高等院校计算机类专业系列教材

从机器学习到深度学习：模型、进展与应用

张善文　李丽萍　张　婷　主编

南京大学出版社

内容简介

人们正处在人工智能深刻重塑世界的时代。本书系统梳理了从经典机器学习到现代深度学习的理论演进与技术脉络，旨在为读者构建一幅完整而清晰的人工智能知识图谱。

全书内容由浅入深，涵盖了机器学习十大经典算法、神经网络基础组件、卷积神经网络（CNN）、序列与图神经网络（如 RNN、Transformer、GCN 及新兴的 Mamba 模型）以及知识图谱等核心主题。不仅深入剖析了模型背后的数学原理与设计思想，更紧密结合了其在计算机视觉、自然语言处理、科学发现等领域的突破性应用。

本书可作为计算机科学、人工智能及相关专业高年级本科生和研究生的教材，也可供广大科研人员与工程师作为系统学习和深入研究的参考指南。它既是一份通往智能技术前沿的地图，也是一位助力探索与实践的可靠向导。

图书在版编目（CIP）数据

从机器学习到深度学习 ：模型、进展与应用 / 张善文，李丽萍，张婷主编. -- 南京 ：南京大学出版社，2026. 10. -- ISBN 978-7-305-30084-4

Ⅰ. TP181

中国国家版本馆 CIP 数据核字第 2026GA5104 号

出版发行　南京大学出版社
社　　址　南京市汉口路 22 号　　邮　　编　210093

书　　名　从机器学习到深度学习：模型、进展与应用
CONG JIQI XUEXI DAO SHENDU XUEXI：MOXING、JINZHAN YU YINGYONG
主　　编　张善文　李丽萍　张　婷
责任编辑　吴　华　　编辑热线　025-83596997

照　　排　南京开卷文化传媒有限公司
印　　刷　南京玉河印刷厂有限公司
开　　本　787 mm×1092 mm　1/16 开　印张 19　字数 474 千
版　　次　2026 年 10 月第 1 版　2026 年 10 月第 1 次印刷
书　　号　978-7-305-30084-4
定　　价　56.00 元

网　　址：http://www.njupco.com
官方微博：http://weibo.com/njupco
微信服务号：NJUYUNSHU
销售咨询热线：(025)83594756

* 版权所有，侵权必究
* 凡购买南大版图书，如有印装质量问题，请与所购图书销售部门联系调换

前言

人工智能(AI)正以超乎想象的速度改变世界,从实验室的算法突破到日常生活中的智能交互,这场由数据与算力驱动的变革正席卷各行各业。在这一波澜壮阔的进程中,深度学习作为AI的基石,赋予了机器从海量数据中自动提取层次化特征的能力,从而推动语音识别、图像理解、自然语言处理等领域取得跨越式进展。

回望AI技术演进的路径,一条清晰的脉络逐渐浮现:AI的每一次飞跃,几乎都伴随着机器学习乃至深度学习架构的创新与优化。经典机器学习为其奠定了统计学习的理论基础,而深度学习则在此基础上,凭借端到端学习与高维数据表征的独特优势,将AI从“能识别”推向“能理解”,乃至“能创造”。系统学习机器学习与深度学习的原理、模型及其应用,不仅是掌握AI核心技术的关键,更是洞察未来AI发展方向的前提。

本书《从机器学习到深度学习:模型、进展与应用》正是在这样的时代与技术背景下应运而生,旨在为计算机及人工智能相关专业的本科生、新一代电子信息技术相关学科的研究生以及AI产业界的工程师,提供一份系统而全面的学习指南。

一、全书结构

本书按照“基础理论—核心模型—前沿拓展—应用实践”逻辑展开,共分为七章:

第一章　智能的演进:从数据中学习的科学

作为全书的开篇,本章从“为什么需要从数据中学习”这一根本问题出发,系统梳理人工智能、机器学习与深度学习的演进脉络,揭示三者之间的内在联系与发展逻辑,帮助读者建立对智能技术体系的整体认知。

第二章　ML十大经典算法

本章深入讲解线性回归、K近邻、支持向量机等十大经典机器学习算法,从数学原理推

导到实践案例分析,夯实机器学习基础,为后续深度学习内容的学习做好理论铺垫。

第三章　DL 基础知识

本章系统剖析神经网络的基本组件,包括卷积、池化、激活函数、注意力机制、归一化、损失函数等核心概念与操作,同时深入讨论参数量计算、过拟合、可解释性等实践中的关键问题,为理解复杂模型奠定坚实基础。

第四章　卷积神经网络

本章专注计算机视觉领域,详解卷积神经网络(CNN)的经典架构与演进历程,涵盖 LeNet、AlexNet、VGG、ResNet、MobileNet、U - Net 及其改进系列、YOLO 系列、R - CNN 系列等代表性模型,全面呈现 CNN 在图像分类、目标检测、语义分割等任务中的应用与发展。

第五章　序列与图结构的 DL

本章从深度学习范式的演进出发,系统探讨循环神经网络(RNN)、长短期记忆网络(LSTM)、门控循环单元(GRU)、Transformer、视觉 Transformer(ViT)及其改进模型,以及新兴的状态空间模型(SSM)、Mamba、视觉 Mamba(VMamba)、VM - UNet 等前沿架构。同时涵盖图卷积网络(GCN)、文本嵌入等关键技术,全面呈现序列建模与图数据挖掘领域的最新进展。

第六章　知识图谱

拓展人工智能的知识边界,本章从“从数据到认知的桥梁”这一视角出发,系统介绍知识图谱的基本概念、知识表示与学习、知识获取与加工、构建方法等核心内容,并结合 EduKG 构建和《红楼梦》知识图谱构建两个实践案例,帮助读者掌握知识图谱的工程实现方法。

第七章　DL 应用案例

以产业应用为导向,本章系统展示深度学习在森林火灾检测、交通标志检测、路面缺陷检测、作物病害叶片分割、焊缝缺陷检测、遥感图像显著性检测与目标检测、无人机航拍图像目标检测等多个领域的创新应用,同时展示知识图谱在教育领域的应用实践。本章旨在帮助读者将理论知识转化为解决实际问题的能力。

二、读者定位与使用建议

本书定位为本科生与研究生双层次教材,兼顾科研人员与产业界工程师等不同层次读

者的学习需求：

读者层次	学习目标	建议阅读路径
本科生	建立系统知识体系，掌握核心算法与模型	系统研读第一至四章，夯实理论基础；第五至七章作为拓展阅读与课程实践参考
研究生	聚焦前沿技术，开展科研与工程实践	快速浏览第一至三章，重点研读第四至七章，关注模型演进、前沿架构（如 Mamba、VM-UNet）与应用创新
产业界工程师	获取工程实践参考，解决实际问题	直接查阅第四、五章中感兴趣的模型架构，以及第七章的应用案例

本书可作为人工智能、计算机科学与技术、软件工程、智能科学与技术、机器人工程等本科专业教材，也可作为高等院校新一代电子信息技术、控制科学与工程、计算机科学与技术、人工智能等相关专业研究生的教学用书，同时可供从事 AI 技术研发、数据分析、智能系统构建的工程技术人员自学参考。

三、编写背景与特色

深度学习作为人工智能的核心技术，正以前所未有的深度和广度推动着科学研究和产业应用的变革。从 CNN 在计算机视觉中的主导，到 Transformer 在自然语言处理领域的突破，再到 Mamba 等状态空间模型对序列建模范式的重新定义，深度学习模型架构正经历着快速迭代。如何系统梳理这些经典模型与前沿进展，并将其有机融合，成为当前教学与科研的重要课题。为此，我们团队在大量调研与实践基础上编写此书，力求做到理论深度与实践广度的有机结合。

本书在内容组织与编写上力求体现以下特色：

➢ **系统性**：从经典机器学习到深度学习前沿，从基础理论到应用实践，构建完整的知识体系。

➢ **前沿性**：涵盖状态空间模型（SSM）、Mamba、视觉 Mamba（VMamba）等近年最新研究成果。

➢ **实践性**：第七章提供十余个真实应用案例，涵盖低空经济、智慧农业、智能交通等多个热点领域。

➢ **可读性**：注重概念的可视化呈现，关键算法辅以图表和实例说明。

四、致谢、免责与求正声明

本书在编写过程中,参考、借鉴并引用了众多专家、学者及研究人员在机器学习、深度学习等领域的理论成果、技术方法与典型案例,在此谨致以衷心的感谢。本书亦获得国家自然科学基金项目(62273284)的资助,特此致谢。

此外,本书部分内容参考并整合了网络上公开可获取的资料以及众多专家学者的研究成果。由于来源较为分散,部分内容未能一一注明出处,在此一并感谢并致歉。尽管编者力求内容系统、准确,但受限于深度学习领域新技术的快速演进及作者自身学术水平,书中难免存在疏漏与不足之处。编者承诺,在本书再版时,将于书末补充“主要参考网络资源列表”或详细来源说明(包括主要网站、机构或知识库名称等),以进一步规范学术引用。

诚邀广大同行专家与读者不吝批评指正(请致信:wjdw716@163.com),以便再版时予以补充、修订与完善。

编　者

2026 年 5 月

目录

第一章　智能的演进:从数据中学习的科学

人工智能(AI)的核心目标是赋予机器感知、理解、学习与决策的能力,以胜任人类智慧主导的任务。实现这一目标的关键范式转移,是从“直接编程定义规则”转向“让机器从数据中自主发现规律”,由此催生了**从数据中学习的科学**。本章系统梳理机器学习(ML)与深度学习(DL)的核心理念、演进脉络及价值,为理解智能技术的应用奠定基础。

1.1　为什么需要从数据中学习?

在 ML 系统中,数据是驱动智能化的核心动力,如同滋养“数字大脑”的知识养分与经验来源。按用途可分为三类:**训练数据**用于模型参数学习,**验证数据**用于模型选择与超参数调优,**测试数据**用于独立评估模型性能。

1. 数据:智能系统的生命力源泉

数据是 ML 流程的基础,其重要性堪比血液之于生命体。从特征提取、模型训练到预测决策,每个环节均依赖数据支撑。**数据的质量、数量与多样性直接决定智能系统的性能上限**:优质丰富的数据能帮助模型构建完整的认知框架,持续更新的数据流则保障系统的动态适应性;反之,低质或匮乏的数据会导致模型性能衰退甚至失效。

2. 从白纸到专家:ML 的内在机理

ML 通过数学框架实现模型从“无知”到“精通”的自主进化。初始模型如同一张白纸,通过接触海量训练数据,识别数据中的潜在模式,再借助优化算法迭代调整内部参数,逐步逼近输入与输出的真实映射关系。

这一过程与人类学习高度相似:如同婴儿通过观察世界建立认知,模型通过数据反馈积累知识。区别在于机器能以远超人类的速度处理海量信息,在特定领域达到超越人类的专业水准。

3. 数据驱动的革命性价值

数据驱动方法突破了传统规则驱动方法的局限,带来三大核心变革:

(1) 突破认知瓶颈:面对图像识别、自然语言理解等复杂任务,传统方法难以穷尽所有规则,而数据驱动方法无需预先定义规律,让机器从样本中自主总结模式,巧妙绕开了“规律形式化”的难题。

(2) 实现决策客观化:传统决策易受人类经验与偏见影响,数据驱动决策则基于可量化、可验证的证据,能捕捉人眼难以察觉的细微模式,在金融反欺诈、医疗早期诊断等场景中

提升决策精准度。

(3) 提升效率与创新能力:训练成熟的模型可近乎零边际成本处理海量数据,实现知识获取自动化;同时,算法能挖掘人类未发现的模式与趋势,催生新的解决方案与商业模式。

4. 迈向智能新纪元

从数据中学习不仅是技术突破,更是全新的认知范式——将数据转化为"可计算的洞察力",拓展人类认识与改造世界的边界。随着数据资源与处理技术的发展,数据驱动的智能系统正推动 AI 向更高层次迈进,为解决复杂问题、构建高效智能社会开辟了全新可能。

1.2 为什么需要 ML?

ML 是 AI 的核心分支,其本质是赋予计算机从数据中**自主学习规律**的能力,而非执行预设的固定指令。

- **传统编程**:如同"手把手教孩子认猫",需精确定义"尖耳朵、胡须、四条腿"等规则,适用于逻辑清晰、规则明确的场景;
- **ML**:如同"给孩子看上万张猫的图片,让其自主归纳特征",适用于规则复杂、难以形式化的任务(如图像识别、语音理解)。

1. 解决复杂现实问题的核心路径

在应对图像识别、自然语言理解等复杂任务时,传统方法依赖人工编写确定规则,往往面临"规则难以穷举"的困境,导致程序僵化、扩展性差。例如,在语音识别或自动驾驶场景中,企图构建一个覆盖所有可能情况的规则库几乎不可行。ML 则从根本上绕开了这一瓶颈,转而采用数据驱动的方式,通过算法从样本中自动学习潜在规律,从而实现对复杂模式的捕捉与泛化。ML 是实现 AI 从理论走向应用的关键桥梁,它为解决各行各业中那些规则不明确、逻辑复杂或动态变化的问题提供了可行的技术路径。无论是在医疗领域辅助疾病诊断、在金融行业进行风险控制,还是在推动自动驾驶技术成熟等方面,ML 都已成为不可或缺的基础工具。

2. 应对大数据时代的机遇与挑战

当今时代,数据以前所未有的规模积累,已成为一种关键的生产要素。然而,海量数据本身并不直接产生价值,如何从中高效、精准地提取有用信息和规律,已远超人工处理的极限。ML 恰恰扮演了"数据炼金术"的角色,作为将原始数据转化为结构化洞察与智能决策能力的关键技术载体。在"大数据时代",人们面临的挑战从"数据匮乏"转向了"信息过载"。传统的数据处理方法难以从海量、高维的数据中高效地提取有效模式。ML 提供了必要的技术手段,能够自动、快速地处理和分析这些数据,将其转化为有价值的洞察和可操作的知识,从而支持科学发现和商业决策。

3. ML 的核心价值

ML 通过算法自动从数据中挖掘模式、构建预测模型,并能够对未知数据进行推断与决

策，从而将原始数据转化为可重复利用的知识与生产力。更重要的是，其所构建的系统具备动态演进的能力：随着新数据的持续输入，模型可以不断迭代优化，展现出传统程序所不具备的环境适应性与进化潜力。ML 的根本目标在于让机器掌握“学习”这一人类核心智能行为。它通过计算模型模拟了人类从经验中学习、归纳和总结规律的过程。这不仅使计算机能够自动改进其性能，更使其在特定领域（如处理高维数据、进行高速计算）突破人类固有的认知极限，获得超越人类的学习与分析能力。归纳为：

（1）自动化知识构建：从数据中挖掘模式、构建预测模型，实现对未知数据的推断，将原始数据转化为可复用的知识与生产力；

（2）动态自适应进化：模型可通过持续输入新数据迭代优化，具备传统软件不具备的环境适应性；

（3）突破人类认知极限：在高维数据处理、高速计算等场景中，机器能发现人类忽略的规律，实现超越人类的分析能力。

4. 从人工判断到智能决策的范式转移

基于数据驱动的 ML 模型，能够实现决策过程的自动化与最优化。它通过严格的统计推理替代了依赖直觉的人工判断，显著减少了主观偏见，提升了决策的速度、准确性与一致性，从而在推荐系统、供应链管理等场景中大幅提升了运营效率。

5. 构建自适应系统需求

传统的软件系统在部署后功能便固化，而 ML 模型的核心优势在于其持续的进化能力。当新的数据可用时，模型可以不断地进行自我调整与优化，使其能够适应真实世界中快速变化的环境与需求，构建出真正具有**生命力**和**长期价值**的智能系统。

在 AI 时代，人们始终在解决一个问题：如何让机器感知、理解并解决不确定性的复杂问题？早期，人们试图通过精密编程为机器构建一套完备的“世界规则”，但很快便触碰到天花板。许多任务（如识别图像中的物体、理解自然语言的微妙之处）的底层规律，难以被人类穷尽地形式化定义。在此背景下，ML 应运而生，并迅速成长为 AI 领域最具活力的分支。其核心理念实现了从“授人以鱼”到“授人以渔”的根本性转变：人们不再执着于告诉机器“世界是什么”，而是为它设计能够从海量数据中自行发现“世界如何运作”的算法。

1.3　为什么需要 DL？

DL 作为 ML 的一个重要分支，其核心突破在于能够从原始数据中自动学习多层次、高抽象的“特征表示”，从而解决了传统 ML 方法在处理非结构化、高维度复杂数据时的根本性瓶颈。

1. 突破传统特征工程的局限

在图像、语音、文本等非结构化数据面前，传统 ML 方法严重依赖专家经验进行“特征工程”——即手工设计和提取对任务有效的特征。这一过程不仅耗时费力，且难以捕捉数据中深层次、本质的复杂模式。DL 通过构建具有多个隐层的神经网络，实现了“端到端”的学习：

模型直接从原始数据（如像素、波形、字符）开始，自动逐层抽象，从低级特征（如边缘、纹理）逐步组合形成高级语义概念（如物体部件、实体含义），从而彻底将人类从繁复且不完美的特征设计工作中解放出来。

2. **驾驭海量高维数据的核心能力**

在大数据时代，我们面对的数据不仅在规模上爆炸性增长，其维度（如图像的百万像素、文本的词汇表大小）也急剧增加。传统线性或浅层模型的学习能力很快会达到饱和，无法有效利用海量数据中蕴含的复杂信息。深度神经网络因其庞大的参数量和非线性变换堆叠，具备极强的“容量”，能够吸收海量数据的知识，学习极其复杂的映射关系，从而在处理高维数据时展现出压倒性的性能优势。

3. **解锁 AI 的感知能力**

DL 的崛起直接推动了 AI“感知智能”的革命性进步。以卷积神经网络（CNN）为代表的架构，在计算机视觉领域（如图像分类、目标检测）达到了超越人类的精度；以循环神经网络（RNN）和 Transformer 为核心的模型，则彻底改变了自然语言处理（如机器翻译、文本生成）和语音识别的技术格局。正是 DL，让机器第一次真正拥有了“看懂”图像、“听懂”语音、“理解”文本的可靠能力。

4. **实现更强大的表示与泛化**

DL 模型学习到的分层特征表示，往往具有更好的“可分离性”和“可转移性”。这意味着在大量数据上学到的特征，可以有效地迁移到其他相关任务上（迁移学习），或仅用少量标注数据就能获得良好性能（少样本学习）。这种强大的表示学习能力，是构建通用、鲁棒、可复用 AI 系统的基础。

简言之，DL 并非是对传统 ML 的简单替代，而是一次关键的范式升级。它将学习的重点从“如何设计特征”转向了“如何设计能够自动学习特征的模型架构”。当人们面对的问题涉及非结构化数据、蕴含复杂的非线性关系且拥有海量样本时，DL 便从一种可选项，变成了解决问题的必需品，成为推动当前 AI 浪潮走向深入的核心引擎。

1.4 走向深度：AI、ML、DL 的演进脉络

在技术浪潮的推动下，AI、ML 与 DL 已成为引领社会变革的核心力量。理解这三者间环环相扣、层层递进的发展过程，不仅是掌握当代智能技术的关键，更是预见未来创新的基石。本节将系统梳理从 AI 的宏大概括，到 ML 的实用化突破，再到 DL 的革命性进展，这一完整的技术演进史诗。

1. **AI 的兴起：从宏伟蓝图到初步探索**

AI 早期遭遇的挑战主要源于计算能力不足、数据匮乏以及处理复杂现实问题的能力有限。过于乐观的预期未能实现，导致资金和兴趣锐减，AI 领域经历了数次“寒冬”。现代 AI 源于一个充满想象力的宏伟蓝图：**让机器模拟人类的智能行为。**

理论奠基：1950 年，**艾伦·图灵**在其论文《计算机器与智能》中提出了著名的“图灵测试”，为判断机器是否具备智能设立了标准。1956 年的达特茅斯会议首次正式提出“AI”这一术语，标志着 AI 成为一个独立的学科。早期的研究者们乐观地认为，在数年内就能制造出具备人类智能的机器。

符号主义路径：早期 AI 的主流路径，认为智能源于对符号的操纵和逻辑推理。专家系统是其中的典型成果，它通过将人类专家的知识编码成规则，在特定领域（如医疗诊断）内解决问题。然而，符号主义在处理感知、理解不确定性问题等时显得力不从心。

连接主义路径：这一路径试图通过模拟人脑神经元的连接结构来实现智能，即神经网络模型。1958 年，弗兰克·罗森布拉特提出的感知机是首个可学习的神经网络模型，虽因局限性一度沉寂，却为未来的 DL 埋下了火种。

2. ML 的崛起：从“编程智能”到“学习智能”的范式转移

当传统 AI 研究在专家系统等符号主义方法中陷入瓶颈时，一种全新的范式正在悄然孕育——ML。这一革命性理念的核心转变在于：不再试图将人类知识直接编码输入计算机，而是让计算机能够像人类一样，通过观察和经验自动改进其性能。这种从“编程智能”到“学习智能”的范式转移，标志着 AI 研究进入了全新的发展阶段。

范式转变：从编程到学习。ML 的核心价值在于，它不再依赖于人类手动输入所有规则，而是通过算法让计算机从数据中自动学习，生成模型，从而具备预测和决策的能力。这标志着从“机械执行”到“自主归纳”的根本性转变。传统编程范式要求开发者明确定义每一个处理步骤和规则，这种方法在处理图像识别、语音理解等复杂任务时遇到了根本性挑战，很难将人类直觉性的认知过程完全形式化为明确的规则。ML 巧妙地绕过了这一困境，它将智能的实现方式从“教授规则”转变为“提供经验”。

这一转变的重要意义在于，它使得计算机能够处理那些人类知其然但不知其所以然的复杂问题。通过从大量实例中学习，ML 系统可以发现人类专家可能忽略的微妙模式和复杂关联，实现了智能系统构建方法的根本性革新。

理论基础与算法演进：ML 的数学基础可以追溯到多个学科的深度交叉融合。概率论为其提供了处理不确定性的框架，统计学为其贡献了从样本推断总体的方法，而优化理论则为其提供了寻找最优模型的数学工具。这种多学科的融合，为 ML 奠定了坚实的理论基础。

在算法层面，ML 经历了从浅层模型到复杂系统的演进过程。早期的线性模型、决策树等算法展现了基础学习能力，随后支持向量机（SVM）等方法的引入显著提升了模型的表达能力。而集成学习方法的出现，通过组合多个弱学习器来构建强学习器，进一步推动了 ML 性能的边界。

技术突破与应用拓展：ML 的崛起得益于多个关键因素的协同作用。计算能力的持续提升使得复杂算法的训练成为可能，互联网的普及提供了前所未有的大规模数据集，而理论研究的突破则不断推动着算法性能的进步。在应用层面，ML 技术迅速渗透到各个领域。从搜索引擎的排序算法到电商平台的推荐系统，从金融领域的信用评分到医疗行业的辅助诊断，ML 正在重塑传统行业的运作方式。这种广泛的应用反过来又推动了技术的进一步发展和完善，形成了良性的创新循环。

三大学习范式的确立:

监督学习:从带标签数据中学习输入输出映射,解决分类、回归问题(如垃圾邮件过滤);

无监督学习:从无标签数据中探索内在结构,通过聚类、降维发现数据规律(如市场细分);

强化学习:通过与环境交互试错,以奖励信号优化决策策略(如游戏 AI、机器人控制)。

随着研究的深入,ML 逐渐形成了三种基本的学习范式,每种范式都针对不同类型的学习问题。**监督学习**通过带有标签的训练数据,学习输入到输出的映射关系,主要解决分类和回归问题。这种范式在垃圾邮件过滤、风险评估等任务中取得了显著成功。**无监督学习**则在没有标签的数据中探索内在结构和分布模式,通过聚类、降维等技术发现数据的自然分组和简化表示。这种方法在市场细分、异常检测等领域展现出独特价值。**强化学习**采用与环境交互试错的方式,通过奖励信号来学习最优决策策略。这种范式在游戏 AI、机器人控制等序列决策问题中表现出色,展示了 ML 在复杂动态环境中的适应能力。

技术落地与应用拓展:计算能力提升、互联网海量数据支撑,推动 ML 渗透到搜索引擎、推荐系统、医疗诊断等领域,形成"技术进步—应用拓展"的良性循环。

1.5 DL 的革命:驱动 AI 的第二次浪潮

DL 的复兴并非简单的技术轮回,而是在数据、算力、算法三重驱动下的一场**范式革命**,突破了 ML 的瓶颈。当 ML 在诸多领域取得成功的同时,其在处理图像、语音、自然语言等非结构化数据时的局限性也逐渐显现。就在这个关键时刻,DL 以前所未有的力量登上历史舞台,开启了 AI 的第二次发展浪潮。这场革命不仅突破了传统 ML 的性能瓶颈,更重新定义了 AI 技术的可能性边界。

1. DL 复兴的三大核心驱动力

DL 并非一个全新的概念,其核心基础——神经网络,早在 20 世纪中期就已经被提出。然而,直到 21 世纪初,在三大关键因素的共同推动下,神经网络才以"DL"的名义实现强势回归:

(1) 数据可用性: ImageNet 等大型标注数据集的出现,使得训练深层神经网络成为可能。这些包含数百万图像的数据集,为模型学习复杂的视觉特征提供了充足的"养料"。

(2) 计算硬件大算力:计算机 GPU 的大规模并行计算能力,使得原本需要数月训练时间的深层网络,能够在数天甚至数小时内完成训练。这种计算效率的跃升,极大地加速了 DL 的研究和实验周期。

(3) 算法创新: ReLU 等新型激活函数的引入有效缓解了梯度消失问题,Dropout 等正则化技术显著改善了模型泛化能力,而批量归一化等技术则大幅提升了训练稳定性。这些创新共同构成了 DL 快速发展的技术基础。

2. 核心架构的创新突破

DL 的革命性进展,很大程度上得益于一系列专门化神经网络架构的提出和完善。这些创新架构突破了传统神经网络的局限,使得 DL 在各项任务中展现出前所未有的性能。

循环神经网络(RNN)及其改进型长短期记忆网络(LSTM):通过引入内部状态和门控机制,有效地处理序列数据的时序依赖性。这些架构在机器翻译、语音识别等任务中表现出色,为序列建模提供了强有力的工具。针对序列数据的时序特性,循环神经网络及其改进架构提供了有效的解决方案。传统 RNN 通过循环连接维持内部状态,理论上能够处理任意长度的序列信息。然而,其在实践中面临着梯度消失和长期依赖学习困难等挑战。长短期记忆网络(LSTM)通过精妙的门控机制解决了这些难题。输入门、遗忘门和输出门的协同工作,使模型能够自主决定信息的保留与遗忘,从而有效地捕捉序列中的长期依赖关系。随后出现的门控循环单元(GRU)通过简化门控结构,在保持相当性能的同时提升了计算效率。这些架构在机器翻译、文本生成、语音识别等序列建模任务中取得了突破性进展。

卷积神经网络(CNN):CNN 通过局部连接、权重共享和空间下采样等机制,极大地提升了图像处理的效率和性能。从 AlexNet 到 ResNet,CNN 架构的不断深化,在图像分类、目标检测等任务中持续刷新性能纪录,甚至超越了人类水平的识别精度。从 AlexNet 在 2012 年 ImageNet 竞赛中以超越第二名 10 个百分点的惊人成绩夺冠,到 VGGNet 通过堆叠小型卷积核构建更深的网络结构,再到 ResNet 创新性地引入残差连接解决深度网络梯度消失问题,CNN 的不断发展推动着图像识别准确率持续突破,甚至在特定任务上超越了人类视觉的识别精度。CNN 通过其独特的结构设计,彻底改变了计算机视觉领域的发展轨迹。其核心创新体现在三个关键机制:局部感受野通过关注图像的局部区域而非全连接,显著减少了参数数量;权重共享机制使得不同位置的相同特征能够被同一组参数检测,极大提升了模型效率;空间下采样则通过池化操作逐步压缩特征图尺寸,在保留重要信息的同时增强模型的平移不变性。

Transformer 架构:Transformer 的提出标志着 DL 领域的又一次范式转移。基于自注意力机制的 Transformer 模型,不仅在处理长序列时表现出更好的性能,而且为大规模预训练模型的发展奠定了基础,直接催生了当前的大语言模型浪潮。2017 年提出的 Transformer 架构标志着 DL 领域的又一次重大范式转移。其核心的自注意力机制使模型能够直接计算序列中任意两个位置之间的关联强度,从而更好地捕捉全局依赖关系。Transformer 的并行计算特性使其在处理长序列时显著优于 RNN 系列架构。基于 Transformer 的预训练模型,如 BERT、GPT 系列等,在自然语言处理领域掀起了一场革命。这些模型通过大规模无监督预训练获得通用的语言理解能力,再通过微调适配到具体下游任务,极大地推动了自然语言处理技术的发展。

DL 从根本上改变了人们构建智能系统的方法论,通过**足够深的神经网络层次**、**海量的训练数据**和**强大的计算资源**,机器能够自动学习从低级信号到高级概念的复杂映射。

在**计算机视觉**领域,DL 使得图像分类、目标检测、语义分割等任务的准确率实现了质的飞跃。从医学影像分析到自动驾驶视觉感知,DL 已经成为相关应用的核心技术。

在**自然语言处理**领域,基于 DL 的词向量表示、序列到序列模型等技术,彻底改变了传统语言处理方法。机器翻译、文本生成、情感分析等任务的性能得到了前所未有的提升。

在**语音技术**领域,DL 的引入显著提升了语音识别和合成的准确度与自然度,推动了智能助手、实时转录等应用的快速普及。

更重要的是,DL 展现出了强大的**特征学习能力**。与传统 ML 依赖人工特征工程不同,DL 能够自动从原始数据中学习具有层次结构的特征表示,这种端到端的学习方式极大地简

化了系统构建流程,提升了开发效率。

3. **推动 AI 民主化与产业变革**

DL 的革命不仅体现在技术进步上,还深刻地影响了 AI 的研发模式和产业生态,开发人员无需深谙数据科学的复杂技能,也能通过调用高级 API 和预训练模型,快速构建出强大的数据驱动应用。

开源框架的繁荣,如 TensorFlow、PyTorch 等的出现,极大地降低了 DL 的技术门槛,推动了 AI 技术的民主化进程。研究人员和开发者可以基于这些成熟的工具快速实现和验证创新想法。

预训练模型的普及改变了 AI 应用的开发范式。通过迁移学习,开发者在大规模预训练模型的基础上,使用相对较少的数据快速构建特定领域的应用解决方案,显著提升了开发效率。

产业应用的广度不断扩展。从互联网公司的推荐系统,到制造业的质量检测,再到金融业的风险控制,DL 正在重塑各行各业的运作方式,创造着巨大的经济价值和社会效益。

4. **未来展望与挑战**

尽管 DL 取得了令人瞩目的成就,但其发展仍然面临着重要的挑战和机遇:

可解释性的缺乏限制了 DL 在关键决策场景中的应用。如何理解和解释深度模型的决策过程,建立人类对 AI 系统的信任,是亟待解决的重要问题。

计算效率的挑战随着模型规模的扩大而日益凸显。如何在保持性能的同时降低模型的计算和能耗成本,推动 DL 向边缘设备扩展,是影响技术普及的关键因素。

多模态学习的发展预示着新的突破方向。如何让 DL 模型更好地理解和处理视觉、语言、声音等不同模态信息之间的复杂关联,将是实现更通用 AI 的重要路径。

DL 革命仍在继续,它不仅是技术能力的突破,更是思维方式的转变。这场革命告诉人们,通过构建足够深度的网络结构,配合大规模数据和强大算力,机器能够学习到前所未有的复杂模式和抽象概念。正如 DL 先驱 Geoffrey Hinton 所言:“人们已经找到了一种让神经网络学习层次化概念的方法,这就像是发现了新的科学原理。”这个“原理”正在继续推动着 AI 向前发展,迈向更加智能的未来。

1.6 人工智能(AI)

AI,顾名思义,为让机器像人一样思考、学习和决策的科学与技术。AI 旨在创造能够执行通常需要人类智能的任务的系统。其研究范畴建立在几个相互关联的核心概念之上,这些概念构成了传统“符号主义 AI”的基石,并与现代数据驱动的方法相辅相成。包括:

思考→像人类一样推理、规划和解决问题。

学习→从数据和经验中自动改进,无需被重新编程。

决策→在复杂情况下做出判断和选择。

感知→理解和解释周围的世界(如看懂图像、听懂语音)。

一个简单比喻:若把制造智能机器看作“造车”,则 AI 为要造的“车”本身。它是一个宏

大的总目标和愿景。为了实现这个目标,主要发展出了三条技术路径:

AI 目前分为弱 AI、强 AI 和超 AI。

(1) 弱 AI:弱 AI(ANI),只专注于完成某个特定的任务,例如语音识别、图像识别和翻译等,是擅长于单个方面的 AI。它们只是用于解决特定的具体类的任务问题而存在,大都是统计数据,以此从中归纳出模型。

(2) 强 AI:强 AI(AGI),属于人类级别的 AI,在各方面都能和人类比肩,它能够进行思考、计划、解决问题、抽象思维、理解复杂理念、快速学习和从经验中学习等操作,并且和人类一样得心应手。

(3) 超 AI:超 AI(ASI),在几乎所有领域都比最聪明的人类大脑都聪明许多,包括科学创新、通识和社交技能。在超 AI 阶段,AI 已经跨过"奇点",其计算和思维能力已经远超人脑。此时的 AI 已经不是人类可以理解和想象。AI 将打破人脑受到的维度限制,其所观察和思考的内容,人脑已经无法理解,AI 将形成一个新的社会。

目前人们仍处于弱 AI 阶段。传统 AI(规则驱动):像教小孩算术,人们先告诉它所有规则(如 1+1=2),它根据规则进行计算。这种方式精确但僵化,难以处理模糊和复杂的问题。

现代 AI(数据驱动):像教小孩认猫,人们不定义规则,而是给它看成千上万张猫的图片,让它自己总结出猫的特征。这种方式灵活且强大,是当前 AI 爆发的主流方法,其核心为 ML,特别是 DL。

AI 的未来将更加注重跨学科的整合,包括认知科学、心理学、神经科学等,以更好地模拟和理解人类智能。人们常听到的 ML 和 DL 是实现 AI 的重要方法。

1.7　交融与未来:迈向更通用的智能

1. 三者的核心关系

如图 1-1 可视化地展现出它们三者的关系,表示:AI⊃ML⊃DL。三者如同"大树生态":AI 是树根,奠定发展方向;ML 是主干,支撑核心技术;DL 是繁茂枝干,延伸出具体能力,三者协同演进,形成持续进化的技术体系。

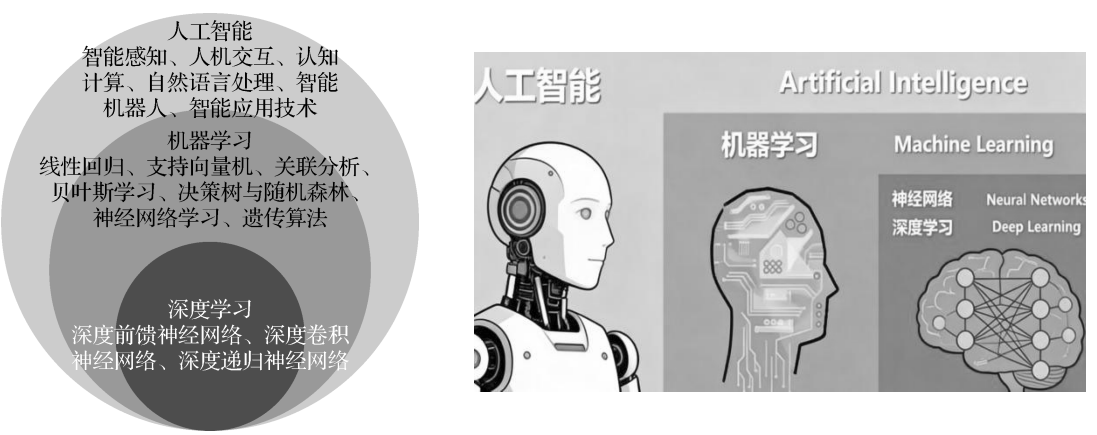


图 1-1　AI、ML 和 DL 三者关系

ML是一种实现AI的方法,DL是一种实现ML的技术。AI包含智能感知、人机交互、ML、认知计算、自然语言处理、智能机器人、智能应用技术等;ML包含线性回归、支持向量机、关联分析、贝叶斯学习、决策树与随机森林、神经网络学习、遗传算法等;DL包含深度前馈神经网络、深度CNN、深度递归神经网络等。

2. 三者应用场景的区别与联系

AI的研究领域在不断地扩大,包括专家系统、ML、进化计算、模糊逻辑、计算机视觉、自然语言处理、推荐系统等,并且目前的科研工作都集中在弱AI这部分。ML研究直接来源于早期的AI领域,传统的算法包括决策树、聚类、贝叶斯分类、支持向量机、EM、Adaboost等。从学习方法上来分,ML可以分为监督学习(如分类问题)、无监督学习(如聚类问题)、半监督学习、集成学习、DL和强化学习。传统的ML算法在指纹识别、人脸检测、特征物体检测等领域的应用基本达到了商业化的要求或特定场景的商业化水平。

在当今数字化时代,AI、ML、DL已经成为推动技术进步和创新的关键力量。这些技术不仅改变了人们与机器的互动方式,还在医疗、金融、交通、教育等多个领域产生了深远影响。但是,AI、ML、DL三者的关系并非取代,而是**深度融合、协同演进**。如图1-1所示,AI是最宏大的范畴,旨在创造智能机器;ML是实现AI最成功、最核心的路径;而DL是ML的一个子领域,它使用多层神经网络来模拟人脑处理信息的方式,是ML中目前最强大、最前沿的分支,它利用深度神经网络来处理复杂问题。

DL本来并不是一种独立的学习方法,其本身也会用到监督学习和无监督学习方法来训练深度神经网络,但由于近年来该领域发展迅猛,一些特有的学习手段相继被提出(如残差网络),因此越来越多的人将其单独看作一种学习方法。主要应用在互联网、安防、金融、智能硬件、医疗、教育等行业,人脸技术、图像识别、智能监控、文字识别、语义分析等领域。

3. ML与DL的选择

其选择并非孰优孰劣,而是取决于具体的应用场景和资源约束。ML可以看作是“手工打造的精巧工具”,它在数据量有限、需要可解释性或处理结构化数据时非常高效。而DL则像是“能够自我演化的超级大脑”,它通过吞噬海量数据和算力,在解决人类直觉擅长但难以形式化的复杂感知任务上展现出无可比拟的威力。它们是数据科学工具箱中不同但互补的强大工具,具体区别如表1-1所示。

表1-1 ML与DL的区别

	ML	DL
定义	计算机通过数据和算法改进自身性能的方法,涵盖多种学习范式	ML的一个子集,专注于使用包含多个处理层的深度神经网络进行学习
数据依赖与特征工程	依赖特征工程:需要领域专家手动设计和提取特征,模型性能严重依赖于特征的质量	自动特征学习:自动从原始数据中学习特征,能够直接从原始数据(如图像像素、文本)中自动学习多层次的特征表示,无需繁琐的手工特征工程

续　表

	ML	DL
模型复杂度与可解释性	相对简单，例如决策树、线性回归	复杂得多，包含许多层的神经网络
数据需求	模型相对简单、透明（如决策树、线性模型），可以处理小到中等规模的数据集，可解释性强，易于理解和调试	通常需要大规模的数据集，包含数百万甚至数十亿参数的复杂“黑箱”，可解释性差，难以理解其内部决策逻辑
计算资源	相对较少，可以在普通硬件上运行，需求相对较低，大部分算法可以在 CPU 上于合理时间内完成训练	需求极高，依赖高性能 GPU 进行大规模并行计算，训练过程耗时且耗能
应用场景	数据挖掘、预测分析、推荐系统等，数据挖掘、预测分析、推荐系统、欺诈检测、客户分群等	图像识别、自然语言处理、语音识别、自动驾驶、高级游戏 AI 等
代表性算法	决策树、支持向量机（SVM）、随机森林	CNN、循环神经网络（RNN）、生成对抗网络（GAN），Transformer
灵活性	较高，可以根据具体任务调整算法	较低，主要是通过调整网络架构和超参数
硬件依赖	对硬件无特殊要求，通用计算机即可满足	高度依赖 GPU 集群和专门的 AI 加速芯片（如 TPU、NPU）

4. 发展核心趋势

AI、ML 和 DL 三者之间并非简单的替代关系，而是一种层层递进、深度融合的协同演进体系。AI 作为最宏观的概念，定义了让机器具备智能的终极目标；ML 作为实现这一目标最有效的路径，提供了从数据中学习规律的方法论；而 DL 则作为 ML 领域中最强大的工具之一，通过深层神经网络突破了传统 ML 的技术瓶颈。这种关系就像一棵不断生长的大树：AI 是树根，奠定发展方向；ML 是主干，支撑核心技术；DL 则是其中最繁茂的枝干，不断延伸出新的能力。三者相互促进——DL 的突破推动了 ML 的边界，ML 的进步则让 AI 的愿景不断成为现实，形成了一个持续进化的技术生态系统。AI、ML、DL 的发展正从技术突破早期走向与产业和科学研究的深度融合，其未来的演进并非单一技术的直线攀升，而是呈现出多路径、多维度融合发展的态势，具体如表 1-2 所示。

表 1-2　AI、ML 和 DL 的发展趋势

维度	核心趋势	具体表现
算力与效率	从集中式训练走向分布式推理	大规模 AI 数据中心建设；端侧设备推理优化；自适应计算降低智能成本
模型与算法	从单一模态走向协同与自主	多模态融合；AI 智能体（Agents）自主决策；世界模型与具身智能
应用与赋能	从工具自动化走向产业与科学重塑	AI 与人类深度协作；驱动科学研究范式变革（AI for Science）；催生 AI 原生应用与超级应用
控制与治理	从追求能力走向可信与负责任	可解释性（XAI）与安全对齐；合成数据的使用；完善 AI 安全治理体系

5. 创新

从“替代人力”到“增强人智”:业界共识正从“AI 取代人类”转向“AI 与人类协作”。

○ 融合创新:将 DL 的感知能力与知识图谱、符号推理相结合,发展神经符号 AI,以解决需要可解释性和逻辑推理的复杂任务。

○ 追求效率与可信:研究重点正从单纯追求性能,转向构建更高效(如模型压缩)、更安全、更可靠、更符合伦理的下一代 AI 系统。

○ 探索前沿:强化学习与 DL 的结合(深度强化学习),以及在科学发现(AI for Science)等领域的应用,正在不断拓展智能的边界。

○ 智能正成为一种可规模化的商品:随着算力基础设施的完善和模型效率的提升,智能本身的成本正在急剧下降。获取公开知识的时间趋近于零,这使得 AI 能力能够像电力一样,被各行各业便捷、低成本地使用,从而引发大范围的产业变革。

○ AI 的“思维方式”在进化:未来的 AI 系统将不再仅仅被动响应指令,而是能够主动参与。这体现在 AI 智能体(Agents)能自主规划并执行复杂任务;同时,科研领域正探索“无微调”技术,让模型能不改变底层参数就快速适应新任务,这类似于人类凭借一本不断更新的操作手册就能学会新技能,而非每次都重塑大脑。

AI 将主要处理海量信息分析和模式识别,而人类则专注于需要价值判断、情感共鸣和伦理权衡的再决策。这种协同模式旨在放大人类的智慧,而非替代它。从 AI 的宏伟梦想,到 ML 的实用化路径,再到 DL 的革命性突破,这是一段智能技术不断聚焦、深化并再爆发的壮丽历程。理解这一发展过程,意味着人们不仅掌握了技术工具,更获得了洞察未来的历史视角。总的来说,AI、ML 和 DL 的未来并非通向一个全知全能的“超级 AI”,而是走向一个人机深度协作、智能无处不在的世界。技术发展将更加注重实用性、经济性和可持续性。

本章习题

1. 概念辨析:AI、ML 和 DL 三者之间是什么关系?请用一句话概括,并举例说明它们在实际应用中的区别。

2. 数据的重要性:在 ML 中,数据常被比喻为“智能系统的生命力源泉”。请简述训练数据、验证数据和测试数据各自的作用,并解释为什么需要将数据集划分为这三部分。

3. 学习范式对比:监督学习、无监督学习和强化学习是 ML 的三大范式。请分别说明它们的特点、典型任务及一种代表性算法。

4. 算法原理理解:梯度下降法是 ML 和 DL 中常用的优化算法。请简要描述其基本原理,并比较批量梯度下降、随机梯度下降和小批量梯度下降的优缺点。

5. 线性回归与逻辑回归:线性回归和逻辑回归都是经典的 ML 算法。请说明:

(1) 它们分别解决什么类型的问题?

