1.4 Roots

We have just reviewed a small subset of problems that machine learning can address. For a diverse set of machine learning problems, deep learning provides powerful tools for their solution. Although many deep learning methods are recent inventions, the core ideas behind learning from data have been studied for centuries. In fact, humans have held the desire to analyze data and to predict future outcomes for ages, and it is this desire that is at the root of much of natural science and mathematics. Two examples are the Bernoulli distribution, named after Jacob Bernoulli (1655–1705)[23], and the Gaussian distribution discovered by Carl Friedrich Gauss (1777–1855)[24]. Gauss invented, for instance, the least mean squares algorithm, which is still used today for a multitude of problems from insurance calculations to medical diagnostics. Such tools enhanced the experimental approach in the natural sciences—for instance, Ohm's law relating current and voltage in a resistor is perfectly described by a linear model.

Even in the middle ages, mathematicians had a keen intuition of estimates. For instance, the geometry book of Jacob Köbel (1460–1533)[25] illustrates averaging the length of 16 adult men's feet to estimate the typical foot length in the population (Fig. 1.4.1).

As a group of individuals exited a church, 16 adult men were asked to line up in a row and have their feet measured. The sum of these measurements was then divided by 16 to obtain an estimate for what now is called one foot. This "algorithm" was later improved to deal with misshapen feet; The two men with the shortest and longest feet were sent away, averaging only over the remainder. This is among the earliest examples of a trimmed mean estimate.

Statistics really took off with the availability and collection of data. One of its pioneers, Ronald Fisher (1890–1962)[26], contributed significantly to its theory and also its applications in genetics. Many of his algorithms (such as linear discriminant analysis) and concepts (such as the Fisher information matrix) still hold a prominent place in the foundations of modern statistics. Even his data resources had a lasting impact. The Iris dataset that Fisher released in 1936 is still sometimes used to demonstrate machine learning algorithms. Fisher was also a proponent of eugenics, which should remind us that the morally

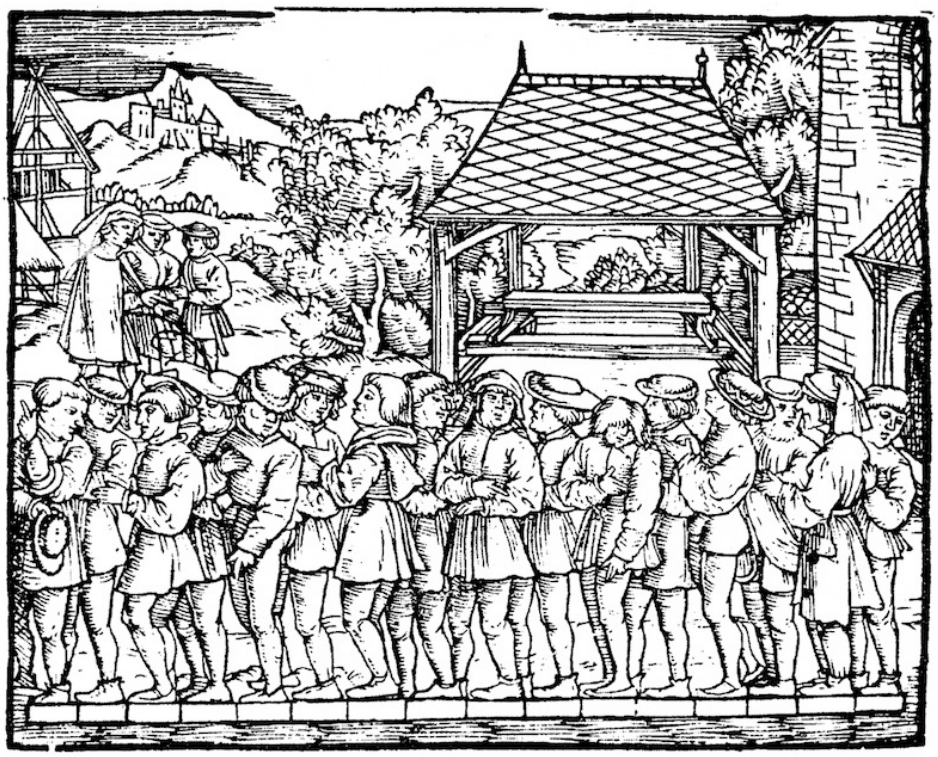

Fig. 1.4.1 Estimating the length of a foot.

dubious use of data science has as long and enduring a history as its productive use in industry and the natural sciences.

