1 An introduction to artificial intelligence

This chapter covers

- Gaining perspective about the history of artificial intelligence
- Understanding machine learning and its relationship to AI
- Exploring the drivers of the explosion in AI applications

Artificial intelligence (AI) is not a new technology. For decades, computer scientists have tried different approaches to reach the holy grail of computing: intelligent machines. While we are still far away from replicating the wonders of the human brain, AI applications have started to fill our daily lives and power our electronic devices, from smartphones to home alarm systems.

Why this seemingly sudden explosion? This chapter will answer this question by teaching you about modern AI—including the core principles behind it, and how and why we got to where we are now.

1.1 The path to modern AI

As humans, we've always tried to find ways to understand the world around us and bend nature to meet our goals. To do so, we have always relied on external tools that amplify our brain's capabilities.

The abacus was probably the first such tool, invented about 5,000 to 6,000 years ago to help people make calculations. Although it's still used in schools to help children visualize simple mathematical operations, it doesn't really save us from the labor of actually performing them. We had to wait until the 1960s for the first machines that could add and subtract numbers automatically. Computers have come a long way since then, but deep down their capability has still been pretty simple: executing calculations exactly as some (expert) human has instructed them to do. There's little "intelligence" in them.

The two words *artificial* and *intelligence* were first put together on August 31, 1955, when professor John McCarthy from Dartmouth College, together with M.L Minsky from Harvard University, N. Rochester from IBM, and C. E. Shannon from Bell Telephone Laboratories, asked the Rockefeller Foundation to fund a summer of research on artificial intelligence. Their proposal stated the following:

> *We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. . . . An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. 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.*

The researchers knew that tackling intelligence as a whole was too tough of a challenge, both because of technical limitations *and* the inherent complexity of the task. Instead of solving the broad concept of intelligence, they decided to focus on subproblems, like language. Later, these applications would be called *narrow AI*. An artificial intelligence capable of matching or surpassing human capabilities would instead be called *general AI*. In other words:

- *General AI* (or *strong AI*)—An artificial intelligence program capable of tackling every kind of task it's presented. This is similar to an extremely resourceful human, and you can think of it as the robot from *The Terminator* (or, hopefully, a more peaceful version of it).
- *Narrow AI*—An artificial intelligence program capable of solving a single, well-defined task. It can be broad (recognizing objects from pictures) or extremely specific (predicting which customers who bought product A are more likely to purchase product B as well). This means one task at a time, and not any other: an AI that recognizes cats in images can't translate English to Italian, and vice versa.

General AI is still far away: researchers still don't know when we'll finally get it. Some argue that we'll never get there. Even though general AI is still a distant, fuzzy dream, this is what many people have in mind when AI is mentioned in the news. If you were one of those people, and are now disappointed that general AI is not here yet, don't despair. Narrow AI applications are still capable of creating immense value. For example, AI that can detect lung cancer is a narrow application but nevertheless extremely useful.

The results of the Dartmouth research summer of 1956 were so interesting that they sparked a wave of excitement and hope among the participants. The enthusiasm of the scientists spread to the US government, which started heavily funding research on a specific application: English/Russian translation. Finding trustworthy Russian translators must not have been easy in the midst of the Cold War.

After the first few years of work, a government committee produced the infamous 1966 Automatic Language Processing Advisory Committee (ALPAC) report. The document featured the opinions of many researchers about the state of AI research. Most were not very positive:

> *Early machine translations of simple or selected text . . . were as deceptively encouraging as "machine translations" of general scientific text have been uniformly discouraging. . . . No one can guarantee, of course, that we will not suddenly or at least quickly attain machine translation, but we feel that this is very unlikely.*
>
> *. . . there is no immediate or predictable prospect of useful machine translation.*

The ALPAC report marks the beginning of a period called the *first AI winter*: public funding for AI research stopped, excitement cooled, and researchers focused their work on other fields.

Interest in AI faded until the 1980s, when private companies such as IBM and Xerox started investing in a new AI spring. New hopes were fueled by a technology called *expert systems*: computer programs that encode the knowledge of a human expert in a certain field in the form of precise, *if-then* rules. An example will help you understand how expert systems were designed to work.

