1 Introduction

Making *intelligent* machines, that is, machines capable of learning and utilizing the gathered knowledge in thinking and reasoning, is a long-lived dream of human civilization. The more we know about the human brain, intelligence, and psychology, the more challenging it seems. However, despite the many obstacles and challenges in creating artificial intelligence (AI), the joint effort of researchers working in the natural, cognitive, mathematical, and computer sciences has produced impressive machinery that is already revolutionizing our daily life, industry, and science.

1.1 How do computers learn?

The ultimate goal of AI is to endow machines with the ability to conceptualize and create abstractions. Both of these features are mechanisms that underlie learning representations of knowledge and reasoning based on experience in humans. We have multiple ways of representing ideas. For example, we can encode a piece of music in a digital format on a computer, in an analog format on a vinyl disk, or we can write it down in a music score. Although the representations are entirely different, the piece of music is the same. Therefore, the properties of abstract ideas do not depend on the data source.

Furthermore, conceptualization and abstraction bring the possibility of considering various levels of details within a particular representation or the ability to switch from one level to another while preserving the relevant information [7–11]. Our brain excels at extracting abstract ideas from different representations of knowledge. In our daily lives, we constantly process information from multiple sources that represent the same concept in completely different ways. For example, we can identify the concept of a dog by seeing one, hearing or smelling it, reading the word "dog," painting a snout on someone's face, or even casting shadows with our hands that resemble the shade of a dog. This level of abstraction and conceptualization enables us to reason, connecting high-level ideas. All the properties of our brain mentioned above form what we call intelligence. Conferring these properties to a computer would result in a general problem-solving machine.

Today, we are at a point in our technological advances at which the human brain and computers have a disjoint set of tasks in which they naturally excel.[1] Some tasks are easy for computers but difficult for humans. These are problems that can be described by a list of formal, mathematical rules. Therefore, computers excel at solving logic, algebra, geometry, and optimization problems, which we can tackle with hard-coded solutions or knowledge-based AI. However, we would like to tackle problems that are not easy to present in a formal mathematical way, such as face

[1]This observation was first made in the 1980s, and it is called Moravec's paradox. As Moravec wrote in 1988 [12], "It is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility" (p. 15).

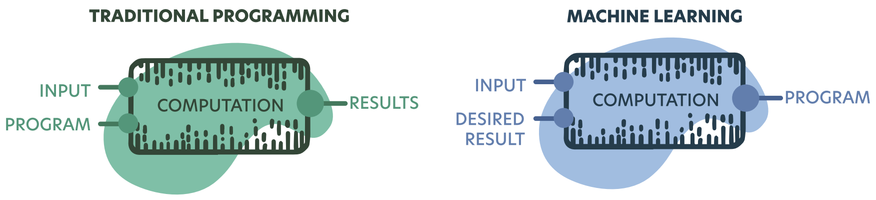

Figure 1.1 Schematic representation of the difference between traditional programming, based on the algorithmic approach, and the experience-based/data-driven approach, which is the backbone of the ML paradigm. The ML paradigm is the first step toward learning abstractions by computers through the extraction of common features from data.

recognition, or whose exact mathematical formulation is not yet known, such as detecting new quantum phases.

A particularly exciting direction is the development of algorithms that are not explicitly programmed. The main principle is to enable computers to learn from experience (or data). The shift toward this data-driven paradigm led to the birth of machine learning (ML), schematically depicted in Fig. 1.1. This field leverages fundamental concepts of applied statistics, emphasizing the use of computers to estimate complicated functions and with a decreased emphasis on proving confidence intervals around them [13]. This trend has accelerated with the rise of deep learning (DL), where enormous and heavily parametrized hierarchical models are used to deal with complex patterns from real-world data and do this with unprecedented accuracy. Interestingly, many DL architectures are designed to mimic some of the properties of the human thinking process, such as understanding correlations in visual patterns or recurrence in sound signals. We present a schematic representation of the relationship between these three fields (AI, ML, and DL) in Fig. 1.2.

To make a computer learn, we need three main ingredients:

1. A *task* to solve (Section 1.4).
2. *Data* that can be considered as an equivalent of *experience*. The latter can be provided in the form of, for example, an interacting environment, and allows for solving the task (Section 1.5).
3. A *model* that learns how to solve the task (Section 2.4).

To check whether a computer successfully learns how to solve a task, we need to define a performance measure that can be as simple as the comparison between the prediction of the model and the expected answer. In these terms, the learning process can be described as the iterative minimization of the model error or maximization of the model performance on the given task and data.