Other influences for machine learning came from the information theory of Claude Shannon (1916–2001)[27] and the theory of computation proposed by Alan Turing (1912–1954)[28]. Turing posed the question "can machines think?" in his famous paper *Computing Machinery and Intelligence* (Turing, 1950). Describing what is now known as the Turing test, he proposed that a machine can be considered *intelligent* if it is difficult for a human evaluator to distinguish between the replies from a machine and those of a human, based purely on textual interactions.

27

28

Further influences came from neuroscience and psychology. After all, humans clearly exhibit intelligent behavior. Many scholars have asked whether one could explain and possibly reverse engineer this capacity. One of the first biologically inspired algorithms was formulated by Donald Hebb (1904–1985)[29]. In his groundbreaking book *The Organization of Behavior* (Hebb, 1949), he posited that neurons learn by positive reinforcement. This became known as the Hebbian learning rule. These ideas inspired later work, such as Rosenblatt's perceptron learning algorithm, and laid the foundations of many stochastic gradient descent algorithms that underpin deep learning today: reinforce desirable behavior and diminish undesirable behavior to obtain good settings of the parameters in a neural network.

29

Biological inspiration is what gave *neural networks* their name. For over a century (dating back to the models of Alexander Bain, 1873, and James Sherrington, 1890), researchers have tried to assemble computational circuits that resemble networks of interacting neurons. Over time, the interpretation of biology has become less literal, but the name stuck. At its heart lie a few key principles that can be found in most networks today:

- The alternation of linear and nonlinear processing units, often referred to as *layers*.

- The use of the chain rule (also known as *backpropagation*) for adjusting parameters in the entire network at once.

After initial rapid progress, research in neural networks languished from around 1995 until 2005. This was mainly due to two reasons. First, training a network is computationally very expensive. While random-access memory was plentiful at the end of the past century, computational power was scarce. Second, datasets were relatively small. In fact, Fisher's Iris dataset from 1936 was still a popular tool for testing the efficacy of algorithms. The MNIST dataset with its 60,000 handwritten digits was considered huge.

Given the scarcity of data and computation, strong statistical tools such as kernel methods, decision trees, and graphical models proved empirically superior in many applications. Moreover, unlike neural networks, they did not require weeks to train and provided predictable results with strong theoretical guarantees.

1.5 The Road to Deep Learning

Much of this changed with the availability of massive amounts of data, thanks to the World Wide Web, the advent of companies serving hundreds of millions of users online, a dissemination of low-cost, high-quality sensors, inexpensive data storage (Kryder's law), and cheap computation (Moore's law). In particular, the landscape of computation in deep learning was revolutionized by advances in GPUs that were originally engineered for computer gaming. Suddenly algorithms and models that seemed computationally infeasible were within reach. This is best illustrated in `tab_intro_decade`.

:Dataset vs. computer memory and computational power

Table 1.5.1: label:`tab_intro_decade`

Decade	Dataset	Memory	Floating point calculations per second
1970	100 (Iris)	1 KB	100 KF (Intel 8080)
1980	1 K (house prices in Boston)	100 KB	1 MF (Intel 80186)
1990	10 K (optical character recognition)	10 MB	10 MF (Intel 80486)
2000	10 M (web pages)	100 MB	1 GF (Intel Core)
2010	10 G (advertising)	1 GB	1 TF (NVIDIA C2050)
2020	1 T (social network)	100 GB	1 PF (NVIDIA DGX-2)

Note that random-access memory has not kept pace with the growth in data. At the same time, increases in computational power have outpaced the growth in datasets. This means that statistical models need to become more memory efficient, and so they are free to spend more computer cycles optimizing parameters, thanks to the increased compute budget. Consequently, the sweet spot in machine learning and statistics moved from (generalized) linear models and kernel methods to deep neural networks. This is also one of the reasons why many of the mainstays of deep learning, such as multilayer perceptrons (McCulloch and Pitts, 1943), convolutional neural networks (LeCun *et al.*, 1998), long short-term memory (Hochreiter and Schmidhuber, 1997), and Q-Learning (Watkins and Dayan, 1992), were essentially "rediscovered" in the past decade, after lying comparatively dormant for considerable time.

