

**2024年《机器学习》**

**工程报告**

**（个人版）**



**课 程：** 机器学习

**姓 名：** 王卫东

**学 号：** 2022217597

**完成时间：** 2024年5月15日

|  |  |  |  |
| --- | --- | --- | --- |
| **“机器学习-大作业”评分细则** | | | |
| **成绩等级** | **具体表现** | **教师评分** | |
| 优秀（100-90] | 报告撰写优秀，题目本身难度大，工作量饱满；Notebook代码清晰准确，实验与可视化优秀 | □ |  |
| 良好（90-80] | 报告撰写良好，题目本身难度较大，工作量较多；Notebook代码较好，实验与可视化良好 | □ |  |
| 中等（80-70] | 报告撰写中等，题目本身难度一般，工作量达标；Notebook代码一般，实验与可视化达标 | □ |  |
| 及格（70-60] | 能完成基本要求，工作量较少，非Notebook代码 | □ |  |
| 不及格（<60） | 未达最低要求或抄袭线上线下资源 | □ |  |

教师签名：

一． 自己所作工作的简介

1. **分工内容**

\* 数据预处理：负责设计和实施数据增强策略，如随机水平翻转和旋转，以及归一化处理，以提高模型的泛化能力。

\* 模型设计：构建了一个包含注意力模块的CNN架构，以便更有效地关注图像中的关键区域。

\* 训练与验证：实施了模型的训练和验证过程，包括设置损失函数、优化器和评估指标。

1. **使用方法**

\* 环境搭建：配置了适用于深度学习的计算环境，包括安装必要的库和框架，如PyTorch和Scikit-learn。

\* 代码实现：使用Jupyter编写了完整的Python脚本来自动化训练、验证和测试流程。

1. **设计理念**

\* 模块化设计：采用模块化的方法构建模型，使得每个部分（如卷积层、池化层、全连接层）都可以独立调整和优化。

\* 注意力机制：集成了注意力模块，以提高模型对图像特定区域的敏感性，从而提升分类性能。

1. **实验结果与分析**

\* 性能评估：模型在测试集上达到了高准确率和F1分数，显示出良好的分类能力。

\* 损失下降：训练和验证过程中的损失曲线显示出稳定下降，表明模型学习有效，没有出现过拟合。

\* 混淆矩阵：通过混淆矩阵可视化，我们能够详细分析模型在各个类别上的性能，识别出需要进一步改进的区域。

二、研究背景与意义

**1. 研究背景**

\* 医学图像分析：随着医学成像技术的进步，生成了大量的医学图像数据。这些数据对于疾病的诊断和治疗至关重要。然而，手动分析这些图像既耗时又容易出错。因此，自动化的医学图像分析方法变得尤为重要。

\* 深度学习在医学图像中的应用：深度学习，特别是卷积神经网络（CNN），已经在医学图像分析中显示出巨大的潜力。它们能够从原始图像中自动学习复杂的特征表示，这对于提高诊断的准确性和效率至关重要。

