*Article*

# A Review on Models and Applications of Quantum Computing

**Eduard Grigoryan [1], Sachin Kumar [1,*] and Placido Rogério Pinheiro [2,*]**

[1] Akian College of Science and Engineering, American University of Armenia, Yerevan 0019, Armenia; eduard_grigoryan@edu.aua.am
[2] Graduate Program in Applied Informatics, University of Fortaleza, Fortaleza 60811905, Brazil
* Correspondence: sachin.kumar@aua.am (S.K.); placido@unifor.br (P.R.P.)

**Abstract**

This manuscript is intended for readers who have a general interest in the subject of quantum computation and provides an overview of the most significant developments in the field. It begins by introducing foundational concepts from quantum mechanics—such as superposition, entanglement, and the no-cloning theorem—that underpin quantum computation. The primary computational models are discussed, including gate-based (circuit) quantum computing, adiabatic quantum computing, measurement-based quantum computing and the quantum Turing machine. A selection of significant quantum algorithms are reviewed, notably Grover's search algorithm, Shor's factoring algorithm, and Quantum Singular Value Transformation (QSVT), which enables efficient solutions to linear algebra problems on quantum devices. To assess practical performance, we compare quantum and classical implementations of support vector machines (SVMs) using several synthetic datasets. These experiments offer insight into the capabilities and limitations of near-term quantum classifiers relative to classical counterparts. Finally, we review leading quantum programming platforms—including Qiskit, PennyLane, and Cirq—and discuss their roles in bridging theoretical models with real-world quantum hardware. The paper aims to provide a concise yet comprehensive guide for those looking to understand both the theoretical foundations and applied aspects of quantum computing.

**Keywords:** quantum computing; quantum algorithms; quantum machine learning

Academic Editor: Michel Planat

Received: 1 July 2025
Revised: 25 August 2025
Accepted: 3 September 2025
Published: 4 September 2025

**Citation:** Grigoryan, E.; Kumar, S.; Pinheiro, P.R. A Review on Models and Applications of Quantum Computing. *Quantum Rep.* **2025**, *7*, 39. https://doi.org/10.3390/quantum7030039

## 1. Introduction

Quantum computing finds its roots in the realization that information is fundamentally physical and must obey the laws of quantum mechanics. This idea was expressed when a reversible Turing machine governed by a quantum–mechanical Hamiltonian was described [1]. Richard Feynman observed that classical computers cannot efficiently simulate quantum systems and proposed the concept of a quantum simulator to overcome this boundary [2]. Later, David Deutsch formally proposed the idea of universal quantum computer, showing that quantum parallelism could at its core solve certain problems that are beyond reach of any classical Turing machine [3]. Regardless of these theoretical advancements, practical quantum hardware remained unreachable until the past decade, when improvements in qubit control, coherence time, and error mitigation led the way to Noisy Intermediate-scale Quantum (NISQ) era hardware. This hardware is characterized by systems with tens of to a few hundred qubits that lack full error correction but can perform nontrivial quantum behavior [4]. Currently, NISQ systems allow experimental discoveries of quantum algorithms in chemistry, optimization, and machine learning, even as researchers try to overcome noise and scalability problems. Industry and academia have already applied NISQ devices to simulate small molecules' electronic structure, enabling potential breakthroughs in drug discovery and materials science by evaluating ground-state energies more efficiently than classical methods [5]. At the same time, combinatorial optimization problems, such as graph partitioning and portfolio optimization, have been tackled using the quantum approximate optimization algorithm (QAOA). This is a hybrid quantum classical approach which is particularly well suited for NISQ hardware [4,6]. Along with classical approaches, quantum machine learning can be advanced with variational quantum algorithms (VQAs) offering flexible frameworks for tasks such as classification and clustering by parametrizing quantum circuits and training them using classical optimizers [7]. On the quantum networking side, experimental distributed quantum computing was realized across an optical link between two modules separated by two meters. Grover's search algorithm was executed as the first fully distributed quantum algorithm using quantum gate teleportation [8]. Existing review papers on quantum computing offer valuable insights into the field's theoretical foundations, applications, and challenges. Gill et al. [9,10] provide a broad taxonomy and systematic review, emphasizing future directions across various domains. Paudel et al. [11] focus on energy applications, highlighting quantum simulations' potential, while Rietsche et al. [12] outline quantum computing layers and application areas. Putranto et al. [13] analyze the quantum start-up ecosystem, and the Qiskit paper [14] details software tools for quantum computing. However, these reviews have limitations; they often lack comprehensive case studies, focus narrowly on specific sectors, omit detailed experimental comparisons, or are software-centric without broad industry analysis. The novelty of this review lies in its integration of theoretical foundations, computational models, core algorithms, and a detailed case study comparing quantum and classical SVMs using synthetic datasets. This experimental approach provides empirical insights into capabilities and limitations, addressing a gap in practical performance evaluation. Limitations include its reliance on simulated quantum data and lack of access to fault-tolerant hardware. The review spans works from the 1980s to recent advancements, providing a comprehensive historical and contemporary analysis. This paper is intended for a diverse audience, including researchers and students in computational science seeking a comprehensive understanding of quantum computing's theoretical and practical aspects, engineers and developers interested in quantum software platforms and hardware applications, and industry professionals in fields like chemistry, optimization, and machine learning looking to explore quantum-enhanced solutions. The inclusion of a detailed case study and software ecosystem analysis makes it particularly valuable for practitioners aiming to bridge theory and real-world implementation.

## 2. Theoretical Foundations

### *2.1. Qubits and Superposition*

In classical computing we fundamentally distinguish two mutually exclusive physical states, which follow classical laws of physics, but with the development of quantum mechanics, we have an alternative view. What if these states can exist simultaneously? For a moment this could be counterintuitive; however, along with other things, quantum mechanics breaks boundaries. Imagine a solid barrier with two narrow, parallel slits cut into it; on one side of the barrier is placed an emitting device and on the other side is a capturing screen (image). The device starts to emit electrons towards the barrier one at a time; in terms of classical physics it is expected that the particle should pass either through the first or second slit. Eventually the screen should have two lines (projections of slits); however, instead, a dotted diffraction pattern is observed. This experiment shows the duality principle of electrons being both waves and particles. The interference pattern expresses a fundamental concept in quantum mechanics, the superposition. When a single electron approaches the slits, it does not pass through one of them. Instead it exists in a superposition and passes through both of them at the same time. The wave nature makes the electron interfere with itself. The detector device can be placed right before the slit entrance, to observe particles before they enter the slit. Surprisingly, while emitting several electrons, two lines are observed on the screen. This indicates that while an electron is being watched or measured, it collapses into one definite path, either the first or second slit, and behaves like a classical particle [15].

The mathematical definition of a quantum bit (qubit) in Dirac's notation is as follows:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{1}$$

where $\alpha$ and $\beta$ are complex numbers and

$$|\alpha|^2 + |\beta|^2 = 1. \tag{2}$$

(1) is a linear combination of states, otherwise known as the superposition of a state. Here the values of $\alpha$ and $\beta$ are called amplitudes of states and their modulus squares define the probability that after measurement, the state will collapse to the state $|0\rangle$ or $|1\rangle$. Naturally, the reader may come up with the questions of why the amplitude of a state is a complex number and why we should take its modulus square as a probability of the measuring state. First, the amplitudes are defined as complex numbers because the sum of two real numbers between 0 and 1 will always be greater than each of these numbers; in the case of complex numbers, it is not necessary that the sum is greater. In other words, complex numbers enable us to manipulate state probabilities not only by increasing but also by decreasing them, which in the case of real numbers is not possible [16]. To answer the second question, Born's Rule states that the square of the magnitude of a particle's wave function gives the probability density of finding the particle at a particular position when a measurement is made

$$P(a) = |\langle a|\Psi\rangle|^2, \tag{3}$$

where $\langle a|\Psi\rangle$ is the inner product between the state $|a\rangle$ and the quantum state $|\Psi\rangle$, which represents the system's state before measurement. The idea of using the $L_2$ norm is important in the context of probability, as it should be in the range from 0 to 1. More importantly it satisfies the Schrödinger wave function equation:

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{x},t) = \hat{H}\Psi(\mathbf{x},t), \tag{4}$$

where $\Psi(\mathbf{x},t)$ is the wave function, $\hat{H}$ is the Hamiltonian operator, and $\hbar$ is the reduced Planck constant and after arbitrary transformations the amplitude remains normalized [15]. Geometrically it is interpreted that the qubit's state should be normalized to the length 1. In Figure 1 the unit vector $|\Psi\rangle$ represents the state vector of a single qubit system (for obvious reasons it is not possible to represent a multi-qubit system in 3D space). The Bloch sphere maps a qubit's state to a point on a unit sphere. The standard form $|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\phi}\sin(\theta/2)\,|1\rangle$ uses $\theta/2$ to ensure that as $\theta$ goes from 0 to $\pi$, the state moves smoothly from $|0\rangle$ (north pole) to $|1\rangle$ (south pole) without repeating. A simpler form, $\cos\theta\,|0\rangle + e^{i\phi}\sin\theta\,|1\rangle$, would cause a problem: at $\theta = \pi$, it gives $-|0\rangle$, which is equivalent to $|0\rangle$ (since global phase does not affect measurements), not $|1\rangle$. This would map the sphere twice, wasting space and misplacing states. The $\theta/2$ factor fixes this, making the sphere's geometry match the qubit's states perfectly, with opposite points representing orthogonal states like $|0\rangle$ and $|1\rangle$. Here, intuitively, if the vector is in the upper sphere, then with higher probability it will collapse into the basis state $|0\rangle$.

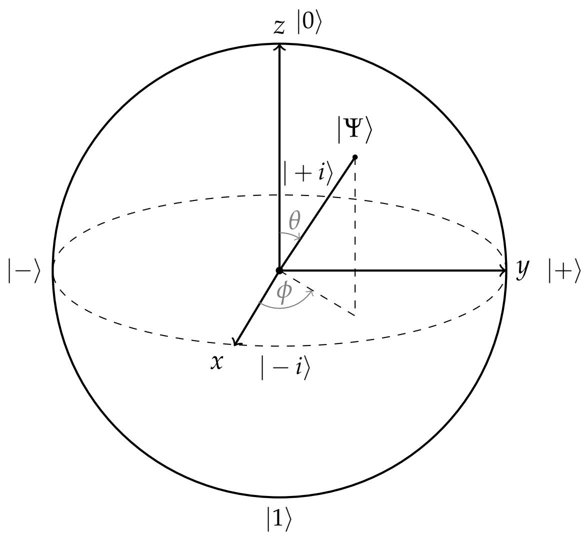

**Figure 1.** Bloch sphere representation of a single qubit state, $\Psi$. The computational basis states $|0\rangle$ and $|1\rangle$ are located at the north and south poles, respectively. Superposition states such as $|+\rangle$, $|-\rangle$, $|+i\rangle$, and $|-i\rangle$ are positioned along the equator. The state $\Psi = \cos(\theta/2)\,|0\rangle + e^{i\phi}\sin(\theta/2)\,|1\rangle$ is represented by a vector on the sphere, defined by the spherical coordinates $\theta$ and $\phi$.

