深度学习训练营

案例 2：构建自己的多层感知机: MNIST 手写数字识别

1 数据集简介

MNIST 手写数字识别数据集是图像分类领域最常用的数据集之一，它包含 60,000 张训练图片，10,000 张测试图片，图片中的数字均被缩放到同一尺寸且置于图像中央，图片大小为 28×28。MNIST 数据集中的每个样本都是一个大小为 784×1 的矩阵(从 28×28 转换得到)。MNIST 数据集中的数字包括 0 到 9 共 10 类，如下图所示。注意，任何关于测试集的信息都不该被引入训练过程。

在本次案例中，我们将构建多层感知机来完成 MNIST 手写数字识别。

2 构建多层感知机

本次案例提供了若干初始代码，可基于初始代码完成案例，各文件简介如下：（运行初始代码之前请自行安装 TensorFlow 2.0 及以上版本，仅用于处理数据集，禁止直接调用 TensorFlow 函数）

- mlp.ipynb 包含了本案例的主要内容，运行文件需安装 Jupyter Noterbook.
- network.py 定义了网络，包括其前向和后向计算。
- optimizer.py 定义了随机梯度下降(SGD)，用于完成反向传播和参数更新。
- solver.py 定义了训练和测试过程需要用到的函数。
- plot.py 用来绘制损失函数和准确率的曲线图。

此外，在/criterion/和/layers/路径下使用模块化的思路定义了多个层，其中每个层均包含三个函数：__init__用来定义和初始化一些变量，forward 和 backward 函数分别用来完成前向和后向计算：

- FCLayer 为全连接层，输入为一组向量（必要时需要改变输入尺寸以满足要求），与权重矩阵作矩阵乘法并加上偏置项，得到输出向量：$\boldsymbol{u} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$.

- SigmoidLayer 为 sigmoid 激活层，根据$f(\boldsymbol{u}) = \frac{1}{1+\exp(-\boldsymbol{u})}$计算输出。
- ReLULayer 为 ReLU 激活层，根据$f(\boldsymbol{u}) = \max(\mathbf{0}, \mathbf{u})$ 计算输出。
- EuclideanLossLayer 为欧式距离损失层，计算各样本误差的平方和得到：$\frac{1}{2}\sum_n \|logits(n) - gt(n)\|_2^2$。
- SoftmaxCrossEntropyLossLayer 可以看成是输入到如下概率分布的映射：

$$P(t_k = 1|\boldsymbol{x}) = \frac{\exp(x_k)}{\sum_{j=1}^{K}\exp(x_j)}$$

其中x_k是输入向量$\boldsymbol{x}$中的第 k 个元素，$P(t_k = 1|\boldsymbol{x})$表示该输入被分到第$k$个类别的概率。由于 softmax 层的输出可以看成一组概率分布，我们可以计算 delta 似然及其对数形式，称为 Cross Entropy 误差函数：

$$E = -\ln p\left(t^{(1)}, \ldots, t^{(N)}\right) = \sum_{n=1}^{N} E^{(n)}$$

其中

$$E^{(n)} = -\sum_{k=1}^{K} t_k^{(n)} \ln h_k^{(n)}$$

$$h_k^{(n)} = P\left(t_k^{(n)} = 1|x^{(n)}\right) = \frac{\exp(x_k^{(n)})}{\sum_{j=1}^{K}\exp(x_j^{(n)})}$$

注意：此处的 softmax 损失层与案例 1 中有所差异，本次案例中的 softmax 层不包含可训练的参数，这些可训练的参数被独立成一个全连接层。

3 案例要求

完成上述文件里的‘#TODO’部分(红色标记的文件)，提交全部代码及一份案例报告，要求如下：

- 记录训练和测试准确率，绘制损失函数和准确率曲线图；
- 比较分别使用 Sigmoid 和 ReLU 激活函数时的结果，可以从收敛情况、准确率等方面比较；
- 比较分别使用欧式距离损失和交叉熵损失时的结果；
- 构造具有两个隐含层的多层感知机，自行选取合适的激活函数和损失函数，与只有一个隐含层的结果相比较；
- 本案例中给定的超参数可能表现不佳，请自行调整超参数尝试取得更好的结果，记录下每组超参数的结果，并作比较和分析。

4 注意事项

- 提交所有代码和一份案例报告；

- 注意程序的运行效率，尽量使用矩阵运算，而不是 for 循环；
- 本案例中不允许直接使用 TensorFlow, Caffe, PyTorch 等深度学习框架；
- 禁止任何形式的抄袭。