

学年论文

(课程论文、课程设计)

题 目： 基于CNN的脱机手写汉字识别

作 者： 梁浩铂

所在学院： 计算机科学与技术学院

专业年级： 计算机21-1

指导教师： 程述立

职 称： 副教授

2024 年 5 月 31 日

新疆大学课程论文（设计）、学年论文评分表

题 目	基于 CNN 的脱机手写汉字识别				
作 者	梁浩铂	专业年级	计算机 21-1	指导教师	程述立
指导教师评语及评分建议	<p>该生积极完成《机器学习（专创）》课程作业并参与课程研讨，上课认真听讲，论文格式较为规范，图表清晰，文字表述通顺，用语符合专业要求，逻辑性较强，课程设计较为合理，论述过程较为详细。整体而言，论文结构科学合理，思路清晰，观点表达准确，语言流畅，论述方法清晰合理，参考文献与课程主题符合，论文内容与网络舆情分析课程内容相符。</p> <p>根据《机器学习（专创）》的课程要求，结合该生考勤和作业情况，平时成绩为：（ ）分，根据课程论文撰写质量和课程论文工作量，课程论文成绩评定为：（ ）分，综合成绩为平时成绩占 30%，课程论文占 70%，该生综合成绩评定为：（ ）分。</p> <p>根据以上意见，建议该生课程论文最终成绩等级为：（ ）。</p> <p>指导教师：</p> <p>2024 年 5 月 31 日</p>				
院（部）或教研室意见	<p>同意指导老师意见。</p> <p>学院或教研室主任：</p> <p>年 月 日</p>				

摘要

随着人工智能技术的飞速发展,手写汉字识别作为自然语言处理领域的一个重要分支,其研究和应用价值日益显著。本文提出了一种基于卷积神经网络(Convolutional Neural Network, CNN)的手写汉字识别方法,并采用 TensorFlow 2.0 框架进行模型的构建和训练。与常见的 MNIST 或 Fashion-MNIST 数据集相比,汉字识别任务面临更大的挑战,主要体现在其庞大的搜索空间和复杂的数据集处理需求。

汉字识别的数据集规模庞大,类别多达 3000+ 以上,这要求模型设计必须更加精细,同时数据处理的复杂性也显著增加。汉字手写字体识别数据集稀缺,且预处理难度大,对模型设计能力提出了更高的要求。此外,模型的复杂度需要精心平衡,以避免训练过程中的发散问题。

尽管存在诸多挑战,本项目通过深入研究和实践,成功实现了基于 CNN 的中文手写字识别系统。本系统采用 TensorFlow 2.0 API 构建,不仅能够处理复杂的汉字手写识别任务,而且经过精心设计,可以直接应用于工业场合,如票据识别和手写文本自动扫描等。与依赖外部 API 接口相比,本系统具有可优化性、免费性和本地性等显著优势。

本文的实验结果表明,所提出的模型在标准手写汉字数据集上取得了较高的识别准确率,与现有方法相比,具有更好的性能和鲁棒性。此外,本文还探讨了卷积神经网络在手写汉字识别中的一些关键问题,如网络结构的选择、超参数的调整以及模型的优化策略。

关键词: 卷积神经网络; 手写汉字识别; TensorFlow; 模型设计; 数据预处理

Abstract

With the rapid advancement of artificial intelligence technologies, handwritten Chinese character recognition, as a significant branch of natural language processing, has become increasingly prominent in research and application value. This paper presents a method for handwritten Chinese character recognition based on Convolutional Neural Networks (CNNs) and utilizes the TensorFlow 2.0 framework for model construction and training. Compared to common datasets such as MNIST or Fashion-MNIST, the task of Chinese character recognition faces greater challenges, mainly reflected in its vast search space and complex dataset processing requirements.

The scale of Chinese character recognition datasets is enormous, with categories exceeding 3000+, necessitating more refined model design and significantly increasing the complexity of data processing. Handwritten Chinese font recognition datasets are scarce, and the difficulty of preprocessing poses higher demands on model design capabilities. Moreover, the complexity of the model must be carefully balanced to avoid divergence issues during training.

Despite the numerous challenges, this project has successfully implemented a CNN-based Chinese handwriting recognition system through in-depth research and practice. The system, constructed using the TensorFlow 2.0 API, is not only capable of handling complex Chinese handwriting recognition tasks but also, after meticulous design, can be directly applied to industrial scenarios, such as bill recognition and automatic scanning of handwritten text. Compared to reliance on external API interfaces, this system offers significant advantages in terms of optimization, cost-free, and local deployment.

The experimental results of this paper demonstrate that the proposed model achieves a high recognition accuracy rate on standard handwritten Chinese character datasets, exhibiting better performance and robustness compared to existing methods. Additionally, this paper explores several key issues in the application of CNNs to handwritten Chinese character recognition, such as the selection of network architecture, adjustment of hyperparameters, and optimization strategies for the model.

Keywords: Convolutional Neural Networks; Handwritten Chinese Character Recognition; TensorFlow; Model Design; Data Preprocessing

目录

第一章 绪论	1
1.1 研究背景及意义	1
1.2 国内外研究现状	3
1.2.1 国内研究现状	3
1.2.2 国外研究现状	4
1.3 本文研究内容及章节安排	4
第二章 相关理论和技术介绍	6
2.1 神经网络概述	6
2.1.1 神经网络算法	6
2.1.2 神经网络架构	8
2.2 卷积神经网络	9
2.2.1 卷积神经网络的结构	9
2.2.2 卷积神经网络的特点	10
2.2.3 卷积神经网络的算法	11
2.3 特征提取	13
2.3.1 基于结构的特征提取方法	14
2.3.2 基于统计的特征提取方法	14
2.4 本章小结	15
第三章 基于CNN的手写汉字识别	16
3.1 数据准备	16
3.2 模型构建	17
3.2.1 构建模型	17
3.2.2 数据输入	18
3.3 模型训练	19
3.4 模型测试	20
3.5 本章小结	21
第四章 总结和展望	22
4.1 工作总结	22
4.2 展望未来	22
参考文献	24

第一章 绪论

1.1 研究背景及意义

手写汉字识别技术作为计算机视觉与模式识别领域的重要研究方向之一，其研究背景和意义随着信息技术的飞速发展而日益凸显。在数字化时代背景下，快速准确地将手写文字转换为电子数据，不仅能够提高信息处理的效率，还能促进文化传承与保护。

首先，汉字作为一种独特的文字系统，具有悠久的历史和丰富的文化内涵。从古代碑刻、书画到现代的文档记录，汉字承载了中华民族数千年的文明史。然而，随着时间的推移，许多珍贵的历史文献因保存条件恶劣而逐渐模糊或损毁，急需通过现代技术手段进行恢复和保存。手写汉字识别技术的应用，为这些珍贵文物的数字化保护提供了有效的解决方案。

手写汉字识别技术在日常生活和商业应用中展现出巨大的潜力。它在多个领域中能够显著提升工作效率并降低人力成本。例如，在银行业务中，自动处理支票；在法律行业，实现文书资料的数字化管理；以及在个人使用场景下，快速转换日常笔记，手写汉字识别技术都发挥着重要作用。

