entropy

IMPACT FACTOR 2.1

Indexed in: PubMed

CITESCORE 4.9

Article

Quantum Machine Learning—Quo Vadis?

Andreas Wichert

Special Issue

The Future of Quantum Machine Learning and Quantum AI

Edited by

Prof. Dr. Andreas (Andrzej) Wichert

MDPI

https://doi.org/10.3390/e26110905

entropy

MDPI

Article

Quantum Machine Learning—Quo Vadis?

Andreas Wichert

Department of Computer Science and Engineering, INESC-ID & Instituto Superior Técnico, University of Lisbon, 2744-016 Porto Salvo, Portugal; andreas.wichert@tecnico.ulisboa.pt

Abstract: The book *Quantum Machine Learning: What Quantum Computing Means to Data Mining*, by Peter Wittek, made quantum machine learning popular to a wider audience. The promise of quantum machine learning for big data is that it will lead to new applications due to the exponential speed-up and the possibility of compressed data representation. However, can we really apply quantum machine learning for real-world applications? What are the advantages of quantum machine learning algorithms in addition to some proposed artificial problems? Is the promised exponential or quadratic speed-up realistic, assuming that real quantum computers exist? Quantum machine learning is based on statistical machine learning. We cannot port the classical algorithms directly into quantum algorithms due to quantum physical constraints, like the input–output problem or the normalized representation of vectors. Theoretical speed-ups of quantum machine learning are usually analyzed in the literature by ignoring the input destruction problem, which is the main bottleneck for data encoding. The dilemma results from the following question: should we ignore or marginalize those constraints or not?

Keywords: quantum machine learning; basis encoding; amplitude encoding; input destruction problem; HHL; quantum kernels; variational algorithm

1. Introduction

Deep learning has achieved tremendous successes; it is based on error minimizing algorithms that approximate a distribution of a population. The approximation is based on the error minimization of the prediction of a very large training sample, with the assumption that the large sample describes the population sufficiently well. The error minimization is based on a loss function and the back-propagation algorithm. The back-propagation algorithm applied to deep learning architectures requires huge computational resources in hardware, energy, and time. The promise of quantum machine learning is that it will overcome these problems due to quadratic or even exponential speed-up in time and the possibility of compressed data representation. However, can we really apply quantum machine learning for real-world applications? What are the algorithmic constraints of quantum machine learning? Currently there is a huge body of literature on quantum machine learning, which we are unable to review. Instead we will deal with four fundamental categories of quantum machine learning including quantum encoding.

- The process of transferring classical data into quantum states is an essential step in quantum algorithms. This process is called quantum encoding or quantum state preparation. Binary patterns representation of a training sample using quantum encoding and the application of the Grover's algorithm is presented. It leads to quadratic speed-up of popular machine learning algorithms, like k-nearest neighbor, clustering, and associative memory.
- We describe the representation of a training sample by amplitude encoding. It is used in the quantum algorithm for linear systems of equations (HHL) based on Kitaev's phase estimation algorithm leading to an exponential speed-up. Since almost all machine learning algorithms use some form of a linear system of equations, it is assumed that the HHL algorithm is going to be one of the most useful subroutines.

check for updates

Citation: Wichert, A. Quantum Machine Learning—Quo Vadis? *Entropy* **2024**, *26*, 905. https://doi.org/10.3390/e26110905

Academic Editors: Osamu Hirota and Giuliano Benenti

Received: 13 September 2024
Revised: 14 October 2024
Accepted: 23 October 2024
Published: 24 October 2024

- Quantum kernels that are not based on the kernel trick but map the vectors directly into high-dimensional space and may lead to new kernel functions.
- Variational approaches are characterized using a classical optimization algorithm to iteratively update a parameterized quantum trial solution that can open new insights in real-world problems, like the study of the behavior of complex physical systems that cannot be tackled with classical machine learning algorithms.

2. Binary Patterns and the Grover's Algorithm

Basis encoding encodes a n dimensional binary vector to a n-qubit quantum basis state. Ventura and Martinez [1,2] proposed a method to encode m binary linear independent vectors with dimension n (with $n > m$) into a superposition of a n-qubit quantum state. We describe the simplified method by [3]. The procedure successively divides a present superposition into processing and memory branches. The input patterns are loaded into new generated memory branches step by step. The cost of the method is linear in the number of stored patterns and their dimension [4]. At the initial step, the system is in the basis state with load qubits, memory qubits and the control qubits c_1, c_2

$$|memory; c_2, c_1; load\rangle$$

(using little endian notation). The basis states are split step by step using the control qubits c_1, c_2 until the required superposition is present

$$\frac{1}{\sqrt{m}} \sum_{j=1}^{m} |memory; c_2, c_1; load\rangle_j$$

and with the memory register in the required superposition

$$\frac{1}{\sqrt{m}} \sum_{j=1}^{m} |memory; 0, 0; 0 \cdots 0\rangle_j = \left(\frac{1}{\sqrt{m}} \sum_{j=1}^{m} |memory\rangle_j \right) \otimes |0, 0; 0 \cdots 0\rangle.$$