### 2.2. *Entanglement*

Another phenomenon of quantum physics which is hardly explained from a classical perspective is qubit entanglement. Given two qubits, all possible states of a system are

$$|00\rangle\,,|01\rangle\,,|10\rangle\,,|11\rangle \tag{5}$$

or equivalently

$$|0\rangle\,,|1\rangle\,,|2\rangle\,,|3\rangle\,.$$

Suppose that the first qubit is in an equally likely superposition,

$$\frac{|0\rangle+|1\rangle}{\sqrt{2}} \tag{6}$$

and the second qubit is in the base state $|0\rangle$; then their combined state is a tensor product of individual states:

$$\frac{|0\rangle+|1\rangle}{\sqrt{2}}\otimes|0\rangle = \frac{|00\rangle+|10\rangle}{\sqrt{2}}. \tag{7}$$

These qubits are separable as we can take out $|0\rangle$. Let us introduce a binary function called CNOT which negates qubits depending on controlled qubits. In other words, with highly controlled qubits, the target qubit is flipped. This function (actually the gate, covered in the next sections), applied to (7), evaluates the state as follows:

$$\frac{|00\rangle+|10\rangle}{\sqrt{2}} \xrightarrow{\text{CNOT}} \frac{|00\rangle+|11\rangle}{\sqrt{2}}. \tag{8}$$

Mathematically, the new state is not separable; no common qubits can be taken out. Physically this means that while measuring one of the qubits, the other also collapses into a definite state. For example, if the first bit is measured and it is 1, then without looking into the content of the second qubit, it can be stated that it is also 1; the same applies for the 0 state. It is proven that even if the qubits are separated in the distance, they still remain entangled. Examples of separable states include all computational basis states (5), as well as product superpositions such as (7) or

$$|1\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}, \tag{9}$$

in general form

$$\begin{aligned} &(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (\gamma\,|0\rangle + \delta\,|1\rangle) = \\ &(\alpha\gamma)\,|00\rangle + (\alpha\delta)\,|01\rangle + (\beta\gamma)\,|10\rangle + (\beta\delta)\,|11\rangle\,. \end{aligned} \tag{10}$$

where $|\alpha|^2 + |\beta|^2 = 1$ and $|\gamma|^2 + |\delta|^2 = 1$. On the other hand, a state is entangled if it cannot be factorized into a product of states of its individual subsystems. The canonical maximally entangled states for two qubits are the Bell states:

$$|\Phi^{\pm}\rangle = \frac{|00\rangle \pm |11\rangle}{\sqrt{2}}, \quad |\Psi^{\pm}\rangle = \frac{|01\rangle \pm |10\rangle}{\sqrt{2}}. \tag{11}$$

With this definition, if a quantum state is entangled, it exhibits correlations between its subsystems that cannot be explained by any local, classical means and are a direct result of the quantum entanglement. From a statistical perspective, entanglement can be described using the density matrix formalism. The density matrix unifies key quantum properties, namely, entanglement, coherence, and entropic uncertainty, and provides alternative state representation. The density matrix $\rho$, satisfying $Tr(\rho) = \sum_i \langle i|\rho|i\rangle = 1$, describes a system's state, both for single qubits and multi-qubit systems. A pure state, similar to a single qubit (6) or a two-qubit Bell state (11), is a fully known quantum state with no uncertainty, represented by

$$\rho = |\psi\rangle\,\langle\psi| \tag{12}$$

and satisfying $Tr(\rho^2) = 1$. In contrast, a mixed state, caused by noise or incomplete knowledge, is an ensemble of pure states:

$$\rho = \sum_i p_i\,|\psi_i\rangle\,\langle\psi_i| \tag{13}$$

where $p_i$ are probabilities representing the likelihood of the system being in each pure state $|\psi_i\rangle$ with $Tr(\rho^2) < 1$, indicating uncertainty. A separable state or any product state is independent and can be written as a tensor product, $\rho_{AB} = \rho_A \otimes \rho_B$. However, entangled states, such as the maximally entangled Bell states (11), link qubits; therefore, measuring one instantly affects the other, and $\rho_{AB} \neq \rho_A \otimes \rho_B$. In case of an entangled pair, the individual qubits states can be expressed by the reduced density matrix $\rho_A$ or $\rho_B$ as subsystems. The main advantage of the density matrix formalism is that it allows us to describe entangled states and their subsystems, whereas using the state vector representation, the individual qubit states are uncertain and correlated and therefore cannot be written. For example, for two-qubit system in the state $|\Phi^+\rangle$, the density matrix is given by

$$\begin{aligned} &\rho_{AB} = |\Phi^+\rangle\,\langle\Phi^+| = \\ &\frac{1}{2}(|00\rangle\,\langle 00| + |00\rangle\,\langle 11| + |11\rangle\,\langle 00| + |11\rangle\,\langle 11|) \end{aligned} \tag{14}$$

or in matrix form:

$$\rho_{AB} = \frac{1}{2}\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

Using the reduced density matrix, the individual qubit state can be expressed as $\rho_A = Tr_B(\rho_{AB})$, and it is obtained by summing over B's states using the partial trace, averaging over B to isolate A's state:

$$\rho_A = \mathrm{Tr}_B(\rho_{AB}) = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{16}$$

This $\rho_A$ becomes mixed ($Tr(\rho_A^2) < 1$) for entangled states, unlike pure separable states. In open systems, noise affects $\rho_{AB}$ via the master equation:

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H, \rho] + \mathcal{L}(\rho), \tag{17}$$

where $H$ drives qubit–resonator coupling and $\mathcal{L}(\rho)$ models noise, namely, amplitude damping and dephasing [17,18]. This causes entanglement to oscillate, decay, or revive under detuning, which is a frequency mismatch between qubits and resonators.

The ability to preserve superpositions is measured by the coherence $L_1$-norm $C_{L_1}(\rho) = \sum_{i \neq j} |\rho_{ij}|$, where $\rho_{ij}$ are off-diagonal elements. Noise reduces these elements, making $\rho_{AB}$ more classical and less quantum. This loss is connected to entropic uncertainty, which measures unpredictability in measuring two properties, like spin in X and Z directions. It is given by

$$H(X|B) + H(Z|B) \geq \log_2 \frac{1}{c} + S(A|B), \tag{18}$$

where $H(X|B) = S(\rho_{XB}) - S(\rho_B)$ is the conditional entropy. $H(X|B)$ quantifies uncertainty in measuring observable X on qubit A given access to qubit B's state, with $\rho_{XB}$ denoting the joint state after measuring X [19]. Here, $S(\rho) = -Tr(\rho \log_2 \rho)$ is the von Neumann entropy, measuring disorder in $\rho$. The term $c$ is compatibility, indicating how much X and Z overlap, and $S(A|B) = S(\rho_{AB}) - S(\rho_B)$ decreases with entanglement, tightening the uncertainty bound. Noise diminishes entanglement and coherence, at the same time increasing entropy and uncertainty. However, detuning can revive these quantum features by preserving $\rho_{AB}$'s structure. This relationship, encoded in $\rho$, is essential for robust quantum computation. The discussion of entanglement has primarily focused on its theoretical formalisms. Another crucial aspect concerns the generation and control of entanglement in practice, particularly through nonlinear interactions in physical systems. In quantum physics, nonlinear processes are interactions between quantum systems, such as qubits, resonators, and oscillators, where a small change in one system causes large, complex, or unpredictable effects in another. Nonlinear processes in quantum entanglement provide powerful mechanisms both for generating and controlling entangled states on optical, mechanical, and hybrid platforms. The main method for entangled photon generation is the second-order $\chi^2$ process known as spontaneous parametric down-conversion (SPDC), where a high-energy pump photon is converted inside a nonlinear crystal into two lower-energy photons whose properties are quantum-correlated [20,21]. Recent integrated implementations achieve high-brightness, narrow-band emission in CMOS-compatible devices [22]. Theoretical advances now enable nonperturbative treatment of SPDC in the high-gain regime, expanding its applicability to on-chip and multimode scenarios [23]. Beyond $\chi^2$ interactions, third-order

($\chi^3$) Kerr nonlinearities have been harnessed to mediate deterministic entangling gates in photonic circuits [24] and to enhance bipartite entanglement in hybrid magnon–photon systems [25], while Duffing-type mechanical nonlinearities can increase steady-state entanglement in optomechanical setups [26]. Advances in engineered materials, including ultrathin van der Waals heterostructures [27] and nanophotonic platforms [28], have significantly boosted nonlinear interaction strengths, enabling brighter and more scalable entanglement sources. Comprehensive theoretical frameworks that incorporate multimode structure and mode-matching considerations [29] are increasingly essential for accurately modeling and optimizing these nonlinear entanglement processes.

### *2.3. No-Cloning Theorem*

Copying is one of the fundamental operations in computer science; however, surprisingly, quantum mechanics does not allow such operation [30,31]. Suppose that we have a given state, $|\psi\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$. Clearly, if the first qubit is 0, then the second qubit is also evaluated to 0 (the same for the case with 1). However, what if the state of the second qubit is also an arbitrary state? Suppose that we have the arbitrary two qubit system state

$$|\psi\rangle \otimes |e\rangle$$

Then applying some function, $U$ (which can be treated as Hamiltonian applied to a state), the content of the second qubit will be evaluated to the content of the first qubit or equivalently

$$U\,|\psi\rangle \otimes |e\rangle = |\psi\rangle \otimes |\psi\rangle\,. \tag{19}$$

Now let us show that it is not possible to come up with such a Hamiltonian, $U$. Assuming that there is such a Hamiltonian, $U$, and the states $|\psi_1\rangle\,, |\psi_2\rangle$, then the following must hold:

$$U\,|\psi_1\rangle \otimes |e\rangle = |\psi_1\rangle \otimes |\psi_1\rangle \tag{20}$$

$$U\,|\psi_2\rangle \otimes |e\rangle = |\psi_2\rangle \otimes |\psi_2\rangle \tag{21}$$

Next, let us define the linear superposition

$$|\phi\rangle = \alpha\,|\psi_1\rangle + \beta\,|\psi_2\rangle$$

Applying the $U$ Hamiltonian,

$$U\,|\phi\rangle \otimes |e\rangle = U(\alpha\,|\psi_1\rangle + \beta\,|\psi_2\rangle) \otimes |e\rangle$$

Then, by linearity,

$$U\,|\phi\rangle \otimes |e\rangle = \alpha U\,|\psi_1\rangle \otimes |e\rangle + \beta U\,|\psi_2\rangle \otimes |e\rangle\,.$$