此外，随着人工智能技术的持续进步，手写汉字识别正逐渐成为促进人机交互发展的关键技术之一。它不仅提高了数据处理的自动化水平，还为人机之间的无缝交流提供了可能，进一步推动了智能系统在多样化场景中的应用和普及。

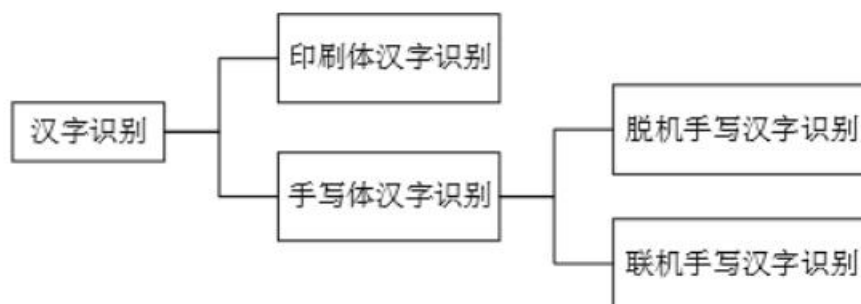


图 1-1 汉字识别分类

汉字识别技术依据识别对象的差异，主要划分为两大类别：首先是印刷汉

字的识别，其次是手写体汉字的识别。在手写体汉字识别领域，根据数据采集方式的不同，又进一步细分为脱机手写体汉字识别和联机手写体汉字识别两种形式，如图 1-1 所示。

联机手写体汉字识别技术涉及的是书写者利用数字笔、手写板或触摸屏等物理设备进行实时书写，在此过程中，文字信号被在线捕获并输入至计算机系统。相对而言，脱机手写体汉字识别技术处理的是那些通过扫描仪或摄像头等图像捕捉设备所获取的手写汉字的二维图像。

这种分类反映了汉字识别技术的多样性和适应性，同时也指出了不同技术路径的特定应用场景和需求。

手写体汉字的识别率一直未能达到实际应用水平，其难度被公认为模式识别领域的一大挑战。这主要表现在以下几个方面：

(1) 常用汉字的数量庞大，约有 3000 至 4000 个。一般而言，识别种类越多，分类任务就越为复杂和困难。

(2) 汉字的笔画数量通常较多，且组合方式多样，变化无穷。对于笔画稠密的汉字，如“龘”、“𪚩”和“𪚪”，书写过程中可能出现的问题会导致特征提取变得尤为困难。

(3) 手写体的形态与印刷体汉字的规整性不同，书写时的连笔以及风格迥异的书写方式（如行书、草书等）会导致汉字结构的变化。此外，每个人的书写习惯都有所不同，即使是同一个人在不同时间书写的汉字也会有所差异。

(4) 在众多汉字中，存在一些外形相似的字符，例如：“人”与“入”，“木”、“本”与“术”，“未”与“末”等。这些字符本身就具有高度的相似性，而手写的随意性进一步增加了计算机识别和特征提取的难度。

(5) 尽管全面的数据设计和训练模型是基础，但目前仍然缺乏权威且统一的手写体汉字数据库，这也是制约手写体汉字识别发展的一个重要因素。

深度神经网络技术的迅猛发展，显著提高了手写体汉字识别的准确度。特别是卷积神经网络（Convolutional Neural Network, CNN）作为基础网络架构，以及近年来兴起的注意力机制模型，为脱机手写体汉字识别的准确度提升做出了重要贡献。与传统手写体汉字识别方法相比，CNN 在特征提取方面具有显著优势，能够高效地捕捉汉字图像的关键特征，并展现出对不同书写风格和类型手写体汉

字的优良适应性。因此，基于 CNN 的识别方法在手写体汉字识别领域取得了较为显著的成效。

然而，由于 CNN 模型的多样性，不同的模型可能适用于不同的识别任务。因此，为了实现高效的手写体汉字识别，研究者需要深入学习 CNN 模型的理论基础，探究其高识别率的原因，并不断优化模型结构，以寻找最适合特定识别任务的 CNN 模型。

1.2 国内外研究现状

1.2.1 国内研究现状

随着计算机视觉和人工智能技术的快速发展，中国在手写体汉字识别研究领域已取得显著进展，正逐步缩小与国际先进水平的差距。国内学者和研究机构通过深入研究和创新，推动了该领域的技术进步。

在深度学习技术，尤其是卷积神经网络（CNN）的应用上，国内手写体汉字识别研究已取得突破性成果。研究者通过构建深层多层网络模型，有效提升了从复杂汉字图像中的特征提取能力，并实现了高效的分类与识别。此外，迁移学习、增强学习等深度学习技术也被集成应用于汉字识别任务，进一步提高了系统的准确性和鲁棒性。

针对复杂场景下的手写体汉字识别问题，国内研究人员开始探索多模态识别方法，整合笔迹动态信息、书写顺序及上下文关系等多维度数据，以期实现更为精准的综合判断和识别。

为了训练更为健壮的识别模型，国内多个研究机构和企业已建立大规模的手写体汉字数据集，这些数据集覆盖了常用汉字、生僻字和多种变形字体，为模型训练提供了丰富的样本资源。同时，对这些数据集的深入分析有助于研究者更准确地把握汉字的结构特点和识别难点。

深度学习模型在手写体汉字识别中的广泛应用，虽然带来了显著的性能提升，但也伴随着对计算资源的高需求。为了解决这一问题，国内研究者正在积极探索算法的硬件加速和优化策略。通过利用 GPU、TPU 等专用硬件加速设备，并结合模型压缩和优化技术，研究者们已经显著提高了模型在实际部署中的效率和性能。

手写体汉字识别技术的应用场景也在不断拓展。从最初的单一字符识别，到现在的整个文本处理流程，这项技术的适用性越来越广泛。目前，国内众多研究团队正将手写体汉字识别技术应用于电子签名验证、智能文档批改、自动字幕生成等多个领域，有效拓宽了其应用范围。

总体来看，国内在手写体汉字识别领域已经建立起一套涵盖理论研究、技术创新、应用开发等多方面的完善研究体系。展望未来，随着新技术的不断涌现和跨学科融合的深化，预计国内在手写体汉字识别领域的研究将取得更多突破性进展，为相关技术的发展和應用提供更加坚实的基础。

1.2.2 国外研究现状

在国外，手写体汉字识别技术的研究起步较早，特别是在日本和欧美国家，这一领域的研究较为活跃。早期的研究主要集中在印刷体汉字的识别上，随着技术的发展，手写体汉字识别逐渐成为研究的热点。

自 20 世纪 60 年代起，IBM 公司的 R. Casey 和 G. Nagy 便开始了印刷汉字的识别研究。随后，日本东芝公司在 1977 年开发出首个可以识别 2000 个不同汉字的系统。这些早期研究为后续的手写体汉字识别技术奠定了基础。进入 80 年代后，日文部分文字与汉字结构相似的特点使得日本在该领域取得显著成果。而联机手写汉字识别系统则由 IBM 公司在 1981 年设计出较为成熟的技术。

