国家教材委员会专家委员会审核通过
2020

普通高中教科书

信息技术

选择性必修 4 人工智能初步

华东师范大学出版社

普通高中教科书

信息技术

选择性必修 4

人工智能初步

华东师范大学出版社

总　主　编：李晓明

副 总 主 编：赵　健

本 册 主 编：于晓雅

本册副主编：刘　焱　李　粤

编 写 人 员（按姓氏笔画排序）：
于晓雅　刘　焱　李　粤　唐斯斯

责 任 编 辑：曹祖红

美 术 设 计：储　平

普通高中教科书　信息技术　选择性必修 4　人工智能初步

上海市中小学(幼儿园)课程改革委员会组织编写

出版发行　华东师范大学出版社(上海市中山北路 3663 号)

印　　刷　上海四维数字图文有限公司

版　　次　2021 年 3 月第 1 版

印　　次　2021 年 3 月第 1 次

开　　本　890 毫米×1240 毫米　1/16

印　　张　8

字　　数　140 千字

书　　号　ISBN 978-7-5760-0552-3

定　　价　10.10 元

致同学们

亲爱的同学们：

当今，信息技术的发展日新月异，物联网、大数据、人工智能等新技术、新工具扑面而来，显著地改变着人们的生活、学习和工作模式。生存于信息社会中，我们每一个人都不可避免地会接触信息技术、应用信息技术，甚至去创造新的信息技术。在具备了基本信息技术应用能力的基础上，高中阶段我们要进一步学习信息技术知识与技能，能够利用信息技术负责任地解决生活与学习中的问题，全面提升信息素养，迎接信息社会的挑战。

“人工智能初步”是高中信息技术学科的一个选择性必修模块。本教科书是在同学们已学习过高中信息技术学科必修模块的基础上，以全面深化对人工智能的理解和认识、掌握人工智能基本思想和方法及应用为目标的进阶性教科书。

本教科书沿用高中信息技术系列教科书的特点，以“项目活动”为主线来组织学习内容，通过一系列精心设置的体验性、思考性和实践性项目，将同学们带入生活和学习中的各种应用场景中，并循序渐进、由浅入深地系统介绍人工智能的核心思想和方法及技术、负责任地使用智能技术。

为方便同学们学习，本教科书的每章都围绕信息技术学科核心素养的要求和人工智能学习的特点来设计学习目标，并以“本章知识结构”图示呈现本章知识脉络，帮助同学们从总体上了解本章学习内容。同时，本教科书精心设计了帮助同学们学习的各种工具和环节。例如：“体验思考”栏目将教材内容与现实中的问题及个人经验和知识技能相关联，使同学们始终带着问题学习；“探究活动”和“项目实践”栏目将“做中学”与“学中做”的学习方法相互融合，帮助同学们把知识技能应用于解决实际问题中；“技术支持”栏目提供了一些与人工智能相

关的新技术、新工具与新平台的信息，鼓励同学们在实践中负责任地思考，提高应用及把握人工智能技术的能力；“知识延伸”栏目则补充关于人工智能研究、开发、应用及伦理等方面的最新成果信息，助力同学们拓展视野。

同学们，人工智能是一项面向未来的技术，它必将对我们的社会、生活、学习、工作产生深远的影响。希望同学们通过本教科书的学习，能够学会、掌握和负责任地使用强大的智能技术，充满信心地迎接未来智能社会的种种挑战，成长为新时代合格的社会主义建设者和接班人。

编　者

目　录

第一章

人工智能初识

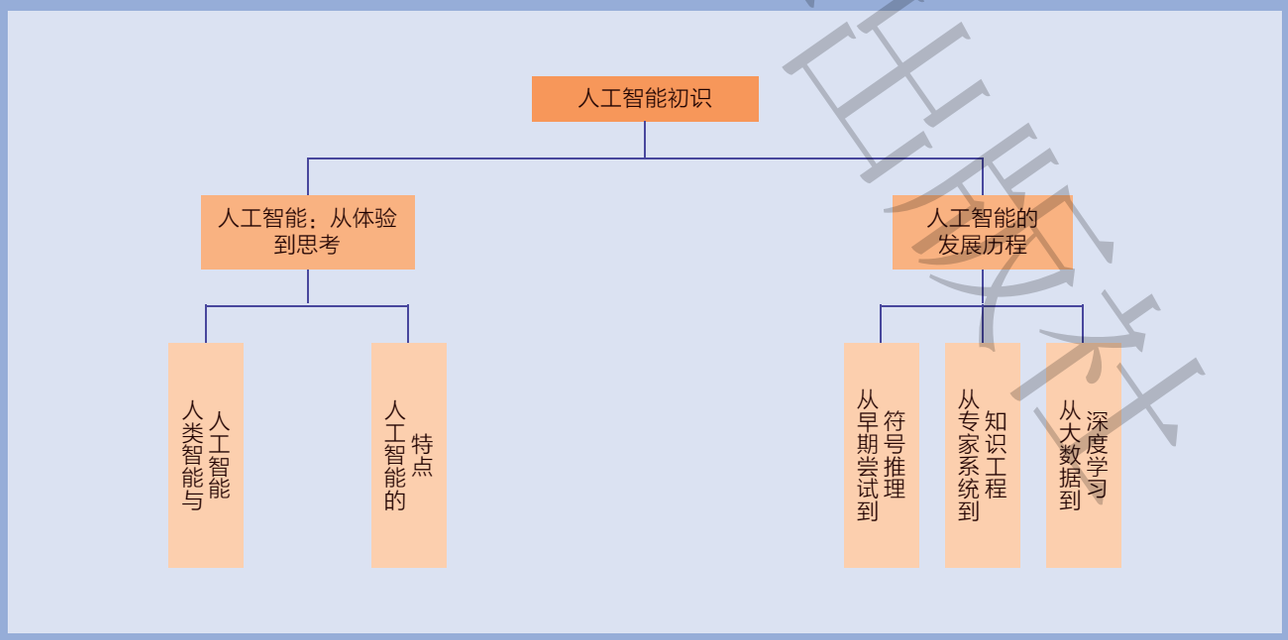
本章学习目标

- 了解围绕智能、人工智能及相关概念的主要观点和争论，熟悉人工智能发展历程中的重要人物和事件，初步认识人工智能观念的复杂性及多学科属性。
- 领会图灵测试、中文屋问题等检验人工智能的设想和方案，理解其中的辩证关系、论据及其局限性。
- 分析人工智能发展历史中产生高潮和低谷的原因，认识大数据、算法和计算力对人工智能复兴的共同推动作用。
- 体验搭建身边的人工智能应用，初步形成自己的人工智能观念。

1956年，五位学者发起美国达特茅斯夏季研讨会，会上正式提出“人工智能”（artificial intelligence缩写为AI）一词，标志着人工智能这一领域的诞生。在随后的60多年中，人工智能历经“进展—争议—突破—期待—应用—失望—寂静—复兴”的发展轮回，其间有过“高潮”“低谷”甚至“寒冬”。进入21世纪后，在大数据逐渐积累、高性能计算的能力持续提高和深度学习算法取得突破等因素的共同驱动下，人工智能进入以应用为主要特征的第三次复兴时期。在这一时期，被赋予了“学习能力”的计算机不仅在下棋、游戏、识图和辨音等方面的智力竞赛中赢了人类，更重要的是催生了各种全新行业，给人们的生活带来了智能化便利，也深刻地改变了人类的思维和行为方式。掌握人工智能知识、驾驭人工智能技术、学会智能地解决问题是人类面向未来、迎接挑战的绝佳途径。

本章将引领同学们从接触身边的人工智能应用开始，走进人工智能的世界，通过阅读、思考、分享、体验等学习活动，以全局的、历史的和发展的多重视角，了解围绕着人工智能所激发的各种思想、概念和争论，思考“智能”的本质，动手体验人工智能的应用，初步认识新一代人工智能的背景、技术和应用特点，感受对智能社会美好未来的热切期待。

本章知识结构



项目主题

智慧之旅

项·目·情·境

人工智能的历史就是人类在各个领域中不断创新的一个缩影，其间涌现出许多不同的思想、理论、方法、人物、流派、事件和应用。让我们追随大师们的脚步，从智能的本质、人工智能和机器智能的基本内涵等问题开始，探索充满各种奇思妙想的人工智能世界，亲自体验和搭建带有人工智能特征的应用，了解人工智能曲折跌宕、波澜壮阔的发展历程。

学校要举办一次以“走进人工智能世界”为主题的创新科技周活动，要求分“人工智能与人的智能——智能的本质”“图灵测试与智能定义——人工智能的测试”“多样的人工智能观点——人工智能的发展”三个板块，向同学和老师们深入展示人工智能世界的研究问题和应用实践，并设立“身边的人工智能”体验应用展台，设计一些具有吸引力的人工智能应用体验项目，吸引同学和老师们来体验。

项·目·任·务

任务 1

讨论人的智能和人工智能的相同与不同，学习人工智能发展历史中不同的观点和观念，了解不同领域对人工智能概念的理解，总结出本小组认同的人工智能定义。

任务 2

列举用于图灵测试的应用准则，规划自己版本的图灵测试游戏。设计制作介绍图灵测试规则和应用及反对观点的展板，布置体验自己版本图灵测试游戏的展台和道具。完成“图灵测试与智能定义——人工智能的测试”板块。

任务 3

通过搜索相关网站和在线数据库、查阅图书馆和科技馆的资料、采访领域专家等途径，汇集文本、图片、视频、应用软件等资源，设计制作人工智能发展历史大事记图表，作为“多样的人工智能观点——人工智能的发展”的主展板内容。

任务 4

搜集整理生活中、学习中常用的人工智能应用软件或者人工智能工具，布置“身边的人工智能”体验应用展台。

第一节　人工智能：从体验到思考

从人类文明发端之际，人们就开始了对自身“意识”“智能”和“灵魂”的思考和探索，这种对自身的“好奇”和探究，成为科学与宗教起源的重要端倪。人工智能的研究直指人类自主意识层面上的核心要素——智慧、思考和心灵种种，因此该领域中的几乎一切重要思想、结果和产品都会引起强烈反响。正是由于人工智能的这种特别属性和极其广泛的覆盖面，有关人工智能的概念、问题、研究领域、方法和观点的界定才十分复杂和多样化。人们提出了形形色色的观点、假说和解释，但到目前为止，尚没有任何单一的视角和观点能客观完整地描述人工智能。

为了给出人工智能的一个合理界定，首先要了解什么是智能，构成智能的要素是什么。

一、人类智能与人工智能

人类对智能的探索自古有之。在中国古籍《荀子·正名篇》中有这样的描述：“所以知之在人者谓之知，知有所合谓之智。”意思是：人身上所具有的用来认识事物的能力叫做知觉，知觉同所认识的事物有所符合，叫做智（慧）。早在计算机出现之前，人们就一直尝试着发明“能思考、有智慧”的机器，可以帮助解决更复杂的问题，甚至幻想着有朝一日这些机器会比人类更“聪明”。

但总的来讲，我们对人类智能本质的认识还处于很无知的状态：我们不知道智能是如何产生的，不了解什么样的机制对于智能是必须的，更无法制造出像人一样聪明的“机械智能”。即便是对“什么是智能”这样一个根本性的问题，我们至今也没有一个令所有人都满意的答案。

然而，所有这些概念上和实践上的困难都不能阻止我们对“智能”和建造“智能机器”的不懈探索。我们虽然还无法了解智能的本质，但是却能敏锐地察觉和识别出智能行为，这对于研究并制造能展现出人类智能行为和能力的机器似乎已经够用了。

1. 智能与智能行为

既然无法给出智能的完美定义，我们不妨从更实用的角度来界定

智能。如荀子所表达的意思，人们能够说话、学习、思考，并将所学所想付诸行动，通过对话、学习和与周围环境的互动（行动），使自己这些方面的能力不断得到改进提高，这就是“智慧”。人的这种具有智慧的能力，就是“智能”。

1994 年，美国心理学家罗伯特·斯腾博格（Robert J. Sternberg）对智能给出了如下定义：“（个体）智能是个人从经验中学习、理性思考、记忆重要信息，以及应付日常生活需求的认知能力。”

除了个体智能外，人类及其他生物还存在“集体智能”（也称群体智能）的现象。例如，某种群体可能会形成并遵循某些规则。如蚂蚁，个体的蚂蚁很难称为具有智能，但一个蚁群却能呈现出令人惊叹的集体智能行为。

在日常用语中，“人工”带有“人造的”“人工合成的”含义。基于这种含义，对人工智能的一种理解是：研究产生智能行为及能力的机制和理论（理解智能），制造能呈现智能行为和能力的机器（制造智能）。

探 究 活 动　从数据到智慧

在信息技术领域中，有一个经典的“数据—信息—知识—智慧”的金字塔，如图 1.1所示。

请同学们搜索并阅读有关文献，在小组内就以下问题进行研讨交流：

1 这个金字塔的含义是什么？

2 有一种观点认为：智慧就是产生知识、运用知识和创造知识的能力。上述观点与斯腾博格的智能定义有什么联系？

3 在以上关于智慧的观念的基础上，有人给出“人工智能就是关于知识的科学”这一定义，你认为这个定义合理吗？

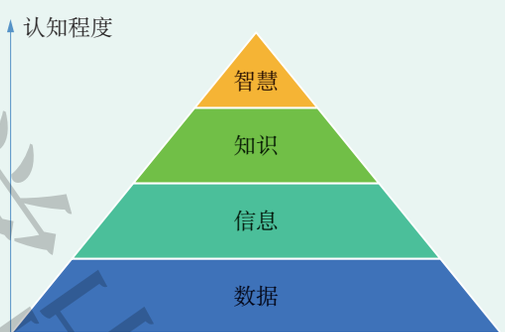


图 1.1　从数据到智慧

那么，什么样的行为或能力可算作是智能的呢？一般认为，（人的）智能是人类个体认识客观事物和运用与事物有关的知识解决问题的能力。因此，人的智能就体现在认识事物和解决问题时所呈现出的各种能力，包括语言能力、数学与逻辑推理能力、沟通交流能力、学习与理解能力、情感感知能力、规划行动能力、创造性思考与问题解决能力等。

体 验 思 考　搭建简易聊天机器人

我们将使用 Python的自然语言处理扩展库 NLTK(natural language toolkit),尝试搭建一个简易的英语聊天机器人,来帮助同学们学习英语。

请同学们完成下列任务:

1 按照课程资源的说明,下载和安装 NLTK。打开并运行配套程序 NLTK_chat_bot.py,创建一个简易聊天机器人。

2 尝试与聊天机器人进行简单的英语会话,体验并初步评估机器人(以及你自己)的英语水平。

3 你认为这样的聊天机器人是智能的吗?为你的结论至少提供三条依据。

4 以小组为单位,阅读 NLTK扩展库的文档,尝试为该程序增加新的功能。

在本活动中,我们使用了 NLTK来使聊天机器人具备自然语言处理的能力。NLTK是一个用于自然语言处理的工具库,提供了分类、标记、词干分离、解析和语义推理等基本文本处理功能,可从 Python程序中直接调用。

尝试搭建并体验了简易聊天机器人 AI 应用后,你认为聊天机器人是否展现出了某些智能行为呢?你认为它有智能吗?

以上所述的聊天机器人程序虽然简单,但其背后的支撑却是人工智能研究者和语言学家们数十年辛勤研究的积累。

概 念 解 释　自然语言处理

自然语言处理(natural language processing, NLP)是计算机与人工智能领域中的一个重要研究方向,其目标是实现人与计算机之间用自然语言(即我们日常使用的语言,而不是符号的或人工的语言)进行有效通信。

自然语言处理大体上包括自然语言理解和自然语言合成两大部分。自然语言处理的研究经历了漫长而曲折的发展过程。人们曾经认为:要想让机器达到人类水平的语言处理能力,机器必须先了解语言的规则(语法的观点),然后需要真正懂得语言所表达的意思(语义的观点)。基于语言大数据(语料库)的统计语言处理方法打破了这种观念上的禁锢,并为后续深度学习方法的应用奠定了基础。语料库汇集了各个行业的语言文字及语言材料,如新闻类、科技类、文化艺术类、体育类等等,是当前支撑自然语言处理研发的基础性大数据,在自然语言处理的发展中起到了十分关键的作用。

图 1.2　AI合成主播

近年来,自然语言处理领域,特别是关于中文自然语言的处理方面,取得了长足进步,在一些方面已经接近人类语言专家的水平,如机器翻译和自然语言理解等。2018年 11月 7日,在第五届世界互联网大会上,全球首个“AI合成主播”发布(如图 1.2所示),实现了 AI语音及视频同时实时合成

的新突破。在实现 AI合成主播的各项技术中，声音合成技术致力于模拟人的声音和新闻播报的特点，是人工智能技术的典型应用。当然，除了自然语言处理技术之外，AI合成主播还涉及人脸合成、运动控制等多项 AI技术。

2. 图灵测试

探 究 活 动　　哪些 App可以算作是智能的？

在智能手机或平板电脑上，像聊天机器人那样声称具有 AI功能的应用比比皆是：有的可支持专门的 AI芯片，有的带智能加速，有的带智能感知，有的带智能拍照，等等。你是否想过，这些应用真的如其所称的那样是“智能”的吗？智能手机的“智能”体现在哪里？又如何衡量其“智能”的水平呢？

请列举出你常用或熟悉的 10个手机 App，将这些 App按照各自所展现出的智能水平从低到高进行排序，然后认真思考并回答下列问题：

1 你依据什么标准来衡量 App的智能水平？请至少给出三条理由来支持你的衡量标准。

2 邀请你的同学和朋友一起来进行同样的探究，分享讨论彼此衡量智能水平的标准，并形成一个大家都认同的标准。

3 根据你们衡量智能水平的标准，现在的智能手机以及智能 App在智能方面还欠缺些什么？

4 根据你们的了解和衡量智能水平的标准，如果将所有手机 App划分为非智能的和智能的两大类，请你估计一下，智能的 App大约占多少比例？

5 平均来看，哪种类型的 App智能水平最高（或最低）？

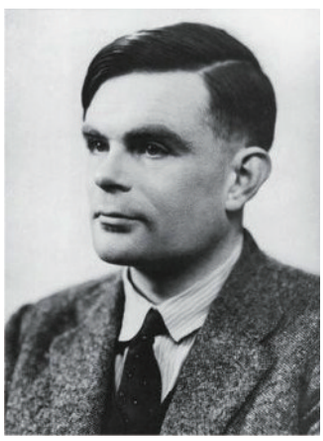


图 1.3 艾伦·图灵（1912~1954）

通过上述的思考和探索，我们发现：判断一台机器或一个程序是否具有智能以及是否具有像人一样的智能绝非易事，而且每个人都会有自己的判断标准。

英国数学家、逻辑学家、计算机科学家艾伦·图灵（Alan Turing）（如图 1.3 所示）在 1950 年发表了一篇名为《计算机器与智能》（*Computing Machinery and Intelligence*）的论文，最早提出机器能否思考，以及如何判断机器智能达到了人类的水平等关键问题。图灵写这篇文章的时候，“人工智能”一词还没有出现，这篇论文的第一句话就是：“我建议考虑这个问题：‘机器能思考吗？’”

图灵在论文中提出了一种“模仿游戏”，用以判断计算机能否在智能行为上表现得和人“无法区分”。这种模仿游戏后来衍生出各种不同的版本，既有严肃的和学术性的，又有商业的和大众化的，统

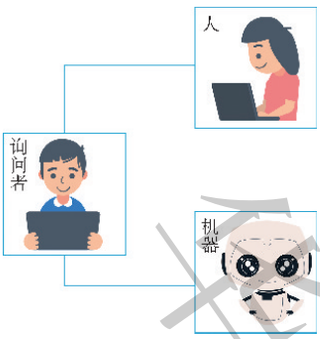


图 1.4　图灵测试

称为“图灵测试”。

图灵测试通过观察机器在回答人们提问时的表现来判断其是否具有智能。假设让一个人与一台机器同在一个房间中，询问者（也是判别者）待在另一个房间中，且看不见两者中的任何一方。询问者通过一个终端对人和机器提问并接收两者的回答，如图 1.4 所示。如果经过对双方足够多次的问询和接收回答后，询问者仍无法分辨出哪个是人，哪个是机器，那么就认为机器是智能的。

图灵在论文中写道：“我认为在大约 50 年的时间里……经过 5 分钟的提问后，一般询问者作出正确判断的概率，不会超过 70%。”

图灵测试被广泛用于判断机器或程序是否具有“人类级别”的智能水平。

规　划　设　计　　规划自己的模仿游戏

图灵所提出的原始模仿游戏其实由两个阶段构成，以上所述只是其第二个阶段。图灵测试的第一个阶段是询问者通过不断的问询来判断面对的人是男人还是女人。假设让一个男人与一个女人同在一个房间中，其中一方，比如男方，希望能够骗过询问者（也就是判别者），让他认为自己是女人，而另一方则要向询问者证明自己才是女人。实验的限制与第二个阶段的情形类似，询问者必须待在另一个房间中，看不见两者中的任何一方，只能通过终端对两人进行提问，并接收两者的回答，如图 1.5所示。如果经过对双方足够多次的问询和接收回答后，询问者仍无法分辨出哪一方是男人，哪一方是女人，那么就认定是男方获胜。

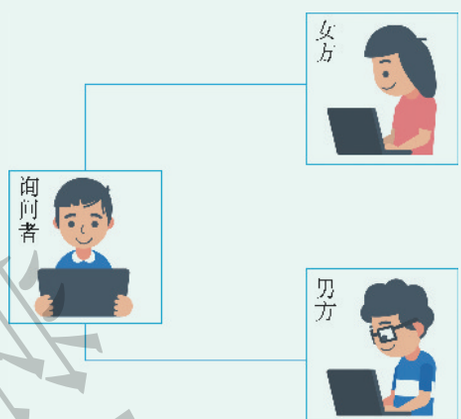


图 1.5　模仿游戏的第一个阶段

1　讨论图灵测试引入第一阶段的目的是什么。后人使用图灵测试的时候经常忽略第一阶段，分析这种忽略对于图灵测试的结果是否有影响。

2　请利用身边的终端，设计一个类似的模仿游戏，不必拘泥于图灵设想的原始形式。注意：你不但需要设计游戏规则和判定胜负的标准，还要事先准备好询问中允许提出的各种问题。

3　游戏中是否应该允许抛开问题清单任意提问？为什么？

4　对游戏规则和问题清单做出改进，使之更有趣味性也更合理。

图灵测试虽说只是一种假想的思想实验，但却具有以下两个显著特征：

（1）对智能的概念给出了一个客观的判定标准，即根据机器对一

系列特定问题的反应来判断它是否有智能行为，从而为判断智能与否提供了一个可操作的标准，绕开了针对智能“究竟是什么”之类的哲学争论。

（2）严格的实验条件限制排除了可能影响判别者得出结论的其他因素，这些因素既包括简单的外观差异，比如生命体征、外观表情和姿态，也包括诸如人和机器的思考方式这样的深层区别等。

由于这两个特征，图灵测试被普遍用来作为判别机器智能及人工智能的智能程度的依据。

图灵测试及其各种推广、限制和变体形式，经常被用来评判现代AI程序的智能水平。2011年，一台名为“沃森”的超级计算机在美国知名的电视智力竞赛节目《危险边缘》中击败了之前这档节目中人类智力竞赛的冠军。这是一个标志性成果，被认为是在智力竞赛这个狭窄的领域内，智能计算机通过了图灵测试。另一个颇具影响的现代图灵测试是1991年开始的“洛伯纳大奖赛”，该比赛每年举办一次，由国际人工智能协会主持并挂靠在人工智能的全球性学术会议上，因此带有某种“专业的”意味。有趣的是，“洛伯纳大奖赛”的目标并非决定参赛的人与机器谁是获胜者，而是判定谁表现得“最具人性”，因此其获胜者被称为是最具人性的“人”。

探 究 活 动　对图灵测试的质疑——塞尔的“中文屋”问题

图灵测试的想法被提出以后，它事实上已经成为判断机器是否具备智能的一个“标准”，但各种反对和质疑声也从未停止过，其中最有影响力的一种反对观点是美国哲学家约翰·塞尔（J.R.Searle）在1980年提出的。塞尔设计了一个叫做“中文屋”的思想实验，以此来证明机器即使能够表现出与人类似的“智能”行为，其实质上也仍然是没有任何思维和感情的“机器”，因此不能认为是“智能”的。

塞尔设计了一个与图灵的模仿游戏类似的场景。如图1.6所示，假设一个完全不懂中文的人被关在一间屋子里，屋外的人通过向屋里传递纸条的方式用中文来提问，屋内的人需要将答案写在纸条上传递出来。屋内提供了一个计算机程序，该程序能够用中文来回答其收到的任何以中文叙述的问题，并指示屋内人如何用中文作答，然后将写有答案的纸条传递给提问者。塞尔问道：如果屋外的人通过这样的问答无法判断屋内人的母语是否为中文，那么屋内的人是不是就该算作懂中文？

图1.6　塞尔的“中文屋”问题

塞尔认为：即使屋内人通过了这样的图灵测试，他仍然是完全不懂中文的。

思考：

1. 塞尔使用“中文屋”问题质疑图灵测试的关键点在哪里？

2. 你认为用“中文屋”问题来反驳图灵测试，其论据是否充分？

也有一些计算机科学家对如何判断“机器是否能思考”这个问题提出过有趣的观点。比如，著名计算机科学家艾兹赫尔·戴克斯特拉 (EdsgerW ybe D ijkstra)就曾评论：问“机器是否能思考”，与问“潜水艇能不能游泳”是类似的。如果将游泳定义为“利用四肢、鳍或尾巴在水中前进”，那么潜水艇肯定不会游泳，因为它既没有胳膊和腿，也没有鳍或尾巴。因此，我们完全可以说机器能“思考”，只不过机器的思考不是我们所设想的人类那种思考。

3. 图灵测试的一个“另类”应用

随着互联网应用的普及，信息安全问题日显突出，日常应用的便利与信息安全成为一对矛盾。通常，由这对矛盾产生的问题都没有绝对完美的解决方法。比如，你在网络上的虚拟私有空间需要用用户标识和密码的组合来进行身份鉴别以保护其安全。想侵入你私有空间的恶意用户经常会利用一些破解工具，在短时间内大量尝试各种密码的组合。在通常情况下，密码越复杂，成功抵御这种被破解的风险的概率就越大，你的网络空间安全也就越有保证。这也是很多应用要求你不能设置太简单的密码的原因所在！

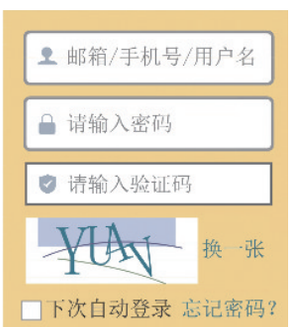


图 1.7　图形验证码

但是，复杂的密码会给我们的日常使用带来不便，尤其是在需要同时使用多个“标识 + 密码”的组合的场合。“图形验证码”为解决这个问题提供了一个简单而颇具创意的思路。同学们应该都熟悉，在很多需要输入用户标识和密码的场合，都会伴随着输入一个图形验证码，如图 1.7 所示。

所谓的图形验证码通常是一幅小图片，里面有几个或歪歪扭扭，或重叠交叉在一起的符号（汉字、字母或数字），你必须要认出这些符号并将其填写在验证码的输入框中。图形验证码的英文名称为 CAPTCHA，它是“completely automated public turing test to tell computers and humans apart”的缩写，大意是“用于区分计算机和人的一种全自动公共图灵测试”。图形验证码所依据的原理正是图灵测试，只不过这时我们的目的不是看计算机是否具有智能，而是正相反：我们相信计算机识别图形中的不规则符号的能力还算不上“智能”，所以不可能在限定的时间内准确识别并填入到输入框中。

如此一来，在短时间内大量尝试各种密码组合也就成为不可能完成的任务了。

注意：上述解决方法只是显著降低了计算机自动快速尝试多个密码的速度和可能性，不能保证绝对的安全。但对于多数普通应用，这样的安全缓冲就足够了。

探 究 活 动　CAPTCHA的变体

随着计算机图像识别能力的提升，辨别 CAPTCHA中的符号成为一个相对简单的任务，因此原始的CAPTCHA验证方式的安全性也受到了挑战。

1 假设图 1.8所示是某企业员工账号的登录界面。你认为这个系统有什么安全隐患？如何加以改进？

图 1.8　账号登录界面

图 1.9　改进的图形验证码

2 为增加安全性，某网站采用了如图 1.9所示的登录方式。这种验证码能否提升安全性能？为什么？

3 当前的很多应用采用短信验证码的方式来改善安全性能。与图形验证码相比，短信验证码的优势在哪里？可能的安全隐患是什么？

二、关于人工智能的观念

随着计算技术的飞速发展，人工智能已经演化成为一个十分复杂的科学与技术领域，横跨计算机科学、生物学、心理学、神经与认知科学、数学和哲学等多个学科，曾经被视为人工智能代言者的智能机器人也已经成为极具发展潜力的庞大产业。与人工智能相关的观点、新闻、影视和文学作品每天都充斥于我们的周围。

“人工智能”一词是在1956年的达特茅斯会议上由约翰·麦卡锡(John McCarthy)提出的,他认为:“人工智能就是制造智能的机器,更特指制作人工智能的程序。人工智能模仿人类的思考方式使计算机能智能地思考问题,人工智能通过研究人类大脑的思考、学习和工作方式,将研究结果作为开发智能软件和系统的基础。”

探 究 活 动　关于人工智能的各种观点

1 智能体的观点

智能体(Agent也称为智能代理)是指能够智能地感知环境,从环境中学习并与环境进行交互的系统(如图 1.10所示)。智能体的观点认为:人工智能的目标就是创建能够执行某类任务,并具备认知功能的智能体。

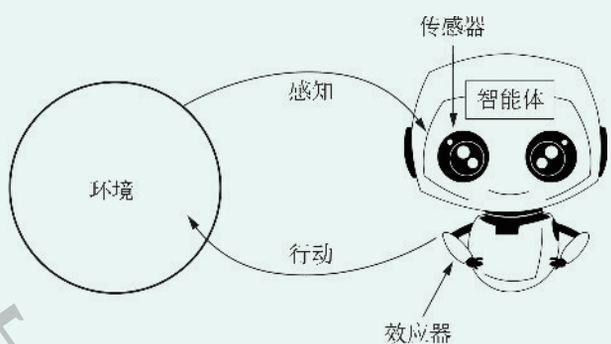


图 1.10 智能体示意图

智能体明确地将“环境”纳入智能系统的组成部分,并特别强调了感知、交互、认知和学习能力是智能的核心。将人工智能视为智能体,既保留了人工智能的整体特色,同时又在很大程度上避免了诸如“中文屋”之类问题的困扰,因此这一观点已成为近年来人们普遍采用的一种观点。

智能体可根据其所呈现的形态大致分成两大类:软件智能体和物理智能体。

思考:

1. 请对比智能体与机器人的概念,指出它们有哪些相同之处,又有哪些不同之处。

2. “清扫机器人”是日常生活中常见的一种智能设备,可以把它看成一个智能体。请详细说明这个智能体的核心组成部分有哪些,它们分别要承担哪些任务。

3. 智能手机是否应该被视为一个智能体?将其看作是多个智能体的组合是否更合适?如果将其视为“多智能体”,那么它最主要的几个部分是什么?

2 人工智能的三种流派

在人工智能的发展历程中,由于对智能本身及产生智能的机理理解不同,曾经出现了很多风格迥异的观念和流派,但总体上来讲大致可以分为三种。

(1)符号主义

符号主义的观点基于所谓的“物理符号假设”,即将物理世界的问题等同于可用符号表示的世界,而智能等同于对符号的所有可能操作。早期的符号推理以及基于规则的专家系统都可以看成是符号主义观念的体现。

(2)连接主义

连接主义的观点认为:大脑是一切智能活动的基础,人工智能应集中关注大脑神经元及其连接机制,只有揭示出大脑结构和信息处理活动的本质,才能在机器上实现对大脑功能的模拟。神经网络研究一直

受到大脑神经元构造的启发和指引，被认为是连接主义最主要的表现形式。

(3)行为主义

行为主义的观点认为智能仅仅体现在感知和行为上，一个机器（或智能体）只要在“感知—行动”上看起来具备智能特征，其内部甚至无需具有知识或推理能力。行为主义采用自底向上的方法构建智能体，即构成智能体的单元本身似乎只有非常简单的“感知—反应”功能，如没有“大脑”的六足机器人。但根据一定的机制将多个单元组合在一起构成的智能体可以呈现出令人惊叹的复杂智能行为。

行为主义观点在自然界的其他生物体智能中找到了某些依据。比如，单只蚂蚁很难说其具有智能行为，但蚁群作为一个智能体能表现出记忆、学习和规划路径等非凡的智能行为。在鱼群、鸟群、蜂群中也可以发现类似的情况。

在行为主义观点影响下，人工智能中也形成了一个子领域——仿生智能（bio-inspired intelligence），其成为计算智能或软计算的一部分。

给人工智能这样一个复杂的、具有多学科属性而且仍在高速发展中的领域下一个完美定义注定是十分困难的，甚至是不可能的。许多学者从不同的视角对人工智能的内涵进行了阐述，适当了解这些观点对于我们全面了解人工智能的特点、认识人工智能的价值和潜力非常必要。

思考：

1. 从学习小组成员所汇集的文献中，列举出至少三种关于人工智能的定义或界定，研讨这些定义或界定之间的关联与区别。

2. 阅读百科全书、权威媒体以及著名人物关于人工智能的观点，结合自己使用人工智能应用的体会，讨论每种观点所强调的和欠缺的方面。

3. 待本书的学习全部完成后，请重新做一次本活动，总结自己在哪些方面的观念有所变化。

本书中，我们采用下列定义：人工智能指由人创造出来的，具有感知、认知、决策、学习、执行和社会协作能力，符合人类情感、伦理与道德观念的虚拟的或人工的系统。

人工智能学科大致分为六个领域（或称分支）：

（1）计算机视觉：研究模式识别、图像处理等子领域和问题；

（2）自然语言理解与交流：研究语音识别、合成、对话等子领域和问题；

（3）认知与推理：研究对物理和社会方面各种知识与常识的辨识、采集、表征、认知以及推理等方面的相关问题；

（4）机器人学：研究各种类人形态或非人形态的实体机器人的设计、机械、能源、控制、运动以及任务规划等方面的问题；

（5）博弈与伦理：研究多代理智能体的交互、对抗与合作，以及机器人与社会融合等问题；

（6）机器学习：研究机器（软件）如何从数据中学习，包括各种基于统计的机器学习模型、基于神经网络的学习模型等问题。

第二节　人工智能的发展历程

人工智能自诞生至今，已经有 60 多年的历史，经历了多次高潮和低谷，堪称是人类科技发展史中一部跌宕起伏的大剧。

一、从早期尝试到符号推理

人工智能诞生于 1956 年的达特茅斯会议。从 1956 年到 20 世纪 70 年代中期的约 20 年时间里，人工智能从萌芽迅速走向繁荣，产生了大量的原创思想与成果，初步形成了自然语言处理、专家系统、定理自动证明与自动推理等特色研究方向，建造了很多经典的智能系统原型。包括后来才得到大发展的一些新兴领域，如神经网络（感知机）及 AI 中的贝叶斯方法等，也在这个时期萌芽，这个时期堪称人工智能“古老的黄金年代”。

总体来讲，这一时期的主流还是基于数理逻辑的符号知识表示与推理，这个特色也成为人工智能第一次浪潮的标志。

这个阶段的主要成果及事件包括：

（1）因编写计算机弈棋程序的需要，提出了各种“智能”搜索算法，其中很多算法现今都已经成为经典。一些算法（比如“深度优先”和“广度优先”搜索算法等）已如此普及，以至于完全不需要再冠以“智能”一词了。

（2）艾伦·纽厄尔（Allen Newell）、约翰·克里夫·肖（John Cliff Shaw）和赫伯特·西蒙（Herbert A. Simon）开发了“逻辑理论家”及“通用问题求解机”，为早期专家系统和符号推理系统奠定了基础。

（3）乔舒亚·莱德伯格（Joshua Lederberg）和布鲁斯·布坎南（Bruce G. Buchanan）等人设计了第一个具有实用价值的、基于知识的专家系统 DENDRAL。

（4）约翰·麦卡锡设计了表处理语言——Lisp 编程语言，阿兰·科莫劳尔（Alain Colmerauer）设计了 Prolog 编程语言，这两种语言是第一批专门面向人工智能的计算机编程语言，其引入的语言范式和特点（如面向对象、逻辑编程语言）成为后世同类语言的典范。后来，在 20 世纪 80 年代，日本提出的第五代计算机研究计划就以 Prolog 语言为基础。

（5）茱莉亚·罗宾逊（Julia Robinson）提出逻辑消解算法，成功解

决了一类符号推理中的关键问题。爱德华·费根鲍姆(Edward A. Feigenbaum)和乔尔·摩西(Joel Moses)开发出了第一个成功的、基于知识的数学推理程序 Macsyma。我国著名数学家吴文俊发明了几何定理机械化证明的“吴氏方法”。

(6) 在 1942 年提出的麦卡洛克-皮茨神经元模型的基础上,弗兰克·罗森布拉特(Frank Rosenblatt)提出了感知机模型,这是最早的一类神经网络模型。

体 验 思 考　Python中的符号数学

符号数学系统是自动推理程序中比较成熟的一类,其中比较有名的商业系统包括 Mathematica(及其后来的智能搜索引擎 Wolfram Alpha)、MATLAB和 Maple,它们基本上都遵循了 Macsyma的模式。

