

# Chapter 1

## Introduction

Inventors have long dreamed of creating machines that think. Ancient Greek myths tell of intelligent objects, such as animated statues of human beings and tables that arrive full of food and drink when called.

When programmable computers were first conceived, people wondered whether they might become intelligent, over a hundred years before one was built (Lovelace, 1842). Today, *artificial intelligence (AI)* is a thriving field with many practical applications and active research topics. We look to intelligent software to automate routine labor, understand speech or images, make diagnoses in medicine and to support basic scientific research.

In the early days of artificial intelligence, the field rapidly tackled and solved problems that are intellectually difficult for human beings but relatively straightforward for computers—problems that can be described by a list of formal, mathematical rules. The true challenge to artificial intelligence proved to be solving the tasks that are easy for people to perform but hard for people to describe formally—problems that we solve intuitively, that feel automatic, like recognizing spoken words or faces in images.

This book is about a solution to these more intuitive problems. This solution is to allow computers to learn from experience and understand the world in terms of a hierarchy of concepts, with each concept defined in terms of its relation to simpler concepts. By gathering knowledge from experience, this approach avoids the need for human operators to formally specify all of the knowledge that the computer needs. The hierarchy of concepts allows the computer to learn complicated concepts by building them out of simpler ones. If we draw a graph showing how these concepts are built on top of each other, the graph is deep, with many layers. For this reason, we call this approach to AI *deep learning*.

Many of the early successes of AI took place in relatively sterile and formal environments and did not require computers to have much knowledge about

the world. For example, IBM's Deep Blue chess-playing system defeated world champion Garry Kasparov in 1997 (Hsu, 2002). Chess is of course a very simple world, containing only sixty-four locations and thirty-two pieces that can move in only rigidly circumscribed ways. Devising a successful chess strategy is a tremendous accomplishment, but the challenge is not due to the difficulty of describing the relevant concepts to the computer. Chess can be completely described by a very brief list of completely formal rules, easily provided ahead of time by the programmer.

Ironically, abstract and formal tasks that are among the most difficult mental undertakings for a human being are among the easiest for a computer. Computers have long been able to defeat even the best human chess player, but are only recently matching some of the abilities of average human beings to recognize objects or speech. A person's everyday life requires an immense amount of knowledge about the world. Much of this knowledge is subjective and intuitive, and therefore difficult to articulate in a formal way. Computers need to capture this same knowledge in order to behave in an intelligent way. One of the key challenges in artificial intelligence is how to get this informal knowledge into a computer.

Several artificial intelligence projects have sought to hard-code knowledge about the world in formal languages. A computer can reason about statements in these formal languages automatically using logical inference rules. This is known as the *knowledge base* approach to artificial intelligence. None of these projects has led to a major success. One of the most famous such projects is Cyc (Lenat and Guha, 1989). Cyc is an inference engine and a database of statements in a language called CycL. These statements are entered by a staff of human supervisors. It is an unwieldy process. People struggle to devise formal rules with enough complexity to accurately describe the world. For example, Cyc failed to understand a story about a person named Fred shaving in the morning (Linde, 1992). Its inference engine detected an inconsistency in the story: it knew that people do not have electrical parts, but because Fred was holding an electric razor, it believed the entity "FredWhileShaving" contained electrical parts. It therefore asked whether Fred was still a person while he was shaving.

The difficulties faced by systems relying on hard-coded knowledge suggest that AI systems need the ability to acquire their own knowledge, by extracting patterns from raw data. This capability is known as *machine learning*. The introduction of machine learning allowed computers to tackle problems involving knowledge of the real world and make decisions that appear subjective. A simple machine learning algorithm called *logistic regression* can determine whether to recommend cesarean delivery (Mor-Yosef *et al.*, 1990). A simple machine learning algorithm called *naive Bayes* can separate legitimate e-mail from spam e-mail.

The performance of these simple machine learning algorithms depends heavily

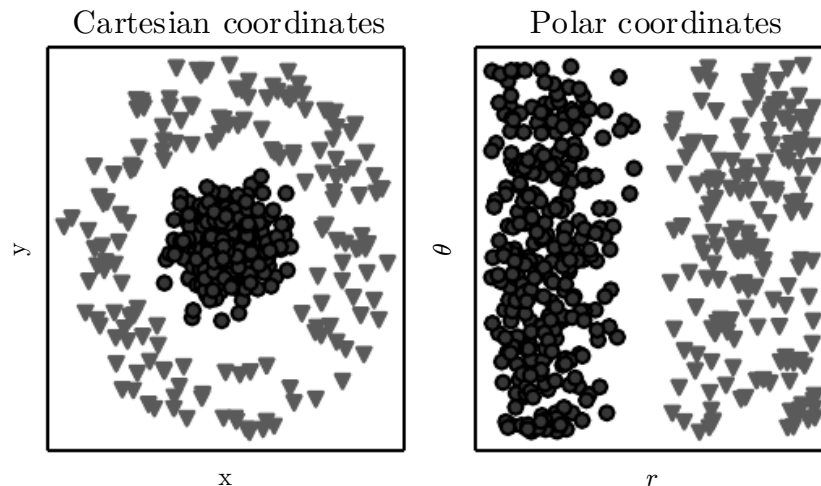


Figure 1.1: Example of different representations: suppose we want to separate two categories of data by drawing a line between them in a scatterplot. In the plot on the left, we represent some data using Cartesian coordinates, and the task is impossible. In the plot on the right, we represent the data with polar coordinates and the task becomes simple to solve with a vertical line. (Figure produced in collaboration with David Warde-Farley)

on the *representation* of the data they are given. For example, when logistic regression is used to recommend cesarean delivery, the AI system does not examine the patient directly. Instead, the doctor tells the system several pieces of relevant information, such as the presence or absence of a uterine scar. Each piece of information included in the representation of the patient is known as a *feature*. Logistic regression learns how each of these features of the patient correlates with various outcomes. However, it cannot influence the way that the features are defined in any way. If logistic regression was given a 3-D MRI image of the patient, rather than the doctor’s formalized report, it would not be able to make useful predictions. Individual voxels<sup>1</sup> in an MRI scan have negligible correlation with any complications that might occur during delivery.

This dependence on representations is a general phenomenon that appears throughout computer science and even daily life. In computer science, operations such as searching a collection of data can proceed exponentially faster if the collection is structured and indexed intelligently. People can easily perform arithmetic on Arabic numerals, but find arithmetic on Roman numerals much more time consuming. It is not surprising that the choice of representation has an enormous effect on the performance of machine learning algorithms. For a simple visual example, see Fig. 1.1.

Many artificial intelligence tasks can be solved by designing the right set of

---

<sup>1</sup>A voxel is the value at a single point in a 3-D scan, much as a pixel is the value at a single point in an image.

features to extract for that task, then providing these features to a simple machine learning algorithm. For example, a useful feature for speaker identification from sound is the pitch. The pitch can be formally specified—it is the lowest frequency major peak of the spectrogram. It is useful for speaker identification because it is determined by the size of the vocal tract, and therefore gives a strong clue as to whether the speaker is a man, woman, or child.

However, for many tasks, it is difficult to know what features should be extracted. For example, suppose that we would like to write a program to detect cars in photographs. We know that cars have wheels, so we might like to use the presence of a wheel as a feature. Unfortunately, it is difficult to describe exactly what a wheel looks like in terms of pixel values. A wheel has a simple geometric shape but its image may be complicated by shadows falling on the wheel, the sun glaring off the metal parts of the wheel, the fender of the car or an object in the foreground obscuring part of the wheel, and so on.

One solution to this problem is to use machine learning to discover not only the mapping from representation to output but also the representation itself. This approach is known as *representation learning*. Learned representations often result in much better performance than can be obtained with hand-designed representations. They also allow AI systems to rapidly adapt to new tasks, with minimal human intervention. A representation learning algorithm can discover a good set of features for a simple task in minutes, or a complex task in hours to months. Manually designing features for a complex task requires a great deal of human time and effort; it can take decades for an entire community of researchers.

