23 春 Python 与深度学习基础学习笔记

学生姓名：杨益

学号： PB21000308

课程名称：23 春 Python 与深度学习基础

更新至：Lec9

中国科学技术大学

May 7, 2023

1 python 基础知识

Lec1 导言

- 循环，rang(start,end,step), 注意 range 为左闭右开, enumerate

```
enumerate 的使用
>>> seasons = ['Spring', 'Summer', 'Fall', 'Winter']
>>> list(enumerate(seasons))
[(0, 'Spring'), (1, 'Summer'), (2, 'Fall'), (3, 'Winter')]
```

Lec2 python 基本语法

```
生成式方案
for val in data1:
    data.append(val)

data = [val for val in data1]

data = [val for val in data1 if val%2 == 0]
```

- 生成式:data=[1,2,3] 与生成器 data=(1,2,3)，生成器不会占用过多内存，可以直接使用 for 循环
- 可以使用.format() 对字符串进行格式化替换
- 类（Class ClassName), 有 init 和其他类, 可以直接在 init 中调用父类, father name.init 调用，并且可以重写函数覆盖父类的方法, 当没有表明父类，默认集成 object 类
- 异常处理，try：，except ZeroDivisonError as e：...
- 文件读写 f = open(”text.txt”,”r”),f.close(), 也可以用 with open(”text.txt”,”r”) as f, 可以用 csv 读写处理比较方便，文件操作完使用 f.close c 才能确保所做的修改确实保存到了文件中，也可以 save 到其他目录中

```
str.format()使用方法，使用{}作为分隔符
#named indexes:
txt1 = "My name is {fname}, I'm {age}".format(fname = "John", age = 36)
#numbered indexes:
txt2 = "My name is {0}, I'm {1}".format("John",36)
#empty placeholders:
txt3 = "My name is {}, I'm {}".format("John",36)
```

Lec3 字符串和爬虫

- 修饰器（decorator），接受其他函数作为参数并且对其进行改造并且返回新函数，利用 @before,def wrapper（*args，*kwargs）
- requests and beautifulsoup

Lec4 Numpy

- python 中 list 类元素可以是任意对象，浪费了 cpu 时间和内存，numpy 以 ndarray 存储单一数据类型的多维数组，并以 ufunc 函数进行处理
- ndarray 切片的数据共享一块数据存储空间，区分需要.copy
- 随机数,rand,randint,randn,choice,normal,uniform,poission,shuffle

2 神经网络

Lec5：机器学习入门

- 学习（观察，学习，技能）；机器学习（数据，机器学习，提高某种指标）
- 通过算法使得机器能从样本数据中学习规律从而对新的样本做决策
- 机器学习类型（分类模型预测离散值，回归模型预测连续值）；监督，非监督，半监督
- KNN(K-Nearest Neighbors), 线性分类，最小二乘法
- 不使用 MSE，交叉熵（最大似然估计解读）
- 过拟合（模型拟合了样本中的噪声）；正则化，惩罚模型复杂度
- SVM 和核方法，将原始向量从低维映射到高维是一个解决非线性可分性的重要手段

Lec6：机器学习基本知识

- 机器学习三要素：模型，学习准则，优化方法
- 机器学习：通过数据 D，计算得出一个近似目标函数 f 的假设 g
- 通用近似定理：利用线性输出层和具有单调增、有界连续函数性质的激活函数，但层数足够宽，近似函数可以与原函数任意接近
- 训练方法：反向传播法（利用链式法则进行梯度求导）

模型和特点

- 全连接网络：参数量很大（内存，计算量很大，训练相对困难），且当网络层数加深，梯度容易消失
- 图卷积神经网络：利用图片信息的结构化信息，用卷积核进行模型学习。降低维度方法：权值共享
- ResNet，单纯堆叠网络会造成比较大的训练和测试误差，网络更难以训练 (Shortcut Connections)
- AIGC 网络：RNN，LSTM，transformer

Lec7：深度学习框架

- Pytorch: Tensors and Dynamic neural networks in Python with strong GPU acceleration
- 数值数据存储在张量中，框架定义了张量的函数，函数的导数可以自动求得，导数用于优化模型参数

```
<class "torch.Tensor">
torch.tensor()
torch.from_numpy()
torch.zeros()/rand()/ones()
torch.zeros_like()/rand_like()/one_like: 生成一个值全为0的、维度与输入尺寸相同的矩阵
```

- torch.nn.functional, 包含常见的数学运算，数据的维度 [N,C,W,H], 包含基本的神经网络运算（卷积，池化，激活函数，损失等）
- torch.autograd, 调用 Tensor.backward 计算梯度，求导部分指定 requires grad = True, 求出的导数放在 Tensor.grad 中，也可以通过指定 torch.no grad 来固定某部分导数
- torch.nn.module, 需要重载 init 和 forward 方法，可以调用 nn.functional 的计算函数和参数，并且可以直接调用 Module.parameters() 获得所有的参数，使用 nn.sequential 直接组合流水式的神经网络
- 对梯度下降法的改良（使用更多历史梯度信息，确定学习率），SGD，Adam 等
- 优化器，torch.optim，使用 Optimizer.step() 进行梯度更新，请可以使用 torch.lr-schedule 控制学习率的更变
- 数据集 torch.utils.data，使用 torch.utils.data.Dataset 和 torch.utils.dataloader 指定数据集和数据加载方式，可以读取 minibatch 数据，进行 shuffle 处理

GPU 基本概念与分布式训练

- GPU: Graphical Processing Unit
- Tensor 中包含 device 属性，a = torch.tensor([1,1],device='cuda'), 可以使用 Tensor.to 和 Module.to 进行转换
- 有多个 GPU 时，可以指定 device="cuda:X"