SymPy是 Python的一个开源符号数学扩展库,其开发目的是提供一个功能上可与上述商业软件相匹敌的便利工具。在 SymPy中,“符号”一词与数学中的“未知量”“参数”“变量”这些对象很类似。SymPy可以定义并操作非常复杂的、包含有“符号”的数学表达式和数学函数,从而实现了数学层面的“符号推理”。

SymPy中的符号计算与通常的数学实践非常接近,所以学习使用不会有什么困难。请同学们遵循如下步骤体验 SymPy的一些自动运算和推理功能:

1 使用 import语句导入 SymPy库,具体使用方式依需要而定(见随后的代码)。

2 阅读并在交互模式下运行下列程序片段:

```
# 导入SymPy中的合并(collect)、展开(expand)、分解(factor)和化简(simplify)函数。
from sympy import collect, expand, factor, simplify

# 导入SymPy中用于定义一个符号(Symbol)和多个符号(symbols)的函数。
from sympy import Symbol, symbols

# 导入SymPy中的正弦及余弦函数。
from sympy import sin, cos

# 定义六个数学符号。
x, y, a, b, c, d = symbols('x y a b c d')

# 定义一个含有符号的“代数表达式”。
代数表达式 = 5*x**2 + 2*b*x**2 + cos(x) + 51*x**2

simplify(代数表达式)

factor(x**2 + x - 30)      # 将代数式x**2 + x - 30做因式分解。

expand((x - 5)*(x + 6))    # 展开代数式(x - 5)*(x + 6)。

# 将代数式关于x合并同类项。
collect(x**3 + a*x**2 + b*x**2 + c*x + d, x)
```

```
# 定义一个“三角函数表达式”。
三角函数表达式 = sin(x)*sin(x) + cos(x)*cos(x)

# 在表达式中用x = 5, y = 25代入，并计算精确值。
三角函数表达式.subs({x : 5, y : 25})

# 同上代入，计算结果的近似值。
三角函数表达式.subs({x : 5, y : 25}).n()
```

但是，繁荣中也隐藏着危机。1969 年，马文·闵斯基（Marvin Minsky）和西摩·佩珀特（Seymour Papert）合作出版了《感知机：计算几何导论》。在该书中，闵斯基和佩珀特指出罗森布拉特的感知机（即简单神经网络）模型有重大缺陷，它不能模拟逻辑异或函数（XOR），说明神经网络的计算能力实在有限。这直接导致神经网络的研究陷入低谷，也为若干年后人工智能领域的全面衰落埋下了伏笔。1971～1972 年，斯蒂芬·库克（Stephen A. Cook）和理查德·卡普（Richard Karp）等人建立了计算复杂性理论，明确指出很多计算问题不存在“能行解”，同时也暗示了基于纯符号推理的人工智能之路不可行。1973 年，经过历时近两年的调查评估，英国著名应用数学家詹姆斯·莱特希尔（James Lighthill）提交了一份报告（人工智能历史中称为《莱特希尔报告》），不仅抨击了机器翻译方面的工作，而且对英国政府资助的所有人工智能研究都做出了全面的否定性结论。随后，英国以及其他国家陆续终止或大幅消减了对人工智能研究的资助，人工智能进入了第一个“冬天”。

知 识 延 伸　　**两本著名的“反”AI著作**

人工智能从诞生之日起，就有不少反对者。

1《计算机不能做什么》

早期阶段人工智能的最坚定的反对者之一是美国哲学家休伯特·德雷福斯（Hubert L. Dreyfus），他在 1965年发表了《炼金术与人工智能》一文，对人工智能进行了激烈的批判。1972年，德雷福斯将其批判人工智能的文章汇集成一本书，名为《计算机不能做什么》，并于 1979年和 1992年两次再版，其实两个新版本都没有本质的变化。此书也是人工智能及认知科学发展史上的重要文献，在 20世纪 80年代中期曾经被翻译成中文出版（如图 1.11中左图所示）。

有趣的是，麦卡锡在达特茅斯会议上所提出的“人工智

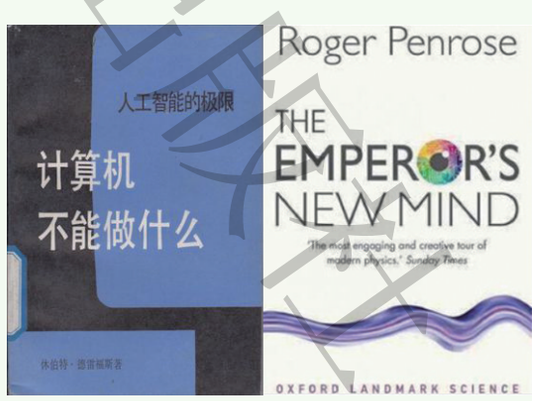


图 1.11 《计算机不能做什么》（左）和《皇帝的新脑》（*THE EMPEROR'S NEW MIND*）（右）

能”这一术语，开始并没有得到学术界的广泛认同和接受，相近领域的研究工作也常被冠以“机器智能”“复杂信息处理”“思维过程机械化”“人工思维”“控制论”等不同的名称。德雷福斯对人工智能的激烈批判反而引起了学术界对这个新兴领域的广泛兴趣，此后“人工智能”一词也得到了普遍接受。

2《皇帝的新脑》

讨论 AI 的另一本著名的书是由杰出物理学家罗杰·彭罗斯 (Roger Penrose)所著的《皇帝的新脑》(*THE EMPEROR'S NEW MIND*)，由牛津大学出版社于 1989年出版(如图 1.11中右图所示，湖南科学技术出版社出版了该书的中文版)。在这本书中，彭罗斯基于对计算机科学、数学、物理学、脑科学和哲学的深邃洞察和理解，对当时甚嚣尘上的“强人工智能”观点提出强烈质疑。彭罗斯认为：正如寓言故事中的皇帝没有穿衣服一样，机器没有头脑，因此机器智能也不可能超越人脑。彭罗斯不认同将通过图灵测试等同于具有机器智能或智慧的简单化理解，他通过一系列的数学、物理学和哲学上的论证，试图说明呈现智能行为与具有智能(如理解力)是完全不同层面上的问题，用图灵测试来检验和定义智能还远远不够。

德雷福斯对人工智能的批判主要是在哲学的层面上，与此不同，彭罗斯对强人工智能的质疑以及他自己的观点有深厚的科学根基。在出版后的五六年中，《皇帝的新脑》甚至间接掀起了人工智能研究的一个小高潮，成为尚处于第二个“冬天”中的 AI领域的一道独特风景。

彭罗斯后来还撰写了《皇帝的新脑》的续篇《大脑的影子》(1994年出版)，回应其他学者的反馈，并继续从更宽广的视角探讨宇宙、人、智能和未来等问题。

二、从专家系统到知识工程

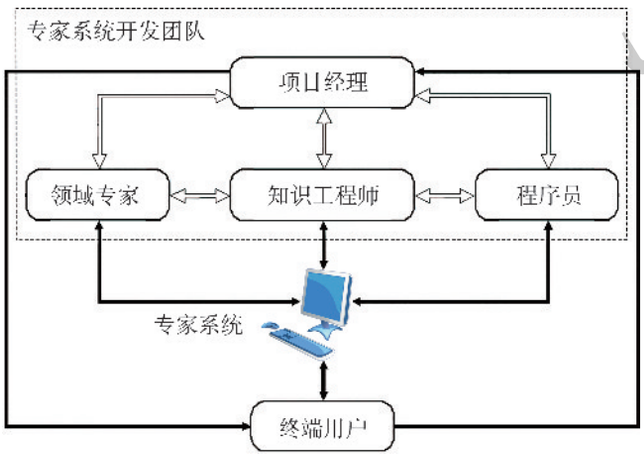


图 1.12　专家系统开发的示意图

从 20 世纪 80 年代初开始，人们认识到，要想让机器能够解决更现实、更复杂的问题，仅有推理的框架是远远不够的，必须为这些框架添加领域“知识”。于是，知识逐渐成为人工智能领域关注的核心，在各种人工智能系统中导入“人类知识”或“领域知识”，使之能解决更实用的问题成为研究主流。专家系统开始以全新的面目复兴，研究者提出了各种“基于知识”的专家系统，包括可处理“非确定性”及“模糊性”知识的专家系统、可“自主进化的”“演化型”专家系统、可“自主学习的”神经专家系统，以及各类“混合型”专家系统等。专家系统开发的示意图如图 1.12 所示。

基于知识的大规模专家系统的开发还催生了“知识工程”这个子领域。

这一阶段最具代表性的专家系统有：

（1）由费根鲍姆主持设计的 MYCIN 和 EMYCIN。MYCIN 是一个用于检查血液传染病的专用系统，后续的 EMYCIN 则是一个通用型专家系统（也称为专家系统外壳），其通过增加某个专门领域的知识库就可以成为专用的专家系统。

（2）由斯坦福研究院开发的用于矿物勘探的 PROSPECTOR。PROSPECTOR 是一个概率型专家系统，其推理机制基于贝叶斯推理，是首批将贝叶斯推理付诸实践的 AI 系统。

据 1994 年的一项调查，当时已经有 2 500 种以上的专家系统投入使用，其中用于商业及制造业领域的系统占到 60%。这标志着人工智能中的专家系统（或知识工程）这个子领域已趋于成熟。在 20 世纪 80 年代初到 90 年代初的这十几年间，人们在人工智能的基础研究方面，包括神经网络、统计学习与并行信息处理等领域，都取得了长足的进步，事实上也为日后机器学习领域的全面爆发奠定了基础。1982 年，日本政府启动了“第五代计算机系统计划”，英国政府也全面恢复了自《莱特希尔报告》以来一直停滞的对人工智能研究的资助，这些都标志着人工智能领域迎来了第二次快速发展。

探 究 活 动　身边的专家系统

将你手机中经常使用的 App做成一个列表，并思考如下问题：

1 哪些 App可以认为是专家系统？

2 那些算作专家系统的 App有哪些共性？又各自有哪些独特之处？

3 专家系统 App知识库的丰富程度是如何影响 App功能的？（举例说明）

但是，由于构成专家系统核心的知识表示与推理机制具有内在的局限性，加之当时在数据来源和计算力方面的严重制约，大规模专家系统的应用开发遭遇瓶颈，第二次人工智能的热潮也逐渐冷却。到了 1995 年前后，人工智能领域再次接近停顿状态，并陷入长达十多年的“AI 冬天”。

知 识 延 伸　知识工程时代的典型——第五代计算机系统计划和 Cyc项目

在人工智能的第二次发展热潮期间，诞生了很多规模庞大的 A项目，其中最有影响力的当属日本的“第五代计算机系统计划”和美国的“Cyc项目”。

1 第五代计算机系统计划

1981年，日本的一个专家委员会向政府提交了一份报告，题目是《知识信息处理系统的挑战：第五代计算机系统初步报告》。报告认为：传统的冯·诺依曼体系计算机已经不能适应未来的知识信息处理的挑战，新型的第五代计算机系统不应再以硬件工艺为主，而应更看重体系结构和软件。报告提出了 6种被认为是更先进的体系结构，其中多数都是当时流行的基于知识的结构。 1982年，日本通产省制订了为期 10年的第五代计算机系统计划，试图以先进体系结构和 Prolog语言为基础，建立知识水平能与人脑相匹敌的计算机，计划总投资预计为 1 000亿日元。

日本的第五代计算机系统计划在世界范围内引起了强烈的反响，美国和欧洲的主要发达国家都先后制订了类似的应对计划。 1982年，美国政府决定成立“微电子与计算机联盟”（Microelectronics and Computer Consortium，缩写为 MCC），每年投资 7 500万美元。后面将要提到的，被称为历史上最大的知识工程项目 Cyc，就是从 MCC开始的。英国于 1982年启动了“阿尔维计划”，欧洲其他国家则是在次年启动了“欧洲信息技术研究发展战略计划”（ESPRIT）。这些国家和地区主导的大型项目，极大地刺激了人工智能的研发，也助推了人工智能的第二次热潮的兴起。

但是，到了 1988年，人们开始认识到，当时的技术水平离达到第五代计算机的要求还甚为遥远。事实上，第五代计算机系统计划没有在任何相关领域取得实质性突破，其也以失败结束。当时的一位知名专家在谈到失败的原因时认为，大部分第五代计算机所做的工作都是试着用逻辑程序去解决其他手段早就已经解决的问题，而不是去尝试解决其他手段不能解决或解决得不好的问题。

当然，第五代计算机系统计划也并非毫无价值，该计划的实施为日本聚集和培养了大批优秀 AI人才，特别是在机器人领域。同时，在第五代计算机系统计划中得到发扬光大的模糊信息处理技术，推动了“模糊”家电在世界范围的广泛流行。

2 Cyc项目

Cyc项目于 1984年始于 MCC，由道格拉斯·莱纳特（Douglas Lenat）领导。该项目最初的目标是将已知的数以百万条的人类常识知识全部编码成机器可读的形式，以形成人类历史上最大的知识百科全书（Cyc就是来源于“encyclopedia”）。莱纳特曾预测：要想完成 Cyc这样庞大的常识知识库系统，至少要涉及 25万条规则，350人花费一年才能做到。事实上，Cyc知识库涉及的编码知识条目数已经上亿，规则有数千万条，而且该项目至今仍未完成。

Cyc项目被称为是“人工智能历史上最大规模的、也是最有争议的项目”之一，特别是在现今的数据爆炸时代，还依靠手工添加所有知识的方式来创建百科全书式的知识库，这很难令人相信。不过近年来，Cyc项目所积累的巨量知识库却为知识图谱技术发展提供了其所必需的资源，这也使该项目有了复兴的机会。

三、从大数据到深度学习

20 世纪 90 年代中后期，受浏览器和搜索引擎技术突破的推动，互联网开始迅速普及并呈现出爆炸式发展，尤其是其后智能手机的迅速普及，海量数据的处理和利用问题日益凸显，但又完全超出人工处理

所能达到的极限，人们对全自动化智能信息处理的需求与日俱增。在这种背景下，以采集、存储及处理海量数据为目标的新一代计算基础架构（如大规模并行处理、高性能计算、大数据、云计算与物联网技术等）和各种“智能化的”信息处理技术（如语义网及本体技术、数据可视化技术、数据挖掘与知识发现和机器学习等）趋向成熟，并迅速显示出巨大的价值，人工智能研发复兴的速度明显提升，并最终在 2012 年开始了全面爆发。基于深度架构的神经网络不仅在 ILSVRC 图像分类竞赛中获胜，而且大幅提升了分类准确度。至此，人工智能进入了第三个发展阶段——机器学习/深度学习阶段。

第三个发展阶段的最典型特征就是以深度学习为代表的一系列突破性算法和应用。深度学习这一概念源于乔弗瑞·欣顿（Geoffrey Hinton）于 2006 年提出的“深度信念网络”，是一种含有多个隐藏层的神经网络结构，“深度”一词即指隐藏层的数目，如图 1.13 所示。

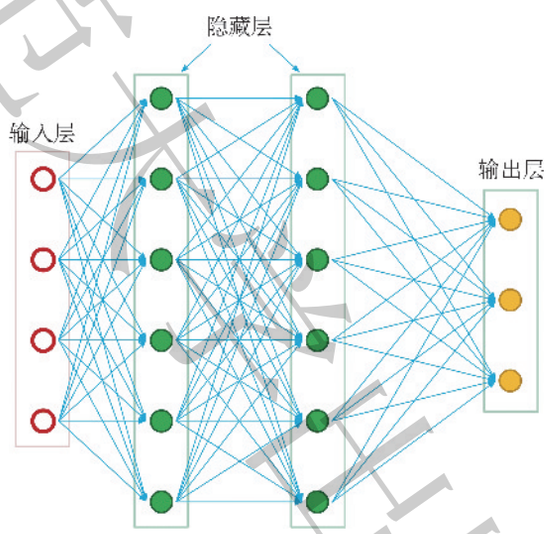


图 1.13　含有多个隐藏层的深度神经网络

深度神经网络通过组合一系列较简单的隐藏层（卷积层和池化层），从简单到复杂、由局部到整体依次提取原始数据中的各类属性及特征，并逐层提高这些特征表示的抽象级别，以得到原始数据的尽可能抽象的特征表示。

例如，如图 1.14 所示，用于识别图像中物体（猫）的深度学习过程带有若干隐藏层，每个层分别提取猫的某个局部特征，然后逐渐组合成完整的猫的特征。

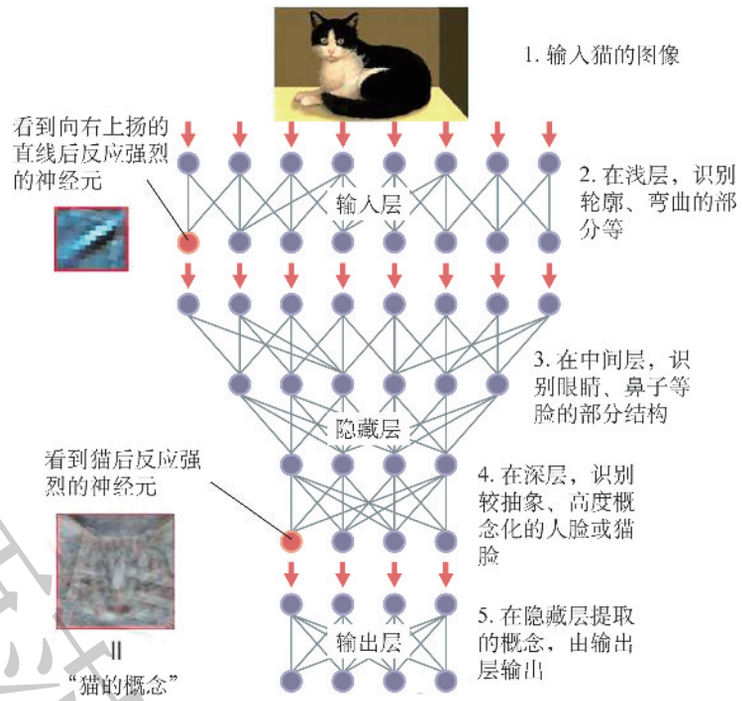


图 1.14　深度学习算法逐层识别图像中物体的特征

事物特征的提取以往都是靠人工来完成的，这种方法不仅低效，而且在多事物、多特征的情况下几乎成为不可能完成的任务。举例来说，在人工智能发展的第二阶段，构建大型专家系统之所以遭遇瓶颈，一个重要原因就是人类积累了大量知识（即使在一个相对封闭的小领域也是如此），如果仅靠手工方式来提取和进行标记的话，很快就会陷入“爆炸”状态。深度学习突破了“特征表示”这个重要的瓶颈问题，是过去几十年间机器学习乃至整个人工智能领域中的重大进展。因此可以说，深度学习事实上开启了人工智能在几乎所有领域和行业各个层次应用的大门。

从 2012 年的标志性事件——ILSVRC 竞赛开始，深度学习先后在自动翻译、自然语言处理、图像识别、人机博弈等人工智能传统领域取得大幅进步，同时也为机器人与智能制造、自然人机交互、自动驾驶、机器应答系统、垃圾邮件过滤、网络安全取证、推荐系统等全新应用领域提供了发展基石和动力。2016 年和 2017 年，人工智能程序阿尔法围棋（AlphaGo）以绝对优势战胜人类顶尖的围棋高手，更是将以深度学习为代表的新一代人工智能的应用水准推向一个历史性的新高度。

从 2015 年开始，一些国际信息技术公司相继发布了自己的人工智能开放平台，我国也宣布进行国家级 AI 行业应用开放平台建设，一

批新兴 AI 公司推出了各具特色的人工智能应用开发平台，这些为开展人工智能在各个行业的创新应用提供了无限的可能。

人工智能第三次浪潮的兴起很大程度上是因为克服了第二阶段发展中所遭遇的主要瓶颈问题，因此也是直接奠基于传统人工智能（相对于新一代人工智能而言）的研发积累之上。具体地讲，新一代人工智能的兴起依赖于三个紧密关联的方面：基于深度学习的新一代机器学习架构和算法，各行业数十年信息化及因特网发展所积累的大量数据，廉价通用的并行 GPU（图形处理单元）被广泛应用于训练学习模型。

深度学习算法虽然在一系列问题上取得了突破，但其本身还有很多局限性和未解决的问题，比如对数据集的敏感依赖性、基于小数据及无标记数据的无监督学习算法进展迟缓、深度学习应用领域的不可迁移性等。此外，深度学习算法很大程度上是基于经验的，尚未建立稳固可靠的数学基础，甚至无法解释某些很基本的现象。比如，一些看似通用的深度学习算法为什么对一类应用高效，而对另一类应用却很低效。

为了解决深度学习存在的问题，人工智能科学家们开始构想未来人工智能的形态，并尝试各种新型体系及模型，以突破和超越深度学习的原有框架。例如，引进强化学习及对抗网络，增强深度学习的性能及扩大应用领域。欣顿本人在 2017 年提出了一种基于“胶囊”的全新理论体系，试图攻克无监督学习问题；因发明“因果推理”而获得图灵奖的朱迪亚·珀尔（Judea Pearl）及其合作者则倡导采用“因果推理及解释”机制，使机器智能超越目前机器学习基于数据相关性的“数据推理”范式，让人工智能从大数据、小任务的架构逐渐走向小数据、大任务的架构。

体 验 思 考　体验现代机器翻译

本章前面提到，20世纪 70年代初的《莱特希尔报告》曾对当时的机器翻译所做的承诺和实际水准提出猛烈抨击，那么过了几十年后，现代的机器翻译和一般自然语言处理的水平提升到了怎样的程度呢？

在本章第一节的“搭建简易聊天机器人”活动中，我们已经实现了一个可以用简单自然语言进行交流的机器人例子。下面我们再做一个现代机器翻译的任务。

1 准备好一幅包含待翻译文字的图片，比如图 1.15。

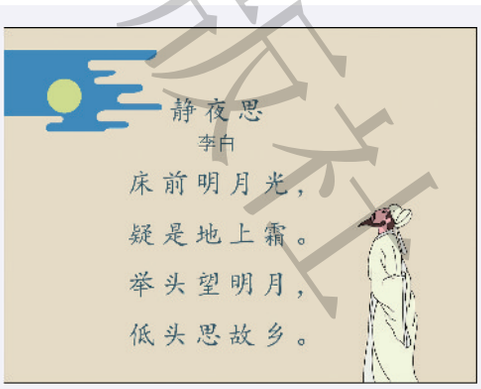


图 1.15　包含待翻译文字的图片

2 打开一个在线自动翻译系统，设置原始文档语种（选择中文或自动）及翻译后的语种（选择英语），将图片直接拖入输入框中（如图 1.16所示），右侧的输出框中将显示翻译后的内容。

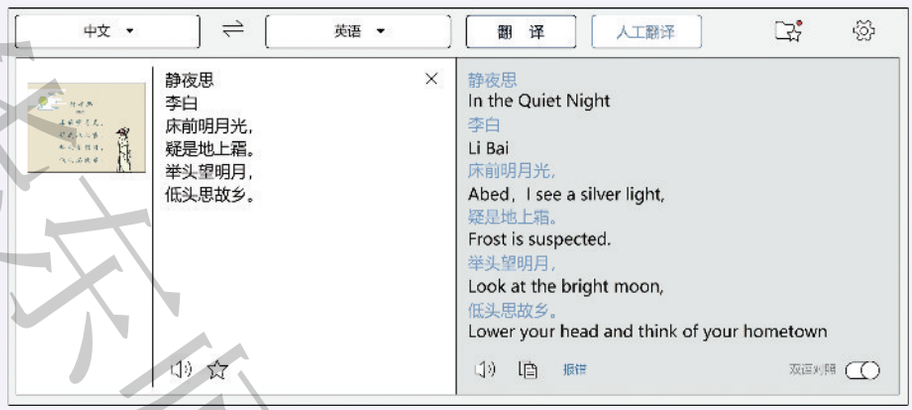


图 1.16 在线自动翻译系统

3 尝试使用系统的其他选项，如直接在输入框中输入文字、变更翻译语种等。

4 建议同学们在使用机器翻译之前，先自己翻译一下，然后将你翻译的结果与机器翻译的结果进行对比。

5 大家相互交流讨论：你认为机器翻译的水平如何？能否达到或高于普通人的水平？

编 程 实 践 使用 Python调用语音识别功能

我们将通过调用 AI平台的功能，实现一个简易的语音识别小工具。

1 设计语音识别工具应具备的功能，规划所需要的资源（素材及 AI功能等），明确要完成的开发任务，如图 1.17所示。

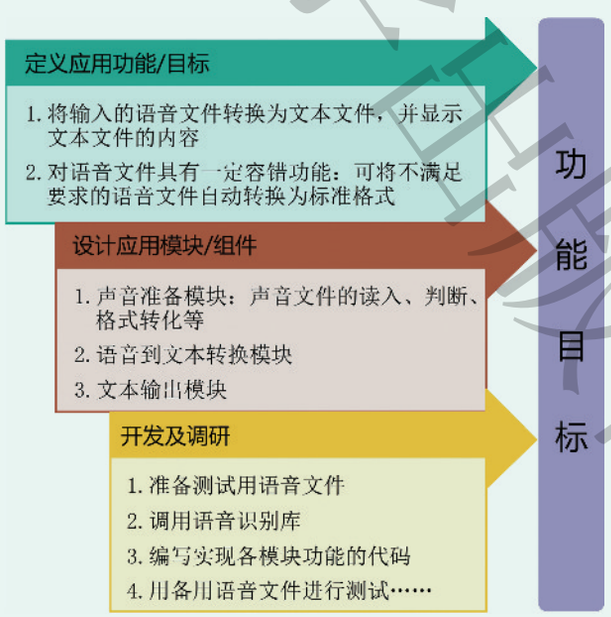


图 1.17 语音识别小工具的设计

2 从课程资源包中下载本活动的压缩文件包，其中包含 Python程序脚本文件 AipSpeech.py、说明文件和示范用语音文件。

3 在联网状态下进入命令行环境，使用如下命令安装所需要的 AI库。

```
pip install baidu-aip
```

4 在命令行环境下运行如下命令，检验语音识别功能实现的效果，其中 testxx.wav是课程资源包所提供的样例语音文件，你也可以使用自己准备的语音文件。

```
python AipSpeech.py testxx.wav
```

第二章

人工智能实现

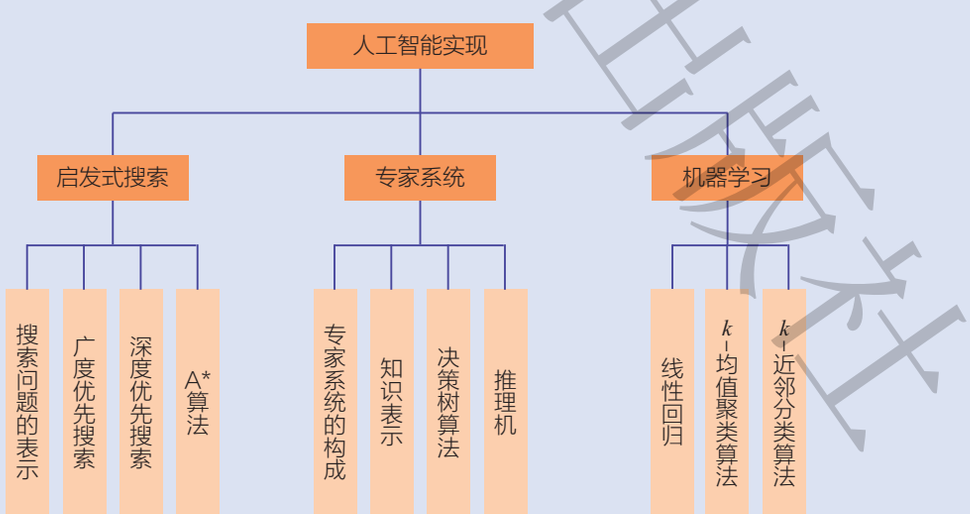
本章学习目标

- 了解搜索问题的形式化表示，掌握广度优先搜索、深度优先搜索、A* 算法的基本原理，用编程语言实现一个简单的启发式搜索算法。
- 理解专家系统的基本概念，掌握运用决策树算法自动构建知识库的方法，了解知识推理的方法及其在专家系统中的应用。
- 了解机器学习的基本概念，理解机器学习技术分类方法，掌握 k-均值聚类算法和 k-近邻分类算法。

人工智能已历经六十余载，其思想方法愈来愈深刻地影响着人类对科技、对社会以及对自身未来命运的认知。与此同时，人工智能应用也逐渐成为我们每个人生活中不可缺少的一部分。

本章以“创新园区自由行”的情境为背景，选取了参观中经常遇到的路径规划、主题展馆选择、集合地点选取等问题，将人工智能领域的基本理念、典型算法设计过程有机地融入问题求解中，并使用现有的数字化工具和编程方法实现问题求解的关键环节，使同学们能够在实践中学习人工智能方法，在应用中掌握人工智能工具。

本章知识结构



项目主题

创新园区自由行

项·目·情·境

学校组织同学们参观新近落成的人工智能创新园区。园区内空间开阔，环境优美，上百家公司已经入驻。同学们将要参观的若干主题展馆分散矗立在园区的不同地方。园区提供了自动导航、智能推荐等小程序，方便同学们自行安排参观行程。

项·目·任·务

任务 1

以规划创新园区的参观路径为背景，设计并实现一个启发式搜索算法。

任务 2

以主题展馆的个性化推荐为例，设计并实现一个决策树推荐算法。

任务 3

以确定集合地点为例，设计并实现一个 k-均值聚类算法。

第一节　启发式搜索

搜索是人工智能最基本的一类问题求解方法。最简单的搜索算法就是逐一列举出所有可能性，直到找到问题的解，因此被称为盲目搜索或蛮力搜索。对于复杂一点的问题，可能的状态数目通常非常庞大，盲目搜索基本上是不可行的。启发式搜索算法则是利用经验性法则帮助问题求解。这些经验性法则没有定式，因此启发式搜索算法通常要根据不同的问题场景来进行针对性的设计。

体　验　思　考　　**规划创新园区的参观路径**

人工智能创新园区鸟瞰图如图 2.1所示。图中标记的圆圈内的字母代表某个展馆，其中 A代表科普展馆，B代表生物科技展馆，C 代表智慧城市展馆，D代表自动驾驶展馆，E代表电子商务展馆，F代表金融科技展馆，G 代表无人机展馆。如果两个展馆之间有道路直接相连，则图中就用白色的线标示出来。如果两个展馆之间没有道路直接相连，就要选择从其他道路绕行。例如 E 馆和 D 馆之间无道路直接相连，我们可以选择“E 馆→A 馆→B 馆→D 馆”的路径，或者“E 馆→A 馆→B 馆→C 馆→D 馆”的路径，或者其他可行的路径。

图 2.1　人工智能创新园区鸟瞰图

如果有道路直接相连的展馆之间步行所需的时间是已知的，如何规划一条从 A馆到 G 馆的可行路径呢？能否找到一条从 A馆到 G 馆的最快步行路径呢？

一、搜索问题的表示

要解决路径规划问题，首先需要用精确的形式化语言（如数学语言）对“问题是什么”，以及“什么构成问题的解”等进行严格表述。问题的形式化表示是问题建模的第一步。

下面我们就将创新园区的路径规划问题形式化，即仅保留与解决问题有关的要素。在规划创新园区参观路径的场景中，假设某个地点的具体大小、形状外貌、道路曲直这样的物理信息，对于解决问题没有影响，则可以将地点抽象为小圆圈，连接两个地点的道路抽象为一条线段，并且忽略圆圈的大小、形状，以及连线的画法对问题的影响。这

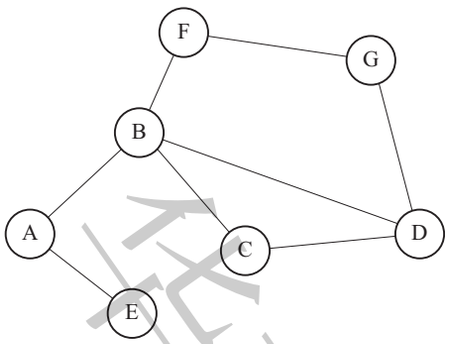


图 2.2　用图表示展馆之间的道路连接

种由若干顶点(也称为节点)以及顶点之间的一些连线(称为边或弧)定义的表示方法就是数学中的图。例如,图 2.2 就是图 2.1 的图表示。在一个图中,我们集中关注顶点与顶点之间的连接状况,而不关心图具体是什么形状。

接下来,我们还要将问题所涉及的其他要素进行形式化。

1. 状态

状态用来描述问题中的各种场景,所有可能的状态构成状态空间。初始状态是问题开始时的状态,终止状态则是问题得以解决时的状态。例如,在创新园区参观路径规划问题中,如果你当前在 A 馆,就可以用符号表示为 In(A)。在本例中,定义的状态空间是有限的,只考虑参观者出现在七个展馆中的某一个展馆的情况,并不讨论同时出现在其中两个的情况,这一点是与实际问题相符的。同时,我们也不讨论参观者不在任何一个展馆的情况,而这一点就是对实际问题的简化了。

2. 行动

行动(也称为动作)可以导致状态的转移,即从一种状态转换到另一种状态。例如,如果你实施了从 A 馆走向 B 馆这一“行动”,就会发生状态转移,即从 In(A)转移到 In(B)。所有可能实施的行动和行动的组合构成了行动集合。例如,如果你处于 In(A)状态,先走到 B 馆,然后又走到 C 馆,那么这两个连续的行动就记为{Go(B), Go(C)}。

3. 状态转移

如果从状态 In(A),执行行动 Go(B),进入新状态 In(B),这个状态转移的过程可以形式化描述为

$$\text{Result}(\text{In}(A),\text{Go}(B))=\text{In}(B)$$

4. 目标测试

目标测试就是确定给定状态是否为目标状态。在本例中,目标测

试就是检验当前状态是否为 In(G)。

5. 路径耗散

路径耗散就是通过该路径的消耗。消耗的表现形式因问题而异，可能是时间的消耗，也可能是距离的消耗等。在本例中，我们假设从任意一个展馆步行到与其有道路直接相连的展馆所需要的时间都是已知的，比如可以从指路牌上看到，则将步行时间定义为相连展馆之间的耗散值是一个不错的选择。

例如，假定从 A 馆到 B 馆所需时间为 10 分钟，则一个单步耗散可表示为

$$c(In(A), Go(B), In(B)) = 10$$

多步耗散值，可以简化为单步耗散值的和。假如要从 A 馆通过 B 馆到 F 馆，从 B 馆到 F 馆所需时间为 8 分钟，则该路径的多步耗散为

$$\begin{aligned} &c(In(A), \{Go(B), Go(F)\}, In(F)) \\ &= c(In(A), Go(B), In(B)) + c(In(B), Go(F), In(F)) \\ &= 10 + 8 = 18 \end{aligned}$$

项 目 实 践　　参观路径的形式化表示

通过上述过程，我们将从 A 馆走到 G 馆这样一个简单规划问题的场景抽象成图 2.3。

与图 2.2 有所不同的是，图 2.3 中各条边上都标记了数值，这些数值分别表示从一个顶点到另一个顶点的单步耗散。现在，问题的解就是一个能将状态 In(A) 转移为状态 In(G) 的行动序列。

1 验证 {Go(B), Go(F), Go(G)} 是一个能到达解状态的行动序列。

2 计算上述行动序列的耗散值。

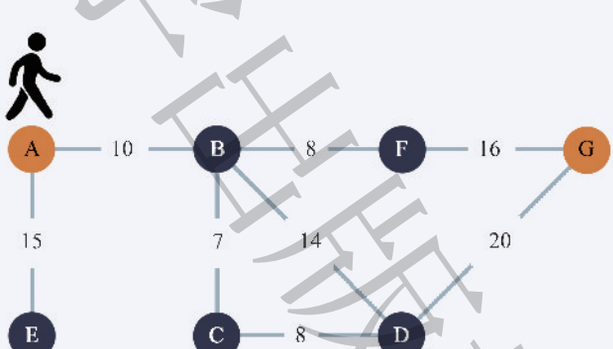
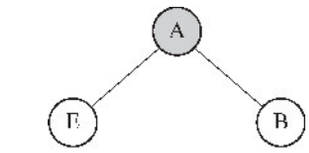


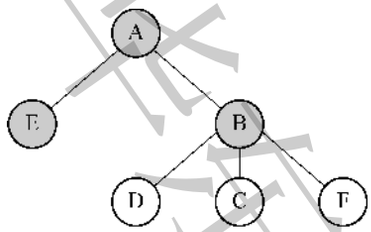
图 2.3　参观路径规划问题的形式化表示

二、广度优先搜索

基于问题形式化的结果，我们需要找到一个行动的序列，从初始



第一步扩展，产生两个子节点 B 和 E



继续扩展节点 B，产生三个子节点 C、D、F

图 2.4　从根节点开始向终止节点扩展

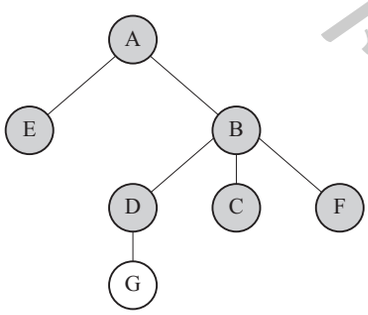


图 2.5　节点 D 的下一个扩展节点是目标节点

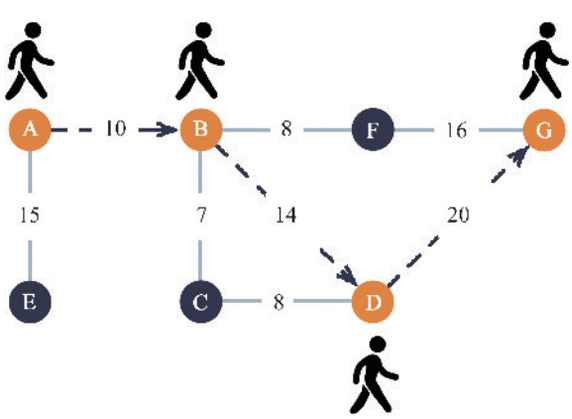


图 2.6　广度优先搜索找到的解路径

状态转移到目标状态。搜索树算法以节点代表状态，以分支代表可能的行动，建立一个树状结构，寻找可能的解决方案。我们以搜索树的根节点代表初始状态，分支代表基于当前状态的可能行动，设计搜索策略，决定在当前状态下采取行动的顺序，直至到达目标状态。本节将以经典的广度优先搜索为例，介绍搜索树的建立方法以及如何应用搜索树来求解问题。