The quintessential example of a representation learning algorithm is the *autoencoder*. An autoencoder is the combination of an *encoder* function that converts the input data into a different representation, and a *decoder* function that converts the new representation back into the original format. Autoencoders are trained to preserve as much information as possible when an input is run through the encoder and then the decoder, but are also trained to make the new representation have various nice properties. Different kinds of autoencoders aim to achieve different kinds of properties.

When designing features or algorithms for learning features, our goal is usually to separate the *factors of variation* that explain the observed data. In this context, we use the word “factors” simply to refer to separate sources of influence; the factors are usually not combined by multiplication. Such factors are often not quantities that are directly observed but they may exist either as unobserved objects or forces in the physical world that affect observable quantities, or they are constructs in the human mind that provide useful simplifying explanations or inferred causes of the observed data. They can be thought of as concepts or abstractions that help us make sense of the rich variability in the data. When analyzing a speech

recording, the factors of variation include the speaker’s age, their sex, their accent and the words that they are speaking. When analyzing an image of a car, the factors of variation include the position of the car, its color, and the angle and brightness of the sun.

A major source of difficulty in many real-world artificial intelligence applications is that many of the factors of variation influence every single piece of data we are able to observe. The individual pixels in an image of a red car might be very close to black at night. The shape of the car’s silhouette depends on the viewing angle. Most applications require us to *disentangle* the factors of variation and discard the ones that we do not care about.

Of course, it can be very difficult to extract such high-level, abstract features from raw data. Many of these factors of variation, such as a speaker’s accent, can only be identified using sophisticated, nearly human-level understanding of the data. When it is nearly as difficult to obtain a representation as to solve the original problem, representation learning does not, at first glance, seem to help us.

*Deep learning* solves this central problem in representation learning by introducing representations that are expressed in terms of other, simpler representations. Deep learning allows the computer to build complex concepts out of simpler concepts. Fig. 1.2 shows how a deep learning system can represent the concept of an image of a person by combining simpler concepts, such as corners and contours, which are in turn defined in terms of edges.

The quintessential example of a deep learning model is the feedforward deep network or *multilayer perceptron* (MLP). A multilayer perceptron is just a mathematical function mapping some set of input values to output values. The function is formed by composing many simpler functions. We can think of each application of a different mathematical function as providing a new representation of the input.

The idea of learning the right representation for the data provides one perspective on deep learning. Another perspective on deep learning is that it allows the computer to learn a multi-step computer program. Each layer of the representation can be thought of as the state of the computer’s memory after executing another set of instructions in parallel. Networks with greater depth can execute more instructions in sequence. Being able to execute instructions sequentially offers great power because later instructions can refer back to the results of earlier instructions. According to this view of deep learning, not all of the information in a layer’s representation of the input necessarily encodes factors of variation that explain the input. The representation is also used to store state information that helps to execute a program that can make sense of the input. This state information could be analogous to a counter or pointer in a traditional computer program. It has nothing to do with the content of the input specifically, but it helps the model to organize its processing.

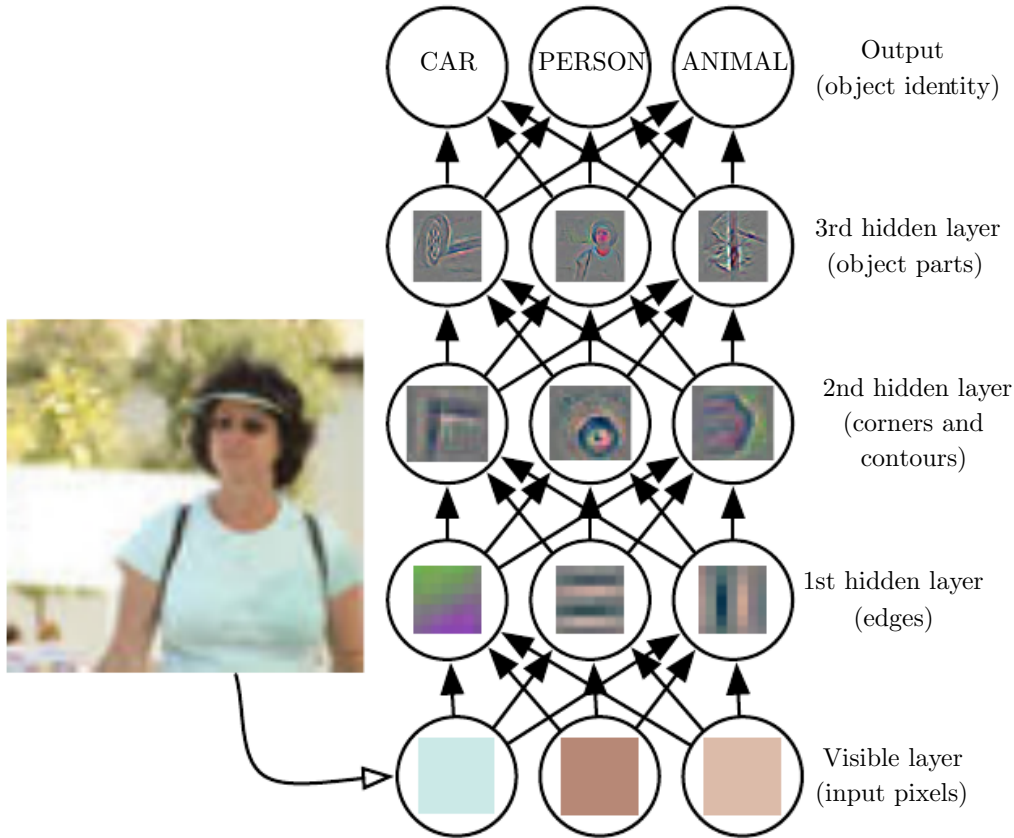


Figure 1.2: Illustration of a deep learning model. It is difficult for a computer to understand the meaning of raw sensory input data, such as this image represented as a collection of pixel values. The function mapping from a set of pixels to an object identity is very complicated. Learning or evaluating this mapping seems insurmountable if tackled directly. Deep learning resolves this difficulty by breaking the desired complicated mapping into a series of nested simple mappings, each described by a different layer of the model. The input is presented at the *visible layer*, so named because it contains the variables that we are able to observe. Then a series of *hidden layers* extracts increasingly abstract features from the image. These layers are called “hidden” because their values are not given in the data; instead the model must determine which concepts are useful for explaining the relationships in the observed data. The images here are visualizations of the kind of feature represented by each hidden unit. Given the pixels, the first layer can easily identify edges, by comparing the brightness of neighboring pixels. Given the first hidden layer’s description of the edges, the second hidden layer can easily search for corners and extended contours, which are recognizable as collections of edges. Given the second hidden layer’s description of the image in terms of corners and contours, the third hidden layer can detect entire parts of specific objects, by finding specific collections of contours and corners. Finally, this description of the image in terms of the object parts it contains can be used to recognize the objects present in the image. Images reproduced with permission from Zeiler and Fergus (2014).

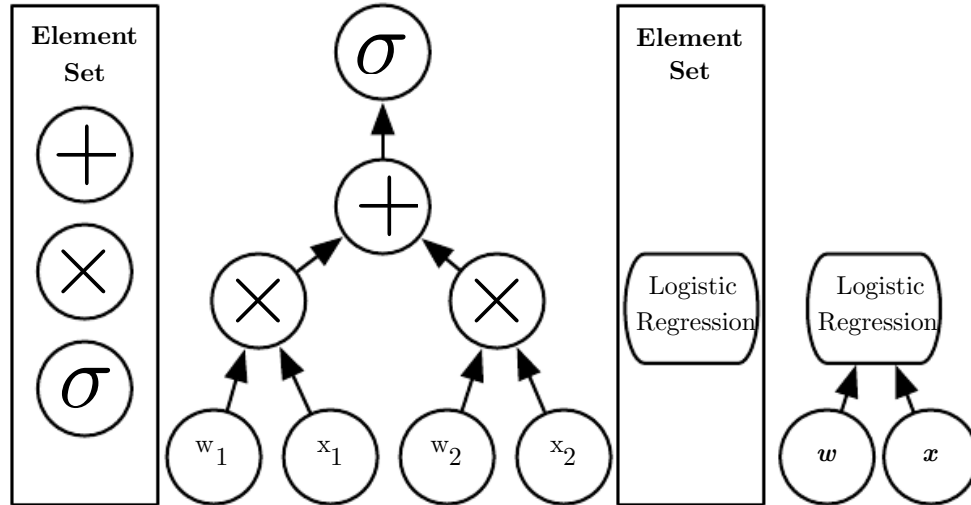


Figure 1.3: Illustration of computational graphs mapping an input to an output where each node performs an operation. Depth is the length of the longest path from input to output but depends on the definition of what constitutes a possible computational step. The computation depicted in these graphs is the output of a logistic regression model,  $\sigma(\mathbf{w}^T \mathbf{x})$ , where  $\sigma$  is the logistic sigmoid function. If we use addition, multiplication and logistic sigmoids as the elements of our computer language, then this model has depth three. If we view logistic regression as an element itself, then this model has depth one.