Suppose you want to build an AI system that can stand in for a gastroenterologist. This is how you do it with an expert system: you ask a doctor to describe with extreme precision how they make decisions about patients. You then ask a programmer to painstakingly transform the doctor's knowledge and diagnosis flow to if-then rules that can be understood and executed by a computer. An extremely simplified version would look something like this:

> *If the patient has a stomachache and the body temperature is high, then the patient has the flu.*
>
> *If the patient has a stomachache and has eaten expired food, then the patient has food poisoning.*

And so on. Once the doctor's knowledge is encoded into the software and a patient comes in, the software follows the same decision path as the doctor and (hopefully) comes up with the same diagnosis. This approach has several problems:

- *Poor adaptability*—The only way for the software to improve is to go back to the drawing board with a computer scientist and the expert (in this case, the doctor).
- *Extreme brittleness*—The system will fail in situations that weren't part of the original design. What if a patient has a stomachache but normal body temperature, and hasn't eaten spoiled food?
- *Tough to maintain*—The complexity of such a system is huge. When thousands of rules are put together, improving it or changing it is incredibly complicated, slow, and expensive. Have you ever worked with a huge Microsoft Excel sheet and struggled to find the root cause of a mistake? Imagine an Excel sheet 100 times bigger.

Expert systems were a commercial failure. By the end of the 1980s, many of the companies that were developing them went out of business, marking the beginning of the *second AI winter*. It wasn't until the early 2000s that the next generation of AI successes came along, fueled by an old idea that became new again: machine learning.

1.2 The engine of the AI revolution: machine learning

The first definition of *machine learning* dates back to 1959, from American AI pioneer Arthur Samuel:

> *Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed.*

The key elements here are *learning* and *without being explicitly programmed*. Let's focus on the latter first. Explicitly programming a computer means defining the rules and instructions it must follow to perform a specific task. This is what software engineers do when they write software that handles your everyday tasks like doing taxes or filling out spreadsheets.

People without programming experience often feel like software engineers are powerful creatures who can bend machines to their will. Unfortunately, things are not always that easy. Try to think about the various decisions you make as you perform some trivial actions: can you explain the process you follow to recognize your friends when you see them? All the split-second decisions you make while driving? Can you list all the English grammar rules you apply as you talk? If you can't precisely explain how you do something, there's no chance that you can instruct a computer to do it.

Samuel proposed to replace "instructing computers" with "giving them the ability to learn." If you think about it, learning instead of following instructions is (coincidentally?) what human beings do all the time. Our mothers and fathers don't teach us their native tongue by giving us grammar books at the tender age of one. They just speak to us naturally, and we learn from their example, applying thousands of grammar rules without even knowing it. In fact, our brain is capable of automatically extracting rules way before it becomes capable of rationally understanding grammar at school! Even for us humans, it looks like learning rules from examples can be easier than being told about them.

In the same way we learn from experience, machine learning (ML) techniques allow computers to learn from data. Let's make it more concrete with a classic toy example: teaching a computer to tell dogs from cats in pictures. If you had to teach a kid to perform this task, you wouldn't pick up a veterinary book and start reading about the differences in ear shape or fur color. Instead, you'd probably just point them to a few pictures and let their brain do its magic.

An ML solution to the "dog or cat" problem is similar to our childhood learning experiences. We feed the computer thousands of images of cats and tell it "these are cats," and then thousands of images of dogs and tell it "these are dogs." Finally, we let it figure out the difference between the two pets automatically. We don't have to explain the key elements that distinguish dogs from cats. A good ML application learns to figure that out from the examples it receives. Figure 1.1 shows the difference between traditional programming and machine learning.

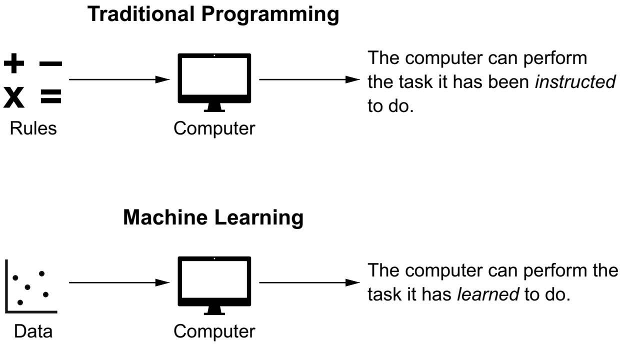
Traditional Programming
+ −
x =
Rules
Computer
The computer can perform the task it has been *instructed* to do.
Machine Learning
Data
Computer
The computer can perform the task it has *learned* to do.

Figure 1.1 The difference between the traditional programming approach and machine learning: the first relies on precise rules and instructions, the latter on data and learning.

