手写数字识别实验报告

本实验使用 PyTorch 实现了使用卷积神经网络对 MNIST 手写数字识别数据集进行分类的任务，并分析了学习率的影响。

数据集

MNIST 数据集是一个手写数字识别数据集，它包含 60000 张训练图像和 10000 张测试图像。每个样本都是一张 28×28 像素的灰度手写数字图片，对应了 0~9 十个数字的其中一个。

实验设置

卷积神经网络结构：

Layer (type)	Output Shape	Parameters	Kernels	Kernel Size	Stride	Padding
Conv2d-1	[-1, 16, 28, 28]	416	16	5	1	2
BatchNorm2d-2	[-1, 16, 28, 28]	32				
ReLU-3	[-1, 16, 28, 28]	0				
MaxPool2d-4	[-1, 16, 14, 14]	0		2	0	0
Conv2d-5	[-1, 32, 14, 14]	12,832	32	5	1	2
BatchNorm2d-6	[-1, 32, 14, 14]	64				
ReLU-7	[-1, 32, 14, 14]	0				
MaxPool2d-8	[-1, 32, 7, 7]	0		2	0	0
Linear-9	[-1, 10]	15,690				

超参数设置：

- 初始学习率尝试了三种：$10^{-3}, 5 \times 10^{-4}, 10^{-4}$。
- 优化算法：Adam

$$
\begin{aligned}
m_t &= \beta_1 * m_{t-1} + (1-\beta_1) * \nabla w_t \\
v_t &= \beta_2 * v_{t-1} + (1-\beta_2) * (\nabla w_t)^2 \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t} \qquad \hat{v}_t = \frac{v_t}{1-\beta_2^t} \\
w_{t+1} &= w_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} * \hat{m}_t
\end{aligned}
$$

- 批量大小：128
- 迭代次数：50
- 激活函数：ReLU

$$
\sigma(x) = \begin{cases} max(0,x) & ,x >= 0 \\ 0 & ,x < 0 \end{cases}
$$

代码说明

所有代码均在 `main.py` 文件中。

核心的函数和类：

- `CNN`：实现了一个卷积神经网络
- `train_epoch`：使用训练集对模型参数进行一轮完整更新，并计算准确率和样本平均损失
- `train`：通过调用 `train_epoch` 函数，对模型参数进行 `EPOCH` 轮更新，并保存测试集表现最好的模型参数，绘制训练中的损失曲线
- `evaluate`：对数据集进行预测，计算准确率和平均损失
- `get_loader`：加载 MNIST 数据集
- `main`：函数入口

此外还实现了一些辅助功能：

- `seed_evetything`：固定随机数种子，确保实验可复现
- `args`：命令行参数
- `log`：日志记录
- `show_error_images`：展示预测错误的图片

本实验提交了三个不同学习率模型的训练结果，分别在 `lr_1e-3`，`lr_1e-4`，`lr_5e-4` 三个文件夹中，其中包含：损失曲线 `figure.png`、模型权重 `model.pth`、日志 `model.log`。

运行代码

安装依赖：

```
pip install torch torchvision torchsummary matplotlib tqdm
```

训练模型：（参数均为可选）

```
python main.py --batch_size 128 --epoch 50 --learning_rate 0.001
```

测试模型：（`model_path` 为模型参数文件路径）

```
python main.py --test true --model_path lr_1e-3/model.pth
```

实验结果和分析

一共训练了三个不同学习率（$10^{-3}, 5 \times 10^{-4}, 10^{-4}$）的模型，分别进行完整 50 个 epoch 的训练，选取测试集损失最小的模型作为最终模型，结果如下：

学习率	训练集损失	训练集准确率	测试集损失	测试集准确率	最佳epoch
0.001	0.000088	0.9964	0.000209	0.9918	10
0.0005	0.000101	0.9962	0.000206	0.9916	10
0.0001	0.000035	0.9997	0.000203	0.9907	36