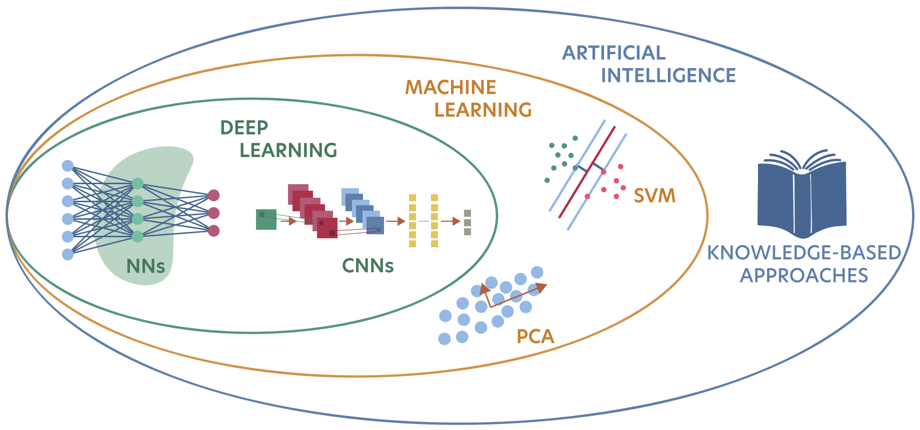

Figure 1.2 Sketch of the relation between AI, ML, and DL with examples from each field, including support vector machines (SVMs), principal component analysis (PCA), neural networks (NNs), and convolutional neural networks (CNNs).

1.2 Historical view on learning machines

The foundations of the theory of learning were established already in the 1940s. Its development has followed two parallel paths: a knowledge-based approach, which dominated the AI research field for decades, and a data-based one, which is currently on the rise. Throughout the years, ML has gone under various names (such as cybernetics or connectionism) and experienced a few cycles of intense popularity,[2] followed by criticism and disappointment, followed by funding cuts, followed by renewed interest years or decades later [13]. To give the reader some insights into the giants on whose shoulders we stand, we briefly present milestones in the development of ML following [13, 15–17]:

- 1943 – Walter Pitts and Warren McCulloch create a computer model inspired by the neural networks (NNs) of the human brain called the *threshold logic*. Their field of expertise is called *cybernetics*.
- 1949 – Donald Hebb hypothesizes how learning in biological systems works and formulates *Hebbian learning*. For example, if certain neurons "fire together, they wire together."
- 1957 – Frank Rosenblatt introduces a *Rosenblatt perceptron* modeling a single neuron. A perceptron is also called "an artificial neuron" and, after modifications in 1969 by Marvin Minsky and Seymour Papert, to this day, remains widely used as a building block of artificial neural networks (ANNs).
- 1962 – David Hubel and Torsten Wiesel present, for the first time, the response properties of single biological neurons recorded with a microelectrode.

[2]Some argue that the "AI winter" is upon us unless we rethink AI or combine it with knowledge-based approaches [14]. It is also important to remember that such hype cycles are frequent with emerging new technologies.

- 1969 – Marvin Minsky and Seymour Papert point out the computational limitations and disadvantages of linear models, including a single artificial neuron, contributing to the first "AI winter."
- 1986 – David Rumelhart, Geoffrey Hinton, and Ronald Williams use *backpropagation* to train an NN with one or two hidden layers, which, next to the revival of Hebb's ideas, causes renewed interest in the field that at this time is called *connectionism*. In the same year, David Rummelhart *et al.* publish a widely discussed two-volume book *Parallel Distributed Processing*, which collected original contributions from the field, including backpropagation and Boltzmann machines.
- The mid-1990s – Second AI winter whose appearance is ascribed [13] to exceedingly ambitious claims of the community, which led to the disappointment of investors, and the simultaneous progress of kernel methods, which require less computational resources.

Interestingly, we can see how closely the development of AI was intertwined with neuroscience. This makes sense, as the human brain provides proof by example that intelligent behavior is possible. A natural approach to AI would be to try to reverse engineer the brain to reproduce its functionality. However, while the perceptron was inspired by biological neurons and some ML models are loosely inspired by neurological discoveries, there is nowadays a consensus that models should not be designed to be realistic simulators of biological functions [13].[3] Instead, scientists attempt to solve the mysteries of the human brain using ML.

