揭秘会学习的机器：人工智能与机器学习基础原理及其实际应用威力的生动入门

第一章：机械之梦：电力时代之前的计算

现代计算机的嗡嗡声、屏幕发出的光芒、对海量信息库的即时访问——这些现象已深深植根于当代生活，以至于需要刻意努力才能想象一个没有它们的世界。然而，追求计算自动化、制造能够处理数字和符号的机器，其历史比电力本身的发明还要早几个世纪。这个由齿轮、杠杆和精巧的钟表机械驱动的时代，代表了计算的机械之梦。这是一个由人类与手工计算的易错性持续斗争、对精确度和速度的渴望，以及对机器能够承担先前专属于人类智力任务的初步构想所定义的时期。从简单的计数辅助工具到复杂但未实现的、用于可编程引擎的设计，这些早期的努力为即将到来的数字时代奠定了概念和机械基础，展示了一种将思维的某些方面外化和机械化的深刻而长远的雄心。

早期计算辅助工具：从算盘到对数

这段旅程并非始于复杂的机械，而是始于旨在辅助人脑完成计数和计算基础任务的基本工具。其中最早且最持久的是算盘。其起源可追溯至数千年前的古美索不达米亚，并在罗马、中国、日本和俄罗斯等不同文化中出现演变形式，算盘提供了一个使用沿杆或线移动的珠子或石子来表示数字的实体系统。虽然看似简单，它极大地减轻了算术的认知负担，减少了计数和基本运算中的错误，并且至今仍在世界部分地区作为实用工具。它体现了第一步：创建一个外部物理系统来管理数字信息。

一个更为神秘的早期机械精密度的惊鸿一瞥来自安提基特拉机械装置（Antikythera mechanism），这是一件从希腊安提基特拉岛附近一艘罗马时代沉船中打捞出的惊人复杂的文物。这个可追溯至公元前2世纪的装置，由错综复杂的青铜齿轮组成，被认为是世界上已知最古老的模拟计算机。它被设计用来预测天文位置、日食，甚至古代奥林匹克运动会的周期。安提基特拉机械装置揭示了对齿轮传动和周期性现象的非凡理解，表明通过机械方式建模和预测复杂系统的雄心在几千年前就已存在，即使这些知识后来失传或未能广泛传播。

算盘辅助算术，安提基特拉机械装置模拟宇宙，而计算领域的一次重大概念飞跃发生在17世纪初，苏格兰数学家约翰·纳皮尔（John Napier）于1614年发明了对数。对数提供了一种革命性的数学捷径：通过使用预先计算好的数表，将乘法和除法转换为简单得多的加法和减法运算，它们极大地减少了复杂科学计算所需的时间和精力，尤其是在天文学和航海领域，这些领域大数计算司空见惯。纳皮尔在1617年左右发明了“纳

皮尔骨头”（Napier's Bones），进一步辅助了实际计算。这是一套带有标记的杆，通过基于格子乘法的巧妙视觉系统简化了多位数乘法。

直接建立在对数原理之上的是计算尺，它在不久之后出现，由埃德蒙·甘特（Edmund Gunter）和威廉·奥特雷德（William Oughtred）等人在17世纪20年代和30年代贡献发展而成。通过在滑动条或圆盘上刻上对数刻度，计算尺允许用户通过物理上在刻度上加减长度来快速执行乘法、除法、开方和乘方运算。虽然计算尺是一个提供近似结果而非精确数字和的模拟设备，但它成为了工程师、科学家和学生们长达三个多世纪的普遍计算工具，其统治地位直到20世纪70年代廉价电子计算器的出现才告终。

这些前机械时代的辅助工具，从算盘的有形计数框架到计算尺所体现的对数概念力量，都是至关重要的垫脚石。它们解决了计算中对更高速度和准确性的基本需求，突显了繁琐计算中人为错误的持续问题，并为通过物理机制自动化这些过程奠定了基础。它们代表了无辅助人类计算和手动辅助工具的局限性，为第一批真正的计算机器铺平了道路。

最早的机械计算器：帕斯卡和莱布尼茨

17世纪标志着从计算辅助工具向自动化计算机器的过渡。虽然德国博学家威廉·施卡德（Wilhelm Schickard）在1623年设计了一台“计算钟”，将纳皮尔骨头与加法机制相结合，但历史的不幸意味着他的工作直到20世纪才广为人知。因此，第一台被广泛认可且具有影响力的机械计算器诞生于法国。布莱士·帕斯卡（Blaise Pascal），一位才华横溢的数学家和哲学家，大约在1642年，年仅18岁时就开始了这项挑战。为了减轻他父亲（一位税务监督员）繁重的算术劳动，帕斯卡开发了一种能够直接执行加法和减法（乘法和除法可通过繁琐的重复实现）的机器。

经过多年的改进和无数原型机，帕斯卡于1645年公开展示了他的“帕斯卡计算器”（Pascaline）。其核心创新在于其可靠的进位机制。当一个数字轮从9转到0时，一个复杂的棘爪装置——*sautoir*——会短暂地抬起一个小重物，然后重物落下，推动下一个轮前进一位。这使得进位能够有效地跨多个数字位传播，这是机械设计中的一个重大挑战。帕斯卡于1649年获得了皇家特权，授予他在法国生产此类机器的独家权利。然而，以当时的技术水平制造帕斯卡计算器既复杂又昂贵。可能总共只制造了不到二十几台，限制了它的实际影响，但它建立了一个关键的概念验证：通过齿轮机制实现自动化计算是可能的。

受到帕斯卡成就的启发，但旨在实现更强大的功能，德国博学家戈特弗里德·威廉·莱布尼茨（Gottfried Wilhelm Leibniz）大约在1671年开始设计自己的计算器。莱布尼茨，微积分的发明者之一，明确寻求一种能够自动化所有四种基本算术运算的机器：加法、减法、乘法和除法。他最终的设计，“步进计算器”（Stepped Reckoner），引入了一项关键创新：“莱布尼茨轮”或称步进轮。这是一个圆柱体，其侧面有九个长度不等的齿。旋转该轮会带动相应的齿轮，根据轮的旋转位置和齿轮沿轮的横向位置，将特定的数字（0到9）加到累加器中。通过使用多个步进轮和一个可移动的托架（概

念上类似于后来的计算器），步进计算器能够比帕斯卡计算器更系统地通过重复加法执行乘法，通过重复减法执行除法。

莱布尼茨大约在1694年完成了他的第一个原型，并在1706年左右完成了第二个。然而，与帕斯卡的机器一样，步进计算器也受到17世纪机械制造能力的限制。达到齿轮切割所需的精度，尤其是在乘法过程中跨多个数字位的复杂进位机制，被证明极其困难。最终只制造了两台原型机，且只有一台幸存至今。尽管缺乏实际成功，莱布尼茨的步进轮机制被证明具有极高的影响力，成为接下来200年许多成功制造的机械计算器的基础。

帕斯卡和莱布尼茨的工作确立了机械计算的基本原理。帕斯卡展示了一种带有有效进位机制的可靠加法器，而莱布尼茨构想了第一个四则运算计算器的设计，并引入了经久不衰的步进轮。他们的挣扎凸显了杰出概念设计与制造技术的实际限制之间的差距。又过了一个半世纪，制造技术才得以充分发展，通过查尔斯·泽维尔·托马斯·德·科尔马（Charles Xavier Thomas de Colmar）的算术计（Arithmometer），在商业上可行的形式下实现了莱布尼茨的雄心。该算术计于1820年获得专利，并从1851年开始成功量产。基于莱布尼茨的步进轮，算术计成为第一款足够坚固、可供广泛办公室使用的计算器，开启了机械计算器产业，并最终实现了自动化算术的早期机械之梦。

巴贝奇的引擎：计算的蓝图

当帕斯卡、莱布尼茨及其后继者专注于自动化算术时，英国博学家查尔斯·巴贝奇（Charles Babbage，1791-1871）构想了更为深远的东西：不仅能计算，而且能根据预设的操作序列进行*运算*的机器。他的工作代表了从计算器到计算机的巨大概念飞跃，奠定了一个世纪后将定义计算机器的架构原则。

巴贝奇最初的动机源于他对广泛用于航海、天文学和工程学的手工计算数学用表（如对数表和三角函数表）中普遍存在的错误感到沮丧。他认为这些源于人类在计算和抄写中易犯错误的误差不仅不方便，而且代价高昂，可能导致海难和工程失败。大约在1821-22年，他构想了他的第一台伟大机器：差分机（Difference Engine）。

差分机是为一项特定但至关重要的任务而设计的：自动计算和制表多项式函数。其天才之处在于采用了有限差分法。这种方法允许仅使用重复加法来计算多项式值，避免了乘法或除法，而后者在机械上实现可靠性要困难得多。巴贝奇的设计要求一个由黄铜齿轮和杆组成的、代表十进制数的大型复杂装置。独特的是，它还被设计成能自动将其结果直接打印到铅版上，从而消除了人工抄写和排版过程中出错的风险。

认识到该项目的规模和潜在重要性，巴贝奇获得了大量的政府资助，这是国家资助技术研发的最早实例之一。然而，该项目困难重重。制造设计所需的数千个、达到前所未有精度水平的部件，对19世纪工程技术的能力构成了严峻挑战。成本不断攀升，工期延误，巴贝奇与他的总工程师约瑟夫·克莱门特（Joseph Clement）之间也出现了争端。此外，巴贝奇自己活跃的思维开始转向一个更宏伟的想法。经过多年的工作和

巨额开销，到1832年，差分机1号（能够处理高达20位数字并计算七阶差分）仅完成了一部分。政府资助在1842年正式撤销。巴贝奇后来设计了一款更优雅、更高效的差分机2号（1847-49），吸取了经验教训，但这个版本在他有生之年并未建成。几十年后，在20世纪末，一台能工作的差分机2号根据巴贝奇的详细图纸被忠实地建造出来，证明了他的机械设计的健全性。

在差分机项目受挫的同时，巴贝奇大约在1834年构思了他最具革命性的机器：分析机（Analytical Engine）。这并非一台专门的计算器，而是一台通用、可编程、自动化的机械计算机。它代表了从自动化算术到自动化计算本身的根本性转变。该设计包含了许多与现代计算机惊人相似的、具有预见性的特性：

1. **存储与处理分离：** 分析机拥有一个专门的“存储库”（Store，内存），用于存放数字（计划容量：1000个50位十进制数）和中间结果，与进行实际计算的“处理单元”（Mill，算术逻辑单元）分开。这种分离至今仍是计算机体系结构（冯·诺依曼结构，尽管是后来独立构思的）的基本原则。
2. **可编程性：** 指令和数据通过穿孔卡片序列输入分析机，这是巴贝奇从雅卡尔织布机借鉴的概念，该织布机使用穿孔卡片控制复杂图案的编织。这使得机器变得可编程，能够根据卡片上提供的指令执行不同的计算。
3. **条件分支：** 设计包含了“条件控制转移”机制。这意味着分析机可以根据先前计算的结果（例如，测试一个数是正数还是负数）来改变其操作顺序，从而在一个程序内实现决策——这是复杂算法的关键特性。
4. **输入/输出：** 计划包括用于输入的穿孔卡片阅读器、用于输出的卡片打孔机和打印机，甚至还有用于图形输出的绘图仪。

分析机被设想为一个由蒸汽驱动的、体现了工业时代力量的巨大机器。然而，其机械复杂性比差分机高出几个数量级。它需要精密复杂的齿轮、杠杆和连杆系统，远远超出了那个时代的制造能力。再加上政府在差分机经历后的失望情绪，资金从未得到保障。巴贝奇一生都在不断完善设计，但分析机从未被建造出来。

尽管只停留在蓝图阶段，分析机仍是一项卓越的智力成就。它是第一个通用、可编程计算设备的完整设计，阐明了一个世纪后将在电子形式中被独立重新发现和实现的核心架构概念。巴贝奇的工作表明，通用计算的思想远在建造它的技术出现之前就已可以构想。他的挣扎也凸显了技术愿景对制造物质现实的关键依赖。可悲的是，他开创性的工作在他去世后相对默默无闻，后来建造第一批电子计算机的先驱们大多对此一无所知。

阿达·洛夫莱斯：计算机时代的预言家

若非奥古斯塔·阿达·金，洛夫莱斯伯爵夫人（Augusta Ada King, Countess of Lovelace，1815-1852）的洞察力，巴贝奇分析机的概念意义和潜力可能仍被锁在他

的复杂设计图纸之中。作为著名诗人拜伦勋爵的女儿，洛夫莱斯拥有非凡的数学天赋，尽管19世纪女性在科学领域面临社会限制，她的这份热情仍得到了培养。她于1833年结识查尔斯·巴贝奇，并被他精妙的机械设计所吸引，由此开启了一段长期而富有成果的智力友谊与合作。

洛夫莱斯的决定性贡献源于她翻译的一篇关于分析机的论文，该论文由意大利工程师路易吉·梅纳布雷亚（Luigi Menabrea，后任意大利首相）用法文撰写，基于巴贝奇1840年在都灵所做的讲座。当巴贝奇的朋友查尔斯·惠斯通（Charles Wheatstone）邀请她将该文翻译成英文并在英国发表（载于1843年的*泰勒科学回忆录*）时，洛夫莱斯承担了这项任务。然而，她所做的远不止翻译；在巴贝奇的鼓励下，她加入了自己详尽的注释，标题仅为“笔记”（Notes）。这些笔记最终的篇幅几乎是梅纳布雷亚原文的三倍，并包含了对分析机的性质和可能性的深刻见解。

在被称为“笔记G”的部分，洛夫莱斯细致地描述了分析机计算伯努利数（数论中出现的一个复杂序列）的操作序列。她概述了如何使用提议的穿孔卡片系统，一步步地指示机器执行必要的公式。这份详细的计划被计算机历史学家广泛认为是世界上第一个公开发表的计算机程序或算法。它提供了具体的证据，证明分析机的抽象设计可以应用于解决简单的算术之外的复杂数学问题。

然而，洛夫莱斯的贡献远不止于这个具体的算法。她对分析机的潜力有着独特的概念把握，不仅视其为数字处理机器，更视其为符号操纵器。她认识到，如果机器能根据既定规则操作数字，那么它原则上也能操作任何其相互基本关系可以用抽象符号表达的实体——正如她所阐述的，“数字以外的其他事物”。在一个引人注目的段落中，她推测：

> “分析机或许能作用于数字以外的其他事物，只要能找到其相互基本关系可用运算抽象科学表达的对象……譬如，假设和声学与音乐创作科学中音高的基本关系能被如此表达和调整，那么引擎便能创作出任何复杂程度或篇幅的精细且科学的乐曲。”

她设想机器能编织“代数图案，正如雅卡尔织布机编织花朵和叶子一样”。这种对通用符号操纵潜力的洞察，预见了一个世纪后现代计算的通用性——其处理文本、逻辑、图像和声音，而不仅仅是数字的能力。她洞察了我们现在称之为计算的本质：根据规则操纵符号。

洛夫莱斯还就机器在创造性和原创性方面的内在局限性提出了一个关键观点。在她笔记中一段常被引用的文字中，她写道：

> “分析机丝毫没有声称能创造任何东西。它能做任何我们知道如何命令它去执行的事情。它能遵循分析；但它没有任何能力去预见任

> 何分析关系或真理。”

