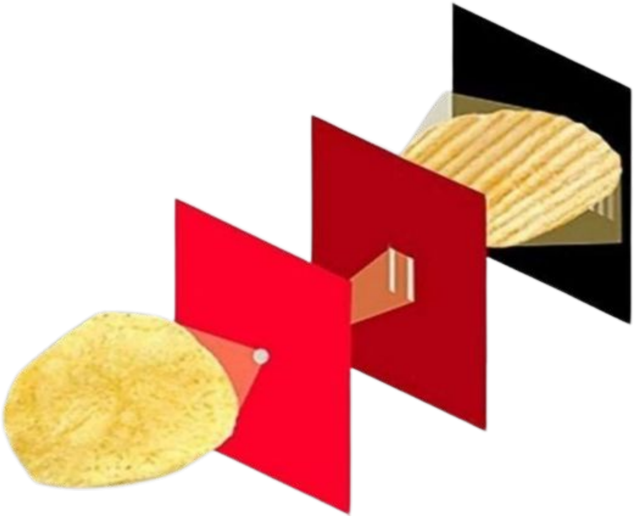
Quantum Machine Learning
or Quantum Mechanics meets Machine Learning



“When mixing machine learning with ‘quantum’,
you catalyse a hype-condensate.”

- Jacob Biamonte, a contributor to the theory of quantum computation

“When mixing machine learning with ‘quantum’, you catalyse a hype-condensate.”

*There exist important caveats that can limit the applicability of the method[51].

- Jacob Biamonte, a contributor to the theory of quantum computation

*There exist important caveats that can limit the applicability of the method[51].

“When mixing machine learning with ‘quantum’, you catalyse a hype-condensate.”

*There exist important caveats that can limit the applicability of the method[51].

- Jacob Biamonte, a contributor to the theory of quantum computation

*There exist important caveats that can limit the applicability of the method[51].

"When mixing machine learning with 'quantum', you catalyse a hype-condensate."

*There exist important caveats that can limit the applicability of the method[51].

*There exist important caveats that can limit the applicability of the method[51].

*There exist important caveats that can limit the applicability of the method[51].

e exist important caveats that can limit the applicability
d[51].

*There exist important caveats that can limit the applicability of the method[51].

*There exist important caveats that can limit the applicability of the method[51].

*There exist important caveats that can limit the applicability of the method[51].

1. General purpose digital quantum computer

Classic computer

Uses bits b:

$b \in \{0, 1\}$

Uses logic gates:

Basic logic gates are operations like AND, OR, NOT…

Quantum computer

Uses qubits q:

$q \in C^2$ (linear combination of orthogonal vectors/states $|0\rangle$ and $|1\rangle$)

Uses quantum logic gates:

Basic quantum logic gates are the basis of hermitian 2x2 matrices
(3 Pauli matrices + Identity matrix)

1. Measurement in quantum mechanics

Let us have a qubit $|q\rangle$ in some unknown state $|q\rangle = a|0\rangle + b|1\rangle$. We want to find *a* and *b.*

To do that we need to measure this qubit. Measurement in quantum mechanics is a projection operation.

E.g., we can measure if $|q\rangle$ is in a state $|0\rangle$ by projecting it on state $|0\rangle$ (finding inner product of $|0\rangle$ and $|q\rangle$):
$\langle 0|q\rangle = a\langle 0|0\rangle + b\langle 0|1\rangle = a\cdot 1 + b\cdot 0 = a.$

However, by doing a single measurement we can only find if $|q\rangle$ is in the state $|0\rangle$ or not. To find the actual (probability) *a* we need to make a series of measurement, gather the statistics, and only then we can derive *a.*

Moreover, by doing a measurement we are changing the state of $|q\rangle$: if we found it in the state $|0\rangle$, then now it is in the state $|0\rangle$ and not in the state $a|0\rangle + b|1\rangle$.

If $|q\rangle = a|0\rangle + b|1\rangle$ is a result of some computations, we have to repeat them from scratch to obtain
$|q\rangle = a|0\rangle + b|1\rangle$ again.

