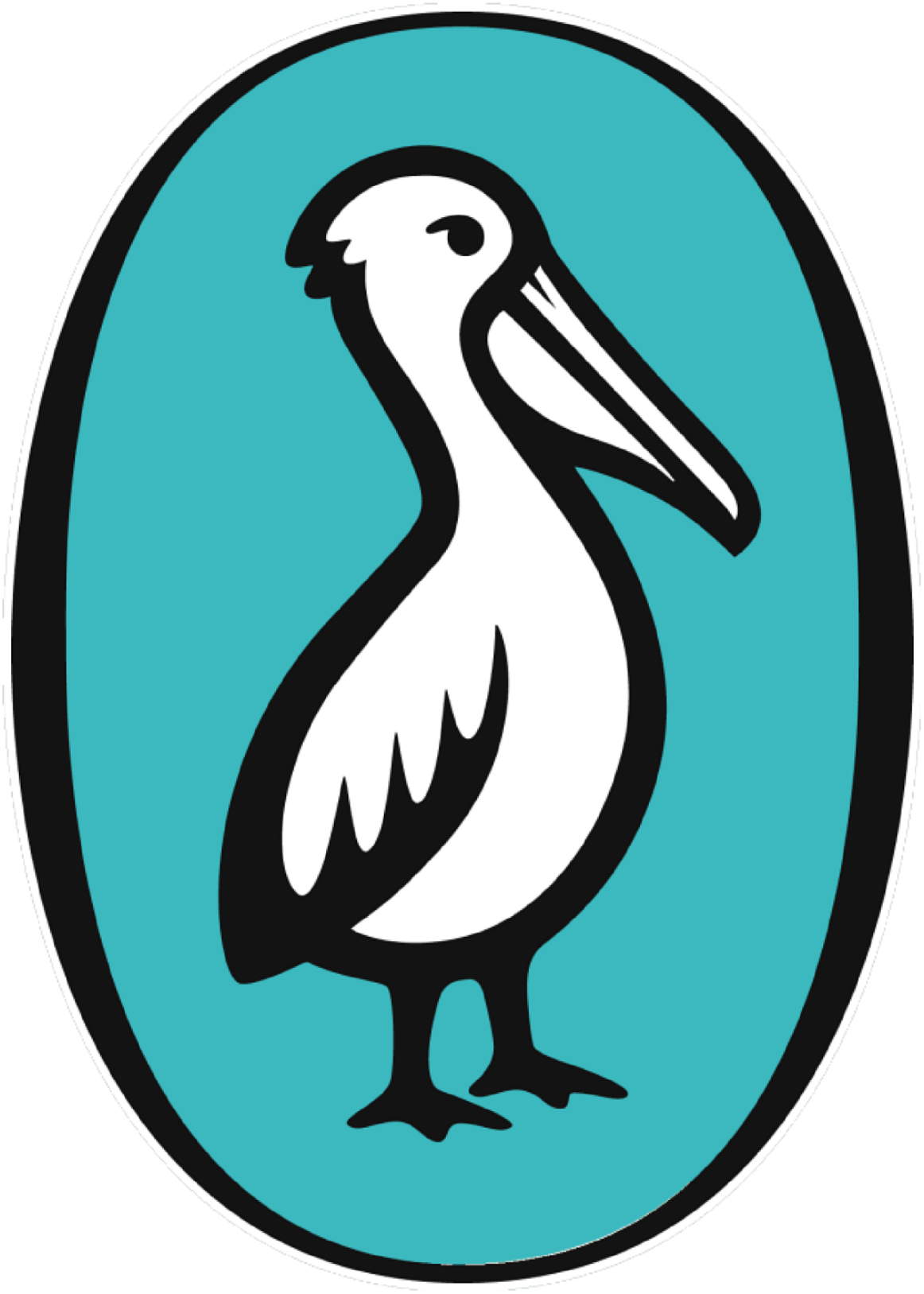

Contents

About the Author

Melanie Mitchell is a professor of Computer Science at Portland State University and External Professor at the Santa Fe Institute. She is the author of *An Introduction to Genetic Algorithms*, a widely known introductory book, and *Complexity: A Guided Tour*, which won the 2010 Phi Beta Kappa Science Book Award.

To my parents, who taught me how to be a
thinking human, and so much more

Illustration Credits

Figure 1: Drawing of neuron adapted from C. Ling, M. L. Hendrickson and R. E. Kalil, 'Resolving the Detailed Structure of Cortical and Thalamic Neurons in the Adult Rat Brain with Refined Biotinylated Dextran Amine Labeling', *PLOS ONE* 7, no. 11 (2012), e45886. Image licensed under Creative Commons Attribution 4.0 International license (creativecommons.org/licenses/by/4.0/).

Figure 2: Handwritten characters image by Josef Steppan, commons.wikimedia.org/wiki/File:MnistExamples.png. Image licensed under Creative Commons Attribution-ShareAlike 4.0 International license (creativecommons.org/licenses/by-sa/4.0/deed.en).

Figure 3: Author.

Figure 4: Author.

Figure 5: Author.

Figure 6: media.defense.gov/2015/May/15/2001047923/-1/-1/0/150506-F-BD468-053.JPG, accessed Dec. 4, 2018 (public domain).

Figure 7: Author.

Figure 8: Author.

Figure 9: Author.

Figure 10: Author.

Figure 11: Author.

Figure 12: Author.

Figure 13: Author.

Figure 14: From twitter.com/amywebb/status/841292068488118273, accessed Dec. 7, 2018. Reprinted by permission of Amy Webb.

Figure 15: www.nps.gov/yell/learn/nature/osprey.htm (public domain); www.fs.usda.gov/Internet/FSE_MEDIA/stelprdb5371680.jpg (public domain).

Figure 16: From twitter.com/jackyalcine/status/615329515909156865, accessed Dec. 7, 2018. Reprinted by permission of Jacky Alcine.

Figure 17: From www.flickr.com/photos/jozjozjoz/352910684, accessed Dec. 7, 2018. Reprinted by permission of Joz Wang of jozjozjoz.com.

Figure 18: From C. Szegedy et al., 'Intriguing Properties of Neural Networks', in *Proceedings of the International Conference on Learning Representations* (2014). Reprinted by permission of Christian Szegedy.

Figure 19: From A. Nguyen, J. Yosinski and J. Clune, 'Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), 427–36. Reprinted by permission of the authors.

Figure 20: Figure adapted from M. Sharif et al., 'Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition', in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016), 1528–40. Reprinted by permission of the authors. Milla Jovovich photograph is from commons.wikimedia.org/wiki/File:Milla_Jovovich.png, by Georges Biard, licensed under Creative Commons Attribution-Share Alike 3.0 Unported license (creativecommons.org/licenses/by-sa/3.0/deed.en).

Figure 21: Author.

Figure 22: From www.cs.cmu.edu/~robosoccer/image-gallery/legged/2003/aibo-with-ball12.jpg. Reprinted by permission of Manuela Veloso.

Figure 23: Author.

Figure 24: Author.

Figure 25: Author.

Figure 26: Author.

Figure 27: Author.

Figure 28: Author.

Figure 29: Author.

Figure 30: Author.

Figure 31: Author.

Figure 32: Author.

Figure 33: Author.

Figure 34: Author.

Figure 35: Author.

Figure 36: Author.

Figure 37: Adapted from T. Mikolov et al., 'Distributed Representations of Words and Phrases and Their Compositionality', in *Advances in Neural Information Processing Systems* (2013), 3111–19. Adapted by permission of Tomas Mikolov.

Figure 38: Author.

Figure 39: Author. Photogrpah is from the Microsoft COCO set: cocodataset.org.

Figure 40: Photographs and captions are from the Microsoft COCO set: cocodataset.org.

Figure 41: Photographs from nic.droppages.com. Reprinted by permission of Oriol Vinyals.

Figure 42: Top row: Photogrpahs from O. Vinyals et al., 'Show and Tell: A Neural Image Caption Generator', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), 3156–64. Reprinted by permission of Oriol Vinyals. Bottom row, left: Road-Tech Safety Services. Reprinted by permission of Ben Jeffrey. Bottom row, right: Nikoretro, https://www.flickr.com/photos/bellatrix6/4727507323/in/album-72057594083648059. Licensed under Creative Commons Attribution-ShareAlike 2.0 Generic license: https://creativecommons.org/licenses/by-sa/2.0/.

Figure 43: Photographs from H. Chen et al., 'Attacking Visual Language Grounding with Adversarial Examples: A Case Study on Neural Image Captioning', in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, vol. 1, *Long Papers* (2018), 2587–97. Reprinted with permission of Hongge Chen and the Association for Computational Linguistics.

Figure 44: Dorothy Alexander / Alamy Stock Photo.

Figure 45: From www.foundalis.com/res/bps/bpidx.htm. Original images are from M. Bongard, *Pattern Recognition* (New York: Spartan Books, 1970).

Figure 46: From www.foundalis.com/res/bps/bpidx.htm. Original images are from M. Bongard, *Pattern Recognition* (New York: Spartan Books, 1970).

Figure 47: Author.

Figure 48: Photographs taken by author.

Figure 49: www.nps.gov/dena/planyourvisit/pets.htm (public domain); pxhere.com/en/photo/1394259 (public domain); Peter Titmuss / Alamy Stock Photo; Thang Nguyen, www.flickr.com/photos/70209763@N00/399996115, licensed under Creative Commons Attribution-ShareAlike 2.0 Generic licence (creativecommons.org/licenses/by-sa/2.0/).

Figure 50: P. Souza, *Obama: An Intimate Portrait* (New York: Little, Brown, 2018), 102 (public domain).

PROLOGUE:

Terrified

Computers seem to be getting smarter at an alarming rate, but one thing they still can't do is appreciate irony. That's what was on my mind a few years ago, when, on my way to a discussion about artificial intelligence (AI), I got lost in the capital of searching and finding – the Googleplex, Google's world headquarters in Mountain View, California. What's more, I was lost inside the Google Maps building. Irony squared.

The Maps building itself had been easy to find. A Google Street View car was parked by the front door, a hulking appendage crowned by a red-and-black soccer ball of a camera sticking up from its roof. However, once inside, with my prominent 'Visitor' badge assigned by security, I wandered, embarrassed, among warrens of cubicles occupied by packs of Google workers, headphones over ears, intently typing on Apple desktops. After some (map-less) random search, I finally found the conference room assigned for the day-long meeting and joined the group gathered there.

The meeting, in May 2014, had been organized by Blaise Agüera y Arcas, a young computer scientist who had recently left a top position at Microsoft to help lead Google's machine intelligence effort. Google started out in 1998 with one 'product': a website that used a novel, extraordinarily successful method for searching the web. Over the years, Google has evolved into the world's most important tech company and now offers a vast array of products and services, including Gmail, Google Docs, Google Translate, YouTube, Android, many more that you might use every day, and some that you've probably never heard of.

Google's founders, Larry Page and Sergey Brin, have long been motivated by the idea of creating artificial intelligence in computers, and this quest has become a major focus at Google. In the last decade, the company has hired a profusion of AI experts, most notably Ray Kurzweil, a well-known inventor and a controversial futurist who promotes the idea of an AI 'Singularity', a time in the near future when computers will become smarter than humans. Google hired Kurzweil to help realize this vision. In 2011, Google created an internal AI research group called Google Brain; since then, the company has also acquired an impressive array of AI start-up companies with equally optimistic names: Applied Semantics, DeepMind and Vision Factory, among others.

In short, Google is no longer merely a web-search portal – not by a long shot. It is rapidly becoming an applied AI company. AI is the glue that unifies the diverse products, services and blue-sky research efforts offered by Google and its parent company, Alphabet. The company's ultimate aspiration is reflected in the original

mission statement of its DeepMind group: 'Solve intelligence and use it to solve everything else.'[1]

AI and *GEB*

I was pretty excited to attend an AI meeting at Google. I had been working on various aspects of AI since graduate school in the 1980s and had been tremendously impressed by what Google had accomplished. I also thought I had some good ideas to contribute. But I have to admit that I was there only as a tag-along. The meeting was happening so that a group of select Google AI researchers could hear from and converse with Douglas Hofstadter, a legend in AI and the author of a famous book cryptically titled *Gödel, Escher, Bach: an Eternal Golden Braid*, or more succinctly, *GEB* (pronounced 'gee-ee-bee'). If you're a computer scientist, or a computer enthusiast, you've probably never heard of it, or read it, or tried to read it.