Substituting (20) and (21),

$$U\,|\phi\rangle \otimes |e\rangle = \alpha\,|\psi_1\rangle \otimes |\psi_1\rangle + \beta\,|\psi_2\rangle \otimes |\psi_2\rangle \tag{22}$$

but if cloning were truly possible, then the state should be

$$|\phi\rangle \otimes |\phi\rangle = (\alpha\,|\psi_1\rangle + \beta\,|\psi_2\rangle) \otimes (\alpha\,|\psi_1\rangle + \beta\,|\psi_2\rangle)$$

$$\begin{aligned} &= \alpha^2\,|\psi_1\rangle \otimes |\psi_1\rangle + \alpha\beta\,|\psi_1\rangle \otimes |\psi_2\rangle \\ &+\beta\alpha\,|\psi_2\rangle \otimes |\psi_1\rangle + \beta^2\,|\psi_2\rangle \otimes |\psi_2\rangle \end{aligned}$$

which contradicts (22). This expression includes the cross-terms $\alpha\beta|\psi_1\rangle \otimes |\psi_2\rangle$ and $\beta\alpha|\psi_2\rangle \otimes |\psi_1\rangle$, which are absent in (22). For these expressions to be equivalent for arbitrary $|\psi_1\rangle$ and $|\psi_2\rangle$, the tensor product would need to exhibit a commutative isomorphism, meaning that the order of states in the tensor product (e.g., $|\psi_1\rangle \otimes |\psi_2\rangle$ vs. $|\psi_2\rangle \otimes |\psi_1\rangle$) would not affect the result. However, quantum mechanics does not permit such commutativity for arbitrary states, leading to a contradiction. Thus, no unitary $U$ can clone arbitrary quantum states, proving the no-cloning theorem. This limitation poses a significant challenge in quantum computing, where manipulating quantum states is central. Unlike classical computing, where data can be duplicated for backups or checkpoints, the inability to clone quantum states prevents such operations, complicating error correction and state recovery during computations. At the same time, this limitation can be leveraged for certain quantum protocols, such as Quantum Secure Direct Communication (QSDC), a protocol for securely transmitting messages without a separate key exchange; quantum states are encoded into the polarization of photons and transported via optical fibers [32]. Recent advancements in QSDC demonstrate its potential for secure communication using photonic qubits. A device-independent QSDC protocol, utilizing high-efficiency single-photon sources and a heralded entanglement architecture, achieves secure communication over distances approximately six times longer than those of earlier protocols, with efficiency improved by a factor of 600 [33]. Additionally, a hybrid entanglement-based QSDC protocol combines continuous and discrete variable encoding, leveraging simple linear optical elements to enhance communication efficiency and fidelity, maintaining security for bit error rates below 0.073 [34]. Unlike classical systems, where data duplication is straightforward, the no-cloning constraint in quantum computing leads to novel approaches to error correction and state management.

## 3. Quantum Computation Models

### 3.1. Gate Model

Operations on qubits are performed by some model of computation, which aligns with quantum mechanics rules. As in classical computing, quantum computing also defines a gate model of computation [35]. First, all the operations should be performed by unitary operators, meaning that $UU^\dagger = I$. This states that every operation is reversible, which is a fundamental rule in quantum mechanics [36]. In the context of logical operations on qubits (logical gates), this reversibility implies that for a given set of output qubit states, it is possible to unambiguously recover the input set of qubit states. Unitarity ensures that during time the total probability of the system is preserved and no information is lost while performing operations. The most essential gate is the Hadamard gate, a single-qubit gate, H, which is defined as

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{23}$$

This gate is the first step in quantum circuit design as it creates an equally likely superposition state given the initial qubit

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \tag{24}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{25}$$

for $|0\rangle$, $|1\rangle$ basis states, respectively. It is easy to verify that the gate is unitary and reversible, $HH^\dagger = I$. The phase shift operator is the next building block. It is defined as

$$R_\theta = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix} \tag{26}$$

or, equivalently,

$$R_\theta|0\rangle = |0\rangle, \quad R_\theta|1\rangle = e^{i\theta}|1\rangle \tag{27}$$

This operator shifts the phase of the state; on the Bloch sphere (Figure 1), the change affects the $\phi$ angle, leaving $\theta$ the same. After applying phase shift, the probability does not change; however, the state is modified in phase. The Pauli-X gate is a basic single-qubit operation, which is known as the quantum NOT gate. It flips the state of a qubit from $|0\rangle$ to $|1\rangle$ and vice versa:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad X|0\rangle = |1\rangle, \quad X|1\rangle = |0\rangle. \tag{28}$$

Geometrically, on the Bloch sphere, the Pauli-X gate represents a rotation by $\pi$ radians around the X-axis. Next is the CNOT gate which was mentioned earlier in the entanglement discussion. Given two qubit inputs, the first qubit acts as a controlling qubit and the second as a target qubit in the computational basis $\{|0\rangle, |1\rangle\}$. However, the designation of control and target qubits can depend on the basis used. For example, in the Hadamard-transformed basis $\{|+\rangle, |-\rangle\}$, the roles interchange, the state of the second qubit remains unchanged, while the state of the first qubit is flipped based on the state of the second qubit. In practice, this basis dependence is accounted for by carefully designing quantum circuits in the standard computational basis, where the control and target roles are well defined and transform to other bases only when necessary through additional gates like Hadamard gates applied before and after CNOT. In a matrix form, CNOT is defined:

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{29}$$

When the first qubit is $|1\rangle$, the second qubit is flipped; otherwise, it remains unchanged. Alternatively, CNOT can be interpreted as the XOR of qubits written as target qubits:

$$\text{CNOT}(|a\rangle \otimes |b\rangle) = |a\rangle \otimes |a \oplus b\rangle \tag{30}$$

The general *controlled-U* gate is defined as follows. If the control qubit is $|0\rangle$, the unitary $U$ is not applied; otherwise, the gate is applied to the target qubit:

$$|0\rangle \otimes |x\rangle \mapsto |0\rangle \otimes |x\rangle, \quad |1\rangle \otimes |x\rangle \mapsto |1\rangle \otimes U|x\rangle. \tag{31}$$

The Toffoli gate is an extended version of CNOT with three input qubits. Given three input bits, if the first two are $|1\rangle$, then the last is flipped; otherwise, it remains the same.

$$|x, y, z\rangle \mapsto |x, y, z \oplus (x \cdot y)\rangle. \tag{32}$$

and the corresponding $8 \times 8$ matrix is

$$\text{Toffoli} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (33)$$

### *3.2. Quantum Adiabatic Model*

The quantum adiabatic model (QAM) of computation is based on adiabatic theory [37]. Given a quantum system described by the Schrödinger Equation (4), the theory states that if the system starts in the ground state $\Psi_I$ of an initial Hamiltonian, $H_I$, at time $t = 0$ and the Hamiltonian changes sufficiently slowly to $H_T$ at time $t = T$, then the system will remain in the ground state of the final Hamiltonian $H_T$. The function of $H(t)$ can be formulated as

$$H(t) = (1 - \frac{t}{T})H_I + \frac{t}{T}H_T. \quad (34)$$

This method of computation is generally used in optimization problems where the optimal solution is hard to find. The first general algorithm was demonstrated on the *3-SAT* problem [38]. Later, it was shown that the adiabatic model is polynomially equivalent to gate model [39]. However, the main challenge is how slowly the Hamiltonian must evolve to ensure that the system reaches the optimal final state. This requirement poses practical difficulties in implementing adiabatic quantum systems [40]. The formal computational cost of the adiabatic algorithm is determined by the total evolution time $T$, which scales as $T \sim 1/\Delta^2_{\text{min}}$, where $\Delta_{\text{min}}$ is the minimal spectral gap along the Hamiltonian path, to ensure that the system remains in the ground state with high probability [41]. This cost is intimately connected to the spectral gap, as the adiabatic theorem requires the evolution to be sufficiently slow relative to $\Delta_{\text{min}}$, specifically satisfying $T \gg \max_t |\frac{dH}{dt}|/\Delta^2_{\text{min}}$, where smaller gaps necessitate longer times to avoid excitations [41,42]. In the context of simulating a quantum circuit with $L$ gates using the adiabatic model, the minimal gap scales as $\Delta_{\text{min}} \sim \pi^2/[8(L+1)^2]$ for large $L$, leading to a polynomial evolution time, $T \sim \text{poly}(L)$, that establishes polynomial equivalence between the adiabatic and gate models [43]. The Hamiltonian must evolve slowly enough to maintain adiabaticity, with the required slowness inversely proportional to the square of the spectral gap, posing challenges when $\Delta_{\text{min}}$ closes exponentially for hard optimization instances [44]. Key drawbacks of the QAM include exponentially small spectral gaps in problems with first-order phase transitions, leading to prohibitively long evolution times, sensitivity to decoherence and thermal noise that can excite the system out of the ground state, and classical simulability in certain regimes, such as one-dimensional systems with constant gaps or stoquastic Hamiltonians without a sign problem [41,45]. Suggestions to mitigate these drawbacks involve shortcuts to adiabaticity via counter-diabatic driving to suppress transitions, diabatic annealing protocols that allow controlled excitations for faster convergence, variational quantum adiabatic algorithms (VQAAs) that optimize nonlinear paths using gradient-based methods to reduce time by factors of $\sim$10 while maintaining high fidelity, and reinforcement learning to design adaptive Hamiltonian schedules that enhance success probabilities, as demonstrated for the 3-SAT and Grover search problems [42,44,46,47].

## 3.3. Measurement-Based Quantum Computation

Measurement-based quantum computation (MBQC), also known as one-way quantum computation, is a model of quantum processing where universal computation is achieved through adaptive single-qubit measurements in a pre-entangled multi-qubit resource state, rather than sequential unitary gates. In MBQC, computation starts with a special quantum state called a *cluster state*, which is a grid of qubits all tangled together in a highly organized way, created by applying specific controlled-phase gates to a starting state where all qubits are in a $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ state. Mathematically, a cluster state on a lattice is defined by *stabilizers*, which are rules like

$$K(a) = \sigma_x^{(a)} \prod_{b \in \mathrm{nbgh}(a)} \sigma_z^{(b)} |\Phi\rangle_C = |\Phi\rangle_C, \tag{35}$$

where $\sigma_x$ and $\sigma_z$ are Pauli operators and $\mathrm{nbgh}(a)$ are neighboring qubits on the grid; this setup ensures the qubits are deeply entangled, meaning that their states are interconnected [48,49]. Computation takes the following process; each qubit, one by one, is measured in a specific basis, and these measurements control the quantum state for performing calculations, with results adjusted using simple corrections (applying $\sigma_x$ or $\sigma_z$ based on random measurement outcomes) to obtain the desired answer [50]. MBQC's advantages are the following: it is used for systems with photons or trapped ions where creating entanglement upfront is easier than applying gates repeatedly, it can perform many operations at once to speed things up (certain calculations collapse nearly instantly), and it works with error-correcting codes to protect against mistakes [51]. However, there are downsides; a huge number of qubits are used (sometimes thousands more than those in other methods), measurements destroy the state and, therefore, a new one needs to be created each time, and choosing the right measurement angles in noisy conditions is tricky [49]. Compared to the gate model, MBQC simplifies hardware needs but uses more qubits; unlike adiabatic quantum computing, MBQC is faster but sensitive to measurement errors [52]. Recent advancements make MBQC even more promising; for example, new methods use continuous-variable systems (like light waves) with special error-correcting codes (GKP codes) to achieve fault tolerance with a squeezing threshold of 12.7 dB, meaning that they can handle errors well in photonic systems [53]. Other advances include efficient designs using color-code states for simpler hardware [54], low-overhead error correction with Gaussian operations [55], and even combining MBQC with machine learning to generate complex data patterns [56]. These developments show MBQC's potential in future quantum computers, especially for systems where measurements are easier than gates, though it still needs work to reduce qubit demands and handle noise effectively [50].

