A Leap among Quantum Computing and Quantum Neural Networks: A Survey

FABIO VALERIO MASSOLI, LUCIA VADICAMO, GIUSEPPE AMATO, and FABRIZIO FALCHI, Istituto di Scienza e Tecnologie dell'Informazione "Alessandro Faedo", CNR, Italy

In recent years, Quantum Computing witnessed massive improvements in terms of available resources and algorithms development. The ability to harness quantum phenomena to solve computational problems is a long-standing dream that has drawn the scientific community's interest since the late '80s. In such a context, we propose our contribution. First, we introduce basic concepts related to quantum computations, and then we explain the core functionalities of technologies that implement the Gate Model and Adiabatic Quantum Computing paradigms. Finally, we gather, compare, and analyze the current state-of-the-art concerning Quantum Perceptrons and Quantum Neural Networks implementations.

CCS Concepts: • **Computer systems organization** → **Quantum computing**; **Neural networks**; • **Computing methodologies** → **Neural networks**; **Machine learning**; • **Applied computing** → *Physics*; • **Hardware** → Quantum technologies;

Additional Key Words and Phrases: Quantum computing, quantum machine learning, quantum neural network, quantum deep learning

ACM Reference format:
Fabio Valerio Massoli, Lucia Vadicamo, Giuseppe Amato, and Fabrizio Falchi. 2022. A Leap among Quantum Computing and Quantum Neural Networks: A Survey. *ACM Comput. Surv.* 55, 5, Article 98 (December 2022), 37 pages.
https://doi.org/10.1145/3529756

1 INTRODUCTION

Artificial Intelligence (AI) has intrigued and puzzled many generations of scientists and largely fueled novelists' imaginations. The modern definition of AI—as the ensemble of computer systems empowered with the ability to learn from data through statistical techniques—dates back to 1959. **Machine Learning (ML)**, a subclass of AI, is a discipline that aims to study algorithms that can learn from data to perform tasks without following explicit instructions. Often, these algorithms are based on a computational model that belongs to differentiable programming techniques, called **Neural Networks (NNs)**. The success of such algorithms resides in their ability to learn to achieve

The work was partially supported by H2020 project AI4EU under GA 825619, by H2020 project AI4Media under GA 951911, by WAC@Lucca funded by Fondazione Cassa di Risparmio di Lucca.
Authors' address: F. V. Massoli (corresponding author), L. Vadicamo, G. Amato, and F. Falchi, Istituto di Scienza e Tecnologie dell'Informazione "Alessandro Faedo," CNR, Via G. Moruzzi 1, Pisa, Italy, 56124; emails: {fabio.massoli, lucia.vadicamo, giuseppe.amato, fabrizio.falchi}@isti.cnr.it.

0360-0300/2022/12-ART98 $15.00
https://doi.org/10.1145/3529756

a specific goal [93, 116], i.e., they learn to discover hidden patterns and relations among data to fulfill the task at hand [87, 115]. Mathematically, NNs are made of a sequence of transformations, called layers, composed of linear operators and elementwise nonlinearities. Then, the goal of learning is to modify the transformations' parameters to fulfill a task successfully. Whenever a model accounts for more than a couple of such layers, it is called a **Deep Learning (DL)** model or a **Deep Neural Network (DNN)**. Thanks to their representation power and the development of new technologies and training algorithms, DL models obtained astonishing results in the past two decades, achieving superhuman performance on certain tasks [174]. However, higher performances require more complex models and larger datasets to train them, thus constantly increasing the hunger for resources and power to learn to solve a given problem.

In this regard, quantum computers might offer new solutions that exploit quantum phenomena such as interference, superposition, and entanglement. Such a characteristic is expected to speed up the computational time and to reduce the requirements for extensive resources, yielding the concepts of *quantum advantage* and *quantum supremacy* [134, 145]. The mentioned quantum phenomena are described within the framework of quantum mechanics [61, 91, 159], a "young" physics theory formalized at the beginning of the 20th century. Such a theory unveils the intrinsic statistical characteristic of nature, a behavior that unfortunately is hidden from us, at least in the macroscopic world.

The quest for a quantum computer started with the ideas of Benoff [23] and Feynmann [73, 121], in the 1980s, pointing out that quantum computers should be more efficient in solving specific problems. For example, quantum devices might help in studying very complex, e.g., entangled, systems by emulating or simulating [66, 74] their behavior in chips that are quantum by their nature [150, 193].

From a computer science point of view, a quantum computer represents a computational model based on the principles and postulates of quantum mechanics. Such techniques aim at embracing the power of quantum theory into a device that can be conveniently "programmed" to fulfill a given task. Moreover, the result of the computation itself might represent a quantum object encoding different answers, each solving a specific problem [59]. It is only recently that researchers have succesfully realized a quantum processor capable of performing controllable computations based on quantum phenomena. Several industrial applications already benefited from such a technology, such as: chemistry [104], optimization problems [132], finance [67], quantum sensing [56], and quantum imaging [48], among others. Besides the mentioned applications, quantum computers might offer advantages in terms of energy management compared to classical ones. Indeed, quantum computers are expected to be more energy-efficient than supercomputers considering Reference [10]. For that reason, hybrid approaches might offer exciting solutions to lower the energy consumption for a given computation by moving the high-energy portion of the computation on the **Quantum Processing Unit (QPU)** while leaving the low-energy one to the cloud [182]. A fundamental result of quantum information theory is the observation that, although quantum phenomena allow solving some classical problems more efficiently than classical computations [27], quantum computers cannot compute any function that is not Turing-computable [58].

Concerning the context of **Quantum Machine Learning (QML)** [30], a relevant contribution comes from the ability of quantum devices to speed up linear algebra-related tasks. For example, it has been shown that using the **Harrow-Hassidim-Lloyd (HHL)** quantum algorithm [85, 198] to sample solutions for a system of linear equations offers an exponential speedup over its classical counterpart. Furthermore, Shor's algorithm [171, 172] for integer factorization and Grover's algorithm [80] for searching unstructured databases are additional examples of procedures that benefit from a quantum formulation. Although general QML is of great interest, it is not the central topic of our work, as it has already been covered extensively in the literature. Therefore, we

focus on the QML sub-area concerning recent **Quantum Neural Networks (QNNs)** approaches. Despite the fact that their name recalls the neural network structure mentioned earlier, they are characterized by a completely different design.

Concerning quantum computations, the most commonly adopted paradigms are the **Gate Model (GM)** [134] and the **Adiabatic Quantum Computation (AQC)** [11]. In spite of being equivalent up to a polynomial overhead, they represent two profoundly different computation approaches. The first one is based on the concept of "gate": a unitary transformation applied to one or more quantum bits (*qubits*), i.e., the basic units of quantum information (see Section 3). Instead, in the AQC, one typically encodes the desired objective function into a quantum system and then lets it evolve. However, in both paradigms, the QPU state at the end of the evolution embodies the answer to the given problem. Thus, we can highlight the main differences between the two approaches as follows: The GM allows users to control the transformation to apply on the qubits directly and is discrete in time, while AQC does not allow to control single qubits directly and does not discretize the time property of the system.

Unfortunately, quantum technologies are still at their dawn, having minimal computational capacity. Furthermore, significant technological challenges arise from the requirement for quantum systems to be isolated from the environment to avoid decoherence, which causes lack of information stored in the quantum device. Therefore, researchers typically rely on quantum simulators to test their ideas while waiting for the next generation of quantum computers. Whether quantum supremacy is real or not is still an open question. For example, we do not expect quantum computers to solve efficiently worst-case NP-hard problems like combinatorial ones. Instead, we do expect that we will be able to find a better approximation to the solution or find it faster than a classical computer [109, 146].

Stemming from those considerations, we conceived this survey to offer both the neophyte and the more experienced reader insights into several fundamental topics in the quantum computation field. Moreover, noticing that the literature lacks a detailed discussion about the latest achievements concerning **Quantum Perceptrons (QPs)** and QNNs, we gather, analyze and discuss state-of-the-art approaches related to those topics. Notably, we can summarize our work as follows:

- We review the current state-of-the-art regarding QPs and QNNs by discerning among theoretical formulations, simulations, and implementations on real quantum devices;
- We report the main achievements by different research groups concerning the topic of quantum supremacy;
- We provide a gentle introduction to several basic notions about quantum mechanics, quantum information, and quantum computational models;
- We collect and organize the most relevant papers to this survey on a GitHub[1] page allowing interested readers to easily and quickly browse through the literature.

Moreover, to ease the understanding of the article's content, we summarize basic notions about the Dirac notation, postulates of quantum mechanics, the physical realization of qubits, the Bloch Sphere representation, the Variational Principle, and the Adiabatic Theorem, in the electronic appendix available in the ACM Digital Library. We suggest the reader who is unfamiliar with these topics to peruse the Appendix B before reading any further.

Concerning the remaining part of the article, it is organized as follows: In Section 2, we report on other surveys on the topic at hand. In Section 3, we introduce the fundamental concept of a qubit, and we give the reader a brief overview of the currently most widespread quantum computational models. Then, in Section 4, we tackle the concept of quantum speedup. Subsequently, in Section 5,

[1] https://github.com/fvmassoli/survey-quantum-computation.

Table 1. Summary of Notation Used throughout This Article

Symbol	Definition
$\mathrm{Re}(z)$, $\mathrm{Im}(z)$	Real and imaginary parts of the complex number z
z^*	Complex conjugate of the complex number z
A^*	Complex conjugate of the matrix A
A^T	Transpose of the matrix A
$A^\dagger$	Complex conjugate transpose of the matrix A, i.e., $A^\dagger = (A^*)^T$
$A \otimes B$	Kronecker product of the matrix A with the matrix B
$\hat{A}$	Linear operator whose matrix representation is A
I, $\hat{I}$	Identity matrix and Identity operator
$\hat{H}$	Hamiltonian
ξ_i	Eigenvalues of the Hamiltonian
$\dot{f}$	Newton's notation for the derivative of a function $f(t)$ with respect to t
$\lvert\psi\rangle$	Vector of a Hilbert space, also called *ket*
$\langle\psi\rvert$	Vector dual to $\lvert\psi\rangle$, also called *bra*
$\langle\psi\vert\phi\rangle$	Inner product between $\lvert\psi\rangle$ and $\lvert\phi\rangle$ whose results in scalar
$\lvert\psi\rangle\langle\phi\rvert$	Outer product between $\langle\psi\rvert$ and $\lvert\phi\rangle$ whose results in a matrix
$\lVert\lvert\phi\rangle\rVert = \sqrt{\langle\phi\vert\phi\rangle}$	Norm of $\lvert\phi\rangle$
δ_{ij}	Kronecker delta that is defined to be equal to 1 whenever $i = j$, and 0 otherwise
$\lvert\psi\rangle\otimes\lvert\phi\rangle$, $\lvert\psi\rangle\lvert\phi\rangle$, $\lvert\psi, \phi\rangle$, $\lvert\psi\phi\rangle$	Tensor product between $\lvert\psi\rangle$ and $\lvert\phi\rangle$ and its abbreviated notations
$\lvert\psi\rangle^{\otimes n}$	Tensor product between n identical states $\lvert\psi\rangle$, i.e. $\lvert\psi\rangle^{\otimes n} = \lvert\psi\rangle\otimes\cdots\otimes\lvert\psi\rangle$
$\langle\psi\rvert\hat{A}\lvert\phi\rangle$	Inner product between $\lvert\psi\rangle$ and $\hat{A}\lvert\phi\rangle$ where $\hat{A}$ is a linear operator
$\langle\hat{A}\rangle$	Average value of an operator w.r.t. to a given state $\lvert\psi\rangle$: $\langle\psi\rvert\hat{A}\lvert\psi\rangle$
$\{\lvert0\rangle, \lvert1\rangle\}$ or $\{\lvert\uparrow\rangle, \lvert\downarrow\rangle\}$	Computational basis
$\lvert+\rangle$, $\lvert-\rangle$	Hadamard basis states, defined by $\lvert+\rangle = \frac{1}{\sqrt{2}}(\lvert0\rangle + \lvert1\rangle)$, and $\lvert-\rangle = \frac{1}{\sqrt{2}}(\lvert0\rangle - \lvert1\rangle)$
ρ	Density matrix describing a given system's state: $\rho = \sum_i \lambda_i \lvert\psi_i\rangle\langle\psi_i\rvert$
$\hat{\sigma}_x$, $\hat{\sigma}_y$, $\hat{\sigma}_z$	Pauli operators
H, —$\boxed{H}$—	Hadamard operator/gate
Tr	trace operator

we move to the core topic of this survey, i.e., QNN approaches. Finally, the conclusions are drawn in Section 6. In Table 1, we report a summary of the notations used throughout the manuscript, while a summary of the used abbreviations is available in Table 1 in the Appendix A.

2 OTHER SURVEYS

In the literature, many review papers have been devoted to the realm of quantum computation and information. Several surveys address quantum ML and its applications. However, very little has been said about QNNs, and reviews that include this topic often cover it marginally or lack a detailed discussion on the most recent achievements. For this reason, we devote our work mainly to reviewing Variational Learning approaches that encompass Quantum Approximate Optimization Algorithms, Variational Quantum Eigensolver, Quantum Perceptrons, and Quantum Neural Networks.

Gyongyosi and Imre [82] reviewed the technological status of quantum computation and information systems by discussing several papers that can be useful for researchers interested in delving deep into specific aspects of quantum computations. Savchuk and Fesenko [158] reported a brief analysis of some of the fundamental principles beneath quantum computing. McGeoch [130] presented a comparison between expected results based on a purely theoretical calculation about complexity theory and what is currently achievable and achieved on the modern physical realization of quantum processors based on quantum annealing. They also gave interesting

insights on basic notions concerning the quantum annealing process, which is at the heart of the D-Wave quantum platforms. Preskill [145, 146] debated the concept of "quantum supremacy" and the potential of quantum computing.

In the past two decades, there has been a strong interest and commitment in the scientific community to develop quantum algorithms to solve ML problems, giving life to the field of QML. Hitherto, there is not a universally accepted definition of QML yet. However, Aïmeur et al. [8] and Adcock et al. [5] provided useful categorizations based on the employed learning paradigms. For example, in Reference [5], the term QML is used to refer to two learning categories. The first one concerns learning processes performed using classical computations aided by sub-routines relying on quantum computations. The second one concerns approaches that exploit quantum computations only and for which there are no sub-routines that can be performed equally well on a classical computer. Notwithstanding, this categorization excludes from the QML class all the learning approaches performed on a classical machine but that use data coming from quantum devices.

A viable approach to QML stems from the quantum algorithms that speedup linear algebra-related tasks (see, e.g., References [30, 88, 164, 196]). In this context, several quantum extensions of classical algorithms have already been proposed, for example, for clustering [105, 137], Support Vector Machines [149], semidefinite programs [36], gradient descent for linear systems [107], PCA [122], and variational generation for adversarial learning [151], just to cite some. Biamonte et al. [30] and Schuld et al. [164] gave an overview of the achievements and challenges of quantum-enhanced machine learning algorithms, considering aspects of both gate and adiabatic computational models. Basic algorithms for QML, such as Grover's algorithm [80], quantum state"similarity" estimation, and the HLL algorithm [85], are reviewed in Reference [197]. The authors also illustrate how these algorithms (based on the quantum gate model) have been used in literature to speed up standard ML algorithms, such as Support vector machine, k-means clustering, PCA, and Linear discriminant analysis. Furthermore, in References [2, 3] the authors review quantum solutions for binary classification and k-Nearest Neighbor problems. Ciliberto et al. [50] covered quantum machine learning approaches with emphasis on theoretical aspects, such as computational complexity analysis, discussion on limitations of QML, and challenging aspects like learning under noise. Theoretical aspects of ML using quantum computers are also discussed in Reference [15]. Many other survey papers overview quantum algorithms for ML and their applications; see for example References [9, 29, 64, 65, 125, 148, 162] and references therein.

Benedetti et al. [22] reviewed parametrized quantum circuits, which are arguably the quantum analogous of NN models. The Variational Quantum Algorithms, which employ parameterized quantum circuits, are also discussed in Reference [44]. Kamruzzaman et al. [103] reviewed the status of (theoretical) Quantum Neural Networks and their limitations. Allcock and Zhang [13] focused on quantum generalizations and applications of some popular neural-network concepts, such as Boltzmann machine, generative adversarial networks, and autoencoders. Recently, Magini et al. [124] gave a concise overview of the main directions that have been taken to develop quantum artificial neural networks.

3 QUBIT AND QUANTUM COMPUTATIONAL MODELS

In this section, we first introduce the concept of *qubit* and then give a brief overview of the two main paradigms of quantum computation: the ***Gate Model* (GM)**, or Universal Quantum Computing, and the ***Adiabatic Quantum Computation* (AQC)**. A comprehensive analysis of the mentioned approaches is out of scope for our work. However, in what follows, we try to empower the reader with a basic grasp of those two types of computations.

3.1 Qubit

The *qubit*, short for "quantum bit," represents the fundamental unit of information for quantum devices. It is the quantum correspondent of the bit. However, apart from the last part of its name, the qubit does not share much with its classical cousin. Similar to the bit, which assumes two values only (0 or 1), the qubit can be "observed" (i.e., measured) in two possible states, typically indicated as $|0\rangle$ and $|1\rangle$. However, differently from the bit, a qubit can also be in a so-called *superposition* of states before the measurement. Intuitively, that means that the qubit can be in either the state $|0\rangle$ or $|1\rangle$ with a certain probability. Notwithstanding, when measured, the qubit state "collapses" into one of them. The qubit is a rather abstract concept, since such an object does not exist in the real world. Instead, we relate it with artificial atoms, i.e., physical systems able to emulate the behavior of an atom. Specifically, we are typically interested in emulating the behavior of two-state systems that satisfy a certain number of hard constraints such as small dissipation and isolation from the environment.

