# 实 验 报 告

## 实验名称： SoftMax回归

## 课程名称： 机器学习实验

院系： 人工智能学院

班级： 2021级机器学习A班

姓名： 张宇鹏

学号： 21311071

日期： 2024年 5月1日

**中山大学**

1. 实验内容

本次实验的主要任务是**了解并掌握训练softmax回归做多类别分类的流程**，学习两个机器学习库pytorch和torchvision的基本使用。

softmax回归是将**回归应用到分类任务**中的一种方法，线性回归模型适用于输出值为连续值的情况，在我们需要的输出为离散值的情况，我们不关心值具体是多少，只关心的是哪个的类别值更大，更有可能是哪一类的。对于softmax回归，根据数据有多少个特征，就能确定有多少个输入，而有多少个类别，模型也就能确定有多少个输出。本次实验，我将基于Fashion-MNIST数据集来**实现softmax回归，并将最终结果以混淆矩阵的形式可视化。**

1. 实验环境

操作系统： Windows 11 Version 23H2

Python 版本： Python 3.11.4 ('base':conda)

1. 实验步骤

3.1 实现softmax的基本流程

首先需要读取数据集中的数据，并将**数据统一转化为张量形式**，通过load\_data模块加载Fashion-MNIST数据集的训练和测试数据。定义softmax模型类，继承自nn.Moudle，在前向传播中**只包含一个线性层**用于将输入特征映射到输出类别，这是因为softmax回归模型的定义。

softmax回归定义了模型的输出值，模型的特征，模型的权重系数以及偏置系数。

根据上面的式子，再使用softmax函数将输出的离散值归一化为概率值：

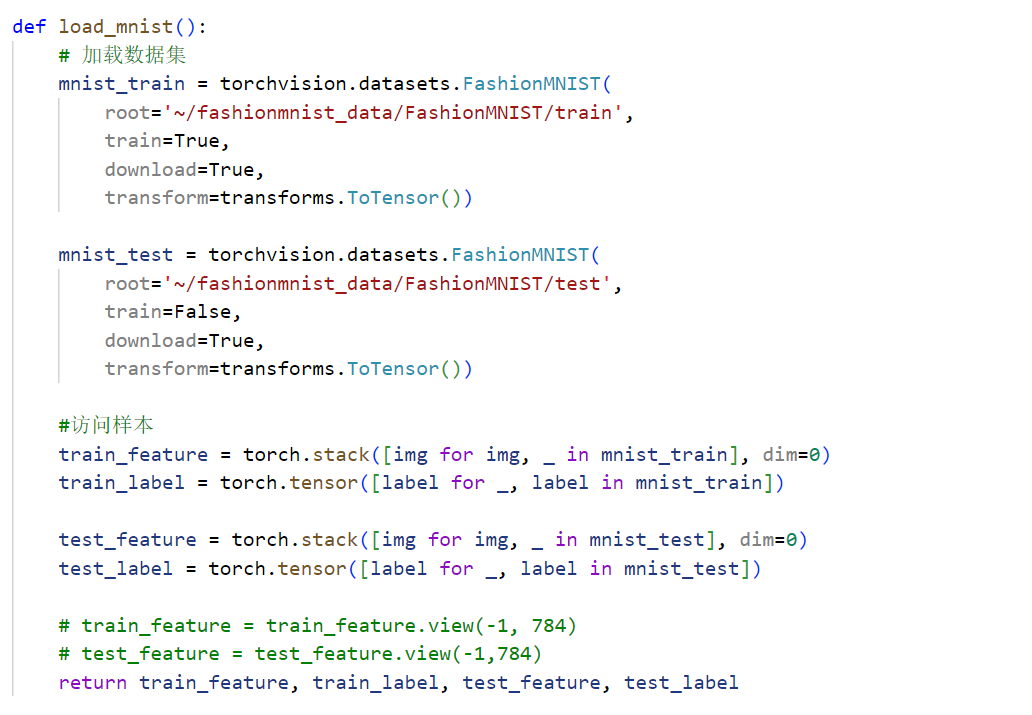
其中，是由sigmoid函数得到的。

定义训练函数train，使用**小批量梯度下降优化器**和**交叉熵损失函数**来训练模型，并在每个训练步骤中打印损失值。小批量梯度下降优化器是在随机梯度下降优化器(SGD)的基础上，由数据加载器实现的，数据加载器用于将数据分批，再统一输入到SGD中。

