南 开 大 学

本 科 生 毕 业 论 文（设 计）

中文题目：文档图像分类方法研究与分析

外文题目：Research and analysis of document image classification methods

学　　号：2011631

姓　　名：梁奕宸

年　　级：2020 级

专　　业：软件工程

系　　别：软件工程

学　　院：软件学院

指导教师：张玉志

完成日期：2024 年 5 月 9 日

关于南开大学本科生毕业论文（设计）的声明

本人郑重声明：所呈交的学位论文，是本人在指导教师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或没有公开发表的作品内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：

2024 年 5 月 9 日

本人声明：该学位论文是本人指导学生完成的研究成果，已经审阅过论文的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

学位论文指导教师签名：

2024 年 5 月 9 日

摘　要

如今，随着信息化时代的到来，越来越多的文档被数字化，因此如何有效地管理、搜索和利用这些文档变成一个非常重要的问题。在这样的时代背景下，深度学习方法有了更广阔的前景，开展对于文档图像分类方法的研究与分析具有重要的理论价值和现实意义。文中首先对图像分类的相关知识和一些经典的深度学习框架进行了研究和对比，谈到了相比于传统机器学习方法的一些优势；介绍了注意力机制和移动窗口的引入，并深入研究了 Swin Transformer 模型是如何基于卷积神经网络和 ViT 的先验知识优化而来。

然后重点介绍了 Swin Transformer 模型中的关键模块，包括移动窗口内的自注意力机制和相对位置编码的部分；在初步了解模型后，笔者进行了多组对比实验来进一步研究影响 Swin Transformer 性能的因素，分别是预处理方法、数据集的特征、学习率调整策略、预训练模型和优化器等。

最后提出了模型未来的优化方向和应用场景，希望未来能够在更多的场景中看到更多性能更优秀的模型。

关键词：图像分类；深度学习；Swin Transformer；移动窗口机制；对比实验；

Abstract

With the advent of the information age, an increasing number of documents are being digitized, making it crucial to effectively manage, search, and utilize these documents. In this context, research and analysis of document image classification methods have significant theoretical value and practical significance. This paper first studies and compares relevant knowledge of image classification and some classical deep learning frameworks, highlighting the advantages compared to traditional machine learning methods. The introduction of attention mechanism and shifted window is discussed, followed by an in-depth investigation into how the Swin Transformer model is optimized based on prior knowledge of convolutional neural networks and ViT.

The key modules of the Swin Transformer model are then emphasized, including the self-attention mechanism within the moving window and the part related to relative position encoding. After gaining a preliminary understanding of the model, three sets of comparative experiments are conducted to further study the factors affecting the performance of the Swin Transformer: Preprocessing methods, dataset features, learning rate adjustment strategies, pre-trained models and optimizers.

Finally, future optimization directions and application scenarios of the model are proposed, hoping to see more superior-performing models in various scenarios in the future.

Keywords: Image classification; Deep learning; Swin Transformer; Shifted window mechanism; Comparative experiments.

目 录

第一章 绪论

第一节 研究背景和意义

当谈论到文档图像分类时，实际上是在谈论一种能够使我们的日常工作更加高效、智能化的技术。现代社会已经进入了数字化时代，大量的文档被电子化处理，这包括从合同、报告到表格等各种类型的文档。但是，这些文档的数量往往是庞大的，因此如何有效地管理、搜索和利用这些文档就成了一个非常重要的问题。

首先，可以考虑一个办公室的场景。办公室里可能会有大量的文件，包括合同、报告、会议记录等。通过文档图像分类技术，我们可以自动将这些文档按照其内容或特征分类，比如将所有的合同放在一起，将所有的报告放在一起，这样就可以更方便地管理和查找这些文档了。而且，一旦文档被分类，就可以轻松地建立一个智能搜索系统，用户可以通过关键词或其他条件快速地找到他们需要的文档，而不必手动查阅每一个文件。

除了办公室场景，文档图像分类技术还可以在许多其他领域发挥作用。比如，在图书馆或档案馆里，有大量的文献和档案需要管理和分类。通过文档图像分类技术，可以将这些文献和档案按照主题、作者、出版年份等信息进行分类，从而方便读者或研究人员查阅和利用这些资源。在金融领域，文档图像分类技术可以帮助银行或保险公司对大量的合同和文件进行自动化处理和管理，提高工作效率和准确性。在医疗领域，文档图像分类技术可以帮助医院或诊所对患者的病历和医疗记录进行分类和管理，支持临床决策和医疗服务。

除了提高工作效率和智能化服务外，文档图像分类技术还有助于信息安全和隐私保护。通过对文档中的敏感信息进行识别和分类，可以帮助组织和机构加强对这些信息的保护和管理，防止泄漏和滥用。

综上所述，文档图像分类技术在促进信息化、提高工作效率、支持智能化服务等方面具有重要的研究价值和实际意义。随着科技的不断进步和应用场景的不断拓展，我们相信文档图像分类技术将会在未来发挥越来越重要的作用，并为我们的生活带来更多的便利和智能化体验。

第二节 国内外研究现状

图像分类领域，国内外的研究现状都在不断地发展和演进[1]。以下是一些国内外研究现状的主要方面：

传统方法：在早期，文档图像分类主要依靠传统的特征提取和机器学习方法，如 SIFT、HOG 等特征提取方法结合 SVM、KNN 等分类器进行分类。这些方法在一定程度上取得了一定的效果，但受限于特征的表达能力和泛化能力。

深度学习方法：随着深度学习[2]的兴起，特别是卷积神经网络（CNN）的成功应用，文档图像分类领域也开始使用深度学习方法。通过使用卷积神经网络进行端到端的特征学习和分类，取得了较好的效果。其中，一些经典的模型如 AlexNet、VGG、ResNet 等被广泛应用于文档图像分类任务。

迁移学习和预训练模型： 迁移学习和预训练模型的出现为文档图像分类带来了新的机遇。研究者们发现，通过在大规模数据集上预训练的模型，如 ImageNet 上预训练的模型，可以在文档图像分类任务上进行微调，取得更好的效果。这种方法能够利用大规模数据集中学到的通用特征，提升文档图像分类的性能。

注意力机制和 Transformer 模型：最近，注意力机制和 Transformer 模型在文档图像分类领域也得到了广泛关注。这些模型能够有效地捕捉文档图像中的长距离依赖关系和语义信息，提升了分类性能。Swin Transformer 等新型模型的出现，进一步推动了文档图像分类的发展。

数据增强和对抗训练：数据增强和对抗训练等方法也被应用于文档图像分类任务中，以提升模型的泛化能力和抗干扰能力。通过在训练过程中引入随机扰动或对抗样本，可以增加模型对不同条件下的适应能力。

综上所述，国内外在文档图像分类领域的研究现状呈现出多样化和多方面的发展趋势，不断涌现出新的方法和技术，推动着文档图像分类技术的不断进步和完善。

第三节 研究的关键问题

作为当下深度学习领域的骨干网络，对 Swin Transformer[1]模型具体结构的进一步了解是非常必要的，鉴于传统机器学习和卷积神经网络的先验知识，来研

究 Swin Transformer 是如何在 ViT 上进行改进，在移动窗口内计算自注意力，并结合卷积神经网络的层次化结构，形成一个层次化的 Transformer 模型；除此之外，模型的准确性还涉及很多超参数的设置和数据集本身的问题，预训练模型涉及的迁移学习的方法等，因此，为了研究不同因素对于模型性能的具体影响程度，笔者进行了多组的对比试验来进一步了解 Swin Transformer 模型。

1.3.1 传统机器学习分类方法的局限性

1．传统方法通常依赖于手工设计的特征，例如颜色直方图、纹理特征和形状描述符等。这些特征需要领域专家花费大量时间和精力来设计，且往往难以捕捉到文档图像中的高级语义信息。

2．传统方法的特征表示通常较为简单，缺乏对文档图像中复杂结构和内容的泛化能力。因此，当面对图像多样性和类别不平衡等挑战时，传统方法的性能往往受到限制。

3．传统方法往往对图像的变换（如旋转、缩放、平移等）和噪声非常敏感，导致模型的鲁棒性较差。这使得传统方法在实际应用中往往难以处理现实场景中的复杂图像数据。

4．传统方法通常采用基于统计学习的分类器（如支持向量机、随机森林等），在处理大规模数据时计算和存储成本较高，且模型的训练和推理速度较慢。

5．传统方法中的参数通常需要手动调整，例如分类器的超参数和特征选择的阈值等。这需要领域专家具有丰富的经验和专业知识，且往往需要进行大量的实验来找到最优的参数设置。

1.3.2 卷积神经网络在文档图像分类任务中的局限性

数据需求量大：CNN 通常需要大量的标记数据进行训练，以学习有效的特征表示。对于某些文档图像分类任务，可用的标记数据可能有限，这可能限制了 CNN 模型的性能。

对位置和尺度敏感：CNN 是局部连接的神经网络，因此对于输入图像的位置和尺度变化非常敏感。这可能导致在不同分辨率或不同排列方式的文档图像上的性能下降。

可解释性差：CNN 模型通常被认为是黑盒模型，难以解释其内部如何进行决策。在一些应用场景下，尤其是对于需要解释分类决策的任务，这可能是一个限制因素。

过拟合：对于小样本问题，CNN 容易过拟合，特别是在训练数据不足时。这可能导致模型在未见过的数据上的泛化性能下降。

计算和存储成本高：CNN 通常具有大量的参数和复杂的计算结构，因此在训练和推理时需要大量的计算资源和存储空间。这可能对于资源受限的环境来说是一个限制因素。

对输入数据质量敏感：CNN 对输入数据质量的要求较高，例如噪声、模糊或失真可能会对其性能产生负面影响。

1.3.3 ViT 和 Swin Transformer

其实 Swin Transformer 就是一个使用了移动窗口的层级式的 Vision Transformer，Swin 来自 Shifted Window（移动窗口），希望 Vision Transformer 像卷积神经网络一样，也能够分成几个 block，做层级式的特征提取，从而导致提出来的特征有多尺度的概念。

下面本文对两种模型做一些关键性的对比：

在建模地尺度方面，ViT[2]将整个输入图像拆分成固定大小的图像块，并将每个图像块视为序列输入，然后通过 Transformer 模型进行处理。这种方法限制了模型对图像中局部结构的建模能力；而 Swin Transformer 引入了分层的注意力机制，将输入图像分解为一系列的图像块，并在不同阶段（stage）上交换局部和全局信息。这样，模型能够更好地捕获图像中的局部和全局特征，提高了模型对图像语义结构的建模能力。

在窗口划分和自注意力的计算方面：ViT 在整个图像上直接应用自注意力机制，导致模型的计算复杂度较高，尤其是在处理大规模输入图像时；Swin Transformer 则引入了窗口式的自注意力机制，将输入图像分割成一系列的小图像块，并在每个窗口上独立地应用自注意力机制。这样，这样只要窗口大小是固定的，自注意力的计算复杂度就是固定的，整张图的计算复杂度就会跟图片的大小而成的线性增长关系，就是说图片增大了 x 倍，窗口数量也增大了 x 倍，计算

复杂度也就乘以 x，而不是乘以 x 的平方，因此能够减少计算的复杂度，使模型能够更高效地处理大规模的输入图像。

除此之外，二者的区别还在于 ViT 使用单一的 Transformer 编码器层来处理整个输入序列，缺乏对不同尺度信息的有效建模能力；Swin Transformer 则基于 CNN 的先验知识，使用分层的特征表示，将输入图像划分为多个 stage，每个 stage 包含多个 Transformer 编码器层。这样，模型能够更好地处理不同尺度的信息，并且有助于提高模型的泛化能力。

通过以上改进，Swin Transformer 在图像分类任务中能够更好地捕获图像的局部和全局特征，提高了模型的性能和泛化能力，并且在一些情况下能够使用更小的模型规模取得类似甚至更好的性能。

1.3.4 特殊文档数据集的针对性研究

图片本次使用的数据集是由近代书籍中提取出来的页面图像，相较于传统的公共数据集 ImageNet[3]、CIFAR-10 以及 MINIST 来说，本次实验使用的数据集明显具有更大的规模，图片的尺寸基本为 1300×2000 像素，而更大尺寸的图片，给实验使用的模型也提出了更高的要求：

计算资源要求：更大尺寸的图片通常意味着更多的像素和更高的分辨率，这会增加模型的计算负担。因此，处理大尺寸的图片可能需要更多的计算资源，包括更多的内存和更长的处理时间。