The processing branch is indicated by the control qubit c_2 with the value one ($c_2 = 1$) and the memory branch representing the current superposition with the control qubit c_2 with the value zero ($c_2 = 0$). The the qubit c_2 is split by the operator CS_p represented by the parametrized U gate $U(\theta, \phi, \lambda) =$ with $\phi = \pi, \lambda = \pi$, and $\theta = \arcsin\left(\frac{1}{\sqrt{p}}\right) \cdot 2$

$$CS_p = CU(\arcsin\left(\frac{1}{\sqrt{p}}\right) \cdot 2, \pi, \pi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sqrt{\frac{p-1}{p}} & \frac{1}{\sqrt{p}} \\ 0 & 0 & \frac{-1}{\sqrt{p}} & \sqrt{\frac{p-1}{p}} \end{pmatrix}$$

with $CS_p|c_2, c_1\rangle$

$$CS_p|01\rangle = |01\rangle, \quad CS_p|11\rangle = \frac{1}{\sqrt{p}} \cdot |10\rangle + \sqrt{\frac{p-1}{p}} \cdot |11\rangle.$$

The control qubit c_1 indicates the split of the qubit c_2. Since the control qubit $c_1 = 1$ is entangled with the memory register we create the memory branch ($c_2 = 0$) with $\frac{1}{\sqrt{p}} \cdot |memory; 01\rangle$ and processing branch ($c_2 = 1$) $\sqrt{\frac{p-1}{p}} \cdot |memory; 11\rangle$ by the split operation on the preceding processing branch. A new pattern is stored in the generated memory register of the new generated memory branch. We repeat the procedure until we arrive in the final state

$$|\psi\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} |memory; 0, 0; 0 \cdots 0\rangle_j$$

representing the binary patterns in superposition of m basis states. In Figure 1 we indicate a circuit for the preparation of the three states $|01\rangle, |10\rangle, |11\rangle$.

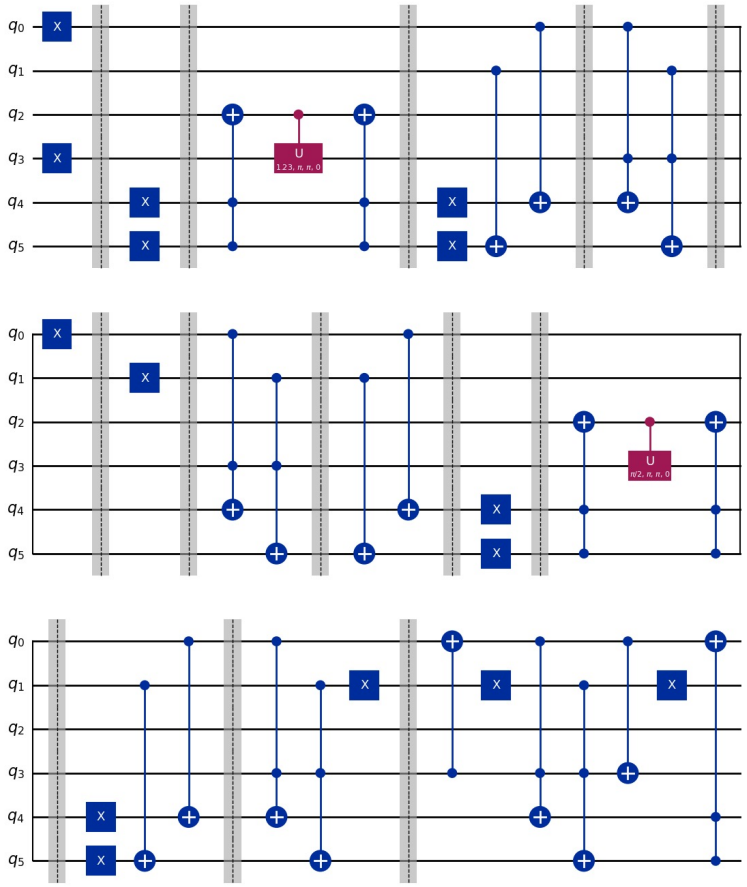

Figure 1. In this example, we store three binary patterns, $|01\rangle, |10\rangle, |11\rangle$. After applying the circuit the resulting state is $\frac{1}{\sqrt{3}}|0,0;1,0;0,0\rangle + \frac{1}{\sqrt{3}}|0,1;0,0;0,0\rangle + \frac{1}{\sqrt{3}}|1,0;0,0;0,0\rangle$. For more details see [5]. The dashed lines are visual indicators of the grouping of a circuit sections.

We can apply Grover's algorithm with a quadratic speed-up of $\sqrt{m}$, since the represented m basis states are known to us. The constructed Grover operator amplifies only the m basis states and the $n - m$ states with the amplitudes zero are unchanged.

Alternatively, we could entangle the index qubits that are in the superposition with the binary patterns.

$$|pattern_m\rangle, |pattern_{m-1}\rangle, \cdots |pattern_1\rangle.$$

To store m binary patterns we entangle the index qubits using multi-controlled NOT gates by generating $v = log_2(m)$ index qubits using v Hadamard gates

$$H^{\otimes v}|0\rangle^{\otimes v} = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} |index_j\rangle.$$

and entangle m binary pattern with the index qubits with a resulting superposition

$$\frac{1}{\sqrt{m}} \left(\sum_{j=1}^{m} |index_j\rangle |pattern_j\rangle \right). \tag{1}$$

Using an oracle $o()$ we can mark the corresponding index state and un-compute the entangled patterns resulting in a state

$$|\psi\rangle = \frac{1}{\sqrt{m}} \left(\sum_{j=1}^{m} (-1)^{o(index_j)} \cdot |index_j\rangle |0 \cdots 0\rangle \right). \tag{2}$$

We can now apply Grover's algorithm with a quadratic speed-up of $\sqrt{m}$ to the index states that point to our binary patterns.