国外研究者在早期便开始布局，积累了丰富的经验，并不断尝试将各种先进算法和技术应用于汉字识别中。例如，深度可分离卷积的 CNN 结构和注意力机制等先进技术的应用，推动了该领域的技术进步和应用发展。此外，国外的研究机构和企业也在大规模数据集的建立、多模态识别方法的探索、硬件加速与优化等方面取得了显著成果。

当前，国外在手写体汉字识别领域的研究已经形成了一套较为完善的体系。从理论研究到技术创新，再到应用开发，各方面都取得了长足的进步。

1.3 本文研究内容及章节安排

本文包含四章内容，各章节内容如下：

第一章 绪论。本章旨在阐释手写汉字识别技术的研究背景、研究意义，并

进一步分析了手写汉字识别面临的主要挑战，包括汉字的多样性、结构复杂性以及书写风格的差异性。此外，本章还系统地回顾了国内外在手写汉字识别领域的研究进展，指出了现有研究的优势与不足。最后，本文概述了研究的主要内容，并对论文的结构安排进行了清晰的规划。

第二章 相关理论概述。本章详细介绍了卷积神经网络的理论基础，包括其在图像识别中的数学模型、工作原理以及关键的网络结构。同时，对特征提取方法进行了详尽的讨论，强调了这些方法在手写汉字识别中的重要性和应用效果。

第三章 基于 CNN 手写汉字识别。本章作为实验研究的核心部分，详细阐述了使用卷积神经网络进行手写体汉字识别的全过程。内容包括数据准备、模型构建、模型训练以及模型测试。每一部分都力求详尽，旨在展示实验的完整性和科学性，确保实验结果的可靠性和有效性。

第四章 总结与展望。本章对全文进行了综合性的总结，回顾了实验过程中存在的不足之处和遭遇的问题，并对研究中遇到的问题进行了批判性思考。最后，本章提出了未来研究的重点方向和可能的发展趋势，旨在为后续的研究工作提供参考和指导。

第二章 相关理论和技术介绍

前面章节主要阐述了手写汉字识别技术的研究背景、研究意义，并进一步分析了手写汉字识别面临的主要挑战，包括汉字的多样性、结构复杂性以及书写风格的差异性。本章节将主要讲述在进行手写汉字识别的过程中需要用到的理论和技术介绍。主要包括神经网络算法和架构，卷积神经网络的结构、特点以及具体的实现算法，以及特征提取的方法，这些理论和技术为基于卷积神经网络的脱机手写汉字识别的研究提供了坚实的理论支撑和关键技术保障。

2.1 神经网络概述

2.1.1 神经网络算法

神经网络（Neural Networks，简称 NNs）是一种模仿生物神经系统结构与功能构建的数学模型，其设计灵感来源于人类大脑的神经网络。在生物神经系统中，神经元能够存储信息并协同工作以处理复杂信息。类似地，人工神经网络由大量节点（或称为人工神经元）组成，这些节点通过加权的连接相互连接，形成一个高度复杂的网络结构。每个节点能够接收输入信号，并通过激活函数进行处理，以产生输出信号。

尽管单个神经元的结构相对简单，处理能力有限，但当众多神经元以网络形式组织起来时，它们能够执行高度复杂的计算和操作。如图 2-1 所示，神经元的模型结构被简化以便于数学建模和计算实现。在深度学习领域，神经网络的深度（即层数）是衡量其处理能力的关键因素，网络层数的增加通常意味着更强的处理能力和更高级的功能实现。

神经网络算法本质上是一种逻辑推理和演算过程，它通过模拟人类大脑的神经思维模式，将目标信息转换为计算机能够理解的语言符号，进而编写成计算机可执行的指令。这种算法能够对数据进行高效的特征提取和模式识别，从而在图像识别、语音处理、自然语言理解等多个领域展现出卓越的性能。随着深度学习

技术的不断进步，神经网络在解决复杂问题方面的能力也在不断增强，为人工智能的发展提供了强大的技术支撑。

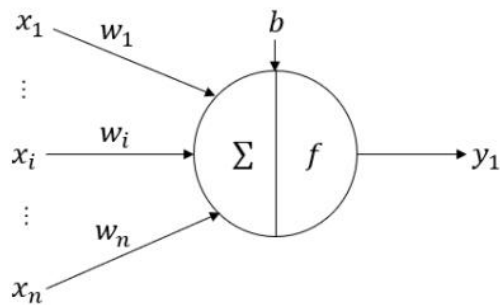


图 2-1 神经元的模型结构

在神经网络模型中常用到的算法有 K 最近邻算法（K-Nearest Neighbours, KNN）、批归一化算法（Batch Normalization, BN）、随机梯度下降算法（Stochastic Gradient Descent, SGD）、超图学习算法（Hypergraph Learning）、反向传播算法（Back Propagation, BP）等。如表 2-1 所示，对各算法进行对比。

表2-1 神经网络算法优缺点对比

神经网络算法	优点	缺点
K最近邻算法	无监督学习，最简单的神经网络算法，可以实现分类和回归。	运行速度低，数据集样本很大时，计算的时间会变长，效率变差。
批归一化算法	提高模型训练速度，增加模型稳定性，防止过拟合的正则化。	不能使用小批量进行训练，会在成均值和方差有很大的偏差。
随机梯度下降算法	可以提前获得损失值，将误差进行泛化。	计算量大，训练时间长，模型方差大，很难设定合适的学习率。
超图学习算法	通过节点特征相似性让网络自我进化，可以不断地迭代构建模型中不存在的关系属性。	需要为数据集构建简单的图，然后用简单图谱聚类作为基础。

表2-1 神经网络算法优缺点对比（续）

神经网络算法	优点	缺点
反向传播算法	可以将训练学习的结果应用于新的样本上，可以重新整理误差，提取更多的关键特征。	算法会出现局部极值，导致网络训练失败，收敛速度慢，对训练样本存在依赖。

2.1.2 神经网络架构

神经网络是一种模拟人脑神经系统的计算模型，它由大量的人工神经元（也称为节点或单元）通过复杂的连接模式组成。这些人工神经元分为两种类型：一种是输入神经元，负责接收来自外部环境的数据信号；另一种是输出神经元，负责将网络处理后的信息传递出去，形成最终的响应或决策。输入神经元和输出神经元之间，以及它们与隐藏层神经元的相互作用，共同构成了一个强大的信息处理和转换系统。

隐藏层神经元是神经网络中的中间处理单元，它们既不直接接收外部输入，也不直接输出信息，而是通过复杂的数学变换对输入数据进行处理，提取特征并逐步抽象和概括信息。这些隐藏层使得神经网络能够学习和模拟复杂的函数映射，从而解决诸如分类、回归、模式识别等任务。

神经网络的架构设计对其性能至关重要。一个典型的神经网络包括输入层、一个或多个隐藏层以及输出层。每一层由多个神经元组成，每两个相邻层之间的神经元通过权重连接。权重是神经网络学习过程中不断调整的参数，它们决定了信号在网络中的传递和转换方式。