广度优先搜索策略是在寻找从初始状态到目标状态的路径时，每一步都是先向尽可能广的方向扩展，等这一步所有可能的方向都走遍了，再进入到下一步。我们从初始状态 In(A)开始，这就是搜索树的根节点。该状态非目标状态，因此需要行动以转移到下一个状态。此时可作用的行动有两个：Go(B)、Go(E)。这两个行动的目标状态就是根节点 A 后的两个新节点，如图 2.4 上图所示。

经检验，In(E)和 In(B)都不是目标状态，需要进一步再做展开。在广度优先搜索中，优先扩展离当前节点最近的未扩展节点，本例中，节点 B 和节点 E 都在节点 A 的下一层，我们可以随机选择扩展节点 B 或节点 E。假如我们先扩展节点 E，会发现节点 E 无法再继续扩展。在这种情况下，节点 B 是距离当前节点 A 最近且尚未扩展的节点，因此扩展节点 B，得到图 2.4 下图所示的结果。

依照前面的搜索策略，继续扩展以后的层，直至到达目标状态，或者所有的节点都已扩展过，如图 2.5 所示。

这样，我们就通过广度优先搜索找到了一条从初始节点 A 到目标节点 G 的路径 A→B→D→G，或者表示为行动序列{Go(B)，Go(D)，Go(G)}，该路径的耗散值为 44，如图 2.6 所示。

体　验　思　考

用广度优先搜索找出其他的解路径。

三、深度优先搜索

深度优先搜索策略是：在扩展根节点后面的子节点时，每次都选择距离根节点最远的未扩展节点，如果没有能继续扩展的节点了，则返回根节点重新开始，直至到达目标状态或所有节点都已扩展完毕。

深度优先搜索的第一步与广度优先搜索的情况完全一样，我们通过展开离根节点最远的下一层节点 B 和节点 E，得到图 2.7。具体过程不再重复。

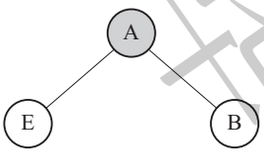


图 2.7　扩展根节点 A，产生子节点 B 和 E

In(E)和 In(B)都不是目标状态，需要继续扩展。但我们是从状态 In(B)开始继续搜索，还是从状态 In(E)开始继续搜索呢？深度优先搜索策略是每次都扩展距离根节点最远的未扩展节点，节点 B 和节点 E 都在根节点 A 的下一层，深度相同，都等于 1。所以，可以随意选择先扩展哪个节点。

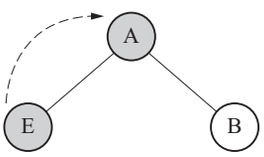


图 2.8　节点 E 无后续节点，回溯到上一层节点 A

假如我们先扩展节点 E。但节点 E 没有后续节点了，所以我们重新回到其上一层节点 A，继续来扩展节点 B，如图 2.8 所示。

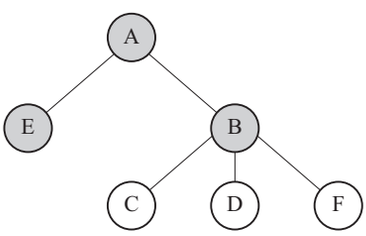


图 2.9　扩展节点 B，产生子节点 C、D、F

扩展节点 B 产生三个新的子节点：C、D、F。这三个子节点都不是目标节点，且深度相同，因此可以从任意一个节点开始继续扩展，如图 2.9 所示。

按照类似的步骤继续进行，直到遇到目标节点 G，如图 2.10 所示。

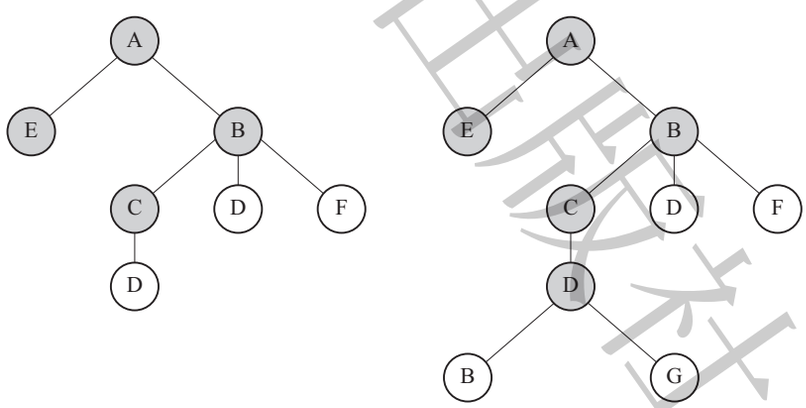


图 2.10　继续扩展节点 C(左)、节点 D(右)，直到产生目标节点 G

体 验 思 考　　深度优先搜索的路径耗散

根据图 2.11中所示深度优先搜索的解路径，计算路径耗散。

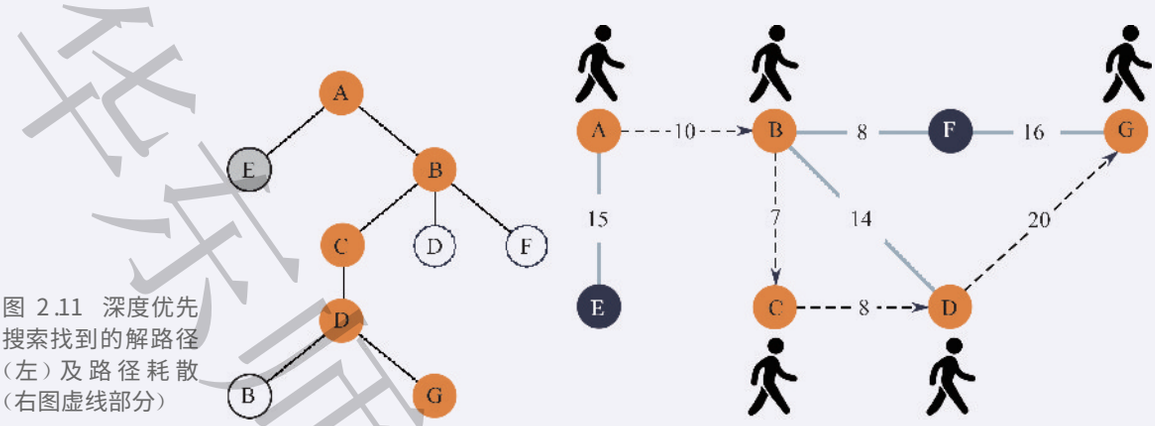


图 2.11　深度优先搜索找到的解路径（左）及路径耗散（右图虚线部分）

广度优先搜索和深度优先搜索反映了人工智能技术两种不同的设计思路。广度优先搜索强调的是齐头并进的设计理念，即每次算法向前推进之前都查找了所有的可能性，主要应用于小到中等的数据集，并且如果存在解决方案一定需要找到的场景。深度优先搜索强调的是一条路走到底的设计理念，如果在查找过程中碰了壁，例如走到头都没有看到目标状态，这时再回头查看其他的解决方案。它主要应用于比较大的数据集，并且对于找到的解决方案要求不是很严苛的场景，比如很多搜索引擎就会采用深度优先搜索。

四、A* 算法

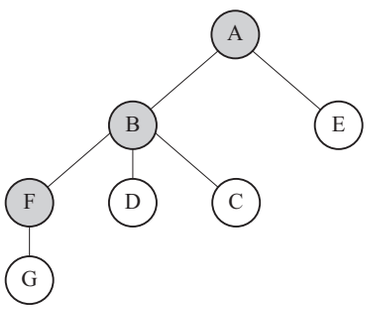


图 2.12　深度优先搜索树找到最优解的例子

广度优先搜索和深度优先搜索主要根据尚未搜索到的节点的深度来决定扩展顺序。只要起始节点和目标节点之间存在可通行的路径，广度优先搜索总能找到一条解路径，但这条解路径不一定是耗散最小的。深度优先搜索不能保证一定能找到解路径，但在“运气好”的情况下不仅能找到解路径，而且还能找到耗散最小的路径。

例如，如果我们在展开节点 B 的后续节点时，选择先展开节点 F，则我们得到一个最佳路径 A→B→F→G（如图 2.12 所示），其耗散等于 34。

广度优先搜索和深度优先搜索只使用了相连展馆间的局部信息，却没有利用有可能获得的额外的全局信息。比如，我们可以按照地图先估计每个节点到目标节点的直线距离和步行时间，而不管它们之间是否有直接相连的道路。根据经验，如果两个展馆之间直线距离比较短，那么从其中一个展馆到另一个展馆的实际步行时间也常常比较短。虽然这种信息是间接的，即不完整的，但也为问题求解提供了更多的线索。我们把这种经验法则叫做启发式规则。

A* 算法就是利用这种额外的启发式规则来“指导”路径展开的次序，更快更好地找到问题的解。

如图 2.13 所示，我们在图的右侧提供了启发式规则，即从某个节点到目标节点之间的直线步行时间。本例中的启发式规则是这样计算的：先根据不同场馆所在的位置计算两个场馆的直线距离，再用直线距离除以人行走的速度，得出两个场馆之间直线行走时间的启发式信息。

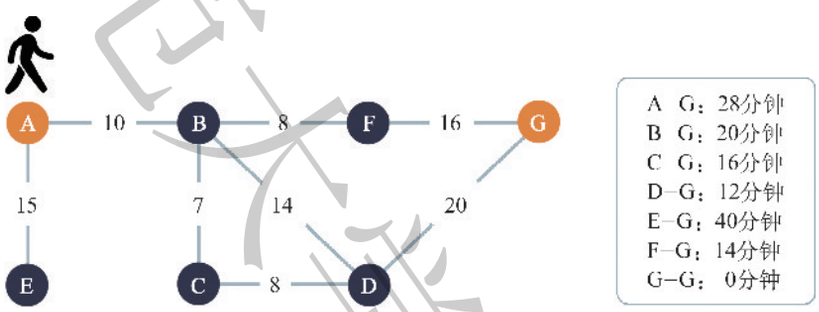


图 2.13　给路径规划图中加入启发式信息

A* 算法的具体实施步骤与广度优先搜索或深度优先搜索非常类似，只不过 A* 算法不是依据节点的深度来决定下一个要展开的节点，而是根据后续节点与目标节点的估值表来决定展开的次序。

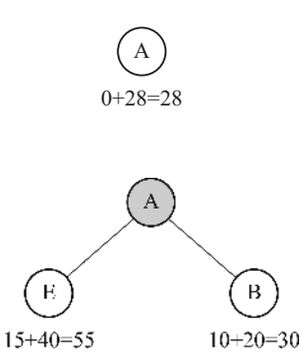


图 2.14　A* 算法的根节点（上）与展开节点 A 的后续节点 B 和 E（下）

（1）将初始状态 In(A) 设为根节点，并将该节点与目标节点的估值标记在下面，如图 2.14 上图所示。因为该状态不是目标状态 In(G)，所以要展开其后续子节点 B 和 E，并将从节点 A 经过每个子节点到达目标节点的估值标记在该节点的下面，这个估值是子节点到目标节点的估值加上其与根节点的实际值，如图 2.14 下图所示。

（2）In(E) 和 In(B) 都不是目标状态，需要继续展开其后的子节点。但从哪个开始继续搜索呢？A* 算法的策略是：每次都先扩展总耗散估值最小的节点。于是，我们选择从总耗散估值最小的节点 B 开始做下一步扩展，产生下一层子节点 C、D、F，如图 2.15 所示。

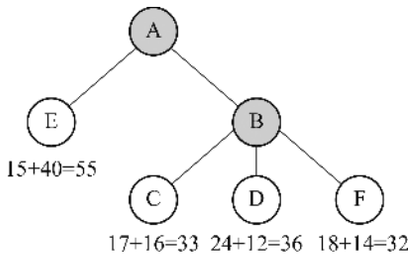


图 2.15　选择总耗散估值最小的子节点进行扩展

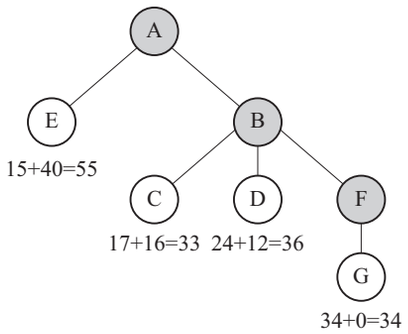


图 2.16　继续选总耗散估值最小的子节点进行扩展，找到目标节点

（3）与上一步类似，我们再选择节点 B 的总耗散估值最小的子节点 F 进行扩展，得到的结果如图 2.16 所示。A 点到 G 点的总耗散估值就等于 A 点到 B 点、B 点到 F 点、F 点到 G 点的实际耗散加上 G 点到 G 点的时间估值。

现在，目标节点 G 虽然已经出现，但 A* 算法还不能结束。因为根据 A* 算法的规则，还要继续选择总耗散估值最小的节点继续展开，而节点 C 的总耗散估值小于节点 G 的估值。

（4）继续扩展节点 C，直到所有叶子节点的总耗散估值都大于目标节点的总耗散估值，搜索停止，如图 2.17 所示。

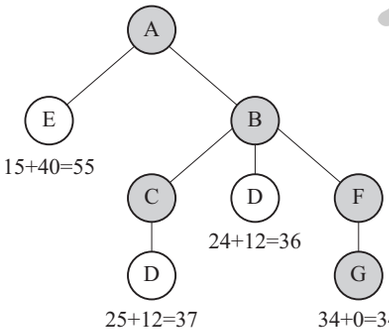


图 2.17　直到所有叶子节点的总耗散估值都大于目标节点的总耗散估值，搜索停止

这样，我们通过 A* 算法找到了一条从初始节点 A 到目标节点 G 的解路径 A→B→F→G（如图 2.18 中左图所示）或行动序列{Go(B)，Go(F)，Go(G)}，其路径耗散为 34 分钟（如图 2.18 中右图所示），这是最佳路径。

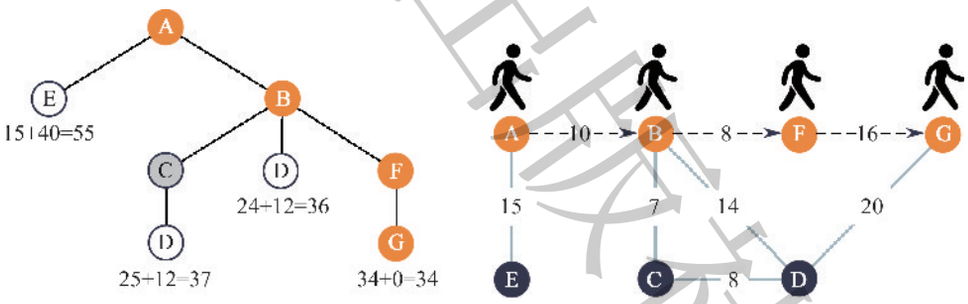


图 2.18　A* 算法找到的解路径（左）和解路径耗散（右）

虽然我们开始估计的 A 馆到 G 馆的步行时间并不太准确，但通过这个整体估值的确找到了花费时间最短的步行路径，这就是很典型的“启发式信息”。当然，A* 算法并非在所有情况下都可以保证找到最优解，例如启发式信息定义不当就可能会影响到计算的结果。

项 目 实 践

1 用 Python语言实现本节中的任意一种搜索算法。

2 在本节的 A* 算法案例中，如果开始时的时间估计值与实际值相差很大，实际实施搜索时可能会产生什么结果？

3 图 2.19展示的是上海地铁线路的局部。如果你想从徐家汇出发去陆家嘴，以途经车站数目最少为最优，你该如何选择换乘路线？若以换乘次数最少为最优，又该如何选择换乘路线呢？

图 2.19 上海地铁线路局部图

作业练习

8- 数码游戏

8-数码游戏的棋盘内有 9个方块，其中 8个方块分别放置数码 1~8，留下一个空格。游戏初始布局（即 9个方块的排列）是随机的，目标布局是空格在中间，且数码 1~8按顺时针方向排列。游戏规则是：空格周边的任何数码都可以滑动，每滑动一次就产生一个状态。

假设初始状态和目标状态如图 2.20所示。若用 A* 算法来求解这个问题，该如何定义启发式信息呢？可以这样定义：假设对每一个数字而言，如果该数字不在目标状态，那么其在初始状态的基础上至少需要移动一步才可以到达目标状态。在图 2.20中，数字 1、2、8都不在目标状态，它们都至少需要移动一步；数字 3、4、5、6、7在目标状态，它们都至少需要移动 0步。因此，完成游戏总共至少需要 3步。是否还有其他可能的定义方法呢？同学们可以思考并讨论。

初始状态 目标状态

图 2.20 8-数码游戏

知 识 延 伸

旅行商问题

在计算机科学中，与寻找路径有关的最有名的难题是旅行商问题（即 Traveling Salesman Problem，简称 TSP问题），也称为旅行推销员问题、货郎担问题、环游城市问题等。TSP问题叙述如下：有若干个城市，任何两个城市之间的旅行距离都是确定的，旅行商从某个城市出发，如果他必须途经所有城市，而且每个城市只能去一次，最后再回到出发的城市，如何规划出一条最短路径？图 2.21是 5个城市的 TSP问题的简单示意图。

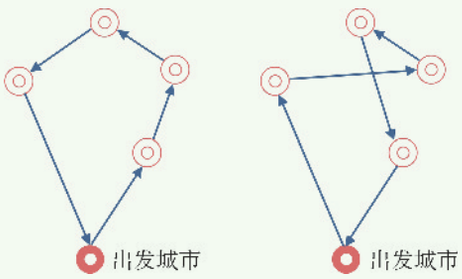


图 2.21 5个城市的 TSP问题（左图的路程小于右图的路程）

初看起来，TSP问题和本节讨论的最短路径问题有些相似，但事实上 TSP问题要难得多。例如，以上面的 5个城市的 TSP问题为例，假设任意两个城市都有道路连接，那么可选的路径总数为 $4\times 3\times 2\times 1=24$条。如果将城市的数目增加到 100个，那么总的可选路径大约是 9.33×10^{157} 条。如此巨大的计算量，使得我们必须要思考有没有更智能的方法来提高搜索的效率。对于像 TSP问题这样的问题，简单的启发式搜索方法（包括 A^* 算法）没有太大的价值。

TSP问题虽然难，但其求解算法有很大的理论和应用价值，所以备受关注。至今，我们仍无法为一般 TSP问题找到一个低计算量的最优解，但使用与启发式算法相类似的智能算法可以找到近似最优解，即使这些解只是近似解，却也有非常广泛的应用。

从理论的观点看，TSP问题之所以重要，其主要原因是：TSP问题是一个所谓的 NP完全问题。

那么，什么是 NP完全问题呢？

一般来讲，可以通过算法来求解的问题大致分为两类："易解"的问题和"难解"的问题。如果可以找到一个问题的算法，使得对给定输入尺度为 n的问题实例，求解该问题实例的算法所需要的时间（步骤）尺度不超过 n的某次幂，此时我们称该问题属于 P类（P代表多项式）。属于 P类的问题被认为是"易解的问题"，其存在可在"多项式时间内"完成的求解算法，这种算法也称为多项式时间算法。

例如，将两个 n位数 x和 y相加的算法，其步骤最多不会超过 n的某个一次多项式 c_0+c_1n（常数 c_0 和 c_1 的具体值并不重要）。请同学们想一想：两个 n位数 x和 y相乘的算法是否为多项式时间算法？

还有一类问题被称为 NP类（NP代表非确定性多项式），被认为是"难解的问题"。一个问题属于 NP类的意思是：我们可以在多项式时间内检验某个解是否正确，但不知道是否有多项式时间的解。

是否有 P等于 NP？这是计算机科学领域中最著名的未解决难题之一。1971年，计算机科学家库克和卡普等人发现了一个惊人的事实：在 NP中存在这样一类问题，如果能为这类问题中的任何一个找到多项式时间算法，那么也就能为 NP中的所有问题找到多项式时间算法。具有这样属性的 NP问题被称为 NP完全问题，TSP问题就是一个最著名的 NP完全问题，另一个人们熟知的 NP完全问题是 Windows中经典的扫雷游戏。

人们普遍认为 NP类不等于 P类，但这仅仅是出于对算法复杂程度的感觉和猜测，要严格证明这个结论尚缺乏充分的论据。

第二节　专家系统

人类在解决问题时需要知识，例如一般的常识性知识和与问题相关的专业知识等。与知识有关的内容是人工智能重要的组成部分之一。在人工智能领域中获取知识、使用知识的方法和流派有很多，专家系统代表了其中的一大类方法。

专家系统指的是这样的计算机程序或应用，其内部含有某个领域专家水平的知识与经验，能够利用人类专家所积累的知识、方法和经验，来解决该领域中的实际问题。专家系统最核心的部分是它所包含的知识，因此，有时也将专家系统称为基于知识的系统。新一代人工智能兴起后，基于大数据和深度学习的方法使从海量无结构多信息源中自动构建知识库并生成推理规则成为可能，这也为专家系统带来了新的发展机遇。

体　验　思　考　构建个性化的推荐系统

人工智能创新园区有大小展馆上百个，选择合适的参观展馆不是一件很容易的事。为方便参观者，园区提供了一个智能小程序，也可视为微型专家系统，它可以根据参观者的情况推荐合适的展馆，协助安排行程。

表 2.1是园区的自动驾驶展馆所收集的 20份用户反馈意见。

表 2.1　自动驾驶展馆用户反馈汇总

样本	首次参观	参观人数	天气	专业人士	参观体验
1	是	多	雨	是	良好
2	是	多	晴	是	良好
3	否	少	雨	否	良好
4	否	多	阴	是	一般
5	是	多	晴	否	一般
6	是	少	阴	否	良好
7	是	多	雨	否	一般
8	否	多	雨	是	一般
9	否	少	晴	是	一般
10	否	多	阴	是	一般
11	否	少	晴	否	良好
12	否	多	雨	是	一般
13	否	少	晴	是	一般

（续　表）

样本	首次参观	参观人数	天气	专业人士	参观体验
14	是	少	阴	否	良好
15	否	多	雨	是	一般
16	是	少	雨	否	良好
17	否	少	晴	否	良好
18	否	少	雨	是	一般
19	否	少	阴	否	良好
20	否	少	晴	是	一般

这次来参观创新园区的都是学生，是非专业人士，其中有些是首次前来，有些曾经跟随父母参观过其他的自动驾驶展馆。当天天气晴朗，参观人数不多。

思考：微型专家系统可否根据以往经验数据和每个学生的特征，预测学生的参观体验，从而给出是否推荐参观自动驾驶展馆的建议呢？

一、专家系统的构成

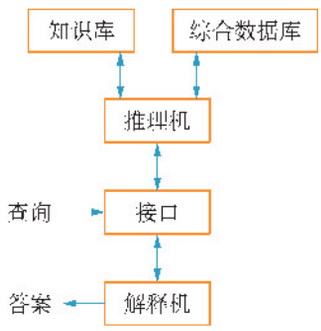


图 2.22　专家系统的构成

与普通信息系统相比，专家系统最明显的特点在于将应用于领域问题求解的知识单独组成一个实体，即知识库，而且对知识库的操作和处理是通过与知识库分离的若干策略进行的，这种机制也被称为推理机或推理引擎。而普通的系统都是把知识隐含在代码中，对知识的处理也由代码直接完成。

如图 2.22 所示，专家系统主要由以下所述的几个部分构成。

1. 知识库

知识库是由领域专家的经验、知识、方法汇集而成的库。每个专家系统通常都按照其知识库中知识的表示方法（即知识表示）来称呼。比如，基于规则的专家系统就是指该专家系统中的知识是以一条条规则的形式来表示的。

2. 综合数据库

综合数据库指专家系统中所使用的除知识库以外的其他类型的

数据，如通用信息、背景信息和一些常识等。

3. 推理机

推理机也称为推理引擎或推理机制，指专家系统处理知识和使用知识获得结论的机制。

4. 接口

和普通的信息系统一样，接口指用户与专家系统进行交互的方式。

5. 解释机

专家系统不但要响应用户的请求，得出结论，还要能对导出结论的依据给出解释和说明，这部分即解释机。

二、知识表示

如同语言是人类思想的载体一样，人工智能中用于表达知识的语言就是承载领域知识的基本方式。人类在用一种语言表达思想时，一般会从表达的形式和表达的内容，即语法和语义两个方面切入。同样地，用知识表示的语言表达人类的知识时也包括这两个方面，即结构（语法方面）与事实（语义方面）。

基于规则的专家系统是最简单、最典型的一种专家系统，这种系统使用一组规则来表示知识。一条规则表示当指定的条件 A 满足时，能推导出某个结论 B，表现为如下形式：

$$\text{if A then B，或 A}\rightarrow\text{B，}$$

其中 A 称为规则的前提，B 称为结论。举例来说，如果 A 是下雨，B 是带伞，那么 A→B 就是一条简单规则：如果下雨，就带伞。

规则被用来从已知事实（即综合数据库中的数据）推导出新的事实，这个过程称为（基于规则的）知识推理。于是，基于规则的知识库也就是由一些形如“if…then…”这样的语句所构成的规则库。

以上述展馆推荐系统为例，假如我们希望该系统的基本功能为：能根据以往的经验和用户特征来预测参观者参观某个展馆的体验，并

基于这种预测为新的参观者推荐合适的展馆。

表 2.1 的内容实际上构成了我们推荐系统的综合数据库——表中的每一行都是若干条“事实”。以第一行为例，它描述的事实为：参观者 1 是专业人士，是首次参观且体验良好，同时参观那天下雨且参观人数很多。

我们可以进一步将表 2.1 第一行所描述的事实分解为更简单的语句的复合形式。

令：

A1：参观者 1 是专业人士

A2：参观者 1 是首次参观

A3：参观者 1 参观体验良好

B1：参观者 1 参观的那天下雨

B2：参观者 1 参观的那天该展馆参观人数很多

则第一行所描述的事实可表示为

A1∧A2∧A3∧B1∧B2

其中“∧”表示逻辑连接词“且(and)”。

在以往的案例(数据)中，此类参观者的参观体验都是良好。因此，我们可以考虑为知识库添加一条简单的规则：

如果(专业人士＝是)并且(首次参观＝是)那么(参观体验＝良好)，

或者写成

if[(专业人士＝是)and(首次参观＝是)]then(参观体验＝良好)。

项 目 实 践　通过互联网，了解身边的专家系统

通过互联网查找自己身边被普遍使用或者公众普遍关注的专家系统，根据它们的特点，讨论专家系统与普通信息系统的关系，了解专家系统在现实生活中的作用。

从以上两个规则的例子中，我们可以看到，一个参观者的参观体验不仅与其是否为专业人士相关，还与天气和参观人数相关，而且表中的“事实”数据可能会产生不同的规则。当不同的事实中的不同因素导致不同的结论时，如何来决定哪些因素更重要，哪些因素并非那么重要，甚至是不相关的呢？

要解决这个问题，需要某种算法，这种算法能从“事实”数据库中计算出哪些因素对于决策而言最重要，同时总结归纳出相应的决策规则。后面将要介绍的决策树算法就是这样一种算法的典型例子。

知 识 延 伸　　知识表示方法

人工智能中用于表示知识的方法其实还有很多，比如我们都很熟悉的思维导图（mind map）或概念图（concept map）。相比之下，规则只是一种最简单、最基本的知识表示方法。

以下我们简要介绍另外几种常用的知识表示方法。

1 语义网

语义网（semantic network）是20世纪60年代提出的一种知识表示方法，在早期的专家系统构建中曾经起到很重要的作用。同时，语义网的基本思想也显著地影响到了更近的知识表示方法。

语义网使用带标记的有向图来表示各种知识，其中图的节点表示某种概念或概念的实例，带方向的边表示概念（实际例子）之间的关系，而边上的标记用来精确地描述这种关系。

语义网中的关系可以表示子类（超类）——边的方向从子类指向超类；也可以表示实例——边的方向从实例指向它所属的类；或者表示一个对象的某种属性（颜色、大小等）。此外，有向边还可以定义一个对象的所有权（例如拥有另一个对象）。

例如，图2.23描述了一个与动物相关的微型知识库。在这个例子中，从“哺乳动物”节点指向“动物”节点的有向边表示的是子类（超类）关系——“哺乳动物”是“动物”；从“猫”节点指向“哺乳动物”节点的有向边表示的是实例关系——“猫”是一种“哺乳动物”；从“加菲”节点指向“公猫”节点的有向边表示的是实例的一种属性——“加菲”是一只“公猫”。

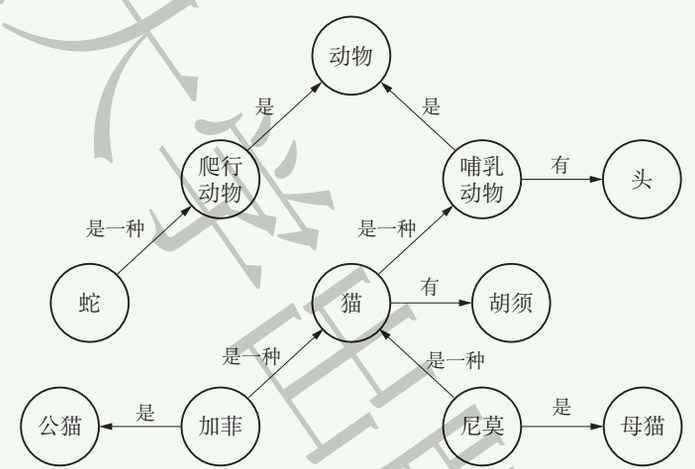


图 2.23　一个微型语义网

注意：不要将语义网与后文的语义万维网（semantic web）相混淆，尽管两者无论从名称上还是从内涵上都有类似的地方。

2 框架方法

框架方法与语义网方法有非常紧密的关联：框架就是将语义网中具有“相同层次”的那些知识进行归类表示，每个这样的类就构成一个框架。图2.24所示就是用框架的表示方式来实现图2.23中语义网所表示的部分知识。

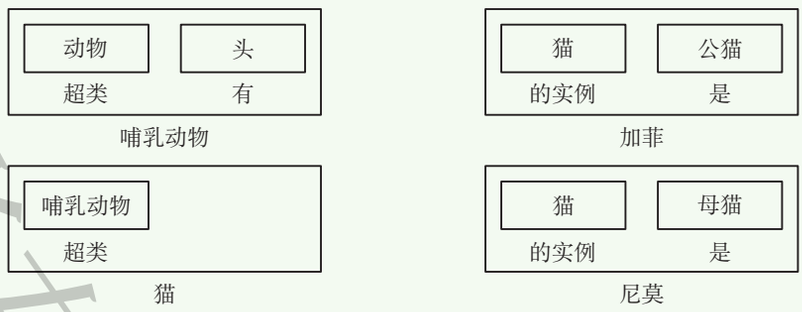


图 2.24 知识的框架表示

3 语义万维网

语义万维网是由万维网联盟(W3C)所发起的一个项目,其主要倡导者为万维网的发明者蒂姆·伯纳斯·李(Tim Berners-Lee)。语义万维网的目标是发展一种"机器可理解"的表示方式来描述万维网上的内容,以便于对万维网信息进行自动化的语义处理。

语义万维网的一大特色是使用了"本体"(ontology)作为形式概念来描述一个领域的知识结构。

简单地讲,一个本体就是某个特定领域中的概念及概念之间的各种关系,如层级关系、从属关系等。

本体为来自不同领域的知识提供了一种语义上的"共识",从而在一定程度上解决了对跨领域知识的表示及应用中存在的一词多义和一义多词等问题,提高了万维网上知识的可互操作性。

在 XML(可扩展标记语言)的基础上,W3C 为语义万维网制订了一系列的标准描述语言,包括 RDF、RDF Schema、OWL/OWL2。

前面提到的语义网和框架虽然也能描述概念和类之间的层次及从属关系,但对类中的具体实例(在 OWL 中称为"个体")却很难做进一步的精确描述。例如,在上面猫的例子中,如果我们想知道场景中有几只猫,又有几只公猫(母猫),猫与猫之间有何关系等等。

使用 OWL 语言就可以很容易地将这些更细致的信息和关联描述清楚,从而为基于类和类中个体之间依赖关系的知识推理打下基础。

下面是用 OWL 语言写的几个小片段:

```
class:Cat                                    (一个猫的类 Cat)

_:x rdf:type owl:Class;
owl:unionOf(:MaleCat:FemaleCat). (类 Cat 是类 MaleCat 和类 FemaleCat 的并)

:Cat owl:equivalentClass _:x.
_:x rdf:type owl:Class;
owl:unionOf(:MaleCat:FemaleCat). (类 Cat 与类 MaleCat 和类 FemaleCat 的并两者等价)

:Garfield rdf:type:Cat.                  (Garfield 是类 Cat 中的一个个体)
:Nimo rdf:type:Cat.                      (Nimo 是类 Cat 中的一个个体)
```

4 知识图谱

知识图谱可以视为知识本体表示的一种可视化形式，图 2.25显示了深度学习的知识图谱。

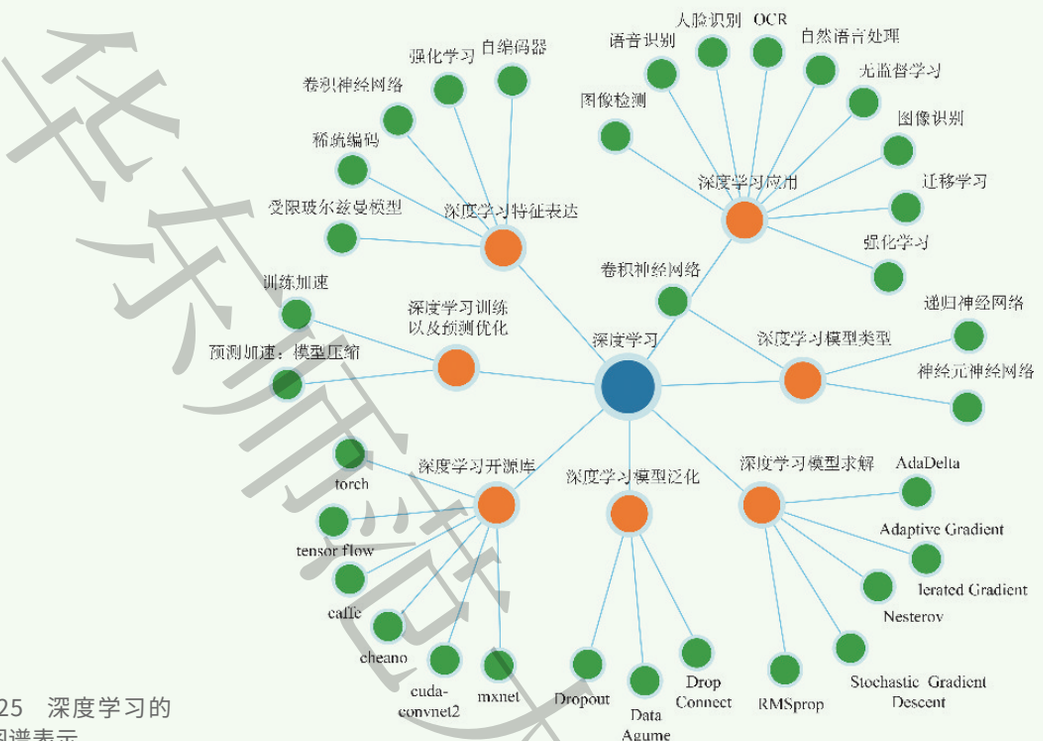


图 2.25 深度学习的知识图谱表示

三、决策树算法

基于表 2.1，我们希望将是否去参观的决策建立在参观体验的基础上，即使用“参观体验”作为我们的决策标签，而是否首次参观、参观人数、天气情况、是否专业人士则是特征，每个样本在这些特征上的具体取值称为特征值。

在之前的学习中我们已经看到，如果用手工的方法，即使对这样一个“小数据”样本，从数据中抽取知识（规则）也绝非易事。以下我们讨论如何使用决策树算法自动从数据中总结出规律和知识，从而让专家系统能自行生成或更新知识库。

1. 信息熵

信息熵（information entropy，也简称为熵）是一个用来描述数据

不确定性的量：数据的不确定性越高，信息熵的值就越大；反之，数据的确定性越高，信息熵的值就越小。例如，上海夏天高温的可能性是非常高的，所以不确定性就低，上海夏天高温的熵就小。同样道理，上海夏天下雪的可能性是非常低的，所以不确定性也低，熵就小。但是上海秋天多雨的不确定性就高，上海的秋天可能多雨，也可能少雨，所以上海秋天多雨的熵就远远大于上海夏天高温的熵。就离散的样本数据而言，其不确定性就是数据的分布情况。例如，在表 2.1 的 20 个样本中，有 9 个样本体验良好，11 个样本体验一般，参观体验的信息熵接近于 1。

