第三天

2019年6月9日 9:22

pytorch中数据加载

batch：数据打乱顺序，组成一波一波的数据，批处理

epoch：拿所有的数据训练一次

Dataset基类，数据集类

1. torch.utils.data.Dataset
2. 两个重要的方法：
    a. __getitem__(index) :能够对实例进行索引
    b. __len__: len(实例) 调用实例的__len__方法

迭代数据集

1. torch.utils.data.DataLoader（dataset，batch_size,shuffle）

手写数字识别的思路：

1. 准备数据，通过dataset和DataLoader准备
2. 模型构建
3. 模型训练，模型保存和加载
4. 模型的评估

a. 准备Mnist数据
    a. torchvision.transforms.ToTensor
        i. 把ndarray转化为tensor
        ii. PIL中iamge对象对转化为tensor
    b. torchvision.transforms.Normlize(mean,str)
        i. mean，std的形状和通道数相同
    c. torchvision.transforms.Compose
        i. 把不同的实例组合使用
b. 交叉熵损失
    a. nn.CrossEntropyLoss()
    b. 使用带权损失计算交叉熵损失
        i. softmax(out)
        ii. $$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_K}}, j = 1 \cdots k$$

iii. output = F.log_softmax(out) #log(P)
iv. F.nll_loss(output,target) #$-\sum Y\log(p)$

c. 带权损失
i. $loss = -\sum w_i x_i$

c. 训练
a. 遍历dataloader
b. tqdm(可迭代对象,total=迭代总次数)

d. 模型的评估
a. 不需要计算梯度
b. 计算损失和准确率
c. 准确率的计算
i. 获取概率最大值的位置作为预测值
ii. 预测值和真实值判断相等，结果取均值

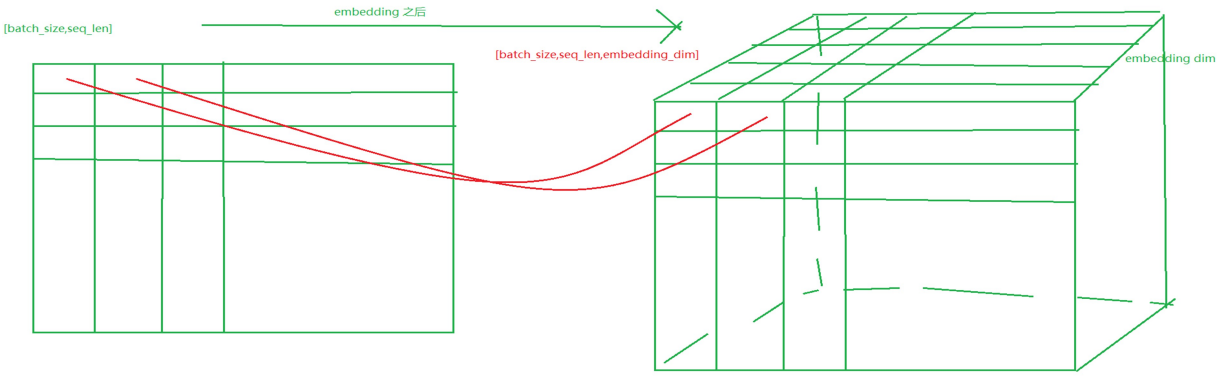
文本分词

N-gram：用连续的N个token左右一个特征，N往往取2或者3。

1. 考虑了句子中词语的顺序

文本向量化

1. one-hot
2. word emebdding:
   a. 用一个向量表示每一个词语，向量中的每个值都是参数，都会在后续通过训练得到
   b.



作业

构造数据（有三个特征值，一个目标值），构造模型进行训练，进行模型评估