There are two main ways of measuring the depth of a model.

The first view is based on the number of sequential instructions that must be executed to evaluate the architecture. We can think of this as the length of the longest path through a flow chart that describes how to compute each of the model's outputs given its inputs. Just as two equivalent computer programs will have different lengths depending on which language the program is written in, the same function may be drawn as a flow chart with different depths depending on which functions we allow to be used as individual steps in the flow chart. Fig. 1.3 illustrates how this choice of language can give two different measurements for the same architecture.

Another approach, used by deep probabilistic models, illustrates not the depth of the computational graph but the depth of the graph describing how concepts are related to each other. In this case, the depth of the flow-chart of the computations needed to compute the representation of each concept may be much deeper than the graph of the concepts themselves. This is because the system's understanding of the simpler concepts can be refined given information about the more complex concepts. For example, an AI system observing an image of a face with one eye in shadow may initially only see one eye. After detecting that a face is present, it can then infer that a second eye is probably present as well. In this case, the graph of concepts only includes two layers—a layer for eyes and a layer for faces—but the

graph of computations includes  $2n$  layers if we refine our estimate of each concept given the other  $n$  times.

Because it is not always clear which of these two views—the depth of the computational graph, or the depth of the probabilistic modeling graph—is most relevant, and because different people choose different sets of smallest elements from which to construct their graphs, there is no single correct value for the depth of an architecture, just as there is no single correct value for length of a computer program. Nor is there a consensus about how much depth a model requires to qualify as “deep.” However, deep learning can safely be regarded as the study of models that either involve a greater amount of composition of learned functions or learned concepts than traditional machine learning does.

To summarize, deep learning, the subject of this book, is an approach to AI. Specifically, it is a type of machine learning, a technique that allows computer systems to improve with experience and data. According to the authors of this book, machine learning is the only viable approach to building AI systems that can operate in complicated, real-world environments. Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts and representations, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones. Fig. 1.4 illustrates the relationship between these different AI disciplines. Fig. 1.5 gives a high-level schematic of how each works.

## 1.1 Who Should Read This Book?

This book can be useful for a variety of readers, but we wrote it with two main target audiences in mind. One of these target audiences is university students (undergraduate or graduate) learning about machine learning, including those who are beginning a career in deep learning and artificial intelligence research. The other target audience is software engineers who do not have a machine learning or statistics background, but want to rapidly acquire one and begin using deep learning in their product or platform. Software engineers working in a wide variety of industries are likely to find deep learning to be useful, as it has already proven successful in many areas including computer vision, speech and audio processing, natural language processing, robotics, bioinformatics and chemistry, video games, search engines, online advertising and finance.

This book has been organized into three parts in order to best accommodate a variety of readers. Part 1 introduces basic mathematical tools and machine learning concepts. Part 2 describes the most established deep learning algorithms that are essentially solved technologies. Part 3 describes more speculative ideas



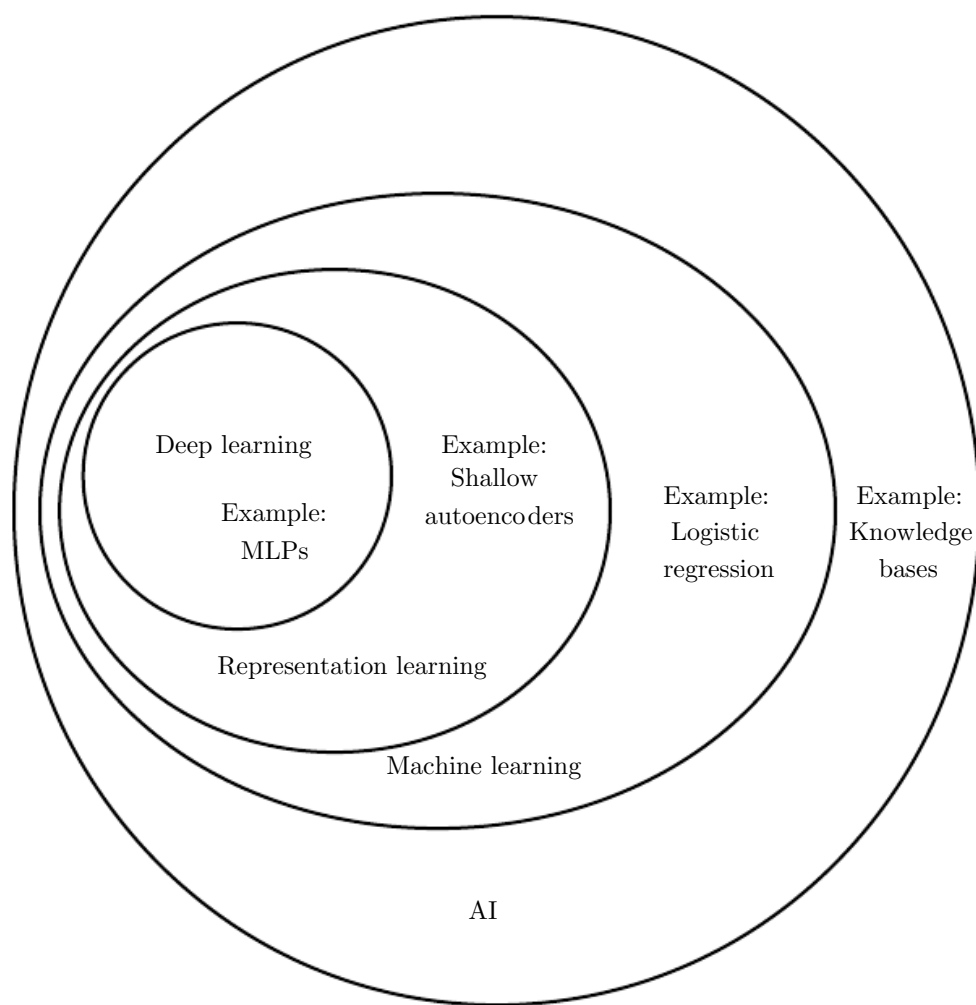


Figure 1.4: A Venn diagram showing how deep learning is a kind of representation learning, which is in turn a kind of machine learning, which is used for many but not all approaches to AI. Each section of the Venn diagram includes an example of an AI technology.