Since 2006, DL has been thriving again thanks to a breakthrough in the efficient training of deep NNs [18] via backpropagation, followed by multiple analyses confirming the importance of its depth. At the same time, there has been a rapid improvement in computational power in recent decades, which has allowed the exploration of larger ML models. Here, the development of graphics processing units (GPUs) [19, 20] has played a particularly important role: highly parallelizable algorithms, such as NNs, which are based on matrix and vector operations, can profit immensely from the parallel architecture of GPUs, enabling them to process large amounts of data more efficiently than central processing units (CPUs). Furthermore, we have started to produce and store large amounts of easily accessible electronic data throughout the world [21–23], enabling data-driven programming approaches. Since then, progress in the field has enabled realizations of concepts known, so far, only in science-fiction literature, such as self-driving cars or robots mimicking human emotions on their artificial faces (even if we are still far from human-like intelligence [24]). DL has dominated the field of computer vision for years and has found great success in time series analysis, with applications such as stock market and weather forecasting [25]. Another fruitful direction is natural language processing, where sequence-to-sequence models have achieved great feats, even combining text with images [26, 27]. The DL-based algorithms obtained superhuman performance in video games [28, 29] and complex board games, such as Go [30].

[3] Interestingly, we know that actual biological neurons compute very different functions than the perceptrons constituting our modern NNs, but greater realism has not yet led to any improvement in model performance [13].

Overall, the continuous progress in the field of ML is supported by the steady increase in computational power and its easy applicability to real-world problems. The increasing amount of data produced by our society and the monetary benefit of its processing have made the largest technological companies focus enormous economic efforts on the development of ML models. It is, hence, not a coincidence that the most important research groups in the field are associated with such companies. Importantly, one should understand the extent to which the trends of the field are dictated by the thirst for scientific discovery or by the particular needs of one or another technological giant. In summary, ML has become a day-to-day tool, acting in the shades of multiple technological tools we use today [24], with the potential to solve some of the most important problems of the modern world and thus contribute to improving the quality of life of people around the world.

1.3 Learning machines viewed by a statistical physics

It is also worth noting that the above-sketched developments of AI, data science, cognitive science, and neuroscience, related to ML and NN, were also intertwined with the development of the statistical physics of spin glasses and NN. A wonderful retrospective of these developments can be found in the lecture of the late Naftali Tishby, "Statistical physics and ML: A 30-year perspective." Therefore, here we present a similar list of historical milestones as in Section 1.2 but focused on statistical physics achievements:

- 1975 – Philipp W. Anderson and Samuel F. Edwards formulate the Edwards–Anderson spin glass model with short-range random interactions between Ising spins.
- 1975 – A little later, David Sherrington and Scott Kirkpatrick formulate the Sherrington–Kirkpatrick spin-glass model with infinite-range interactions, for which the mean-field solution should be exact. They propose to solve it using the replica trick, but this approximate solution turns out to be clearly incorrect at low temperatures.
- 1979 – Giorgio Parisi proposes an ingenious replica symmetry-breaking solution of the Sherrington–Kirkpatrick model.
- 1982 – John J. Hopfield publishes his seminal paper on attractor NNs, where, by assuming the symmetry of interneuron coupling, he relates the model to a disordered Ising model of N spins, very much analogous to spin glasses. The maximal storage capacity is found to be $0.14\,N$.
- 1985 – Daniel Amit, Hannoch Gutfreund, and Haim Sompolinski formulate the statistical physics of the Hopfield model and relate limited storage capacity to the spin-glass transition.
- 1987 – Marc Mezard, Giorgio Parisi, and Miguel Angel Virasoro publish the book *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*. Interestingly, it is one of the first works bringing together statistical physics and NNs but also putting them in a more general context of complex systems like optimization and protein folding.

- 1988 – Elisabeth Gardner formulates the so-called Gardner's program to ML, where learning abilities are related to the relative volume in the space of those NNs that realize learning tasks and teacher–student scenarios (see Section 8.1.1).
- 1989 – Daniel Amit publishes the book *Modeling Brain Function: The World of Attractor NNs* where he brings closer neurophysiology and artificial NNs by introducing dynamical patterns whose temporal sequence encodes the information.
- 1990 – Géza Györgyi shows that sharp phase transitions from bad to good generalization can occur in learning using Gardner's program on the perceptron.
- 1995 – David Saad, Sara Solla, Michael Biehl, and Holm Schwarze adapt Gardner's idea to study the dynamics of gradient descent in perceptrons and simple two-layer NNs called committee machines.
- Late 2010s – The statistical mechanics predictions for the perceptron and the committee machine start being made mathematically rigorous by Nicolas Macris, Jean Barbier, Lenka Zdeborová, and Florent Krzakala.
- 2010s–today – With the explosion of DLs, interest in the statistical mechanics approach to learning is rekindled. Analyses are developed for increasingly complex models beginning to bridge the gap from perceptrons to deep NNs.
- 2021 – Giorgio Parisi receives the Nobel Prize in Physics "for the discovery of the interplay of disorder and fluctuations in physical systems from atomic to planetary scales."
- 2024 – John J. Hopfield and Geoffrey Hinton receive the Nobel Prize in Physics "for foundational discoveries and inventions that enable machine learning with artificial neural networks."