(2) 逻辑回归如何通过 Sigmoid 函数将线性回归的输出转化为概率?

6. 决策树与随机森林:决策树和随机森林都属于基于树的模型。请解释:

(1) 决策树是如何进行特征选择和分裂的?

(2) 随机森林如何通过“集成学习”提升模型的泛化能力?

7. SVM 与核函数：支持向量机（SVM）在处理非线性可分数据时，常使用核函数。请说明：

（1）SVM 的基本思想是什么？

（2）为什么需要核函数？列举两种常见的核函数并简述其特点。

8. 聚类与降维：K-means 和 PCA 分别是聚类和降维的经典算法。请简述：

（1）K-means 算法的步骤；

（2）PCA 的主要思想及其与 LDA 的区别。

第二章　ML 十大经典算法

本章将系统介绍 ML 领域的十大经典算法，它们是 AI 领域的基石，历经时间考验，至今仍在工业界和学术界被广泛应用。理解这些算法，不仅能让人们掌握解决实际问题的强大工具，更能为人们后续深入 DL 奠定坚实的数学与思想基础。

2.1　引言：ML 的基本思路与流程

ML 解决现实问题的基本思路可抽象为三个步骤：

第一步　**问题建模**：将现实问题抽象为数学模型，并明确模型中各参数的意义。

第二步　**模型求解**：利用数学方法对模型进行求解。

第三步　**评估验证**：评估该模型是否真正解决了实际问题。

一个直观的例子：预测芒果质量

假设任务是从市场上挑选好芒果。我们首先收集一批芒果样本作为训练数据，记录每个芒果的特征（如颜色、大小、形状、产地）及其质量标签（如“香甜”“多汁”“成熟”）。目标是设计一个学习算法，从这些数据中学习特征与质量之间的关联模型。下次购买时，面对一个新芒果（测试数据），便可使用该模型根据其特征来预测其质量。

图 2－1　芒果质量预测过程

如图 2－2 所示，一个完整的 ML 项目通常遵循一个系统化、可迭代的流程：

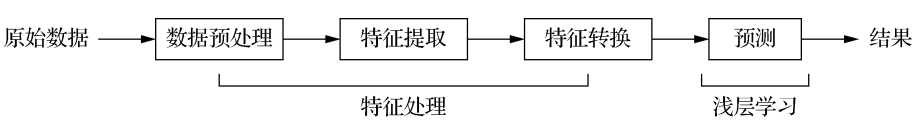


图 2－2　模式识别流程

ML 模型的工作流程是一个系统化的迭代过程：

第一步　**数据划分**：将数据分为训练集、验证集和测试集。

第二步　**模型训练与调优**：在训练集上构建模型，并用验证集评估表现、调整超参数。

第三步　**再评估**：在测试集上对优化后的模型进行再性能评估。

第四步　**部署与迭代**：若模型达标则投入应用；若不达标，则需通过增加数据、优化特征或调整模型等方式进行迭代优化。

这一闭环流程确保了模型从开发、验证到部署的全生命周期管理。

ML 在解决问题的方法论上，处于传统编程与高级智能应用之间的关键位置。与传统编程不同，它并非由人类直接编写解决问题的具体规则和流程，而是通过将大量数据“喂”给计算机，让机器从数据中自动学习和归纳出完成任务所需的内在规律与模式（即模型）。这个过程类似于统计学，但更侧重于利用计算能力从大数据中自动发现复杂的关联并进行预测。通过这种方式训练出的模型构成了各类智能应用的核心引擎，使其能够自主执行识别、判断和预测等任务，从而实现了从“执行指令”到“自主决策”的能力跃迁。

ML 的五大思想流派从不同哲学视角探索了实现智能的路径：符号主义基于逻辑与规则，试图用清晰的符号系统模拟人类的理性思维；贝叶斯派聚焦于不确定性，通过概率推理来更新对世界的认知；联结主义受大脑启发，通过神经网络的自组织与学习来识别复杂模式；进化主义模拟自然选择过程，利用遗传算法等进化策略在解空间中寻找最优方案；而 Analogizer（类推主义）则强调从相似性中学习，通过对比样本间的约束关系（如支持向量机）来构建决策边界。这五大流派各有侧重，共同构成了 ML 丰富而多元的理论基础与方法论体系。

ML 与模式识别、统计学习、数据挖掘、计算机视觉、语音识别和自然语言处理等领域有着很深的联系，与其他领域处理技术的结合，形成了计算机视觉、语音识别、自然语言处理等交叉学科，如图 2－3 所示，由此产生了 ML 的很多任务和很多方法。

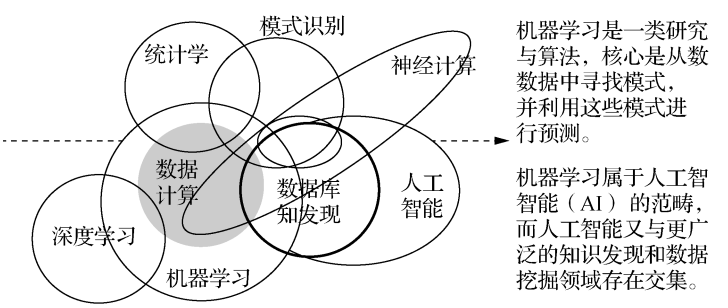


图 2－3　ML 的多领域结合

2.2　ML 方法分类

根据在建模和模型训练过程中有没有利用训练样本的标签，将 ML 算法可以分为 4 类：监督学习、无监督学习、半监督学习和强化学习，分类框架如图 2－4 所示：

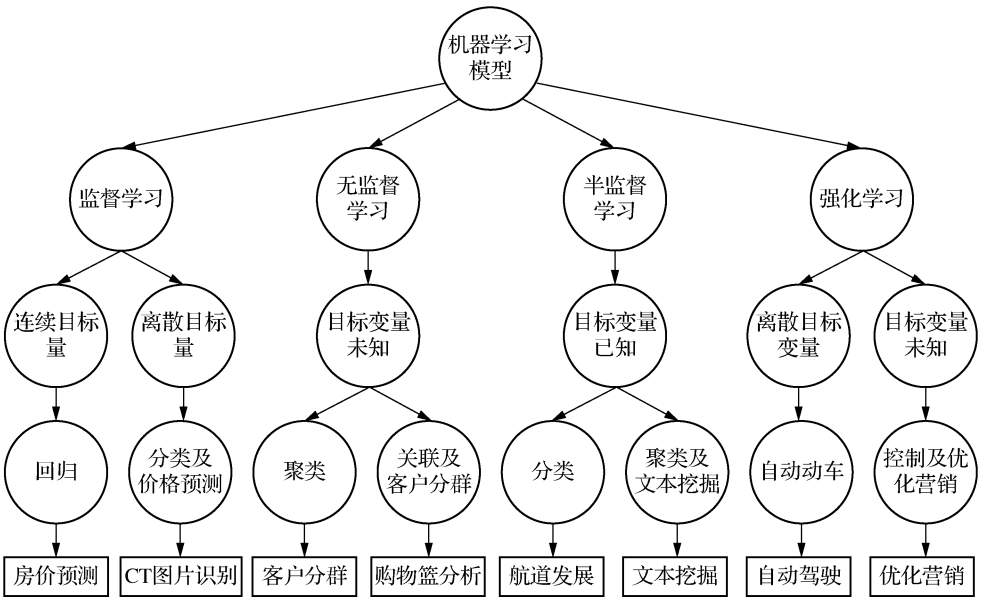


图 2-4　ML 的分类框架图

1. 监督学习

监督学习是 ML 的一种核心范式,它通过已标注的训练数据集(即包含“标准答案”的样本)来构建预测模型,其本质是学习输入数据与输出标签之间的映射关系。在这一过程中,系统通过不断比较模型预测结果与真实标签的差异,持续优化模型参数,再目标是使模型能够对新出现的未知数据做出准确的分类或回归预测。该方法的有效性高度依赖于标注数据的质量和代表性,因此被广泛应用于自然语言处理、图像识别、医疗诊断等需要精确推断的领域。主要算法包括神经网络、支持向量机、最近邻居法、朴素贝叶斯法、决策树等。

核心思想与过程:监督学习的过程可以概括为“从已知答案中学习规律”。其输入是一组标注了分类标签或目标值的样本集,这组数据为学习过程提供了一组“标准答案”。系统通过学习,建立一个从输入到输出的映射函数(即模型)。当新的、未标记的数据到来时,此模型便可依据已学的规律对其输出进行预测。在模型构建阶段,监督学习会持续将模型的预测结果与训练数据的真实结果进行比较,通过不断调整模型内部参数来缩小预测值与真实值之间的差异,直至模型在训练数据上达到预期的准确率。

基本要素与要求:训练数据是监督学习的基础,要求样本的分类标签已知且精确。标签的精确度越高,样本越具代表性,再学习所得模型的准确度通常也越高。

学习目标:学习一个从输入 X 到输出 Y 的映射函数 $Y=f(X)$。

监督学习主要解决两大类问题:分类和回归。分类是预测离散的类别标签。例如,在反垃圾邮件系统中,将邮件判断为“垃圾邮件”或“非垃圾邮件”;在手写体识别中,判断数字是“0”到“9”中的哪一个;回归是预测连续的数值。例如,根据房屋面积、位置等特征预测其市场价格。

2. 无监督学习

无监督学习是ML的重要分支，通过未标注的原始数据集自主挖掘数据内在结构、分布规律或隐藏关联。在这一过程中，系统通过分析数据本身的统计特性和分布模式，自动发现数据中存在的自然分组、关联规则或简洁表示，再目标是让模型能够识别数据的内在组织结构，为数据理解和知识发现提供支持。该方法的有效性高度依赖于数据本身的质量和分布特性，因此，被广泛应用于市场分析、异常检测、推荐系统等需要探索性数据分析的领域。主要算法包括K均值聚类、主成分分析、关联规则挖掘和自编码器等。

核心思想与过程：无监督学习的过程可以概括为“从无序数据中发现隐藏结构”。其输入是一组未标注的原始数据样本，这些数据不包含任何明确的“标准答案”。系统通过特定的算法探索数据内在的分布模式和组织结构，建立数据的压缩表示或分组结构。当新的数据到来时，此模型可以依据已发现的数据结构对其进行归类、重构或异常判断。在模型构建阶段，无监督学习会持续优化数据的表示或分组效果，通过不断调整模型参数来提升数据表示的紧凑性或分组的内在一致性，直至模型能够很好地捕捉数据的内在结构。

基本要素与要求：原始数据是无监督学习的基础，要求数据具有一定的内在结构和分布规律。数据的质量越高，内在结构越明显，学习所得模型的效果通常也越好。

学习目标：学习数据的底层结构或分布特性，发现数据中的自然分组 $Z=g(X)$ 或数据的低维表示。

无监督学习的核心任务包括三类：

(1) 聚类：将数据划分为有意义的组别。例如，在客户细分中，根据消费行为将客户划分为不同的群体；在文档分析中，将相似的文档自动归类。

(2) 降维：在保留主要信息的前提下减少数据维度。例如，在数据可视化中，将高维数据投影到二维或三维空间；在特征工程中，提取数据的最主要特征。

(3) 关联规则挖掘：发现数据项之间的有趣联系。例如，在购物篮分析中，发现商品之间的购买关联关系；在医疗诊断中，发现症状与疾病之间的潜在联系。

3. 半监督学习

半监督学习是介于监督学习与无监督学习之间的重要范式，核心思想是结合少量标注样本与大量未标注样本训练模型，在保证预测性能的同时显著降低数据标注成本。系统通过挖掘未标注数据蕴含的分布结构信息，辅助建立更准确的输入—输出映射关系，再目标是利用有限标注预算构建强泛化能力的预测模型。该方法有效性既依赖标注数据的准确性，受益于未标注数据揭示的底层结构，广泛应用于文本分类、医学影像分析、网络内容审核等标注成本高但未标注数据易得的领域。

核心思想与过程：半监督学习的过程可以概括为“借助未标注数据增强监督学习”。其输入由少量标注样本和大量未标注样本共同组成，标注样本提供直接的监督信号，而未标注样本则揭示数据的整体分布特性。系统首先从标注数据中学习初步的预测模型，同时利用未标注数据探索数据的流形结构或分布规律，进而优化决策边界的位置。当新的未知数据到来时，模型能够基于学习到的数据结构和映射关系做出更准确的预测。在模型构建阶段，半监督学习会协调利用两种信息来源：一方面确保在标注数据上的预测准确性，另一方面保

证模型与数据整体分布结构的一致性。

基本要素与要求：标注数据的质量和未标注数据的分布覆盖范围是半监督学习成功的关键。要求标注样本具有代表性，未标注样本与标注样本来自相同的分布，且数据本身具有一定的聚类假设或流形结构。

学习目标：学习一个从输入 X 到输出 Y 的映射函数 $Y=f(X)$，同时满足在标注数据上的准确性要求和在整体数据分布上的结构性约束。

半监督学习主要解决两类问题：

（1）稀缺标注场景的分类与回归：在标注样本有限但未标注数据丰富的任务中构建预测模型。例如，在医学影像诊断中，专家标注成本高昂，但可以利用大量未标注影像提升疾病识别准确率；在网络评论情感分析中，人工标注的评论有限，但海量未标注评论有助于建立更稳健的情感分类器。

（2）数据分布探索与模型正则化：利用未标注数据揭示的分布特性改善模型泛化能力。例如，在语音识别中，通过大量未标注语音数据学习声学特征的流形结构，提升识别系统的鲁棒性；在异常检测中，结合少量已知异常样本和大量正常样本，更精确地界定正常与异常行为的边界。

典型算法包括图推理算法、拉普拉斯支持向量机、伪标签方法、一致性正则化方法等。这类算法通常先对未标注数据的分布结构建模，再利用标注数据建立准确预测函数，在平衡标注成本与模型性能的现实场景中极具价值，是现代 ML 系统的重要技术路径。

4. 强化学习

强化学习是 ML 的另一个重要分支，它通过智能体与环境的持续交互，基于获得的奖励信号来学习最优决策策略。在这一过程中，系统通过不断尝试不同的行动并观察环境反馈，逐步调整决策策略，再目标是使智能体能够在复杂环境中学会达成长期目标的最优行为序列。该方法的有效性高度依赖于环境反馈的设计和探索策略的选择，因此，被广泛应用于游戏 AI、机器人控制、自动驾驶等需要序列决策的领域。主要算法包括 Q 学习、策略梯度方法、深度强化学习等。

核心思想与过程：强化学习的过程可以概括为“在试错中学习最优决策”。其输入是环境状态和奖励信号，不包含明确的行动指导。智能体通过与环境交互，根据当前状态选择行动，环境随之转移到新的状态并给予相应的奖励反馈。系统通过评估行动的价值来建立决策策略。当面临新的环境状态时，智能体可以依据学习到的策略选择最优行动。在学习过程中，强化学习会持续更新行动的价值估计，通过不断调整策略参数来最大化长期累积奖励，直至策略收敛到最优解。

基本要素与要求：环境模型和奖励函数是强化学习的基础，要求环境具有马尔可夫性，奖励信号能够有效引导学习方向。环境反馈越明确，探索越充分，学习所得策略通常也越优。

学习目标：学习一个从环境状态到最优行动的映射策略 $\pi=h(S)$，以最大化长期累积奖励。

强化学习的算法体系主要包括：

Q-learning：基于值函数的经典表格方法，通过建立状态-动作值函数来指导决策；

深度 Q 网络(DQN):结合 DL 与 Q-learning,使用神经网络近似值函数,能够处理高维状态空间;

策略梯度方法:直接优化策略函数,特别适用于连续动作空间的复杂决策问题。

强化学习主要解决三类问题:

(1) 序列决策问题:需要在多个时间步上连续做出决策的任务。例如,在围棋对弈中,需要规划一系列的落子决策;在机器人控制中,需要连续调整关节动作。

(2) 延迟奖励问题:行动的效果需要经过多个时间步才能显现的任务。例如,在投资决策中,当前的投资行动可能需要很长时间才能看到回报;在游戏通关中,某些关键决策直到游戏后期才显现价值。

(3) 探索与利用权衡问题:需要在尝试新行动和利用已知有效行动之间保持平衡的任务。例如,在新药研发中,需要在尝试新化合物和优化已知有效化合物之间做出权衡;在在线广告推荐中,需要在探索新广告类型和利用已知有效广告之间取得平衡。

在应用层面,强化学习已在游戏 AI(如 AlphaGo)、机器人控制、自动驾驶、智能资源调度等领域展现出巨大潜力。随着深度强化学习的持续发展,这一范式在解决现实世界复杂序列决策问题方面正发挥着越来越重要的作用。

与监督学习和无监督学习相比,强化学习具有以下独特优势:

时序决策能力:专门适用于需要连续决策的序列问题;

延迟奖励处理:能够有效处理行动与再奖励之间存在时间延迟的复杂场景;

在线学习特性:支持在与环境实时交互的过程中持续改进和优化策略;

自主决策能力:不需要预先标注的训练数据,能够通过与环境互动自主学习和进化。

根据其目标和输出类型,将 ML 划分为分类、回归、聚类和生成四类:

(1) 分类:分类任务旨在将输入数据划分到预定义的离散类别中。其输出结果为有限的类别标签,核心在于实现从数据特征到类别标签的映射。根据类别数量可分为二分类(如真/假判断)与多分类(如动植物种类识别)。该任务广泛应用于垃圾邮件过滤、人脸识别、情感分析等需要明确分类结论的场景,如判断一封邮件是“垃圾邮件”还是“正常邮件”。

(2) 回归:回归任务专注于预测与输入数据相关联的连续数值目标。其输出为具有实际意义的数值量,重点在于构建特征与连续目标变量之间的映射关系。依据变量间关系特性,可分为线性回归与非线性回归。典型应用场景包括房价评估、股市趋势分析与生物年龄预测等数值预测领域,如根据房屋大小和地段,预测其市场价格。

(3) 聚类:聚类任务通过分析数据内在的相似性特征,将未标注的数据样本自动划分为具有意义的群组。其输出为基于数据自然分布形成的簇结构,可分为硬聚类(样本唯一归属)与软聚类(样本概率归属)两种模式。该技术主要应用于客户细分、图像区域划分和社交网络社群发现等无监督学习场景,如根据顾客的购买行为,自动将他们分成不同的群体,以便精准营销。

(4) 生成:生成任务的核心在于学习训练数据的潜在分布规律,进而创造具有相似特征的新颖数据样本。其输出为符合原始数据特征的新生成内容,可根据生成条件分为条件生成与无条件生成两种范式。这一前沿领域在艺术创作、智能写作和语音合成等创造性应用中展现出巨大潜力。典型应用包括 AI 绘画、对话机器人、虚拟人物创作,如看过无数人脸照片后,AI 可以画出一张世界上并不存在的逼真人脸。

2.3 10个经典的ML方法

在ML领域,十大基础算法构成了解决各类问题的核心工具集:线性回归和逻辑回归分别预测与二分类问题;决策树及其集成版本随机森林、梯度提升通过树模型实现高精度的分类与回归;支持向量机(SVM)和K近邻(KNN)分别依靠最大化分类间隔和邻近样本投票进行预测;朴素贝叶斯基于概率论实现高效分类;K均值作为经典聚类算法探索数据内在分组结构;而降维算法(如PCA)则通过压缩数据维度来提升计算效率和可视化效果。这些算法覆盖了监督学习、无监督学习和集成学习的主要范畴,为处理分类、回归、聚类、降维等基础任务提供了坚实的技术基础。下面介绍常用的5种方法。

2.3.1 梯度下降法

梯度下降法是一种广泛应用于ML和DL领域的优化算法,其核心目标是通过迭代方式寻找函数的最小值点。该算法的基本原理是沿着函数梯度(导数)的反方向逐步调整参数,因为梯度方向指示了函数值增长最快的方向,其反方向则是函数值下降最快的路径。

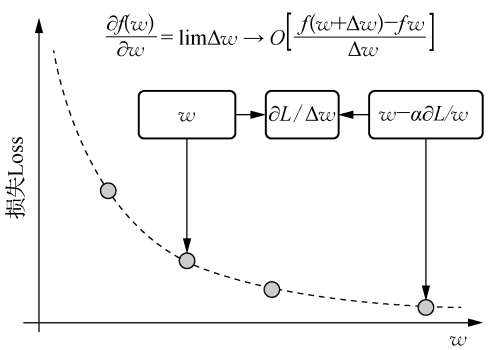


图2-5 梯度下降法

梯度下降法是一种通过迭代寻找函数最小值的优化算法。其核心是沿函数梯度(增长最快方向)的反方向逐步调整参数。

- **基本原理:**新参数=旧参数−学习率×梯度。
- **执行步骤:**初始化参数→计算梯度→更新参数→判断收敛。

第一步 **初始化参数:**随机选择参数的初始值作为搜索起点;

第二步 **梯度计算:**计算当前参数位置处损失函数的梯度;

第三步 **参数更新:**按照公式新参数=旧参数−学习率×梯度调整参数值;

第四步 **收敛判断:**重复第二步和第三步,直到梯度接近零或达到预设迭代次数。

根据计算梯度时使用的数据量不同,梯度下降法分为三种主要变体:

(1) 批量梯度下降:使用全部训练数据计算梯度,方向稳定但计算成本高;

(2) 随机梯度下降:每次随机选择一个样本计算梯度,速度快但波动较大;

(3) 小批量梯度下降:折中方案,使用小批量样本计算梯度,兼顾效率与稳定性。

在微积分中，对多元函数的参数求偏导，求得参数的偏导数以向量的形式表示梯度。如图2-6所示，θ_0 到 θ_1 的距离为 θ_0 在函数 $L(\theta)$ 上的梯度，记作 $\partial L/\partial\theta$。数学上，梯度越大，则函数的变化越大。对于函数 $L(\theta)$ 在点 θ_0 处，梯度向量 $\partial L/\partial\theta$ 的方向为函数 $L(\theta)$ 增加最快的方向。也就是说，沿着梯度向量的方向容易找到函数的最大值。反之亦然，沿着与梯度相反的方向，梯度减小最快，容易找到函数的最小值。

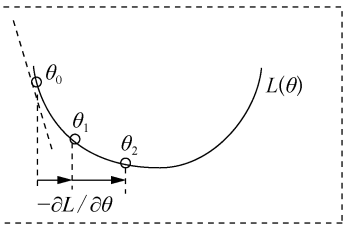


图2-6　梯度下降算法

假设函数 $L(\theta)$ 为损失函数，为了找到损失函数的最小值，需要沿着与梯度向量相反的方向 $-\partial L/\partial\theta$ 更新变量 θ，这样可以使梯度减小得最快，直至损失函数收敛至最小值。该算法称为梯度下降算法，表示：

$$\theta \leftarrow \theta - \eta \frac{\partial L}{\partial \theta} \tag{2-1}$$

式中，$\eta \in \mathbf{R}$ 为学习率，用于控制梯度下降的幅度（快慢）。在神经网络中，上式中的变量 θ 是由神经网络的参数所组成的向量 $\theta=\{w_1,w_2,\cdots,w_n,b_1,b_2,\cdots,b_n\}$。梯度下降的目标为找到参数 θ^* 使得神经网络总损失 L 最小，从而确定神经网络中的权重向量 $\boldsymbol{W}$ 和偏置 $\boldsymbol{b}$。

在图2-6中，梯度下降算法每次计算参数 θ_i 在当前位置的梯度，然后让参数 θ_i 顺着梯度的反方向前进一段距离，不断重复该过程，直到梯度接近于零时，算法认为找到了损失函数的最小值并停止计算。此时可以认为参数 θ^* 恰好到达让损失函数位于最小值的状态，也为神经网络预测值接近于真实值的状态。

梯度下降算法的特点是：越接近目标值，变化越小，梯度下降的速度越慢。梯度下降的算法有很多变种形式。三个经典变体为批量梯度下降算法（BGD）、随机梯度下降算法（SGD）和小批量随机梯度下降算法（Min-batch SGD）。

1. 批量梯度下降算法

该算法是梯度下降法最基础、最经典的形式。核心特点为在每一次参数更新迭代中，使用训练集中的全部样本来计算损失函数关于参数的梯度，所有样本都参与参数更新。

为了找到使损失函数 $J(\theta)$ 最小化的参数 θ，算法沿着整个数据集所确定的最准确的梯度方向（即全局梯度）前进一小步（步长由学习率 α 控制）。

假设有 m 个样本的训练集，m 个样本都参与调整参数 θ，得到的是一个标准的梯度。主要任务是构建最小化损失函数、基于所有 m 个样本计算得到的梯度和设计参数更新规则。更新规则可以理解为：根据整个数据集提供的“集体意见”来调整模型参数。

▲ 优点：易于得到全局最优解；总体迭代次数不多。

▲ 缺点：当样本数目很多时，训练时间过长，收敛速度变慢。

2. 随机梯度下降算法

随机梯度算法的原理与批量梯度下降算法的原理类似，区别在于随机梯度是从 m 个样本中随机抽取 n 个样本求解其梯度。

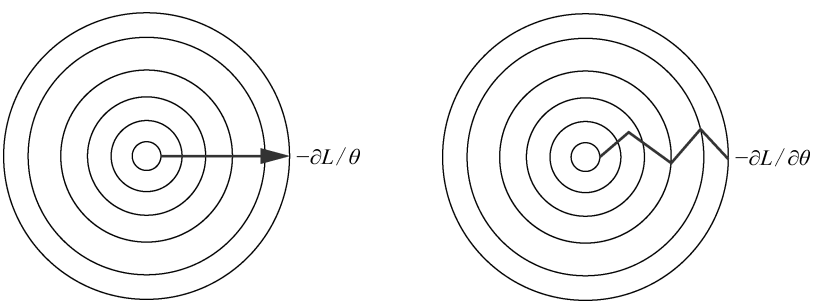
▲ 优点：训练速度快，每次迭代计算量少。

▲ 缺点:准确度下降,得到的不一定是全局最优解;总体迭代次数比较多。

3. 小批量随机梯度下降算法

小批量随机梯度下降算法是对批量梯度下降算法和随机梯度下降算法的折中方法:每次随机从 m 个样本中抽取 k 个进行迭代求梯度,每一次迭代的抽取方式都是随机的,因此部分样本会重复。这样做的好处是计算梯度时让数据与数据之间产生关联,避免数据只能收敛到局部最优解。最大的优点是集合了前两个算法的优点。

如图 2-7 所示,(a)批量梯度下降算法的下降轨迹平滑,经过多次迭代后达到全局最优解处;而(b)小批量随机梯度下降算法在下降过程中持续的小幅振荡,但每次迭代的数据量少,能够更快地找到全局最优解。因此,小批量随机梯度下降算法在实际工程中使用的频率更高。



(a) 批量梯度下降算法 BGD　　(b) 小批量随机梯度下降算法 Min-batch SGD

图 2-7　BGD 和 Min-batch SGD 梯度下降示意图

▲ **优点:训练速度快。**相比一次只用一个样本(真正的 SGD),它能利用计算硬件的并行能力,容易找到全局最优解,计算效率更高。

▲ **缺点:**需要手动设置"批量大小"这个超参数,大小会影响训练速度和稳定性。

在 ML 中,梯度下降法主要用于优化模型的损失函数,通过最小化预测误差来提升模型准确性。然而,该算法存在一个显著局限:容易陷入局部最优解而非全局最优解。算法的搜索结果高度依赖于初始点的选择,这可能使得再找到的只是某个局部极小值点,而非整个函数的最小值点。因此,在实际应用中常常需要多次运行算法或采用更先进的优化策略来克服这一局限。

2.3.2　线性回归

回归算法是一种通过最小化预测值与实际结果值之间的差距,而得到输入特征之间的最佳组合方式的一类算法。线性回归是一种经典的回归分析方法,通过建立自变量(特征)与因变量(目标)之间的线性关系进行预测。该算法的核心目标是寻找最优的线性函数,使得所有观测数据点到该线性模型的预测误差平方和最小化,表示如下:

简单线性回归:

$$y = B_0 + B_1 x \tag{2-2}$$

多元线性回归:

$$y = B_0 + B_1 x_1 + B_2 x_2 + \cdots + B_n x_n \tag{2-3}$$

式中，B_0 为截距项，B_1 至 B_n 为特征权重系数，共同构成了模型的参数集合。

将在给定输入值 x 的条件下预测 y，线性回归学习算法的目的是找到系数 B_0 和 B_1 的值。

线性回归的求解主要通过两种经典方法实现：普通最小二乘法通过矩阵运算 $\boldsymbol{B} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$ 直接求得解析解，具有计算精确的优点，但计算复杂度较高；梯度下降法则通过迭代更新规则 $\boldsymbol{B} = \boldsymbol{B} - \boldsymbol{\alpha} \cdot \nabla \boldsymbol{J}(\boldsymbol{B})$ 逐步优化参数，适用于大规模数据集，但需要仔细选择学习率。这两种方法各具特色，为不同规模和数据特性的问题提供了解决方案。

线性回归作为回归算法家族的基础成员，与普通最小二乘回归、逻辑回归、逐步回归、局部加权回归、多元自适应回归样条等方法共同构成了完整的回归分析体系。多元线性回归的实用意义在于能够同时考虑多个影响因素，更符合现实问题的复杂性，既能提供更准确的预测效果，又可以通过系数分析各变量的贡献度。其独特优势体现在模型简单且计算效率高，结果具有很好的可解释性，既为理解复杂模型奠定基础，又适合作为ML的入门算法。

在实际应用中，线性回归不仅为连续值预测提供了有效的解决方案，其基本思想更为逻辑回归等广义线性模型提供了理论框架。这种承上启下的特性使其成为ML算法体系中不可或缺的基础组成部分，既保持了数学上的严谨性，又具备工程实践中的实用性，为后续更复杂模型的学习和应用奠定了坚实的理论基础。

(1) 梯度下降法求解线性回归。假设函数 $g(z) = \dfrac{1}{1+\mathrm{e}^{-z}}$，将其转换为线性回归模型，$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$，则 $g(z) = \dfrac{1}{1+\mathrm{e}^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p)}} = h_\beta(X)$，得到条件概率表示：

$$\begin{aligned} P(y=1 \mid X;\beta) &= h_\beta(X) = p \\ P(y=0 \mid X;\beta) &= 1 - h_\beta(X) = 1 - p \end{aligned} \tag{2-4}$$

对其求对数得：

$$\log\left(\frac{p}{1-p}\right) = \log(\mathrm{e}^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p \tag{2-5}$$

将两个式合并得 $P(y \mid X;\beta) = h_\beta(X)^y \times (1 - h_\beta(X))^{1-y}$，似然函数表示：

$$L(\beta) = \prod_{i=1}^{n} h_\beta(x^{(i)})^{(i)} (1 - h_\beta(x^{(i)}))^{1-y(i)} \tag{2-6}$$

将似然函数进行对数处理：

$$l(\beta) = \log(L(\beta)) = \sum_{i=1}^{n} (y^{(i)} \log(h_\beta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\beta(x^{(i)}))) \tag{2-7}$$

对目标函数求导：

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{n} (y^{(i)} - h_\beta(x^{(i)}))(x^{(i)}) \tag{2-8}$$

令导函数为零,求解:

$$\sum_{i=1}^{n}(y^{(i)}-h_{\beta}(x^{(i)}))(x^{(i)})=0 \tag{2-9}$$

上面式中无法得到未知数的解,迭代构成会使用到经典的梯度下降算法。由于对似然函数求的是最大值,直接采用梯度下降方法不合适,因为梯度下降专门用于解决最小值问题。为了能够适用梯度下降方法,需要在目标函数的基础上乘以−1,即新的目标函数可以表示:

$$J(\beta)=-l(\beta)=-\sum_{i=1}^{n}(y^{(i)}\log(h_{\beta}(x^{(i)}))+(1-y^{(i)})\log(1-h_{\beta}(x^{(i)}))) \tag{2-10}$$

梯度下降法得到的更新过程:

$$\beta_j:=\beta_j-\alpha\frac{\partial J(\beta)}{\partial\beta_j}\quad(j=1,2,\cdots,p) \tag{2-11}$$

式中,α 为学习率。

α 过小时,迭代次数过多、收敛慢;α 过大时,可能导致陷入局部最小的状态,错过最优解。

(2) 最小二乘法求解线性回归。假设一组数据包含 n 个样本,对于第 i 个样本,有一个特征 x_i 和一个对应的真实值 y_i,多元线性回归是要求解线性方程:

$$y=\beta_0+\beta_1x+\beta_2x_2+\cdots+\beta_px_p+\varepsilon \tag{2-12}$$

矩阵表示:

$$\boldsymbol{y}=\boldsymbol{X\beta}+\boldsymbol{\varepsilon} \tag{2-13}$$

代价函数为:

$$J(\beta)=\sum\varepsilon^2=\sum(y-X\beta)^2 \tag{2-14}$$

最小二乘的目标为找到最佳的 β,使得这个损失函数 J 最小。对参数求偏导

$$\frac{\partial J(\beta)}{\partial\beta}=(0-2X'y-2X'X\beta)=0 \tag{2-15}$$

计算回归系数,得:

$$\beta=(X'X)^{-1}X'y \tag{2-16}$$

线性回归的优点是思想简单、实现容易、建模迅速,对小数据量和简单线性关系可快速构建有效模型,且结果可解释性强,为决策分析提供直观依据。作为诸多非线性模型的基础,其蕴含的最小化误差思想是 ML 的核心思想之一,能有效解决典型回归问题。但该算法的局限性也较为明显:难以处理非线性数据或特征间存在强相关性的复杂数据集,无法准确表达高度复杂的数据关系。

2.3.3 逻辑回归

逻辑回归是一种广义线性模型,专门用于处理分类问题。其核心逻辑是:先对特征进行

线性组合，再通过Sigmoid函数将组合结果映射到(0,1)区间，以此预测样本属于某一类别的概率。逻辑回归不仅能输出分类标签，还能提供概率估计，兼具简单性、可解释性与良好性能，是分类任务的首选基准算法之一，为理解复杂分类模型奠定基础。以下详细介绍其模型构建、损失函数与求解方法。

逻辑回归的目标是解决二分类问题($y\in\{0,1\}$)，模型构建分为两步：

第一步　线性组合：计算特征与权重的线性关系$z=w\cdot x+b$(w为权重向量，b为偏置项)；

第二步　非线性映射：将Z映射到(0,1)区间，得到类别为1的概率，表示：

$$p(y=1\mid x)=\sigma(w^{\mathrm{T}}x+b) \tag{2-17}$$

式中，x是输入特征(多个特征组成的向量)，w是权重向量，b是偏置项，$\sigma(z)=\dfrac{1}{1+\mathrm{e}^{-z}}$是Sigmoid函数。Sigmoid函数将模型的输出值($w^{\mathrm{T}}x+b$)映射到0到1之间，可以看作是属于类别1的概率。

损失函数：逻辑回归的损失函数是对数损失函数，其形式如下，

$$J(w,b)=-\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(h_\beta(x^{(i)}))+(1-y^{(i)})\log(1-h_\beta(x^{(i)}))\right] \tag{2-18}$$

式中，m是训练样本的数量，$h_\beta(x)=\sigma(w^{\mathrm{T}}x+b)$是逻辑回归的预测概率。

梯度下降法求解：与线性回归一样，逻辑回归通常也使用梯度下降法来优化损失函数，求解参数w和b。参数更新规则为：

$$\begin{aligned}w&=w-\alpha\frac{\partial J(w,b)}{\partial w}=w-\alpha\frac{1}{m}\sum_{i=1}^{n}(h_\beta(x^{(i)})-y^{(i)})x^{(i)}\\ b&=b-\alpha\frac{\partial J(w,b)}{\partial b}=b-\alpha\frac{1}{m}\sum_{i=1}^{n}(h_\beta(x^{(i)})-y^{(i)})\end{aligned} \tag{2-19}$$

逻辑回归通过不断更新w和b，直到损失函数收敛。逻辑回归通过使用Sigmoid函数将线性回归的输出转换为概率值，用于解决二分类问题，训练过程通过最小化对数损失函数来优化模型参数，梯度下降法是常用的优化方法，用来更新模型参数w和b，Python中的scikit-learn库提供了简单易用的接口来实现逻辑回归，并且能够轻松地进行模型训练、评估和可视化。

逻辑回归的核心优势在于：

(1) 可解释性极佳：通过权重系数可直接判断各特征对分类结果的贡献程度(正相关/负相关、贡献大小)，在医疗诊断、金融风控等需决策依据的领域极具价值；

(2) 输出概率估计：不仅能给出分类标签，还能提供样本属于某一类别的概率，支持风险量化评估；

(3) 计算高效：模型简单、训练速度快，不易过拟合，适用于线性可分问题和高维稀疏数据(如文本分类)。

其局限性主要包括：

(1) 线性决策边界：默认只能处理线性可分问题，需手动引入特征交叉或多项式变换才

能捕捉非线性关系;

(2) 对多重共线性敏感:特征间存在强相关性时,会影响参数估计的稳定性;

(3) 依赖特征工程:样本量不足或特征工程不充分时,模型性能会显著下降。

在实践中,逻辑回归常作为基准模型,其线性建模思想也为神经网络的学习奠定了基础。Python 的 scikit-learn 库提供了简洁的逻辑回归实现接口,支持快速完成模型训练、评估与可视化。

2.3.4 K 最近邻与 K -均值算法

1. K 最近邻(KNN)

属于“惰性学习”算法,其核心特点是不预先生成固定模型,而是将模型构建与未知样本预测同步进行。核心思想是:通过计算未知样本与已知标签样本的相似度(距离),选取最相似的 K 个样本(近邻),基于这 K 个样本的标签预测未知样本的类别(分类任务)或取值(回归任务)。

相似度度量以距离为核心:距离越小,相似度越高;距离越大,相似度越低。常见距离度量方法包括欧氏距离、曼哈顿距离、余弦相似度等。预测规则为:分类任务采用“投票法”,选取 K 个近邻中出现频率最高的类别;回归任务采用“平均法”,取 K 个近邻的均值作为预测值。

KNN 的预测步骤可概括为:

第一步　确定近邻个数 K 值。

第二步　基于选定的距离度量,计算未知样本与所有已知样本的距离。

第三步　筛选距离最小的 K 个已知样本,形成近邻集合。

第四步　分类任务:对近邻集合的类别进行投票,频率最高的类别为预测结果;回归任务:计算近邻集合的均值,作为预测结果。

若 K 值过小就会导致模型过拟合;反之又可能使模型进入欠拟合的状态。为了获得最佳 K 值,可以考虑两种解决方案,一种是设置 K 近邻样本的投票权重,若已知样本距离未知样本比较远,则对应的权重就设置得低一些,否则权重就高一些,通常可以将权重设置为距离的倒数;另一种是采用多重交叉验证法,该方法是目前比较流行的方案,其核心为将 K 取不同的值,然后在每种值下执行 m 重的交叉验证,最后选出平均误差最小的 K 值。当然,还可以将两种方法的优点相结合,选出理想的 K 值。

(1) 加权投票:为近邻分配权重,距离越近权重越大(通常为距离的倒数),降低远邻对预测结果的干扰。

(2) 交叉验证:尝试不同 K 值,通过多重交叉验证计算各 K 值对应的平均误差,选取误差最小的 K 值。实际应用中常结合两种方案,提升模型稳定性。

常见距离度量公式如下。已知两点 $A(x_1,x_2,\cdots,x_n)$ 和 $B(y_1,y_2,\cdots,y_n)$,A 和 B 之间的距离度量计算如下:

(1) 欧氏距离是两点 A 和 B 之间的直线距离表示:

$$d_{AB}=\sqrt{(y_1-x_1)^2+(y_2-x_2)^2+\cdots+(y_n-x_n)^2} \tag{2-20}$$

(2) 曼哈顿距离是两点 A 和 B 在轴上的相对距离综合可表示：

$$d_{AB}=|y_1-x_1|+|y_2-x_2|+\cdots+|y_n-x_n| \tag{2-21}$$

(3) 余弦相似度是计算两点所构成向量夹角的余弦值，夹角越小，则余弦值越接近于1，进而能够说明两点之间越相似，A 和 B 之间的余弦相似度可以用向量表示：

$$Similarity=\cos\theta=\frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\|\ \|\vec{B}\|} \tag{2-22}$$

图2-8展示了KNN的分类示例：蓝色正方形与绿色三角形为已知类别样本，黄色圆形为待分类样本。当 $K=3$ 时，近邻包含2个绿色三角形和1个蓝色正方形，投票后预测为绿色三角形；当 $K=5$ 时，近邻包含2个绿色三角形和3个蓝色正方形，投票后预测为蓝色正方形。

在图2-8中，$K=3$，则离黄色点最近的有2个绿色的三角形和1个蓝色的正方形，这三个点进行投票，于是黄色的待分类点就属于绿色的三角形。而若 $K=5$，则离黄色点最近的有2个绿色的三角形和3个蓝色的正方形，这五个点进行投票，于是黄色的待分类点就属于蓝色的正方形。

特殊情况：当 $K=1$ 时，KNN退化为“最近邻算法”，直接将距离最近样本的类别赋予未知样本。该方法对噪声极其敏感，易过拟合。图2-9展示 K 值对分类结果的影响：1-近邻将查询点分类为正例(+)，而5-近邻将其分类为反例(−)，其中，图左为一系列的正反训练样例和一个要分类的查询实例 x_q。1-近邻算法把 x_q 分类为正例，然而5-近邻算法把 x_q 分类为反例；图右对于一个典型的训练样例集合1-近邻算法导致的决策面。围绕每个训练样例的凸多边形表示最靠近这个点的实例空间(即这个空间中的实例会被1-近邻算法赋予该训练样例所属的分类)。