Written in the 1970s, *GEB* was an outpouring of Hofstadter's many intellectual passions – mathematics, art, music, language, humour and wordplay, all brought together to address the deep questions of how intelligence, consciousness and the sense of self-awareness that each human experiences so fundamentally can emerge from the non-intelligent, nonconscious substrate of biological cells. It's also about how intelligence and self-awareness might eventually be attained by computers. It's a unique book; I don't know of any other book remotely like it. It's not an easy read, and yet it became a bestseller and won both the Pulitzer Prize and the National Book Award. Without a doubt, *GEB* inspired more young people to pursue AI than any other book. I was one of those young people.

In the early 1980s, after graduating from college with a maths degree, I was living in New York City, teaching maths in a prep school, unhappy, and casting about for what I really wanted to do in life. I discovered *GEB* after reading a rave review in *Scientific American*. I went out and bought the book immediately. Over the next several weeks, I devoured it, becoming increasingly convinced that not only did I want to become an AI researcher but I specifically wanted to work with Douglas Hofstadter. I had never before felt so strongly about a book, or a career choice.

At the time, Hofstadter was a professor in computer science at Indiana University, and my quixotic plan was to apply to the computer science PhD programme there, arrive, and then persuade Hofstadter to accept me as a student. One minor problem was that I had never taken even one computer science course. I had grown up with computers; my father was a hardware engineer at a 1960s tech start-up company, and as a hobby he built a mainframe computer in our family's den. The refrigerator-sized Sigma 2 machine wore a magnetic button proclaiming, 'I pray in FORTRAN,' and as a child I was half-convinced it did, quietly at night, while the rest of the family was asleep. Growing up in the 1960s and '70s, I learned a bit of each of the popular languages of the day: FORTRAN, then BASIC, then Pascal, but I knew next to nothing about proper programming techniques, not to mention anything else an incoming computer science graduate student needs to know.

To speed along my plan, I quit my teaching job at the end of the school year, moved to Boston and started taking introductory computer science courses to prepare for my new career. A few months into my new life, I was on the campus of the Massachusetts Institute of Technology, waiting for a class to begin, and I caught sight of a poster advertising a lecture by Douglas Hofstadter, to take place in two days on that very campus. I did a double take; I couldn't believe my good fortune. I went to the lecture, and after a long wait for my turn in a crowd of admirers I managed to speak to Hofstadter. It turned out he was in the middle of a year-long sabbatical at MIT, after which he was moving from Indiana to the University of Michigan in Ann Arbor.

To make a long story short, after some persistent pursuit on my part, I persuaded Hofstadter to take me on as a research assistant, first for a summer, and then for the next six years as a graduate student, after which I graduated with a doctorate in computer science from Michigan. Hofstadter and I have kept in close touch over the years and have had many discussions about AI. He knew of my interest in Google's AI research and was nice enough to invite me to accompany him to the Google meeting.

Chess and the First Seed of Doubt

The group in the hard-to-locate conference room consisted of about twenty Google engineers (plus Douglas Hofstadter and myself), all of whom were members of various Google AI teams. The meeting started with the usual going around the room and having people introduce themselves. Several noted that their own careers in AI had been spurred by reading *GEB* at a young age. They were all excited and curious to hear what the legendary Hofstadter would say about AI. Then Hofstadter got up to speak. 'I have some remarks about AI research in general, and here at Google in particular.' His voice became passionate. 'I am terrified. Terrified.'

Hofstadter went on.[2] He described how, when he first started working on AI in the 1970s, it was an exciting prospect but seemed so far from being realized that there was no 'danger on the horizon, no sense of it actually *happening*'. Creating machines with humanlike intelligence was a profound intellectual adventure, a long-term research project whose fruition, it had been said, lay at least 'one hundred Nobel prizes away'.[3] Hofstadter believed AI was possible in principle: 'The "enemy" were people like John Searle, Hubert Dreyfus, and other sceptics, who were saying it was impossible. They did not understand that a brain is a hunk of matter that obeys physical law and the computer can simulate anything ... the level of neurons, neurotransmitters, et cetera. In theory, it can be done.' Indeed, Hofstadter's ideas about simulating intelligence at various levels – from neurons to consciousness – were discussed at length in *GEB* and had been the focus of his own research for decades. But in practice, until recently, it seemed to Hofstadter that general 'human-level' AI had no chance of occurring in his (or even his children's) lifetime, so he didn't worry much about it.

Near the end of *GEB*, Hofstadter had listed 'Ten Questions and Speculations' about artificial intelligence. Here's one of them: 'Will there be chess programs that can beat anyone?' Hofstadter's speculation was 'no'. 'There may be programs which can beat anyone at chess, but they will not be exclusively chess players. They will be programs of *general* intelligence.'[4]

At the Google meeting in 2014, Hofstadter admitted that he had been 'dead wrong'. The rapid improvement in chess programs in the 1980s and '90s had sown the first seed of doubt in his appraisal of AI's short-term prospects. Although the AI pioneer Herbert Simon had predicted in 1957 that a chess program would be world champion 'within 10 years', by the mid-1970s, when Hofstadter was writing *GEB*, the best computer chess programs played only at the level of a good (but not great) amateur. Hofstadter had befriended Eliot Hearst, a chess champion and psychology professor who had written extensively on how human chess experts differ from computer chess programs. Experiments showed that expert human players rely on quick recognition of patterns on the chessboard to decide on a move rather than the extensive brute-force look-ahead search that all chess programs use. During a game, the best human players can perceive a configuration of pieces as a particular 'kind of position' that requires a certain 'kind of strategy'. That is, these players can quickly recognize particular configurations and strategies as instances of higher-level concepts. Hearst argued that without such a general ability to perceive patterns and recognize abstract concepts,

chess programs would never reach the level of the best humans. Hofstadter was persuaded by Hearst's arguments.

However, in the 1980s and '90s, computer chess saw a big jump in improvement, mostly due to the steep increase in computer speed. The best programs still played in a very unhuman way: performing extensive look-ahead to decide on the next move. By the mid-1990s, IBM's Deep Blue machine, with specialized hardware for playing chess, had reached the Grandmaster level, and in 1997 the program defeated the reigning world chess champion, Garry Kasparov, in a six-game match. Chess mastery, once seen as a pinnacle of human intelligence, had succumbed to a brute-force approach.

Music: The Bastion of Humanity

Although Deep Blue's win generated a lot of hand-wringing in the press about the rise of intelligent machines, 'true' AI still seemed quite distant. Deep Blue could play chess, but it couldn't do anything else. Hofstadter had been wrong about chess, but he still stood by the other speculations in *GEB*, especially the one he had listed first:

> Question: Will a computer ever write beautiful music? Speculation: Yes, but not soon.

Hofstadter continued,

> Music is a language of emotions, and until programs have emotions as complex as ours, there is no way a program will write anything beautiful. There can be 'forgeries' – shallow imitations of the syntax of earlier music – but despite what one might think at first, there is much more to musical expression than can be captured in syntactic rules ... To think ... that we might soon be able to command a preprogrammed mass-produced mail-order twenty-dollar desk-model 'music box' to bring forth from its sterile circuitry pieces which Chopin or Bach might have written had they lived longer is a grotesque and shameful misestimation of the depth of the human spirit.[5]

Hofstadter described this speculation as 'one of the most important parts of *GEB* – I would have staked my life on it'.

In the mid-1990s, Hofstadter's confidence in his assessment of AI was again shaken, this time quite profoundly, when he encountered a program written by a musician, David Cope. The program was called Experiments in Musical Intelligence, or EMI (pronounced 'Emmy'). Cope, a composer and music professor, had originally developed EMI to aid him in his own composing process by automatically creating pieces in Cope's specific style. However, EMI became famous for creating pieces in the style of classical composers such as Bach and Chopin. EMI composes by following a large set of rules, developed by Cope, that are meant to capture a general syntax of composition. These rules are applied to copious examples from a particular composer's opus in order to produce a new piece 'in the style' of that composer.

Back at our Google meeting, Hofstadter spoke with extraordinary emotion about his encounters with EMI:

> I sat down at my piano and I played one of EMI's mazurkas 'in the style of Chopin'. It didn't sound exactly like Chopin, but it sounded enough like Chopin, and like coherent music, that I just felt *deeply* troubled.
>
> Ever since I was a child, music has thrilled me and moved me to the very core. And every piece that I love feels like it's a direct message from the emotional heart of the human being who composed it. It feels like it is giving

me access to their innermost soul. And it feels like there is *nothing* more human in the world than that expression of music. Nothing. The idea that pattern manipulation of the most superficial sort can yield things that sound as if they are coming from a human being's heart is very, very troubling. I was just completely thrown by this.

Hofstadter then recounted a lecture he gave at the prestigious Eastman School of Music, in Rochester, New York. After describing EMI, Hofstadter had asked the Eastman audience – including several music theory and composition faculty – to guess which of two pieces a pianist played for them was a (little-known) mazurka by Chopin and which had been composed by EMI. As one audience member described later, 'The first mazurka had grace and charm, but not "true-Chopin" degrees of invention and large-scale fluidity ... The second was clearly the genuine Chopin, with a lyrical melody; large-scale, graceful chromatic modulations; and a natural, balanced form.'[6] Many of the faculty agreed and, to Hofstadter's shock, voted EMI for the first piece and 'real-Chopin' for the second piece. The correct answers were the reverse.

In the Google conference room, Hofstadter paused, peering into our faces. No one said a word. At last he went on. 'I was terrified by EMI. Terrified. I hated it, and was extremely threatened by it. It was threatening to destroy what I most cherished about humanity. I think EMI was the most quintessential example of the fears that I have about artificial intelligence.'

Google and the Singularity

Hofstadter then spoke of his deep ambivalence about what Google itself was trying to accomplish in AI – self-driving cars, speech recognition, natural-language understanding, translation between languages, computer-generated art, music composition, and more. Hofstadter's worries were underlined by Google's embrace of Ray Kurzweil and his vision of the Singularity, in which AI, empowered by its ability to improve itself and learn on its own, will quickly reach, and then exceed, human-level intelligence. Google, it seemed, was doing everything it could to accelerate that vision. While Hofstadter strongly doubted the premise of the Singularity, he admitted that Kurzweil's predictions still disturbed him. 'I was terrified by the scenarios. Very skeptical, but at the same time, I thought, maybe their timescale is off, but maybe they're right. We'll be completely caught off guard. We'll think nothing is happening and all of a sudden, before we know it, computers will be smarter than us.'

If this actually happens, 'we will be superseded. We will be relics. We will be left in the dust.

'Maybe this is going to happen, but I don't want it to happen *soon*. I don't want my children to be left in the dust.'

Hofstadter ended his talk with a direct reference to the very Google engineers in that room, all listening intently: 'I find it very scary, very troubling, very sad, and I find it terrible, horrifying, bizarre, baffling, bewildering, that people are rushing ahead blindly and deliriously in creating these things.'

Why is Hofstadter Terrified?

I looked around the room. The audience appeared mystified, embarrassed even. To these Google AI researchers, none of this was the least bit terrifying. In fact, it was old news. When Deep Blue beat Kasparov, when EMI started composing Chopin-like mazurkas, and when Kurzweil wrote his first book on the Singularity, many of these engineers had been in high school, probably reading *GEB* and loving it, even though its

AI prognostications were a bit out of date. The reason they were working at Google was precisely to make AI happen – not in a hundred years, but now, as soon as possible. They didn't understand what Hofstadter was so stressed out about.

People who work in AI are used to encountering the fears of people outside the field, who have presumably been influenced by the many science fiction movies depicting superintelligent machines that turn evil. AI researchers are also familiar with the worries that increasingly sophisticated AI will replace humans in some jobs, that AI applied to big data sets could subvert privacy and enable subtle discrimination, and that ill-understood AI systems allowed to make autonomous decisions have the potential to cause havoc.