知 识 延 伸　　熵的概念

熵（entropy）的概念最初是由德国物理学家鲁道夫·克劳修斯（Rudolf Clausius）在 1854年提出的，后来奥地利物理学家路德维希·爱德华·玻尔兹曼（Ludwig Edward Boltzmann）将其用于统计热力学的研究中，用来度量热力学中体系的“混乱程度”或“无序程度”。热力学第二定律指出：在孤立系统中，体系总是自发地向混乱度增大的方向变化，即使整个系统的熵值增大。 1948年，信息论奠基人克劳德·香农（Claude Shannon）借用热力学中熵的概念来命名信息熵，从而解决了信息的度量问题。信息中的不确定性可以看作是一种信息的“混乱程度”或“无序程度”，而信息熵指的是为了消除信息中的不确定性所需要的信息量。因此，一个事件的不确定性越高，信息熵就越大。

比如说，五月的上海，第二天下雪的概率非常小（接近确定了），所以这个事件的信息熵就很小。但是第二天会不会下雨呢？上海五月份下雨的概率要远大于下雪的概率（是否下雨的不确定性增加了），所以上海五月份下雨的信息熵就比下雪的信息熵要大很多。

如果我们有当天的气象信息，例如湿度很大、周边地区都在下雨等，那么第二天下雨的概率也就非常高。有趣的是，在这种情况下信息熵反而会减小，因为是否下雨的不确定性减小了。如图 2.26所示，我们用横轴表示下雨的概率，0就是不下雨，1就是一定下雨；用纵轴表示下雨这个事件的信息熵。无论下雨的概率很大还是很小，信息熵都小。但是当下雨的概率是 50%时，此时是否下雨的不确定性最大，信息熵也就最大。于是，表示信息熵的函数的图像大致应该如图 2.26所示。

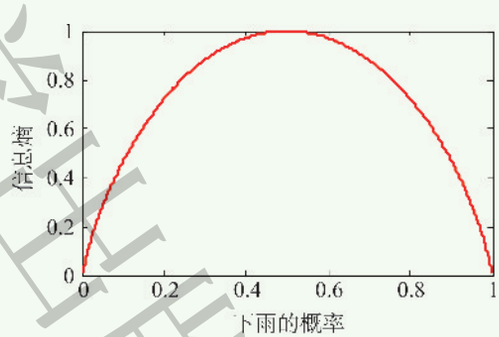


图 2.26　信息熵示意图

在数学中，定义在实数轴上具有上述形状的最简单函数是 $y=-x\log_2 x$，因此这个函数被用来作为定义熵的基础。

若设一个场景可能有 x_i 个事件，用 $p(x_i)$ 表示事件 x_i 发生的概率，则该场景的信息熵定义为：

$$E(p)=-\sum_i p(x_i)\log_2 p(x_i)$$

当 x_i 仅取有限个离散值时，概率 $p(x_i)$ 经常使用频率来代替。

2. 信息增益

除了可以通过计算信息熵来描述参观体验的不确定性外，也可以在某些特定的特征值下计算信息熵。在表 2.1 中，有 11 个专业人士，其中 2 个体验良好，9 个体验一般，所以专业人士参观体验的信息熵就比较低，约为 0.68。也就是说，如果知道了参观者是专业人士，就可以比较容易地估计参观体验。

那么，可否据此得出结论，参观者是否为专业人士是决定参观体验的重要特征呢？答案是否定的。因为表 2.1 中还有 9 个非专业人士，还需要计算非专业人士参观体验的信息熵。

综合专业人士这一特征在不同取值下的信息熵，计算信息增益(information gain)的表达式如下：

信息增益(专业人士)
=信息熵(参观体验)
$$-\sum_{\text{特征值}\in\text{特征取值范围}}$$
特征值比例×信息熵(该特征值下的参观体验)。

信息增益可以衡量某个特征在预测参观体验中的重要性。例如，信息增益(专业人士)的计算结果为 0.27，就是说通过专业人士这一特征，可以使参观体验的信息熵下降 0.27。同理，可以计算出信息增益(首次参观)为 0.11，信息增益(参观人数)为 0.13，信息增益(天气)为 0.02。

比较首次参观、参观人数、天气和专业人士这 4 个特征，可以发现专业人士这个特征的信息增益最大。这表明专业人士这个特征在决定参观体验上是最有贡献的。

知 识 延 伸　　基尼不纯度

在计算信息增益时，信息熵是作为数据的不确定性或不纯度的度量而出现的。在产生决策树的算法中，用于度量数据不纯度的方法不仅有信息熵，还有一个常用的度量叫做基尼(Gini)不纯度。

关于特征 F 的基尼不纯度可以通过以下公式来计算：

$$G(F)=\sum_{i=1}^{N} p_i(1-p_i)=1-\sum_{i=1}^{N} p_i^2,$$

其中 p_i 表示标签为 i 的数据项在样本中所占的比例。请同学们思考一下，为什么这样定义的值可以衡量数据的不确定性或不纯度？

用基尼不纯度代替信息熵，我们可以得到类似的信息增益计算公式。

3. 决策树的建立

如图 2.27 所示，通过专业人士这一特征，数据被分为是和否两个分支，但无论是在哪一支，都同时存在着体验良好和体验一般两种可能性。这说明仅凭是否为专业人士这一特征不足以预测参观体验，还需要其他特征来共同决定。

对于 11 个专业人士，计算参观体验的信息熵，结果约为 0.68。在此基础上，分别计算首次参观、参观人数和天气在这 11 个数据上的信息增益。经比较，参观人数的信息增益最高，约为 0.13。以参观人数为第二层特征，把数据分成参观人数多的一组和参观人数少的一组。参观人数少的一组都体验良好，参观人数多的一组都体验一般。

对于 9 个非专业人士，分别计算首次参观、参观人数和天气在这 9 个数据上的信息增益。经比较，首次参观的信息增益最高，约为 0.76。以首次参观为第二层特征，把数据分成首次参观的一组和非首次参观的一组。首次参观的一组都体验良好，而非首次参观的一组都体验一般。

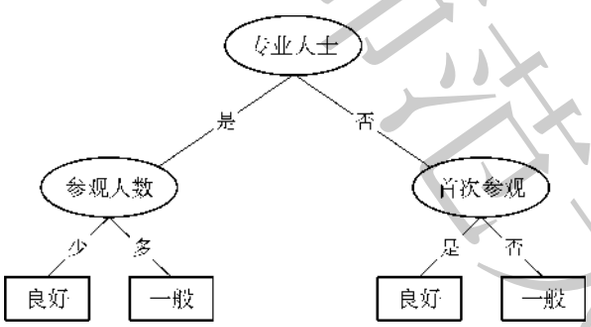


图 2.27　基于表 2.1中的数据计算推荐决策树

汇总以上的计算，我们可以建立一棵如图 2.27 所示的决策树，其中每一个非叶节点（图中的椭圆形节点）都代表一个特征，通过该特征的取值来判断样本属于正样本还是负样本。决策树的每个叶节点（图中的矩形节点）数据都是一种标签。

决策树算法通过对数据信息增益的计算，将经验提升到规则，再把规则显性地表现出来。

探　究　活　动　　**决策树中的特征**

在图 2.27中，“天气”这个特征并没有出现在决策树上，那么是否可以得出结论：天气状况与是否推荐参观没有关系？请分组讨论通过现有的数据是否足以找到通用的规则，以及怎样判别由数据所造成的规则偏颇。

四、推理机

拥有知识库之后，专家系统仍然不知道如何运用知识来解决实际问题，推理机就是如何运用知识库中的知识进行推断的机制和规则。

我们将通过一个实例来讨论专家系统中两种最基本、最常用的推理方法：正向推理和反向推理。

我们以一个用逻辑规则表示的微型知识库为例，来说明正向推理和反向推理的最基本应用方式。此知识库里有 11 个规则（也称为语句），前面 3 个是基本事实（推理的起点），后面 8 个则是非常简单的逻辑关系：

（1）A
（2）B
（3）C
（4）A∧B⇒D
（5）B∧C⇒E
（6）A∧C⇒F
（7）A∧F⇒G
（8）D∧F⇒K
（9）G∧K⇒Q1
（10）E⇒H
（11）H∧C⇒Q2

我们用一种网状的示意图来表示知识库中各个语句之间的关系，如图 2.28 所示。在图 2.28 中，白色的大圆点表示基本事实或者由基本事实通过规则所推导出来的衍生事实；灰色的小圆点表示应用一次知识库中的规则。

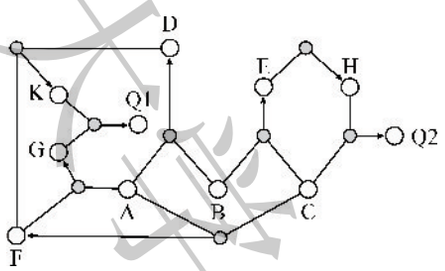


图 2.28　微型知识库示意图

假设用户询问由知识库能否推出 Q2，我们看看正向推理和反向推理是如何解决这个问题的。

正向推理就是从事实出发，逐步运用规则推导出结论的过程。建议同学们拿一支彩色笔，将推导的过程标记出来。首先，根据基本事实 A、B、C（图 2.29 甲中标记为红色），我们能推导出 D、E、F（图 2.29 乙中标记为红色）；再进一步能推导出 K、G、H（图 2.29 丙中标记为红色）；最后一步推导出 Q2 和 Q1（图 2.29 丁中标记为红色）。

我们不但推导出了 Q2，还顺带推导出了 Q1，这似乎是一件好事。但是，实际使用的专家系统的知识库可能有数万条规则，可能的推理路径会达到数十亿条，这意味着正向推理的效率会很低，甚至不可能进行正常推理。

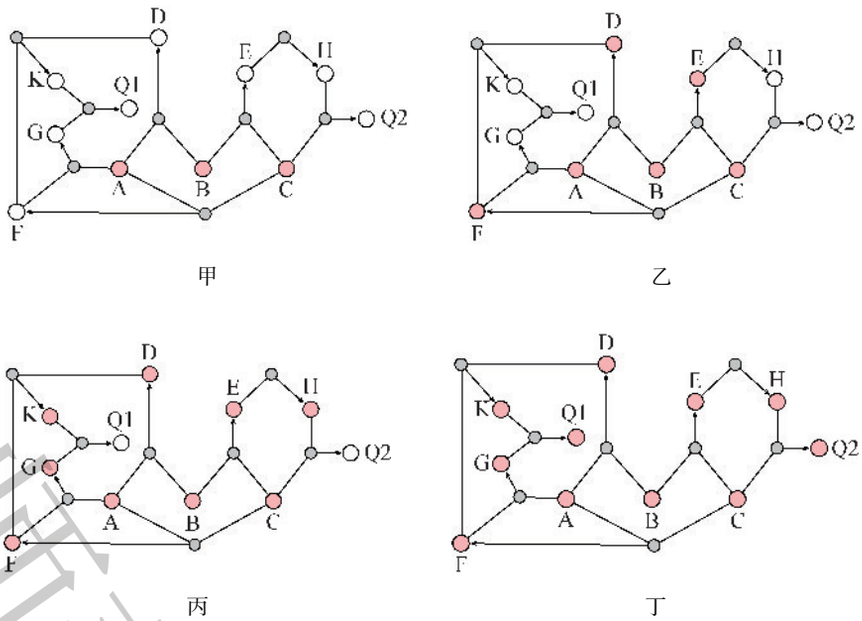


图 2.29　正向推理

反向推理采取从结论逐步回溯到事实的方式：先自结论开始，标示出能够导出结论的相关知识，然后再由标示的部分根据规则推导出结论，从而完成验证。

同样是问是否能推导出 Q2 这个问题，反向推理先查找哪些知识与 Q2 相关，发现是 C 和 H（如图 2.30 乙所示）；C 是基本事实，不需再继续回溯了；要导出 H 需要 E，而 E 与 B 和 C 相关（如图 2.30 丙所示），现在 B 和 C 都是基本事实（如图 2.30 丁所示）。

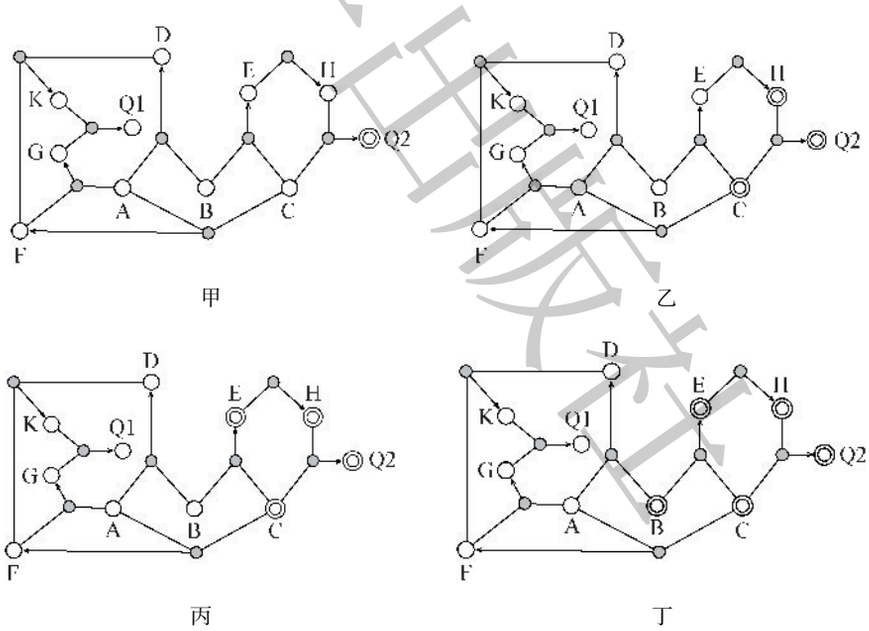


图 2.30　反向推理：标示范围

至此，我们就标示出了与 Q2 相关的知识范围。然后，通过事实和使用规则共同推导出 Q2，如图 2.31 所示。

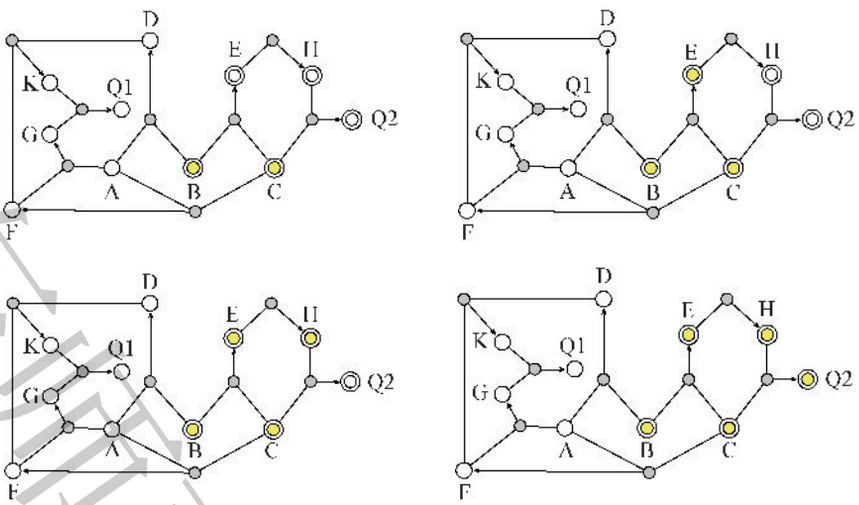


图 2.31　反向推理：验证阶段

探　究　活　动　由决策树产生推理规则

就图 2.27所示的决策树而言，正向推理基于用户的专业程度、是否首次参观、参观人数来决定推荐意见，而反向推理则由推荐意见开始，先确定知识库中哪些信息与推荐意见相关，再根据相关信息得出推荐意见。

1　将图 2.27中的决策树表示为规则，并比较正向推理和反向推理的结果。

2　尝试通过改变图 2.27中的决策树结构，从而产生不同的正向推理和反向推理结果。

3　在专家系统中，什么样的情况适合正向推理，什么样的情况适合反向推理？

4　除正向推理和反向推理外，还可能有其他形式的推理机制吗？

作业练习

构建专家系统和知识库

根据参观者的情况来推荐合适的展馆，可以抽象为一个微型专家系统，其中用户的反馈意见以及相关信息可以看作是综合数据库中的原始数据。请定义个性化展馆推荐系统的知识库、综合数据库、推理机、接口、解释机。

第三节　机器学习

学习是一种比较复杂的智能行为，迄今为止，人类对于自身的学习机理还没有完全理解。但同时，学习是人类获取和使用知识解决问题的重要手段，是从数据通向智能的必由之路。机器学习是人工智能中最关注如何从数据中进行学习的部分，是新一代人工智能的核心。在本书中，我们将机器学习理解为从数据和经验中进行学习，以获得、改善或逼近问题求解模型的一个领域。

机器学习的最大特点在于系统并非由程序硬性规定，而是可以通过对实际案例的学习逐步训练出来。正因为机器学习是从数据中学习，所以其学习的方式强烈依赖于可用数据集的特性。例如在第二节的专家系统中，通过对参观者的数据调查，人工智能系统可以自动学习到：就某个展馆而言，参观者是否是该领域的专家，对参观体验的影响很大，而参观当天的天气状况不太能影响到参观者的体验。因为这些规律都是计算机系统自动学到的，所以我们把这类模型归类为机器学习模型。如果在建立专家系统的时候，我们直接提供规则，比如“因为科学园都是室内场馆，所以参观当天的天气状况不会影响参观体验”，当有用户来参观的时候，专家系统根据事先给定的规则来预测用户的参观体验，这就不是机器学习模型。但是这种模型是可以做出自动预测的，所以还是属于人工智能技术的。

在机器学习中，还可以根据模型在学习时是否获得了标准答案，把机器学习技术细分为监督学习和非监督学习。例如我们收集了一些学生的信息，发现：小张和小王喜欢闲暇时打电子游戏，喜欢阅读与兵器知识有关的杂志；小李和小赵喜欢参观古迹，喜欢阅读散文和诗歌。物以类聚，人以群分，我们会有意识甚至无意识地把小张和小王分成一群，把小李和小赵分成一群，这就是非监督学习中的经典问题——聚类，而小张和小王的群组就被定义为聚类问题中的一个簇，小李和小赵的群组被定义为另一个簇。每个簇的成员都比较相似，而两个不同簇的成员的差别就相对比较大了。那什么是监督学习呢？沿用同一个例子，班上新转来一位同学小陈，他是从另一个城市搬来的，第一次去科学园，想知道无人机展馆是否好玩，本节中我们将介绍一种新的分类方法，即 k-近邻分类算法来帮他做决定。比如说，小陈和小张、小王一样，都喜欢玩电子游戏和看兵器杂志，但是对于历史古迹和诗歌散文没有兴趣。小张和小王强烈推荐无人机展馆，如果我们据此推断小陈也会喜欢无人机展馆，那就是根据与他相似的学生的偏

好来进行预测，这就是 k-近邻分类算法的基本理念。由于小张和小王对无人机展馆的喜欢与否我们已经知道，所以这一类机器学习算法被归类为监督学习，也就是说，我们在预测小陈对于无人机展馆的态度时，已有了小张和小王的正确答案作为监督的样本。

一、线性回归

线性回归是机器学习中的一个基础模型，其基本思想常常被用在各种不同的监督学习和非监督学习技术中。我们将以自动驾驶电动汽车的耗电量为例，讨论如何用最小二乘法求解线性回归方程。

体 验 思 考　　预测自动驾驶电动汽车的耗电量

科学园的自动驾驶汽车展馆提供了一组电动汽车耗电量和行驶距离之间的数据，如图 2.32所示。同学们可以讨论一下：人工智能系统可否根据这些数据预测同一类型的自动驾驶电动汽车从一个地点行驶到另一个地点的耗电量？

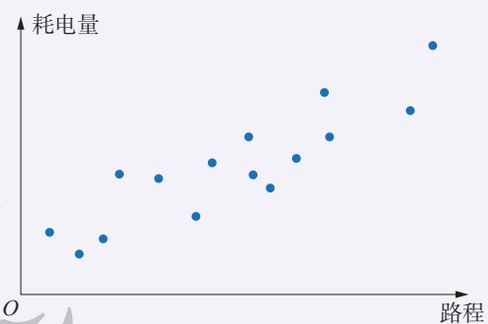


图 2.32　自动驾驶电动汽车的路程和耗电量

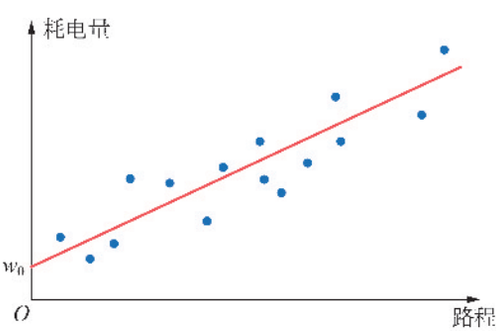


图 2.33　简单线性回归模型对于耗电量的求解示意图

在预测自动驾驶电动汽车耗电量的场景中，我们观察到自动驾驶电动汽车的耗电量和其行驶路程之间存在着一定的关联。这种关联总的趋势是路程越长，耗电量越高，但是对于某些数据点而言，也可能出现一个数据点的路程比另一个数据点的路程略长，但是其耗电量却比较低。如果引入一个假设——路程和耗电量之间的关系可以用一个线性函数表示（例如图 2.33 中的红色直线），那么对于每一段路程，人工智能系统就可以大致估算出相应的耗电量。这种分析一个或者多个自变量和一个因变量之间的线性关系的方法被定义为线性回归。

在自动驾驶电动汽车的例子中，自变量只有一个，就是路程，我们把它定义为 x，因变量是耗电量，我们把它定义为 y。这种单变量的模

型 $y=f(x)$ 属于简单线性回归。我们的目标是找到一条最优直线，可以更好地表达图 2.32 中的蓝色点在二维空间里的线性关系，即确定下式中 w_1 和 w_0 的值。

$$f(x)=w_1x+w_0。$$

在上式中，w_1 是直线的斜率，表达的是耗电量随路程增加的幅度；w_0 是方程的偏置，例如在汽车刚刚发动的时候，虽然路程为零，但是启动汽车也是需要消耗电量的。

如图 2.33 所示，在路程－耗电量的二维空间中，存在着无数条直线，但似乎没有任何一条直线可以覆盖所有蓝色点。每一条直线都可能造成某些蓝色点的耗电量不能被准确地预测。那么如何定义最优直线呢？在线性回归问题中，误差平方和是一个常用的目标函数，即：

$$\sum_j(y_j-(w_1x_j+w_0))^2。$$

上式中，j 指的是所有已知数据里的第 j 个数据。我们的目标是最小化所有已知数据的误差平方和。对于一个二次函数，可以通过求导的方式找到最小值所对应的 w_1 和 w_0。具体的实现方法如下：

首先，把目标函数对于 w_1 和 w_0 求取偏导，使得偏导值等于 0，即：

$$\partial\sum_j(y_j-(w_1x_j+w_0))^2/\partial w_1=0,$$
$$\partial\sum_j(y_j-(w_1x_j+w_0))^2/\partial w_0=0。$$

当上面的两个方程的系数向量线性无关时，可以通过求解联列方程得到唯一的解析解，即：

$$w_1=(N\sum x_jy_j-\sum x_j\sum y_j)\Big/(N\sum x_j^2-(\sum x_j)^2),$$
$$w_0=(\sum y_j-w_1\sum x_j)\Big/N。$$

也就是说，如果把最小化已知数据的误差平方和定义为最优直线的标准，那么就能够直接计算出最优的 w_1 和 w_0。

二、k-均值聚类算法

聚类学习是无监督学习的经典问题，主要研究的是数据在特征空间内的相互关系。聚类学习的技术被广泛应用于数据挖掘、自然语言处理和计算机视觉。

体 验 思 考

集合地点的选择

结束智能创新园区的参观任务后，同学们需要在园区的不同地点集合，由大巴车一起送离园区。如果我们通过定位系统得到了各位同学在园区的位置，思考如何通过聚类方法找到合适的集合地点。

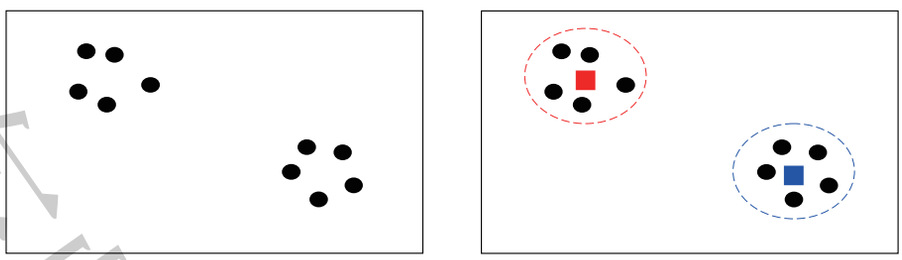


图 2.34　聚类问题示意图

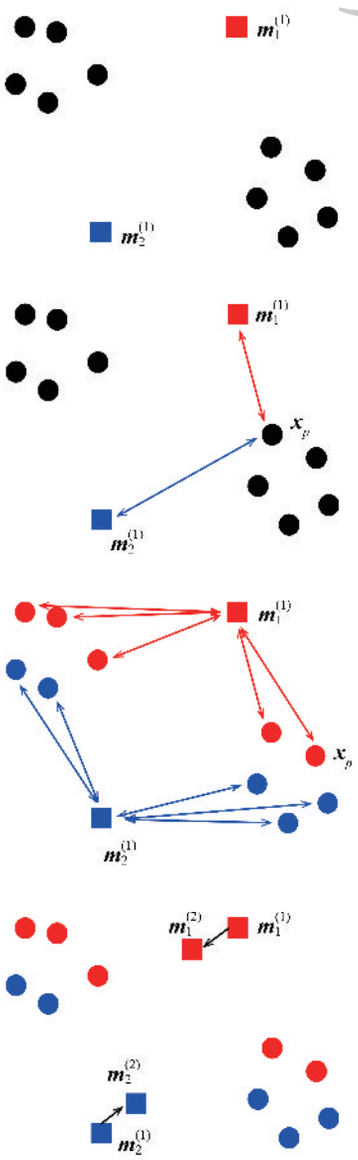


图 2.35　第一次聚类结果

如图 2.34 左图所示，10 个数据分布在一个二维的空间，一些数据相距得比较近，一些数据相距得比较远。如果把相距比较近的点分成同一个组，把相距比较远的点分成不同组，那么这类问题就可以被定义为聚类问题，而这些组则被定义为簇。如图 2.34 右图所示，10 个数据被分成红色和蓝色两簇。直观地看，圈内的点互相之间的距离比较近，不同圈之间的点距离比较远。

在聚类算法设计中，我们首先要考虑两个问题：第一，数据点应该被分成几个簇；第二，数据点之间的距离如何度量。在图 2.34 所示的例子中，我们假设图中的点可以被分成两个簇，并且数据点之间的距离由欧氏距离来定义。如果用“暴力”搜索的方式，那么找到最佳答案的计算量会非常大。是否可以用更智能的方式来聚类呢？k-均值聚类算法巧妙地应用了人工智能中“贪婪”和“随机”两个最基本的设计理念，可以快速而准确地把数据聚类。

k-均值聚类算法首先假设我们已经找到了簇的中心。如图 2.35 所示，$\boldsymbol{m}_1$ 是红色簇的中心，$\boldsymbol{m}_2$ 是蓝色簇的中心。$\boldsymbol{m}_1$ 和 $\boldsymbol{m}_2$ 在初始状态其实是随机选取的。计算图中每一个点与中心 $\boldsymbol{m}_1$ 和中心 $\boldsymbol{m}_2$ 的距离。如果数据点 $\boldsymbol{x}_p$ 与 $\boldsymbol{m}_1$ 的距离较近，我们就把数据点 $\boldsymbol{x}_p$ 归在红色簇内，反之，则归在蓝色簇内。对所有数据点做相同的操作，就得到图 2.35 所示的第一次迭代的聚类结果。

根据这个结果，我们用所有簇内点坐标的均值作为新的簇中心。有了新的簇中心，我们可以重复第一次迭代的做法，生成图 2.36 上排两图所示的第二次聚类结果，并且产生第三次迭代的簇中心。

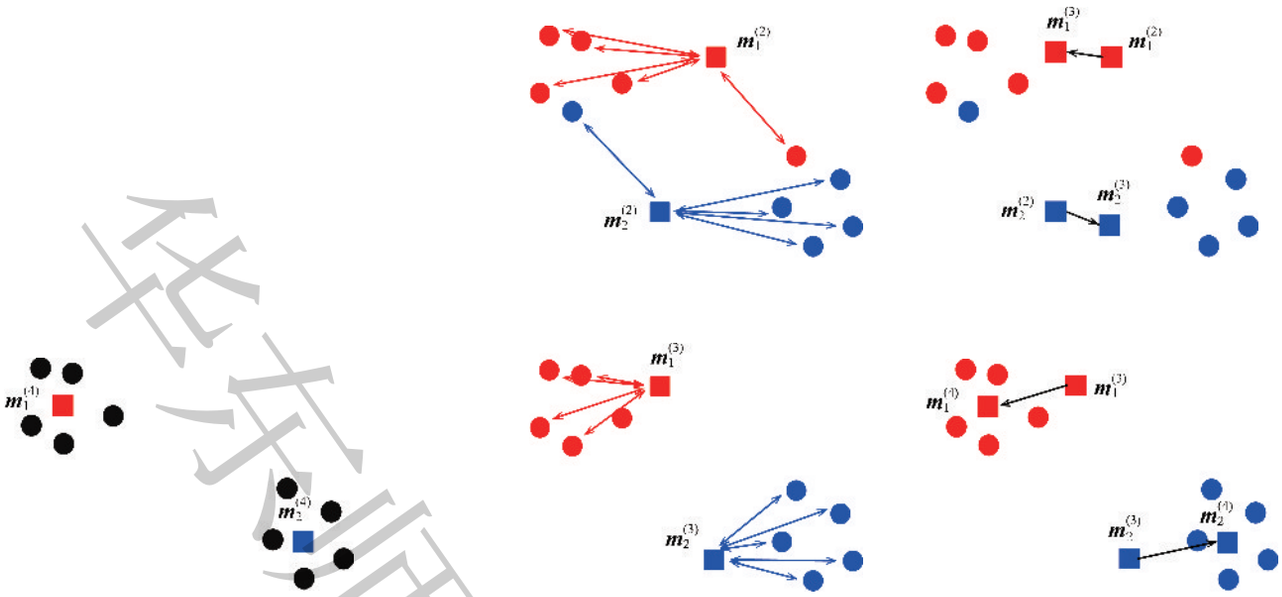


图 2.36　第二次、第三次聚类结果

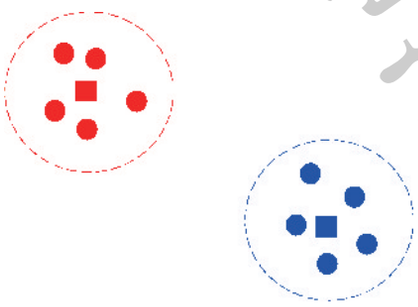


图 2.37　第四次聚类结果

根据第三次迭代的簇中心，我们生成了如图 2.36 下排两图所示的第三次聚类结果，并且产生第四次迭代的簇中心。

根据第四次迭代的簇中心，我们生成了第四次聚类结果，而这次的聚类结果和第三次相同，所以簇中心也没有产生变化，此时，我们称聚类结果“收敛”了，如图 2.37 所示。我们用 k-均值聚类算法得到了符合预期的聚类结果。

项　目　实　践　使用 Python语言实现典型的无监督学习算法

1　运用 k-均值聚类算法实现如图 2.38中左图所示问题的聚类。

2　思考当数据分布如图 2.38中右图所示时，如何用 k-均值聚类算法实现聚类的结果。

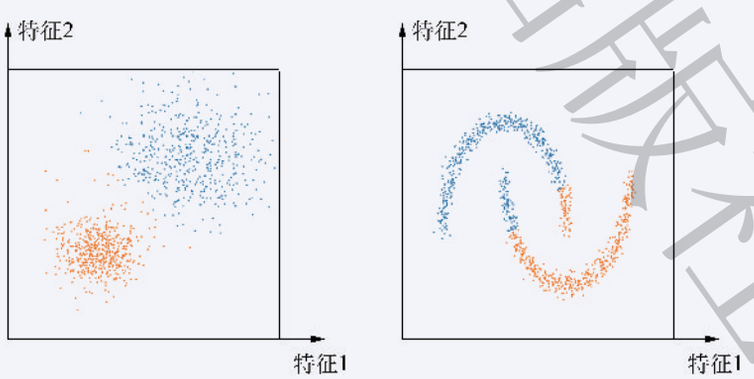


图 2.38　两组不同数据的聚类结果

三、k-近邻分类算法

k-近邻分类算法是监督学习的经典技术，其设计理念源于一个众所周知的简单道理，即“物以类聚，人以群分”。

探 究 活 动　参观者行程安排

在本章第二节中我们以自动驾驶展馆为例，介绍了如何根据参观者的情况推荐合适的展馆。请同学们讨论：如果我们把情况推广到 10个展馆，如何用分类算法协助参观者安排行程。

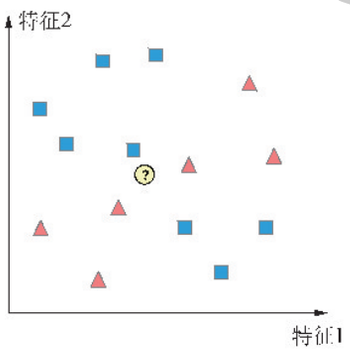


图 2.39　二类分类问题

如图 2.39 所示，数据分布在一个二维的特征空间中。红色的三角形代表一类数据，蓝色的正方形代表着另一类数据。现在出现了一个新的数据（用黄色的圆圈表示），这个数据可能属于红色三角形的类型，也可能属于蓝色正方形的类型。我们可否根据已有的标签信息，确定这个未知数据的标签呢？在这个问题中，数据的值可以用一个二维的向量来表示，一个维度表示特征 1 的值，而另一个维度表示特征 2 的值。红色的三角形和蓝色的正方形是这些数据的标签。已经有了标签的数据被定义为训练数据，可以用来训练分类模型。黄色圆圈数据属于我们要预测标签的数据，被定义为测试数据。

k-近邻分类算法的基本思路非常简单，即以每一个测试数据为中心，看看哪些训练数据离测试数据比较近，根据在特征空间中与测试数据相邻的训练数据的标签来确定测试数据的标签。具体到预测图 2.39 中黄色圆圈数据的标签，我们要考虑以下几个问题：

（1）怎样计算那些和测试数据相邻的训练数据与测试数据之间的距离？k-近邻分类算法中有不同的距离计算方法。在图 2.39 中，我们采取的是训练数据与测试数据在二维特征空间中的欧氏距离。

（2）如何判定哪些训练数据与测试数据相邻？k-近邻分类算法里的参数 k 就是用来描述相邻关系的信息。比如，我们定义 k 等于 3，就意味着我们认为与测试数据最相邻的三个点与测试数据有着相邻关系；如果定义 k 等于 5，就意味着我们认为与测试数据最相邻的五个点与测试数据有着相邻关系。确定 k 有多种方法，最简单的方法是根据以往经验事先定义一个 k 值，最常见的 k 值是 3 和 5。当然，对于不同的应用，k 值的差异是非常大的。

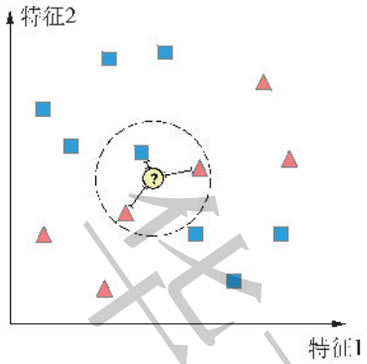


图 2.40　k-近邻分类算法

（3）如图 2.40 所示，当 k 等于 3 时，我们根据二维特征空间的欧氏距离确定了离测试数据最近的三个点。那么接下来的问题是，如何根据相邻训练数据确定测试数据的标签。一个直观的方法是直接投票。还是以图 2.40 为例，在三个与测试数据相邻的点中，一个点是蓝色正方形标签，两个点是红色三角形标签，那么采取简单多数原则，测试数据的标签被预测为红色三角形。

项　目　实　践　使用 Python实现 k-近邻分类算法

1 运用 k-近邻分类算法实现对于二维空间数据的两类分类，使用欧氏距离计算测试数据与训练数据之间的相邻关系。

2 运用 k-近邻分类算法实现对于二维空间数据的五类分类，使用欧氏距离计算测试数据与训练数据之间的相邻关系。

知　识　延　伸　集成算法

与 k-近邻分类算法类似，集成算法的基本思想也来源于生活实践。在现实生活中，我们常常采取博采众长的方法，来解决困难的问题。集成算法就是这种集体智慧在机器学习技术上的具体体现。

绝大多数集成算法都是在监督学习的框架下实现的。以分类问题为例，集成算法在解决一个分类问题的时候，会训练一系列分类器，最后的分类结果由众多分类器共同决定。这里，我们基于前面学习的决策树，介绍一个集成学习的经典算法——随机森林。

假设训练数据有 N 个，用 M 个特征来表示。建立一个决策树，决策树的每一个训练数据都是从 N 个数据中随机取出的，对于决策树的每一个节点，从 M 个特征中随机取出 m 个特征，按照信息增益最大的原则，从 m 个特征中选择合适的特征构建决策树。重复这个过程，构建 k个决策树，组成随机森林。