From a mathematical point of view, a *qubit state* $|\psi\rangle$ is a unit vector in $\mathbb{C}^2$. The symbol $|\cdot\rangle$ we just used is inherited from Dirac [60] formalism[2] employed in the quantum mechanics. In a nutshell, the object $|\psi\rangle$, called *ket*, represents a vector in a Hilbert space. Similarly, $\langle\psi|$, called *bra*, is defined as the adjoint (or dual) of such a ket. We remind the reader unfamiliar with the Dirac formalism that the symbol within a ket or bra, e.g., the "ψ" used above, can be an arbitrary label (e.g., a letter or a number). Note that often binary labels are used, especially to indicate the vectors of a space basis. For example, the basis $\{\left[\begin{smallmatrix}1\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]\}$ of $\mathbb{C}^2$, referred to as the *computational basis*, is typically indicated as $\{|0\rangle, |1\rangle\}$, or sometimes also as $\{|\uparrow\rangle, |\downarrow\rangle\}$. Using this formalism, a qubit state can be represented as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, where α and β, called *amplitudes*, are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$.[3] Therefore, a qubit state is a coherent superposition of the computational basis states $|0\rangle$ and $|1\rangle$. However, this does not mean that a qubit has a value between $|0\rangle$ and $|1\rangle$ but rather that it is not possible to say whether the qubit is definitively in the state $|0\rangle$, or definitively in the state $|1\rangle$. In fact, when we measure a qubit, we observe either $|0\rangle$ with probability $|\alpha|^2$, or $|1\rangle$ with probability $|\beta|^2$. After the measurement, the qubit state collapses to whatever value ($|0\rangle$ or $|1\rangle$) was observed, irreversibly losing memory of the former α and β amplitudes. Please note that different kinds of measurements exist; the one we referred to above is called measurement concerning the computational basis, which is among the most widely used.[4]

We now turn our attention from the theoretical definition to a more physical one. Any physical system whose state space can be described by $\mathbb{C}^2$ can serve as a qubit's physical realization.[5] These systems are referred to as quantum *two-level* systems, as their state can be described by a vector in a 2-dimensional Hilbert space. The state of an isolated quantum mechanical system composed by n two-level system, called *quantum n-register*, is described by a vector in a 2^n-dimensional Hilbert space ($\mathcal{H}^{2^n}$). This means that a state of an n-register can represent the superposition of 2^n basis states, which is the cornerstone of quantum parallelism. Formally, an n-register state can be described by the linear combination of the basis vector $\{|0\rangle, \ldots |2^n - 1\rangle\}$:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle, \qquad \text{with} \quad \sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1. \tag{1}$$

[2]Please refer to section B.1 in Appendix B for an extensive introduction to this formalism.

[3]Since a qubit has more states available than simply two levels, often it is useful to visualize it as a point on the so-called *Bloch sphere*, which is a unit sphere in a three-dimensional Euclidean space with the north and south pole corresponding to the computational basis states $|0\rangle$ and $|1\rangle$, respectively (Section B.4 in Appendix B).

[4]Other measurements are discussed in the Appendix (Section B.2.3).

[5]Some physical realizations of qubits are discussed in the Appendix (Section B.3).

If we measure the state $|\psi\rangle$ with respect to the computational basis, then we get one of the basis state labels i with probability $p(i) = |\alpha_i|^2$ and the corresponding post-measurement state is $|i\rangle$. Usually, binary labeling is used to denote the 2^n basis states. For example, the state of a two-qubit system can be expressed as $|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$ with $\sum_{i,j} |\alpha_{ij}|^2 = 1$.

The composition of two or more quantum systems is represented by the tensor product. For example, if two systems A and B have the states $|\psi_A\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|\psi_B\rangle = \gamma|0\rangle + \delta|1\rangle$, respectively, then the system C composed by A and B has the state $|\psi_C\rangle = |\psi_A\rangle \otimes |\psi_B\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle$. Finally, note that a state $|\psi\rangle \in \mathcal{H}^{2^n}$ of an n-register is called *entangled* if it that cannot be decomposed as the tensor product of n single qubit states.

3.2 Gate Model

At the heart of the GM, there is the concept of *circuit* model, i.e., a sequence of building blocks that realize elementary operations. Such building blocks are called *gates*. Thus, a gate encodes a well-controlled operation acting on a single qubit or a subset of qubits [134, 146] in a given system. When acting on more than one qubit, these gates can give rise to the phenomenon of entanglement establishing a strong correlation among qubits.

As a direct consequence of the postulates of quantum mechanics,[6] quantum gates are represented by unitary operators $\hat{U}$, i.e., $\hat{U}^\dagger\hat{U} = \hat{I}$ [18, 134]. This property automatically translates into saying that all the quantum gates must be reversible, unlike classical logic gates. However, there are exceptions to such a rule. For example, measurements are transformations on qubits that are allowed not to be reversible.

Being unitary operators, gates can be represented in different ways. Although it might be easier to formalize them as matrices, typically the Dirac notation, or the outer product among state vector, is leveraged:

$$U = \begin{bmatrix} u_{00} & \cdots & u_{0(2^n-1)} \\ \vdots & \ddots & \vdots \\ u_{(2^n-1)0} & \cdots & u_{(2^n-1)(2^n-1)} \end{bmatrix} = \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} u_{ij}|i\rangle\langle j|, \tag{2}$$

where $\{|i\rangle\}_{i=0,\ldots,2^n-1}$ is the currently used basis. Hence, in the simplest case of a single-qubit gate, it is represented by a 2×2 unitary matrix whose coordinates are completely specified by its action on the basis states. More formally, single-qubit gates describe transformations that belong to the Lie group of 2×2 unitary matrices with determinant 1, called special unitary group of degree 2 ($SU(2)$) [18].

A particularly useful set of one-qubit gates are the *Pauli gates*[7] that, together with the identity operator, span the vector space formed by all one-qubit unitary operators. In other words, any one-qubit gate can be expressed as a linear combination of the Pauli gates. It is worth mentioning that the Pauli gates $\hat{\sigma}_x$, $\hat{\sigma}_y$, and $\hat{\sigma}_z$ correspond to rotations around the x-, y-, and z-axes of the Bloch sphere, respectively. As an example, the $\hat{\sigma}_x$ matrix represents an operation called the "bit flip," or NOT gate, which maps the $|0\rangle$ to $|1\rangle$ and vice versa, in the computational basis:

$$\hat{\sigma}_x|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} |0\rangle = (|0\rangle\langle 1| + |1\rangle\langle 0|)\,|0\rangle = |0\rangle\langle 1|0\rangle + |1\rangle\langle 0|0\rangle = |1\rangle. \tag{3}$$

Interestingly, we can notice that the operation in Equation (3) resembles the NOT gate in classical computations.

[6] Postulates of quantum mechanics are summarized in the Appendix (Section B.2).

[7] Please refer to the Appendix (Table 2 in Section B.5) for the definition of the Pauli gates and other single- and multi-qubit gates mentioned in this manuscript.

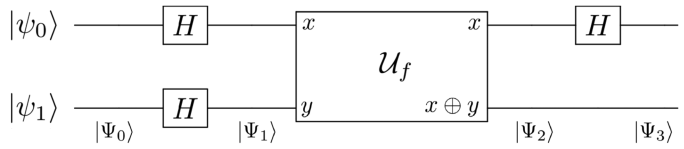


Fig. 1. Quantum circuit implementing Deutsch's algorithm [59, 134].

Apart from the single-qubit gates, there are unitaries that involve two or more qubits. Perhaps, the most famous ones are the **Controlled-NOT (CNOT)** and Toffoli gates. Note that many multi-qubit gates are designed to perform "controlled" operations, meaning that an operation is executed on a qubit, named the *target* qubit, if another qubit (called *control* qubit) is in a specific state. For example, the CNOT gate applies the NOT gate operation to the target qubit only when the control one is in the state $|1\rangle$. It is formalized as: $CNOT(|\psi\rangle \otimes |\gamma\rangle) = |\psi\rangle \otimes |\psi \texttt{ XOR } \gamma\rangle$ where $|\psi\rangle$ and $|\gamma\rangle$ are the control and target qubit, respectively.

An exciting result coming from the GM paradigm is that not all gates are "fundamental" for computations. Indeed, as it happens in the classical case, there is a set of quantum gates, named *universal*, that can be used to approximate to arbitrary accuracy any quantum circuit [134]. The Hadamard (H), Controlled-NOT (CNOT), phase (S), and $\pi/8$ (T) gates constitute a universal quantum computation set of gates. Although the S-gate can be constructed from the T-gate, it is typically considered an element of the universal set due to its extensive usage in fault-tolerant circuit construction.

As mentioned above, a circuit is made of a sequence of gates acting on a given set of qubits. To characterize a circuit, two metrics are typically reported, namely, the *width* and the *depth*. The first one refers to the number of qubits involved in the calculations, while the second one represents the longest path in the circuit, resembling the largest number of operations applied to a given qubit from the beginning to the end of the computations. In Figure 1, we report an example of a circuit in which we applied single- and two-qubit gates. Specifically, the circuit represents one of the most famous algorithms in the scientific community called the Deutsch's algorithm [59], which determines if a given Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is balanced or constant.

As we can see from Figure 1, a quantum circuit is depicted as a set of qubits, represented by lines, to which the gates (squares) are applied, and where the order from left to right represents the flow of time. However, it is worth noting that there is not an actual "flow" of the qubits in a quantum device. Indeed, the state of qubits at each given time represents the state of the QPU, while gates are the transformations applied to them to change the device's status.

3.3 Adiabatic Quantum Computation

In the previous section, we introduced the GM in which a qubits' state is evolved by applying a series of gates. In such a design, the time evolution is discretized, since the different operations are performed at subsequent time instants. Quite differently, the Adiabatic approach to quantum computation [71] leverages a time-continuous evolution of the qubits' state according to the Schrödinger equation [159]. Moreover, it does not require nor allow to apply any transformations, such as gates, directly on qubits. Instead, everything is encoded into a quantum operator, called ***Hamiltonian*** **($\hat{H}$)**, that describes the forces to apply to a given system of qubits to move it into the desired state over time. However, it is possible to show that these two approaches are polynomially equivalent [7, 71] by using the technique introduced in Reference [121]. Such a goal is reached by discretizing the evolution time interval and applying the Trotter formula [71] to each time segment.

For a deep dive into the AQC, we refer the reader to the comprehensive and fascinating review by Reference [11]. In what follows, we assume that the reader knows what a Hamiltonian is or at least its definition in classical mechanics. For our purposes, we need to consider that $\hat{H}$ can be interpreted as the energy operator, in the sense that its eigenvectors are the energy eigenstates, and its eigenvalues are the energies of the corresponding eigenvectors. Hence, as it will become obvious from the following discussion, in the AQC, any problem is recast as an energy minimization one. The key ingredient of the AQC is that $\hat{H}$ is not constant, rather, it varies with time: $\hat{H} \rightarrow \hat{H}(t)$. The AQC derives its name from the Adiabatic Theorem [131], which asserts how to follow the evolution of a system when $\hat{H}(t)$ varies slowly enough over time [6, 35, 70].[8,9]

We now briefly report what a "slowly enough varying" Hamiltonian means. As previously mentioned, AQC exploits the continuous time-evolution of a system of qubits within a time interval $[0, T]$, where T represents the end of the adiabatic evolution. Commonly, the varying Hamiltonian is expressed as a one-parameter function, given by $\hat{H}(s)$ where $s = t/T \in [0, 1]$. Note that the eigenvalues of $\hat{H}(s)$, indicated as $\xi_i(s)$, represent different energy levels of the system ordered with increasing values of $i \geq 0$ (e.g., the ground state energy is $\xi_0(s)$). To ease the understanding of the following concepts, we indicate the eigenstate for $\hat{H}(s)$ related to the energy eigenvalue $\xi_i(s)$ as $|\phi_i(s)\rangle$, where we stressed the time dependence through the parameter s. Therefore, the instantaneous eigenstates for $\hat{H}(s)$ are the states satisfying $\hat{H}(s)|\phi_i(s)\rangle = \xi_i(s)|\phi_i(s)\rangle$. For example, the ground state of $\hat{H}(s)$, $|\phi_0(s)\rangle$, is characterized by $\hat{H}(s)|\phi_0(s)\rangle = \xi_0(s)|\phi_0(s)\rangle$. Given such definitions and restricting our interest to the ground state of $\hat{H}(s)$, the adiabatic theorem ensures that if it is guaranteed that the difference among the ground and the first excited state energies, $\Delta = \min_{0 \leq s \leq 1}(\xi_1(s) - \xi_0(s))$, remains large enough throughout the entire adiabatic evolution, then the system is likely to lie in the ground state of the instantaneous Hamiltonian. Specifically, such a requirement directly affects the length of the adiabatic evolution, since $T \propto \Delta^{-2}$. As mentioned above, $\hat{H}(s)$ is interpolated between an initial ($\hat{H}_I$) and a final ($\hat{H}_P$), also called the problem, Hamiltonians. Specifically, the "problem" operator encodes the solution to a given problem in such a way that its ground state configuration, $|\phi_0(s = 1)\rangle$, expresses the solution to the optimization problem at hand. Concerning the AQC, the strategy is first to prepare the qubits in the ground state of $\hat{H}_I$, and then the operator is evolved towards $\hat{H}_P$ by following the so-called adiabatic path $\hat{H}(s) = (1 - s)\hat{H}_I + s\hat{H}_P$. From a mathematical perspective, $\hat{H}_I$ and $\hat{H}_P$ can be defined in terms of the Pauli-matrices. Concerning a system made by n-qubits, the initial Hamiltonian can be defined as: $\hat{H}_I = \sum_{i=0}^{n-1} \frac{1}{2}(1 - \hat{\sigma}_x^i)$, where $\hat{\sigma}_x$ is the Pauli-x operator. The ground state of $\hat{H}_I$ is characterized by all qubits aligned with the x-axis, i.e., the state $|+\rangle^{\otimes n} = \frac{1}{2^{n/2}} \sum_{z_1} \sum_{z_2} \cdots \sum_{z_n} |z_i\rangle^{\otimes n}$, where $|z_i\rangle \in \{|0\rangle, |1\rangle\}$, which resembles the uniform superposition of the computational basis states. To obtain such a state, one couples a magnetic field, called the transverse or disordering field, in the x-direction to each quantum bit. The choice of such a shape for $\hat{H}_I$ relates to specific symmetries that the operator must satisfy to allow for the sought computations. For example, such a choice is mandatory to avoid the so-called "level crossing" [71]. Moreover, $\hat{H}_P$ is typically expressed in terms of the $\hat{\sigma}_z$ operator so the final configuration of the qubit system is represented by eigenvectors of $\hat{\sigma}_z$, i.e., each qubit can be found either in the $|0\rangle$ or $|1\rangle$ state. Such a configuration is nothing more than a classical sequence of bits. Thus, by starting from an initial quantum

[8]For the formal definition and proof of the Adiabatic Theorem, we refer the reader to Section B.7 in the Appendix.

[9]The Adiabatic Theorem states that if a system is initially in the kth state of well-defined energy, then it will stay in this state when the Hamiltonian is changed sufficiently slowly [31]. However, we focus our attention on the ground state because, in adiabatic quantum computing, a problem is typically encoded as a Hamiltonian whose ground state is the problem solution.

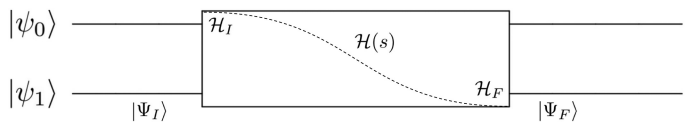


Fig. 2. Schematic adiabatic evolution for the Deutsch's algorithm [59].

superposition, the adiabatic process evolves the state into a classical configuration that can then be read to return the solution to the problem at hand. One of the first "proof of concept" to show the effectiveness of such a computation model was realized in Reference [70] for solving instances of the satisfiability problem. Apart from such a first example, many optimization problems can be recast as an energy minimization problem, such as the Grover search algorithm [71, 72, 80]. We report in Figure 2 a schematic representation of an adiabatic path representing the Deutsch's algorithm [59]. The reader might compare such an image with Figure 1 to notice the difference between the two designs.

4 THE QUEST FOR QUANTUM SUPREMACY

The first claim about an attainable advantage with a hypothetic quantum computer upon a classical one is attributed to Feynmann [73]. However, at that time, such an idea concerned quantum simulations of physical systems rather than universal computations. Indeed, considering the simulation of a dynamical system characterized by $N = 2^d$ degrees of freedom, one can immediately see that a classical simulation living in a 2^d-dimensional Hilbert space requires resources that grow exponentially with the size of the problem, while only polynomially if using a quantum device [121].

However, soon after, researchers started to think about gaining an advantage over generic classical computations by leveraging quantum effects. Thus, the concepts of "quantum advantage" (or "quantum speedup") and "quantum supremacy" [86, 145] appeared in the literature. Despite the fact that both terms refer to the same principle, loosely speaking, one can distinguish them considering what follows. The first one typically refers to the ability of a quantum computer to perform a given computation faster than a classical one. Differently, the second one assesses the ability of a quantum computer to find solutions to a problem not resolvable by a classical computer (or at least not in a reasonable amount of time). Thus, even though both terms express the same principle, sometimes they are used with slightly different purposes.

One of the most important phenomena for quantum computations is the entanglement, which allows propagating the action on a qubit to others and compressing the resource requirements to describe a given state. Highly entangled states cannot be simulated efficiently by classical systems, therefore proving the advantage of quantum devices over classical ones. Interestingly, assuming the existence of computational tasks beyond any classical computer's capability that can be solved with a universal quantum computer, then in that case, it could be possible to refute the "extended Church-Turing thesis." As we mentioned previously, concerning an algorithmic point of view, the size of the Hilbert space grows exponentially for a classical algorithm while only polynomially in the quantum case. Moreover, quantum phenomena, such as entanglement, superposition, and tunnelling, help to navigate such a vast space and should allow for a quantum speedup [4, 85, 121]. Concerning the GM, Deutsch [58] was among the first ones to show that a quantum circuit can be built of reversible quantum logic gates to compute any classical function f defined on n bits. In 1992, the first example of quantum advantage in computation was formalized in Reference [59] in which the authors showed that for a specific class of problems, a quantum device required exponentially less time to solve them than any deterministic classical approach.

When talking about quantum advantage or speedup, two algorithms are typically cited: Grover's search [80] and Shor's factoring [171] algorithms. However, there is a fundamental difference in how they obtain the advantage. Indeed, while for the first one the "modest" quadratic speedup over the classical formulation is mathematically demonstrated, concerning the second one this is not the case. Specifically, all that can be said about Shor's algorithm is that it reaches an exponential speedup over the most efficient classical analogous available today. However, nothing forbids that a new classical formulation might reduce or eliminate such a gap.

Generally speaking, there are several definitions of quantum speedup. "Provable quantum speedup" stems from the existence of a mathematical proof that there is not any classical algorithm that can perform better than the quantum one [25, 80]. A different concept is expressed by the "strong quantum speedup" [139], which considers the performance of the best classical algorithm. Shor's algorithm [171] is an instance of such a definition. Indeed, although classical algorithms require super-polynomial cost in the number of digits, the proof of an exponential lower bound for classical factorization has not been found yet. Thus, one typically adopts the concept of quantum speedup by referring to the comparison among the best-known classical algorithm, which might not be the best possible one, and its quantum counterpart.