You may start to sense why ML couldn't possibly have blossomed before the 2000s. The main ingredient of this set of techniques is *data*, and the internet has made collecting data much easier. The other crucial ingredient for ML is *computing power*: learning from data doesn't happen for free, and computers need fast processors to perform this task. Thanks to cloud computing and increases in processing power, access to powerful computers has never been so easy and cheap.

To give you a sense of how much things have changed in just a few years, we asked Alex Waibel, one of the pioneers of AI in speech recognition and among the first hires of Facebook's AI team, how different it was to work on ML 20 years ago. The most powerful computer he could use in the early 2000s was as big as an apartment, cost a few million, and he needed to rent it to train his models. Today, he has much more computing power sitting on his desk for a few thousand dollars. Your phone is probably more powerful than what top researchers had available just 20 years ago.

Availability of data and cheap computing power created the perfect environment for machine learning to bloom. Indeed, many (most) of the coolest consumer-facing applications of what we call AI today rely heavily on ML: the Siri voice virtual assistant, Google Translate, self-driving cars, and many more.

Going back to the history of AI, it seems that ML is the engine that powered today's AI explosion, finally bringing some hope after the last AI winter of the 1980s. In fact, the success of modern AI has been so dependent on ML techniques that people are often confused about the difference between the two. What is artificial intelligence, then? Let's find out.

1.3 What is artificial intelligence, after all?

In our experience as technologists, consultants, and public speakers, we are constantly meeting people with different opinions about the definitions of *AI*, *data science*, and *ML*. Although many are quite opinionated, few can defend their position. Indeed, finding a universal definition of AI is not as trivial as it might look.

Going by its name, we might try to define *artificial intelligence* by finding the human traits that we associate with intelligence. Once we agree on what makes humans intelligent, we can say that any computer that does the same thing is AI. It makes sense, right? Although this is a common approach, it falls apart even with simple scenarios. For instance, a human who can divide 13.856 by 13 down to the tenth decimal number in a split second would definitely be called intelligent, yet its artificial counterpart is a $2 pocket calculator that nobody would dare call AI. At the same time, we would never call someone intelligent just because they're able to drive in heavy traffic, yet a self-driving car is generally considered one of the toughest forms of AI the tech industry is working on today. We shouldn't be surprised by how hard defining *intelligence* is; after all, philosophers and scientists have been debating about it for centuries.

Not only do we have different weights to measure human and machine intelligence, but we also seem to be changing our mind pretty fast about what is AI and what isn't. Let's take an example from Paul Graham, founder of Y Combinator, the most successful Silicon Valley startup accelerator, and arguably one of the most forward-looking people in tech. In 2002, Graham wrote an essay proposing a new solution to detect spam emails. Back then, email was just getting off the ground, and spam (unwanted email) was one of the most serious threats to widespread use of the internet by nontechies. It seems hard to imagine now, but the best computer scientists were busy trying to write complex rules to let computers automatically sort through Viagra advertisements.

In his essay, Graham thought about a new ML-based approach that would learn to classify an email by processing thousands of "good" and spam emails. Paul's simple software learned to recognize spam better than the complex rules concocted by engineers. Fast-forward 20 years, and automatic spam detectors are such a boring technology that we would be laughed out of the room if we dared call it AI.

In fact, it seems like AI is about mastering tasks that our imagination suggests computers shouldn't be able to do. Once we get used to a technology in our daily life, we remove the AI badge of honor and start calling it just computer software. This is a well-studied phenomenon called the *AI effect*.

Because of the AI effect, the goalposts for what we call AI keep moving just as quickly as technology improves. The definition of AI we draw from these considerations is "a temporary label to a piece of software that does something cool and surprising, until we get used to it." We don't know about you, but that just doesn't feel like a satisfying definition.

We hope we have convinced you that it is extremely hard to find a definition that makes everyone happy and can be valid as technology evolves. With the AI effect in mind, we decided to avoid a narrow definition of AI that rewards "flashy" applications just to ditch them once the hype is gone. We embrace a broader definition that includes less flashy applications. This is our definition of AI:

> *Software that solves a problem without explicit human instruction.*

As you can see, our definition focuses on the outcome of the technology rather than the specific techniques used to build it. Some people will not agree with it, because it's almost equivalent to what we said about machine learning earlier in the chapter. The truth is, learning *is* an intelligent trait, and while ML is just a tool, it is *the* tool behind 99% of the successful applications we happen to call AI today. This may change in the future, but we don't see any new approaches on the horizon that hold the same promise as ML. This is why every AI application we'll cover in this book is based on ML: it's simply the most accurate picture of the AI landscape of today and the near future.