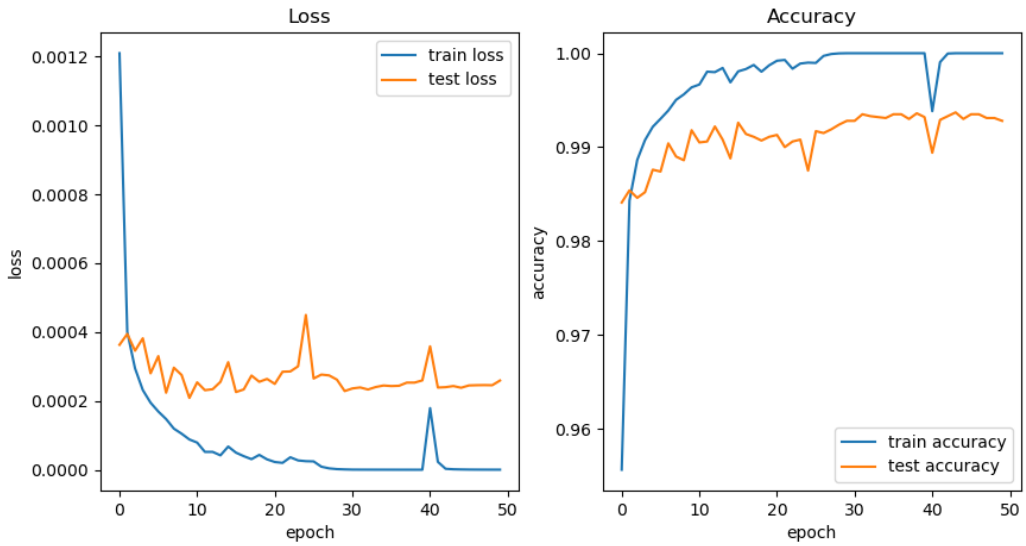
第 50 个 epoch 的结果如下：

学习率	训练集损失	训练集准确率	测试集损失	测试集准确率
0.001	0.000000	1.0000	0.000259	0.9928
0.0005	0.000059	0.9978	0.000539	0.9863
0.0001	0.000013	0.9999	0.000229	0.9903

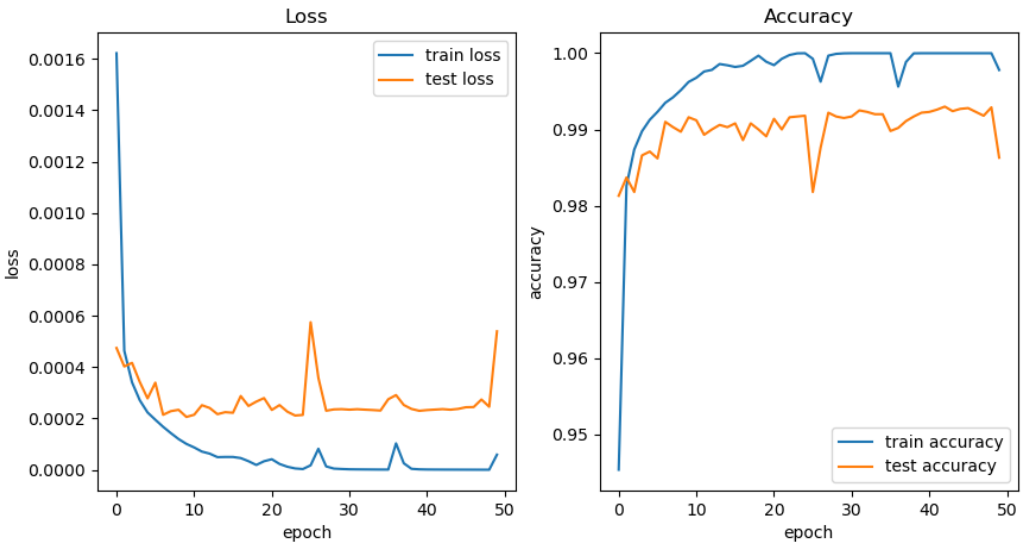
分析：

- 三个模型均很好地拟合了训练集和测试集，训练集准确率均高于 99.60%，测试集准确率均高于 99%。
- 初始学习率越低，模型拟合训练集的速度越慢，过拟合的时间也更晚。
- 由于 Adam 算法拥有自适应学习率的能力，数据集也较为简单，降低初始学习率并没有使得模型的过拟合更加严重。

