1.1

* AI can be described as the effort to automate intellectual tasks normally performed by humans
* As such, AI is a general field that encompasses machine learning and deep learning, but that also includes many more approaches that may not involve any learning
* In fact, for a fairly long time, most experts believed that human-level artificial intelligence could be achieved by having programmers handcraft a sufficiently large set of explicit rules for manipulating knowledge stored in explicit databases. This approach is known as *symbolic AI*.
* A new approach arose to take symbolic AI’s place: *machine learning*

1.1.2

* *Analytical Engine*: the first-known general-purpose mechanical computer.
  + It was merely meant as a way to use mechanical operations to automate certain computations from the field of mathematical analysis—hence the name Analytical Engine
* The usual way to make a computer do useful work is to have a human programmer write down rules—a computer program—to be followed to turn input data into appropriate answers
* A machine learning system is trained rather than explicitly programmed

1.1.3.

* To do machine learning, we need three things:
  + Input data points—For instance, if the task is speech recognition, these data points could be sound files of people speaking. If the task is image tagging, they could be pictures.
  + Examples of the expected output—In a speech-recognition task, these could be human generated transcripts of sound files. In an image task, expected outputs could be tags such as “dog,” “cat,” and so on.
  + A way to measure whether the algorithm is doing a good job—This is necessary in order to determine the distance between the algorithm’s current output and its expected output. The measurement is used as a feedback signal to adjust the way the algorithm works. This adjustment step is what we call learning.
* The central problem in machine learning and deep learning is to meaningfully transform data: in other words, to learn useful representations of the input data at hand—representations that get us closer to the expected output
* what’s a representation
  + it’s a different way to look at data—to represent or encode data.
* **Learning**, in the context of machine learning, describes an automatic search process for data transformations that produce useful representations of some data, guided by some feedback signal—representations that are amenable to simpler rules solving the task at hand.
* Machine learning algorithms aren’t usually creative in finding these transformations; they’re merely searching through a predefined set of operations, called a hypothesis space.

1.1.4.

How many layers contribute to a

model of the data is called the *depth* of the model.

The “deep” in “deep learning” isn’t a reference to

any kind of deeper understanding achieved by the approach; rather, it stands for

this idea of successive layers of representations

In deep learning, these layered representations are learned via models called *neural*

*networks*,

Meanwhile, other approaches to machine learning tend to focus on learning

only one or two layers of representations of the data (say, taking a pixel histogram

and then applying a classification rule); hence, they’re sometimes called *shallow*

*learning*.

You can think of a deep network as a multistage *informationdistillation*

process, where information goes through successive filters and comes out

increasingly *purified*

So that’s what deep learning is, technically: a multistage way to learn data representations.

1.1.5.

The specification of what a layer does to its input data is stored in the layer’s

*weights*, which in essence are a bunch of numbers

In technical terms, we’d say that the

transformation implemented by a layer is *parameterized* by its weights (see figure 1.7).

(Weights are also sometimes called the *parameters* of a layer.)

*learning*

means finding a set of values for the weights of all layers in a network, such that the

network will correctly map example inputs to their associated targets

you need to be able to measure how far this output is from what you

expected. This is the job of the *loss function* of the network, also sometimes called the

*objective function* or *cost function*. The loss function takes the predictions of the network

and the true target (what you wanted the network to output) and computes a distance

score, capturing how well the network has done on this specific example

The fundamental trick in deep learning is to use this score as a feedback signal to

adjust the value of the weights a little, in a direction that will lower the loss score for

the current example (see figure 1.9). This adjustment is the job of the *optimizer*, which

implements what’s called the *Backpropagation* algorithm: the central algorithm in deep

learning.

This is the *training loop*, which, repeated a sufficient

number of times (typically tens of iterations over thousands of examples), yields

weight values that minimize the loss function.

1.1.6

Niks

1.1.7

Ai winter = Funding dried up after after followed by disappointment

1.1.8

Niks

1.2

It’s safe to say that most of the machine learning algorithms used in the industry

today aren’t deep learning algorithms. Deep learning isn’t always the right tool for the

job—sometimes there isn’t enough data for deep learning to be applicable, and sometimes

the problem is better solved by a different algorithm.

1.2.1. Probabilistic Modeling

*Probabilistic modeling* is the application of the principles of statistics to data analysis.

One of the best-known algorithms in this category is the Naive Bayes algorithm

A closely related model is *logistic regression* (logreg for short), which is sometimes

considered to be the “Hello World” of modern machine learning.

logreg is a classification algorithm rather than a regression algorithm

It’s often the first thing a data scientist will try on a dataset to get a feel for the classification task at hand

1.2.2. Early Neural Networks

For a long time, the missing

piece was an efficient way to train large neural networks. This changed in the mid-

1980s, when multiple people independently rediscovered the Backpropagation algorithm—

a way to train chains of parametric operations using gradient-descent optimization

(we’ll precisely define these concepts later in the book)—and started applying

it to neural networks.

1.2.3. Kernel Methods

*Kernel methods* are a group of classification algorithms, the best known of which is the *Support Vector Machine* (SVM).