这段论述，有时被称为“洛夫莱斯夫人的异议”（Lady Lovelace's Objection），强调了执行编程指令（无论多么复杂）与真正的独立思考或创造力之间的区别。尽管后来的阿兰·图灵（Alan Turing）等人物会在人工智能的背景下讨论这一异议，但洛夫莱斯的观察提出了一个关于机器能力和界限的基本问题，至今仍在人工智能的讨论中具有现实意义。

如同巴贝奇的引擎一样，阿达·洛夫莱斯的贡献在她生前及36岁因癌症早逝后的几十年里基本上未获赏识。她的工作直到20世纪中期电子计算出现后才被重新发现并获得重视。如今，她被恰当地颂扬为一位有远见的人物：第一位计算机程序员和一位敏锐的分析家，她在计算机器成为现实很久之前就预见到了其变革性的潜力。她的遗产不仅体现在Ada编程语言的命名上，也体现在她对计算的机械之梦所蕴含可能性的深刻阐述中，这个梦想等待着电子的火花来完全实现。

第二章：电子黎明：锻造数字时代

帕斯卡、莱布尼茨和巴贝奇的复杂钟表式机械代表了机械之梦的顶峰，推动了齿轮和杠杆所能达到的极限。然而，这些依赖物理运动的机器在速度、可靠性和复杂性方面面临着固有的局限。摩擦、磨损、惯性以及制造数千个精密、相互啮合部件的巨大困难限制了进一步的发展。需要一种根本性的新技术来打破这些障碍，释放自动化计算的真正潜力。这项技术就是电子学，利用电子的流动而非齿轮的啮合。20世纪中期见证了“电子黎明”，这是一个变革时期，理论突破与工程独创性相结合，常常在全球冲突的熔炉中加速发展，锻造出第一批真正的电子数字计算机，并确立了至今仍支撑着计算的架构范式。

理论基础：图灵、香农、冯·诺依曼

在电子计算机能够被建造出来之前，一个理论基础是必需的。这涉及到理解电子元件如何逻辑地表示和操纵信息，定义计算本身的极限和本质，并为这些新机器设计出高效的架构。三位人物因其对这一基础的开创性贡献而脱颖而出：克劳德·香农（Claude Shannon）、艾伦·图灵（Alan Turing）和约翰·冯·诺依曼（John von Neumann）。

第一个挑战是弥合抽象逻辑与物理电路之间的鸿沟。虽然像乔治·布尔（George Boole）这样的19世纪逻辑学家已经发展出用于表示逻辑命题（真/假）的代数系统，查尔斯·桑德斯·皮尔士（Charles Sanders Peirce）也注意到了逻辑与电子开关电路之间的联系，但正是克劳德·香农在20世纪正式确立了这一关键联系。在他开创性的1937年麻省理工学院硕士论文《继电器和开关电路的符号分析》（A Symbolic Analysis of Relay and Switching Circuits）中，香农证明了布尔代数可以直接应用于由充当开关（开/关，代表真/假或1/0）的机电继电器或真空管构建的电路的设计和分析。他展示了如何通过这些开关网络实现复杂的逻辑运算，为工程师设计数字电路

提供了强大的数学工具包。大约在同一时期，日本的中岛章（Akira Nakashima）和苏联的维克托·舍斯塔科夫（Victor Shestakov）也独立地发展了类似的见解。然而，香农的工作成为了西方数字电路设计的基础，将抽象逻辑转化为实用的工程原理。

香农提供了构建计算电路的语言，而艾伦·图灵则解决了一个更根本的问题：什么*可以*被计算？在他里程碑式的1936年论文《论可计算数及其在判定问题上的应用》（On Computable Numbers, with an Application to the Entscheidungsproblem）中，图灵引入了一个抽象的计算数学模型，现在被称为图灵机。这个理论装置由一条无限长的、划分成单元格的纸带、一个可以在纸带上移动的读写头，以及一组有限的状态和规则组成。根据读取的符号及其当前状态，机器写入一个符号，移动读写头，并改变状态。尽管结构简单，图灵证明了这个模型原则上可以执行任何可以用算法描述的可想象的数学计算。他定义了“图灵完备性”的概念——一个计算系统（如编程语言或计算机）能够模拟通用图灵机的属性。论文中还介绍了通用图灵机，这是一种特殊的图灵机，能够模拟*任何*其他图灵机，只要在其纸带上给出适当的描述（程序）。这为通用计算机——能够执行任何有效指令集的机器——奠定了理论基础。图灵的工作确立了可计算性理论领域，定义了机器能够计算和不能计算的理论极限（例如，停机问题的不可解性）。

随着计算的理论可能性得以确立，以及在电路中实现逻辑的方法被定义，最后一块拼图是组织电子计算机的实用蓝图。虽然像ENIAC这样的早期机器功能强大，但它们的编程非常繁琐。关键的洞见，产生于ENIAC及其计划中的后继者EDVAC的设计者们激烈的讨论中，是存储程序概念。这个想法由数学家约翰·冯·诺依曼在他1945年的文件《EDVAC报告初稿》（First Draft of a Report on the EDVAC）中著名地阐述出来。综合了与J. Presper Eckert, John Mauchly, Herman Goldstine, Arthur Burks以及参与项目的其他人共同发展的想法，冯·诺依曼勾勒出一种架构，其中程序指令和被处理的数据都存储在同一个快速、可寻址的电子存储器中。

这是革命性的。指令不再需要外部接线或从磁带缓慢输入，而是可以像数据一样以电子速度从内存中取出。关键是，这意味着程序可以被当作数据对待——它们可以被轻松加载、修改，甚至由其他程序生成。“冯·诺依曼结构”，正如后来人们所熟知的（尽管承认整个EDVAC团队的贡献至关重要），通常包括：

1. 一个中央处理单元（CPU），包含用于计算和逻辑运算的算术逻辑单元（ALU），以及用于取指、解码和执行指令的控制单元。
2. 一个主存储器单元，同时存储数据和指令。
3. 用于与外部世界交互的输入/输出（I/O）机制。
4. 连接这些组件的通信路径（总线）。

这种以指令和数据共享内存空间以及顺序指令处理（一次取一条指令）为特征的架构，成为了接下来几十年计算机设计的主导范式，并且至今仍然具有高度影响力。香

农的逻辑电路综合、图灵的可计算性理论，以及冯·诺依曼的存储程序结构共同提供了构建电子数字计算机所必需的理论支柱。

早期电子和机电计算机：战时催化剂及以后

从纯机械计算向电子计算的过渡并非一蹴而就。几台使用机电继电器构建的重要机器弥合了差距，而第二次世界大战的巨大压力极大地加速了电子技术的发展和应用。

在德国，由于战争原因相对孤立地工作，康拉德·楚泽（Konrad Zuse）独立开创了几个计算概念。他的Z1（1938年）是一台机械二进制计算器，但他的Z3（1941年完成）是一项了不起的成就。它使用大约2300个电话继电器构建，采用二进制浮点运算，并由打孔在废弃电影胶片上的程序控制。尽管在1943年被盟军轰炸摧毁，Z3被许多历史学家认为是第一台可运行的、程序控制的、通用的（尽管实际上并未如此使用）计算机。楚泽的工作证明了使用现成继电器技术进行复杂计算的可行性。

在美国，贝尔电话实验室的乔治·斯蒂比茨（George Stibitz）也开发了一系列基于继电器的计算器，始于他在自家厨房桌子上搭建的“K模型”。他的复数计算器（Complex Number Calculator, CNC），于1939年完成，可以对复数进行算术运算。在1940年的一次里程碑式的演示中，斯蒂比茨在新罕布什尔州达特茅斯学院的一次会议上，通过电传打字机连接远程操作了位于纽约市的CNC，预示了远程计算和网络。

与此同时，在哈佛大学，霍华德·艾肯（Howard Aiken）在IBM的大力支持下，开发了哈佛Mark I（也称为IBM自动序列控制计算器或ASCC）。这台机电巨兽于1944年完成，长超过50英尺，重约5吨，包含数百英里的电线和数千个继电器及机械计数器。它从穿孔纸带上顺序读取指令，并在战争期间被美国海军广泛用于计算。尽管规模宏大且具有影响力，Mark I代表的是机电技术的顶峰，而非电子时代的黎明。

真正向电子化的飞跃始于阿塔纳索夫-贝瑞计算机（Atanasoff-Berry Computer, ABC），由约翰·阿塔纳索夫（John Atanasoff）和他的研究生克利福德·贝瑞（Clifford Berry）于1937年至1942年间在爱荷华州立学院（Iowa State College）建造。ABC是第一台使用真空管（约300个）进行数字逻辑运算的计算设备。它包含了几项创新：二进制算术、一种使用安装在旋转鼓上的电容器的再生存储器形式（需要周期性刷新以维持电荷，是现代DRAM的前身），以及并行处理架构。然而，ABC是专门为解决线性方程组而设计的，并不是通用意义上的可编程。它的发展因战争而中断，从未完全投入常规使用。关键的是，约翰·莫奇利（John Mauchly）在1941年访问了阿塔纳索夫并观察了ABC；这次访问后来成为1973年专利诉讼案（霍尼韦尔诉斯佩里兰德案，Honeywell v. Sperry Rand）的核心，该案使ENIAC专利无效，并宣布阿塔纳索夫是电子数字计算机若干关键概念的创始人。

第二次世界大战为更快的计算提供了巨大的推动力。密码学（破译密码）和弹道学（计算火炮射击表）的需求，要求计算速度远超机械或机电装置所能提供的。在英

国，布莱切利园（Bletchley Park）为破译加密的德国通信而进行的绝密工作，导致了Colossus（巨人）计算机的开发。主要由汤米·弗劳尔斯（Tommy Flowers）设计，并从1943年底开始运行，Colossus机器是第一批大规模电子计算机。它们使用大约1500-2400个真空管，高速执行逻辑运算（布尔比较），以帮助破译由复杂的德国洛伦兹（Lorenz）密码机加密的消息。Colossus机器在一定程度上是可编程的，通过插线板上的插头和开关为不同任务进行配置，但它们是专用的密码破译设备，不是通用计算机。它们的存在在战后几十年间一直保密，但它们的成功明确证明了电子计算在处理复杂逻辑任务上的速度和能力优势。（在布莱切利园同样至关重要的是机电式的Bombe（炸弹）机，它在波兰工作的基础上设计，并由图灵和哈罗德·基恩（Harold Keen）改进，用于破解恩尼格玛（Enigma）密码）。

被广泛认为是第一台通用电子数字计算机的机器是ENIAC（电子数值积分计算机，Electronic Numerical Integrator And Computer）。由宾夕法尼亚大学摩尔电气工程学院的约翰·莫奇利和J.普雷斯珀·埃克特（J. Presper Eckert）开发，并由美国陆军弹道研究实验室资助，ENIAC于1945年完成，并在1946年2月公开亮相。它是一个庞然大物：包含近18000个真空管、70000个电阻器、10000个电容器和6000个手动开关，占据了一个大房间，重约30吨，消耗大量电力。但它速度很快，计算能力大约是像哈佛Mark I这样的机电机器的1000倍。ENIAC内部使用十进制算术，并且是图灵完备的。然而，对它编程是出了名的困难，需要手动设置数千个开关并将电缆插入插线板来定义操作序列——这个过程可能需要几天或几周。大部分这种艰苦的编程工作是由六位女性数学家（凯·麦克纳尔蒂（Kay McNulty）、贝蒂·詹宁斯（Betty Jennings）、贝蒂·斯奈德（Betty Snyder）、玛琳·韦斯科夫（Marlyn Wescoff）、弗兰·比拉斯（Fran Bilas）和露丝·利希特曼（Ruth Lichterman））完成的，她们现在被公认为编程先驱。

这个时代生动地展示了理论概念、工程实力和紧迫需求之间的相互作用。战时需求催化了从较慢的继电器向更快但可靠性较低且耗电量大的真空管的转变。像ABC和Colossus这样的机器证明了电子计算在特定任务上的可行性，而ENIAC尽管存在编程限制，却代表了通用电子计算机器的首次实现，为下一个关键创新——存储程序——奠定了基础。

存储程序革命：曼彻斯特宝贝、EDVAC、UNIVAC

通过手动重新布线来重新编程ENIAC的繁琐过程凸显了一个主要瓶颈。在ENIAC/EDVAC开发期间构思并由冯·诺依曼阐述的解决方案是优雅且具有变革性的：将程序指令与数据一起存储在计算机的高速电子存储器中。这个存储程序概念将允许程序快速加载和更改，将计算机从一台功能强大但僵化的计算器转变为一台真正多功能、由软件控制的机器。

1940年代末，英国和美国的几个团队竞相建造第一台可运行的存储程序计算机。第一台可证明在其内存中电子存储并运行程序的机器是曼彻斯特小型实验机（Manchester

Small-Scale Experimental Machine, SSEM），昵称为“曼彻斯特宝贝”（Manchester Baby）。由弗雷德里克·威廉姆斯（Frederic Williams）、汤姆·基尔伯恩（Tom Kilburn）和杰夫·图蒂尔（Geoff Tootill）在曼彻斯特大学建造，它于1948年6月21日成功执行了第一个简单程序（找到一个数的最大真因子）。“宝贝”主要是作为威廉姆斯-基尔伯恩管（Williams-Kilburn tube）的试验平台而设计的，这是一种使用阴极射线管（CRT）将比特存储为电荷点的新型电子存储器。尽管规模小且是实验性的，它的成功运行标志着一个关键时刻——现代意义上软件的诞生。

此后不久，在1949年5月，EDSAC（Electronic Delay Storage Automatic Calculator，电子延迟存储自动计算器）在剑桥大学数学实验室投入运行，由莫里斯·威尔克斯（Maurice Wilkes）领导。受到冯·诺依曼《初稿》的启发，EDSAC使用水银延迟线作为存储器（将比特存储为在水银管中传播的声脉冲）。虽然不是最早的，但EDSAC可以说是第一台实用且完全可运行的存储程序计算机，从一开始就旨在为大学研究人员提供常规计算服务。它具有许多创新，包括汇编器和子程序库，为早期软件开发实践做出了重大贡献。

在美国，EDVAC（Electronic Discrete Variable Automatic Computer，电子离散变量自动计算机），即冯·诺依曼报告最初为其撰写的机器，面临各种技术和人员延误，直到1949年才完成，此时英国的机器已经运行。其他早期的存储程序计算机包括BINAC（Binary Automatic Computer，二进制自动计算机），由埃克特和莫奇利新成立的公司为诺斯罗普飞机公司（Northrop Aircraft）建造（1949年完成，但问题缠身），以及澳大利亚的CSIRAC（1949年末运行），它现在是现存最古老的第一代电子计算机。