The recent progress in statistical models, applications, and algorithms has sometimes been likened to the Cambrian explosion: a moment of rapid progress in the evolution of species. Indeed, the state of the art is not just a mere consequence of available resources applied to decades-old algorithms. Note that the list of ideas below barely scratches the surface of what has helped researchers achieve tremendous progress over the past decade.

- Novel methods for capacity control, such as *dropout* (Srivastava *et al.*, 2014), have helped to mitigate overfitting. Here, noise is injected (Bishop, 1995) throughout the neural network during training.

- *Attention mechanisms* solved a second problem that had plagued statistics for over a century: how to increase the memory and complexity of a system without increasing the number of learnable parameters. Researchers found an elegant solution by using what can only be viewed as a *learnable pointer structure* (Bahdanau *et al.*, 2014). Rather than having to remember an entire text sequence, e.g., for machine translation in a fixed-dimensional representation, all that needed to be stored was a pointer to the intermediate state of the translation process. This allowed for significantly increased accuracy for long sequences, since the model no longer needed to remember the entire sequence before commencing the generation of a new one.

- Built solely on attention mechanisms, the *Transformer* architecture (Vaswani *et al.*, 2017)

has demonstrated superior *scaling* behavior: it performs better with an increase in dataset size, model size, and amount of training compute (Kaplan *et al.*, 2020). This architecture has demonstrated compelling success in a wide range of areas, such as natural language processing (Brown *et al.*, 2020, Devlin *et al.*, 2018), computer vision (Dosovitskiy *et al.*, 2021, Liu *et al.*, 2021), speech recognition (Gulati *et al.*, 2020), reinforcement learning (Chen *et al.*, 2021), and graph neural networks (Dwivedi and Bresson, 2020). For example, a single Transformer pretrained on modalities as diverse as text, images, joint torques, and button presses can play Atari, caption images, chat, and control a robot (Reed *et al.*, 2022).

- Modeling probabilities of text sequences, *language models* can predict text given other text. Scaling up the data, model, and compute has unlocked a growing number of capabilities of language models to perform desired tasks via human-like text generation based on input text (Anil *et al.*, 2023, Brown *et al.*, 2020, Chowdhery *et al.*, 2022, Hoffmann *et al.*, 2022, OpenAI, 2023, Rae *et al.*, 2021, Touvron *et al.*, 2023a, Touvron *et al.*, 2023b). For instance, aligning language models with human intent (Ouyang *et al.*, 2022), OpenAI's ChatGPT[30] allows users to interact with it in a conversational way to solve problems, such as code debugging and creative writing.

[30] 

- Multi-stage designs, e.g., via the memory networks (Sukhbaatar *et al.*, 2015) and the neural programmer-interpreter (Reed and De Freitas, 2015) permitted statistical modelers to describe iterative approaches to reasoning. These tools allow for an internal state of the deep neural network to be modified repeatedly, thus carrying out subsequent steps in a chain of reasoning, just as a processor can modify memory for a computation.

- A key development in *deep generative modeling* was the invention of *generative adversarial networks* (Goodfellow *et al.*, 2014). Traditionally, statistical methods for density estimation and generative models focused on finding proper probability distributions and (often approximate) algorithms for sampling from them. As a result, these algorithms were largely limited by the lack of flexibility inherent in the statistical models. The crucial innovation in generative adversarial networks was to replace the sampler by an arbitrary algorithm with differentiable parameters. These are then adjusted in such a way that the discriminator (effectively a two-sample test) cannot distinguish fake from real data. Through the ability to use arbitrary algorithms to generate data, density estimation was opened up to a wide variety of techniques. Examples of galloping zebras (Zhu *et al.*, 2017) and of fake celebrity faces (Karras *et al.*, 2017) are each testimony to this progress. Even amateur doodlers can produce photorealistic images just based on sketches describing the layout of a scene (Park *et al.*, 2019).