Another class of problems probed to prove and harness a quantum advantage is the sampling problems' class [86], such as: constant-depth circuits [181], boson sampling [1], and random quantum circuits containing commuting and non-commuting gates [33, 38, 169]. Such examples are somehow in between the factoring algorithm [171] and analog quantum simulators [49, 51, 74].

In Reference [153], the authors performed experiments on a "D-Wave Two" quantum annealer. As a benchmark, they tasked the QPU with finding the ground state of a 2D planar graph of an Ising spin glass model, which is known to be an NP-hard problem [17]. The authors compared the results from simulated annealing [112], simulated quantum annealing [127, 157], and the actual quantum device from D-Wave [26, 84, 98, 99]. However, they did not observe any evidence for a genuine quantum speedup from the experiments. Recently, Google claimed to have reached quantum supremacy [16]. In their work, the authors proposed an ad hoc experiment involving a QPU containing 53 qubits and a circuit depth of 20 and claimed that a classical computer would have required thousands of years to simulate the obtained results. However, several research groups immediately replied to such a claim by showing that it was possible to simulate such a quantum computer in just slightly more than two days [140] or even less [199]. Such profoundly different results witness the difficulty that researchers typically face when trying to estimate the real power of **Noisy Intermediate-Scale Quantum (NISQ)** devices.

We conclude this brief overview over the concept of quantum supremacy and the various attempts to harness it by observing that, despite the tremendous efforts that scientists are devoting to embracing the quantum phenomena, such a goal is still far from being reached. Moreover, it is clear that any claim about quantum supremacy must undertake a detailed analysis based on the most recent achievements in classical simulation algorithms [39, 102, 147].

5 QUANTUM LEARNING

The research on Quantum Neural Networks is a quest started more than 20 years ago [14, 68, 163, 165]. Studies from Behrman et al. [21] and Toth et al. [184] are examples of seminal works that introduced the concept of QNNs. Behrman et al. [21] described a mathematical model based on quantum dot molecules and showed by simulations that such a model was able to realize any classical logic gate. Instead, Toth et al. [184] proposed a more biologically inspired architecture, where quantum dots were coupled to form a cellular structure in which near-neighbor connectivity allowed the information to flow. The realization of a quantum analogue of an **Artificial Neural Network (ANN)**, as an algorithm able to combine features from both the classical and quantum

worlds, is a long-standing dream for the scientific community [163]. In the classical realm, a feed-forward ANN is a universal approximator of continuous functions [95]. Thus, a QNN should at least satisfy such a requirement.

Classically, an ANN is made by a sequence of layers. Each of them applies a specific mathematical transformation to its input and produces an output taken as input by the subsequent layer in the network. Moreover, the various layers are typically interleaved by non-linear functions to enhance the ANN representation power. Indeed, if only linear operations are considered, then one could reduce the entire network to a single affine mapping. ANNs owe their name to their structure loosely inspired by the human brain. Moreover, still based on a biological analogy, the basic building block of ANNs is the artificial neuron, called *perceptron*, a computational model proposed initially by Rosenblatt [154]. These algorithms typically deal with large amounts of data aiming at finding patterns to describe them by surfing the parameters' space. Thus, they are perfect candidates to leverage the advantage of quantum phenomena that allows navigating more efficiently high-dimensional spaces.

It is worth mentioning that classical deep ANNs have been exploited to solve quantum problems in disciplines such as chemistry [118, 144, 167] and physics [40, 47, 135], among others. In such contexts, a DL model can be trained, for example, to predict the quantum mechanical wavefunction for a given many-body system [42], to predict the interatomic potential energy surfaces [176], to identify distinct phases of matter and the transitions between them [43], to mention some. In spite of this topic being fascinating, a detailed discussion about these techniques is out of scope for our work. Thus, in what follows, we will focus on approaches based on quantum models only.

Concerning the quantum realm, in 1994, Lewenstein [117] was among the first to propose a quantum-mechanical perceptron as an ideal basic element that processes input quantum states through unitary transformations. Since then, several studies have been conducted to formulate QNNs as made up of basic cells such as QPs or as entirely new computing architectures. Indeed, the design of QNNs can be much different from that of classical ANNs, meaning that the first ones do not require to be made of several layers of QPs. For example, they can be realized as variational circuits (see Section 5.4) without relying on direct implementation of a basic cell resembling the quantum analogue of a perceptron.

In the following sections, we first introduce the concept of quantum embedding (Section 5.1) and then we describe variational algoritmhs (Section 5.2). Subsequently, we report several designs for QPs (Section 5.3) and QNNs (Section 5.4). Finally, we discuss trainability issues, e.g., barren plateaus (Section 5.5). We focus primarily on the GM computational paradigm, since it is currently the most widely adopted approach in such a context, and it offers a closer analogy to classical approaches, especially from a computer scientist's point of view. We also report some interesting formulations based on AQC. We leave a more detailed analysis of the results obtained with the AQC paradigm for future work.

5.1 Encodings of Data

A quantum machine learning algorithm,[10] such as a QNN, accepts quantum states as input by its nature. Hence, classical data must be translated into quantum states and transferred from a classical memory to the quantum device, a process called "state preparation."

The *basis* and the *quantum amplitudes encodings* [160] are two fundamental examples of quantum data embeddings. The basis encoding (also called bit encoding) associates each data input with a computational basis state of a qubits system. This is equivalent to saying that each data

[10]Following the categorization used in References [8, 160], we use the term quantum machine learning for approaches that use a quantum device to process classical or quantum data.

item $\mathbf{x} \in \mathbb{R}^d$ is first transformed to a binary string $\mathbf{b}$ of length n and that each binary string is uniquely associated with a computational basis state of an n-qubit system, $|\mathbf{b}\rangle$. For example, given a vector $\mathbf{x} = [0.3, -0.4]$, we may use a bit to encode the sign and T bits to encode each number in the interval $[0, 1)$ according to a binary fraction representation $\sum_{k=1}^{T} b_k \frac{1}{2^k}$ of precision T. In the case of $T = 4$, we obtain $\mathbf{0}.3 \rightarrow 0\,0100$ and $-0.4 \rightarrow 1\,0110$, therefore, the vector $\mathbf{x}$ is mapped to the binary string $\mathbf{b} = 0010010110$ that is represented by the quantum state $|0010010110\rangle$ of 10 qubits. Amplitudes encoding, instead, associates real vector components with quantum amplitudes with the only caveat that the original real vectors must be normalized. Formally, given a vector $x \in \mathbb{R}^d$, where $d = 2^n$, we have the following mapping:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{d-1} \end{bmatrix} \quad \rightarrow \quad |x\rangle = \frac{1}{\|\mathbf{x}\|} \sum_{i=0}^{d-1} x_i |i\rangle, \tag{4}$$

where $|i\rangle$ represents the ith computational basis state for an n-qubit system. For example, if $\mathbf{x} = [0.3, -0.4]$, then $|x\rangle = (0.3\,|0\rangle - 0.4\,|1\rangle)\,/0.5$, upon normalization. Note that the amplitude encoding requires exponentially fewer qubits than the base encoding to embed a vector into a quantum state. An entire dataset can be represented in the computational basis by considering the amplitude encoding of the concatenation of all the input data. Hence, the main advantage of this encoding is that it only requires $\log_2 Md$ qubits to encode a dataset of size M and dimensionality d [160].

5.2 Variational Algorithms

Currently, the leading approach to the optimization of quantum circuits [22] exploits **Variational Quantum Algorithms (VQA)** [44, 81, 129, 142], hybrid quantum-classical algorithms that leverage both quantum as well as classical components. Their name is inspired by the well-known result from physics called the *Variational Principle*, which provides an upper bound of the ground state's energy of a quantum system.[11] Generally speaking, the Variational Principle gives a recipe to construct a trial state (i.e., a quantum state parameterized by learnable parameters) for optimization purposes.

From a computer science point of view, one can formulate a VQA as follows: A hybrid quantum-classical architecture comprising a set of quantum operations, the *ansatz*, that are applied to the initial qubits state to produce a final state, called *trial state*. The ansatz is characterized by a set of gates controlled by some classical parameters $\boldsymbol{\theta}$ whose values are modified with the help of a classical procedure, e.g., gradient descent. Therefore, the design of such algorithms typically relies on three main components. The first one is the ansatz, also referred to as "parametrized quantum circuit" or "variational circuit," which sometimes includes the embedding of classical data into a quantum state. Subsequently, one must specify an objective function $\mathcal{L}_{\boldsymbol{\theta}}$, whose value is used to drive the optimization procedure, and finally a classical optimization procedure is used to modify the ansatz's parameters $\boldsymbol{\theta}$ that define the transformations applied on qubits. Hence, it is clear that VQAs represent the practical embodiment of the idea of training quantum computers as we train classical neural networks. We report in Figure 3 a schematic view of a VQA.

The first application of such class of methods is the ***Variational Quantum Eigensolver* (VQE)** [104, 142, 191], initially proposed to find the lowest energy state (the ground state) of a

[11] We refer the interested reader to Section B.6 in the Appendix where we report the formal definition and proof of the Variational Principle.

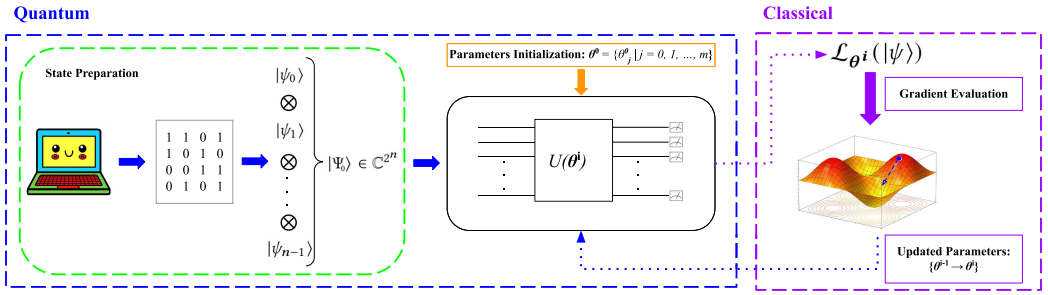


Fig. 3. Schematic view of a VQA. The quantum device implements a parametrized quantum circuit $U(\boldsymbol{\theta})$ that given an initial state $|\Psi_0\rangle$ prepares the trial state $|\Psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|\Psi_0\rangle$. Observable measurements on the trial state are used to return estimates of expectation values (e.g., the state of a certain qubit), which are used to evaluate an objective function $\mathcal{L}_{\boldsymbol{\theta}}$. A classical optimization algorithm (e.g., gradient descent) is employed to update the circuit parameters.

quantum system. Given the Hamiltonian $\hat{H}$ of a system, the Variational Principle tells us that the ground state $|\psi^*\rangle$ corresponds to the normalized state $|\psi\rangle$ that minimizes the quantity $\langle\psi|\hat{H}|\psi\rangle$ expressing the energy of the system in the state $|\psi\rangle$. That is, $|\psi^*\rangle = \text{argmin}_{|\psi\rangle}\langle\psi|\hat{H}|\psi\rangle$, and $E_{min} = \langle\psi^*|\hat{H}|\psi^*\rangle$. Therefore, the ground state can be approximated by optimizing the parameters $\boldsymbol{\theta}$ of an ansatz $U(\boldsymbol{\theta})$ to minimize the expectation value $\mathcal{L}_{\boldsymbol{\theta}} = \langle\psi(\boldsymbol{\theta})|\hat{H}|\psi(\boldsymbol{\theta})\rangle$, where $|\psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|\Psi_0\rangle$ for a given initial state $|\Psi_0\rangle$. In other words, given $\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}}\,\mathcal{L}_{\boldsymbol{\theta}}$, the state $|\psi(\boldsymbol{\theta}^*)\rangle$ approximates $|\psi^*\rangle$ and $\mathcal{L}_{\boldsymbol{\theta}^*}$ provides an upper bound for E_{min}. VQE can solve general optimization problems as long as the problem is formulated using a Hamiltonian such that its ground state corresponds to the solution of the original problem. For example, to find the binary solution x that minimizes a cost function $C(x)$ one may first find a Hamiltonian H_C that encodes the cost function $C(x)$, i.e., $H_C|x\rangle = C(x)|x\rangle$, and then use the VQE to approximate the ground state of this Hamiltonian.

The ***Quantum Approximate Optimization Algorithm*** **(QAOA)** [69] is a famous variational algorithm that defines an ansatz to generate approximate solutions to a given problem. It was initially proposed to solve combinatorial optimization problems, and then it was generalized as a standard ansatz [44, 83]. QAOA finds the trial state by repeatedly applying unitary evolutions according to a two-terms Hamiltonian [69], namely, $\hat{H}_C$ and $\hat{H}_B$. The first one encodes the classical cost function $C(x)$ to be minimized, and it is typically diagonal in the computational basis. In contrast, the second one is the mixing Hamiltonian, which coherently moves the system through different configurations in the Hilbert space seeking the ground state of $\hat{H}_C$. Typically, $\hat{H}_B$ is a global transverse field, i.e., $\hat{H}_B = -\sum_i \hat{\sigma}_x^i$. Thus, given the problem Hamiltonian $\hat{H}_C$, one can use QAOA to find its ground state by applying the evolution operator $\hat{U}$:

$$|\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle = \hat{U}(\boldsymbol{\beta},\boldsymbol{\gamma})|\psi_0\rangle = \prod_{i=0}^{p} e^{-i\beta_i\hat{H}_B}e^{-i\gamma_i\hat{H}_C}|\psi_0\rangle = e^{-i\beta_p\hat{H}_B}e^{-i\gamma_p\hat{H}_C}\cdots e^{-i\beta_1\hat{H}_B}e^{-i\gamma_1\hat{H}_C}|\psi_0\rangle, \quad (5)$$

where the set of angles $\{\boldsymbol{\beta},\boldsymbol{\gamma}\}$ are the parameters to be optimized. Starting from an initial state $|\psi_0\rangle$, e.g., the uniform superposition of the basis states $H^{\otimes n}|0\rangle$, one can evolve it by using Equation (5) and obtain $|\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle$. The parameters are then adjusted such as to minimize the objective function expressed as:

$$\mathcal{L}(\boldsymbol{\beta},\boldsymbol{\gamma}) = \langle\psi(\boldsymbol{\beta},\boldsymbol{\gamma})|\hat{H}_C|\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle = \langle\psi_0|\hat{U}^\dagger(\boldsymbol{\beta},\boldsymbol{\gamma})\hat{H}_C\hat{U}(\boldsymbol{\beta},\boldsymbol{\gamma})|\psi_0\rangle. \quad (6)$$

The parametrized evolution operator $\hat{U}(\boldsymbol{\beta}, \boldsymbol{\gamma})$ can be viewed as a layerized variational circuit:

$$|\psi_0\rangle \text{—} \boxed{U_C(\gamma_1)} \text{—} \boxed{U_B(\beta_1)} \text{—} \cdots \text{—} \boxed{U_C(\gamma_p)} \text{—} \boxed{U_B(\beta_p)} \tag{7}$$

with p repetitions of alternating cost layers, $U_C(\gamma_i) = e^{-i\gamma_i \hat{H}_C}$, and mixing layers, $U_B(\beta_i) = e^{-i\beta_i \hat{H}_B}$. Note that alternating the cost layers with the mixer layers allows enlarging the space of quantum states that can be generated with the circuit (since a circuit consisting of only cost layers is equivalent to a circuit with only one parameterized cost layer). The layers of QAOA correspond to the trotterized segments of the time evolution operator associated to the adiabatic evolution $\hat{H}(s) = (1-s)H_B + sH_C$, $s \in [0,1]$. The initial QAOA state $|\psi_0\rangle$ is set as the ground state of the initial Hamiltonian H_B (e.g., $|\psi_0\rangle = H^{\otimes n}|0\rangle$ if $\hat{H}_B = -\sum_i \hat{\sigma}_x^i$) so the adiabatic process evolves to the ground state of the problem Hamiltonian H_C.

A fundamental ingredient for QAOA is the parameter p in Equation (5), since it regulates the precision of the approximate solution. Growing p means reducing the "distance" from the optimal solution. However, for large p, decoherence starts to play a significant role in the computation, since the ansatz will contain more gates. We recommend to the reader the original paper [69] for a complete discussion about the impact of the p parameter and QAOA in general.

In the following two sections, we move to the topics of QPs and QNNs. Unlike classical learning approaches, quantum ansatzes are typically specialized to build perceptrons or represent an entire network. From such an observation, we first delve deep into the QPs and then to full QNNs proposals.

5.3 Quantum Perceptron

From the classical perspective, a perceptron is a computational model characterized by a non-linear response to its input and is parametrized as:

$$s_j = f\left(\mathbf{w}_j^T \mathbf{x} + b_j\right), \tag{8}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input, $\mathbf{w}_j \in \mathbb{R}^d$ and $b_j \in \mathbb{R}$ are learnable parameters (weights and bias), and f is a non-linear activation function. Depending on the value of s_j, the perceptron is said to be either active or at rest. One of the simplest forms of an artificial neural network is the **Multilayer Perceptron (MLP)**, i.e., a network composed of multiple layers of perceptrons (called inner-product or fully connected layers). Each layer is composed of many perceptrons whose action is a linear projection of the input followed by a non-linear element-wise activation function, i.e., $\mathbf{s} = f(\mathbf{W}^T\mathbf{x} + \mathbf{b}) \in \mathbb{R}^m$, where $\mathbf{W} \in \mathbb{R}^{d \times m}$ and $\mathbf{b} \in \mathbb{R}^m$ are the weight matrix and the bias vector, respectively. The fully connected layer is one of the basic building blocks for Deep Neural Networks, so the perceptron can be considered a "fundamental unit" of classical NNs. Therefore, the research in the field of QNN initially focused on defining quantum algorithms capable of reproducing the functionalities of classical perceptrons, e.g., for classification tasks.

The implementation of the Quantum Perceptron is still an active research topic. We know that classically there is one reference implementation of such objects (Equation (8)). However, due to the intrinsic differences among the available quantum hardware platforms, different formulations report advantages and disadvantages, depending on the context in which they are used, thus allowing for distinct implementations equally valid in the respective fields of application. To ease the reader's understanding of the current state-of-the-art QPs, we report a summary in Table 2. It should be noted that some approaches have been implemented on real QPUs (e.g., IBM Q 5), while other methods have only been tested using numerical simulations so far.