## 3.4. Quantum Turing Machine

With an advancement in quantum mechanics, researchers began rethinking the Church–Turing thesis. The thesis states that each problem, naturally regarded as computable, can be simulated by a Turing machine. Furthermore, the universal Turing machine showed that a single machine could simulate any other Turing machine. However, a question arose: whether this classical model was sufficient when considering the physical world, which is governed by quantum mechanics. Eventually, a new formulation of the Church–Turing thesis was proposed accounting quantum mechanics, which states that "*Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means [3]*". Deutsch [3] defines a finite processor which consists of two-state observables, $\{\hat{n}_i\}$. The tape is infinite with the two-state observables $\{\hat{m}_i\}$. The computation is fixed in time, $T$, during which the processor and finite part of the memory tape interact. The observable $\{\hat{x}\}$ represents the address of the tape at which the reading head is currently located. The state of the machine is determined by the superposition $|x, n, m\rangle$ which is in the composite Hilbert space of $\mathcal{H} = \mathcal{H}_{\text{control}} \otimes \mathcal{H}_{\text{tape}} \otimes \mathcal{H}_{\text{head}}$. The transition, as in the Turing machine, is the evolution of the quantum state governed by a unitary operator, $U$, which acts on the entire system, namely, the control, tape, and head. It is crucial to note that two consecutive states cannot be identical after non-trivial evolution of the system. Moreover, the state should not be observed before the end of computation as it will collapse and remaining transformations would not be possible to resume. To signal that the computation is halted, a separate bit is needed in the processor $\hat{n}_0$. If the program is valid, then $n_0$ is set to 1 to indicate halting; otherwise, no interaction is needed. Therefore, periodically observing this qubit, we can determine when to measure the state and finish the computation. Later, Berstein and Vizari [57] proposed a widely accepted formal definition of the quantum Turing machine (QTM) as follows:

$$M = (Q, \Sigma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}, \mathcal{H}) \tag{36}$$

where

- $Q$ is a finite set of internal (control) states;
- $\Sigma$ is a finite tape alphabet containing a blank symbol, #;
- $\delta$ is the transition function, a map defining amplitudes of transitions:

$$\delta : Q \times \Sigma \times Q \times \Sigma \times \{L, R\} \to \mathbb{C}$$

- $q_0 \in Q$ is the initial state;
- $q_{\text{accept}} \in Q$ is the accepting state;
- $q_{\text{reject}} \in Q$ is the rejecting state ($q_{\text{accept}} \neq q_{\text{reject}}$);
- $\mathcal{H}$ is the Hilbert space spanned by basis states:

$$|q, x, h\rangle$$

where $q \in Q$, $x \in \Sigma^{\mathbb{Z}}$ is the tape content, and $h \in \mathbb{Z}$ is the position of the head.

The major difference between Deutsch's proposal is that this a QTM allows the $U$ operator and system state to be defined by complex numbers rather than positive reals. Furthermore, Bounded-error Quantum Polynomial (BQP) time class of decision problems was introduced, which is the quantum equivalent of classical Bounded-error Probabilistic Polynomial (BPP) time. Additionally, a universal QTM was shown, capable of simulating any other QTM machine with only polynomial slowdown [57].

## 4. Quantum Algorithms

### *4.1. Quantum Search*

Classically, we are bound linearly to find some element in an arbitrary array, which means that we should look through each element and compare it with our sample. However, in quantum settings, this can be performed more effectively. For simplicity, $N = 2^n$; in other words, with $N$ elements, they are encoded by $n$ qubits. Let us define a function, $f : \{0, 1\}^n \to \{0, 1\}$, such that there exists exactly one binary string, $x_0$, for wich $f(x_0) = 1$, and for the rest, $x \neq x_0$, $f(x) = 0$. The general idea of the algorithm is to amplify the target state amplitude and measure it. First, the superposition of $n$ qubits is prepared by applying the Hadamard gate to each qubit. Then, applying the phase inversion or $U_f$ gate, the amplitude of the desired state is inverted:

$$U_f |x\rangle |t\rangle = |x\rangle |t \oplus f(x)\rangle \tag{37}$$

where $f(x)$ encodes desired target state. After this operation, all the amplitudes remain unchanged except the target's, which is inverted. If the measurement is conducted at this point, it will not be possible to differentiate the target since, although it became inverted, the probability of measuring it remains equally likely compared with that of bad states. The next step is to apply a *diffusion* operator or inversion about the mean. This operator is defined as

$$D = 2|\psi\rangle\langle\psi| - I \tag{38}$$

where $|\psi\rangle$ is the equal superposition state:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \tag{39}$$

The effect of this operator is that all amplitudes are inverted around the mean. As a result, our target amplitude is amplified and the bad states' amplitudes are shrinked. However, applying these two operators once does not guarantee that after measurement the desired state will collapse. It was proven that approximately $\sqrt{N}$ iterations should be performed to amplify the target state, making the algorithm $\mathcal{O}(\sqrt{N})$ complex [58]. Later, a multiple-target approach was shown which extends quantum search with the possibility of multiple targets present in array; in this case the number iterations becomes smaller and overall complexity becomes $\mathcal{O}(\sqrt{N/t})$ where t is the number of solutions [59]. Moreover, it was proven that any quantum algorithm needs at least $\Omega(\sqrt{N})$ queries to solve the unstructured search problem, with $N$ elements [60].

### *4.2. Quantum Factoring*

Number factoring is the cornerstone of modern cryptography; more precisely, the absence of effective factoring algorithms makes encryption algorithms work. With the development of quantum computing, one of the most influential algorithms is Shor's factorization algorithm which is exponentially faster than existing classical algorithms, ensuring, at least by now, theoretical quantum speedup [61]. Although Shor's algorithm is complex in its mathematical formulation, the actual quantum routine is what makes it efficient. Given a number, $N$, composed of prime factors, $p$ and $q$, the task is to efficiently find prime factors, knowing only $N$. The algorithm begins by choosing an arbitrary number, $a$, such that

$$1 < a < N \text{ and } gcd(a, N) = 1, \tag{40}$$

meaning that $a$ does not have common factors with $N$. Next, with

$$a^r \equiv 1 \mod N, \tag{41}$$

the main task becomes to find the minimum order $r$. If the order can be found, using the facts that

$$a^r - 1 = (a^{r/2} + 1)(a^{r/2} - 1) \tag{42}$$

and

$$a^r \equiv 1 \mod N \tag{43}$$

it follows that

$$(a^{r/2} + 1)(a^{r/2} - 1) \equiv 0 \mod N, \tag{44}$$

which means that $N$ divides this product. Therefore, calculating the greatest common divisor (GCD),

$$\text{gcd}\left(a^{r/2} \pm 1, N\right) \tag{45}$$

can reveal the prime factor. The classical bottleneck is the computation of the order $r$ which can take time up to $\mathcal{O}(r \log N)$ where r can be up to N, which is impractical for large N. However, in quantum settings there is a method of order finding which boosts the process dramatically. Quantum order finding is implemented as follows: The first register initialized with $|0\rangle$ handles the superposition of $m$ qubits; in this register the number of qubits represents the precision of our calculations. The second register has $n = \lceil \log N \rceil$ qubits initialized with $|1\rangle$, to represent the output of the function $f(x) = a^x \mod N$, which takes values in the range $\{0, 1, \ldots, N-1\}$. After initializing the system, the quantum state is prepared as follows:

$$|0\rangle^{\otimes m} |1\rangle^{\otimes n} \xrightarrow{H^{\otimes m}} \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle |1\rangle \tag{46}$$

The modular exponentiation unitary $U_f$ is defined as

$$U_f : |x\rangle |1\rangle \mapsto |x\rangle |a^x \bmod N\rangle \tag{47}$$

and applied to the state

$$\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x\rangle |a^x \bmod N\rangle \tag{48}$$

Without going into details of the quantum Fourier transform, after applying it, the state has the form

$$\frac{1}{\sqrt{2^m r}} \sum_{s=0}^{r-1} \sum_{k=0}^{2^m-1} e^{2\pi i s k / r} |k\rangle |u_s\rangle, \tag{49}$$

where $|u_s\rangle$ is an eigenstate of the modular exponentiation unitary operator $U_f$ and $s$ represents different frequencies ranging from 0 to $r-1$. After applying the inverse quantum Fourier transform, the registers evolve to the state

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |s/r\rangle |u_s\rangle. \tag{50}$$

At this point, measuring the state will give a number, $s$, which is close to a rational multiple of $2^n/r$. This is because the quantum Fourier transform of the first register of n qubits (where 2n is the register size) resolves frequencies associated with the period r, producing peaks at multiples of 2n/r in the measurement probabilities. Finally, using continued fractions, the value $s/2^n$ can be approximated, based on the fact that

$$\frac{s}{2^n} \approx \frac{k}{r} \tag{51}$$

where $k$ is some integer. Therefore the smallest value of $r$ among possible candidates can be determined. As described previously, using the classical Euclidean algorithm for computing the GCD, the prime factors of $N$ can be extracted. It should be noted that if the randomly chosen $a$ does not have even order, or if the resulting $r$ does not lead to a successful factorization, then the entire process is repeated. Since the set of valid $a$ values is exponentially large (up to $N-1$), the probability of eventually finding an appropriate $a$ is high. The complexity of this algorithm is $\mathcal{O}(\log^3 N)$, which demonstrates an exponential speedup compared to the best-known classical factoring algorithms. Although the result is groundbreaking, the practical implementation of this algorithm requires a multi-qubit reliable quantum system and error-correction techniques for scaling [62]. Using Shor's algorithm to break traditional cryptographic schemes like RSA can be done with 20 million noisy qubits; RSA-2048 could be factored in as little as 8 h [63].