训练完成后，模型参数被保存到一个检查点文件中。然后定义测试函数test，使用模型对测试数据进行预测，并计算分类准确率。

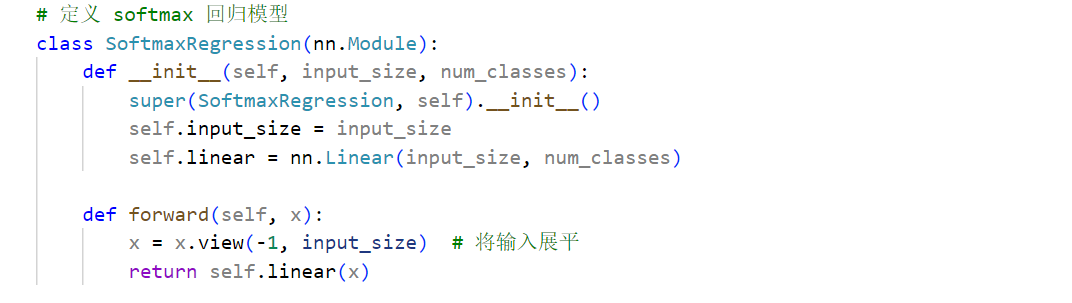
3.2 数据加载

函数load\_mnist用于加载Fashion-MNIST数据集，它自动下载并处理训练集和测试集的数据，将图像转换为Tensor格式，并提取出图像特征和对应的标签。通过调用这个函数，我们可以获取Fashion-MNIST数据集的特征和标签的Tensor，用于后续的模型训练和测试。