- Furthermore, while the diffusion process gradually adds random noise to data samples, *diffusion models* (Ho *et al.*, 2020, Sohl-Dickstein *et al.*, 2015) learn the denoising process to gradually construct data samples from random noise, reversing the diffusion process. They have started to replace generative adversarial networks in more recent deep generative models, such as in DALL-E 2 (Ramesh *et al.*, 2022) and Imagen (Saharia *et al.*, 2022) for creative art and image generation based on text descriptions.

- In many cases, a single GPU is insufficient for processing the large amounts of data

available for training. Over the past decade the ability to build parallel and distributed training algorithms has improved significantly. One of the key challenges in designing scalable algorithms is that the workhorse of deep learning optimization, stochastic gradient descent, relies on relatively small minibatches of data to be processed. At the same time, small batches limit the efficiency of GPUs. Hence, training on 1,024 GPUs with a minibatch size of, say, 32 images per batch amounts to an aggregate minibatch of about 32,000 images. Work, first by Li (2017) and subsequently by You *et al.* (2017) and Jia *et al.* (2018) pushed the size up to 64,000 observations, reducing training time for the ResNet-50 model on the ImageNet dataset to less than 7 minutes. By comparison, training times were initially of the order of days.

- The ability to parallelize computation has also contributed to progress in *reinforcement learning*. This has led to significant progress in computers achieving superhuman performance on tasks like Go, Atari games, Starcraft, and in physics simulations (e.g., using MuJoCo) where environment simulators are available. See, e.g., Silver *et al.* (2016) for a description of such achievements in AlphaGo. In a nutshell, reinforcement learning works best if plenty of (state, action, reward) tuples are available. Simulation provides such an avenue.

- Deep learning frameworks have played a crucial role in disseminating ideas. The first generation of open-source frameworks for neural network modeling consisted of Caffe [31], Torch [32], and Theano [33]. Many seminal papers were written using these tools. These have now been superseded by TensorFlow [34] (often used via its high-level API Keras [35]), CNTK [36], Caffe 2 [37], and Apache MXNet [38]. The third generation of frameworks consists of so-called *imperative* tools for deep learning, a trend that was arguably ignited by Chainer [39], which used a syntax similar to Python NumPy to describe models. This idea was adopted by both PyTorch [40], the Gluon API [41] of MXNet, and JAX [42].

The division of labor between system researchers building better tools and statistical modelers building better neural networks has greatly simplified things. For instance, training a linear logistic regression model used to be a nontrivial homework problem, worthy to give to new machine learning Ph.D. students at Carnegie Mellon University in 2014. By now, this task can be accomplished with under 10 lines of code, putting it firmly within the reach of any programmer.

1.6 Success Stories

Artificial intelligence has a long history of delivering results that would be difficult to accomplish otherwise. For instance, mail sorting systems using optical character recognition have been deployed since the 1990s. This is, after all, the source of the famous MNIST dataset of handwritten digits. The same applies to reading checks for bank deposits and scoring creditworthiness of applicants. Financial transactions are checked for fraud auto-

matically. This forms the backbone of many e-commerce payment systems, such as PayPal, Stripe, AliPay, WeChat, Apple, Visa, and MasterCard. Computer programs for chess have been competitive for decades. Machine learning feeds search, recommendation, personalization, and ranking on the Internet. In other words, machine learning is pervasive, albeit often hidden from sight.

It is only recently that AI has been in the limelight, mostly due to solutions to problems that were considered intractable previously and that are directly related to consumers. Many of such advances are attributed to deep learning.

- Intelligent assistants, such as Apple's Siri, Amazon's Alexa, and Google's assistant, are able to respond to spoken requests with a reasonable degree of accuracy. This includes menial jobs, like turning on light switches, and more complex tasks, such as arranging barber's appointments and offering phone support dialog. This is likely the most noticeable sign that AI is affecting our lives.