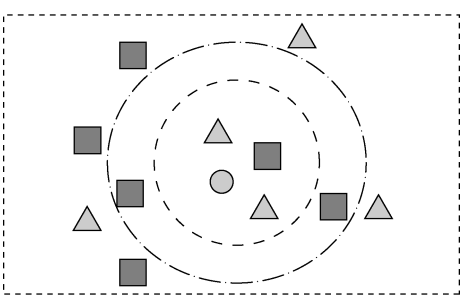


图2-8　KNN分类示例

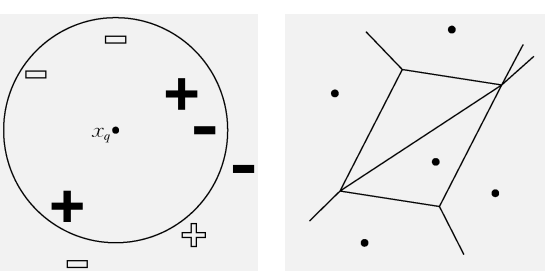


图2-9　K-最近邻算法决策边界示意图

对前面的 k-近邻算法作简单的修改后，用于逼近连续值的目标函数，再计算 k 个最接近样例的平均值，而不是计算其中的最普遍的值。为了逼近这个实值目标函数 $f:R_n\rightarrow R$，只要把算法中的公式替换为：

$$f(x_q)\leftarrow\frac{\sum_{i=1}^{k}f(x_i)}{k} \tag{2-23}$$

KNN的主要不足：

(1) 样本不平衡敏感：当某类样本数量远大于其他类别时，K 个近邻中该类样本可能占

比过高，导致预测偏差；

（2）计算复杂度高：对每个未知样本，需计算与所有已知样本的距离，样本量较大时效率低下；

（3）对高维数据不友好：高维数据中距离度量的区分度降低（维度灾难），模型性能下降。

2. K-均值算法（K-means）

K-means 算法是一种典型的基于划分的聚类算法，该算法具有运算速度快、执行过程简单的优点，在很多大数据处理领域得到了广泛的应用。利用相似性度量方法来衡量数据集中所有数据之间的关系，将关系比较密切的数据划分到一个集合中。之所以称为 K-means，是因为该算法可以将数据划分为指定的 K 个簇，并且簇的中心点由各簇样本均值计算所得。该聚类算法的思路非常通俗易懂，为不断地计算各样本点与簇中心之间的距离，直到收敛为止，其具体的步骤如下：

第一步　初始化质心：从数据集中随机选取 K 个样本作为初始簇中心。

第二步　分配样本：计算每个剩余样本与各簇质心的距离，将样本标记为距离最近的簇类别。

第三步　更新质心：重新计算每个簇内所有样本的均值，将均值作为新的簇质心。

第四步　迭代优化：重复步骤二、三，直至满足收敛条件。

第五步　收敛判断：常见条件包括“质心位置变化小于预设阈值”“达到最大迭代次数”“簇分配结果不再变化”。重复执行分配和更新步骤，直到满足以下任一条件：

（1）质心位置变化小于预设阈值。

（2）达到最大迭代次数。

（3）簇分配结果不再发生变化。

K-means：该方法的思想为所有簇内样本的离差平方和之和达到最小，所以目标函数为：

$$J(c_1,c_2,\cdots,c_k)=\sum_{j=1}^{k}\sum_{i=1}^{n_j}(x_i-c_j)^2 \tag{2-24}$$

可表明当质心为簇内样本均值时，目标函数取得最小值，对目标函数求偏导数：

$$\frac{\partial J}{\partial c_j}=\sum_{j=1}^{k}\sum_{i=1}^{n_j}\frac{(x_i-c_j)^2}{\partial c_j}=\sum_{i=1}^{n_j}-2(x_i-c_j) \tag{2-25}$$

令偏导数为零，得：

$$c_j=\frac{\sum_{i=1}^{n_j}x_i}{n_j}=\mu_j \tag{2-26}$$

K-means 的核心是确定 K 值。K 值需预先指定，但现实问题中往往无法直接知晓最优簇数。常用的 K 值选择方法包括肘部法则、轮廓系数法等：肘部法则通过绘制目标函数 J 随 K 值的变化曲线，选取曲线“肘部”（J 下降速率骤减的点）对应的 K 值；轮廓系数法通过计算样本的轮廓系数（衡量样本与本簇的相似度及与其他簇的差异度），选取平均轮廓系数

最大的 K 值。

算法优势与适用场景：K-means 的核心优势是概念直观、实现简单，对大规模数据集具有较高的计算效率（时间复杂度为 $O(nkt)$，n 为样本量、k 为簇数、t 为迭代次数）。其适用于客户分群、图像分割、文档聚类、异常检测等需要快速实现数据分组的场景，是实践中应用最常用的聚类算法之一。

局限性：对初始质心敏感，不同初始质心可能导致不同聚类结果（易收敛至局部最优）；需预先指定 K 值，难以适应数据内在簇结构未知的场景；对噪声和异常值敏感（异常值会显著影响质心计算）；仅适用于凸形簇结构，对非凸、不规则簇结构的聚类效果较差。这种方法假设已知 K 值，但目前如何选择 K 值比较难。

2.3.5　决策树

决策树算法是一种逼近离散函数的监督学习方法，核心用于分类任务，也可扩展用于回归任务。其本质是通过归纳训练数据生成可读的决策规则与树形结构，再利用该结构对新数据进行分类。决策树的核心价值在于可解释性强，生成的规则直观易懂，无需专业背景即可理解。构建精度高、规模小的决策树是算法的核心目标，通常分为“决策树生成”与“决策树剪枝”两个步骤。

1. 决策树学习的核心目标与策略

决策树学习的目标是基于训练数据构建一个能对实例进行准确分类的模型。其本质是从训练数据中归纳出一组分类规则，通过递归选择最优特征并对数据进行分割，再形成树形结构的决策模型。由于可能存在多个能正确分类训练数据的决策树，理想模型应该在保证训练准确性的同时具备良好的泛化能力。

学习策略与优化方法：决策树学习采用正则化的极大似然函数作为损失函数，以损失函数最小化为优化目标。由于寻找全局最优决策树是 NP 完全问题，实践中采用启发式方法获得次优解。学习过程包含以下核心环节：

特征选择：从所有特征中筛选出具有足够分类能力的特征，剔除无关特征（如姓名等不具有普遍分类意义的属性）。

决策树生成：递归地选择最优分割特征，基于局部最优原则构建树结构。

决策树剪枝：通过去除过于细分的节点来简化模型结构，提升泛化能力。

剪枝策略与特征选择：剪枝处理作为关键的后处理步骤，通过将过度细分的叶节点回退到父节点，有效控制了模型复杂度。这一过程体现了从局部最优到全局最优的优化思想，决策树生成阶段关注局部最优分割，而剪枝阶段则从全局角度优化模型性能。

特征选择在决策树构建初期尤为重要，通过保留具有强分类能力的特征，排除干扰项，为构建简洁有效的决策树奠定基础。这种预处理不仅提高了模型性能，也增强了结果的可解释性。

2. 决策树的构建流程

决策树的构建分为两个核心阶段：

第一阶段：决策树生成。基于带分类标签的训练数据集，以自顶向下递归方式生成决策

树。训练数据集需具备一定代表性与综合性,能够反映数据的内在规律。生成过程中,每个内部节点对应一个特征测试,根据特征取值划分数据为若干子集合,每个子集合对应一条分支;叶节点存放分类标签,代表该分支的决策结果。

第二阶段:决策树剪枝。对生成的初始决策树进行校验与修正,核心是用测试数据集验证初始规则,剪除影响预测准确性的冗余分支。初始生成的“最大树”通常存在过拟合问题——不仅拟合了训练数据的核心规律,还拟合了噪声,剪枝通过去除这些冗余分支,实现从局部最优到全局最优的优化。

构建决策树的流程如图 2-10,如图 2-11 所示为一棵简单的决策树。

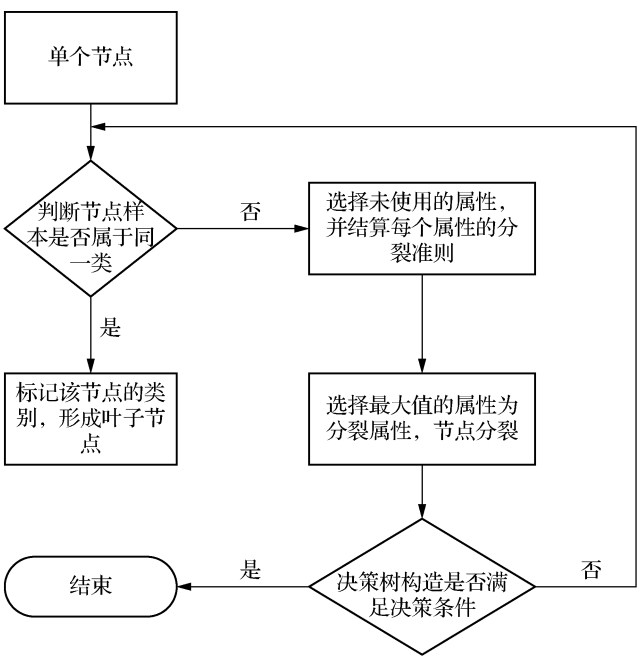


图 2-10 决策树构造流程图

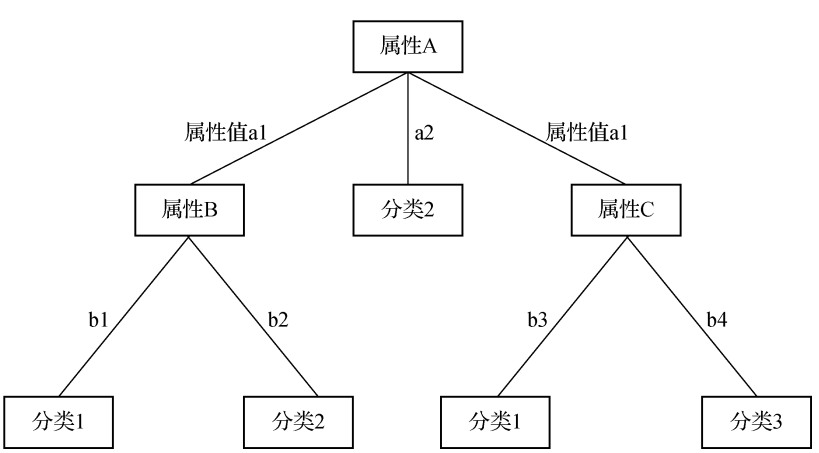


图 2-11 简单的决策树

决策树算法属于监督学习，包括两个阶段：第一阶段利用带有分类标号的训练数据集构建决策树分类模型，第二阶段使用验证数据集来检验模型得到分类结果。决策树算法构建的分类模型是树形结构，决策树的每一个分枝都代表数据的一个属性与该属性取值类型的映射关系，一棵决策树从根节点开始生长，每一个内部节点对应于数据的一个测试属性，选择合适的属性值将数据划分为若干子集合，对应的分枝路径对应一个子集合，每个叶子节点存放一个分类标号。利用具有关联的类标号的训练数据集，以自顶向下递归的方式生成决策树分类模型，叶子节点内包含的类别是这条数据的分类。对一个决策问题必须选择终结的评价时间点。也就是说，全部策略应有同时间点被评价，且全部收支值应是同时间点上。

3. 决策树的绘制规则与决策步骤

为确保决策树逻辑严谨、表达清晰，绘图时应遵循以下三条核心规则：

(1) 时间基准统一：明确决策评价的时点，所有策略选项及其对应的收益/成本均需折算至同一时间点进行比较。

(2) 路径单向展开：从决策点或结局点引出的各条路径(支路)应按时间顺序向右展开。除起始节点外，每个节点仅能由一条支路进入，确保路径序列不交叉、不合并。

(3) 完备性与互斥性：从任一节点(尤其是“机会点”)引出的所有可能分支，必须彼此互斥，并共同覆盖该情况下所有可能的后续状态，不能有遗漏或重叠。

运用决策树进行分析，通常遵循以下6个步骤：

第一步　建树：从左至右绘制树状图，依次画出决策点、方案枝、机会点、概率枝和结局点，系统展示所有可能路径。为清晰起见，可对节点按顺序编号。

第二步　标注参数信息：将每条路径对应的收益(损失)数值、自然状态及其发生概率等关键信息，准确标注在树图相应的分支旁。需特别注意状态概率估计的准确性。

第三步　计算期望值(回滚计算)：从决策树的末端叶节点(最右侧)开始，从右向左进行反向计算。对于机会节点，计算其所有后续分支的期望收益(各分支收益乘以对应概率之和)；对于决策节点，比较各方案分支的期望值。将计算结果标注在相应节点上方。

第四步　选择最优方案：依据期望值准则(如期望收益最大或期望损失最小)进行决策。将最优方案对应的期望值标注在初始决策点上方，此即该决策问题在考虑时间价值后的整体净效果(再期望值)。

第五步　剪枝与确定路径：不加限制地生长会产生一棵与训练数据拟合过度的“最大树”，它可能学习了噪声而非一般规律。**剪枝**旨在简化模型、防止过拟合，提升泛化能力。

预剪枝：在树生长过程中，提前停止分支(如设定最大深度、最小样本数阈值)。

后剪枝：首先生成一棵最大树，然后自底向上考察非叶节点，若将其替换为叶节点能提升整体模型在验证集上的性能，则进行剪枝。常用方法包括“成本复杂度剪枝”。

第六步　确定再模型与预测：通过剪枝得到一系列复杂度不同的子树后，通常使用验证集或交叉验证选择性能最佳的一棵作为再模型。对于新样本的预测，则从根节点开始，根据其特征值沿树向下遍历，直至到达某个叶节点，并以该叶节点的类别(或均值)作为预测结果。

若是多阶段或多级决策，则需要重复第二、三、四步工作。如图2-12所示：

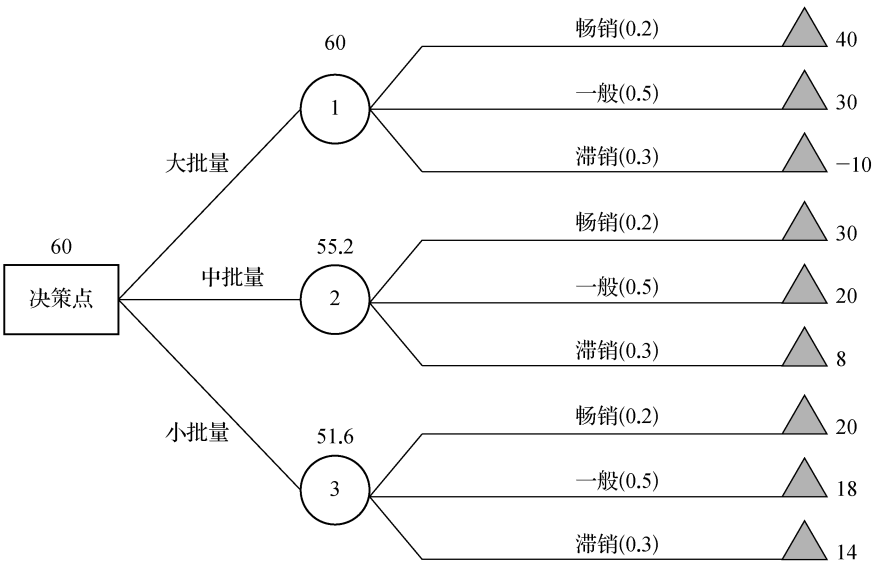


图 2-12　决策树多级决策

4. **经典决策树算法**:ID3

ID3 算法(迭代二叉树 3 代)是基于奥卡姆剃刀原理的决策树算法,其核心是追求"以最简洁的规则解决最多的问题"。该算法采用自顶向下的贪婪搜索策略,遍历可能的决策空间,其关键在于**使用信息增益作为属性选择的度量标准**,即在每个节点选择能够带来最大信息增益的特征进行分裂,从而不断提升数据划分的纯度。从信息论角度看,系统的期望信息(熵)越小,其纯度越高;通过特征分裂减少的不确定性越多,所获得的信息增益就越大。决策树构建的核心是度量数据的不确定性与特征对不确定性的减少能力,这依赖于以下三个基本概念:

信息熵(香农熵):信息熵表示随机变量不确定性的度量,也就是信息熵越大,变量的不确定性就越大。设 X 是个有限值的离散随机变量,其概率分布为:$P(X=x_i)=p_i, i=1,2,\cdots,n$,则随机变量 X 为:

$$H(X)=-\sum_{i=1}^{n}\log_2 p_i(\text{若 } p_i=0,\text{定义 } 0\log 0=0) \tag{2-27}$$

条件熵:条件 $H(Y|X)$ 表示在已知随机变量 X 条件下随机变量 Y 的不确定性。随机变量 X 给定的条件下随机变量 Y 的条件为:

$$H(Y\mid X)=\sum_{i=1}^{n}p_i H(Y\mid X=x_i), p_i=P(X=x_i) \tag{2-28}$$

信息增益:特征 A 对训练数据集 D 的信息增益 $g(D,A)$,定义为集合 D 的经验 $H(D)$ 与特征 A 给定条件下 D 的经验条件 $H(D|A)$ 之差,即:

$$g(D,A)=H(D)-H(D\mid A) \tag{2-29}$$

信息增益大的特征具有更强的分类能力。给定训练数据集 D 和特征 A，经验熵 $H(D)$ 表示对数据集 D 进行分类的不确定性，经验条件熵 $H(D|A)$ 表示在特征 A 给定的条件下对数据集 D 进行分类的不确定性，$g(D,A)=H(D)-H(D\mid A)$ 表示由于特征 A 而使得对数据 D 的分类的不确定性减少的程度。

基于上述原理，ID3算法的基本流程为：从根节点开始，计算所有特征的信息增益，选择增益最大的特征作为划分属性；然后根据该特征的每个取值创建分支，并对各分支子集递归地重复此过程，直至满足停止条件（如子集中样本全属同一类，或无特征可用）。由此生成的树结构可用于对未知数据进行分类预测。

ID3的优点：

模型直观：生成的决策树易于理解和解释，规则可读性强。

无需大量参数设置：算法流程简单，除停止阈值外超参数少。

适合处理离散特征：天然适用于分类问题中的离散属性。

ID3的局限与缺点：

倾向于选择取值多的特征：信息增益准则对取值数目较多的特征有偏好，可能产生泛化能力弱的树。

无法处理连续特征：ID3本身不能直接处理数值型连续属性。

对缺失值敏感：未提供处理训练数据缺失值的机制。

容易过拟合：若不加以剪枝，可能会生成过于复杂、对训练数据过拟合的树。

未考虑剪枝：ID3本身没有内置剪枝步骤，易受噪声影响。

ID3是决策树算法发展史上的重要里程碑，它首次系统地将信息论应用于ML中的特征选择。C4.5算法（引入信息增益比解决取值偏移问题）、CART算法（支持连续型特征与剪枝）均是对ID3的改进。例如，C4.5算法通过引入**信息增益率**克服了对多值特征的偏好，并增加了对连续特征的处理、缺失值处理以及剪枝功能，成为ID3全面且实用的增强版本。因此，学习ID3不仅有助于理解决策树的基本构建原理，也为理解更先进的树模型奠定了基础。

2.3.6　降维算法

在AI与ML领域，高维数据是一把“双刃剑”——既蕴含丰富信息，也带来计算复杂度高、过拟合风险大、数据稀疏等问题。降维算法的核心价值在于将高维数据映射到低维空间，在减少维度的同时最大程度保留关键信息。与聚类算法类似，降维算法旨在发现训练数据的固有结构，但更侧重用更少的低维信息总结原始数据的核心内容。降维算法广泛应用于数据可视化、降低计算成本等场景，常作为数据预处理步骤，将处理后的数据输入其他ML算法训练。主成分分析（PCA）与线性判别分析（LDA）作为两种经典的降维算法，在AI领域应用广泛。

1. PCA

PCA是最常用、最经典的线性降维方法之一。它通过正交变换，将可能存在相关性的原始变量转换为一组线性不相关的**主成分**。这些主成分按方差从大到小排列，前者尽可能保留原始数据的信息。PCA是一种无监督的方法，**以方差衡量信息**，其优点是实现简单、计算

高效,但主成分的含义往往不如原始特征清晰。实现 PCA 有两种方法:

(1) 基于特征值分解协方差矩阵的 PCA 实现

输入:数据集,需降至 k 维。

第一步　去中心化处理,即对每一位特征的所有样本值减去该特征的平均值;

第二步　计算去中心化后数据的协方差矩阵;

第三步　通过特征值分解或奇异值分解(SVD)方法,求解协方差矩阵的特征值与对应的特征向量;

第四步　将特征值按从大到小的顺序排序,选取前 k 个最大的特征值,并将其对应的 k 个特征向量分别作为行向量,构建特征向量矩阵 $\boldsymbol{P}$;

第五步　将原始数据映射到 k 个特征向量构建的新空间中,即通过矩阵运算 $\boldsymbol{Y}=\boldsymbol{PX}$ 得到降维后的数据集 $\boldsymbol{Y}$。

(2) 基于 SVD 分解协方差矩阵的 PCA 实现

输入:数据集,需降至 k 维。

第一步　去中心化处理,即对每一位特征的所有样本值减去该特征的平均值;

第二步　计算去中心化后数据的协方差矩阵;

第三步　通过奇异值分解(SVD)求解协方差矩阵的特征值与对应的特征向量;

第四步　将特征值按从大到小的顺序排序,选取前 k 个最大的特征值,并将其对应的 k 个特征向量分别作为列向量,构建特征向量矩阵;

第五步　将原始数据映射到 k 个特征向量构建的新空间中,得到降维后的数据集。

作为经典的无监督降维方法,PCA 的核心优势在于通过正交变换,将存在相关性的原始特征转换为线性无关的主成分,在有效保留数据全局结构和主要方差信息的同时,实现数据压缩与噪声过滤,且计算过程具有解析解,效率较高。然而,PCA 也存在显著局限性:其一,作为线性降维方法,难以捕捉数据中复杂的非线性结构;其二,降维后的主成分失去了原始特征的实际业务含义,可解释性较差;其三,其以方差衡量信息重要性的机制,可能导致部分具有类别判别价值的低方差特征被忽略。

2. LDA

LDA 是一种经典的有监督 ML 算法,主要用于分类任务中的降维和特征提取。其核心思想与主成分分析(PCA)截然不同:PCA 是一种无监督方法,旨在保留数据的最大方差;而 LDA 则利用类别标签信息,寻找能够最有效区分不同类别的最优投影方向。

LDA 的中心思想可以概括为:在低维空间中进行投影时,要同时实现类内方差最小化和类间方差最大化。

类内方差最小化:确保投影后,同一类别的数据点尽可能紧凑地聚集在一起。

类间方差最大化:确保投影后,不同类别的数据的中心(均值)尽可能远离。

这个目标是找到一个新坐标轴,使得数据投影后,不同类别之间的“分离度”达到最高,从而极大地方便后续的分类器进行分类。

算法原理:LDA 数学目标是最大化一个判别标准——Fisher 判别准则,即类间散度与类内散度的比值;

目标函数：寻找一个投影方向 w，使得以下函数最大化 $J(w)=\frac{w^{\mathrm{T}}S_b w}{w^{\mathrm{T}}S_w w}$，其中 S_w 为类内散度矩阵，S_b 为类间散度矩阵。

LDA算法的实现步骤

输入：数据集 $D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)\}$，其中 x_i 任意样本为 n 维向量，$y_i\in\{C_1,C_2,\cdots,C_k\}$，降维到的维度 d。

第一步　计算类内散度矩阵 S_W、类间散度矩阵 S_b；

第二步　求解矩阵 $S_W^{-1}S_b$ 的特征值与对应的特征向量；

第三步　将特征值按从大到小的顺序排序，选取前 d 个最大的特征值及其对应的 d 个特征向量 $(w_1,w_2,\cdots,w_d)$ 构建投影矩阵 W；

第四步　对样本集中的每一个样本特征 x_i，通过矩阵运算 $x_i'=W^{\mathrm{T}}x_i$ 转换为新的样本；

第五步　整合所有转换后的样本，得到降维后的输出样本集 X'。

作为有监督降维方法，LDA的核心优势在于能够直接利用类别标签信息，通过最大化类间差异、最小化类内差异的双重目标，寻找最具判别力的投影方向。这一特性使得LDA在人脸识别、基因分类等分类任务中的降维效果通常优于无监督方法，且能有效提升后续分类器的性能。然而，LDA的局限性也较为明显：其性能依赖于三个强假设——数据线性可分，各类别服从高斯分布，不同类别具有相同协方差矩阵，当数据存在复杂非线性关系或违背上述假定时，降维效果会显著下降；同时，其降维维度受类别数量限制，对于C类问题，最多只能降至C－1维。

LDA算法和PCA算法异同之处如下。

相同点：

(1) 核心功能一致：均能实现数据降维，简化模型计算复杂度。

(2) 核心思想相通：降维过程均基于矩阵特征分解的数学思想。

(3) 数据假设相同：均默认输入数据符合高斯分布。

不同点：

(1) 监督属性不同：LDA是有监督降维方法，需利用类别标签信息；PCA是无监督降维方法，无需依赖任何标签信息。

(2) 降维维度限制不同：LDA的降维维度存在上限，对于C类问题最多可降至C－1维；PCA无此限制，可根据方差贡献度灵活选择降维维度。

(3) 应用场景不同：LDA兼具降维和分类功能，更适用于分类导向的任务；PCA仅用于降维，适用于数据压缩、可视化等无监督场景。

(4) 投影方向选择标准不同：LDA以“提升分类性能”为目标，选择类间分离度最大的投影方向；PCA以“保留数据最大方差”为目标，选择样本投影后方差最大的方向。

2.3.7　支持向量机(SVM)

SVM是一种经典的监督式ML算法，既可用于离散因变量的分类问题，也可处理连续因变量的回归预测任务。与逻辑回归、决策树、朴素贝叶斯、K近邻(KNN)等单一分类算法相比，SVM通常具备更高的预测精度，核心优势在于其能够将低维空间中线性不可分的样本映射至高维空间，使其在高维空间内实现线性可分。该算法的核心思想是构建一个由若

干支持向量确定的“超平面”,以此实现对不同类别样本的有效划分。无论样本属于线性可分、近似线性可分,还是完全非线性可分场景,SVM 均能通过该超平面实现较高精度的分类;对于非线性可分情形,可借助核函数技术将样本映射至核空间,进而完成线性划分。

1. **核心思想与模型定义**

SVM 本质上是一个二分类模型,其基本型定义为特征空间中间隔最大的线性分类器。这一“间隔最大化”策略使其区别于其他线性分类器(如感知机),并可通过该技巧灵活处理非线性问题。从优化角度看,SVM 的学习过程可形式化为一个凸二次规划问题,等价于最小化带正则化的合约损失函数。

2. **基本类型**

根据数据可分性,SVM 主要分为三类:

- **线性可分 SVM**:适用于训练数据完全线性可分的情形。
- **线性 SVM(软间隔)**:适用于数据近似线性可分,通过引入松弛变量容忍少量误分类,提升模型鲁棒性。
- **非线性 SVM**:适用于数据线性不可分的情形,通过核函数将数据映射到高维特征空间,在高维空间中实现线性划分。

3. **从线性分类器到 SVM**

线性分类器是最基础的分类模型,其决策函数为 $g(x)=w^{\mathrm{T}}x+b$,通过符号函数 $\mathrm{sgn}(g(x))$ 输出类别标签。SVM 在此基础上,通过最大化分类间隔来优化决策超平面。样本点 (x_i,y_i) 到超平面 $w^{\mathrm{T}}x+b=0$ 的函数间隔定义为:

$$\tilde{\gamma}_i=y_i(w^{\mathrm{T}}x+b) \tag{2-30}$$

所有样本中最小函数间隔的绝对值,体现了分类的确信度。但函数间隔会随 w 和 b 的缩放而改变,因此需对其规范化,得到几何间隔:

$$\gamma_i=\frac{y_i(w^{\mathrm{T}}x+b)}{\|w\|} \tag{2-31}$$

SVM 的目标即最大化最小几何间隔。

4. **线性可分 SVM 与最大间隔划分**

线性分类器(一定意义上,也可以叫作感知机)是最简单也很有效的分类器形式。在一个线性分类器中,可以看到 SVM 形成的思路,并接触很多 SVM 的核心概念。用一个二维空间里仅有两类样本的分类问题来举个小例子,如图 2-13 所示:

C_1 和 C_2 是要区分的两个类别,在二维平面中它们的样本如图 2-13 所示。中间的直线为一个

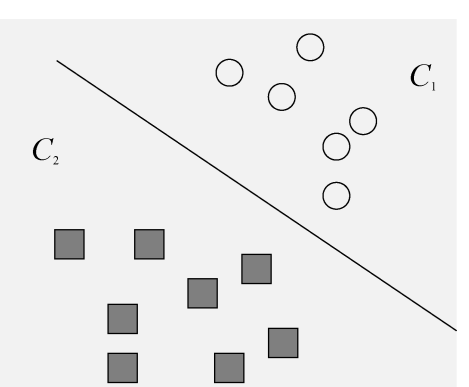


图 2-13 线性分类器

分类函数，它可以将两类样本完全分开。一般的，若一个线性函数能够将样本完全正确地分开，就称这些数据是线性可分的，否则称为非线性可分的。线性函数若在一维空间里，为一个点；在二维空间里为一条直线，三维空间里为一个平面，若不关注空间的维数，这种线性函数还有一个统一的名称——超平面。实际上，一个线性函数是一个实值函数（即函数的值是连续的实数），而分类问题需要离散的输出值，例如用1表示某个样本属于类别 C_1，而用0表示不属于（不属于 C_1 也就意味着属于 C_2），这时候只需要简单地在实值函数的基础上附加一个阈值即可，通过分类函数执行时得到的值大于还是小于这个阈值来确定类别归属。例如，一个线性函数 $g(x)=wx+b$，可以取阈值为0，根据 $g(x_i)$ 判别任意一个样本 x_i 的类别：

若 $g(x_i)>0$，则判别 x_i 为类别 C_1；若 $g(x_i)<0$，则判别 x_i 为类别 C_2。$g(x)$ 等价于符号函数 sgn()，即 $f(x)=\text{sgn}[g(x)]$，即判别函数。

在进行分类时，可以让计算机这样来看待人们提供给它的训练样本，每一个样本由一个向量（为那些数据特征所组成的向量）和一个标记（标示出该样本属于哪个类别）组成，表示：

$$D_i=(x_i,y_i) \tag{2-32}$$

式中，x_i 为数据向量（维数很高），y_i 为分类标记，在二元的线性分类中，这个表示分类的标记只有两个值，1和−1（用来表示属于还是不属于这个类）。有了这种表示法，就可以定义一个样本点到某个超平面的间隔，表示：

$$\delta_i=y_i(\omega^{\mathrm{T}}x_i+b) \tag{2-33}$$

若某个样本属于该类别的话，则 $\omega^{\mathrm{T}}x+b>0$ 而 y_i 也大于0；若不属于该类别的话，则 $\omega^{\mathrm{T}}x+b<0$，而 y_i 也小于0，这意味着 $y_i(\omega^{\mathrm{T}}x+b)$ 总是大于0，而且它的值就等于 $|\omega^{\mathrm{T}}x+b|$，也为 $|g(x_i)|$。当 $y_i=1$ 时，$\omega^{\mathrm{T}}x+b$ 应该是个大正数，反之是个大负数。因此，函数间隔为人们认为特征是正例还是反例的确信度。针对某一个样本的函数间隔为 $|\omega^{\mathrm{T}}x+b|$，定义全局样本上的函数间隔：$\min|\omega^{\mathrm{T}}x_i+b|$，其中 $i=1,2,\cdots,m$。即在训练样本上分类正例和负例确信度最小那个函数间隔。图2-14更加直观地展示出了几何间隔的现实含义：

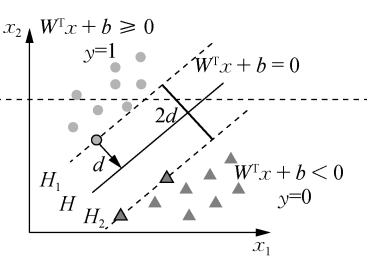


图2-14　几何间隔

H 是分类面，而 H_1 和 H_2 是平行于 H，且离 H 最近的两类样本的直线，H_1 与 H，H_2 与 H 之间的距离为几何间隔。

最大间隔的优化模型：现在人们已知如何去求数据点到超平面的距离，在超平面确定的情况下，人们就能够找出所有支持向量，然后计算出间隔 margin。每一个超平面都对应一个 margin，人们的目标为找出所有 margin 中最大的那个值对应的超平面。因此，用数学语言描述为确定 ω、b 使得 margin 最大。这是一个优化问题，其目标函数可写成：

$$\arg\max_{\omega,b}\left(\min(y(\omega^{\mathrm{T}}x+b))\frac{1}{\|W\|}\right) \tag{2-34}$$

式中，y 表示数据点的标签，且为−1或1。用 $y(\omega^{\mathrm{T}}x+b)$ 计算距离，这时就体现出−1和1

的好处。若数据点在平面的正方向(即+1类),则 $y(\omega^{\mathrm{T}}x+b)$ 是一个正数,而当数据点在平面的负方向时(即−1类),$y(\omega^{\mathrm{T}}x+b)$ 依然是一个正数,这样就能够保证始终大于零。注意到当 ω 和 b 等比例放大时,d 的结果是不会改变的。可以令所有支持向量的 u 为非零,而其他点的 u 为0,可通过调节 ω 和 b 求到。因此,上面的问题简化为:

$$\begin{aligned}&\min_{w,b}\frac{1}{2}\|W\|^2\\&\text{s.t.}\quad y(\omega^{\mathrm{T}}x+b)-1\geqslant 0,i=1,2,\cdots,n\end{aligned}\tag{2-35}$$

对于线性可分的情况,SVM优化问题表述为:

$$\begin{aligned}&\min_{w,b}\frac{1}{2}\|\omega\|^2\\&\text{s.t.}\quad y_i(w\cdot x_i+b)\geqslant 1,i=1,2,\cdots,n\end{aligned}\tag{2-36}$$

式中,ω 为超平面的法向量,b 为偏置项,x_i 为训练样本,y_i 为对应的标签(+1或−1)。

通过引入拉格朗日乘子 α_i(即对偶变量),上述原始问题可以转换为对偶问题,以便通过求解对偶问题获得最优解:

$$\max_{\alpha}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i\cdot x_j\tag{2-37}$$

式中,约束条件为:

$$\sum_{i=1}^{n}\alpha_i y_i=0,\alpha_i\geqslant 0,i=1,2,\cdots,n\tag{2-38}$$

求解后,最优超平面的法向量可表示 $w^*=\sum_{i=1}^{n}a_i^* y_i x_i$,其中 $a_i^*>0$ 对应的样本即为支持向量。分类决策函数为:

$$f(x)=\operatorname{sgn}\Big(\sum_{i=1}^{n}a_i^* y_i x_i^{\mathrm{T}}x+b^*\Big)\tag{2-39}$$

5. 近似线性可分SVM

对于数据“基本可以用一条线划分,但存在少数‘不听话’的异常点或噪声”情况,标准线性可分SVM(硬间隔)将无法找到一个可行的超平面。应使用近似线性可分SVM(软间隔SVM)。软间隔SVM的核心思想是引入妥协机制:它不再要求所有样本都必须严格满足间隔约束,而是允许少数样本以一定的“代价”被误分类或落入间隔带内。这种妥协通过引入松弛变量来实现,旨在模型复杂性与分类精度之间取得最佳平衡,从而获得更好的泛化能力。

对于给定训练集 $\{(x_i,y_i)\}$,软间隔SVM通过引入松弛变量 $\xi_i\geqslant 0$ 放宽约束。优化问题表述为:

$$\begin{aligned}&\min_{w,b,\xi}\left[\frac{1}{2}\|w\|^2+C\sum_{i=1}^{n}\xi_i\right]\\&\text{s.t.}\quad y_i(w\cdot x_i+b)\geqslant 1-\xi_i,\xi_i\geqslant 0,i=1,2,\cdots,n\end{aligned}\tag{2-40}$$

约束条件 $y_i(w \cdot x_i + b) \geqslant 1 - \xi$ 含义：

当 $\xi_i = 0$ 时，样本被完全正确分类且位于间隔外（或恰在边界上）；

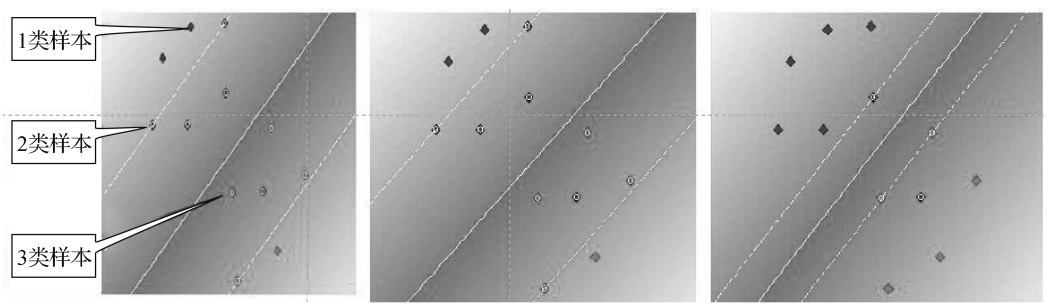
当 $0 < \xi_i \leqslant 1$ 时，样本被正确分类，但落入了间隔带内部；

当 $\xi_i > 1$ 时，样本将被误分类。

松弛变量 ξ_i 量化第 i 个样本偏离理想分类边界的程度，如图 2-15(a)所示。

目标函数中的 $C > 0$ 是一个关键的惩罚参数或正则化参数，它控制着模型对误分类的容忍程度。

C 值很大意味着对误分类的惩罚极重，模型将倾向于尽可能严格地分类所有样本，迫使大多数 ξ_i 趋近于 0，其表现趋近于硬间隔 SVM，可能导致过拟合。不同的权衡因子得到的不同的分类面如图 2-15(b)(c)所示。



(a) 不同松弛变量　(b) C=10　(c) C=100

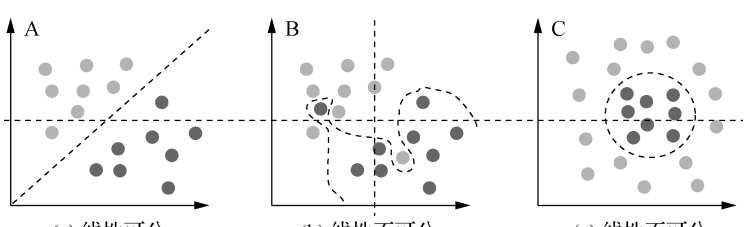
图 2-15　不同松弛变量和不同惩罚参数

C 值很小意味着对误分类的惩罚较轻，模型会为了获得更大的分类间隔而容忍更多的样本落入间隔内或被误分类，可能导致欠拟合。因此，C 实质上是“最大化间隔”与“最小化训练误差”两个目标之间的权衡系数。在实际应用中，C 通常通过交叉验证来确定。

近似线性可分 SVM 通过一个简单而巧妙的数学扩充（引入松弛变量和惩罚参数），将线性可分 SVM 的刚性框架扩展为能适应现实数据噪声的柔性模型。其成功的关键在于认识到：追求绝对完美的分类有时会损害模型的整体泛化能力，适度的妥协反而能得到更稳健的决策边界。

6. 非线性可分 SVM

当数据分布完全无法用一条直线（或超平面）划分时，无论是硬间隔还是软间隔的线性 SVM 都将失效，如图 2-16(b)(c)所示。



(a) 线性可分　(b) 线性不可分　(c) 线性不可分

图 2-16　线性可分与线性不可分

这类“线性不可分”问题,例如环形分布、异或问题,是现实世界中更常见的挑战。通过核函数将数据映射到高维空间,在新空间里建立线性边界。从理论角度看,只要选用合适的核函数(如多项式核、高斯径向基核等),即使原始数据分布复杂,也有望实现有效划分。其原理是,通过非线性映射将原始数据扩展到高维空间,从而在高维空间中实现样本的线性可分。主要包括两个关键步骤:将原始空间中的样本点映射到高维特征空间;再在该高维空间中寻找一个能够有效区分不同类别样本的线性“超平面”。表 2-1 清晰地展示两者的核心区别。

表 2-1 线性 SVM 和非线性 SVM 的核心区别

对比维度	近似线性可分 SVM(线性 SVM)	非线性可分 SVM(非线性 SVM)
数据特征	数据近似线性可分,即主体分布是线性的,但存在少数异常点或噪声	数据完全线性不可分,其内在分布是复杂非线性(如环形、交叉分布)
核心思想	软间隔:为了获得更好的泛化能力,允许一些样本被误分类,在“间隔最大化”和“误分类惩罚”之间取得平衡	核技巧:通过一个非线性映射,将数据从原始低维空间变换到一个高维特征空间,从而在这个新空间里实现线性可分
关键技术	松弛变量和惩罚参数 C 用于处理噪声和近似可分,在间隔宽度与分类损失间进行权衡	核函数,隐式定义高维空间的内积,避免显式映射的复杂计算,避免复杂的高维映射计算,直接在内积中完成(核技巧)
决策边界	在原始特征空间中,它仍然是一条直线或一个超平面	在原始特征空间中,决策边界是一条复杂的曲线或曲面(即高维空间中超平面投影)
目标函数	在原始空间中优化	通过对偶问题在核空间中优化
直观理解	画一条线把两类基本分开,但允许个别点跑到线的另一边,并用参数 C 控制对这种行为的容忍度	将数据“抛”到一个更高的维度,然后在这个新视角下用一个“平面”干净利落地切开

7. 核化 SVM

近似线性可分是“线性方法的小修小补”,而非线性可分则是“彻头彻尾的空间变换”。在实际应用中,人们通常先尝试线性 SVM,若效果不佳,再通过引入不同的核函数来使用非线性 SVM。在非线性 SVM 中,核函数的选择对模型性能具有决定性影响,尤其是在处理线性不可分数据时。核函数的核心作用是将输入空间中的线性不可分数据映射到某个高维特征空间,使其在该空间中转化为线性可分。这一过程本质上是一种非线性变换,它将原始优化问题转化为高维空间中的一个约束最优化问题。假设原始空间中的样本点为 x,将样本通过某种转换 $\phi(x)$ 映射到高维空间中,则非线性 SVM 模型的目标函数可表示:

$$\begin{aligned}&\min_{\alpha}\left[\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\phi(x_i)\phi(x_j))-\sum_{i=1}^{n}\alpha_i\right]\\&\text{s. t.}\quad \sum_{i=1}^{n}\alpha_i y_i=0,0\leqslant\alpha_i\leqslant C\end{aligned}\tag{2-41}$$

由于从输入空间到特征空间的这种映射会使得维度发生爆炸式的增长,因此,上述约束问题中内积 $\phi(x_i)\phi(x_j)$ 运算量会非常大以至于无法承受,可通过构造一个核函数,将输入

空间内线性不可分的数据映射到一个高维特征空间内使得数据在特征空间内可分，$k(x_i \cdot x_j)=\phi(x_i)\cdot\phi(x_j)$，避免在特征空间内的运算，只需要在输入空间内就可以进行特征空间的内积运算。核 SVM 就可以把求解约束最优化问题变为：

$$\begin{aligned}&\min_{w,b}\left[\frac{1}{2}\|w\|^2+C\sum_{i=1}^{n}\xi_i\right]\\&y_i[w^{\mathrm{T}}\phi(x_i)+b]\geqslant 1-\xi_i,\xi_i\geqslant 0,i=1,2,\cdots,n\end{aligned}\tag{2-42}$$

式中，w 是超平面的法向量，b 是偏置项，ξ_i 是松弛变量，C 是惩罚参数。

C 用于调节“最大化间隔”与“最小化误分类惩罚”的权重，C 值越大，对误分类样本的惩罚越重，模型越倾向于严格分类；C 值越小，对误分类的容忍度越高，模型更注重间隔最大化。

计算最优的拉格朗日乘子，得到线性“超平面”的值：

$$\hat{w}=\sum_{i=1}^{n}{}_j\hat{\alpha}_i y_i\phi(x_i),\hat{b}=y_i-\sum_{i=1}^{n}\hat{\alpha}_i y_i K(x_i\cdot x_j)\tag{2-43}$$

若数据集中存在噪点，则在求超平时就会出现很大问题。从图 2-17 中可以看出其中一个深色点偏差太大，把它作为支持向量所求出来的 margin 就会比不算入它时要小得多。但若这个深色点落在了浅色点之间就找不出超平面，则无法对样本进行分类。