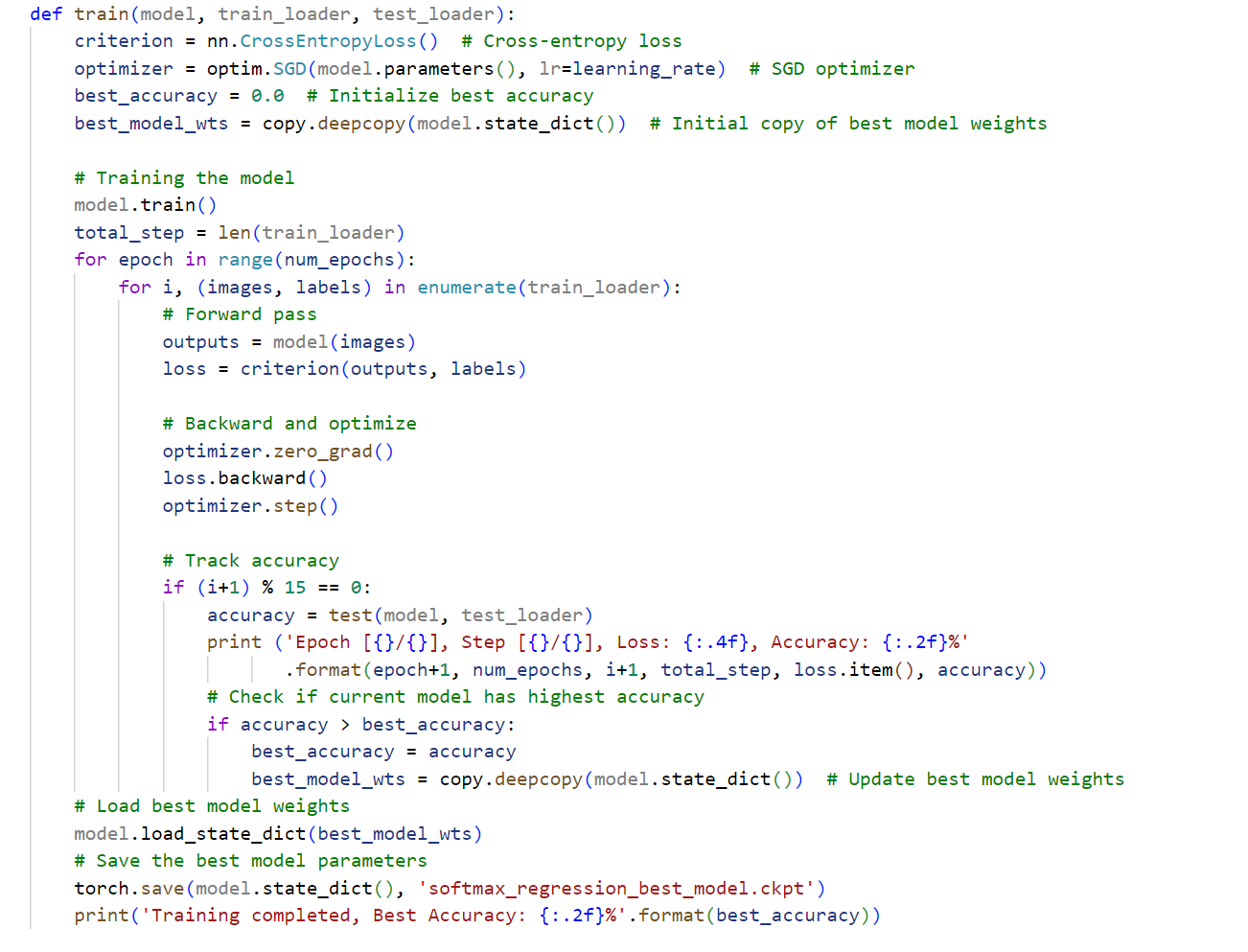
3.3 模型定义

接下来定义softmax模型类SoftmaxRegression，继承自nn.Moudle，在前向传播中**只包含一个线性层**用于将输入特征映射到输出类别。为了处理维度为(1×28×28)的Tensor图像信息，这里我还做了一个重要的步骤，就是将图像的输入展平，展平后的维度为(1×784)，这是将图像数据读入模型的一种常用操作。值得注意的是，我们没有添加softmax层，这是因为在 Softmax 回归模型中，通常不需要显式地添加 Softmax 层，因为在交叉熵损失函数中已经包含了Softmax操作。交叉熵损失函数会在模型的最后一层输出进行 Softmax 操作，然后计算预测值与真实标签之间的交叉熵损失，可以直接返回线性输出。



3.3 模型训练

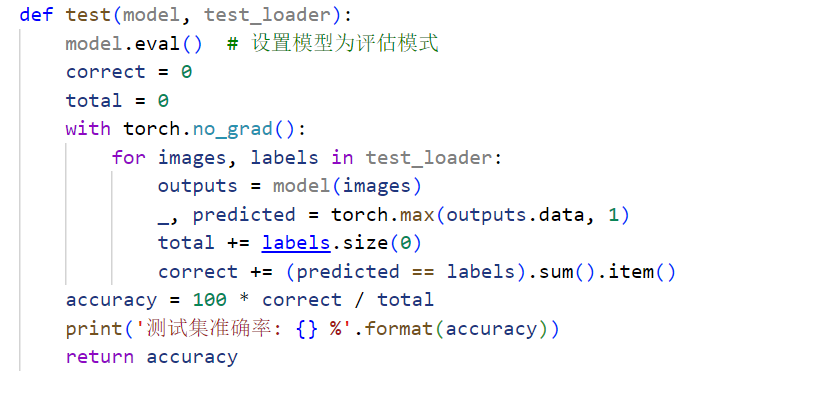
设计一个用于训练softmax回归模型的函数，接受一个模型(model)和一个训练数据加载器，也就是刚刚分批处理的数据集(train\_loader)作为输入，同时**使用交叉熵损失函数和小批量梯度下降优化器(SGD)来训练模型**，并记录训练过程中的最小损失值及对应的模型参数。



在每次迭代中，执行前向传播计算损失，然后执行反向传播和优化步骤来更新模型参数。此外，我们选择**保存了训练过程中具有最小损失的模型参数**，并在训练完成后将其加载到模型中，最后**保存的是损失值最小的模型的参数**。

3.4 模型测试

设计测试函数用于评估神经网络模型在测试集上的性能，首先通过调用model.eval()将模型设置为**评估模式**，确保评估时表现正确。然后使用torch.no\_grad()上下文管理器来禁止梯度计算，从而节省内存和计算资源。在循环中，函数迭代测试数据加载器中的每一批图像和标签，将图像输入模型得到预测输出，并使用torch.max函数找到预测类别。通过计算预测正确的样本数并累加总样本数，最后计算并打印出测试集的准确率。