2.1. Input Destruction Problem

The naïve assumption that the speed-up is quadratic is not realistic due to the input destruction problem (ID problem) [6–9]:

- The input (reading) problem: The quantum state is initialized by state preparation of m binary patterns. Although the existing quantum algorithm requires only $\sqrt{m}$ steps and is faster than the classical algorithms, m data points must be read—prepared. Hence, the complexity of the algorithm does not decrease.
- The destruction problem: We are required to read m data points and are allowed to query only once because of the collapse during the measurement (destruction).

Additionally, the output quantum state in many quantum machine learning algorithms must be fully read, but measuring the state collapses it to a single value, requiring many measurements to interpret the full state.

2.2. Quantum Random Access Memory

To avoid input destruction problem [10] quantum random access memory ($qRAM$) [11] was proposed. A $qRAM$ access basis states by copy operation [11]. A register $|i\rangle$ is queried and the ith binary patter is loaded into the second register

$$|i\rangle |0\rangle \rightarrow |i\rangle |x^i\rangle, \tag{3}$$

with $|x^i\rangle$ being a basis state representing a binary vector. The query operation can be executed in parallel with

$$\frac{1}{\sqrt{m}} \sum_{i=1}^{m} |i\rangle |0\rangle \rightarrow \frac{1}{\sqrt{m}} \sum_{i=1}^{m} |i\rangle |x^i\rangle, \tag{4}$$

with the time complexity ignoring the preparation cost of (due to the input problem) is $O(\log(m))$. The $qRAM$ suffers from the input destruction problem. Its usage leads to a circular argument.

3. Amplitude Encoding and the HHL Algorithm

Amplitude encoding encodes data into the amplitudes ω_i of a quantum state

$$|\psi\rangle = \sum_{i=1}^{N} \omega_i \cdot |x\rangle. \tag{5}$$

For example, a complex normalized vector $\mathbf{x}$

$$\mathbf{x} = \begin{pmatrix} \sqrt{0.03} \\ \sqrt{0.07} \\ \sqrt{0.15} \\ \sqrt{0.05} \\ \sqrt{0.1} \\ \sqrt{0.3} \\ \sqrt{0.2} \\ \sqrt{0.1} \end{pmatrix}.$$

is represented as

$$|\psi\rangle = \sqrt{0.03} \cdot |000\rangle + \sqrt{0.07} \cdot |001\rangle + \sqrt{0.15} \cdot |010\rangle + \sqrt{0.05} \cdot |011\rangle +$$
$$+\sqrt{0.1} \cdot |100\rangle + \sqrt{0.3} \cdot |101\rangle + \sqrt{0.2} \cdot |110\rangle + \sqrt{0.1} \cdot |111\rangle$$

by a top-down divide strategy using parametrized rotation gates with a linear complexity. Amplitude coding can only represent normalized vectors and also suffers from the input destruction problem.

3.1. Quantum Random Access Memory for Amplitude Coding

An operation that would produce a copy of an arbitrary quantum state such as $|\psi\rangle$ is not possible; we cannot copy non-basis states because of the linearity of quantum mechanics. However, we can, to some extent, simulate the copy of non-basis states using quantum random access memories ($qRAM$) as proposed by [9]. The resulting complexity would be $O(n \log(m))$ where n is the dimension of the resulting superposition vector [9]. We divide the binary vector of the dimension 2^m into v substrings; each substring $code_i$ represents a real number by fractional representation (binary representation of a real number)

$$|code_1 code_2 \cdots code_n\rangle \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle. \tag{6}$$

with $n = 2^m/v$ and a fractional real number α_i that is smaller than one

$$code_i = \alpha_i < 1.$$

We add an auxiliary state $|0^{\otimes n}\rangle$

$$|code_1 \cdots code_n\rangle \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle \rightarrow |code_1 \cdots code_n\rangle \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle |0^{\otimes n}\rangle \tag{7}$$

For each α_i we preform a controlled rotation $R(\alpha_i)$

$$\left(C \cdot \alpha_i |1\rangle + \sqrt{1 - C^2 \cdot \alpha_i^2} |0\rangle \right)$$

and measure. By measuring the corresponding auxiliary register with the result $|1\rangle$ we know that the resulting state is correct. However, if we measure in auxiliary register $|0\rangle$, we have to repeat the whole procedure. The success rate of measuring n $|1\rangle$ is very low (0.5^n) and converges to zero for large n. For large n it is simply not feasible. If we succeed, the resulting state would be

$$\sum_{i=1}^{n} \alpha_i |i\rangle. \tag{8}$$

The described routine is non reversible since it is based on measurement. If the probability of success was not very low, the complexity reading m vectors of dimension n would be $O(n\log(m))$ ignoring the preparation costs, compared to $O(n \cdot m)$ on a classical RAM. However, the $qRAM$ for amplitude coding is not feasible and does not solve the input destruction problem.

3.2. Quantum Algorithm for Linear Systems of Equations

Systems of linear equations can be solved by Gauss elimination with $O(n^3)$. The approximate solution for a sparse matrix via conjugate gradient descent requires much lower costs, $\tilde{O}(n)$ [12].

By ignoring certain constraints, the quantum algorithm for linear systems of equations on a quantum computer is exponentially faster for sparse matrices than any algorithm that solves linear systems on the classical computer [10]. It is based on amplitude coding and is called the HHL algorithm according to its inventors Aram Harrow, Avinatan Hassidim, and Seth Lloyd. For an invertible complex matrix $n \times n$ A and a complex vector $\mathbf{b}$

$$A \cdot \mathbf{x} = \mathbf{b} \tag{9}$$

we want to find $\mathbf{x}$. The following constraints are present.