1950年代初标志着从这些通常作为一次性项目建造的实验室机器向面向政府和商业用户的商业制造计算机的关键过渡。埃克特和莫奇利离开宾夕法尼亚大学后，成立了埃克特-莫奇利计算机公司（Eckert-Mauchly Computer Corporation，后被雷明顿兰德公司（Remington Rand）收购）。他们的旗舰产品是UNIVAC I（Universal Automatic Computer I，通用自动计算机I）。第一台UNIVAC于1951年3月交付给美国人口普查局，成为美国第一台商业生产的电子数字计算机。UNIVAC使用水银延迟线存储器和磁带进行输入/输出。当它在1952年总统大选中根据早期民意调查结果正确预测了德怀特·D·艾森豪威尔（Dwight D. Eisenhower）的压倒性胜利时，获得了显著的公众关注，震惊了评论员，并展示了计算机在纯粹科学计算之外处理大规模数据的潜力。UNIVAC在一段时间内成为了公众心目中“计算机”的代名词。

在英国，受到EDSAC的启发，餐饮公司J. Lyons & Co.资助开发了LEO I（Lyons Electronic Office，里昂电子办公室）。LEO于1951年投入运行，成为世界上第一台致力于常规商业应用的计算机，运行诸如工资单和库存管理之类的任务。

IBM最初在投资了机电式的Mark I之后对电子计算机持谨慎态度，但在1950年代初果断进入市场。IBM 701（国防计算器），于1952年发布，面向科学和技术市场，使用威廉姆斯-基尔伯恩管存储器。紧随其后的是1954年大获成功的IBM 650。650使用旋转

磁鼓存储器，它比CRT或延迟线慢，但更便宜、更可靠，使其对商业用途具有吸引力。IBM利用其既有的销售网络和在商用机器市场的声誉，生产了近2000台650。这一成功将IBM推向了在新兴的大型计算机行业中持续数十年的主导地位。

存储程序概念的开发和实施是一个分水岭时刻，可以说是计算历史上最重要的单一架构创新。它将程序与硬件配置分离，实现了现代计算机所特有的灵活性和强大功能。随后，雷明顿兰德（UNIVAC）和IBM等公司转向商业生产，确立了计算机产业的基础，为未来几十年的快速技术进步以及最终将计算融入现代生活几乎所有方面奠定了舞台。电子黎明不仅已经到来，而且开始照亮世界。

第三章：人工智能的诞生：一个新领域的形成

电子计算机的发明和普及标志着人类历史的一个深刻转折点，其意义远不止于加速计算。当这些由真空管和电线组成的复杂装置开始以前所未有的速度执行复杂的指令序列时，一个诱人的问题出现了，从科幻小说的领域进入了切实的科学探究：这些能够如此灵巧地操纵数字和符号的机器，能否有一天被赋予*思考*的能力？智能本身，这种定义人类认知的难以捉摸的品质，能否在硅和电路中被复制或模拟？20世纪中期见证了一个致力于追求这一大胆目标的新科学学科的正式诞生：人工智能（Artificial Intelligence, AI）。这个领域并非由单一思想凭空产生，而是源于哲学探究、数理逻辑、生物学启发、工程实力以及电子计算新近可用的潜力所释放出的巨大能量的汇合。

早期的种子：图灵测试、控制论和基础思想

就在工程师们努力应对建造第一批电子巨兽的实际挑战时，富有远见的思想家们开始思考它们的最终潜力。其中最具影响力的是艾伦·图灵，他早期的工作为可计算性奠定了理论基础。在他于1950年发表在哲学期刊*Mind*上的开创性论文《计算机器与智能》（Computing Machinery and Intelligence）中，图灵探讨了这个棘手的问题：“机器能思考吗？”认识到定义“思考”的固有模糊性，他提出了一个务实的替代方案：“模仿游戏”（Imitation Game），现在普遍称为图灵测试（Turing Test）。

该测试涉及一个人类询问者与两个看不见的参与者——一个是人类，另一个是机器——进行基于文本的对话。图灵认为，如果在持续的对话后，询问者无法可靠地区分机器和人类，那么这台机器就可以被认为具有智能行为的能力。图灵测试巧妙地回避了关于意识和意向性的深层哲学辩论，而是提供了一个基于对话能力的机器智能操作基准。尽管其有效性和实用性此后一直备受争论，但该测试为这个 nascent 领域提供了一个强大的框架概念和一个明确的目标。在同一篇论文中，图灵有先见之明地回应了潜在的反对意见，包括源自阿达·洛夫莱斯的论点，即机器只能做它们被编程去做的事情（“洛夫莱斯夫人的异议”）。图灵反驳说，足够复杂的机器，可能包含学习机制，可以表现出令人惊讶且看似原创的行为，模糊了遵循指令与真正能力之间的界限。

与此同时，出现了另一股深刻影响早期AI思想的智力潮流：控制论（Cybernetics）。由诺伯特·维纳（Norbert Wiener）在其1948年的著作《控制论：或关于在动物和机器中控制和通信的科学》（Cybernetics: Or Control and Communication in the Animal and the Machine）中率先提出，这个跨学科领域探索了生物有机体和复杂机器共有的反馈、控制、通信和自我调节原理。控制论关注系统如何通过反馈回路——观察行动结果并相应调整行为——来维持稳定、适应变化的环境并追求目标。像体内平衡、信息论和负反馈这样的概念，为理解有目的的行为提供了理论框架，影响了早期AI研究人员思考如何设计能够智能地与世界互动并做出响应的系统。

除了哲学探究和控制理论，生物学的启发也起到了关键作用。如果目标是复制智能，那么人脑似乎是显而易见的模型。1943年，神经生理学家沃伦·麦卡洛克（Warren McCulloch）和逻辑学家沃尔特·皮茨（Walter Pitts）发表了《神经活动中内在思想的逻辑演算》（A Logical Calculus of the Ideas Immanent in Nervous Activity）。他们提出了一个高度简化的生物神经元数学模型，一个抽象单元，接收输入，将它们求和，如果总和超过某个阈值，则“激发”一个输出。至关重要的是，麦卡洛克和皮茨证明了由这些简单单元构建的网络原则上可以计算任何逻辑函数（与、或、非）。这一理论结果表明，复杂的认知过程可能源于许多简单处理元素的互连活动，为连接主义（connectionism）——即智能可能源于分布式网络而非单一、集中的处理器，模仿大脑结构的观点——提供了早期的理论基础。

这些理论和概念探索与早期尝试将现有计算机编程以执行被认为是人类智能标志的任务并行进行。游戏，特别是国际象棋和跳棋，迅速成为一个有吸引力的试验平台。这些游戏涉及逻辑、策略、远见以及在巨大的组合搜索空间中导航的能力——这些挑战似乎需要智力。克劳德·香农，已因其连接逻辑和电路的工作而闻名，在1950年发表了一篇论文，概述了编程计算机下国际象棋的策略，讨论了诸如评估函数和搜索算法（极小化极大算法，minimax）等概念。英国的迪特里希·普林茨（Dietrich Prinz）大约在1951年为费兰提Mark 1（Ferranti Mark 1）计算机开发了一个初步的国际象棋程序。

也许早期AI实践中最显著的例子是亚瑟·塞缪尔（Arthur Samuel）的跳棋（draughts）程序，从20世纪50年代初开始在IBM机器上迭代开发。塞缪尔的程序之所以重要，不仅在于它能下跳棋，更在于它开创性地融入了机器学习。它可以通过两种关键机制随时间提高其性能：死记硬背学习（记住先前遇到的棋盘位置及其结果），以及更重要的参数调整（根据与自身和人类对手对弈的结果调整其评估函数中的权重）。到20世纪60年代初，塞缪尔的程序能够达到可敬的业余水平，为机器确实可以从经验中学习提供了有力的证据。

这些不同的线索——图灵的哲学挑战和操作测试、维纳的控制论原理（控制与反馈）、麦卡洛克和皮茨的简化神经模型，以及早期在游戏领域的编程实验——共同构成了人工智能领域将正式萌芽的智力土壤。它们表明，机器智能问题不再纯粹是推测性的，而是变得可以通过理论分析和实践实验来研究。

达特茅斯研讨会（1956）：AI得其名

随着电子计算机能力的增强以及在研究人员中流传的基础思想，对机器智能的兴趣日益浓厚，并在1956年夏天达到高潮。一个关键事件，现在被广泛认为是人工智能作为一个独特领域的正式诞生，在新罕布什尔州汉诺威市的达特茅斯学院（Dartmouth College）持续了数周。

这个事件是“达特茅斯人工智能夏季研究项目”（Dartmouth Summer Research Project on Artificial Intelligence），由四位杰出的研究人员构思和组织：约翰·麦卡锡（John McCarthy，当时是达特茅斯的一位年轻数学家，后在麻省理工学院和斯坦福大学任职）、马文·明斯基（Marvin Minsky，哈佛大学的初级研究员，后成为麻省理工学院AI实验室的基石）、纳撒尼尔·罗切斯特（Nathaniel Rochester，IBM波基普西实验室信息研究经理）和克劳德·香农（Claude Shannon，来自贝尔实验室的著名信息理论家）。他们在1955年撰写的提案大胆陈述了项目的基本猜想：“……学习的每一个方面或智能的任何其他特征原则上都可以被精确地描述，以至于可以制造一台机器来模拟它。”

至关重要的是，正是约翰·麦卡锡为该提案创造了“人工智能”（Artificial Intelligence）这个术语。他特意选择了这个新的、有点挑衅性的名称，以将这项事业与现有的相关领域如控制论、运筹学和自动机理论区分开来，并明确表明该项目雄心勃勃地专注于复制更高级别的认知功能。提案概述了几个关键的研究领域，包括自动计算机、用于AI的编程语言、神经网络、可计算性理论、抽象、随机性、创造力和语言处理。

研讨会召集了十位研究人员，大约持续了六到八周（出席情况有所不同）。除了四位组织者，主要参与者包括来自麻省理工学院的特伦查德·莫尔（Trenchard More）和奥利弗·塞尔弗里奇（Oliver Selfridge）、来自IBM的亚瑟·塞缪尔、雷·索洛蒙诺夫（Ray Solomonoff，专注于归纳推理的独立研究员），以及也许最重要的是来自卡内基理工学院（现卡内基梅隆大学）的艾伦·纽厄尔（Allen Newell）和赫伯特·A·西蒙（Herbert A. Simon）。

虽然研讨会本身并未产生单一的、统一的突破，也没有形成传统意义上的正式会议记录（它更像是一次长时间的头脑风暴和思想交流），但其历史意义是巨大的。它发挥了几个关键作用：

1. **命名与身份认同：** 它正式将该领域命名为“人工智能”，赋予了它一个名称和一个独特的身份。
2. **社群形成：** 它汇集了将在未来二十年成为AI研究的奠基人和领导者的人物，促进了合作和共同目标感。
3. **定义愿景：** 它阐述了该领域的宏伟、雄心勃勃的目标——模拟智能的所有方面——为未来的研究设定了高标准。

4. **展示早期成功：** 关键的是，研讨会成为了展示早期、引人注目的AI实践例子的场所。纽厄尔和西蒙展示了他们的逻辑理论家（Logic Theorist）程序，该程序能够使用启发式搜索技术独立证明怀特海和罗素的《数学原理》（*Principia Mathematica*）中的定理。这是一项里程碑式的成就，表明计算机可以执行先前被认为需要人类独创性和逻辑推理的任务。亚瑟·塞缪尔也演示了他的学习型跳棋程序，展示了机器通过经验改进的潜力。

达特茅斯研讨会，在这些早期成功所产生的乐观情绪及其参与者共同愿景的推动下，有效地将AI作为一个正式的研究项目启动了。它标志着从分散的个人努力和基础思想到一个拥有自己名称、核心研究群体和一系列具有挑战性的长期目标的公认科学学科的转变。AI时代正式开始了。

基础范式：逻辑、符号和搜索

从达特茅斯研讨会产生并在此后四分之一个世纪里主导该领域的人工智能方法，通常被称为符号AI（Symbolic AI），或有时回顾性地称为“老式人工智能”（Good Old-Fashioned AI, GOFAI）。这一范式建立在艾伦·纽厄尔和赫伯特·A·西蒙最有力阐述的一个中心假设之上：物理符号系统假设（Physical Symbol System Hypothesis）。该假设断言，“一个物理符号系统拥有进行一般智能行动的必要和充分手段。” 本质上，它提出智能源于根据一组形式化规则（逻辑）对符号（代表对象、概念或思想的物理模式）的操作。在这种观点下，思考从根本上说是在符号表示上进行的计算过程。

这种观点自然导致将形式逻辑作为表示知识和进行推理的主要工具。如果智能涉及根据规则操纵符号，那么完善的数理逻辑系统似乎就是理想的框架。程序被设计用来使用谓词演算等逻辑形式主义来表示问题和知识，并使用逻辑推理规则（如*modus ponens*，肯定前件式）来推导出新的事实或解决问题。

纽厄尔和西蒙在达特茅斯展示的逻辑理论家（1956年）就是这种方法的典型范例。它以符号方式表示数学定理和公理，并尝试通过应用逻辑规则来寻找证明。一个关键挑战很快变得明显：可能规则应用的数量之多，创造了一个巨大的潜在推理步骤“搜索空间”。对于除了最简单的问题之外的所有问题，详尽地探索这个空间在计算上是不可行的——这种现象被称为“组合爆炸”（combinatorial explosion）。

因此，早期AI研究的一个主要焦点是开发高效的搜索（Search）算法。从定理证明、游戏到路径规划等问题，都被框定为在状态空间（代表可能情况或配置）中搜索，以找到从初始状态到期望目标状态的路径。由于详尽搜索通常是不可能的，研究人员开发了启发式搜索（Heuristic Search）技术。启发式是“经验法则”或有根据的猜测，有助于引导搜索走向有希望的路径，并剪除搜索空间中没有希望的分支，从而使复杂问题变得易于处理。逻辑理论家就采用了启发式方法来选择可能有用的公理和推理步骤。

在逻辑理论家工作的基础上，纽厄尔和西蒙于1959年开发了通用问题求解器（General Problem Solver, GPS）程序。GPS试图创建一个更通用的问题解决架构，不局限于像逻辑这样的特定领域。它采用了一种称为“手段-目的分析”（means-ends analysis）的启发式技术。这涉及到将当前状态与目标状态进行比较，识别差异，并选择被认为能减少这些差异的操作符（行动或规则）。GPS可以解决各种形式化的问题，如逻辑谜题和符号积分任务，进一步巩固了通用智能行为可以通过符号处理和启发式搜索实现的思想。

为了促进这些符号操作程序的开发，需要专门的编程语言。虽然早期程序通常用较低级语言编写，但约翰·麦卡锡认识到需要一种为符号计算优化的语言。1958年，他开发了Lisp（List Processing，表处理）。Lisp的核心数据结构是列表，事实证明它在表示复杂的符号信息方面极其灵活，包括逻辑表达式、规则，甚至其他Lisp程序（从而实现了强大的元编程能力）。凭借递归和自动内存管理（垃圾回收）等特性，Lisp成为了AI研究的主导编程语言，尤其是在美国，长达数十年。

符号范式在20世纪70年代和80年代初专家系统（Expert Systems）的发展中达到了其实际应用的高峰。这些系统旨在捕获狭窄、定义明确领域中人类专家的专业知识，并使其可供咨询。通常，一个专家系统由两个主要部分组成：