We discuss the intersection of statistical physics and ML in more detail in Section 8.1.

1.4 Examples of tasks

As stated in Section 1.1, the first ingredient needed for a computer to learn is the notion of a *learning task*. The archetypical ML task is the study of a response variable, $y(x)$, influenced by an explanatory variable x. In principle, there is no restriction on whether y or x or both are continuous, discrete, or even categorical.[4] Throughout the book, we restrict both variables, possibly encoded accordingly, to be of quantitative nature. That is, we can treat variables straightforwardly from a numerical perspective and easily adjust them to fit our needs.

[4]When the inputs are, for example, words in a sentence as they are in the field of natural language processing, we can still process them by representing words by a suitable encoding, which can be either continuous or discrete.

Regression. We start by considering *regression* tasks. In this setting, we typically assume an immediate relationship between the two variables x and y, which is often deterministic. More precisely, we seek to express the variable $\boldsymbol{y}$, also known as (a.k.a.) the output or target, in terms of the variable $\boldsymbol{x}$, a.k.a. the input. In general, both variables can be multidimensional, as indicated by our notation. The objective of regression is to find the function f that yields the mapping $\boldsymbol{y} = f(\boldsymbol{x})$ for all possible tuples of $(\boldsymbol{x}, \boldsymbol{y})$. Of course, from a practical point of view, we can neither optimize over the set of all possible functions nor over the entire domain of $\boldsymbol{x}$. Instead, we resort to a finite dataset for which we opt to find a model that maps every input $\boldsymbol{x}$ to its corresponding target $\boldsymbol{y}$. Usually, the model is predefined up to some parameters,[5] which are tuned to fit the dataset. The most simple model assumes a linear relationship between the input and the output. We give more details of this model archetype in Section 2.4.1. From here, there is a multitude of ways to extend the model by incorporating nonlinear dependencies on both the model parameters and the input $\boldsymbol{x}$. We find interesting regression problems in a large range of study fields, such as sociology (e.g., annual salary as a function of years of work experience), psychology (e.g., perceived happiness relative to wealth), finance (e.g., housing market prices depending on socioeconomic factors), and, of course (quantum) physics and chemistry. We cover some examples in this book, for instance, the prediction of potential energy surfaces (PESs) in quantum chemistry in Section 4.5, or the estimation of Hamiltonian's parameters given the measurement data in Section 7.3.

Classification. Another large class of tasks is *classification*. In this case, our goal is to use an algorithm to assign *discrete* class labels to examples. In contrast to regression, we are optimizing a model to find a mapping from an input vector $\boldsymbol{x}$ to a target $\boldsymbol{y}$, which encodes a representation of the different possible classes. The simplest example of this kind of task is binary classification, in which an algorithm has to distinguish between two classes, for example, true or false. When the task involves more than two classes, we speak of multi-class classification. A canonical example for such a task is the classification of the images of handwritten digits contained in the famous MNIST [31] dataset (named after the Modified National Institute of Standards and Technology) over ten classes, one for each number from zero to nine. Other famous ML classification datasets are Iris [32], CIFAR-10 and 100 [33], and ImageNet [34].[6] A popular example from physics is the classification of different classical and quantum phases of matter, described in Chapter 3. Another set of examples is provided by the classification subroutines in the automation of (quantum) experiments highlighted in Section 7.3.

Both regression and classification tasks require a training dataset consisting of examples of inputs $\boldsymbol{x}$ together with their corresponding labels $\boldsymbol{y}$. Nonetheless, there are also tasks that do not require explicit labels. An example of such is *density estimation*, where the aim is to infer the probability density function of the dataset. This is

[5]There are also nonparametric approaches, for example, see Sections 4.4.2 and 7.2.2.

[6]The Iris database contains 150 data points with four features of three species of iris. The CIFAR-10 dataset consists of 60 000 32×32 color images in 10 classes and was named after the Canadian Institute for Advanced Research. Finally, the ImageNet is a gigantic project with over 10 million labeled images whose most popular subset spans 1 000 object classes.

directly related to the field of *generative problems*, where the goal is to *generate* new data instances that resemble some given input data. The distinction between the two fields is that the latter does not require explicit knowledge or reconstruction of the underlying data distribution to sample new instances. We present more details on density estimation in Section 7.2.