Hofstadter's terror was in response to something entirely different. It was not about AI becoming too smart, too invasive, too malicious, or even too useful. Instead, he was terrified that intelligence, creativity, emotions, and maybe even consciousness itself would be too *easy* to produce – that what he valued most in humanity would end up being nothing more than a 'bag of tricks', that a superficial set of brute-force algorithms could explain the human spirit.

As *GEB* made abundantly clear, Hofstadter firmly believes that the mind and all its characteristics emerge wholly from the physical substrate of the brain and the rest of the body, along with the body's interaction with the physical world. There is nothing immaterial or incorporeal lurking there. The issue that worries him is really one of complexity. He fears that AI might show us that the human qualities we most value are disappointingly simple to mechanize. As Hofstadter explained to me after the meeting, here referring to Chopin, Bach, and other paragons of humanity, 'If such minds of infinite subtlety and complexity and emotional depth could be trivialized by a small chip, it would destroy my sense of what humanity is about.'

I Am Confused

Following Hofstadter's remarks, there was a short discussion, in which the nonplussed audience prodded Hofstadter to further explain his fears about AI and about Google in particular. But a communication barrier remained. The meeting continued, with project presentations, group discussion, coffee breaks, the usual – none of it really touching on Hofstadter's comments. Close to the end of the meeting, Hofstadter asked the participants for their thoughts about the near-term future of AI. Several of the Google researchers predicted that general human-level AI would probably emerge within the next thirty years, in large part due to Google's own advances on the brain-inspired method of 'deep learning'.

I left the meeting scratching my head in confusion. I knew that Hofstadter had been troubled by some of Kurzweil's Singularity writings, but I had never before appreciated the degree of his emotion and anxiety. I also had known that Google was pushing hard on AI research, but I was startled by the optimism several people there expressed about how soon AI would reach a general 'human' level. My own view had been that AI had progressed a lot in some narrow areas but was still nowhere close to having the broad, general intelligence of humans, and it would not get there in a century, let alone thirty years. And I had thought that people who believed otherwise were vastly underestimating the complexity of human intelligence. I had read Kurzweil's books and had found them largely ridiculous. However, listening to all the comments at the meeting, from people I respected and admired, forced me to critically examine my own views. While assuming that these AI researchers underestimated humans, had I in turn underestimated the power and promise of current-day AI?

Over the months that followed, I started paying more attention to the discussion surrounding these questions. I started to notice the slew of articles, blog posts, and

entire books by prominent people suddenly telling us we should start worrying, right now, about the perils of 'superhuman' AI. In 2014, the physicist Stephen Hawking proclaimed, 'The development of full artificial intelligence could spell the end of the human race.'[7] In the same year, the entrepreneur Elon Musk, founder of the Tesla and SpaceX companies, said that artificial intelligence is probably 'our biggest existential threat' and that 'with artificial intelligence we are summoning the demon.'[8] Microsoft's co-founder Bill Gates concurred: 'I agree with Elon Musk and some others on this and don't understand why some people are not concerned.'[9] The philosopher Nick Bostrom's book *Superintelligence*, on the potential dangers of machines becoming smarter than humans, became a surprise bestseller, despite its dry and ponderous style.

Other prominent thinkers were pushing back. Yes, they said, we should make sure that AI programs are safe and don't risk harming humans, but any reports of near-term superhuman AI are greatly exaggerated. The entrepreneur and activist Mitchell Kapor advised, 'Human intelligence is a marvelous, subtle, and poorly understood phenomenon. There is no danger of duplicating it anytime soon.'[10] The roboticist (and former director of MIT's AI Lab) Rodney Brooks agreed, stating that we 'grossly overestimate the capabilities of machines – those of today and of the next few decades'.[11] The psychologist and AI researcher Gary Marcus went so far as to assert that in the quest to create 'strong AI' – that is, *general* human-level AI – 'there has been almost no progress.'[12]

I could go on and on with duelling quotations. In short, what I found is that the field of AI is in turmoil. Either a huge amount of progress has been made, or almost none at all. Either we are within spitting distance of 'true' AI, or it is centuries away. AI will solve all our problems, put us all out of a job, destroy the human race, or cheapen our humanity. It's either a noble quest or 'summoning the demon'.

What This Book is About

This book arose from my attempt to understand the true state of affairs in artificial intelligence – what computers can do now, and what we can expect from them over the next decades. Hofstadter's provocative comments at the Google meeting were something of a wake-up call for me, as were the Google researchers' confident responses about AI's near-term future. In the chapters that follow, I try to sort out how far artificial intelligence has come, as well as elucidate its disparate – and sometimes conflicting – goals. In doing so, I consider how some of the most prominent AI systems actually work, and investigate how successful they are and where their limitations lie. I look at the extent to which computers can now do things that we believe to require high levels of intelligence – beating humans at the most intellectually demanding games, translating between languages, answering complex questions, navigating vehicles in challenging terrain. And I examine how they fare at the things we take for granted, the everyday tasks we humans perform without conscious thought: recognizing faces and objects in images, understanding spoken language and written text, and using the most basic common sense.

I also try to make sense of the broader questions that have fuelled debates about AI since its inception: What do we actually mean by 'general human' or even 'superhuman' intelligence? Is current AI close to this level, or even on a trajectory to get there? What are the dangers? What aspects of our intelligence do we most cherish, and to what extent would human-level AI challenge how we think about our own humanness? To use Hofstadter's terms, how terrified should we be?

This book is not a general survey or history of artificial intelligence. Rather, it is an in-depth exploration of some of the AI methods that probably affect your life, or will

soon, as well as the AI efforts that perhaps go furthest in challenging our sense of human uniqueness. My aim is for you to share in my own exploration and, like me, to come away with a clearer sense of what the field has accomplished and how much further there is to go before our machines can argue for their own humanity.

PART ONE

Background

CHAPTER 1

The Roots of Artificial Intelligence

Two Months and Ten Men at Dartmouth

The dream of creating an intelligent machine – one that is as smart as or smarter than humans – is centuries old but became part of modern science with the rise of digital computers. In fact, the ideas that led to the first programmable computers came out of mathematicians' attempts to understand human thought – particularly logic – as a mechanical process of 'symbol manipulation'. Digital computers are essentially symbol manipulators, pushing around combinations of the symbols 0 and 1. To pioneers of computing like Alan Turing and John von Neumann, there were strong analogies between computers and the human brain, and it seemed obvious to them that human intelligence could be replicated in computer programs.

Most people in artificial intelligence trace the field's official founding to a small workshop in 1956 at Dartmouth College organized by a young mathematician named John McCarthy.

In 1955, McCarthy, aged twenty-eight, joined the mathematics faculty at Dartmouth. As an undergraduate, he had learned a bit about both psychology and the nascent field of 'automata theory' (later to become computer science) and had become intrigued with the idea of creating a thinking machine. In graduate school in the mathematics department at Princeton, McCarthy had met a fellow student, Marvin Minsky, who shared his fascination with the potential of intelligent computers. After graduating, McCarthy had short-lived stints at Bell Labs and IBM, where he collaborated, respectively, with Claude Shannon, the inventor of information theory, and Nathaniel Rochester, a pioneering electrical engineer. Once at Dartmouth, McCarthy persuaded Minsky, Shannon and Rochester to help him organize 'a 2 month, 10 man study of artificial intelligence to be carried out during the summer of 1956'.[1] The term *artificial intelligence* was McCarthy's invention; he wanted to distinguish this field from a related effort called cybernetics.[2] McCarthy later admitted that no one really liked the name – after all, the goal was *genuine*, not 'artificial', intelligence – but 'I had to call it something, so I called it "Artificial Intelligence".'[3]

The four organizers submitted a proposal to the Rockefeller Foundation asking for funding for the summer workshop. The proposed study was, they wrote, based on 'the

conjecture that every aspect of learning or any other feature of intelligence can be in principle so precisely described that a machine can be made to simulate it'.[4] The proposal listed a set of topics to be discussed – natural-language processing, neural networks, machine learning, abstract concepts and reasoning, creativity – that have continued to define the field to the present day.

Even though the most advanced computers in 1956 were about a million times slower than today's smartphones, McCarthy and colleagues were optimistic that AI was in close reach: 'We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.'[5]

Obstacles soon arose that would be familiar to anyone organizing a scientific workshop today. The Rockefeller Foundation came through with only half the requested amount of funding. And it turned out to be harder than McCarthy had thought to persuade the participants to actually come and then stay, not to mention agree on anything. There were lots of interesting discussions but not a lot of coherence. As usual in such meetings, 'Everyone had a different idea, a hearty ego, and much enthusiasm for their own plan.'[6] However, the Dartmouth summer of AI did produce a few very important outcomes. The field itself was named, and its general goals were outlined. The soon-to-be 'big four' pioneers of the field – McCarthy, Minsky, Allen Newell and Herbert Simon – met and did some planning for the future. And for whatever reason, these four came out of the meeting with tremendous optimism for the field. In the early 1960s, McCarthy founded the Stanford Artificial Intelligence Project, with the 'goal of building a fully intelligent machine in a decade'.[7] Around the same time, the future Nobel laureate Herbert Simon predicted, 'Machines will be capable, within twenty years, of doing any work that a man can do.'[8] Soon after, Marvin Minsky, founder of the MIT AI Lab, forecasted that 'within a generation ... the problems of creating "artificial intelligence" will be substantially solved.'[9]

Definitions, and Getting On with It

None of these predicted events have yet come to pass. So how far do we remain from the goal of building a 'fully intelligent machine'? Would such a machine require us to reverse engineer the human brain in all its complexity, or is there a shortcut, a clever set of yet-unknown algorithms, that can produce what we recognize as full intelligence? What does 'full intelligence' even mean?

'Define your terms ... or we shall never understand one another.'[10] This admonition from the eighteenth-century philosopher Voltaire is a challenge for anyone talking about artificial intelligence, because its central notion – intelligence – remains so ill-defined. Marvin Minsky himself coined the phrase 'suitcase word'[11] for terms like *intelligence* and its many cousins, such as *thinking*, *cognition*, *consciousness* and *emotion*. Each is packed like a suitcase with a jumble of different meanings. *Artificial intelligence* inherits this packing problem, sporting different meanings in different contexts.

Most people would agree that humans are intelligent and specks of dust are not. Likewise, we generally believe that humans are more intelligent than worms. As for human intelligence, IQ is measured on a single scale, but we also talk about the different dimensions of intelligence: emotional, verbal, spatial, logical, artistic, social, and so forth. Thus, intelligence can be binary (something is or is not intelligent), on a continuum (one thing is more intelligent than another thing), or multidimensional (someone can have high verbal intelligence but low emotional intelligence). Indeed, the word *intelligence* is an over-packed suitcase, zip on the verge of breaking.

For better or worse, the field of AI has largely ignored these various distinctions. Instead, it has focused on two efforts: one scientific and one practical. On the scientific

side, AI researchers are investigating the mechanisms of 'natural' (that is, biological) intelligence by trying to embed it in computers. On the practical side, AI proponents simply want to create computer programs that perform tasks as well as or better than humans, without worrying about whether these programs are actually *thinking* in the way humans think. When asked if their motivations are practical or scientific, many AI people joke that it depends on where their funding currently comes from.

In a recent report on the current state of AI, a committee of prominent researchers defined the field as 'a branch of computer science that studies the properties of intelligence by synthesizing intelligence'.[12] A bit circular, yes. But the same committee also admitted that it's hard to define the field, and that may be a good thing: 'The lack of a precise, universally accepted definition of AI probably has helped the field to grow, blossom, and advance at an ever-accelerating pace.'[13] Furthermore, the committee notes, 'Practitioners, researchers, and developers of AI are instead guided by a rough sense of direction and an imperative to "get on with it".'

