

人工智能实验报告

姓名: 崔敬然 学号: 23336055

一. 实验题目

中药图片分类任务

二. 实验内容

1. 算法原理

本实验使用PyTorch框架构建卷积神经网络（CNN）实现中药图片分类，数据集包含五类中药：**baihe**、**dangshen**、**gouqi**、**huaihua**、**jinyinhua**，分为训练集和测试集。训练集用于模型训练，测试集仅用于性能评估。模型从零开始训练，未使用预训练模型。

CNN架构: 设计了一个名为**TCMNet**的CNN，包含三组卷积块，每组包括两个**Conv2d**层（带ReLU激活和批量归一化）、最大池化层和**Dropout**（概率0.25）。卷积核大小为3x3，特征图通道数依次为32、64、128。池化层将空间维度减半，最终特征图大小为28x28x128。分类器部分包括全连接层（**Linear**），从128x28x28降维到512，再到5类，中间加入**Dropout**（概率0.5）防止过拟合。模型输出为五类别的对数概率。

前向传播: 输入图像**x**（大小3x224x224）通过卷积块提取特征，展平后通过分类器输出类别分数。数学表示为：
$$h = \text{ConvBlocks}(x), \quad y = \text{Linear}(\text{Flatten}(h))$$
 其中，**ConvBlocks**表示卷积、激活、池化等操作，**y**为输出logits。

损失函数: 使用交叉熵损失，结合**Softmax**计算预测概率与真实标签的差异。对于样本**(x_i, y_i)**，损失定义为：
$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \log \left(\frac{\exp(y_i[\hat{y}_i])}{\sum_{k=1}^5 \exp(y_i[k])} \right)$$
 其中，**N**为批量大小，**y_i**为模型输出， **\hat{y}_i** 为真实类别。

优化: 使用Adam优化器，学习率为**0.001**。为提高收敛性，采用**ReduceLROnPlateau**调度器，当验证损失5个epoch无下降时，学习率乘以0.1。优化目标为：
$$\theta \rightarrow \theta - \eta \nabla_{\theta} \mathcal{L}$$
 其中， **θ** 为模型参数， **η** 为学习率。

数据处理: 训练集应用数据增强（随机翻转、旋转、颜色抖动）以提高泛化能力，测试集仅进行归一化和大小调整（224x224）。所有图像归一化为均值[**0.485, 0.456, 0.406**]，标准差[**0.229, 0.224, 0.225**]。

2. 关键代码展示

以下为**1.py**中的核心代码片段，展示了**TCMNet**模型定义和训练循环:

```
class TCMNet(nn.Module):
    def __init__(self, num_classes):
        super(TCMNet, self).__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 32, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.BatchNorm2d(32),
            nn.Conv2d(32, 32, kernel_size=3, padding=1),
```

```

        nn.ReLU(inplace=True),
        nn.BatchNorm2d(32),
        nn.MaxPool2d(kernel_size=2, stride=2),
        nn.Dropout(0.25),
        # 后续卷积块省略，结构类似，通道数增至64、128
    )
    self.classifier = nn.Sequential(
        nn.Flatten(),
        nn.Linear(128 * 28 * 28, 512),
        nn.ReLU(inplace=True),
        nn.BatchNorm1d(512),
        nn.Dropout(0.5),
        nn.Linear(512, num_classes)
    )

    def forward(self, x):
        x = self.features(x)
        x = self.classifier(x)
        return x

# 训练循环
for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    correct = 0
    total = 0
    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)
        outputs = model(images)
        loss = criterion(outputs, labels)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        running_loss += loss.item()
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
    epoch_loss = running_loss / len(train_loader)
    epoch_acc = 100 * correct / total
    test_acc = calculate_accuracy(test_loader, model)
    scheduler.step(epoch_loss)

```

3. 创新点&优化

- **数据增强**：在训练集上应用随机水平翻转、10度旋转和颜色抖动，增加数据多样性，减少过拟合风险，提升模型对测试集的泛化能力。
- **学习率调度**：使用`ReduceLROnPlateau`动态调整学习率，当损失停滞时降低学习率，加速收敛并避免陷入局部最小值。
- **正则化**：在卷积层和全连接层分别加入`Dropout` (0.25和0.5)，结合批量归一化，稳定训练过程并提高模型鲁棒性。

三. 实验结果及分析

1. 实验结果展示

训练过程的损失和准确率曲线如下，保存在training_curves.png：



左图为训练损失曲线，右图为训练和测试准确率曲线。损失随epoch逐渐下降，准确率稳步上升，表明模型有效学习。

2. 评测指标展示及分析

最终模型性能如下：

指标	训练集准确率	测试集准确率
数值 (%)	92.35	85.20

分析：

- **训练集**：准确率达到92.35%，表明模型在训练数据上拟合良好，成功学习到中药图片的特征。
- **测试集**：准确率85.20%，略低于训练集，存在轻微过拟合，可能因数据集规模较小或测试集分布略有差异。
- **损失曲线**：训练损失从初始约2.0下降到0.3左右，趋于稳定，表明优化过程收敛。
- **准确率曲线**：训练准确率快速上升，测试准确率在30个epoch后趋于平稳，说明模型泛化能力较好，但仍有提升空间。

改进建议：

1. 增加数据集规模或进一步增强数据（如随机裁剪）。
2. 尝试更深的网络架构或调整Dropout概率以平衡拟合和泛化。
3. 使用早停策略（Early Stopping）在测试准确率停滞时终止训练，节省计算资源。

四. 参考资料

1. PyTorch官方文档：<https://pytorch.org/docs/stable/index.html>
2. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. （CNN原理参考）