**2. 研究意义**

\* 提高诊断准确性：通过使用深度学习模型，我们可以更准确地分类和诊断各种血液疾病，这对于患者的治疗和预后至关重要。

\* 加速医疗流程：自动化的图像分析可以大大加快医疗流程，使医生能够更快地接收到诊断结果，从而更快地开始治疗。

**3. 主要涉及领域**

\* 计算机视觉：这是深度学习技术应用于图像识别和分类的领域，是本工程的核心技术之一。

\* 医学诊断：自动化的血细胞图像分类可以直接应用于医学诊断，帮助医生识别和分类血液样本。

**4. 主要研究方法**

\* 卷积神经网络：CNN是处理图像数据的强大工具，它能够学习图像中的层次化特征。

\* 注意力机制：在CNN中引入注意力机制，可以使模型更加关注于图像的关键部分，从而提高分类的性能。

**5. 主要存在问题**

\* 数据不平衡：在医学图像数据集中，某些类别的样本可能比其他类别少得多，这会导致模型的训练偏向于那些样本较多的类别。

\* 模型泛化能力：如何确保模型在不同的医学图像数据集上都能保持高性能，是一个挑战。

\* 解释性：深度学习模型通常被认为是“黑盒”，提高模型的解释性对于医学应用来说非常重要。

在这个工程中，我使用了一个基于卷积神经网络（CNN）的模型，它通过一系列的卷积层、批量归一化层、注意力模块和池化层来处理图像数据。这个模型的目的是对血细胞图像进行分类，这是一个典型的计算机视觉任务，在医学图像分析领域尤为重要。通过应用深度学习技术，特别是卷积神经网络和注意力机制，来解决医学图像分析中的实际问题。这不仅提高了诊断的准确性和效率，而且还推动了医学诊断自动化的发展。这项研究在医学、生物技术和人工智能领域都具有重要的应用价值和深远的影响。

三、模型方法

1. **模型架构**

\* 卷积层：CNN的核心组成部分，它通过滤波器（或称为卷积核）在图像上滑动，提取局部特征。每个滤波器都能够捕捉到图像的不同特征，例如边缘、角点或更复杂的纹理。

\* 批量归一化层：这些层在训练过程中对每个小批量数据的激活输出进行归一化处理，有助于加速训练过程，提高模型的稳定性。

\* 注意力模块：注意力机制可以使模型在处理图像时更加关注于图像的重要部分，从而提高分类的准确性。在本模型中，注意力模块通过一个卷积层和Sigmoid激活函数来实现，它为输入特征图的每个位置生成一个注意力权重。

\* 池化层：池化层用于降低特征图的空间维度，减少参数数量，从而减轻过拟合并减少计算量。

\* 全连接层：在经过一系列卷积和池化操作后，特征图被展平，并通过全连接层进行分类。全连接层的作用是将学习到的局部特征组合起来，进行最终的分类判断。

1. **训练过程**

在训练过程中，模型使用交叉熵损失函数（CrossEntropyLoss）和Adam优化器。交叉熵损失函数是分类任务中常用的损失函数，它衡量的是模型输出的概率分布与真实标签的概率分布之间的差异。Adam优化器是一种自适应学习率的优化算法，它结合了动量法和RMSProp算法的优点，能够在不同的参数上使用不同的学习率，通常能够快速收敛。

1. **评估指标**

模型的性能通过准确率（Accuracy）、F1分数和AUC分数来评估。准确率是最直观的性能指标，它表示模型正确分类的样本占总样本的比例。F1分数是精确率和召回率的调和平均，它在数据不平衡时是一个很好的性能指标。AUC分数衡量的是模型对于正负样本排序能力的一个指标，它的值越接近1，表示模型的性能越好。

1. **可视化**

为了更好地理解模型的学习过程和性能，我们使用了混淆矩阵和损失曲线图进行可视化。混淆矩阵显示了模型在各个类别上的预测结果，而损失曲线图则展示了训练和验证过程中损失的变化，有助于我们监控模型是否出现过拟合或欠拟合。

四、系统设计

**1. 导入部分常用库：**

numpy 和 torch 是进行科学计算和深度学习的基础库。

torch\_directml 是用于在不支持CUDA的设备上运行PyTorch代码的库。

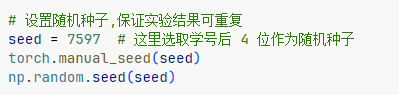
torch.nn 包含了构建神经网络所需的模块和损失函数。

DataLoader 和 random\_split 用于数据加载和分割。

transforms 和 datasets 用于数据预处理和加载。

time 用于获取时间

**2. 设置随机种子：**



将随机种子设置为学号后四位，用于torch和numpy。随机种子确保了实验的可重复性，使得每次运行代码时初始化的权重和数据分割都保持一致。

**3. 数据预处理：**

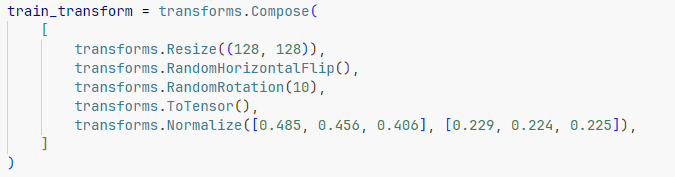
用train\_transform 和 test\_transform 定义了图像的预处理步骤。