3.5 数据可视化

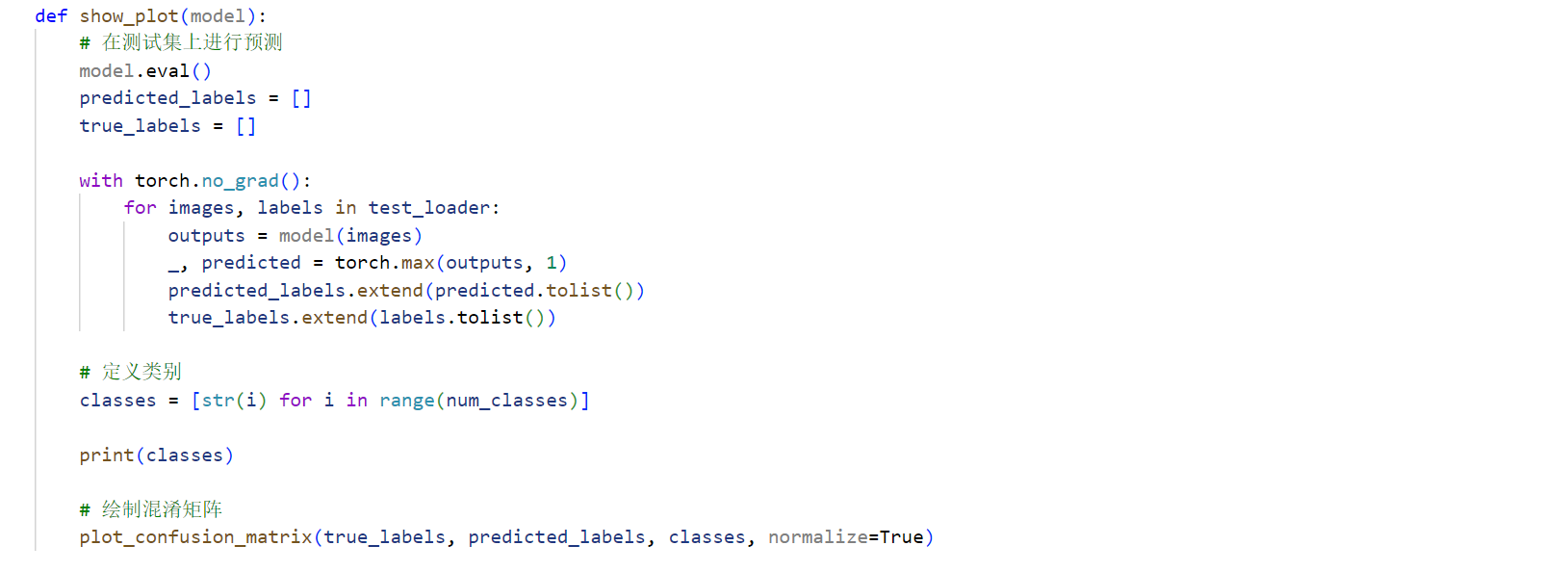
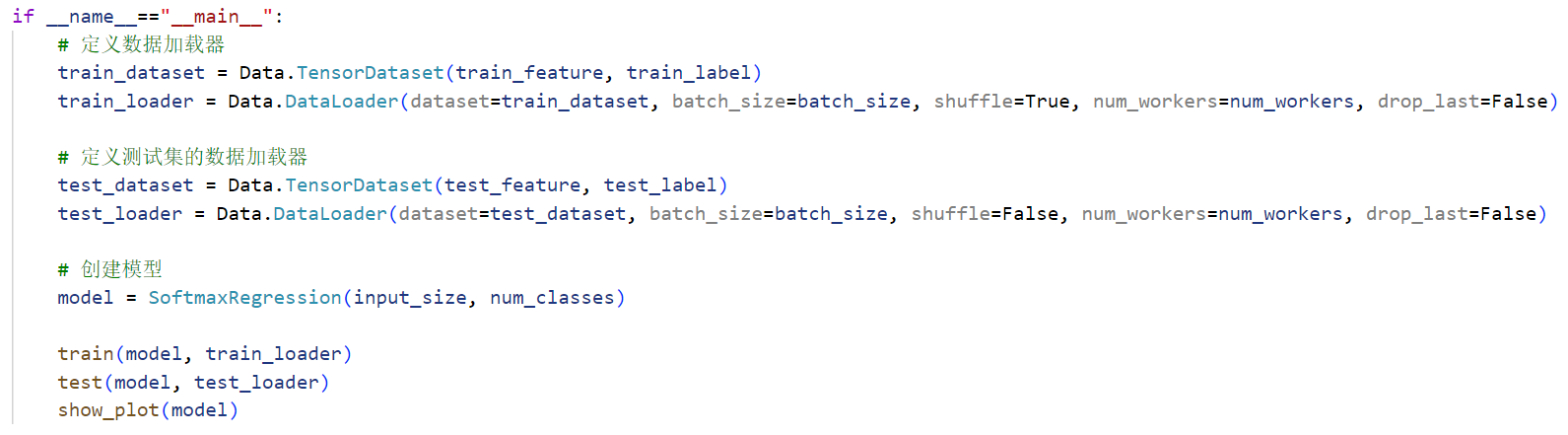
设计函数用于计算并绘制混淆矩阵，根据预测结果和真实标签，以热力图的形式展示模型的分类性能。根据给定的真实标签和预测标签计算混淆矩阵 cm。然后，根据参数normalize决定是否对混淆矩阵进行归一化。之后只需要利用Seaborn绘制热力图，并在每个单元格中添加了概率值或原始计数，同时设置标题、标签和字号，最后显示绘制好的混淆矩阵。