We now have a clear view of what ML is, a working definition of modern AI, and some perspective about how these terms evolved. We are just missing the third buzzword you've probably heard about: data science.

Data science (*DS*) is a broad, multidisciplinary field that uses scientific methods and processes to analyze data and extract insights. ML techniques are some of the tools in the DS toolbox. In practice, when people refer to a *data science project*, they often mean something *static*: extracting insights from data and presenting them as a presentation or report. On the other hand, AI is more commonly used in the context of live software.

For instance, analyzing traffic data to design a new urban plan for a city to minimize congestion likely falls into the realm of data science. However, if you use the same data to control traffic in real time and direct cars through less-congested routes, most people would say the project is about AI. In the first case, the output of your project is a report, and in the second, it's "live" software that runs 24/7. Keep in mind that this division is mostly conventional: there really are no hard-and-fast rules about what's AI and what's data science. Table 1.1 summarizes the differences as we see them.

Table 1.1 The main differences between AI and data science

Artificial intelligence	Data science
Automates tasks or predicts future events based on data.	Produces insights based on data.
Is commonly used "live": it continuously elaborates new data and produces answers.	Is commonly "one-off": it produces some insights that inform decisions.
It commonly has the form of software.	It commonly has the form of a presentation or report.

Hopefully, these sections helped demystify some commonly misunderstood terms and created context for these technologies. Now you can start learning the core principles of AI, what you can potentially do with it, and how to bring this transformative technology into your organization. In the next section, we'll explain the steps of this journey and how this book guides you through them.

1.4 Our teaching method

If you want to productively use AI in your work life, it's paramount that you understand its nuts and bolts first. We noticed that nontechnical people who approach AI without a solid understanding of its principles often end up dreaming about projects that are simply impossible to build, or miss low-hanging fruit that could be easily tackled. After the first part of the book, you'll know all the AI principles you need to avoid these dead ends and get the best out of the technology.

Even after just this first chapter, you already understand that virtually all modern AI applications rely on machine learning, and machine learning is all about learning from data. This is why we used data as your guide to understanding AI. Each chapter of the first part of the book focuses on one specific kind of data, showing you how to spot it in your organization, what you can do with it, and how it fits into the world of AI.

Each chapter in part 1 uses a toy example to introduce the ML concepts you need. We found this to be the most efficient way to teach ML concepts that would otherwise be too dry and abstract. We didn't dig deep into technological aspects for two simple reasons:

- Technology changes so rapidly that implementation details would soon become obsolete.
- Simply put, you don't need it. Unless you want to pivot your career to writing code, we believe there's more value in adding AI to your wealth of knowledge and letting someone else practically implement your vision in computer terms.

This doesn't mean that we'll completely spare you from technicalities. From our experience as engineers, we know that it can be difficult for your technical team to communicate with people without the smallest bit of technical understanding. We don't want them to have trouble talking to *you*, so we made sure that you'll be learning the most important technical aspects of AI tools. Even as you leave them in the hands of your team, knowing about them will help you plan and manage the efforts.

Each chapter includes one or more real-world business cases about companies that achieved extraordinary results. To the extent that we mention specific companies, products, or services, keep in mind that we do so because we want you to develop awareness, but you shouldn't feel limited to them in any way. We have no affiliation or stake in any of the companies in the case studies; it just so happens that they're building great products we can all learn from.

When presenting cases, we followed a methodology inspired by the Harvard Business School case method: we'll first present the case in the most neutral way possible, and ask you open-ended questions at the end. Right after that, we include our thoughts about these questions and prompts for further discussion. We recommend you don't read these answers right away, but rather try thinking about how *you* would answer based on your knowledge and what you've read in the case, and only then read our take. Be aware that there's no unique solution to the questions we asked: if you found an interesting take on the cases that we didn't include in the answers, good job! This means you've learned what you needed and are able to extract insights on your own (so if that happens, we reached our goal with this book as well).

Summary

- AI has had a long history of successes and failures that dates back to the 1950s.
- General AI is the pipe dream of having an all-knowing machine. All AI applications we have today are instead narrow; they focus on specific tasks.
- Machine learning is the prevalent way to implement AI today, and is based on letting machines learn autonomously from data.
- Data science is related to AI and ML, but focuses more on extracting insights than persistent intelligence.