在计算测试数据分类结果的时候，首先用 k个已有的决策树对测试数据独立分类，得出 k个分类结果。再通过投票的方式，确定测试数据的最终分类结果。

与经典的决策树算法相比，随机森林在精度、效率和鲁棒性 (robustness)上都有着明显的优势。第一，由于随机性的引入，随机森林算法不容易陷入过拟合，所以有着很多的泛化性。第二，算法实现简单，学习速度快，还可以支持并行计算，尤其适合于处理高维数据。第三，随机森林具有很好的抗噪声能力，对于不平衡的分类数据集，也有良好的稳定性。

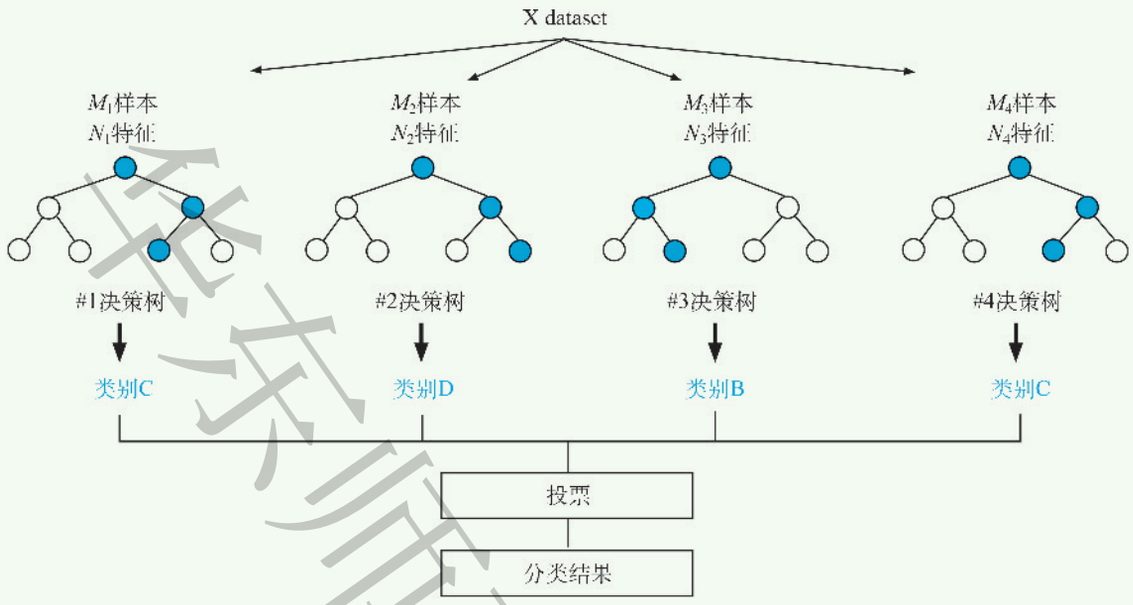


图 2.41　随机森林算法示例

第三章

漫游深度学习的世界

本章学习目标

- 了解人工神经元的工作机制，掌握单层感知机的基本算法。
- 掌握人工神经网络的基本学习算法，并用编程语言实现一个简单的人工神经网络。
- 了解深度学习模型的概念、特点，以及其与大数据之间的关系。学习卷积神经网络的基本架构，分步骤实现其中的核心功能。
- 学习开源人工智能应用框架，搭建简单的深度学习应用框架，熟悉根据不同环境和任务配置参数。

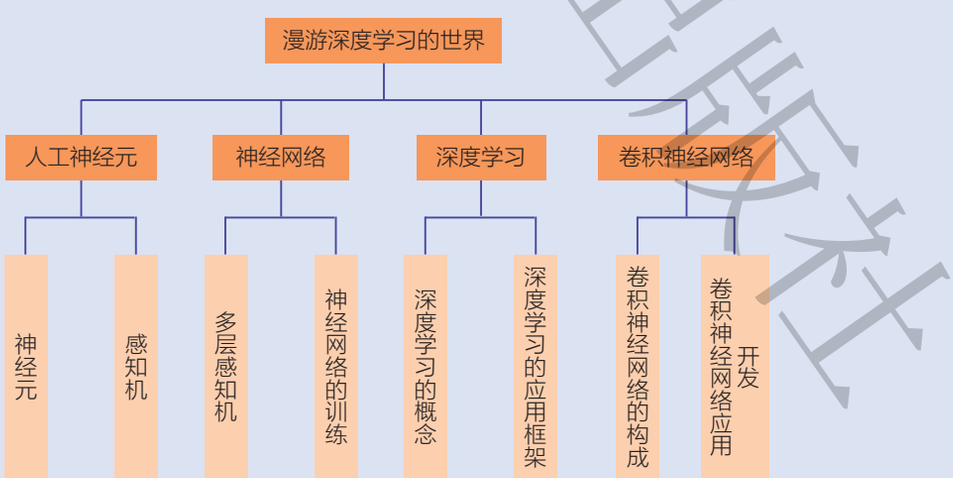
在我们前面的人工智能之旅中，同学们已经初步体验到了新一代人工智能应用的强大威力，一定迫不及待地想要了解其背后更多的技术秘密，本章我们就将带领大家漫游其中的一项核心技术——深度学习的世界。

在新一代人工智能的崛起过程中，深度学习是最核心的技术。现今人工智能的各种突破性应用中几乎都能看到深度学习的影子，而将深度学习这样一种高端科技带入各行各业的助推力量就是各种人工智能开放平台。很多知名的研究机构、企业等将主要的人工智能算法、模型、数据集和典型案例等研发积累汇集成工具库，按照相对简单和统一的模式开放给开发人员及普通爱好者使用。这些开放平台隐藏了艰深的理论和繁杂的技术细节，使普通人工智能使用者和初级开发人员能够很快上手开发自己的应用。由于开放平台包含的内容多经过广泛测试和应用考验，集应用推广与人才培养于一体，所以极大地推动了新一代人工智能的普及。

本章从深度学习的基本单元人工神经元开始，逐层推进，结合几种典型的人工智能开放平台，通过若干现实世界中的应用，直观详细地介绍了深度学习背后的一些基本思想、重要概念、算法设计和编程语言实现。

现在，就让我们开始深度学习的奇妙探索之旅吧！

本章知识结构



项目主题

探索深度学习技术

项·目·情·境

深度学习技术的快速发展和广泛应用造就了人工智能的再次繁荣。本章我们通过体验深度学习的应用，理解深度学习的概念，实现深度学习算法的关键模块，熟悉机器学习应用开发的一般过程；最后，借助人工智能开放平台和案例设计实施一个简单的人工智能应用。

我们常常用像眼睛一样宝贵来强调某一事物的重要性。在本项目中，我们逐层解构机器学习算法如何重要、单个生物神经元、视觉神经网络、卷积神经网络，并用于手写体识别和图像分类。

项·目·任·务

任务 1	任务 2	任务 3	任务 4
以单层感知机模拟与门电路。	以随机梯度下降法实现神经网络的学习。	以卷积神经网络实现 MNIST 手写识别。	使用 VGG19 卷积神经网络实现图像风格迁移。

第一节　人工神经元与单层感知机

人工神经元是生物神经元的一种数学模型，它模拟了生物神经元中处理感知信号的基本机理。感知机则仿照生物神经元对输入信号的处理和传输，通过学习算法实现了对输入信号的调谐。

一、生物神经元和人工神经元

如图 3.1 中左图所示，一个生物神经元由树突、轴突、细胞体和神经末梢组成，其中树突是神经元的输入端，用于接收来自其他神经元的电脉冲信号。一个细胞体与多个树突相连，可以同时接收来自多个树突的脉冲信号。这些输入电信号会在细胞体内外产生一个电压差，当电压差达到一个阈值后，就会在与细胞体相连接的轴突上产生一个神经脉冲信号，这个脉冲信号再由神经末梢输出到其他的神经元的输入端。

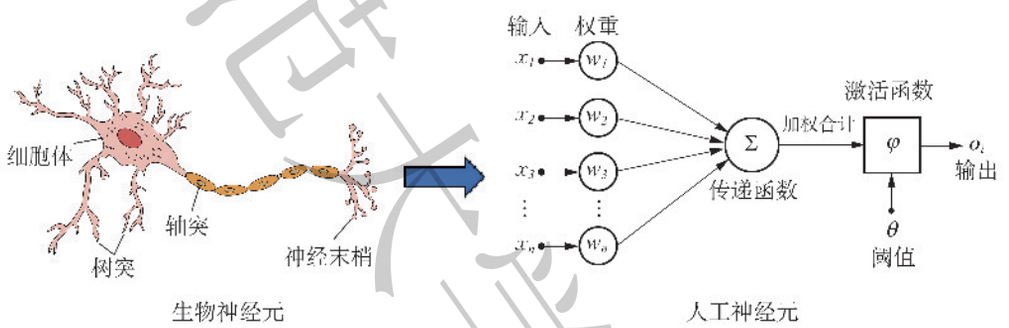


图 3.1　生物神经元（左）与人工神经元（右）

人工神经元是对生物神经元的一种模拟，如图 3.1 中右图所示，用 x_1，…，x_n 表示 n 个输入，模拟与细胞体相连的多个树突。每个输入 x_i 配置一个对应的“权重”w_i，用来平衡和调节不同的输入值，因为我们无法事先确切地了解究竟是哪些树突传来的信号对细胞体而言“更重要”一些。

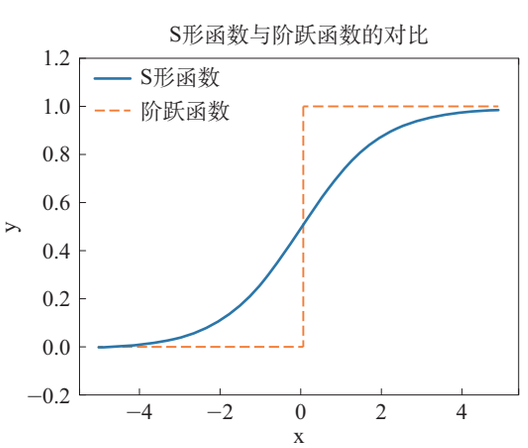


图 3.2　阶跃函数（虚线）与 S 形函数（实线）

同样地，我们无法确切地知道在真正的神经元中，所有这些输入信号是如何进行组合并向外传递的，因此只能采取一种“简单粗暴”的模拟：将加了权重的输入信号累加在一起，即让传递函数为加权求和。

现在，人工神经元需要一个激活函数，模拟达到一定阈值后神经元通过轴突向神经末梢输出脉冲信号的过程。早期人工神经元经常使用的激活函数为阶跃函数（也称为阶梯函数）和 S 形函数（因其形状类似一个大写的 S 而得名），如图 3.2 所示。

项 目 实 践　使用 Python实现阶跃函数与 S形函数

1 阶跃函数 $h(x)$定义为：

$$h(x)=\begin{cases}0, & x\leqslant 0,\\ 1, & x>0。\end{cases}$$

2 使用 Python实现阶跃函数非常简单，比如如下代码给出了一种实现：

```
import numpy as np
def step_function(x):
    if x > 0:
        return 1
    else:
        return 0
```

3 S形函数的定义使用了指数函数 e^x：

$$h(x)=\frac{1}{1+e^{-x}}。$$

从图像（如图 3.2所示）中可以看出：当 x的值越来越小（趋近于负无穷大）时，S形函数的值越来越接近于 0（趋近于 0）；当 x的值越来越大（趋近于正无穷大）时，S形函数的值越来越接近于 1（趋近于 1）。这个特性与真实神经元的脉冲输出形式非常类似。

4 下面是 S形函数的一种 Python实现（使用 NumPy）：

```
import numpy as np
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
```

5 使用 matplotlib库画出阶跃函数和 S形函数的图像（如图 3.2所示）：

```
# 绘制两种激活函数的图形
import numpy as np
import matplotlib.pylab as plt
import matplotlib.font_manager as fm

# 使用中文字体替换默认字体
myfont = fm.FontProperties(fname = r'C:\Windows\Fonts\simsun.ttc')

x = np.arange(-5.0, 5.0, 0.1)
y1 = sigmoid(x)
y2 = step_function(x)

plt.plot(x, y1, label = 'S形函数')
plt.plot(x, y2, linestyle = '--', label = '阶跃函数')
plt.ylim(-0.2, 1.2)

plt.xlabel("x")                          # x轴标签
plt.ylabel("y")                          # y轴标签
plt.title('S形函数与阶跃函数的对比', fontproperties = myfont,fontsize = 12)
plt.legend(loc = 'upper left', prop = myfont)                    # 显示图例

plt.savefig('fig3-2.png', dpi = 400, bbox_inches = 'tight')
plt.show()
```

二、单层感知机

与人工神经元相比，单层感知机不仅有输入、输出和激活机制，还带有明确的产生所有权重参数和偏置的方法，以及根据数据来调整参数的学习算法，因此是一种真正需要学习（或训练）的模型。

一个（单层）感知机是如图 3.3 所示形式的人工神经元：

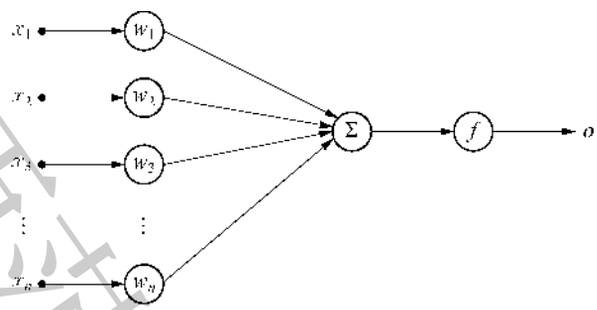


图 3.3　单层感知机的基本结构

其中 x_1，…，x_n 是 n 个输入，w_1，…，w_n 是对应的权重，o 表示输出。若用 θ 表示阈值，f 表示阶跃函数，则感知机从输入到输出的计算公式为：

$$o = f(w_1x_1 + \cdots + w_nx_n - \theta) = \begin{cases} 0, & w_1x_1 + \cdots + w_nx_n \leqslant \theta, \\ 1, & w_1x_1 + \cdots + w_nx_n > \theta。 \end{cases}$$

知　识　延　伸　　**使用向量记号简化表达式**

我们可以使用向量记号来化简感知机从输入到输出的计算公式。首先，我们将 n 个输入 x_1，…，x_n 和 n 个权重 w_1，…，w_n 都写成“列”向量的形式：

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \boldsymbol{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}。$$

则输入的加权和恰好是这两个向量的点积（或内积）：

$$\boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} = w_1x_1 + \cdots + w_nx_n,$$

其中向量 $\boldsymbol{x}$ 右上角的符号 T 表示“转置”运算，即将 $\boldsymbol{x}$ 的行（或列）换成同序号的列（或行）。

现在感知机输出的计算公式可以写为：

$$o = f(\boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w} - \theta) = \begin{cases} 0, & \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w} \leqslant \theta, \\ 1, & \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w} > \theta。 \end{cases}$$

此时的 θ 不仅是阈值，它也能起到调节和平衡输入值的作用，因此有时也被视为一种“偏置量”。

项 目 实 践　感知机实现与门电路

逻辑门电路是构成计算机硬件的基础。逻辑门主要有四种：与门（对应于逻辑与运算）、或门（对应于逻辑或运算）、与非门（对应于逻辑与非运算的复合）及异或门（对应于几种基本逻辑运算的复合）。从理论上讲，如果一种计算模型能实现这四种基本逻辑门电路，就能实现所有的计算。因此，一种模型能否实现四种基本逻辑门电路经常被用作检验模型的理论计算能力的方法。

在本次实践中，我们将证明感知机可以模拟（即实现）与门、或门及与非门，但不能模拟异或门。如第一章所讲到的，这个缺陷曾经引起人们对神经网络的强烈质疑，也是导致人工智能进入第一次寒冬的原因之一。

事实上，这个问题可以通过引进多层感知机来弥补（见本章第二节）。多层感知机不但轻易地解决了基本逻辑门的模拟问题，也为后来一般神经网络的发展奠定了坚实的基础。

我们用感知机来模拟逻辑门电路中的与门。与门（AND gate）是有两个输入和一个输出的门电路，其输入信号和输出信号的对应表（真值表）如表 3.1所示。

表 3.1　输入和输出信号对应表

x_1(输入)	x_2(输入)	y(输出)	x_1(输入)	x_2(输入)	y(输出)
0	0	0	0	1	0
1	0	0	1	1	1

换言之，与门仅在两个输入均为 1时才输出 1，否则均输出 0。我们构造一个两输入的感知机，如图 3.4所示。

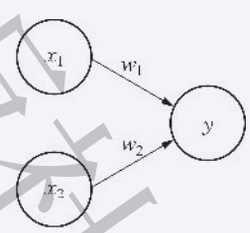


图 3.4　两输入的感知机

用这个感知机表示与门需要做的就是确定能满足上述真值表的各个参数 w_1、w_2、θ 的值。事实上，满足条件的参数组合有无数种。比如，我们取 $w_1 = 0.5$，$w_2 = 0.5$，$\theta = 0.7$，则容易计算：这样设定参数后，仅当 $x_1 = 1$，$x_2 = 1$时，输入信号的加权和为

$$w_1 x_1 + w_2 x_2 = 0.5 + 0.5 = 1 > 0.7 = \theta,$$

因此输出为 1。

下面是这个感知机的 Python实现：

```
#---与函数的实现---
def AND(x1, x2):
    w1, w2, 阈值 = 0.5, 0.5, 0.7          # 使用预设好的权重和阈值。
    输入加权和 = x1*w1 + x2*w2             # 计算加权和。

    if 输入加权和 <= 阈值:                 # 加权和小于或等于阈值，返回0。
        return 0
    elif 输入加权和 > 阈值:                # 加权和大于阈值，返回1。
        return 1
```

探索：

1. 用类似的方法，给出与非门及或门的感知机模拟。

2. 使用 Python 实现模拟与非门及或门的感知机。

如上所述，使用感知机可以很容易实现与门、与非门、或门这三种逻辑电路，但是却无法实现异或门（XOR gate）。这是为什么呢？我们来详细分析一下其中的原因。

对任何含两个输入、一个输出的感知机而言，其输出的一般形式为：

$$y=\begin{cases}0, & w_1x_1+w_2x_2\leqslant\theta,\\ 1, & w_1x_1+w_2x_2>\theta,\end{cases}$$

其中 x_1、x_2 是输入信号，y 是输出信号，w_1、w_2 表示权重，θ 是阈值（或临界值）。

在 (x_1, x_2)-平面上，方程 $0=w_1x_1+w_2x_2-\theta$ 表示一条直线。因此，感知机实际上将位于该直线下方的所有点（包含直线上的点）输出为 0，而将直线上方的所有点（不包含直线上的点）输出为 1。

对于与门、与非门、或门而言，其对应于输出 0和 1的那些输入 x_1、x_2，恰好都可以用 (x_1, x_2)-平面上的一条直线将其分离在直线的两侧，因此可以用感知机来实现。

例如，我们考虑或门的情况。输入 x_1、x_2 中只要有一个为 1，输出就是 1，否则输出为 0。在图 3.5的上图中，标注为小三角形的三个点对应于产生输出 1的输入值，而标注为小圆点的一个点则产生输出 0。

我们可以很轻松地画出一条直线（相当于确定直线方程中的参数 w_1、w_2 和 θ），将产生不同输出的输入值分割在直线的两侧。

请同学们思考一下：是否有方法能简单地确定这条直线（即快速地确定 w_1、w_2 和 θ 的值）？

现在，我们就可以解释为什么异或门不能用单层感知机来模拟。异或门的逻辑运算规则是：仅当 x_1 或者 x_2 中恰好有一个值为 1时，异或门的输出为 1。表示为 (x_1, x_2)-平面上的点如图 3.5中下图所示（小三角形表示产生输出 1的输入值的点，小圆点表示产生输出 0的输入值）。

无论你如何画直线，都无法将圆点和三角形点用直线分离开，使得直线的同一侧只含一类点。这就表明单层感知机无法模拟异或门。

由此也可以看出，单层感知机的局限性在于：它只能解决输入可由直线来分割的一类问题，即“线性问题”。

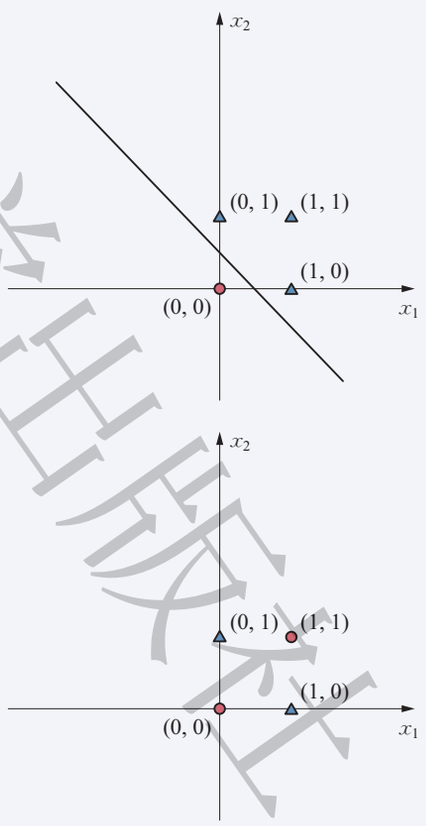


图 3.5 用一条直线分离产生不同输出的点（上），异或门的输入点不能用一条直线分割（下）

第二节　多层感知机与人工神经网络

单层感知机只有输出层的神经元进行激活函数处理，因此其学习能力非常有限，只能表征线性可分的数据（即可以用直线、平面或高维的超平面将数据点完全隔离开的场合），如与门、或门及与非门之类的函数。但对于像异或门这样的非线性可分问题，单层感知机就无能为力了。

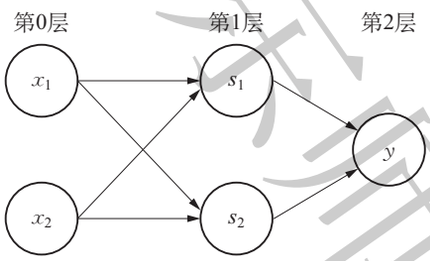


图 3.6　用多层感知机模拟异或门

一、多层感知机

在图 3.6 所示的多层感知机中，如果 s_1 实现了一个输入为 x_1、x_2 的与非门，s_2 实现了输入为 x_1、x_2 的或门，y 实现了输入为 s_1、s_2 的与门，则整个多层感知机的网络就实现了异或门。

我们可以使用如下的 Python 程序来验证这一点：

```
#--异或门（函数）的实现--
def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    return y
```

像图 3.6 这样表示异或门的网络就是多层结构的神经网络：最左边的一列称为第 0 层（输入层），中间一列称为第 1 层（也称为中间层或隐藏层），最右边一列称为第 2 层（输出层）。像这样叠加了多层的感知机的神经网络称为多层感知机。注意：通常我们不把输入层算作一个独立的层，因为毕竟这个层除了把数据输送到网络中来，其他什么也没做。

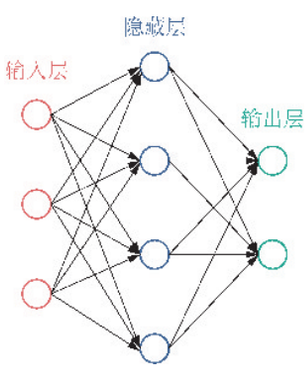


图 3.7　二层感知机

一般地，多层感知机具有如图 3.7 所示的结构。

我们前面用于模拟逻辑电路的单层和多层感知机，其权重都是事先指定好的，这仅对简单问题才有可能。现实中的神经网络（多层感知机）通常涉及成千上万个参数，甚至可以上亿，再想用人工决定参数值的方法是不可能的。因此，对于一般神经网络，我们需要有训练算法（也称为学习算法），其实也就是通过比对感知机的预测输出与真实值的差别，选择某种固定的策略来不断更新权重的值。

下面是训练多层感知机的一种算法（一种著名的反向传播算法的形式）。

第 0 步(初始化)：用 0 或者小随机数值初始化所有的权重和偏置值；
重复以下步骤，直到达到指定的训练次数或参数已达到要求：
第 1 步：计算输入特征和权重值的线性组合(外加上偏置)；
第 2 步：将第 1 步结果输入到激活函数(阶跃函数)，返回一个二元值(激活或抑制)；
第 3 步：对给定学习率使用学习规则，计算权重和偏置的更新值；
第 4 步：将更新值加到新权重值和偏置上.

二、从多层感知机到人工神经网络

神经网络是对多层感知机的推广，因此它们之间有很多共同之处，比如关于多层感知机的术语和训练算法基本上可不加以改变就适用于神经网络，但两者最重要的共同性质是都可以从数据中学习到合适的权重参数。

另一方面，神经网络与多层感知机的最大不同之处在于其激活函数不再限制为阶跃函数。当然，反过来也可以这样讲：(多层)感知机就是以阶跃函数作为激活函数的神经网络。

神经网络中更常使用的激活函数是 S 形函数，这个函数比阶跃函数要复杂很多，使用反向传播学习算法会很低效。但 S 形函数的一个优势是其光滑性(不像阶跃函数那样有一个跳跃间断点)，因此可以用于更强大的分析手段。

与多层感知机一样，神经网络的学习是指从训练数据中自动获取最优权重参数的过程。

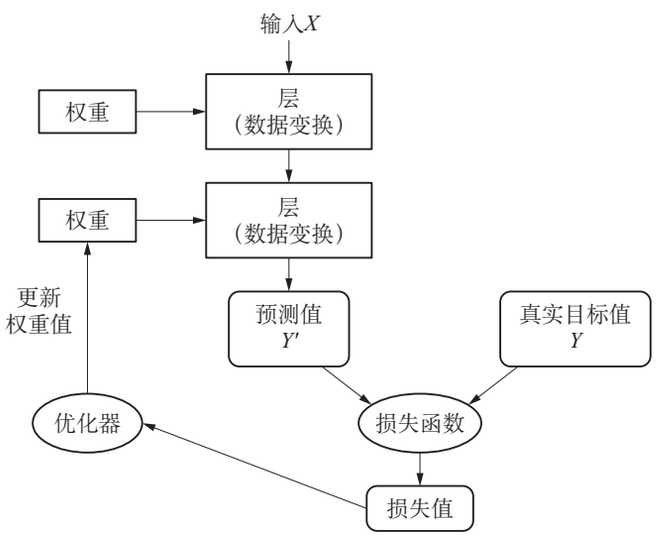


图 3.8　损失函数值衡量输出结果的质量，从而可作为评估神经网络学习性能的指标

1. 损失函数

引入损失函数是为了衡量神经网络学习的性能，同时也作为学习算法寻找最优权重参数的线索。神经网络学习的目的就是要找出能使损失函数值达到最小的那些权重参数。这样一来，一旦确定了损失函数，训练的目标就变成通过选择一组权重参数，使得损失函数的值最小，如图 3.8 所示。在数学中，求函数在某种约束条件下取最小值的问题被称为最优化。

原则上，损失函数可以是任何函数，但

通常都使用比较有意义的函数，如均方误差或交叉熵误差等。

均方误差是最常见的损失函数，它由下式给出：

$$E=\frac{1}{2}\sum_{i=1}^{n}(y_i-t_i)^2,$$

其中 y_i 表示神经网络的输出，t_i 表示实际的输出值，n 是训练数据的个数。

例如，如果某个识别问题产生了一组数据：

$$y=[0.1,\ 0.05,\ 0.6,\ 0.0,\ 0.05,\ 0.1,\ 0.0,\ 0.1,\ 0.0,\ 0.0],$$
$$t=[0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0],$$

则根据公式可计算出其均方误差为 0.097 5。

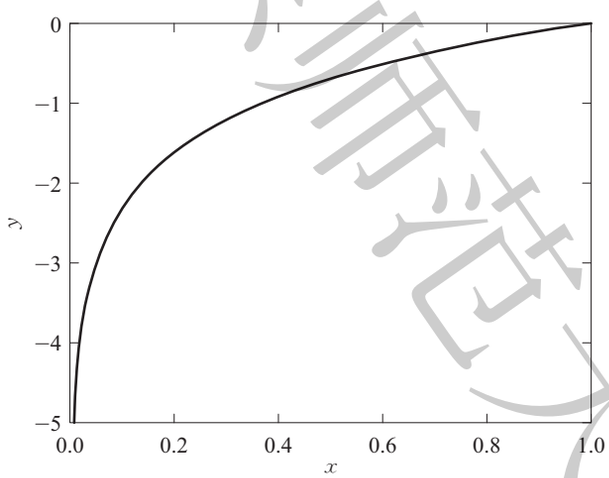


图 3.9　对数函数的图像

交叉熵误差也经常被用作损失函数。采用与前面相同的记号，交叉熵误差定义如下：

$$E=-\sum_{i=1}^{n}t_i\log y_i。$$

根据对数函数的性质（如图 3.9 所示），正确标签所对应位置上的输出值越大，交叉熵误差值越接近于 0；而当输出值为 1 时，交叉熵误差为 0。

2. 梯度下降法

一般神经网络最著名的训练算法是梯度下降法。所谓梯度下降法，就是让损失函数从当前位置的取值沿着下降速度最快的方向（梯度下降的反方向）移动一小段距离，然后在新位置上重新求梯度，再沿新的梯度下降方向移动……如此反复不断地沿梯度下降方向移动，从而逐渐减小损失函数的值。

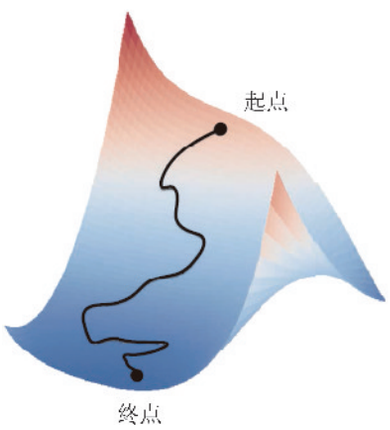


图 3.10　沿梯度方向下降

直观地讲，在某个点的梯度就是函数形状在该点最“陡”的方向，因此沿着这个方向的下降速度最快，如图 3.10 所示。

使用梯度下降法时通常还需要设置一个“学习率”，用来调节损失函数沿梯度（反）方向变化的幅度，防止出现变化幅度太大或太小的情况。

如果梯度的变化幅度过大，就有可能错过损失函数的最低点（如图 3.11 中左图所示）；相反，如果梯度的变化幅度过小，损失函数向最小值点移动的速度过慢，就可能导致算法效率下降（如图 3.11 中右图所示）。

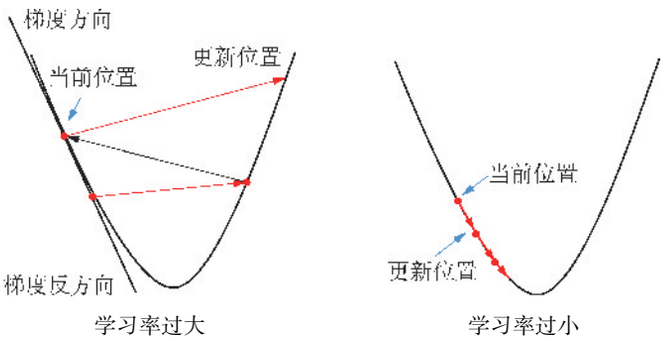


图 3.11　梯度下降法中的学习率

由此可见，在使用梯度下降法训练模型时，学习率过大或过小，都无法让损失函数达到或逼近一个“好的位置”。

学习率需要事先确定为某个值，比如 0.01 或 0.005，然后视学习过程的需要再进行动态调整。

像学习率这样需要预置的参数与神经网络中的其他参数（权重和偏置量）的性质有所不同：神经网络的权重和偏置量之类的参数是通过使用训练数据和学习算法而获得的，因此也叫做“可学习参数”。相比之下，学习率需要人工来设定，这类参数称为“超参数”。

知　识　延　伸　　梯度下降法

我们给出神经网络随机梯度下降法（英文缩写为 SGD，是梯度下降法的一个改良版本）描述如下：

第 1步：从训练数据中随机选出一部分（称为 mini-batch），训练的目标是减小针对 mini-batch损失函数的值。

第 2步（计算梯度）：梯度表示损失函数值减小最快的方向。为了减小 mini-batch损失函数的值，需要求出损失函数关于每个权重的梯度。

第 3步（更新参数）：沿梯度方向对权重参数进行微小更新。

第 4步（重复）：重复上述的步骤 1~3，直到达到所需要的精度或最大迭代次数。

图 3.12所示是使用 SGD 的一个实例，图中圆点表示每次迭代的更新值，显示出其逐渐向最低点靠近。

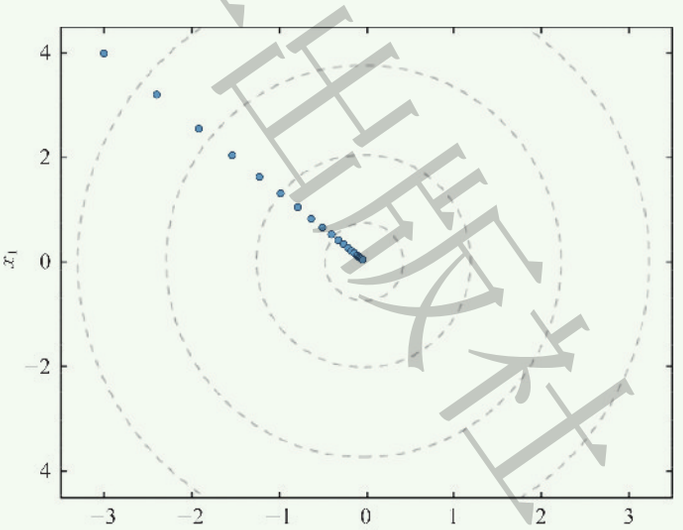


图 3.12　随机梯度下降法的例子

作业练习

使用随机梯度下降法实现神经网络的学习

在本练习中，我们将实现一个简单的 2层神经网络，其输入层有 2个神经元（偏置量为 1）；隐藏层有 6个神经元（偏置量为 1），使用双曲正切 tanh作为激活函数；输出层有 1个神经元，使用 S形函数作为激活函数，见图 3.13。

1.从课程资源包中下载所需要的程序代码。

2.阅读并分析样例代码，调节其中的参数并观察对性能的影响。

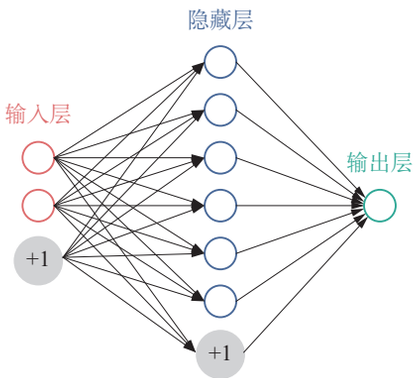


图 3.13　使用 SGD 实现神经网络学习

第三节　深度学习

深度学习是在人工神经网络的基础上，通过更深的结构和更智能的学习算法，达到更好的大数据处理能力的一种计算模型。在这次人工智能的热潮中，深度学习发挥了举足轻重的作用，并被广泛应用于图像识别、文件处理、声音合成、金融科技等各个领域。

体　验　思　考　　**人脸识别是如何实现的？**

小李同学喜欢摄影。他手机上有一个应用，能将包含某人照片的所有照片自动归类，他很喜欢这个给他带来诸多便利的应用。同时，他也在思考：从包含很多人的海量照片库中选出那些只包含某个人的照片，就是人来做也未必能分得很准确，更别说速度了。手机程序是怎么做到的呢？

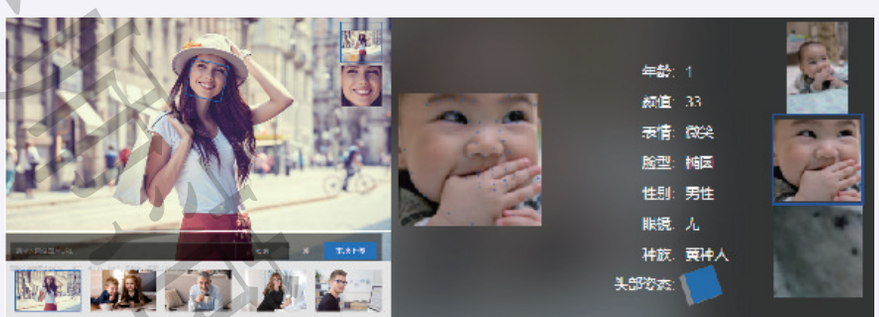


图 3.14　体验人脸识别：上传本地照片（左），识别结果（右）

为此，他打开一个能在线完成一些图像识别任务的应用，上传了一些自己找到的照片做了实验，如图 3.14所示。

小李阅读了该应用中对人脸识别技术原理的介绍，了解到这些功能都是通过深度学习技术实现的。

探索：

1. 尝试使用该应用的其他图像识别功能，如动植物识别、图像中的文字提取等。
2. 上传一张卡通人物图片，看系统是否能正确识别。由此讨论人工智能识别人脸的局限性。

思考：

1. 从技术上看，完成这些应用中的各种识别任务有哪些共性？
2. 我们人类是如何辨别不同物体的？

一、深度学习的概念

深度学习其实说的是“深度神经网络的学习”，因此，深度学习泛指包含很多中间层（隐藏层）的多层神经网络，如图 3. 15 所示。

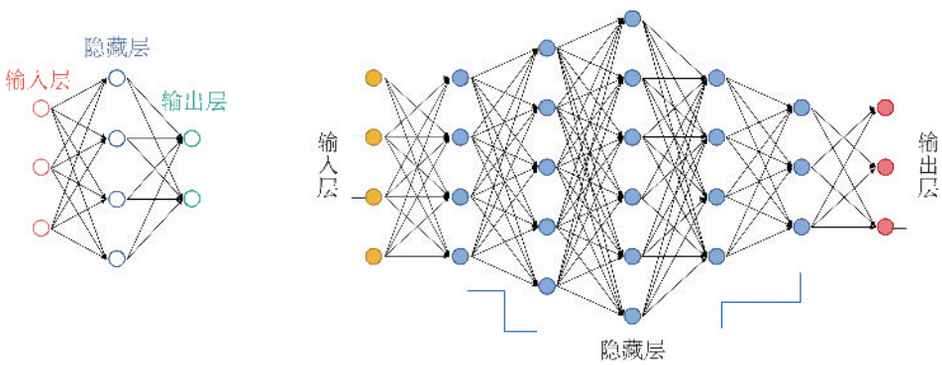


图 3.15　深度神经网络：带有隐藏层的神经网络（左），深度网络（右）

那么，包含多少个隐藏层才算是“深度”的网络呢？这个问题的答案其实并没有严格的限定，通常要根据具体的网络架构而定，而网络架构则由网络中各神经元节点的构成、隐藏层的个数、每个层中神经元的个数、不同层之间神经元的连接模式、所传输信号的类型及连接权重等诸多因素共同决定。在实际应用中所使用的深度网络经常有几十层至数百层不等。