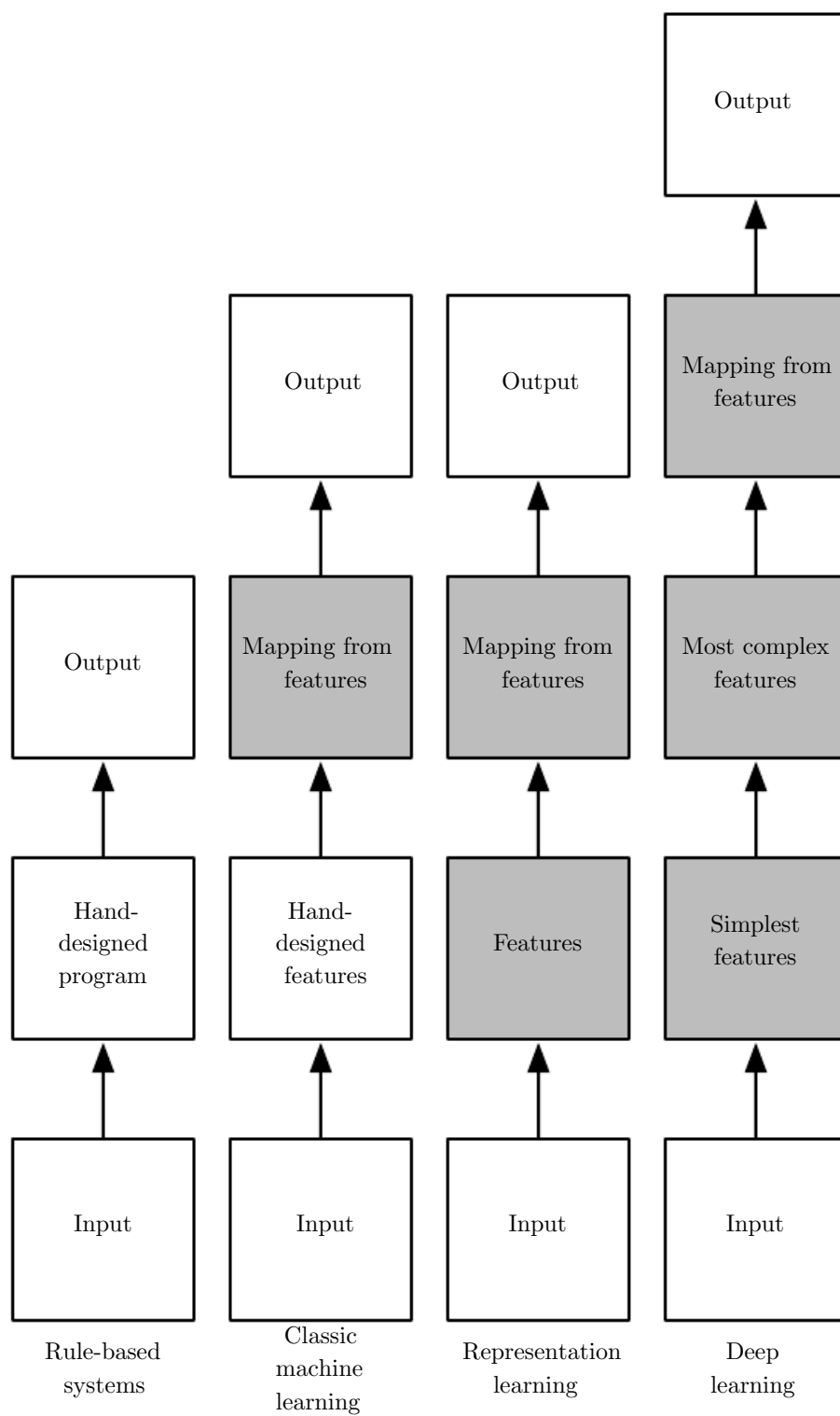


Figure 1.5: Flow-charts showing how the different parts of an AI system relate to each other within different AI disciplines. Shaded boxes indicate components that are able to learn from data.

that are widely believed to be important for future research in deep learning.

Readers should feel free to skip parts that are not relevant given their interests or background. Readers familiar with linear algebra, probability, and fundamental machine learning concepts can skip part 1, for example, while readers who just want to implement a working system need not read beyond part 2.

We do assume that all readers come from a computer science background. We assume familiarity with programming, a basic understanding of computational performance issues, complexity theory, introductory level calculus and some of the terminology of graph theory.

## 1.2 Historical Trends in Deep Learning

It is easiest to understand deep learning with some historical context. Rather than providing a detailed history of deep learning, we identify a few key trends:

- Deep learning has had a long and rich history, but has gone by many names reflecting different philosophical viewpoints, and has waxed and waned in popularity.
- Deep learning has become more useful as the amount of available training data has increased.
- Deep learning models have grown in size over time as computer hardware and software infrastructure for deep learning has improved.
- Deep learning has solved increasingly complicated applications with increasing accuracy over time.

### 1.2.1 The Many Names and Changing Fortunes of Neural Networks

We expect that many readers of this book have heard of deep learning as an exciting new technology, and are surprised to see a mention of “history” in a book about an emerging field. In fact, deep learning has a long and rich history. Deep learning only *appears* to be new, because it was relatively unpopular for several years preceding its current popularity, and because it has gone through many different names. While the term “deep learning” is relatively new, the field dates back to the 1950s. The field has been rebranded many times, reflecting the influence of different researchers and different perspectives.

A comprehensive history of deep learning is beyond the scope of this pedagogical textbook. However, some basic context is useful for understanding deep learning. Broadly speaking, there have been three waves of development of deep learning:

deep learning known as *cybernetics* in the 1940s-1960s, deep learning known as *connectionism* in the 1980s-1990s, and the current resurgence under the name deep learning beginning in 2006. This is quantitatively illustrated in Fig. 1.6.

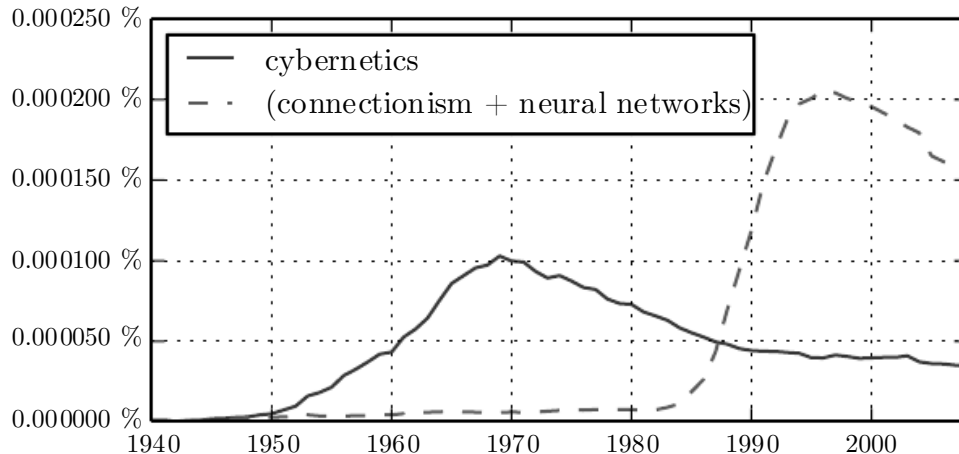


Figure 1.6: The figure shows two of the three historical waves of artificial neural nets research. The first wave started with cybernetics in the 1940-1960’s, with the perceptron (Rosenblatt, 1958) to train a single neuron. The second wave started with the connectionist approach of the 1980-1995 period, with back-propagation (Rumelhart *et al.*, 1986a) to train a neural network with one or two hidden layers. The current and third wave, deep learning, started around 2006 (Hinton *et al.*, 2006; Bengio *et al.*, 2007a; Ranzato *et al.*, 2007a), which allowed us to train very deep networks, is not shown for lack of books on the subject. The figure shows on the vertical axis the Google books  $n$ -grams relative counts for “cybernetics” (solid curve) and “connectionism” or “neural networks” (dashed curve), for years (horizontal axis) between 1940 and 2004. Keep in mind that the scientific advances tend to happen years before they are discussed in many places in books, hence these curves incorporate a lag of several years compared to the scientific activity itself.

Some of the earliest learning algorithms we recognize today were intended to be computational models of biological learning, i.e. models of how learning happens or could happen in the brain. As a result, one of the names that deep learning has gone by is *artificial neural networks* (ANNs). The corresponding perspective on deep learning models is that they are engineered systems inspired by the biological brain (whether the human brain or the brain of another animal). The neural perspective on deep learning is motivated by two main ideas. One idea is that the brain provides a proof by example that intelligent behavior is possible, and a conceptually straightforward path to building intelligence is to reverse engineer the computational principles behind the brain and duplicate its functionality. Another perspective is that it would be deeply interesting to understand the brain and the principles that underlie human intelligence, so machine learning models that shed light on these basic scientific questions are useful apart from their ability to solve

engineering applications.

The modern term “deep learning” goes beyond the neuroscientific perspective on the current breed of machine learning models. It appeals to a more general principle of learning *multiple levels of composition*, which can be applied in machine learning frameworks that are not necessarily neurally inspired.

The earliest predecessors of modern deep learning were simple linear models motivated from a neuroscientific perspective. These models were designed to take a set of  $n$  input values  $x_1, \dots, x_n$  and associate them with an output  $y$ . These models would learn a set of weights  $w_1, \dots, w_n$  and compute their output  $f(\mathbf{x}, \mathbf{w}) = x_1 w_1 + \dots + x_n w_n$ . This first wave of neural networks research was known as *cybernetics*, as illustrated in Fig. 1.6.