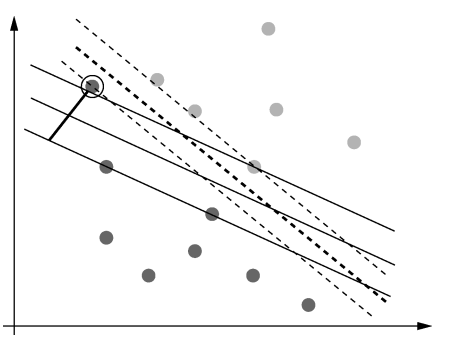


图 2-17　噪点

引入一个松弛变量 ξ，允许一些数据可以处于分隔面错误的一侧，则约束条件变为：

$$y_i(\omega^{\mathrm{T}}x_i+b)\geqslant 1-\xi_i,i=1,2,\cdots,n\tag{2-44}$$

式中，ξ_i 的含义为允许第 i 个数据点允许偏离的间隔。若让 ξ 任意大的话，则任意的超平面都符合条件。所以在原有目标的基础之上，让 ξ 的总量也尽可能地小。所以新的目标函数为：

$$\begin{aligned}&\min\frac{1}{2}\|\omega\|^2+C\sum_{i=1}^{n}\xi_i\\&\text{s.t.}\quad\sum_{i=1}^{n}\alpha_i\cdot label^i=0\text{ 和 }C\geqslant\alpha\geqslant 0\end{aligned}\tag{2-45}$$

式中，C 是用于控制“最大化间隔”和“保证大部分的点的函数间隔都小于 1”这两个目标的权重。将上述模型完整表示：

$$\begin{aligned}&\min\frac{1}{2}\|\omega\|^2+C\sum_{i=1}^{n}\xi_i\\&\text{s.t.},\xi_i\geqslant 0,i=1,2,\cdots,n\\&y_i(\omega^{\mathrm{T}}x_i+b)\geqslant 1-\xi_i,i=1,2,\cdots,n\end{aligned}\tag{2-46}$$

新的拉格朗日函数变为:

$$L(\omega,b,\xi(\omega^{\mathrm{T}}))=\frac{1}{2}\|\omega\|^{2}+C\sum_{i=1}^{n}\xi_i-\sum_{i=1}^{n}\alpha_i(y_i(\omega^{\mathrm{T}}x_i+b)-1+\xi_i)-\sum_{i=1}^{n}\tau_i\xi_i \tag{2-47}$$

将拉格朗日函数转化为其对偶函数,首先对 L 分别求 ω,b,ξ 的偏导,并令其为 0,得:

$$\begin{aligned}&\frac{\partial L}{\partial\omega}=0\Rightarrow\omega=\sum_{i=1}^{n}\alpha_i y_i x_i\\&\frac{\partial L}{\partial b}=0\Rightarrow\omega=\sum_{i=1}^{n}\alpha_i y_i=0,\text{其中},i=1,2,\cdots,n\\&\frac{\partial L}{\partial\xi_i}=0\Rightarrow C-\alpha_i-\tau_i=0\end{aligned} \tag{2-48}$$

代入原式化简之后得到和原来一样的目标函数:

$$\max L(a)=\sum_{i=1}^{n}a_i-\frac{1}{2}\sum_{i,j=1}^{n}a_i a_j y_i y_j x_i^{\mathrm{T}}x_j \tag{2-49}$$

经过添加松弛变量的方法,现在能够解决数据更加混乱的问题。通过修改参数 C,可以得到不同的结果,而 C 的大小到底取多少比较合适,需要根据实际问题进行调节。

8. SVM 核函数的选择

SVM 核函数的选择对于其性能的表现有重要的作用,尤其是针对那些线性不可分的数据,核函数的选择在 SVM 算法中就显得至关重要。

(1) 线性核函数

$$k(x\cdot x_i)=x\cdot x_i \tag{2-50}$$

线性核主要用于线性可分的情况,可以看到特征空间到输入空间的维度是一样的,其参数少速度快,对于线性可分数据,其分类效果很理想,因此,人们通常首先尝试用线性核函数来做分类。

(2) 多项式核函数

$$k(x\cdot x_i)=((x\cdot x_i)+1)^{d} \tag{2-51}$$

多项式核函数可以实现将低维的输入空间映射到高维的特征空间,但是多项式核函数的参数多,当多项式的阶数比较高时,核矩阵的元素值将趋于无穷大或者无穷小,计算复杂度会大到无法计算。

(3) 高斯(RBF)核函数

$$k(x\cdot x_i)=\exp\left(-\frac{\|x\cdot x_i\|^{2}}{\delta^{2}}\right) \tag{2-52}$$

高斯径向基函数是一种局部性强的核函数,其可以将一个样本映射到一个更高维的空间内,该核函数是应用最广的一个,无论大样本还是小样本都有比较好的性能,而且其相对

于多项式核函数参数要少，因此大多数情况下在不知道用什么核函数时，优先使用高斯核函数。

(4) Sigmoid 核函数

$$k(x \cdot x_i) = \tanh(\eta < x, x_i > + \theta) \tag{2-53}$$

采用 Sigmoid 核函数，支持向量机实现的为一种多层神经网络。

因此，在选用核函数时，若人们对数据有一定的先验知识，就利用先验来选择符合数据分布的核函数；若不知道的话，通常使用交叉验证的方法，来试用不同的核函数，误差最小的即为效果最好的核函数，或者也可以将多个核函数结合起来，形成混合核函数。总结有三类情况：

① 若特征的数量大到和样本数量差不多，则选用 LR 或者线性核的 SVM；

② 若特征的数量小，样本的数量正常，则选用 SVM+高斯核函数；

③ 若特征的数量小，而样本的数量很大，则需要手工添加一些特征从而变成第一种情况。

9. SVM 的优缺点

SVM 的核心优势主要体现在以下方面：

(1) 鲁棒性强：SVM 再形成的分类器仅依赖少量支持向量，因此，增加或删除非支持向量的样本点，不会改变分类器的决策边界与分类效果；同时可有效避免“维度灾难”，模型计算复杂度不会随数据维度的提升而显著增加。

(2) 泛化能力优异：通过间隔最大化策略，可在一定程度上避免模型过拟合，同时因求解的是凸二次规划问题，能有效规避局部最优解，确保得到全局最优解。

与此同时，SVM 的局限性也较为明显：

(1) 不适用于大样本场景：处理大规模数据集时，会消耗大量计算资源与时间，训练效率较低。

(2) 对缺失样本敏感：建模前需对样本数据进行严格清洗，缺失值会严重影响模型性能。

(3) 核函数选择敏感：虽可通过核函数解决非线性可分问题，但核函数的类型与参数设置对模型效果起决定性作用，需大量实验验证。

(4) 模型可解释性差：属于“黑盒模型”，相较于回归分析、决策树等算法，无法直观解释分类结果的生成逻辑。

2.3.8 贝叶斯模型

朴素贝叶斯模型同样是流行的十大挖掘算法之一，属于有监督的学习算法，是专门用于解决分类问题的模型，该模型的数学理论简单。该分类器的实现思想非常简单，即通过已知类别的训练数据集，计算样本的先验概率，然后利用贝叶斯概率公式测算未知类别样本属于某个类别的后验概率，再以最大后验概率所对应的类别作为样本的预测值。

对未知类别的样本进行预测时具有以下优点：首先，算法在运算过程中简单而高效；其次，算法拥有古典概率的理论支撑，分类效率稳定；最后，算法对缺失数据和异常数据不太敏

感。有以下缺点:模型的判断结果依赖于先验概率,所以分类结果存在一定的错误率;对输入的自变量 X 要求具有相同的特征(如变量均为数值型或离散型或 0—1 型);模型的前提假设在实际应用中很难满足等。

全概率公式:

$$P(A)=\sum_{i=1}^{n}P(AB_i)=\sum_{i=1}^{n}P(B_i)P(A\mid B_i) \tag{2-54}$$

贝叶斯公式:

$$P(C_i\mid X)=\operatorname{argmax}\frac{P(C_i)P(X\mid C_i)}{\sum_{i=1}^{k}P(C_i)P(X\mid C_i)} \tag{2-55}$$

主要任务是计算 $P(X\mid C_i)$。假设数据集中一共包含 P 个自变量,则 X 可表示 $(x_1,x_2,\cdots,x_p)$,则条件概率表示:

$$P(X\mid C_i)=P(x_1,x_2,\cdots,x_p\mid C_i)=P(x_1\mid C_i)P(x_2\mid C_i)\cdots P(x_p\mid C_i)$$

自变量 X 的数据类型可以是连续的数值型,也可以是离散的字符型,或者是仅含有 0—1 两种值的二元类型,通常会根据不同的数据类型选择不同的贝叶斯分类器。

(1) 高斯贝叶斯分类器。若数据集中的自变量均为连续的数值型,则在计算 $P(X\mid C_i)$ 时会假设自变量 X 服从高斯正态分布,所以自变量 X 的条件概率可表示:

$$P(x_j\mid C_i)=\frac{1}{\sqrt{2\pi}\sigma_{ij}}\exp\left(-\frac{(x_i-\mu_{ij})^2}{2\sigma_{ij}^2}\right) \tag{2-56}$$

高斯贝叶斯分类器的核心是假设数值型变量服从正态分布,若实际数据近似服从正态分布,分类结果会更加准确。Sklearn 模块提供了实现该分类器的计算功能,它为 naive_bayes 子模块中的 GaussianNB 类。语法为:

```
GaussianNB(priors=None)
```

式中,priors 用于指定因变量各类别的先验概率,默认以数据集中的类别频率作为先验概率,所以在这个分类器中可以不传递任何参数值。

(2) 多项式贝叶斯分类器。若数据集中的自变量 X 均为离散型变量时,就无法使用高斯贝叶斯分类器,而应该选择多项式贝叶斯分类器。计算概率 $P(X\mid C_i)$ 可以表示:

$$P(x_j=x_{jk}\mid C_i)=\frac{N_{ik}+\alpha}{N_i+n\alpha} \tag{2-57}$$

(3) 伯努利贝叶斯分类器。当数据中的自变量 X 均为 0—1 二元值时(例如在文本挖掘中判断某个词语是否出现在句子中,出现时为 1,不出现时为 0),通常会优先选择伯努利分类器。利用该分类器计算概率值 $P(X\mid C_i)$ 时,会假设自变量 X 的条件概率满足伯努利分布,故概率值的计算公式可以表示:

$$P(x_j\mid C_i)=px_j+(1-p)(1-x_j) \tag{2-58}$$

伯努利贝叶斯分类器的计算与多项式贝叶斯分类器的计算非常相似,在文本分类问题

中，若构造的数据集是关于词语出现的次数，通常会选择多项式贝叶斯分类器进行预测；若构造的数据集是关于词语是否会出现的0—1值，则会选择伯努利贝叶斯分类器进行预测。伯努利贝叶斯分类器可以直接调用sklearn子模块naive_bayes中的BernoulliNB类。语法和参数含义如下：BernoulliNB(alpha＝1.0，binarize＝0.0，fit_prior＝True，class_prior＝None)，其中alpha用于指定平滑系数 a 的值，默认为1.0。

binarize：当自变量的值大于该值时，自变量的值将被转换为1，否则被转换为0；若该参数为None时，则默认训练数据集的自变量均为0—1值。

fit_prior：是否以数据集中各类别的比例作为先验概率。

class_prior：用于人工指定各类别的先验概率，若指定该参数，则参数fit_prior不再有效。

2.3.9 Bagging与随机森林

Bagging(自助聚合)是一种经典的集成学习技术，其核心思想非常直观：通过“少数服从多数”或“平均意见”来获得比单一模型更稳定、更准确的预测结果。随机森林(RF)是Bagging的一个扩展变体。随机森林是一种基于集成学习思想的强大ML算法，通过构建多棵决策树并整合其预测结果来完成分类或回归任务。

1. Bagging

给定包含 m 个样本的数据集，人们先随机取出一个样本放入采样集中，再把该样本放回初始数据集，使得下次采样时该样本仍有可能被选中，这样，经过 m 次随机采样操作，人们得到含 m 个样本的采样集，初始训练集中有的样本在采样集里多次出现，有的则从未出现。初始训练集中约有63.2%的样本出现在采样集中。于是，人们可以采样出 T 个含 m 个训练样本的采样集，然后基于每个采样集训练出一个基学习器，再集成，这为Bagging的基本流程。在对预测输出进行结合时，Bagging通常对分类任务采用简单投票法，对回归任务使用简单平均法。若分类预测时出现两个类收到同样票数的情形，则最简单的做法是随机选择一个，也可进一步考察学习器投票的置信度来确定再胜者。

2. 随机森林

随机森林在以决策树作为基学习器构建Bagging集成的基础上，进一步在决策树的训练过程中引入了随机属性选择。具体来说，传统决策树在选择划分属性时是在当前结点的属性集合(假定有 d 个属性)中选择一个最优属性；而在RF中，对基决策树的每个结点，先从该结点的属性集合中随机选择一个包含 k 个属性的子集，然后再从这个子集中选择一个最优属性用于划分。这里的参数 k 控制了随机性的引入程度：若令 $k＝d$，则基决策树的构建与传统决策树相同；若令 $k＝1$，则随机选择一个属性用于划分；一般情况下，推荐值 $k＝\log 2d$，RF简单、容易实现、计算开销小，而令人惊奇的是，它在很多学习任务中展现出强大的性能，被誉为“代表集成学习技术水平的方法”。RF的收敛性与Bagging相似。随机森林的起始性能往往相对较差，特别是在集成中只包含一个基学习器时，这很容易理解，因为通过引入属性扰动，随机森林中个体学习器的性能往往有所降低。然而，随着个体学习器数目增加，随机森林通常会收敛到更低的泛化误差。

该算法的核心优势在于通过两种随机性来提升模型性能：首先使用 Bootstrap 抽样从原始数据集中有放回地生成多个训练子集，为每棵树提供略有差异的训练数据；其次在每棵树的节点分裂时引入特征随机选择，仅从全部特征中随机选取部分特征进行评估，从而确保森林中每棵决策树都具有足够的多样性。归纳为：

（1）训练可以高度并行化，对于大数据时代的大样本训练速度有优势，这是最主要的优点。

（2）由于可以随机选择决策树节点划分特征，这样在样本特征维度很高时，仍然能高效地训练模型。

（3）在训练后，可以给出各个特征对于输出的重要性。

（4）由于采用了随机采样，训练出的模型的方差小，泛化能力强。

（5）相对于 Boosting 系列的 Adaboost 和 GBDT，RF 实现比较简单。

（6）对部分特征缺失不敏感。

RF 的主要缺点有：

（1）在某些噪音比较大的样本集上，RF 模型容易陷入过拟合。

（2）取值划分比较多的特征容易对 RF 的决策产生更大的影响，从而影响拟合的模型的效果。

结合策略：假设集成中包含 T 个基学习器 $h_1, h_2, \cdots, h_T$，其中 h_i 在示例 x 上的输出为 $h_i(x)$。则对 h_i 进行结合的常见策略有以下几种：平均法、投票法和学习法，平均法主要针对回归类任务，对数值型输出 $h_i(x) \in \mathbf{R}$，最常见的结合策略是平均法。

3. Stacking

Stacking 是一种集成学习算法，很多算法可以视作它的特例，也是一种很重要的结合策略。主要的原理也很简单，根据训练集训练出 T 个初级学习器，这些初级学习器更多时是互不相同的，比如可能是决策树、神经网络等，当然也可能是相同的，比如都是决策树。然后还有一个次级学习器（也叫元学习器），拿所有的初级学习器的输出作为次级学习器的输入来训练次级学习器，最后得到一个总的学习器。

算法本身不复杂，没有像 AdaBoost 那样的公式。但在训练中要注意为了防止过拟合，不能直接使用初级学习器的训练集产生次级学习器的训练集，而要使用交叉验证法，对初始的训练集进行划分，训练集用来训练初级学习器，验证集用来产生次级训练集。Stacking 是一种比较重要的思想，假如人们有一个学习器效果不是很好，想要提升它的效果，可以用它和其他的学习器集成起来，从而达到减小误差提升效果的目的。

尽管性能优异，随机森林也存在一些固有局限。模型解释性不足：相比单棵决策树，整个森林的决策逻辑变得不透明；计算资源需求较高：树数量较多时，训练和推理速度会明显下降；对特定数据类型效果有限：在处理稀疏特征或高度相关特征时提升不明显。

随机森林已在多个领域表明其价值，金融领域：包括信用评分、欺诈检测、风险控制；医疗健康：包括疾病诊断、预后预测、药物发现；商业分析：包括客户细分、流失预警、推荐系统；工业应用：包括设备故障预测、质量监控、价格预测等。

通过合理调整超参数（如树的数量、最大深度等）并利用袋外样本评估技术，实践者可以在具体项目中充分发挥随机森林的潜力，构建高效可靠的预测模型。

2.3.10　神经网络

神经网络(NN),亦称为人工 NN(ANN),是一种受生物神经系统信息处理机制启发而构建的计算模型。通过模拟神经元之间的连接与信号传递方式,从数据中自动学习复杂的非线性映射关系,现已成为 ML 和 AI 领域的核心技术。

人脑包含约 860 亿个相互连接的神经元。每个神经元通过树突接收信号,在细胞体内整合处理,若信号超过一定阈值,则通过轴突传递给下游神经元。ANN 正是对这一生物过程的极度抽象与简化。

(1) 神经元模型。ANN 可看成以人工神经元为节点,用有向弧加权连接起来的有向图。人工神经元为对生物神经元的模拟,而有向弧则是轴突—突触—树突对的模拟。有向弧的权值表示相互连接的两个人工神经元间相互作用的强弱。人工神经元是 NN 的基本计算单元,其结构如图 2-18 所示。

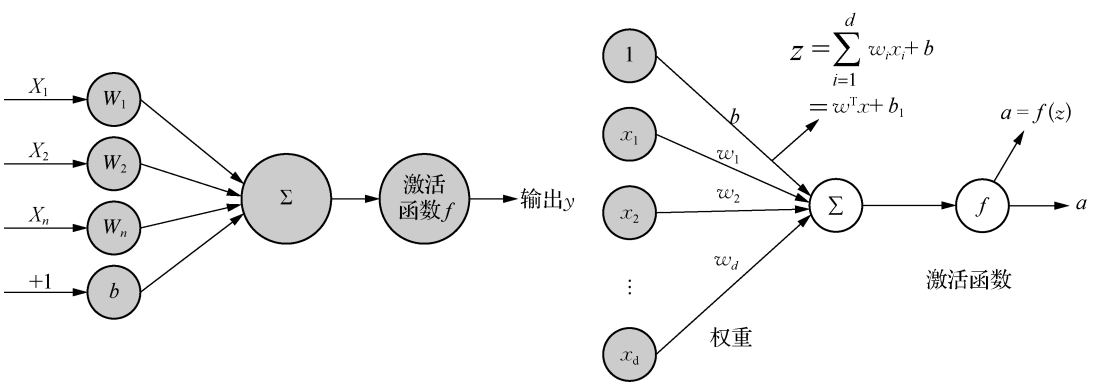


图 2-18　神经元模型

NN 从两个方面模拟大脑:NN 获取的知识是从外界环境中学习得来;内部神经元的连接强度,即突触权值,用于储存获取的知识。

NN 系统由能够处理人类大脑不同部分之间信息传递的由大量神经元连接形成的拓扑结构组成,依赖于这些庞大的神经元数目和它们之间的联系,人类的大脑能够收到输入信息的刺激,由分布式并行处理的神经元相互连接进行非线性映射处理,从而实现复杂的信息处理和推理任务。

(2) 网络架构与学习机制。典型的 ANN 采用分层前馈结构,如图 2-19 所示的三层网络,由四个基本组成部分构成:

① **输入层**:负责接收外部原始数据。接收来自数据或前一层神经元的信号,每个输入对应一个特征(如图像中的像素值、文本中的词向量)。

② **隐藏层**:承担信息抽象与特征转换的核心功能。该层每个神经元接收来自前一层所有节点的输入,每个输入连接都有一个可学习的**权重** w_i(表征该连接的重要性)和一个共同的**偏置** b(用于调节神经元激活的难易度)。神经元计算输入的加权和,然后通过一个非线性**激活函数**产生输出。激活函数(如 Sigmoid、ReLU)的引入是 NN 能够学习复杂非线性模式的关键。表示如下:

$$y=f\left(\sum_{i=0}^{n-1}w_i x_i+b\right) \tag{2-59}$$

式中,x_i 为第 i 个元素的输入,w_i 为第 i 个处理单元与本处理单元的互联权重即神经元连接权值。f 称为激活函数或作用函数,它决定节点(神经元)的输出,b 表示偏置,用于调节神经元激活的难易程度。

③ **激活函数**:通过引入 Sigmoid、ReLU、Tanh 等非线性变换激活函数使 NN 能够突破线性模型的限制,学习并表征复杂的数据模式。这些函数作为非线性阈值过滤机制,决定了神经元是否被激活以及激活的程度。计算输入的加权和 $z=\sum_{i=0}^{n-1}w_i x_i+b$,然后通过一个非线性的激活函数 $f(z)$ 产生输出。这个非线性是 NN 能够学习复杂模式的关键。

④ **输出层**:生成再的预测或分类结果。

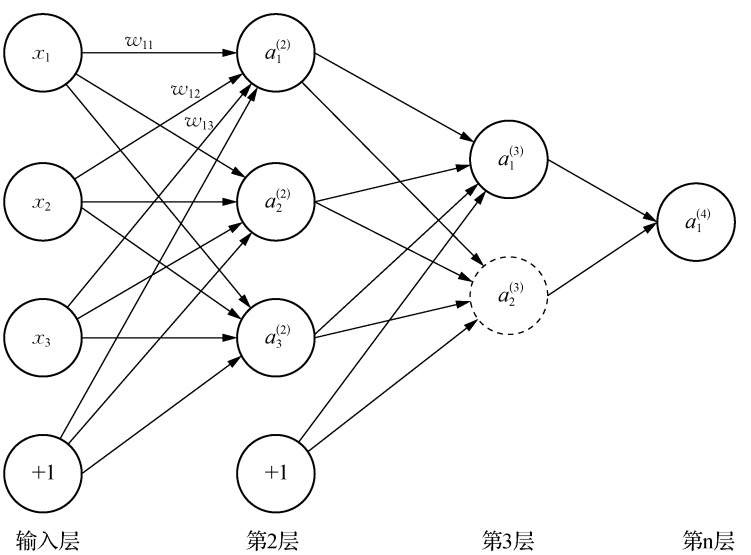


图 2-19　三层 ANN 模型

在图 2-19 中,w_{ij} 是相邻两层神经元之间的权值,它们是在训练过程中需要学习的参数,$a_1^{(2)}$ 中表示第 2 层的第 1 个神经元,分别求出 3 个神经元的输出结果,而 $h_{W,b}(x)$ 表示第 3 层第 1 个神经元 $a_1^{(3)}$ 的值。图中箭头所指的方向为前向传播的过程,即所有输入参数经过加权求和后,将结果值依次向下一层传递,直到最后输出层,层数越多,层中神经元越多,形成的权重值参数就越多。图中网络结构对应的计算如下:

$$\begin{aligned}
a_1^{(2)}&=f(w_{11}^{(1)}x_1+w_{12}^{(1)}x_2+w_{13}^{(1)}x_3+b_{11}^{(1)})\\
a_2^{(2)}&=f(w_{21}^{(1)}x_1+w_{22}^{(1)}x_2+w_{23}^{(1)}x_3+b_{12}^{(1)})\\
a_3^{(2)}&=f(w_{31}^{(1)}x_1+w_{32}^{(1)}x_2+w_{33}^{(1)}x_3+b_{13}^{(1)})\\
h_{W,b}(x)&=a_1^{(3)}=f(w_{11}^{(2)}a_1+w_{12}^{(2)}a_2+w_{13}^{(2)}a_3+b_{21}^{(2)})
\end{aligned} \tag{2-60}$$

层次化特征学习:包括浅层、中层和深层。其中,浅层学习基础、局部的特征(如边缘、角落、基本词性);中层组合浅层特征,形成更复杂的模式(如眼睛、鼻子、短语结构);深层进一

步组合，再识别出高级的、抽象的概念（如一张脸、一个句子的情感）。这种自动的、由简至繁的特征提取能力，使其无需依赖昂贵且可能不完善的人工特征工程。

端到端学习：给定输入（如原始图像）和期望输出（如“猫”），NN可以自动调整内部数百万甚至数十亿的参数（权重和偏置），以最小化预测误差。这个过程通过反向传播算法实现，它有效地将输出层的误差反向传播到网络各层，指导参数更新。

NN的力量并非来自单个神经元，而是源于其层次化结构和端到端学习能力。

这种基于神经元互连的架构，结合非线性激活和优化算法，使得ANN具备了强大的特征学习和模式识别能力，为DL的发展奠定了坚实基础。

1. BPNN（反向传播NN）

BPNN是最经典、应用最广泛的多层前馈NN之一。其名称来源于其核心训练算法——误差反向传播。BP网络由输入层、至少一个隐含层和输出层构成，相邻层间全连接，层内无连接。信息单向从输入层经隐含层传至输出层（前向传播），误差则从输出层反向传播至输入层以指导权重更新。给定一个训练样本(X, t)，其学习循环如下：

第一步 **前向传播**：计算网络的实际输出Y。

第二步 **误差计算与反向传播**：计算输出Y与目标t的误差，并将该误差沿网络反向传播，计算各层神经元对总误差的贡献（梯度）。

第三步 **权重更新**：根据梯度下降法则，沿误差减小的方向调整各层间的连接权重和偏置。

通过反复迭代上述过程，网络参数逐渐收敛，使得网络能够以最小误差学习输入到输出的复杂映射。

BP网络主要功能：

（1）函数逼近：用输入向量和相应的输出向量训练一个网络以逼近一个函数。

（2）模式识别：用一个待定的输出向量将它与输入向量联系起来。

（3）分类：把输入向量所定义的合适方式进行分类。

（4）数据压缩：减少输出向量维数以便传输或存储。

2. 其他经典NN模型

（1）Hopfield网络：Hopfield网络是一种反馈式类型的网络。Hopfield网络比BP出现得还早一些，它的学习规则是基于灌输式学习，即网络的权值不是通过训练出来的，而是按照一定规则计算出来的。Hopfield网络采用了这种学习方式，其权值一旦确定就不再改变，而网络中各神经元的状态在运行过程中不断更新，网络演变到稳定时得出的当前各神经元的状态为再的解集。

主要功能：Hopfield网络早期功能是被用作内容寻址存储器，模数转换及优化组合计算等。具有代表意义的是解决TSP问题。在经过DL的浪潮和AI的发展后，Hopfield网络在ML、联想记忆、模式识别、优化计算、VLSI和光学设备的并行实现等方面都有着广泛应用。用于联想记忆时训练完成后权重不变，网络中可变的参数只有一直在更新的神经元状态和神经元输出。由于神经元随机更新，所以称此模型为离散随机型。当网络更新时，若权重矩阵与非负对角线对称，则下面这个能量函数可以保证最小化，直到系统收敛到其稳定状态

之一。

工作机制如下：

① 权重固定与状态更新：在用于联想记忆时，其连接权重在训练完成后即保持不变。网络运行时，仅有神经元的状态与输出随时间动态更新。

② 离散随机型：由于神经元通常按随机顺序异步更新，因此，该模型常被称为离散随机型 Hopfield 网络。

③ 能量函数与收敛性：当网络权重矩阵对称且对角线元素非负时，系统定义一个能量函数。在每次状态更新中，该能量函数均保证单调下降，直至网络收敛至某个局部极小值点，即系统的稳定状态。这一特性是其能够应用于优化问题的数学基础。

Hopfield 网络作为典型的反馈型 NN，主要具备以下功能与应用：

早期应用：主要被设计用作内容寻址存储器，并可用于实现模数转换及求解组合优化问题。其最具代表性的成果之一是成功应用于旅行商问题的求解。

现代发展：随着 DL 与 AI 技术的演进，Hopfield 网络在联想记忆、模式识别、优化计算等领域持续发挥重要作用，同时在超大规模集成电路与光学设备的并行实现中也展现出应用潜力。

(2) 径向基 NN(RBF)。RBF 网络是一种结构清晰的三层前馈网络，由输入层、单个隐含层和输出层构成。其设计灵感源于神经科学中的感受野概念——大脑皮层的神经元仅对特定局部区域的刺激产生响应。网络借鉴了这种局部感知与重叠覆盖的机制，通过隐层中的径向基函数(如高斯函数)对输入空间的局部区域进行响应，从而成为一种强大的局部逼近网络。理论上，它能够以任意精度逼近任何连续函数，因此，在处理复杂的非线性分类与回归问题上具有显著优势。

得益于其精确的局部拟合能力和往往更快的训练速度，网络常被应用于对精度要求较高的多个领域，主要包括：图像处理与语音识别、时间序列预测与系统建模、雷达目标定位与医疗诊断、故障检测与模式识别。

在诸多应用中，RBF 网络最核心且最常见的用途仍是分类任务，尤其在模式识别领域表现突出。此外，它在时间序列分析与预测中也是一类非常有效的工具。

本章习题

1. ML 基本流程

简述 ML 从问题抽象到模型评估的完整流程，并以“芒果质量预测”为例，具体说明每个阶段的任务。

2. 梯度下降法对比

梯度下降法包含批量梯度下降(BGD)、随机梯度下降(SGD)和小批量梯度下降(Mini-batch GD)三种主要变体。请从**计算效率、收敛稳定性和适用场景**三个方面对比它们的优缺点。

3. 线性回归与逻辑回归

线性回归与逻辑回归均为基于线性模型的经典算法，请回答：

(1) 它们分别用于解决什么类型的问题？

(2) 逻辑回归如何借助 Sigmoid 函数实现从回归到分类的转换?
(3) 写出逻辑回归的损失函数(对数损失)表达式。
4. KNN 与 K-means 辨析
K 最近邻(KNN)与 K 均值(K-means)名称相似但本质不同。请说明:
(1) 它们分别属于哪类学习范式(监督/无监督)?
(2) 简述各自的核心思想与算法步骤。
(3) 列举两种常见的距离度量方法,并给出计算公式。
5. 决策树与信息增益
在决策树算法中,信息增益是常用的特征选择准则。请回答:
(1) 什么是信息熵?写出其数学定义式。
(2) 如何计算特征 AA 对数据集 DD 的信息增益?
(3) 信息增益为何会偏向取值较多的特征?有哪些改进方法?
6. PCA 降维原理
主成分分析(PCA)是一种经典的无监督降维方法。请简述:
(1) PCA 的核心思想是什么?
(2) 基于特征值分解的 PCA 算法主要步骤。
(3) PCA 与 LDA 在目标与应用场景上有何主要区别?
7. SVM 与核函数
支持向量机(SVM)在处理非线性问题时广泛使用核函数。请解释:
(1) SVM 的“最大间隔”原则是指什么?
(2) 什么是“核技巧”?为何它能避免显式的高维映射计算?
(3) 列出三种常见核函数并简述其特点。
8. 朴素贝叶斯分类器
朴素贝叶斯是基于贝叶斯定理的分类算法。请回答:
(1) 写出贝叶斯公式,并说明其在分类任务中如何应用。
(2) “朴素”体现在何处?该假设在实际问题中是否合理?
(3) 针对连续型特征和离散型特征,通常分别选用哪种贝叶斯分类器?
9. 集成学习与随机森林
随机森林是集成学习的典型代表。请说明:
(1) 什么是 Bagging?它与 Boosting 的主要区别是什么?
(2) 随机森林在 Bagging 的基础上引入了哪两种随机性?各自作用是什么?
(3) 随机森林如何评估特征的重要性?
10. 神经网络基础
人工神经网络是 DL 的基石。请回答:
(1) 一个典型的人工神经元如何计算输出?写出其数学表达式。
(2) 什么是激活函数?为什么神经网络必须使用非线性激活函数?
(3) 简述 BP 神经网络的前向传播与反向传播过程。

第三章　DL 基础知识

DL 通过一系列核心操作处理复杂数据，构建起高效的特征学习与模式识别体系：卷积操作精准提取数据局部特征，池化操作在降低特征维度的同时增强平移不变性，激活函数为网络注入非线性变换能力以适配复杂模式学习，全连接层则实现高层特征的整合与目标映射。在此基础上，损失函数量化预测输出与真实标签的差异程度，反向传播算法与优化器协同工作，通过梯度计算动态调整网络权重参数，推动模型持续学习有效特征表示，稳步提升预测精度与泛化能力。

3.1　引言

DL 和传统 ML 都是 ML 领域的重要分支，但它们在方法和应用上存在明显的区别与独特的优势。表现如下：

(1) 特征提取与学习不同：传统 ML 通常依赖于特征工程，这意味着专家需要人为地对数据进行提炼和清洗，选择或构造最相关的特征来训练模型。DL 模型自身能够从原始数据中自动学习和提取有用的特征。这种方法不需要手动选择特征、压缩维度或转换格式。

(2) 数据依赖性不同：传统 ML 通常需要大量的标记数据来训练模型，因为模型的性能很大程度上取决于输入的数据质量；而对于 DL 来说，尤其是当使用无监督学习方法时，可以处理大量未标记的数据。此外，深度网络的多层结构使其能够学习数据的多层次表示。

(3) 计算资源不同：传统 ML 通常需要的计算资源较少，因为它们的模型结构简单。DL 由于其复杂的网络结构和大量的参数，通常需要更多的计算资源，如 GPU 加速。

(4) 模型解释性不同：许多传统的 ML 算法(如决策树、支持向量机等)提供相对较高的模型解释性，因为它们的决策过程往往是直观的；而 DL 的模型，尤其是深层神经网络，通常被视为“黑箱”，因为它们的内部工作机制很难解释。

当前，DL 领域已形成多个成熟且功能强大的开源框架，它们在设计理念、适用场景与生态系统方面各具特色，为学术研究及工业应用提供了坚实基础。主流包括 TensorFlow、PyTorch、Keras、PaddlePaddle 与 Caffe 等。

1. TensorFlow：生态系统完备的工业级框架

TensorFlow 由 Google Brain 团队开发，是一个以其完整的数据流图计算架构和强大生态系统著称的开源 ML 框架。2.0 版本后的 TensorFlow 深度融合 Keras 高级 API，同时支持动态图(Eager Execution)与静态图两种模式，显著简化神经网络构建流程，保持对动态图与静态图两种计算模式的双重支持，兼顾研究灵活性与生产部署的高效性。

强大的生产链路：提供从模型开发、训练优化到多端部署（包括移动端、边缘端）的全流程解决方案。

跨平台硬件支持：具备完善的硬件适配能力，可无缝运行于CPU、GPU及Google专有的TPU等计算平台，并提供从模型开发、训练优化到多端部署的全流程解决方案。

生态系统：依托TensorBoard可视化工具、TFX端到端平台和活跃的开发者社区，构建了完整的ML开发生态，在图像分类、机器翻译等复杂任务中持续保持领先地位，平衡了研究灵活性与生产稳定性的特点，使其成为学术界和工业界首选的DL框架之一。

2. PyTorch：以研究灵活性与开发效率见长

PyTorch由Facebook AI Research团队开发，以其卓越的灵活性和执行效率在学术界获得极高人气，并迅速向工业界扩展。采用“Define-by-Run”机制，支持研究人员在模型构建过程中进行实时调整与调试，极大地简化了原型开发和实验迭代流程。

直观的Pythonic接口：其设计哲学强调开发效率与用户体验，API设计直观，使开发者能够专注于网络结构本身。

繁荣的生态系统：提供完整的GPU加速支持，并集成了TorchVision、TorchText等丰富的领域库，以及强大的Hugging Face社区。

发展动态：2022年，PyTorch基金会的成立及其纳入Linux基金会管理体系的重要举措，进一步巩固了其开源生态系统的可持续发展。考虑到PyTorch简洁的神经网络语法、完善的算法复现基础以及显著降低的开发成本，很多研究选择该框架作为实验平台。

3. Keras：用户友好的高阶API

Keras作为高阶神经网络API，基于TensorFlow、Theano或CNTK等后端运行。通过提供简洁的模块化接口，将层、模型、优化器等组件高度抽象，极大地降低了模型开发的复杂度。它本身不是一个底层框架，而是可以作为前端接口运行在TensorFlow、Theano或CNTK等后端之上。其设计注重用户友好性与快速实验，通过简洁的模块化接口大幅降低模型开发复杂度，非常适合初学者入门与中小规模建模任务。

适用场景：非常适合初学者入门、概念验证以及中小规模的建模任务。

4. PaddlePaddle：百度自主研发的产业级DL平台

PaddlePaddle是源自中国自主研发的产业级DL平台，提供了一套覆盖训练到部署全流程的完整工具链，已在多个工业场景中得到广泛应用。PaddlePaddle具备大规模分布式训练、高效推理引擎以及丰富的产业级预训练模型，其内置的动态图/静态图统一编程范式兼顾了开发灵活性与部署性能。

适用场景：已在工业制造、自动驾驶、智慧城市等多个工业场景中得到广泛应用，特别适合需要将模型部署到实际生产环境中的任务。

5. Caffe：经典高效的视觉库

Caffe由Berkeley AI Research团队开发的开源DL框架，以其出色的执行效率与清晰的模块化设计著称，在CNN和视觉任务领域一度是事实上的标准。在图像分类、目标检测

等视觉应用中表现优异，执行速度快，配置式网络定义清晰。为提升模型可解释性，Caffe 还集成了 TensorBoard 等可视化工具，助力开发者深入理解网络内部工作机制。

主要局限：尽管功能强大，Caffe 在应用层面存在一定局限。其配置过程较为复杂，不仅对计算硬件有较高要求，还需依赖众多第三方库的协同工作，这对初学者及非专业用户构成了相当的门槛。对计算硬件和第三方库依赖较多，入门门槛较高；灵活性不足，网络结构需预先静态定义，动态调整极为不便，限制了研究迭代速度；应用领域受限，对循环神经网络等序列建模任务支持较弱，在 NLP 等非视觉领域应用较少。

为便于直观比较 5 种框架，表 3－1 汇总了 5 种 DL 框架的核心特征。

表 3－1　DL 框架的核心特征

框架名称	核心定位	主要优势	潜在局限
TensorFlow	工业级全栈平台	生态完整、部署能力强、支持多种计算图模式	API 较为复杂、学习曲线相对陡峭
PyTorch	研究导向的灵活框架	动态图、直观易用、调试方便、社区活跃	生产环境部署工具链相对年轻
Keras	高阶 API，入门首选	接口简洁、模块化、快速实验、易于上手	底层灵活性受限于高级 API 封装
PaddlePaddle	产业级全栈平台	大规模分布式训练、端到端部署、产业级模型库	全球社区与生态影响力仍在扩张中
Caffe	高效视觉库	执行速度快、模块化、视觉任务成熟稳定	配置复杂、灵活性差、对动态网络支持弱

DL 技术已在 AI 和计算机等相关领域得到了广泛应用与突破：

- **计算机视觉**：涵盖图像分类、目标检测、图像生成（如 GAN）与语义分割等，广泛应用于自动驾驶、医疗影像分析与安防监控等领域。
- **自然语言处理**：包括文本分类、机器翻译、语音识别与对话系统。以 Transformer 为代表的模型显著提升了语言理解与生成的性能。
- **语音与音频处理**：支持语音识别、音频事件检测、音乐分类等任务，为智能助手、内容审核等场景提供核心能力。
- **生物信息与医疗健康**：应用于基因序列分析、蛋白质结构预测、医学影像辅助诊断等方向，推动精准医疗与药物研发。
- **金融科技**：用于欺诈检测、信用评估、市场预测等，提升金融业务智能化与风险控制水平。
- **游戏与机器人**：在游戏 AI、机器人环境感知与决策控制等方面发挥重要作用，促进智能体与复杂环境的交互能力。

未来 DL 技术将持续在以下方向演进：

- **模型架构创新**：Transformer、BERT 等新型网络结构不断涌现，在多模态学习、跨任务泛化等方面展现出更强潜力。
- **多模态融合**：融合视觉、语言、语音等多源信息的统一建模成为重要方向，推动跨媒体

分析与人机交互能力升级。

- **自动化ML**:通过神经架构搜索与超参数优化等技术,降低模型设计对专业知识的依赖,提升开发效率。
- **边缘计算部署**:轻量化模型与推理优化技术促进AI在终端设备中的普及,满足实时性与数据隐私需求。
- **可解释性与安全性**:增强模型决策的透明性与鲁棒性,建立可信任的AI系统,应对对抗攻击与伦理挑战。
- **算法与优化进步**:改进训练策略与优化方法,提升模型收敛速度与泛化性能,促进更大规模与更高效地学习。

3.2 卷积与图像卷积

卷积是信号处理和图像分析中一个非常核心的运算,也是DL中最重要的过程之一,也是CNN得以成功的理论基础。从本质上看,卷积是一种信息混合技术,它将两种不同来源的信息通过特定的数学规则进行融合,产生新的特征表示。

1. 数学上的卷积

在数学(特别是泛函分析)中,卷积是一种定义在两个函数上的数学运算,目的是衡量两个函数在滑动重叠过程中的积分效应。对于两个函数,一个函数是"信号",另一个是"滤波器"或"窗口"。卷积操作为让这个"窗口"在"信号"上滑动,在每一个位置,计算它们重叠部分的乘积之和。对于函数 $f(x)$ 和 $g(x)$,它们在 t 时刻的卷积定义为:

$$\begin{aligned}\mathrm{Cov}(x) &= f(x) \otimes g(x) \\ &= \int_{-\infty}^{+\infty} f(t)g(x-t)\mathrm{d}t\end{aligned} \tag{3-1}$$

将连续卷积中的积分替换为求和,就得到离散序列 $f(k)$ 和 $g(k)$ 的离散卷积定义为:

$$\mathrm{Cov}(k) = \sum_j f(j)g(k+1-j) \tag{3-2}$$

卷积过程:翻转+滑动+点积求和。这是数学卷积的精髓,即先将滤波器函数进行翻转,然后在信号函数上滑动并进行积分(或求和)。需要注意,虽然积分区间(求和)是从负无穷到正无穷,但由于函数和序列的定义域(范围)的限制,实际的有效积分(求和)范围往往有限。若超出 $f(x)$和 $f(k)$的定义域时,被积函数值(求和序列值)为零,对积分(求和)没有贡献。

想象一下,有两个装满信息的"容器":一个是输入数据(如图像),另一个是卷积核(一组可学习的参数)。卷积运算为将这两个容器中的信息倒入一个新的容器中,按照既定的规则进行"搅拌"和融合。这种融合过程不是简单的叠加,而是一种有组织的、局部相关的混合方式。在数学上,卷积是一种特殊的积分变换,但与加减乘除等基本运算一样,它具有明确的数学定义和计算规则。虽然卷积运算本身相对复杂,但它能够极大地简化更复杂的表达式,这正是它在工程和科学领域得到广泛应用的原因。两个函数 $f(x)$和 $g(x)$

的卷积定义为:

考虑一个生动例子:七品县令用打板子惩罚无赖。假设单个板子的疼痛响应会随时间衰减,稀疏打击(每天一板子),则疼痛有足够时间衰减,总痛苦程度不叠加;连续打击(每秒一板子),前次疼痛未衰减,后续打击又至,痛苦程度叠加。用卷积来描述:总痛苦程度 = $\sum$ 第 τ 次打击的疼痛 × 衰减系数($t-\tau$)。数学表达式为:

$$S(k)=\sum T_i \cdot H(k-t_i) \tag{3-3}$$

式中,$S(k)$为在时间 k 的总痛苦,T_i 为第 i 次打击的疼痛强度,$H(k-t_i)$第 i 次打击的衰减函数,t_i 为这次打击发生的时间,Σ 为对所有发生过的打击($i=1$ 到 N)进行求和。