Due to the linear, unitary, and non-dissipative nature of the quantum transformations, one of the hardest challenges to design a QP resides in designing and implementing non-linear activation

Table 2. Summary Comparison among Various Quantum Perceptron Proposals

Work	Formulation	Impl.	Code
	Using the Gate Model		
Schuld et al. [165]	Quantum circuit using inverse quantum Fourier transform to reproduce a step activation function. The input is encoded into an n-qubit state.	Simulation	✗
Cao et al. [41]	Circuit based on RUS (using a tangent-based non-linear function). The d-dimensional input data is encoded into a d-qubit state.	Simulation	✗
Hu [96]	Circuit based on RUS (using a sigmoid-based non-linear function). The d-dimensional input data is encoded into a d-qubit state.	IBM Q 5	✗
Du et al. [62]	Variational quantum circuit based on a parametrized Grover oracle learned with a classical optimization approach.	Simulation (pyQuil)	✗
Wiebe et al. [194]	Quantum search problem, based on the version space interpretation of the perceptron, based on the Grover's algorithm.	nr	✗
Liu et al. [120]	Given a supervised dataset, construct the weights' matrix as the tensor product of input and target and then apply SVD to it.	nr	✗
Wiersema et al. [195]	Qubit represented as a density matrix that describes the classifier as a Boltzmann-like finite-temperature distribution.	Simulation (Tensorflow)	✓*
Tacchino et al. [180]	Quantum variational circuit in which the d-dimensional input datum is encoded into the d coefficients of an n-qubit state ($d = 2^n$).	IBM Q 5 "Tenerife"	✓°
	Using the Adiabatic Model		
Soloviev et al. [178]	Perceptrons and synapses are implemented as JJ-based superconducting circuits.	nr	✗
Torrontegui et al. [183]	Two-level system (qubit) with a non-linear excitation response to the input field.	Simulation (Tensorflow)	✓•

The symbol "nr" stands for "not reported," meaning that the authors did not explicitly report that specific information. In the column "Impl," we reported whether the method was implemented on an actual QPU or tested using numerical simulations only (reporting in brackets the library used when the information was available).
Code: ✗ the code is not publicly available, ✓ the code is available either on GitHub (∗), upon request to the authors of the original work (○), or in the supplementary material of the original work (•).

functions for quantum computations. Schuld et al. [165] proposed one of the first designs for a quantum perceptron model imitating the step-activation function of a classical binary perceptron. Their main idea was to encode a normalized version $\psi \in [0, 1)$ of the input signal $\mathbf{w}^T\mathbf{x}$ into the phase of a quantum state of n qubits and then use a phase estimation algorithm with a precision of τ, implemented using an inverse quantum Fourier transform. They proved that measurements on the first qubit of the output quantum state could have been used to estimate if ψ was larger than $1/2$, hence reproducing the behavior of a step activation function.

In 2017, Cao et al. [41] proposed an approach based on the ***Repeat-Until-Success*** **(RUS)** [32, 138] technique to synthesize quantum gates. As we mentioned previously, the classical perceptron evaluates the non-linear transformation $s = f(\mathbf{w}^T\mathbf{x} + b)$ that, once thresholded, defines the activation status of the perceptron. In such a context, $\theta = \mathbf{w}^T\mathbf{x} + b$ is the input signal to the neuron and f represents the non-linear operation whose output s lies in $[l, u]$, where typically $l = -1$ or 0 and $u = 1$. In Reference [41], such an operation is mapped to the quantum realm by defining the perceptron as a qubit whose state is defined as: $\hat{R}_y(s\frac{\pi}{2} + \frac{\pi}{2})|0\rangle = \cos(s\frac{\pi}{4} + \frac{\pi}{4})|0\rangle + \sin(s\frac{\pi}{4} + \frac{\pi}{4})|1\rangle$, where $s \in [-1, 1]$ and $\hat{R}_y$ represents a rotation around the y-axis as described by $\hat{\sigma}_y$. Note that as far as the extreme values of s are concerned, they recover a classical behavior being the system in the state $|0\rangle$ or $|1\rangle$. Instead, in all other cases it behaves quantum-mechanically. The quantity $s = f(\theta) = f(\mathbf{w}^T\mathbf{x} + b)$ is computed using a quantum circuit where the input data is encoded into a state vector $|x\rangle = |x_1 \cdots x_d\rangle$ acting as the control register over an ancilla qubit. Initially, a rotation $R_y(2w_i)$ conditioned on $|x_i\rangle$ followed by a rotation $R_y(2b)$ is applied on the ancilla qubit, which is equivalent to say that $R_y(2\theta)$ is applied on the ancilla qubit conditioned on the state $|x\rangle$.

Then, the non-linearity is realized by using the rotation $R_y(2f(\theta))$ that is approximated by a class of RUS [138] circuits each implementing a rotation with an angle equal to $\arctan(\tan^2(\theta))$. Repeating the RUS circuit k times gives a rotation with an angle $\arctan(\tan^{2^k}(\theta))$ that is a sigmoid-like, tangent-based, non-linear function. Stemming from the same approach, Hu [96] implemented a different RUS circuit to represent a sigmoid-based activation function, $f(x) = \arcsin(\sqrt{\text{sigmoid}(x)})$, thus avoiding the drawbacks of using periodic functions, such as the tangent. Wiebe et al. [194] adopted a very interesting and completely different perspective. Indeed, the authors reported that the only way to unveil a quantum advantage is to formalize approaches that do not try to emulate classical algorithms in the quantum world. Instead, a new point of view is needed. Thus, they based their approach on the *Version Space* interpretation of the perceptron and Grover's search algorithm [80]. Specifically, they exploited a general method referred to as amplitude amplification [37] for which Grover's algorithm represents a particular case. Using the *Version Space*, the authors posit the problem of training a perceptron as a search problem (i.e., search for the optimal parameters rather than "learn" them), hence fully exploiting the quantum advantage offered by the amplitude amplification technique. Wiersema et al. [195] exploited the formalism of density matrices to cast the perceptron learning problem in terms of a quantum cross-entropy:

$$\mathcal{L}_q = -\sum_{\mathbf{x}} q(\mathbf{x}) \text{Tr}\{\eta_{\mathbf{x}} \ln \rho_{\mathbf{x}}\}, \tag{9}$$

where $q(\mathbf{x})$ is the empirical distribution of observing the datum $\mathbf{x} \in \{1, -1\}^d$, $\eta_{\mathbf{x}} \equiv |\Psi\rangle\langle\Psi|$, with $|\Psi\rangle = \sqrt{q(y|\mathbf{x})}|0\rangle + \sqrt{q(-y|\mathbf{x})}|1\rangle$, is the density matrix associated with the distribution $q(y|\mathbf{x})$ of observing the label $y \in \{1, -1\}$ given $\mathbf{x}$, while $\rho_{\mathbf{x}} \equiv \rho(\mathbf{x}, \mathbf{w}; \mathbf{y})$ is the density matrix associated to the model conditional distribution $p(y|\mathbf{x}; \mathbf{w})$. Note that such a formulation resembles the classical cross-entropy objective: $\mathcal{L}_{cl} = -\sum_{\mathbf{x}} q(\mathbf{x}) \sum_y q(y|\mathbf{x}) \ln p(y|\mathbf{x}; \mathbf{w})$. The matrix $\rho_{\mathbf{x}}$, which embodies the perceptron's description, represents a finite-temperature description of a qubit formulated as:

$$\rho_{\mathbf{x}} = \exp\left(-\beta \sum_{k \in \{x,y,z\}} h_k \hat{\sigma}_k\right)/Z, \tag{10}$$

where the inverse temperature β is set equal to -1, $\hat{\sigma}_k$ are the Pauli matrices, the term $h_k \in \mathbb{R}$ is evaluated as $\mathbf{w}_k^T \mathbf{x}$ (i.e., for each Pauli matrix there is one set of weights), and Z is such that $\text{Tr}\{\rho_{\mathbf{x}}\} = 1$. Thus, there is a direct dependence from the classical input data of the perceptron state. When the dataset $\{\mathbf{x}\}$ is linearly separable, the convergence point is similar to the classical case. Instead, in the presence of label noise, there is a contribution also from Pauli operators other than $\hat{\sigma}_z$ that allow the quantum perceptron to perform better than its classical counterpart.

Torrontegui et al. [183] based their approach on the similarity between the classical perceptron and a single qubit interpreted as a two-level system. Indeed, they implemented a perceptron with sigmoid non-linear activation as a qubit that undergoes a unitary transformation, parametrized by an external input field $\hat{\theta}_j$, given by

$$\hat{U}_j(\hat{\theta}_j; f) = \exp\left(\text{i} \arcsin\left(f(\hat{\theta}_j)^{1/2}\right)(\hat{\sigma}_y)_j\right) \quad \text{with} \quad \hat{\theta}_j = \sum_{k<j} w_{jk}(\hat{\sigma}_z)_k - b_j, \tag{11}$$

where j is the label of jth perceptron of a layer, $\hat{\theta}_j$ is the quantum field generated by neurons in earlier layers, $\hat{\sigma}_y$ and $\hat{\sigma}_z$ are the Pauli matrices, and $f(\cdot)$ encodes the non-linear activation function. The authors formalized the implementation of such a model using an adiabatic process. Specifically, they proposed implementing the system's evolution as a single adiabatic path for an Ising model of interacting spins. The authors proved that such a perceptron acts as a universal approximator

with such a definition. Moreover, they report how it is possible to treat such objects as the basic building block for large neural networks.

In Liu et al. [120], the authors proposed a quantum algorithm based on unitary weights, suitable to implement a one-iteration learning approach. The weights' operator is constructed as the tensor product $\hat{w} = \sum_j |y_j\rangle\otimes\langle x_j|$, where $\langle x_j|$ represents the conjugate transpose matrix of a training input and $|y_j\rangle$ is the desired output. The Singular Value Decomposition procedure is then applied to $\hat{w}$ to decompose it as the product $\hat{F}\Sigma\hat{w}_{\text{new}}$, where $\hat{F}$ and $\hat{w}_{\text{new}}$ are unitary matrices, and Σ is the diagonal matrix of singular values. Since Σ may not be unitary, it is replaced with a diagonal unitary matrix and the output of the quantum perceptron on an input $|x\rangle$ is defined as $\hat{F}\Sigma_{\text{new}}\hat{w}_{\text{new}}|x\rangle$. The authors showed how to approximate several quantum gates with such an algorithm in their work. A similar approach is also exploited by Khan et al. [108]. Du et al. [62] proposed a variational QP implementing a parametrized oracle for a Grover-like search algorithm. The parametrized quantum circuit learns features from the input dataset, and a classical optimization algorithm guides the parameters update. Differently from the previously introduced approaches, Soloviev et al. [178] reported a discussion about how to exploit superconducting circuits equipped with **Josephson junctions (JJ)** [100, 101] to physically realize perceptrons and synapses to build up a neural network. Specifically, the authors exploited the operational principles of adiabatic superconducting cells to implement multilayer perceptrons. The current-phase relation is used to implement the non-linear response in such a design. Instead, the system's input is supplied through magnetic fluxes to which the circuits, operating at a few kelvins, couple.

Finally, in 2019, Tacchino et al. [180] proposed a binary perceptron model (where both input and weight vectors are limited to binary values) that can be implemented on NISQ [146] devices. To reduce the resource demand of the algorithm, the authors formalized a new procedure to generate multipartite entangled states based on the so-called hypergraph states [75, 155]. First, the input vector $\mathbf{x} \in \{-1, 1\}^d$ and the weight vector $\mathbf{w} \in \{-1, 1\}^d$ are encoded on the quantum states $|\psi_{\mathbf{x}}\rangle$ and $|\psi_{\mathbf{w}}\rangle$, respectively, using $n = \log_2 d$ qubits (see Equation (4)). Then, given the preparation state as $|0\rangle^{\otimes n}$, the authors evolved the state employing two unitaries, $\hat{U}_{\mathbf{x}}$ and $\hat{U}_{\mathbf{w}}$, such that $\hat{U}_{\mathbf{x}}|0\rangle^{\otimes n} = |\psi_{\mathbf{x}}\rangle$ and $\hat{U}_{\mathbf{w}}|\psi_{\mathbf{w}}\rangle = |1\rangle^{\otimes n}$. The authors proved the final N-qubit state $|\psi_{\mathbf{x},\mathbf{w}}\rangle = \hat{U}_{\mathbf{w}}|\psi_{\mathbf{x}}\rangle$, obtained by applying $\hat{U}_{\mathbf{w}}$ after $\hat{U}_{\mathbf{x}}$, contains the scalar product among the input and the weight vectors, up to a normalization factor, in its last amplitude coefficient, i.e., $\langle 1\cdots 1|\psi_{\mathbf{x},\mathbf{w}}\rangle = \langle\psi_{\mathbf{w}}|\psi_{\mathbf{x}}\rangle = \frac{1}{n}\mathbf{w}^T\mathbf{x}$. Finally, the non-linearity was implemented as a measurement whose output represented the probability for the system of being activated by the given input pattern. The authors successfully proved that for any given weight vector, $\mathbf{w}$, the perceptron was able to single out from the 16 possible input patterns only $\mathbf{i} = \mathbf{w}$ (with output probability of 1, i.e., the perfect activation of the neuron), while all other inputs gave outputs smaller than 0.25.

5.4 Quantum Neural Networks

Before delving deep into the most recent achievements in the field of QNNs, we briefly mention a few seminal works that, although published several years ago, played a relevant role in advancing the discipline at hand. One of the first proposals concerning a hybrid algorithm for QNNs dates back to 2000 with the work of Narayanan and Menneer [133]. Using simulations, the authors compared the performance of a classical artificial network in which, from one experiment to the following, different components were substituted with their quantum analogous. Interestingly, the authors found that a fully quantum network did not report any tangible advantage compared to a "partially" quantum one. Ventura and Martinez [185] proposed a quantum version of associative memories to harness the exponential increase in storage capacity, originated from superposition effects that quantum mechanics allows, concerning classical algorithms. To their aim, the authors exploited a generalization of Grover's algorithm [80] able to fulfill the search and completion tasks.

***Parametrized Quantum Circuits* (PQCs)**, which include variational quantum algorithms, are the most commonly adopted choice to design quantum neural networks. In such a context, the qubits' states are manipulated by unitaries characterized by a set of parameters that are classically learned at training time. Finally, the value of a given observable is accessed through a measurement that outputs a discrete quantity. Even if we focus on those algorithms in what follows, it is worth mentioning that another class of quantum algorithms exploits continuous variables, namely, **Continuous-Variable (CV)** [34, 110] models. These models do not use qubits to store quantum information. Instead, they exploit continuous degrees of freedom such as the amplitude of the electromagnetic field, thus making them suited for photonic hardware. In what follows, we review the state-of-the-art QNN models. To ease the comparison among the analyzed approaches, we report in Table 3 a summary of the most relevant properties of the models.

5.4.1 The Expressivity of PQC Models. Classical feedforward networks have a layered structure where each layer processes the previous layer's output using functions depending on some trainable parameter. Since a layered structure with learnable parameters also characterizes PQC-based algorithms, they have been widely used as quantum versions of neural networks. Typically, each layer of a PQC accounts for three types of blocks: data-encoding circuit blocks $\hat{S}_i(\mathbf{x})$ (depending on the input variable), parametrized circuit blocks $\hat{U}_i(\boldsymbol{\theta}_i)$ (depending on a set of trainable parameters), and non-parametrized circuit blocks $\hat{V}_i$ (e.g., entangling operators). Given these blocks, a PQC with L layers is usually described by a unitary $\hat{\mathcal{U}}(\mathbf{x};\Theta) = \prod_i^L \hat{V}_i\hat{U}_i(\boldsymbol{\theta}_i)\hat{S}_i(\mathbf{x})$. The prediction from such a model is then evaluated as the expectation value of an observable O, over m runs of the circuit, with respect to the final state of the quantum circuit, e.g., $y_{\text{out}}(\mathbf{x};\Theta) = \langle\psi_0|\hat{\mathcal{U}}(\mathbf{x};\Theta)^\dagger O\hat{\mathcal{U}}(\mathbf{x};\Theta)|\psi_0\rangle$ where $|\psi_0\rangle$ is some initial state of the quantum system. Different choices for the implementation of the data encoding, parametric and non-parametric circuit blocks, and their mutual interaction correspond to different QNN architectures. Hereafter, to ease the notation, we merge the non-parametrized blocks with the parametrized ones, i.e., we will use $\hat{U}_i(\boldsymbol{\theta}_i)$ to denote the composition $\hat{V}_i\hat{U}_i(\boldsymbol{\theta}_i)$ referred to as a *trainable block*.

An important difference between this type of QNNs and classical NNs is that the information does not flow from one layer to the next only. Instead, the input data $\mathbf{x}$ is uploaded at each input layer $\hat{S}_i(\mathbf{x})$ (see, e.g., Figure 4(a)). Such a design directly impacts the class of functions the PQC can approximate References [76, 141, 166]. Indeed, note that a PQC made by only trainable blocks would be equivalent to a circuit made by a single trainable layer (since all the $\hat{U}_i(\boldsymbol{\theta}_i)$ are linear unitary operators, thus their composition is a linear unitary operator as well). To draw a parallel with deep learning, data-encoding layers play a role reminiscent of the non-linearity of activation functions in classical DNNs, increasing the expressive power of models as function approximators. Indeed, a classical NN composed only of linear layers is equivalent to a network consisting of only one linear layer. To have a "deep" network, i.e., a network composed of multiple layers potentially able to generate progressively more abstract features with a higher representation power, the linear operations must be interleaved with non-linearities.