模型设计和参数调整：需要设计具有更大容量的模型，以便能够充分利用图像中的信息。此外，可能需要对模型的参数进行调整，以适应更大尺寸的输入数据。

内存和存储需求：更大尺寸的图片需要更多的内存来存储和处理，这可能会导致内存不足或存储空间不足的问题。因此，处理更大尺寸的图片可能需要更大的内存和存储资源。

数据预处理和增强：更大尺寸的图片可能需要更复杂的数据预处理和增强技术，以确保模型能够有效地学习和泛化。这可能涉及裁剪、缩放、旋转等操作，以及其他数据增强技术。

模型的鲁棒性：更大尺寸的图片可能包含更多的细节和复杂性，这可能会增

加模型的复杂性和过拟合的风险。因此，需要设计具有更强鲁棒性的模型，以应对更大尺寸图片带来的挑战。

除此之外，数据集本身还存在各类型数据分布不均衡的问题，从书籍中提出的页面图像，正文页的占比显然是更高的，这是在做文档图像分类时不可避免的事情。所以本文需要研究数据不均衡时对模型正确率的影响。

第四节 论文的研究目标

本文首先主要研究 Swin Transformer 模型本身，与 CNN 以及 Vision Transformer 相比所提出的移动窗口内的自注意力机制，以及类似于卷积神经网络的层级式结构，其次就是针对特殊数据集的针对性研究，在不同情况下参数和数据本身的调整对模型分类准确率的影响。

在之前的研究中，对于文档图像分类的主流方法仍是基于 CNN 模型（如 ResNet、VGG 等）对文档图像进行特征提取，然后使用全连接层或其他分类器对提取的特征进行分类，相较于之前的传统机器学习算法已经在准确率和性能上有了更好的表现。但自从 2021 ICCV 的最佳论文《Swin Transformer: Hierarchical Vision Transformer using Shifted Windows》的发表，结合其之后在各大数据集上优良的表现，都已经充分说明了 Transformer 在视觉领域上的强大潜力，也将在未来各个领域中取代卷积神经网络成为计算机视觉领域的骨干网络。如今本文的自有数据集中包含的都是大尺寸的图像，而 Swin Transformer 模型中通过下采样层达到的对高分辨率图像较好的处理效果正好满足本文的实验需求，所以此次实验本文也选用 Swin Transformer 来作为主干网络，下面是本篇论文的主要研究内容：

1．在原始论文以及相关文献的基础上，了解模型的整体结构、自注意力机制和层归一化等每个组件的功能以及如何组合这些组件来构建完整的模型；并基于模型架构图研究各个组件之间的连接方式和信息流动，理解该模型是如何使用分层架构来处理大尺寸图像。

2．通过对学习率的不同调整策略和优化器的选择来优化模型训练过程，以期望在比较稳定的情况下加快模型收敛的速度，以更快得到预测效果较好的模型。

3．由于本文的数据集是由近代书籍中提取得来，所以原始的页面类型比例

显然是不均衡的，正文页的比例将会远超其他类型的页面。因此为了更客观地评估模型的预测效果，我引入常见的文档图像的开源数据集 Tobacco3482 与本文的自由数据集作预测结果上的对比；此外也会对自由数据集本身作一些调整，以研究数据集均衡以及不同预处理方法与否对模型的准确率有影响。

4．研究不同规模的预训练模型和预处理方法对预测准确率的影响。本文选用在同样规模数据集上进行训练的两个层数和尺寸不同的 Swin Transformer 模型来进行对比，研究图片调整的尺寸大小和模型的层数对于模型性能的影响。

第五节 论文的内容组织

本文共分为七章，各章的主要内容如下：

第一章 绪论。简要介绍文章的研究背景和国内外的研究现状，并针对本次研究的关键问题进行总结，作为本文的基础内容。

第二章 图像处理相关知识

第三章 Swin Transformer 模型的基本框架

第四章 基于 Swin Transformer 的文档图像分类网络

第五章 总结与展望

参考文献

致谢

第二章　图像处理相关知识

第一节 图像分类方法

2.1.1　卷积神经网络

卷积神经网络（Convolutional Neural Networks，CNN）作为一种优秀的深度学习模型，出于其良好的特征提取能力、参数共享和稀疏连接的优势、适应性强、泛化能力强等特点，在计算机视觉领域以及 NLP 领域都已经有比较成熟且广泛的应用。

2.1.1.1　输入层

卷积神经网络的输入层用于接收数据的输入，由原始的图像或文本转化为可以被卷积神经网络处理的向量或矩阵（由像素值大小组成）。在这个过程中，还包括了数据预处理、特征提取等多种功能，对后续的模型处理性能都会产生重要的影响。

2.1.1.2　卷积层

这是卷积神经网络非常重要的一个结构，也是对输入矩阵进行特征提取的关键一步。每个卷积层中都包含多个卷积核，通过卷积核在输入层传来的矩阵上滑动来完成特征提取的工作；卷积核本身就是一个规模小于或等于输入图像矩阵，通过设置步长（Stride）和填充（Padding）操作来控制卷积核的移动，每移动到新的位置就做一次对应矩阵的乘积和求和，最后得到一个新的矩阵，本文叫做特征图（Feature Map），其中值的大小就代表着该局部区域内的特征。第一层的卷积核提取到的信息是有限的，往往只能得到比较低级的特征，但通过多层的迭代，就可以从低级的 Feature Map 中提取更复杂的特征。

2.1.1.3　激活层

激活层通过激活函数将输入值映射到非线性空间，使神经网络能够学习非线性的关系，对复杂的数据进行更好的建模和拟合，有些激活函数还可以缓解梯度消失的问题，从而使得神经网络的训练更加稳定和高效，具有更好的性能。

2.1.1.4　池化层

但为了充分提取特征，我们不得不接收非常多的特征图，然而得到的大量的特征可能并不是笔者想要的，其中很多冗余的特征往往会带来过拟合和维度过高的问题，为了解决这个问题，本文引入了池化层以及下采样的概念。具体来讲，就是在卷积操作过后，对得到的 Feature Map 在进行特征提取，将其中最具代表性的特征提取出来，这个过程类似于卷积的操作，对划分好的区域中进行下采样从而只得到一个值。常见的池化操作有最大池化（Max Pooling）和平均池化（Average Pooling），区别就在于：最大池化是选择区域中的最大值作为最具代表性的特征，而平均池化是取所有值的平均值，充分考虑区域中每个位置的值对特征的影响。所以由上可知，池化层通常与卷积层是紧密相连的，对卷积层的结果进行降维，减少网络中的参数个数。

2.1.1.5　全连接层

在通过前面的多层结构提取出图像的特征后，笔者需要做的就是利用这些特征来完成本次实验最终的任务，全连接层（Fully Connected Layers）的目的就是通过对上层特征进行连接，进行加权后用于将输入分为不同的类。全连接层的每一个节点都和上一层的每个节点全部相连，通过权重矩阵和输入层相乘并加上一个偏置（bias），以得到最后的全连接层的输出矩阵。

具体的计算公式如下：

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \times \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{23} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \tag{3.1}$$

其中$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$表示的是全连接层的结果，$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$表示的是全连接层的输入，

$\begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}$表示的是权重大小，$\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$表示的是偏置量。

2.1.1.6　输出层

输出层是用来输出不同类别的概率分布，具体的概率数值是应用 Softmax 函数得到的，通过将输出值转换为范围在[0,1]并且和为 1 的概率分布。其具体的计算公式如下：

$$\mathrm{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{3.2}$$

其中z_i是第 i 个类别的输出值，下面是 Softmax 的一个示例，从最后的输出结果可知，分类结果是类别 1 的可能性最大。

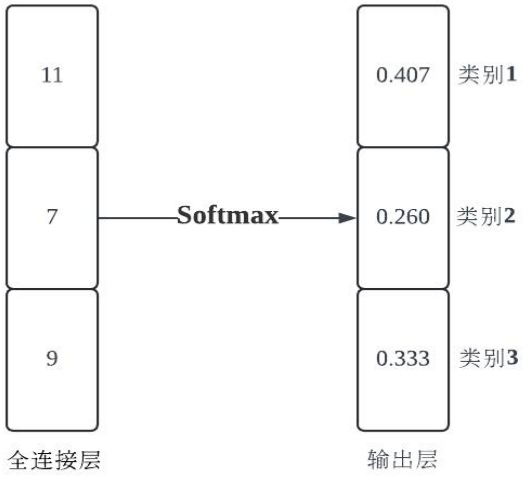


图 2.1 Softmax 示例

2.1.2　图像分类经典框架

2.1.2.1　AlexNet

AlexNet[4]是由 Alex Krizhevsky、Ilya Sutskever 和 Geoffrey Hinton 在 2012 年提出的一个经典的卷积神经网络模型，用于在 ImageNet 数据集上进行图像分类，是深度学习在计算机视觉领域的重要里程碑之一。它作为新一代的神经网络，

下面是它一些比较重要的创新点：

如图 2.2 所示，AlexNet 是一个相对较深的卷积神经网络，包含 8 个卷积层和 3 个全连接层，其中包含的大量参数可以保证模型学习到更复杂的图像特征；与此同时，AlexNet 使用 ReLU 作为激活函数，取代传统的 Sigmoid 函数，具有

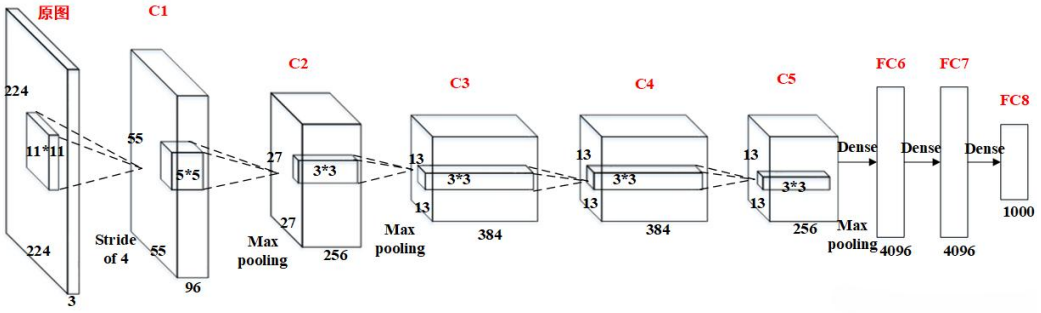


图 2.2 AlexNet 结构示意图

非线性的特性和稀疏激活性的特点，有助于加速模型的收敛速度和避免梯度消失的问题；此外，模型引入 Dropout 正则化技术，通过随机地将一部分神经元的输出设置为零，以减少模型的过拟合程度，以提高其泛化能力和性能。最后 LRN 层的引入，对神经元的输出进行了局部的归一化处理，进一步增强了模型的泛化能力，提高了模型的鲁棒性。但由于 AlexNet 包含大量的参数，训练过程十分耗时，因此为了加速训练，需要将训练数据分配到多个 GPU 上进行并行计算，这也在一定程度上增加了训练成本。

2.1.2.2　VGG

VGG[5]是 Karen Simonyan 和 Andrew Zisserman 在 2014 年提出的卷积神经网络模型，同样在 ImageNet 上完成了图像分类任务。相比于之前的网络，他的主要贡献则在于提出了一种简单有效的网络结构设计原则，即使用连续的 3×3 卷积核和池化层，以及全连接层，构建了一个具有较深层次的卷积神经网络。

在搭建过程中，卷积层都采用 3×3 的卷积核，池化层采用 2×2 的池化核，这样统一的网络结构设计被证明是更加简单有效的；相较于之前的神经网络，VGG 包含 16 或 19 个卷积层，更深的层数使得 VGG 可以学习到更加复杂的图像特征；此外，用小卷积核代替较大的卷积核，可以增加网络的深度，并且减少参数的数量，多个小卷积核的组合可以学习到更复杂的特征表示。

2.1.2.3　ResNet

ResNet[6]是由 Kaiming He 等人在 2015 年提出的一个经典的深度卷积神经网络模型，其主要的创新点在于引入了残差连接的概念，通过跨层的直接连接来解决梯度消失和梯度爆炸的问题，使得可以训练更深的网络结构。下面是一些 ResNet 的一些重要特点和创新点；

1．残差连接： ResNet 通过在每个卷积层的输入和输出之间添加一个残差连接来构建残差块（Residual Block）。这个残差连接将输入信号直接添加到了卷积层的输出之中，即 output=input+F(input)，其中 F(・)是残差函数。这种跨层的直接连接可以有效地传递梯度信息，解决了深度网络中梯度消失和梯度爆炸的问题，使得可以训练更深的网络结构。

2．深度：ResNet 是一个非常深的卷积神经网络模型，可以堆叠成几十甚至上百层。这使得 ResNet 可以学习到非常复杂的特征表示，提高了模型的性能和泛化能力。

3．批量归一化： ResNet 在每个卷积层之后都采用了批量归一化操作，将每个特征通道的输入归一化为均值为 0、标准差为 1 的分布。这有助于加速模型的收敛速度和稳定性，同时还可以减少模型对初始参数的依赖。