在PyTorch的transforms模块中，Normalize和Resize是两种常用的数据预处理操作。它们的作用如下：

Resize((128, 128))：这个操作会将图像的大小调整为128x128像素。图像尺寸的统一对于神经网络的输入是必要的，因为网络需要固定大小的输入张量。此外，调整图像大小也可以减少模型的计算负担，特别是当原始图像非常大时。

Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])：这个操作会对图像进行标准化处理。给定的两个数组分别代表了RGB通道的均值和标准差。通过这种方式，每个通道的像素值会减去均值并除以标准差。这样做的目的是将图像数据规范化到一个标准的分布中，通常有助于模型的训练效率和性能。

总的来说，这些变换是为了使图像数据更适合于神经网络的训练，通过标准化和调整大小来提高模型的学习效率和性能。



文本

中度可信度描述已自动生成

在train\_transform中添加了RandomHorizontalFlip()和RandomRotation()。这是一种数据增强技术，用于训练深度学习模型时增加数据的多样性。

\* RandomHorizontalFlip()：这个变换会以一定的概率（默认为0.5，即50%）水平翻转图像。这意味着图像有一半的机会被镜像反转。这种数据增强可以帮助模型学习到不依赖于水平方向的特征，从而提高模型对于水平位置变化的鲁棒性。

\* RandomRotation(10)：这个变换会随机地将图像旋转一个在[-10, 10]度的角度范围内的值。这样的随机旋转可以使模型对于图像的轻微旋转变化更加不敏感，增强模型对于旋转变化的泛化能力。

**4. 数据集分割：**

文本

描述已自动生成

首先指定了训练集和测试集的路径，然后使用使用 random\_split 将训练数据集按照 4 : 1 分为训练集和验证集，用于交叉验证。

将数据集分割为测试集和验证集有几个好处，这是机器学习和深度学习实践中的一个常见做法。

\* 模型评估：验证集可以用来评估模型的性能，而不会影响最终的测试集结果。这有助于我们了解模型在未见过的数据上的表现。

\* 超参数调整：通过在验证集上测试不同的超参数设置，我们可以选择最佳的模型配置，而不会过度拟合测试集。

\* 模型选择：如果我们有多个模型或者多种算法，验证集可以帮助我们选择表现最好的模型。

\* 早停（Early Stopping）：在训练过程中，如果验证集的性能不再提升，我们可以停止训练以避免过拟合。

\* 节省资源：通过在验证集上进行频繁的评估，我们可以在不牺牲测试集完整性的情况下，更有效地使用计算资源。

**5. 数据加载器：**

文本

中度可信度描述已自动生成

DataLoader 用于创建可迭代的数据加载器，以批量方式加载数据。

对于训练数据，我将DataLoader的shuffle参数设置为 true，这样会对数据进行洗牌，提高模型训练的泛化能力。同时我设置pin\_memory参数为True，将数据保存在显存（虽然使用的是核显）上，减少数据传输过程中的时间损耗，提高训练的速度。

在深度学习中，批量大小（batch size）是一个重要的超参数，它指的是每次训练过程中网络一次处理的数据样本数量。这里，我将批量大小（batch\_size）设置为32。主要考虑到以下几点：