- The vectors $|b\rangle$ and $|x\rangle$ represented by $\log_2 n$ qubits have a length of one in the l_2 norm with
$$|b\rangle = \frac{\sum_{i=1}^{n} b_i|i\rangle}{\|\sum_{i=1}^{n} b_i|i\rangle\|} \tag{10}$$
and
$$|x\rangle = \frac{\sum_{i=1}^{n} x_i|i\rangle}{\|\sum_{i=1}^{n} x_i|i\rangle\|}. \tag{11}$$
- $|b\rangle$ has to be prepared efficiently with the cost no bigger than $\log(n)$.
- The matrix A is sparse.
- For the output we are interested in the global properties of $|x\rangle$ rather than the coefficients x_i.

If A is Hermitian $A^* = A$ (for real matrix $A^T = A$) then A can be represented by the spectral decomposition as

$$A = \lambda_1 \cdot |u_1\rangle\langle u_1| + \lambda_2 \cdot |u_2\rangle\langle u_2| + \cdots + \lambda_n \cdot |u_n\rangle\langle u_n|. \tag{12}$$

and

$$A^{-1} = \frac{1}{\lambda_1} \cdot |u_1\rangle\langle u_1| + \frac{1}{\lambda_2} \cdot |u_2\rangle\langle u_2| + \cdots + \frac{1}{\lambda_n} \cdot |u_n\rangle\langle u_n|. \tag{13}$$

It follows

$$A^{-1} \cdot |u_j\rangle = \frac{1}{\lambda_j}|u_j\rangle \tag{14}$$

and writing $|b\rangle$ as a linear combination of the eigenvectors of A

$$|b\rangle = \sum_j |u_j\rangle\langle u_j|b\rangle \tag{15}$$

leads to

$$A \cdot |b\rangle = \sum_j \lambda_j|u_j\rangle\langle u_j|b\rangle \tag{16}$$

and

$$|x\rangle = A^{-1} \cdot |b\rangle = \sum_j \lambda_j^{-1}|u_j\rangle\langle u_j|b\rangle. \tag{17}$$

We estimate the eigenvalues using Kitaev's phase estimation algorithm. We then estimate the unknown eigenvalue $e^{2\cdot\pi\cdot i\cdot\theta_j}$. If we apply U to $|u_j\rangle$ we obtain

$$U \cdot |u\rangle = e^{2\cdot\pi\cdot i\cdot\theta_j} \cdot |u_j\rangle = e^{i\cdot\lambda_j} \cdot |u_j\rangle. \tag{18}$$

This representation is similar to the evolutionary operator $U_t = e^{-i\cdot t\cdot H}$ for $t = 1$ and $H := A$ is the Hamiltonian operator. The process of implementing a given Hamiltonian evolution on a quantum computer is called Hamiltonian simulation [13]. The challenge in Hamiltonian simulation is due to the fact that the application of matrix exponentials is computationally expensive [4]. The dimension of the Hilbert space grows exponentially with the number of qubits, and thus any operator will be of exponential dimension. The computation of the matrix exponential is difficult, and this is still a topic of considerable current research. Only for a sparse hermitian matrix H, the Hamiltonian simulation can be implemented efficiently. We do not need to know the eigenvector $|u_j\rangle$ of U. Since a quantum state $|b\rangle$ can be decomposed into an orthogonal basis

$$|b\rangle = \sum_j |u_j\rangle\langle u_j|b\rangle = \sum_j \beta_j |u_j\rangle. \tag{19}$$

After applying Kitaev's phase estimation algorithm to U that we estimated by the Hamiltonian simulation for each value j the values $\tilde{\lambda}_{k|j}$ approximate the true value λ_j. For simplicity, we assume

$$\sum_{j=1}^{n} \beta_j \sum_{k=0}^{T-1} \alpha_{k|j} \cdot |\tilde{\lambda}_{k|j}|u_j\rangle \approx \sum_{j=1}^{n} \beta_j \cdot |\tilde{\lambda}_j\rangle|u_j\rangle. \tag{20}$$

We have to measure the corresponding eigenvalues, so that we would be able to define a circuit that performs the conditioned rotation. To conduct the conditional rotation we add an auxiliary state $|0\rangle$

$$\sum_{j=1}^{n} \beta_j \cdot |\tilde{\lambda}_j\rangle|u_j\rangle|0\rangle \tag{21}$$

and perform the conditioned rotation on the auxiliary state $|0\rangle$ by the operator R

$$R = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}. \tag{22}$$

with the relation

$$\alpha = \arccos\left(\frac{C}{\tilde{\lambda}}\right) \tag{23}$$

with C being a constant of normalization. Each eigenvalue indicates a special rotation

$$\sum_{j=1}^{n} \left(\beta_j \cdot |\tilde{\lambda}_j\rangle|u_j\rangle\left(R\left(\tilde{\lambda}_j^{-1}\right)|0\rangle)\right)\right) =$$

$$\sum_{j=1}^{n} \left(\beta_j \cdot |\tilde{\lambda}_j\rangle|u_j\rangle\left(\frac{C}{\tilde{\lambda}_j}|1\rangle + \sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}}|0\rangle\right)\right). \tag{24}$$