4．1x1 卷积核：ResNet 在残差块中使用了 1x1 的卷积核来降低输入的通道数，然后再使用 3x3 的卷积核进行特征提取。这样可以减少模型的计算复杂度和参数数量，同时还可以增加网络的非线性表达能力。

5．全局平均池化：ResNet 在最后的卷积层之后采用了全局平均池化操作，将每个特征图的尺寸降低到 1x1，然后通过一个全连接层进行分类。这种操作可以减少模型的参数数量，同时还可以提高模型的泛化能力。

如图所示，X 是作为残差块的输入，F(x)是经过第一层线性变化并激活后的输出，在第二层进行线性变化后面的激活层之前，将 F(x)与输入值 X 相加，然后再经过激活函数 relu 后输出。在第二层输出值激活前加入 X 的这条路径称作 shortcut connection。

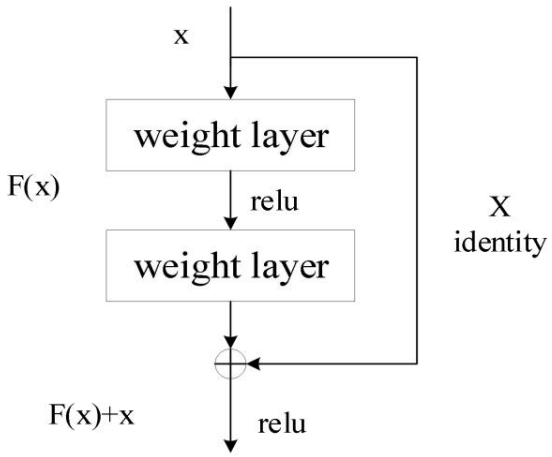


图 2.3 残差结构图

2.1.3　注意力机制

注意力机制与人类的注意力机制在其灵感来源上有关联。人类的注意力机制指的是人类在感知、认知和决策过程中，根据外界环境和任务需求，有选择性地关注和处理特定的信息。这种选择性关注使得人类能够更有效地处理复杂的信息，快速做出正确的决策。

深度学习中的注意力机制[9]（Attention Mechanism[10]）借鉴了这种人类的注意力机制的思想，通过动态调整输入数据的权重，使模型在处理信息时能够有选择性地关注特定的部分，从而提高模型的性能和效率。因此，注意力机制在一定程度上模拟了人类的注意力机制，使得深度学习模型更接近于人类的认知方式，更有效地处理复杂的任务。

下面可以用一个示例来帮助笔者更好地理解注意力机制，展示了人类在看到一幅图时如何高效分配有限注意力资源的，其中未模糊的“锦江饭店”招牌就是人类的视觉系统更加关注的目标，从图中可以看出：人们会把注意力更多地投入到离更近、更大，颜色对比更明显的区域。

图 2.4 人类视觉系统的注意力机制说明

当前主流应用中，最典型的注意力机制包括自注意力机制、空间注意力机制和时间注意力机制。这些注意力机制允许模型对输入序列的不同位置分配不同的权重，以便在处理每个序列元素时专注于最相关的部分。

在本次实验研究的 Swin Transformer 模型中，包含的是自注意力机制[11]（Self-Attention Mechanism）。自注意力机制是一种用于处理序列数据的注意力机制，其中每个元素都可以与序列中的其他元素进行交互，并根据它们的相关性动态调整其表示，其基本思想如下：

1．Query、Key 和 Value 的计算：给定输入序列，首先通过线性变换将其映射到三个不同的表示空间，分别为 Query、Key 和 Value。这些变换可以用矩阵乘法实现，以将输入向量投影到更高维度的空间中。

2．计算注意力权重：对于每个 Query 向量，计算其与所有 Key 向量之间的相似度（通常使用点积或其他相似性度量），并将这些相似度转换为权重值。这些权重表示了 Query 与不同元素之间的关联程度，用于加权求和 Value 向量以得到最终的表示。

3．加权求和：将上一步中计算得到的权重与对应的 Value 向量相乘并求和，得到 Query 对应的加权表示。这个加权求和的过程可以看作是模型在给定

Query 时对序列中其他元素的关注程度，其中注意力权重越高的元素对最终表示的贡献越大。

最后在实际应用中，多会选用多头注意力机制作为神经网络中的重要模块，用来提高模型的表达能力和稳健性，通常会并行地计算多组注意力权重，并将它们拼接或叠加在一起，然后通过额外的线性变换将其投影到最终的表示空间。

第二节 常见预处理算法

图像预处理[12]在图像处理和计算机视觉任务中扮演着至关重要的角色，无论是传统方法还是深度学习方法，其分类性能都与输入的图像质量有紧密的关联。因此如何选择合理的预处理算法来提高图像的质量，对于本文的实验来说亚视需要考虑的重要因素，下面对于常用的预处理算法进行简单的介绍：

2.2.1　标准化

将数据集中的特征按照一定的方式进行缩放，使得它们具有零均值和单位方差。具体一般是通过每个特征减去其均值，然后除以其标准差来实现。这有助于加速模型的训练过程，并且可以使不同特征之间的权重更加均匀，提高模型的稳定性和收敛速度。

2.2.2　数据增强

通过对训练数据进行随机变换，如旋转、翻转、裁剪、缩放、平移、添加噪声等。这些变换可以生成与原始数据相似但又略有不同的数据样本，增加数据的多样性，从而提高模型的泛化能力。

2.2.3　噪声过滤

对输入数据进行去噪处理，如高斯模糊、中值滤波等，可以减少数据中的噪声对模型的影响，提高模型的鲁棒性和性能表现。

2.2.4　正则化

正则化是在模型训练过程中引入额外的约束条件，以减少模型的复杂度并防止过拟合。常见的正则化方法包括 L1 正则化、L2 正则化等，这些方法可以惩罚模型中的大权重，从而促使模型更加简单和泛化能力更强。

2.2.5　缺失值处理

缺失值处理是指对输入数据中的缺失值进行处理，以确保模型能够正确处理这些缺失值。常见的处理方法包括填充缺失值、删除包含缺失值的样本或特征、使用插值等。这些方法可以确保模型能够充分利用可用的信息，并且能够正确地处理缺失值。

第三节 常见图像特征提取算法

作为计算机视觉领域中至关重要的一步，特征提取方法也经历了多次的更迭，不断为后续任务的推进提供着动力。一个好的特征提取方法，直接影响着特征表示的质量；好的特征表示能够更好地捕捉图像的语义信息和抽象特征，减少输入图像受光照、视角等因素的影响，对这些变化提供一定鲁棒性的特征表示，进而直接影响着分类器的性能；除此之外，好的特征提取方法能将原始数据的高维特征空间有效地降低到一个更低维度的表示空间，减少特征的冗余，从而节省计算资源，提高分类器的泛化能力。

常见的图像特征提取算法[13]分为传统方法和深度学习方法。传统方法通常是通过手工设计特征提取算法来获取数据的有效表示，而基于深度学习的方法可以端到端地学习图像的特征表示，通常能够获得更高质量的特征表示。相较于传统方法，深度学习方法具有更强的泛化性，且能处理更复杂的数据，也不需要很强的专业性来完成参数的设计，显然具有更广泛的应用前景，因此在多数领域深度学习方法都已经逐渐成为主流，相比之下，PCA 等传统方法在图像特征提取方面的应用已经相对较少，更多地被用于其他领域或者作为基准方法进行比较。

下面就对其中的一些常见方法做一些了解：

2.3.1　主成分分析

主成分分析[15]作为一种简单高效的降维技术，用于将高维数据集投影到低维空间，保留主要特征并去除噪声。下面是其主要的工作原理：

1．计算协方差矩阵：首先，PCA 基于数据的协方差矩阵来理解数据之间的关系。对于一个包含 n 个特征的数据集，协方差矩阵描述了这些特征之间的线性关系。

2．特征值分解：PCA 通过对协方差矩阵进行特征值分解来找到数据中的主成分。特征值分解将协方差矩阵分解为特征值和对应的特征向量。特征值表示了数据中的方差，而特征向量则表示了方差最大的方向。

3．选择主成分：根据特征值的大小，选择前 k 个特征值对应的特征向量作为主成分。这些主成分捕捉了数据中最大方差的方向，因此包含了数据的大部分信息。

4．降维：通过将原始数据投影到选定的主成分上，实现数据的降维。这样可以减少数据的维度，同时保留了数据中最重要的信息。

5．重构：如果需要，可以通过将降维后的数据投影回原始空间来重构数据，尽管在这个过程中可能会丢失一些信息。

2.3.2　预训练模型的特征提取

预训练模型的特征提取是指利用在大规模数据集上预训练的深度学习模型，将其作为特征提取器来获取输入数据的高层次特征表示。

由于自己用来实验的数据集往往不够大，也不希望花费太多的时间和资源进行训练，但又希望尽可能获得性能更好的模型，于是本文引入了预训练模型来学习到更丰富的特征表示。

首先需要选择一个在大规模数据集上预训练的深度学习模型，常见的包括 VGG、ResNet、Inception、EfficientNet 等。通常选择的模型应该具有良好的性能和泛化能力，并且在相似的图像分类任务上已经取得了良好的效果，

然后移除掉原模型中的全连接层，而保留卷积层部分，起到提取特征的作用；

在接收了大量的图像输入后，这些特征表示通常具有较高的语义信息和抽象能力，可以用于各种后续任务，如图像分类、目标检测等。根据实际任务的需要，

具体模型可以选择不同层次的特征表示，一般来说，底层的特征更加局部化和具体化，而顶层的特征更加抽象和语义化。这些通过预训练模型得到的特征表示，既可以作为新任务的输入，也可以作为基础特征，进一步在新的任务上进行微调或训练新的分类器。

与传统的特征提取方法相比，预训练模型的方法具有很强的通用性和灵活性，能够快速、有效地获取输入数据的高层次特征表示，并且在很多视觉任务中都已经被证明切实能够取得更好的性能，在越来越多的计算机视觉任务中扮演着重要的角色。

第三章　Swin Transformer 模型的基本框架

通过之前的分析可以得知，因为需要针对大尺寸图片进行分类任务，所以本文选择 Swin Transformer 模型完成后续实验是比较理想可行的，下面本文就来详细介绍一下模型本身是由哪些模块构成的。

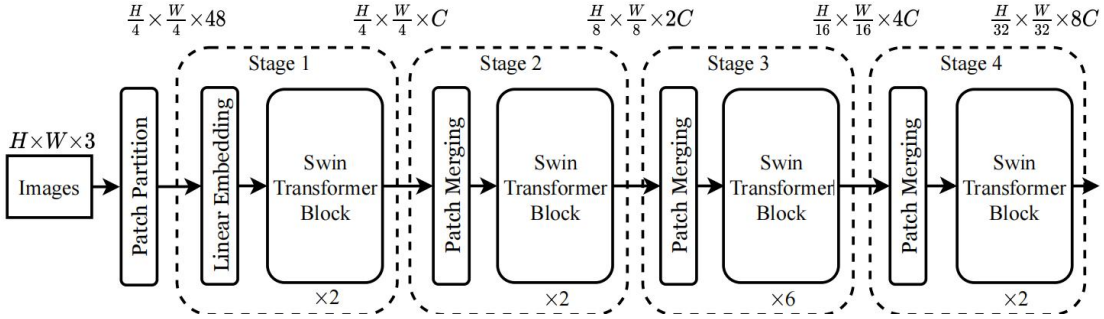


图 3.1 Swin Transformer 网络结构图

第一节 Swin Transformer 模型基本结构

通过研究原论文的模型结构，首先，将输入的 RGB 图像大小为 H×W×3 的图像进行分块，每个 4×4 像素的相邻区域称为一个 Patch，并在通道方向进行展平。每个 Patch 包含 16 个像素，每个像素有 3 个通道，因此展平后的值为 48。这样，经过 Patch Partition 后，图像的形状从[H,W,3]变为[H/4, W/4, 48]。接着，通过线性嵌入层对每个像素的通道数据进行线性变换，将 48 个通道减少为 C 个通道，从而将图像的形状从[H/4, W/4, 48]变为[H/4, W/4, C]。

随后，通过 4 个阶段构建不同大小的特征图。第一个阶段通过线性嵌入层开始，而其余三个阶段通过 Patch Merging 层进行下采样。最后，通过堆叠两种不同结构的 Swin Transformer Block 来完成处理。这两种结构分别是 W-MSA 结构和 SW-MSA 结构，它们成对使用，先使用一个 W-MSA 结构再使用一个 SW-MSA 结构。