1. Advantage: Quantum parallelism

Let us have L qubits.

$$\underbrace{\frac{|0\rangle + |1\rangle}{\sqrt{2}} \cdots \frac{|0\rangle + |1\rangle}{\sqrt{2}} \; \frac{|0\rangle + |1\rangle}{\sqrt{2}}}_{L} =$$

$$= \frac{1}{2^{L/2}} \underbrace{\left(\underbrace{|0\rangle \cdots |0\rangle |0\rangle}_{L} + \underbrace{|0\rangle \cdots |0\rangle |1\rangle}_{L} + \underbrace{|0\rangle \cdots |1\rangle |0\rangle}_{L} + \underbrace{|0\rangle \cdots |1\rangle |1\rangle}_{L} + \cdots + \underbrace{|1\rangle \cdots |1\rangle |1\rangle}_{L} \right)}_{2^L}.$$

1. Advantage: Quantum parallelism

Let us have some function $f : \{0, \ldots, 2^L - 1\} \to \{0, \ldots, 2^{L'} - 1\}$

We can match it to an operator/matrix $\hat{U}_f$ such that:

$$\hat{U}_f | \underbrace{n}_{L} ; \underbrace{0}_{L'} \rangle = | \underbrace{n}_{L} ; \underbrace{f(n)}_{L'} \rangle .$$

Given that L bits can be in a superposition of all 2^L - 1 integers at once, we are able to calculate the result of *f* over those 2^L - 1 integers in a single run!

Or not?

1. Advantage: Quantum parallelism

*There exist important caveats that can limit the applicability of the method[51].

You cannot extract more than N bits of classic data from N qubits.

Hence, unless you find a way to bypass this limitation

(e.g., by using such operation as an intermediate subroutine and later reducing the data),

quantum parallelism is not such a promising advantage.

1. Limitation: No-cloning theorem

No-cloning theorem states that it is impossible to create an independent and identical copy of an **arbitrary unknown** quantum state.

It is still possible to reproduce a known quantum state.

No-cloning theorem makes it harder to design quantum logic gates and circuits and generally makes life harder.

At the same time, it lies in the basis of quantum cryptography, ensuring that no cipher broadcasted through a quantum channel can be intercepted without being noticed.

Box 1 Table | Speedup techniques for given quantum machine learning subroutines

Method	Speedup	Amplitude amplification	HHL	Adiabatic	qRAM
Bayesian inference[106,107]	$O(\sqrt{N})$	Yes	Yes	No	No
Online perceptron[108]	$O(\sqrt{N})$	Yes	No	No	Optional
Least-squares fitting[9]	$O(\log N)$*	Yes	Yes	No	Yes
Classical Boltzmann machine[20]	$O(\sqrt{N})$	Yes/No	Optional/No	No/Yes	Optional
Quantum Boltzmann machine[22,61]	$O(\log N)$*	Optional/No	No	No/Yes	No
Quantum PCA[11]	$O(\log N)$*	No	Yes	No	Optional
Quantum support vector machine[13]	$O(\log N)$*	No	Yes	No	Yes
Quantum reinforcement learning[30]	$O(\sqrt{N})$	Yes	No	No	No

*There exist important caveats that can limit the applicability of the method[51].

Other algorithms:

- Grover's search over N-sized database (quadratic speed-up)
- Fourier transform
- Quantum eigenvalue estimation algorithm
- solving linear sets of equations over 2^n-dimensional vector spaces (HHL algorithm)

The last 3 show exponential speed-up compared to their classic counterpart, sometimes in exchange for precision and/or with several caveats

2. Common problems of quantum algorithms

1. **The input problem**. Although quantum algorithms can provide dramatic speedups for processing data, they seldom provide advantages in reading data. This means that the cost of reading in the input can in some cases dominate the cost of quantum algorithms.

Understanding this factor is an ongoing challenge.

2. Common problems of quantum algorithms

2. **The output problem**. Obtaining the full solution from some quantum algorithms as a string of bits requires learning an exponential number of bits. This makes some applications of quantum machine learning algorithms infeasible.

This problem can potentially be sidestepped by learning only summary statistics for the solution state.

2. Common problems of quantum algorithms

3. **The costing problem**. Closely related to the input/output problems, at present very little is known about the true number of gates required by quantum machine learning algorithms.

Bounds on the complexity suggest that for sufficiently large problems they will offer huge advantages, but it is still unclear when that crossover point occurs.

2. Common problems of quantum algorithms

4. **The benchmarking problem**. It is often difficult to assert that a quantum algorithm is ever better than all known classical machine algorithms in practice

because this would require extensive benchmarking against modern heuristic methods. Establishing lower bounds for quantum machine learning would partially address this issue.

Box 1 Table | Speedup techniques for given quantum machine learning subroutines

Method	Speedup	Amplitude amplification	HHL	Adiabatic	qRAM
Bayesian inference[106,107]	$O(\sqrt{N})$	Yes	Yes	No	No
Online perceptron[108]	$O(\sqrt{N})$	Yes	No	No	Optional
Least-squares fitting[9]	$O(\log N)$*	Yes	Yes	No	Yes
Classical Boltzmann machine[20]	$O(\sqrt{N})$	Yes/No	Optional/No	No/Yes	Optional
Quantum Boltzmann machine[22,61]	$O(\log N)$*	Optional/No	No	No/Yes	No
Quantum PCA[11]	$O(\log N)$*	No	Yes	No	Optional
Quantum support vector machine[13]	$O(\log N)$*	No	Yes	No	Yes
Quantum reinforcement learning[30]	$O(\sqrt{N})$	Yes	No	No	No

*There exist important caveats that can limit the applicability of the method[51].