We un-compute the phase estimation procedure, resulting in the state

$$|0\rangle \sum_{j=1}^{n} \left(\beta_j \cdot |u_j\rangle\left(\frac{C}{\tilde{\lambda}_j}|1\rangle + \sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}}|0\rangle\right)\right) =$$

$$|0\rangle \sum_{j=1}^{n} \left(C \cdot \frac{\beta_j}{\tilde{\lambda}_j}|u_j\rangle|1\rangle + \sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}} \cdot \beta_j|u_j\rangle|0\rangle\right). \tag{25}$$

By measuring the auxiliary qubit with the result 0 we obtain

$$|0\rangle \sum_{j=1}^{n} \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} \cdot \beta_j |u_j\rangle \right) \tag{26}$$

and with the result 1

$$C \cdot |0\rangle \sum_{j=1}^{n} \left(\frac{\beta_j}{\tilde{\lambda}_j} |u_j\rangle \right) = C \cdot |0\rangle A^{-1} |b\rangle \approx C \cdot |0\rangle |x\rangle. \tag{27}$$

We have to select the outcome of the measurement 1

$$|x\rangle = A^{-1}|b\rangle = \sum_{j=1}^{n} \frac{\beta_j}{\tilde{\lambda}_j} |u_j\rangle \tag{28}$$

which requires several measurements. After the preceding measurements we cannot obtain the solution $|x\rangle$ efficiently. Obtaining the required coefficient x_i from $|x\rangle$ would require at least n measurements, so the complexity of the algorithm would be $O(n)$ which is the same cost for an approximate solution for a sparse matrix via conjugate gradient descent on a classical computer.

For example,

the solution to the problem

$$A = \begin{pmatrix} 1 & -\frac{1}{3} \\ -\frac{1}{3} & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

is represented by

$$A^{-1} = \begin{pmatrix} \frac{9}{8} & \frac{3}{8} \\ \frac{3}{8} & \frac{9}{8} \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} \frac{9}{8} \\ \frac{3}{8} \end{pmatrix}.$$

with the normalized vector being

$$\mathbf{x}_n = \frac{\mathbf{x}}{\|\mathbf{x}\|} = \begin{pmatrix} 0.948683 \\ 0.316228 \end{pmatrix}.$$

We represent

$$|b\rangle = |0\rangle = \mathbf{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

with

$$|u_1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}, \quad |u_2\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

and with

$$|b\rangle = \sum_j \beta_j |u_j\rangle = |0\rangle = \frac{1}{\sqrt{2}} \cdot |u_1\rangle + \frac{1}{\sqrt{2}} \cdot |u_2\rangle$$

$$|b\rangle = \frac{1}{\sqrt{2}} \cdot \frac{|0\rangle - |1\rangle}{\sqrt{2}} + \frac{1}{\sqrt{2}} \cdot \frac{|0\rangle + |1\rangle}{\sqrt{2}} = |0\rangle.$$

We perform conditioned rotation on the auxiliary state $|0\rangle$ by

$$R_Y(\alpha) = \begin{pmatrix} \cos\left(\frac{\alpha}{2}\right) & -\sin\left(\frac{\alpha}{2}\right) \\ \sin\left(\frac{\alpha}{2}\right) & \cos\left(\frac{\alpha}{2}\right) \end{pmatrix}$$

with measured two control qubits being $|10\rangle$ representing $\tilde{\lambda}_1 = 2$ and $|01\rangle$ representing $\tilde{\lambda}_2 = 1$,

$$\alpha_1 = 2 \cdot \arccos\left(\frac{1}{\tilde{\lambda}_1}\right) = 2 \cdot \arccos\left(\frac{1}{2}\right) = \frac{\pi}{3} \tag{29}$$

$$\alpha_2 = 2 \cdot \arccos\left(\frac{1}{\tilde{\lambda}_2}\right)) = 2 \cdot \arccos\left(\frac{1}{1}\right) = \pi \tag{30}$$

The measured estimated probability value (requiring several measurements) of the HHL simulation are 0.562 and 0.0622 resulting in

$$\mathbf{x}_m^2 = \begin{pmatrix} \frac{0.56}{0.562+0.0622} \\ \frac{0.0622}{0.562+0.062} \end{pmatrix}$$

with

$$\mathbf{x}_m = \begin{pmatrix} 0.948869 \\ 0.31567 \end{pmatrix} \approx \mathbf{x}_n = \begin{pmatrix} 0.948683 \\ 0.316228 \end{pmatrix}.$$

4. Quantum Kernels

To avoid the input destruction problem quantum kernels were proposed. A quantum computer can estimate a quantum kernel and the estimate can be used by a kernel method on a classical computer, [4]. Using a quantum computer results in an exponential advantage in evaluating inner products, allowing us to estimate the quantum kernel directly in the higher dimensional space by a function $\phi(\mathbf{x})$ with

$$k(\mathbf{x}, \mathbf{y}) = |\langle \phi(\mathbf{x}) | \phi(\mathbf{y}) \rangle|^2. \tag{31}$$

For large high-dimensional space, such a procedure is not tractable on a classic computer [14]. However, classically, we do not compute the function $\phi(\mathbf{x})$; instead we compute inner products specified by the kernel $k(\mathbf{x}, \mathbf{y})$

$$k(\mathbf{x}, \mathbf{y}) = \Phi^T(\mathbf{x})\Phi(\mathbf{y}) = \langle \Phi(\mathbf{x}) | \Phi(\mathbf{y}) \rangle. \tag{32}$$

and the feature space could be of infinite dimensionality

$$k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}) | \Phi(\mathbf{y}) \rangle = \sum_{j=1}^{\infty} \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}). \tag{33}$$