最后对于分类网络，后面接上一个 Layer Norm 层、全局池化层以及全连接层得到最终输出，这些都是在源码中的实现方法。

下面对这个模型中的关键模块进行详细的研究：

1. Patch Merging：该模块的作用是在每个 Stage 开始前做下采样，用于缩小分辨

率，调整通道数，进而形成层次化的设计，同时节省一定的运算量。由于每次下采样两倍，在行和列的方向上间隔 2 选取元素，然后拼接在一起作为整个向量：

① x = x.view(B, H, W, C)：将输入特征向量重塑为四维向量，形状为 [B, H, W, C]，其中 B 是批大小，H 和 W 是输入特征的高度和宽度，C 是通道数。

② x0 = x[:, 0::2, 0::2, :]、x1 = x[:, 1::2, 0::2, :]、x2 = x[:, 0::2, 1::2, :]、x3 = x[:, 1::2, 1::2, :]：分别从输入特征向量中提取每个 2x2 区域的 4 个像素位置的特征，形状均为 H/2*W/2*C。

③ x = torch.cat([x0, x1, x2, x3], -1)：将四个特征向量沿着通道维度进行拼接，通道维度方向拉长，得到形状为 H/2*W/2*4C 的向量，相当于用空间上的维度换了更多的通道数。

④ x = x.view(B, -1, 4*C)：将拼接后的向量展平为二维向量，形状为[B, H/2*W/2, 4 * C]，这就是合并后最终得到的特征向量。

⑤ 上述过程将原始张量进行了尺寸和通道数的减少，类似于卷积神经网络中的池化操作。然而，与池化操作不同的是，在降维后通道数仅翻倍而不是增加四倍。为了与卷积神经网络保持一致，通过在通道维度上进行一次卷积操作，将通道数减半，变为原来的两倍。这一步将原始的 H*W*C 张量转换为 H/2*W/2*2C 的张量，空间大小减半，通道数翻倍，使其与卷积神经网络相对应。综上，每经过一个 Stage，下采样率两倍，H 和 W 都会减半，通道数翻倍。

2. Window Attention 和 Shifted Window Attention：传统的 Transformer 都是基于全局来计算注意力的，因此计算复杂度十分高。而 Swin Transformer 则将注意力的计算限制在每个窗口内，进而减少了计算量。基于窗口计算注意力的方式（W-MSA（Windows Multi-head Self-Attention））虽然可以很好地解决内存和计算量的问题，但是窗口和窗口之间没有通信，这样就达不到全局建模的效果，也就会限制模型的能力，所以文章中提出了一种移动窗口（Shifted Window）的方式，也就是把原来的窗口往右下角移动一半窗口的距离，如果 Transformer 是上下两层连着做这种操作，先是 window 再是 shifted window 的话，就能起到窗口和窗口之间互相通信的目的。

3. Transformer Block：每次输入进来先做一次 Layernorm 和基于窗口的多头自注意力，然后再做一次 Layernorm 和 MLP，第一个 block 就结束了；接下来是基于

移动窗口的多头自注意力，最后同样通过 MLP 得到输出。两个 block 结合起来才是 Swin Transformer 的一个基本的计算单元，因此也就解释了为什么 Stage1，2，3，4 中的 block 是 depths=(2, 2, 6, 2)这样安排的，即无论有多少层 block，其中的 block 的数字都是偶数，因为他始终需要两层的 block 连在一起作为一个基本单元，所以一定是 2 的倍数。

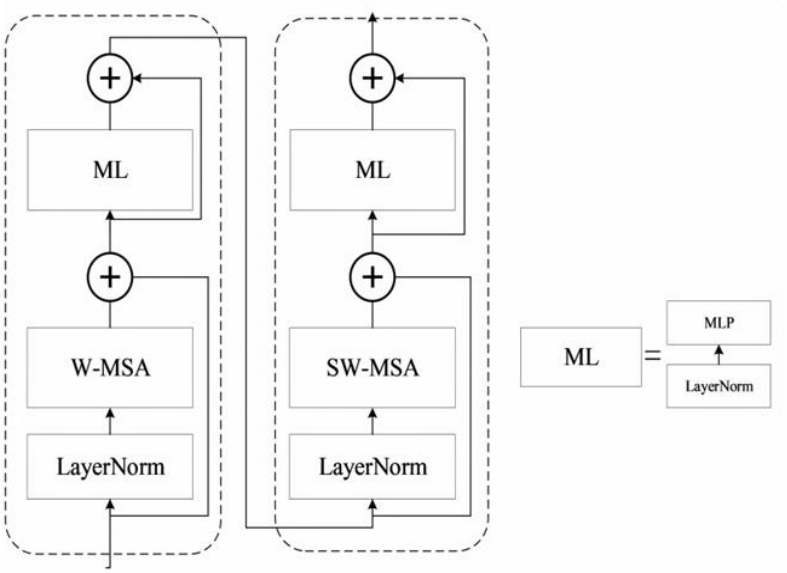


图 3.2　Swin Transformer Block 网络结构图

第二节 Swin Transformer 重要原理

3.2.1　SW-MSA 与 mask 的实现

首先在原论文中，为了解释窗口机制的引入，笔者需要对 W-MSA 和 MSA 的计算复杂度在维度上进行一个估计：

$$\Omega(\mathrm{W\text{-}MSA}) = 4hwC^2 + 2M^2hwC \tag{3.1}$$

$$\Omega(\mathrm{MSA}) = 4hwC^2 + 2(hw)^2C \tag{3.2}$$

其中，公式（3.1）对应的是标准的多头自注意力的计算复杂度，每一个图片有 h*w 个 patch，C 是特征的维度；公式（3.2）对应的是基于窗口的自注意力计算的复杂度，注意每个窗口并不是最小的计算单元，这里的 M 就是一个窗口的某条边上有多少个 patch，所以整个窗口中就有 M*M 个 patch。综上，前者的复杂度和 patch 的数量成平方关系，后者在 M 固定时则与 patch 呈线性关系，对

于较大的 h 和 w，基于全局的自注意力计算通常难以承受，而基于窗口的自注意力所需的计算就会减少很多。

如图 3.3 所示，如果仅仅是使用左图所示的窗口划分方法，那就仅仅计算到了块内的自注意力，而忽略了和他相邻的其他块中的信息，为了解决这个问题，Swin Transformer 团队引入了 Shifted Windows Multi-Head Self-Attention（SW-MSA）模块，也就是加上偏移的 W-MSA。

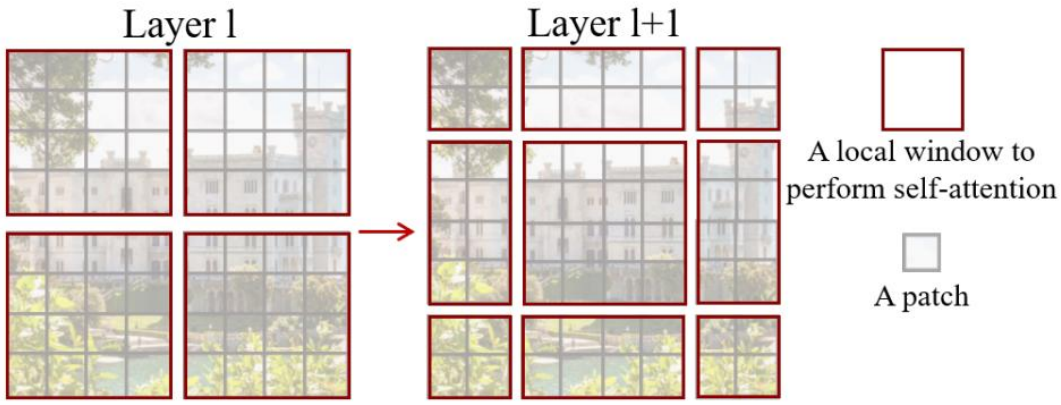


图 3.3 shifted window 中计算 self-attention 的示意图

具体操作就是先将窗口从左上角向右下各偏移$\left\lfloor\frac{M}{2}\right\rfloor$个像素得到右侧的 Layerl+1 层，通过这样的操作，就使得多个窗口间的信息得到了交流，其中的 C 窗口沟通了原来上面两个窗口的信息，而中间的 4*4 的窗口沟通了 4 个窗口的信息。但进行了窗口的偏移后由原来的 4 个窗口变成了 9 个窗口，窗口数量的变多，显然会增加 MSA 的计算量，所以如图 3.4 是作者提出的一种在移位窗口配置中的更高效批处理算法：

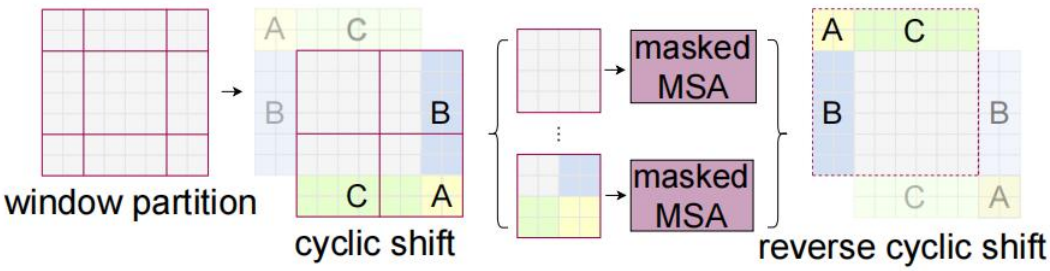


图 3.4 在 shifted window 中计算 self-attention 的高效批处理方法

如图 3.4，先对窗口进行编号然后再翻转划分，首先将 A 和 C 两个窗口翻转到最下面，然后将 A 和 B 翻转到最右边，使用网格再次对窗口进行划分，划分

完成后，4 号仍是原来的窗口；过去的 5 号和 3 号合并、7 号和 1 号合并得到两个新的 4*4 的窗口；其余的 4 个小窗口合并，这样又得到与原来相同的 4 个窗口，得以保证计算量是一样的。

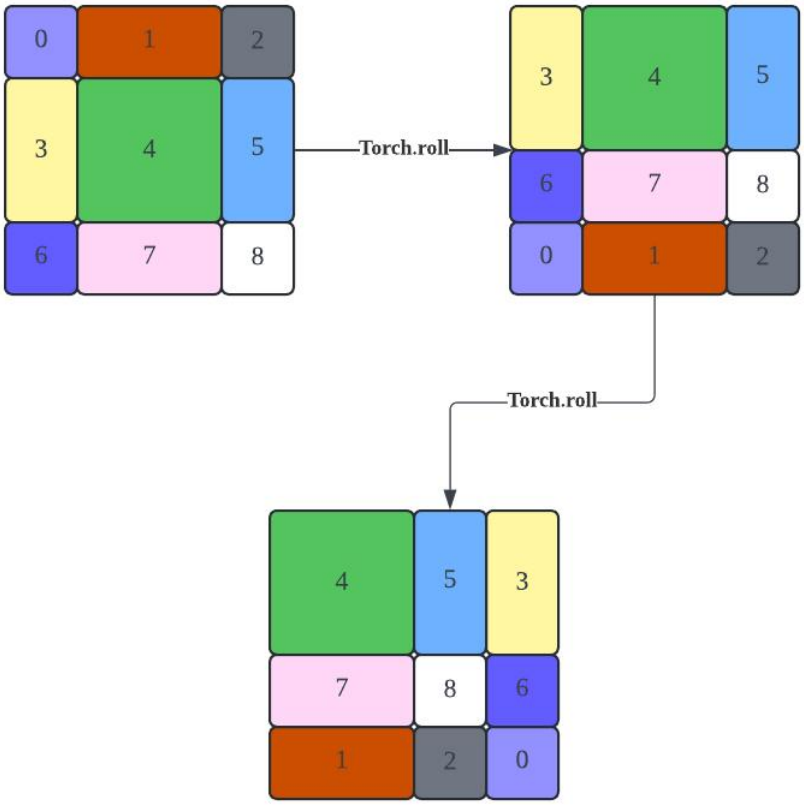


图 3.5 cyclic shift

但是这样子，就导致原本并不相邻的图片连在了一起，直接在窗口内计算自注意力，就会导致并不相关的信息反而产生了关系，并不是笔者想要的结果。所以模型使用的是带蒙版(mask)的 MSA，这样就能通过设置蒙版来隔绝不同区域的信息。具体来讲，为了保证 shifted window self-attention 计算的正确性，只能计算相同子窗口的 self-attention，不同子窗口的 self-attention 结果要归 0，在进行 cyclic shift 之前，需要给子窗口进行编码，如图 3.5，编码之后通过 torch.roll 对窗口进行滚动，在(5,3)(7,1)(8,6,2,0)组成的新窗口中，只有相同编码的部分才能计算 self-attention，不同编码位置间计算 self-attention 需要归 0，根据 self-attention 公式，最后需要进行 Softmax 操作，不同编码位置间计算的 self-attention 结果通过 mask 加上-100，在 Softmax 计算过程中，Softmax(-100)无限趋近于 0，达到归