show\_plot函数通过给定模型 model 对测试集进行预测，并将预测结果和真实标签用于绘制混淆矩阵。首先将模型设置为评估模式，然后对测试集进行迭代，获取预测结果和真实标签，并将它们传递给 plot\_confusion\_matrix 函数进行绘制。最终，该函数展示了模型在测试集上的分类情况。两个函数结合，提供了一种直观地评估模型性能的方法，通过可视化展示模型对每个类别的分类准确性和误差情况。

3.6 主函数调用

在主函数中调用上面提到的五个函数。

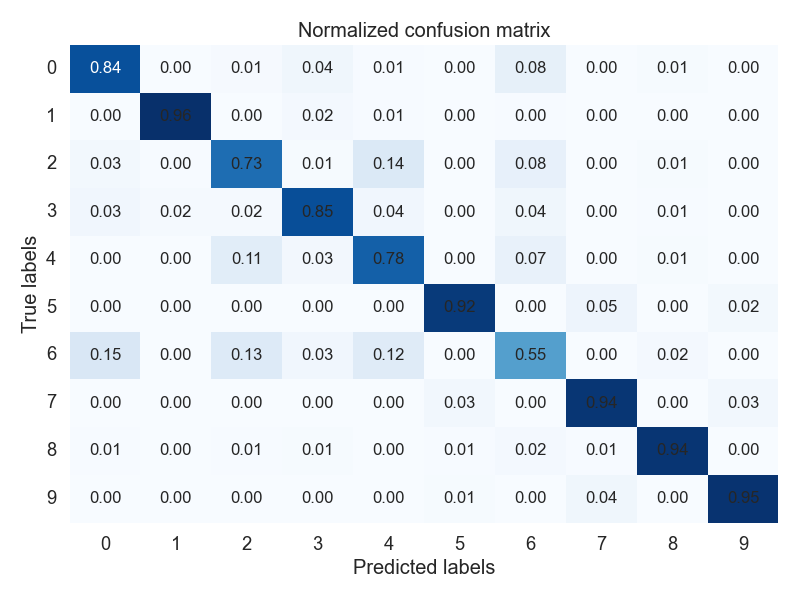


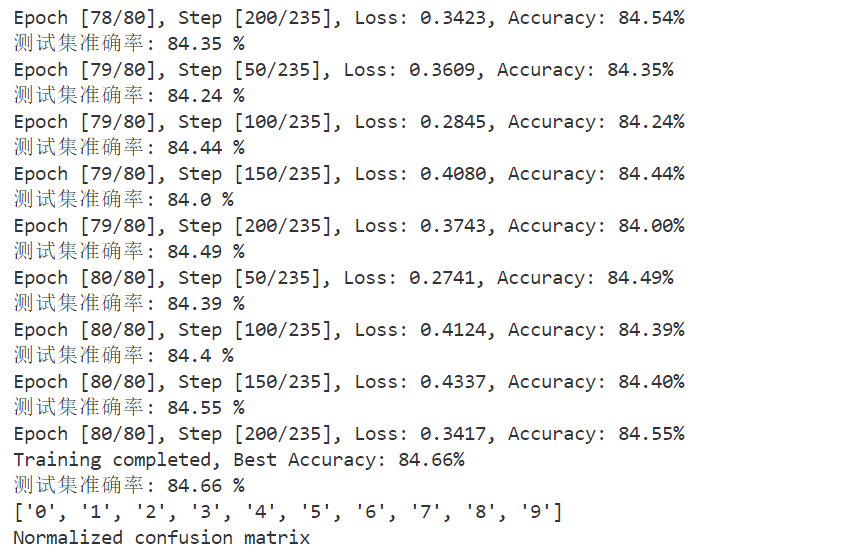
1. 实验结果与分析



将实验的结果**用混淆矩阵可视化**，其中每一行代表了图像的真实标签，每一列代表了图像的预测标签，而混淆矩阵内的数字代表了对应真实结果和预测结果的概率，**概率越高颜色越深，可以发现在对角线上的颜色最深**，也就是说模型是**比较有效**的，大部分情况下可以分辨出图像的类别，通过对模型超参数的一些调整，总的**正确率维持在84%~86%之间**。



下面展示的是结果的正确率大小和训练时的损失值变化，正确率会在一定程度上变化，基本稳定在84%~86%之间，最小损失值基本在0.25~0.30之间，这里是每100步打印一次损失值。输出信息保存到result/epoch.txt中，可以进行查看，下面展示部分epoch的截图。



由于没有办法把所有的训练步的损失都打印出来，所以我每100步打印一次损失值，而我最后保存的是损失值最小的那个模型，用这个模型对测试集进行测试。实验结果表明，模型在测试集上的准确率稳定在84%到86%之间，展示的模型在测试集的正确率是84.66%，损失值维持在0.25到0.30之间。通过绘制混淆矩阵，我们可以观察到模型在对大部分类别的图像进行分类时表现良好，对角线上的概率值较高，颜色深度最深，这意味着模型能够有效地识别出图像的类别。然而，在类别标签为6的时候仍然存在一些混淆，模型对于某些类别的分类性能相对较差，这是值得注意的点。**未来可以考虑继续对模型的超参数进行修改，或者使用刚学习的特征工程技术对特征进行优化设计，或许可以得到更高的正确率**。