如图 2-2 所示的神经网络基本架构图，清晰地展示了从输入层到输出层的数据流动和处理过程。

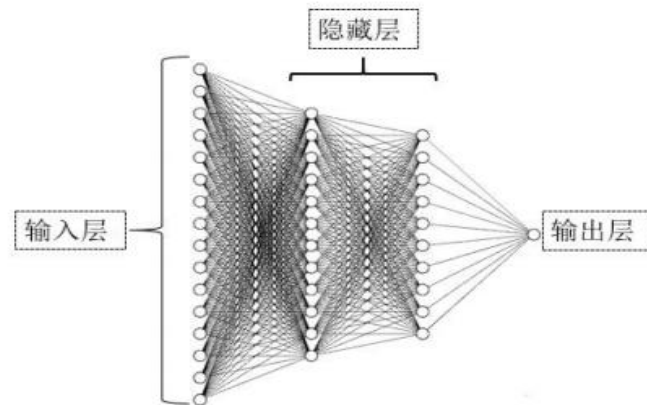


图 2-2 神经网络基本架构图

2.2 卷积神经网络

卷积神经网络（Convolutional Neural Network, CNN）是一种深度学习模型，是一种具有局部连接、全职共享等特性的神经网络，在图像和视频识别、分类以及相关视觉任务中表现出色。启发与生物学家对动物视觉皮层的研究，卷积神经网络使用“局部感受野”的方式处理输入图像，即卷积层中的神经元只接受其覆盖区域内的信号，这种特性极大地减少神经元之间连接权值的数量，降低了模型的复杂度。

2.2.1 卷积神经网络的结构

卷积神经网络一般由四层组成，分别是卷积层、激活层、池化层和全连接层，如图 2-3 所示，各层的主要功能如下：

卷积层：卷积神经网络的核心，通过使用一组可学习的滤波器（或称为卷积核）在输入数据上滑动，计算局部区域的特征图。卷积操作能够捕捉图像中的局部特征，并且由于卷积核的共享权重，网络能够以参数数量较少的方式学习到特征。

激活层：通常跟在卷积层之后，目的是引入非线性，使得网络能够学习更加复杂的函数映射。常用的激活函数有 Sigmoid 函数、Tanh 函数（双曲正切函数）、ReLU 函数等。

池化层：用于降低特征图的空间维度，从而减少参数数量和计算量。同时，池化操作也特高了特征检测的不变形，使得网络对输入图像的平移、缩放和旋转

等变化更加鲁棒。常见的池化操作有最大池化和平均池化。

全连接层：在网络的末端，全连接层将前面层中学到的特征映射到最终的输出。在全连接层中，每个神经元都与前一层的所有神经元相连，这使得网络能够进行最终的分类或其他任务。

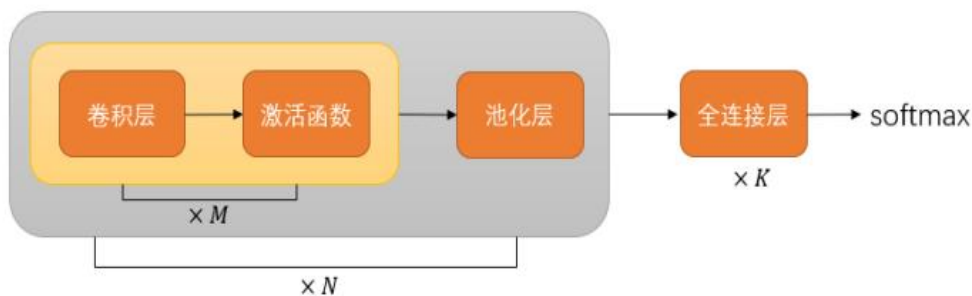


图 2-3 卷积神经网络结构示意图

2.2.2 卷积神经网络的特点

卷积神经网络（CNN）的核心优势在于其独特的局部连接和权值共享机制，这些特性显著减少了模型的计算复杂度和参数数量。

局部连接特性意味着在卷积层中，每个神经元只与输入数据的一个局部区域相连接，而非全连接。如图 2-4 所示，这种设计使得卷积层中的连接参数数量大幅减少。假设 N^m 为第 m 层的神经元格式， N^{m-1} 为第 $m-1$ 等神经元的个数， K 为卷积核的尺寸，则在全连接层中，总共需要 $N^m \times N^{m-1}$ 个连接参数，而在卷积层中，连接参数仅为 $N^m \times K$ ，一般情况下， K 远远小于 N^{m-1} 。

权值共享则体现在卷积层中卷积核的重复使用。在 CNN 中，卷积核的参数在整个网络中是共享的，每个卷积核在整个输入特征图上滑动，对不同的局部区域执行相同的卷积操作。这样，即使在不同的感受野区域，相同的卷积核也能够提取出相似的特征。这种设计不仅使得模型能够捕捉到局部特征，而且还能够识别出这些特征在不同位置的出现，极大地提高了模型的泛化能力。

通过局部连接和权值共享，CNN 能够有效地捕捉图像的局部特征并保持参数数量的精简，这使得 CNN 在图像识别和分类任务中表现出色，尤其是在处理具有复杂结构和丰富细节的图像时。这些特性共同构成了 CNN 在深度学习领域广泛应用的基础。

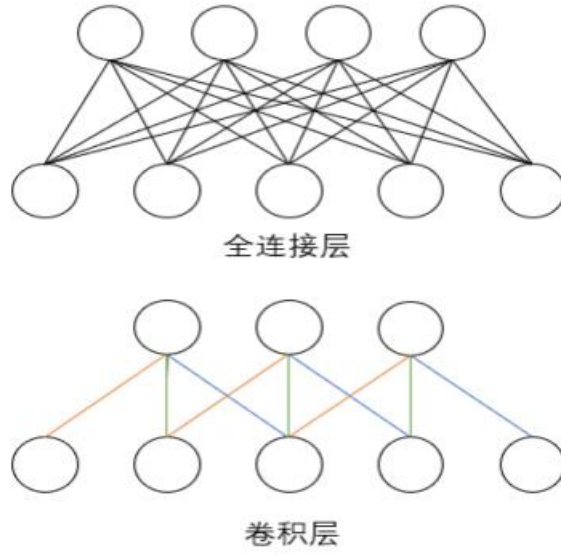


图 2-4 卷积神经网络特点示意图

2.2.3 卷积神经网络的算法

卷积运算在卷积神经网络结构中也称为互相关运算。通常情况下，卷积层的输入为一组特征图， $X \in R^{M \times N \times D}$ ，其中每一个特征图 $x \in R^{M \times N}$ 。模型的参数卷积核通常为四维的张量 $W \in R^{U \times V \times P \times D}$ ，其中一个切片为 $w \in R^{U \times V}$ 的二维矩阵。