Other algorithms:

- Grover's search over N-sized database (quadratic speed-up)
- Fourier transform
- Quantum eigenvalue estimation algorithm
- solving linear sets of equations over 2^n-dimensional vector spaces (HHL algorithm)

The last 3 show exponential speed-up compared to their classic counterpart, sometimes in exchange for precision and/or with several caveats

2. HHL algorithm for solving linear systems of equations

An algorithm for solving A**x** = **b** linear system of equations.

Requires preparation of $|b\rangle$ for solving an $A|x\rangle = |b\rangle$ problem.

The HHL algorithm takes $O[(\log N)^2]$ quantum steps to output $|x\rangle$,
compared with the $O(N \log N)$* steps required to find **x** using the best known method on a classical computer.
→ We have an exponential speed-up!

Caveats:

1) Requires preparation of $|b\rangle$ (can be costly on time or resources)
2) Obtaining **x** from $|x\rangle$ requires measurements (unless it is left as it is and is used later in a dimensionality-reducing operation)
3) Does not work for any matrix A

2. Quantum Principal Component Analysis

Given that there exists a Quantum eigenvalue estimation algorithm,
it is stated that qPCA can be performed in $O[(\log d)^2]$ steps, where d is is the dimension of the data.

It is then compared with the classic algorithms’ complexity $O[d^2]$*,
which is strange, since it is $O[d^3]$ in general, and there seems to be no special assumptions that could make it true.

Nonetheless, another exponential speed-up.

2. Quantum Support Vector Machine

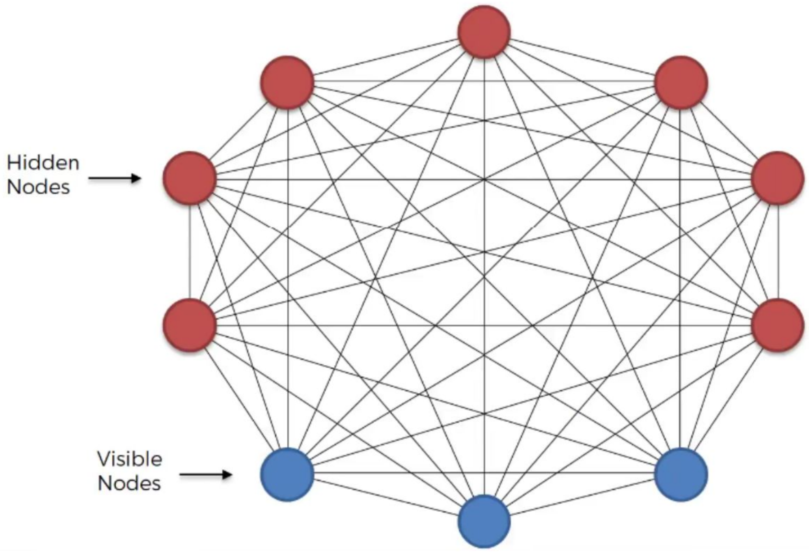
The problem of finding the desired SVM separating hyperplane can be formalized as a minimization problem.

1. A modified Groover search algorithm (Rem.: database search) can be used for finding *s* support vectors in $\sqrt{(N/s)}$ iterations
2. Quantum Linear Algebra algorithms can further improve SVM by log N speed-up.

3. Quantum Analog Computer

What is Boltzmann Machine (BM)?

1. From a Machine Learning perspective, BM is a class recurrent neural network with undirected edges; provides a generative model for the data
2. Nodes can be in the states {0, 1}.
3. The probability of BM being in a certain state (visible nodes, hidden nodes) (or (*v, h*)) is described by Boltzmann distribution



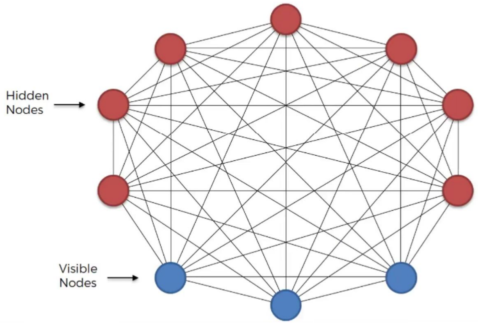
$$P(v,h) = e^{-E(v,h)}/Z,$$

where $E(v,h) = -\sum_i v_i b_i - \sum_j h_j d_j - \sum_{i,j} w_{ij}^{vh} v_i h_j - \sum_{i,j} w_{i,j}^{v} v_i v_j - \sum_{i,j} w_{i,j}^{h} h_i h_j.$

3. Quantum Analog Computer

Given some training data, we can optimize BM weights *b, d, w* for a maximum probability of the given data by maximizing objective function

$$O_{\mathrm{ML}} := \frac{1}{N_{\mathrm{train}}} \sum_{v \in x_{\mathrm{train}}} \log \left(\sum_{h=1}^{n_h} P(v,h) \right) - \frac{\lambda}{2} w^T w$$



However, this problem becomes exponentially difficult with the increase of BM nodes.

