HW2 MNIST手写体数字识别

1 pytorch环境安装（10）

pytorch是一个非常常见的深度学习框架

对于已经配置好conda环境的同学：

```
conda install pytorch torchvision -c pytorch
conda install nb_conda_kernels
```

不知道怎么安装的conda的详见README.md

pip安装详见：https://pytorch.org/，根据自己的环境选择合适的安装语句。

pytorch官方文档（推荐！）：https://pytorch.org/docs/stable/index.html

pytorch中文文档：https://pytorch.apachecn.org/#/

一些pytorch常见语句介绍

CONV2D

```
torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0,
dilation=1, groups=1, bias=True, padding_mode='zeros', device=None, dtype=None)
```

二维卷积，返回值为一个函数

- in_channels：输入channel数
- out_channels：输出channel数
- kernel_size：卷积核的大小，一般使用3比较多，配合上stride=1和padding=1可以保证卷积前后尺寸不变

LINEAR

```
torch.nn.Linear(in_features, out_features, bias=True, device=None, dtype=None)
```

即线性层、全连接层，返回值是一个函数

- in_features：输入特征维度

- out_features：输出特征维度

CROSSENTROPYLOSS

```
torch.nn.CrossEntropyLoss(weight=None, size_average=None, ignore_index=-100,
reduce=None, reduction='mean')
```

即交叉熵损失函数，返回值为一个函数

```
usage:
CE = CrossEntropyLoss()
loss = CE(output, label)
```

- output：torch.tensor类型，shape为（batch_size, 类别数量）
- label：torch.tensor类型，shape为（batch_size, ），存储的是类别的index（非one-hot）

SGD

```
torch.optim.SGD(params, lr=<required parameter>, momentum=0, dampening=0,
weight_decay=0, nesterov=False)
```

随机梯度下降优化器

- params：模型参数
- lr：学习率
- momentum：动量
- weight_decay：正则化参数

2 MNIST识别（90）

2.1 输入数据集 （10）

data文件夹中是MNIST分类数据集，训练数据集包含 60000 个样本, 测试数据集包含 10000 样本。每个样本为单通道图片，长宽均为28。

2.2 数据集可视化（20）

使用matplotlib工具包将数据集可视化。

2.3 模型建立（20）

- 利用卷积层和线性层建立一个深度模型
- 如果熟悉的同学，可以尝试其他层，例如dropout、batchnorm、layernorm、maxpooling等，具体的函数接口可以在官方文档中找到

2.4 模型训练（20）

- 训练3个epoch，观察loss的变化
- 将loss可视化出来

2.5 性能测试（20）

- 将训练好的模型在测试集上进行测试，观察准确率

3 后言

据许多小伙伴反应，第一次的作业好像比较难。那么这一次作业对你们来说应该更是一个挑战。但MNIST任务属于深度学习中的”hello world”，是每个入门深度学习必然需要经历的项目。

希望大家可以懂得查阅官方文档，这是程序员必备的基本功。

然后有问题的小伙伴可以积极在issue上进行提问，我也希望能把issue区当成大家讨论作业讨论代码的一个社区，要学会借助社区的力量，你遇到的问题必然别人也会遇到，大家就可以在社区讨论解决，我如果有时间的话也会回复一下提问的小伙伴。

我会在issue上开一个帖子叫【mnist benchmark】mnist benchmark · Issue #2 · mousecpn/MachineLearning_HW · GitHub，大家可以把自己的测试集性能发上来，并分享自己的模型训练心得（模型结构、优化器参数等等）