1. **知识库（Knowledge Base）：** 特定领域的事实集合，更重要的是，启发式规则，通常以IF-THEN语句的形式表达（例如，“IF 病人发烧 AND 颈部僵硬, THEN 怀疑脑膜炎”）。这些知识是通过 painstakingly 从人类专家那里获取的。
2. **推理引擎（Inference Engine）：** 一个通用的推理机制，将知识库中的规则应用于特定案例的事实（由用户提供），以得出结论或建议。

著名的早期专家系统包括DENDRAL（分析质谱数据以识别化学结构）、MYCIN（诊断传染性血液病并推荐抗生素治疗，达到了与人类专家相当的水平）和PROSPECTOR（协助地质学家进行矿产勘探）。专家系统代表了AI的第一个重要的商业成功案例，证明了符号方法可以在专业应用中提供切实的价值。它们证明了在受限领域内编码显式知识和使用逻辑推理进行复杂决策的力量。

这个由符号范式主导的早期阶段，确立了逻辑、知识表示和搜索作为AI研究的基础支柱。它证明了创造能够执行需要推理和问题解决任务的机器的可行性，为未来的进步奠定了基础，同时也揭示了复制人类智能的全部广度和深度所涉及的深刻挑战。

第四章：硅之心：硬件的指数级飞跃

可计算性的理论框架和最初的架构蓝图，结合人工智能的初步构想，描绘了一幅引人入胜的未来图景。然而，这一愿景只能通过切实的物理硬件来实现。早期的电子计算机，由数千个耗电、不可靠且笨重的真空管构成，是工程上的奇迹，但它们也昂贵、易于故障，并且其最终计算能力有限。要真正释放计算的潜力，并为AI研究人员所设想的复杂性铺平道路，底层电子元件的根本性革命是必需的。这场革命随着固态电子

学的出现而到来，开启了一个持续微型化和计算能力指数级增长的时代，从根本上重塑了技术和社会。计算机的“硅之心”开始以每年更快、更强、更小的节奏跳动。

晶体管与集成电路：微型化的开端

第一代电子计算机严重依赖真空管。虽然比机电继电器快得多，但真空管存在显著的缺点。它们本质上是改良的灯泡——易碎的玻璃管内含需要加热的灯丝，消耗大量电力并产生大量热量。这种热量需要复杂的冷却系统，而且真空管固有的易碎性和最终烧毁导致了频繁的维护问题和有限的可靠性。此外，它们的物理尺寸限制了元件的密度，从而限制了能够实际建造和容纳的计算机的复杂性。一个典型的真空管可能有几英寸长；建造一台像ENIAC那样拥有数万个真空管的机器，结果就是一个房间大小的庞然大物。

使真空管在大多数计算应用中过时的突破发生在1947年的贝尔电话实验室。在相对较新的固态物理领域工作，科学家约翰·巴丁（John Bardeen）、沃尔特·布拉顿（Walter Brattain）和威廉·肖克利（William Shockley）发明了晶体管（transistor）。这种微小的器件，最初由锗制成，后来主要由硅制成，可以执行与真空管相同的基本功能——放大电信号和充当电子开关（打开或关闭电流以表示二进制的1和0）——但没有其固有的缺点。晶体管是：

- **固态的（Solid-State）：** 由半导体材料制成，没有需要加热或烧毁的灯丝，没有需要破坏的真空密封，也没有易碎的玻璃外壳。
- **更小的（Smaller）：** 比真空管小几个数量级。
- **低功耗（Lower Power）：** 消耗的电力显著减少。
- **更冷的（Cooler）：** 产生的热量少得多。
- **更快的（Faster）：** 可以更快地切换状态。
- **更可靠的（More Reliable）：** 具有更长的使用寿命。

晶体管的发明——巴丁、布拉顿和肖克利因此获得了1956年的诺贝尔物理学奖——在20世纪50年代末和60年代初开启了计算机的“第二代”。在电路板上用分立晶体管取代真空管，使得机器体积大大缩小、速度更快、建造成本和运营成本更低、更可靠、更节能。这使得计算机对更广泛的企业、大学和政府机构来说更加实用和易于获取。晶体管计算机的著名例子包括广受欢迎的IBM 1401商用计算机和像数字设备公司（Digital Equipment Corporation, DEC）的PDP-1这样的早期小型计算机。

虽然晶体管解决了真空管的问题，但构建复杂电路仍然需要 painstakingly 地将数千个单独的晶体管、电阻器、电容器和其他分立元件焊接到印刷电路板（PCB）上。这个过程劳动密集、成本高昂，并且在焊点处产生了许多潜在的故障点。微型化的下一次飞跃正是解决了这个挑战。

在20世纪50年代末，德州仪器（Texas Instruments）的杰克·基尔比（Jack Kilby）和仙童半导体（Fairchild Semiconductor）的罗伯特·诺伊斯（Robert Noyce）独立开发了集成电路（Integrated Circuit, IC），通常称为微芯片（microchip）。其核心思想是革命性的：与其将分立元件连接在一起，为什么不在一块单一的、整体的半导体材料（通常是硅）上同时制造多个元件及其互连呢？基尔比在1958年使用锗展示了一个初步的“整体思想”，而诺伊斯不久后开发了一个更实用的基于硅的版本，采用了平面处理工艺和改进的互连方法。

集成电路允许将数十个、然后是数百个、数千个，最终是数十亿个微观晶体管和其他元件蚀刻到一个微小的硅片上。这提供了巨大的优势：

- **极端微型化（Extreme Miniaturization）：** 将更多元件封装到更小的空间。
- **降低成本（Reduced Cost）：** 大规模生产技术使得制造复杂电路比组装分立元件便宜得多。
- **提高可靠性（Increased Reliability）：** 消除了无数焊点，显著减少了潜在故障点。
- **提高速度（Increased Speed）：** 电子在芯片上元件之间传输的距离大大缩短，从而实现更快的操作。
- **降低功耗（Lower Power Consumption）：** 更小的元件通常消耗更少的功率。

IC的出现标志着从20世纪60年代中期开始的计算机“第三代”的开端。IBM具有影响力的System/360系列，虽然主要使用混合电路（固态逻辑技术，将微小的分立元件安装到陶瓷基板上），但代表了行业向集成化的转变。真正的单片IC推动了像DEC PDP-8（第一台商业上成功的小型计算机）和PDP-11这样强大的小型计算机的兴起，它们以大型机成本的一小部分，为实验室和小型组织带来了显著的计算能力。IC不仅从根本上改变了计算，也改变了整个电子领域，为便携式计算器、数字手表以及定义了20世纪下半叶的无数其他设备铺平了道路。硅芯片成为了现代数字世界无处不在的构建模块。

微处理器与摩尔定律：芯片上的计算

集成电路技术驱动的持续进步逻辑上导向了下一个主要里程碑：将计算机中央处理单元（CPU）的所有核心功能——算术逻辑单元、控制单元和寄存器——集成到单个硅芯片上。这种“芯片上的计算机”就是微处理器（microprocessor）。

虽然存在早期的实验性工作，但第一款商业上可用的单芯片微处理器被广泛认为是英特尔（Intel）于1971年11月发布的4004。最初是为一家日本计算器公司（Busicom）设计的，这款4位4004包含2300个晶体管，并证明了将完整的处理单元置于一块硅片上的可行性。英特尔迅速跟进，推出了8位处理器，包括8008（1972年）以及更重要的8080（1974年），后者成为了点燃个人计算机革命的Altair 8800套件计算机的大

脑。其他公司如德州仪器（其TMS 1000，可以说是第一款将RAM和ROM集成在芯片上的微控制器）和加勒特艾雷塞奇（Garrett AiResearch，其用于F-14战斗机的中央空气数据计算机或CADC）也在同一时期生产了早期的微处理器。

微处理器是革命性的。它极大地减小了构建功能性计算机所需的物理尺寸、成本和复杂性。突然之间，计算智能可以被嵌入到传统计算机之外的广泛设备中。处理器复杂性迅速增长，从4位设计发展到8位，然后是16位（如1978年的英特尔8086，它开创了在个人电脑中主导数十年的x86架构），以及32位处理器（如1980年贝尔实验室的BELLMAC-32和用于早期苹果Macintosh计算机的摩托罗拉68000）。这项发明标志着计算机“第四代”的开始，并且至关重要地，它普及了计算能力的使用权，为个人计算机、工作站，并最终为智能手机和嵌入式系统奠定了基础。

支撑集成电路复杂性惊人快速进步的是戈登·摩尔（Gordon Moore）——仙童半导体及后来的英特尔联合创始人——在1965年做出的一项观察。在为*电子*杂志撰写的一篇文章中，摩尔指出，自集成电路发明以来，可以经济地放置在集成电路上的元件（主要是晶体管）数量大约每年翻一番。他最初预测这一趋势将至少再持续十年。后来，他将翻倍周期修正为大约每两年一次，现在通常被引述为每18到24个月翻一番。

这个经验性观察，现在普遍称为摩尔定律（Moore's Law），其意义远不止于一个预测。它演变成了整个半导体行业的指导原则和自我实现的预言。芯片制造商根据维持这种指数级改进速度来设定他们的研发目标。摩尔定律描述了一个良性循环：随着更多晶体管可以被封装到芯片上，处理器变得更强大，内存容量增加，而每个晶体管的成本骤降。这种性能的持续提升和成本的持续下降，推动了整个计算领域的创新。它使得开发日益复杂的软件、图形用户界面、多媒体应用、互联网成为可能，并且至关重要地，为现代人工智能，特别是深度学习所需的计算密集型算法提供了支持。半个多世纪以来，摩尔定律驱动了定义数字时代的指数级增长，将房间大小的计算器变成了口袋大小的超级计算机。

内存与存储的演进

没有同样强大的手段来存储其需要执行的指令和需要处理的数据，快速的处理器是无用的。计算机内存（用于活动程序和数据的快速、易失性存储，通常称为RAM - 随机存取存储器）和存储（用于长期持久化的较慢、非易失性存储）技术的演进与处理能力的进步并行，通常由相同的底层半导体进展驱动。

早期计算机采用了各种巧妙但往往笨重的内存技术。除了使用真空管或继电器本身作为基本的内存元件外，还开发了专门的技术：

- **声延迟线（Acoustic Delay Lines）：** 用于像UNIVAC I和EDSAC这样的机器。数据位被存储为通过介质（通常是充满水银的管子或磁致伸缩线）传播的声波序列。波在末端被检测并重新循环。访问是串行的（比特必须等待轮到它们出现），并且设备对温度和振动敏感。

- **威廉姆斯-基尔伯恩管（Williams-Kilburn Tubes）：** 用于曼彻斯特宝贝和IBM 701。这些是改进的阴极射线管（CRT），将比特存储为屏幕荧光涂层上微小的电荷点。与延迟线相比，它们提供了更快的随机访问，但它们是易失性的（数据会迅速消失，需要不断刷新）并且密度相对较低。
- **磁鼓存储器（Magnetic Drum Memory）：** 在广受欢迎的IBM 650中显著使用。一个涂有磁性材料的旋转圆柱体将数据存储为磁化点。沿着鼓定位的读写头在数据旋转到其下方时访问数据。它是非易失性的，比电子存储器便宜，但由于机械旋转而慢得多。

一个显著的改进出现在20世纪50年代中期，即磁芯存储器（magnetic core memory），它主导了主内存设计近二十年。磁芯存储器由微小的、甜甜圈形状的铁磁材料环（磁芯）组成，串在网格状的电线上。每个磁芯可以在两个方向之一（顺时针或逆时针）被磁化，以表示二进制的0或1。穿过磁芯的电线用于“写入”（设置磁化方向）和“读取”（检测方向，不幸的是这需要重写数据——一种“破坏性读取”）。磁芯存储器提供了随机访问（任何磁芯都可以直接访问）并且是非易失性的（断电后仍能保留数据），使其在当时是健壮和可靠的。然而，制造过程涉及到将电线精细地手工编织穿过磁芯，使其成本高昂，并且其速度与后来的电子方法相比有限。

主内存的真正革命伴随着半导体存储器的出现，它利用了改变处理器的相同集成电路技术。在20世纪60年代末商业上可行并在70年代中期占据主导地位的半导体RAM，使用构建在硅芯片上的晶体管制成的微小电子电路（触发器或电容器）来存储比特。出现了两种主要类型：

- **静态RAM（SRAM）：** 使用触发器电路（通常每个比特6个晶体管）。它速度非常快，不需要刷新，但密度较低且更昂贵。常用于高速缓存。
- **动态RAM（DRAM）：** 每个比特使用单个晶体管和电容器来存储电荷。它比SRAM密度高得多且更便宜，但需要周期性刷新以防止电荷泄漏。成为了主计算机内存的标准。

在摩尔定律的推动下，半导体RAM与磁芯存储器相比，提供了大大提高的速度、更高的密度以及迅速下降的每比特成本，尽管它是易失性的（断电时丢失数据）。这种性能和经济性的结合使其成为几乎所有现代计算机主内存无可争议的选择。

同样重要的是二级存储（secondary storage）——用于永久存储大量数据的技术——的演进。早期系统依赖于速度极慢且容量极低的介质，如穿孔卡片和纸带。磁带在容量和顺序速度方面提供了显著改进，但不适合随机访问数据。突破在于磁性磁盘驱动器的发展：

- **硬盘驱动器（HDDs）：** 由IBM在20世纪50年代推出（RAMAC 305系统），HDD将数据存储在涂有磁性材料的旋转盘片上，由可移动的读写头访问。提供随机访

问的HDD成为了操作系统、应用程序和用户数据数十年的主要存储介质。容量和速度随着时间的推移急剧增加，而物理尺寸则缩小了。

- **软盘（Floppy Disks）：** 在20世纪70年代推出，这些在装在保护套中的柔性磁盘上提供了便携式磁存储，对于早期个人计算机时代的软件分发和数据传输至关重要。

后来，像CD-ROM、DVD和蓝光光盘这样的光盘提供了高容量，特别适合软件分发和多媒体内容，尽管写入速度较慢。

最近，固态硬盘（SSDs）已开始在许多应用中取代HDD，特别是在笔记本电脑和高性能系统中。基于闪存（一种非易失性半导体存储器，也用于USB驱动器和存储卡），SSD没有移动部件。与传统HDD相比，这导致了显著更快的数据访问速度、更高的耐用性、更低的功耗和静音操作，尽管历史上每GB成本更高。

处理能力、主内存和二级存储的并行演进至关重要。更快的CPU需要更大、更快的RAM来高效地为其提供指令和数据，而更大的程序和数据集则需要更高容量、更快的持久存储。硅技术的持续进步，包含了晶体管、集成电路、微处理器以及摩尔定律所描述的指数级扩展，提供了强大且日益廉价的硬件基础——硅之心——整个数字革命，包括现代人工智能的复杂需求，都建立在其之上。

第五章：软件、网络与个人革命

硅之心所带来的持续微型化和指数级性能提升，提供了原始动力，即数字时代赖以建立的物理基底。但仅有硬件是惰性的潜力。它需要指令、组织和连接才能变得真正有用。因此，计算崛起的历史与软件——赋予硅生命力的无形逻辑——的演进，以及连接这些日益强大的机器的网络的发展，密不可分，最终 culminate 于一场个人计算革命，将这种能力直接交到全球个人手中。本章探讨了用于指挥机器的语言如何演变，操作系统如何驯服其复杂性，计算机如何从房间大小的庞然大物缩小到桌面设备，以及一个全球网络如何涌现以连接它们，从而改变了社会。