Recently, Schuld et al. [166] proved that if the data-encoding block $S(\mathbf{x})$ of a PQC can be expressed via a Hamiltonian time evolution, i.e., $S(\mathbf{x}) = e^{-\mathrm{i}H_1x_1} \otimes \cdots \otimes e^{-\mathrm{i}H_dx_d}$, and it is the same in every layer, then the PQC is capable of computing a partial Fourier sum, $y_{\text{out}}(\mathbf{x};\Theta) = \langle\psi_0|\hat{\mathcal{U}}(\mathbf{x};\Theta)^\dagger O\hat{\mathcal{U}}(\mathbf{x};\Theta)|\psi_0\rangle = \sum_\omega c_\omega(\Theta)e^{\mathrm{i}\omega^T\mathbf{x}}$. Thus, it is able to approximate any square integrable function in a given interval. In such a context, the output of the PQC model is a linear combination of complex exponential functions with a set of frequencies ω and coefficients c_ω determined by the topology of the network. In particular, trainable blocks determine the expressiveness of Fourier coefficients c_ω, while the spectrum $\Omega = \{\omega\} \subset \mathbb{R}^d$ of the available

Table 3. Comparison among the Analyzed Quantum Neural Networks Designs

Work	Aim	Loss Function	Variational Parameters (n-qubits)	Optimization	Impl.		Appl. Field	Input Type	Test Datasets
					Sim.	QPU			
Perez et al. [141]	PQC as a Universal Quantum Classifier	$\sum_{i=1}^{D}\left(1-\left\lvert\langle\psi_i^{\text{true}}\lvert\hat{\mathcal{U}}(x_i;\Theta)\rvert\psi_0\rangle\right\rvert^2\right)$	nr	L-BFGS-B	✓	✗	CS	C	
Schuld et al. [161]	PQC for Binary Classification	$\frac{1}{2}\sum_{i=1}^{D}\lvert y_{\text{out}}(x_i;\Theta)-y_i\rvert^2$	$O((\log n)^k)$	Gradient Descent	✓	✗	CS	C	MNIST *, CANCER, SONAR, WINE, SEMEION
Macaluso et al. [123]	Quantum MLP for Binary Classification	$\sum_{i=1}^{D}\lvert y_{\text{out}}(x_i;\Theta)-y_i\rvert^2$	nr	Nesterov	✓ $^{\S\,\ddagger}$	✓ $^{\bullet}$	CS	C	
Beer et al. [20]	Quantum Dissipative Multilayer Perceptron	$\frac{1}{D}\sum_{i=1}^{D}\langle\psi_i^{\text{true}}\lvert\rho_i^{\text{out}}\rvert\psi_i^{\text{true}}\rangle$	$O\emptyset(2^{2N})$	Gradient Descent	✓	✗	QI	Q	
Verdon et al. [187]	PQC for Learning Hamiltonian Dynamics	$1-\frac{1}{B}\sum_{j=1}^{B}\lvert\langle\psi_j^{\text{true}}\lvert U^j(\Theta)\rvert\psi_0\rangle\rvert^{2\,\star}$	nr	Adam	✓	✗	Phys.	Q	
Li et al. [119]	Quantum CNN for image recognition	$-\sum_k^C(\mathbf{y}_i)_k\log\left(g(x_i;\Theta)_k\right)^{\dagger}$	nr	Adam	✓	✗	CS	C	MNIST, GTSRB *
Henderson et al. [92]	Hybrid classical-quantum CNN for image recognition	nr	nr	nr	✓ ††	✗	CS	C	MNIST
Cong et al. [52]	CNN-inspired QNN for quantum data analysis	$\frac{1}{D}\sum_{i=1}^{D}\lvert y_{\text{out}}(\lvert x_i\rangle;\Theta)-y_i\rvert^2$	$O(\log n)$	Gradient Descent	✓	✗	Phys.	Q	
Grant et al. [78]	Hierarchical Quantum Classifiers	$\frac{1}{D}\sum_{i=1}^{D}\lvert y_{\text{out}}(\lvert x_i\rangle;\Theta)-y_i\rvert^2$	nr	Adam	✗	✓ *	CS, Phys.	C, Q	MNIST, IRIS
Jaderberg et al. [97]	Quantum Self-Supervised Learning	$\frac{1}{D}\sum_{i=1}^{D}\log\frac{-\exp(\mathbf{z}_i^1\cdot\mathbf{z}_i^2/\tau)}{\exp(\mathbf{z}_i^1\cdot\mathbf{z}_i^2/\tau)+\sum_\alpha\exp(\mathbf{z}_i^1\cdot\mathbf{z}_i^\alpha/\tau)}$	nr	Adam	✓	✓ $^{\circ}$	CS	C	CIFAR10*
Mari et al. [126]	Transfer Learning	$-\frac{1}{D}\sum_{i=1}^{D}p_i[y_i-\log(\sum_k^C e^{y_k})]$	nr	Adam	✓ §	✓ $^{*\,\circ}$	CS	C	ImageNet*, CIFAR10*
Wan et al. [190]	Quantum Feedforward Networks	$\sum_{i=1}^{3}\sum_{j\in\{q_{\text{out}}\}}\eta_{ij}\left(\langle\hat{\sigma}_i^{(j)}\rangle_{\text{out}}-\langle\hat{\sigma}_i^{(j)}\rangle_{\text{true}}\right)^{2\,\ddagger}$	nr	Gradient Descent	✓	✗	QI	Q	
Chen et al. [46]	Quantum State Discrimination	$J(P_{\text{succ}}(\lvert\psi_0\rangle),P_{\text{err}}(\lvert\psi_0\rangle),P_{\text{inc}}(\lvert\psi_0\rangle))^{\bullet}$	$O(n^k)$	Adam	✓	✗	Inf. Theory	Q	
Romero et al. [152]	QAE for quantum data compression	$\sum_{i=0}^{S-1}p_i\cdot F\left(\text{Tr}_A\left[U_\theta\lvert x_i\rangle\langle x_i\rvert_{AB}U_\theta^\dagger\right],\lvert\phi\rangle_B\right)^{\S}$	$O(n^2)$	L-BFGS-B, Basin-Hopping	✓ †	✗	Chem.	Q	

The symbol "nr" stands for "not reported" meaning that the authors did not explicit reported that specific information or that the specific attribute cannot be applied to the referred work.

Loss Function: D: training dataset size, B: batch size, C: number of the output classes, Θ: QNN parameters, $|\psi_0\rangle$: initial state of the quantum system, x_i: ith classical training data, $|x_i\rangle$: ith quantum training data, y_i: the ground truth label of the ith training data, $|\psi_i^{\text{true}}\rangle$ is the ground truth label state of the ith training data, $y_{\text{out}}(z;\Theta)$: the expected QNN output value for the input z (z may be a quantum input $|x_i\rangle$ or a classical input x_i depending on the considered case), $\star$ average fidelity evaluated over a batch of size B, $\dagger$ $\mathbf{y}_i=[(y_i)_1,\ldots,(y_i)_C]$ is the one-hot encoding of the ground truth label of the ith training data and $g(x_i;\Theta)\in\mathbb{R}^C$ is the probability distribution of the QNN output for the ith classical training data, $\ddagger$ j runs on the output qubits only, $\bullet$ $P_{\text{succ}}(|\psi_0\rangle)$, $P_{\text{err}}(|\psi_0\rangle)$, and $P_{\text{inc}}(|\psi_0\rangle)$ are the probabilities of correct/erroneous/inconclusive measurement outcome,§ sum runs over an ensemble S of input states.

Implementation - Sim.: Numerical Simulation. † QuTiP library, †† QxBranch library, § PennyLane library, ‡ QASM library. **Implementation - QPU.:** • IBM Q 5 "Vigo", ◦ IBM Q Paris, ∗ IBM Q 4, ⋄ Rigetti Q 4.

Application Field: Quantum Information (QI), Chemistry (Chem.), Computer Science (CS), Physics (Phys.), Information Theory (Inf. Theory). **Input Type:** Quantum (Q). Classical (C).

Test Datasets: We reported only real-world and publicly available datasets (synthetic datasets are omitted in the table), ∗ a subset of the dataset is used.

+ We only considered the Quantum Graph Recurrent Neural Networks.

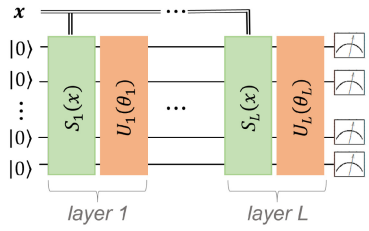
frequencies is determined only by the data-encoding blocks of the circuit. In particular, the available frequencies are a combination of the eigenvalues of the Hamiltonians $\{H_j\}$ defining the data-encoding blocks. Moreover, the size of the spectrum, defined as the number of independent non-zero frequencies, increases linearly with the number of data encoding repetitions: The more times the original data is loaded, the richer the frequency spectrum will be. Finally, note that there are mainly two ways of repeating the data encoding: in series (by uploading several times the inputs while quantum computations proceed, as discussed above) or in parallel (by repeating the encoding on different subsystems at the same layer, so the depth of the circuit will be less but more qubits will be used) [166]. Connections between Fourier series and PQC models with repeated data encoding have also been established by Gil Vidal and Theis [76].

Similar results on the expressivity of PQCs were reported by Perez-Salinas et al. [141], who showed that it is possible to use a single-qubit quantum circuit to implement a universal quantum classifier. The **Universal Approximation Theorem (UAT)** [95] represents a milestone in the context of machine learning. Practically speaking, it ensures that a neural network comprising a single hidden layer with enough units can approximate any continuous function. Interestingly, Perez-Salinas et al. [141] presented an equivalence between UAT and the data re-uploading strategy. In their framework, data are uploaded by single-qubit rotations followed by another set of unitaries representing the evolution of the quantum state. These two operations can be compressed into one by defining quantum layers with tunable parameters as: $\hat{\mathcal{L}}(i) = \hat{U}(\boldsymbol{\theta}_i + \mathbf{w}_i^T \mathbf{x})$, where $\mathbf{x}$ represents the input data, $\mathbf{w}_i$ are weights that play a similar role as in artificial neural networks, while $\boldsymbol{\theta}_i$ are a set of variational parameters. Although, at first sight, it seems that there is no non-linearity in such a construction, the authors reported that they come directly from the structure of the used gates. The output state generated by the circuit is $\hat{\mathcal{U}}(\mathbf{x};\Theta)|\psi_0\rangle$, where $\hat{\mathcal{U}}(\mathbf{x};\Theta) = \prod_{i=1}^{L} \hat{\mathcal{L}}(i)$ and $\Theta = \{\boldsymbol{\theta}_i, \mathbf{w}_i\}_i$ is the set of all circuit parameters. To train their classifier, the authors employed the L-BFGS-B optimization algorithm and used the average fidelity among the output state and the expected one, $|\psi^{\text{true}}\rangle$.

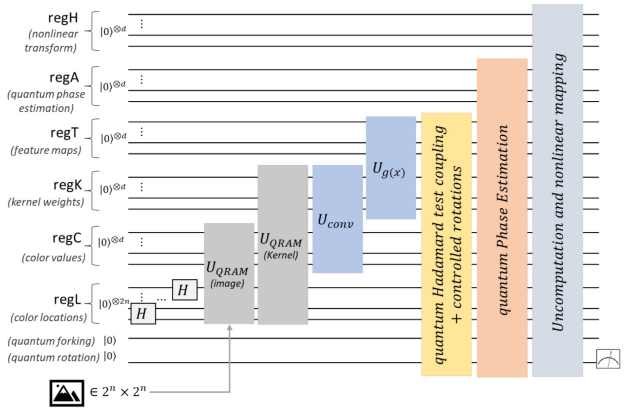
In 2018, Schuld et al. [161] proposed a low-depth parametrized quantum circuit acting as a binary classifier. Their approach is particularly relevant, since it was among the first ones to apply a quantum ansatz to solve machine learning problems using variational circuits. The proposed circuit uses a data-encoding block $\hat{S}(\mathbf{x})$, based on the amplitude encoding technique, to embed classical input data into the state of a quantum system, which is then evolved by a trainable ansatz $\hat{U}(\boldsymbol{\theta})$. The circuit ansatz accounted for a set of single- and multi-controlled qubit gates divided into the so-called "cyclic codes." Once the initial state is evolved, the first qubit is measured, and the probability of finding it in the state $|1\rangle$ is used to infer the input label. Specifically, such a probability is given by:

$$p(q_0 = 1, \mathbf{x}; \boldsymbol{\theta}) = \sum_{k=2^{d-1}+1}^{2^d} \left| \left(\hat{U}(\boldsymbol{\theta}) |\phi(x)\rangle \right)_k \right|^2, \tag{12}$$

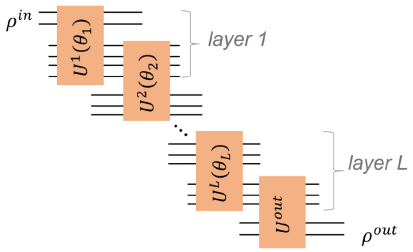
where $|\phi(x)\rangle = \hat{S}(\mathbf{x})|\mathbf{0}\rangle$ is the state prepared by the data-encoding block, and $(\hat{U}(\boldsymbol{\theta})|\phi(x)\rangle)_k$ is the kth entry of the result after the application of the operator $\hat{U}(\boldsymbol{\theta})$ to $|\phi(x)\rangle$. The sum over the second half of the resulting vector corresponds to the single-qubit measurement resulting in state 1. Postprocessing adds the bias b and thresholds to compute a binary prediction: $f(x;\boldsymbol{\theta}, b) = p(q_0 = 1, x; \boldsymbol{\theta}) + b$. To estimate such a probability, the circuit must be run several times. The overall operation is equivalent to evaluate the expectation value of the $\hat{\sigma}_z$ operator acting on the first qubit of the evolved state: $\mathbb{E}(\hat{\sigma}_z) = \langle \phi(x)|\hat{U}(\boldsymbol{\theta})^\dagger (\hat{\sigma}_z \otimes \mathbb{I} \otimes \ldots \otimes \mathbb{I}) \hat{U}(\boldsymbol{\theta})|\phi(x)\rangle$. To validate their ansatz, the authors' employed numerical simulation on several UCI datasets, namely, CANCER, SONAR, WINE, and SEMEION, and on the MNIST dataset. While the first two are binary classification



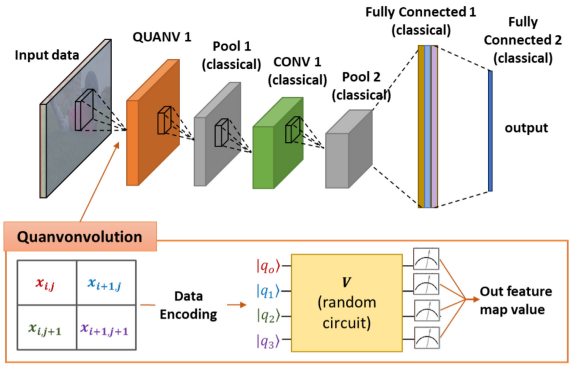
(a) PQC as a universal function approximator [141, 166]



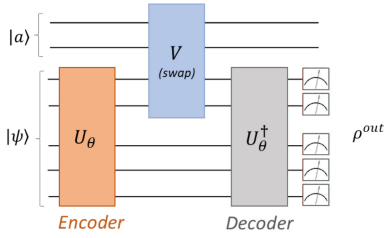
(b) QDCNN (direct quantum implementation of a CNN) [119]



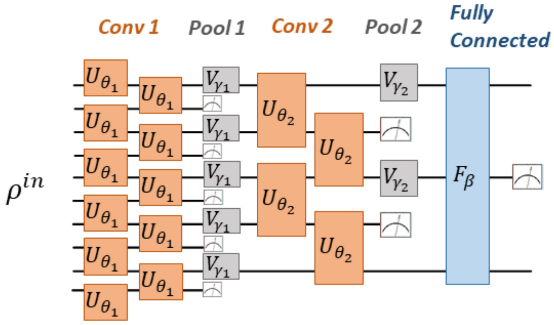
(c) Quantum Dissipative Multilayer Perceptron [20]



(d) Example of a Quanvolutional Neural Network (QVNN) [92]



(e) Quantum AutoEncoder (QAE)[152]



(f) Quantum Convolutional Neural Network (QCNN) [52]

Fig. 4. Schematic circuit representations of some representative QNNs.

problems, the last three are multiclass problems. In such last cases, the authors employed the "one-vs.-all" approach, thus allowing them to use their binary classifier in these cases, too.

Building on top of Reference [161], Macaluso et al. [123] are among the first ones to propose a learning framework for a **quantum Single Layer Perceptron (qSLP)** that is implemented on an actual quantum device, representing the quantum analogous of a single layer perceptron

containing two neurons. Thus, resembling the first attempt at a quantum neural network. Specifically, the circuit computes linear operations in superposition, each characterized by a different set of parameters. The approach is based on three quantum registers: control, data, and temporary. At the end of the computation, the second one stores the results of the two linear operations in superposition. The final response of the qSLP can be obtained as the expected value of $\hat{\sigma}_z$ acting on the data register. Notably, a non-linear activation function is not implemented in the quantum circuit in such a work. Instead, it is part of post-processing operations. Finally, to train the qSLP, the authors used the **Sum of Squared Errors (SSE)** loss and the Nesterov [156] accelerated gradient method.

Instead, Beer et al. [20] proposed a quantum MLP, also referred to as q*uantum dissipative neural network*. In their model, each quantum perceptron is formulated as a unitary operator acting on n input and m output qubits. The idea was to stack this kind of perceptrons to form "layers" that emulated the behavior of a classical deep model. Each layer interacts with the following one before being dissipated (discarded); the interaction is implemented by coupling the qubits in the ith layer to some ancillary qubits in the $(i+1)$-layer (Figure 4(c)). The overall QNN can be then described as follows: $\hat{\mathcal{U}} = \prod_l^L \hat{U}^l$, where $\hat{U}^l = \prod_i \hat{U}_i^l$ represents the set of unitaries of layer l, i.e., the QPs of that specific layer. The QNN is applied to the input state, ρ^{in} and produces an output state, ρ^{out}. To assess the performance of their approach, the authors used numerical simulations with a QNN concerning input and output spaces of 2 and 3 qubits. To train their model, they employed a gradient descent technique, while as a loss function, they employed the fidelity among the output state and the desired (ground-truth) state averaged over the training data.

Graph Neural Networks [179] were introduced to exploit the geometrical structure in the input data and accounted for the application of neural networks to acyclic graphs. Based on a similar principle, a general PQC was proposed by Verdon et al. [187], named ***Quantum Graph Neural Networks*** **(QGNN)**, consisting of a sequence of Q different Hamiltonian evolutions repeated P times:

$$\hat{U}(\boldsymbol{\eta}, \boldsymbol{\theta}) = \prod_p^P \left[\prod_q^Q e^{-i\eta_{pq}\hat{H}_q(\boldsymbol{\theta})} \right], \tag{13}$$

where the $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ are the sets of variational parameters, and the $\hat{H}_q(\boldsymbol{\theta})$ are the Hamiltonians encoding the problem to be solved. The authors proposed different variants for the general QGNN ansatz concerning the following tasks: learning Hamiltonian dynamics of quantum systems, learning how to create multipartite entanglement in a quantum network, unsupervised learning for spectral clustering, and supervised learning for graph isomorphism classification. Concerning the first task, they specialized the ansatz, naming it **Quantum Graph Recurrent Neural Network (QGRNN)** by tying the temporal parameters between iterations ($\eta_{pq} \rightarrow \eta_q$), thus resembling the classical sequence networks in which the parameters are shared among the various timesteps of the representation evolution. Given a target Hamiltonian $\hat{H}_{\text{target}}$ the authors train the model parameters by using the Adam [111] optimizer to minimize the average infidelity:

$$1 - \frac{1}{B} \sum_{j=1}^{B} |\langle \psi_{T_j} | \hat{U}^j(\Delta, \theta) | \psi_0 \rangle|^2, \tag{14}$$

where the $|\psi_{T_j}\rangle = e^{-iT_j\hat{H}_{\text{target}}}|\psi_0\rangle$ is the state evolved from $|\psi_0\rangle$ for a time T_j, $\hat{U}^j(\Delta, \theta)|\psi_0\rangle$ is the state obtained by evolving $|\psi_0\rangle$ according to the QGRNN for $P \sim T_j/\Delta$ iterations with Δ being a hyperparameter determining the Trotter step size, and the average is evaluated over batch sizes of B different times T_j.