实质上，深度学习不仅增加了中间隐藏层的层次，而且在构成网络的各种元素中也有很多全新的思想，比如本章后面将要讲到的最典型的深度学习架构——卷积神经网络（CNN），就引入了卷积和池化这两种新操作。

仅仅就神经元之间的连接数目而言，大量增加的层（以及每层的神经元数量）使得神经元之间连接的数目和对应的权重数目呈爆炸式增长，使得传统“全连接”型神经网络（即每层的所有神经元与下一层的所有神经元之间均有连接）的训练算法很快变得无效。

探　究　活　动　　**深度所带来的挑战**

多层神经网络是一种全连接的结构，其连接权重的学习算法（如梯度下降法）需要大量算力，这正是导致人工智能领域中神经网络方向在 20世纪 90年代陷入发展瓶颈的主要原因之一。我们通过下列问题来亲身体验一下多层全连接所带来的困扰。

图 3.15的左图是只有一个隐藏层的简单全连接神经网络，它共有 20个连接。

1　图 3.15的右图中，全连接网络里共有多少个连接？

2　假设给图 3.15右图所示网络的第三个隐藏层的后面再增加一个包含 8个神经元的隐藏层，如果使用全连接的话，则共有多少个连接？

为了能进行有效学习，深度网络引入了诸多的创新思想，包括改

变神经元结构、大幅减少连接数目、优化训练算法等多种手段，从而达到在减少计算量的同时不降低网络功能的目的。正所谓“一个好汉三个帮”，即使有如此多的改进，深度学习的有效应用仍然需要超强的并行计算力（以图形处理器 GPU 的方式）及充足的大数据支持（以各种数据集的形式）。

体 验 思 考　　为什么是 GPU，而不是 CPU？

GPU是 graphic processing unit图形处理单元）的缩写，是推动深度学习普及的“三驾马车”之一。

请通过数字化学习平台查找有关 GPU 的信息，了解在深度学习算法中使用 GPU 的原因，以及使用 GPU 所带来的效率上的改进情况。

二、数据集与深度学习

大数据在以深度学习为代表的机器算法的发展中起着至关重要的作用。那么机器学习算法的设计者和使用者是如何找到并使用这些数据的呢？

通常，经由各种渠道积累下来的大数据，会以各种数据集的形式提供给使用者。这些数据集或者因为已经失去时效性和价值而成为学习使用范例，或者是各国政府、企业和机构所公布的各种开放数据，出于学习研究的目的可以免费使用。

对于机器学习和深度学习的初学者而言，获取示例数据集的最方便有效的方法就是使用行业标准测试或竞赛用数据集，这些数据集都经过仔细挑选和整理，并在各种应用和教学中得到了广泛的测试，具有很好的示范性。本书所使用的数据集也采自这些标准数据集。

下面我们简要介绍几个使用最广泛的数据集平台：

（1）Kaggle 数据集。Kaggle 其实是一个全球性的数据建模及数据分析竞赛的平台，学习者、研究者和企业等各种用户（无论是否参与竞赛）均可通过平台下载或发布数据。Kaggle 会定期发布用于机器学习算法竞赛的各种数据集及最佳排名等信息，以方便使用者了解自己所训练的模型能达到的性能。

（2）MNIST 数据集。MNIST 是 modified national institute of standards and technology（意思是修订的 NIST 数据集）的缩写，是机器学习领域历史最悠久的数据集来源之一。深度学习领域的很多最

基础应用的数据集，比如被称为建立机器学习领域第一个真正应用（“Hello World!”应用）的手写数字图像数据库，就出自 MNIST。此外，最早出现的深度网络架构——卷积网络 LeNet 也是首先用于这个数据集。在本章的后面，我们将使用 MNIST 数据集来完成基于 AI 开放平台 Keras 的第一个深度学习应用。

（3）ImageNet 数据集。MNIST 普及了机器学习，将初学者和非专业人员带进了深度学习领域。而对深度学习浪潮真正起到巨大推动作用的则是另一个图像数据集——ImageNet。如其名字所暗示的那样，ImageNet 是一个通过网络众包方式建立起来的庞大图像数据库，包含 1 400 多万幅图片，涵盖了 2 万多个类别。ImageNet 中有超过百万幅的图片上有明确的类别标注和图像中物体位置标注，用于训练各种机器学习的算法。现今，ImageNet 数据集几乎成为检验深度学习在图像相关领域算法性能的“标准”。ImageNet 也开放到 Kaggle 平台上，供学习和研究使用。

（4）程序数据集。除这些开放数据集来源外，在广泛使用的 Python 数据处理扩展包 Pandas 和机器学习扩展包 scikit-learn（也称为 ski-learn）中，也包含了一些用于学习和测试机器学习算法的经典数据集，比如 Iris 数据集等。有些扩展包甚至还有专门用于产生测试用简单数据集的函数。在本章第二节最后训练神经网络的例子中，就使用了 scikit-learn 的内建函数所生成的随机数据集。

相比于上述几个数据集来源，Pandas 和 scikit-learn 中的数据集主要是方便学习用的，其所包含数据集的数据点（数据条目）数目通常较小。

探 究 活 动　探究 MNIST 手写数字数据集

使用机器识别手写字符（阿拉伯数字 0到 9）曾经是人工智能所面临的重要挑战之一，其中 10个手写数字的识别是字符识别中最简单、也是最典型的任务。每个人的书写习惯不同，写出来的数字千差万别，无论是轮廓清晰、字迹工整的，还是潦草涂鸦的，人们只要看上一眼就基本上能认出来是哪个数字。但教会机器自动识别这些数字就困难多了，其中的原因之一是手写的数字会比较潦草，表示特定数字的特征有些模糊，而传统的识别算法则是精确的，不容易处理这种模糊特性。

例如，图 3.16中的手写数字究竟表示的是哪个数？

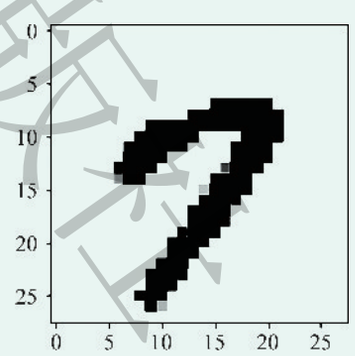


图 3.16　手写数字的一个实例

1995年，美国国家标准技术研究所（NIST）创建了一个手写体数字的数据库，用于研究和测试机器学习和图像处理系统。该数据库经过适当的清洗、正规化并剔除了一些有问题的数据，在1998年开始对外开放，并命名为MNIST数据集。现在，手写数字识别的问题已经解决得非常完美，所以在机器学习领域中，MNIST数据集经常被用于进行教育目的或基准测试，用来检验一个模型是否能正常工作，相当于编程语言中的第一个程序“Hello,World!”。

MNIST数据集包含有70000幅28×28像素的灰度图像（灰度值0~255之间），每个手写体数字是一幅独立的图像，其中60000幅图像带有标签，标明该图像代表的是哪个数字，用于训练模型；另外的10000幅图像用于测试或诊断模型。

图3.17显示了该数据集中的几幅典型图像。

图3.17　MNIST数据集中的若干图像

原始MNIST数据集中的图像格式不太容易使用，本书的资源包中包含了已经转换成*.CSV格式的文件，由于*.CSV格式文件本质上是纯文本文件，使用最简单的文本编辑器就可以打开查看。

图3.18是使用文本编辑器显示的*.CSV格式的MNIST数据集的数据片段。

5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,18,18,18,126,136,175,26,166,255,247,127,0,0,0,0,0,0,0,0,0,0,0,0,30,36,94,154,170,253,253,253,253,253,225,172,2
53,242,195,64,0,0,0,0,0,0,0,0,0,0,0,49,238,253,253,253,253,253,253,253,253,251,93,82,82,56,39,0,0,0,0,0,0,0,0,0,0,0,0,18,219,253,253,253,25
3,253,198,182,247,241,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,80,156,107,253,253,205,11,0,43,154,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,1,154,
253,90,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,139,253,190,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11,190,253,70,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,35,241,225,160,108,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,81,240,253,253,119,25,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,45,186,253,253,150,27,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,16,93,252,253,187,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,249,253,249,64,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,46,130,183,253,253,207,2,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,39,148,229,253,253,253,250,182,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,24,114,221,253,253,253,253,201,78,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,23,66,213,253,253,253,253,198,81,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,18,171,219,253,253,253,253,195,80,9,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,55,172,226,253,253,253,253,244,133,11,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,136,253,253,253,212,135,132,16,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,25,100,122,7,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,33,151,208,252,252,252,146,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,40,152,244,252,253,224,211,252,232,40,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,15,152,239,252,252,252,216,31,37,252,252,60,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,96,252,252,252,252,217,29,0,37,252,252,60,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,181,252,252,220,167,30,0,0,77,252,252,60,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,26,128,58,22,0,0,0,0,100,252,252,60,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,157,252,252,60,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,110,121,122,121,202,252,194,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,53,179,253,253,255,253,253,228,35,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,5,54,227,252,243,228,170,242,252,252,231,117,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6,78,252,252,125,59,0,18,208,252,252,25
2,252,87,7,0,0,0,0,0,0,0,0,0,0,0,0,5,135,252,252,180,16,0,21,203,253,247,129,173,252,252,184,66,49,49,0,0,0,0,0,0,0,0,0,3,136,252,241,106,17,
0,53,200,252,216,65,0,14,72,163,241,252,252,223,0,0,0,0,0,0,0,0,105,252,242,88,18,73,170,244,252,126,29,0,0,0,0,0,89,180,180,37,0,0,0,0,0,0,
0,0,231,252,245,205,216,252,252,252,124,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,207,252,252,252,252,178,116,36,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,13,93,143,121,23,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

7,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,115,121,162,253,253,213,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,63,107,170,251,252,252,252,252,250,214,0,0,0,0,0,0,0,0,0,0,0,0,25,19
2,226,226,241,252,253,202,252,252,252,252,252,225,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,68,223,252,252,252,252,252,39,19,39,65,224,252,252,183,0,0,0,0,
0,0,0,0,0,0,0,0,0,186,252,252,252,245,108,53,0,0,0,150,252,252,220,20,0,0,0,0,0,0,0,0,0,0,0,0,0,70,242,252,252,222,59,0,0,0,0,0,178,252,252,14
1,0,0,0,0,0,0,0,0,0,0,0,185,252,252,194,67,0,0,0,0,17,90,240,252,194,67,0,0,0,0,0,0,0,0,0,0,0,0,0,83,205,190,24,0,0,0,0,0,121,252,252,20
9,24,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77,247,252,248,106,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,253,252,252,102,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,134,255,253,253,39,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6,183,253,252,107,2,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,10,102,252,253,163,16,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,168,252,252,110,2,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,41,252,252,217,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,40,155,252,214,31,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,16
5,252,252,106,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,43,179,252,150,39,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,137,252,221,39,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,67,252,79,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0

图3.18　以*.CSV格式显示的MNIST数据集

如图3.18所示，每个手写数字的图像由28×28=784个数组成（不包含用作分割符号的逗号），加上第一个数表示所代表的真实数字（该图像的标签，图中红色箭头所指的数字）。除第一个数外，其余的数值都在0~255之间，表示28×28图像在该位置上的灰度值。

下列 Python程序片段显示了如何读入 MNIST数据集：

```
# 打开.csv文件，并将其读入到一个Python列表中，文件的行为列表元素。
MNIST数据文件 = open("mnist_train_data.csv", 'r')
MNIST数据列表 = MNIST数据文件.readlines()

# 关闭.csv文件。
MNIST数据文件.close()

# 显示列表中的第一个元素。
MNIST数据列表[0]
```

图 3.19显示的是程序的运行结果：

```
'5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,18,18,18,126,1
36,175,26,166,255,247,127,0,0,0,0,0,0,0,0,0,0,0,0,30,36,94,154,170,253,253,253,253,253,225,172,253,242,195,6
4,0,0,0,0,0,0,0,0,0,0,0,49,238,253,253,253,253,253,253,253,253,251,93,82,82,56,39,0,0,0,0,0,0,0,0,0,0,0,0,1
8,219,253,253,253,253,253,198,182,247,241,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,80,156,107,253,253,205,11,0,4
3,154,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,1,154,253,90,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,139,253,190,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11,190,253,70,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,35,241,225,160,108,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,81,240,253,253,119,2
5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,45,186,253,253,150,27,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,16,93,252,253,187,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,249,253,249,64,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,46,130,183,253,253,207,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,39,148,229,253,253,
253,250,182,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,24,114,221,253,253,253,253,201,78,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,23,66,213,253,253,253,253,198,81,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,18,171,219,253,253,253,253,195,8
0,9,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,55,172,226,253,253,253,253,244,133,11,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,136,253,253,253,212,135,132,16,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0\n'
```

图 3.19 以＊.CSV格式显示的 MNIST数据集

上述程序虽然可以显示 MNIST数据集中任意数据项，但却是以文本方式显示的，效果相当于把原来 28×28的二维图像“摊平”为一维的数据。

下列 Python代码（注意要接着前面的程序来运行）可以将 MNIST数据集中的每个小图像真正以图像的方式显示出来：

```
# 取出一条记录中的数据，将其重排为28*28数组，并作为图像画出来。
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

# 选择列表中的一个元素（一行数据），加入分隔符","。
选定的数据行 = MNIST数据列表[15].split(',')

# 从"选定的数据行"中剔除第一个值（标签值）。
截取的数据部分 = np.asfarray(选定的数据行[1:])

# 将"截取的数据部分"重塑为28x28的二维数组。
重塑的二维图像数据 = 截取的数据部分.reshape((28, 28))

plt.imshow(重塑的二维图像数据, cmap = 'Greys', interpolation = 'None')
```

现在你可以尝试修改程序，以文本或图像的方式显示 MNIST数据集中的任何内容。

项目练习：在本书的配套资源库中，有一个使用 MNIST数据集训练的基本神经网络模型的完整代码，请同学们课下调试并运行测试数据，并与本章后面使用其他 AI开放平台开发的手写数字识别程序进行对比。

技 术 支 持　　Python语言的扩展库 Pandas和 scikit-learn

1 Pandas

Pandas是 Python的一个用于处理表格型数据(二维)及时间级数(一维)的库,包含很多常用函数,功能上类似于 Excel电子表格的计算。

Pandas中有两种最重要的数据对象:Series和 DataFrame。其中 Series是带有索引(从 0开始)的一维列表,比如按特定的时间间隔所采集到的一维数据(称为时间级数),如数字声音信号;而 DataFrame(数据帧)则是类似于普通电子表格的二维表对象。

2 scikit-learn(即前面曾经用过的 sklearn)

scikit-learn实现了常用的机器学习算法,尤其是基于统计的算法,包括分类、回归、聚类、决策树等。

使用 scikit-learn实现机器学习应用非常简单,只需将适当的数据提供给表示某个模型的函数,就能训练模型。

1. 数据集的使用

在实际应用中,数据集要根据需要划分为训练集和测试集,这个过程称为数据划分。其中,训练集用于训练(也称为学习)机器学习模型中的各种参数,以达到最佳效果;测试集用于检验已经训练好的模型解决当前问题的性能。

训练集和测试集的划分比例默认为 3∶1,即训练集的数据点数量占整个数据集的 75%,测试集占 25%。注意:训练集和测试集的数据一般不允许有公共的数据点。

2. 开发机器学习应用的步骤

体 验 思 考　　机器学习编程与传统计算机编程方法的区别

思考对比机器学习与人类学习之间的相似之处与不同之处。

1 如何比较机器学习与人类学习?

2 机器学习在哪些方面可能优于人类学习?在哪些方面难以超越人类学习?

传统计算机编程或问题求解是面向算法的:我们针对求解某个问题的要求,设计用于处理数据的若干规则和一系列步骤(算法),然后用一种计算机编程语言实现这些算法(如图 3.20 中上图所示)。与

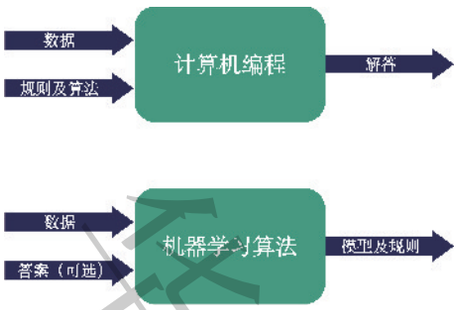


图 3.20　传统计算机编程与机器学习编程的对比

传统计算机编程不同，机器学习编程的显著特点是“通过数据来训练模型”：针对求解某个问题的要求，准备好以往处理这类问题所获得的数据（集），选择若干可能用于解决这类问题的模型（或算法、规则），使用数据集对模型的各种参数进行训练，以获得最佳的问题解决模型（如图 3.20 中下图所示）。

通过使用数据集来训练模型参数，机器学习算法将以往解决同类问题所获得的经验“教给”了模型，使之在解决当前的问题时性能达到最优。

知　识　延　伸　　**机器学习应用开发流程**

机器学习应用是一类特别的智能应用，因此也是信息系统的一种。根据信息系统开发的一般流程，结合机器学习应用的特点，可以总结出机器学习应用开发的一般流程，如图 3.21所示。

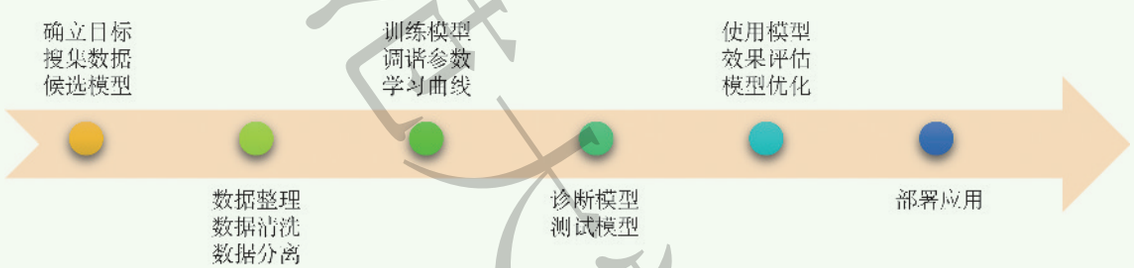


图 3.21　机器学习应用开发流程

我们在本书中仅涉及简单的机器学习应用程序，对于这类应用，其具体开发步骤可简化如下：

（1）确立目标。明确我们的应用程序要解决什么问题，或者更明确地讲，针对当前的问题，需要解决到什么程度，允许有多大的误差才算是得到了解决。

（2）准备数据。搜集为解决问题而获得的数据集，了解数据集中数据的背景和特点，尽量选择适用的模型，并防止掉入因果陷阱。

（3）数据预处理。通常，数据预处理阶段分成两步。第一步，去除或矫正数据集（或数据点）中的异常项及当前问题用不到的那些特征。这项工作也称为数据整理，或者更形象地称为数据清洗。第二步，根据需要，将经整理/清洗后的数据集划分为训练数据集和测试数据集。

（4）选择并训练模型。选择一个已经被证明是有效的机器学习

模型/算法，用训练集数据对模型中的各种参数进行训练（学习），以获得最佳配置的模型。

（5）评估模型性能。用测试数据集的数据对训练好的模型进行测试，评估其解决当前问题的性能。

三、深度学习框架

Python 的扩展库 scikit-learn 虽然实现了机器学习的很多算法，但绝大多数都属于经典的基于统计的算法，很少涉及深度学习的内容。此外，scikit-learn 的计算基本依赖于 Python 的数值计算库 NumPy，没有针对机器学习中多维数组的高强度计算进行专门优化，在实施真实的应用时显得有些力不从心。

深度学习框架由国际上有影响力的企业或机构，将其在该领域中的最新研究开发成果用工具包及可调用编程接口等形式，开放给普通用户使用。这些框架隐藏了复杂的深度学习算法和冗长的实现代码，使用户无需深度学习的专门知识就能学习或者开发出相当专业的、效果惊人的 AI 应用。

下文概述了当前几种最常用的深度学习框架。

1. TensorFlow

TensorFlow 是最早开源的，也是目前最流行的深度学习框架之一。TensorFlow 结合了其他开源框架的一些设计思想，使用张量（tensor）为基本计算原语，高效实现了多维数组运算。此外，TensorFlow 还将计算图的概念引入到算法设计中，使高维计算变得更形象直观。

知 识 延 伸　　张量与计算图

从字面上理解，TensorFlow就是“张量流”的意思，比喻张量在计算图中“流动”的样子。那什么是张量呢？什么又是计算图呢？

简单地讲，张量就是多维数组。一维张量就是一维数组，二维张量就是二维数组（也称为矩阵），……，N维张量就是 N维数组，等等。在 TensorFlow中，所有参与计算的值都是某种张量。

比如，下列指令定义了一个 NumPy的三维张量：

```
import numpy as np
t=[[[1, 2, 3], [2, 3, 4]], [[0, -1, 2], [2, -5, -7]], [[1, 2, 3], [2, 3, 4]]]
np.shape(t)
```

其中“np.shape(t)”显示数组 t的“形状”，即该数组在每个维度上元素的个数。该例中显示为 (3,2,3)，表示这个三维张量在三个维度上的元素分别为 3个、2个、3个。

计算图也称为数据流图或计算流图，是用来表示计算过程的有向图。在数据流图中，节点表示某种数学运算或对数据所进行的变换，节点之间的有向边表示节点之间通过的数据。

例如，如图 3.22所示的计算图表示运算 $z=(a+b)\times c=d\times c$。

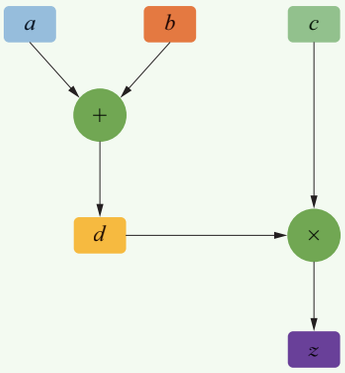


图 3.22 计算图

项 目 实 践 安装 TensorFlow

1 在命令行下运行下列指令：

```
pip install - - upgrade tensorflow%无 GPU 版本%
```

或者

```
pip install - - upgrade tensorflow-gpu%带 GPU 版本%
```

2 运行你的第一个 TensorFlow程序。

键入下列 Python指令，检验安装是否成功：

```
import tensorflow as tf
hello = tf.constant("Hello, TensorFlow!")
sess = tf.Sesstion()
print(run.sess(hello))
```

注意： 安装完全版的 TensorFlow，需要运行 64位操作系统。

TensorFlow 也有网页版本——tensorflow.js，支持 JavaScript 语言，可利用现有模型或者重新训练的模型，无需编程或仅需少量编程就能开发网页上运行的深度学习应用。tensorflow.js 示范案例包括：

真实世界的表情包、基于网络相机的吃豆子游戏、教会计算机识别图片与语音等。

2. Theano 与 Keras

Theano 是第一批发布的神经网络和深度学习框架，同时也引进了很多原始思想，包括透明使用 GPU 方法的张量和计算图等。Theano 直接建立在 Python 的数值计算库 NumPy 和高精度数值计算库 SciPy 的基础上，可以高效地处理与多维数组相关的数学表达式，特别是其实现的高速符号微分方法（梯度下降法的核心）和最优化算法（用于计算损失函数），对于神经网络而言具有非常重要的意义。

2017 年 9 月，Theano 宣布在其 1.0 版本发布后不再做重大更新。尽管如此，Theano 仍然是一个功能非常强大的深度学习库，可以支撑关于神经网络和深度学习的常规学习、测试和研究。

Keras 是用 Python 编写的一个高级神经网络 API（应用程序编程接口），能够和 TensorFlow、Theano 或 CNTK（见“知识延伸　其他深度学习框架”）配合使用。与 TensorFlow 和 Theano 这些更偏向“底层”实现的框架相比，Keras 更像是一个整合了其他框架能力的、更和蔼可亲的“前台”，使机器学习模型的训练更加容易。

项　目　实　践　　**安装 Keras**

Keras建立在其他深度学习框架之上，因此在安装 Keras之前，需要至少安装了 TensorFlow、Theano 或 CNTK之中的一个框架，我们假定已经安装了 TensorFlow。除此之外，Keras的安装过程与 TensorFlow类似，都可以使用 pip来完成。

在命令行状态下运行下列指令：

```
pip install keras
```

图 3.23 显示了 Keras、Theano 与 Python 语言及其数值计算扩展库之间的依赖关系。

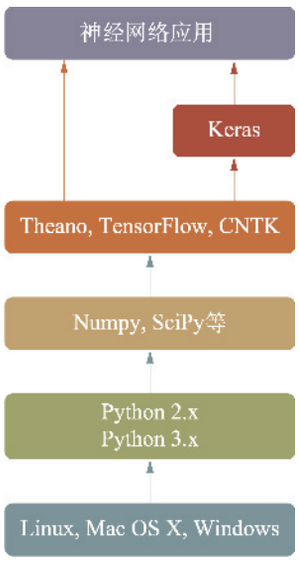


图 3.23　Keras、Theano与 Python语言及其数值计算扩展库之间的依赖关系

3. PaddlePaddle

PaddlePaddle是一种高效、灵活、可扩展的深度学习平台，其旨在将深度学习方法应用于各种场景中。PaddlePaddle Suite（PPS）则是面向工业应用的全功能深度学习套件（如图 3.24 所示），支持企业和开发者自主构建基于深度学习的智能应用。

图 3.24　PaddlePaddle Suite的核心框架

知 识 延 伸　其他深度学习框架

1 CNTK

CNTK是 computational network toolkit的缩写，现在称为 microsoft cognitive toolkit是一个通用开源深度学习工具包，尤其适合开发与自然语言处理相关的机器学习应用。与 TensorFlow类似，CNTK将神经网络描述成一个有向图，图的叶节点代表输入或者网络的参数，其他节点则表示计算步骤。

2 Caffe

Caffe也是一个非常普及的现代深度学习框架，长于模型表达能力、速度和模块化，其后代 Caffe2在Caffe的基础上进行了多项改进。

3 PyTorch

PyTorch也是常用的深度学习框架，具有与 Python紧密融合、支持 GPU增强张量计算和动态神经网络框架等特点。

项 目 实 践　PyTorch安装

PyTorch是 2017年开源的一个深度学习新框架。相比其他同类框架，PyTorch具有如下特点：

- 与 Python编程语言，特别是与 NumPy紧密集成，学习难度大大降低；
- 支持在 GPU(及多 GPU)上直接定义并计算 PyTorch张量；
- 可生成并导出已训练好的机器学习模型，便于应用部署；
- 支持动态计算图生成。

安装 PyTorch的步骤如下：

1.进入 PyTorch官网并找到安装界面；

2.在安装阵列中选择要安装的 PyTorch版本、计算机所运行的操作系统、GPU版本、使用的 Python版本、CUDA版本等信息(如图 3.25所示，红色的框表示已确定的选项)，安装引导程序会自动生成安装指令，安装指令出现在安装阵列的最后一行：Run this Command:

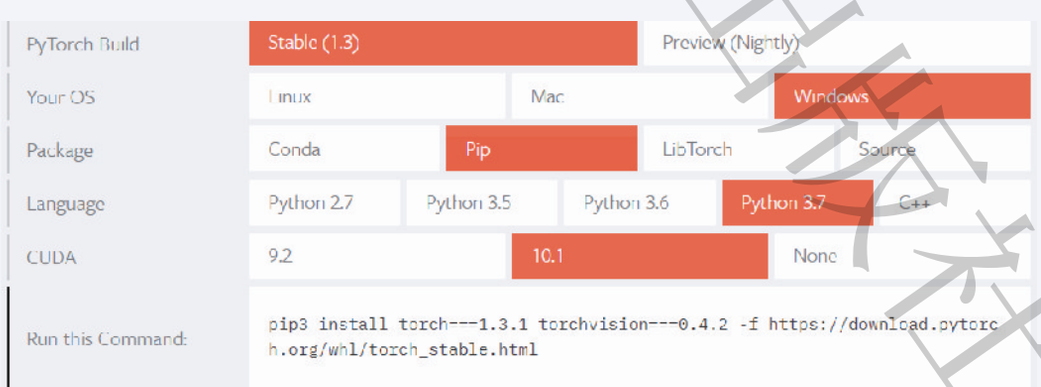


图 3.25

3.进入终端环境，并运行安装指令。

第四节　卷积神经网络

卷积神经网络(convolutional neural network,缩写为 CNN)是一种特殊的神经网络结构,是深度学习领域的重要基石,目前基于深度学习的现代机器学习应用几乎都以 CNN 为基础。

CNN 在视觉信息处理方面的表现尤为出色。早在 1989 年,CNN 就被用于手写数字分类和脸部识别,但并未受到太多关注。直到 2012 年,基于 CNN 的深度学习算法在 ImageNet 的图像分类竞赛中取得了突破性进展,直接推动了人工智能第三次浪潮的兴起。2012 年以后,CNN 被广泛应用于图像识别、语音识别等各种场合中。

CNN 网络的整体结构及框架与之前介绍的神经网络相似,可以像搭积木一样通过组装多个神经元的层来构建越来越复杂的深度网络。不过,在 CNN 中出现了两种新的构造——卷积层(convolution layer)和池化层(pooling layer)。此外,CNN 中所使用的激活函数也与普通神经网络不同。

一、卷积层

深度学习通过一系列能完成不同任务的神经元层,将复杂任务(如图像识别)分解为一个个相对更简单、更底层的工作。比如,一些层只识别图像中的局部线条信息,像人脸的局部轮廓、植物叶子边缘等,而其他的层可能只负责图像中的那些色彩或亮度信息。每个层所“发现”的这些信息在下一个层中被汇集到一起继续进行分析。

普通神经网络存在以下问题:首先,随着隐藏层的增加,普通的全连接神经网络会面临权重参数在数量上和训练上的“爆炸”问题。此外,全连接神经网络还存在另一个严重问题:忽略了输入数据的“形状”。举例来讲,图像数据通常包含着图像的高度、长度及通道等信息,因此其形状是三维的。使用全连接层进行处理时,需要将这样的三维数据“摊平”为一维数据。使用网络感知机处理 MNIST 手写体数字数据集时,就是将形状为(1, 28, 28)的三维图像数据(1 个通道、28 像素高、28 像素长)先排成 1 列、784 个数据,然后以一维数据的形式来输入的。

知 识 延 伸

数据维度、数据语义与数据降维

多维数据通常都包含很多有价值的附加信息，包括有关数据来源、属性、应用场景，以及其所表达的内容等方面的信息，这些信息统称为数据语义。

即使是一维的数据（数组或数值序列）也可以包含语义，比如一维数组中数据元素的先后顺序可能反映了采集数据的时间次序。三维形状的图像数据中通常包含重要的空间位置信息。比如，空间上相邻的像素其值也应该接近，RBG 各个通道之间各自有密切关联性，相距较远的像素之间没有关联等。因此，三维形状数据中可能隐藏着很有提取价值的本质模式。全连接层会完全忽视数据形状，将所有输入数据均视为神经元的输入等同处理，因此会白白丢掉语义信息，而这些信息对于完成任务可能是至关重要的。

数据语义在我们更好地处理、管理和应用数据方面具有很大的价值。

显而易见，数据的维度越高，处理数据的算法和计算就越复杂，因此在很多场合，需要在尽可能保留数据语义的前提下进行“数据降维”，即将多维数据投影到较低维度的空间上。数据降维是改善机器学习算法效率的一类重要方法。

针对以上问题，卷积神经网络被提出来了。何为卷积呢？想象有一个人想仔细检查一幅图像。为此，他准备了一组倍数不同的放大镜。他从图像的左上角开始逐渐向右下方扫描，直到看完整个图像。每次他只能看到图像的一个局部小块。

开始的时候由于放大倍数大，他看到的只是一些堆在一起的像素块，但仍然看得出这些像素在颜色上的不同。现在，到了下一层，他先把上次扫描的各个小图像块拼接起来，构成一幅与原图相匹配的完整图像，在上面反映出原图像中的各种物体的边缘等信息。然后，他又拿起另一个“放大镜”，准备把这些物体一个一个分离出来，于是再次从左上角开始扫描图像……

基本上，这就是一个卷积层所要做的事情，卷积层使用的“放大镜”是一种操作，称为“卷积核”。其实卷积核更像是一种可以局部放大的滤镜，在放大的同时还能滤掉一些内容，因此卷积核也被称为“滤波器”。与普通神经网络不同，卷积层可以保持原始数据中的形状信息不变，当输入三维的图像数据时，卷积层不但以三维形式接收输入数据，还同样以三维数据的形式将处理结果输出至下一层。因此，卷积层能够利用甚至理解图像的语义。

CNN 中的卷积操作就是依次对图像作用卷积核，即选择一小片区域，过滤掉该区域图像中的一些内容，保留并突显当前所关注的内容，比如图像中各个物体的边缘信息等。显然，使用不同的卷积核/滤波器会产生不同效果。卷积操作的计算过程参见图 3.26。

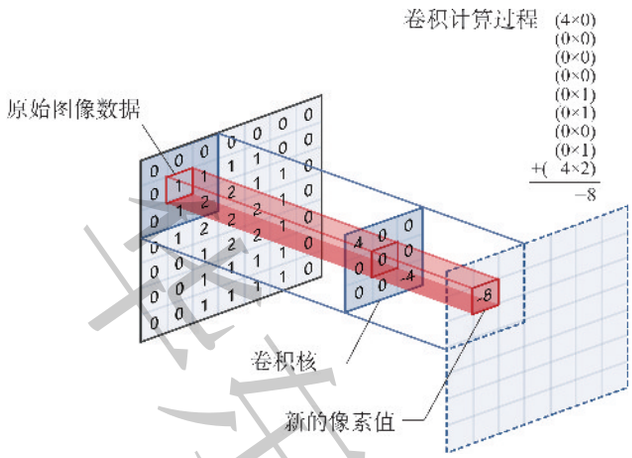


图 3.26　卷积操作与卷积核

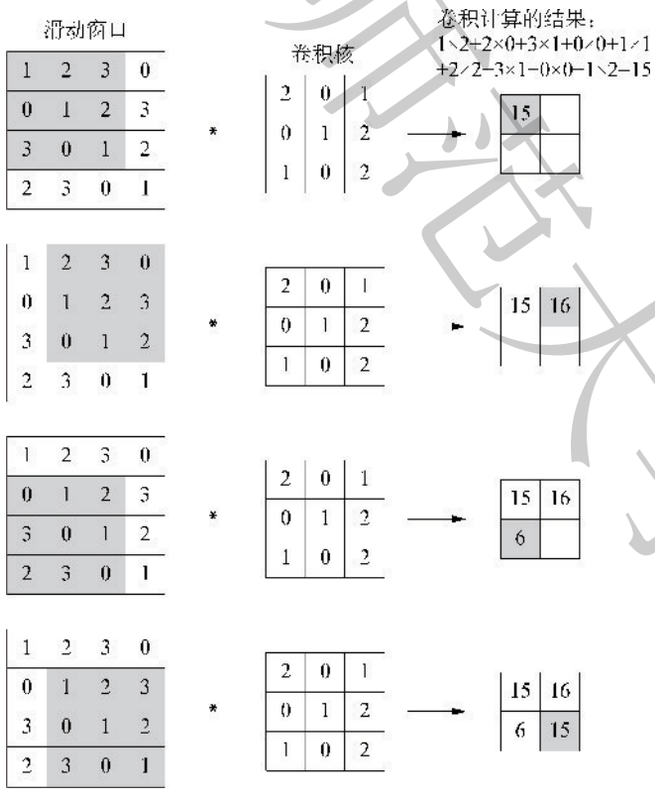


图 3.27　滑动窗口 (其中“*”表示卷积计算)

卷积核的尺寸一般比输入数据的尺寸要小，要对整个输入图像实施卷积操作，需要先选择一个“滑动窗口”，窗口的尺寸与卷积核尺寸相同，然后让窗口从左上角开始依次向右下角“滑动”。滑动窗口每进入一个新位置（窗口），滤波器与覆盖在该窗口下的那部分数据进行“卷积和”运算，即将每个位置上卷积核的元素与输入数据的对应元素相乘，然后再求和（如图 3.26 所示）。每个滑动窗口的运算结果依次保存到一个输出的相应位置上。将这个过程对输入数据的所有位置都进行一遍，就可以得到卷积运算的输出。

图 3.27 显示了滑动窗口的移动次序及相应的卷积操作。

卷积操作将卷积核每个位置上的值与原始输入数据对应的值相乘后再相加，因此卷积核的值与普通神经网络的权重很相像。由于这个原因，卷积核学习与权重学习的算法也非常类似，如都可以使用误差反向传播算法等。图 3.28 显示了一组 5×5 的卷积核，其取值均经过了适当“正规化”，即将最小值对应为黑色（0），最大值对应为白色（255）。