机器的语言：从机器码到高级语言

对最早的电子计算机进行编程是一项神秘且极其困难的任务。在最基础的层面上，处理器只理解机器码（machine code）——代表特定操作（如将两个数相加、移动数据或跳转到不同指令）和内存地址的原始二进制数字（1和0）序列。直接用机器码编写程序极其繁琐、容易出错，并且需要对计算机特定硬件架构有深入的了解。一个比特放错位置就可能导致整个程序以不可预测的方式失败。

向可用性迈出的一小步是汇编语言（assembly language）。汇编语言为机器指令提供了助记符（简短、人类可读的缩写，如 `ADD`、`MOV`、`JMP`），并允许程序员使用符号名称来表示内存位置，而不是原始的二进制地址。然后，汇编器（assembler）程序

会将这些助记符翻译回相应的机器码。虽然仍然与硬件紧密相连，需要详细的架构知识，但与原始二进制相比，汇编语言使编程稍微快一些，并且不易出现低级错误。

然而，使编程易于访问和高效的真正突破是高级编程语言（high-level programming languages）的发展。这些语言允许程序员使用更抽象、人类可理解的结构来表达指令，通常类似于数学符号或自然语言元素，而不是关注寄存器和内存地址的低级细节。这种抽象将程序员从特定硬件的复杂性中解放出来，使他们能够专注于他们试图解决的问题的逻辑。虽然康拉德·楚泽在20世纪40年代中期构思了一种复杂的高级语言Plankalkül，但由于时代背景，它在很大程度上仍停留在理论层面。

第一个广泛成功的高级语言是FORTRAN（Formula Translation，公式翻译），由约翰·巴科斯（John Backus）领导的IBM团队在1954年至1957年间开发。FORTRAN主要为科学和工程计算设计，允许程序员直接编写代数公式（例如，`Y = A*X**2 + B*X + C`），FORTRAN编译器会将其翻译成高效的机器码。它在技术领域非常受欢迎，至今仍在一些遗留系统和高性能计算领域使用。

认识到商业世界的独特需求，即数据处理（处理记录、文件、财务计算）至关重要，而非复杂的数学运算，COBOL（Common Business-Oriented Language，通用面向商业的语言）由一个深受格蕾丝·霍珀（Grace Hopper）影响的委员会从1959年开始开发。COBOL旨在实现类似英语的可读性和自文档化，使用冗长的语法。它成为大型机上商业数据处理的主导语言长达数十年，处理了数万亿美元的交易。

对于新兴的人工智能领域，操纵符号和复杂数据结构至关重要，约翰·麦卡锡于1958年在麻省理工学院创造了Lisp（List Processing，表处理）。Lisp的基本数据结构——列表，被证明非常适合表示符号信息、逻辑表达式乃至代码本身。Lisp的优雅、灵活性以及递归等特性使其成为AI研究多年来的首选语言。

其他有影响力的早期语言也相继出现。ALGOL（Algorithmic Language，算法语言），由欧美计算机科学家在20世纪50年代末和60年代合作开发，引入了基本概念，如块结构（用`begin`和`end`分组语句）、词法作用域和形式化语言定义。虽然商业上不如FORTRAN或COBOL成功，但ALGOL深刻影响了许多后续语言的设计，包括Pascal和C。

为了让学生和初学者更容易接触计算，达特茅斯学院的约翰·凯梅尼（John Kemeny）和托马斯·库尔茨（Thomas Kurtz）于1964年创建了BASIC（Beginner's All-purpose Symbolic Instruction Code，初学者通用符号指令代码）。BASIC专为易学和在达特茅斯分时系统内交互使用而设计，成为20世纪70年代和80年代许多早期个人计算机附带的默认语言，向数百万人介绍了编程。

也许有史以来最具影响力的语言之一是C语言，由丹尼斯·里奇（Dennis Ritchie）于1972年左右在贝尔实验室开发，与肯·汤普森（Ken Thompson）在Unix操作系统上的工作并行进行。C语言在高级抽象和低级控制之间取得了平衡，允许高效地操作硬件资源，同时提供结构化编程特性。它与Unix的紧密联系及其相对的可移植性使其成

为系统编程（编写操作系统、编译器、实用程序）和应用程序开发数十年的主导语言，其语法深刻影响了C++、Java、C#以及许多其他现代语言。

对高级语言的成功至关重要的是开发了用于将它们翻译成可执行机器码的复杂软件工具：编译器（compilers）和解释器（interpreters）。编译器在执行前将整个高级程序（源代码）翻译成机器码（目标代码）作为一个单独的步骤。相比之下，解释器逐行或逐条语句地翻译并执行程序。编译器通常生成运行更快的程序，而解释器在开发过程中提供了更大的灵活性和调试便利性。这些翻译器的创建自动化了弥合人类可读指令与机器二进制语言之间鸿沟的复杂任务。

从神秘的机器码到表达力强的高级语言的演变，代表了计算领域的一次深刻转变。它极大地提高了程序员的生产力，减少了出错的可能性，使得创建远为复杂的软件系统成为可能，并在不同硬件平台之间促进了一定程度的代码可移植性。语言的多样化反映了计算机在科学、商业、人工智能、系统开发和教育等领域应用的不断扩大。

操作系统：管理复杂性

随着计算机变得越来越强大，能够运行更大、更复杂的程序，管理底层硬件资源——CPU时间、内存、输入/输出设备——变得日益具有挑战性。早期计算机通常根本没有操作系统（OS）；程序员直接与硬件交互，通过开关或纸带手动加载程序。后来，出现了基本的监控程序来处理基本的输入/输出和程序加载。

第一个真正的操作系统出现在20世纪50年代和60年代，用于管理昂贵的大型计算机上的批处理（batch processing）。为了最大限度地提高利用率，作业（程序及其数据，通常以打孔卡片组的形式提交）被分组批处理。操作系统会自动一个接一个地加载作业，控制打印机和磁带驱动器，编译程序，并管理作业核算，从而最大限度地减少昂贵硬件闲置的时间。例子包括通用汽车公司和北美航空公司为IBM机器开发的GM-NAA I/O，以及IBM自己的IBSYS和后来的综合性OS/360系列。

一个重大的飞跃是20世纪60年代分时系统（time-sharing systems）的发展。分时系统允许多个用户通过各自的终端同时与一台强大的计算机进行交互，而不是按顺序处理作业。操作系统快速地在用户之间切换CPU的注意力，给每个用户一小部分处理时间片。这创造了每个用户都拥有专用机器访问权限的错觉，从而实现了交互式编程、调试和程序使用。这需要复杂的操作系统能力来管理CPU调度、内存分配（保护用户程序互不干扰）以及处理并发的I/O请求。开创性的分时系统包括麻省理工学院开发的兼容分时系统（Compatible Time-Sharing System, CTSS），以及极具雄心的Multics（Multiplexed Information and Computing Service，多路复用信息与计算服务）项目，这是麻省理工学院、贝尔实验室和通用电气公司的合作项目。虽然Multics本身的商业成功有限，但它具有巨大的影响力，开创了许多现代操作系统中仍在使用的概念。

肯·汤普森和丹尼斯·里奇以及贝尔实验室的其他人，对Multics的复杂性感到沮丧，但受到其思想的启发，大约在1969年开始开发一个更简单、更优雅的操作系统，最初是为一台未被充分利用的DEC PDP-7小型计算机设计的。这个系统成为了Unix。Unix体现了几种强大的设计哲学：一个层级文件系统，其中所有东西（包括设备）都表示为文件；使用小型的、单一用途的实用程序；以及“管道”（pipes）的概念，用于将这些实用程序链接起来以执行复杂的任务。至关重要的是，Unix在C语言创建后不久就主要用C语言重写，这使得它相对容易移植（改编）到不同的硬件平台。这种可移植性，加上其强大的功能和优雅的设计，使得Unix在学术界、研究实验室以及最终在商业世界中变得极具影响力。它的后代和变种（如Linux、macOS、iOS、Android）构成了当今绝大多数计算设备的基础。

随着计算机在尺寸和成本上的缩小，操作系统也为微型计算机发展起来。像数字研究公司（Digital Research）的CP/M（Control Program for Microcomputers，微型计算机控制程序）这样的早期系统主导了IBM PC之前的市场。随着1981年IBM PC的推出，微软的MS-DOS（Microsoft Disk Operating System，微软磁盘操作系统）——从西雅图计算机产品公司（Seattle Computer Products）的QDOS收购并改编而来——成为IBM兼容机的标准，主要提供命令行界面（CLI），用户通过键入文本命令进行操作。苹果公司为其Apple II系列开发了自己的Apple DOS和后来的ProDOS。

用户交互方式的一次革命性转变是图形用户界面（Graphical User Interface, GUI）的出现。由施乐公司（Xerox）的帕洛阿尔托研究中心（Palo Alto Research Center, PARC）在20世纪70年代在其试验性的Alto计算机上开创，GUI采用了鼠标指针、窗口、图标、菜单和桌面隐喻，使得交互比键入命令直观和可视化得多。虽然施乐未能有效地将这些创新商业化，但苹果电脑公司（Apple Computer）采纳了它们，首先是在昂贵的Lisa计算机（1983年）上，然后，最成功的是在开创性的Macintosh（1984年）上。Macintosh以其集成的GUI和鼠标，极大地降低了普通用户的入门门槛。微软最终以其Windows操作系统作为回应，最初运行在MS-DOS之上，后来演变成一个独立的图形操作系统，最终主导了IBM兼容PC市场。

因此，操作系统从简单的监控程序演变成了复杂的软件套件，它们管理硬件资源，为应用程序提供基本服务（如文件管理和网络），并定义了用户与计算机交互的基本方式，无论是通过命令行还是图形界面。它们成为了调解硬件复杂性与软件应用程序及用户需求之间不可或缺的层。

个人计算的兴起

20世纪70年代初微处理器的发明是点燃个人计算机（Personal Computer, PC）革命的技术火花。通过将整个CPU封装到单个、价格合理的芯片上，设计和制造足够小、足够便宜，可供个人、学校和小型企业拥有和使用的计算机变得可行。

这场革命并非始于企业实验室，而是始于业余爱好者和电子发烧友的车库和地下室。分水岭时刻通常被认为是1975年1月，《大众电子》（*Popular Electronics*）杂志封面

刊登了Altair 8800。由阿尔伯克基的一家小公司MITS以套件形式销售，基于英特尔8080微处理器的Altair，本质上是一个带有开关和指示灯的盒子——它没有键盘，没有屏幕，最初几乎没有软件。然而，它抓住了成千上万看到个人计算潜力的爱好者的想象力。比尔·盖茨（Bill Gates）和保罗·艾伦（Paul Allen）为Altair编写了一个BASIC解释器，并在此过程中创立了微软（Microsoft）。

虽然Altair吸引了愿意自己动手搭建和编程机器的爱好者，但市场在1977年随着有时被称为个人计算"三巨头"的推出而转向主流消费者：

- **Apple II：** 主要由史蒂夫·沃兹尼亚克（Steve Wozniak）设计，并由史蒂夫·乔布斯（Steve Jobs）出色地营销，Apple II是一款面向家庭和学校的预装计算机。它具有彩色图形、声音功能、易于扩展，并内置了BASIC。其用户友好的设计和不断增长的软件库使其取得了巨大成功。
- **Commodore PET（个人电子处理机）：** 一体化设计，带有集成的单色显示器和磁带驱动器，在教育市场很受欢迎。
- **Tandy Radio Shack TRS-80：** 被亲切地昵称为"Trash-80"，通过Radio Shack商店销售，使其广泛可及，尽管最初与Apple II相比功能有限。

这些机器，通常提供4KB到48KB的RAM，销量达数百万台，将计算带出了大型组织的专属领域，进入了日常环境。它们的成功极大地得益于"杀手级应用"（killer applications）的出现——那些极具吸引力以至于人们会为了运行它们而购买硬件的软件程序。典型的例子是VisiCalc，第一个电子表格程序，最初于1979年为Apple II发布。VisiCalc将PC从爱好者玩具或教育工具转变为强大的商业生产力机器，展示了其在财务建模、预算编制和预测方面的价值。

随着IBM于1981年8月进入市场，个人计算机市场获得了显著的合法性并进入了商业主流。IBM个人计算机（IBM PC），采用英特尔8088处理器和微软的MS-DOS操作系统，特意采用相对开放的架构（使用现成的组件，尽管BIOS芯片是专有的）。这一策略鼓励了第三方公司为该平台开发扩展卡、外围设备和软件。它也使得其他制造商（如康柏（Compaq）、戴尔（Dell）和惠普（HP））能够逆向工程BIOS并创建能够运行相同软件（主要是MS-DOS和后来的Windows）但通常价格更低或功能增强的"IBM兼容机"或"克隆机"。这促进了激烈的竞争，压低了成本，并导致MS-DOS/Windows平台最终主导了个人计算机市场。

1984年苹果Macintosh的推出，以其内置的图形用户界面和鼠标，标志着可用性的又一重大进步，使得被命令行界面吓倒的更广泛受众能够接触计算机。虽然Mac最初在市场上与IBM兼容机相比份额较小，但其对界面设计的影响是深远的。

计算也摆脱了桌面束缚的限制。早期对便携性的尝试产生了像Osborne 1（1981年）这样笨重的"便携式"机器，它将计算机、小型CRT屏幕和软盘驱动器装入一个手提箱大小的外壳中。第一批被称为"笔记本电脑"（laptops）的机器，具有现在熟悉的铰链

式翻盖屏幕设计，大约在1983年出现，型号如Gavilan SC和Tandy Model 100。虽然最初受电池寿命、屏幕质量（通常是单色LCD）和存储容量的限制，便携式计算迅速改进，最终演变成今天常见的强大的笔记本电脑、笔记本和（tablets）平板电脑。

个人计算机革命，由微处理器推动，由创业精神驱动，由操作系统和杀手级应用等基本软件赋能，并通过GUI等创新变得易于访问，从根本上改变了人类与计算的关系。它广泛地分配了处理能力，赋予了个人力量，并改变了无数行业。

互联网：连接全人类

与个人计算机的兴起并行的是另一项深刻的技术发展：创建连接它们的网络，最终形成了全球性的互联网（Internet）。互联计算的需求最初源于军事和研究领域对弹性通信和资源共享的需求。

互联网的起源在于ARPANET项目，该项目于20世纪60年代末由美国国防部高级研究计划局（ARPA，后来的DARPA）发起。ARPANET旨在创建一个能够承受部分中断的分散式通信网络（这是冷战紧张局势加剧的一个担忧）。它开创了分组交换（packet switching）的使用，这是一种将数据分解成称为数据包（packets）的、带有地址的小块的技术。这些数据包可以独立地通过网络中的各种路径路由，并在目的地重新组装。这与传统电话网络中使用的电路交换（circuit-switching）方法根本不同，后者在通话期间建立专用连接。分组交换被证明对于数据通信非常高效和健壮。