### *4.3. Quantum Singular Value Transformation*

In classical linear algebra, Singular Value Decomposition (SVD) is a matrix factorization method which decomposes an arbitrary matrix, $A$, as

$$A = USV^{\dagger}, \tag{52}$$

where $U$ is a unitary matrix of left singular vectors, $S$ is a diagonal matrix of singular values, and $V^{\dagger}$ is the conjugate transpose of a unitary matrix of right singular vectors. Alternatively, the matrix can be written in sum form as

$$A = \sum_{i=1}^{r} s_i u_i v_i^{\dagger}, \tag{53}$$

where $s_i$ are the singular values, and $\mathbf{u}_i, \mathbf{v}_i$ are the left and right singular vectors, respectively. These singular values encode how the matrix stretches or compresses input vectors. Classically, to apply a function, $f$, to $A$, it is necessary to decompose $A$ and apply the function to its singular values which is a computationally expensive task. In a novel quantum approach, it is shown that the function $f$ can be applied without explicitly calculating the singular values [64]. First, the matrix $A$ must be embedded into a larger unitary matrix, $U$, a process known as *block encoding*, typically in the form

$$U = \begin{bmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{bmatrix}. \tag{54}$$

Since quantum circuits cannot directly encode arbitrary functions, the target function must be approximated. This is achieved using *Chebyshev polynomial approximation*, where any continuous function, $f$, can be written as

$$f(x) \approx \sum_{k=0}^{d} a_k T_k(x), \tag{55}$$

where $T_k(x)$ are the Chebyshev polynomials. The result is a polynomial, $P(x)$, of the degree $d$ that approximates $f(x)$ over a bounded interval. This is implemented through a sequence of quantum gates generated by a series of phases, $\{\phi_1, \phi_2, \ldots \phi_d\}$. It is proven that computing these phases has the complexity $\mathcal{O}(d^3 \cdot \mathrm{poly}(d/\epsilon))$ [65]. The general procedure to construct the $f$ function is as follows. After classically computing the phase sequence, the combined operator is defined as

$$U_{\mathrm{QSVT}} = P_0(\phi_1) U P_0(\phi_2) U^{\dagger} \ldots, \tag{56}$$

where $P_0(\phi_i)$ is a conditional phase gate that applies a phase shift only when the ancilla qubits are in the $|0\ldots0\rangle$ state, ensuring that the transformation targets the encoded $A$ block. Depending on the parity of $d$, if $d$ is even, then the final gate that is applied is $U^{\dagger}$; otherwise, it is $U$. This gate encodes the polynomial $P(A)$, that is,

$$U_{\mathrm{QSVT}} \approx f(A), \tag{57}$$

and can then be applied to a general $|\phi\rangle$ input state. The complexity of the algorithm depends on several factors: the degree $d$ of the target function approximation, the cost of block-encoding $A$, and the cost of preparing the initial state $|\phi\rangle$. Quantum Singular Value Transformation (QSVT) offers a powerful route to exponential speedups for a wide range of linear algebra problems. However, the actual performance gain depends critically on the structure of the problem and the output required. One of the most important applications of this algorithm is matrix inversion. Defining a function, $f = 1/x$, constrained over a domain of nonzero singular values, it is possible to efficiently calculate the polynomial approximation using QSVT. Moreover, QSVT is a core method in quantum machine learning, namely in the implementation of quantum principal component analysis and quantum support vector machines [64]. Additionally, a dequantized version was proposed which demonstrates that for low-degree polynomial transformations, QSVT can be efficiently simulated classically with arbitrarily small constant precision [66].

## 5. Quantum Machine Learning

### *5.1. Quantum Hopfield Networks*

One influential work in Artificial Intelligence is the associative memory model introduced by Hopfield [67]. The model is a neural network with fully connected neurons. The weights are symmetric and binary, meaning that the weight going from neuron $i$ to $j$ is the same as that from $j$ to $i$ with values $\{-1, 1\}$. The energy function is defined as follows

$$E = -\frac{1}{2}\sum_{i \neq j} w_{ij} s_i s_j + \sum_i \theta_i s_i, \tag{58}$$

where $w_{ij}$ is the weight between neurons $i$ and $j$, $s_i$ is the state of neuron $i$, and $\theta_i$ is the threshold for neuron $i$. The objective is to minimize the energy and drive the system into a stable configuration. The network learns using the Hebbian learning rule,

$$w_{ij} = \frac{1}{P}\sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu, \quad w_{ii} = 0, \tag{59}$$

where $P$ is the number of patterns to be stored, and $\xi_i^\mu$ is the $i$-th bit of the $\mu$-th pattern. It is proven that if there are $N$ neurons, the maximum number of storable patterns is approximately 0.138 $N$ [68]. Once patterns are selected, the Hebbian rule is applied to compute $w_{ij}$; then the network can recover a noisy version of a stored pattern. Quantum Hopfield networks extend the classical idea by encoding weights and neuron states in the amplitudes of quantum states [69]. The quantum Hebbian learning rule defines the weight matrix $W$ as

$$W = \frac{1}{P}\sum_{\mu=1}^{P} |\xi^\mu\rangle\langle\xi^\mu|, \tag{60}$$

where $|\xi^\mu\rangle$ are the quantum states representing patterns. A pattern state, $|\xi^\mu\rangle$, is prepared using the encoding

$$|\xi^\mu\rangle = \frac{1}{|\xi^\mu|}\sum_{i=1}^{d} \xi_i^\mu |i\rangle, \tag{61}$$

where $d$ is the length and $|\xi^\mu|$ denotes the Euclidean (L$_2$) norm of the pattern, defined as $|\xi^\mu| = \sqrt{\sum_{i=1}^{d}(\xi_i^\mu)^2}$. The pattern retrieval problem is reduced to solving the linear system $Ax = b$, where $A$ is constructed from $W$ and a regularization term, and $b$ is the input pattern vector. The solution vector $x$ approximates the stored pattern. This system is solved using the Harrow–Hassidim–Lloyd algorithm [70], involving quantum phase estimation, controlled rotations, and inverse phase estimation, to approximate $A^{-1}b$. Then the output state $|x\rangle$ is measured to retrieve the pattern. Due to the probabilistic nature of quantum measurement, multiple runs may be necessary for high-confidence recovery. The quantum algorithm has a runtime complexity of

$$\mathcal{O}(\text{poly}(P, \log d, 1/\epsilon, 1/\mu)), \tag{62}$$

where $\epsilon$ is the desired error tolerance, $\mu$ is the condition number of the matrix, and $d$ is the pattern size. For comparison, the best-known classical matrix inversion algorithm [71] has the complexity

$$\mathcal{O}(\text{poly}(d, 1/\sqrt{\mu}, \log(1/\epsilon), s)), \tag{63}$$

where $s$ is the sparsity of the matrix. Therefore, the quantum approach offers exponential speedup in the dimensionality $d$. Additionally, there are quantum versions with neuron embedding directly in qubits [72], using quantum search [73]. Moreover, the usage of quantum annealing was shown to address memory recall as an energy minimization task, where the stored patterns are mapped to low-energy levels [74].

### 5.2. Quantum Support Vector Machines (QSVMs)

Support vector machines (SVMs) are widely used supervised learning algorithms for classification and regression tasks [75]. The central idea is to find the optimal hyperplane in a given dataset that separates classes. Given the labeled samples $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, the algorithm constructs a hyperplane characterized by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ such that

$$y_i(w^T x_i + b) \geq 1 \quad \forall i. \tag{64}$$

The margin distance between the hyperplane and the nearest points (support vectors) is maximized by minimizing

$$\min_{w,b} \frac{1}{2}\|w\|^2. \tag{65}$$

This is known as the primal optimization form of the SVM, which can be applied to linearly separable data. If the data are nonlinear, the optimization problem becomes

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{66}$$

with the following constraints:

$$0 \leq \alpha_i \leq C, \quad \sum_i \alpha_i y_i = 0, \tag{67}$$

where $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ is the kernel function. The kernel function is used to map data into a higher dimensional space using a feature map, $\phi$, where separation may be possible. There are three major kernel families, namely, linear, $K(x, x') = x^T x'$, polynomial, $K(x, x') = (x^T x' + c)^d$, and Gaussian or radial base function (RBF), $K(x, x') = \exp(-\gamma\|x - x'\|^2)$ [76]. Quantum support vector machines (QSVMs) are defined based on quantum kernel methods to find solutions in high-dimensional feature spaces [77]. The kernel function is defined by the inner product of quantum states:

$$K(x_i, x_j) = |\langle \phi(x_i) | \phi(x_j) \rangle|^2. \tag{68}$$

The key advantage is that this kernel captures complex relationships in the data that might be inaccessible in the classical domain. The kernel measures the similarity between data points after encoding them into a quantum state using a quantum feature map. The classical input $x$ is encoded by applying the $U(x)$ circuit feature map such that

$$|\phi(x)\rangle = U_\phi(x)|0\rangle, \tag{69}$$

where $|0\rangle$ is the initial quantum state, and $U(x)$ is a parameterized circuit that encodes specific $x$ using data-dependent rotations and entanglement. $K(x_i, x_j)$ is computed using either the overlap test or the SWAP test [78]. The overlap test prepares the state

$$U^{\dagger}(x_i)U(x_j)|0\rangle, \tag{70}$$

and the probability of measuring $|0\rangle$ gives an estimate of the inner product $|\langle\phi(x_i)|\phi(x_j)\rangle|^2$. The circuit is repeated $R$ times to estimate the probability of returning to the $|0\rangle$ state. The SWAP test prepares identical quantum registers, $|\phi(x_i)\rangle$ and $|\phi(x_j)\rangle$, and applies the Hadamard $\rightarrow$ controlled-SWAP $\rightarrow$ Hadamard gate sequence to an ancilla qubit; then the ancilla is measured. Controlled-SWAP is a three-qubit gate that conditionally swaps the states of two target qubits depending on the state of a control qubit. In the case of multi-qubit quantum states, the CSWAP operation is applied in parallel across all corresponding qubit pairs. The goal is to fetch the probability of the ancilla qubit collapsing to $|0\rangle$. To estimate the probability, we iterate the circuit multiple times to obtain the count. Using the SWAP test identity

$$|\langle\phi(x_i)|\phi(x_j)\rangle|^2 = 2P(0) - 1, \tag{71}$$

the $K(x_i, x_j)$ value is defined. Finally, the SVM dual problem is solved using the quantum kernel classically, making this algorithm a hybrid, both classical and quantum. As the optimization problem solver is the same in both computations, in terms of complexity, the kernel estimation running time should be considered. For the classical version, depending on the kernel type, the complexity is $\mathcal{O}(n^2 \cdot k(d))$ where $d$ is the dimension of the feature space, and $k(d)$ is a function of cost for a specific kernel; it should be noted that most of the classical kernels are $\mathcal{O}(d)$. In quantum settings, the complexity is $\mathcal{O}(n^2 \cdot R \cdot d_q)$ where $R$ is the number of iterations for probability estimation and $d_q$ is the depth of a quantum circuit, which is usually polynomial in data size. The classical algorithm is efficient in the case of low-dimensional feature space and a small number of samples. In contrast, the potential advantage of the quantum version is that it can operate in high-dimensional spaces that may be computationally infeasible using classical methods. Another quantum SVM approach reformulates the SVM problem as least squares optimization, eventually reducing it to the task of solving a linear system of equations [79]. This system is then solved using the HHL quantum algorithm [70], allowing for potential exponential speedup under certain conditions.