5.4.2 CNN-inspired Quantum Neural Networks. **Convolutional Neural Networks (CNNs)** are feedforward networks particularly suitable for data that have a grid-like topology such as images or time-series data. This class of models owes its name to the use of convolutional operations in at least one of their layers. A convolutional layer takes a tensor as input and processes it using a set of trainable kernels (filters) that slide along the input tensor, computing the inner product between the filter and the input entries, producing as output the so-called feature map. Each element of the output is then transformed by applying a non-linear activation function. Finally, a pooling operation is typically used to replace portions of the feature map with a summary statistic (e.g., max, average, sum), which makes it more manageable in the successive layers (since the dimension is reduced), the network more resilient to small input changes, and helps to avoid overfitting issues. Convolutional layers can then be followed by fully connected layers where each perceptron receives information from all the units of the previous layer. This latter typology of a layer is also used to produce the output of the network, e.g., a vector containing the probabilities that the input data belong to certain classes for classification tasks or real values for regression tasks.

Given the success of deep CNNs on the task of pattern recognition [113, 114], many researchers proposed quantum algorithms inspired by these models. In References [106, 119], the authors proposed quantum models, running on an actual quantum hardware, capable of performing all the classical CNN operations. The main limitation of these approaches is that they usually require a QRAM [77] to interface classical data and quantum states and may suffer from classical data encoding bottleneck. Moreover, the overall algorithm may require subroutines that are not easy to implement with many qubits (e.g., quantum phase estimation). For example, Li et al. [119] proposed the ***Quantum Deep Convolutional Neural Network*** **(QDCNN)**, based on a PQC, whose architecture consists of several successive quantum convolutional layers and a quantum classifier layer (Figure 4(b)). Their model does not implement a quantum version for the pooling layer, however, the dimension of the generated feature maps is reduced by preserving some qubits in the location register and disentangling the other ones. Quantum measurement is performed to output a category prediction, and a hybrid quantum-classical training algorithm is used to optimize the parameters of the circuit. The classical input (e.g., an image) and the convolutional kernels are prepared using a QRAM. Similarly, Kerenidis et al. [106] proposed to vectorize the convolution operations in a CNN to exploit linear-algebra subroutines to carry out the calculation exploiting quantum phenomena. To their aim, the authors proposed algorithms both for the forward and backward passes while training a QNN. Data and convolutional kernels are stored into QRAMs registers that are resumed when needed.

Henderson et al. [92] proposed a hybrid classical-quantum approach to empower a classical CNN with layers containing filters implemented as quantum circuits, called quanvolutional layers, that can be stacked on top of any layer of a traditional CNN (see Figure 4(d)). The resulting architecture was named *Quanvolutional Neural Network* (QVNN). Such a proposal stemmed from the observation that, on the one side, machine learning algorithms benefit from random non-linear features in terms of accuracy and training time, while on the other, quantum circuits can model complex functional relationships. Each quanvolutional layer is characterized by a certain number of filters, designed as a sequence of single- or single-controlled qubit gates, and produces feature maps by locally transforming input data. Specifically, each circuit is built by considering each qubit as a node in a graph and assigning what the authors named a "connection probability" between each pair of qubits. A key difference of such a formulation of QVNN compared to others is the lack of a variational tuning of the circuits' parameters (they use random quantum circuits in the quanvolutional filters). Interestingly, such an architecture does not require a QRAM, thus reducing the overhead due to its usage, and it is compatible and directly integrable with existing classical architectures. QVNNs were tested against the MNIST dataset using the QxBranch Quantum Computer

Simulation System, and, unfortunately, the authors did not find any advantage over the classical counterpart. Indeed, the performance of the QVNN was indistinguishable from that of a model using a classical random non-linear transformation instead of the quanvolutional layer.

In Cong et al. [52] a variational ansatz, named ***Quantum Convolutional Neural Network* (QCNN)**, is proposed for analyzing quantum data (Figure 4(f)). As typically happens for variational circuits, hyperparameters such as the number of gates are fixed, while the parameters characterizing them are learned by classical optimization. The similarity between CNN and QCNN stems from the interpretation given to each "layer" of the last one. However, it is fundamental not to confuse the concept of layer between the classical and quantum architectures. Indeed, concerning the last one, a layer is represented by a unitary operator applied to the data register. With such a difference in mind, one can think of the QCNN structure as a sequence of layers as follows: The first layer applies a single quasilocal unitary, $\hat{U}_\theta$, in a translationally invariant manner followed by a measurement on a subset of the input qubits, which resembles the pooling operation. The outcome of such an operation determines the unitary operation, V_γ, applied to the nearby qubits. These two operations are applied a few times until the system is small enough. A final unitary, F_β, is then applied to emulate the action of a fully connected layer. Finally, the outcome of the QCNN is obtained by measuring a specific set of output qubits. Such a model was applied successfully to solve the problem of Quantum Phase Recognition, which aims at recognizing whether a given input quantum state ρ_i belongs to a particular quantum phase of matter. Given a QCNN, the training set comprised training vectors $\{(|\psi_i\rangle, y_i)|i = 1, \ldots, D\}$, where the binary labels y_i identified a given input state vector. As objective, the authors exploited the mean squared error between the label y_i and the QCNN output. Thus, the learning process consisted of initializing all the unitaries and then optimizing them until convergence, for example, via gradient descent.

In Grant et al. [78], the authors exploited the tensor networks' hierarchical structure to represent quantum circuits.[12] Specifically, they employed tree tensor networks [170] and the **multi-scale entanglement renormalization ansatz (MERA)** [188] to implement a quantum classifier subsequently tested against the Iris, MNIST, and synthetic quantum datasets. The classifiers were implemented on the ibmqx4 quantum computer and trained using the mean squared error loss evaluated between the circuit's output and the expected one. Specifically, concerning the first quantity, it represented the outcome of a measurement on the target qubit: $\langle \hat{M}_\theta(|\psi\rangle)\rangle = \langle\psi|\hat{U}_\theta^\dagger \hat{M}\hat{U}_\theta|\psi\rangle$. To minimize the objective function, the authors employed the Adam [111] optimizer. As an example of hierarchical structure, the tree tensor networks-inspired circuit begins by applying a set of two-qubit unitaries among the nearest qubits. After each operation, one qubit is discarded, thus halving the total number of qubits to the next layer. After a sequence of such unitaries, the state of the final remaining qubit is measured.

5.4.3 Hybrid Approaches for Different Learning Paradigms. Jaderberg et al. [97] proposed a VQA to solve a classification task by employing a self-supervised training approach [55]. They formulated the ansatz to be part of a hybrid encoder network tested against the classification task on five classes of the CIFAR10 dataset. To train the model, the authors exploited the contrastive learning procedure [89, 136] applied to the representations generated by their models. As the objective function, they used the normalized temperature-scaled cross-entropy loss [177]. As mentioned, the QNN was part of a hybrid network whose first part was classical. Specifically, the overall architecture began with a ResNet-18 [90] followed by a single-layer convolutional network that

[12]This method can be interpreted as a CNN-inspired model, since tensor networks with hierarchical structure exhibit many similarities with neural networks and in some cases have been shown to be equivalent to CCNs with rectified linear activation [78]

compressed the features representation down to an eight-dimensional vector, $\mathbf{v}$, such that it was encoded in an entangled state $|\psi_{\mathbf{v}}\rangle$ comprising $n = 8$ qubits. The quantum data encoding is achieved by applying a single qubit rotation $\hat{R}_x$ (as generated by the $\hat{\sigma}_x$ operator) to each qubit of the register, i.e., $|\psi_{\mathbf{v}}\rangle = \otimes_i^n \hat{R}_x(v_i)|\mathbf{0}\rangle$, where v_i is the ith element of the n-dimensional features vector $\mathbf{v}$. The state is evolved using an ansatz made by parametrized controlled qubit rotations. Finally, to get the output from the QNN, the authors measured the qubits state in the computational basis. The authors tested their approach on quantum state vector simulations and real quantum devices. In both cases, they observed that the hybrid architecture reported performance similar or higher than the full classical implementation, thus giving an interesting hint about the capability of such hybrid models applied to computer vision-related tasks.

Transfer learning is a widely used technique in deep learning that exploits pre-trained neural models on a different task. Interestingly, such an approach can be "hybridized" by combining classical and quantum algorithms. Indeed, Mari et al. [126] considered three different scenarios for transfer learning, namely, classical-to-quantum, quantum-to-classical, and quantum-to-quantum. In what follows, we focus on the classical-to-quantum configuration. In such a case, a ResNet-18 [90] network trained on the ImageNet dataset [57] was used as a features extractor for the quantum circuit. The parametrized quantum circuit was interpreted as made by three components: $Q = \mathcal{M} \circ \mathcal{Q} \circ \mathcal{E}$. The first component, $\mathcal{E}$, represents the quantum embedding of the classical features vector into the Hilbert space depending on $\mathbf{x}$. The second one, $\mathcal{Q}$, is a variational circuit of a given depth composing single and two-qubit gates, while the $\mathcal{M}$ term represents the measurement on the circuit's output that maps a quantum state to a classical vector. The authors trained such a circuit to classify images of bees and ants utilizing the Adam [111] optimizer and the cross-entropy loss. The authors assessed the performance of their models through numerical simulations by using the PennyLane framework and then implemented the quantum circuit on the ibmqx4 and the Aspen-4-4Q-A quantum processors from IBM and Rigetti, respectively. The authors' results demonstrated that there was already the possibility to investigate efficient algorithms to process high-resolution images in the NISQ era [146] concerning the classical-to-quantum transfer learning scheme.

5.4.4 Other Approaches. Differently from the variational approach, a quantum generalization of the classical feedforward neural networks was proposed by Wan et al. [190]. The quantum neural network accounted for a reformulation of the perceptron model by making each transformation reversible and unitary. Although different from VQA, such networks have shown, by employing numerical simulation, to be trainable using gradient descent for a given objective function. In such a case, the goal was to minimize the difference between the quantum circuit's output and the expected one. A viable way to physically realize such an architecture was employing quantum photonics. For example, to prove the effectiveness of their approach, the authors proposed a quantum autoencoder to compress two quantum states with high accuracy.

Quantum State Discrimination (QSD) [19] requires the design of measurements to optimally identify the ρ that represents the unknown state of a quantum device that is believed to belong to a non-orthogonal set of states $\{\rho_i\}$. Such a capability has a broad range of applications. For example, quantum state discrimination by itself plays a crucial role in quantum information processing protocols and is used in quantum cryptography [24], quantum cloning [63], quantum state separation, and entanglement concentration [19]. Typically, distinguishing among non-orthogonal states is challenging, and specialized measurements are required. An exciting approach for solving such a problem was proposed by Chen et al. [46], who formalized a framework to learn to simulate the unknown structure of the generalized quantum measurement, or **Positive-Operator-Value-Measure (POVM)** [134]. The circuit designed by the authors contained only a single-qubit and CNOT gates, and the proper non-linearities were introduced by measuring qubits states.

Specifically, they assessed the performance of their circuits by employing numerical simulation on classical machines and by using the Adam [111] optimizer to learn the circuit's parameters. However, differently from the previously cited variational approaches, in the current case, the authors aimed at maximizing their circuit's generalization performance given a specific range of the parameters rather than looking through all the available space.

Classical AutoEncoders are tasked with data reduction, i.e., given an $(n + k)$-bit representation, the encoder generates a compressed n-bit datum with the same discrimination power of the original input. Romero et al. [152] proposed a ***Quantum AutoEncoder*** **(QAE)** as a solution to the compression problem concerning quantum states represented by $(n + k)$-qubits. Specifically, they were meant to facilitate quantum data compression, especially concerning quantum simulations. Such models should recognize patterns due to quantum phenomena, such as superposition and entanglement, not accessible to classical algorithms. The authors tested their architecture to compress ground states of the Hubbard model and molecular Hamiltonians. The task of a QAE was then to get rid of a certain number of qubits, say, k, while maintaining the same amount of quantum information in the remaining ones. Thus, the QAE must entangle qubits, and for such an aim, the authors exploited unitary gates to build the variational circuit (Figure 4(e)). As a cost function, QAE leverages the expected fidelity [134], $F(|\psi_i\rangle, \rho_i^{out})$, which quantifies the deviation from the initial state $|\psi_i\rangle$ to the output one ρ_i^{out}. The authors used a hybrid approach to train the model in which the quantum device took care of the state preparation and the measurement while the optimization procedure was carried out through classical techniques. The authors exploited two different optimizers to train the circuit, namely, L-BFGS-B and Basin-Hopping. After training, the authors proved that QAE could compress the ground state of the two-sites Hubbard model from 4 qubits down to 1 with an error of below 10^{-4}. It is relevant to notice that such a compression allows moving from a space of size 2^4 to one of size 2.

Finally, it is worth mentioning that there exist approaches based on an optimization procedure different from the variational approaches we have seen so far. For example, Silva et al. [53, 173] proposed a learning algorithm in which all the input patterns were presented concurrently in superposition to the neural networks. Their model was based on a quantum version of the RAM-based neural networks [12], termed **quantum-RAM (q-RAM)** node [54], a weightless network in which the learning procedure consisted upon simply writing the proper output value in the corresponding look-up table entries of the q-RAM node. The model was designed as a quantum circuit based on the gate model paradigm. Moreover, the authors commented on the possibility of stacking mode nodes to form a deeper network. A fundamental step in the training procedure accounted for Grover's algorithm [80] applied to the input until the desired output was returned.

5.5 Trainability Issues and Barren Plateau

As discussed in the previous section, the most common approach for implementing a QNN is using a PQC characterized by a set of parameters $\boldsymbol{\theta}$ that are typically learned by optimizing a cost function $C(\boldsymbol{\theta})$ using a classical optimization algorithm. One of the main issues with the trainability of a QNN is that the gradient of the cost function with respect to $\boldsymbol{\theta}$ may exponentially vanish during the training as a consequence of the progressive flattening of the cost function's landscape—a phenomenon referred to as *barren plateaus*. This phenomenon is particularly present when using a random initialization for "deep" circuits, or more precisely, for circuits characterized by a number of layers that depend polynomially on the number n of qubits [128]. Indeed, McClean et al. [128] stated that for a wide class of PQCs with a sufficient depth and number of qubits, the probability that the gradient along any reasonable direction is non-zero to some fixed precision is exponentially small as a function of the number of qubits. In other words, the expected value of $\frac{\partial C(\boldsymbol{\theta})}{\partial \theta_i}$ is

zero and its variance decreases exponentially with n. Unfortunately, the region where the gradient is zero does not correspond to local minima of interest. Instead, it corresponds to a large plateau of states for which no interesting search direction can be pursued to exiting the barren plateau, or more precisely, an exponential precision is required to determine a minimizing direction to navigate the cost landscape. The use of a shallow circuit does not solve the problem *a priori*, because as long as a *global* cost function is used, i.e., a measure obtained using an observable that involves all the qubits of the circuits, the barren plateau can occur in any layer of the circuit [128]. This is one of the main differences between the gradient vanishing problem for classical DNNs versus QNNs: In the case of classical DNNs, the gradient can vanish exponentially in the number of layers, while for QNNs, the gradient can vanish exponentially in the number of qubits [128].

Cerezo et al. [45] showed that a promising approach to overcome the problem of exponentially vanishing gradients is to use shallow circuits with *local* cost functions, which are constructed by observing only a limited set of qubits at a time. Indeed, they demonstrated that local cost functions lead, at worst, to a polynomially vanishing gradient if the circuit depth is $O(\log n)$, with n being the number of qubits, and therefore the circuit will be trainable with a polynomial scaling with the system size (i.e., a polynomial number of shots per optimization step are needed to estimate the gradient). However, note that using local cost functions alone is not sufficient to avoid the problem: Deep networks with local observables can still give rise to barren plateaus.

Interestingly, some QNNs have not exhibited barren plateau in the cost function landscape, such as dissipative quantum neural networks with shallow local perceptrons [20, 168] and Quantum Convolutional Neural Networks [52, 78, 143]. Moreover, several strategies have been proposed for mitigating barren plateaus, including the use of local cost function [45], specific parameter initialization/pre-training [79, 186], clever problem-inspired ansatz [28, 83], layerwise learning [175], and parameter correlation [94, 189], just to name a few.

All the methods mentioned so far concern the problem of noise-free barren plateaus, since they do not consider quantum hardware noise. However, a different kind of barren plateau can occur in the presence of noise. Wang et al. [192] provided an analytical study on noise-induced barren plateaus for a generic variational ansatz. They proved that in the presence of local noise, the magnitude of the gradient decays exponentially with the depth of the circuit. Therefore, while noise-free barren plateaus are mainly connected to the structure of the ansatz (number of qubits, parameter initialization, locality of the cost function), the noise-induced barren plateaus are mainly affected by the ansatz depth. Moreover, for noise-induced barren plateaus, the magnitude of the cost function's minimum is also flattening, in addition to its gradient. Due to the diverse nature of noise-induced barren plateaus, strategies to mitigate noise-free barren plateaus, e.g., References [45, 79, 175, 189], do not solve the problem, and the strategies to adopt should rather rely on simplifying the complexity of the circuit by reducing its depth or reducing the hardware noise level.

5.6 Final Remarks

As we have seen, PQC currently represents the most adopted approach to QNNs. Perhaps, their capability to allow implementing learning circuits on NISQ devices and the similarities between the ansatz and the architecture of classical neural networks are two of the reasons for the variational algorithms' success. However, other research lines try to exploit intrinsic properties of quantum systems without the necessity of emulating a classical procedure, such as in the work of Reference [194]. Hence, as we have seen so far, the research for QNNs that one day will be applied to real-world problems represents a very exciting and active field.