0 的效果，所以对于像素 0 而言，实际上还是只和区域 5 内的像素进行 MSA，在计算完成后还要把数据挪回到原来的位置。

通过设置合理的 mask，让 Shifted Window Attention 在与 Window Attention 相同的窗口个数下，达到等价的计算结果。在模型实现的过程中，Mask 是初始就给定的，即只有特征发生滚动去迎合 Mask。

```
def create_mask(self, x, H, W):
# calculate attention mask for SW-MSA
# 1.先计算输入图像的高度 Hp 和宽度 Wp，使其能够被 window_size 整除，然后创建一
 #个形状为(1,Hp,Wp,1)的全零张量 img_mask，用于存储图像的掩码信息。
Hp = int(np.ceil(H / self.window_size)) * self.window_size
Wp = int(np.ceil(W / self.window_size)) * self.window_size
#  拥有和 feature map 一样的通道排列顺序，方便后续 window_partition
img_mask = torch.zeros((1, Hp, Wp, 1), device=x.device)  # [1, Hp, Wp, 1]
# 2.定义两个切片对象 h_slices 和 w_slices，分别表示高度和宽度上的切片范围。这些切
 #片范围将被用于将图像掩码赋值给 img_mask 张量的不同部分。
h_slices = (slice(0, -self.window_size),
            slice(-self.window_size, -self.shift_size),
            slice(-self.shift_size, None))
w_slices = (slice(0, -self.window_size),
            slice(-self.window_size, -self.shift_size),
            slice(-self.shift_size, None))
# 3.使用两个嵌套的循环遍历 h_slices 和 w_slices 中的切片范围，并为每个切片位置赋
#予一个递增的值 cnt，这样就可以将相应的索引值赋给 img_mask 张量，以表示每个窗
 #口的位置。
cnt = 0
for h in h_slices:
    for w in w_slices:
        img_mask[:, h, w, :] = cnt
        cnt += 1
 # 4.使用 window_patition 函数将 img_mask 张量划分为大小为 window_size 的窗口，得
到  #形状为(nW,window_size,window_size,1)的张量 mask_windows。将 mask_windows 重塑
#为形状为(nW,window_size*window_size)的二维张量。
mask_windows = window_partition(img_mask, self.window_size)
mask_windows = mask_windows.view(-1, self.window_size * self.window_size)
#5.通过计算 mask_windows 与自身转置之间的差异来计算注意力掩码(attn_mask)，将
 #attn_mask 中不为零的位置用-100.0 来替换，将为零的位置用 0.0 替换。这样可以为注
#意力机制建立一个掩码，以自注意力计算的过程中过滤不相关的位置。
attn_mask = mask_windows.unsqueeze(1) - mask_windows.unsqueeze(2)
attn_mask = attn_mask.masked_fill(attn_mask != 0, float(-100.0)).masked_fill(attn_mask ==
0, float(0.0))
```

3.2.2　相对位置编码

经 Swin Transformer 团队实验，发现 Swin-T 网络在 Attention 计算中引入相对位置偏置机制，其准确率能够提高 1.2%~2.3%不等。

下面通过源码来具体了解其实现原理：

1．定义一个相对位置偏置参数表：

```
self.relative_position_bias_table = nn.Parameter(
torch.zeros((2 * window_size[0] - 1) * (2 * window_size[1] - 1), num_heads))
# [2*Mh-1 * 2*Mw-1, nH]
```

2．定义不参与网络学习的变量的相对位置索引，首先使用 torch.arange 函数创建两个张量 coords_h 和 coords_w，分别表示窗口大小的高度和宽度范围内的坐标：

```
coords_h = torch.arange(self.window_size[0])
 coords_w = torch.arange(self.window_size[1])
```

然后，利用 torch.meshgrid 函数将 coords_h 和 coords_w 组合起来，生成一个二维坐标矩阵 coords，其中包含了所有可能的坐标组合。coords 的 shape 是 (2,Wh,Ww)，其中 Wh 和 Ww 分别代表窗口的高度和宽度的大小：

```
coords = torch.stack(torch.meshgrid([coords_h, coords_w]))
```

3．接下来，通过 torch.flatten 函数将 coords 展平为一维张量 coords_flatten，shape 为 (2, Wh*Ww)，表示所有坐标点的组合：

```
coords_flatten = torch.flatten(coords, 1)
```

计算相对坐标 relative_coords，这里通过广播操作实现了两两坐标之间的相对位置计算。计算结果的 shape 是 (2, Wh*Ww, Wh*Ww)，因为对每个坐标点，需要计算它与所有其他坐标点的相对位置：

```
relative_coords = coords_flatten[:, :, None] - coords_flatten[:, None, :]
relative_coords = relative_coords.permute(1, 2, 0).contiguous()
```

根据 Swin Transformer 中相对位置索引的计算规则，通过对相对坐标进行数据变换和求和操作，得到最终的相对位置索引矩阵 relative_position_index，shape 为 (Wh*Ww, Wh*Ww)。

```
relative_coords[:, :, 0] *= 2 * self.window_size[1] - 1
relative_position_index = relative_coords.sum(-1)
```

4．最后，通过 self.register_buffer 将计算得到的相对位置索引矩阵 relative_position_index 注册为网络的缓冲区（buffer），以便在网络的前向传播过程中进行使用。

```
self.register_buffer("relative_position_index", relative_position_index)
```

5．在前向计算中，相对位置偏置（relative_position_bias）被计算为相对位置索引矩阵（relative_position_index）对应的预先计算好的相对位置偏置表格（relative_position_bias_table）中的值：

首先，利用 relative_position_index.view(-1) 将相对位置索引矩阵展平为一维张量，并使用它来索引相对位置偏置表格 relative_position_bias_table 中的值。这样得到的 relative_position_bias 是一个形状为 (window_size*window_size, window_size*window_size, nhead) 的三维张量。

接着，通过 permute(2, 0, 1) 操作对 relative_position_bias 进行维度交换，将 nhead 维度移动到最前面，得到新的相对位置偏置张量。

```
relative_position_bias =
self.relative_position_bias_table[self.relative_position_index.view(-1)].view(self.window_size[0]
* self.window_size[1], self.window_size[0] * self.window_size[1], -1)
relative_position_bias = relative_position_bias.permute(2, 0, 1).contiguous()
```

第四章　基于 Swin Transformer 模型的文档图像分类实验

第一节 实验环境

本文的算法主要是依托于深度学习主流开源框架 PyTorch，Python 版本是 Python3.11.8，初始学习率设置为 0.001，epoch 设置为 100，学习率调整算法使用的是余弦退火学习率调整器，并通过租用云服务器来完成实验。

表 4.1 实验环境及参数设置

配置	型号	参数设置	内容
操作系统	Windows 11	迭代次数	100
CPU	Intel(R) Xeon(R) CPU E5-2673 v3	批次值	16
GPU	4090-24G	学习率	0.001
Python 版本	3.11.8	损失函数	CrossEntropyLoss
CUDA 版本	12.1	类别数目	10

第二节 数据集来源和划分

本次实验的数据集来源是近代的 300 本书籍，排除了一些数量过少的类别，最后得到 10 个分类，图像尺寸基本分布在 1300×2000 像素左右，相较于传统的公共数据集，这样的尺寸显然是更大的，也就意味着有更多的特征需要提取，更多的细节需要把握，对模型的能力提出了更高的要求。

下面是数据集的划分：

表 4.2 数据集划分

页面类别（10 个）	训练集 train/张	测试集 test/张	数据集总数/张
BodyPage_Annotation	410	121	531
BodyPage_Common	865	214	1079
BodyPage_ImageTitle	113	23	136

BodyPage_PageFooter	308	71	379
BodyPage_PageHeader	409	103	512
BodyPage_Table	802	191	993
BodyPage_TableTitle	244	45	289
CatalogPage	305	63	368
CopyrightPage	136	38	174
CoverPage	216	41	257

第三节 对比试验

作为科学实验的重要方法，对比试验一直以来都能给研究者带来丰富的实验结果，尤其是针对比较复杂的深度学习模型，大量的参数和层数的堆叠，让研究者对它的研究变得更加困难，所以本文需要借助对比试验的方法，来研究一些超参数以及其他可能影响模型预测结果的因素是否真的如笔者所想。但同时要注意的是，这样的实验结果也只能在一定程度上证明这些因素是有关联的，但可能也仅限于当前的实验环境和数据集，不是绝对的因果关系。下面是我进行的一些对比试验，希望可以对 Swin Transformer 模型的理解有一些帮助。

4.3.1　预处理操作对比

4.3.1.1　鲁棒性研究

在数据预处理的过程中，笔者尝试了多种的预处理方法，包括图像随机旋转±15°来增强对图像方向的鲁棒性，以及调整图像的亮度、对比度、饱和度和色调来增强模型对光照和颜色变化的鲁棒性。但经过实验证明，如图 4.1 可知，复杂的预处理方法有时会对图像进行过度变换，导致图像特征变得过于复杂或失真，使得模型最终不仅在准确率上有所下降，收敛的速度也大幅下滑，在经过长达 350 个 epoch 后仍没有达到很好的效果，所以过于复杂的预处理方法显然是不适用于本实验的。这种情况尤其可能发生在预处理方法与数据集特性不匹配时。例如：在本次实验中，数据集中包含的都是相同方向的书籍图片，对于其进行一些旋转值会增加特征提取的难度和准确率，色彩抖动可能使得图像颜色分布与原始

数据集不同，从而影响模型的学习。

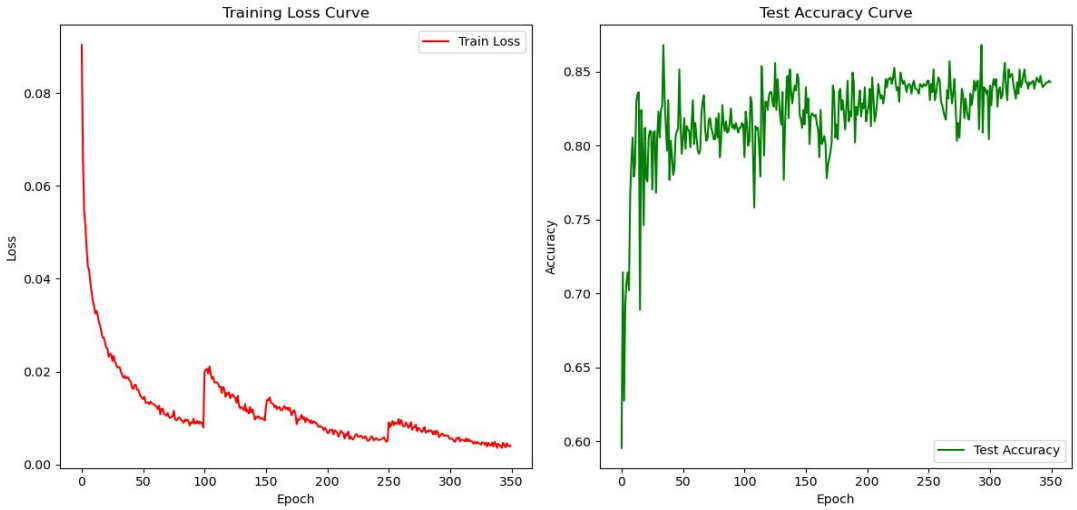


图 4.1　修改预处理方法后的曲线图

4.3.1.2　标准化参数研究

除此之外，标准化作为深度学习的图像处理任务中重要的数据预处理方法，若数据未经标准化就进入模型训练，可能导致训练过程中出现梯度消失或爆炸的问题，模型权重更新的步长可能过大或过小。而标准化过程通过调整数据的尺度使其符合特定的统计分布，输入数据的各个特征维度处于相同的尺度范围内，这有助于避免某些特征在学习过程中因尺度大而占主导地位，从而提高模型的稳定性和泛化能力，并有助于优化算法更快地收敛，减少内部协变量偏移。

因此针对标准化的参数均值和协方差，笔者进行了以下的实验进行研究，实验数据包括三种不同方法得来的参数：通用数据集 ImageNet 的均值和标准差、均设置为 0.5 以及根据本次实验的具体数据集来计算特有的均值和标准差。实验结果如下表：

表 4.3　参数设置和结果展示

参数	accuracy	epoch	备注
[0.5, 0.5, 0.5] [0.5, 0.5, 0.5]	85.16%	100	通用设置
[0.485, 0.456, 0.406] [0.229, 0.224, 0.225]	83.74%	100	ImageNet
[0.872, 0.872, 0.871] [0.220, 0.221, 0.220]	85.27%	150	计算得出