An Anarchy of Methods

At the 1956 Dartmouth workshop, different participants espoused divergent opinions about the correct approach to take to develop AI. Some people – generally mathematicians – promoted mathematical logic and deductive reasoning as the language of rational thought. Others championed inductive methods in which programs extract statistics from data and use probabilities to deal with uncertainty. Still others believed firmly in taking inspiration from biology and psychology to create brain-like programs. What you may find surprising is that the arguments among proponents of these various approaches persist to this day. And each approach has generated its own panoply of principles and techniques, fortified by speciality conferences and journals, with little communication among the subspecialities. A recent AI survey paper summed it up: 'Because we don't deeply understand intelligence or know how to produce general AI, rather than cutting off any avenues of exploration, to truly make progress we should embrace AI's "anarchy of methods".'[14]

But since the 2010s, one family of AI methods – collectively called deep learning (or deep neural networks) – has risen above the anarchy to become the dominant AI paradigm. In fact, in much of the popular media, the term *artificial intelligence* itself has come to mean 'deep learning'. This is an unfortunate inaccuracy, and I need to clarify the distinction. AI is a field that includes a broad set of approaches, with the goal of creating machines with intelligence. Deep learning is only one such approach. Deep learning is itself one method among many in the field of *machine learning*, a subfield of AI in which machines 'learn' from data or from their own 'experiences'. To better understand these various distinctions, it's important to understand a philosophical split that occurred early in the AI research community: the split between so-called symbolic and subsymbolic AI.

Symbolic AI

First let's look at *symbolic AI*. A symbolic AI program's knowledge consists of words or phrases (the 'symbols'), typically understandable to a human, along with rules by which the program can combine and process these symbols in order to perform its assigned task.

I'll give you an example. One early AI program was confidently called the General Problem Solver,[15] or GPS for short. (Sorry about the confusing acronym; the General Problem Solver pre-dated the Global Positioning System.) GPS could solve problems such as the 'Missionaries and Cannibals' puzzle, which you might have tackled yourself

as a child. In this well-known conundrum, three missionaries and three cannibals all need to cross a river, but their boat holds only two people. If at any time the (hungry) cannibals outnumber the (tasty-looking) missionaries on one side of the river ... well, you probably know what happens. How do all six get across the river intact?

The creators of the General Problem Solver, the cognitive scientists Herbert Simon and Allen Newell, had recorded several students 'thinking out loud' while solving this and other logic puzzles. Simon and Newell then designed their program to mimic what they believed were the students' thought processes.

I won't go into the details of how GPS worked, but its symbolic nature can be seen by the way the program's instructions were encoded. To set up the problem, a human would write code for GPS that looked something like this:

```
CURRENT STATE:
LEFT-BANK = [3 MISSIONARIES, 3 CANNIBALS, 1 BOAT]
RIGHT-BANK = [EMPTY]

DESIRED STATE:
LEFT-BANK = [EMPTY]
RIGHT-BANK = [3 MISSIONARIES, 3 CANNIBALS, 1 BOAT]
```

In English, these lines represent the fact that initially the left bank of the river 'contains' three missionaries, three cannibals and one boat, whereas the right bank doesn't contain any of these. The desired state represents the goal of the program – get everyone to the right bank of the river.

At each step in its procedure, GPS attempts to change its current state to make it more similar to the desired state. In its code, the program has 'operators' (in the form of subprograms) that can transform the current state into a new state and 'rules' that encode the constraints of the task. For example, there is an operator that moves some number of missionaries and cannibals from one side of the river to the other:

```
MOVE (#MISSIONARIES, #CANNIBALS, FROM-SIDE, TO-SIDE)
```

The words inside the parentheses are called arguments, and when the program runs, it replaces these words with numbers or other words. That is, #MISSIONARIES is replaced with the number of missionaries to move, #CANNIBALS with the number of cannibals to move, and FROM-SIDE and TO-SIDE are replaced with 'LEFT-BANK' or 'RIGHT-BANK', depending on which riverbank the missionaries and cannibals are to be moved from. Encoded into the program is the knowledge that the boat is moved along with the missionaries and cannibals.

Before being able to apply this operator with specific values replacing the arguments, the program must check its encoded rules; for example, the maximum number of people that can move at a time is two, and the operator cannot be used if it will result in cannibals outnumbering missionaries on a riverbank.

While these symbols represent human-interpretable concepts such as *missionaries*, *cannibals*, *boat* and *left bank*, the computer running this program of course has no knowledge of the meaning of these symbols. You could replace all occurrences of 'MISSIONARIES' with 'Z372B' or any other nonsense string, and the program would work in exactly the same way. This is part of what the term *General* refers to in *General Problem Solver*. To the computer, the 'meaning' of the symbols derives from the ways in which they can be combined, related to one another and operated on.

Advocates of the symbolic approach to AI argued that to attain intelligence in computers, it would not be necessary to build programs that mimic the brain. Instead, the argument goes, general intelligence can be captured entirely by the right kind of

symbol-processing program. Agreed, the workings of such a program would be vastly more complex than the Missionaries and Cannibals example, but it would still consist of symbols, combinations of symbols, and rules and operations on symbols. Symbolic AI of the kind illustrated by GPS ended up dominating the field for its first three decades, most notably in the form of *expert systems*, in which human experts devised rules for computer programs to use in tasks such as medical diagnosis and legal decision-making. There are several active branches of AI that still employ symbolic AI; I'll describe examples of it later, particularly in discussions of AI approaches to reasoning and common sense.

Subsymbolic AI: Perceptrons

Symbolic AI was originally inspired by mathematical logic as well as by the way people described their conscious thought processes. In contrast, *subsymbolic* approaches to AI took inspiration from neuroscience and sought to capture the sometimes *unconscious* thought processes underlying what some have called fast perception, such as recognizing faces or identifying spoken words. Subsymbolic AI programs do not contain the kind of human-understandable language we saw in the Missionaries and Cannibals example above. Instead, a subsymbolic program is essentially a stack of equations – a thicket of often hard-to-interpret operations on numbers. As I'll explain shortly, such systems are designed to learn from data how to perform a task.

An early example of a subsymbolic, brain-inspired AI program was the perceptron, invented in the late 1950s by the psychologist Frank Rosenblatt.[16] The term 'perceptron' may sound a bit 1950s science-fiction-y to our modern ears (as we'll see, it was soon followed by the 'cognitron' and the 'neocognitron'), but the perceptron was an important milestone in AI and was the influential great-grandparent of modern AI's most successful tool, deep neural networks.

Rosenblatt's invention of perceptrons was inspired by the way in which neurons process information. A neuron is a cell in the brain that receives electrical or chemical input from other neurons that connect to it. Roughly speaking, a neuron sums up all the inputs it receives from other neurons, and if the total sum reaches a certain threshold level, the neuron fires. Importantly, different connections (*synapses*) from other neurons to a given neuron have different strengths; in calculating the sum of its inputs, the given neuron gives more weight to inputs from stronger connections than inputs from weaker connections. Neuroscientists believe that adjustments to the strength of connections between neurons is a key part of how learning takes place in the brain.

To a computer scientist (or, in Rosenblatt's case, a psychologist), information processing in neurons can be simulated by a computer program – a perceptron – that has multiple numerical inputs and one output. The analogy between a neuron and a perceptron is illustrated in figure 1. Figure 1*a* shows a neuron, with its branching dendrites (fibres that carry inputs to the cell), cell body and axon (that is, output channel) labelled. Figure 1*b* shows a simple perceptron. Analogous to the neuron, the perceptron adds up its inputs, and if the resulting sum is equal to or greater than the perceptron's *threshold*, the perceptron outputs the value 1 (it 'fires'); otherwise it outputs the value 0 (it 'does not fire'). To simulate the different strengths of connections to a neuron, Rosenblatt proposed that a numerical *weight* be assigned to each of a perceptron's inputs; each input is multiplied by its weight before being added to the sum. A perceptron's *threshold* is simply a number set by the programmer (or, as we'll see, learned by the perceptron itself).

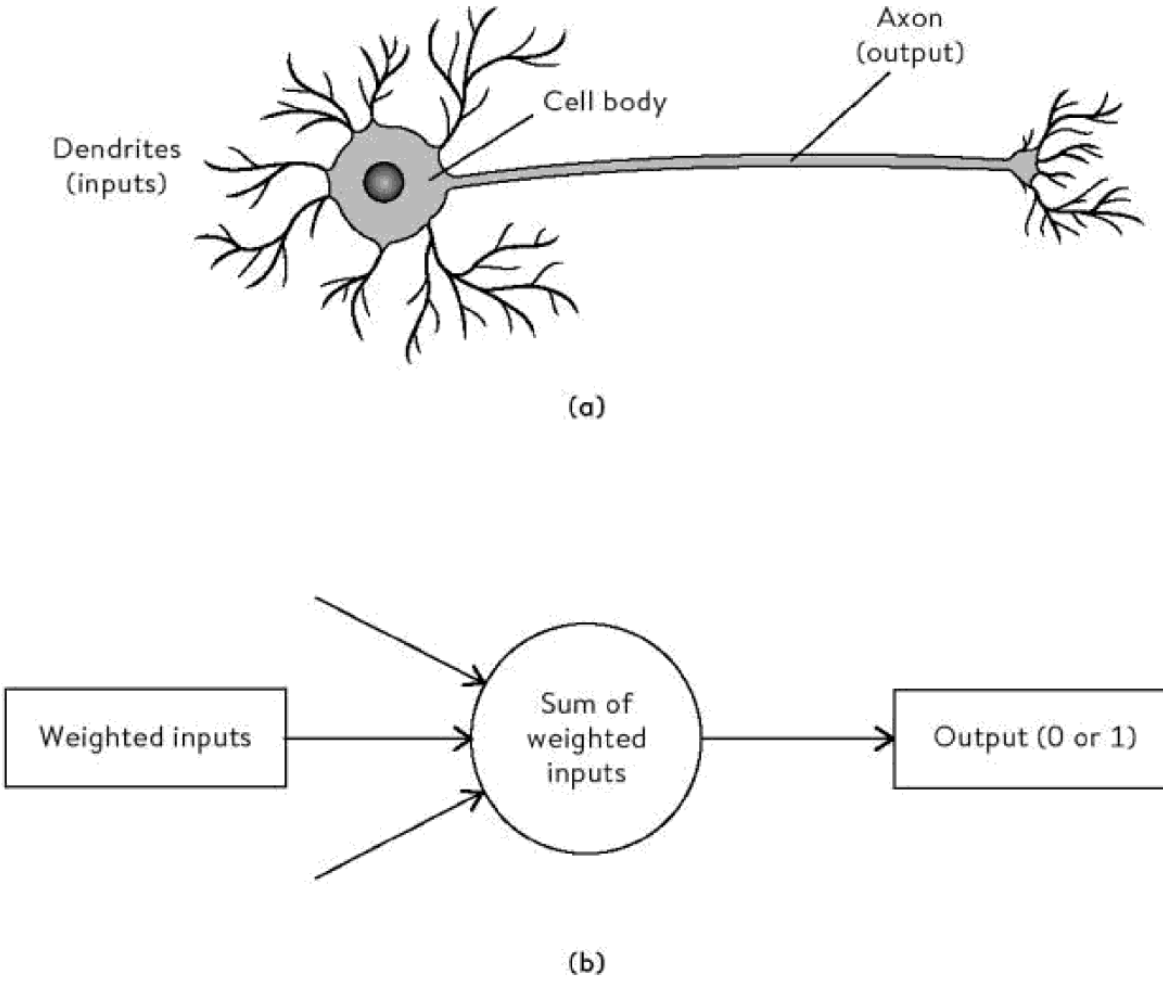

Figure 1
a, a neuron in the brain; *b*, a simple perceptron

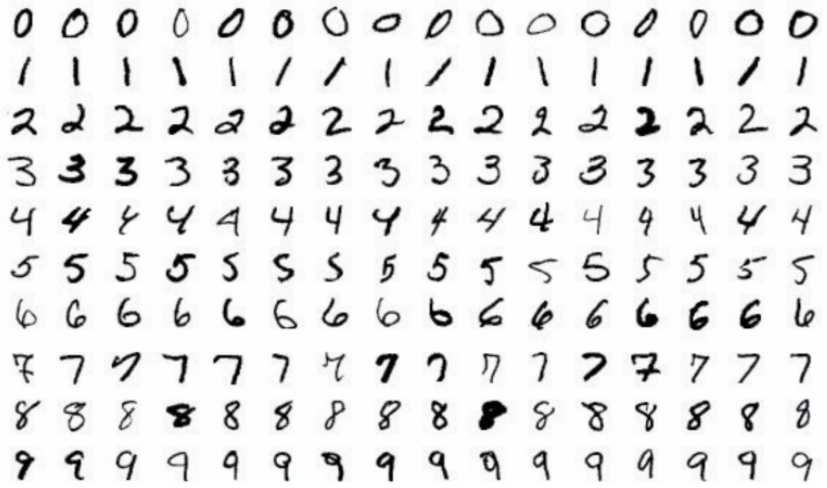

Figure 2
Examples of handwritten digits

In short, a perceptron is a simple program that makes a yes-or-no (1 or 0) decision based on whether the sum of its weighted inputs meets a threshold value. You probably make some decisions like this in your life. For example, you might get input from several friends on how much they liked a particular movie, but you trust some of those friends' taste in movies more than others'. If the total amount of 'friend enthusiasm' – giving more weight to your more trusted friends – is high enough (that is, greater than some unconscious threshold), you decide to go to the movie. This is how a perceptron would decide about movies, if only it had friends.