For quantum kernels the dimension of the feature space is finite, since we map the vectors directly with the aid of a quantum computer and do not specify the kernel function. The classical data are encoded into quantum data by quantum feature maps via a parametrized quantum circuit [15]. The feature vector defines by m parameters of the parametrized quantum circuit $U_{\phi(\mathbf{x})}$

$$|\phi(\mathbf{x})\rangle = U_{\phi(\mathbf{x})}|0\rangle^{\otimes m} \tag{34}$$

with the dimension of $\phi(\mathbf{x})$ being 2^m. If we map for an input state $|0\rangle^{\otimes m}$ with a parametrized quantum circuit $U_{\phi(\mathbf{x})}$ with parameters that are defined by $\mathbf{x}$ and un-compute it by $U^{\dagger}_{\phi(\mathbf{x})}$, the inverse of the parametrized quantum circuit $U_{\phi(\mathbf{x})}$, then the probability of measuring the state $|0\rangle^{\otimes m}$ is one. If we parametrize the quantum circuit U by $\mathbf{x}$ ($U_{\phi(\mathbf{x})}$) and the inverse of the parametrized quantum $U^{\dagger}$ by $\mathbf{y}$ ($U^{\dagger}_{\phi(\mathbf{y})}$) and if $\mathbf{x}$ and $\mathbf{y}$ are similar, the probability of measuring $|0\rangle^{\otimes m}$ for the input $|0\rangle^{\otimes m}$

$$U^{\dagger}_{\phi(\mathbf{y})} U_{\phi(\mathbf{x})} |0^{\otimes m}\rangle \tag{35}$$

should be near 1. If $\mathbf{x}$ and $\mathbf{y}$ differ a lot, this probability is smaller. The quantum kernel is represented after measurement as

$$k(\mathbf{x}, \mathbf{y}) = |\langle \phi(\mathbf{x}) | \phi(\mathbf{y}) \rangle|^2 = |\langle 0^{\otimes m} | U^{\dagger}_{\phi(\mathbf{y})} | U_{\phi(\mathbf{x})} | 0^{\otimes m} \rangle|^2 \tag{36}$$

We have to measure the final state several times and record the number of $|0^{\otimes m}\rangle$ and estimate the value $k(\mathbf{x}, \mathbf{y})$. The parametrized quantum circuit is based on superposition and entanglement and rotation gates that map the feature vectors in a periodic feature space resulting in periodic receptive fields with a fixed center. Quantum kernels can achieve lower training error on real data. However, this improvement leads to poor generalization on the test set and classical models can outperform quantum models [16] due to the periodic receptive fields with a fixed center, see Figure 2.

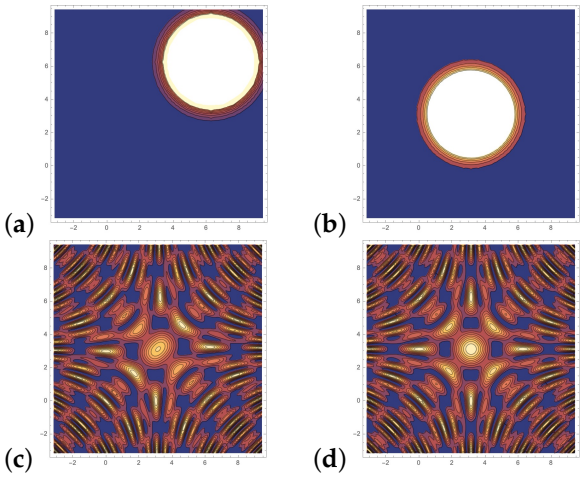

Figure 2. (**a**) RBF kernel with the center $\mathbf{y} = (2 \cdot \pi, 2 \cdot \pi)^T$. (**b**) RBF kernel with the center $\mathbf{y} = (\pi, \pi)^T$. (**c**) Quantum kernel with the center $\mathbf{y} = (2 \cdot \pi, 2 \cdot \pi)^T$. (**d**) Quantum kernel with the center $\mathbf{y} = (\pi, \pi)^T$. For the RBF kernel the position in the space defined its center; this is not the case with the quantum kernel, the center of which remains fixed; instead its wave distribution changes. In the contour plot the third dimension is indicated by the color, high values are indicated by white to yellow color, low values by blue color.

Because of this it is doubtful if quantum kennels offer any advantage over classical kernels of real-world applications.

5. Variational Approaches

Variational approaches are seen as one of the most promising direction in quantum machine learning [16]. They do not suffer from the input destruction problem, and have no limitations from the HHL algorithm or the failure of generalization of the quantum kernels.

Variational approaches iteratively update a parameterized quantum trial solution also called ansatz (from German ansatz = approach) using a classical optimization algorithm. The parameterized quantum trial solution is represented by a parametrized quantum circuit in the same way as a quantum kernel.

