括号 {} 来
结束符	一般以新行作为语句的结束符	使用';'作为语句结
引号	可以使用引号(')、双引号(")、三引号("' 或 """) 来表示字符串	单引号和双引号

2.1.4 循环

python提供了for循环和while循环（没有do...while循环）

```
for epoch in range(epochs):

    net.train()
    running_loss = 0.0
    train_bar = tqdm(train_loader, file=sys.stdout)

    for step, data in enumerate(train_bar):
        images, labels = data
        optimizer.zero_grad()
        logits = net(images.to(device))
        loss = loss_function(logits, labels.to(device))
        loss.backward()
        optimizer.step()

        running_loss +=loss.item()
        train_bar.desc = "train epoch[{}/{} loss:{:3f}".format(epoch+1, epochs, loss)
```

图 3: for循环

2.1.5 数据类型

python主要的数据类型：字符串，列表，元组，字典

```
data_transform = {
    "train": transforms.Compose([transforms.RandomResizedCrop(224),
                                 transforms.RandomHorizontalFlip(),
                                 transforms.ToTensor(),
                                 transforms.Normalize([0.485, 0.456, 0.406], [0.229 ,0.2
    "val": transforms.Compose([transforms.Resize(256),
                               transforms.CenterCrop(224),
                               transforms.ToTensor(),
                               transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224
}
```

图 4: 字典（键值对）

2.2 pytorch视觉应用

2.2.1 PIL:python图像处理类库

PIL提供通用图像处理功能：缩放，裁剪，旋转，颜色转换

- 读取图像：

```
from PIL import Image
pil_im = Image.open('empire.jpg')
```

- 颜色转换

```
pil_im = Image.open('empire.jpg').convert('L')
```

- 图像保存

```
Image.open(infile).save(outfile)
```

- 创建缩略图

```
pil_im.thumbnail((128,128))
```

- 复制和粘贴图像区域

```
box = (100,100,400,400)
region = pil_im.crop(box)
pil_im.paste(region,box)
```

- 调整尺寸

```
pil_im.resize((128,128))
```

- 旋转

```
pil_im.rotate(45)
```

2.2.2 Matplotlib

- 绘制图像、点和线

  1. 导入模块

```
from pylab import *
x = [100,100,400,400]
```

  2. 绘制点

```
plot(x,y,'r*')
```

  3. 绘制线

```
plot(x[:2],y[:2])
```

  4. 显示图像

```
imshow(im)
```

- 图像轮廓和直方图

图像轮廓

```
figure()
gray()
contour(im,origin='image')
```

直方图

```
figure()
hist(im.flatten(),128)
show()
```

- 交互

```
x = ginput(3)
```

2.2.3 NumPy

NumPy帮助实现数组中的操作

- 图像数组表示

```
im = array(Image.open('empire.jpg').convert('L','f'))
```

- 灰度变换

```
im2 = 255 - im
im3 =(100.0/255)*im + 100
im4 = 255.0*(im/255.0)**2
```

- 图像缩放

```
def imresize(im,sz):
    pil_im = Image.fromarray(uint8(im))
    return array(pil_im.resize(sz))
```

- 主成分分析

```
def pca(X):
   num_data,dim = X.shape
   mean_X = X.mean(axis=0)
   X = X - mean_X
   ...
   return V,S,mean_X
```

2.2.4 SciPy

SciPy建立于NumPy基础上，用于数值计算

- 图像模糊

使用scipy.ndimage.filters模块计算卷积

```
from scipy.ndimage import filters
im2 = filters.gaussian_filter(im,5)
```

模糊彩色图像只需对每一个颜色通道进行高斯模糊

- 图像导数

sobel导数滤波器

```
filters.sobel(im,2,imx)
magnitude = sqrt(imx**2+imy**2)
```

3 实验心得

通过本次实验，我学会了python的基础语法，安装了pycharm并在本地配置完成了python环境我学会了pytorch处理图像的基本方法，了解了pytorch处理图像的常用库及对应方法