Inspired by networks of neurons in the brain, Rosenblatt proposed that networks of perceptrons could perform visual tasks such as recognizing faces and objects. To get a flavour of how that might work, let's explore how a perceptron might be used for a particular visual task: recognizing handwritten digits like those in figure 2.

In particular, let's design a perceptron to be an 8 detector – that is, to output a 1 if its inputs are from an image depicting an 8, and to output a 0 if the image depicts some other digit. Designing such a detector requires us to (1) figure out how to turn an image into a set of numerical inputs, and (2) determine numbers to use for the perceptron's weights and threshold, so that it will give the correct output (1 for 8s, 0 for other digits). I'll go into some detail here because many of the same ideas will arise later in my discussions of neural networks and their applications in computer vision.

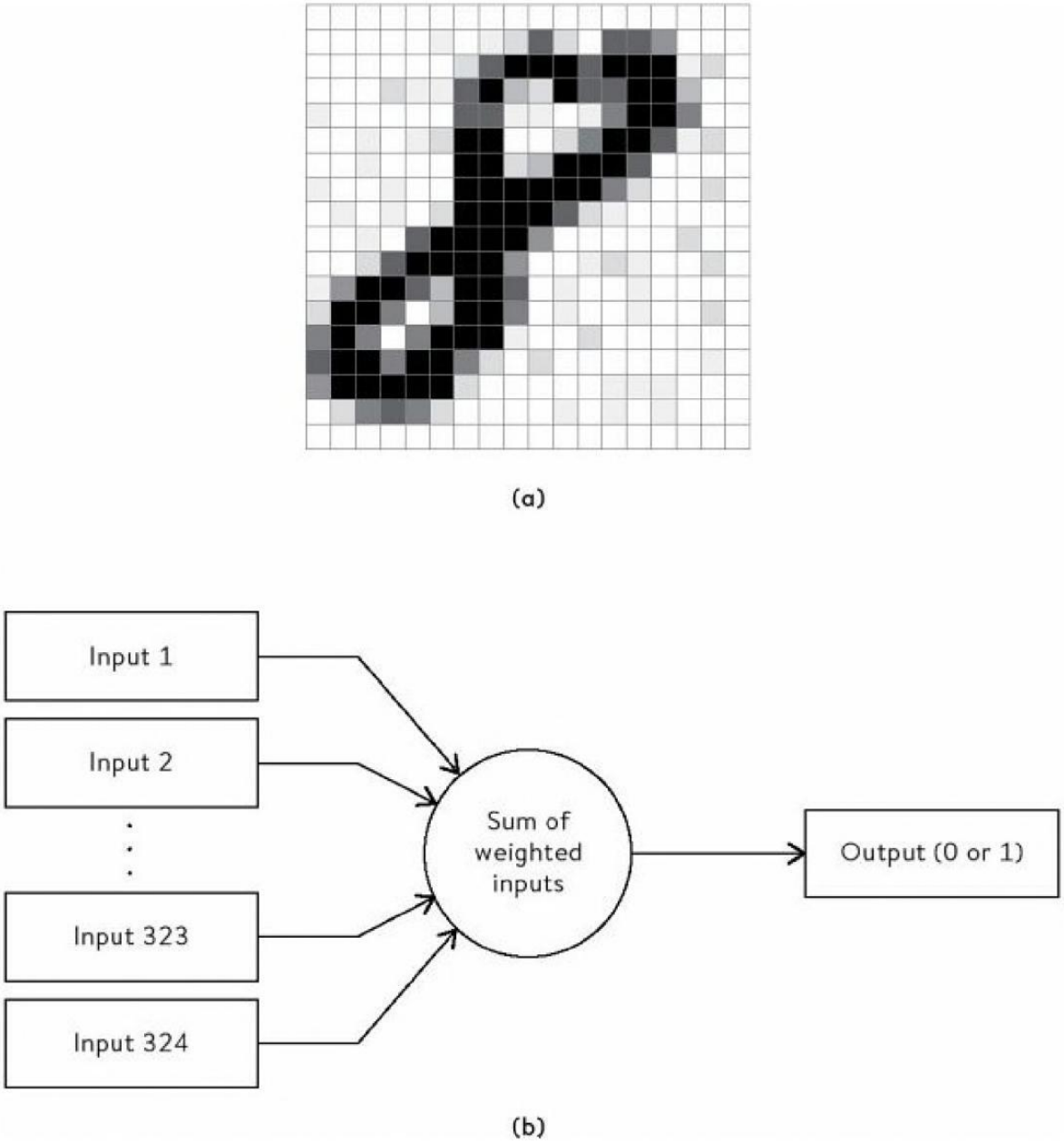

Figure 3
An illustration of a perceptron that recognizes handwritten 8s. Each pixel in the 18 × 18-pixel image corresponds to an input to the perceptron, yielding 324 (= 18 × 18) inputs.

Our Perceptron's Inputs

Figure 3*a* shows an enlarged handwritten 8. Each grid square is a pixel with a numerical 'intensity' value: white squares have an intensity of 0, black squares have an intensity of 1 and grey squares are in between. Let's assume that the images we give to our perceptron have been adjusted to be the same size as this one: 18 × 18 pixels. Figure 3*b* illustrates a perceptron for recognizing 8s. This perceptron has 324 (that is, 18 × 18) inputs, each of which corresponds to one of the pixels in the 18 × 18 grid. Given an image like the one in figure 3*a*, each of the perceptron's inputs is set to the corresponding pixel's intensity. Each of the inputs would have its own weight value (not shown in the figure).

Learning the Perceptron's Weights and Threshold

Unlike the symbolic General Problem Solver system that I described earlier, a perceptron doesn't have any explicit rules for performing its task; all of its 'knowledge' is encoded in the numbers making up its weights and threshold. In his various papers, Rosenblatt showed that given the correct weight and threshold values, a perceptron like the one in figure 3*b* can perform fairly well on perceptual tasks such as recognizing simple handwritten digits. But how, exactly, can we determine the correct weights and threshold for a given task? Again, Rosenblatt proposed a brain-inspired answer: the perceptron should *learn* these values on its own. And how is it supposed to learn the correct values? Like the behavioural psychology theories popular at the time, Rosenblatt's idea was that perceptrons should learn via *conditioning*. Inspired in part by the behaviourist psychologist B. F. Skinner, who trained rats and pigeons to perform tasks by giving them positive and negative reinforcement, Rosenblatt's idea was that the perceptron should similarly be *trained* on examples: it should be rewarded when it fires correctly and punished when it errs. This form of conditioning is now known in AI as supervised learning. During training, the learning system is given an example, it produces an output, and it is then given a 'supervision signal', which tells how much the system's output differs from the correct output. The system then uses this signal to adjust its weights and threshold.

The concept of supervised learning is a key part of modern AI, so it's worth discussing in more detail. Supervised learning typically requires a large set of positive examples (for instance, a collection of 8s written by different people) and negative examples (for instance, a collection of other handwritten digits, not including 8s). Each example is *labelled* by a human with its category – here, 8 or not-8. This label will be used as the supervision signal. Some of the positive and negative examples are used to *train* the system; these are called the *training set*. The remainder – the *test set* – is used to evaluate the system's performance after it has been trained, to see how well it has learned to answer correctly in general, not just on the training examples.

Perhaps the most important term in computer science is *algorithm*, which refers to a 'recipe' of steps a computer can take in order to solve a particular problem. Frank Rosenblatt's primary contribution to AI was his design of a specific algorithm, called the perceptron-learning algorithm, by which a perceptron could be trained from examples to determine the weights and threshold that would produce correct answers. Here's how it works. Initially, the weights and threshold are set to random values between -1 and 1. In our example, the weight on the first input might be set to 0.2, the weight on the second input set to -0.6, and so on, and the threshold set to 0.7. A computer program called a random-number generator can easily generate these initial values.

Now we can start the training process. The first training example is given to the perceptron; at this point, the perceptron doesn't see the correct category label. The perceptron multiplies each input by its weight, sums up all the results, compares the sum with the threshold, and outputs either 1 or 0. Here, the output 1 means a guess of 8, and the output 0 means a guess of not-8. Now, the training process compares the perceptron's output with the correct answer given by the human-provided label (that is, 8 or not-8). If the perceptron is correct, the weights and threshold don't change. But if the perceptron is wrong, the weights and threshold are changed a little bit, making the perceptron's sum on this training example closer to producing the right answer. Moreover, the amount each weight is changed depends on its associated input value; that is, the blame for the error is meted out depending on which inputs had the most impact. For example, in the 8 of figure 3*a*, the higher-intensity (here, black) pixels would have the most impact, and the pixels with 0 intensity (here, white) would have no impact. (For interested readers, I have included some mathematical details in the Notes.[17])

The whole process is repeated for the next training example. The training process goes through all the training examples multiple times, modifying the weights and threshold a little bit each time the perceptron makes an error. Just as the psychologist B. F. Skinner found when training pigeons, it's better to learn gradually over many trials; if the weights and threshold are changed too much on any one trial, then the system might end up learning the wrong thing (such as an overgeneralization that 'the bottom and top halves of an 8 are always equal in size'). After many repetitions on each training example, the system eventually (we hope) settles on a set of weights and a threshold that result in correct answers for all the training examples. At that point, we can evaluate the perceptron on the test examples to see how it performs on images it hasn't been trained on.

An 8 detector is useful if you care only about 8s. But what about recognizing other digits? It's fairly straightforward to extend our perceptron to have ten outputs, one for each digit. Given an example handwritten digit, the output corresponding to that digit should be 1, and all the other outputs should be 0. This extended perceptron can learn all of its weights and thresholds using the perceptron-learning algorithm; the system just needs enough examples.

Rosenblatt and others showed that networks of perceptrons could learn to perform relatively simple perceptual tasks; moreover, Rosenblatt proved mathematically that for a certain, albeit very limited, class of tasks, perceptrons with sufficient training could, in principle, learn to perform these tasks without error. What wasn't clear was how well perceptrons could perform on more general AI tasks. This uncertainty didn't seem to stop Rosenblatt and his funders at the Office of Naval Research from making ridiculously optimistic predictions about their algorithm. Reporting on a press conference Rosenblatt held in July 1958, *The New York Times* featured this recap:

> The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself, and be conscious of its existence. Later perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech and writing in another language, it was predicted.[18]

Yes, even at its beginning, AI suffered from a hype problem. I'll talk more about the unhappy results of such hype shortly. But for now, I want to use perceptrons to highlight a major difference between symbolic and subsymbolic approaches to AI.

The fact that a perceptron's 'knowledge' consists of a set of numbers – namely, the weights and threshold it has learned – means that it is hard to uncover the rules the perceptron is using in performing its recognition task. The perceptron's rules are not

symbolic; unlike the General Problem Solver's symbols, such as LEFT-BANK, #MISSIONARIES and MOVE, a perceptron's weights and threshold don't stand for particular concepts. It's not easy to translate these numbers into rules that are understandable by humans. The situation gets much worse with modern neural networks that have millions of weights.