Unlike the literature on classical neural networks, identifying state-of-the-art QNNs for solving a given task is not easy. This issue is mainly due to the lack of benchmark procedures for comparing these algorithms. Many approaches have only been tested on synthetic data generated by authors,

Table 4. Performance Comparison between Some QNNs on the MNIST Dataset

Reference	Impl.	Input dim (after preprocessing)	# Samples	# Classes	Accuracy (%)
Schuld et al. [161]*	Simulation	256 real values	2,766 (train/test split not reported)	10	67.00
Li et al. [119]	Simulation	28 × 28 (784 real values)	60,000 train/10,000 test	10	98.97
Henderson et al. [92]°	Simulation	28 × 28 (784 real values)	60,000 train/10,000 test	10	~98.00
Grant et al. [78]⋆	QPU (IBM Q4)	8 real values	60,000 train/10,000 test	2 (is >4 or <4)	79.10 ± 0.90
				2 (is even or odd)	84.85 ± 0.20
				2 (is 0 or 1)	99.87 ± 0.02
				2 (is 2 or 7)	98.86 ± 0.07

The original MNIST dataset contains 70,000 (60,000 training and 10,000 test) 28-by-28 greyscale pixel images of handwritten digits (10 classes). The test accuracy of the QNNs is reported from the original papers.
*Multilabel classification problems casted as a set of "one-versus-all" binary classification subtasks; fivefold cross-validation with one repetition was carried out.
°Approximate accuracy estimated from the graph of test accuracy versus training iterations available in the original paper.
⋆Four distinct binary classification tasks were considered in the paper.

and only a few have publicly released their code. Furthermore, the use of different implementations (simulations or different QPUs) makes a direct comparison among the various approaches very difficult. Even when the same benchmark dataset is used (see, for example, the MINST dataset for handwritten recognition in Table 4), it is not easy to compare the performance of the various methods, because there is a variety in the number of samples used, the classes or the specific tasks considered.

Due to the limitations of current quantum platforms, several works have remained only theoretical, waiting for QPUs to become powerful enough and less noisy to allow the exploitation of a large number of entangled units. Indeed, the dragon of decoherence is always on the hunt and, unfortunately, qubits easily fall prey, thus strongly limiting the number of computations that can be performed. Perhaps, future quantum devices will open the frontiers to a new generation of neural networks based on quantum phenomena.

6 CONCLUSIONS

Artificial intelligence has played a central role in academic and industrial debates in the past couple of decades, especially concerning machine learning and neural network techniques. Although these algorithms have shown an incredible generalization capability when adequately trained to solve a given problem, how and why these algorithms behave as they do is a topic that still dazzles scientists worldwide. Nevertheless, notwithstanding astonishing results, these algorithms will always and irremediably be tied to the classical interpretation of the real world.

Differently, quantum computations might overcome such a limit and exploit phenomena such as superposition and entanglement that belong to the quantum realm. Although still at their dawn, quantum technologies represent a promising and fascinating alternative to classical computing techniques, perhaps realizing practical quantum supremacy soon.

Stemming from these considerations, we conceived this survey to offer both the neophyte and the more experienced reader insights into several fundamental topics in the quantum computation field such as the qubit, the Gate Model, and the Adiabatic Quantum Computation paradigms, to mention some. Moreover, noticing that the literature lacks a detailed discussion about the latest achievements concerning Quantum Perceptrons and Quantum Neural Networks, we gather,

analyze, and discuss state-of-the-art approaches related to these topics. From our work, it is clear that quantum neural networks and algorithms, in general, are still far from proving decisive supremacy over the classical ones. Too often, such a goal has been claimed. Nevertheless, it is still an open quest. It is not clear if quantum devices will replace classical chips, becoming the core of a new generation of personal computers. However, recent improvements for quantum hardware and algorithms have opened the gates towards a new world, the quantum world, characterized by phenomena that were completely unknown until the last century.

REFERENCES

[1] Scott Aaronson and Alex Arkhipov. 2011. The computational complexity of linear optics. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*. Association for Computing Machinery, 333–342.

[2] Farid Ablayev, Marat Ablayev, Joshua Zhexue Huang, Kamil Khadiev, Nailya Salikhova, and Dingming Wu. 2020. On quantum methods for machine learning problems part I: Quantum tools. *Big Data Mining. Anal.* 3, 1 (2020), 41–55.

[3] Farid Ablayev, Marat Ablayev, Joshua Zhexue Huang, Kamil Khadiev, Nailya Salikhova, and Dingming Wu. 2020. On quantum methods for machine learning problems part II: Quantum classification algorithms. *Big Data Mining Anal.* 3, 1 (2020), 56–67.

[4] Daniel S. Abrams and Seth Lloyd. 1999. Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.* 83, 24 (1999), 5162.

[5] Jeremy Adcock, Euan Allen, Matthew Day, Stefan Frick, Janna Hinchliff, Mack Johnson, Sam Morley-Short, Sam Pallister, Alasdair Price, and Stasja Stanisic. 2015. Advances in quantum machine learning. *arXiv preprint arXiv:1512.02900* (2015).

[6] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. 2004. Adiabatic computation is equivalent to standard quantum computation. *arXiv preprint quant-ph/0405098* (2004).

[7] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. 2008. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Rev.* 50, 4 (2008), 755–787.

[8] Esma Aïmeur, Gilles Brassard, and Sébastien Gambs. 2006. Machine learning in a quantum world. In *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 431–442.

[9] Esma Aïmeur, Gilles Brassard, and Sébastien Gambs. 2013. Quantum speed-up for unsupervised learning. *Mach. Learn.* 90, 2 (2013), 261–287.

[10] Akshay Ajagekar and Fengqi You. 2019. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy* 179 (2019), 76–89.

[11] Tameem Albash and Daniel A. Lidar. 2018. Adiabatic quantum computation. *Rev. Mod. Phys.* 90, 1 (2018), 015002.

[12] Igor Aleksander. 1966. Self-adaptive universal logic circuits. *Electron. Lett.* 2, 8 (1966), 321–322.

[13] Jonathan Allcock and Shengyu Zhang. 2019. Quantum machine learning. *Natl. Sci. Rev.* 6, 1 (2019), 26–28.

[14] M. V. Altaisky. 2001. Quantum neural network. *arXiv preprint quant-ph/0107012* (2001).

[15] Srinivasan Arunachalam and Ronald de Wolf. 2017. Guest column: A survey of quantum learning theory. *ACM SIGACT News* 48, 2 (2017), 41–67.

[16] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, et al. 2019. Quantum supremacy using a programmable superconducting processor. *Nature* 574, 7779 (2019), 505–510.

[17] Francisco Barahona. 1982. On the computational complexity of Ising spin glass models. *J. Phys. A: Math. Gen.* 15, 10 (1982), 3241.

[18] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. 1995. Elementary gates for quantum computation. *Phys. Rev. A* 52, 5 (1995), 3457.

[19] Stephen M. Barnett and Sarah Croke. 2009. Quantum state discrimination. *Adv. Opt. Photon.* 1, 2 (2009), 238–278.

[20] Kerstin Beer, Dmytro Bondarenko, Terry Farrelly, Tobias J. Osborne, Robert Salzmann, Daniel Scheiermann, and Ramona Wolf. 2020. Training deep quantum neural networks. *Nat. Commun.* 11, 1 (2020), 1–6.

[21] Elizabeth C. Behrman, John Niemel, James E. Steck, and Steve R. Skinner. 1996. A quantum dot neural network. In *Proceedings of the 4th Workshop on Physics of Computation*. 22–24.

[22] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. 2019. Parameterized quantum circuits as machine learning models. *Quant. Sci. Technol.* 4, 4 (2019), 043001.

[23] Paul Benioff. 1980. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *J. Stat. Phys.* 22, 5 (1980), 563–591.

[24] Charles H. Bennett. 1992. Quantum cryptography using any two non-orthogonal states. *Phys. Rev. Lett.* 68, 21 (1992), 3121.
[25] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. 1997. Strengths and weaknesses of quantum computing. *SIAM J. Comput.* 26, 5 (1997), 1510–1523.
[26] A. J. Berkley, M. W. Johnson, P. Bunyk, R. Harris, J. Johansson, T. Lanting, E. Ladizinsky, E. Tolkacheva, M. H. S. Amin, and G. Rose. 2010. A scalable readout system for a superconducting adiabatic quantum optimization system. *Supercond. Sci. Technol.* 23, 10 (2010), 105014.
[27] Ethan Bernstein and Umesh Vazirani. 1997. Quantum complexity theory. *SIAM J. Comput.* 26, 5 (1997), 1411–1473.
[28] Kishor Bharti and Tobias Haug. 2021. Iterative quantum-assisted eigensolver. *Phys. Rev. A* 104, 5 (2021), L050401.
[29] Kishor Bharti, Tobias Haug, Vlatko Vedral, and Leong-Chuan Kwek. 2020. Machine learning meets quantum foundations: A brief survey. *AVS Quant. Sci.* 2, 3 (2020), 034101.
[30] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549, 7671 (2017), 195–202.
[31] James Binney and David Skinner. 2013. *The Physics of Quantum Mechanics*. Oxford University Press.
[32] Alex Bocharov, Martin Roetteler, and Krysta M. Svore. 2015. Efficient synthesis of universal repeat-until-success quantum circuits. *Phys. Rev. Lett.* 114, 8 (2015), 080502.
[33] Sergio Boixo, Sergei V. Isakov, Vadim N. Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J. Bremner, John M. Martinis, and Hartmut Neven. 2018. Characterizing quantum supremacy in near-term devices. *Nat. Phys.* 14, 6 (2018), 595–600.
[34] Roberto Bondesan and Max Welling. 2021. The Hintons in your neural network: A quantum field theory view of deep learning. *arXiv preprint arXiv:2103.04913* (2021).
[35] Max Born and Vladimir Fock. 1928. Beweis des adiabatensatzes. *Zeitschrift für Physik* 51, 3–4 (1928), 165–180.
[36] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. 2019. Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. *arXiv preprint arXiv:1710.02581* (2019).
[37] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. 2002. Quantum amplitude amplification and estimation. *Contemp. Math.* 305 (2002), 53–74.
[38] Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd. 2011. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceed. Roy. Societ. A: Math., Phys. Eng. Sci.* 467, 2126 (2011), 459–472.
[39] Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. 2017. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum* 1 (2017), 8.
[40] Zi Cai and Jinguo Liu. 2018. Approximating quantum many-body wave functions using artificial neural networks. *Phys. Rev. B* 97, 3 (2018), 035116.
[41] Yudong Cao, Gian Giacomo Guerreschi, and Alán Aspuru-Guzik. 2017. Quantum neuron: An elementary building block for machine learning on quantum computers. *arXiv preprint arXiv:1711.11240* (2017).
[42] Giuseppe Carleo and Matthias Troyer. 2017. Solving the quantum many-body problem with artificial neural networks. *Science* 355, 6325 (2017), 602–606.
[43] Juan Carrasquilla and Roger G. Melko. 2017. Machine learning phases of matter. *Nat. Phys.* 13, 5 (2017), 431–434.
[44] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. 2020. Variational quantum algorithms. *arXiv preprint arXiv:2012.09265* (2020).
[45] Marco Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J. Coles. 2021. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.* 12, 1 (2021), 1–12.
[46] Hongxiang Chen, Leonard Wossnig, Simone Severini, Hartmut Neven, and Masoud Mohseni. 2021. Universal discriminative quantum neural networks. *Quant. Mach. Intell.* 3, 1 (2021), 1–11.
[47] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. 2018. Equivalence of restricted Boltzmann machines and tensor network states. *Phys. Rev. B* 97, 8 (2018), 085104.
[48] Jian Chen, Jonathan L. Habif, Zachary Dutton, Richard Lazarus, and Saikat Guha. 2012. Optical codeword demodulation with error rates below the standard quantum limit using a conditional nulling receiver. *Nat. Photon.* 6, 6 (2012), 374–379.
[49] Lawrence W. Cheuk, Matthew A. Nichols, Katherine R. Lawrence, Melih Okan, Hao Zhang, Ehsan Khatami, Nandini Trivedi, Thereza Paiva, Marcos Rigol, and Martin W. Zwierlein. 2016. Observation of spatial charge and spin correlations in the 2D Fermi-Hubbard model. *Science* 353, 6305 (2016), 1260–1264.
[50] Carlo Ciliberto, Mark Herbster, Alessandro Davide Ialongo, Massimiliano Pontil, Andrea Rocchetto, Simone Severini, and Leonard Wossnig. 2018. Quantum machine learning: A classical perspective. *Proc. Roy. Societ. A: Math., Phys. Eng. Sci.* 474, 2209 (2018), 20170551.

[51] J. Ignacio Cirac and Peter Zoller. 2012. Goals and opportunities in quantum simulation. *Nat. Phys.* 8, 4 (2012), 264–266.
[52] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. 2019. Quantum convolutional neural networks. *Nat. Phys.* 15, 12 (2019), 1273–1278.
[53] Adenilton J. Da Silva, Wilson R. De Oliveira, and Teresa B. Ludermir. 2012. Classical and superposed learning for quantum weightless neural networks. *Neurocomputing* 75, 1 (2012), 52–60.
[54] Wilson Rosa de Oliveira. 2009. Quantum RAM based neural networks. In *Proceedings of the European Symposium on Artificial Neural Networks*. Citeseer, 331–336.
[55] Virginia R. de Sa. 1994. Learning classification with unlabeled data. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. Citeseer, 112–119.
[56] Christian L. Degen, F. Reinhard, and Paola Cappellaro. 2017. Quantum sensing. *Rev. Mod. Phys.* 89, 3 (2017), 035002.
[57] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 248–255.
[58] David Deutsch. 1985. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. Roy. Societ. London. A. Math. Phys. Sci.* 400, 1818 (1985), 97–117.
[59] David Deutsch and Richard Jozsa. 1992. Rapid solution of problems by quantum computation. *Proc. Roy. Societ. London. A. Math. Phys. Sci.* 439, 1907 (1992), 553–558.
[60] Paul Adrien Maurice Dirac. 1939. A new notation for quantum mechanics. *Math. Proc. Cambridge Philos. Soc.* 35, 3 (1939), 416–418.
[61] Paul Adrien Maurice Dirac. 1981. *The Principles of Quantum Mechanics*. Clarendon Press, Oxford.
[62] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. 2018. Implementable quantum classifier for nonlinear data. *arXiv preprint arXiv:1809.06056* (2018).
[63] Lu-Ming Duan and Guang-Can Guo. 1998. Probabilistic cloning and identification of linearly independent quantum states. *Phys. Rev. Lett.* 80, 22 (1998), 4999.
[64] Vedran Dunjko and Hans J. Briegel. 2018. Machine learning & artificial intelligence in the quantum domain: A review of recent progress. *Rep. Prog. Phys.* 81, 7 (2018), 074001.
[65] Vedran Dunjko and Peter Wittek. 2020. A non-review of quantum machine learning: Trends and explorations. *Quant. Views* 4 (2020), 32.
[66] Stavros Efthymiou, Sergi Ramos-Calderer, Carlos Bravo-Prieto, Adriian Perez-Salinas, Diego García-Martín, Artur Garcia-Saez, José Ignacio Latorre, and Stefano Carrazza. 2021. Qibo: A framework for quantum simulation with hardware acceleration. *Quant. Sci. Technol.* (2021).
[67] Daniel J. Egger, Claudio Gambella, Jakub Marecek, Scott McFaddin, Martin Mevissen, Rudy Raymond, Andrea Simonetto, Stefan Woerner, and Elena Yndurain. 2020. Quantum computing for finance: State of the art and future prospects. *IEEE Trans. Quant. Eng.* (2020).
[68] Jean Faber and Gilson A. Giraldi. 2002. Quantum models of artificial neural networks. *Retrieved from: http://arquivosweb. lncc. br/pdfs/QNN-Review. pdf.*
[69] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028* (2014).
[70] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. 2001. A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science* 292, 5516 (2001), 472–475.
[71] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. 2000. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106* (2000).
[72] Edward Farhi and Sam Gutmann. 1998. Analog analogue of a digital quantum computation. *Phys. Rev. A* 57, 4 (Apr. 1998), 2403–2406.
[73] Richard P. Feynman. 1982. Simulating physics with computers. *Int. J. Theor. Phys.* 21, 6/7 (1982).
[74] Iulia M. Georgescu, Sahel Ashhab, and Franco Nori. 2014. Quantum simulation. *Rev. Mod. Phys.* 86, 1 (2014), 153.
[75] Maddalena Ghio, Daniele Malpetti, Matteo Rossi, Dagmar Bruß, and Chiara Macchiavello. 2017. Multipartite entanglement detection for hypergraph states. *J. Phys. A: Math. Theor.* 51, 4 (2017), 045302.
[76] Francisco Javier Gil Vidal and Dirk Oliver Theis. 2020. Input redundancy for parameterized quantum circuits. *Front. Phys.* 8 (2020), 297.
[77] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. 2008. Quantum random access memory. *Phys. Rev. Lett.* 100, 16 (2008), 160501.
[78] Edward Grant, Marcello Benedetti, Shuxiang Cao, Andrew Hallam, Joshua Lockhart, Vid Stojevic, Andrew G. Green, and Simone Severini. 2018. Hierarchical quantum classifiers. *npj Quantum Inf.* 4, 1 (2018), 1–8.
[79] Edward Grant, Leonard Wossnig, Mateusz Ostaszewski, and Marcello Benedetti. 2019. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum* 3 (2019), 214.