The McCulloch-Pitts Neuron (McCulloch and Pitts, 1943) was an early model of brain function. This linear model could recognize two different categories of inputs by testing whether  $f(\mathbf{x}, \mathbf{w})$  is positive or negative. Of course, for the model to correspond to the desired definition of the categories, the weights needed to be set correctly. These weights could be set by the human operator. In the 1950s, the perceptron (Rosenblatt, 1958, 1962) became the first model that could learn the weights defining the categories given examples of inputs from each category. The Adaptive Linear Element (ADALINE), which dates from about the same time, simply returned the value of  $f(\mathbf{x})$  itself to predict a real number (Widrow and Hoff, 1960), and could also learn to predict these numbers from data.

These simple learning algorithms greatly affected the modern landscape of machine learning. The training algorithm used to adapt the weights of the ADALINE was a special case of an algorithm called *stochastic gradient descent*. Slightly modified versions of the stochastic gradient descent algorithm remain the dominant training algorithms for deep learning models today.

Models based on the  $f(\mathbf{x}, \mathbf{w})$  used by the perceptron and ADALINE are called *linear models*. These models remain some of the most widely used machine learning models, though in many cases they are *trained* in different ways than the original models were trained.

Linear models have many limitations. Most famously, they cannot learn the XOR function, where  $f([0, 1], \mathbf{w}) = 1$  and  $f([1, 0], \mathbf{w}) = 1$  but  $f([1, 1], \mathbf{w}) = 0$  and  $f([0, 0], \mathbf{w}) = 0$ . Critics who observed these flaws in linear models caused a backlash against biologically inspired learning in general (Minsky and Papert, 1969). This was the first major dip in the popularity of neural networks.

Today, neuroscience is regarded as an important source of inspiration for deep learning researchers, but it is no longer the predominant guide for the field.

The main reason for the diminished role of neuroscience in deep learning research today is that we simply do not have enough information about the brain to use it as a guide. To obtain a deep understanding of the actual algorithms used

by the brain, we would need to be able to monitor the activity of (at the very least) thousands of interconnected neurons simultaneously. Because we are not able to do this, we are far from understanding even some of the most simple and well-studied parts of the brain (Olshausen and Field, 2005).

Neuroscience has given us a reason to hope that a single deep learning algorithm can solve many different tasks. Neuroscientists have found that ferrets can learn to “see” with the auditory processing region of their brain if their brains are rewired to send visual signals to that area (Von Melchner *et al.*, 2000). This suggests that much of the mammalian brain might use a single algorithm to solve most of the different tasks that the brain solves. Before this hypothesis, machine learning research was more fragmented, with different communities of researchers studying natural language processing, vision, motion planning and speech recognition. Today, these application communities are still separate, but it is common for deep learning research groups to study many or even all of these application areas simultaneously.

We are able to draw some rough guidelines from neuroscience. The basic idea of having many computational units that become intelligent only via their interactions with each other is inspired by the brain. The Neocognitron (Fukushima, 1980) introduced a powerful model architecture for processing images that was inspired by the structure of the mammalian visual system and later became the basis for the modern convolutional network (LeCun *et al.*, 1998b), as we will see in Chapter 9.10. Most neural networks today are based on a model neuron called the *rectified linear unit*. These units were developed from a variety of viewpoints, with (Nair and Hinton, 2010b) and Glorot *et al.* (2011a) citing neuroscience as an influence, and Jarrett *et al.* (2009a) citing more engineering-oriented influences. While neuroscience is an important source of inspiration, it need not be taken as a rigid guide. We know that actual neurons compute very different functions than modern rectified linear units, but greater neural realism has not yet found a machine learning value or interpretation. Also, while neuroscience has successfully inspired several neural network *architectures*, we do not yet know enough about biological learning for neuroscience to offer much guidance for the *learning algorithms* we use to train these architectures.

Media accounts often emphasize the similarity of deep learning to the brain. While it is true that deep learning researchers are more likely to cite the brain as an influence than researchers working in other machine learning fields such as kernel machines or Bayesian statistics, one should not view deep learning as an attempt to simulate the brain. Modern deep learning draws inspiration from many fields, especially applied math fundamentals like linear algebra, probability, information theory, and numerical optimization. While some deep learning researchers cite neuroscience as an important influence, others are not concerned with neuroscience at all.

It is worth noting that the effort to understand how the brain works on an algorithmic level is alive and well. This endeavor is primarily known as “computational neuroscience” and is a separate field of study from deep learning. It is common for researchers to move back and forth between both fields. The field of deep learning is primarily concerned with how to build computer systems that are able to successfully solve tasks requiring intelligence, while the field of computational neuroscience is primarily concerned with building more accurate models of how the brain actually works.

In the 1980s, the second wave of neural network research emerged in great part via a movement called *connectionism* or *parallel distributed processing* (Rumelhart *et al.*, 1986d). Connectionism arose in the context of cognitive science. Cognitive science is an interdisciplinary approach to understanding the mind, combining multiple different levels of analysis. During the early 1980s, most cognitive scientists studied models of symbolic reasoning. Despite their popularity, symbolic models were difficult to explain in terms of how the brain could actually implement them using neurons. The connectionists began to study models of cognition that could actually be grounded in neural implementations, reviving many ideas dating back to the work of psychologist Donald Hebb in the 1940s (Hebb, 1949).

The central idea in connectionism is that a large number of simple computational units can achieve intelligent behavior when networked together. This insight applies equally to neurons in biological nervous systems and to hidden units in computational models.

Several key concepts arose during the connectionism movement of the 1980s that remain central to today’s deep learning.

One of these concepts is that of *distributed representation*. This is the idea that each input to a system should be represented by many features, and each feature should be involved in the representation of many possible inputs. For example, suppose we have a vision system that can recognize cars, trucks, and birds and these objects can each be red, green, or blue. One way of representing these inputs would be to have a separate neuron or hidden unit that activates for each of the nine possible combinations: red truck, red car, red bird, green truck, and so on. This requires nine different neurons, and each neuron must independently learn the concept of color and object identity. One way to improve on this situation is to use a distributed representation, with three neurons describing the color and three neurons describing the object identity. This requires only six neurons total instead of nine, and the neuron describing redness is able to learn about redness from images of cars, trucks and birds, not only from images of one specific category of objects. The concept of distributed representation is central to this book, and will be described in greater detail in Chapter 14.

Another major accomplishment of the connectionist movement was the suc-

successful use of back-propagation to train deep neural networks with internal representations and the popularization of the back-propagation algorithm (Rumelhart *et al.*, 1986a; LeCun, 1987). This algorithm has waxed and waned in popularity but as of this writing is currently the dominant approach to training deep models.

The second wave of neural networks research lasted until the mid-1990s. At that point, the popularity of neural networks declined again. This was in part due to a negative reaction to the failure of neural networks (and AI research in general) to fulfill excessive promises made by a variety of people seeking investment in neural network-based ventures, but also due to improvements in other fields of machine learning: kernel machines (Boser *et al.*, 1992; Cortes and Vapnik, 1995; Schölkopf *et al.*, 1999) and graphical models (Jordan, 1998).

Kernel machines enjoy many nice theoretical guarantees. In particular, training a kernel machine is a *convex optimization problem* (this will be explained in more detail in Chapter 4) which means that the training process can be guaranteed to find the optimal model efficiently. This made kernel machines very amenable to software implementations that “just work” without much need for the human operator to understand the underlying ideas. Soon, most machine learning applications consisted of manually designing good features to provide to a kernel machine for each different application area.

During this time, neural networks continued to obtain impressive performance on some tasks (LeCun *et al.*, 1998c; Bengio *et al.*, 2001a). The Canadian Institute for Advanced Research (CIFAR) helped to keep neural networks research alive via its Neural Computation and Adaptive Perception research initiative. This program united machine research groups led by Geoffrey Hinton at University of Toronto, Yoshua Bengio at University of Montreal, and Yann LeCun at New York University. It had a multi-disciplinary nature that also included neuroscientists and experts in human and computer vision.