One might make a rough analogy between perceptrons and the human brain. If I could open up your head and watch some subset of your hundred billion neurons firing, I would probably not get any insight into what you were thinking or the 'rules' you used to make a particular decision. However, the human brain has given rise to language, which allows you to use symbols (words and phrases) to tell me – often imperfectly – what your thoughts are about or why you did a certain thing. In this sense, our neural firings can be considered *subsymbolic*, in that they underlie the symbols our brains somehow create. Perceptrons, as well as more complicated networks of simulated neurons, have been dubbed 'subsymbolic' in analogy to the brain. Their advocates believe that to achieve artificial intelligence, language-like symbols and the rules that govern symbol processing cannot be programmed directly, as was done in the General Problem Solver, but must emerge from neural-like architectures similar to the way that intelligent symbol processing emerges from the brain.

The Limitations of Perceptrons

After the 1956 Dartmouth meeting, the symbolic camp dominated the AI landscape. In the early 1960s, while Rosenblatt was working avidly on the perceptron, the big four 'founders' of AI, all strong devotees of the symbolic camp, had created influential – and well-funded – AI laboratories: Marvin Minsky at MIT, John McCarthy at Stanford, and Herbert Simon and Allen Newell at Carnegie Mellon. (Remarkably, these three universities remain to this day among the most prestigious places to study AI.) Minsky, in particular, felt that Rosenblatt's brain-inspired approach to AI was a dead end, and moreover was stealing away research dollars from more worthy symbolic AI efforts.[19] In 1969, Minsky and his MIT colleague Seymour Papert published a book, *Perceptrons*,[20] in which they gave a mathematical proof showing that the types of problems a perceptron could solve *perfectly* were very limited and that the perceptron-learning algorithm would not do well in scaling up to tasks requiring a large number of weights and thresholds.

Minsky and Papert pointed out that if a perceptron is augmented by adding a 'layer' of simulated neurons, the types of problems that the device can solve is, in principle, much broader.[21] A perceptron with such an added layer is called a multilayer neural network. Such networks form the foundations of much of modern AI; I'll describe them in detail in the next chapter. But for now, I'll note that at the time of Minsky and Papert's book, multilayer neural networks were not broadly studied, largely because there was no general algorithm, analogous to the perceptron-learning algorithm, for learning weights and thresholds.

The limitations Minsky and Papert proved for simple perceptrons were already known to people working in this area.[22] Frank Rosenblatt himself had done extensive work on multilayer perceptrons and recognized the difficulty of training them.[23] It wasn't Minsky and Papert's mathematics that put the final nail in the perceptron's coffin; rather, it was their speculation on multilayer neural networks:

> [The perceptron] has many features to attract attention: its linearity; its intriguing learning theorem; its clear paradigmatic simplicity as a kind of parallel computation. There is no reason to suppose that any of these virtues

> carry over to the many-layered version. Nevertheless, we consider it to be an important research problem to elucidate (or reject) our intuitive judgment that the extension is sterile.[24]

Ouch. In today's vernacular that final sentence might be termed 'passive-aggressive'. Such negative speculations were at least part of the reason that funding for neural network research dried up in the late 1960s, at the same time that symbolic AI was flush with government dollars. In 1971, at the age of forty-three, Frank Rosenblatt died in a boating accident. Without its most prominent proponent, and without much government funding, research on perceptrons and other subsymbolic AI methods largely halted, except in a few isolated academic groups.

AI Winter

In the meantime, proponents of symbolic AI were writing grant proposals promising impending breakthroughs in areas such as speech and language understanding, commonsense reasoning, robot navigation and autonomous vehicles. By the mid-1970s, while some very narrowly focused expert systems were successfully deployed, the more general AI breakthroughs that had been promised had not materialized.

The funding agencies noticed. Two reports, solicited respectively by the Science Research Council in the UK and the Department of Defense in the United States, reported very negatively on the progress and prospects for AI research. The UK report in particular acknowledged that there was promise in the area of specialized expert systems – 'programs written to perform in highly specialized problem domains, when the programming takes very full account of the results of human experience and human intelligence within the relevant domain' – but concluded that the results to date were 'wholly discouraging about general-purpose programs seeking to mimic the problem-solving aspects of human [brain] activity over a rather wide field. Such a general-purpose program, the coveted long-term goal of AI activity, seems as remote as ever.'[25] This report led to a sharp decrease in government funding for AI research in the UK; similarly, the Department of Defense drastically cut funding for basic AI research in the United States.

This was an early example of a repeating cycle of bubbles and crashes in the field of AI. The two-part cycle goes like this. Phase 1: New ideas create a lot of optimism in the research community. Results of imminent AI breakthroughs are promised, and often hyped in the news media. Money pours in from government funders and venture capitalists for both academic research and commercial start-ups. Phase 2: The promised breakthroughs don't occur, or are much less impressive than promised. Government funding and venture capital dry up. Start-up companies fold, and AI research slows. This pattern became familiar to the AI community: 'AI spring', followed by overpromising and media hype, followed by 'AI winter'. This has happened, to various degrees, in cycles of five to ten years. When I got out of graduate school in 1990, the field was in one of its winters and had garnered such a bad image that I was even advised to leave the term 'artificial intelligence' off my job applications.

Easy Things are Hard

The cold AI winters taught practitioners some important lessons. The simplest lesson was noted by John McCarthy, fifty years after the Dartmouth conference: 'AI was harder than we thought.'[26] Marvin Minsky pointed out that in fact AI research had uncovered a paradox: 'Easy things are hard.' The original goals of AI – computers that could converse with us in natural language, describe what they saw through their

camera eyes, learn new concepts after seeing only a few examples – are things that young children can easily do, but, surprisingly, these 'easy things' have turned out to be harder for AI to achieve than diagnosing complex diseases, beating human champions at chess and Go, and solving complex algebraic problems. As Minsky went on, 'In general, we're least aware of what our minds do best.'[27] The attempt to create artificial intelligence has, at the very least, helped elucidate how complex and subtle are our own minds.

CHAPTER 2

Neural Networks and the Ascent of Machine Learning

Spoiler alert: Multilayer neural networks – the extension of perceptrons that was dismissed by Minsky and Papert as likely to be 'sterile' – have instead turned out to form the foundation of much of modern artificial intelligence. Because they are the basis of several of the methods I'll describe in later chapters, I'll take some time here to describe how these networks work.

Multilayer Neural Networks

A *network* is simply a set of elements that are connected to one another in various ways. We're all familiar with social networks, in which the elements are people, and computer networks, in which the elements are, naturally, computers. In neural networks, the elements are simulated neurons akin to the perceptrons I described in the previous chapter.

In figure 4, I've sketched a simple multilayer neural network, designed to recognize handwritten digits. The network has two rows (*layers*) of perceptron-like simulated neurons (circles). For simplicity (and probably to the relief of any neuroscientists reading this), I'll use the term *unit* instead of *simulated neuron* to describe the elements of this network. Like the 8-detecting perceptron from chapter 1, the network in figure 4 has 324 (18 × 18) inputs, each of which is set to the intensity value of the corresponding pixel in the input image. But unlike the perceptron, this network has a layer of three so-called hidden units, along with its layer of ten output units. Each output unit corresponds to one of the possible digit categories.

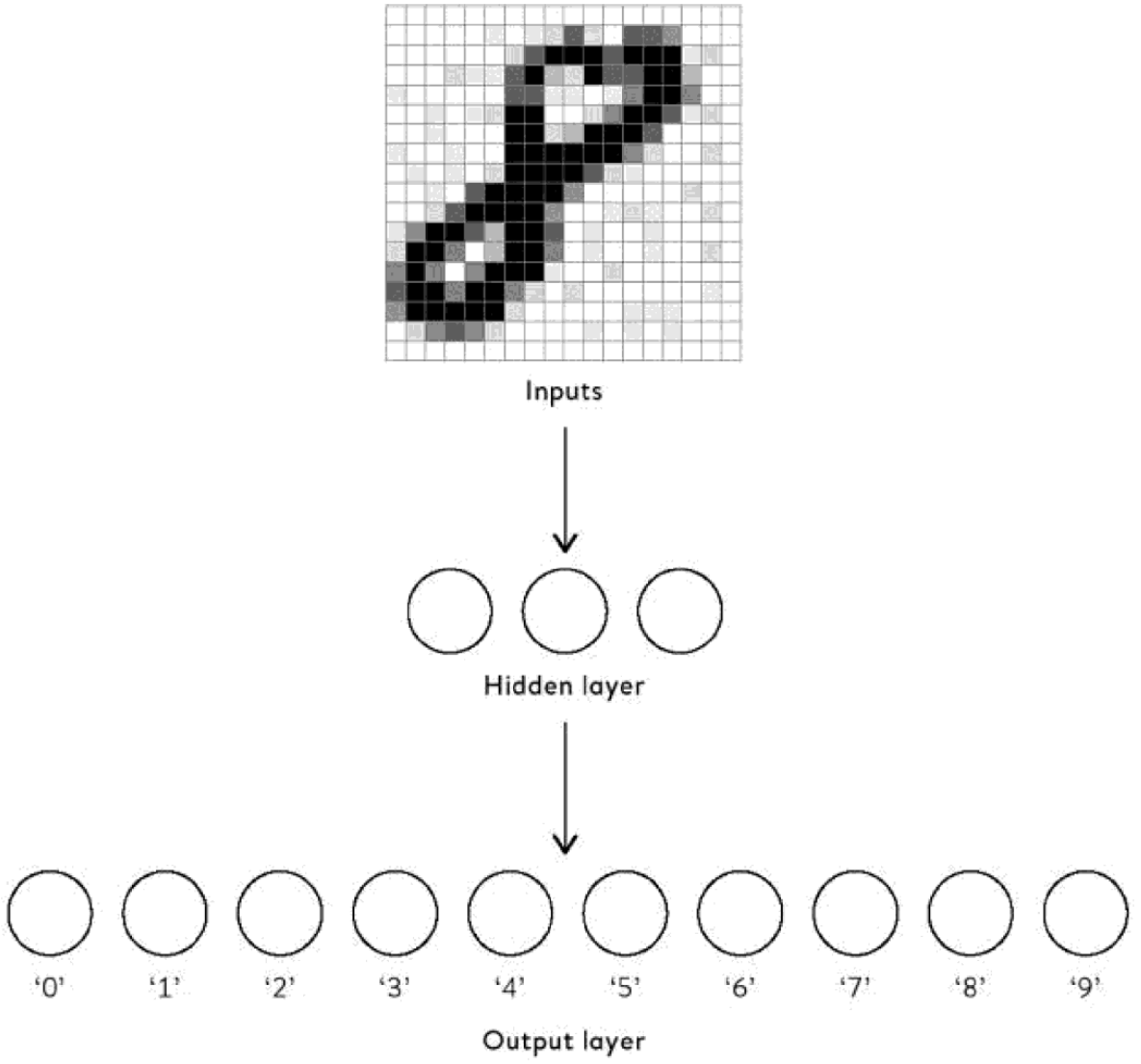

Figure 4
A two-layer neural network for recognizing handwritten digits

The arrows signify that each input has a weighted connection to each hidden unit, and each hidden unit has a weighted connection to each output unit. The mysterious-sounding term *hidden unit* comes from the neural network literature; it simply means a non-output unit. A better name might have been *interior unit*.

Think of the structure of your brain, in which some neurons directly control 'outputs' such as your muscle movements but most neurons simply communicate with other neurons. These could be called the brain's hidden neurons.

The network shown in figure 4 is referred to as 'multilayered' because it has two layers of units (hidden and output) instead of just an output layer. In principle, a multilayer network can have multiple layers of hidden units; networks that have more than one layer of hidden units are called deep networks. The 'depth' of a network is simply its number of hidden layers. I'll have much more to say about deep networks in upcoming chapters.