In all the previous cases, we try to infer properties of a given predefined dataset. However, there are other tasks that involve starting from scratch and building a dataset on the fly, from which we can then learn. A paradigmatic example of such a task is learning how to play a game. In this case, we start tabula rasa and progressively build a dataset with the experience gathered as we play the game. From these data (or during their retrieval), our goal is to learn a function that chooses the best possible action or move according to the current state of the game. In this example, we can periodically alternate between collecting experience and learning, or we can do both at the same time.

This list of tasks is, of course, not exhaustive. Other examples that do not directly fall into the previous categories include text translation, imputation of missing values, anomaly detection, and data denoising, to name a few.

1.5 Types of learning

The second learning ingredient is *data*, whose accessibility also often determines the type of learning we have to consider. It is clear, of course, that the notions of task, as presented in Section 1.4, and data are intertwined: Certain tasks can only be solved if sufficient data are available and, in turn, a richer dataset allows us to transfer from one task to another with seemingly low effort. Although the term data is often used for a variety of concepts across many fields, there is a precise definition of it in the ML community. We usually refer to *data* in terms of a dataset $\mathcal{D}$, containing a finite amount of data instances often called *data points* $\boldsymbol{x}_i$, which may be presented as is, that is, $\mathcal{D} = \{\boldsymbol{x}_i\}$ or may be accompanied by predefined labels or targets $\boldsymbol{y}_i$, that is, $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$. To shorten the notation, we also represent the input data points $\{\boldsymbol{x}_i\}$ by a matrix $\boldsymbol{X}$, which can either be stacked row- or column-wise.

Although the notation is clear, there is much less convention and an even lesser understanding of how the data should be *represented*. This is because, on the one hand, the data can be arbitrarily preprocessed (e.g., the data mean is often subtracted prior to any further analysis), which already provides some degree of freedom. On the other hand, even choosing the right descriptors to characterize our object of interest is challenging: Too few might not capture all relevant aspects of the object, whereas too many can lead to spurious correlations that can interfere with the conclusions that we want to draw from data. We refer to each element at each data point $\boldsymbol{x}_i$ as a *feature*. As stated in Section 1.1, a central problem in ML relates to the correct representation of the data and its features. This is the core of the field of representation learning, which we only touch upon, for example, by means of principal component analysis (PCA) and autoencoders (AEs) in Chapter 3 and Section 7.2, respectively.

Finally, we emphasize that data can, loosely speaking, be identified with experience: Data can be produced as a result of a repeated interaction with an entity (such as an experiment or a simulation) that then leaves us with a certain amount of experience about its underlying mechanism. In some cases, this experience may be used to further interact with such an entity and learn from it. To this end, we set up a model. In summary, the type of data to which we have access effectively defines the types of learning with which our model can be faced. These are usually divided into three: supervised, unsupervised, and reinforcement learning (RL).

Supervised learning. *Supervised learning* can be seen as a generalized notion of regression and classification, introduced in Section 1.4, and describes ML algorithms that learn from *labeled* data, that is, $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$. There exist various approaches to supervised learning, ranging from statistical methods to classical ML and DL, both introduced in Section 2.4. The concept of supervised learning appears repeatedly in this book and forms the basis of many chapters, including phase classification (Chapter 3), Gaussian processes (Chapter 4), as well as the selected topics of DL for quantum sciences (Chapter 8). Importantly, some of the latter are specially suited to deal with experimental data, as, for instance, in the efficient readout of quantum dots or the identification of Hamiltonian parameters describing quantum experimental setups. In most of these examples (but there are notable exceptions), large amounts of data are required for the training process. On top of the data, as stated above, supervised learning requires correctly labeled data. This is usually considered one of its most prominent downsides, as perfectly matching labels are not always accessible or have to be added manually by humans.

Unsupervised learning. Supervised learning is not always the best option: the scarcity of labeled data is an example in which a classical input–output design might fail. Instead, we often have access to data where no prior information, for example, in terms of labels, is given (i.e., $\mathcal{D} = \{\boldsymbol{x}_i\}$). In this case, we can employ *unsupervised learning*. Unsupervised learning can either be used for preliminary preprocessing steps, such as dimensionality reduction, or for representation learning, such as in clustering. In contrast, dimensionality can also be increased by adding features via generative models. In this book, we discuss the application of unsupervised learning for phase classification in Chapter 3 and density estimation in Section 7.2. This example is particularly interesting because it demonstrates how the choice of unsupervised learning over supervised learning can aid in the automated discovery of new physics when the interpretation of a process, for example, the nature of two different phases in a transition is unknown.