At this point in time, deep networks were generally believed to be very difficult to train. We now know that algorithms that have existed since the 1980s work quite well, but this was not apparent circa 2006. The issue is perhaps simply that these algorithms were too computationally costly to allow much experimentation with the hardware available at the time.

The third wave of neural networks research began with a breakthrough in 2006. Geoffrey Hinton showed that a kind of neural network called a deep belief network could be efficiently trained using a strategy called greedy layer-wise pretraining (Hinton *et al.*, 2006), which will be described in more detail in Chapter 14.1. The other CIFAR-affiliated research groups quickly showed that the same strategy could be used to train many other kinds of deep networks (Bengio *et al.*, 2007a; Ranzato *et al.*, 2007a) and systematically helped to improve generalization on test examples. This wave of neural networks research popularized the use of the



term *deep learning* to emphasize that researchers were now able to train deeper neural networks than had been possible before, and to emphasize the theoretical importance of depth (Bengio and LeCun, 2007a; Delalleau and Bengio, 2011; Pascanu *et al.*, 2014a; Montufar *et al.*, 2014). Deep neural networks displaced kernel machines with manually designed features for several important application areas during this time—in part because the time and memory cost of training a kernel machine is quadratic in the size of the dataset, and datasets grew to be large enough for this cost to outweigh the benefits of convex optimization. This third wave of popularity of neural networks continues to the time of this writing, though the focus of deep learning research has changed dramatically within the time of this wave. The third wave began with a focus on new unsupervised learning techniques and the ability of deep models to generalize well from small datasets, but today there is more interest in much older supervised learning algorithms and the ability of deep models to leverage large labeled datasets.

### 1.2.2 Increasing Dataset Sizes

One may wonder why deep learning has only recently become recognized as a crucial technology if it has existed since the 1950s. Deep learning has been successfully used in commercial applications since the 1990s, but was often regarded as being more of an art than a technology and something that only an expert could use, until recently. It is true that some skill is required to get good performance from a deep learning algorithm. Fortunately, the amount of skill required reduces as the amount of training data increases. The learning algorithms reaching human performance on complex tasks today are nearly identical to the learning algorithms that struggled to solve toy problems in the 1980s, though the models we train with these algorithms have undergone changes that simplify the training of very deep architectures. The most important new development is that today we can provide these algorithms with the resources they need to succeed. Fig. 1.7 shows how the size of benchmark datasets has increased remarkably over time. This trend is driven by the increasing digitization of society. As more and more of our activities take place on computers, more and more of what we do is recorded. As our computers are increasingly networked together, it becomes easier to centralize these records and curate them into a dataset appropriate for machine learning applications. The age of “Big Data” has made machine learning much easier because the key burden of statistical estimation—generalizing well to new data after observing only a small amount of data—has been considerably lightened. As of 2015, a rough rule of thumb is that a supervised deep learning algorithm will generally achieve acceptable performance with around 5,000 labeled examples per category, and will match or exceed human performance when trained with a dataset containing at least 10 million labeled examples. Working successfully with datasets smaller

than this is an important research area, focusing in particular on how we can take advantage of large quantities of unlabeled examples, with unsupervised or semi-supervised learning.

### 1.2.3 Increasing Model Sizes

Another key reason that neural networks are wildly successful today after enjoying comparatively little success since the 1980s is that we have the computational resources to run much larger models today. One of the main insights of connectionism is that animals become intelligent when many of their neurons work together. An individual neuron or small collection of neurons is not particularly useful.

Biological neurons are not especially densely connected. As seen in Fig. 1.9, our machine learning models have had a number of connections per neuron that was within an order of magnitude of even mammalian brains for decades.

In terms of the total number of neurons, neural networks have been astonishingly small until quite recently, as shown in Fig. 1.10. Since the introduction of hidden units, artificial neural networks have doubled in size roughly every 2.4 years. This growth is driven by faster computers with larger memory and by the availability of larger datasets. Larger networks are able to achieve higher accuracy on more complex tasks. This trend looks set to continue for decades. Unless new technologies allow faster scaling, artificial neural networks will not have the same number of neurons as the human brain until at least the 2050s. Biological neurons may represent more complicated functions than current artificial neurons, so biological neural networks may be even larger than this plot portrays.

In retrospect, it is not particularly surprising that neural networks with fewer neurons than a leech were unable to solve sophisticated artificial intelligence problems. Even today's networks, which we consider quite large from a computational systems point of view, are smaller than the nervous system of even relatively primitive vertebrate animals like frogs.

The increase in model size over time, due to the availability of faster CPUs, the advent of general purpose GPUs, faster network connectivity and better software infrastructure for distributed computing, is one of the most important trends in the history of deep learning. This trend is generally expected to continue well into the future.

### 1.2.4 Increasing Accuracy, Complexity and Real-World Impact

Since the 1980s, deep learning has consistently improved in its ability to provide accurate recognition or prediction. Moreover, deep learning has consistently been applied with success to broader and broader sets of applications.

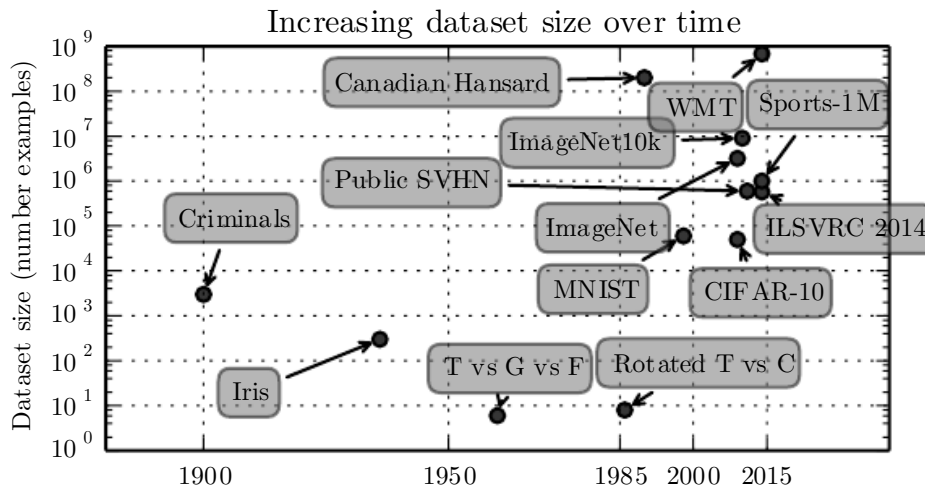


Figure 1.7: Dataset sizes have increased greatly over time. In the early 1900s, statisticians studied datasets using hundreds or thousands of manually compiled measurements (Garson, 1900; Gosset, 1908; Anderson, 1935; Fisher, 1936). In the 1950s through 1980s, the pioneers of biologically-inspired machine learning often worked with small, synthetic datasets, such as low-resolution bitmaps of letters, that were designed to incur low computational cost and demonstrate that neural networks were able to learn specific kinds of functions (Widrow and Hoff, 1960; Rumelhart *et al.*, 1986b). In the 1980s and 1990s, machine learning became more statistical in nature and began to leverage larger datasets containing tens of thousands of examples such as the MNIST dataset (shown in Fig. 1.8) of scans of handwritten numbers (LeCun *et al.*, 1998c). In the first decade of the 2000s, more sophisticated datasets of this same size, such as the CIFAR-10 dataset (Krizhevsky and Hinton, 2009) continued to be produced. Toward the end of that decade and throughout the first half of the 2010s, significantly larger datasets, containing hundreds of thousands to tens of millions of examples, completely changed what was possible with deep learning. These datasets included the public Street View House Numbers dataset (Netzer *et al.*, 2011), various versions of the ImageNet dataset (Deng *et al.*, 2009, 2010a; Russakovsky *et al.*, 2014a), and the Sports-1M dataset (Karpathy *et al.*, 2014). At the top of the graph, we see that datasets of translated sentences, such as IBM’s dataset constructed from the Canadian Hansard (Brown *et al.*, 1990) and the WMT 2014 English to French dataset (Schwenk, 2014) are typically far ahead of other dataset sizes.