虽然ARPANET允许互联的研究计算机进行通信，但将不同类型的网络连接在一起构成了重大挑战。关键突破是20世纪70年代初TCP/IP协议套件（传输控制协议/互联网协议，Transmission Control Protocol/Internet Protocol）的开发，主要由文特·瑟夫（Vint Cerf）和鲍勃·卡恩（Bob Kahn）完成。TCP/IP提供了一套通用规则，定义了数据如何在各种互联网络（“互联网络”或“互联网”）之间进行打包、寻址、传输、路由和接收。TCP处理可靠的数据传输（确保数据包正确、按序到达），而IP处理数据包在网络中的寻址和路由。TCP/IP于1983年被采纳为ARPANET的标准，成为了现代互联网的技术基础，使全球不同的网络能够无缝通信。早期的应用利用了这些协议，包括电子邮件（email，迅速成为网络的“杀手级应用”）、远程登录（Telnet）和文件传输（FTP）。

网络逐渐扩展到其军事和计算机科学研究起源之外。在20世纪80年代中期，美国国家科学基金会（National Science Foundation, NSF）创建了NSFNET，一个连接美国各地超级计算中心和大学网络的高速骨干网。NSFNET采用了TCP/IP，并最终取代ARPANET成为主要的骨干网，显著增加了学术和研究界的网络容量和接入。对商业用途的限制逐渐解除，商业互联网服务提供商（ISPs）在20世纪80年代末和90年代初开始出现，为企业并最终为公众提供拨号接入服务。

尽管有了这些发展，互联网在很大程度上仍然是熟悉基于文本的命令和复杂寻址方案的技术用户的领域。公众使用量的爆炸式增长是由万维网（World Wide Web）的发

明点燃的，该发明由在瑞士欧洲核子研究组织（CERN）工作的英国物理学家蒂姆·伯纳斯-李（Tim Berners-Lee）在1989年至1991年间完成。伯纳斯-李开发了使浏览互联网信息资源变得直观和易于访问的关键组件：

- **HTML（超文本标记语言，Hypertext Markup Language）：** 一种用于创建包含文本、图像以及至关重要的、可以指向网络上任何其他文档的超链接的文档（“网页”）的简单语言。
- **URL（统一资源定位符，Uniform Resource Locator）：** 一种用于在互联网上定位资源（如网页或文件）的标准化方式。
- **HTTP（超文本传输协议，Hypertext Transfer Protocol）：** 一种定义客户端（浏览器）如何发出Web资源请求以及服务器如何满足这些请求的协议。

伯纳斯-李还创建了第一个Web浏览器（名为WorldWideWeb，后更名为Nexus）和第一个Web服务器。Web轻松链接和显示多媒体信息的能力，将互联网从文件存储库转变为一个巨大的、相互连接的知识网络。

美国国家超级计算应用中心（NCSA）于1993年发布的Mosaic网页浏览器是一个关键的催化剂。Mosaic是第一个将图像与文本内联显示（而不是在单独窗口中）的浏览器，并被移植到包括Windows和Macintosh在内的多个平台。其用户友好的图形界面使得浏览Web变得轻松且视觉上吸引人，导致Web服务器和用户数量呈指数级增长。随后浏览器的商业化，由Netscape Navigator（由许多原始Mosaic团队成员开发）和微软的Internet Explorer引领，引发了激烈的竞争（“浏览器大战”），进一步加速了Web技术的发展，并在20世纪90年代中后期推动了大规模的公众采用。

互联网迅速从一个专业的研究网络演变成一个全球性的、公众可访问的通信和信息平台。它彻底改变了商业、教育、娱乐、政治、社交互动以及现代社会几乎所有其他方面。它连接了数十亿人和设备，为云计算、社交媒体、流媒体服务、电子商务创造了基础设施，并且对AI的发展轨迹至关重要，它提供了训练强大机器学习模型所需的、前所未有的大量数据。

软件语言、操作系统、个人计算机和互联网的相互交织的演进，建立在指数级改进的硬件基础之上。它们共同将计算从专家的专业工具转变为无处不在的、个性化的、互联的力量，重塑了世界，并为接下来由数据驱动的人工智能时代奠定了基础。

第六章：AI的旅程：炒作与进步的周期

旨在赋予机器智能的探索，自达特茅斯研讨会的乐观主义启动以来，并未遵循一条简单、线性的、能力不断增强的轨迹。相反，人工智能的历史以剧烈的周期为特征：狂热的热情、雄心勃勃的预测和重大突破，往往随后是冷静的现实检验、幻灭和投资减少的时期——这种现象通常被称为“AI寒冬”（AI Winters）。这种周期性模式反映了复制人类认知的巨大固有难度、理论理解、可用计算资源、实际成功之间的复杂相互作

用，以及研究资金和公众认知的起伏。理解这些周期对于认识该领域的韧性，以及为其现代成功铺平道路的知识和技术的逐步（虽然常常不均衡）积累至关重要。

机器学习生根发芽：从数据中学习

虽然早期AI的主导范式侧重于符号推理——明确地用逻辑规则和知识表示来编程机器（GOFAI）——但一个并行存在（尽管最初不那么突出）的思路探索了智能也可能源于*从经验或数据中学习*的能力。与其煞费苦心地手工制作每一条规则，或许机器可以自动获取知识或随着时间的推移改进其性能。这个概念，现在是现代AI的核心，属于机器学习（Machine Learning, ML）范畴，其根源可以追溯到该领域最早期的日子。

亚瑟·塞缪尔的跳棋程序，在整个20世纪50年代开发，提供了首批引人注目的演示之一。如前所述，塞缪尔的程序不仅仅是基于固定规则下跳棋；它融入了学习机制。通过记住它遇到过的棋盘位置（死记硬背学习），以及更显著地，通过根据与自身和人类对弈的结果调整其评估函数中分配给不同棋盘特征的权重（参数调整或强化学习），该程序逐渐提高了其技能，最终达到了强大的业余水平。塞缪尔的工作表明，机器可以通过自动化经验显著提高其性能，这是机器学习的核心原则。

与此同时，模式识别（Pattern Recognition）领域的研究人员开始探索能够将数据分类的算法。这涉及到开发能够基于示例学习识别模式的系统——例如，识别手写字符或对图像中的对象进行分类。该领域一个显著的早期发展是弗兰克·罗森布拉特（Frank Rosenblatt）于1957年提出的感知器（Perceptron）。受麦卡洛克-皮茨模型神经元的启发，感知器是一个单层人工神经网络。它可以通过根据其对训练样本的分类是否正确来调整其连接权重（使用简单的学习规则）来学习对输入模式进行分类。罗森布拉特构建了硬件实现，并展示了感知器学习识别简单模式的能力，引发了人们对受大脑启发的学习机器潜力的极大兴奋。

伴随着这些努力，统计方法开始应用于与AI相关的任务。植根于概率论的技术，如贝叶斯推断（根据证据更新信念）和回归分析（建模变量之间的关系），提供了基于不确定或嘈杂数据进行预测和决策的数学严谨方法。虽然与基于逻辑的符号方法不同，但这些统计学习技术为在数据中寻找模式和结构提供了替代工具。

即使在符号AI阵营内部，也出现了从示例中归纳规则或知识的方法，而不仅仅是依赖专家启发。归纳逻辑编程（inductive logic programming）的研究试图寻找从观察中自动生成逻辑规则的方法。一个特别有影响力的研究方向集中在学习决策树（decision trees）——通过询问一系列简单问题来对数据进行分类的层次结构。罗斯·昆兰（Ross Quinlan）借鉴了厄尔·B·亨特（Earl B. Hunt）等人的早期工作，在20世纪70年代末和80年代初开发了ID3算法。ID3可以通过迭代选择根据信息论度量（如信息增益）最能分割数据的属性，来高效地从数据集中构建决策树。决策树被证明对分类任务有效，并提供了产生相对人类可理解规则的优势。

这些早期的分支——塞缪尔的游戏学习、罗森布拉特的感知器、统计模式识别，以及像ID3这样的规则归纳算法——将机器学习确立为AI内部（或有时与之并列）的一个独特子领域。它们共享一个共同主题：超越仅仅编程显式知识，转向能够通过自动处理数据或经验来获取知识或提高性能的系统。虽然符号AI在几十年内保持主导地位，但数据驱动学习的种子在AI旅程的早期就已经播下。

AI寒冬：现实检验与资金削减

AI研究的最初几十年充满了相当大的乐观情绪、像逻辑理论家和塞缪尔跳棋手这样的突破性演示，以及有时来自领先研究人员关于在机器翻译、通用问题解决甚至在一代人内实现人类水平智能等领域即将取得突破的大胆预测。然而，早期的成功往往涉及精心约束的问题或“微观世界”，将这些技术扩展到处理现实世界的复杂性和模糊性被证明比预期的要困难得多。这种期望与现实之间的差距导致了显著的幻灭、批评和研究资金减少的时期——即AI寒冬。

第一个主要的AI寒冬大约发生在20世纪70年代中期到80年代初。几个汇集因素促成了这次低迷：

1. **承诺过高与预测失败：** 早期研究人员沉浸在初步成功的兴奋中，有时对AI的近期潜力做出了过于乐观的声明。备受瞩目的项目，特别是在机器翻译领域（早期系统难以处理语义模糊性和上下文），未能兑现雄心勃勃的承诺，导致资助机构产生怀疑。由美国政府委托撰写的、具有影响力的1966年ALPAC（自动语言处理咨询委员会）报告对机器翻译研究提出了高度批评，结论是它比人工翻译更慢、更不准确、更昂贵，导致该领域的资金被大幅削减。
2. **理论局限性暴露：** 基础性工作揭示了一些早期方法的固有局限性。最重大的打击来自马文·明斯基和西摩尔·帕尔特（Seymour Papert）1969年的著作《感知器》（*Perceptrons*）。虽然该书特别关注像罗森布拉特模型那样的单层感知器的局限性，但它提供了严谨的数学证明，表明这种简单的网络从根本上无法学习某些重要的模式类别，最著名的是XOR（异或）逻辑函数。尽管该分析不适用于更复杂的多层网络，但这本书被广泛解读为证明了整个连接主义（神经网络）方法的根本不足，显著抑制了对该研究方向的热情和资助超过十年。
3. **组合爆炸：** 许多AI问题，特别是涉及搜索的问题（如博弈、规划或定理证明），都受到“组合爆炸”的影响。随着问题规模或复杂性的增长，需要探索的可能状态或解决方案路径的数量呈指数级增长，迅速压垮了当时计算机有限的处理能力和内存容量。启发式方法有所帮助，但通常不足以应对现实世界规模的问题。
4. **框架问题与常识：** 符号AI系统在表示和推理人类毫不费力地用来理解世界的大量隐性常识知识方面遇到了巨大困难（例如，知道掉落物体会导致其下落，或者容器倾斜液体会溢出）。明确地表示这些知识被证明极其困难，而有效地推理当一个

动作发生时什么不会改变（框架问题，frame problem）是一个主要的理论障碍。

5. **关键政府报告：** 除了ALPAC报告，1973年由英国政府委托撰写的莱特希尔报告（Lighthill Report）对整个AI领域给出了高度批判性的评估，将进展归类为令人失望，并反对资助广泛的、无方向的AI研究。这导致英国AI资金大幅削减。

由于这些因素，美国的主要资助机构如DARPA和英国的政府机构将焦点从探索性的、基础的AI研究转向了具有特定、可实现目标和较短时间表的更应用型的项目。第一次AI寒冬是一次严酷的现实检验，迫使研究人员面对创造智能所固有的深层困难。

第二次，也许不那么严重的AI寒冬发生在20世纪80年代末和90年代初。这个周期主要是由专家系统（Expert Systems）专业市场的崩溃引发的。在20世纪80年代初期的初步兴奋和大量商业投资之后，专家系统的局限性变得明显：

- **高成本与高投入：** 构建和维护大型、复杂的规则库需要在知识工程上投入大量资金—— painstakingly 地从稀缺的人类专家那里提取知识。
- **脆弱性（Brittleness）：** 专家系统通常在其狭窄的专业领域内表现良好，但当面临略微超出其编程知识或常识边界的情况时，往往会突然且不可预测地失败。
- **可扩展性问题：** 添加新知识可能变得越来越困难，可能在现有规则库中引入矛盾或不可预见的相互作用。维护很复杂。
- **对专用硬件的依赖：** 许多早期专家系统是使用Lisp编程语言开发的，并运行在专门的、昂贵的“Lisp机器”上。功能强大、更便宜的、运行Unix和C的通用工作站的兴起，削弱了这种专用硬件的市场。

随着企业认识到这些局限性和所涉及的高昂成本，围绕专家系统的商业炒作逐渐消退。许多在20世纪80年代初成立的AI公司破产或转移了焦点，导致投资再次下滑，该领域经历了一段知名度降低和被认为停滞不前的时期。

这些AI寒冬，虽然在当时对研究人员和行业来说是痛苦的，但最终起到了作用。它们抑制了不切实际的期望，鼓励了更严谨的科学方法论，强调了更好理论理解的必要性，并推动研究人员探索替代方法和解决更易于管理的子问题。它们强调了通往人工智能的道路将是漫长而艰巨的，其特点是渐进式进步，偶尔穿插着突破，而不是向有感知能力的机器快速迈进。

神经网络复兴：连接主义的复苏

尽管《感知器》一书产生了抑制作用，并且在第一次AI寒冬期间符号AI占据主导地位，但对人工神经网络的研究并未完全停止。一小部分研究人员继续探索受大脑启发的计算模型。重新点燃广泛兴趣并克服了明斯基和帕尔特所强调的主要局限性的关键突破，是随着一种有效的多层网络训练算法的开发和普及而到来的：反向传播（backpropagation）。

虽然反向传播的核心思想（使用微积分的链式法则将误差信号反向传播回网络层）早先已被几位研究人员独立探索过（包括保罗·韦尔博斯（Paul Werbos）在他1974年的博士论文中），但大卫·鲁梅尔哈特（David Rumelhart）、杰弗里·辛顿（Geoffrey Hinton）和罗纳德·威廉姆斯（Ronald Williams）的工作，发表在包括1986年开创性著作《并行分布式处理：认知微观结构的探索》（*Parallel Distributed Processing: Explorations in the Microstructure of Cognition*）中的一章等有影响力的论文中，广泛传播了该算法并展示了其威力。

反向传播提供了一种有效的方法，用于计算多层神经网络中每个连接权重对网络输出与给定输入的期望目标输出之间的总误差的贡献程度。通过系统地将这些误差信号从输出层反向传播到输入层，该算法允许网络迭代地调整其权重以最小化误差。至关重要的是，多层网络（也称为多层感知器或MLP），与明斯基和帕尔特分析的单层感知器不同，被证明能够学习数据中复杂的非线性关系，包括以前有问题的XOR函数。只要有足够的隐藏单元，它们就可以充当通用函数逼近器。

这一突破在整个20世纪80年代和90年代初引发了对神经网络和受大脑启发的计算的重大兴趣复苏，通常被称为连接主义（Connectionist）复兴或并行分布式处理（PDP）的兴起。连接主义提供了一种与符号AI根本不同的视角。它提出，知识和智能可能并不存在于由中央处理器操纵的显式符号规则中，而是可以隐式地编码在大量简单的、类似神经元的处理单元之间并行操作的连接模式和强度中——非常像生物大脑的结构。在这种观点下，学习对应于根据经验修改这些连接的强度。