Reinforcement learning. In contrast to the two previous types of learning, in RL, we usually do not have a dataset available *at all*. Instead, we have an *environment* with which we have to interact to achieve a certain task. This interaction is augmented with feedback, that is, some extra information on whether the action has been beneficial or harmful in achieving the task at hand. The collection of visited environment states, actions taken, and rewards or penalizations received take the role of a dataset. Feedback is very important in RL because we do not have a clear-cut route in achieving our task. In fact, initially, we typically do not even know the necessary ingredients for achieving the task. Often, we only know that we *achieved* a specific goal but not *why* we did it. The field of RL is precisely concerned with tackling the issue of how. To introduce it properly, we devote Chapter 6 to it.

Other types of learning. While supervised, unsupervised, and RL are the most common learning schemes in the ML applications in quantum sciences, there are ML approaches that go beyond this classification. An interesting example is active learning. This field includes selection strategies that allow for an iterative construction of a model's training set in interaction with a human expert or environment. The aim of active learning is to select the most informative examples and minimize the cost of labeling [35, 36]. We only touch on this topic by means of BO[7] in Section 4.3 and local ensembles (LEs) in Section 3.5.3. Another example of learning is semi-supervised learning in which a large amount of unlabeled data is explored to get better feature representations and improve the models trained on the small number of labeled data [37].

1.6 How to read this book

This book aims at providing an educational and self-contained overview of modern applications of ML in quantum sciences. As such, Chapter 2 is devoted to the ML prerequisites that are necessary to fully enjoy all the more advanced and further contents of this book. We discuss in detail four main ML paradigms that have been successfully explored in quantum physics and chemistry: In Chapter 3, we describe how supervised and unsupervised learning can be utilized to classify phases of matter. In Chapter 4, we introduce kernel methods with a special focus on Gaussian processes (GPs) and BO. Chapter 5 presents an overview of various representations of quantum states based on NNs. Finally, in Chapter 6, we dive into the foundations of RL and how it can be applied to quantum experiments.

In addition to these four pillars of ML in quantum sciences, there exists an exciting two-way interplay between the natural sciences and AI. Chapter 7 focuses on more specialized examples of how ML-related methods revolutionize quantum science. In particular, we introduce the paradigm of differentiable programming (∂P) and describe how it is becoming an important numerical research tool. Moreover, we discuss how ML methods assist researchers in tasks related to density estimation, as well as optimizations and speedup of scientific experiments. There exists a vibrant reverse influence on ML coming from statistical physics (which we discuss in Section 8.1) and finally quantum computing. We describe the promises of quantum machine learning (QML) in Section 8.2. All in all, this book discusses the fruitful interplay of AI and physical sciences. Its content with references to relevant sections is illustrated in Fig. 1.3.

We encourage the reader to start with Chapters 1 and 2, that is, "Introduction" and "Basics of Machine Learning." Then, the reader is free to wander into any of the independent Chapters 3–6 covering the four main paradigms, Section 7.1 on

[7] Bayesian optimization (BO) and active learning, while similar, are not the same. Active learning aims to determine optimal sampling, while BO aims to find an extremum of a black-box function with as few function evaluations as possible.