单个板子的响应(脉冲响应):无赖挨一个板子后,疼痛并非在瞬间达到峰值然后消失,而是会持续一段时间,并随着时间逐渐衰减。这个"疼痛随时间衰减的模式",为系统的脉冲响应,在数学上表示衰减函数 $H(t-\tau)$。它描述了一个单一事件(一个板子)在系统中留下的"痕迹"是如何随时间演变的。

多个板子的总效应(卷积):当板子以很短的时间间隔接连落下时,情况就变了。第二个板子落下时,第一个板子造成的疼痛还未完全消退;第三个板子落下时,前两个的疼痛痕迹依然存在。卷积所做的,正是在某个特定时刻 t,将所有过去发生的板子(在时间 τ)所残留的疼痛痕迹,进行求和(积分)。

卷积的物理意义:描述了系统如何将一系列连续的输入(板子),通过其自身的特性(疼痛会衰减但不立即消失),转化为一个连续累积的输出(总痛苦程度)。它是历史对现在的总影响的精确量化。

2. 图像卷积

图像卷积操作,也称为核操作或算子操作,是图像处理中用于实现模糊、锐化及边缘检测等功能的常用技术。它是一种邻域操作,其核心过程是使用一个称为卷积核(或算子、滤波器)的小型矩阵,与图像中的每个像素点及其邻域像素进行加权求和,从而生成一幅新的输出图像。简单来说,卷积为让一个预设的"模式"(卷积核)滑过图像的每一个像素,并通过一种特定的计算方式,将当前像素及其周围像素的信息"混合"在一起,从而在输出图像中产生新的效果。

(1) 卷积核。卷积核是一个包含特定权重的数组,其设计遵循以下关键原则:

➢ **尺寸**:通常选用奇数尺寸,如 1×1、3×3、5×5 或 7×7。这确保了卷积核具有明确的中心点(即锚点),便于与图像像素精确对齐。

➢ 对称性:多数卷积核设计为对称结构(如 3×3、5×5),以便在空间域中平等地利用像素各方向上的邻域信息。非对称核(如 1×3 或 3×1)则用于特定方向(如水平或垂直)的处理。

图像卷积是一个系统的计算过程:

➢ 将卷积核的**锚点**对准输入图像的当前待处理像素;

➢ 将卷积核覆盖区域内的每个像素值,与核中对应位置的权重值相乘;

➢ 将所有乘积结果求和,所得数值即为输出图像中对应位置的像素新值;

➢ 按设定的**步长**滑动卷积核，重复上述过程，直至遍历完整幅图像。

通过这一过程，图像中的每个像素值都被其自身及邻域像素的加权组合所替代，从而实现对图像特征的增强、提取或平滑。在图像处理中，卷积常以滤波形式呈现。常见的高斯滤波、导向滤波、双边滤波等，本质上均为卷积操作的具体实现，表示：

$$F[i,j]=\sum_{u,v} I[i+u,j+v]\cdot w[u,v] \tag{3-4}$$

式中，I 代表输入图像，F 为输出特征图，w 是卷积核参数，都是二维数组。

(2) 图像卷积。图像卷积运算为卷积核与输入图像对应的元素相乘，然后再相加。该过程相当于将滤波器作为滑动模板，在图像每个局部区域进行逐元素相乘并求和，从而实现特征的增强或提取。卷积过程如图 3-1 所示。

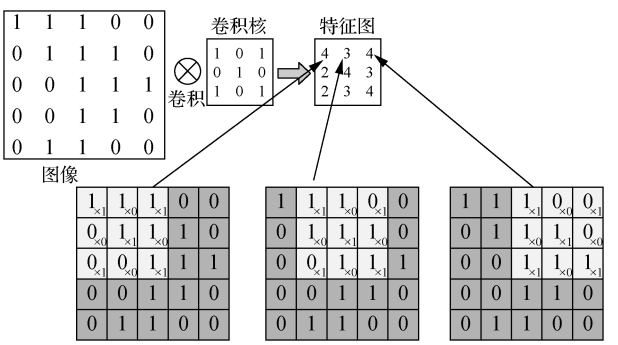


图 3-1 卷积过程

卷积核 w 也称为滤波器或卷积算子，类似一个模板。滑动核，使其中心位于输入图像 g 的(i,j)像素上；利用式(3-4)求和，得到输出图像的(i,j)像素值；继续上面操作，直到求出输出图像的所有像素值。卷积后的大小：W 为矩阵宽，H 为矩阵高，F 为卷积核宽和高，P 为 padding(需要填充的 0 的个数)，N 为卷积核的个数，S 为步长，width 为卷积后输出矩阵的宽，height 为卷积后输出矩阵的高，得

$$\text{width}=(W-F+2P)/S+1(\text{向下取整}) \tag{3-5}$$

$$\text{height}=(H-F+2P)/S+1(\text{向下取整}) \tag{3-6}$$

在 Matlab 二维卷积 conv2d()中，若最大池化 max_pool()的 padding＝SAME 时，width=W，height=H，则保证输入输出尺寸图像大小相等，当 padding='valid' 时，$P=0$，相当于不填充。

输出图像大小：(width，height，N)中 N 为卷积核的个数，在卷积操作中起到分为 N 个通道数的作用；当卷积操作中含有膨胀因子 d，且 padding 为 pad，k 为卷积核尺寸，stride 为步长时，特征图维数为：out=$(W-(F\times d-1)+2\times \text{pad})/S+1$。

不同的卷积核对图像产生截然不同的处理效果。5 个著名的卷积核：锐化(Sharpen)、模糊(Blur)、边缘增强(Edge-enhance)、边缘检测(Edge-detect)、凸印(Emboss)在图像处理中起着重要作用。表 3-2 为 5 个卷积核的性能比较。

表 3-2　5 个卷积核的性能比较

效果	卷积核	视觉描述与核心原理
锐化(Sharpen)	[0,−1,0] [−1,5,−1] [0,−1,0]	效果:使图像细节更清晰、边缘更锋利。 原理:通过增强中心像素与周围邻居的差异来实现。强大的中心权重(+5)会强化当前像素,而负的邻域权重(−1)则相当于减去一个模糊版本,从而提升了边缘的对比度。
模糊(Blur)	[1/9,1/9,1/9] [1/9,1/9,1/9] [1/9,1/9,1/9]	效果:使图像变得平滑、柔和,能有效抑制噪点。 原理:这是一个均值滤波器。它将每个像素的值替换为其自身及其周围像素的平均值。这个过程减少了像素间的剧烈变化,从而达到平滑和模糊的效果。
边缘增强 (Edge Enhance)	[0,−1,0] [−1,6,−1] [0,−1,0]	效果:在保留图像原内容的基础上,让物体的轮廓和内部线条更加突出、鲜明。 原理:可以看作是更激进的锐化。它使用比锐化核更大的中心权重(+6),从而更强烈地突出已有的边缘,但不会像边缘检测那样丢弃其他图像信息。
边缘检测 (Edge Detect)	[−1,0,1] [−2,0,2] [−1,0,1]	效果:生成一幅只显示图像中边缘和轮廓的图,非边缘区域变为黑色。 原理:此核(Sobel 算子)用于计算像素亮度的梯度(变化率)。它对垂直方向的边缘特别敏感。当经过均匀区域时,输出为 0(黑色);当经过明暗变化的边缘时,则输出高值(亮色)。
凸印(Emboss)	[−2,−1,0] [−1,1,1] [0,1,2]	效果:创造一种 3D 浮雕或压印效果,仿佛光线从左上角照射。 原理:通过非对称的权重来模拟定向光照。右下角为正权重(高光区),左上角为负权重(阴影区)。在平坦区域输出中性灰,在有纹理的区域则产生明暗变化,从而形成立体感。

(3) 卷积应用。根据不同用途,应选择不同的卷积核,进行卷积:

突出差异(锐化、边缘增强、边缘检测):通过中心正权重与周围负权重的组合来放大或提取像素间的变化。

平滑差异(模糊):通过均匀的正权重来平均化像素间的变化。

模拟光照(凸印):通过非对称且总和近于 0 的权重来创造光影效果。

CNN 的核心在于其卷积核的可学习性与层次化特征提取能力。网络通过数据驱动自动学习大量卷积核的权重,使其能够识别从底层边缘、纹理到高层语义结构的多级视觉模式,实现对图像内容的层次化表征。卷积核作为可学习参数,通过在输入特征图上滑动计算并加权求和,生成对应的特征响应图,即完成“卷积计算”。每个卷积核提取一种局部特征,多个卷积核得到的特征图共同形成图像的多维度抽象表达,这也是“利用 CNN 提取图像特征”的本质过程。

3. 一些常见的卷积方式

(1) 标准卷积。例如,对于一张 6×6 像素、三通道彩色输入图像(shape 为 6×6×3)。经过 3×3 卷积核的卷积层(假设输出通道数为 4,则卷积核 shape 为 3×3×3×4),再输出 4 个特征图,若有相同 padding,则尺寸与输入层相同(6×6),若没有,则尺寸变为 3×3。基于单卷积核的卷积过程如图 3-2(a)所示,无论是单层还是多层,一个卷积核只能检测一种特征。要检测多个特征需要多个卷积核,多个卷积核的卷积过程如图 3-2(b)和图 3-2(c)所示。

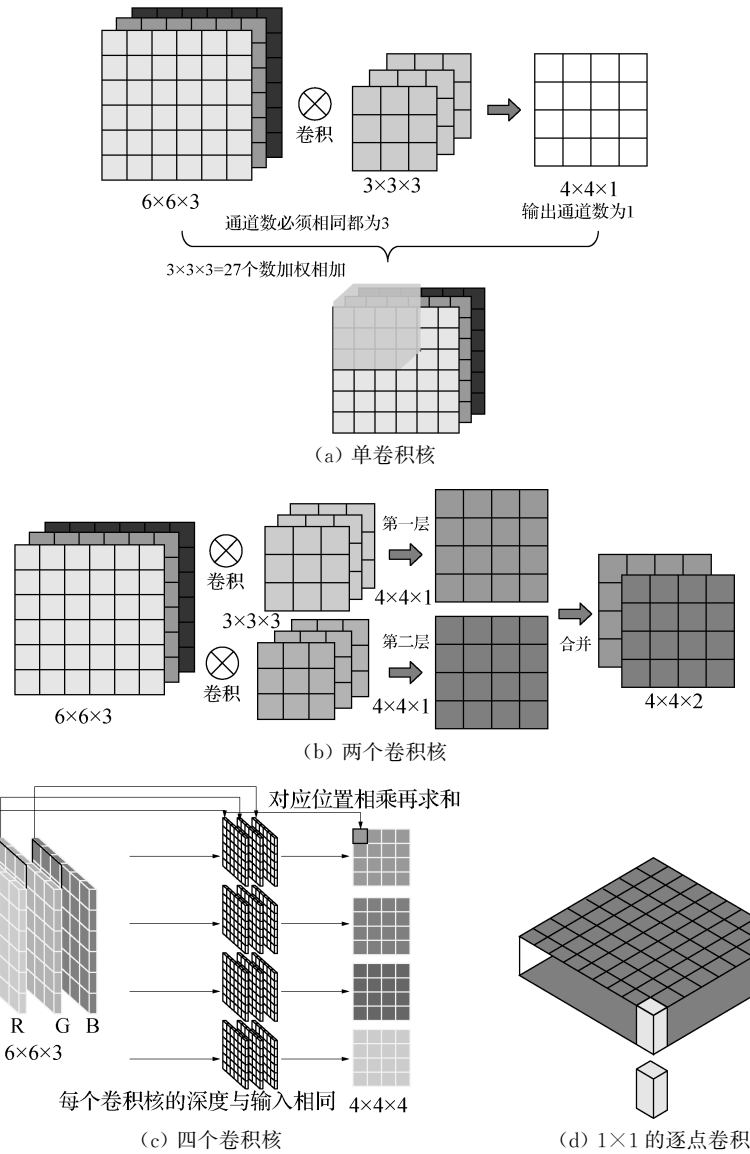


图 3-2　标准卷积过程

标准卷积的参数计算：假设有一个卷积层，其输入特征图的通道数为 16，期望输出特征图的通道数为 32，并使用 3×3 大小的卷积核。在此设置下，需要创建 32 个独立的卷积核。关键在于，每个卷积核本身都是一个三维张量，其通道维度必须与输入特征图的通道数保持一致。因此，每个卷积核的实际尺寸是 3(高)×3(宽)×16(通道)。

单个卷积核操作与参数量：一个卷积核会在输入的 16 通道特征图上进行滑动窗口计算。在每个空间位置上，它执行一次三维的卷积运算：将卷积核的 16 个通道分别与输入特征的 16 个通道对应位置相乘后求和，输出一个单通道的响应值。因此，单个卷积核的参数个数为 3×3×16＝144。

总参数量的计算：由于需要 32 个这样的卷积核来生成 32 个输出通道，因此该卷积层的总参数量为：(3×3×16)×32＝4 608。

假设卷积核大小为 $D_K \times D_K$，输入通道为 M，输出通道为 N，输出的特征图尺寸为 $D_F \times D_F$，则通过标准卷积后，参数量为 $D_K \times D_K \times M \times N$；计算量为 $D_K \times D_K \times M \times N \times D_F \times D_F$。

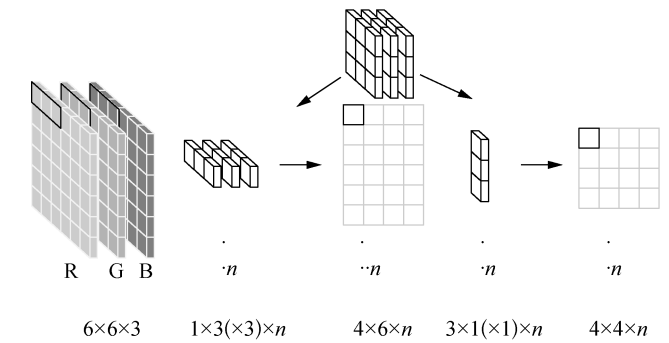
标准卷积的一个关键特性：每个输出通道都是由一个专门的三维卷积核生成的，而该卷积核的通道数由输入决定。这种全通道连接的方式虽然强大，但也导致了较高的计算和存储开销，这也是催生深度可分离卷积等轻量化技术的重要原因。

(2) 1×1 卷积。 1×1 卷积是使用大小为 1×1 的卷积核进行的卷积操作。它在空间维度（高和宽）上不做任何聚合，其核心作用在于对输入特征图的每个像素点通过所有通道进行跨通道的交互与变换。虽然 1×1 卷积结构简单，如图 3-2(d)所示，但却是现代深度神经网络中不可或缺的“神器”，其功能概括为两方面：

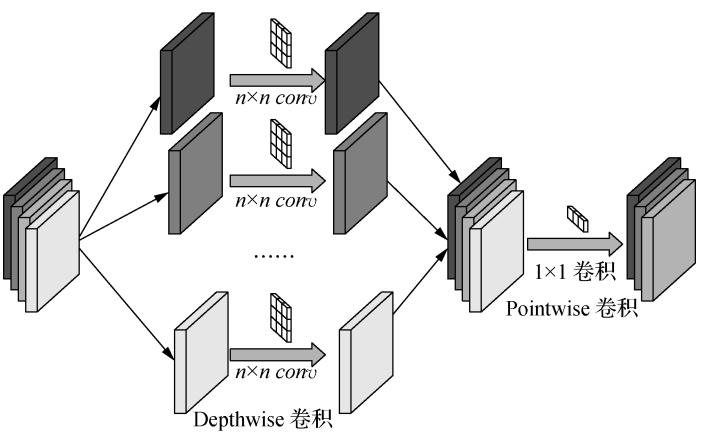
- 跨通道信息交互与降维/升维：通过控制 1×1 卷积核的数量，可以灵活地减少或增加输出特征图的通道数。
- 降维（压缩通道数）：通过使用少于输入通道数的 1×1 卷积核，可以有效减少计算量和参数量，成为一个“瓶颈”层，这在诸如 Inception 模块中至关重要。比如一张 500×500 且通道数为 100 的图像在 20 个卷积核上做 1×1 的卷积，输出特征图的尺寸变为 500×500×20。当卷积核数量为 200 时，输出特征图的尺寸会变为 500×500×200，实现网络加宽或升维。
- 升维：同样，通过使用更多的卷积核，可以在保持空间尺寸不变的情况下增加通道数，以编码更丰富的特征。
- 引入非线性：在卷积操作后通常会接一个非线性激活函数（如 ReLU）。这使得简单的线性加权和之后具备了非线性表达能力，让网络能够学习更复杂的特征。若没有非线性，多层 1×1 卷积的堆叠就退化为一个单一的线性变换。
- 计算效率高：通过先降维再进行后续昂贵的 3×3 或 5×5 卷积，可极大降低整体计算成本。
- 参数少：1×1 卷积本身的参数量极少（输入通道数×输出通道数），是进行通道变换最经济的方式。
- 灵活性：可以无缝插入到网络的任何位置，实现复杂的特征调控。
- Inception 网络系列：在 GoogLeNet 的 Inception 模块中，1×1 卷积被用于在进行大尺寸卷积之前先进行降维，以控制计算爆炸。
- 深度可分离卷积：作为深度可分离卷积的第二阶段——逐点卷积，负责将深度卷积后的结果进行通道融合，是轻量化网络的基石。
- 注意力机制：在如 SENet 的通道注意力机制中，1×1 卷积被用来生成通道权重，对特征通道进行重标定。
- 替代全连接层：在全卷积网络中，1×1 卷积可以被用来替代末端的全连接层，使网络可以接受任意尺寸的输入。

1×1 卷积即逐点卷积是构建深层网络的重要组件。尽管在二维图像上看似无效，但在处理三维特征图时，它实则为在通道维度上的加权求和操作。在卷积操作后通常会接一个非线性激活函数（如 ReLU）。这使得简单的线性加权和之后具备了非线性表达能力，让网络能够学习更复杂的特征。若没有非线性，多层 1×1 卷积的堆叠就退化为一个单一的线性变换。

(3) 可分离卷积。 可分离卷积包括空间可分为空间卷积核和深度可分卷积，如图 3-3 所示。



(a) 空间可分离卷积



(b) 深度可分离卷积

图 3-3 可分离卷积过程

空间可分离卷积:生活中的图像一般为彩色图像,所以有三个维度:高、宽、深度。空间可分离卷积是在图像的高、宽维度上。如图所示,空间可分离卷积是将原来 $3\times3(\times3)\times n$ 的卷积核分成了 $1\times3(\times3)\times n+3\times1(\times1)\times n$ 的卷积核。

深度可分离卷积:作为一种高效的卷积操作,通过将标准卷积分解为通道独立的空域卷积与通道融合的 1×1 卷积两个步骤,在保持模型表达能力的同时显著降低了计算复杂度。该结构已成为轻量化神经网络设计的核心组件,广泛应用于移动端等资源受限场景。

与标准卷积不同,深度卷积并不涉及网络的"深度",而是指其按输入通道进行独立处理的方式。在标准卷积中,每个卷积核的通道数与输入特征图的通道数相同,通过设定卷积核数量控制输出通道维度;而在深度卷积中,每个卷积核为单通道结构,维度为$(1,1,k,k)$,其数量与输入通道数 i_C 保持一致,每个卷积核仅与输入特征图的对应通道进行二维卷积运算,输出维度为$(1,i_C,o_H,o_W)$。由于深度卷积无法改变输出通道数,通常在其后接一个 1×1 标准卷积(维度为 $o_C,i_C,1,1$)以完成通道融合与维度变换,从而替代原本的 3×3 或更大尺寸的标准卷积操作。

该结构的整体计算量为 $k\times k\times i_C\times o_H\times o_W+i_C\times o_C\times o_H\times o_W$,为标准卷积的约 $1/o_C+1/(k\times k)$倍,在显著降低参数与计算负担的同时,仍能保持相近的模型表达能力。该结构由 MobileNet V_1 首次系统提出并验证,现已逐渐成为轻量化神经网络设计的标准模块

之一，推动了 DL 模型从依赖参数堆叠向注重计算效率的重要范式转变。

空间卷积核和深度可分卷积两者都旨在通过“分解”来减少计算复杂度和模型参数，但存在着本质区别：

- 空间可分离卷积分解的是单个卷积核的空间维度（从 2D 到两个 1D）。
- 深度可分离卷积分解的是卷积操作的功能（将空间聚合与通道聚合分开）。
- 实际影响：由于深度可分离卷积的通用性、稳定性和显著的效率提升，已成为现代轻量化神经网络架构的标配组件；而空间可分离卷积更多作为一种补充性的优化技巧，在特定场景下使用。因此，在当前的学术论文和技术讨论中，当提及“可分离卷积”时，若无特殊说明，通常默认指的是**“深度可分离卷积”**。

（4）逐通道卷积。逐通道卷积（Depthwise 卷积）为深度可分离卷积中关键组成部分，是一种专为降低计算量与参数量而设计的轻量级卷积操作，尤其适用于移动端等资源受限的场景。与标准卷积不同，逐通道卷积不进行跨通道的特征融合，而是对输入特征的每个通道分别施加独立的卷积操作。具体而言：

在标准卷积中，每个卷积核会同时作用于所有输入通道，并将卷积结果求和，合并为单一输出通道；而在逐通道卷积中，每个输入通道都对应一个独立的二维卷积核，输出通道数因此与输入通道数保持一致。

以一张尺寸为 5×5 像素、通道数为 3 的输入图像（形状为 5×5×3）为例，逐通道卷积将使用 3 个独立的二维卷积核，分别作用于对应的输入通道。每个卷积核仅在当前通道的二维空间内滑动计算。若采用“same”填充策略，每个通道输出的特征图尺寸与输入相同（即 5×5），如图 3-4(a)所示。因此，3 个通道共生成 3 个特征图，整个过程不涉及任何跨通道的信息交互。这种独立处理机制显著降低了模型的参数量与计算负担，为后续的通道融合操作（如 1×1 卷积）提供了高效的预处理基础。然而，逐通道卷积也存在一定局限：其输出特征图数量与输入通道数相同，无法扩展特征维度；同时，由于其对各通道独立处理，未能有效融合不同通道在同一空间位置上的特征信息。因此，在实际应用中通常需要配合逐点卷积，以将各通道特征图进行组合，从而生成新的、更具表达力的特征图。

（5）逐点卷积。逐点卷积（Pointwise 卷积）的运算与常规卷积运算非常相似，它的卷积核的尺寸为 $1\times1\times M$，M 为上一层的通道数。所以这里的卷积运算会将上一步的 map 在深度方向上进行加权组合，生成新的特征图。有几个卷积核就有几个输出特征图，如图 3-4(b)所示。由图 3-4 可得：

常规卷积的参数个数为：N_std=4×3×3×3=108；

逐通道卷积的参数个数为：N_depthwise=3×3×3=27；

逐点卷积的参数个数为：N_pointwise=1×1×3×4=12。

深度可分离卷积的参数由逐通道和逐点卷积两部分相加得到：

N_separable=N_depthwise+N_pointwise=39。

相同的输入，得到 4 张特征图，深度可分离卷积的参数个数是常规卷积的约 1/3。因此，在参数量相同的前提下，采用深度可分离卷积的神经网络层数可以做的更深。在可分离卷积中，通常将卷积操作拆分成多个步骤；而在神经网络中通常使用的为深度可分离卷积。

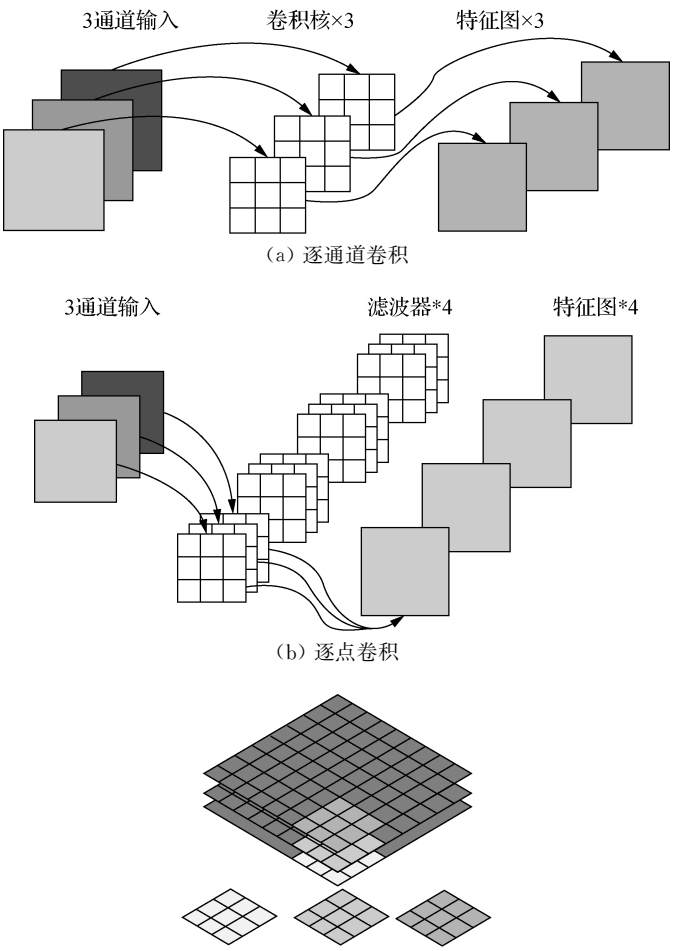


图 3-4 逐通道和逐点卷积过程

举个例子，假设有一个3×3大小的卷积层，其输入通道为16、输出通道为32，则一般的操作为用32个3×3的卷积核来分别同输入数据卷积，这样每个卷积核需要3×3×16个参数，得到的输出是只有一个通道的数据。之所以会得到一通道的数据，是因为刚开始3×3×16的卷积核的每个通道会在输入数据的每个对应通道上做卷积，然后叠加每一个通道对应位置的值，使之变成了单通道，则32个卷积核一共需要(3×3×16)×32=4 068个参数。

由于采用的是1×1卷积的方式，此步中卷积涉及的参数个数计算为：N_pointwise=1×1×3×4=12。经过Pointwise卷积后，同样输出了4张特征图，与常规卷积的输出维度相同。

(6) 分组卷积。分组卷积是一种通过减少通道连接来降低计算复杂度和模型参数的卷积策略。其核心操作可分解为以下三个步骤，如图3-5所示：

- 分组：将输入特征图在通道维度上均匀分为 g 组。同时，卷积核也相应地被分为 g 组。每一组卷积核只与对应组的输入特征图进行卷积运算。
- 组内卷积：在每个组内独立进行标准的卷积操作。每个组都会产生一个输出特征图。
- 拼接：将所有 g 组产生的输出特征图在通道维度上进行拼接，形成再的输出。

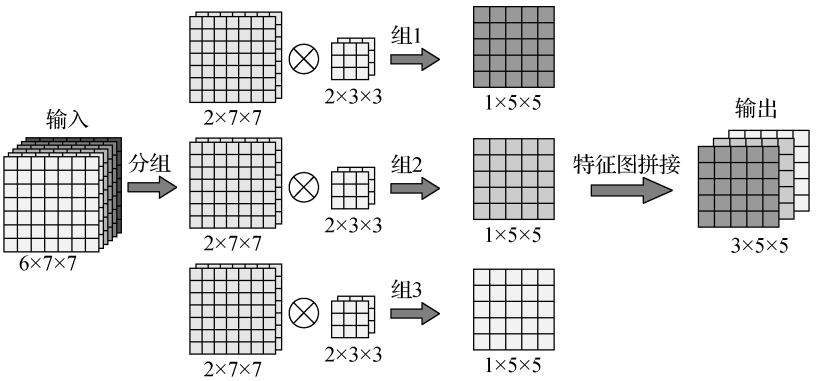


图 3-5　分组卷积

核心优势如下：计算量和参数量均减少为标准卷积的 1/groups。这在构建轻量级模型时至关重要；强制模型在不同的组内学习不同的特征模式，可以看作是一种结构化的正则化。

与深度可分离卷积的关系：当 groups 数量等于输入通道数时，分组卷积就演变为逐通道卷积。因此，深度可分离卷积可以看作是“分组数＝通道数”的分组卷积＋1×1 逐点卷积的组合。

现代应用：除了用于构建轻量级模型（如 MobileNet），在 ResNeXt 等高性能模型中，通过使用分组卷积并增加组的数量（称为“基数”），可以在不显著增加计算成本的前提下提升模型的表达能力。

(7) 转置卷积。转置卷积（亦称反卷积或分数步长卷积）是一种可实现特征图上采样的可学习操作。与使用固定插值系数的传统上采样方法不同，转置卷积通过可训练的卷积核参数，使网络能够自适应地学习最优的特征重建方式。其核心原理可理解为标准卷积操作的逆向映射过程：通过对原始卷积核进行转置排列，并将输入元素与卷积核权重相乘后累加到输出的对应位置，从而实现输入特征图到高分辨率输出特征图的空间维度恢复。需要特别说明的是，转置卷积并非卷积的精确逆运算，而是一种结构上的对称操作，其本质是通过学习得到的填充和重叠模式来重建特征图的空间维度。

标准卷积是一个输出像素是其输入局部“感受野”内像素的加权和；而转置卷积为一个输入像素将其值乘以卷积核权重，并将其“贡献”到输出中一个更大的局部区域（可称为“影响域”）。这是一种一对多的映射关系。转置卷积可以通过在输入特征图元素间插入空白（填充 0），然后进行标准卷积来实现。其具体步骤如图 3-6 所示：

输入准备：在输入特征图的每个元素之间插入（stride－1）个零值。这是实现“一对多”映射和扩大尺寸的关键。

边缘填充：在输入特征图的外围进行（kernel_size-padding－1）的零填充。

执行卷积：使用旋转了 180 度的原始卷积核，在填充后的输入上进行步长为 1 的标准卷积。

转置卷积可学习的上采样：与双线性插值等固定算法不同，转置卷积的核权重在训练中学习，能更好地适应特定任务。其核心应用场景在于：

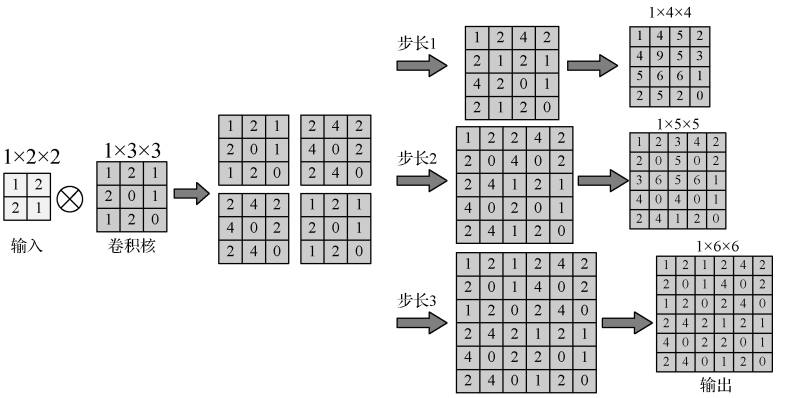


图 3-6　转置卷积过程

语义分割：在编码器—解码器结构中，解码器使用转置卷积从低分辨率特征图恢复出高分辨率分割图。

生成对抗网络：从随机噪声生成高分辨率图像。

自编码器：在解码器部分重建输入尺寸。

转置卷积是一种通过插入零值和填充来实现上采样的可学习操作。它实现了从小的输入到大的输出的“一对多”映射，是现代深度网络中实现精细重建和生成任务的关键组件。在编码器—解码器类架构（如 U-Net）中，转置卷积发挥着关键作用。编码器通过卷积和下采样不断压缩空间维度并提取抽象特征，而解码器则通过转置卷积逐步恢复特征图尺寸，同时结合跳跃连接传递的浅层细节特征，再实现精确的像素级定位与分割。这种可学习的上采样机制，使网络能够更有效地平衡深层语义信息与浅层空间细节，显著提升分割边界的重建质量。

（8）多尺度卷积。核心思想是让神经网络能够“同时”看到不同大小、不同抽象层次的特征。

想象一下，当你辨认一张图像中的物体时，你既会关注局部的细微纹理（如猫的胡须），也会关注整体的轮廓形状（如猫的整个身体）。多尺度卷积是为了让网络具备这种能力而设计的，它通过并行或串行地使用具有不同感受野的卷积核，来捕获图像中从细节到全局的多种尺度信息。

多尺度卷积有多种实现方式，它们在结构上各有千秋，但目标一致。

应对尺度变化：实际中物体大小各异，多尺度结构使模型对不同尺寸的目标都具有鲁棒性。

丰富特征表征：融合了细节和上下文信息，使得特征表达更加全面和强大。

提升任务性能：在图像分类、目标检测、语义分割等几乎所有核心视觉任务上，引入多尺度思想都带来了显著的性能提升。

效率与精度的平衡：通过精心设计（如 1×1 卷积降维），可以在增加有限计算成本的前提下，获得巨大的精度收益。

总结来说，多尺度卷积是现代 DL 架构设计中一项关键的技术，它赋予了模型更接近人类视觉的、多层次的场景理解能力。Inception 模块为比较典型的多尺度卷积模块。该模块

通过并行连接多种不同尺度的卷积与池化操作,并在深度维度进行拼接,以高效模拟多尺度视觉处理过程。这种设计允许网络在增加深度和宽度的同时,有效控制计算复杂度并缓解过拟合。

(9) 空洞卷积。空洞卷积是一种不通过池化来增大感受野的巧妙方法。空洞卷积,也称为膨胀卷积或带孔卷积,是一种通过向标准卷积核内部注入空洞(零元素)以扩大其感受野的特殊卷积方法,而无需增加参数量或进行下采样。核心思想与工作原理为空洞卷积的核心是在标准卷积核的权重元素之间,系统地插入固定间隔的零值。

工作原理:在标准卷积核的权重之间"插入"空格(零值),通过一个空洞率参数来控制间隔大小。空洞率越大,感受野也越大,而参数量和计算量保持不变。

优势:在不丢失分辨率(不下采样)情况下,快速获得巨大的感受野,捕获更广阔的上下文信息,非常适合需要精确定位的任务(如语义分割)。

经典模型:DeepLab 系列模型广泛使用了空洞卷积和它的变体空洞空间金字塔池化(ASPP),后者可以看作是 Inception 思想与空洞卷积的结合。

空洞率:这是空洞卷积的关键超参数,记为 R(或 Rate),它定义了卷积核元素之间的间隔距离,当 $k=1$ 时,即为标准卷积;当 $k>1$ 时,空洞被引入。对于空洞率为 R 的空洞卷积,实际卷积核感受野大小为:

$$k_r = k + (k-1) \cdot (R-1) \tag{3-7}$$

式中,k 为原始卷积核大小,k_r 为扩张后的卷积核大小。

空洞卷积表示:

$$y(i) = \sum_{l=1}^{k} x(i+l) \cdot h(l) \tag{3-8}$$

式中,$x(i)$ 为输入图像,$w(k)$ 为长度为 R 的卷积核,r 为步长,$y(i)$ 为空洞卷积输出。

示例:一个 3×3 卷积核,$R=2$ 的空洞卷积,其有效感受野相当于一个 3+(3−1)×(2−1)=5×5 的标准卷积;$R=3$ 的空洞卷积的有效感受野相当于 3+(3−1)×(3−1)=7×7。图 3-7 分别为 Rate=1,2,3,4,不同步长的卷积核。

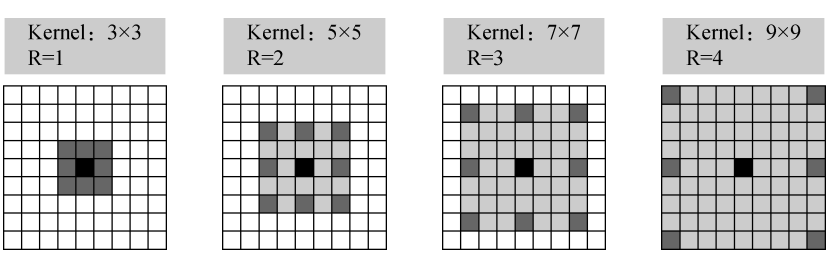


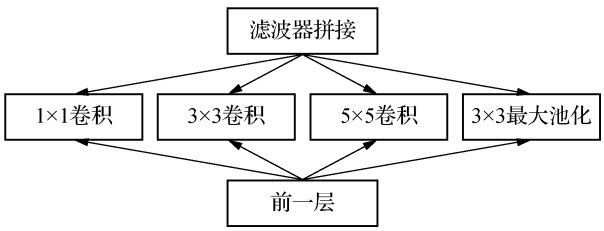
图 3-7 不同空洞率的卷积核

(10) 空洞 Inception 模块。Inception 模块的设计灵感来源于一个直观的视觉处理假设:图像中的关键信息可能以不同尺寸分布。因此,一个理想的网络应在同一层级提供多尺度特征提取能力:

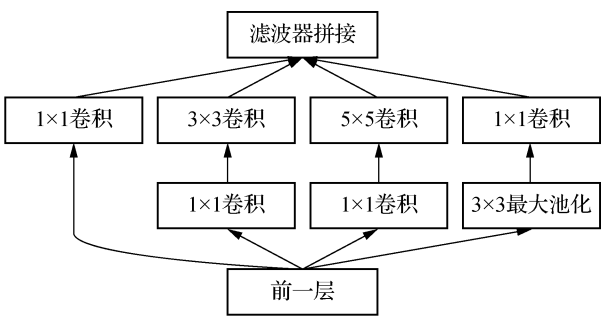
当感兴趣区域分布更全局时,网络可倾向于利用较大卷积核(如 5×5)路径的输出。

当感兴趣区域分布更局部时,网络可倾向于利用较小卷积核(如 1×1)路径的输出。

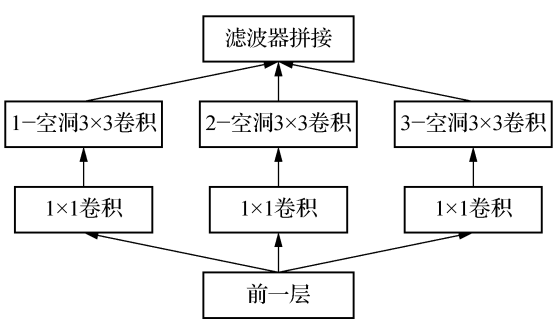
通过并行堆叠这些路径并将结果拼接，网络能够自适应地从不同尺度聚合视觉信息，从而增强其特征表达能力。Inception 改进模块版本很多，图 3-8(a)和(b)为经典的两个模型。



(a) Inception 模块



(b) 降维后的空洞 Inception 模块



(c) 空洞 Inception 模块

图 3-8 Inception 模块

原始的 Inception 模块结构如图所示，它并行地集成了四种不同类型的操作：

1×1 卷积：用于捕获极局部的特征交互，并作为廉价的线性变换与降维工具。

3×3 卷积：用于捕获中等范围的视觉模式。

5×5 卷积：用于捕获更大范围的上下文信息。

3×3 最大池化：用于保留特征的原始空间响应，并引入某种程度的平移不变性。

关键创新为 1×1 卷积的引入：为了避免简单拼接带来的计算爆炸，Inception 模块创造性地在 3×3 和 5×5 卷积之前，以及池化操作之后，大量使用了 1×1 卷积。1×1 卷积在此扮演了两个关键角色：

降维：通过减少输入特征的通道数，显著降低了后续大卷积核操作的计算量。

增加非线性：在每个 1×1 卷积后接 ReLU 激活函数，在不显著改变感受野的情况下，增强了模型的非线性表达能力。

Inception 模块通过其独特的“并行处理、深度拼接”架构，实现了对多尺度特征的高效、自适应提取，为后续更复杂的网络设计（如 Inception－$v_2/v_3/v_4$）奠定了坚实基础。核心优势为：

多尺度特征提取：并行结构使网络能够同时捕捉不同粒度与范围的视觉特征。

计算效率高：通过 1×1 卷积进行降维，在增加网络宽度和深度的同时，巧妙地控制了参数数量和计算成本。

缓解过拟合：模块的稀疏连接结构（由 1×1 卷积实现）与深度拼接方式，作为一种正则化，有助于防止模型过拟合。

空洞 Inception 模块保留 U－Net 中的 3×3 卷积核，选用空洞 Inception 进行遥感飞机图像的多尺度特征提取，该结构如图 3－8(b)所示。由 4 个分支组成，包含 4 个 1×1 卷积、3 个 3×3 空洞卷积，空洞率分别为 1、2、3，1 个 3×3 最大池化和一个连接操作组成，设置空洞率 Rate＝1，2，3，卷积核大小为 3×3，空洞卷积后感受野分别为 3×3、5×5、7×7，感受野增大，但是网络参数不增加。因此，不同膨胀率的空洞卷积可以在不增加计算量的情况下，获得更强的分类特征。

4. 卷积应用

在计算机视觉领域，从 Inception 模型的并行多分支架构，到特征金字塔网络（FPN）的层级融合机制，再到空洞卷积的高效感受野扩展，这些多尺度处理方法共同推动了视觉技术的持续进步。其中，空洞卷积通过其巧妙的“内部填充”机制，成功实现了感受野的指数级扩展与上下文信息的有效捕获，成为语义分割等需要兼顾全局语境与局部细节的视觉任务中不可或缺的核心工具。其在保持计算效率的同时提升模型性能的特点，使其在现代 DL 架构中占据重要地位。

下面介绍空洞卷积的核心机制与应用价值。

（1）感受野的指数级扩展。空洞卷积通过引入空洞率这一超参数，在标准卷积核的权重之间插入零值，从而在不增加卷积核物理尺寸的情况下显著扩大感受野。例如，一个空洞率为 2 的 3×3 空洞卷积，其有效感受野等同于 5×5 的标准卷积，却仅需 9 个参数，计算成本与 3×3 标准卷积一致。

（2）语义分割中的关键作用。这是空洞卷积最经典的应用。通过堆叠多个不同空洞率的空洞卷积层（构成“空洞卷积空间金字塔”，如 ASPP 模块），网络能够在保持高分辨率特征图的同时，捕获从局部细节到全局上下文的多种尺度信息，从而精细地描绘物体边界。作为空洞卷积最经典的应用场景，语义分割任务通过堆叠多个不同空洞率的卷积层构建空洞空间金字塔池化（ASPP）模块。该设计使网络能够在保持特征图高分辨率的同时，捕获从局部细节到全局上下文的多种尺度信息，从而实现对物体边界的精细描绘。

（3）目标检测与特征提取。在需要精确定位的目标检测任务中，空洞卷积用于提取包含更丰富上下文信息的高分辨率特征，有效改善了小目标检测和边界定位的准确性。

（4）传统下采样的替代方案。与传统 CNN 依赖池化层或步长卷积进行下采样的方法不同，空洞卷积提供了一种无需降低空间分辨率即可获得大感受野的创新方案，有助于生成更精细、空间信息更完整的输出特征。

（5）潜在挑战与优化策略。空洞卷积的主要局限在于网格效应——当堆叠多个高空洞

率的卷积层时,有效权重可能分布在稀疏的网格点上,导致局部信息丢失。现代架构通过设计更复杂的空洞率模式(如“混合扩张卷积”)来缓解此问题,确保权重分布的连续性和完整性。

(6) 空洞空间金字塔池化(ASPP)模块。在DeepLab系列等先进的语义分割模型中,ASPP模块已成为标准配置。它使网络能够同时“看到”树木(局部细节)和森林(全局语境),从而在复杂场景中实现更加准确和鲁棒的分割效果,特别在医学影像分析、自动驾驶场景理解等对精度要求极高的领域展现出卓越性能。这种基于空洞卷积的多尺度融合机制,为语义分割任务提供了一种既保持细节又兼顾上下文的优雅解决方案。在语义分割任务中,空洞卷积发挥着至关重要的作用,其最经典的应用体现在ASPP模块的设计中。该模块通过并行堆叠多个具有不同空洞率的空洞卷积层,构建了一个高效的多尺度特征提取结构。ASPP模块的核心组成与工作机制:

① 多尺度上下文捕获

并行设置多个空洞卷积分支,分别采用不同的空洞率(如1,6,12,18);

小空洞率分支专注于局部细节和细粒度特征;

大空洞率分支负责捕获全局上下文和远程依赖;

各分支输出的特征图在通道维度进行融合。

② 分辨率的保持

所有空洞卷积操作均不涉及下采样;

始终保持特征图的空间分辨率不变;

避免了传统池化操作导致的信息损失。

③ 边界精细化

通过融合局部细节与全局语境信息;

显著提升了对物体边界的识别精度;

能够生成更加精细的分割掩码。

空洞卷积通过一种巧妙的“内部填充”机制,实现了感受野的指数级扩张,成为在语义分割等需要兼顾全局上下文与局部细节的任务中不可或缺的工具。

3.3 池化

池化(Pooling)是DL中的一种关键操作,主要用于对特征图进行下采样,以缩减其空间尺寸(高度和宽度),从而显著减少模型的计算复杂度和参数数量。池化层通常置于卷积层后,通过对局部区域进行聚合,提取具有鲁棒性的主要特征,同时增强模型对输入位置变化的鲁棒性(即平移不变性)。

1. 池化定义

与卷积层不同,池化层本身不包含可学习的参数,其行为完全由超参数(如池化核大小、步长和填充方式)决定。

池化操作是逐通道独立进行,输出通道因数与输入通道数保持不变,输出尺寸通用表示:

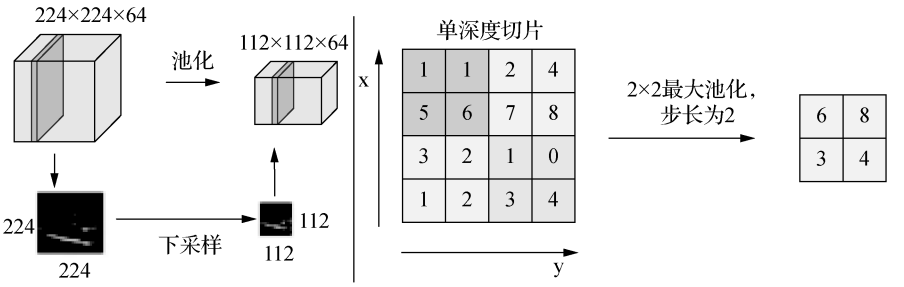


图 3-9　池化

$$H_{\text{out}}=\frac{\left[(H_{\text{in}}+2\times\text{padding}_h-\text{dilation}_h(\text{kernel\_size}_h-1)-1)/S_h\right]}{\text{stride}_h}+1 \quad (3-9)$$

式中,kernel_size(池化窗口大小)指定池化窗口的尺寸,类型为整数或元组(如 2 或(3.2)),默认值为 2,该参数决定了每次池化操作所覆盖的局部区域范围。stride(步长)定义池化窗口每次移动的像素距离,类型为整数或元组。默认值通常等于 kernel_size,此时为非重叠池化;若步长小于窗口尺寸,则形成重叠池化。

步长直接影响输出特征图的空间尺寸。padding(填充)控制在输入特征图边界四周添加的零值填充层数,类型可为整数、元组或字符串(如 'valid' 或 'same')。默认值为 0(即 'valid',无填充)。填充可用于调控输出特征图的尺寸。dilation(膨胀率)定义池化窗口中各元素之间的间距,类型为整数或元组,默认值为 1(表示连续窗口)。增大膨胀率可形成稀疏的池化窗口,在不增加参数的情况下扩大感受野。

池化作用归纳如下:

(1) 特征不变性:池化操作使模型更加关注特征是否存在,而不是特征的具体位置。这种不变性包括平移不变性、旋转不变性和尺度不变性。

(2) 特征降维:池化通过在空间范围内进行维度约减,使模型可以抽取更广范围的特征,同时减小了下一层的输入大小,减少计算量和参数数量。

(3) 防止过拟合:池化在一定程度上防止了过拟合,更方便优化。

(4) 实现非线性(类似 ReLU),扩大感受野。

2. 常见的池化类型

(1) 最大池化:最大池化是最常见的池化方法之一。它通过在池化窗口内选择最大值来进行下采样。如图 3-10 所示,对于一个 2×2 的池化窗口,最大池化会选择窗口内的最大值作为输出。

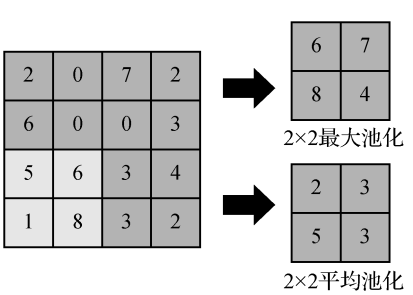


图 3-10　最大和平均池化

(2) 平均池化:平均池化通过计算池化窗口内的平均值来进行下采样,如图 3-10 所示。与最大池化不同,平均池化更关注整体信息,而不是局部最大值。

(3) 全局池化:包括全局平均池化与全局最大池化,将整个特征图的每个通道压缩为单一数值,常用于替代全连接层以减少参数并增强模型泛化能力。

(4) 自适应池化:允许用户直接指定输出特征图的尺寸,而非依赖固定的池化核与步长,适用于处理多尺度输入或需要固定尺寸输出的场景。

(5) 重叠池化:重叠池化是指相邻池化窗口之间有重叠区域,可以更细致地捕捉特征。当池化步长小于池化核尺寸时,相邻池化窗口存在重叠区域。若定义池化窗口的大小为sizeX,定义两个相邻池化窗口的水平位移或竖直位移为 stride,此时 sizeX>stride,可在下采样过程中保留更丰富的特征信息。

随着网络结构的发展,也出现了如空间金字塔池化、随机池化(Stochastic Pooling)、分数最大池化(Fractional Max-Pooling)等变体,以适应不同任务对特征保留与泛化的需求。池化类型的合理选择与组合,对模型的效率与性能具有重要影响。

(6) 全局平均池化:在 CNN 训练初期,卷积层通过池化层后一般要接多个全连接层进行降维,再利用 Softmax 分类,该做法使得全连接层参数很多,降低了网络训练速度,且容易出现过拟合的情况。在这种背景下,有学者提出使用全局平均池化来取代最后的全连接层。用很小的计算代价实现了降维,更重要的是 GAP 极大减少了网络参数(CNN 网络中全连接层占据了很大的参数)。该池化是一种特殊的平均池化,它不划分若干矩形区域,而是将整个特征图 Q 中所有的元素取平均输出到下一层。其定义如下:

$$y_k = \frac{1}{|R|}\sum_{(i,j)\in R} x_{kij} \tag{3-10}$$

式中,y_k 为与第 k 个特征图的全局平均池化输出值,x_{kij} 为第 k 个特征图区域 R 中位于(i,j)处的元素,$|R|$表示第 k 个特征图全部元素的个数。

(7) 全局最大池化:全局最大池化是一种常用的池化操作方法,用于降低特征图的空间维度。在全局最大池化中,将特征图中的每个通道的所有值取最大值,得到一个与通道数相同的向量,并将这些向量拼接在一起形成再的特征向量。该池化的主要作用是提取图像或特征图中最显著的特征。通过取最大值,可以保留重要的特征并丢弃不重要的信息。这种操作对于图像分类、目标检测和图像分割等任务都有很好的效果。

(8) 空间金字塔池化:空间金字塔池化(SPP)是一种能够处理任意输入尺寸的 CNN 结构,通过在不同尺度上进行池化与融合,生成固定长度的特征表示,能够处理不同尺寸的输入图像,从而消除网络对输入图像尺寸的限制。这一过程通过在不同尺度上进行池化来实现,从而保留了空间信息。SPP 层位于最后一个卷积层和第一个全连接层之间,其工作流程如下:

特征图输入:SPP 层接收来自卷积层的特征图,这些特征图可以是任意大小。

分层池化:SPP 层将特征图划分为多个空间区域(bins),并在每个区域内执行最大池化(max pooling)。通常使用多层次的池化策略,例如:

第一层:1 个大区域(全局池化)

第二层:4 个区域(2×2)

第三层:16 个区域(4×4)

固定输出:通过上述步骤,SPP 层能够输出统一维度的特征向量,便于直接馈入后续的全连接层进行分类或下游任务,增强了网络对不同尺度输入的适应性和全连接层的兼容性。从结构图 3-11 可以看出,其通过 3 个不同扩展率的感受野组成,再由 1 个全局均值池化层合并 3 个通道的输出。

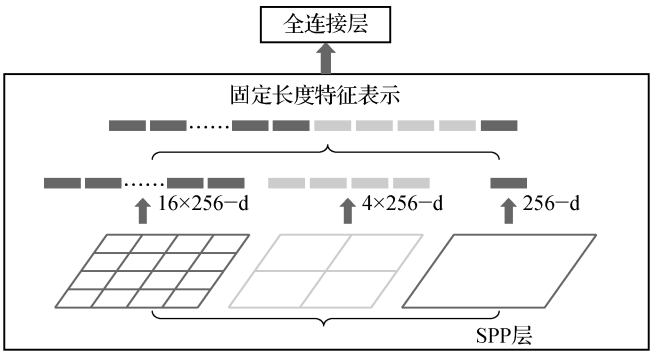


图 3-11　SPP 基本结构

(9) 自适应平均池化层:自适应平均池化是一种对输入进行自适应调整大小的池化操作。在传统的平均池化中,人们需要指定池化操作的输出尺寸,而自适应平均池化则可以根据输入的大小自动决定输出的尺寸。

具体来说,自适应平均池化将输入张量划分成不同大小的区域,并对每个区域中的元素取平均值作为输出。每个区域的大小根据输入的尺寸和输出的尺寸进行自适应计算,从而达到自适应调整大小的效果。自适应池化 Adaptive Pooling 与标准的 Max/AvgPoolingl 区别在于,自适应池化 Adaptive Pooling 会根据输入的参数来控制输出 output size,而标准的 Max/AvgPooling 是通过 kernel size,stride 与 padding 来计算 output size。

自适应平均池化在 DL 中常用于处理输入尺寸变化的情况,特别是当需要使用全连接层或者固定尺寸的卷积层时。通过自适应平均池化,可以使得网络对输入的大小更加鲁棒,不受固定尺寸限制。

(10) 自适应最大池化:自适应最大池化是一种特殊类型的最大池化操作,特别是计算机视觉任务中一种非常重要且实用的池化操作。用户无需手动指定池化核的大小和步长,而是直接指定期望的输出尺寸。该操作会自动计算出所需的池化核参数,以确保无论输入特征图的尺寸如何,都能精确地输出目标大小的特征图。

具体地说,它计算输入张量中每个通道的最大值,并使用这些通道的不同最大值对每个通道进行汇总,从而产生输出张量。池化大小可以在每个通道上是不同的。这使得自适应最大池化比传统最大池化更灵活,并且在处理不同大小的输入时特别有用。

假设:输入特征图尺寸为 $H_{in}\times W_{in}$,指定的输出尺寸为 $H\_out\times W\_out$。系统会自动计算出自适应池化核的有效大小和步长:

池化核高度=ceil(H_{in}/H_out);池化核宽度=ceil(W_{in}/W_out)

步长高度=floor(H_{in}/H_out);步长宽度=floor(W_{in}/W_out)

然后,使用这些计算出的核大小和步长,在输入特征图上执行标准的**最大池化**操作。

(11) 混合池化。为了提高训练较大 CNN 模型的正则化性能,受 Dropout(将一半激活函数随机设置为 0)的启发,Dingjun 等人提出了一种随机池化 MiixPooling 的方法,随机池化用随机过程代替了常规的确定性池化操作,在模型训练期间随机采用了最大池化和平均池化方法,并在一定程度上有助于防止网络过拟合现象。其定义如下:

$$y_{kij}=\lambda\cdot\max_{(p,q)\in R_{ij}}x_{kpq}+(1-\lambda)\cdot\frac{1}{|R_{ij}|}\sum_{(p,q)\in R_{ij}}x_{kpq} \tag{3-11}$$

式中，λ 是 0 或 1 的随机值。混合池化以随机方式改变了池调节的规则，这将在一定程度上解决最大池和平均池所遇到的问题。

(12) 组合池化。组合池化则是同时利用最大值池化与均值池化两种优势而引申的一种池化策略。常见组合策略有两种：C_t 与 Add。常常被当作分类任务的一个 trick，其作用为丰富特征层，maxpool 更关注重要的局部特征，而 average pooling 更关注全局特征。

(13) 随机池化。随机池化是一种旨在为模型引入随机性的下采样方法，其核心思想是根据特征图中局部区域内各元素的数值大小作为概率，进行随机采样，而非像最大池化那样直接选择最大值。这种机制为模型提供了一定的正则化效果，有助于抑制过拟合。

随机池化的计算过程清晰且易于实现，主要分为以下三个步骤：

计算概率分布：对于池化窗口（如 3×3）内的所有元素，计算其数值总和，然后将每个元素值除以该总和，得到一个概率矩阵。该矩阵中每个位置的值为其被选中作为输出的概率。

依概率随机采样：将整个池化窗口视为一个多项分布，根据上一步得到的概率矩阵，随机选择一个位置。

输出该位置值：将被选中位置的原始元素值（而非概率值）作为池化操作的输出结果。

假设 3×3 特征图中 Pooling 区域元素值如下：

元素值和 sum=0+1.1+2.5+0.9+2.0+1.0+0+1.5+1.0=10，方格中的元素同时除以 sum 后得到的矩阵元素为：每个元素值表示对应位置处值的概率，现在只需要按照该概率来随机选一个。将其看作是 9 个变量的多项式分布，然后对该多项式分布采样即可，theano 中有直接的 multinomial()来函数完成。当然也可以自己用 0—1 均匀分布来采样，将单位长度 1 按照那 9 个概率值分成 9 个区间（概率越大，覆盖的区域越长，每个区间对应一个位置），随机生成一个数后看它落在哪个区间。比如若随机采样后的矩阵为：

0	1.1	2.5
0.9	2.0	1.0
0	1.5	1.0

0	0.11	0.25
0.09	0.2	0.1
0	0.15	0.1

0	0	0
0	0	0
0	1	0

图 3-12 随机采样后的矩阵

则 pooling 值为 1.5，使用随机 pooling 时（即 test 过程），其推理过程也很简单，对矩阵区域求加权平均即可。比如对上面的例子求值过程为：说明此时对小矩形 pooling 后为 1.625。在反向传播求导时，只需保留前向传播已经记录被选中节点的位置的值，其他值都为 0，这和 max-pooling 的反向传播非常类似。

核心优势：

正则化效果：通过引入随机性，防止模型过于依赖某些特定的神经元，从而提升泛化能力，有助于防止过拟合。

软性最大值选择：它不像最大池化那样总是选择最强的特征，也不像平均池化那样进行平滑，而是以一种概率性的方式兼顾强特征与次强特征。

局限性：

随机性导致不稳定：由于每次前向传播的采样结果可能不同，使得模型的输出存在不确定性，这在某些需要确定性行为的应用中可能不适用。

现代替代方案：随着 Dropout、数据增强等更简便有效的正则化技术的普及，以及像 DropBlock 这种专门针对卷积层的正则化方法的出现，随机池化在实际应用中的普及度已不如前。

随机池化是一种构思巧妙的下采样策略，它将概率采样融入池化过程，为模型提供了有效的随机性。尽管在当前实践中已不常用，但理解其思想对于深入认识网络正则化与特征下采样的多样性仍具有重要价值。

(14) 局部重要性池化。在 CNN 中，空间下采样层（如池化层或步长卷积）被广泛用于扩大感受野并降低计算内存消耗。然而，此类操作通常依赖于预定义的固定策略（如取最大值或平均值），往往无法有效保留对任务至关重要的细节信息，从而导致模型精度损失。

为解决这一问题，有研究者从局部重要性的角度出发，提出了局部重要性池化。该方法的核心思想是通过基于输入数据学习得到的自适应重要性权重，引导下采样过程自动聚焦于更具判别性的特征，从而在压缩特征图的同时增强其表征能力。

常见的下采样方法存在固有缺陷，存在的问题如下：

最大池化：其选取的“最大值”未必是当前区域内最具判别力的特征。在梯度反向传播过程中，除非当前最大值被抑制，否则网络难以更新并学习到那些更具区分度的非最大值特征。

平均池化：对所有特征响应进行均等化处理，无法突出关键特征。

步长卷积：其采样位置是固定且均匀间隔的，这种刚性结构缺乏对输入内容的自适应能力，容易忽略分布在非采样点上的重要信息。

一个理想的池化操作应满足以下两个关键条件：

自适应的采样位置：下采样的位置应能根据输入内容动态调整，打破固定间隔的采样模式。

可学习的重要性函数：用于评估特征显著性的函数 FF 本身应能通过数据驱动的方式进行学习，而非人为预设。

局部重要性池化通过为每个局部区域内的元素学习一个重要性权重，并依据此权重对特征进行加权聚合，从而同时满足了以上两点要求。这使得下采样过程不再是简单的信号压缩，而转变为一种保留并增强关键特征的判别性操作。

(15) 软池化(Soft-pooling)。软池化是一种基于 Softmax 加权的池化方法，能够在保留输入特征基本属性的同时，增强响应更显著的特征激活。与直接选取最大值的最大池化(Max-pooling)不同，软池化是可微操作，在反向传播过程中能为每个输入分配梯度，从而有助于提升模型训练效果。计算流程如下：通过滑动窗口在特征图上选取局部区域；对该区域内的每个特征值进行指数运算，得到对应的权重；再将各特征值与其权重相乘；最后对加权后的结果进行求和。这种方式使得局部区域内所有特征值均参与计算，且重要性更高的特征会获得更大权重。

给定一个特征图 A，设 R 为某个局部区域（如 $k\times k$ 的窗口），a_i 为该区域内的第 i 个特征值。Soft-pooling 的计算步骤如下：

对区域 R 内的每个特征值 a_i，通过 Softmax 函数计算其权重 w_i：

$$w_i = \frac{\exp(a_i)}{\sum_{j \in R} a_j} \tag{3-12}$$

加权聚合：使用权重对特征值进行加权平均，得到该区域的输出：

$$\text{output} = \sum_{j \in R} w_i \cdot \alpha_i \tag{3-13}$$

对于一个局部区域 R（包含 n 个特征值）：

$$\text{Soft-pooling}(X) = \sum_{i=1}^{N} \left(\exp(a_i) \Big/ \sum_{j=1}^{N} \exp(a_j) \right) \cdot a_i \tag{3-14}$$

Soft-pooling 的特点如下：

- **可微**：由于使用指数和归一化，整个操作完全可微，适合端到端训练。
- **保留更多信息**：相比最大池化（仅保留最大值）和平均池化（平等对待所有值），Soft-pooling 通过权重分配，既强调较大值，又保留了一定程度的结构信息。
- **自适应权重**：权重根据特征值的大小动态调整，响应较大的特征获得更高权重。

(16) 双线性池化。双线性池化是一种主要用于细粒度图像分类任务的高级特征融合技术，通过计算两个特征向量之间的外积来捕捉它们的二阶交互信息，进而生成一个融合后的全局特征描述符。具体而言，该方法对两个特征图在空间对应位置上的特征向量进行外积运算，并对所有位置的结果进行聚合，从而得到一个能表征特征间复杂关联的高维表示。与依赖一阶组合的传统方法（如拼接、求和或池化）相比，双线性池化能够更有效地建模特征之间的相互关系。它尤其适用于融合功能互补的特征，例如将表示物体部件位置的信息与描述部件外观的识别信息相结合，从而实现对细微视觉差异的精准判别。

根据两个输入特征的来源，可将其分为两类：

同源双线性池化：当特征 x 与 y 来源于同一个特征提取器（即 $x=y$）时，该操作退化为计算特征的协方差矩阵估计，因此，也被称为二阶池化。

多模双线性池化：当特征 x 与 y 来源于两个不同的特征提取器时，该操作旨在融合两种异构或互补的特征模式。

双线性池化的计算过程可系统性地分解为以下四个步骤：

计算局部外积：对于图像特征图上的每一个空间位置 i，计算该位置上两个特征向量 x_i 和 y_i 的外积，得到一个二阶矩阵：

$$b_i = x_i y_i^{\mathrm{T}} \tag{3-15}$$

全局池化聚合：对所有空间位置 i 上得到的外积矩阵 b_i 进行求和池化，得到一个全局的矩阵 ξ：

$$\xi = \sum_{i=R}^{N} b_i \tag{3-16}$$

式中，R 表示所有空间位置的集合。

向量化：将聚合得到的矩阵 ξ 展平为一个高维向量。

归一化:对得到的双线性向量进行归一化处理,以提升数值稳定性并优化其用于后续分类器的性能,得到的 z 即为融合后的特征表示。

3. 池化应用

池化在保留最活跃特征的同时,显著降低数据的空间维度。以下是池化操作的几个核心应用与价值:

(1) 赋予模型平移不变性。这是池化最重要的作用之一。平移不变性指的是当输入图像中的目标物体发生微小平移时,模型的输出结果能保持不变。

工作原理:以最大池化为例,它只记录一个局部区域内的最强响应。只要这个最强特征(如"猫耳朵")仍然存在于池化窗口的某个位置,无论它在该窗口内如何轻微移动,最大池化的输出都是相同的。

价值:这使得模型不再过分关心特征的确切位置,而更关注特征是否存在。这极大地提升了模型对目标位置变化的鲁棒性,是图像分类任务成功的关键之一。

(2) 显著降低计算与内存开销。池化通过降低特征图的分辨率(例如,一个 2×2 最大池化会将特征图的宽和高各减半,使像素点总数变为原来的 1/4),带来两大直接好处:

减少参数数量:后续层(如全连接层或卷积层)需要处理的输入数据量大幅减少,从而有效控制了整个模型的参数量。

加速训练与推断:更少的数据意味着更少的浮点运算和内存占用,使得训练更高效,模型也更容易部署在资源受限的环境中。

(3) 扩大感受野,整合上下文信息。在 CNN 中,每一层的感受野定义了其神经元能"看到"的原始输入图像的范围。池化层通过合并相邻神经元的输出,使后续层中单个神经元的感受野能够覆盖原始图像中更广的区域。这有助于网络在更高层级学习到更加抽象和全局的特征。

(4) 在一定程度上防止过拟合。池化通过提供一种抽象的、局部平移不变的特征表示,在一定程度上减少了模型对训练数据中特定噪声和细节的过度敏感,从而有助于降低过拟合的风险,可以将其视为一种轻量的正则化手段。

常见的池化类型与应用场景如表 3-3 所示。

表 3-3 池化类型与应用场景

池化类型	操作方式	特点与适用场景
最大池化	取池化窗口内的最大值	最常用。能更好地保留纹理等尖锐特征,对噪声不敏感。广泛应用于图像分类、目标检测等大多数视觉任务。
平均池化	取池化窗口内的平均值	生成更平滑的特征图,有时会模糊重要细节。常用于网络的最后阶段(如 ResNet),将特征图平滑地转换为一个向量。
全局平均池化	对整个特征图求平均值	常用作全连接层的替代品。每个特征图输出一个值,极大减少参数,有效防止过拟合,常用于轻量级网络和现代架构(如 GoogleNet,SqueezeNet)。

双线性池化作为一种基于二阶统计的特征融合机制,为 DL 模型提供了强大的结构化表征能力。随着紧凑化技术的不断发展,其在处理高维、多模态数据方面的潜力将进一步释

放，持续推动复杂视觉任务的性能边界。双线性池化凭借其强大的特征交互建模能力，在多个视觉与多模态任务中展现出显著效果，典型应用包括：

细粒度图像分类：通过融合来自不同图像区域的局部特征，有效区分高度相似的子类别（如不同鸟种、车型）。

视觉问答：将图像特征与文本问题进行双线性交互，实现跨模态语义对齐与推理。

人员重识别：结合人体的全局外观与局部结构特征，增强身份判别能力。

三维物体识别：融合多视角或几何特征，提升对三维结构的表征能力。

在这些任务中，双线性池化通过显式建模特征间二阶交互，显著提升了模型在复杂场景下的判别性能。

尽管双线性池化的改进性能优异，原始双线性池化输出的特征维度为两输入特征维度的乘积，存在维度灾难问题，导致计算与存储开销巨大。为克服这一瓶颈，研究者提出多种紧凑化方法：利用核技巧或随机投影（如 Count Sketch＋FFT）将高维外积近似映射至低维空间，在保持判别力的同时大幅降低特征维度；多模态紧凑双线性池化：进一步扩展 CBP 至多模态场景，支持异构特征（如图像与文本）的高效融合。

这些改进在基本保留双线性交互能力的前提下，显著提升了方法的实用性与可扩展性。

4. 池化操作的局限性

尽管池化作用巨大，但它并非没有缺点，缺点如下。

信息丢失：池化是一种有损压缩。在保留最强特征的同时，不可避免地会丢弃大量细节信息，这对于需要精确定位的任务（如语义分割、实例分割）是不利的。

固定模式：传统的池化使用固定的窗口尺寸和步长，可能无法最优地适应所有类型的数据结构。

池化是经典 CNN 架构（如 LeNet，AlexNet，VGG）的基石，它以便捷的方式实现了平移不变性和维度压缩，为处理高维图像数据提供了关键支持。然而，在现代架构中，池化的角色正在演变。为了克服其信息丢失的缺点，研究者们采用了以下替代或补充方案：

使用步长大于 1 的卷积直接进行下采样；

使用空洞卷积在不降低分辨率的情况下扩大感受野。

在编码器—解码器结构中引入跳跃连接（如 U－Net），将编码器中池化前的细节特征直接传递给解码器，以弥补池化造成的信息损失。尽管如此，池化因其简单、高效和有效的特性，至今仍在许多模型中占据一席之地，是理解 CNN 工作原理不可或缺的一环。

3.4 激活函数

激活函数为网络引入非线性，是 DL 模型强大的根源，使神经网络能够逼近任意复杂函数，是模型具备强大表示能力的根本原因。在 CNN 中，激活函数直接决定了模型的非线性表达能力、特征抽象层次与梯度传播效率，进而影响收敛过程与再性能。激活函数在提升网络性能的同时也可能带来训练挑战，特别是在反向传播过程中可能出现的梯度消失或梯度爆炸问题。

1. 激活函数定义

神经网络中常用的激活函数主要有 ReLU、Sigmoid、Tanh、PReLU、ELU、Swish 等激活函数,这些激活函数的公式与图像如下所示。

$$\mathrm{ReLU}(x)=\begin{cases}x & x\geqslant 0\\ 0 & x<0\end{cases} \tag{3-17}$$

$$\mathrm{Sigmoid}(x)=\frac{1}{1+\mathrm{e}^{-x}} \tag{3-18}$$

$$\mathrm{Tanh}(x)=\frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}} \tag{3-19}$$

$$\mathrm{PReLU}(x)=\max(\alpha x,x) \tag{3-20}$$

$$\mathrm{ELU}(x)=\begin{cases}x & x\geqslant 0\\ \alpha(\mathrm{e}^{x}-1) & x<0\end{cases} \tag{3-21}$$

为系统对比 ReLU、Sigmoid、Tanh 与 Swish 四种常用函数的核心特性及其在域自适应任务中的适配性,表 3-4 从非线性程度、梯度特性、计算效率等多个维度进行系统梳理。

表 3-4 数学特性对比与任务适配总结

激活函数	输出范围	梯度特性	适用场景	域自适应价值
ReLU	$[0,+\infty)$	正区间梯度 1,负区间梯度 0	浅层特征提取、高计算效率需求	过滤域相关噪声,保留显著性特征
Sigmoid	$(0,1)$	两端梯度饱和,中部梯度最大	二分类输出层、概率判别任务	域判别器输出概率,支撑对抗训练
Tanh	$(-1,1)$	零中心化,中部梯度最大	中间层特征归一化、生成对抗网络	稳定特征分布,缓解域间偏移
Swish	$(-\infty,+\infty)$	平滑非单调,自适应梯度调节	复杂特征建模、跨域分布适配	动态对齐源域与目标域特征响应强度

Sigmoid 函数是一种经典的 S 型激活函数,能够将任意实数输入映射到$[0,1]$区间内。其特性表现为:输入值极大时输出趋近于 1,输入值极小时输出趋近于 0,浅层广泛用于二分类任务中概率输出的建模。

Tanh 函数能够将输入映射到$[-1,1]$区间,其输出以零为中心的特点有效改善 Sigmoid 函数均值偏移问题。该函数对输入的响应呈现对称饱和特性,输入绝对值较大时输出逐渐趋近于± 1,这一特性使其在需要平衡正负激活值的场景中表现优异。

ReLU 函数凭借其稀疏激活特性成为 DL 的标准配置,其分段线性设计从根本上解决梯度消失问题。该函数在正值区域保持线性传递,负值区域完全抑制的特点大幅提升计算效率。

Leaky ReLU 是针对传统 ReLU 激活函数中可能出现的“神经元死亡”问题提出的改进方案。其核心思想是在负值区间引入一个微小的固定斜率(通常设为 0.01),使得当输入为

负时仍能保持微弱的信号传递能力。这种设计使得神经元在负值区域的梯度不再完全归零，有助于维持神经网络训练过程中参数的持续更新，从而有效降低神经元永久失活的风险。

ELU激活函数通过引入指数修正机制构建平滑的负值响应区域，在保持ReLU抗梯度消失特性的基础上优化梯度传播效果。其数学表达式定义为分段函数，当输入为正时直接输出原值，负值时采用$\alpha(e^x-1)$的非线性变换。这种设计使得函数曲线在零点处保持连续可导，理论上既能避免ReLU的“死神经元”现象，又能通过负值响应促进激活值的零均值化，从而改善深层网络的训练动态。

Swish函数通过引入自门控机制实现激活函数的自适应调节，其平滑非单调特性有效增强深层网络的表达能力。该函数在输入为负值时仍保持小幅度激活，避免ReLU系列的“死亡”缺陷，表示：

$$f(x)=x\cdot\sigma(\beta x) \tag{3-22}$$

式中，σ是Sigmoid函数，β为可训练参数。β的引入增加超参数调优成本，需要配合复杂的初始化策略才能发挥最佳效果。Sigmoid分量的持续存在导致计算效率仍低于纯线性激活函数，在需要实时推理的场景中可能产生延迟。

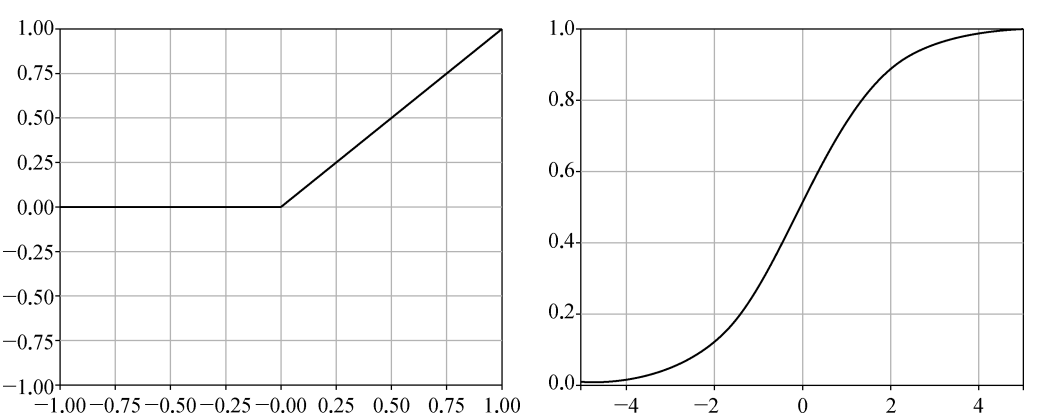
Leaky ReLU函数。为解决ReLU的神经元坏死问题，研究者提出了Leaky ReLU这一改进型激活函数。其数学表达式为公式(3-23)所示：

$$f(x)=\begin{cases}\alpha x & x\leqslant 0\\ x & x>0\end{cases} \tag{3-23}$$

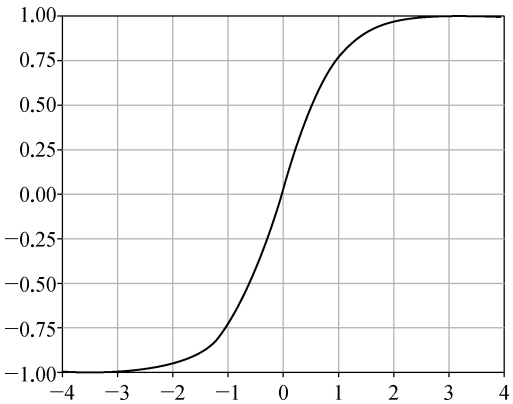
式中，α为一个较小的正常数(通常设为0.01)。

当输入为负时，函数输出为一个斜率很小的线性函数(αx)，而非像ReLU那样完全截断为0。这一改进确保了负值输入区域仍具有非零梯度，从而保证神经元参数能够持续更新，有效避免了“死亡神经元”现象。然而，这种改进也带来了相应的计算代价——由于需要在负半轴引入额外的线性计算，相比标准ReLU增加了计算复杂度与参数数量(需引入并调整α参数)。尽管如此，在许多实际应用中，Leaky ReLU在模型性能与训练稳定性方面带来的提升往往超过了其额外的计算开销。

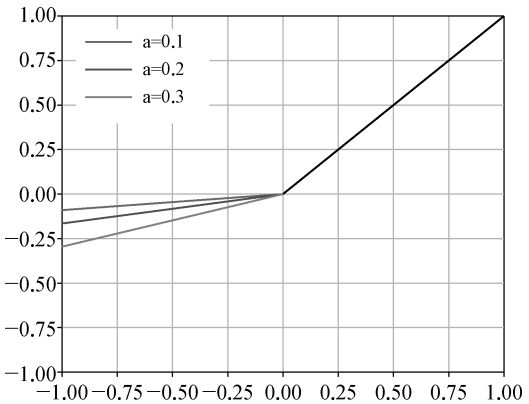
上述公式的函数图像如图3-13所示。



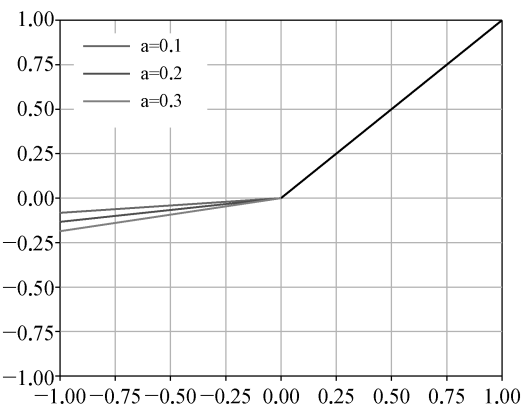
(a) ReLU函数图　　(b) Sigmoid函数图



(c) Tanh 函数图



(d) PReLU 函数图



(e) ELU 函数图

图 3-13 函数图像图

2. **激活函数机制**

在视觉任务中(如图像分类、目标检测与语义分割),激活函数具有以下关键作用:

引入非线性映射:打破网络的线性叠加约束,使其能够学习图像中目标的复杂形态与空间分布规律,如裂缝结构、物体边缘等。

实现特征筛选:通过阈值机制抑制非关键区域响应(如背景纹理),同时增强与任务强相关特征的显著性(如裂缝边缘、角点等)。

优化特征空间:在域自适应等任务中,通过非线性映射为不同域样本提供可分性更强的特征表示,支撑跨域特征对齐与模型迁移。

在 DL 领域中,激活函数的选择对模型性能具有决定性影响,尤其在域自适应等复杂任务中,其作用更为关键。不同激活函数凭借其独特的数学特性,通常在网络结构的不同部位承担专门职能,通过功能互补协同促进跨域知识的有效迁移。

3.5 注意力机制

注意力机制的核心思想源于人类认知系统处理信息的方式:面对复杂输入时,人类大脑

会选择性关注与当前任务最相关的信息，同时抑制无关干扰。DL中的注意力机制正是对这一生物过程的计算建模，它使模型能够根据上下文与任务目标，动态调整对不同输入部分的关注程度，从而提升对关键信息的提取能力。**在目标检测中的应用价值：**目标检测作为计算机视觉的基础任务，面临着检测场景多样、物体遮挡、背景干扰、尺度变化和光照影响等多重挑战。研究表明，引入注意力机制能够有效增强特征表示能力。具体表现在：

选择性特征提取，使得网络更倾向于提取与目标物体相关的关键特征；上下文信息融合，充分建立局部特征与全局上下文之间的联系；干扰抑制，通过权重分配抑制背景噪声的干扰影响。

1. 定义和数学描述

在技术实现上，注意力机制是一个基于查询的加权分配过程。其核心可概括为：根据“查询”的内容，在“信息库”中动态找出最重要的部分，并为其分配更高的权重。在预测功率中并非所有的数据特征值都与功率有关，因此，在预测过程中着重关注数据相关的特征温度和湿度，结合本次实验所用数据多特征的特点，在网络中添加注意力机制，通过区分不同特征的重要性来提高分类的准确性。设 s 是权重系数 α_i 和原隐层状态 h_i 乘积的累加和，e_i 为原隐层状态 h_i 的权重，v_i 和 ω_i 为权重矩阵，b_i^* 为偏差。注意力机制通常通过以下三个步骤实现：

(1) 相似度计算：计算一个查询向量与一组键向量的相似度。这里的“查询”可以理解为当前任务的目标或上下文，“键”则是输入信息中各个部分的标识：

$$e_i = v_i \tanh(\omega_i h_i + b_i) \tag{3-24}$$

(2) 权重归一化：将上一步得到的相似度分数通过 Softmax 等函数进行归一化，转化为一个概率分布，即注意力权重。这确保了所有权重之和为 1，且权重的大小直接反映其对应信息的重要性：

$$a_i = \exp(e_i) \Big/ \sum_{j=1}^{n} \exp(e_j) \tag{3-25}$$

(3) 加权求和：使用归一化后的注意力权重，对另一组值向量进行加权求和。这里的“值”为输入信息本身的实际内容。再得到的上下文向量即融合了注意力权重的关键信息汇总：

$$s = \sum_{i=1}^{n} a_i h_i \tag{3-26}$$

注意力机制通过“查询—键”交互计算出权重，再用此权重对“值”进行聚合，实现了信息筛选与聚焦。这一精巧的设计使其成为提升模型判别力和可解释性的关键技术。核心思想可概括为：**对重要信息分配更多关注，即赋予较大权重**。

2. 三类注意力机制

根据作用域的不同，注意力机制主要分为通道域、空间域和通道空间混合域三种注意力机制：

(1) 通道域注意力。通道域注意力是 DL 中的一种重要机制,其核心在于对特征通道间的关系进行建模,通过评估各通道的重要性,自适应地重新校准通道维度上的特征响应。该机制使网络能够聚焦于信息量更丰富的特征通道,同时抑制次要或冗余的通道特征。

在各类通道注意力方法中,SE(Squeeze-and-Excitation)模块是最具代表性的实现之一。SE 机制完整体现了通道注意力的核心思想,即通过建模通道间的差异化,使网络能够自主学习各通道特征的重要性权重。如图 3-14 所示,SE 模块的执行流程包含三个关键步骤:首先通过全局平均池化聚合特征图的全局空间信息,利用全连接层学习通道间的非线性依赖关系,再通过 Sigmoid 激活函数生成通道注意力权重,对原始特征进行重标定。

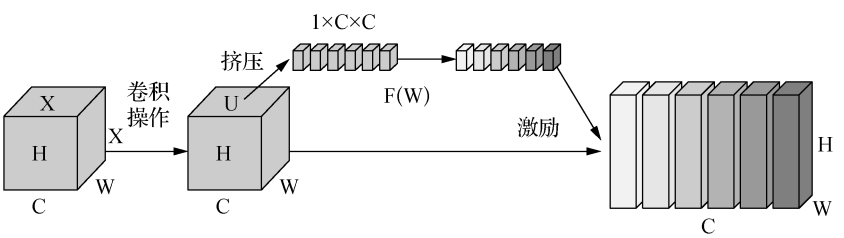


图 3-14 SE 注意力机制结构图

Squeeze(压缩):通过全局平均池化操作,将特征图在空间维度上进行压缩,聚合为每个通道的全局描述符。对输入特征图 $u \in R^{H\times W\times C}$ 沿空间维度 $H\times W$ 进行全局平均池化,将每张特征图压缩为一个通道描述符,得到向量 $z \in R^C$,其中每个元素 Z_C 承载了对应通道的全局空间信息。

$$Z_C = \frac{1}{HXW}\sum_{i=1}^{H}\sum_{j=1}^{W} u_c(i,j) \tag{3-27}$$

Excitation(激励):利用全连接层构成的瓶颈结构,学习通道间的非线性依赖关系,生成表示各通道重要性的权重。将 z 输入由两个全连接层构成的门控机制,通过非线性变换学习各通道的权重:

$$s = \sigma(W_2\delta(W_1 z)) \tag{3-28}$$

式中,δ 为 ReLU 激活函数,σ 为 Sigmoid 激活函数,W_1 和 W_2 为两个全连接层的权重,r 为压缩比,得到 $s \in R^C$ 表示各通道的重要性权重。

Scale(重标定):将学习得到的通道权重通过 Sigmoid 激活函数归一化后,与原始特征图进行逐通道相乘,完成特征重标定。将权重向量 s 与原始特征图 u 逐通道相乘,完成特征自适应加权:

$$X_c = s_c u_c \tag{3-29}$$

该操作强化重要特征通道的响应,抑制次要通道,使网络朝着有利于目标任务的方向更新特征表示。

SE 注意力机制因其有效性和通用性,被广泛应用于各类视觉任务中,其提出的模型也在 ImageNet 2017 图像分类竞赛中取得冠军,显著推动了注意力机制在计算机视觉领域的

发展。

在 SE 模块中，通道间关系的建模完全依赖于全连接层的隐式学习，这种以块结构形式深度嵌入网络的设计会带来显著的参数与计算开销。为解决 SE 注意力机制在通道关系学习中存在的计算效率问题，Yang 等人提出了 GCT(Gated Channel Transformation)注意力机制。如图 3－15 所示，GCT 通过引入轻量级的通道归一化层来重构通道注意力结构。该设计的核心创新在于使用基于 ℓ_2 归一化的简洁操作，取代了 SE 模块中复杂的全连接层。这种归一化层能够显式地调控通道间的响应竞争与依赖关系，在基本保持模型性能的前提下，显著降低了参数数量与计算复杂度。这一改进为注意力机制在资源受限环境下的实际应用提供了更高效的解决方案。

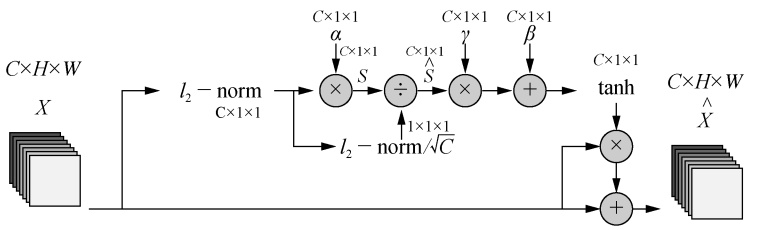


图 3－15　GCT 注意力机制结构图

(2) 空间域注意力。在空间维度上建立注意力图，旨在识别特征图中不同空间位置的重要性差异，从而实现对关键区域的精准聚焦。

SE 模块的提出有效解决了卷积池化过程中因忽略特征图通道间重要性差异而导致的信息损失问题。然而，该设计仅聚焦于通道维度，未能充分探索空间位置信息的重要性。为弥补这一局限，Hu 等人在 SE 的基础上提出了改进模型 GE，其结构如图 3－16 所示。GE 模块引入了 Gather 与 Excite 两个核心操作符：Gather 操作负责从大范围空间区域内聚合上下文特征信息，Excite 操作则将聚合得到的全局上下文信息重新分配到局部特征位置。通过这一机制，GE 能够自适应地调整不同空间位置的特征响应强度，在实现空间注意力功能的同时，其参数量较 SE 模块更为轻量化，体现了在注意力机制设计上对效率与性能的综合考量。