假设输入数据的一张二维的特征图为 $x \in R^{M \times N}$ 。模型定义的一个卷积核为 $w \in R^{U \times V}$ ，则经过卷积计算的结果为：

$$z = \sum_{d=1}^D w \otimes x + b \quad (2-1)$$

式 (2-1) 中 \otimes 表示卷积操作， d 表示卷积计算的区域。其中，

$$Z_{ij} = \sum_{u=1}^U \sum_{v=1}^V w_{uv} x_{i+u-1, j+v-1} + b \quad (2-2)$$

式 (2-2) 中， b 为偏置值， z_{ij} 为该卷积核对所计算区域的输出。详情如图 2-4 所示。

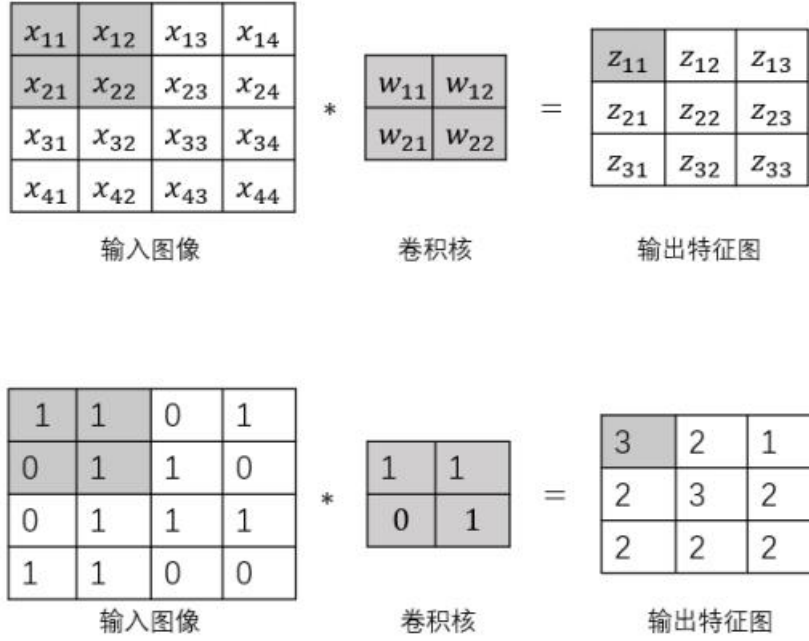


图 2-5 卷积计算示意图

在图 2-5 中，卷积核作用于输入特征图的左上角位置，覆盖区域为 2×2 ，卷积计算的输出结果为：

$$z_{i,j} = w_{i,j}x_{i,j} + w_{i,j+1}x_{i,j+1} + w_{i+1,j}x_{i+1,j} + w_{i+1,j+1}x_{i+1,j+1} + b \quad (2-3)$$

通常在卷积计算过后需要对计算结果进行非线性激活处理，即由卷积层进入到激活层。将卷积计算的结果输入到激活函数中，对应的公式为：

$$a = f(z) \quad (2-4)$$

公式 (2-4) 中， f 为激活函数，常用的激活函数是 ReLU。

整个卷积的计算过程就是一个三维的输入特征图与四维的卷积核进行互相计算，计算后的输出结果为一个全新的三维特征图。

由激活层进入到池化层后，需要进行特征选择和特征降维。池化层按照运算的过程可分为最大池化层和平均池化层。

对于最大池化层而言，池化的作用就是在输入特征图中某个区域，遍历挑选出该区域内的最大值、假设某个输入特征图 $x \in R^{M \times N}$ ，池化得到区域为 R_{mn} ，其中 $1 \leq m \leq M$ ， $1 \leq n \leq N$ ，则对该区域的最大池化结果为：

$$y = \max(x_i) \quad (2-5)$$

式 (2-5) 中， x_i 为区域 R_{mn} 中的值。

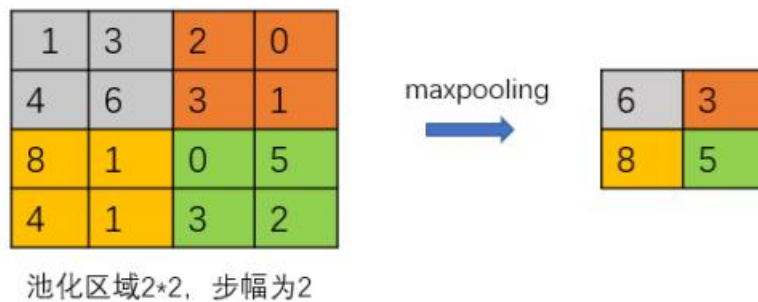


图 2-6 最大池化计算示意图

对于平均池化层，输出值为区域 R_{mn} 内对应特征值的平均信息。

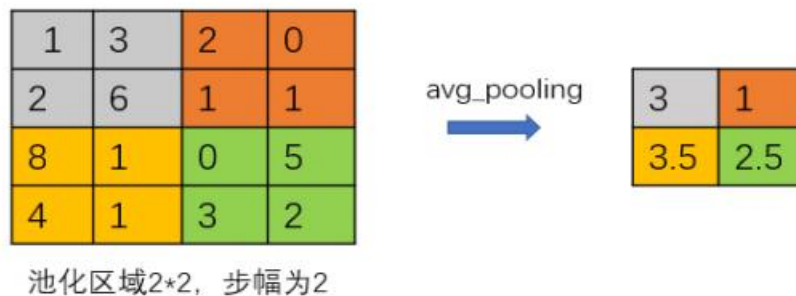


图 2-7 最大池化计算示意图

2.3 特征提取

在脱机手写体汉字识别的传统流程中，特征提取构成了整个识别任务的核心环节，其成效直接关乎识别的准确性。为了有效地捕捉手写汉字的关键信息，研究者们已经开发出多种特征提取技术。这些技术主要分为两大类：结构特征提取和统计特征提取。

结构特征提取方法侧重于分析汉字的几何结构，如笔画、端点和交点等，这些特征对于理解汉字的形状和构造至关重要。而统计特征提取方法则关注于图像的局部或全局统计信息，如像素分布、梯度方向等，这种方法能够揭示汉字的内在统计规律。

两种方法各有优劣，结构特征提取方法能够精确描述汉字的形态特征，适合于规范书写文本的识别；统计特征提取方法则在处理非规范书写或复杂背景下的汉字识别时表现出较强的鲁棒性。

2.3.1 基于结构的特征提取方法

基于结构的特征提取方法致力于捕捉汉字图像的结构性信息。汉字的结构之所以具有稳定性，是因为它主要由“一”、“丨”、“丿”、“㇏”这四种最基本笔画构成，同时辅以端点、拐角和交叉点等特征。这种稳定性使得基于结构的特征提取方法能够较好地适应非规范书写的手写字体，展现出其优势。

然而，这种方法也存在一些不足之处。首先，它需要进行大量的计算，这导致特征提取过程耗时较长，速度较慢。其次，当手写体的书写风格较为随意时，可能会对汉字的结构特征造成干扰。这种随意性会降低特征提取的稳定性，使得提取结果受到负面影响。

综上所述，基于结构的特征提取方法虽然在适应不同书写风格方面表现出了一定的灵活性，但其在计算效率和对书写变异的敏感性方面仍有改进空间。

2.3.2 基于统计的特征提取方法

基于统计的特征提取方法是一种高效的汉字图像处理技术。这种方法的核心在于将汉字图像划分为若干小块，然后对每个区域的局部统计信息进行分析和提取。具体流程包括：首先将汉字图像分割成若干小块；接着对每一块图像进行信息特征的统计；最终，综合这些局部特征来构建整个汉字的特征表示。这种方法的一个显著优势是能够快速提取特征，并且具有较高的识别率。在脱机手写体汉字识别中，广泛使用的统计特征提取方法有弹性网格特征、方向梯度直方图(Histogram of Orientation)、Gradients, HOG 特征、方向性特征(Directional Element Features), DAG、Gabor 特征和 IICA 等特征等。