Similar to perceptrons, each unit here multiplies each of its inputs by the weight on that input's connection and then sums the results. However, unlike in a perceptron, a unit here doesn't simply 'fire' or 'not fire' (that is, produce 1 or 0) based on a threshold; instead, each unit uses its sum to compute a number between 0 and 1 that is called the unit's 'activation'. If the sum that a unit computes is low, the unit's activation is close to 0; if the sum is high, the activation is close to 1. (For interested readers, I've included some of the mathematical details in the Notes.[1])

To process an image such as the handwritten 8 in figure 4, the network performs its computations layer by layer, from top to bottom. Each hidden unit computes its activation value; these activation values then become the inputs for the output units, which then compute their own activations. In the network of figure 4, the activation of an output unit can be thought of as the network's confidence that it is 'seeing' the corresponding digit; the digit category with the highest confidence can be taken as the network's answer – its *classification*.

In principle, a multilayer neural network can learn to use its hidden units to recognize more abstract features (for example, visual shapes, such as the top and bottom 'circles' on a handwritten 8) than the simple features (for example, pixels) encoded by the input. In general, it's hard to know ahead of time how many layers of hidden units are needed, or how many hidden units should be included in a layer, for a network to perform well on a given task. Most neural network researchers use a form of trial and error to find the best settings.

Learning via Back-Propagation

In their book *Perceptrons*, Minsky and Papert were sceptical that a successful algorithm could be designed for learning the weights in a multilayer neural network. Their scepticism (along with doubts from others in the symbolic AI community) was largely responsible for the sharp decrease in funding for neural network research in the 1970s. But despite the chilling effect of Minsky and Papert's book on the field, a small core of neural network researchers persisted, especially in Frank Rosenblatt's own field of cognitive psychology. And by the late 1970s and early '80s, several of these groups had definitively rebutted Minsky and Papert's speculations on the 'sterility' of multilayer neural networks by developing a general learning algorithm – called back-propagation – for training these networks.

As its name implies, back-propagation is a way to take an error observed at the output units (for example, a high confidence for the wrong digit in the example of figure 4) and to 'propagate' the blame for that error backwards (in figure 4, this would be from bottom to top) so as to assign proper blame to each of the weights in the network. This allows back-propagation to determine how much to change each weight

in order to reduce the error. *Learning* in neural networks simply consists in gradually modifying the weights on connections so that each output's error gets as close to 0 as possible on all training examples. While the mathematics of back-propagation is beyond the scope of my discussion here, I've included some details in the Notes.[2]

Back-propagation will work (in principle at least) no matter how many inputs, hidden units or output units your neural network has. While there is no mathematical guarantee that back-propagation will settle on the correct weights for a network, in practice it has worked very well on many tasks that are too hard for simple perceptrons. For example, I trained both a perceptron and a two-layer neural network, each with 324 inputs and 10 outputs, on the handwritten-digit-recognition task, using sixty thousand examples, and then tested how well each was able to recognize ten thousand new examples. The perceptron was correct on about 80 per cent of the new examples, whereas the neural network, with 50 hidden units, was correct on a whopping 94 per cent of those new examples. Kudos to the hidden units! But what exactly has the neural network learned that allowed it to soar past the perceptron? I don't know. It's possible that I could find a way to visualize the neural network's 16,700 weights[3] to get some insight into its performance, but I haven't done so, and in general it's not at all easy to understand how these networks make their decisions.

It's important to note that while I've used the example of handwritten digits, neural networks can be applied not just to images but to any kind of data. Neural networks have been applied in areas as diverse as speech recognition, stock-market prediction, language translation and music composition.

Connectionism

In the 1980s, the most visible group working on neural networks was a team at the University of California at San Diego headed by two psychologists, David Rumelhart and James McClelland. What we now call neural networks were then generally referred to as connectionist networks, where the term *connectionist* refers to the idea that knowledge in these networks resides in weighted *connections* between units. The team led by Rumelhart and McClelland is known for writing the so-called bible of connectionism – a two-volume treatise, published in 1986, called *Parallel Distributed Processing*. In the midst of an AI landscape dominated by symbolic AI, the book was a pep talk for the subsymbolic approach, arguing that 'people are smarter than today's computers because the brain employs a basic computational architecture that is more suited to ... the natural information-processing tasks that people are so good at,' for example, 'perceiving objects in natural scenes and noting their relations ... understanding language, and retrieving contextually appropriate information from memory'.[4] The authors speculated that 'symbolic systems such as those favored by Minsky and Papert'[5] would not be able to capture these humanlike abilities.

Indeed, by the mid-1980s, expert systems – symbolic AI approaches that rely on humans to create rules that reflect expert knowledge of a particular domain – were increasingly revealing themselves to be *brittle*: that is, error-prone and often unable to generalize or adapt when presented with new situations. In analysing the limitations of these systems, researchers were discovering how much the human experts writing the rules actually rely on subconscious knowledge – what you might call common sense – in order to act intelligently. This kind of common sense could not easily be captured in programmed rules or logical deduction, and the lack of it severely limited any broad application of symbolic AI methods. In short, after a cycle of grand promises, immense funding and media hype, symbolic AI was facing yet another AI winter.

CHAPTER 3

AI Spring

Spring Fever

Have you ever taken a video of your cat and uploaded it to YouTube? If so, you are not alone. More than a billion videos have been uploaded to YouTube, and a lot of them feature cats. In 2012, an AI team at Google constructed a multilayer neural network with over a billion weights that 'viewed' millions of random YouTube videos while it adjusted these weights in order to successively compress, and then decompress, selected frames from the videos. The Google researchers didn't tell the system to learn about any particular objects, but after a week of training, when they probed the innards of the network, what did they find? A 'neuron' (unit) that seemed to encode cats.[1] This self-taught cat-recognition machine was one of a series of impressive AI feats that have captured the public's attention over the last decade. Most of these achievements rely on a set of neural network algorithms known as deep learning.

Until recently, AI's popular image came largely from the many movies and TV shows in which it played a starring role; think *2001: A Space Odyssey* or *The Terminator*. Real-world AI wasn't very noticeable in our everyday lives or mainstream media. If you came of age in the 1990s or earlier, you might recall frustrating encounters with customer service speech-recognition systems, the robotic word-learning toy Furby, or Microsoft's annoying and ill-fated Clippy, the paper-clip virtual assistant. Full-blown AI didn't seem imminent.

Maybe this is why so many people were shocked and upset when, in 1997, IBM's Deep Blue chess-playing system defeated the world chess champion Garry Kasparov. This event so stunned Kasparov that he accused the IBM team of cheating; he assumed that for the machine to play so well, it must have received help from human experts.[2] (In a nice bit of irony, during the 2006 World Chess Championship matches the tables were turned, with one player accusing the other of cheating by receiving help from a computer chess program.[3])

Our collective human angst over Deep Blue quickly receded. We accepted that chess could yield to brute-force machinery; playing chess well, we allowed, didn't require general intelligence after all. This seems to be a common response when computers surpass humans on a particular task; we conclude that the task doesn't actually require intelligence. As John McCarthy lamented, 'As soon as it works, no one calls it AI any more.'[4]

However, by the mid-2000s and beyond, a more pervasive succession of AI accomplishments started sneaking up on us and then proliferating at a dizzying pace. Google launched its automated language-translation service, Google Translate. It wasn't perfect, but it worked surprisingly well, and it has since improved significantly. Shortly thereafter, Google's self-driving cars showed up on the roads of Northern California, careful and timid, but commuting on their own in full traffic. Virtual assistants such as Apple's Siri and Amazon's Alexa were installed on our phones and in our homes and could deal with many of our spoken requests. YouTube started providing impressively accurate automated subtitles for videos, and Skype offered simultaneous translation between languages in video calls. Suddenly Facebook could recognize your face eerily well in uploaded photos, and the photo-sharing website Flickr began automatically labelling photos with text describing their content.

In 2011, IBM's Watson program roundly defeated human champions on television's *Jeopardy!* game show, adroitly interpreting pun-laden clues and prompting its challenger Ken Jennings to 'welcome our new computer overlords'. Just five years later, millions of internet viewers were introduced to the complex game of Go, a long-time grand challenge for AI, when a program called AlphaGo stunningly defeated one of the world's best players in four out of five games.

The buzz over artificial intelligence was quickly becoming deafening, and the commercial world took notice. All of the largest technology companies have poured billions of dollars into AI research and development, either hiring AI experts directly or acquiring smaller start-up companies for the sole purpose of grabbing ('acqui-hiring') their talented employees. The potential of being acquired, with its promise of instant millionaire status, has fuelled a proliferation of start-ups, often founded and run by former university professors, each with his or her own twist on AI. As the technology journalist Kevin Kelly observed, 'The business plans of the next 10,000 startups are easy to forecast: Take X and add AI.'[5] And, crucially, for nearly all of these companies, AI has meant 'deep learning'.

AI spring is once again in full bloom.

AI: Narrow and General, Weak and Strong

Like every AI spring before it, our current one features experts predicting that 'general AI' – AI that equals or surpasses humans in most ways – will be here soon. 'Human level AI will be passed in the mid-2020s,'[6] predicted Shane Legg, co-founder of Google DeepMind, in 2016. A year earlier, Facebook's CEO Mark Zuckerberg declared, 'One of our goals for the next five to 10 years is to basically get better than human level at all of the primary human senses: vision, hearing, language, general cognition.'[7] The AI philosophers Vincent Müller and Nick Bostrom published a 2013 poll of AI researchers in which many assigned a 50 per cent chance of human-level AI by the year 2040.[8]

While much of this optimism is based on the recent successes of deep learning, these programs – like *all* instances of AI to date – are still examples of what is called 'narrow' or 'weak' AI. These terms are not as derogatory as they sound; they simply refer to a system that can perform only one narrowly defined task (or a small set of related tasks). AlphaGo is possibly the world's best Go player, but it can't do anything else; it can't even play checkers, tic-tac-toe or Candy Land. Google Translate can render an English movie review into Chinese, but it can't tell you if the reviewer liked the movie or not, and it certainly can't watch and review the movie itself.

The terms *narrow* and *weak* are used to contrast with *strong*, *human-level*, *general* or *full-blown* AI (sometimes called AGI, or artificial general intelligence) – that is, the AI that we see in movies, that can do most everything we humans can do, and possibly much more. General AI might have been the original goal of the field, but achieving it

has turned out to be much harder than expected. Over time, efforts in AI have become focused on particular well-defined tasks – speech recognition, chess playing, autonomous driving, and so on. Creating machines that perform such functions is useful and often lucrative, and it could be argued that each of these tasks individually requires 'intelligence'. But no AI program has been created yet that could be called intelligent in any general sense. A recent appraisal of the field stated this well: 'A pile of narrow intelligences will never add up to a general intelligence. General intelligence isn't about the number of abilities, but about the integration between those abilities.'[9]

But wait. Given the rapidly increasing pile of narrow intelligences, how long will it be before someone figures out how to integrate them and produce all of the broad, deep and subtle features of human intelligence? Do we believe the cognitive scientist Steven Pinker, who thinks all this is business as usual? 'Human-level AI is still the standard fifteen to twenty-five years away, just as it always has been, and many of its recently touted advances have shallow roots,' Pinker declared.[10] Or should we pay more attention to the AI optimists, who are certain that this time around, this AI spring, things will be different?

Not surprisingly, in the AI research community there is considerable controversy over what human-level AI would entail. How can we know if we have succeeded in building such a 'thinking machine'? Would such a system be required to have *consciousness* or *self-awareness* in the way humans do? Would it need to *understand* things in the same way a human understands them? Given that we're talking about a machine here, would we be more correct to say it is 'simulating thought', or could we say it is truly thinking?

Could Machines Think?

Such philosophical questions have dogged the field of AI since its inception. Alan Turing, the British mathematician who in the 1930s sketched out the first framework for programmable computers, published a paper in 1950 asking what we might mean when we ask, 'Can machines think?' After proposing his famous 'imitation game' (now called the Turing test – more on this in a bit), Turing listed nine possible objections to the prospect of a machine actually thinking, all of which he tried to refute. These imagined objections range from the theological – 'Thinking is a function of man's immortal soul. God has given an immortal soul to every man and woman, but not to any other animal or to machines. Hence no animal or machine can think' – to the parapsychological, something along the lines of 'Humans can use telepathy to communicate while machines cannot.' Strangely enough, Turing judged this last argument as 'quite a strong one', because 'the statistical evidence, at least for telepathy, is overwhelming.'