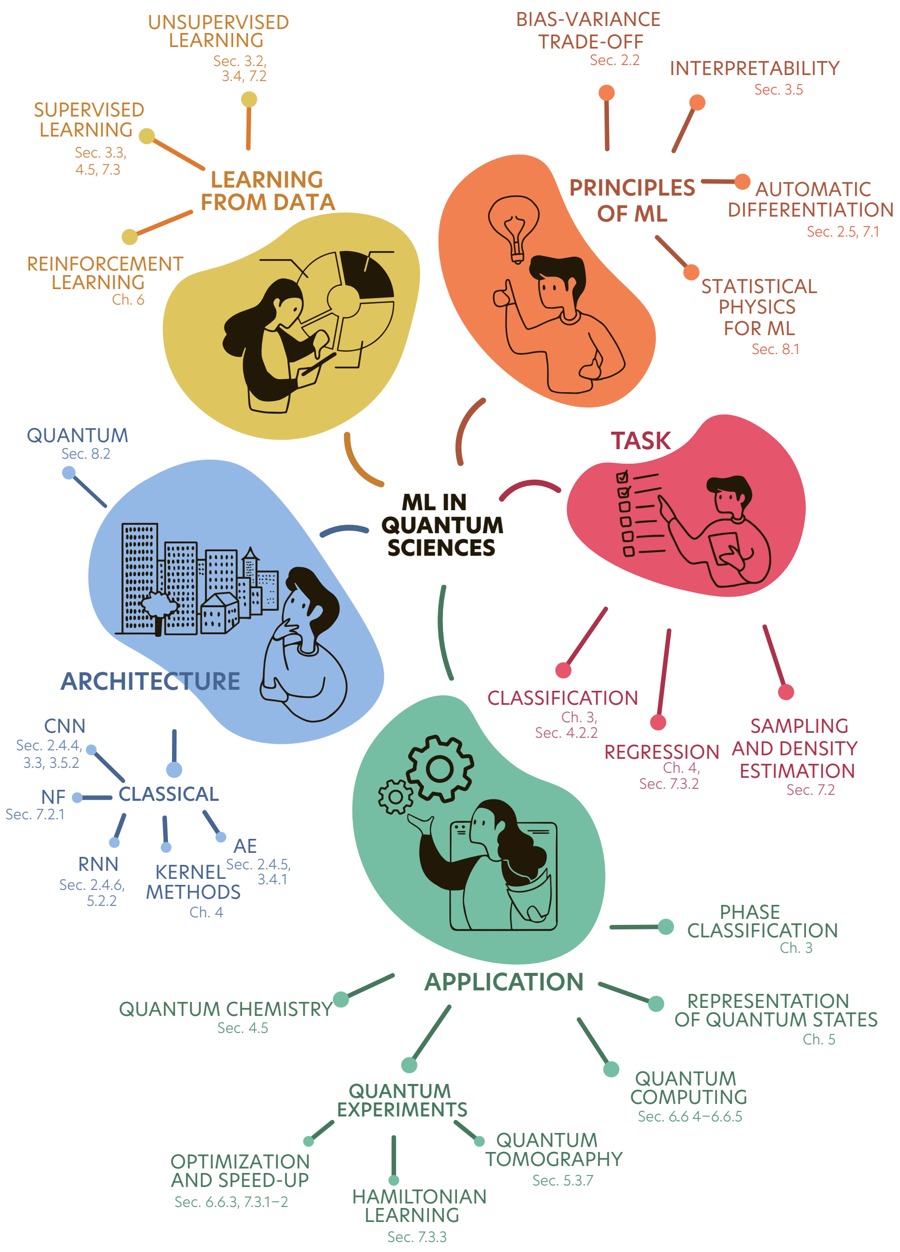
ML IN QUANTUM SCIENCES
LEARNING FROM DATA
UNSUPERVISED LEARNING Sec. 3.2, 3.4, 7.2
SUPERVISED LEARNING Sec. 3.3, 4.5, 7.3
REINFORCEMENT LEARNING Ch. 6
PRINCIPLES OF ML
BIAS-VARIANCE TRADE-OFF Sec. 2.2
INTERPRETABILITY Sec. 3.5
AUTOMATIC DIFFERENTIATION Sec. 2.5, 7.1
STATISTICAL PHYSICS FOR ML Sec. 8.1
TASK
CLASSIFICATION Ch. 3, Sec. 4.2.2
REGRESSION Ch. 4, Sec. 7.3.2
SAMPLING AND DENSITY ESTIMATION Sec. 7.2
ARCHITECTURE
QUANTUM Sec. 8.2
CLASSICAL
CNN Sec. 2.4.4, 3.3, 3.5.2
NF Sec. 7.2.1
RNN Sec. 2.4.6, 5.2.2
KERNEL METHODS Ch. 4
AE Sec. 2.4.5, 3.4.1
APPLICATION
PHASE CLASSIFICATION Ch. 3
REPRESENTATION OF QUANTUM STATES Ch. 5
QUANTUM COMPUTING Sec. 6.6 4–6.6.5
QUANTUM CHEMISTRY Sec. 4.5
QUANTUM EXPERIMENTS
OPTIMIZATION AND SPEED-UP Sec. 6.6.3, 7.3.1–2
HAMILTONIAN LEARNING Sec. 7.3.3
QUANTUM TOMOGRAPHY Sec. 5.3.7

Figure 1.3 Content of this book. We cover three main learning schemes: supervised, unsupervised, and reinforcement learning (RL), examples of ML tasks like classification, regression, and density estimation, various applications in quantum sciences, quantum and classical ML architectures. We also dive into the principles of ML.