## 6. Software Ecosystem

Quantum computing software provides a bridge between theoretical and practical applications. It is a critical layer in research and development of quantum algorithms which enables the development, testing, and execution of quantum algorithms on both simulated and physical quantum devices. Currently, advancements in quantum software development include high-level programing frameworks, low-level compilers, simulators, and domain-specific libraries, namely, machine learning, financial, and optimization problem libraries. Quantum software can generally be categorized based on its abstraction level and functional scope. At the highest level, there are software development kits (SDKs) that enable native scientific interfaces for building and executing quantum circuits. One widely used SDK is Qiskit, developed by IBM [14]. Qiskit provides various functionalities for quantum algorithm development including transpilation, simulation, and optimization. Qiskit supports the Python programming language interface and can be integrated with IBM's quantum hardware. It has a specialized module for machine learning algorithms that supports designing methods such as quantum neural networks, quantum kernel methods, and variational quantum algorithms. PennyLane is another high-level SDK, designed by Xanadu [80]. In contrast, PennyLane stresses a wider range of integrations, giving one interface for various simulators and quantum hardware vendors. The design and implementation are particularly well suited for machine learning applications supporting state-of-the-art machine learning methods. Another notable SDK is Cirq, developed by Google, which is optimized to support near-term quantum devices and provides full control over quantum gates [81]. Amazon Bracket offers an SDK that is capable of running on various backends, both simulated and real. In contrast to these high-level frameworks, low-level solutions offer more specific control over quantum circuit compilation [82]. ProjectQ is developed taking into account modular architecture that highlights compiler optimization [83]. Q# is a domain specific language which is integrated in the Azure Quantum Development Kit, supporting a strong type of checking and simulation environment [84]. Quantum simulators are an important layer in the development lifecycle of quantum algorithms by enabling the emulation of quantum behavior on classical hardware. Qiskit Aer is a quantum state vector simulator which simulates the evolution of quantum states with high accuracy using a noise-aware simulation technique which injects parameterized amounts of noise into a system that reflects real gate and measurement errors. ITensor is a C++ library for tensor network calculations, primarily designed for many efficient noise-free quantum body system simulations using tensor contraction techniques [85]. Another class of quantum software supports quantum annealing, which is a model focused on quantum tunneling to find low-energy configurations of a cost function used in optimization problems [86]. The most famous software ecosystem of quantum annealers is offered by D-Wave Systems. Their Ocean SDK provides Python tools for problem formulation, embedding, and execution on D-Wave quantum annealers. It is designed to support quadratic unconstrained binary optimization problems and hybrid quantum–classical pipelines. Quantum annealers are specifically useful for applications in logistics, finance, and machine learning such as clustering and sampling. Quantum machine learning encompasses quantum computing and machine learning algorithms that try to utilize quantum advantages in data processing and model training. Qiskit provides a dedicated module for machine learning that integrates essential tools for implementing quantum classifiers, regressors and other algorithms. PennyLane offers similar capabilities with more emphasis on optimization through interfaces with PyTorch and TensorFlow.

### *6.1. Case Study: QSVM vs. Classical SVM*

This case study explores the performance of a classical SVM versus a QSVM on various synthetic binary classification datasets, highlighting their capabilities in handling different data complexities. The datasets were generated using scikit-learn functions and split into training (80%) and testing (20%) sets. The linearly separable dataset consisted of 100 samples with two features, created with two centers and a cluster standard deviation of 1.0, resulting in well-separated clusters. The nonlinearly separable dataset had 100 samples and noise of 0.1 and was scaled by 10 for larger values, producing interlocking crescent shapes. The overlapping dataset employed 100 samples, two features, and two centers but a higher cluster standard deviation of 3.0, leading to significant class overlap. The high-dimensional dataset featured 100 samples across 10 dimensions (7 informative, 3 redundant), introducing complexity through higher feature space. Finally, the imbalanced dataset also featured 100 samples with two features but had the class weights [0.9, 0.1] and was scaled by 10, resulting in a majority class dominating a sparse minority (as illustrated in Figure 2).

The classical SVM pipeline leverages scikit-learn's SVC class. For linear and simple cases, it uses a linear kernel by default, but for nonlinear scenarios like moon-shaped data, it employs a Radial Basis Function (RBF) kernel to implicitly map data into a higher-dimensional space via the kernel trick, enabling curved decision boundaries. The process involves fitting the model using training data, predicting using test data, and evaluating metrics, namely, accuracy, precision, recall, and F1-score.

In contrast, the quantum SVM pipeline integrates quantum computing elements using Qiskit's tools on a simulated backend. Processing begins by scaling features to the [0, $\pi$] range to suit quantum encoding and then applies a ZZFeatureMap (with four repetitions) to embed classical data into a quantum Hilbert space [14,87]. (68) is used for state overlap estimation, a kernel matrix representing quantum similarities. This matrix is then fed into a classical SVC for training and prediction, mirroring the classical evaluation metrics. The key difference lies in the kernel computation; the classical RBF uses mathematical approximations in Euclidean space, while the QSVM exploits quantum superposition and entanglement for potentially more expressive feature maps in exponentially larger spaces, though it was simulated here classically, which limited true quantum advantages.

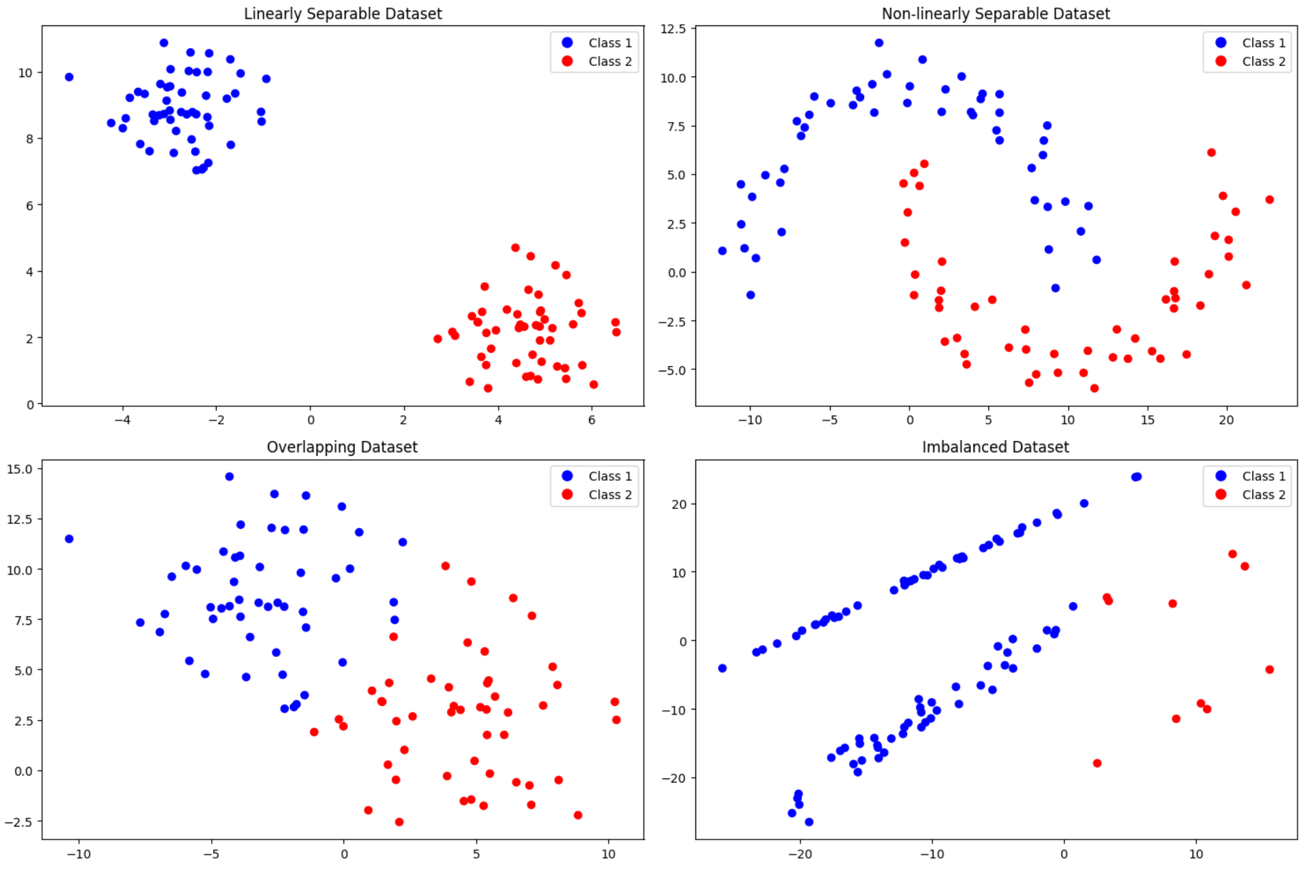

**Figure 2.** Plots of the generated datasets: linearly separable (**top left**), nonlinearly separable (**top right**), overlapping (**bottom left**), and imbalanced (**bottom right**).

Analyzing the results, the classical SVM consistently outperformed the QSVM across all datasets on the test sets (Table 1). For the linearly separable data, classical achieved perfect scores, while the QSVM dropped to 0.85 accuracy, possibly due to quantum encoding overhead introducing noise in simulation. On nonlinear data, classical again perfected at 1.0, leveraging the RBF effectively, but the QSVM yielded 0.80 accuracy with lower precision (0.625), indicating challenges in capturing nonlinearities via the ZZFeatureMap. Overlapping data resulted in classical being at 0.95 accuracy versus the QSVM's poor 0.45, highlighting the QSVM's sensitivity to noise and overlap. In high-dimensional space, classical managed 0.75 accuracy, but QSVM's plummeted to 0.30, struggling with the 10-feature encoding into qubits. For imbalanced data, classical perfected at 1.0, while the

QSVM (with class weights) reached 0.70 weighted accuracy, better handling minority recall but overall being inferior.

The classical SVM's advantages include robustness, efficiency on small datasets, and mature implementations, yielding high performance without specialized hardware, making it practical for real-world use. However, it may scale poorly in high dimensions due to computational costs of kernel matrices. The QSVM offers potential advantages in theoretically handling complex, high-dimensional data via quantum speedup, as seen in slightly better recall in some cases (e.g., 1.0 in high dimensions), suggesting promise for entangled patterns which are unfeasible classically. Disadvantages include simulation limitations introducing errors, higher computational demands (even when simulated), sensitivity to scaling and feature mapping choices, and current immaturity leading to lower metrics (0.3–0.85 accuracy), underscoring that true benefits may emerge only on fault-tolerant quantum hardware for larger datasets. Overall, while the classical SVM excels in this study, the QSVM hints at future potential for intractable problems, warranting further optimization like advanced feature maps or hybrid approaches.