From the vantage of many decades, my own vote for the strongest of Turing's possible arguments is the 'argument from consciousness', which he summarizes by quoting the neurologist Geoffrey Jefferson:

> Not until a machine can write a sonnet or compose a concerto because of thoughts and emotions felt, and not by the chance fall of symbols, could we agree that machine equals brain – that is, not only write it but know that it had written it. No mechanism could feel (and not merely artificially signal, an easy contrivance) pleasure at its successes, grief when its valves fuse, be warmed by flattery, be made miserable by its mistakes, be charmed by sex, be angry or depressed when it cannot get what it wants.[11]

Note that this argument is saying the following: (1) Only when a machine *feels* things and is aware of its own actions and feelings – in short, is conscious – could we consider

it actually *thinking*, and (2) No machine could ever do this. Ergo, no machine could ever *actually* think.

I think it's a strong argument, even though I don't agree with it. It resonates with our intuitions about what machines are and how they are limited. Over the years, I've talked with any number of friends, relatives and students about the possibility of machine intelligence, and this is the argument many of them stand by. For example, I was recently talking with my mother, a retired lawyer, after she had read a *New York Times* article about advances in the Google Translate program:

> Mom: The problem with people in the field of AI is that they anthropomorphize so much!
> Me: What do you mean, anthropomorphize?
> Mom: The language they use implies that machines might be able to actually think, rather than to just simulate thinking.
> Me: What's the difference between 'actually thinking' and 'simulating thinking'?
> Mom: Actual thinking is done with a brain, and simulating is done with computers.
> Me: What's so special about a brain that it allows 'actual' thinking? What's missing in computers?
> Mom: I don't know. I think there's a human quality to thinking that can't ever be completely mimicked by computers.

My mother isn't the only one who has this intuition. In fact, to many people it seems so obvious as to require no argument. And like many of these people, my mother would claim to be a philosophical materialist; that is, she doesn't believe in any nonphysical 'soul' or 'life force' that imbues living things with intelligence. It's just that she doesn't think machines could ever have the right stuff to 'actually think'.

In the academic realm, the most famous version of this argument was put forth by the philosopher John Searle. In 1980, Searle published an article called 'Minds, Brains, and Programs'[12] in which he vigorously argued against the possibility of machines *actually thinking*. In this widely read, controversial piece, Searle introduced the concepts of 'strong' and 'weak' AI in order to distinguish between two philosophical claims made about AI programs. While many people today use the phrase *strong AI* to mean 'AI that can perform most tasks as well as a human' and *weak AI* to mean the kind of narrow AI that currently exists, Searle meant something different by these terms. For Searle, the *strong AI* claim would be that 'the appropriately programmed digital computer does not just simulate having a mind; it literally has a mind.'[13] In contrast, in Searle's terminology, *weak AI* views computers as tools to simulate human intelligence and does not make any claims about them 'literally' having a mind.[14] We're back to the philosophical question I was discussing with my mother: Is there a difference between 'simulating a mind' and 'literally having a mind'? Like my mother, Searle believes there is a fundamental difference, and he argued that strong AI is impossible even in principle.[15]

The Turing Test

Searle's article was spurred in part by Alan Turing's 1950 paper, 'Computing Machinery and Intelligence', which had proposed a way to cut through the Gordian knot of 'simulated' versus 'actual' intelligence. Declaring that 'the original question "Can a machine think?" is too meaningless to deserve discussion,' Turing proposed an operational method to give it meaning. In his 'imitation game', now called the Turing test, there are two contestants: a computer and a human. Each is questioned separately

by a (human) judge who tries to determine which is which. The judge is physically separated from the two contestants so cannot rely on visual or auditory cues; only typed text is communicated.

Turing suggested the following: 'The question, "Can machines think?" should be replaced by "Are there imaginable digital computers which would do well in the imitation game?"' In other words, if a computer is sufficiently humanlike to be indistinguishable from humans, aside from its physical appearance or what it sounds like (or smells or feels like, for that matter), why shouldn't we consider it to *actually* think? Why should we require an entity to be created out of a particular kind of material (for example, biological cells) to grant it 'thinking' status? As the computer scientist Scott Aaronson put it bluntly, Turing's proposal is 'a plea against meat chauvinism'.[16]

The devil is always in the details, and the Turing test is no exception. Turing did not specify the criteria for selecting the human contestant and the judge, or stipulate how long the test should last, or what conversational topics should be allowed. However, he did make an oddly specific prediction: 'I believe that in about 50 years' time it will be possible to programme computers ... to make them play the imitation game so well that an average interrogator will not have more than 70 per cent chance of making the right identification after five minutes of questioning.' In other words, in a five-minute session, the average judge will be fooled 30 per cent of the time.

Turing's prediction has turned out to be pretty accurate. Several Turing tests have been staged over the years, in which the computer contestants are chatbots – programs specifically built to carry on conversations (they can't do anything else). In 2014, the Royal Society in London was host to a Turing test demonstration featuring five computer programs, thirty human contestants and thirty human judges of different ages and walks of life, including computer experts and non-experts, as well as native and non-native English speakers. Each judge conducted several rounds of five-minute conversations in which he or she conversed (by typing) in parallel with a pair of contestants – one human and one machine – after which the judge had to guess which was which. A chatbot named 'Eugene Goostman', created by a group of Russian and Ukrainian programmers, won the competition by fooling ten (or 33.3 per cent) of the judges. The competition organizers, following Turing's 'more than 30 per cent fooled in five minutes' criterion, quickly flooded the media with reports that the Turing test had finally been passed.

The Eugene Goostman chatbot works in much the same way as most other chatbots. It stores a large set of sentence templates that can be filled in based on a set of programmed rules that are applied to the input text it gets from its conversation partner. The chatbot's programmers have given it linguistic rules that allow it to pinpoint key information in its input and to store that information for later use. In addition, the chatbot stores a database of 'commonsense knowledge', encoded by human programmers, along with some logic rules; these would presumably allow it to answer questions like 'What colour is the sky over Vladivostok?' with the human-sounding 'Blue, I suppose, unless it's a cloudy day.' If none of the chatbot's rules apply to an input, it just changes the subject. The system's rules also encode its 'personality' – in this case, a thirteen-year-old Ukrainian boy whose English is good but (conveniently) not perfect. Here is a sample of Eugene Goostman's conversation with one of the judges who was fooled:

JUDGE: what is your gender
EUGENE: I'm a young boy, if you care to know. And not very ugly, by the way!
JUDGE: what are you aware of at this moment

sages usually do) won the game. The king asked the sage to name his reward. The sage, who enjoyed mathematics, said, 'All I ask for is that you take this chessboard, put two grains of rice on the first square, four grains on the second square, eight grains on the third, and so on, doubling the number of grains on each successive square. After you complete each row, package up the rice on that row and ship it to my village.' The mathematically naive king laughed. 'Is that all you want? I will have my men bring in some rice and fulfil your request post-haste.'

The king's men brought in a large bag of rice. After several minutes they had completed the first eight squares of the board with the requisite grains of rice: 2 on the first square, 4 on the second, 8 on the third, and so on, with 256 grains on the eighth square. They put the collection of grains (511, to be exact) in a tiny bag and sent it off by horseback to the sage's village. They then proceeded on to the second row, with 512 grains on the first square of that row, 1,024 grains on the next square, and 2,048 grains on the following. Each pile of rice no longer fit on a chessboard square, so it was counted into a large bowl instead. By the end of the second row, the counting of grains was taking far too much time, so the court mathematicians started estimating the amounts by weight. They calculated that for the sixteenth square, 65,536 grains – about a kilogram (just over two pounds) – were required. The bag of rice shipped off for the second row weighed about two kilograms.

The king's men started on the third row. The seventeenth square required 2 kilos, the eighteenth required 4, and so on; by the end of the third row (square 24), 512 kilos were needed. The king's subjects were conscripted to bring in additional giant bags of rice. The situation had become dire by the second square of the fourth row (square 26), when the mathematicians calculated that 2,048 kilos (over two tons) of rice were required. This would exhaust the entire rice harvest of the kingdom, even though the chessboard was not even half completed. The king, now realizing the trick that had been played on him, begged the sage to relent and save the kingdom from starvation. The sage, satisfied that the rice already received by his village would be enough, agreed.

Figure 5*a* plots the number of kilos of rice required on each chess square, up to the twenty-fourth square. The first square, with two rice grains, has a scant fraction of a kilo. Similarly, the squares up through 16 have less than 1 kilo. But after square 16, you can see the plot shoot up rapidly, due to the doubling effect. Figure 5*b* shows the values for the twenty-fourth through the sixty-fourth chess square, going from 512 kilos to more than 30 trillion kilos.

The mathematical function describing this graph is $y = 2^x$, where x is the chess square (numbered from 1 to 64) and y is the number of rice grains required on that square. This is called an exponential function, because x is the exponent of the number 2. No matter what scale is plotted, the function will have a characteristic point at which the curve seems to change from slow to explosively fast growth.

Exponential Progress in Computers

For Ray Kurzweil, the computer age has provided a real-world counterpart to the exponential fable. In 1965, Gordon Moore, co-founder of Intel Corporation, identified a trend that has come to be known as Moore's law: the number of components on a computer chip doubles approximately every one to two years. In other words, the components are getting exponentially smaller (and cheaper), and computer speed and memory are increasing at an exponential rate.

Kurzweil's books are full of graphs like the ones in figure 5, and extrapolations of these trends of exponential progress, along the lines of Moore's law, are at the heart of his forecasts for AI. Kurzweil points out that if the trends continue (as he believes they will), a $1,000 computer will 'achieve human brain capability (1016 calculations per second) ... around the year 2023'.[28] At that point, in Kurzweil's view, human-level AI will just be a matter of reverse engineering the brain.

Neural Engineering

Reverse engineering the brain means understanding enough about its workings in order to duplicate it, or at least to use the brain's underlying principles to replicate its intelligence in a computer. Kurzweil believes that such reverse engineering is a practical, near-term approach to creating human-level AI. Most neuroscientists would vehemently disagree, given how little is currently known about how the brain works. But Kurzweil's argument again rests on exponential trends – this time in advancements in neuroscience. In 2002 he wrote, 'A careful analysis of the requisite trends shows that we will understand the principles of operation of the human brain and be in a position to re-create its powers in synthetic substances well within thirty years.'[29]

Few if any neuroscientists agree on this optimistic prediction for their field. But even if a machine operating on the brain's principles can be created, how will it learn all the stuff it needs to know to be considered intelligent? After all, a newborn baby has a brain, but it doesn't yet have what we'd call human-level intelligence. Kurzweil agrees: 'Most of [the brain's] complexity comes from its own interaction with a complex world. Thus, it will be necessary to provide an artificial intelligence with an education just as we do with a natural intelligence.'[30]

Of course, providing an education can take many years. Kurzweil thinks that the process can be vastly speeded up. 'Contemporary electronics is already more than ten million times faster than the human nervous system's electrochemical information processing. Once an AI masters human basic language skills, it will be in a position to expand its language skills and general knowledge by rapidly reading all human literature and by absorbing the knowledge contained on millions of web sites.'[31]

Kurzweil is vague on how all this will happen but assures us that to achieve human-level AI, 'we will not program human intelligence link by link as in some massive expert system. Rather, we will set up an intricate hierarchy of self-organizing systems, based largely on the reverse engineering of the human brain, and then provide for its education ... hundreds if not thousands of times faster than the comparable process for humans.'[32]

Singularity Sceptics and Adherents

Responses to Kurzweil's books *The Age of Spiritual Machines* (1999) and *The Singularity is Near* (2005) are often one of two extremes: enthusiastic embrace or dismissive

PENGUIN BOOKS

UK | USA | Canada | Ireland | Australia
India | New Zealand | South Africa

Penguin Books is part of the Penguin Random House group of companies whose addresses can be found at global.penguinrandomhouse.com.

First published in the USA by Farrar, Straus and Giroux 2019
First Published in Great Britain by Pelican Books 2019

Cover by Matthew Young

Book design by Matthew Young

ISBN: 978-0-241-40484-3