- A key ingredient in digital assistants is their ability to recognize speech accurately. The accuracy of such systems has gradually increased to the point of achieving parity with humans for certain applications (Xiong *et al.*, 2018).

- Object recognition has likewise come a long way. Identifying the object in a picture was a fairly challenging task in 2010. On the ImageNet benchmark researchers from NEC Labs and University of Illinois at Urbana-Champaign achieved a top-five error rate of 28% (Lin *et al.*, 2010). By 2017, this error rate was reduced to 2.25% (Hu *et al.*, 2018). Similarly, stunning results have been achieved for identifying birdsong and for diagnosing skin cancer.

- Prowess in games used to provide a measuring stick for human ability. Starting from TD-Gammon, a program for playing backgammon using temporal difference reinforcement learning, algorithmic and computational progress has led to algorithms for a wide range of applications. Compared with backgammon, chess has a much more complex state space and set of actions. DeepBlue beat Garry Kasparov using massive parallelism, special-purpose hardware and efficient search through the game tree (Campbell *et al.*, 2002). Go is more difficult still, due to its huge state space. AlphaGo reached human parity in 2015, using deep learning combined with Monte Carlo tree sampling (Silver *et al.*, 2016). The challenge in Poker was that the state space is large and only partially observed (we do not know the opponents' cards). Libratus exceeded human performance in Poker using efficiently structured strategies (Brown and Sandholm, 2017).

- Another indication of progress in AI is the advent of self-driving vehicles. While full autonomy is not yet within reach, excellent progress has been made in this direction, with companies such as Tesla, NVIDIA, and Waymo shipping products that enable partial autonomy. What makes full autonomy so challenging is that proper driving requires the ability to perceive, to reason and to incorporate rules into a system. At present, deep learning is used primarily in the visual aspect of these problems. The rest is heavily tuned by engineers.

This barely scratches the surface of significant applications of machine learning. For instance, robotics, logistics, computational biology, particle physics, and astronomy owe some of their most impressive recent advances at least in parts to machine learning, which is thus becoming a ubiquitous tool for engineers and scientists.

Frequently, questions about a coming AI apocalypse and the plausibility of a *singularity* have been raised in non-technical articles. The fear is that somehow machine learning systems will become sentient and make decisions, independently of their programmers, that directly impact the lives of humans. To some extent, AI already affects the livelihood of humans in direct ways: creditworthiness is assessed automatically, autopilots mostly navigate vehicles, decisions about whether to grant bail use statistical data as input. More frivolously, we can ask Alexa to switch on the coffee machine.

Fortunately, we are far from a sentient AI system that could deliberately manipulate its human creators. First, AI systems are engineered, trained, and deployed in a specific, goal-oriented manner. While their behavior might give the illusion of general intelligence, it is a combination of rules, heuristics and statistical models that underlie the design. Second, at present, there are simply no tools for *artificial general intelligence* that are able to improve themselves, reason about themselves, and that are able to modify, extend, and improve their own architecture while trying to solve general tasks.

A much more pressing concern is how AI is being used in our daily lives. It is likely that many routine tasks, currently fulfilled by humans, can and will be automated. Farm robots will likely reduce the costs for organic farmers but they will also automate harvesting operations. This phase of the industrial revolution may have profound consequences for large swaths of society, since menial jobs provide much employment in many countries. Furthermore, statistical models, when applied without care, can lead to racial, gender, or age bias and raise reasonable concerns about procedural fairness if automated to drive consequential decisions. It is important to ensure that these algorithms are used with care. With what we know today, this strikes us as a much more pressing concern than the potential of malevolent superintelligence for destroying humanity.

1.7 The Essence of Deep Learning

Thus far, we have talked in broad terms about machine learning. Deep learning is the subset of machine learning concerned with models based on many-layered neural networks. It is *deep* in precisely the sense that its models learn many *layers* of transformations. While this might sound narrow, deep learning has given rise to a dizzying array of models, techniques, problem formulations, and applications. Many intuitions have been developed to explain the benefits of depth. Arguably, all machine learning has many layers of computation, the first consisting of feature processing steps. What differentiates deep learning is that the operations learned at each of the many layers of representations are learned jointly from data.