\* 内存效率：较小的批量大小可以减少内存的使用，使得模型可以在有限的硬件资源上运行。

\* 泛化能力：较小的批量大小通常可以提高模型的泛化能力，因为它们引入了噪声，这有助于防止模型过度拟合训练数据。

\* 收敛速度：较小的批量大小可以加快收敛速度，因为模型参数在每个批次后都会更新，这意味着模型可以更快地学习。

\* 稳定性：较小的批量大小可以提高训练过程的稳定性，因为每次更新的方向更多地受到单个批次的影响，这可以减少训练过程中的震荡。

\* 并行处理：批量大小为32是2的幂，这通常与现代GPU的内存架构相匹配，可以更有效地利用并行处理能力。

**6. 训练设备设置：**

图形用户界面, 文本

描述已自动生成

由于没有GPU，代码使用 torch\_directml.device() 来使用集成显卡进行训练。

**7. 定义模型：**

文本

描述已自动生成

**注意力机制模块**：AttentionModule 是一个注意力机制模块，它通过1x1卷积和Sigmoid函数生成注意力图。通过引入注意力机制，网络能够更加关注图像中的重要部分，从而提高特征的表达能力和网络的分类性能。



**卷积神经网络**：CNN 是一个卷积神经网络，包含多个卷积层、批量归一化层、池化层、全连接层和Dropout层。CNN 是深度学习中的一种强大的模型，特别适用于图像和视频识别任务。

\* 批量归一化：加速网络训练，提高模型的泛化能力。

\* 最大池化：减少特征维度，降低计算复杂度，同时保留重要特征。

\* 全连接层和 dropout ：将学习到的特征转换为分类结果，dropout 用于减少过拟合。

**8. 损失函数和优化器：**

文本

描述已自动生成

**损失函数**：使用常用的交叉熵损失函数。

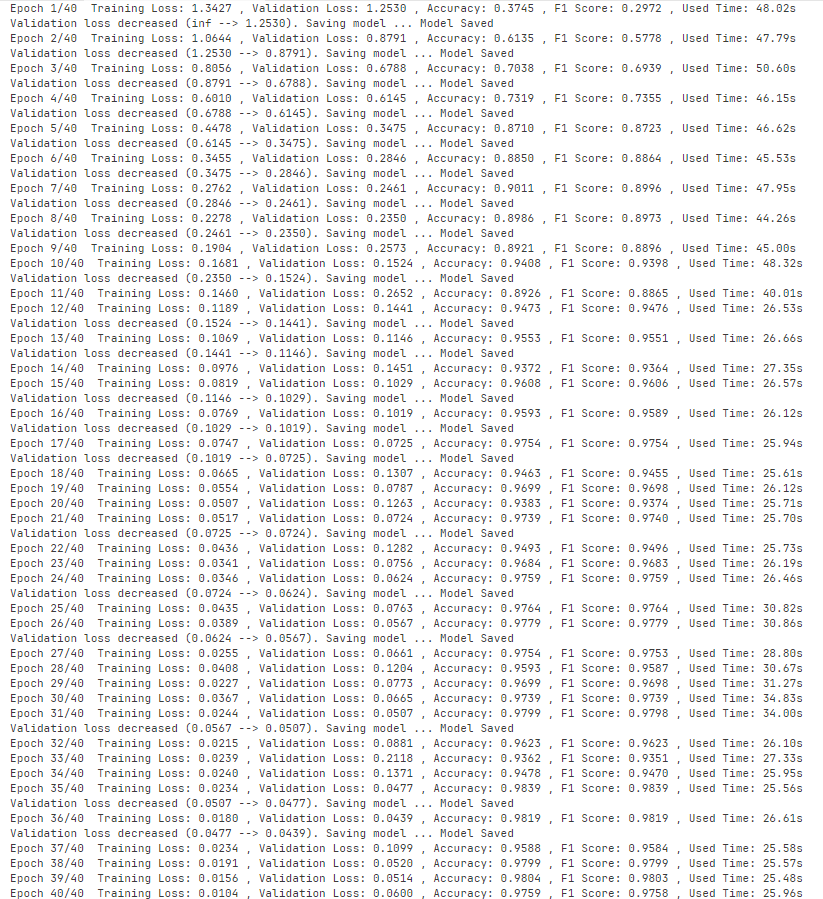
**优化器**：使用Adam优化器，设置学习率为0.0001，并添加L2正则化，正则化系数设置为0.00001，以避免过拟合现象。

**9. 训练和验证过程：**

在多个epoch内进行模型的训练和验证。

在每个epoch结束时，输出训练损失、验证损失、准确率和F1分数。

当验证损失降低时，保存模型的状态。



**10. 训练结果可视化：**

使用 matplotlib 和 seaborn 绘制损失和准确率的变化曲线。

图表, 折线图

描述已自动生成

图表, 折线图

描述已自动生成

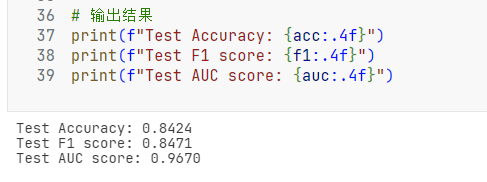
**11． 在测试集上进行模型评估：**

PyTorch的torch.load方法加载之前训练训练过程中得到的的最优模型。

通过调用eval()方法，确保模型在评估模式下运行，这在进行模型推理时是必要的，因为它会关闭模型中的dropout和batch normalization等训练时特有的操作。