The variational approach can be used for binary classifier with two classes represented by the target values 0 and 1. With the input data vectors $\mathbf{x}_k$ of dimension m and the binary output labels t_k with a training set

$$D = \{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \cdots, (\mathbf{x}_N, t_N)\}, \;\; t_k \in \{0, 1\}.$$

A parametrized quantum circuit $U_{\phi(\mathbf{x}_k)}$ with m parameters encodes each input data vector $\mathbf{x}_k$ of the dimension m [4]

$$U_{\phi(\mathbf{x})_k} |0\rangle^{\otimes m}. \tag{37}$$

Additionally, a variational quantum circuit represents the free parameter $\mathbf{w}$

$$U_{W(\mathbf{w})} \tag{38}$$

that will adapt during training by using an optimizer on a classical computer, leading to the state

$$|\psi(\mathbf{x}_k, \mathbf{w})\rangle = U_{W(\mathbf{w})} \cdot U_{\phi(\mathbf{x})_k} |0\rangle^{\otimes m}. \tag{39}$$

The measured state $|\psi(\mathbf{x}_k, \mathbf{w})\rangle$ by a basis state $|q_m \cdots q_1 q_0\rangle$ representing a binary string, see Figure 3. The measured binary string represents a parity function. A Boolean function whose value is one if and only if the input vector has an odd number of ones represents a parity function. For each input data vector $\mathbf{x}_k$ we determine the output function $o_k \in \{0, 1\}$ and perform an adaptation of the parameters $\mathbf{w}$ of the variational quantum circuit $U_{W(\mathbf{w})}$ using an optimizer on a classical computer that minimizes the loss function between the predicted values represented by the parity function of the measured basis state and the target values. The optimizer approximates a stochastic gradient descent by the Simultaneous Perturbation Stochastic Approximation (SPSA). SPSA requires only two measurements of the loss function, regardless of the dimension of the optimization problem [17] to estimate the stochastic gradient.

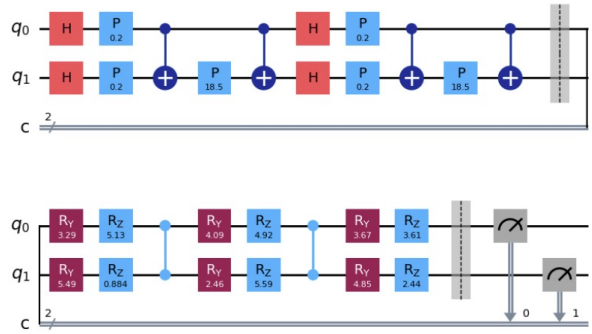

Figure 3. We use the parameterized circuit $U_{\phi(\mathbf{x})} = ZZFeatureMap$ with repetition for the input of dimension two mapping it into 2^2 dimensional space. For $U_{W(\mathbf{w})}$ we use the TwoLocal circuit. It is a parameterized circuit consisting of alternating rotation layers and entanglement layers. An input vector $\mathbf{x}_k$ defines the circuit $U_{\phi(\mathbf{x}_k)}$. We determine the output function $o_k \in \{0, 1\}$. We minimize the loss function by the SPSA optimizer on a classical computer resulting in new parameter $\mathbf{w}$ that defines the adopted variational quantum circuit $U_{W(\mathbf{w})}$. We repeat the process using the adapted variational circuit for the next raining pattern until the error represented by the loss function is minimal. (The dashed lines are visual indicators of the grouping of a circuit sections).

The advantages of classical deep learning models cannot be met by variational algorithms, like the principle of hierarchical organization or the representation of big training sets or regularization. The idea of hierarchical structures is based on the decomposition of a hierarchy into simpler parts leading to a more efficient way of representing information. This principle appears in nature; for example, the structure of the matter itself is hierarchically organized by elementary particles, atomic nuclei, atoms, and molecules [18]. The deep learning approach learns the hierarchical structure from the training data, leading to a high-level abstractions in data by architectures composed of multiple nonlinear transformations using gradient descent through error back-propagation. The back-propagation algorithm leads to a hierarchy from low-level structures to high-level structures, as demonstrated by natural complexity [19–22]. Variational quantum circuits cannot propagate the error backward, since this operation would require the determination of the activity of each layer for the classical optimizer; this operation can only be performed by measurement, which would lead to collapse. By increasing the number of layers in deep learning, we can increase the number of parameters. By doing so, we can add enough degrees of freedom to model large training sets. This is extremely helpful since nowadays, for specific

tasks, a really large amount is collected. Ideally, to overcome overfitting, one has to use a model that has the right capacity. However, this task is difficult and costly since it involves searching through many different architectures. Many experiments with different number of neurons and hidden layers have to be conducted. Using an over-parameterized deep learning network, one can constrain it to the right complexity through regularization. The search for the correct model complexity can be conducted efficiently through empirical experiments. These advantages leading to the success of deep learning cannot be meet by variational algorithms.

6. Conclusions

The input destruction problem has not yet been solved, and theoretical speed-ups are usually analyzed by ignoring the input destruction problem. Other constraints are the normalized representation of vectors by amplitudes and quantum kernels with periodic receptive fields. We arrive at a dilemma: should we ignore or marginalize those constraints or not? Until now, no theoretical advantage over classical algorithms on real data was shown. It is questionable if the input destruction problem will be ever solved and if a $qRAM$ is possible. Usually the constraints are ignored or marginalized.

Instead, we should determine if quantum machine learning (without the discussed constraints) can solve any problems more efficiently than a classical computer. Until now, such problems do not exist and we are far away from showing how to use quantum machine learning algorithms for real-world data. Are there any problems for which a quantum computer is more useful than a classical computer? If we do not answer these questions and overestimate the power of quantum machine, a quantum machine learning winter will follow in analogy to the AI winter.