弹性网格特征：这种方法通过非均匀线条根据汉字像素点的分布来划分图像，形成适应汉字结构的网格。全局弹性网格因其简单和广泛的适用性而受到青睐。尽管它能够较好地反映汉字的结构特征并适应不同书写风格带来的变化，但它无法捕捉到汉字的整体结构，因为图像块之间缺乏联系。

HOG 梯度特征：HOG (Histogram of Oriented Gradients) 特征通过统计图像中局部区域的梯度方向直方图来描述物体。它的优势在于对图像的几何和光学形变具有很好的不变性，同时具备优秀的空间频率域信息特性。HOG 滤波器对光照变化不敏感，这使得它在多变的光照条件下也能保持稳定的识别性能。

Gabor 特征：Gabor 滤波器模仿人脑视觉系统中的简单细胞响应，对图像边缘非常敏感，并提供方向和尺度的选择性。这种滤波器对光照变化不敏感，有助于在不同光照条件下提取汉字特征，适用于频域内不同方向的特征提取。

ICA 特征：独立成分分析（ICA）是一种分析信号高阶统计特性的方法，它考虑了信号概率密度函数的统计独立性。ICA 能够揭示数据间的本质结构，尤其适用于图像像素高阶统计量中所蕴含的重要信息。通过 ICA，可以提取图像的高阶统计特性，从而改善图像识别的准确性。

2.4 本章小结

本章主要介绍了神经元的结构和神经网络常用的算法和框架，并且详细讲述了卷积神经网络的结果、特点以及算法，还补充介绍了手写汉字特征提取的两种方法，即基于结构的特征提取的方法和基于统计的特征提取的方法。

第三章 基于 CNN 的手写汉字识别

之前的章节我们介绍了神经网络概述、卷积神经网络的架构和算法，以及在手写汉字识别过程中用到的特征提取的方法。在本章节我们将构建一个卷积神经网络用于识别手写体汉字，从数据准备到模型构建，再到模型训练，以及最后的模型测试。

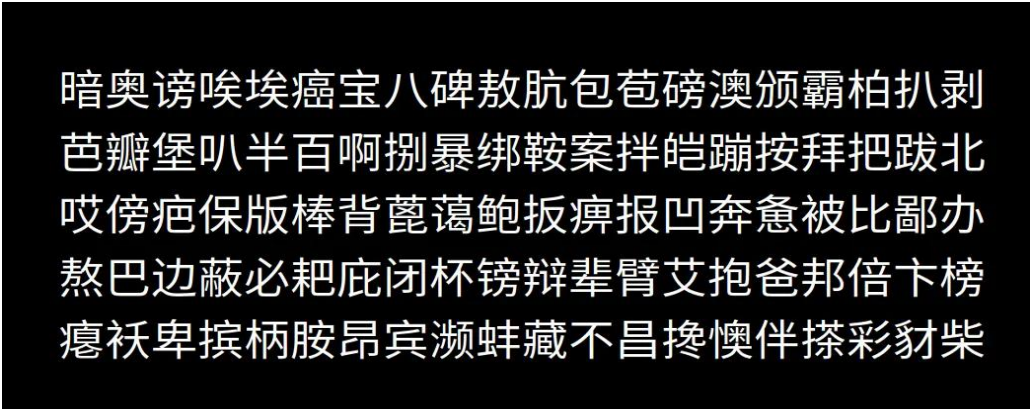
3.1 数据准备

本项目所采用的数据全部来自于 CASIA 的开源中文手写数据集，该数据集分为两部分：

1、CASIA-HWDB：离线的 HWDB，其中 1.0-1.2，这是单字的数据集，2.0-2.0 是整张文本的数据集，本项目中只使用 1.0-1.2 的单字数据集。数据集中包含了约 7185 个汉字以及 171 个英文字母。数字、标点符号等。

2、CASIA-OLHWDB：在线的 HWDB，格式与 CASIA-HWDB 相同，同样包含了 7185 个汉字以及 171 个英文字母。数字、标点符号等。

由于原始数据过于复杂，我们使用一个类来封装数据读取过程。数据集中的部分文字识别出来的结果如图 3-1 所示。



暗奥谤唉埃癌宝八碑敖肮包苞磅澳颁霸柏扒剥
芭瓣堡叭半百啊捌暴绑鞍案拌皑蹦按拜把跋北
哎傍疤保版棒背蓖蔼鲍扳痹报凹奔惫被比鄙办
熬巴边蔽必耙庇闭杯镑辩辈臂艾抱爸邦倍卞榜
瘰祲卑摈柄胺昂宾濒蚌藏不昌揅懊伴捺彩豺柴

图 3-1 数据集部分识别结果

原始数据集，以 trn_gnt.zip 文件的形式从服务器下载，需经过两次解压缩

过程，方可获取包含 gnt 格式文本的目录结构。该目录中，每个 gnt 文件均包含一组汉字及其相应的标注信息。直接对这些文件进行处理不仅操作复杂，而且不利于后续图像提取与操作。

尽管将单个汉字图像化并存储为图片文件在视觉上更为直观，但考虑到汉字图像的尺寸仅为 80×80 像素，这一分辨率并不适合直接转换为图像格式并保存于本地存储。因此，为了更有效的数据管理和后续的批量读取与训练，建议将汉字的原始二进制数据转换为 tfrecord 格式。tfrecord 是一种高效的数据存储格式，能够支持快速的数据读取和处理，适合于机器学习模型的训练过程。通过将数据转换为 tfrecord，可以直接从该格式中读取图像数据，进而进行模型训练和评估。

3.2 模型构建

3.2.1 构建模型

使用 Keras API 快速构建一个网络模型，因为它足够简单，只包含两个卷积层（Conv2D）和两个最大池化层（maxpool）层。部分代码如图 3-2 所示。但是即使再简单模型，有时候也能发挥出巨大的用处，对于某些特定的问题可能比更深的网络更有前途。在构建这部分模型时需要注意：

1、如果只是构建序列模型，没有频繁的跳跃链接，可以直接使用 keras.Sequential 来构建模型。

2、Conv2D 中最好制定每个参数的名字，不要省略，否则无法分别是输入的通道还是过滤器。

```
def build_net_003(input_shape, n_classes):
    model = tf.keras.Sequential([
        layers.Conv2D(input_shape=input_shape, filters=32, kernel_size=
(3, 3), strides=(1, 1),
                        padding='same', activation='relu'),
        layers.MaxPool2D(pool_size=(2, 2), padding='same'),
        layers.Conv2D(filters=64, kernel_size=(3, 3), padding='same'),
        layers.MaxPool2D(pool_size=(2, 2), padding='same'),

        layers.Flatten(),
        layers.Dense(n_classes, activation='softmax')
    ])
    return model
```