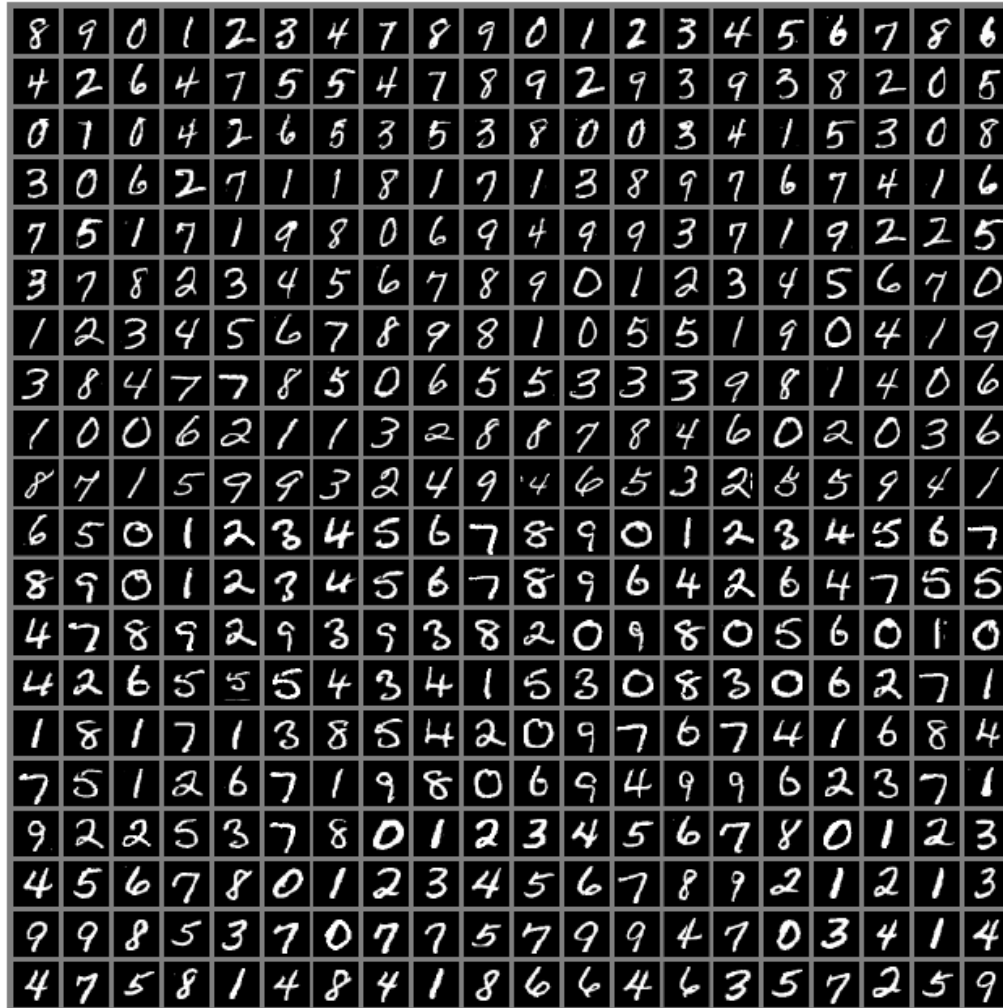


Figure 1.8: Example inputs from the MNIST dataset. The “NIST” stands for National Institute of Standards and Technology, the agency that originally collected this data. The “M” stands for “modified,” since the data has been preprocessed for easier use with machine learning algorithms. The MNIST dataset consists of scans of handwritten digits and associated labels describing which digit 0-9 is contained in each image. This simple classification problem is one of the simplest and most widely used tests in deep learning research. It remains popular despite being quite easy for modern techniques to solve. Geoffrey Hinton has described it as “the *drosophila* of machine learning,” meaning that it allows machine learning researchers to study their algorithms in controlled laboratory conditions, much as biologists often study fruit flies.

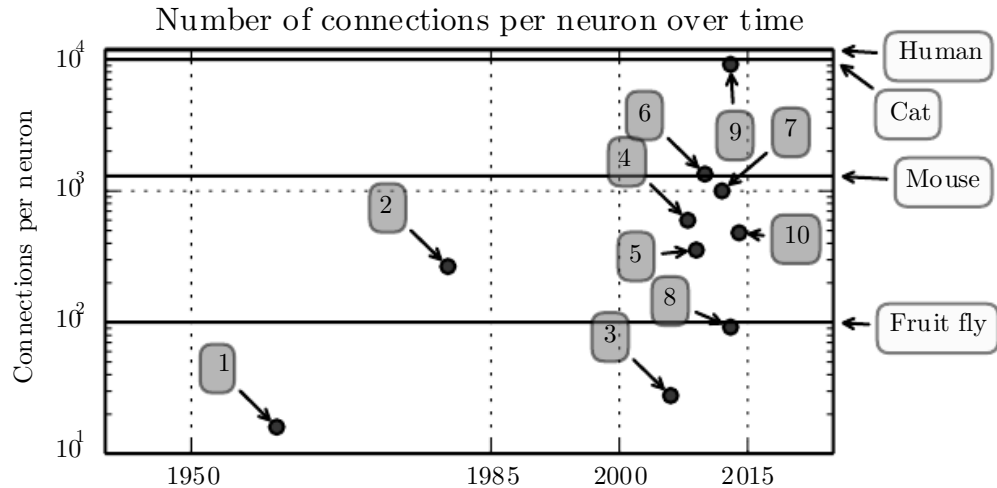


Figure 1.9: Initially, the number of connections between neurons in artificial neural networks was limited by hardware capabilities. Today, the number of connections between neurons is mostly a design consideration. Some artificial neural networks have nearly as many connections per neuron as a cat, and it is quite common for other neural networks to have as many connections per neuron as smaller mammals like mice. Even the human brain does not have an exorbitant amount of connections per neuron. The sparse connectivity of biological neural networks means that our artificial networks are able to match the performance of biological neural networks despite limited hardware. Modern neural networks are much smaller than the brains of any vertebrate animal, but we typically train each network to perform just one task, while an animal’s brain has different areas devoted to different tasks. Biological neural network sizes from Wikipedia (2015).

1. Adaptive Linear Element (Widrow and Hoff, 1960)
2. Neocognitron (Fukushima, 1980)
3. GPU-accelerated convolutional network (Chellapilla *et al.*, 2006)
4. Deep Boltzmann machines (Salakhutdinov and Hinton, 2009a)
5. Unsupervised convolutional network (Jarrett *et al.*, 2009b)
6. GPU-accelerated multilayer perceptron (Ciresan *et al.*, 2010)
7. Distributed autoencoder (Le *et al.*, 2012)
8. Multi-GPU convolutional network (Krizhevsky *et al.*, 2012a)
9. COTS HPC unsupervised convolutional network (Coates *et al.*, 2013)
10. GoogLeNet (Szegedy *et al.*, 2014a)

The earliest deep models were used to recognize individual objects in tightly cropped, extremely small images (Rumelhart *et al.*, 1986a). Since then there has been a gradual increase in the size of images neural networks could process. Modern object recognition networks process rich high-resolution photographs and do not have a requirement that the photo be cropped near the object to be recognized (Krizhevsky *et al.*, 2012b). Similarly, the earliest networks could only recognize two kinds of objects (or in some cases, the absence or presence of a single kind of object), while these modern networks typically recognize at least 1,000 different categories of objects. The largest contest in object recognition is the ImageNet Large-Scale Visual Recognition Competition held each year. A dramatic moment in the meteoric rise of deep learning came when a convolutional network won this challenge for the first time and by a wide margin, bringing down the state-of-the-art error rate from 26.1% to 15.3% (Krizhevsky *et al.*, 2012b). Since then, these competitions are consistently won by deep convolutional nets, and as of this writing, advances in deep learning had brought the latest error rate in this contest down to 6.5% as shown in Fig. 1.11, using even deeper networks (Szegedy *et al.*, 2014a). Outside the framework of the contest, this error rate has now dropped to below 5% (Ioffe and Szegedy, 2015; Wu *et al.*, 2015).