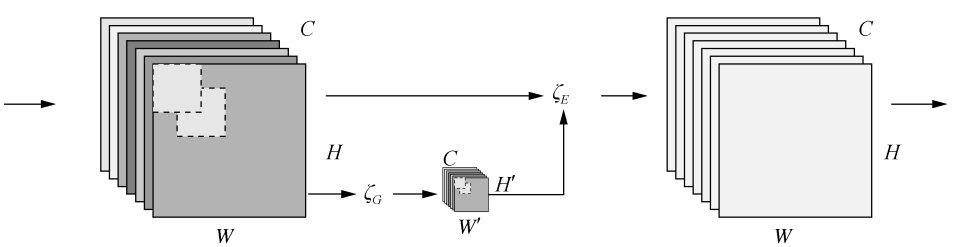


图 3－16　GE 注意力机制结构图

受卷积操作固有特性的限制，CNN 中的信息流动通常被约束在局部邻域范围内，这在一定程度上削弱了模型对复杂场景的整体理解能力。为突破这一局限，提出一种点状空间注意力网络(PSANet)，其结构如图 3－17 所示。PSANet 的核心设计目标在于构建全局信息交互机制，以缓解传统卷积的局部性约束。具体而言，该网络通过在特征图的每个位置处

自适应地学习一个注意力掩码,从而建立该位置与图像中所有其他位置之间的关联,实现全局上下文信息的动态聚合。

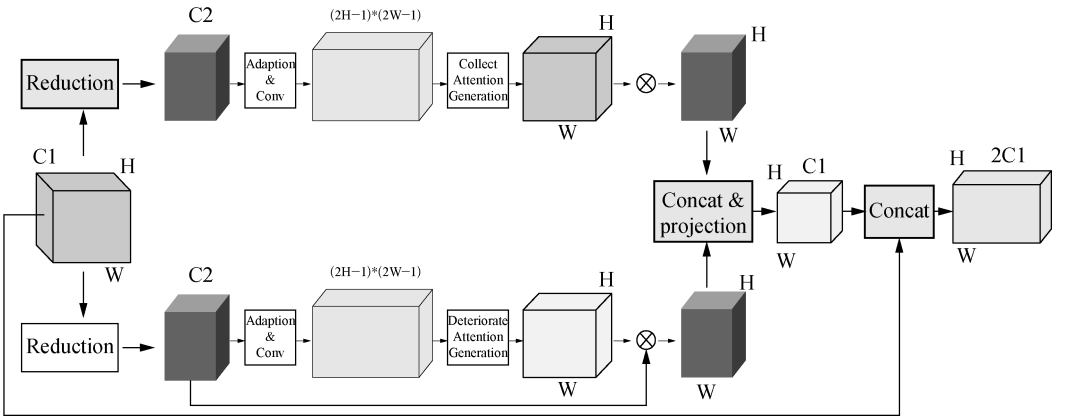
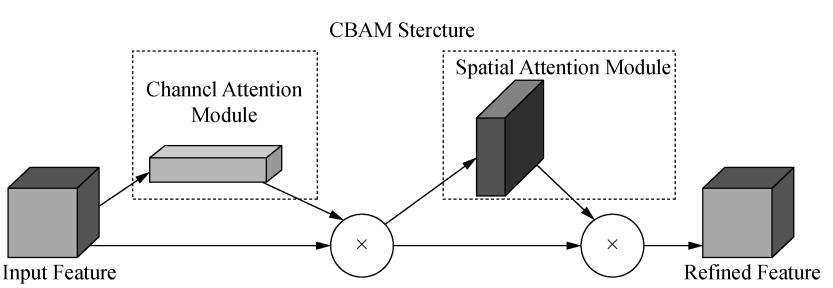


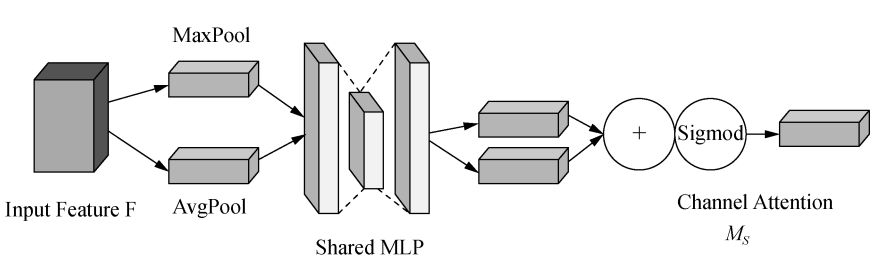
图 3-17　PSANet 注意力网络结构图

此外,PSANet 进一步引入了双向信息传播机制,支持信息在场景中的双向流动:一方面,收集来自其他位置的信息有助于提升对当前位置的语义理解;另一方面,分发当前位置的信息也有助于其他位置的预测。该设计在场景解析等任务中有效增强了模型对全局结构和上下文依赖的建模能力。

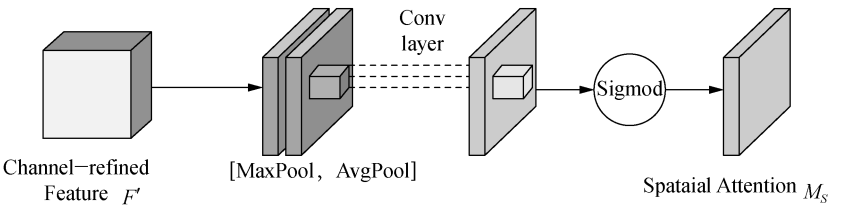
(3) 混合域注意力。结合通道与空间维度的注意力机制,实现更精细的特征重标定,在保持计算效率的同时获得更优的性能表现。为在保持计算效率的同时实现更精细的特征重标定,提出了一种轻量级的卷积注意力模块——CBAM,其结构如图 3-18 所示。CBAM 的核心思想是串联使用通道与空间两个维度的注意力机制,使网络能够自主判断应重点关注哪些通道以及哪些空间区域。



(a) CBAM



(b) 通道注意力模块



(c) 空间注意模块

图 3-18 CBAM 注意力机制

CBAM 依次对输入特征施加通道注意力和空间注意力，具体过程如下：

通道注意力子模块，如图 3-18(b)所示，同时使用全局最大池化和全局平均池化对特征进行压缩，得到两种不同的全局上下文描述。将两种池化结果分别送入一个共享的多层感知机(MLP)，该 MLP 结构为“卷积→ReLU→卷积”。将 MLP 的两路输出进行元素加和，再通过 Sigmoid 激活函数生成通道注意力权重。

空间注意力子模块，如图 3-18(b)所示，将经过通道重标定的特征作为输入，在通道维度上同时进行全局最大池化和全局平均池化，并将结果拼接。使用一个大尺度卷积核(如 7×7)对拼接后的特征进行卷积，以捕获大感受野的空间上下文信息。最后通过 Sigmoid 激活函数生成空间注意力权重。

计算空间注意力时，首先对每个特征点在通道维度上应用平均池化和最大池化，通过这两种池化方式分别聚合输入特征层的通道信息，生成两个二维特征图，一个代表平均池化特性，另一个代表最大池化特性。接着，将这两个特征图堆叠在一起，经过通道数为 1 的卷积层处理，再通过 Sigmoid 激活函数得到二维空间注意力图，从而为输入特征图的每个位置赋予一个权重值。最后，将这些权重逐通道地应用到输入特征层上，完成空间注意力的加权操作，具体计算过程如式(3-30)所示。

$$M_S(F)=\sigma(f^{7\times7}([\text{AvgPool}(F);\text{MaxPooL}(F)])) \tag{3-30}$$

式中，σ 表示 Sigmoid 函数，$f^{7\times7}$ 表示 7×7 大小的卷积核。

核心优势：通过显式建模特征的通道关系和空间位置重要性，有效增强对关键区域的特征响应。

设计轻量级，可无缝集成到任何 CNN 架构中，仅带来可忽略不计的计算开销。

潜在局限：其空间注意力子模块依赖固定尺度的卷积核来生成注意力图，可能因感受野受限而影响对复杂空间上下文的建模能力，为后续改进提供了方向。

3. 自注意力

自注意力机制(Self-Attention)最初由 Vaswani 等人提出于 Transformer 框架中，其核心思想在于通过显式建模序列中任意两个位置之间的相关性，自适应地捕获全局上下文信息，从而克服 CNN 在局部感受野受限以及循环神经网络在长程依赖建模中存在梯度消失与计算瓶颈的问题。设输入特征序列为 $\boldsymbol{X}\in\mathbf{R}^{n\times d}$，其维度为 n 个位置、每个位置 d 维特征。通过三个独立的线性映射将输入投影到查询(Query)、键(Key)和值(Value)：

$$Q=\boldsymbol{X}\boldsymbol{W}_Q,\quad K=\boldsymbol{X}\boldsymbol{W}_K,\quad V=\boldsymbol{X}\boldsymbol{W}_V \tag{3-31}$$

式中,$\boldsymbol{W}_Q,\boldsymbol{W}_K,\boldsymbol{W}_V \in \mathbf{R}^{d\times d_k}$ 为可学习参数矩阵。

Query 矩阵 $\boldsymbol{W}_Q$ 表示当前的关注点或信息需求,用于与 Key 矩阵进行匹配;Key 矩阵 $\boldsymbol{W}_K$ 包含输入序列中各个位置的标识信息,用于被 Query 矩阵查询匹配;Value 矩阵 $\boldsymbol{W}_V$ 存储与 Key 矩阵相对应的实际值或信息内容,当 Query 与某个 Key 匹配时,相应的 Value 将被用来计算输出。自注意力通过缩放点积计算相似度矩阵,并利用 Softmax 归一化得到权重分布:

$$\text{Attention}(Q,K,V)=\text{Softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V \tag{3-32}$$

式中,$\sqrt{d_k}$ 为缩放因子,用于缓解向量内积随维度增大而导致的数值不稳定问题。这一机制可分为三个阶段:首先是相似度计算,即通过查询与键之间的点积来衡量不同位置的相关性;其次是归一化处理,通过 Softmax 将注意力分布转换为概率形式,从而保证特征聚合具有全局平衡性;最后是加权聚合阶段,值向量根据权重分布进行加权求和,得到融合全局上下文信息的表示。相比传统卷积仅在局部范围捕获特征,自注意力能够在单步计算中实现全局建模,大幅提升了特征表达的能力。

4. 多头自注意力

多头自注意力机制是对自注意力的重要扩展。该机制通过将输入向量分割为多个“头”,使模型能够在不同的表示子空间中并行地计算注意力分布;每个头独立学习一组注意力权重,专注于输入序列中不同类型和范围的依赖关系;所有头的输出被拼接并通过线性投影融合,从而整合多种关系模式。这种设计显著提升了模型的特征表达能力,使其能够同时捕捉输入数据中不同方面的关联信息。以高分辨率遥感图像分析为例,同一场景中可能同时包含河流、植被、建筑等多种地物目标,这些目标间的语义关联往往跨越较大空间范围。传统卷积操作受限于局部感受野,难以有效建模这种长程依赖;而多头自注意力则通过其全局交互机制,直接建立跨区域的关联建模,极大增强了模型在复杂场景下的识别能力与鲁棒性。多头自注意力机制通过并行化的注意力计算方式,显著提升了模型对复杂依赖关系的建模能力。其完整计算流程可归纳为以下五个核心步骤:

(1) 输入线性变换。对输入的 Query、Key 和 Value 向量分别进行线性变换,将其映射到不同的表示子空间。这一步骤通过可学习的权重矩阵实现维度转换,为后续的多头分割奠定基础。

(2) 分割多头。将线性变换后的 Query、Key 和 Value 向量沿特征维度均匀分割为多个头(head)。每个头将独立进行注意力计算,专注于学习不同方面的特征关系。

(3) 缩放点积注意力。在每个头内部,使用缩放点积注意力来计算 Query 和 Key 之间的注意力分数。这个分数决定了在生成输出时,模型应该关注 Value 向量的部分。在每个头内部执行标准的缩放点积注意力操作,具体包含:

相似度计算:通过 Query 与 Key 的点积衡量特征间关联强度;

数值缩放:将点积结果除以特征维度的平方根,确保梯度稳定性;

权重归一化:应用 Softmax 函数将相似度转换为概率分布的注意力权重;

信息聚合:使用注意力权重对 Value 向量进行加权求和。

缩放点积注意力是Transformer模型中的多头注意力机制的一个关键组成部分。

(4) 多头拼接。将所有头的输出向量沿特征维度拼接,形成包含多种注意力模式的综合表征。将计算出的注意力权重应用于Value向量,得到加权的中间输出。这个过程可以理解为根据注意力权重对输入信息进行筛选和聚焦。

(5) 线性变换。对拼接后的向量进行线性变换,融合不同头的学习结果,得到多头注意力输出。

由于点积操作的结果可能非常大,尤其是在输入维度较高的情况下,这可能导致Softmax函数在计算注意力权重时进入饱和区。为了避免这个问题,缩放点积注意力引入了一个缩放因子,通常是输入维度的平方根。点积结果除以这个缩放因子,可以使得Softmax函数的输入保持在一个合理的范围内。

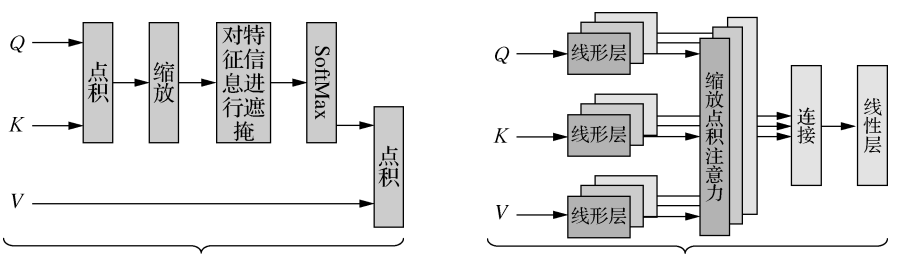


图 3-19 缩放点积注意力和多头自注意力结构图

5. 并行双注意力网络(DANet)

DANet是一种集成通道与空间维度的混合域注意力机制。与CBAM的串行结构不同,DANet采用并行双通路设计,分别从通道和空间两个角度建模特征依赖关系,其整体框架如图3-20所示。

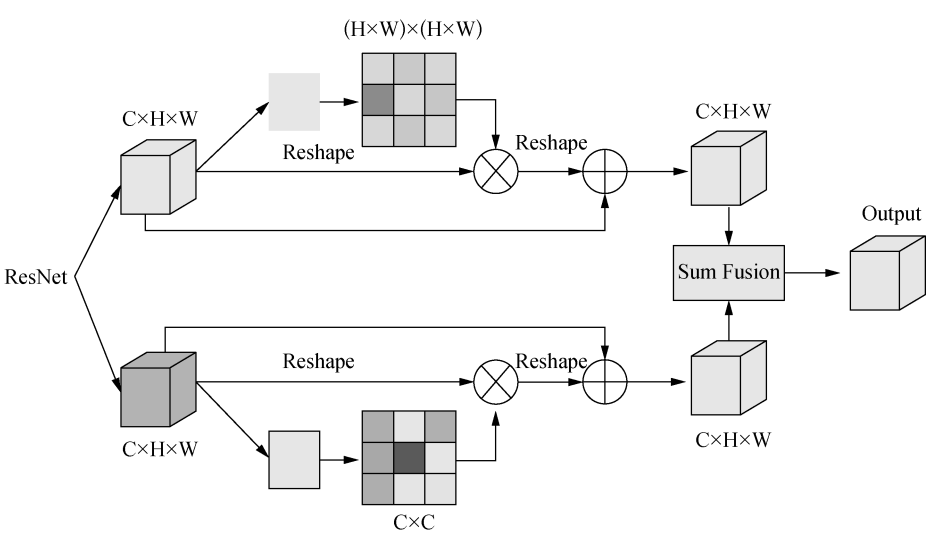


图 3-20 DANet注意力机制结构图

DANet 的核心在于利用自注意力机制构建两种特征关联矩阵。空间自注意力矩阵 $\boldsymbol{S}$:捕捉特征图中不同空间位置之间长程依赖关系;通道自注意力矩阵 $\boldsymbol{X}$:建模不同特征通道之间的相互依赖。

通过这两个并行生成的注意力矩阵,DANet 能够显式地引导特征更新过程,将分布在不同位置和通道上的语义特征进行有效关联。这种设计显著增强了模型对图像整体上下文的理解能力,从而在语义分割等需要精细场景解析的任务中提升了检测精度与语义一致性。

6. Coordinate 注意力机制

Coordinate 注意力机制是 CVPR 2021 提出的一种新颖的注意力机制,它通过将位置信息嵌入到通道注意力中,在轻量级网络设计中表现出色。如图 3-21 所示为其算法框架图。

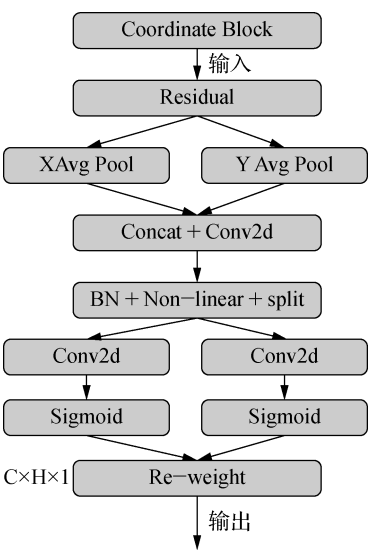


图 3-21 Coordinate 注意力结构图

图 3-21 包括 Coordinate 注意力的两个关键阶段:

(1) Coordinate 信息嵌入:传统通道注意力(如 SE)使用二维全局池化将空间信息压缩成一个通道描述符,这会导致位置信息丢失。Coordinate 注意力将其分解为两个一维编码过程:

水平全局平均池化:使用池化核$(H,1)$,生成尺寸为 $C\times H\times 1$ 的特征图,捕获垂直方向上的长程依赖,同时保留水平方向上的精确位置信息。

垂直全局平均池化:使用池化核$(1,W)$,生成尺寸为 $C\times 1\times W$ 的特征图,捕获水平方向上的长程依赖,同时保留垂直方向上的精确位置信息。

(2) Coordinate 注意力生成。信息融合与变换:将上述两个方向的特征图拼接后,通过共享的 1×1 卷积(F1)进行特征融合和降维。

注意力权重计算:将融合后的特征沿空间维度拆分为独立的水平分量和垂直分量,分别通过 1×1 卷积(F_h 和 F_w)和 Sigmoid 激活函数,生成方向感知和位置敏感的注意力图。

特征重标定:将两个方向的注意力图与输入特征图逐元素相乘,实现对重要特征的增强。

Coordinate注意力的核心思想在于，通过分解池化操作并分别编码水平与垂直方向的空间信息，将位置信息有效地嵌入到通道注意力机制中。这使得它能够同时捕获长程依赖关系和保留精确的位置信息。其优势主要体现在：

增强特征表示：通过引入位置信息，生成的注意力图能更准确地聚焦于关键区域，例如在图像分类中帮助网络更好地识别目标物体。

精准定位：保留的精确位置信息使其在目标检测、语义分割等需要位置信息的任务中表现出色。

高效轻量：整个机制主要由池化层和1×1卷积构成，计算开销小，易于集成到各种轻量级网络中（如MobileNetV2，EfficientNet）。

7. ECA-Net（高效通道注意力）

ECA-Net是一种高效的轻量级通道注意力机制，其核心创新在于使用一维卷积替代传统全连接层，以极低计算成本实现跨通道交互建模。该设计有效解决了SE-Net中因全连接层导致的参数冗余与降维损伤问题。具体而言，ECA-Net摒弃了SE模块中的降维—升维结构，直接对全局平均池化后的特征向量施加一个固定大小的一维卷积核（k）。该卷积核定义了局部跨通道交互的范围，使每个通道的权重能够由其k个邻近通道共同参与计算，既保留了通道间的依赖关系，又避免了全局全连接带来的高昂计算开销。通过这一结构优化，ECA-Net在保持与SE-Net相当甚至更优性能的同时，显著降低了模型的复杂度与参数量，为注意力机制在轻量级网络中的实用化提供了高效解决方案。图3-22为ECA网络的结构图。由以下几个部分组成：

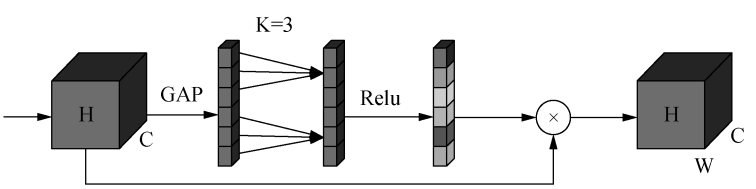


图3-22 ECA网络的结构图

全局平均池化（GAP）：将输入特征图压缩到1×1的大小，得到通道描述。与SENet相同，首先将输入特征图在空间维度（H×W）上进行压缩，为每个通道生成一个全局统计量z_c，形成通道特征向量z。

自适应一维卷积：使用kernel_size为3的一维卷积核，在通道维度上进行运算。padding设置为（kernel_size−1）/2，确保卷积后的输出与输入尺寸相匹配。卷积核大小k自适应确定：k通过与通道总数C关联，$k=\psi(C)$，确保交互范围随网络深度和通道数的变化而自动调整，无需手动调参。

Sigmoid激活函数：将卷积结果映射到0—1之间作为注意力权重。

特征重加权：将计算得到的注意力权重作用于原特征图，从而调整各通道的重要性，完成对重要特征的增强与对次要特征的抑制。

再的权重是综合了各个相邻通道的信息获得。

$$\omega_i=\sigma\left(\sum_{j=1}^{k}\omega_i^j y_i^j\right),y_i^i\in\Omega_i^k \tag{3-33}$$

式中,σ 为激活函数,y^i 代表通道,ω_i 为通道 y_i 的权重,Ω 代表与 y^i 相邻的 k 个通道,k 的值是随着学习自适应变化的。

为了实现这一想法,利用一维卷积层来进行实现,通过核为 k 的一维卷积对通道与其相邻的 $k-1$ 个通道信息进行综合:

$$\omega=\sigma(C1D_k(y)) \tag{3-34}$$

式中,$C1D_k$ 表示核为 k 的一维卷积操作,y 表示通道。

该操作使每个通道的再权重 ω_i 由其自身及其 $k-1$ 个相邻通道共同参与决定,从而实现局部跨通道交互。卷积核大小 k 可根据通道维度 C 自适应确定,进一步提升了模块的灵活性与泛化能力。ECA-Net 与 SE-Net 的对比优势如表 3-5 所示。

表 3-5 ECA-Net 与 SE-Net 的对比

特性	SE-Net	ECA-Net
交互机制	使用两个全连接层进行全通道交互	使用一维卷积进行局部通道交互
参数量	较高,约为 $2*C^2/r$(r 为降维比)	极低,仅为 $k*C$(k 为卷积核大小)
保持性能	降维操作可能损害特征表示	避免降维,直接学习权重,性能通常更优
理念	“压缩—激励”	“高效交互”

ECA-Net 的成功在于它揭示了一个关键洞察:**捕获跨通道交互时,不需要复杂的全连接降维,局部交互已经足够,并且更高效**。它通过一个极其轻量的一维卷积层,实现了性能、速度和参数效率的完美平衡,成为轻量级网络设计中通道注意力的首选方案之一。ECA 的优点如下:

轻量高效:仅增加少量参数(主要由一维卷积核带来),计算开销极低。

避免降维:直接处理通道向量,避免了 SENet 中降维操作对通道间依赖关系的损害。

自适应交互:通过自适应卷积核大小,动态调整交互范围,比固定的全连接层更灵活。

8. 图注意力机制(GAT)

GAT 是注意力机制与空域图神经网络结合的典范。其核心创新在于通过注意力机制动态学习中心节点与其各邻居节点之间的关联强度(注意力权重),而非依赖图结构中预定义的静态权重,从而能更灵活地捕捉节点间复杂的非线性关系。GAT 层对图中每个节点的计算可分解为以下步骤:

线性变换:通过一个共享的权重矩阵 W,将所有节点的特征向量(如 miRNA 或疾病节点特征)映射到一个更高维或更具表达力的隐空间,

$$h'_i=Wh_i,\quad h'_j=Wh_j \tag{3-35}$$

式中,h_i,h_j 表示节点特征,W 表示可学习的参数矩阵。

计算注意力得分:对于中心节点 i 及其邻居节点 $j\in N(i)$,计算它们之间的原始注意力

得分 e_{ij}。通常通过一个加性模型或点积模型实现，并由一个可学习的注意力向量 a 参数化，

$$e_{ij}=a([Wh_i \| Wh_j], j \in N_i) \tag{3-36}$$

式中，e_{ij} 表示两个节点之间的系数，$a(\cdot)$ 表示映射函数，$[\cdot \| \cdot]$ 表示矩阵拼接操作。

归一化注意力权重：使用 Softmax 函数对中心节点所有邻居的原始注意力得分进行归一化，得到标准化的注意力系数 α_{ij}，如下所示：

$$\alpha_{ij}=\mathrm{Softmax}(e_{ij})=\frac{\exp(\mathrm{ReLUe}_{ij})}{\sum_{j\in N_i}\exp(\mathrm{ReLUe}_{ij})} \tag{3-37}$$

加权聚合：将归一化后的注意力权重作为系数，对变换后的邻居节点特征进行加权求和，并通过一个非线性激活函数（如 ELU 或 ReLU），得到中心节点 i 的相关系数，进行归一化：

$$h'_i=\sigma\left(\sum_{j\in N_i}\alpha_{ij}Wh_j\right) \tag{3-38}$$

式中，h'_i 为融合领域信息的节点新特征，$\sigma(\cdot)$ 表示激活函数。

GAT 通过将注意力机制引入图节点特征的聚合过程，实现了对图结构数据更精细、更自适应的建模，已成为图神经网络领域的一个基础且强大的构建模块。核心优势为：

强大而灵活：注意力权重的动态计算使模型能为不同的邻居分配合适的重要性，增强了模型的表达能力。

计算高效：每个节点的操作可以并行 across 其邻居，适用于大规模图数据。

归纳式学习：不依赖整个图结构，易于泛化到未见过的节点或图。

综上所述，注意力机制能够通过特征关联更有效地学习区别性特征，进一步提高目标检测精度，但上述目标检测网络中通道和空间特征 $C\times W\times H$ 的注意力机制通常都是对所有 C 个通道、整个空间 $W\times H$ 进行信息压缩后，实现注意力权重计算和分配。这种粗粒度全局操作方式忽略了不同通道之间，以及不同空间区域上的差异度，导致其注意力权重计算误差较大。此外，如何在特征空间内挖掘出更加有效的上下文信息对于目标检测任务也是一项挑战。

3.6 DL中其他概念和操作

3.6.1 全连接

全连接层通常位于 CNN 的末端，承担着从抽象特征到再决策的映射任务。全连接层的主要作用是将前端通过多轮“卷积-池化”操作所提取的分布式特征进行全局整合与高层推理。它将每个位置的特征综合起来，形成对整个输入的全局性理解，并再将其映射到样本的标签空间，例如在图像分类任务中输出每个类别的概率分布。

1. **概念**

“全连接”是一种网络结构的设计模式或连接方式。它描述的是两层网络之间的一种

极端情况:前一层的每一个神经元,都与后一层的每一个神经元相连接。可以把它想象成一个稠密的网状结构,信息在这个网络里可以任意流动,没有任何限制。它描述一种“关系”,而不是一个具体的“层”。“全连接层”具体实现“全连接”这种模式的网络层。它是DL模型中的一个具体组件。“添加一个全连接层”时,是在模型中插入一个这种全连接结构层。

全连接通常需要将前端输出的三维特征图展平为一维向量,以作为全连接层的输入。该结构通过权重矩阵的线性变换与激活函数的非线性组合,完成复杂的高层特征变换,其典型结构如图3-23所示。

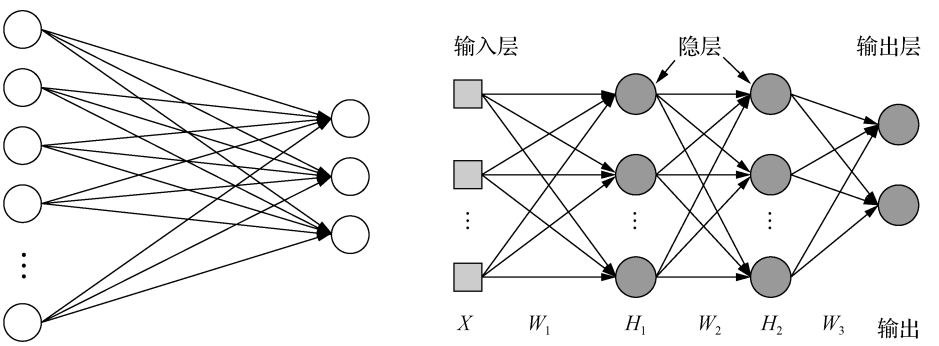


图3-23 全连接网络结构

在全连接层中,每一个神经元均与前一层的所有神经元相连接,实现全局感知。经过线性变换后,通常会再经过一个激活函数(如ReLU,Softmax)来引入非线性,得到再的输出。其计算过程可表述为:

$$\boldsymbol{y}=\sigma(\boldsymbol{w}\times\boldsymbol{x}+\boldsymbol{b}) \tag{3-39}$$

式中,$\boldsymbol{y}$ 为输出向量,σ 为激活函数,$\boldsymbol{w}$ 为权重矩阵,$\boldsymbol{x}$ 为输入向量,偏置向量用 $\boldsymbol{b}$ 表示。

图3-23中的输出为:

$$\mathrm{Out}=H_2*W_3=H_1*W_2*W_3=X*(W_1*W_2*W_3)=X*W \tag{3-40}$$

2. 全连接的特点

巨大的参数量:由于是“全连接”,参数数量是输入维度×输出维度。这使得全连接层通常是整个网络中参数最多、最耗计算资源的部分。

全局感受野:每个输出神经元都“看到”了全部输入信息,适合做全局信息的整合与决策。

失去空间结构:在将图像特征图输入全连接层之前,需要将其展平为一维向量,这完全破坏了特征图原有的空间结构信息。

3. 在CNN中的主要作用

在经典的CNN中,全连接层通常位于网络的末端,扮演着“分类器”的角色。

特征整合与分类:网络前端的卷积层和池化层负责从图像中提取局部、平移不变的特征。这些特征被展平后送入全连接层。全连接层的作用是综合所有这些分布式的特征信

息，并学习它们之间的复杂组合关系，再映射到样本的标签空间（即输出每个类别的分数）。

举例：在一个猫狗分类器中，前面的卷积层可能检测到了“胡须”“耳朵形状”“毛发纹理”等特征。全连接层则负责判断：若“胡须”特征很强且“尖耳朵”特征也很强，则它更可能是猫；若“吐舌头”特征很强且“圆耳朵”特征很强，则它更可能是狗。

4. 演进与替代方案

由于全连接层参数巨大的缺点，现代网络架构正在逐渐减少对其的依赖，或寻找替代方案。

全局平均池化：这是目前最流行的替代方案。在最后的卷积层后，不再使用全连接层，而是直接对每个特征图求平均值，将这个平均值作为该类别的置信度。

优势：极大地减少了参数数量，有效防止过拟合，并且保留了特征图与类别之间的对应关系，使模型更具解释性。

应用：在 ResNet、GoogLeNet 等网络中广泛使用。

全卷积网络：整个网络由卷积层组成，没有全连接层。特别适用于图像分割、目标检测等需要输出空间信息的任务。

3.6.2 感受野

在 CNN 中，感受野决定了特征图上每个神经元能够感知的原始输入图像的范围大小。这一概念揭示了 CNN 层次化处理图像的核心机制：浅层神经元凭借较小的感受野专注于提取边缘、角点等局部细节特征，而随着网络深度递增，感受野呈层级式扩展，使得深层神经元能够整合更广阔的上下文信息，形成具有丰富语义的全局特征表征。这种由局部细节到全局语义的递进式感知架构，不仅构成了 CNN 理解视觉内容的基础，也为迁移学习提供了重要的理论支撑——实践中通过冻结已具备通用特征提取能力的浅层网络参数，既显著提升了模型训练效率，又有效增强了模型在跨域任务中的泛化性能。

在深度 CNN 中，每个神经元节点都对应着输入图像的某个确定区域，仅该区域的图像内容能对相应神经元的激活产生影响，则这个区域称之为该神经元的感受野性质，越靠近感受野中心的区域越重要，各向同性由中心向周围的重要性衰减速度可以通过网络结构控制。从输入特征图到输出特征图尺寸计算：

$$n_{\text{out}}=\frac{n_{\text{in}}+2p-f}{s}+1 \tag{3-41}$$

式中，n_{in} 为输入 size，p 为 Padding 大小，f 为卷积核 size，s 为卷积步长。

感受野的计算与 Padding 无关，当前层的步长会影响下一层的感受野，而不影响当前层输出的感受野。感受野的计算是一个自底向上、逐层迭代的过程。迭代计算公式为：

$$RF_{l+1}=RF_l+(K_{l+1}-1)\prod_{i=1}^{k}S_i \tag{3-42}$$

式中，RF_{l+1} 为当前特征图对应的感受野大小，也为人们要计算的目标感受野，RF_l 为上一层特征图对应的感受野大小，K_{l+1} 为当前卷积层卷积核大小，最后一项连乘项则表示之前

卷积层的步长乘积。RF_0 在输入图像上，一个像素点的感受野为自己（1×1）。$\prod_{i=1}^{k}$ 表示从第1层到第 $l-1$ 层所有步长的累乘，表示前面所有层的步长对当前层感受野的“放大”作用。这是为什么“想让网络更快达到某个感受野，可以让步长大于1的卷积核更靠前”的原因；K_l 表示当前层卷积核在原始图像上能覆盖的额外范围；Padding 虽然不直接出现在公式中，但它通过影响特征图的大小，保证了卷积核能够有效地在特征图上滑动，从而使得 S_l 和 k_l 能够正确地发挥作用。

感受野需要一层一层地计算，后一层的感受野依赖于前一层的感受野。前面所有层的步长会累乘起来，共同决定当前层感受野的扩张速度。这是优化网络结构（如快速扩大感受野）的关键。Padding 不直接参与感受野的计算公式，但它通过维持特征图尺寸，确保了卷积核和步长能够按预期工作。没有 Padding，特征图会迅速缩小，从而限制感受野的有效增长。当前层的步长 s_l 不会影响当前层输出特征图的感受野 RF_l，但它会直接影响下一层的感受野计算，因为它会被计入累乘因子中。

3.6.3 权值共享

权值共享是 CNN 的核心设计原则，指同一卷积核在输入特征图的所有空间位置上重复使用。这不仅大幅减少了模型参数量，更使网络具备了关键的**平移不变性**。其设计基于一个合理的视觉假设：有意义的局部特征（如边缘、纹理）无论在图像的哪个位置出现，都应被同等检测到。因此，用于识别某种特征的卷积核，应当能够滑动扫描整个图像，而无需为每个位置学习不同的参数。

从本质上看，权值共享可理解为一种参数化的模板匹配过程：

卷积核作为特征检测器：每个卷积核是一个小的权重矩阵，用于匹配特定的局部模式；

滑动窗口检测机制：该卷积核作为固定模板，以滑动窗口方式遍历输入特征图的每个位置；

参数复用：在整个扫描过程中，卷积核权重保持不变，仅通过计算其与局部输入的点积，生成输出特征图上对应位置的响应值。

权值共享为 CNN 带来了显著优势：

极高的参数效率：例如，处理三通道 RGB 图像时，一个 5×5 卷积核仅需 5×5×3＝75 个参数。若无权值共享，对一张 100×100 的图像，仅一个全连接层就可能需要数百万参数。

平移不变性：相同特征在任何位置都由同一卷积核检测，使网络更关注特征内容而非其精确位置，从而对平移表现出鲁棒性。

归纳偏置与泛化能力：该机制将“局部性”与“平移不变性”的先验知识编码到网络结构中，使模型更易于训练，并具有更好的泛化性能。

权值共享不仅是降低计算复杂度的关键技术，更是 CNN 能高效处理图像等网格数据并取得卓越性能的架构基础。它充分体现了在 DL 模型中融入领域知识（归纳偏置）所带来的巨大效益，使网络在保持强大表征能力的同时，兼具高效性与实用性。

3.6.4 分辨率、网络深度和宽度

在 DL 中，分辨率、网络深度和宽度是决定模型性能、计算成本和内存消耗的三个最核

心的结构性维度。它们之间存在着深刻的相互制约与权衡关系。

(1) 分辨率。在CNN中,输入图像分辨率的设定与模型的下采样策略密切相关。通常情况下,输入尺寸需根据模型下采样次数 n 和再特征图的预期尺寸 $k\times k$ 共同决定,即输入分辨率与再特征图尺寸满足 $r/2^n=k$ 的关系。

网络的前向传播过程本质上是信息逐层抽象的过程:浅层网络主要保留几何结构和细节特征,而深层网络则提取更具语义意义的高级特征。再特征图尺寸 $k\times k$ 的选择需要在多方面因素间取得平衡:当 k 值较大(如 9×9)时,虽能保留更多空间信息,但会增加计算复杂度且可能限制特征的抽象层次;而当 k 值较小(如 3×3)时,则可能因信息过度压缩导致有效感受野不足,甚至出现输入图像中的关键区域在再特征图中映射不足一个像素的情况,严重影响模型的收敛性和性能。

以典型应用为例:在ImageNet分类任务中,通常采用5次下采样,考虑到原始图像分辨率约300,将再特征图尺寸设定为 7×7,对应输入分辨率为 224×224。

在目标检测任务中,由于多采用全卷积网络结构,输入尺寸设置更为灵活(如 416×416),并可支持多尺度训练策略,仅需保证输入尺寸为32的倍数即可。

多尺度训练策略在图像分类模型中的应用往往受限,主要原因在于分类网络的末端通常包含全连接层,这类层需要固定的输入维度以确保矩阵乘法能够正确执行。提高输入图像的分辨率能够提供更丰富的空间细节与更精确的定位信息,但与此同时,计算开销会随之急剧增加,通常与图像尺寸的平方成正比。

(2) 网络深度。网络深度是影响模型表达能力的关键因素之一,其通常指网络中层(如卷积层、全连接层)的总数。随着网络结构设计的演进,对"深度"的界定方式也在不断发展。在早期的骨干网络中,深度直接等同于逐层堆叠的卷积层总数。而在采用模块化设计的现代网络中(如通过神经架构搜索NAS得到的结构),深度也常被定义为所堆叠的模块数量。

一般而言,增加网络深度有助于模型学习更复杂、更抽象的特征层次(例如从边缘到纹理,再到物体部件乃至整体)。然而,当深度超过某个阈值时,可能导致梯度传递困难(如梯度消失或爆炸)、训练不稳定以及模型退化等问题。因此,最优的网络深度通常需要通过系统的架构搜索与参数调优来确定。

(3) 网络宽度。指每一层(尤其是卷积层)中的通道数或神经元数量。

神经网络的宽度表征了单层网络的信息承载能力,通常由通道数最多的卷积层所定义的通道维度决定。在常规设计中,随着网络深度增加,各层通道数往往呈递增趋势,至深层时达到最大值。这样的设计源于特征演进的内在逻辑:深层特征图虽空间分辨率降低,但语义信息更加抽象和丰富,需要更多卷积核(对应更多输出通道)来捕获和表达这类高阶特征。

更宽的网络每一层可以学习更丰富的特征(例如,同时检测多种不同方向的边缘),模型容量更大,但参数量和计算量也急剧增加。增加网络宽度通常能提升模型表征能力,但也会显著增加计算复杂度和参数规模。因此,宽度配置需在表达能力和计算开销之间寻求平衡,常通过实验调参确定。在实际设计中,通道数通常设置为8的倍数(如64、128、256),这一做法可充分发挥GPU硬件中张量核心的并行计算优势,实现对计算资源的高效利用。

图3-24为分辨率、深度和宽度示意图。

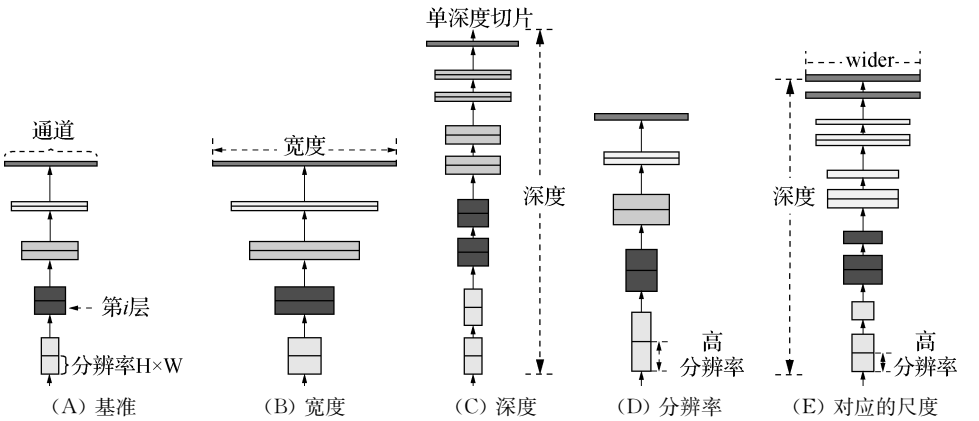


图 3-24　分辨率、深度和宽度

3.6.5　下采样和上采样

DL 中的下采样和上采样是一对核心且对立的概念,它们共同构成了许多现代神经网络架构(尤其是编码器—解码器结构和生成式模型)的骨干。

1. 下采样

(1) 下采样,也称为降采样,其目的是逐步减小特征图的空间尺寸(宽和高),同时增加特征图的通道数。核心目的为:

➢ 增加感受野:让后续层中的一个像素能够"看到"原始输入图像中更大的区域,使得后面的卷积核能够学到更加全局的信息,从而捕获更广泛的上下文信息。

➢ 降低计算复杂度:减少特征图尺寸能显著降低计算量和内存消耗,防止过拟合。

➢ 提取高级语义特征:通过聚合局部信息,形成更抽象、更具语义意义的特征。

➢ 引入平移不变性(尤其是池化层):使模型对目标物体的位置变化不敏感。

(2) 主要实现方法为池化和卷积:

➢ 采用步长(stride)为 2 的池化层,如最大池 Max-pooling 或平均池化 Average-pooling,目前通常使用 Max-pooling,因为它计算简单且最大响应能更好保留纹理特征。最大池化指,取窗口内的最大值,能更好地保留纹理特征;平均池化指取窗口内的平均值,输出更平滑。

➢ 跨步池化:本质是步长大于 1 的池化。

➢ 采用步长(stride)为 2 的卷积层,下采样的过程是一个信息损失的过程,而池化层是不可学习的,用 stride 为 2 的可学习卷积层来代替 pooling 可以得到更好的效果,当然同时也增加了一定的计算量。

➢ 步长卷积:使用步长大于 1 的卷积层,在计算时直接跳过一些像素,实现下采样。这是现代架构中更常用的方法,因为它在降维的同时还能学习参数。

2. 上采样

上采样也称为上采样,其目的与下采样相反,是逐步增大特征图的空间尺寸,同时减少

特征图的通道数。在CNN中,输入图像经过特征提取后,其输出尺寸通常会减小。然而,有些任务(如语义分割)需要将特征图恢复至原始输入尺寸以进行像素级预测。人们需要将图像恢复到原来的尺寸,以便进行进一步的计算(如图像的语义分割),这种将图像由小分辨率映射到大分辨率的操作,叫作上采样。核心目的:

➢ 恢复空间信息:将深层、低分辨率但富含语义信息的特征图恢复到更高的分辨率;

➢ 实现密集预测:对于需要输出与输入尺寸相同或相近的任务(如图像分割、超分辨率、图像生成),上采样是必不可少的步骤;

➢ 精确定位:通过恢复细节和空间结构,实现像素级的精确定位。

主要实现方法

➢ 转置卷积:也称为"反卷积"(但非数学意义上的反卷积)。通过在学习到的权重间插入零值来扩大特征图,并通过卷积操作恢复细节。该方法具有可学习的参数,能够自适应地学习最优的上采样策略,但需注意可能引发棋盘格效应。

➢ 上采样+卷积:先使用最近邻插值或双线性插值等确定性方法将特征图放大到目标尺寸,再紧跟一个标准的卷积层来平滑和优化结果。这种方法简单有效,能避免棋盘效应。其计算复杂度高于最近邻插值,但与卷积运算相比仍属轻量操作,且在保持视觉连续性方面表现优异。

➢ 反池化:尤其是最大反池化,在编码器进行最大池化时记录最大值的位置索引,上采样阶段将特征值还原至对应位置,在解码器中将值放回对应位置,其他位置补零。这种方法能精确恢复池化时丢失的位置信息。这种方法能较好地保留空间结构信息,但会引入非连续性特征。

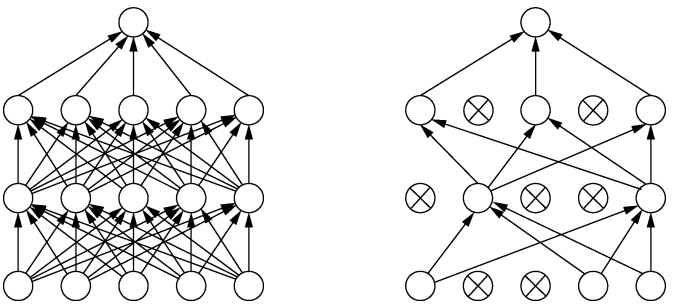
表3-6 上、下采样区别

维度	下采样	上采样
核心作用	信息压缩与抽象化	信息恢复与精细化
空间尺寸	减小	增大
通道数	通常增加	通常减少
感受野	扩大	在恢复的尺寸上相对变小
主要方法	池化、步长卷积	转置卷积、插值+卷积
典型应用	编码器、特征提取骨干网络	解码器、超分辨率、图像生成

下采样是"理解"图像内容的过程,而上采样是"描绘"图像细节的过程。二者相辅相成,使得DL模型既能把握全局语义,又能生成精细的局部结构,从而在复杂的视觉任务中取得卓越成就。

3.6.6 Dropout

Dropout是DL中广泛采用的正则化方法,在训练DL模型时,用于防止或减轻过拟合的正则化技术。其核心思想非常简单,却极其有效。在训练过程的每次前向传播中,随机"丢弃"(即临时隐藏)网络中的一部分神经元。这里的"丢弃"是指将这些神经元的输出暂时设置为零,使其在本次训练迭代中不参与前向传播和反向传播。图3-25为Dropout效果。



(a) Dropout 前　　(b) Dropout 后

图 3-25　Dropout 前后效果

想象一个大型项目团队,没有 Dropout:每次任务都由固定的核心成员完成,其他人产生依赖,一旦核心成员不在,项目可能停滞。使用 Dropout:每次执行任务时,都会随机地从团队中抽选一部分人来完成。这迫使团队中的每一个人都必须具备独立工作和与他人协作的能力,不能过度依赖少数几个"明星"成员。整个团队的鲁棒性和协作能力会变得更强,即使有人请假,项目也能顺利推进。

1. 工作原理与机制

(1) 训练阶段随机失活。对于网络中的每一层(通常是全连接层或卷积层),以预先设定的概率 p(如 0.5)随机选择一部分神经元,将其输出置零。这些被"关闭"的神经元在本次迭代中不贡献任何信息,也不参与权重的更新。每次输入一个批次的数据时,都会随机生成一个新的"丢弃掩码",这意味着网络的结构在每次迭代中都是动态变化的。

(2) 测试/推理阶段全体参与并缩放。在测试阶段,不再使用 Dropout,所有神经元都参与工作。

为了补偿训练时大量神经元被丢弃的情况,在测试时,该层的所有输出都需要乘以一个缩放系数$(1-p)$。例如,若训练时丢弃概率 $p=0.5$,则测试时该层的输出就要乘以 0.5。

现代实现(如 PyTorch,TensorFlow)的技巧:为了不在测试时做额外的操作,通常在训练时就直接对保留下来的神经元的输出进行放大,即乘以 $1/(1-p)$。这样,测试时网络就可以直接使用,无需任何调整,因为输出的期望值在训练和测试时已经保持一致。

2. 主要作用与优点

➢ 防止过拟合:这是最核心的作用。通过随机丢弃神经元,破坏了神经元之间复杂的协同适应关系,迫使它们不能过分依赖于少数几个其他神经元,从而让模型学习到更加鲁棒和泛化能力强的特征。

➢ 实现模型平均的近似效果:理论上,可以训练多个不同的模型并将它们的预测结果进行平均,这通常能提升性能,但计算成本极高。Dropout 在每次迭代中都在训练一个不同的"子网络",再的网络可以看作是指数级不同子网络在测试时的平均。这是一种非常高效且廉价的模型集成方法。

➢ 增加网络稀疏性:类似于一种生物进化压力,让网络在稀疏激活的情况下也能有效工作。

3. Dropout的双重作用机制

(1) 集成学习——宏观的“模型平均”

动态子网络训练:对于包含 n 个神经元的层,每次前向传播都随机采样一个子网络(理论上存在 2^n 种可能组合)。每个批次的数据仅用于更新当前活跃子网络的参数。

测试时的模型平均:在推理阶段,通过缩放激活值(乘以保留概率 $1-p$)来近似实现对所有可能子网络预测结果的加权平均。这相当于汇集了海量不同结构的模型的“集体智慧”,从而显著提升再预测的稳定性和准确率,是一种强大的方差降低技术。

Dropout隐式训练了指数级数量的子网络组合(对于含 n 个神经元的层,理论上可产生 2^n 种子网络结构)。每个训练批次仅更新当前随机子网络的参数,而测试阶段通过整合所有神经元输出,近似实现了多模型集成的平均预测效果,显著增强泛化性能。

(2) 特征协同——微观的“正则化约束”

打破共适应:在没有Dropout的网络中,某些神经元可能会过度依赖其“邻居”的存在,形成脆弱的特征检测链条。Dropout通过随机“静默”一部分神经元,强制每个神经元必须学会在“同伴”缺失的情况下也能有效工作,从而削弱这种复杂的共适应关系。

促进特征鲁棒性:这种机制迫使网络学习到更独立、更冗余、更鲁棒的特征表示。每个神经元都倾向于成为一个更具通用性的特征探测器,而不是一个只能在与特定其他神经元组合时才有效的“专家”。这增强了模型对输入噪声和损坏数据的抵抗能力,是一种降低过拟合的有效手段。

随机神经元失活破坏了神经元间的固定依赖关系,迫使网络避免对特定神经路径的过度依赖。这种机制促使神经元学习更具独立性的特征表示,削弱特征探测器间的共适应现象,从而提升模型对噪声输入的鲁棒性。

Dropout一方面在宏观上通过集成学习提升预测精度,另一方面在微观上通过破坏脆弱的共适应关系来增强模型的泛化能力。这两个机制相辅相成,共同使其成为一种简单而强大的正则化工具。当与批归一化(BN)一起使用时,由于BN本身就具有轻微的正则化效果,并且对网络动态敏感,有时会减少或取消Dropout的使用。需要根据具体任务和架构进行实验。

3.6.7 批归一化(BN)层与归一化(LN)

BN与LN是两种最核心的归一化技术,它们源于同一种思想,但针对不同的应用场景。共同目标是:通过归一化中间层的输出,解决“内部协变量偏移”问题,从而稳定训练过程、加速收敛并缓解梯度消失/爆炸。简单来说,为让每一层的输入分布变得更加稳定。

1. 批归一化(BN)

BN旨在加速神经网络的训练并提高收敛速度。通过对每一层输入的批量数据进行均值为0,方差为1的归一化处理。假定上一层输入结果为 $X=x_1,x_2,\cdots,x_m$,首先计算上一层输出数据均值,其中 m 为此次训练样本的批量:

$$\mu_\beta=\frac{1}{m}\sum_{i=1}^{n}(x_i) \tag{3-43}$$

计算上一层输出数据的标准差:

$$\sigma_\beta^2=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_p)^2 \tag{3-44}$$

$$\hat{x}_i=\frac{x_i-\mu_\beta}{\sqrt{\sigma_\beta^2+\varepsilon}} \tag{3-45}$$

式中,ε 是为了避免分母为 0 而引入的接近于 0 的值。经过对上面归一化处理得到的数据进行重构,得到:

$$y_i=\gamma\hat{x}_i+\beta_{y_i} \tag{3-46}$$

式中,γ 和β 为学习参数。该设计使网络能够自主判断标准化操作的有效性,并在必要时通过调整 γ 和β 恢复原始分布的某些特性。

核心优势:深度网络的训练受限于内部协变量偏移问题,即网络中间层输入的分布会随着前层参数的更新而不断变化。这种不稳定性导致模型训练困难,具体表现为:

(1) 收敛速度缓慢:需使用更小的学习率,且训练过程波动大;

(2) 梯度消失风险:当输入落入 Sigmoid/Tanh 等饱和激活函数的饱和区时,梯度显著减小;

(3) 参数初始化敏感:网络性能高度依赖精细的参数初始化策略。

BN 层被嵌入在神经网络层与激活层之间,通过对每层输出进行标准化和可学习的反变换,实现稳定训练的目的。

优点表现为:

(1) 效果显著:能大幅加速训练收敛,并具有一定的正则化效果。

(2) 广泛验证:在 CNN-based 的图像模型上取得了巨大成功。

缺点(局限性):

(1) 对 Batch Size 敏感:当 Batch Size 很小时,计算的均值和方差噪声很大,导致 BN 效果差甚至失效。

(2) 不适用于动态网络或序列模型:在 RNN/LSTM/Transformer 中,不同样本的序列长度可能不同,且测试时可能遇到比训练时更长的序列,BN 难以应用。

(3) 适用场景:CNN,特别是处理图像的 CNN(如 ResNet)。硬件条件允许使用较大 Batch Size 的训练任务。

2. *层归一化*

LN 是为了解决 BN 的局限性而提出的,它改变了归一化的维度。LN 是一种旨在改善深度神经网络训练稳定性和收敛速度的归一化技术。与批归一化(BN)沿批次维度进行标准化不同,LN 的核心思想是针对单个样本的所有特征通道进行归一化。

LN 的提出主要为了解决 BN 在小批次训练中的局限性:

批次依赖问题:BN 的性能高度依赖于批次大小,当批次较小时(如 BS=1,2),计算的批次统计量极不准确,导致性能显著下降。

序列模型适配:在 RNN/LSTM/Transformer 等处理变长序列的模型中,不同时间步的统计量动态变化,BN 难以有效应用。

LN通过对每个样本的所有通道特征进行独立归一化，完美规避了上述问题，使其特别适合：小批次或在线学习场景、循环神经网络和Transformer架构、生成式模型（如GANs）。

LN的操作可分解为两个阶段：

标准化阶段：对于输入张量 $\boldsymbol{X} \in \mathbf{R}^{B\times C\times H\times W}$（以CNN为例），LN独立地对每个样本计算所有通道的均值和方差：

$$\mu_b = \frac{1}{CHW}\sum_{c=1}^{C}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{bchw}$$

$$\sigma_b^2 = \frac{1}{CHW}\sum_{c=1}^{C}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{bchw}-\mu_b)^2 \tag{3-47}$$

然后进行标准化：

$$\hat{x}_{bchw} = \frac{x_{bchw}-\mu_b}{\sqrt{\sigma_b^2+\varepsilon}} \tag{3-48}$$

可学习重构阶段：与BN类似，LN引入逐通道的可学习参数 γ 和 β 进行缩放和平移：

$$y_{bchw} = \gamma_c \cdot \hat{x}_{bchw} + \beta_c \tag{3-49}$$

这一机制保留了网络的表达能力，使其在必要时可恢复原始分布。

LN通过转向样本内的特征归一化，解决了BN在特定场景下的根本局限性，已成为现代DL架构（尤其是自然语言处理领域）中不可或缺的组成部分。其设计理念体现了DL技术从“批次中心”向“样本中心”的重要范式转变。优势与应用场景：

训练稳定性：通过归一化层内激活值，缓解梯度消失/爆炸问题；

架构灵活性：不受批次大小限制，支持在线学习和动态网络结构；

序列模型友好：在Transformer中成为核心组件，支撑了BERT、GPT等模型的发展；

收敛加速：通常能减少训练迭代次数，提高学习效率。

尽管存在多种归一化方法，BN在CNN中仍占据主导地位，其与LN的关键差异在于归一化方向：BN沿批次维度归一化，对同一特征通道跨样本计算统计量；LN沿通道维度归一化，对单个样本的所有通道计算统计量。由于BN的统计量依赖批次数据，其在批次较小时效果会显著下降；而LN则不受批次大小影响，更适用于RNN/LSTM等序列模型及小批次训练场景。LN与BN的对比如表3-7所示。

表3-7 BN与LN的对比

特性	层归一化（BN）	批归一化（LN）
归一化维度	样本内所有通道 （N，H，W）-沿Batch方向	批次内同通道所有样本 （C，H，W）-沿Channel方向
统计量计算	基于一个Batch内所有样本的同一通道	基于单个样本的所有特征
批次依赖性	无依赖，适用于任意批次大小	强依赖，小批次时性能下降
适用场景	RNN、Transformer、小批次训练	大型CNN、稳定批次训练
统计量稳定性	样本内统计，相对稳定	批次间统计，波动较大

选择 BN 还是 LN,主要取决于模型架构和任务类型。对于图像相关的 CNN(如图像分类、目标检测),BN 通常是首选,因为它能充分利用批次信息,性能卓越。对于序列模型(如自然语言处理、语音识别)、小批量训练或生成式模型,LN 是更优且更普遍的选择,因为它不依赖于批次大小,行为稳定。近年来,也出现了许多新的归一化方法(如 Instance Norm,Group Norm 等),它们都可以被视为在 BN 和 LN 的思想基础上,针对不同维度的折中与探索。

3.6.8 参数量和计算量(FLOPs)

模型的参数量和计算量(FLOPs)是 DL 模型设计、优化和部署的核心环节。它们共同衡量了一个模型的复杂度和对计算资源的需求。

1. 参数量

参数量指所有需要被学习和更新的变量的总数,包括卷积核的权重 weight,BN 和 LN 的缩放系数 γ、偏移系数 β,有些没有 BN 的层可能有偏置 bias,这些都是可学习的参数,即在模型训练开始前被赋予初值,在训练过程根据链式法则中不断迭代更新。整个模型的参数量主要由卷积核的权重 weight 的数量决定,参数量越大,则该结构对运行平台的内存要求越高,参数量的大小是轻量化网络设计的一个重要评价指标。参数量重要,因为:

内存占用:参数直接存储在内存中,参数量越大,模型文件越大,加载模型所需的内存也越多。这对于移动端和嵌入式设备至关重要。

过拟合风险:参数量巨大的模型拥有更强的拟合能力,但也更容易过拟合训练数据,因此需要更多数据或更强的正则化。

通信成本:在分布式训练中,参数需要在不同的计算节点之间同步,参数量直接影响通信带宽的需求。

计算包括:

- 全连接层:参数=(输入维度+1)×输出维度(其中+1 来自偏置项)。
- 卷积层:参数=(卷积核高度×卷积核宽度×输入通道数+1)×输出通道数(其中的+1 同样来自偏置项,现代框架有时会省略)。
- 归一化层(如 BN/LN):参数=2×特征通道数(对应 γ 和 β)。

2. 计算量(FLOPs)

FLOPs 是 Floating Point Operations 的缩写,即浮点运算次数。它用于衡量模型执行一次前向传播(推理)所需的计算负担。注意与 FLOPS 区分,后者是 Floating Point Operations Per Second,即每秒浮点运算次数,是衡量硬件计算能力的单位。FLOPs 重要,因为:

- **速度**:FLOPs 与模型的**推理速度**和**训练速度**高度相关。计算量越大,执行时间通常越长。
- **能耗**:在电池供电的设备上,计算量直接决定了**功耗**。
- **硬件选择**:不同的硬件(如 CPU、GPU、专用 AI 芯片)对不同类型运算的吞吐量不同,但 FLOPs 是一个通用的、重要的评估基准。

表3-8 FLOPs特性、参数量与计算量

特性	参数量	计算量(FLOPs)
衡量对象	模型的静态复杂度与存储开销	模型的动态复杂度与计算开销
主要影响	内存占用、过拟合风险	推理/训练速度、功耗
关系	相关但不直接等价	相关但不直接等价

参数量与计算量并不总是正相关。一个模型可以有很大的参数量但计算量很小,反之亦然。

(1) 高参数量,低FLOPs的典型:

大型全连接层:一个巨大的全连接层可能拥有数十亿参数,但计算它只需要一次矩阵乘法,相对效率较高。

深度可分离卷积:参数量远少于标准卷积,但为了实现相同的变换,其FLOPs的减少比例可能不如参数量则显著。

(2) 低参数量,高FLOPs的典型:

大尺寸输入图像的卷积操作:即使模型本身不大(参数量少),但在高分辨率图像上进行卷积操作会产生巨大的FLOPs。

某些轻量级模块:为了减少参数而设计的模块,有时需要通过增加深度或宽度来补偿性能,这可能会增加FLOPs。

神经网络的前向推理过程基本上都是乘累加计算,所以它的计算量也是指的前向推理过程中乘加运算的次数,通常用FLOPs(即“每秒浮点运算次数”)来表示,即Floating Point Operations(浮点运算数)。计算量越大,在同一平台上模型运行延时越长,尤其是在移动端/嵌入式这种资源受限的平台上想要达到实时性的要求就必须要求模型的计算量尽可能地低,但这个不是严格成正比关系,也跟具体卷积核的计算密集程度(即计算时间与IO时间占比)和该卷积核底层优化的程度有关。

在进行模型设计或选择时,需要根据具体约束条件在参数量、计算量和精度之间进行权衡:

服务器端部署:可能更关心吞吐量,FLOPs是关键指标。

移动端/嵌入式部署:内存和功耗限制严格,因此参数量和FLOPs都需要极小化。

学术研究:在公平比较时,需要汇报在相似参数量或FLOPs下的性能。

总之,参数量和FLOPs是两个互补且至关重要的模型效率指标,是模型评估的重要指标。

3.6.9 损失函数

在DL模型中,损失函数是评估模型性能的核心指标,它通过量化模型预测值与真实值之间的差异,为参数优化提供了明确的指导方向。损失函数的核心价值在于,它能够将模型的拟合能力与泛化性能转化为可优化的数学目标。具体而言,通过计算损失函数相对于模型参数的梯度,引导优化算法(如梯度下降)调整权重与偏置,从而逐步降低损失值,提升模型的预测准确度。通常而言,损失值与模型性能呈负相关——损失越低,表明模型的预测结果与真实情况越接近。损失函数的选择直接决定了模型的学习目标与优化行为。一个合适

的损失函数能够:

- **精准反映任务需求**:确保模型优化方向与再的评价指标一致;
- **影响收敛速度与稳定性**:不同损失函数的梯度特性不同,会直接影响训练过程的效率;
- **提供模型解释性**:通过观察损失的变化,可以诊断模型的训练状态(如是否过拟合、欠拟合)。

表 3-9 为一些损失函数及其适用场景。

表 3-9 损失函数及其适用场景

损失函数	适用场景
交叉熵损失	分类任务
均方误差	回归任务
平滑 L_1 损失	回归任务
对比损失	序列和匹配任务
焦点损失	序列和匹配任务
KL 散度损失	概率分布间差异度量
余弦相似度损失	文本和特征相似性任务
二次熵损失	二分类任务

下面介绍常见的损失函数:

(1) 均方误差(MSE)。在 DL 中,MSE 作为一种经典的回归损失函数,被广泛用于衡量模型预测值与真实值之间的偏离程度。其基本思想是通过最小化预测值与真实值之间差值的平方和,推动模型不断逼近数据的真实分布。MSE 的值越小,表明模型的拟合效果越好,预测精度越高。定义如下:

$$L(Y/f(x))=\frac{1}{n}\sum_{i=1}^{n}(Y_i-f(x_i))^2 \tag{3-50}$$

MSE 具有形式简洁、计算高效、处处可导等优点,因而成为一种稳定且易于优化的距离度量方式。尽管在图像生成、语音合成等强调感知质量的任务中,MSE 对结构相似性和听觉自然性的刻画能力有限,但它仍然是回归任务中评估模型性能的基础性指标。在实际应用中,MSE 既可作为训练过程中的损失函数以引导参数学习,也可作为模型评估阶段的重要参考依据。

与 MSE 密切相关的 L_2 损失是另一种常用的差异度量方式,定义为:

$$L(Y/f(x))=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i-f(x_i))^2} \tag{3-51}$$

L_2 损失函数具有良好的凸性与光滑性,在优化过程中能够提供稳定而有效的梯度信息。从概率视角来看,在误差项满足独立同分布的高斯噪声假设下,最小化 L_2 损失等价于对模型参数进行最大似然估计。这些特性使 L_2 损失成为回归分析、模式识别与图像处理等

领域中构建损失函数的常用基础。

(2) 交叉熵损失函数。在DL中,交叉熵损失函数用于度量模型输出的预测概率分布与真实的标签分布之间的差异。交叉熵的值越小,说明两个概率分布越接近,模型的预测也越准确。该损失函数具有良好的数学性质,其在优化过程中能够提供稳定有效的梯度,有助于缓解梯度消失问题,从而促进模型的高效学习。交叉熵损失函数

$$L(Y/f(x)) = -\sum_{i=1}^{n} Y_i \log f(x_i) \tag{3-52}$$

式中,y_i 是真实标签的one-hot编码(只有真实类别 t_t 的 $y_t=1$,其余为0)。因此,上式可以简化为:

$$L = -\log(S_t) \tag{3-53}$$

式中,S_t 是模型为真实类别 t 所预测的概率。

损失值 $L=-\log(S_t)$ 非常直观。它直接惩罚模型对于正确类别预测概率的低信心度。若模型对正确类别的预测概率是100%($S_t=1$),则损失为0;若预测概率很低(比如 $S_t=0.1$),则损失 $-\log(0.1)$ 会很大。由于交叉熵损失直接作用于概率输出,对错误预测的惩罚随误差增大而增强,这一特性使其特别适用于处理分类任务,包括正负样本不均衡的场景。在二分类任务中,交叉熵损失也常被称为对数损失。目前,该损失函数已成为CNN等DL模型中分类任务的首选损失函数之一。

(3) Softmax损失函数。在DL中,Softmax损失函数(也称为Softmax交叉熵损失函数)是解决多分类任务最核心的损失函数之一。它不是一个独立的新损失函数,而是Softmax函数和交叉熵损失的一个完美结合。对于有 C 个类别的多分类问题,给定一个样本 x,Softmax损失函数

$$S_t = \exp(z_i) \Big/ \sum_{j=1}^{C} \exp(z_j) \tag{3-54}$$

式中,z_i 为样本 x 属于第 i 类的概率。

Softmax损失函数通过将分类输出概率化并利用交叉熵衡量误差,为深度神经网络提供了稳定而强大的梯度信号,使其成为多分类任务中无可争议的默认选择。理解其内部的工作流程和梯度特性,是掌握DL模型优化关键的一步。

在选择损失函数时,需结合任务目标、数据特性和模型架构进行综合考量。以下是几个关键指导原则:

分类任务:优先选择交叉熵。因为它直接衡量概率分布之间的差异,其梯度特性(当误差越大时梯度越大,反之越小)非常有利于模型快速修正错误。

回归任务:优先选择MSE,将它作为你的性能基准。这是衡量其他更复杂损失函数的起点。

均方误差:假设误差服从高斯分布,对大误差施加严厉惩罚。若任务目标是对极端错误零容忍(如金融风险预测),MSE是首选。

平均绝对误差:假设误差服从拉普拉斯分布,对异常值不敏感,更为稳健。当数据中存在显著噪声或异常值时,MAE通常表现更好。

Huber Loss:综合了 MSE 和 MAE 的优点,在误差较小时具有 MSE 的稳定梯度,在误差较大时具有 MAE 的鲁棒性。

生成任务(如图像生成):常使用对抗损失,通过判别器来指导生成器的输出,使其分布逼近真实数据分布。

系统化的损失函数选择策略能够显著提升模型性能,是 DL 应用中的关键设计环节。在 DL 中选择合适的损失函数是一个关键决策,它直接影响模型的收敛速度、再性能和泛化能力。表 3-10 为选择损失函数的系统方法论和实用基本原则。

表 3-10 选择损失函数的系统方法论和实用基本原则

任务类型	推荐损失函数	核心考量
分类任务	交叉熵损失	直接优化预测概率与真实分布的距离
二分类	二元交叉熵	配合 Sigmoid 输出
多分类	分类交叉熵	配合 Softmax 输出
多标签分类	二元交叉熵	每个类别独立判断
回归任务	均方误差(MSE)	对大误差更敏感,假设误差呈高斯分布
—	平均绝对误差(MAE)	对异常值更鲁棒
—	Huber 损失	MSE 和 MAE 的折衷,兼具两者优点
目标检测	多种损失组合	分类损失+回归损失+置信度损失
语义分割	Dice 损失	专门处理类别不平衡,优化重叠度
—	交叉熵+Dice 组合	兼顾边界精度和区域重叠

数据存在特定问题时,需要选择专门的损失函数:

(1) 类别不平衡问题

- Focal Loss:降低易分类样本权重,聚焦困难样本。
- 加权交叉熵:为少数类分配更高权重。
- Dice Loss:直接优化分割区域的重叠度。

(2) 异常值和噪声数据

- Huber Loss:对小误差使用 MSE,大误差使用 MAE。
- Log-Cosh Loss:平滑版本的 MAE,数值更稳定。

(3) 边界敏感任务

- Contrastive Loss/Triplet Loss:优化特征空间中的相对距离。
- 边界相关损失:如目标检测中的 IoU Loss。

3.6.10 过拟合

在 DL 模型中,过拟合是指模型在训练集上表现优异,但在未见过的测试数据上性能显著下降的现象。这是由于模型过度适应训练数据中的特定样本特征(包括噪声),导致其泛化能力不足。以下是系统化的过拟合抑制策略体系:在 DL 模型中,过拟合抑制方法已形成多层次、多角度的技术体系,主要包括以下三类核心策略:

(1) 结构正则化

Dropout:通过随机神经元失活打破网络固有依赖;

参数共享:CNN/RNN中通过权值复用降低模型复杂度;

批归一化:借助激活值分布标准化产生轻微正则效果。

(2) 优化策略

L1/L2正则化:在目标函数中引入参数范数惩罚项;

提前终止:基于验证集性能监控动态调整训练周期;

Bagging集成:通过基模型投票机制降低预测方差。

(3) 数据层面控制

数据增强:通过几何/色彩变换扩充训练样本多样性;

噪声注入:在输入层或隐藏层添加随机扰动提升鲁棒性。

(4) 样本扩增

几何变换:旋转、裁剪、缩放;

色彩调整:亮度、对比度、饱和度扰动;

高级增强:Mixup、Cutmix等混合样本生成。

(5) 噪声注入

输入层:高斯噪声、随机遮挡;

隐藏层:对激活值添加随机扰动。

(6) 数据质量提升

清洗异常样本;

类别平衡处理(重采样/损失加权)。

(7) 现代正则化

标签平滑:软化one-hot标签的绝对确定性;

随机深度:训练时随机跳过网络层;

知识蒸馏:通过教师网络指导学生网络训练。

过拟合控制是DL工程中的核心环节,需要根据具体任务的数据规模、噪声水平和模型复杂度,动态调整策略组合。建议建立系统化的评估流程,通过消融实验确定最有效的策略组合。DL的正则化体系从模型结构、优化过程和数据源三个维度共同构建了泛化能力保障机制。其中Dropout作为最具代表性的结构化正则化技术,通过其独特的随机失活机制,在维持模型表征能力的同时实现了有效的过拟合控制。

3.6.11 DL的可解释性

在图像分类任务中,传统方法依赖人工设计的特征,其语义明确、可解释性强;而DL模型以数据驱动方式自动学习特征表示,虽在性能上表现卓越,却因其内部表示的抽象性而难以被人类理解。为揭示模型的决策机制,可解释性研究应运而生。

目前,可解释性尚缺乏统一的数学定义。从用户认知角度出发,将其定义为连接用户与模型的桥梁,其核心作用是将模型内部的抽象表示映射至人类可理解的语义空间。

如图3-26所示,深度模型从数据中提取的特征可划分为任务相关特征与任务无关特征。尽管模型倾向于利用所有可用特征以提升分类准确率,但精度并非唯一评价标准,模型

的可信度同样至关重要。如图 3-26 中箭头所示,可解释性研究的目标是在保持模型性能的基础上,增强其可信赖性,即借助解释器将模型特征空间中的相关特征转化为符合人类认知习惯的语义表达。然而,由于模型与人类在认知机制上存在本质差异,DL 模型可能学习到某些超出人类认知范畴的非语义特征。尽管这类特征可能在特定任务中提升模型表现,但也可能削弱其泛化能力与鲁棒性,构成了可解释性研究中亟待解决的核心问题。

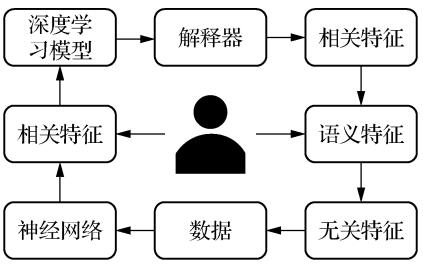


图 3-26　DL 模型可解释性过程

1. 可解释性 DL 方法分类

(1) 模型内部可视化。模型内部可视化方法旨在对网络中的权重参数、神经元或特征检测器进行可视化分析。由于权重直接反映不同特征对预测结果的贡献程度,可视化权重可直观展示模型内部运行机制。类似地,对神经元或特征检测器进行可视化,也能揭示输入特征在模型中的演变过程。

尽管这类方法能提供直观的模型内部视图,但其普适性有限,难以得出通用结论,且解释效果仍有提升空间。典型的模型内部可视化方法包括可视化系统 CNNVis 和可视化工具 Lucid,如表 3-11 所示。

表 3-11　可解释性方法

方法类型	典型方法
模型内部可视化	可视化系统 CNNVis,可视化工具 Lucid
特征统计分析	LIME,CAM,Grad-CAM,解释图表征
本质上可解释模型	DLIME,决策树量化解释,决策树正则化

(2) 可视化系统 CNNVis。CNNVis 通过以下三个模块实现模型内部结构的可视化:

数据预处理模块:将神经网络转换为有向无环图,节点表示神经元,边表示连接关系,并计算神经元的派生特征及其关联性。

聚合模块:对特征图进行聚类,选取代表性图层;进一步对神经元聚类,并选出代表性神经元。

可视化模块:展示神经元集群,支持用户分析网络所学特征、激活模式及其对结果的贡献。系统还提供交互功能,允许用户调整聚合过程以深入观察模型行为。

2. 可视化工具 Lucid

Lucid 是基于 DeepDream 构建的神经网络可视化库,集成了先进的特征可视化技术,适

用于探索神经网络的工作原理。该工具结合特征可视化与其他可解释性方法，使研究者能够“进入网络内部”，观察神经元在特定决策中的激活情况，理解网络识别图像所依据的特征。

尽管可视化技术是当前可解释性研究中最直观的途径之一，其仍存在一定局限：

- 大部分可视化结果仍难以被人直接理解；
- 缺乏统一的评价标准，影响了解释结果的可信度；
- 通常需与其他解释方法结合使用，作为再结果的呈现方式。

目前，可视化技术主要用于局部可解释性分析，以特征图等形式揭示深度神经网络的决策机制。

3. 特征统计分析

与ML相比，DL通常内部结构复杂，导致特征与预测结果之间的因果关系难以追溯，从而形成“黑盒”问题。为解决这一挑战，特征统计分析方法被提出，其核心在于对模型所学习特征进行系统性的量化分析与可视化呈现，以揭示不同特征对再输出的贡献程度，进而增强模型的透明性与可解释性。该方法不依赖模型内部参数，而是通过分析输入特征与输出之间的统计关系实现解释。

以下是两种典型的基于特征统计分析的可解释性方法：

(1) 局部可解释模型无关方法(LIME)

LIME的核心思想是在特定预测样本的局部邻域内，构建一个可解释的代理模型(如线性模型)，以近似原始复杂模型在该区域的行为。具体实现中，LIME通过对输入样本施加符合人类语义理解的扰动(例如对图像区域进行遮挡)，生成一系列扰动样本，并观察原始模型对这些样本的预测变化。根据预测结果的变化与扰动特征之间的关系，LIME拟合出一个局部线性模型，进而识别出对当前预测最重要的特征。

LIME的优势在于其模型无关性，适用于任何黑盒模型，并在文本和图像任务中取得了显著的可解释效果。然而，该方法也存在一定局限性：由于扰动生成的随机性，LIME对同一预测结果可能生成不一致的解释，这种不稳定性可能影响用户对解释结果的信任。

(2) 类激活映射及其梯度增强方法(CAM与Grad-CAM)

该类方法基于CNN中最后一个卷积层所包含的丰富空间与语义信息，通过生成热力图直观展示输入图像中对分类决策起关键作用的区域。

CAM通过使用全局平均池化层(GAP)替代全连接层，将最后一个卷积层的特征图与类别输出直接关联。在对特定类别进行解释时，CAM计算每个特征图对该类别的权重，并通过加权求和生成类激活热力图，突出显示对分类贡献最大的图像区域。尽管CAM增强了对输入的尺度鲁棒性并降低了过拟合风险，但其需要修改网络结构并重新训练，限制了其应用灵活性。

Grad-CAM作为CAM的扩展，无需改变网络结构，适用于包括预训练模型在内的多种架构。其利用目标类别相对于最后一个卷积层特征图的梯度信息，计算各特征图的重要性权重，进而生成精细的类激活热力图。Grad-CAM不仅能定位关键区域，还可与像素空间方法结合，生成高分辨率的视觉解释，在保持解释力的同时显著提升了方法的通用性。

4. 特征图表征

特征图表征是一种通过从深度神经网络中学习具有层次化语义结构的解释图,以解构模型内部知识表示的方法。该方法以无监督方式自动从卷积层的特征图中识别出具有语义意义的视觉部件,无需额外标注信息。

学习得到的解释图具有多层结构,每一层与神经网络中的卷积层相对应,每个节点代表一个特定的语义部件,边则建模了节点间的共现激活关系与空间变换模式。在层次化结构中,高层节点对应较大的部件,低层节点则描述其细分子区域。通过将数百万个神经单元所编码的信息压缩为数千个图节点,特征图表征实现了对深度模型内部知识的有效解构与语义化解释,如图 3-27 所示。

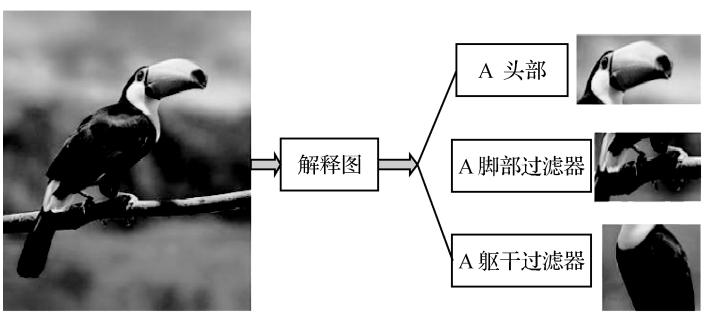


图 3-27 特征图表征

5. 本质可解释模型

随着迁移学习理念的扩展,研究者不仅实现了模型结构的迁移,也探索了将模型可解释性进行迁移的方法。通过将黑盒的 DL 模型映射到本质可解释的模型(如线性模型、决策树等),可以实现对复杂模型的解构与理解。

(1) 线性模型方法

线性模型因其结构简单、参数透明而具有良好的可解释性。在关键应用领域(如医疗诊断)中,决策结果的可信度至关重要,不仅需要模型给出预测,更需要提供可信的解释依据。

DLIME 方法在 LIME 基础上进行了改进:首先使用层次聚类对数据集进行分组,然后通过 K 近邻选择与测试样本最相似的邻域数据点,再训练线性回归模型生成解释。相较于 LIME 的随机扰动策略,DLIME 通过确定性聚类保证了解释结果的稳定性。然而,该方法的效果受数据集规模影响,样本数量不足时可能影响聚类质量与解释准确性。

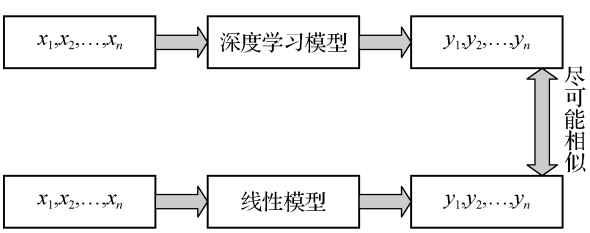


图 3-28 利用线性模型解释黑盒模型

(2) 决策树方法

决策树模型通过树形结构直观展示决策逻辑,为实现深度模型的全局解释提供了有效途径。决策树量化解释方法通过在语义层次上解析神经网络预测逻辑,确定输入图像中哪些物体部件被用于预测,并量化每个部件的贡献度。该方法通过修正神经网络获得解耦的特征表示,并学习决策树来解码全连接层中的决策模式,实现对预测原理的量化分析;决策树正则化方法通过构建模拟决策树来逼近神经网络预测,在训练过程中引入树正则化项,鼓励网络学习更简单、更易解释的决策边界。这种方法在保持神经网络非线性能力的同时,提高了模型的可模仿性与可解释性。

6. **可解释性DL模型构建方法**

(1) 基于模型代理的可解释性建模

代理模型法通过构建简化模型来近似复杂DL系统的行为。反复蒸馏方法是一个典型代表,体现了可解释性迁移的学习思想。

该方法构建教师-学生网络框架:教师网络负责将逻辑规则知识进行建模,学生网络在教师网络的约束下通过反向传播模拟规则化预测。两个网络通过联合训练迭代优化,将一阶逻辑规则的结构化信息转化为神经网络的权重参数。这种基于后验正则化的知识转移机制,使得模型在保持高精度的同时具备逻辑可解释性,已成功应用于CNN和RNN等多种网络结构。

(2) 基于逻辑推理的可解释性建模

逻辑推理方法通过引入形式化逻辑体系来增强模型的可解释性,体现因果关联的推理过程。

神经符号学习系统将论证网络转换为标准神经网络,实现基于权重的论证框架。该方法将论点区分为正面与反面论证,通过网络学习实现论证强度的动态调整,展现类似人类推理的学习过程。另一种深度强化学习方法将逻辑规则作为状态转换函数,通过记忆网络存储关系元组,模仿人类认知中的"图式模式",将推理定义为基于记忆检索的顺序决策过程。

(3) 基于网络节点关联分析的可解释性建模

胶囊网络通过改进传统CNN架构,在节点关联层面提供了解释性。每一组神经元构成一个胶囊,通过活动向量表示实体实例化参数:向量长度表征存在概率,方向编码实例化参数。

胶囊网络采用协议路由机制,为能够更好拟合高层胶囊的低层胶囊分配更高权重。这种机制使得每个胶囊能够编码特定语义概念,清晰展示每个胶囊的功能分工,从而构建出能够识别对象空间结构信息的可解释模型。

(4) 基于传统ML的可解释性建模

传统ML方法为构建可解释DL模型提供了重要的方法论基础。例如,DeepRED方法将基于决策树的规则提取机制扩展至深度网络,通过逆向推导剔除冗余输入特征。尽管该方法能够构建与原始网络高度一致的决策规则,但其可扩展性仍面临一定限制。

gcForest方法则采用深度树集成的创新架构,通过级联多层级联森林实现特征学习。相较于深度神经网络,该框架具有超参数配置简化、内在可解释性增强、训练数据需求降低等优势,在跨领域应用中均展现出稳定的性能表现与良好的解释能力。

当前,可解释性 DL 模型的构建主要围绕以下四个核心维度展开演进:

可信任性:为模型使用者提供明确的决策依据与置信度评估,清晰界定系统行为边界,构成可信 AI 系统的核心基础。

因果关联性:从逻辑推理与特征关联双重视角建立因果机制,如胶囊网络的结构化表征和论证框架的逻辑约束所示。

迁移学习性:将先验知识与结构化信息嵌入神经网络参数,典型代表如知识蒸馏中的师生架构传递机制。

信息提供性:向终端用户提供符合认知习惯的知识表示,包括特征可视化、注意力图谱等与传统 ML 结合的阐释方法。

这些发展维度系统性地推动了 DL 从“黑盒”向“白盒”的范式转变,为构建具备可靠性、透明性与实用性的智能系统奠定了理论基础与方法支撑。

本章习题

1. 核心概念辨析

简要阐述以下概念的区别与关联:(1) 传统机器学习与深度学习在方法论上的核心差异体现在哪些维度?(2) 下采样与上采样的主要目的分别是什么?请各列举两种典型实现方法。

2. 卷积操作理解

(1) 请简述标准卷积、1×1 卷积以及深度可分离卷积(分解为逐通道卷积与逐点卷积)的核心思想与计算特性。

(2) 假设输入特征图尺寸为 64×64×32(高×宽×通道),使用 3×3 卷积核,输出通道数为 64,步长为 1,填充方式为“same”。试计算该卷积层的参数数量与输出特征图尺寸。

3. 池化层的作用与类型

(1) 池化层在卷积神经网络中的主要作用是什么?为何其在 CNN 结构中具有不可替代性?

(2) 最大池化、平均池化与全局平均池化分别适用于哪些典型场景?请结合实例说明。

4. 激活函数的选择

(1) 神经网络中为何必须引入非线性激活函数?若全部使用线性激活函数,模型将出现何种局限?

(2) ReLU 函数为何成为当前最主流的激活函数?它存在哪些缺陷?Leaky ReLU 或 ELU 是如何尝试改进这些缺陷的?

5. 注意力机制分析

(1) 请简述通道注意力(如 SE 模块)与空间注意力(如 CBAM 中的空间注意力子模块)的核心思想与计算流程。

(2) 从特征选择的角度解释,为何注意力机制能够有效提升模型在目标检测等任务中的性能?

6. 批归一化与层归一化

(1) 批归一化(BN)的主要作用是什么?请简述其前向传播中的计算步骤。

(2) BN在哪些场景下可能失效？层归一化(LN)如何应对这些问题？LN更适用于哪类网络架构？

7. 损失函数与任务适配

(1) 均方误差损失与交叉熵损失分别适用于哪类任务？试说明其原理依据。

(2) 针对图像分类任务中存在的严重类别不平衡问题，可选用哪种改进的损失函数？请阐述其核心设计思想。

8. 过拟合与正则化

(1) 什么是过拟合现象？列举三种深度学习中常用的过拟合抑制策略，并简要解释其原理。

(2) Dropout作为一种正则化方法，在训练阶段与测试阶段分别是如何操作的？其为何能起到正则化效果？