SVM is a classification algorithm that works by finding “decision

boundaries” separating two classes

SVMs proceed to find these boundaries in two steps:

1 The data is mapped to a new high-dimensional representation where the decision boundary can be expressed as a hyperplane

2 A good decision boundary (a separation hyperplane) is computed by trying to maximize the distance between the hyperplane and the closest data points from each class, a step called *maximizing the margin*. This allows the boundary to generalize well to new samples outside of the training dataset

to find good decision hyperplanes

in the new representation space, you don’t have to explicitly compute the coordinates

of your points in the new space; you just need to compute the distance

between pairs of points in that space, which can be done efficiently using a kernel

function

A *kernel function* is a computationally tractable operation that maps any two

points in your initial space to the distance between these points in your target representation

space, completely bypassing the explicit computation of the new representation

Because an SVM is a shallow

method, applying an SVM to perceptual problems requires first extracting useful representations

manually (a step called *feature engineering*), which is difficult and brittle

1.2.4. Decision trees, random forests, and gradient bosoting machines

*Decision trees* are flowchart-like structures that let you classify input data points or predict

output values given inputs

*andom Forest* algorithm introduced a robust, practical take on

decision-tree learning that involves building a large number of specialized decision trees

and then ensembling their outputs

A gradient

boosting machine, much like a random forest, is a machine learning technique

based on ensembling weak prediction models, generally decision trees. It uses *gradient*

*boosting*, a way to improve any machine learning model by iteratively training new

models that specialize in addressing the weak points of the previous models.

It may be one of the best, if not *the* best, algorithm for dealing with nonperceptual

data today.

1.2.5. Back to neural neworks

Since 2012, deep convolutional neural networks (*convnets*) have become the go-to

algorithm for all computer vision tasks; more generally, they work on all perceptual tasks.

1.2.6. What makes deep learning different

The primary reason deep learning took off so quickly is that it offered better performance

for many problems.

Deep learning also makes

problem-solving much easier, because it completely automates what used to be the

most crucial step in a machine learning workflow: feature engineering

Previous machine learning techniques—shallow learning—only involved transforming

the input data into one or two successive representation spaces, usually via simple

transformations such as high-dimensional non-linear projections (SVMs) or decision

trees. But the refined representations required by complex problems generally can’t

be attained by such techniques. As such, humans had to go to great lengths to make

the initial input data more amenable to processing by these methods: they had to

manually engineer good layers of representations for their data. This is called feature

engineering.

Deep learning, on the other hand, completely automates this step: with

deep learning, you learn all features in one pass rather than having to engineer them

yourself.

What is transformative about deep learning is that it allows a model to learn all layers of representation jointly, at the same time, rather than in succession (greedily, as it’s called).

These are the two essential characteristics of how deep learning learns from data:

the incremental, layer-by-layer way in which increasingly complex representations are developed, and the fact that these intermediate incremental representations are learned jointly, each layer

being updated to follow both the representational needs of the layer above and the

needs of the layer below

1.2.7. The modern machine learning landscape

gradient boosted trees is used for problems where structured data is available,

whereas deep learning is used for perceptual problems such as image classification.

These are the two techniques you should be the most familiar with in order to be

successful in applied machine learning today: gradient boosted trees, for shallowlearning

problems; and deep learning, for perceptual problems.

1.3. Why deep learning? Why now?

two key ideas of deep learning for computer vision—convolutional neural networks

and backpropagation

In general, three technical forces are driving advances in machine learning:

-Hardware

-Datasets and benchmarks

-Algorithmic advances

1.3.1. Hardare

Interessante stuff

1.3.2. Data

When it comes to data, in

addition to the exponential progress in storage hardware over the past 20 years (following

Moore’s law), the game changer has been the rise of the internet, making it feasible

to collect and distribute very large datasets for machine learning.

1.3.2. Algorithms

The key issue was that of gradient propagation through deep stacks of layers. The feedback signal used to train neural networks would fade away as the number of layers increased.

This changed around 2009–2010 with the advent of several simple but important

algorithmic improvements that allowed for better gradient propagation:

-Better activation functions for neural layers

-Better weight-initialization schemes, starting with layer-wise pretraining, which was

then quickly abandoned

-Better optimization schemes, such as RMSProp and Adam

1.3.4. A new wave of investment

Machine learning—in particular, deep learning—has become central to the product

strategy of these tech giants

1.3.5. the democratization of deep learning

One of the key factors driving this inflow of new faces in deep learning has been the

democratization of the toolsets used in the field.

After its release in

early 2015, Keras quickly became the go-to deep learning solution for large numbers

of new startups, graduate students, and researchers pivoting into the field

1.3.6. Will it last?

These important properties can be broadly sorted into three categories:

 Simplicity—Deep learning removes the need for feature engineering, replacing

complex, brittle, engineering-heavy pipelines with simple, end-to-end trainable

models that are typically built using only five or six different tensor operations.

 Scalability—Deep learning is highly amenable to parallelization on GPUs or TPUs,

so it can take full advantage of Moore’s law. In addition, deep learning models

are trained by iterating over small batches of data, allowing them to be trained on

datasets of arbitrary size. (The only bottleneck is the amount of parallel computational

power available, which, thanks to Moore’s law, is a fast-moving barrier.)

 Versatility and reusability—Unlike many prior machine learning approaches, deep

learning models can be trained on additional data without restarting from scratch, making them viable for continuous online learning—an important

property for very large production models. Furthermore, trained deep learning

models are repurposable and thus reusable: for instance, it’s possible to take a

deep learning model trained for image classification and drop it into a videoprocessing

pipeline. This allows us to reinvest previous work into increasingly

complex and powerful models. This also makes deep learning applicable to

fairly small datasets