Deep learning has also had a dramatic impact on speech recognition. After improving throughout the 1990s, the error rates for speech recognition stagnated starting in about 2000. The introduction of deep learning (Dahl *et al.*, 2010; Deng *et al.*, 2010b; Seide *et al.*, 2011; Hinton *et al.*, 2012a) to speech recognition resulted in a sudden drop of error rates by up to half! We will explore this history in more detail in Sec. 12.3.

Deep networks have also had spectacular successes for pedestrian detection and image segmentation (Sermanet *et al.*, 2013; Farabet *et al.*, 2013a; Couprie *et al.*, 2013) and yielded superhuman performance in traffic sign classification (Ciresan *et al.*, 2012).

At the same time that the scale and accuracy of deep networks has increased, so has the complexity of the tasks that they can solve. Goodfellow *et al.* (2014d) showed that neural networks could learn to output an entire sequence of characters transcribed from an image, rather than just identifying a single object. Previously, it was widely believed that this kind of learning required labeling of the individual elements of the sequence (Gülçehre and Bengio, 2013). Since this time, a neural network designed to model sequences, the Long Short-Term Memory or LSTM (Hochreiter and Schmidhuber, 1997), has enjoyed an explosion in popularity. LSTMs and related models are now used to model relationships between *sequences* and other *sequences* rather than just fixed inputs. This sequence-to-sequence learning seems to be on the cusp of revolutionizing another application: machine translation (Sutskever *et al.*, 2014a; Bahdanau *et al.*, 2014).

This trend of increasing complexity has been pushed to its logical conclusion with the introduction of the Neural Turing Machine (Graves *et al.*, 2014a), a neural network that can learn entire programs. This neural network has been shown to be able to learn how to sort lists of numbers given examples of scrambled and sorted sequences. This self-programming technology is in its infancy, but in the future could in principle be applied to nearly any task.

Another crowning achievement of deep learning is its extension to the domain of *reinforcement learning*. In the context of reinforcement learning, an autonomous agent must learn to perform a task by trial and error, without any guidance from the human operator. Deep Mind demonstrated that a reinforcement learning system based on deep learning is capable of learning to play Atari video games, reaching human-level performance on many tasks (Mnih *et al.*, 2015).

Many of these applications of deep learning are highly profitable, given enough data to apply deep learning to. Deep learning is now used by many top technology companies including Google, Microsoft, Facebook, IBM, Baidu, Apple, Adobe, Netflix, NVIDIA and NEC.

Deep learning has also made contributions back to other sciences. Modern convolutional networks for object recognition provide a model of visual processing that neuroscientists can study (DiCarlo, 2013). Deep learning also provides useful tools for processing massive amounts of data and making useful predictions in scientific fields. It has been successfully used to predict how molecules will interact in order to help pharmaceutical companies design new drugs (Dahl *et al.*, 2014), to search for subatomic particles (Baldi *et al.*, 2014), and to automatically parse microscope images used to construct a 3-D map of the human brain (Knowles-Barley *et al.*, 2014). We expect deep learning to appear in more and more scientific fields in the future.

In summary, deep learning is an approach to machine learning that has drawn heavily on our knowledge of the human brain, statistics and applied math as it developed over the past several decades. In recent years, it has seen tremendous growth in its popularity and usefulness, due in large part to more powerful computers, larger datasets and techniques to train deeper networks. The years ahead are full of challenges and opportunities to improve deep learning even further and bring it to new frontiers.

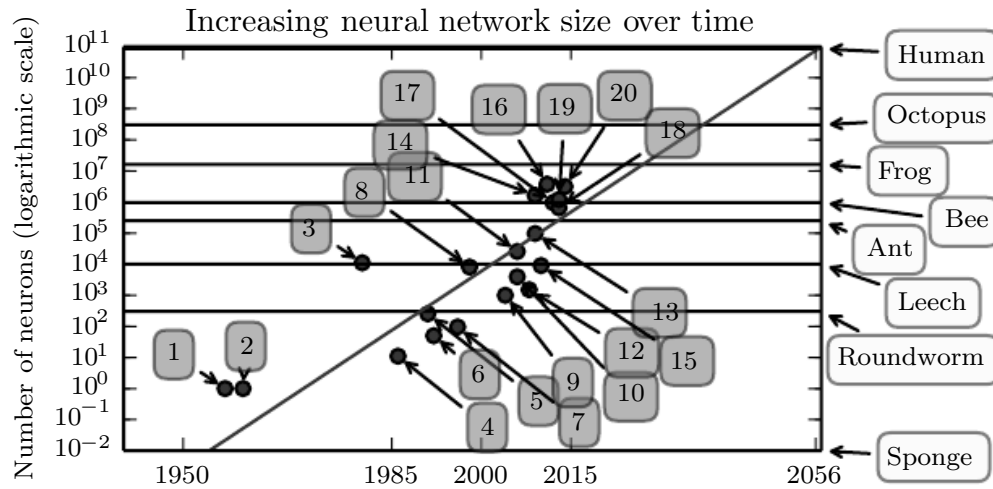


Figure 1.10: Since the introduction of hidden units, artificial neural networks have doubled in size roughly every 2.4 years. Biological neural network sizes from Wikipedia (2015).

1. Perceptron (Rosenblatt, 1958, 1962)
2. Adaptive Linear Element (Widrow and Hoff, 1960)
3. Neocognitron (Fukushima, 1980)
4. Early back-propagation network (Rumelhart *et al.*, 1986b)
5. Recurrent neural network for speech recognition (Robinson and Fallside, 1991)
6. Multilayer perceptron for speech recognition (Bengio *et al.*, 1991)
7. Mean field sigmoid belief network (Saul *et al.*, 1996)
8. LeNet-5 (LeCun *et al.*, 1998b)
9. Echo state network (Jaeger and Haas, 2004)
10. Deep belief network (Hinton *et al.*, 2006)
11. GPU-accelerated convolutional network (Chellapilla *et al.*, 2006)
12. Deep Boltzmann machines (Salakhutdinov and Hinton, 2009a)
13. GPU-accelerated deep belief network (Raina *et al.*, 2009)
14. Unsupervised convolutional network (Jarrett *et al.*, 2009b)
15. GPU-accelerated multilayer perceptron (Ciresan *et al.*, 2010)
16. OMP-1 network (Coates and Ng, 2011)
17. Distributed autoencoder (Le *et al.*, 2012)
18. Multi-GPU convolutional network (Krizhevsky *et al.*, 2012a)
19. COTS HPC unsupervised convolutional network (Coates *et al.*, 2013)
20. GoogLeNet (Szegedy *et al.*, 2014a)



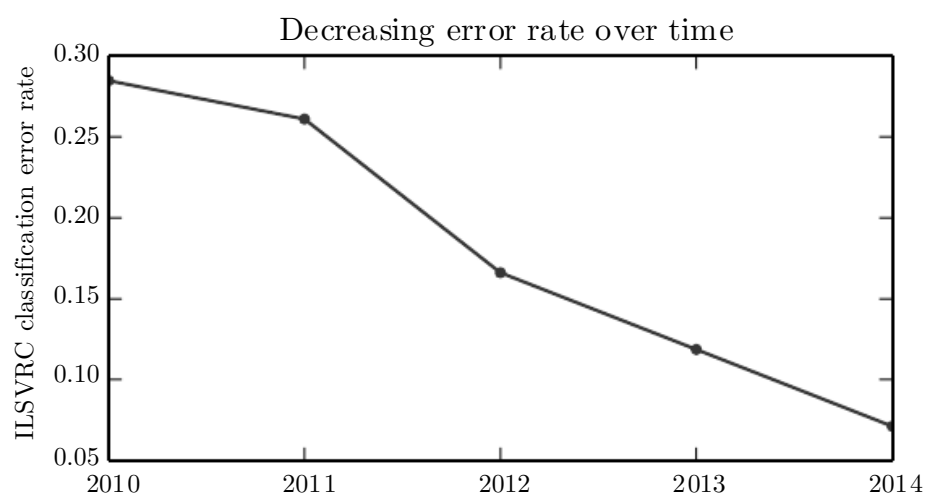


Figure 1.11: Since deep networks reached the scale necessary to compete in the ImageNet Large Scale Visual Recognition, they have consistently won the competition every year, and yielded lower and lower error rates each time. Data from Russakovsky *et al.* (2014b).