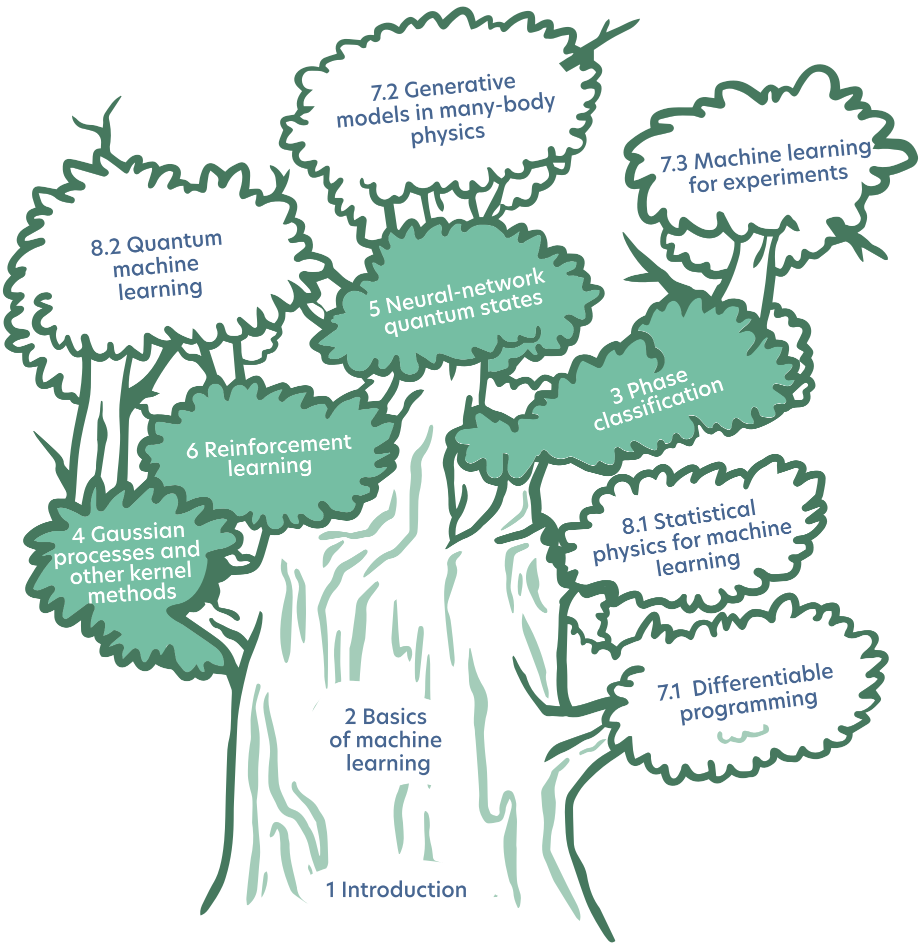
7.2 Generative models in many-body physics
7.3 Machine learning for experiments
8.2 Quantum machine learning
5 Neural-network quantum states
3 Phase classification
6 Reinforcement learning
4 Gaussian processes and other kernel methods
8.1 Statistical physics for machine learning
7.1 Differentiable programming
2 Basics of machine learning
1 Introduction

Figure 1.4 Dependencies between chapters.

differentiable programming or Section 8.1 on how statistical physics tackles the puzzles of ML. Section 7.2 on generative models builds on Chapter 5 on neural quantum states (NQSs), while Section 7.3 on ML for experiments requires knowledge of the methods used for phase classification presented in Chapter 3. Finally, Section 8.2 on QML utilizes concepts introduced in Chapters 4–6. The dependencies between chapters are visualized as a tree in Fig. 1.4.

Further reading

- Carleo, G. *et al.* (2019). *Machine learning and the physical sciences*. Rev. Mod. Phys. 91, 045002. This detailed review summarizes the development of ML in physics and its achievements till 2019 [5].

- Carrasquilla, J. (2020). *Machine learning for quantum matter*. Adv. Phys. X 5, 1. A concise review focusing on phase classification and quantum state representation [6].
- Krenn, M. *et al.* (2023). *Artificial intelligence and machine learning for quantum technologies*. Phys. Rev. A 107(1), 010101. A perspective focusing on how quantum computing, quantum communication, and quantum simulation benefit from the ML revolution [38].
- Chollet, F. (2019). *On the measure of intelligence*. The review of different measures used to quantify *intelligence*, which provides a perspective on AI development [39].
- Krenn, M. *et al.* (2022) *On scientific understanding with artificial intelligence*. Nat. Rev. Phys. 4, 761–769 [40]. A beautiful paper discussing ways in which AI could contribute to scientific discovery. It touches upon the philosophy of understanding and draws conclusions from dozens of anecdotes from scientists on their computer-guided discoveries.
- Recordings of lectures of the Summer School: Machine Learning in Quantum Physics and Chemistry which took place between August 23 and September 03, 2021, in Warsaw, Poland.
- Jupyter notebooks prepared as tutorials for the Summer School: Machine Learning in Quantum Physics and Chemistry [2].