这一时期见证了连接主义范式内重要的理论和架构发展：

- **新的网络架构：** 除了标准的前馈MLP，研究人员还探索了其他网络类型。约翰·霍普菲尔德（John Hopfield）引入了霍普菲尔德网络（Hopfield networks, 1982），这是一种具有对称连接的循环网络，可以充当联想记忆（存储模式并从部分或嘈杂的线索中检索它们）。杰弗里·辛顿、特伦斯·塞诺斯基（Terrence Sejnowski）等人开发了玻尔兹曼机（Boltzmann machines, 1980年代中期），这是一种能够学习其输入概率分布的随机循环网络。
- **处理序列数据：** 标准的前馈网络难以处理像语音或文本这样的序列数据，因为顺序很重要。循环神经网络（Recurrent Neural Networks, RNNs）被开发出来，它包含反馈回路，允许来自先前输入的信息持续存在并影响当前输入的处理，赋予它们一种形式的记忆。虽然早期的RNN被证明在长序列上有效训练很困难，但它们为后来更复杂的循环模型奠定了基础。
- **用于视觉的卷积神经网络（CNNs）：** 受哺乳动物视觉皮层层次结构（Hubel和Wiesel关于简单和复杂细胞的工作）的启发，杨立昆（Yann LeCun）及其合作者在20世纪80年代末和90年代初开发了卷积神经网络（Convolutional Neural Networks, CNNs）。CNN采用专门的层，在输入图像上应用可学习的滤波器（卷积），使它们能够检测局部特征（如边缘或角点）。然后，池化层（Pooling

layers）逐步降低空间分辨率，同时保留重要信息。这种层次结构，随着层级加深学习日益复杂的特征，被证明对图像识别任务非常有效。LeCun的LeNet架构成功应用于手写数字识别以进行支票读取，是展示CNN威力的里程碑式成就。

连接主义的复兴，由反向传播算法以及像RNNs和CNNs这样日益复杂的网络架构的发展所驱动，为纯粹符号方法主导AI早期几十年的局面提供了强大而引人注目的替代方案。虽然仍然面临计算成本、数据需求和理论理解方面的挑战，但这次复苏为深度学习革命——将在21世纪极大地重塑该领域——奠定了关键的基础，无论是在算法上还是在架构上。AI的旅程，在经历了怀疑的寒冬之后，正准备迎接又一个春天。

第七章：深度学习的海啸：AI的现代复兴

经历了乐观与幻灭的周期性旅程后，人工智能领域带着一系列既有技术、来之不易的教训和持续存在的挑战进入了21世纪。虽然符号AI在某些领域仍占有一席之地，连接主义的复苏也奠定了重要基础，但在诸如视觉感知和自然语言理解等复杂任务上实现真正类似人类的能力仍然遥不可及。然后，大约从2000年代末开始，并在2012年后急剧加速，一股进步浪潮开始席卷该领域，以前所未有的速度取得突破。这场现代复兴，常被形容为“海啸”，主要由训练非常深（deep）的人工神经网络——具有许多处理单元层的网络——的进步所驱动。这种方法，恰当地命名为深度学习（Deep Learning），并非源于单一的理论启示，而是源于几个关键促成因素的强有力汇合，释放出的能力最终开始匹敌，甚至在某些特定任务上超越了人类的表现。

三大促成因素：大数据、GPU算力、先进算法

多层神经网络的理论潜力已被理解数十年，但实际限制常常阻碍其有效性，特别是对于足够深以应对现实世界复杂性的网络。深度学习革命的发生，是当三个关键要素最终同时达到临界点时，创造了一场完美的风暴，使得训练大型、深度网络变得可行且异常成功。

1. **大数据：学习的燃料：** 深度神经网络，特别是那些拥有数百万甚至数十亿参数的网络，是极其“饥饿”数据的。它们通过分析大量示例来学习复杂的模式和关系。试图在小数据集上训练如此大的模型通常会导致泛化能力差——模型可能会记住训练数据，但在新的、未见过的数据上表现不佳。然而，数字时代提供了必要的燃料。互联网的爆炸式增长、产生大量用户内容（文本、图像、视频）的社交媒体平台的兴起、带有摄像头和传感器的智能手机的普及、档案和记录的数字化，以及大型科学项目，创造了规模和种类都前所未有的数据集。至关重要的是，人们付出了专门的努力来整理大规模、带标签的数据集，专门用于训练机器学习模型。ImageNet数据集，包含数百万张标记好的、分属于数千个类别的图像，扮演了尤其重要的催化角色。它于2009年启动，其相关的年度竞赛（ILSVRC）成为计算机视觉的关键基准，这个大规模数据集的可获得性对于展示深度学习在图像识别方面的威力至关重要。同样，从网络上抓取的大量文本语料库为训练复杂的

语言模型提供了原材料。这种丰富的数据最终使得深度网络能够学习掌握复杂任务所需的复杂模式。

2. **GPU算力：计算的引擎：** 训练深度神经网络是计算密集型的。核心操作涉及在信号通过网络前向传播和误差梯度在训练期间反向传播（反向传播）时执行大量的矩阵乘法和其他可并行化计算。传统的中央处理单元（CPU），设计用于顺序任务执行，对于在合理时间内在海量数据集上训练大型深度学习模型来说实在太慢了。一个偶然的发现被证明具有变革性：图形处理单元（GPU）。GPU最初是为满足视频游戏中渲染复杂图形的高度并行计算需求而设计的，它包含数百或数千个相对简单的处理核心，优化用于并行执行计算。研究人员意识到，这种架构非常适合深度学习算法基础的矩阵和向量运算。利用GPU进行通用计算（GPGPU），通常由像NVIDIA于2007年推出的CUDA（Compute Unified Device Architecture，统一计算设备架构）这样的编程平台所促进，提供了显著的加速——对于训练深度网络，与CPU相比通常加速10倍到100倍甚至更多。这种硬件加速是一个关键的促成因素，使得在计算上可行地实验和训练比以前可能的大得多、深得多的网络架构成为可能。后来，像谷歌这样的科技巨头甚至会开发更专门化的硬件，如张量处理单元（Tensor Processing Units, TPUs），进一步优化以适应深度学习的特定计算模式。

3. **先进算法：训练的工具：** 虽然反向传播仍然是基本的学习算法，但训练*非常深*的网络（具有数十甚至数百层）带来了重大挑战。梯度在通过许多层反向传播时可能会变得极其微小或爆炸性地增大，从而阻碍了较早层的学习（梯度消失/爆炸问题，vanishing/exploding gradient problem）。此外，在这些复杂、非凸的损失地貌中优化数百万个参数也很困难。几项算法创新对于克服这些障碍至关重要：

    - **更好的激活函数：** 传统的激活函数如sigmoid和双曲正切（tanh）在极端值处会饱和，导致梯度消失。采用修正线性单元（Rectified Linear Unit, ReLU）——一个简单的函数，如果输入为正则输出输入，否则输出零——被证明非常有效。ReLU及其变体（Leaky ReLU, ELU）有助于缓解梯度消失问题，并允许更快的训练。
    - **复杂的优化算法：** 标准的随机梯度下降（Stochastic Gradient Descent, SGD）虽然有效，但可能收敛缓慢或陷入次优点。新的优化算法被开发出来，如AdaGrad、RMSprop，特别是Adam（Adaptive Moment Estimation，自适应矩估计），它们为每个参数单独调整学习率，通常能带来更快的收敛速度和更好的结果。
    - **有效的正则化技术：** 具有大量参数的深度网络容易过拟合——对训练数据学习得太好，包括其噪声，而无法泛化到新数据。像Dropout（在训练期间随机将一部分神经元激活设置为零）、批量归一化（Batch Normalization，在小批

量内归一化激活）以及L1/L2权重正则化等技术成为了提高泛化能力的标准实践。

- **架构创新：** 至关重要的是，研究人员开发了专门设计用于促进深度学习的网络架构。卷积神经网络（CNNs）被证明对视觉任务至关重要，利用了参数共享和分层特征提取。循环神经网络（RNNs），特别是像长短期记忆（Long Short-Term Memory, LSTM）网络和门控循环单元（Gated Recurrent Units, GRUs）这样的变体，通过引入门控机制来控制信息流，提高了处理序列数据的能力。也许最重要的是，2017年引入的Transformer架构，使用一种称为“自注意力”（self-attention）的机制，彻底改变了序列处理，使模型能够比RNN更有效地捕捉长期依赖关系。深度学习的成功很大程度上源于这些深的、分层的架构能够直接从原始数据中自动学习相关特征和表示的能力，从早期层的简单特征开始，到更深层建立起更复杂、抽象的概念。

正是这三个要素——提供原材料的海量数据集、提供计算引擎的强大GPU硬件，以及提供有效训练技术的精炼算法——的强大协同作用，点燃了深度学习的海啸，改变了AI研究和应用。

感知与语言的突破

深度学习革命的影响遍及AI的许多领域，但在那些长期以来对传统AI方法构成挑战的领域，即感官知觉（视觉和语音）和自然语言处理方面，其影响尤其巨大。深度学习直接从像素和原始文本等高维、非结构化数据中学习分层表示的能力，被证明特别适合这些任务。

- **计算机视觉（Computer Vision）：** 几十年来，计算机视觉系统严重依赖手工设计的特征和复杂的流程来试图理解图像。进展稳定但缓慢。深度学习，特别是深度卷积神经网络（Deep Convolutional Neural Networks, CNNs），改变了一切。转折点出现在2012年的ImageNet大规模视觉识别挑战赛（ILSVRC）上。一个名为AlexNet的深度CNN，由Alex Krizhevsky、Ilya Sutskever和Geoffrey Hinton开发，取得了比以往任何非深度学习方法都低得多的错误率，震惊了整个社区。这场胜利展示了在大型数据集（ImageNet）上使用GPU加速训练的深度CNN的威力。几乎一夜之间，深度学习成为了计算机视觉的主导范式。随后的研究产生了更深、更复杂的CNN架构（例如，VGG、GoogLeNet、ResNet），将性能推向更高。深度学习现在在广泛的视觉任务上取得了最先进的结果，通常在特定基准上超越了人类能力，包括物体检测和定位、图像分割（识别属于每个对象的像素）、人脸识别、姿态估计、生成图像标题，以及分析医学图像（如X射线和MRI）以辅助诊断。
- **自然语言处理（Natural Language Processing, NLP）：** 理解和生成人类语言，以其固有的模糊性、上下文依赖性和庞大的词汇量，一直是对AI的巨大挑战。深度学习在NLP领域带来了类似的革命。早期的成功涉及使用从大型文本语料库中

学习到的**词嵌入**（word embeddings，例如Word2Vec、GloVe，大约在2013年开发）。这些嵌入捕捉了语义关系（例如，“国王”和“男人”之间的向量差可能与“女王”和“女人”之间的相似），为下游NLP任务提供了更丰富的输入表示。循环神经网络（RNNs），特别是LSTM和GRU，通过更好地捕捉句子中单词之间的依赖关系，显著提高了机器翻译、情感分析和文本生成等序列建模任务的性能。然而，最重要的NLP突破是**Transformer架构**，由谷歌研究人员在2017年的论文《Attention Is All You Need》中提出。通过完全依赖“自注意力”机制，该机制允许模型在处理每个单词时权衡输入序列中不同单词的重要性，Transformer被证明在捕捉长距离依赖关系方面异常有效，并且在训练期间比RNN可以更有效地并行化。这种架构成为了现代**大型语言模型（Large Language Models, LLMs）**的基础，例如谷歌的BERT和OpenAI的GPT系列（GPT-2、GPT-3、GPT-4及其后继者）。这些模型，使用巨大的计算资源在海量文本数据上训练，在广泛的NLP任务中表现出非凡的流畅性和能力，包括高度连贯的文本生成、复杂的问答、摘要、少样本学习（few-shot learning，用最少的示例适应新任务），甚至计算机代码生成。

- **语音识别（Speech Recognition）：** 自动语音识别（Automatic Speech Recognition, ASR）也从深度学习中受益匪浅。传统的ASR系统涉及复杂的流程，包含独立的声学模型、发音模型和语言模型。深度学习模型，通常采用CNN、RNN/LSTM或基于Transformer的架构的组合，学会了更直接地将原始音频特征映射到文本转录，显著降低了词错误率。这种准确性的急剧提高使得语音交互在日常生活中变得实用，推动了像苹果的Siri、亚马逊的Alexa和谷歌助手这样的虚拟助手在智能手机和智能音箱上的广泛采用。
- **深度强化学习（Deep Reinforcement Learning, DRL）：** 将深度神经网络与强化学习原理相结合，创造了强大的深度强化学习（DRL）框架。深度学习使得RL智能体能够直接从高维感官输入（如游戏像素）中学习，克服了早期需要手工设计特征的RL方法的局限性。里程碑式的成就展示了DRL的力量。DeepMind著名地训练了智能体直接从像素输入玩Atari游戏达到超人水平（2013年，2015年）。最著名的成功是AlphaGo，它在2016年击败了复杂围棋的世界冠军李世石，这一成就此前被认为还需要几十年才能实现。AlphaGo结合了深度神经网络（用于评估棋盘位置和预测走法）和蒙特卡洛树搜索。随后的系统如AlphaGo Zero和AlphaZero取得了更强的能力，完全通过自我对弈学会了掌握围棋、国际象棋和日本将棋，无需人类数据或指导，超越了最强的人类棋手和专门的博弈程序。DRL现在正被应用于机器人学（学习控制策略）、自主系统、资源优化和科学发现等挑战。

深度学习的海啸，由大数据、强大的GPU和算法创新的融合所促成，导致了跨核心AI领域的变革性突破，特别是在理解复杂的感知数据和人类语言方面。它达到了以前被

认为无法企及的性能水平，重新点燃了全球对AI的兴趣和巨额投资，并将该领域推入了当前快速发展和广泛应用的时代。

第八章：计算与AI的当下与未来

从计算表格的机械齿轮到运行复杂学习算法的全球网络，这段旅程将我们带到了历史的一个关键时刻。人工智能，曾是局限于实验室和理论论文的小众学术追求，如今正迅速渗透到我们社会的结构中。在计算能力指数级增长和深度学习变革性成功的推动下，AI正从一种专业工具转变为无处不在的基础技术。我们正处在一个拐点，目睹AI应用重塑行业和日常生活，同时面临着关于其未来发展、社会影响以及智能本身性质的深刻问题。本章审视了当前AI应用的全景，探索了正在进行的研究前沿，深入探讨了围绕其部署的关键伦理对话，并对未来可能的发展轨迹提出了看法。