图 3-2 模型构建部分代码

3.2.2 数据输入

在深度学习的数据准备阶段，将数据集转换为 TensorFlow 的 tfrecord 格式是一种常见的做法。tfrecord 是一种高效的数据存储格式，它允许将图片和标签等数据项序列化并存储为单一的二进制文件。在这一过程中，数据的图片和标签信息被编码为字节流并保存在 tfrecord 中。因此，在数据读取阶段，必须从 tfrecord 中相应地提取这些信息。读取数据的部分代码如图 3-3 所示。在读取数据时需要注意：

1、将 numpy 数组的字节直接存储到 tfrecord 与将文件的字节直接存储到 tfrecord 中，在解码时的处理方式存在差异。由于本研究中使用的图片数据并非来源于本地文件系统，而是通过调用 tobytes() 方法将 numpy 数组转换为字节格式进行存储，因此这种存储方式并不包含图片的维度信息。为了在读取时能够正确恢复图片的原始维度，存储 tfrecord 时必须同时记录图片的宽度和高度信息。

2、在对原始字节流(raw bytes)进行解码时，必须注意字符编码类型。本研究中，数据以 UTF-8 编码格式存储，因此在解码时也应使用 UTF-8 编码。不同的存储编码方式需要对应不同的解码方法，只有使用正确的解码方式，才能确保读取出的数据内容的准确性和完整性。

```
def parse_example_v2(record):
    """
    latest version format
    :param record:
    :return:
    """
    features = tf.io.parse_single_example(record,
                                          features={
                                              'width':
                                                  tf.io.FixedLenFeature([], tf.int64),
                                              'height':
                                                  tf.io.FixedLenFeature([], tf.int64),
                                              'label':
                                                  tf.io.FixedLenFeature([], tf.int64),
                                              'image':
                                                  tf.io.FixedLenFeature([], tf.string),
                                          })
    img = tf.io.decode_raw(features['image'], out_type=tf.uint8)
    # we can not reshape since it stores with original size
    w = features['width']
    h = features['height']
    img = tf.cast(tf.reshape(img, (w, h)), dtype=tf.float32)
    label = tf.cast(features['label'], tf.int64)
    return {'image': img, 'label': label}
```

图 3-3 数据输入部分代码

3.3 模型训练

模型训练代码如图 3-4 和图 3-5 所示。

```
def train():
    all_characters = load_characters()
    num_classes = len(all_characters)
    logging.info('all characters: {}'.format(num_classes))
    train_dataset = load_ds()
    train_dataset = train_dataset.shuffle(100).map(preprocess).batch(32).repeat()

    val_ds = load_val_ds()
    val_ds = val_ds.shuffle(100).map(preprocess).batch(32).repeat()

    for data in train_dataset.take(2):
        print(data)

    # init model
    model = build_net_003((64, 64, 1), num_classes)
    model.summary()
    logging.info('model loaded.')

    start_epoch = 0
    latest_ckpt = tf.train.latest_checkpoint(os.path.dirname(ckpt_path))
    if latest_ckpt:
        start_epoch = int(latest_ckpt.split('-')[1].split('.')[0])
        model.load_weights(latest_ckpt)
        logging.info('model resumed from: {}, start at epoch: {}'.format(latest_ckpt, start_epoch))
    else:
        logging.info('passing resume since weights not there. training from scratch')
```

图 3-4 模型训练部分代码（1）

```
if use_keras_fit:
    model.compile(
        optimizer=tf.keras.optimizers.Adam(),
        loss=tf.keras.losses.SparseCategoricalCrossentropy(),
        metrics=['accuracy'])
    callbacks = [
        tf.keras.callbacks.ModelCheckpoint(ckpt_path,
                                           save_weights_only=True,
                                           verbose=1,
                                           period=500)
    ]
    try:
        model.fit(
            train_dataset,
            validation_data=val_ds,
            validation_steps=1000,
            epochs=15000,
            steps_per_epoch=1024,
            callbacks=callbacks)
    except KeyboardInterrupt:
        model.save_weights(ckpt_path.format(epoch=0))
        logging.info('keras model saved.')
    model.save_weights(ckpt_path.format(epoch=0))
    model.save(os.path.join(os.path.dirname(ckpt_path), 'cn_ocr.h5'))
```

图 3-5 模型训练部分代码（2）

模型训练起来之后，可以达到 95%的准确率，模型训练结果如图 3-6 所示。


```
Epoch 27/15000
1024/1024 [=====] - 30s 29ms/step - loss: 0.8046 - accuracy: 0.8071 - val_loss: 4.3333 - val_accuracy: 0.4497
Epoch 28/15000
1024/1024 [=====] - 32s 31ms/step - loss: 0.4665 - accuracy: 0.8873 - val_loss: 5.0743 - val_accuracy: 0.4170
Epoch 29/15000
1024/1024 [=====] - 32s 31ms/step - loss: 0.1350 - accuracy: 0.9604 - val_loss: 5.2533 - val_accuracy: 0.4228
Epoch 30/15000
1024/1024 [=====] - 31s 30ms/step - loss: 0.2011 - accuracy: 0.9448 - val_loss: 6.2211 - val_accuracy: 0.3953
Epoch 31/15000
1024/1024 [=====] - 31s 30ms/step - loss: 0.1409 - accuracy: 0.9601 - val_loss: 5.7776 - val_accuracy: 0.4423
Epoch 32/15000
1024/1024 [=====] - 30s 30ms/step - loss: 0.1223 - accuracy: 0.9641 - val_loss: 6.3663 - val_accuracy: 0.4061
Epoch 33/15000
1024/1024 [=====] - 32s 31ms/step - loss: 0.1551 - accuracy: 0.9575 - val_loss: 6.3519 - val_accuracy: 0.4017
Epoch 34/15000
603/1024 [=====>.....] - ETA: 11s - loss: 0.1454 - accuracy: 0.9573
```

图 3-6 模型训练结果展示

3.4 模型测试

找一些手写体汉字作为测试集，不分测试集手写汉字如图 3-7 所示：



图 3-8 部分测试集

将测试集输入到模型中，得出测试结果，如图 3-9 所示：



图 3-8 测试集训练结果

3.5 本章小结

本章介绍了从开源中文手写数据集 CASIA 中获取离线的 HWDB，仅使用其中的单字数据集，并且将数据集转换为 tfrecord 格式。使用 Keras API 快速构建一个只包含两个卷积层和两个最大池化层的简单的卷积神经网络。将 tfrecord 格式中的内容读取出来，并将其作为模型输入，对模型进行训练。之后采用一些手写体汉字对模型进行测试。

第四章 总结和展望

4.1 工作总结

手写汉字识别，一个长久以来因汉字种类繁多和手写风格的多变性而充满挑战的研究领域，一直没有得到完美的解决方案。然而，近年来卷积神经网络（CNN）在图像识别领域取得了显著的进展，其出色的特征提取和图像分类能力为这一难题带来了新的希望。本文在通过卷积神经网络对手写汉字进行识别做了如下研究工作：

（1）查阅了大量有关卷积神经网络和手写体汉字识别的文献，分析了目前国内在这方面的研究现状，理清了手写汉字识别的难点。

（2）研究了神经网络的架构，以及在神经网络研究过程中常用的算法。同时侧重于研究卷积神经网络的架构，层级结构，了解了卷积神经网络局部连接和权重共享的特性，以及卷积神经网络在各个层级具体实现算法。