What could be a possible answer? We can only speculate that in the future real quantum data will be generated through quantum physical experiments or that some new physical breakthroughs will lead to new efficient methods for the state preparation. Putting the speculation aside, quantum symbolic AI algorithms offer a better alternative for successful applications. This is because they avoid the input problem, they do not represent large training data by quantum states. Real-world applications, like the simulation of chemical reactions and optimization problems, should be investigated. The simulation of chemical reactions is based on Hamiltonian simulation together with variational quantum Eigensolvers [23]. Variational quantum Eigensolvers are based on a variational algorithm that estimates the ground state of a system [4,23]. Additionally, a promising direction is indicated by the Boltzmann machine [24–26] and deep belief networks [27,28] rather than the back-propagation algorithm.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Ventura, D.; Martinez, T. Quantum associative memory with exponential capacity. In Proceedings of the Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence, Arlington, VA, USA, 24–26 August 1988; Volume 1, pp. 509–513.
2. Ventura, D.; Martinez, T. Quantum associative memory. *Inf. Sci.* **2000**, *124*, 273–296. [CrossRef]
3. Trugenberger, C.A. Probabilistic Quantum Memories. *Phys. Rev. Lett.* **2001**, *87*, 067901 . [CrossRef] [PubMed]
4. Schuld, M.; Petruccione, F. *Supervised Learning with Quantum Computers*; Springer: Berlin/Heidelberg, Germany, 2018.
5. Wichert, A. *Quantum Artificial Intelligence with Qiskit*; CRC Press: Boca Raton, FL, USA, 2024.
6. Wittek, P. *Quantum Machine Learning, What Quantum Computing Means to Data Mining*; Elsevier Insights: Amsterdam, The Netherlands ; Academic Press: Cambridge, MA, USA, 2014.
7. Aïmeur, E.; Brassard, B.; Gambs, S. Quantum speed-up for unsupervised learning. *Mach. Learn.* **2013**, *90*, 261–287. [CrossRef]
8. Aaronson, S. Quantum Machine Learning Algorithms: Read the Fine Print. *Nat. Phys.* **2015**, *11*, 291–293. [CrossRef]

9. Wichert, A. *Principles of Quantum Artificial Intelligence: Quantum Problem Solving and Machine Learning*, 2nd ed.; World Scientific: Singapore, 2020.
10. Harrow, A.; Hassidim, A.; Lloyd, S. Quantum algorithm for solving linear systems of equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [CrossRef] [PubMed]
11. Giovannetti, V.; Lloyd, S.; Maccone, L. Quantum Random Access Memory. *Phys. Rev. Lett.* **2008**, *100*, 160501. [CrossRef] [PubMed]
12. Reif, J.H. Efficient Approximate Solution of SparseLinear Systems. *Comput. Math. Appl.* **1998**, *36*, 37–58. [CrossRef]
13. Lloyd, S. Universal Quantum Simulators. *Science* **1996**, *273*, 1073–1078. [CrossRef] [PubMed]
14. Schuld, M.; Killoran, N. Quantum Machine Learning in Feature Hilbert Spaces. *Phys. Rev. Lett.* **2019**, *122*, 040504 . [CrossRef] [PubMed]
15. Havlicek, V.; Corcoles, 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*, 210–212. [CrossRef] [PubMed]
16. Jerbi, S.; Fiderer, L.J.; Nautrup, H.P.; Kübler, J.M.; Briegel, H.J.; Dunjko, V. Quantum machine learning beyond kernel methods. *Nat. Commun.* **2023**, *14*, 517 . [CrossRef] [PubMed]
17. Bhatnagar, S.; Prasad, H.L.; Prashanth, L.A. *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*; Springer: Berlin/Heidelberg, Germany, 2013.
18. Resnikoff, H.L. *The Illusion of Reality*; Springer: Berlin/Heidelberg, Germany , 1989.
19. LeCun, Y.; Bengio, Y. *Convolutional Networks for Images, Speech, and Time Series*; MIT Press: Cambridge, MA, USA, 1998; pp. 255–258.
20. Riesenhuber, M.; Poggio, T. Hierarchical models of object recognition in cortex. *Nat. Neurosci.* **1999**, *2*, 1019–1025. [CrossRef] [PubMed]
21. Riesenhuber, M.; Poggio, T. Models of object recognition. *Nat. Neurosci.* **2000**, *3*, 1199–1204. [CrossRef] [PubMed]
22. Riesenhuber, M.; Poggio, T. Neural mechanisms of object recognition. *Curr. Opin. Neurobiol.* **2002**, *12*, 162–168. [CrossRef] [PubMed]
23. Hidary, J.D. *Quantum Computing: An Applied Approach*; Springer: Berlin/Heidelberg, Germany, 2019.
24. Hinton, G.E.; Sejnowski, T.J. Optimal perceptual inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 19–23 June 1983; pp. 448–453.
25. Ackley, D.H.; Hinton, G.E.; Sejnowski, T.J. A learning algorithm for Boltzmann machines. *Cogn. Sci.* **1985**, *9*, 147–169.
26. Hinton, G.E.; Sejnowski, T.J. Learning and Relearning in Boltzmann Machines. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*; Rumelhart, D.E., McClelland, J.L., Eds.; The MIT Press: Cambridge, MA, USA, 1986; pp. 282–317.
27. Smolensky, P. Information Processing in Dynamical Systems: Foundations of Harmony Theory. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*; Rumelhart, D.E., McClelland, J.L., Eds.; The MIT Press: Cambridge, MA, USA, 1986; pp. 194–281.
28. Salakhutdinov, R.; Hinton, G. An Efficient Learning Procedure for Deep Boltzmann Machines. *Neural Comput.* **2012**, *24*, 1967–2006. [CrossRef] [PubMed]