由上可知，在相同数据集下标准化的参数对于模型性能还是有明显的影响：

(1) 本次实验的数据集和 ImageNet 的特征显然不同，得到的结果不是特别理想；

(2) 具体计算得出的均值和标准差对于模型的准确率有所提高，但需要更多的计算资源达到这一效果，本次实验训练了 150 个 epoch 后 acc 才相对平稳；

(3) 最简单常见的参数设置得到的模型收敛较快，且效果与具体计算得出的结果相差并不明显。

综上，在之后的应用中，可以结合具体计算得来的参数和常见标准化参数进行对比，以期在收敛速度和准确率上达到一个平衡，使得模型性能更加优越。

4.3.1.3　图像尺寸研究

为了适应预处理模型所需的图像尺寸，在输入层需要对图像进行缩放和裁剪，笔者进行了多组不同尺寸的对比实验来研究这一处理对于模型性能的影响。

表 4.4 图像尺寸影响展示

尺寸	消耗时长/秒	Accuracy
512-384	9347	83.19%
448-384	9223	85.16%
384	9578	87.91%

由表 4.4 结果可知，不同的尺寸处理策略显然对模型来说是非常重要的，会在一定程度上影响模型所需的计算资源和分类准确率，由于模型需要的标准输入尺寸是 384*384，笔者采用三种不同的处理办法：先缩放至 512 或 448 的大小，再进行中心裁剪，保留关键特征；或是直接调整为 384*384，不进行裁剪以保留全部内容。

综上，如果最初设置的缩放尺寸过大，不仅会导致消耗的计算资源增加，且后续的裁剪过程会损失更多特征，降低分类的准确率；如果不进行任何裁剪而直接缩放，在保留更多的特征，得到更好的分类效果同时，也明显需要更多的计算资源，消耗更久的时长。

因此，经过所有以上实验，笔者最终保留了如下一些较为适合本次实验的预处理方法来处理图像输入，希望能够兼顾计算资源的消耗和分类的效果：

1. transforms.Resize(448)：将图像的大小调整为 448×448 像素，这样做可以统

一输入图像的尺寸，便于后续的处理。

2. transforms.CenterCrop(384)：对图像进行中心裁剪，裁剪出 384x384 像素的部分。这样做可以去除图像边缘的噪声，集中在图像的主要内容上。
3. transforms.RandomHorizontalFlip()：随机水平翻转图像。这样做可以增加数据的多样性，提高模型的泛化能力。
4. transforms.ToTensor()：将图像转换为向量格式，由于深度网络模型的输入往往是向量格式的数据，所以只有转化为向量才能用于神经网络的处理。
5. transforms.Normalize([0.5,0.5,0.5],[0.5,0.5,0.5])：对图像进行标准化处理，将图像的像素值缩放到[-1, 1]的范围内。这样做可以加速模型的收敛过程。

综上，图像的预处理方法并不是越复杂越好的，复杂的预处理方法虽然可能会提高模型的鲁棒性，但也意味着可能会引入不必要的复杂性和计算负担，丢失一些很重要的特征。因此我们需要综合考虑多方面的因素和数据集本身的特性来选择预处理的方法，而不是一味追求复杂度和难度，一个好的预处理方法应该能有效地清理、转换和准备数据，以便于模型学习和预测，同时保持数据的重要特征和结构。

4.3.2　数据集影响对比

4.3.2.1　数据集分布影响

前文讲过，原始数据集是具有显著特征的，区别于传统的数据集，在没有进行实验之前，不免会有一些想法，大量的正文页比例或许会影响模型的准确率，所以本文进行了一组对比试验来验证笔者的想法，（从 300 本近代书籍中提取出来的数目最多的 BodyPage_PageHeader 有 36082 张，为了加快实验的速度且不影响实验的最终结果，笔者对数据集进行了两次削减，第一次实验选择了大约一

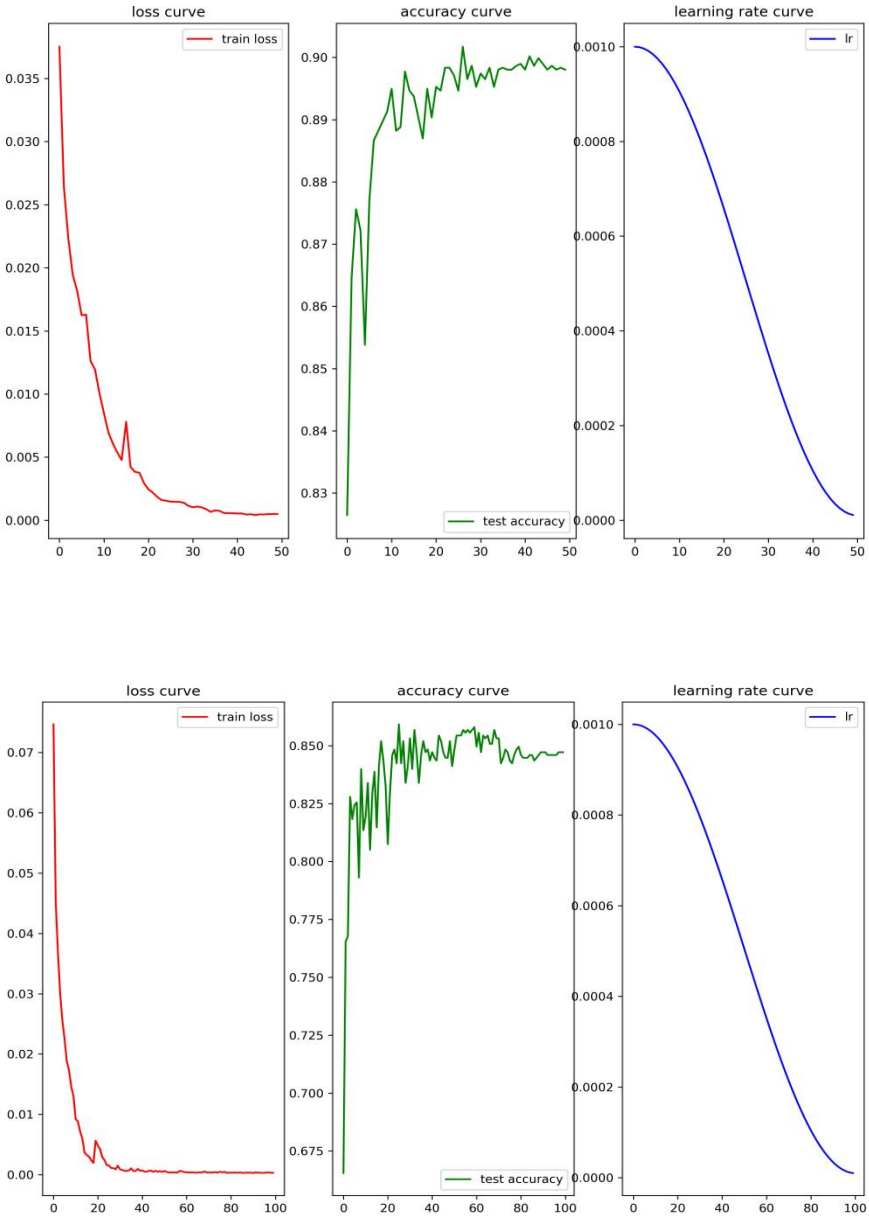


图 4.2 不均衡数据集和均衡数据集的 loss_acc_lr

万张图片，第二次和其余类别的数据作进一步的均衡，留下了 512 张图片作为最终的选择）如图 4.2 是我们具体的实验结果：

由上述的实验结果图 4.2 和表 4.5 可知，当数据集分布不均衡时，数量较大的类别（如：PageFooter 和 PageHeader 是我们数据集中数量最多的类别）往往会有更高的准确率，尤其 BodyPage_PageHeader 数量巨大且具有较高准确率，在将所有类别的准确率混合在一起计算时，将会很大程度上影响最终的结果，如左

图所示，在未削减数据集的情况下，仅需 50 个 epoch，准确率就可以迅速收敛至 89.80%左右，这看起来是非常优秀的结果；但实际上这样的情况并不能很好地反映出模型的真实性能，因为无论是在训练还是测试集中，其中不均衡的数据类别都会很容易得到正确的答案。而右边的结果就是笔者在削减特殊类别的图片数量后进行的二次实验，可以看到训练所需的 epoch 变多了，预测的准确率也出现了明显的下滑，但这才是模型的真实能力，明显更加客观。因此，在这样的结果上对模型进行优化才是有意义的，笔者希望的是模型具有很好的泛化能力，而不是一味追求在特定数据集上的高准确率。

为了验证这一结论，笔者使用 Tobacco-3482 公共数据集来做进一步实验，其中包含 10 个类别的文档图像，例如信件、表格、电子邮件、简历、备忘录等。共有 3482 张图像，作为公共数据集，类别分布更加均衡，分类也比较清晰，显然可以对于模型的性能有更清晰地了解，下面是实验的具体结果：

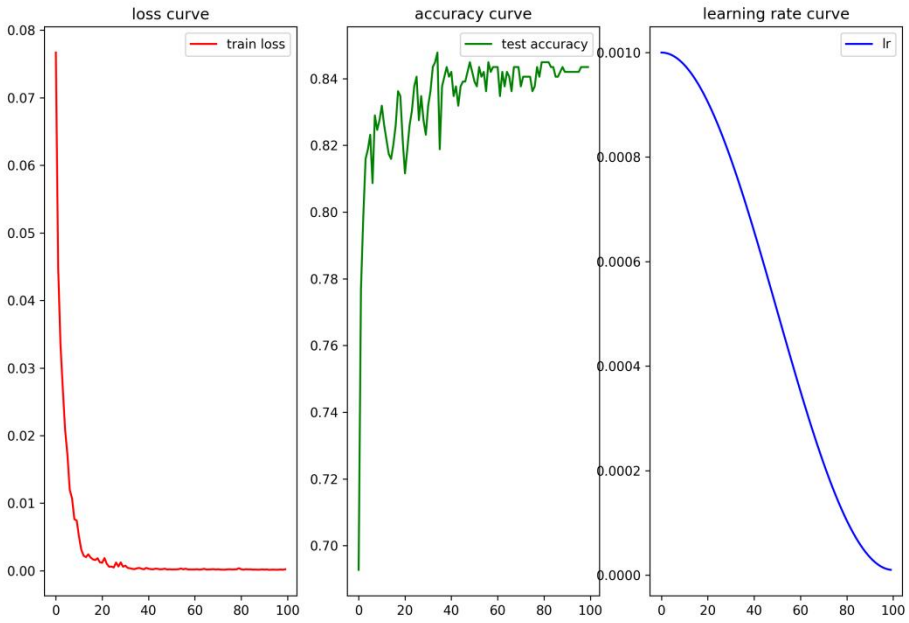


图 4.3 Tobacco-3482 分类效果

4.3.2.2　类别特征影响

除了数据集数量的影响，不同类别的特征提取难度也是影响结果的重要因素，如下表 4.5 中的 CoverPage 和 CtalogPage 两种类别，虽然数量并不占优势，但仍

可以在较小的训练集中得到很好的训练效果，这就在于封面和目录是两种非常有特点的页面类型，在特征提取过程中，不需要很丰富的细节就可以区别于其他类别，从而达到较好的分类效果。

与此相反，还有比较类似的页面类型（如 Table 和 TableTitle 这两个类别），彼此之间的特征差异较小，在 100 个轮次结束后分类效果并不理想，就需要更多的数据和 epoch 来完成训练，以得到较好的效果。在实验过程中，当训练结果显示某个或某几个类别的分类效果出现明显较低的情况时，可以冻结底层网络参数，加载特定类别的训练集来进一步对参数进行微调，通过更多的轮次来寻找更好、更适合特定类别的权重。

```
# 冻结底层网络参数
for param in net.parameters():
    param.requires_grad = False
# 解冻最后一层分类层的参数
for param in net.head.parameters():
    param.requires_grad = True
```

表 4.5 部分类别准确率展示

类别	准确率	类别	准确率
BodyPage_PageFooter	99.54%	CatalogPage	89.38%
BodyPage_PageHeader	92.84%	BodyPage_Annotation	90.53%
Image_Title	87.39%	CoverPage	95.17%