[80] Lov K. Grover. 1997. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.* 79, 2 (1997), 325.
[81] Gian Giacomo Guerreschi and Mikhail Smelyanskiy. 2017. Practical optimization for hybrid quantum-classical algorithms. *arXiv preprint arXiv:1701.01450* (2017).
[82] Laszlo Gyongyosi and Sandor Imre. 2019. A survey on quantum computing technology. *Comput. Sci. Rev.* 31 (2019), 51–71.
[83] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. 2019. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* 12, 2 (2019), 34.
[84] Richard Harris, Mark W. Johnson, T. Lanting, A. J. Berkley, J. Johansson, P. Bunyk, E. Tolkacheva, E. Ladizinsky, N. Ladizinsky, T. Oh, et al. 2010. Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor. *Phys. Rev. B* 82, 2 (2010), 024511.
[85] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. 2009. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* 103, 15 (2009), 150502.
[86] Aram W. Harrow and Ashley Montanaro. 2017. Quantum computational supremacy. *Nature* 549, 7671 (2017), 203–209.
[87] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media.
[88] Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. 2019. Supervised learning with quantum-enhanced feature spaces. *Nature* 567, 7747 (2019), 209–212.
[89] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9729–9738.
[90] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 770–778.
[91] Werner Heisenberg. 1925. Quantum-theoretical re-interpretation of kinematic and mechanical relations. *Zeitschrift für Physik* 33 (1925), 879–893.
[92] Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, and Tristan Cook. 2020. Quanvolutional neural networks: Powering image recognition with quantum circuits. *Quant. Mach. Intell.* 2, 1 (2020), 1–9.
[93] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Comput.* 18, 7 (2006), 1527–1554.
[94] Zoë Holmes, Kunal Sharma, M. Cerezo, and Patrick J. Coles. 2021. Connecting ansatz expressibility to gradient magnitudes and barren plateaus. *arXiv preprint arXiv:2101.02138* (2021).
[95] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Netw.* 2, 5 (1989), 359–366.
[96] Wei Hu. 2018. Towards a real quantum neuron. *Nat. Sci.* 10, 3 (2018), 99–109.
[97] Ben Jaderberg, Lewis W. Anderson, Weidi Xie, Samuel Albanie, Martin Kiffner, and Dieter Jaksch. 2021. Quantum self-supervised learning. *arXiv preprint arXiv:2103.14653* (2021).
[98] Mark W. Johnson, Mohammad H. S. Amin, Suzanne Gildert, Trevor Lanting, Firas Hamze, Neil Dickson, Richard Harris, Andrew J. Berkley, Jan Johansson, Paul Bunyk, et al. 2011. Quantum annealing with manufactured spins. *Nature* 473, 7346 (2011), 194–198.
[99] M. W. Johnson, P. Bunyk, F. Maibaum, E. Tolkacheva, A. J. Berkley, E. M. Chapple, R. Harris, J. Johansson, T. Lanting, I. Perminov, et al. 2010. A scalable control system for a superconducting adiabatic quantum optimization processor. *Supercond. Sci. Technol.* 23, 6 (2010), 065004.
[100] Brian David Josephson. 1962. Possible new effects in superconductive tunnelling. *Phys. Lett.* 1, 7 (1962), 251–253.
[101] Brian David Josephson. 1974. The discovery of tunnelling supercurrents. *Rev. Mod. Phys.* 46, 2 (1974), 251.
[102] Gil Kalai and Guy Kindler. 2014. Gaussian noise sensitivity and bosonsampling. *arXiv preprint arXiv:1409.3093* (2014).
[103] Abu Kamruzzaman, Yousef Alhwaiti, Avery Leider, and Charles C. Tappert. 2019. Quantum deep learning neural networks. In *Proceedings of the Future of Information and Communication Conference*. Springer, 299–311.
[104] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. 2017. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* 549, 7671 (2017), 242–246.
[105] Iordanis Kerenidis, Jonas Landman, Alessandro Luongo, and Anupam Prakash. 2018. q-means: A quantum algorithm for unsupervised machine learning. *arXiv preprint arXiv:1812.03584* (2018).
[106] Iordanis Kerenidis, Jonas Landman, and Anupam Prakash. 2020. Quantum algorithms for deep convolutional neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net.

[107] Iordanis Kerenidis and Anupam Prakash. 2020. Quantum gradient descent for linear systems and least squares. *Phys. Rev. A* 101, 2 (2020), 022316.
[108] Tariq M. Khan and Antonio Robles-Kelly. 2020. A derivative-free method for quantum perceptron training in multi-layered neural networks. In *Proceedings of the International Conference on Neural Information Processing*. Springer, 241–250.
[109] Subhash Khot. 2016. Hardness of approximation. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 3–1.
[110] Nathan Killoran, Thomas R. Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolás Quesada, and Seth Lloyd. 2019. Continuous-variable quantum neural networks. *Phys. Rev. Res.* 1, 3 (2019), 033063.
[111] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[112] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. 1983. Optimization by simulated annealing. *Science* 220, 4598 (1983), 671–680.
[113] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS'12)*. 1097–1105.
[114] Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *Handb. Brain Theor. Neural Netw.* 3361, 10 (1995), 1995.
[115] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
[116] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* 1, 4 (1989), 541–551.
[117] Maciej Lewenstein. 1994. Quantum perceptrons. *J. Mod. Opt.* 41, 12 (1994), 2491–2501.
[118] Haichen Li, Christopher Collins, Matteus Tanha, Geoffrey J. Gordon, and David J. Yaron. 2018. A density functional tight binding layer for deep learning of chemical Hamiltonians. *J. Chem. Theor. Computat.* 14, 11 (2018), 5764–5776.
[119] YaoChong Li, Ri-Gui Zhou, RuQing Xu, Jia Luo, and WenWen Hu. 2020. A quantum deep convolutional neural network for image recognition. *Quant. Sci. Technol.* 5, 4 (2020), 044003.
[120] Wenjie Liu, Peipei Gao, Yuxiang Wang, Wenbin Yu, and Maojun Zhang. 2019. A unitary weights based one-iteration quantum perceptron algorithm for non-ideal training sets. *IEEE Access* 7 (2019), 36854–36865.
[121] Seth Lloyd. 1996. Universal quantum simulators. *Science* (1996), 1073–1078.
[122] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. 2014. Quantum principal component analysis. *Nat. Phys.* 10, 9 (2014), 631–633.
[123] Antonio Macaluso, Luca Clissa, Stefano Lodi, and Claudio Sartori. 2020. A variational algorithm for quantum neural networks. In *Proceedings of the International Conference on Computational Science*. Springer, 591–604.
[124] Stefano Mangini, Francesco Tacchino, Dario Gerace, Daniele Bajoni, and Chiara Macchiavello. 2021. Quantum computing models for artificial neural networks. *EPL (Europhys. Lett.)* 134, 1 (2021), 10002.
[125] A. Manju and Madhav J. Nigam. 2014. Applications of quantum inspired computational intelligence: A survey. *Artif. Intell. Rev.* 42, 1 (2014), 79–156.
[126] Andrea Mari, Thomas R. Bromley, Josh Izaac, Maria Schuld, and Nathan Killoran. 2020. Transfer learning in hybrid classical-quantum neural networks. *Quantum* 4 (2020), 340.
[127] Roman Martoňák, Giuseppe E. Santoro, and Erio Tosatti. 2002. Quantum annealing by the path-integral Monte Carlo method: The two-dimensional random Ising model. *Phys. Rev. B* 66, 9 (2002), 094203.
[128] Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. 2018. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* 9, 1 (2018), 1–6.
[129] Jarrod R. McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. 2016. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* 18, 2 (2016), 023023.
[130] Catherine C. McGeoch. 2020. Theory versus practice in annealing-based quantum computing. *Theoret. Comput. Sci.* 816 (2020), 169–183.
[131] Albert Messiah. 1962. *Quantum Mechanics: Volume II*. North-Holland Publishing Company Amsterdam.
[132] Nikolaj Moll, Panagiotis Barkoutsos, Lev S. Bishop, Jerry M. Chow, Andrew Cross, Daniel J. Egger, Stefan Filipp, Andreas Fuhrer, Jay M. Gambetta, Marc Ganzhorn, et al. 2018. Quantum optimization using variational algorithms on near-term quantum devices. *Quant. Sci. Technol.* 3, 3 (2018), 030503.
[133] Ajit Narayanan and Tammy Menneer. 2000. Quantum artificial neural network architectures and components. *Inf. Sci.* 128, 3–4 (2000), 231–255.
[134] Michael A. Nielsen, Isaac Chuang, M. A. Nielsen, and I. L. Chuang. 2002. *Quantum Computation and Quantum Information*. Cambridge University Press.

[135] Yusuke Nomura. 2021. Helping restricted Boltzmann machines with quantum-state representation by restoring symmetry. *J. Phys.: Condens. Matt.* 33, 17 (2021), 174003.
[136] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).
[137] J. S. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. Schuyler Fried, S. Hong, et al. 2017. Unsupervised machine learning on a hybrid quantum computer. *arXiv preprint arXiv:1712.05771* (2017).
[138] Adam Paetznick and Krysta M. Svore. 2014. Repeat-until-success: Non-deterministic decomposition of single-qubit unitaries. *Quant. Inf. Computat.* 14, 15–16 (2014), 1277–1301.
[139] Anargyros Papageorgiou and Joseph F. Traub. 2013. Measures of quantum computing speedup. *Phys. Rev. A* 88, 2 (2013), 022316.
[140] Edwin Pednault, John A. Gunnels, Giacomo Nannicini, Lior Horesh, and Robert Wisnieff. 2019. Leveraging secondary storage to simulate deep 54-qubit sycamore circuits. *arXiv preprint arXiv:1910.09534* (2019).
[141] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. 2020. Data re-uploading for a universal quantum classifier. *Quantum* 4 (2020), 226.
[142] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. 2014. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* 5, 1 (2014), 1–7.
[143] Arthur Pesah, M. Cerezo, Samson Wang, Tyler Volkoff, Andrew T. Sornborger, and Patrick J. Coles. 2021. Absence of barren plateaus in quantum convolutional neural networks. *Phys. Rev. X* 11, 4 (2021), 041011.
[144] David Pfau, James S. Spencer, Alexander G. D. G. Matthews, and W. Matthew C. Foulkes. 2020. Ab initio solution of the many-electron Schrödinger equation with deep neural networks. *Phys. Rev. Res.* 2, 3 (2020), 033429.
[145] John Preskill. 2012. Quantum computing and the entanglement frontier. *arXiv preprint arXiv:1203.5813* (2012).
[146] John Preskill. 2018. Quantum computing in the NISQ era and beyond. *Quantum* 2 (2018), 79.
[147] Saleh Rahimi-Keshari, Timothy C. Ralph, and Carlton M. Caves. 2016. Sufficient conditions for efficient classical simulation of quantum optics. *Phys. Rev. X* 6, 2 (2016), 021039.
[148] Somayeh Bakhtiari Ramezani, Alexander Sommers, Harish Kumar Manchukonda, Shahram Rahimi, and Amin Amirlatifi. 2020. Machine learning algorithms in quantum computing: A survey. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.
[149] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. 2014. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* 113, 13 (2014), 130503.
[150] Markus Reiher, Nathan Wiebe, Krysta M. Svore, Dave Wecker, and Matthias Troyer. 2017. Elucidating reaction mechanisms on quantum computers. *Proc. Natl. Acad. Sci.* 114, 29 (2017), 7555–7560.
[151] Jonathan Romero and Alán Aspuru-Guzik. 2021. Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions. *Adv. Quant. Technol.* 4, 1 (2021), 2000003.
[152] Jonathan Romero, Jonathan P. Olson, and Alan Aspuru-Guzik. 2017. Quantum autoencoders for efficient compression of quantum data. *Quant. Sci. Technol.* 2, 4 (2017), 045001.
[153] Troels F. Rønnow, Zhihui Wang, Joshua Job, Sergio Boixo, Sergei V. Isakov, David Wecker, John M. Martinis, Daniel A. Lidar, and Matthias Troyer. 2014. Defining and detecting quantum speedup. *Science* 345, 6195 (2014), 420–424.
[154] Frank Rosenblatt. 1957. *The Perceptron, a Perceiving and Recognizing Automaton Project Para.* Cornell Aeronautical Laboratory.
[155] Matteo Rossi, Marcus Huber, Dagmar Bruß, and Chiara Macchiavello. 2013. Quantum hypergraph states. *New J. Phys.* 15, 11 (2013), 113022.
[156] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).
[157] Giuseppe E. Santoro, Roman Martoňák, Erio Tosatti, and Roberto Car. 2002. Theory of quantum annealing of an Ising spin glass. *Science* 295, 5564 (2002), 2427–2430.
[158] M. M. Savchuk and A. V. Fesenko. 2019. Quantum computing: Survey and analysis. *Cybern. Syst. Anal.* 55, 1 (2019), 10–21.
[159] Erwin Schrödinger. 1926. Quantisierung als eigenwertproblem. *Ann. Phys.* 385, 13 (1926), 437–490.
[160] Maria Schuld. 2018. *Supervised Learning with Quantum Computers.* Springer.
[161] Maria Schuld, Alex Bocharov, Krysta Svore, and Nathan Wiebe. 2018. Circuit-centric quantum classifiers. *arXiv preprint arXiv:1804.00633* (2018).
[162] Maria Schuld and Nathan Killoran. 2019. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.* 122, 4 (2019), 040504.
[163] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. 2014. The quest for a quantum neural network. *Quant. Inf. Process.* 13, 11 (2014), 2567–2586.

[164] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. 2015. An introduction to quantum machine learning. *Contemp. Phys.* 56, 2 (2015), 172–185.
[165] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. 2015. Simulating a perceptron on a quantum computer. *Phys. Lett. A* 379, 7 (2015), 660–663.
[166] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. 2021. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A* 103, 3 (2021), 032430.
[167] Kristof T. Schütt, Michael Gastegger, Alexandre Tkatchenko, K.-R. Müller, and Reinhard J. Maurer. 2019. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nat. Commun.* 10, 1 (2019), 1–10.
[168] Kunal Sharma, Marco Cerezo, Lukasz Cincio, and Patrick J. Coles. 2020. Trainability of dissipative perceptron-based quantum neural networks. *arXiv preprint arXiv:2005.12458* (2020).
[169] Dan Shepherd and Michael J. Bremner. 2009. Temporally unstructured quantum computation. *Proc. Roy. Societ. A: Math., Phys. Eng. Sci.* 465, 2105 (2009), 1413–1439.
[170] Y.-Y. Shi, L.-M. Duan, and Guifre Vidal. 2006. Classical simulation of quantum many-body systems with a tree tensor network. *Phys. Rev. A* 74, 2 (2006), 022320.
[171] Peter W. Shor. 1994. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*. IEEE, 124–134.
[172] Peter W. Shor. 1999. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* 41, 2 (1999), 303–332.
[173] Adenilton J. Silva, Teresa B. Ludermir, and Wilson R. de Oliveira Jr. 2010. Superposition based learning algorithm. In *Proceedings of the 11th Brazilian Symposium on Neural Networks*. IEEE, 1–6.
[174] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 6419 (2018), 1140–1144.
[175] Andrea Skolik, Jarrod R. McClean, Masoud Mohseni, Patrick van der Smagt, and Martin Leib. 2021. Layerwise learning for quantum neural networks. *Quant. Mach. Intell.* 3, 1 (2021), 1–11.
[176] Justin S. Smith, Olexandr Isayev, and Adrian E. Roitberg. 2017. ANI-1: An extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* 8, 4 (2017), 3192–3203.
[177] Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 1857–1865.
[178] Igor I. Soloviev, Andrey E. Schegolev, Nikolay V. Klenov, Sergey V. Bakurskiy, Mikhail Yu Kupriyanov, Maxim V. Tereshonok, Anton V. Shadrin, Vasily S. Stolyarov, and Alexander A. Golubov. 2018. Adiabatic superconducting artificial neural network: Basic cells. *J. Appl. Phys.* 124, 15 (2018), 152113.
[179] Alessandro Sperduti and Antonina Starita. 1997. Supervised neural networks for the classification of structures. *IEEE Trans. Neural Netw.* 8, 3 (1997), 714–735.
[180] Francesco Tacchino, Chiara Macchiavello, Dario Gerace, and Daniele Bajoni. 2019. An artificial neuron implemented on an actual quantum processor. *Npj Quant. Inf.* 5, 1 (2019), 1–8.
[181] Barbara M. Terhal and David P. DiVincenzo. 2004. Adptive quantum computation, constant depth quantum circuits and Arthur-Merlin games. *Quant. Inf. Computat.* 4, 2 (Mar. 2004), 134–145.
[182] Adel Nadjaran Toosi, Rodrigo N. Calheiros, and Rajkumar Buyya. 2014. Interconnected cloud computing environments: Challenges, taxonomy, and survey. *ACM Comput. Surv.* 47, 1 (2014), 1–47.
[183] Erik Torrontegui and Juan José García-Ripoll. 2019. Unitary quantum perceptron as efficient universal approximator. *Europhys. Lett.* 125, 3 (2019), 30004.
[184] Géza Tóth, Craig S. Lent, P. Douglas Tougaw, Yuriy Brazhnik, Weiwen Weng, Wolfgang Porod, Ruey-Wen Liu, and Yih-Fang Huang. 1996. Quantum cellular neural networks. *Superlattices Microstruct.* 20, 4 (1996), 473–478.
[185] Dan Ventura and Tony Martinez. 2000. Quantum associative memory. *Inf. Sci.* 124, 1–4 (2000), 273–296.
[186] Guillaume Verdon, Michael Broughton, Jarrod R. McClean, Kevin J. Sung, Ryan Babbush, Zhang Jiang, Hartmut Neven, and Masoud Mohseni. 2019. Learning to learn with quantum neural networks via classical neural networks. *arXiv preprint arXiv:1907.05415* (2019).
[187] Guillaume Verdon, Trevor McCourt, Enxhell Luzhnica, Vikash Singh, Stefan Leichenauer, and Jack Hidary. 2019. Quantum graph neural networks. *arXiv preprint arXiv:1909.12264* (2019).
[188] Guifré Vidal. 2008. Class of quantum many-body states that can be efficiently simulated. *Phys. Rev. Lett.* 101, 11 (2008), 110501.
[189] Tyler Volkoff and Patrick J. Coles. 2021. Large gradients via correlation in random parameterized quantum circuits. *Quant. Sci. Technol.* 6, 2 (2021), 025008.
[190] Kwok Ho Wan, Oscar Dahlsten, Hlér Kristjánsson, Robert Gardner, and M. S. Kim. 2017. Quantum generalisation of feedforward neural networks. *Npj Quant. Inf.* 3, 1 (2017), 1–8.

[191] Daochen Wang, Oscar Higgott, and Stephen Brierley. 2019. Accelerated variational quantum eigensolver. *Phys. Rev. Lett.* 122, 14 (2019), 140504.
[192] Samson Wang, Enrico Fontana, Marco Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J. Coles. 2020. Noise-induced barren plateaus in variational quantum algorithms. *arXiv preprint arXiv:2007.14384* (2020).
[193] Dave Wecker, Matthew B. Hastings, Nathan Wiebe, Bryan K. Clark, Chetan Nayak, and Matthias Troyer. 2015. Solving strongly correlated electron models on a quantum computer. *Phys. Rev. A* 92, 6 (2015), 062318.
[194] Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore. 2018. Quantum perceptron models. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS'16).*
[195] R. C. Wiersema and H. J. Kappen. 2019. Implementing perceptron models with qubits. *Phys. Rev. A* 100, 2 (2019), 020301.
[196] Peter Wittek. 2014. *Quantum Machine Learning: What Quantum Computing Means to Data Mining*. Academic Press.
[197] Yao Zhang and Qiang Ni. 2020. Recent advances in quantum machine learning. *Quant. Eng.* 2, 1 (2020), e34.
[198] Yarui Zheng, Chao Song, Ming-Cheng Chen, Benxiang Xia, Wuxin Liu, Qiujiang Guo, Libo Zhang, Da Xu, Hui Deng, Keqiang Huang, et al. 2017. Solving systems of linear equations with a superconducting quantum processor. *Phys. Rev. Lett.* 118, 21 (2017), 210504.
[199] Yiqing Zhou, E. Miles Stoudenmire, and Xavier Waintal. 2020. What limits the simulation of quantum computers? *Phys. Rev. X* 10, 4 (2020), 041038.

Received 7 July 2021; revised 25 March 2022; accepted 30 March 2022