**Table 1.** Dataset generation parameters and classification results.

| **Dataset** | **Generation Parameters** | | | **Classical SVM Metrics** | | | |
|---|---|---|---|---|---|---|---|
| | **Samples/Features** | **Key Params** | **Split (Train/Test)** | **Acc** | **Prec** | **Rec** | **F1** |
| Linearly Separable | 100/2 | centers = 2, cluster_std = 1.0 | 80/20 | 1.00 | 1.00 | 1.00 | 1.00 |
| Nonlinearly Separable | 100/2 | noise = 0.1, scaled ×10 | 80/20 | 1.00 | 1.00 | 1.00 | 1.00 |
| Overlapping | 100/2 | centers = 2, cluster_std = 3.0 | 80/20 | 0.95 | 0.90 | 1.00 | 0.94 |
| High-Dimensional | 100/10 | informative = 7, redundant = 3 | 80/20 | 0.75 | 0.54 | 1.00 | 0.70 |
| Imbalanced | 100/2 | weights = [0.9, 0.1], scaled ×10 | 80/20 (strat.) | 1.00 | 1.00 | 1.00 | 1.00 |
| Dataset | | | | Quantum SVM Metrics | | | |
| | | | | Acc | Prec | Rec | F1 |
| Linearly Separable | | | | 0.85 | 0.80 | 0.88 | 0.84 |
| Nonlinearly Separable | | | | 0.80 | 0.62 | 0.83 | 0.71 |
| Overlapping | | | | 0.45 | 0.37 | 0.33 | 0.35 |
| High-Dimensional | | | | 0.30 | 0.30 | 1.00 | 0.46 |
| Imbalanced | | | | 0.70 | 0.85 * | 0.70 * | 0.76 * |

* Weighted metrics for imbalanced dataset.

### *6.2. Limitations*

Quantum software faces numerous challenges. One of the core challenges is scalability. Current quantum devices are constrained in terms of qubit count, coherence times, and gate fidelity, which restricts the size and complexity of models that can be executed. Existing approaches can soften current hardware limitations while still using quantum components [88]. Another major field for development is parallel software and hardware design. Optimizing quantum circuits for specific hardware characteristics is essential to maximize performance. Advancements in quantum compilers, transpilers, and error mitigation techniques will be crucial in this regard [7].

## 7. Conclusions and Future Prospects

Quantum computing has made outstanding theoretical progress in computer science, with demonstrated advantages in tasks such us search, factorization, and quantum-enhanced machine learning. However, current quantum devices posses significant limitations. Among major problems are short qubit coherence time, low gate fidelity, and limited qubit systems. These limitations make scaling quantum algorithms to reliably solve practical problems extremely difficult. As a result, implementation of algorithms like Shor's still remain on paper. Other problematic aspects are the development of error correction codes and fault-tolerant quantum architectures. These problems remain active research topics as fault-tolerant computing requires a significantly large amount of physical qubits, which remains a hardware bottleneck.

While quantum computing provides (at least on paper) exponential speedups in particular problems, NP-complete problems still remain intractable in general. Current advancements suggest that quantum computing will not solve all computational problems but rather complement classical computing in hybrid systems. Such systems already show tangible results in optimization, chemistry, and machine learning.

In the future, it is likely that the standard of computational systems will be hybrid machines, where quantum processors act as accelerators for specific subroutines, while classical processors handle the overall orchestration and post-processing. This mixed relationship between classical and quantum devices will define the next era of computational innovation, enabling robust solutions in domains where neither system alone would be sufficient.

**Author Contributions:** Conceptualization: E.G. and S.K.; Methodology, E.G.; Software, E.G.; Formal Analysis, E.G. and P.R.P.; Validation, S.K. and P.R.P.; writing—original draft preparation, E.G. and S.K.; writing—review and editing, E.G., S.K. and P.R.P.; Supervision, S.K. and P.R.P.; Funding acquisition, P.R.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** The funding is provided by the National Council for Technological and Scientific Development (CNPq): 306435/2023–3.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The synthetic data will be made available on request.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Benioff, P. Quantum mechanical Hamiltonian models of Turing machines. *J. Stat. Phys.* **1982**, *29*, 515–546. [CrossRef]
2. Feynman, R.P. Simulating Physics with Computers. *Int. J. Theor. Phys.* **1982**, *21*, 467–488. [CrossRef]
3. Deutsch, D. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. London. A. Math. Phys. Sci.* **1985**, *400*, 97–117.
4. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
5. Barkoutsos, P.K.; Gonthier, J.F.; Sokolov, I.; Moll, N.; Salis, G.; Fuhrer, A.; Ganzhorn, M.; Egger, D.J.; Troyer, M.; Mezzacapo, A.; et al. Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions. *Phys. Rev. A* **2018**, *98*, 022322. [CrossRef]
6. Farhi, E.; Goldstone, J.; Gutmann, S. A quantum approximate optimization algorithm. *arXiv* **2014**, arXiv:1411.4028. [CrossRef]
7. Cerezo, M.; Arrasmith, A.; Babbush, R.; Benjamin, S.C.; Endo, S.; Fujii, K.; McClean, J.R.; Mitarai, K.; Yuan, X.; Cincio, L.; et al. Variational quantum algorithms. *Nat. Rev. Phys.* **2021**, *3*, 625–644. [CrossRef]
8. Main, D.; Drmota, P.; Nadlinger, D.; Ainley, E.; Agrawal, A.; Nichol, B.; Srinivas, R.; Araneda, G.; Lucas, D. Distributed quantum computing across an optical network link. *Nature* **2025**, *638*, 383–388. [CrossRef]
9. Gill, S.S.; Kumar, A.; Singh, H.; Singh, M.; Kaur, K.; Usman, M.; Buyya, R. Quantum computing: A taxonomy, systematic review and future directions. *Software Pract. Exp.* **2022**, *52*, 66–114. [CrossRef]
10. Gill, S.S.; Cetinkaya, O.; Marrone, S.; Claudino, D.; Haunschild, D.; Schlote, L.; Wu, H.; Ottaviani, C.; Liu, X.; Machupalli, S.P.; et al. Quantum computing: Vision and challenges. In *Quantum Computing*; Elsevier: Amsterdam, The Netherlands, 2025; pp. 19–42.
11. Paudel, H.P.; Syamlal, M.; Crawford, S.E.; Lee, Y.L.; Shugayev, R.A.; Lu, P.; Ohodnicki, P.R.; Mollot, D.; Duan, Y. Quantum computing and simulations for energy applications: Review and perspective. *ACS Eng. Au* **2022**, *2*, 151–196. [CrossRef]
12. Rietsche, R.; Dremel, C.; Bosch, S.; Steinacker, L.; Meckel, M.; Leimeister, J.M. Quantum computing. *Electron. Mark.* **2022**, *32*, 2525–2536. [CrossRef]
13. Putranto, D.S.C.; Wardhani, R.W.; Ji, J.; Kim, H. A Deep Inside Quantum Technology Industry Trends and Future Implications. *IEEE Access* **2024**, *12*, 115776–115801. [CrossRef]
14. Javadi-Abhari, A.; Treinish, M.; Krsulich, K.; Wood, C.J.; Lishman, J.; Gacon, J.; Martiel, S.; Nation, P.D.; Bishop, L.S.; Cross, A.W.; et al. Quantum computing with Qiskit. *arXiv* **2024**, arXiv:2405.08810. [CrossRef]

15. Griffiths, D.J.; Schroeter, D.F. *Introduction to Quantum Mechanics*; Cambridge University Press: Cambridge, UK, 2018.
16. Yanofsky, N.S.; Mannucci, M.A. *Quantum computing for Computer Scientists*; Cambridge University Press: Cambridge, UK, 2008.
17. Blais, A.; Grimsmo, A.L.; Girvin, S.M.; Wallraff, A. Circuit quantum electrodynamics. *Rev. Mod. Phys.* **2021**, *93*, 025005. [CrossRef]
18. Manzano, D. A short introduction to the Lindblad master equation. *Aip Adv.* **2020**, *10*, 025106. [CrossRef]
19. Coles, P.J.; Berta, M.; Tomamichel, M.; Wehner, S. Entropic uncertainty relations and their applications. *Rev. Mod. Phys.* **2017**, *89*, 015002. [CrossRef]
20. Caspani, L.; Xiong, C.; Eggleton, B.J.; Bajoni, D.; Liscidini, M.; Galli, M.; Morandotti, R.; Moss, D.J. Integrated sources of photon quantum states based on nonlinear optics. *Light. Sci. Appl.* **2017**, *6*, e17100. [CrossRef] [PubMed]
21. Zhang, C.; Huang, Y.F.; Liu, B.H.; Li, C.F.; Guo, G.C. Spontaneous parametric down-conversion sources for multiphoton experiments. *Adv. Quantum Technol.* **2021**, *4*, 2000132. [CrossRef]
22. Li, B.; Yuan, Z.; Williams, J.; Jin, W.; Beckert, A.; Xie, T.; Guo, J.; Feshali, A.; Paniccia, M.; Faraon, A.; et al. Down-converted photon pairs in a high-Q silicon nitride microresonator. *Nature* **2025**, *639*, 922–927. [CrossRef] [PubMed]
23. Krstić, A.; Setzpfandt, F.; Saravi, S. Nonperturbative theory of spontaneous parametric down-conversion in open and dispersive optical systems. *Phys. Rev. Res.* **2023**, *5*, 043228. [CrossRef]
24. Scala, F.; Nigro, D.; Gerace, D. Deterministic entangling gates with nonlinear quantum photonic interferometers. *Commun. Phys.* **2024**, *7*, 118. [CrossRef]
25. Zhang, Z.; Scully, M.O.; Agarwal, G.S. Quantum entanglement between two magnon modes via Kerr nonlinearity driven far from equilibrium. *Phys. Rev. Res.* **2019**, *1*, 023021. [CrossRef]
26. Massembele, D.K.; Djorwé, P.; Sarma, A.K.; Abdel-Aty, A.H.; Engo, S.N. Quantum entanglement assisted via Duffing nonlinearity. *Phys. Rev. A* **2024**, *110*, 043502. [CrossRef]
27. Lyu, X.; Kallioniemi, L.; Cai, H.; An, L.; Duan, R.; Wu, S.J.; Tan, Q.; Zhang, C.; He, R.; Miao, Y.; et al. Boosting classical and quantum nonlinear processes in ultrathin van der Waals materials. *Nat. Commun.* **2025**, *16*, 4987. [CrossRef]
28. Akin, J.; Zhao, Y.; Kwiat, P.G.; Goldschmidt, E.A.; Fang, K. Faithful quantum teleportation via a nanophotonic nonlinear Bell state analyzer. *Phys. Rev. Lett.* **2025**, *134*, 160802. [CrossRef] [PubMed]
29. Fabre, C.; Treps, N. Modes and states in quantum optics. *Rev. Mod. Phys.* **2020**, *92*, 035005. [CrossRef]
30. Wootters, W.K.; Zurek, W.H. A single quantum cannot be cloned. *Nature* **1982**, *299*, 802–803. [CrossRef]
31. Dieks, D. Communication by EPR devices. *Phys. Lett. A* **1982**, *92*, 271–272. [CrossRef]
32. Pan, D.; Long, G.L.; Yin, L.; Sheng, Y.B.; Ruan, D.; Ng, S.X.; Lu, J.; Hanzo, L. The evolution of quantum secure direct communication: On the road to the qinternet. *IEEE Commun. Surv. Tutorials* **2024**, *26*, 1898–1949. [CrossRef]
33. Zhou, L.; Xu, B.W.; Zhong, W.; Sheng, Y.B. Device-independent quantum secure direct communication with single-photon sources. *Phys. Rev. Appl.* **2023**, *19*, 014036. [CrossRef]
34. Zhao, P.; Zhong, W.; Du, M.M.; Li, X.Y.; Zhou, L.; Sheng, Y.B. Quantum secure direct communication with hybrid entanglement. *Front. Phys.* **2024**, *19*, 51201. [CrossRef]
35. Deutsch, D.E. Quantum computational networks. *Proc. R. Soc. London. A. Math. Phys. Sci.* **1989**, *425*, 73–90.
36. Landauer, R. Irreversibility and heat generation in the computing process. *IBM J. Res. Dev.* **1961**, *5*, 183–191. [CrossRef]
37. Born, M.; Fock, V. Beweis des adiabatensatzes. *Z. Für Phys.* **1928**, *51*, 165–180. [CrossRef]
38. Farhi, E.; Goldstone, J.; Gutmann, S.; Sipser, M. Quantum computation by adiabatic evolution. *arXiv* **2000**, arXiv:quant-ph/0001106.
39. Aharonov, D.; Van Dam, W.; Kempe, J.; Landau, Z.; Lloyd, S.; Regev, O. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Rev.* **2008**, *50*, 755–787. [CrossRef]
40. Sarovar, M.; Young, K.C. Error suppression and error correction in adiabatic quantum computation: Non-equilibrium dynamics. *New J. Phys.* **2013**, *15*, 125032. [CrossRef]
41. Albash, T.; Lidar, D.A. Adiabatic quantum computation. *Rev. Mod. Phys.* **2018**, *90*, 015002. [CrossRef]
42. Schiffer, B.F.; Tura, J.; Cirac, J.I. Adiabatic spectroscopy and a variational quantum adiabatic algorithm. *PRX Quantum* **2022**, *3*, 020347. [CrossRef]
43. Dooley, S.; Kells, G.; Katsura, H.; Dorlas, T.C. Simulating quantum circuits by adiabatic computation: Improved spectral gap bounds. *Phys. Rev. A* **2020**, *101*, 042302. [CrossRef]
44. Crosson, E.; Lidar, D. Prospects for quantum enhancement with diabatic quantum annealing. *Nat. Rev. Phys.* **2021**, *3*, 466–489. [CrossRef]
45. Hastings, M.B. Quantum adiabatic computation with a constant gap is not useful in one dimension. *Phys. Rev. Lett.* **2009**, *103*, 050502. [CrossRef]
46. Yao, J.; Lin, L.; Bukov, M. Reinforcement learning for many-body ground-state preparation inspired by counterdiabatic driving. *Phys. Rev. X* **2021**, *11*, 031070. [CrossRef]
47. Torrontegui, E.; Martínez-Garaot, S.; Ruschhaupt, A.; Muga, J.G. Shortcuts to adiabaticity: Fast-forward approach. *Phys. Rev. A Atomic Mol. Opt. Phys.* **2012**, *86*, 013601. [CrossRef]