4.3.3　动态学习率和固定学习率对比

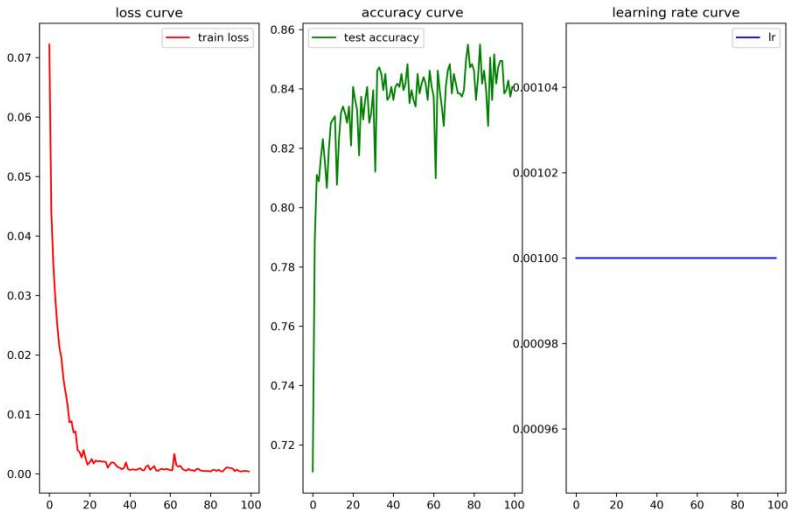
学习率[16]（Learning Rate）是神经网络中一个重要的超参数，控制着模型在每次参数调整时的步长大小，学习率设置的大小都会影响模型的训练过程和最终的性能。

如果学习率设置得过大，可能导致模型的更新过大，在参数空间中来回跳跃，出现震荡或不稳定的情况，更严重可能发生梯度爆炸（Gradient Exploding）和梯度消失（Gradient Vanishing）的问题，导致模型无法收敛到最优解。相反，如果

学习率设置得过小，模型的参数更新步长过小，可能使模型的收敛速度缓慢，不光会浪费更多的训练时间，还有可能导致模型陷入局部最优解或鞍点，更容易受到局部极小值的影响，同样会难以找到全局的最优解。

综上，合理的学习率调整策略可以帮助模型更快地收敛到最优解，节省训练资源，优化模型性能；于是笔者进行了一组对比试验，以研究动态调整学习率和固定学习率对于模型训练过程的影响，本次实验中选取的学习率调整算法是余弦退火学习率调整器，可以通过下面的函数表达式来简单了解其原理，其中的 x 是当前的训练周期数，T 是总训练周期数，args.lrf 是一个超参数，表示学习率的最低比例，即余弦函数的最小值，lr(x)是当前训练周期下的学习率比例：

$$\mathrm{lr}(x)=\left(\frac{1+\cos(\frac{x\pi}{T})}{2}\right)\times(1-\arg s.lrf)+\arg s.lrf. \tag{4.1}$$



下面是动态学习率调整的训练结果：

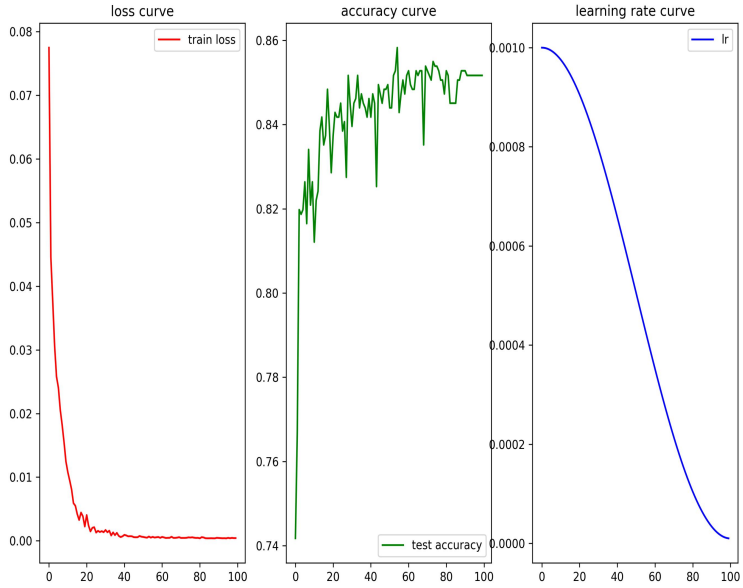


图 4.4 固定学习率和动态学习率 loss_acc_lr 曲线图

通过上述的loss_acc_lr曲线图可以看到，固定学习率使得模型在100个epoch内无法收敛至一个稳定的准确率，而动态调整学习率可以收敛至大约85.18%的准确率，因此动态调整学习率的方式对于模型的训练显然是起到积极作用的。

4.3.4　预训练模型对比

Swin Transformer 模型提出后，团队人员在各种公开的大规模数据集上都进行了相应的实验，并在 github 上传了训练过的一些模型来作为预训练模型，作为迁移学习[17]的基础用来优化后续模型的使用。

预训练模型通过在大规模的数据集上进行训练，学习到了丰富的数据特征，可以作为新任务的良好参数初始化，这样可以使得模型更容易地收敛到最优解，减少了在小规模数据集上训练时的过拟合风险；除此之外，还可以起到迁移学习的效果，可以将在大规模数据集上学习到的知识和特征迁移到目标任务中，通过在少量标注数据上进行微调，从而充分利用大规模数据集上收集到的信息，使得模型具有更强的泛化能力，可以适用于各种领域和任务，节省大量的训练时间和计算资源，提高模型的训练效率。

笔者本次实验使用的是在 ImageNet-22k 上进行图像分类的有监督的 Swin Transformer 预训练模型，ImageNet 作为一个计算机视觉系统识别项目，是目前

世界上最大的图像识别数据库。ImageNet 中目前共有 14197122 幅图像，总共分为 21841 个类别，作为一个超大规模的优质数据库，它已经成为用于训练和评估模型性能的基准。最终笔者选用的分别是 swin_base_patch4_window7_224_22k 和 swin_base_patch4_window12_384_22k 这两个模型，从名字上不难看出两个模型的 Patch 都是 4，window 大小为 7 和 12，输入图像大小分别是 224*224 像素和 384*384 像素。

Swin Transformer 团队提供了四个不同大小的模型以供使用，下面是不同模型的参数 C 和 Swin Transformer block 的层数信息：

$$\text{Swin}-\text{T}: \text{C}=96, \text{layer numbers}=\{2,2,6,2\}$$

$$\text{Swin}-\text{S}: \text{C}=96, \text{layer numbers}=\{2,2,18,2\}$$

$$\text{Swin}-\text{B}: \text{C}=128, \text{layer numbers}=\{2,2,18,2\}$$

$$\text{Swin}-\text{T}: \text{C}=192, \text{layer numbers}=\{2,2,18,2\}$$

下面是笔者实验的结果：

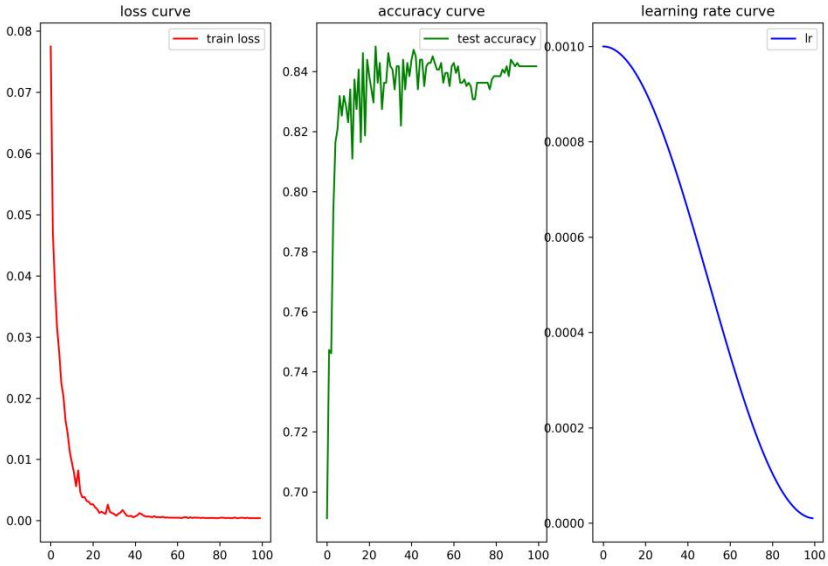


图 4.5 swin_base_patch4_window7_224_22k

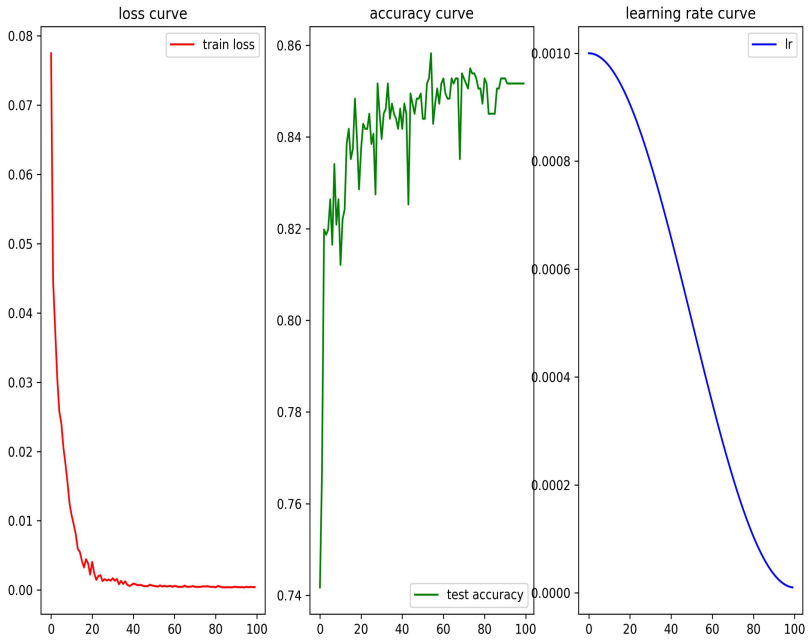


图 4.6 swin_base_patch4_window12_384_22k

由实验结果可知，保留更大尺寸的输入图像，选择更大的 window_size 对于本次大尺寸的图像数据集的实验来说，显然具有更好的训练效果，当预训练模型为 swin_base_patch4_window12_384_22k 时，实验准确率最终收敛于 85.16%，当预训练模型为 swin_base_patch4_window7_224_22k 时，模型最终收敛至 84.18%，二者相差了 1%，在深度学习的方法对比中，这已经足够说明问题。事实证明在本次的实验中，当 loss 都下降至稳定的情况下，较大输入尺寸的保留和窗口大小能够保留更多细节，提取到更丰富的特征，也就能够在测试集中有更好的表现，具有更好的分类性能，这样的实验结果在以后的应用中都将是重要的经验，也是选择预训练模型的重要参考。

4.3.5　优化器效果对比

在机器学习和深度学习中，优化器的核心作用是通过迭代更新模型的参数（如权重和偏置）来最小化损失函数。优化器根据损失函数的梯度调整参数，使用设定的学习率决定更新步长的大小。适当的学习率至关重要，过高可能导致跳过最优解，过低则可能导致训练速度过慢。高级优化器如 Adam，通过自适应学习率技术，针对不同参数自动调整学习率，改善训练速度和稳定性。此外，优化

器还考虑了如何提高收敛速度，例如通过动量项减少更新中的振荡，以及如何通过整合正则化策略帮助模型避免过拟合，从而在未见数据上也能保持良好性能。因此，选择合适的优化器对于训练有效的机器学习模型至关重要，可以显著影响模型的学习效率和最终的预测能力。

本次实验笔者选择了 SGD 和 Adam 两种比较常见的优化器进行对比，下面是调用优化器的过程：

```
optimizer = torch.optim.SGD(net.parameters(), lr=args.lr, momentum=0.9,
            weight_decay=1e-8)
optimizer = torch.optim.Adam(net.parameters(), lr=args.lr, betas=(0.9, 0.999),
            eps=1e-08, weight_decay=1e-4)
```

上述代码使用的都是 PyTorch 框架中的 torch.optim 库来设置优化器，用于训练本次实验中名为 net 的神经网络模型。这里详细解释下代码中的每个部分：

net.parameters()：这个调用返回网络 net 的所有可训练参数（如权重和偏置），这些参数将会被优化器更新。

lr=args.lr：这里设置了学习率 lr，它是优化器在更新参数时使用的步长。

momentum=0.9：动量系数设置为 0.9。动量是一个帮助加速 SGD 在相关方向上的并减缓振荡的方法。它在更新参数时考虑了前一次更新的方向，这可以使得优化过程更快地收敛，并且能够跨越一些小的局部最优。

weight_decay：权重衰减系数，这是一种正则化手段，用于防止过拟合。权重衰减通过在损失函数中加入一个与权重大小成比例的惩罚项来工作，这样可以限制模型的复杂度，使权重的值尽可能小。

betas=(0.9, 0.999)：这是一个元组，用于设置 Adam 优化器中的动量衰减率（beta1）和二阶矩估计的衰减率（beta2）。第一个值 0.9 控制梯度的一阶矩向量（类似于动量的概念），而第二个值 0.999 控制梯度的二阶矩向量的估计。这些值有助于平衡训练过程中的步长调整，提高优化器的稳定性和快速收敛。