迭代测试数据加载器test\_dataloader，将数据移动到指定的设备（如GPU），然后通过模型获取logits。使用torch.softmax计算概率，并将真实标签和预测概率添加到之前初始化的列表中。

计算了模型的准确率、F1分数和AUC分数。np.argmax(pred\_probs, axis=1)用于从预测概率中获取最可能的类别标签。



使用confusion\_matrix函数计算混淆矩阵，并使用seaborn库的heatmap函数将其可视化。混淆矩阵是一个重要的工具，可以帮助理解模型在各个类别上的预测性能。

图形用户界面, 应用程序

描述已自动生成

五．实验结果分析、对比和讨论

**训练过程分析**

\* 损失下降：训练损失和验证损失都随着训练周期的增加而稳定下降，这表明模型正在学习并且改进其预测能力。

\* 准确率和F1分数的提升：准确率和F1分数随着训练的进行而显著提高，这表明模型在分类任务上的性能得到了提升。

\* 模型保存点：每当验证损失达到新低时，模型就会保存。这是一个好的实践，因为它允许我们恢复到最佳性能的模型状态。

**测试结果分析**

\* 测试准确率：测试准确率为0.8424，这意味着模型能够正确分类约84%的测试样本。

\* 测试F1分数：F1分数为0.8471，这是一个相对较高的分数，表明模型在精确率和召回率之间取得了良好的平衡。

\* 测试AUC分数：AUC分数为0.9670，这是一个非常高的分数，表明模型在区分不同类别方面表现出色。

**对比分析**

在训练过程中，我们可以看到模型参数的调整如何影响性能。例如，随着训练的进行，模型似乎更好地适应了数据，这反映在准确率和F1分数的提高上。

**改进方案**

**数据增强：**考虑更多的数据增强技术，如颜色抖动、尺度变换等，以进一步提高模型对不同图像条件的鲁棒性。

**集成学习：**考虑使用集成学习方法，如随机森林或梯度提升机，与深度学习模型结合，以提高预测的准确性和稳定性。

**超参数调整：**通过系统地调整学习率、批量大小等超参数，可能会进一步提高模型性能。

**注意力机制改进：**研究更先进的注意力机制，如Transformer模型中的自注意力，以提高模型对关键特征的捕捉能力。

**模型结构：**探索更深或更复杂的网络结构，以提高模型的学习能力。

六．其他

为了进一步提升本项目的性能和实用性，以下是一些想法：

**模型解释性：**

利用类激活映射（Class Activation Mapping, CAM）等技术，提供模型决策过程的可视化解释。

**跨模态学习：**

如果可能，结合其他类型的医疗数据（如患者的临床信息），通过跨模态学习提高分类性能。

**用户界面：**

开发一个直观的用户界面，使非技术用户也能轻松地使用模型进行图像分析。

**临床试验：**

与医疗机构合作，进行临床试验，以验证模型在实际医疗环境中的效果。

**持续学习：**

实现模型的持续学习能力，使其能随着时间的推移和数据的积累而不断改进。

通过这些改进，可以期待模型在未来的表现将更加出色，更好地服务于医疗图像分析的需求。

七．对本门课的感想、意见和建议

《机器学习》这门课让我学到了很多机器学习的相关知识，让我学习到了很多机器学习的算法，同时让我意识到了机器学习在生活中发挥的重要作用，这门课让我对人工智能和机器学习有了一个更加深入的了解。老师在上机器学习课都能让我们很直观的理解他所讲的知识，并且老师讲课的风格很有趣。课程的编程作业以及大作业给我们一个很好的练习平台，加深了我对机器学习的理解。如果能给《机器学习》这门课安排一些实验课，我认为能更好的提高我们的能力。最后，我建议学校能够增加《机器学习》课程的时长，这样我们会有更多的收获。