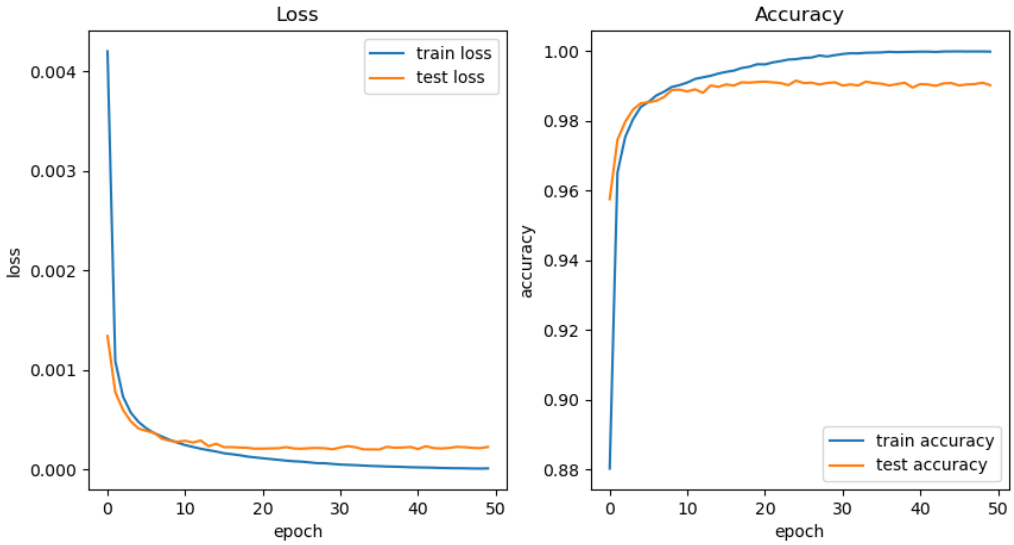
学习率为 10^{-3} 的模型的训练曲线：



学习率为 5×10^{-4} 的模型的训练曲线：



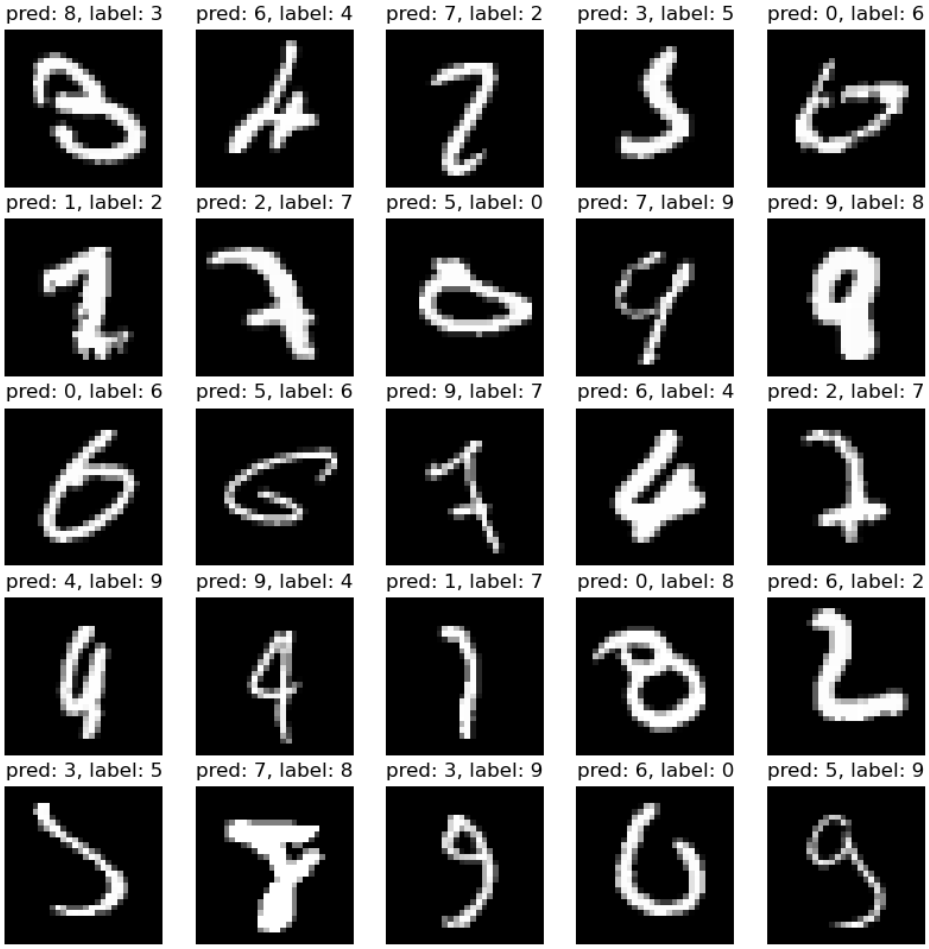
学习率为 10^{-4} 的模型的训练曲线：



分析：

- 初始学习率越低，训练中的参数更新就越稳定，损失曲线也越平滑。

使用学习率为 10^{-3} 的模型进行测试，一共有 82 张图片分类错误，其中的 25 张：



其中部分样本是比较简单的，例如第 3 行第 1 列，第 5 行第 3 列，说明模型还是有部分参数优化不够好；另外大部分都比较困难，例如第 1 行第 4 列，第 3 行第 3 列。