二者的区别主要在于 SGD：标准的 SGD 仅依赖于当前批次的梯度来更新参数，通常使用固定的学习率。虽然可以通过添加动量等技术改进，但它本质上并不考虑历史梯度的信息。而 Adam 结合了梯度的一阶矩（均值）和二阶矩（未中心化的方差）估计来调整每个参数的学习率。这种自适应学习率的方法使得它

在面对各种数据和任务时更为灵活和有效。

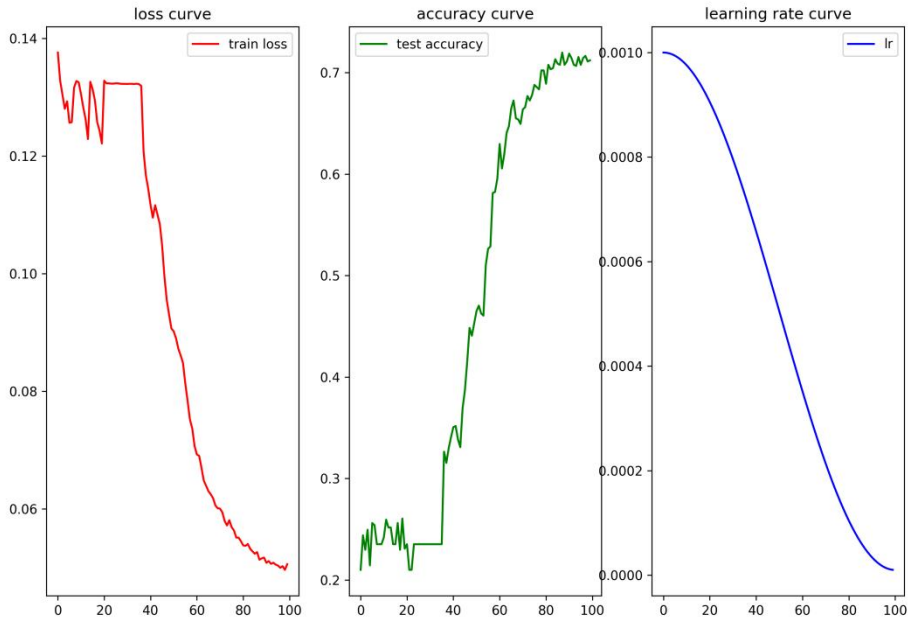


图 4.7 使用 Adam 优化器的曲线图

如图 4.7 是改用 Adam 优化器后的曲线图，由上可知，即使是比较通用的优化器，在特定数据集上也可能表现不佳，与使用 SGD 的优化器相比，模型的性能出现了显著下降，出现这种情况的原因可能有以下几点：

1．泛化能力的差异：多项研究表明，在某些任务和数据集上，尽管 Adam 可以加速初期的收敛，但 SGD 往往能达到更好的最终泛化性能。这可能是因为 Adam 的快速初期收敛让模型在较浅的局部最小值中过早稳定下来，而 SGD 由于更新速度较慢和波动性较大，有更多机会逃离这些局部最小值，最终可能找到更优的全局最小值。

2．梯度估计的差异：SGD 直接使用每个批次的梯度进行更新，而 Adam 使用的是对梯度的一阶和二阶矩的估计，这可能导致在特定的数据分布或者在梯度稀疏的情况下性能表现不佳。特别是在有噪声的数据或非平稳目标函数上，Adam 可能表现不如简单但稳定的 SGD。

3．超参数的敏感性：虽然 Adam 被认为在默认参数下通常表现良好，但实际上它对超参数（特别是学习率和 betas）的选择非常敏感。不当的参数设置可能导致性能下降，而 SGD 通常在较宽的参数范围内表现稳定。

4．优化策略与任务的匹配度：某些类型的网络架构或特定类型的学习任务可能

更适合某种优化策略。例如，SGD 在卷积神经网络和大规模图像分类任务中经常被证明比 Adam 更有效，可能是因为这些任务对梯度估计的准确性和更新策略的稳定性有更高的要求。

在经过对 Adam 优化器参数的一些调整后，笔者仍然没有得到较好的实验结果，因此在本次的图像分类任务中，SGD 优化器被证明是比较合适的优化手段，笔者最终决定在本次实验中使用 SGD 来完成后续的图像分类任务。

第四节 本章小结

本章对于 Swin Transformer 模型进行了简单的介绍，并对其中比较重要的模块做更详细的研究学习，作为如今图像分类任务的骨干网络，这样的研究是必要的。然后通过对不同预处理方法、数据集的均衡情况、学习率调整策略、预训练模型和优化器等多个对比试验，来研究不同因素对实验模型的准确率影响，通过上述实验，笔者对该模型以及深度学习相关知识都有了更充分的理解，得到的结果都将是之后深度学习模型参数调整过程中的重要参考，对于以后多场景下的应用都有非同寻常的意义。

第五章　总结与展望

第一节 主要工作

本文的选题为文档图像分类方法研究与分析，这也是当下比较热门的图像分类深度学习模型的一个具体应用。本文首先介绍了图像处理的相关知识，涉及图像分类方法、图像预处理算法、常见的图像特征提取算法和深度学习的基础知识；之后对于我们本次实验的骨干网络 Swin Transformer 的基本框架以及其中的关键模块做一定的研究；最后就是对比实验的具体内容，通过多组的对比试验，结合控制变量法，来研究图像预处理操作、数据集、学习率、预训练模型和优化器对于模型预测效果的影响程度，以此进一步了解 Swin Transformer 在实际图像分类任务中应该注意的地方和可以调整的参数，从而使它更好地服务于我们的生活。

第二节 展望

在结束了上述实验后，笔者已经对于图像分类任务以及 Swin Transformer 模型有了更深的理解，原论文中提出的移动窗口机制和局部位置编码让后来的研究者们都受益匪浅，也进一步理解数据预处理、数据集本身、学习率的调整、预训练模型和优化率的选取究竟是如何影响模型的性能。

本文所做的文档图像分类方法的研究，通过采用先进的 Swin Transformer 模型，不仅提高了分类精度，也优化了处理速度。在实际工程应用中，该方法可广泛应用于数字档案管理、智能文档处理和自动化办公系统等领域，帮助企业和机构提高文档管理的效率和准确性。

随着云计算和大数据技术的发展，本方法可进一步整合至云服务中，提供 API 接口，方便用户通过网络进行大规模文档图像的分类与处理，扩大其应用范围和市场潜力。

此外，在本次实验中，通过对比不同预处理方法、学习率调整策略以及不同预训练模型的性能，本研究为未来文档图像分类模型的优化提供了实验依据和方向。特别是在处理非平衡数据集和大尺寸图像方面，本研究的发现将指导后续研

究如何选择合适的方法和参数配置以提高模型的泛化能力和准确性。

实验结果也表明，引入 Swin Transformer 模型能有效提高文档图像分类的准确率，未来的研究可以在此基础上探索模型结构的进一步优化，例如通过增加层数、调整窗口大小或优化注意力机制等方式，来应对更复杂或多样化的文档图像分类任务。

具体来讲，针对 Swin Transformer 模型和其在实际生活中的应用，还有很多可以改进和优化的地方：

模型优化：针对 Swin Transformer 模型的局限性和不足之处，可以进一步进行模型优化和改进。例如，设计更加高效的注意力机制、更有效的参数初始化方法、更精细的模型剪枝策略等，以提高模型的性能和泛化能力。

多任务学习：将 Swin Transformer 应用于多任务学习场景，探索其在图像分类、目标检测、图像分割等不同任务中的应用。可以设计多任务联合训练的网络结构，充分利用模型的表示能力和泛化能力，提高多任务学习的效果。

跨领域应用：将 Swin Transformer 应用于其他领域和应用场景，如自然语言处理、语音识别、医疗影像分析等。可以根据不同领域的特点和需求，设计适用于 Swin Transformer 的新型任务和模型结构，拓展其应用范围。

模型解释性：提高 Swin Transformer 模型的解释性和可解释性，探索其内部机制和决策过程，从而更好地理解模型的工作原理和预测结果。可以设计可视化方法、注意力热力图等技术，帮助解释模型的预测结果和推理过程。

硬件加速：结合硬件加速技术，如 GPU、TPU 等，进一步优化 Swin Transformer 模型的训练和推理速度，提高模型的效率和实用性。可以探索模型在不同硬件平台上的部署和优化方法，实现更快速、更高效的模型运行。

参考文献

[1] 田永林,王雨桐,王建功,等.视觉 Transformer 研究的关键问题:现状及展望[J].自动化学报,2022,48(04):957-979.DOI:10.16383/j.aas.c220027.

[2] 郑远攀,李广阳,李晔.深度学习在图像识别中的应用研究综述[J].计算机工程与应用,2019,55(12):20-36.

[3] Liu, Ze et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows.” 2021 IEEE/CVF International Conference on Computer Vision (ICCV) (2021): 9992-10002.

[4] Dosovitskiy, Alexey et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.” ArXiv abs/2010.11929 (2020): n. pag.

[5] J. Deng, Li Fei-Fei et al. "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[6] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[7] Simonyan, Karen and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition.” CoRR abs/1409.1556 (2014): n. pag.

[8] He, Kaiming et al. “Deep Residual Learning for Image Recognition.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 770-778.

[9] 邱锡鹏，神经网络与深度学习，机械工业出版社，https://nndl.github.io/, 2020.

[10] Bahdanau, Dzmitry et al. “Neural Machine Translation by Jointly Learning to Align and Translate.” CoRR abs/1409.0473 (2014): n. pag.

[11] 朱张莉,饶元,吴渊,等.注意力机制在深度学习中的研究进展[J].中文信息学报,2019,33(06):1-11.

[12] 周妮娜,王敏,黄心汉,等.车牌字符识别的预处理算法[J].计算机工程与应用,2003(15):220-221+232.

[13] 常昌. 图像特征提取方法研究及应用[D].华中科技大学,2011.

[14] 陈成,耿晓中,刘柏进,等.一种改进的 Swin Transformer 图像分类识别方法[J].软件工程,2024,27(01):19-22.DOI:10.19644/j.cnki.issn2096-1472.2024.001.005.

[15] 赵蔷.主成分分析方法综述[J].软件工程,2016,19(06):1-3.

[16] 蒋文斌,彭晶,叶阁焰.深度学习自适应学习率算法研究[J].华中科技大学学报(自然科学版),2019,47(05):79-83.DOI:10.13245/j.hust.190515.

[17] 石祥滨,房雪键,张德园,等.基于深度学习混合模型迁移学习的图像分类[J].系统仿真学报,2016,28(01):167-173+182.DOI:10.16182/j.cnki.joss.2016.01.023.

致 谢

时光荏苒，四年的大学生活如白驹过隙般转瞬即逝。站在这段旅程的终点，回首过去，我心中充满了感激与感动。在这段求学的旅程中，我不仅收获了知识和技能，更收获了珍贵的友谊和宝贵的人生经验。在此，我想借此机会，向那些在我成长道路上给予帮助和支持的人们致以最诚挚的谢意。

首先，最深的感激献给我的指导老师张玉志教授。感谢您在这段毕业设计的过程中给予我无尽的耐心与指导。从课题的选择、方案的设计到论文的撰写，每一个环节都凝结着您的心血与智慧。每一次的讨论与交流，不仅让我在专业知识上得到了提升，更让我学会了如何严谨治学，如何面对困难与挑战。您的教诲将成为我今后人生道路上宝贵的财富。

其次，感谢我的父母和家人。你们是我坚强的后盾，是我追逐梦想的动力源泉。在我遇到困难与挫折时，你们总是给予我无条件地支持与鼓励。你们的爱与关怀，化作了我前行路上的光芒，让我在风雨中依然能够坚定地迈步前行。

我要感谢我的同学和朋友们。四年的朝夕相处，我们一起走过了无数个日夜，共同面对学业的压力与生活的挑战。感谢你们在我迷茫时给予的指引，在我困惑时给予的启迪，在我失落时给予的安慰。

最后，我要感谢所有曾经帮助过我的老师和工作人员。你们在教学和管理工作中的辛勤付出，为我们创造了良好的学习环境和生活条件。你们的默默奉献，成就了我们的成长与进步。

毕业在即，离别的时刻终将到来。然而，我相信，未来的我们，无论身处何地，都会带着这份感恩与回忆，继续追逐梦想，勇敢前行。愿我们都能在未来的道路上，不忘初心，砥砺前行。

再次感谢所有曾经帮助和支持过我的人，感谢你们成为我大学生活中最美好的部分。愿你我在未来的日子里，都能在各自热爱的世界里闪闪发光。