（3）通过开源中文手写数据 CASIA 获取离线数据集 HWDB，并将其存储在 tfrecord，使用 Keras 快速构建了一个简单的卷积神经网络。将 tfrecord 中的内容作为模型的输入数据对模型进行训练，最后采用一些简单的手写体汉字对模型进行测试。

4.2 展望未来

本文使用 Keras 快速搭建了一个简单的卷积神经网络实现了手写汉字的识别，并且目前手写汉字识别的技术已取得显著进展，但是仍然存在诸多挑战和改进空间，例如：

（1）模型结构的优化：尽管卷积神经网络及其变体在手写汉字识别领域取得了成功，但是仍需要探索更高效的网络结构，以减少计算资源的消耗，提高识别速度，在做到高精度的同时保证保效率。

（2）实时识别与应用：研究实时手写汉字识别技术，将其应用到在线教育、办公自动化等领域，提供用户体验。

（3）多模态学习：结合手写汉字的多种特征，如笔画顺序、方向性等，进行多模态学习，可能进一步提高识别的准确率。

（4）数据集的扩展和多样性：当前研究中使用的数据集相对有限，未来需要构建更大规模、更多样化的手写汉字数据集，以提高模型的泛化能力和鲁棒性。

（5）艺术性书法识别，针对书法艺术中的古体手写汉字，研究更为精细识别技术，以处理书法作品中的独特风格和变化。

参考文献

- [1] 张通. 基于Swin Transformer和CNN的手写体汉字识别算法研究与应用[D]. 青岛科技大学, 2023. DOI:10.27264/d.cnki.gqdhc.2023.000996.
- [2] 肖婷婷. 基于卷积神经网络的手写汉字识别方法[J]. 科技创新与应用, 2021, 11(25):114-115+118.
- [3] 李龙雪. 深度学习在手写汉字识别中的应用研究[D]. 天津科技大学, 2021. DOI:10.27359/d.cnki.gtqgu.2021.000520.
- [4] 韩梦云. 基于CNN的脱机手写体汉字识别研究[D]. 河南师范大学, 2020. DOI:10.27118/d.cnki.ghesu.2020.000912.
- [5] 袁柱. 基于深度学习的脱机手写汉字识别的研究与应用[D]. 广东工业大学, 2020. DOI:10.27029/d.cnki.ggdgu.2020.000709.
- [6] 黄洋. 基于深度学习的脱机手写汉字识别技术研究[D]. 重庆邮电大学, 2019. DOI:10.27675/d.cnki.gcydx.2019.000059.
- [7] 徐奇. 基于改进卷积神经网络的手写体汉字识别[J]. 电子技术与软件工程, 2022, (09):190-193.
- [8] 王建华, 张雅祺, 肖博怀, 等. 基于改进卷积神经网络的手写汉字识别研究(英文)[J]. 印刷与数字媒体技术研究, 2023, (01):45-56. DOI:10.19370/j.cnki.cn10-1886/ts.2023.01.006.
- [9] Y. Li and Y. Li, "Design and implementation of handwritten Chinese character recognition method based on CNN and TensorFlow," 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 2021, pp. 878-882, doi: 10.1109/ICAICA52286.2021.9498146.
- [10] N. Aleskerova and A. Zhuravlev, "Handwritten Chinese Characters Recognition Using Two-Stage Hierarchical Convolutional Neural Network," 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), Dortmund, Germany, 2020, pp. 343-348, doi: 10.1109/ICFHR2020.2020.00069.
- [11] Q. Hao, X. Wu, S. Zhang, P. Zhang, X. Ma and J. Jiang, "Research on Offline Handwritten Chinese Character Recognition Based on Deep Learning," 2019 9th International Conference on Information Science and Technology (ICIST), Hulunbuir, China, 2019, pp. 470-474, doi: 10.1109/ICIST.2019.8836833.
- [12] S. Katoch, M. Rakhra and D. Singh, "Recognition Of Handwritten English Character Using Convolutional Neural Network," 2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST), Delhi, India, 2022, pp. 1-6, doi: 10.1109/AIST55798.2022.10064860.
- [13] 董春生. 基于文本识别的手写汉字识别平台的设计与实现[D]. 中国科学院大学(中国科学院沈阳计算技术研究所), 2022. DOI:10.27587/d.cnki.gksjs.2022.000049.
- [14] 许兴淼. 高效脱机手写汉字识别网络研究[D]. 南昌航空大

- 学, 2022. DOI:10.27233/d.cnki.gnchc.2022.000245.
- [15] 万茹月, 海玲, 谷铮, 等. 基于深度学习的手写文字识别[J]. 现代信息科技, 2021, 5(19):89-91+96. DOI:10.19850/j.cnki.2096-4706.2021.19.022.
- [16] 甘吉. 手写文字识别及相关问题算法研究[D]. 中国科学院大学(中国科学院计算机科学与技术学院), 2021. DOI:10.44196/d.cnki.gjskx.2021.000003.

个人总结

在本次课程内容实现的过程中，加深了我对卷积神经网络（CNN）的认识。在研究过程中，我深刻体会到理论知识与实践应用相结合的重要性，通过学习CNN的基本原理和结构，我能够更好地理解网络模型和工作原理，将其应用于实际问题中。在构建CNN模型时，对比了不同激活函数、损失函数和优化器，逐渐理解了它们对模型性能的影响，并掌握了如何根据实际情况的需要来调整参数的模型。还有就是撰写论文的时候，学到了一些写论文的格式、要求，以及在写论文的过程中学到的撰写技巧。

当然在实现本次课程内容的过程中也遇到了一些问题，比如说：在第一次读取tfrecord中的内容的时候使用了不合理的解码方式，导致读取失败；在最初构建CNN模型的时候，构建了一个很复杂的模型，使用训练集训练之后发现准确度不到1%，模型发散了等等。对于出现这种问题，我先是查找文献，分析问题所在，之后再网络上查找相关解决问题的方法，最后实在解决不了，会去询问人工智能，让其帮助解决问题，这样培养自己遇到问题时解决问题的能力。

在这次课程内容实现过程中，做的好的部分是能够沉住气在大量的文献中查找有关的文献，并且是第一次尝试阅读英文文献；自己一步步搭建属于自己的神经网络，完整的完成从获取数据集，到搭建模型、训练模型、检测模型这一流程，并且在最后能够实现具体的功能。做的不好的部分就是并没有实现完整的可视化的界面，仅仅是将模型训练好，证明了模型能够识别手写汉字，但并没有将其落到实处，没能实现用户可操作的界面，并且更多的是复现前辈实现过的内容，再加上自己对神经网络的理解，创新性的内容较少。

对于使用卷积神经网络实现手写汉字识别，可以使用效率更高、识别准确度更高的变种模型，再加上用户可操作页面的实现，提高识别手写汉字的实时性，真正将这一功能落到实处。

查重报告截图

查重报告截图

基本信息		结果指标	
文档名称	基于CNN的脱机手写汉字识别	总相似率	12.13%
文档作者	梁浩铂	原创率	87.87%
文档字数	17016	抄袭率	12.13%