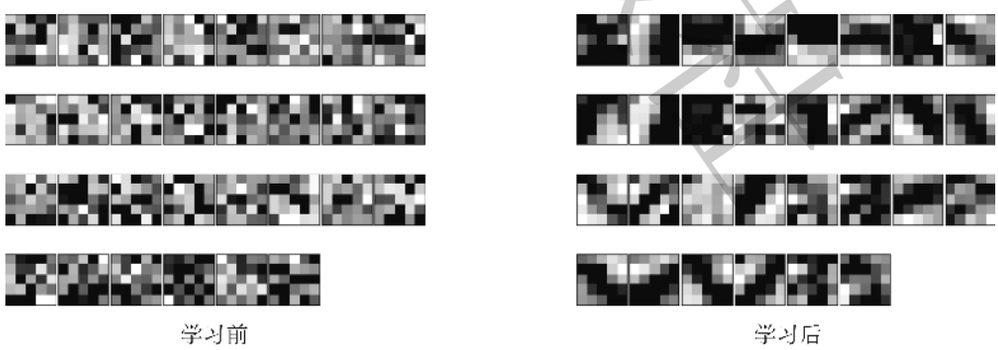


图 3.28　卷积核的学习：左图为随机赋值，右图则是经学习后得到的值

项 目 实 践　　卷积操作的可视化

图 3.28的右图中，学习好的卷积核能识别出图像的边缘（颜色变化的分界线）和色斑块（局部块状区域）等较“低级”的特征。我们从中选择两个典型的卷积核：一个卷积核（滤波器 1）的左半部分大致呈白色，右半部分大致呈黑色；另一个卷积核（滤波器 2）则上半部分大致呈黑色，下半部分大致呈白色。将这两个滤波器分别作用于一幅图像，我们会看到滤波器 1对图像中竖直方向上的边缘有响应，而滤波器 2会对图像中水平方向上的边缘有响应，如图 3.29所示。

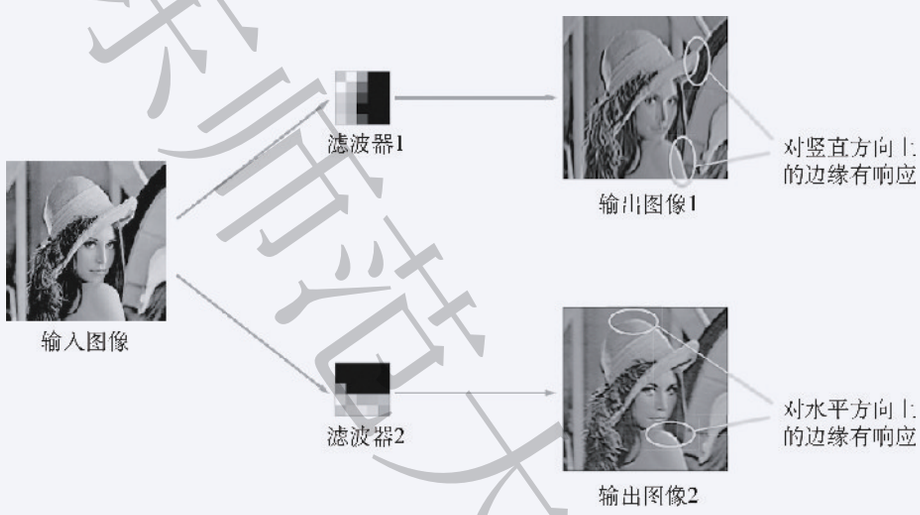


图 3.29　利用卷积核增强图像边缘

请同学们进一步思考并探索：

如果用图 3.28右图中的其他卷积核作用图像，可能会产生哪些效果呢？

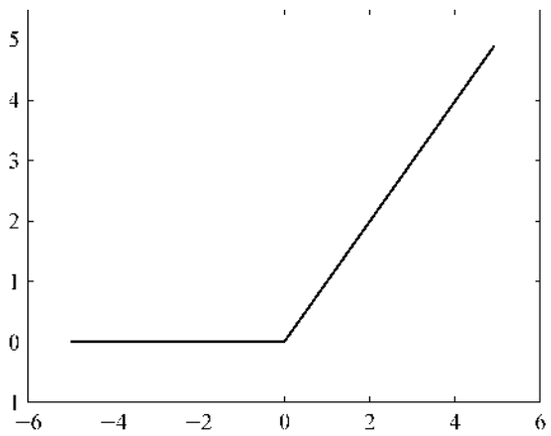


图 3.30　激活函数 ReLU 的图像

二、激活函数

与普通神经网络类似，通过卷积层特征映射的输出数据在传递到下一层之前，会作用一个激活函数。CNN 一般不使用普通神经网络中那些常用的激活函数，而是使用一种名为 ReLU 的激活函数。

ReLU 函数的定义非常简单：当 $x > 0$ 时，ReLU 取值为 x；而当 $x \leqslant 0$ 时，ReLU 取值为 0。函数 ReLU 的图像如图 3.30 所示。

项 目 实 践　　ReLU 激活函数的 Python实现

与其简单的定义相比，函数 ReLU 的 Python实现甚至更简单！这得益于 Python强大的数组操作。ReLU函数的下列实现使用了 NumPy的 maximum函数，即从输入的诸数值中选择最大值输出。

1 运行如下定义的 Python函数：

```
import numpy as np

def relu(x):
    return np.maximum(x)
```

2 使用 matplotlib绘制 ReLU的图像。

大多数深度学习框架都使用自身的高效计算引擎来实现 ReLU函数。

ReLU 函数兼具阶跃函数和 S 形函数两者的优点：差不多与阶跃函数一样简单，同时又像 S 形函数那样是“连续的”，而且几乎是“光滑的”（只有一个不光滑的点）。

ReLU 函数只是屏蔽掉输入中那些负的值，保留所有非负值不变，因此其作用到图像数据上的视觉效果很容易展现出来，如图 3.31 所示。

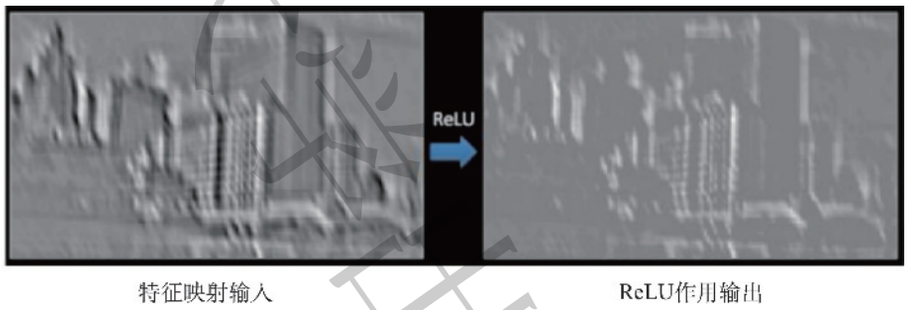


图 3.31 作用 ReLU 激活函数：滤掉图像中黑色的部分（对应于负值的那些像素值）

ReLU 是英文 rectified linear unit 的缩写，意思是“经校正的线性单元”。观察一下 ReLU 作用输出的图像，你觉得这个名称是不是很贴切呢？

三、池化层

CNN 的卷积层后面一般还跟着一个池化层。池化操作从某个像素的周边小区域选择一个典型值代替这块小区域的所有像素值，是一种缩减数据尺寸的简单运算，也称为子采样（subsampling）。

如果把相邻几个位置的值用其中最大的值替换，就叫做最大池化

(max pooling)。最大池化在缩减数据尺寸的同时，增大了周边相邻像素值间的差异，使图像的边缘更突出。当然，有时也会需要达到相反的目的，使边缘与周边的区别更模糊，这时就要使用平均池化(average pooling)。

在图 3.32 中，我们依次将每个 2×2 像素区域缩减为 1 个像素区域，用其中的最大值作为新像素值，以缩减图像尺寸。

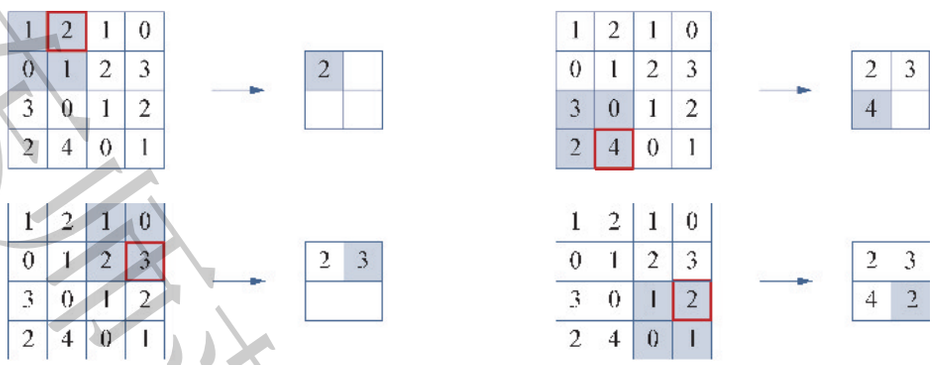


图 3.32　最大池化操作

池化操作与卷积操作都要使用滑动窗口。滑动窗口每次需要向右或者向下移动若干单位，这两个值一般取相同的值。滑动窗口的这个移动值称为步幅。一个 CNN 中可能会有两种步幅：卷积步幅和池化步幅。

例如，图 3.27 中的卷积操作例子是按照步幅 1 向右和向下滑动窗口的，而图 3.32 中池化操作的滑动窗口则使用了步幅 2。

在进行池化操作时，一般会将步幅设定成与池化窗口大小相同的值。比如，3×3 池化窗口的步幅也会设为 3。

知　识　延　伸　　池化层的特点

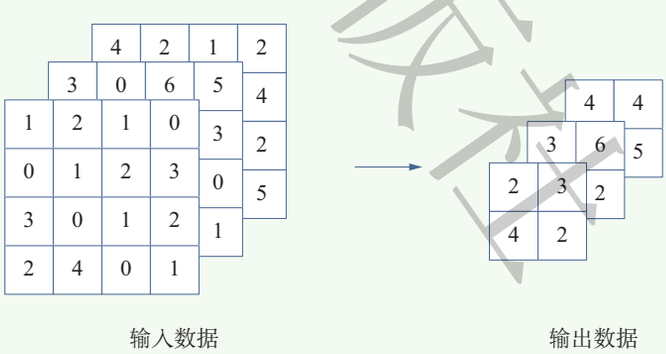
在 CNN 中，虽然也把池化操作算作一个“层”，但这个层与卷积层和激活函数层有所不同，似乎有点“超脱”。具体地讲，池化层具有下列特点：

1　无学习参数

与卷积层不同，池化层只是从目标区域中取出最大值(或平均值)，所以不存在需要学习或训练的参数。

2　通道无关性

输入数据经过池化操作后，产生的输出数据仍在原通道，不会发生变化，换言之，池化操作是独立于通道的，如图 3.33所示。



输入数据　　输出数据

图 3.33　池化操作保持通道信息不变

3 对像素值的局部微小变化具有“健壮性”

当输入数据局部区域的像素值发生微小改变或偏移时，池化仍会返回相同结果。这种能抵抗周边微小变化的稳定性称为健壮性或鲁棒性 如图 3.34所示。

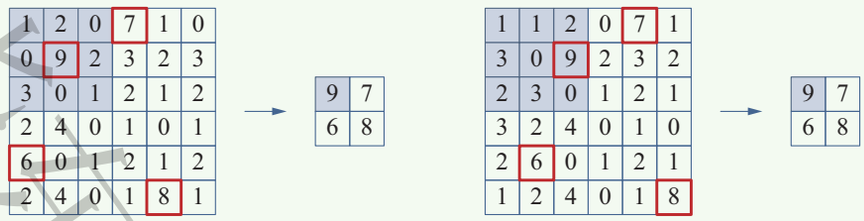


图 3.34 池化操作的健壮性

最大池化作用于特征映射(即卷积层和激活函数层作用)的输出，在保留特征映射的重要特征基础上，缩减了数据尺寸和需要训练的网络参数的数量。此外，最大池化对于输入图像的畸变、失真和平移等具有不变性。

四、卷积神经网络的架构

前面我们讨论了卷积神经网络的主要元素，以下介绍在视觉信息处理领域最常见的卷积神经网络架构，并描述一些著名卷积神经网络的构造特点。

卷积神经网络一般由五个基本成分组成，分别为输入层、卷积层、激活函数层、池化层和全连接层，其中卷积层、激活函数层、池化层经常组合在一起，共同构成整个 CNN 的“隐藏层”，如图 3.35 所示。

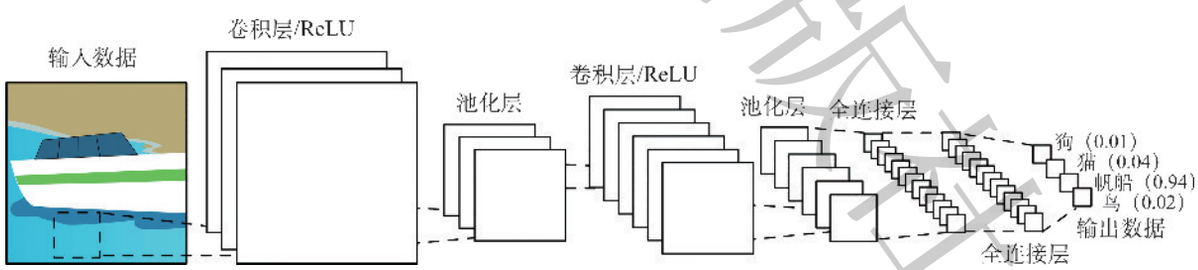


图 3.35 卷积神经网络的架构

表 3.2 列举了 CNN 中每个基本成分所完成的具体功能。

表 3.2　CNN各层的主要功能

层	功　能
输入层	输入图像，保留图像数据结构信息
卷积层	从输入图像中提取特征信息
激活函数层	神经元激活机制
池化层	压缩数据并保留重要特征
全连接层	根据任务实施分类

项　目　实　践　使用 Keras实现 CNN架构

使用 Keras可以非常方便地搭建起一个 CNN的架构。

1 从课程资源包中下载本活动的代码文件并仔细阅读。文件中的代码实现了图 3.35所对应的一个CNN架构。

2 在这个架构基础上，同学们可以通过更改代码中的参数（数字）、激活函数类型和池化窗口大小等，搭建起自己的 CNN架构。

LeNet-5 于 1994 年提出，是最早的深层卷积神经网络之一。由 LeNet-5 首次提出的基本架构及诸多特性现在依然在 CNN 中广泛使用。早期的 LeNet-5 使用 S 形函数作为激活函数，而现代的 CNN 则多采用 ReLU 函数。

图 3.36 显示了一个用于 MNIST 手写数字识别的 LeNet-5 架构（图中符号@前面的数字表示通道数）。

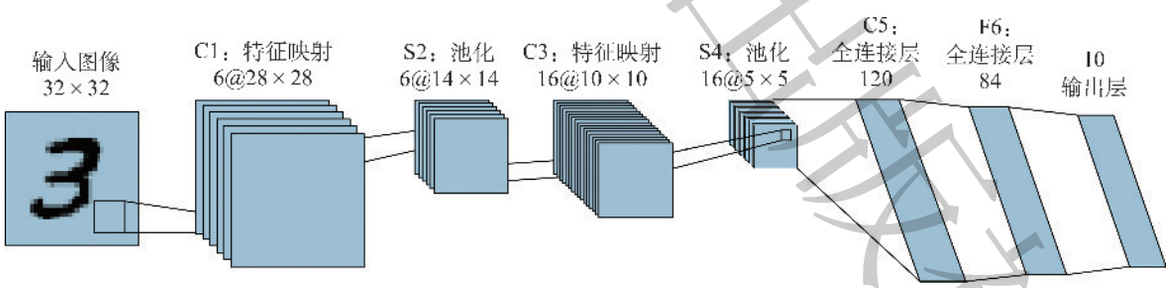


图 3.36　LeNet5的架构

这个网络的输入层接受 32×32 像素的灰度图像作为输入数据，接着有两组“卷积—激活—池化”的组合层 C1－S2 和 C3－S4，其中 C1 和 C3 是卷积层，这两个卷积层及其后面的激活函数层（图中没有标示出来）一起被标记为特征映射；两个组合层后面是两个全连接层 C5

(有 120 个神经元)和 F6(有 84 个神经元),这两个全连接层的结构和功能有所不同,所以使用了不同的名称;网络的最后是有 10 个神经元的输出层。

第一个卷积层 C1 包含 6 个卷积核,对应于 6 个通道,每个卷积核的尺寸为 5×5,每个通道卷积各加上一个偏置,因此第一个卷积层 C1 共有(5×5+1)×6=156 个待学习参数;C1 的后面跟着一个 2×2 平均池化层 S2,再往后则是一个 S 形激活函数。

第二个卷积层 C3 的卷积核尺寸同样为 5×5,共有 16 个卷积核。由于前一个层只有 6 个输出,因此与这 16 个卷积核并非是全连接的,这增加了 CNN 模型的多样性。紧随其后的第二个池化层 S4 也是 2×2 的平均池化。

接下来的 C5 层其实也是卷积层,它有 120 个卷积核,卷积核大小也是 5×5,与上一层的输出数据恰好匹配。与前两个卷积层不同的是,这一层后面没有池化操作,而且实际上是全连接的,所以也算作一个全连接层。其后的 F6 层是一个包含 84 个标准神经元的全连接隐藏层,激活函数为 S 形函数。

LeNet-5 的最后一层是由 10 个单元组成的输出层,用于输出最后的分类结果。

图 3.37 显示了每个层的一些细节。

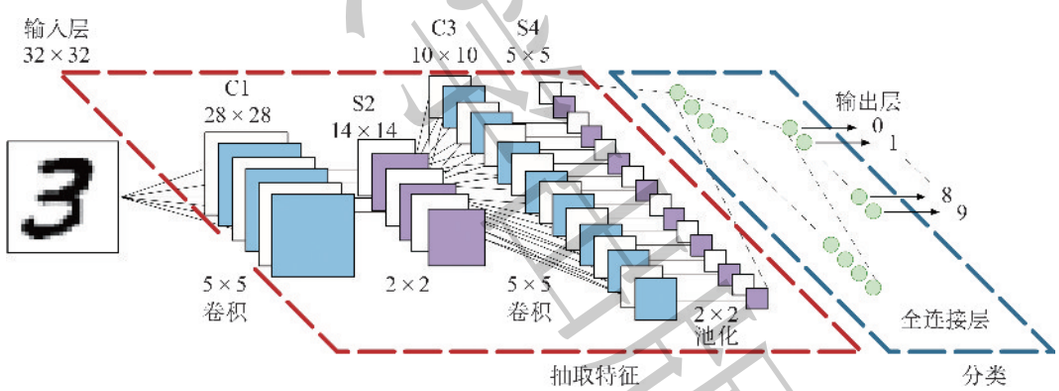


图 3.37 LeNet5网络的内部构成

知 识 延 伸 AlexNet中的新思想

2012年提出的 AlexNet是一个具有历史意义的 CNN网络,事实上,人工智能的新一次浪潮正是由 AlexNet所引起的。在 AlexNet之前,深度学习已经沉寂了很久。2012年,基于 AlexNet架构的 CNN在当年的 ImageNet图像分类竞赛中夺魁,其最好的五个分类成绩的差错率比上一年的冠军成绩下降了十多

个百分点，这是该领域的一个巨大突破。

从网络基本结构上看，AlexNet与 LeNet5并没有太大的不同，其内部同样堆叠了若干“卷积—激活—池化”的组合，加上两个全连接层，如图 3.38所示。

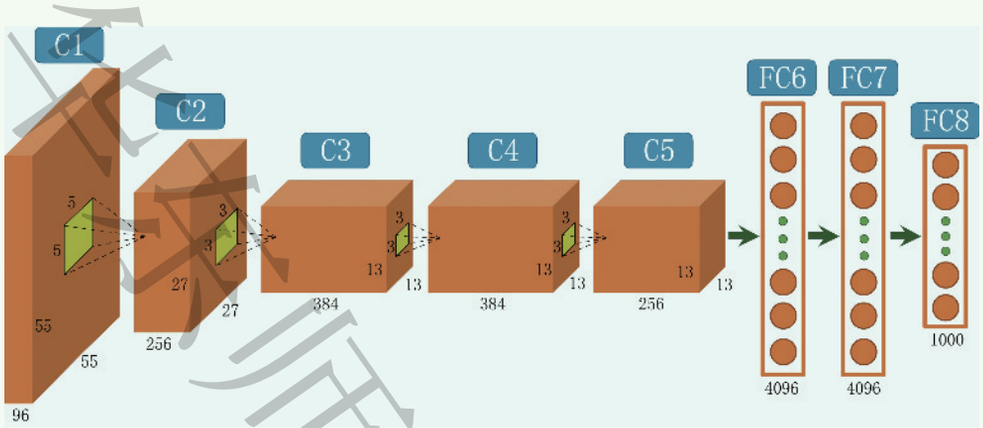


图 3.38 AlexNet的架构

虽然在整体架构上，AlexNet与 LeNet5差别不大，但在内部结构上，AlexNet与 LeNet5还是有很多不同之处，两者的差异主要表现在：

1 AlexNet使用 ReLU作为激活函数，而 LeNet5使用的是 S形函数。使用 ReLU激活函数解决了S形函数在较深的网络中梯度逐渐消失的问题。

2 LeNet5的池化层使用了等步幅的平均池化，而 AlexNet则全部使用最大池化，而且其步幅比池化窗口小，既避免了平均池化带来的模糊效果，同时使池化层的输出之间带有一定的重叠，提升了特征的丰富性。

3 AlexNet使用了局部响应正规化层（local response normalization，缩写为 LRN）。LRN的作用有点像激活函数，直观上为局部神经元活动创建了一种竞争机制，使得其中响应比较大的值变得相对更大，并抑制其他反馈较小的神经元响应。LRN可以增强训练的准确性，但在最新的 CNN结构中，LRN已经被其他构造逐渐代替。

4 AlexNet在最后的全连接层使用了 Dropout（随机失活）手法，随机地忽略一部分神经元的输出信号，减少了连接数量。

虽然在网络结构上，AlexNet 与 LeNet-5 相比差异不算太大，但毕竟比 LeNet-5 晚了十几年，而这十几年恰好是大数据和超级计算能力大发展的年代。随着技术的快速发展，如今 CNN 的深度（构成神经网络的层数）已经从最初 LeNet-5 的 5 层（即 LeNet-5 中的“5”）发展到几十层甚至数百层。

图 3.39 显示了 CNN 的深度近年来发展的情况。

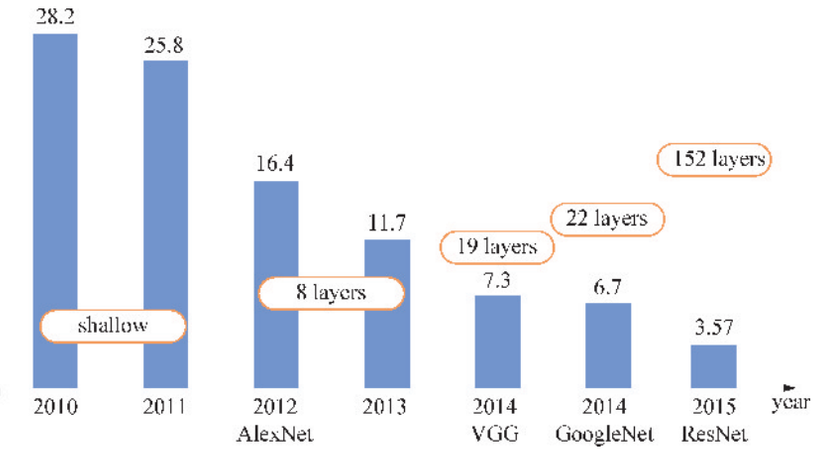


图 3.39　典型 CNN 的深度及图像识别差错率

五、卷积神经网络的应用案例

我们将以项目任务的形式，分别用不同的框架来实现两个实用 CNN 的著名应用案例，即 MNIST 的手写数字识别和 ImageNet 的 CIFAR－10 图像分类应用。

1. 手写数字识别

项　目　实　践　　**MNIST手写数字识别的 TensorFlow 实现**

同学们可以遵循下列步骤，使用 TensorFlow 框架完成一个基于 CNN 的手写数字识别的完整应用。

1　下载课程资源包中的程序，调试并运行，记录每次运行得到的识别准确率。

2　仔细阅读代码及其中的注释。

3　将使用 CNN 的识别结果与使用 *k*-近邻分类算法、双层感知机和多层感知机的识别结果进行比对。

4　列举出应用中的所有可调参数，包括需训练的参数和不需训练的参数；分别调谐各个参数，尝试重新初始化，再运行程序并详细记录运行结果。

5　尝试改变训练集、诊断集和测试集所占的比例，观察并分析对实验结果的影响。

项　目　实　践　　**使用 Keras实现手写数字识别**

Keras建立在 TensorFlow之上，提供了更友好、更方便搭建 CNN模型的功能。请从课程资源包中下载本活动的代码文件，文件中的代码是使用 Keras实现手写数字识别应用，请同学们将其与前面使用 TensorFlow实现的代码进行比较研究。

项 目 实 践

同学们可以自学 Python，用以实现 MNIST手写数字识别，并与 TensorFlow和 Keras实现作比较，完成表 3.3。

表 3.3

正确率	训练时间	开发难度	参数设置

2. CIFAR 图像识别

CIFAR－10/CIFAR－100 数据集包含了 8 000 万张微小尺寸图像，每张小图像都是一张 32×32 像素的彩色图片。CIFAR－10 数据集中的图像有 10 个类，每个类有 6 000 张小图像，总共有 60 000 张。其中 50 000 张用于训练（包含类属的标签），10 000 张用于测试（不含类属标签）。

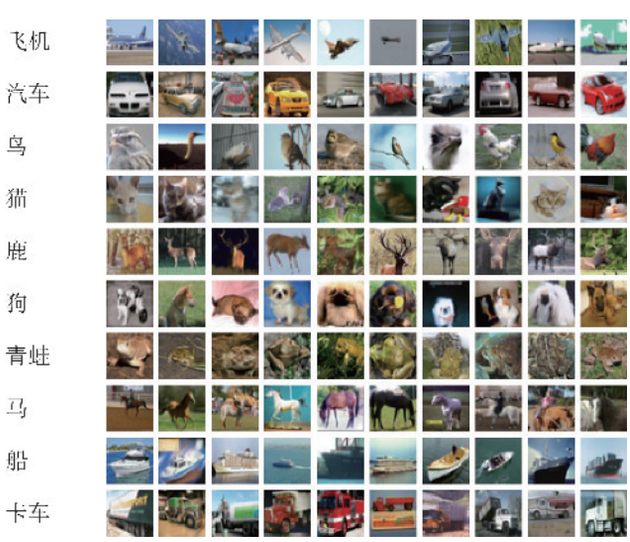


图 3.40 CIFAR－10中的图像样例

图 3.40 显示了从 CIFAR－10 数据集的 10 个类别中分别随机选出的 10 张照片。

CIFAR－10 数据集已经包含在 Keras 中了，可以直接读取，不用重新下载。

CIFAR－100 数据集与 CIFAR－10 的主要差别在于其类属的划分和图像标签的标记方法：CIFAR－100 中的小图像分属于 100 个小类，每个小类含有 600 张图像，500 张用于训练，100 张用于测试。这 100 个小类又分属 20 个大类（CIFAR－100 中被称为超类），每个超类包含 5～6 个小类。此外，每张小图像都带有一个“精细”标签，标记它所属的小类，以及带有一个“粗糙”标签，标记它所属的超类。

表 3.4 列出了 CIFAR－100 数据集中类和超类的划分。

表 3.4　CIFAR-100中的类和超类

Superclass(超类)	Classes(小类)
aquatic	mammals beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food	containers bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household	furniture bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

知　识　延　伸　Keras中的数据集

除 CIFAR-10和 CIFAR-100外，Keras中还包含很多机器学习领域中常用的数据集，比如 MNIST的手写数字数据集、IMDB电影评论数据集、波士顿房屋价格数据集和时装评论数据集等。

在 Keras中导入这些数据集也非常简单，只需几条语句就可以完成。例如，可以使用下面两条语句导入 CIFAR-10数据集，并将其划分为训练集和测试集。

```
from keras.datasets import cifar10
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
```