无处不在的AI：改变社会的应用

人工智能不再是未来的预测；它日益成为现代世界一个不可或缺的、常常是无形的部分。我们日常交互的许多技术都严重依赖于在幕后运行的AI算法。

- **信息获取与过滤：** 像谷歌这样的搜索引擎使用复杂的AI系统，包括大型语言模型，来理解用户查询意图，对数十亿网页进行排名，并直接提供相关答案。电子邮件提供商采用先进的AI驱动的垃圾邮件过滤器来保护用户免受不必要或恶意内容的侵害。社交媒体平台使用AI来策划新闻推送、推荐联系人并审核内容（尽管常常不完美）。
- **个性化与推荐：** 像亚马逊这样的电子商务巨头和像Netflix这样的流媒体服务广泛依赖AI推荐引擎。这些系统分析用户行为、购买历史和偏好，以建议适合个人口味的产品、电影或音乐，从而提高用户参与度和销售额。
- **语言与沟通：** 机器翻译工具，在线上随处可见并集成到浏览器和应用程序中，使用深度学习模型以日益流畅的方式打破语言障碍，促进全球交流和信息获取。像苹果的Siri、亚马逊的Alexa和谷歌助手这样的语音助手利用先进的语音识别和自然语言处理来理解口头命令、回答问题和控制智能家居设备。
- **交通与物流：** 虽然完全自主的自动驾驶汽车仍然是一个重大挑战，但集成AI功能的高级驾驶辅助系统（ADAS），如自适应巡航控制、车道保持和自动紧急制动，正成为新车的标准配置。AI优化配送服务的路线，管理智慧城市的交通流量，并改善供应链中的物流。
- **医疗保健：** AI正在医学领域取得重大进展。机器学习模型分析医学图像（X射线、CT扫描、病理切片），以帮助放射科医生和病理学家更早、更准确地检测癌症等疾病。AI通过分析庞大的生物数据集来加速复杂的新药发现和开发过程。临床决策支持系统根据患者数据和医学文献提供诊断建议或治疗方案。

- **金融与商业：** 金融行业大量利用AI进行算法交易（基于市场预测高速执行交易）、欺诈检测（实时识别可疑交易）、信用评分和风险评估，以及通过“机器人顾问”（robo-advisors）提供个性化财务建议。
- **科学发现：** AI正成为跨学科科学家的必备工具。像DeepMind的AlphaFold这样的系统在根据氨基酸序列预测蛋白质三维结构方面取得了革命性成功，加速了生物学研究。AI分析来自天文学望远镜、物理学粒子加速器、基因组学基因测序仪以及环境科学气候传感器的大量数据集，帮助研究人员揭示以前隐藏的模式和见解。
- **创意与内容生成：** 一个迅速兴起的领域是生成式AI（Generative AI）。像GPT-4这样的模型能够生成极其连贯且与上下文相关的文本，用于写作辅助、内容创作和聊天机器人。像DALL-E 2、Midjourney和Stable Diffusion这样的扩散模型（Diffusion models）可以根据文本描述创建新颖且常常极其逼真的图像。AI工具辅助音乐创作，甚至生成计算机代码。
- **机器人与自动化：** AI增强了机器人的能力，使其超越简单的重复性任务。由AI驱动的感知能力使机器人能够在复杂环境中导航，更有效地在仓库（例如亚马逊机器人）、生产线、农业以及潜在的家庭和老年护理中识别和操作物体。
- **网络安全：** 随着网络威胁变得日益复杂，AI越来越多地用于检测网络流量异常、识别恶意软件签名、预测潜在漏洞以及自动化对安全事件的响应。

这仅仅是一个缩影。AI正被集成到无数其他领域，通常不是作为独立产品，而是作为增强现有系统和流程的嵌入式能力。它正在成为现代技术基础设施的基本层面，提高效率，实现新功能，并从根本上改变我们生活、工作以及与世界互动的方式。

研究前沿：下一步是什么？

尽管取得了显著进展，特别是通过深度学习，但当前的AI系统与人类智能相比仍存在显著局限性。它们通常缺乏鲁棒性、常识、真正的理解力和适应性。因此，AI研究仍然充满活力，在多个前沿推动界限，以克服这些局限性并解锁新能力。活跃的研究领域主要包括：

- **规模化与基础模型（Foundation Models）：** 一个主导趋势是构建越来越大的神经网络，通常在包含文本、图像、代码和其他模态的极其广泛的数据集上进行训练。这些“基础模型”（如GPT-4、PaLM 2、Claude）旨在发展通用能力，然后可以以相对较少的特定任务数据进行适应（微调），用于广泛的特定下游任务。研究重点在于理解控制模型性能如何随规模、数据和计算量提高的“规模定律”（scaling laws）；探索在非常大的模型中意外出现的“涌现能力”（emergent abilities）；开发用于高效训练和推理这些庞大模型的技术；以及构建能够无缝处理和整合来自不同来源信息（例如，根据文本描述理解图像，或为视频生成文本评论）的真正多模态模型。

- **鲁棒性、可靠性与可解释性（XAI）：** 一个主要挑战是当前深度学习模型的“脆弱性”。它们在与其训练集相似的数据上可能表现异常出色，但当面对略有不同的输入或不可预见的情况时，可能会意外失败或做出无意义的预测（例如，旨在欺骗模型的对抗性样本）。确保AI系统的鲁棒性、可靠性和可预测性至关重要，尤其是在医疗保健或自动驾驶等高风险领域的部署。与此密切相关的是可解释性（explainability）或可理解性（interpretability）的挑战。许多深度学习模型像“黑箱”一样运作，使得理解它们*为什么*做出特定决策变得困难。可解释人工智能（Explainable AI, XAI）的研究旨在开发方法来审视这些模型内部，理解它们的推理过程，识别潜在偏差，并为其输出提供理由，这对于建立信任、调试错误和确保问责制至关重要。解决训练数据和算法中存在的固有偏差以确保不同人口群体的公平性，也是构建可信赖AI的关键组成部分。
- **因果AI（Causal AI）：** 当前大多数机器学习擅长识别数据中的相关性和模式。然而，相关性并不意味着因果关系。这些模型通常难以理解真正的因果关系。因果AI领域旨在通过开发能够进行因果推理、从数据中推断因果结构、预测干预效果（如果我们*做了*某事会发生什么？）以及回答反事实问题（如果某事有所不同会发生什么？）的方法来弥合这一差距。实现因果理解被认为是进行更稳健决策、科学发现以及构建具有更深层次世界知识的AI系统的基础。
- **神经符号AI（Neuro-symbolic AI）：** 为了弥合连接主义（深度学习）和符号主义（基于逻辑）方法之间的历史鸿沟，神经符号AI旨在结合两种范式的优势。目标是创建混合系统，利用深度学习从大量原始数据中学习复杂模式的能力，同时融合符号AI在显式知识表示、逻辑推理、抽象和常识推断方面的能力。支持者认为，这种协同作用可能导致AI系统在数据效率、可解释性、鲁棒性以及执行更高级别推理方面更强。
- **通用人工智能（Artificial General Intelligence, AGI）：** 创造具有人类水平认知能力、能够在人类能执行的所有任务范围内表现的AI，这一长期、极具雄心且常常引起争议的目标，仍然是某些研究人员的强大（尽管遥远）动力。AGI代表了复制通用智能的最终愿望，涵盖在复杂、动态环境中的学习、推理、问题解决、创造力、感知和互动。虽然关于AGI的进展——关于其可行性、可能的时间表和精确定义——存在激烈争论，但对其的追求继续激发着对智能本身性质的基础研究。
- **效率与可持续性（绿色AI，Green AI）：** 训练最先进的基础模型需要巨大的计算资源，消耗大量能源，并造成显著的碳足迹。这引发了对环境可持续性和公平获取的担忧，因为通常只有大型组织才能负担得起这样的计算开销。因此，对“绿色AI”的研究日益重要，重点是开发更节能的算法（例如，剪枝、量化、高效架构）、设计专门的低功耗硬件、优化分布式训练技术，以及探索用更小、资源消耗更少的模型实现高性能的方法。

因此，当前的AI研究是多方面的，不仅旨在扩展现有成功，而且旨在解决关于可靠性、理解、推理能力和效率方面的根本弱点，而实现人类水平通用智能的宏伟挑战继续在地平线上若隐若现。

关键对话：伦理、安全与治理

随着AI系统变得越来越强大、自主，并深度融入社会关键基础设施，其伦理影响和潜在的社会后果需要紧迫而审慎的考量。一场涉及研究人员、政策制定者、行业领袖、公民社会组织和公众的关键全球对话正在进行中，以应对复杂的挑战，并确保AI以负责任、安全和公平的方式开发和部署。关注的关键领域包括：

- **偏见与公平：** 在历史数据（通常反映现有社会偏见）上训练的AI模型可能会无意中学习、延续甚至放大这些偏见。这可能导致在招聘（有偏见的简历筛选）、贷款申请（歧视性信用评分）、刑事司法（有偏见的累犯预测）和面部识别（某些人群的错误率更高）等敏感领域产生歧视性结果。定义和确保AI系统的公平性是一个复杂的技术和伦理挑战，需要仔细审计、偏见缓解技术以及开发团队的多元化代表性。
- **隐私：** AI模型对数据的贪婪需求引发了重大的隐私担忧。问题包括收集大量数据集的方法（通常涉及用户数据抓取）、用户同意的充分性、AI驱动监控的潜力（例如，大规模面部识别、行为跟踪），以及用于训练或由AI系统处理的敏感数据的安全性。在数据驱动AI的益处与基本隐私权之间取得平衡是一项关键的持续性任务。
- **问责与透明度：** 当AI系统出错或造成伤害时——例如，自动驾驶车辆的错误导致事故，或医疗AI提供错误诊断——确定责任人可能极其困难，特别是对于复杂且不透明的“黑箱”模型。缺乏透明度阻碍了审计、调试、理解失败模式，并最终阻碍了建立公众信任。这直接关系到对可解释AI（XAI）的需求和明确的问责法律框架。
- **工作岗位流失与经济影响：** 人们持续担忧AI驱动的自动化可能在广泛行业中取代人类工人，从制造业和运输业到客户服务、行政管理甚至创意行业。虽然AI也可能创造新的就业机会，但管理转型、提供充分的再培训和教育，以及制定潜在的社会安全网以应对自动化造成的经济中断，是重大的社会挑战。
- **虚假信息与操纵：** 能够创建逼真但完全虚假的文本、图像、音频和视频（“深度伪造”，deepfakes）的生成式AI模型日益复杂，构成了重大风险。这项技术可能被利用来大规模传播虚假信息和宣传、冒充个人、自动化网络钓鱼诈骗、操纵公众舆论，并侵蚀对机构和数字媒体的信任。开发技术对策（如强大的检测方法）和培养媒体素养是关键的防御措施。
- **自主系统的安全与保障：** 确保在物理世界中运行的AI系统（如自动驾驶汽车、自主无人机、工业机器人或未来的医疗机器人）的安全、可靠和可预测行为至关重

要。这些系统必须在广泛条件下安全运行，妥善处理不可预见的事件，并能抵抗恶意攻击或操纵。致命自主武器（Lethal Autonomous Weapons, LAWs）——能够在没有人类干预的情况下选择和攻击目标——的开发引发了尤其深刻的伦理和人道主义关切，导致国际社会呼吁对其进行监管或禁止。

- **存在风险（Existential Risk）：** 展望更长远的未来，一些研究人员和哲学家对可能发展出超级智能（superintelligence）——在所有方面都远超人类认知能力的AI——所带来的潜在长期风险表示担忧。担忧在于，如果创造出这样的实体，确保其目标与人类价值观保持一致，并防止其造成意想不到的灾难性损害可能极其困难，可能对人类构成存在风险。虽然高度推测性，但这种长期视角为AI安全和对齐（alignment）研究提供了信息。
- **治理与监管：** 认识到这些多方面的挑战，世界各地的政府、国际组织（如欧盟、经合组织、联合国）和行业机构正在积极努力建立治理AI的框架。这包括制定伦理原则（例如，公平、透明、问责）、技术标准、负责任开发和部署的最佳实践，以及颁布具体的法律法规（例如，欧盟AI法案）。找到正确的监管平衡点——既能促进创新，又能减轻风险并保护基本权利——是一项复杂且持续的全球性努力。

应对这些复杂问题需要多方利益相关者的方法，促进AI研究人员、社会科学家、伦理学家、政策制定者、企业和公众之间的合作，以塑造AI的发展和部署，使其符合人类价值观并惠及整个社会。

未来轨迹：推测与愿景

预测技术的精确长期未来是出了名的困难，然而，当前的趋势和研究前沿允许我们对计算和AI的潜在轨迹进行一些有根据的推测。

短期（未来5-10年）： 我们可以预期现有AI技术，特别是大型语言模型和计算机视觉系统，将持续整合和完善到更广泛的产品和服务中。这很可能导致更个性化的用户体验、增强的生产力工具（例如，复杂的编码助手、自动化报告生成、AI驱动的研究辅助工具），以及在特定领域能力日益增强的自动化。自然语言理解和生成、图像分析以及语音识别方面的进展可能会稳步持续。自主系统，特别是在驾驶方面，可能会在辅助功能（特定条件下的L3或L4级自主性）上看到渐进式改进，但完全自主（L5级）车辆的广泛部署可能因安全和监管障碍而仍然难以实现。对AI伦理、公平、透明度的关注以及初步监管框架的实施将会加强。

中期（10-25年）： AI有潜力在各个领域产生更具变革性的影响。我们可能会看到AI分析复杂数据集和生成假设的能力，在医学（基于基因组数据的个性化治疗）、材料科学（设计具有所需特性的新型材料）以及气候变化建模与缓解等领域显著加速科学发现。个性化教育平台可以比当前系统更有效地根据个别学生的需求和进度定制学习体验。人机协作可能在复杂的专业任务中变得司空见惯，增强人类能力而不仅仅是取代它们。由更复杂的AI感知和控制增强的机器人技术，可能在服务行业、物流、医疗保

健（例如，辅助外科医生或提供病人护理）甚至家庭中变得更加通用和普遍。社会对自动化经济影响的适应将变得日益关键，可能需要教育、劳动力培训和社会支持体系发生重大转变。AI的治理框架可能会变得更加成熟和国际协调，尽管挑战无疑仍将存在。

长期（25年以上）： 长期未来更具推测性，并取决于潜在的基础性突破。实现通用人工智能（AGI）的前景仍然是最深刻的不确定性。如果AGI得以实现，它可能代表着与农业革命或工业革命同等重要的技术断裂点，既有潜力解决人类一些最宏大的挑战（疾病、贫困、环境退化），也带来了前面讨论过的巨大风险（对齐、控制、存在威胁）。AGI是否可以实现，以及在什么时间尺度上实现，仍然是激烈辩论的主题。除了AGI，可能还会出现全新的AI范式，也许会超越当前的深度学习方法，或者从生物智能中汲取更深的灵感。人类与日益强大的AI系统持续的共同进化，无疑将继续重塑我们的社会、文化、经济，甚至可能重塑我们对自身的理解。

回顾阿达·洛夫莱斯在19世纪那句著名的断言——“分析机丝毫没有声称能*创造*任何东西”——能够创作音乐、创造令人惊叹的视觉艺术、撰写看似合理的新颖文本的生成式AI的出现，无疑挑战了我们对机器“创造”意味着什么的解释。关于智能、创造力和意识本质的基本问题，最早由洛夫莱斯和图灵等先驱者思考，仍然是AI旅程的核心。前方的道路充满了巨大的希望和重大的危险。通过审慎的研究、深思熟虑的对话和负责任的治理来明智地塑造这一轨迹，将是21世纪决定性的挑战和机遇之一。计算与AI的故事远未结束；其最具变革性的篇章或许尚未书写。