The problems that we have discussed so far, such as learning from the raw audio signal, the raw pixel values of images, or mapping between sentences of arbitrary lengths and their counterparts in foreign languages, are those where deep learning excels and traditional methods falter. It turns out that these many-layered models are capable of addressing low-level perceptual data in a way that previous tools could not. Arguably the most significant commonality in deep learning methods is *end-to-end training*. That is, rather than assembling a system based on components that are individually tuned, one builds the system and then tunes their performance jointly. For instance, in computer vision scientists used to separate the process of *feature engineering* from the process of building machine learning models. The Canny edge detector (Canny, 1987) and Lowe's SIFT feature extractor (Lowe, 2004) reigned supreme for over a decade as algorithms for mapping images into feature vectors. In bygone days, the crucial part of applying machine learning to these problems consisted of coming up with manually-engineered ways of transforming the data into some form amenable to shallow models. Unfortunately, there is only so much that humans can accomplish by ingenuity in comparison with a consistent evaluation over millions of choices carried out automatically by an algorithm. When deep learning took over, these feature extractors were replaced by automatically tuned filters that yielded superior accuracy.

Thus, one key advantage of deep learning is that it replaces not only the shallow models at the end of traditional learning pipelines, but also the labor-intensive process of feature engineering. Moreover, by replacing much of the domain-specific preprocessing, deep learning has eliminated many of the boundaries that previously separated computer vision, speech recognition, natural language processing, medical informatics, and other application areas, thereby offering a unified set of tools for tackling diverse problems.

Beyond end-to-end training, we are experiencing a transition from parametric statistical descriptions to fully nonparametric models. When data is scarce, one needs to rely on simplifying assumptions about reality in order to obtain useful models. When data is abundant, these can be replaced by nonparametric models that better fit the data. To some extent, this mirrors the progress that physics experienced in the middle of the previous century with the availability of computers. Rather than solving by hand parametric approximations of how electrons behave, one can now resort to numerical simulations of the associated partial differential equations. This has led to much more accurate models, albeit often at the expense of interpretation.

Another difference from previous work is the acceptance of suboptimal solutions, dealing with nonconvex nonlinear optimization problems, and the willingness to try things before proving them. This new-found empiricism in dealing with statistical problems, combined with a rapid influx of talent has led to rapid progress in the development of practical algorithms, albeit in many cases at the expense of modifying and re-inventing tools that existed for decades.

In the end, the deep learning community prides itself on sharing tools across academic and corporate boundaries, releasing many excellent libraries, statistical models, and trained networks as open source. It is in this spirit that the notebooks forming this book are freely available for distribution and use. We have worked hard to lower the barriers of access for

anyone wishing to learn about deep learning and we hope that our readers will benefit from this.

1.8 Summary

Machine learning studies how computer systems can leverage experience (often data) to improve performance at specific tasks. It combines ideas from statistics, data mining, and optimization. Often, it is used as a means of implementing AI solutions. As a class of machine learning, representational learning focuses on how to automatically find the appropriate way to represent data. Considered as multi-level representation learning through learning many layers of transformations, deep learning replaces not only the shallow models at the end of traditional machine learning pipelines, but also the labor-intensive process of feature engineering. Much of the recent progress in deep learning has been triggered by an abundance of data arising from cheap sensors and Internet-scale applications, and by significant progress in computation, mostly through GPUs. Furthermore, the availability of efficient deep learning frameworks has made design and implementation of whole system optimization significantly easier, and this is a key component in obtaining high performance.

1.9 Exercises

1. Which parts of code that you are currently writing could be "learned", i.e., improved by learning and automatically determining design choices that are made in your code? Does your code include heuristic design choices? What data might you need to learn the desired behavior?
2. Which problems that you encounter have many examples for their solution, yet no specific way for automating them? These may be prime candidates for using deep learning.
3. Describe the relationships between algorithms, data, and computation. How do characteristics of the data and the current available computational resources influence the appropriateness of various algorithms?
4. Name some settings where end-to-end training is not currently the default approach but where it might be useful.

Discussions[43].

43