下面的语句则对 MNIST数据集做了同样的事情。

```
from keras.datasets import mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

同学们可以仿照下面活动的方法，对更多的数据集进行尝试。

项 目 实 践　　使用 Keras完成 CIFAR-10图像的分类任务

1 加载 CIFAR-10数据集。

2 从每一类中随机挑选一张图像显示，并显示其所属的类别。

运行课程资源包中的对应代码后，将显示 10张带有标签的小图像，如图 3.41所示。

图 3.41 带类属标签的 CIFAR-10图像

3 数据预处理(将类属标签转换为二元标签)。

4 定义卷积神经网络结构。

5 组装、训练 CNN模型，显示模型的识别精度。

请从课程资源包中下载该活动的代码文件并仔细阅读。

3. 图像风格迁移

所谓图像风格迁移，是指将图像 A的风格与图像 B的内容相结合，使形成的新图像在视觉上具有图像 A的风格。图像风格迁移是数字图像技术特有的手段之一，通常认为必须具备一定的艺术造诣才能做到，图像风格随意地复合可能会导致画面整体失调、杂乱，也没有任何美感。

在卷积神经网络方法出现之前，人们进行图像风格迁移的基本思路是：分析某一种风格的图像，为这种风格的图像建立一种模型(数学或统计的)，再改变被迁移图像使之尽量符合所建立的模型。这个思路与艺术家采用不同绘画风格来表现思想有类似之处。但这种图像风格迁移的思路有很大的局限性，即先需要对图像风格进行建模，而且一种模型或程序只能实现一种图像风格的迁移。使用卷积神经网络后，一个程序可以实现很多种的图像风格迁移。

深度卷积神经网络可以经过训练自动学习图像风格，并将选择的图像风格作用于待迁移的图像，从而一次性实现多种图像风格的自动

学习和迁移。使用这种方法，无需对图像进行建模，甚至即便对图像风格一无所知，也能生成具有典型风格特征和强烈视觉冲击力的图像。

在实现风格迁移的过程中，每次需要使用三幅图像：原始内容图像 C（content）、原始风格图像 S（style）和生成后的图像 G（generated）。基本上，我们的 CNN 模型需要完成两项任务：(1)从原始风格图像 S 中提取出图像的内容和风格，我们已经知道 CNN 具有很好的图像特征提取能力，因此这恰好是其发挥作用的地方；(2)度量图像在内容和风格上的相似性，这种相似性指的是图像在特征上的相似，而不是图像的像素相似。

用于同时提取图像特征和图像内容的 VGG19 是 2014 年提出的一种深度 CNN 架构，其构成如图 3.42 所示。

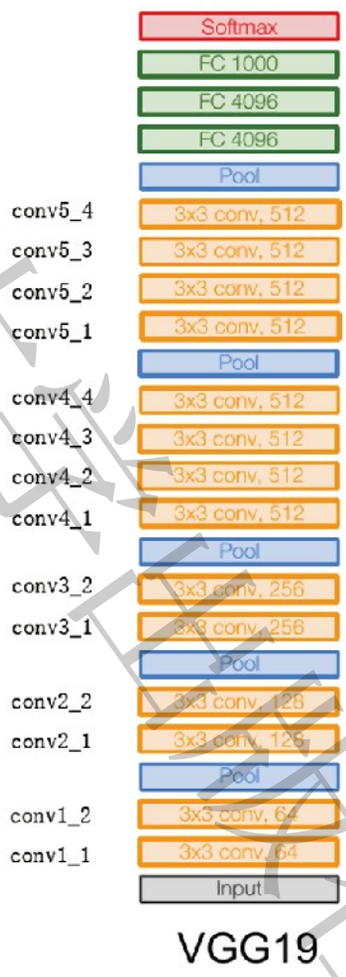


图 3.42　VGG19的架构

VGG19 包含 19 个卷积—池化层（图 3.42 中的橘黄色部分为卷积层，蓝色部分为池化层）、3 个全连接层（图 3.42 中的绿色部分）。

图像数据从 VGG19 的输入层进入，通过卷积层 conv1_1、conv2_1、conv3_1、conv4_1、conv5_1 后的中间结果作为提取出的该图像“风格”特征；而通过卷积层 conv4_2 的中间结果提取出的为图像“内容”特征。

此时，模型学习的目标是图像而不是参数，即学习的目标是使 G 在内容上尽量与 C 接近，而在风格上则尽量与 S 接近。因此，损失函数应该由两个部分构成：(1)内容损失部分，用于计算 C 和 G 的差异；(2)风格损失部分，用于计算 S 和 G 的差异。

于是模型的训练流程就变得非常简单：随机生成一幅 G，使用梯度下降法对模型(G 的值)进行迭代优化，使损失函数值达到最小化；训练完成后，G 在内容上最佳接近 C，在风格上最佳接近 S。

为了调节生成后的图像中风格的强度，可将 C、S、G 分别输入 VGG19，计算内容损失和风格损失的加权和，通过调谐内容损失和风格损失上的两个权重来控制风格的强度。

项 目 实 践　使用 TensorFlow 及 CNN 模型 VGG19，实现图像风格迁移应用

1 下载课程资源包中的程序包。

2 选择几张原始内容图像(C)。例如，选择两幅上海市地标性建筑的照片(如图 3.43所示)作为原始内容图像。

图 3.43 两幅上海市地标性建筑的照片

3 选择几张原始风格图像(S)。例如，选择三幅具有不同风格的名画的照片(如图 3.44所示)作为原始风格图像。

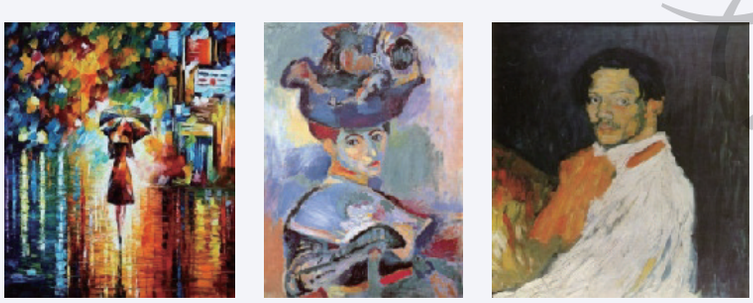


图 3.44 三幅具有不同风格的名画的照片

4 用不同的原始内容图像(C)和不同的原始风格图像(S),运行图像风格迁移程序得到不同的生成图像(G),如图 3.45所示(第一行为原始内容图像,第一列为原始风格图像):

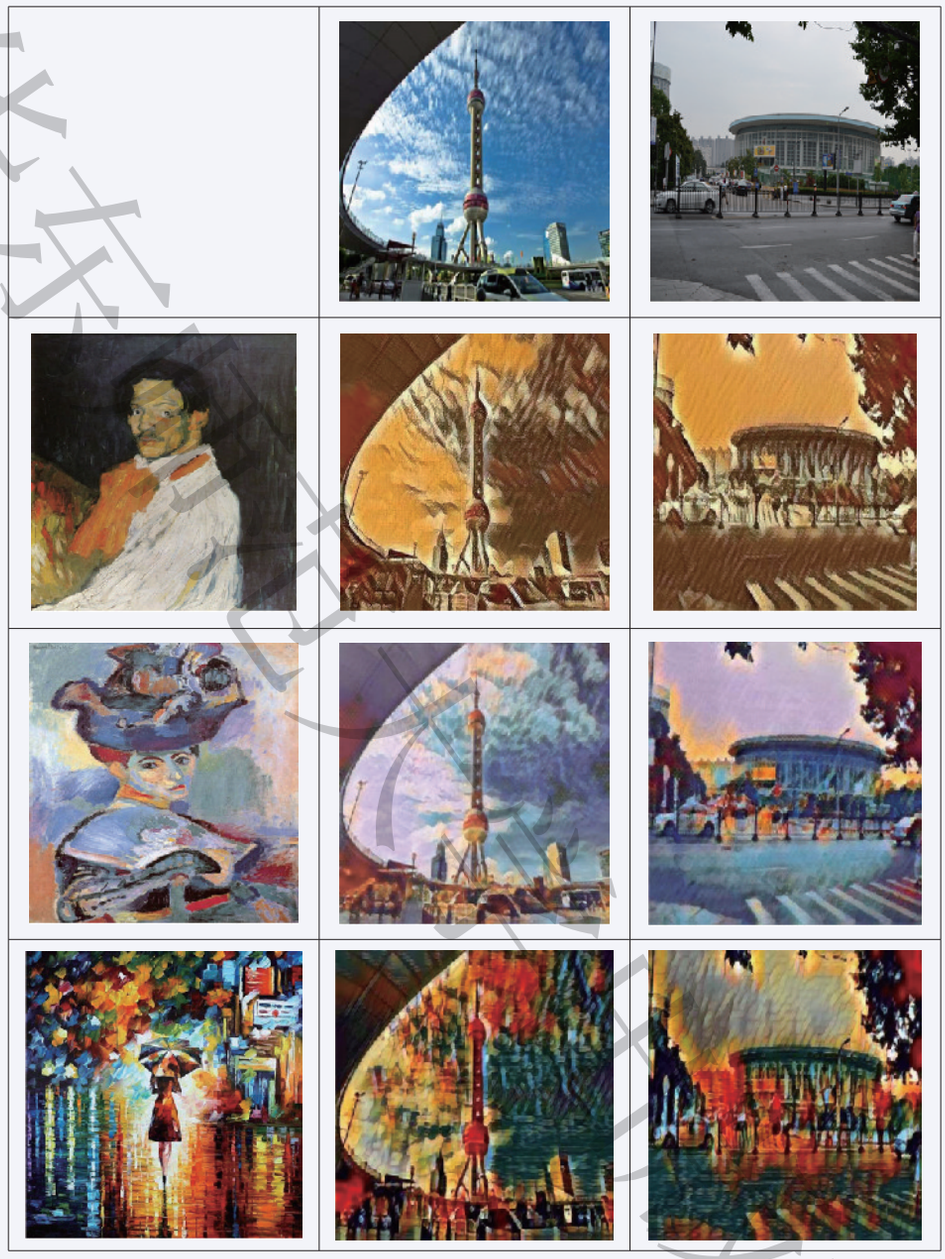


图 3.45 三种不同风格的图像迁移

5 用美图秀秀或 Photoshop,对所选的几张原始内容图像进行增加滤镜的图像处理,分析图像风格迁移和图像滤镜效果的区别。

第四章

人工智能未来发展

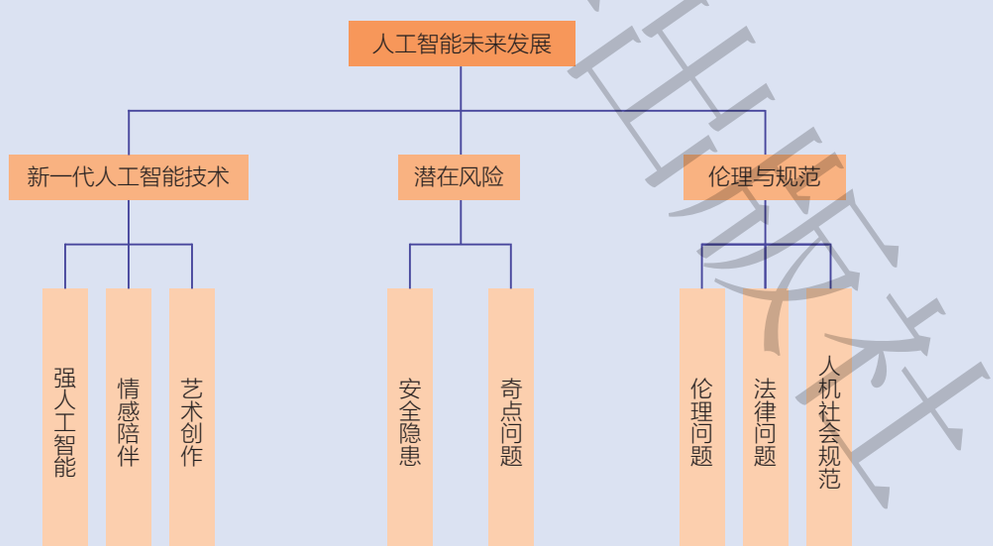
本章学习目标

- 了解新一代人工智能技术跨界融合、人机协同的发展趋势，感受人工智能技术对于人们生活方式和思维模式的改变。
- 辩证认识社会智能化的巨大价值和潜在风险，增强自主学习和探究学习的能力，提升智能社会的公民责任意识。
- 理解人工智能技术发展带来的伦理问题，自觉遵守智能化社会的法规，增强信息系统安全意识，提高综合应用信息技术的能力。

随着人工智能的快速发展，许多科幻电影和文学作品中描述的人工智能场景正在变成现实。自动驾驶汽车悄然上路，机器人陪伴快速普及，就连人类最为自豪的专利，艺术创作领域，也出现了人工智能的身影。日新月异的技术向我们展开了一个充满无限可能的智能时代。

与此同时，人工智能的快速发展也引发了一些不同层面的思考，甚至是争议。例如：人工智能有可能超越人类吗？人工智能技术会否为人类社会带来威胁？如何看待人与机器人之间的情感？在一个数字化、网络化、智能化的社会里，还能保证个人隐私吗？这些问题都是智能时代公民所要面对和思考的。

本章知识结构



项目主题

畅想未来智能生活

项·目·情·境

通过对人工智能发展历程的回溯，参观人工智能创新园区的体验，和新一代人工智能相关技术的学习，同学们初步了解了人工智能的本质，引发了他们对于智能化社会伦理道德和法律问题的强烈关注。大家畅想人工智能为社会生活带来的巨大变化，思考跨入未来智能生活，如何具备与人工智能同样高远、同样广阔的视野，来捍卫人类未来的智能生活，实现人与人工智能和谐相处！

项·目·任·务

任务 1	任务 2	任务 3
以实际生活中人工智能的应用为例，解释新一代人工智能技术的特点，畅想未来人工智能发展趋势。	运用调查访谈、观看相关影视和文学作品、网络采集主流观点等方式，收集人工智能技术应用可能引发的社会问题，并讨论应对策略。	利用整理好的资料，制作“未来的人工智能”电子作品，撰写“未来人工智能应用及社会问题”研究报告。

第一节　新一代人工智能技术

新一代人工智能技术在大数据和高计算能力支撑下，向更加类人的方向高速发展，深入地应用于社会生活的方方面面，体现出跨界融合和人机协同的新特征。虽然其中很多技术还处在发展阶段，但其发展前景和广泛应用都很令人期待。

一、智能驾驶

现有的人工智能系统虽然为人类生活带来了很多的便利，但是其局限性也很明显。一个明显的弱点是这些系统只能在一种或几种应用中使用，不具有通用性，而且对于数据的依赖性很大，往往需要大量的训练数据，欠缺人类举一反三的能力。我们把这类人工智能系统称为弱人工智能系统。相对地，我们把更通用，对数据依赖性更小，具有更好学习能力甚至自主创造能力的系统称为强人工智能系统。

探　究　活　动

2018年 6月，北京市海淀区马路上多了一种引人注目的无人配送快递车，如图 4.1所示。这种自动驾驶汽车可以对周围环境进行 360°观测，识别交通灯、行人、车辆，自动规划路径，绕开障碍物。

自动驾驶汽车是一种智能驾驶汽车，即也是一种人工智能系统应用。请同学们思考这种自动驾驶汽车是属于弱人工智能系统，还是强人工智能系统，并分组讨论。

图 4.1　无人配送快递车

自动驾驶汽车是一种通过车载计算机系统实现驾驶的智能汽车。相比刷脸支付系统，自动驾驶汽车要综合使用更多的人工智能技术，对学习能力和实时计算的要求更高，是人工智能系统迈向强人工智能的重要里程碑。图 4. 2 提供了一个自动驾驶系统的最基本框架，当然，我们也可以加上语言识别等其他功能使整个系统使用起来更方便。

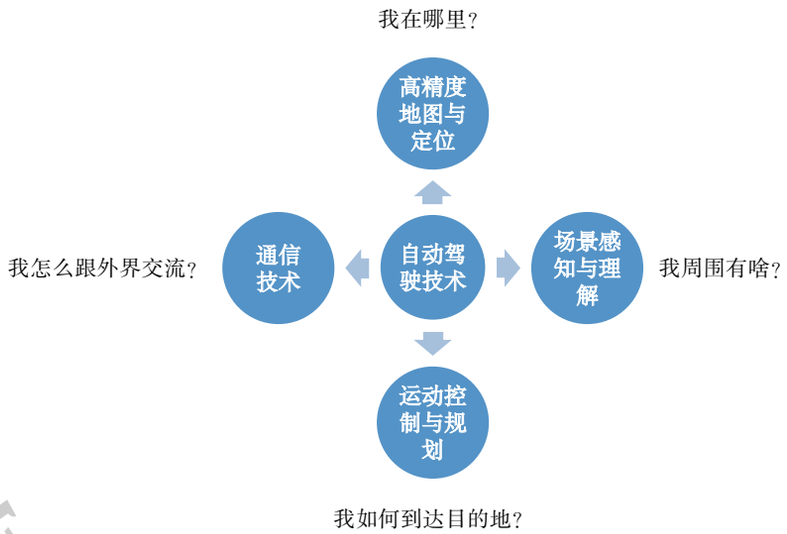


图 4.2　自动驾驶技术关系图

1. 自动驾驶汽车的“大脑”

人工智能芯片是专门用于处理人工智能应用中的大量计算任务的芯片，被称为自动驾驶汽车的“大脑”。人工智能芯片用于实时处理采集到的数据，对数据进行比对，然后规划路径和做出控制选择。与传统的中央处理器芯片(CPU)相比，人工智能芯片在处理人工智能计算时性能更强、功耗更低。

2. 自动驾驶汽车的“眼睛”

司机驾驶汽车时，需要通过眼睛观察周围的环境，对于自动驾驶汽车而言，传感器就相当于“眼睛”。通过传感器，自动驾驶汽车能够识别道路、行人、车辆以及各种交通标志和基础交通设施。目前，自动驾驶汽车上使用的传感器主要有摄像头、毫米波雷达和激光雷达等。以视觉技术为指导的自动驾驶汽车，通过摄像头和毫米波雷达的组合实现相关功能，在成本上比较经济，性价比较高。而以激光雷达技术为主导的自动驾驶汽车，在识别精度上更胜一筹，但成本相对较高。

3. 自动驾驶汽车的导航

自动驾驶汽车需要导航系统引导车辆行驶，这就需要高精度地图

和定位系统(如北斗系统、GPS)来准确定位地形、周边物体和道路轮廓。高精度地图是精度和数据维度更高的电子地图,它的精度可以达到厘米级别,而一般的导航地图精度在米级别。在数据维度方面,一般电子地图只记录道路级别的数据,如道路形状、坡度、曲率、铺设和方向等,高精度地图不仅增加了车道线类型、车道宽度等车道属性数据,还有其他目标数据,如防护栏、树木、道路边缘类型、路边地标和高架物体等。

4. 自动驾驶汽车的通信

为了保证自动驾驶的安全,能够及时应对突发状况并选择最优的行驶路径,需要使用高精度地图。高精度地图对数据的实时性要求更高,有些动态数据需要即时更新。因此,自动驾驶对通信有较高的要求。

自动驾驶汽车在行驶过程中需要跟其他车辆、路侧基础设施和行人进行无线通信,这种技术称为车联网技术(简称 V2X,意为 vehicle to everything,即车与外界的信息交换)。车联网技术使得车与车、车与基站、基站与基站之间能够通信,自动驾驶汽车可以获得实时路况、道路信息、行人信息等一系列交通信息,从而提高驾驶安全性,减少拥堵,提高交通效率,提供车载娱乐信息等。

5. 自动驾驶汽车的外观

在外部结构设计方面,自动驾驶汽车也会与传统汽车有所不同。虽然目前很多自动驾驶汽车还有方向盘之类的结构设计,但随着自动驾驶技术的成熟和智能化程度的提升,自动驾驶汽车将不再需要驾驶员手动控制转向、油门和刹车,外置的方向盘、油门和刹车也将不再需要。自动驾驶汽车在外观结构设计方面,会朝着更人性化、更舒适和更安全的方向发展。

知 识 延 伸 **自动驾驶技术级别划分**

为了更好地区分不同层级的自动驾驶技术,国际汽车工程师学会于 2014年发布了自动驾驶的六级分类体系,目前绝大多数主流自动驾驶研究者和研究机构已将此标准当作通用的分类原则。自动驾驶技术六个级别的定义和区别可通过表 4.1进行了解。

表 4.1　自动驾驶技术的六个级别

级别	描　　述
L0	完全由驾驶员进行驾驶操作，汽车只负责执行命令并不进行驾驶干预
L1	自动系统有时能够辅助驾驶员完成某些驾驶任务，如车道保持系统和主动刹车系统等
L2	自动系统能够完成某些驾驶任务，但驾驶员需要监控驾驶环境，手脚要放在方向盘和刹车踏板上待命，随时准备接管，如自适应巡航和拨动转向灯即可实现自动变道行驶等
L3	驾驶员将不再需要手脚待命，机器可以独立完成几乎全部的驾驶操作，但驾驶员仍需要保持注意力集中，以应对可能出现的人工智能应对不了的情况
L4	在部分场景中，通常是指在城市道路或是高速公路上，汽车已经可以在完全不需要驾驶员介入的情况下进行所有的驾驶操作，驾驶员可以在车上工作或休息
L5	在任何场景中，汽车已经可以在完全不需要驾驶员介入的情况下进行所有的驾驶操作，驾驶员可以在车上工作或休息

二、智能陪伴

体　验　思　考

请同学们想一下，在生活中，你喜欢和什么样的人交朋友。思考你对朋友的要求可能在未来由机器人来满足吗？

图 4.3　智能陪伴机器人

随着人口的老龄化，作为社会中坚力量的年轻人的比例越来越小，他们的生活压力也越来越大，众多老人和儿童得不到亲人必要的陪伴。在这样的大背景下，人类社会未来对智能陪伴机器人的潜在需求就愈发迫切了。

为了拉近与人类的情感距离，智能陪伴机器人的外形一般设计成类人形。这种机器人不仅能做出与人类相似的动作，用人类的语言与我们交流，更重要的是它还能够通过自身的感觉（传感器）来采集环境中的温度、亮度、声音，以及人体的各种生命体征数据，通过内置的智能程序与算法对这些数据进行分析，并以此为依据为人们提供个性化服务。

智能陪伴机器人通常还具备一定的学习能力：陪伴的时间越久，它就会越懂你，知道你的习性和喜好。

比如，如果你喜欢室外运动，当机器人检测到室外空气质量较差时，它会建议你改在室内锻炼，并且会结合你的身体状况计算出合理的运动量。

探 究 活 动

请同学们分组讨论以下问题：

1 人类的哪些任务交给智能陪伴机器人比较合适，哪些不合适？

2 要完成陪护任务，需要解决哪些与人工智能有关的技术难题？

知 识 延 伸 中国工业机器人产业发展

我们国家借鉴了国际机器人联合会（IFR）的机器人分类标准，在国家标准《机器人与机器人装备 词汇》（GB/T 12643-2013）中将机器人分为工业机器人和服务机器人。其中，工业机器人被誉为“制造业皇冠上的明珠”，对国家经济发展具有重要的意义。近年来随着全球自动化趋势影响的扩大，在国家产业政策指引下，我国工业机器人呈现出较好的发展势头，并逐渐向“智能化”方向发展。从 2016年到 2019年，我国已成为全球工业机器人密度增速最快的国家。但同时应该清醒地认识到，我国的机器人产业基础还比较薄弱，核心技术和世界工业机器人领域领先企业相比仍有差距。随着全球机器人整体市场规模持续增长，我国的机器人市场需求潜力巨大，工业领域努力突破机器人关键核心技术，服务领域智能水平快速提升，与国际领先水平基本并跑，颇具成长空间。

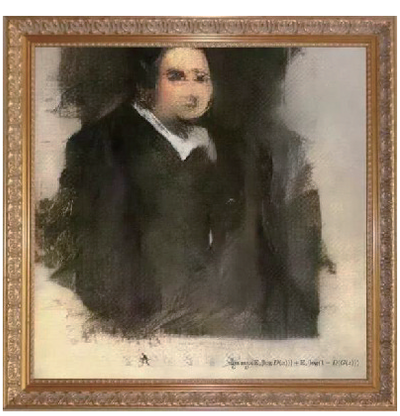


图 4.4 人工智能创作的油画

三、智能艺术创作

艺术创作一直是人类最引以自豪的智能体现。2018 年，一幅用神经网络创作的油画（如图 4.4 所示），在公开拍卖中引起了公众的广泛关注。现在，人工智能已经广泛使用于海报创作、图像风格迁移等不同应用中。除了视觉领域，人工智能也被用于音乐创作，例如作曲、伴奏、歌唱等，大幅降低了音乐制作的成本。

探 究 活 动

通过互联网，查找最新的人工智能艺术作品，分组讨论这些作品是否可以被看作艺术创作，并阐述你的理由。

图 4.5　音乐情感的标签表示方式

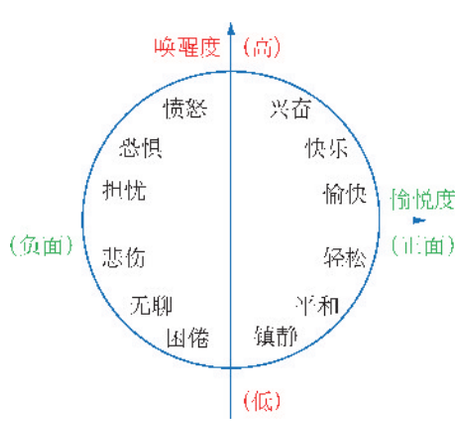


图 4.6　音乐情感的维度空间表示方式

除了创作艺术，人工智能技术也尝试像人一样欣赏艺术，从而达到与人类更好地共鸣。例如音乐情感分析，可以通过人工智能技术预测人类对于音乐的情感反应，在音乐推荐、音乐治疗方面都有重要的应用价值。

音乐是一种音频的艺术形式，好的音乐可以唤起人们强烈的情感反应。千百年来，这一特点令无数研究者为之着迷。尽管迄今为止，现代科学依然不能完整透彻地解释这一现象，但这并不影响各行各业把这一特点应用到实际生活中。例如，音乐疗法在西方心理治疗中已经使用了几十年，用于改善病人的情绪和认知问题。中国古代也有“琴医心，花医肝，香医脾，石医肾，泉医肺，剑医胆”的经典名句。

音乐的情感表示通常有两种方式，一种是标签方式，另一种是维度空间方式。音乐情感的标签方式起源于面部情感分类。如图 4.5 所示，人的面部表情可以大致被分为六个典型的类型。相应地，音乐也可以被归类为快乐的音乐、恐惧的音乐等不同类型。所以，这一类音乐的情感计算可以通过建模变为分类问题：以音乐信号作为特征，例如随时间变化的音高；以音乐的特点作为特征，例如节奏。以情感类型作为分类标签，训练分类模型，可以对输入的音乐进行自动的情感分类。

维度空间方式以二维空间中的不同部位来表示音乐情感。如图 4.6 所示，横轴表示情感的愉悦度，纵轴表示唤醒度，愉悦度和唤醒度有正负之分，分别对应着正负半轴。例如，快乐是一种正面情感，而悲伤是一种负面情感；愤怒是一种高唤醒度情感，而无聊的唤醒度则比较低。这一类音乐情感计算所用的特征和分类模型是一样的，都是音乐信号或者音乐的特点，但是情感表示不是离散的类别，而是一个二维连续空间上的点。由于构成这个点的两个维度，即愉悦度和唤醒度都是连续的，所以常常建模为回归问题。

音乐本身也有擅长表达和不擅长表达的情感，比如嫉妒就是音乐不擅长表达的情感，而像浪漫这样的情感则是音乐非常擅长表达的。一般来讲，音乐中经常出现的情感可大致分为如下类型：快乐、悲伤、平静、唤醒、愉快、喜庆、柔和、有力、浪漫、无忧无虑、扣人心弦、热情、积

极、感人、活泼、愤怒、侵略、悠闲、醇厚、怪诞、奇异等。

项 目 实 践　中国音乐情感计算

中国音乐有源远流长的历史和独特鲜明的艺术魅力。同学们可以综合运用本书中所学的知识，尝试中国音乐的情感分析。

1 下载 100段 20~30秒的中国传统音乐片段，建立一个小数据集，反复聆听这些音乐，并参照图 4.5和图 4.6所示进行手工标注。

2 讨论图 4.5和图 4.6所示表示方式在中国音乐情感表示上的局限性。

3 尝试用从本书中所学的分类模型对收集到的音乐片段进行情感计算。

探 究 活 动

请你与同学从以下角度一起讨论交流人工智能和艺术结合的意义。人工智能能作曲、写诗、画画等，那么：

1 人工智能是具有创造力，还只是在模仿？

2 人工智能是不是真的理解艺术的内涵？

3 从人工智能艺术作品中是否能看出机器的痕迹？

另外，那幅在拍卖中引起公众广泛关注的油画，其收益应该属于使用开源代码生成艺术作品的人？还是应该属于开源代码的贡献者？

知 识 延 伸　能自动生成油画的生成式对抗网络技术

生成式对抗网络技术包含两部分：生成器和判别器。生成器的目标是利用神经网络生成一个虚假对象，希望虚假对象越接近真实对象越好。判别器的目标是利用神经网络判别真实对象和虚假对象，判别准确率越高越好。判别器需要生成器生成的虚假对象和真实对象一起来训练神经网络，而生成器需要判别器训练的梯度来优化生成器的神经网络。生成器和判别器的目标是对抗的，过程又是相互合作的。生成器属于机器学习的无监督学习，判别器属于机器学习的监督学习。（参见图 4.7）

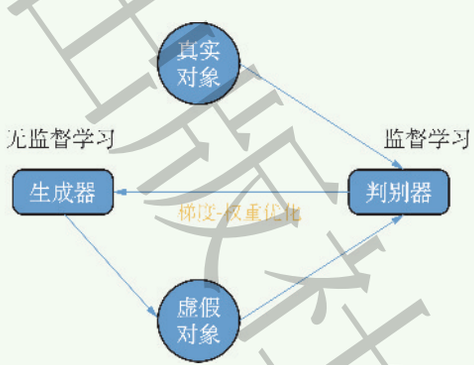


图 4.7　生成式对抗网络

第二节　潜在的风险

人工智能技术的发展和应用加速了人类经济社会的发展，同时也在重塑人与机器的关系，带来了一系列可以预测和无法预测的风险。其引发的伦理问题、法律问题和其他社会问题，如果不能妥善处理，可能会引发社会不安，甚至担忧，有违人类发展科技的初衷。所以，我们应该客观辩证地看待人工智能技术对人类社会和思维方式的改变。

探　究　活　动

在与人工智能有关的许多科幻电影中，经常会出现机器人伤害人类或毁灭人类的场景。请同学们分组讨论这些艺术创作在现实生活中发生的可能性。

一、安全隐患

1. 技术或管理缺陷导致的安全问题

作为一项发展中的新兴技术，人工智能系统当前还不够成熟。某些技术缺陷导致工作异常会使人工智能系统出现安全隐患，比如深度学习采用的黑箱模式会使模型可解释性不强，机器人、无人智能系统的设计、生产不当会导致运行异常等。另外，如果安全防护技术或措施不完善，无人驾驶汽车、机器人和其他人工智能装置可能会受到非法入侵和控制，这些人工智能系统就有可能失控，做出对人类有害的行为。

2. 数据采集中的隐私侵犯

随着各类数据采集设施的广泛使用，智能系统不仅能通过指纹、心跳等生理特征来辨别身份，还能根据不同人的行为喜好自动调节灯光和室内温度、播放音乐，甚至能通过睡眠时间、锻炼情况、饮食习惯以及体征变化等来判断身体是否健康。然而，这些智能技术的使用就意味着智能系统掌握了大量的个人信息，甚至比用户更了解用户。这些数据如果使用得当，可以提升人类的生活质量，但如果出于其他目的非法使用，就会造成隐私侵犯。

3. 技术滥用引发的安全威胁

人工智能对人类的作用很大程度上取决于人们如何使用与管理人工智能技术，比如：如果人工智能技术被犯罪分子滥用，就会带来安全问题。黑客可能通过智能方法发起网络攻击，智能化的网络攻击软件能自我学习，模仿系统中用户的行为，并不断改变方法，以期尽可能长时间地停留在计算机系统中；黑客还可利用人工智能技术非法窃取私人信息；通过定制不同用户阅读的网络内容，人工智能技术甚至会被用来左右和控制公众的认知和判断。

探 究 活 动

调查你身边的同学或亲戚，了解他们都用了什么智能穿戴设备，这些设备都采集了哪些生理信息，这些信息存储在哪里，有没有泄露个人隐私的风险。把你的调查结果分享给本组同学。

二、技术奇点

“奇点”一词的出现由来已久，在经济和科技领域，主要用来描述会产生剧变或引起爆发的临界点。在史蒂文·希尔所著的《经济奇点：共享经济、创造性破坏与未来社会》一书中提及的共享经济本质上是技术更加发达的一种衍生品，而这种技术进步的一个重要表现就是劳动进一步机械化，劳动机械化程度的提升会导致人类社会分化加剧——少部分掌握技术和生产的人在社会资源分配及经济生产中占据绝对优势，往复循环，最终在过了一个临界点后会从内部摧毁整个经济。这个引爆点就被定义为经济奇点。

对于人工智能，很多人有着同样的忧虑。随着人工智能技术的高速发展，它正逐渐渗透到我们工作和生活的方方面面。如果未来的某一天，人工智能的爆发式进展导致机器智能达到人类的水平，甚至超越创造机器智能的人类，智能机器会否按照人类设定的方式不断学习和自我优化呢？正如科幻电影中描述的那样，奇点过后我们将进入一个人机共处的社会，而机器会否为了自身利益的最大化，选择伤害人类呢？当机器有了与人相似甚至更高的智能以后，人类会否有足够的反制能力？

这种想法并非杞人忧天。随着人工智能的迅速发展，在益智类综艺节目、完全信息博弈、非完全信息博弈和模拟人类现实博弈中，人工智能开始逐渐占据优势。

2011 年，在益智类综艺节目《危险边缘》中，美国超级计算机“沃森”挑战该节目的两名冠军肯·詹宁斯和布拉德·鲁特尔，并战胜了这两位，如图 4.8 所示。

图 4.8 “沃森”超级计算机登上智力问答节目《危险边缘》

“沃森”是 2007 年研发出来的一台能迅速回答涉及双关语和文字游戏等复杂问题的机器，虽然比赛时不能接入互联网进行搜索，但“沃森”存储了 2 亿页的数据，包括各种百科全书、词典、新闻，甚至维基百科的全部内容。“沃森”可以在 3 秒内检索数百万条信息并以人类语言输出答案，还能分析题目线索中的微妙含义、讽刺口吻及谜语等。“沃森”还能根据比赛奖金的数额、自己比对手落后或领先的情况、自己擅长的题目领域来选择是否要抢答某一个问题。

在完全信息博弈领域，2017 年 5 月，阿尔法围棋（AlphaGo）人工智能程序与围棋世界冠军、职业九段棋手柯洁进行了围棋人机大战，以 3 ∶ 0 的总比分获胜，如图 4.9 所示。

图 4.9 阿尔法围棋与人类顶尖棋手对弈

2017 年 10 月，新款围棋人工智能程序阿尔法元围棋（AlphaGo Zero）诞生。研究人员只给阿尔法元围棋一个棋盘，让它从空白状态起，在不输入任何人类指导和经验的条件下，凭借强化学习，自我对弈。仅仅用了 3 个小时，阿尔法元围棋就达到了人类初学者的围棋水平。3 天自我对弈后，阿尔法元围棋就以 100 ∶ 0 击败了阿尔法围棋。随着程序训练的进行，它不仅独立发现了人类总结出来的围棋规则，还为这个古老的游戏带来了新见解。

完全信息博弈是指每一个参与者都拥有所有其他参与者的特征、策略及得益函数等方面的准确信息的博弈，围棋就是典型的完全信息博弈。阿尔法围棋的主要工作原理是深度学习，阿尔法元围棋拥有更强的自学能力，可以不断进行自我迭代升级。

不完全信息博弈是指参与者并不完全清楚有关博弈的一些信息。扑克游戏中，选手并不知道其他选手手中的牌，在作决策时，必须对其他选手手中的牌做一个估计，而没有确切的信息，因此扑克游戏属于典型的不完全信息博弈。扑克被喻为人类防守智能机器在游戏领域节节胜利的“最后一道防线”。2017 年，卡内基梅隆大学人工智能系统 Libratus 在长达 20 天的鏖战中，调用了 12 万场的比赛数据，打败 4 名世界顶级德州扑克玩家，赢得了比赛，这是一个非常惊人的结果。Libratus 之所以能在短时间内在与人类的德州扑克比赛中取得傲人成绩，是由于其基于人工智能技术掌握了“规则”原理。德州扑克是“不完美信息”博弈，无法像下围棋一样计算下一步所有的可能性来决策，于是研究人员便改进人工智能的算法，让其应用均衡博弈，通过平衡风险与收益来决定下一步，以达到纳什均衡定义中的完美状态。

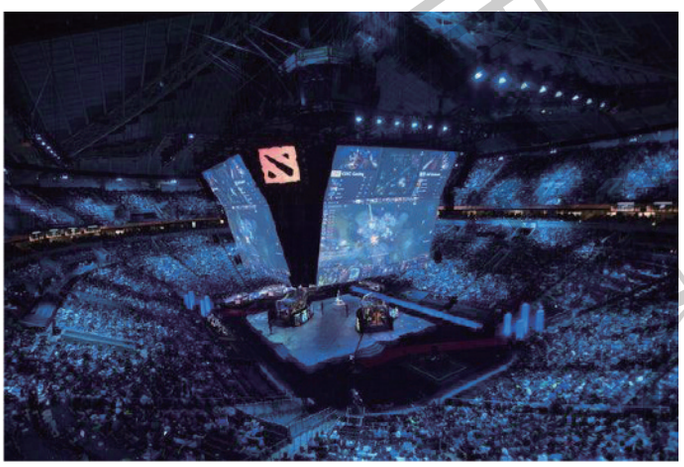


图 4.10　人工智能系统参加电子竞技国际邀请赛

相比围棋这种只有落子一个动作的高度抽象的游戏，电子竞技具有更加现实的意义。人工智能在电子竞技对抗中取胜，则代表了人工智能向通用化又迈进了一大步。2017 年 8 月，研究通用人工智能解决方案的公司 OpenAI 所训练的一款人工智能算法，在著名的国际邀请赛中，参与了 1 对 1 比赛环节，并以压倒性的优势击败了人类顶级电子竞技选手，如图 4.10 所示。

探　究　活　动

通过互联网，了解最新的人机比赛结果，讨论其标志的人工智能发展水平。

第三节　伦理规范

一、伦理问题

当机器人的机械和逻辑处理能力，尤其是自我学习能力和智能化程度获得大幅度提升后，它们可能不再简单地作为人类的工具，而将追求更高的机器人身份与地位。这时，人类就需要思考机器人的身份问题，它究竟是机器人工具，还是与人类身份平等的同类，以及二者之间的关系是什么。

机器人发展到高级阶段，可以与人进行沟通与交流，就可能会产生情感上的需求，进而形成情感伦理问题，这里所说的情感伦理包括爱情、亲情等多种社会关系。科幻电影中常常能看到人类与机器人的爱情故事，但机器人理解的爱情跟人类所理解的爱情可能根本不是一回事。此外，为满足无子女家庭对孩子的渴求，未来也许会出现机器人子女，如电影《人工智能》中的小马丁就是典型的陪伴型子女机器人。假如人与机器人出现类夫妻、类父女等情感，必将拷问现代伦理规范。

体　验　思　考

2017年 10月 26日，沙特阿拉伯授予美国某机器人公司生产的机器人索菲亚公民身份。索菲亚拥有仿生橡胶皮肤，可模拟 62种面部表情，能够识别面部，并与人进行眼神接触，能够理解语言并与人类互动。

该案例中，沙特阿拉伯授予机器人索菲亚公民身份，是否意味着赋予了机器人完整的法律人格？我们应该赋予机器人多少权利？人与机器人的交往会不会影响到人与人之间的交往关系？请结合实际案例和应用场景，谈谈机器人公民会带来哪些问题。

二、法律问题

随着人工智能技术的发展与广泛应用，现有法律的局限性日益凸显，迫使法律法规必须不断做出修改和完善，包括交通、知识产权保护、个人隐私保护等多个领域。

无人驾驶应用将从工厂车间走向普通道路，交管部门应该支持无人驾驶技术创新，但无人驾驶应依法、安全、科学进行，对违反法律法规行为，要依法查处。

人工智能发展迅速，应用呈现井喷式增长，而立法相对滞后，当立法不能适应科技发展时，立法部门就要修改法律。一般立法周期较长，要迅速解决眼前的问题，就需要行政部门制定颁布相关条例和规定。为适应无人驾驶发展的需要，北京市于 2017 年 12 月发布了《北京市自动驾驶车辆道路测试管理实施细则（试行）》，上海市于 2018 年 2 月发布了《上海市智能网联汽车道路测试管理办法（试行）》，同年 3 月，上海市发放了全国首批智能网联汽车开放道路测试号牌，这标志着无人驾驶汽车正式走出封闭园区，进入上路测试阶段。无人驾驶合法化是大势所趋，无人驾驶汽车和由司机驾驶的汽车将会在未来很长一段时期内并存于交通领域，这将给交通事故责任的认定带来挑战。

此外，人工智能在创作领域已经有了初步的发展和应用，人工智能不仅能写歌作曲、写诗作画，还能写新闻稿甚至论文，人工智能作品的知识产权归属问题同样需要法律给予界定。人工智能需要大数据的支撑，在数据采集和数据挖掘过程中，可能会侵犯个人隐私和公司商业利益，甚至泄露政府涉密信息，这同样需要进一步完善相关法律法规，保护个人、公司和政府的合法权益。

体 验 思 考

与组员讨论，人工智能发展和应用还可能带来哪些法律问题？我们应该怎么应对？

三、人机社会的规范

1940 年，科幻作家艾萨克·阿西莫夫就在其科幻小说中提出了“机器人三定律”：

第一条　机器人不得伤害人类，或看到人类受到伤害而袖手旁观。

第二条　机器人必须服从人类的命令，除非这条命令与第一条相矛盾。

第三条　机器人必须保护自己，除非这种保护与以上两条相矛盾。

阿西莫夫利用自己提出的“机器人三定律”，为机器人建立了一套行为规范和道德准则，从而演绎出一系列推理性和逻辑性极强的精彩科幻故事。阿西莫夫的机器人小说系列还涉及如何解释三条定律，如何从心理学的角度预测机器人的智慧，三条定律如何互相抵触而产生矛

盾等。阿西莫夫小说的一个显著特点在于，他第一次站在机器人的角度从三条定律的基本点出发，试图找出一种人类和机器人更加融洽的关系。

2017 年 1 月，来自全球的 844 名人工智能和机器人领域的专家联合签署了“阿西洛马人工智能原则”，呼吁全世界的研究者和研究机构在开展人工智能领域研究的同时严格遵守这些原则，共同保障人类未来的伦理、利益和安全。该原则目前共 23 项，分为科研问题、伦理和价值、更长期的问题三大类。

科研问题类涉及 5 项条款。科学家们普遍认为，人工智能研究的目标应该是创造有益于人类而不是不受人类控制的智能。研究经费方面要保证部分科研经费用于研究如何确保有益地使用人工智能，包括计算机科学、经济学、法律、伦理以及社会研究中的棘手问题。科研文化建设方面，在人工智能研究者和开发者中应该培养一种合作、信任与透明的人文文化。科研要避免竞争，人工智能系统开发团队之间应该积极合作，以避免安全标准上的有机可乘。科学与政策方面，在人工智能研究者和政策制定者之间应该有建设性的、有益的交流。

伦理和价值类的条款有 13 项，以确保人工智能应用安全可靠、故障透明、司法透明、符合人类价值观。包括：要负责任地使用人工智能、尊重人的隐私与自由；应该保证让更多人享受到人工智能发展带来的好处，共享经济繁荣；人工智能的应用场景应该由人类决定，而不是机器自动选择；人工智能应用是为了改善社会秩序而不是颠覆社会；人工智能军备竞赛应该禁止等。

科学家还关注到了人工智能发展更长期的问题。如：警示大家应该避免对于未来人工智能能力上限的过高假设但未达成共识；人类应该有相应的关切和资源来进行计划和管理高级人工智能；必须有针对性地计划和努力减轻可预见的人工智能系统造成的风险冲击；必须有严格的安全和控制标准，控制进行递归自我升级或自我复制的人工智能系统；超级智能的开发是为了服务广泛认可的伦理观念，并且是为了全人类的利益，而不是某个国家或组织。

探 究 活 动

通过阅读课本、上网搜索、实地调研等方式，收集有关人工智能技术开发与应用带来的最新的道德、伦理、法律和社会问题，对收集到的资料进行分类、归纳、提炼，给出立法建议、解决办法或应对策略。利用整理好的资料，形成“未来人工智能应用及社会问题”研究报告。

作业练习

请你观看或阅读一部有关人工智能的科幻电影或科幻著作，分析影片或书籍中人工智能的发展处于哪个阶段，存在哪些主要社会冲突和矛盾，这些冲突和矛盾又是如何化解的，完成表 4.2。

表 4.2

电影(或书籍)名称	
人工智能所处发展阶段	
主要社会冲突和矛盾	
主要社会冲突和矛盾化解办法	

后　记

本册教科书依据教育部《普通高中信息技术课程标准（2017 年版 2020 年修订）》编写，并经国家教材委员会专家委员会审核通过。全体编写人员认真领会国家基础教育改革精神，精心研究当代信息社会的人才培养要求，广泛调研上海及各地高中信息技术教育的现状和挑战，深入了解高中学生的学习需求，并汲取了上海市《普通高中信息科技（试用本）》的编写经验。

编写过程中，上海市中小学（幼儿园）课程改革委员会专家工作委员会，上海市教育委员会教学研究室，上海市课程方案教育教学研究基地、上海市心理教育教学研究基地、上海市基础教育教材建设研究基地、上海市信息科技教育教学研究基地（上海高校“立德树人”人文社会科学重点研究基地）及基地所在单位华东师范大学等单位给予了大力支持，樊磊、袁中果、武迪、刘叶婷等老师作出了重要贡献。在此表示感谢！

本册教科书出版之前，我们已通过多种渠道与教科书选用作品（包括照片、画作）的作者进行了联系，得到了他们的大力支持。对此，我们衷心地表示感谢！恳请尚未联系到的作者与我们联系，以便出版社及时支付相关稿酬。

我们真诚地希望广大教师、学生及家长在使用本册教科书的过程中提出宝贵意见。我们将集思广益，不断修订，使教科书趋于完善。

编　者