1. 实验总结

本次实验的主要目的是让我们**了解并掌握使 softmax回归进行多类别分类的流程**，同时学习PyTorch和torchvision两个机器学习库的基本使用。通过基于Fashion-MNIST数据集实现softmax回归，我成功地构建了一个能够识别不同物品类别的模型，并将结果以混淆矩阵的形式进行了可视化。

实验过程中，我观察到了softmax回归模型在处理多类别分类问题时**表现出了较好的性能**。通过学习Fashion-MNIST数据集中的图像特征，能够较为准确地识别出不同类别的时尚商品。通过混淆矩阵的可视化，我们发现模型在大部分类别的识别上表现良好，对角线上的颜色较深，说明模型能够有效地识别出正确的类别。

然而，实验结果也暴露出了一些问题。尽管模型的正确率稳定在84%到86%之间，但**在某些类别上的分类性能仍然有待提升**。特别是类别标签为6的时尚商品，模型在该类别上表现出了一定的混淆，这可能是由于该类别与其他类别在特征上较为相似，导致模型难以准确区分。

针对这些问题，我觉得或许还可以考虑采取以下的措施进行改进：

①**调整模型超参数**：通过调整学习率、批次大小等超参数，可以进一步优化模型的训练过程，提高模型的分类性能。

②**数据增强**：对训练数据进行增强，如旋转、缩放、裁剪等操作，可以增加模型的泛化能力，提高模型对不同角度、不同尺度的图像的识别能力，其实属于特征工程的范畴。

③**或者使用更复杂的模型结构**：可以尝试使用更复杂的神经网络结构，来提取图像中的更高级特征，比如卷积神经网络，进一步提高模型的分类性能。

通过本次实验，我们不仅掌握了softmax回归的基本原理和实现方法，还学习了PyTorch和torchvision库的基本使用，为后续在深度学习领域的深入研究奠定了坚实的基础。同时，实验结果也为我提供了改进模型的思路和方向，为未来的研究提供了有益的参考。