48. Raussendorf, R.; Briegel, H.J. A one-way quantum computer. *Phys. Rev. Lett.* **2001**, *86*, 5188. [CrossRef] [PubMed]
49. Raussendorf, R.; Browne, D.E.; Briegel, H.J. Measurement-based quantum computation on cluster states. *Phys. Rev. A* **2003**, *68*, 022312. [CrossRef]
50. Wei, T.C. Measurement-based quantum computation. *arXiv* **2021**, arXiv:2109.10111.
51. Lanyon, B.; Jurcevic, P.; Zwerger, M.; Hempel, C.; Martinez, E.; Dür, W.; Briegel, H.; Blatt, R.; Roos, C.F. Measurement-based quantum computation with trapped ions. *Phys. Rev. Lett.* **2013**, *111*, 210501. [CrossRef]
52. Else, D.V.; Schwarz, I.; Bartlett, S.D.; Doherty, A.C. Symmetry-protected phases for measurement-based quantum computation. *Phys. Rev. Lett.* **2012**, *108*, 240505. [CrossRef]
53. Larsen, M.V.; Chamberland, C.; Noh, K.; Neergaard-Nielsen, J.S.; Andersen, U.L. Fault-tolerant continuous-variable measurement-based quantum computation architecture. *Prx Quantum* **2021**, *2*, 030325. [CrossRef]
54. Lee, S.H.; Jeong, H. Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states. *Phys. Rev. Res.* **2022**, *4*, 013010. [CrossRef]
55. Yamasaki, H.; Fukui, K.; Takeuchi, Y.; Tani, S.; Koashi, M. Polylog-overhead highly fault-tolerant measurement-based quantum computation: All-Gaussian implementation with Gottesman-Kitaev-Preskill code. *arXiv* **2020**, arXiv:2006.05416.
56. Majumder, A.; Krumm, M.; Radkohl, T.; Fiderer, L.J.; Nautrup, H.P.; Jerbi, S.; Briegel, H.J. Variational measurement-based quantum computation for generative modeling. *Phys. Rev. A* **2024**, *110*, 062616. [CrossRef]
57. Bernstein, E.; Vazirani, U. Quantum complexity theory. In Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, 16–18 May 1993; pp. 11–20.
58. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.
59. Brassard, G.; Hoyer, P.; Mosca, M.; Tapp, A. Quantum amplitude amplification and estimation. *arXiv* **2000**, arXiv:quant-ph/0005055.
60. Bennett, C.H.; Bernstein, E.; Brassard, G.; Vazirani, U. Strengths and weaknesses of quantum computing. *SIAM J. Comput.* **1997**, *26*, 1510–1523. [CrossRef]
61. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **1999**, *41*, 303–332. [CrossRef]
62. Kitaev, A.Y. Quantum computations: Algorithms and error correction. *Russ. Math. Surv.* **1997**, *52*, 1191. [CrossRef]
63. Gidney, C.; Ekerå, M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* **2021**, *5*, 433. [CrossRef]
64. Gilyén, A.; Su, Y.; Low, G.H.; Wiebe, N. Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, Phoenix, AZ, USA, 23–26 June 2019; pp. 193–204.
65. Haah, J. Product decomposition of periodic functions in quantum signal processing. *Quantum* **2019**, *3*, 190. [CrossRef]
66. Gharibian, S.; Le Gall, F. Dequantizing the quantum singular value transformation: Hardness and applications to quantum chemistry and the quantum PCP conjecture. In Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, 20–24 June 2022; pp. 19–32.
67. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA* **1982**, *79*, 2554–2558. [CrossRef] [PubMed]
68. Folli, V.; Leonetti, M.; Ruocco, G. On the maximum storage capacity of the Hopfield model. *Front. Comput. Neurosci.* **2017**, *10*, 144. [CrossRef] [PubMed]
69. Rebentrost, P.; Bromley, T.R.; Weedbrook, C.; Lloyd, S. Quantum Hopfield neural network. *Phys. Rev. A* **2018**, *98*, 042308. [CrossRef]
70. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [CrossRef] [PubMed]
71. Shewchuk, J.R. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*; Technical Report; Carnegie Mellon University: Pittsburgh, PA, USA, 1994.
72. Rotondo, P.; Marcuzzi, M.; Garrahan, J.P.; Lesanovsky, I.; Müller, M. Open quantum generalisation of Hopfield neural networks. *J. Phys. A Math. Theor.* **2018**, *51*, 115301. [CrossRef]
73. Ventura, D.; Martinez, T. Quantum associative memory with exponential capacity. In Proceedings of the 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227), Anchorage, AK, USA, 4–9 May 1998; Volume 1, pp. 509–513. [CrossRef]
74. Seddiqi, H.; Humble, T.S. Adiabatic quantum optimization for associative memory recall. *Front. Phys.* **2014**, *2*, 79. [CrossRef]
75. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
76. Hofmann, T.; Schölkopf, B.; Smola, A.J. *Kernel Methods in Machine Learning*; Cambridge University Press: Cambridge, UK, 2008.

77. Havlíček, V.; Córcoles, A.D.; Temme, K.; Harrow, A.W.; Kandala, A.; Chow, J.M.; Gambetta, J.M. Supervised learning with quantum-enhanced feature spaces. *Nature* **2019**, *567*, 209–212. [CrossRef] [PubMed]
78. Buhrman, H.; Cleve, R.; Watrous, J.; De Wolf, R. Quantum fingerprinting. *Phys. Rev. Lett.* **2001**, *87*, 167902. [CrossRef]
79. Rebentrost, P.; Mohseni, M.; Lloyd, S. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **2014**, *113*, 130503. [CrossRef]
80. Bergholm, V.; Izaac, J.; Schuld, M.; Gogolin, C.; Ahmed, S.; Ajith, V.; Alam, M.S.; Alonso-Linaje, G.; AkashNarayanan, B.; Asadi, A.; et al. PennyLane: Automatic differentiation of hybrid quantum-classical computations. *arXiv* **2022**, arXiv:1811.04968.
81. Developers, C. *Cirq*; Zenodo: Genève, Switzerland, 2025. [CrossRef]
82. Häner, T.; Steiger, D.S.; Svore, K.; Troyer, M. A software methodology for compiling quantum programs. *Quantum Sci. Technol.* **2018**, *3*, 020501. [CrossRef]
83. Steiger, D.S.; Häner, T.; Troyer, M. ProjectQ: An open source software framework for quantum computing. *Quantum* **2018**, *2*, 49. [CrossRef]
84. Svore, K.; Geller, A.; Troyer, M.; Azariah, J.; Granade, C.; Heim, B.; Kliuchnikov, V.; Mykhailova, M.; Paz, A.; Roetteler, M. Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL. In Proceedings of the Real World Domain Specific Languages Workshop 2018, Vienna, Austria, 24 February 2018; p. RWDSL2018. [CrossRef]
85. Fishman, M.; White, S.R.; Stoudenmire, E.M. The ITensor Software Library for Tensor Network Calculations. *arXiv* **2020**, arXiv:2007.14822. [CrossRef]
86. Kadowaki, T.; Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev. E* **1998**, *58*, 5355. [CrossRef]
87. Schuld, M.; Killoran, N. Quantum machine learning in feature Hilbert spaces. *Phys. Rev. Lett.* **2019**, *122*, 040504. [CrossRef] [PubMed]
88. Farhi, E.; Neven, H. Classification with quantum neural networks on near term processors. *arXiv* **2018**, arXiv:1802.06002. [CrossRef]