This is where (quantum) analog computers come to rescue.

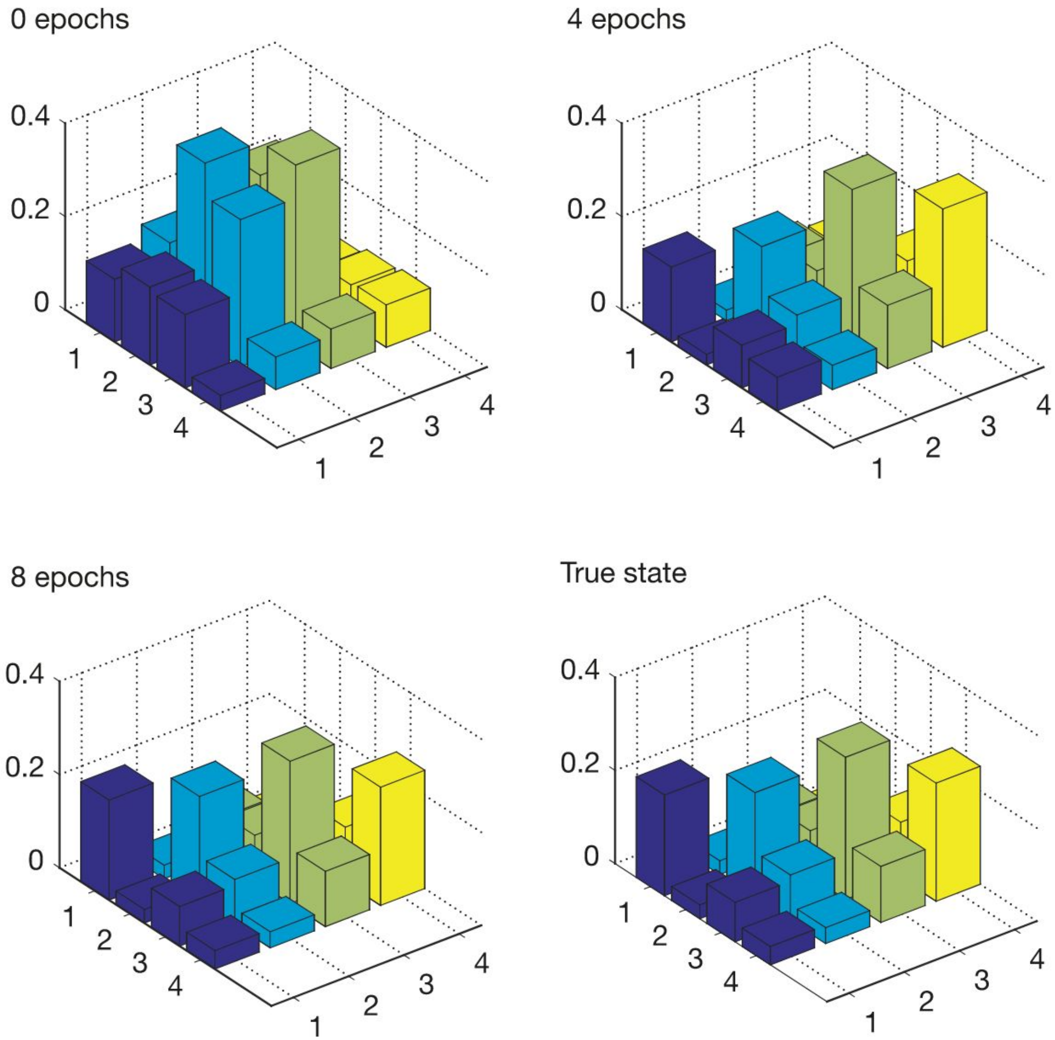
Analog computer can simulate physical systems (in BM case, thermal equilibrium), and we can tune them

3. From D-Wave promotions

Quantum Annealing

D-Wave systems use a process called quantum annealing to search for solutions to a problem.

In nature, physical systems tend to evolve toward their lowest energy state: objects slide down hills, hot things cool down, and so on. This behavior also applies to quantum systems. To imagine this, think of a traveler looking for the best solution by finding the lowest valley in the energy landscape that represents the problem.


0 epochs
4 epochs
8 epochs
True state
0.4
0.2
0
1
2
3
4