REVIEW

doi:10.1038/nature23474

Quantum machine learning

Jacob Biamonte[1,2], Peter Wittek[3], Nicola Pancotti[4], Patrick Rebentrost[5], Nathan Wiebe[6] & Seth Lloyd[7]

Fuelled by increasing computer power and algorithmic advances, machine learning techniques have become powerful tools for finding patterns in data. Quantum systems produce atypical patterns that classical systems are thought not to produce efficiently, so it is reasonable to postulate that quantum computers may outperform classical computers on machine learning tasks. The field of quantum machine learning explores how to devise and implement quantum software that could enable machine learning that is faster than that of classical computers. Recent work has produced quantum algorithms that could act as the building blocks of machine learning programs, but the hardware and software challenges are still considerable.

Long before we possessed computers, human beings strove to find patterns in data. Ptolemy fitted observations of the motions of the stars to a geocentric model of the cosmos, with complex epicycles to explain the retrograde motions of the planets. In the sixteenth century, Kepler analysed the data of Copernicus and Brahe to reveal a previously hidden pattern: planets move in ellipses with the Sun at one focus of the ellipse. The analysis of astronomical data to reveal such patterns gave rise to mathematical techniques such as methods for solving linear equations (Newton–Gauss), learning optima via gradient descent (Newton), polynomial interpolation (Lagrange), and least-squares fitting (Laplace). The nineteenth and early twentieth centuries gave rise to a broad range of mathematical methods for analysing data to reveal the patterns that it contained.

The construction of digital computers in the mid-twentieth century allowed the automation of data analysis techniques. Over the past half-century, the rapid progression of computer power has allowed the implementation of linear algebraic data analysis techniques such as regression and principal component analysis, and has led to more complex learning methods such as support vector machines. Over the same time frame, the development and rapid advance of digital computers spawned novel machine learning methods. Artificial neural networks such as perceptrons were implemented in the 1950s (ref. 1), as soon as computers had the power to realize them. Deep learning built on neural networks (such as Hopfield networks and Boltzmann machines) and training methods (such as back propagation) were introduced and implemented in the 1960s to 1990s (ref. 2). In the past decade, particularly in the past five years, the combination of powerful computers and special-purpose information processors capable of implementing deep networks with billions of weights[3], together with their application to very large datasets, has revealed that such deep learning networks are capable of identifying complex and subtle patterns in data.

Quantum mechanics is well known to produce atypical patterns in data. Classical machine learning methods such as deep neural networks frequently have the feature that they can both recognize statistical patterns in data and produce data that possess the same statistical patterns: they recognize the patterns that they produce. This observation suggests the following hope. If small quantum information processors can produce statistical patterns that are computationally difficult for a classical computer to produce, then perhaps they can also recognize patterns that are equally difficult to recognize classically.

The realization of this hope depends on whether efficient quantum algorithms can be found for machine learning. A quantum algorithm is a set of instructions solving a problem, such as determining whether two graphs are isomorphic, that can be performed on a quantum computer. Quantum machine learning software makes use of quantum algorithms as part of a larger implementation. By analysing the steps that quantum algorithms prescribe, it becomes clear that they have the potential to outperform classical algorithms for specific problems (that is, reduce the number of steps required). This potential is known as quantum speedup.

The notion of a quantum speedup depends on whether one takes a formal computer science perspective—which demands mathematical proofs—or a perspective based on what can be done with realistic, finite-size devices—which requires solid statistical evidence of a scaling advantage over some finite range of problem sizes. For the case of quantum machine learning, the best possible performance of classical algorithms is not always known. This is similar to the case of Shor's polynomial-time quantum algorithm for integer factorization: no sub-exponential-time classical algorithm has been found, but the possibility is not provably ruled out.

Determination of a scaling advantage contrasting quantum and classical machine learning would rely on the existence of a quantum computer and is called a 'benchmarking' problem. Such advantages could include improved classification accuracy and sampling of classically inaccessible systems. Accordingly, quantum speedups in machine learning are currently characterized using idealized measures from complexity theory: query complexity and gate complexity (see Box 1 and Box 1 Table). Query complexity measures the number of queries to the information source for the classical or quantum algorithm. A quantum speedup results if the number of queries needed to solve a problem is lower for the quantum algorithm than for the classical algorithm. To determine the gate complexity, the number of elementary quantum operations (or gates) required to obtain the desired result are counted.

Query and gate complexity are idealized models that quantify the necessary resources to solve a problem class. Without knowing how to map this idealization to reality, not much can be said about the necessary resource scaling in a real-world scenario. Therefore, the required resources of classical machine learning algorithms are mostly quantified by numerical experimentation. The resource requirements of quantum machine learning algorithms are likely to be similarly difficult to quantify in practice. The analysis of their practical feasibility is a central subject of this review.

As will be seen throughout the review, there are quantum algorithms for machine learning that exhibit quantum speedups[4–7]. For example, the

[1]Quantum Complexity Science Initiative, Skolkovo Institute of Science and Technology, Skoltech Building 3, Moscow 143026, Russia. [2]Institute for Quantum Computing, University of Waterloo, Waterloo, N2L 3G1 Ontario, Canada. [3]ICFO—The Institute of Photonic Sciences, Castelldefels, Barcelona 08860 Spain. [4]Max Planck Institute of Quantum Optics, 1 Hans-Kopfermannstrasse, D-85748 Garching, Germany. [5]Massachusetts Institute of Technology, Research Laboratory of Electronics, Cambridge, Massachusetts 02139, USA. [6]Station Q Quantum Architectures and Computation Group, Microsoft Research, Redmond, Washington 98052, USA. [7]Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts 02139, USA.

BOX 1

Quantum speedups

Quantum computers use effects such as quantum coherence and entanglement to process information in ways that classical computers cannot. The past two decades have seen steady advances in constructing more powerful quantum computers. A quantum algorithm is a stepwise procedure performed on a quantum computer to solve a problem, such as searching a database. Quantum machine learning software makes use of quantum algorithms to process information. Quantum algorithms can in principle outperform the best known classical algorithms when solving certain problems. This is known as a quantum speedup[105].

For example, quantum computers can search an unsorted database with N entries in time proportional to $\sqrt{N}$—that is, $O(\sqrt{N})$—where a classical computer given blackbox access to the same database takes time proportional to N: thus the quantum computer exhibits a square-root speedup over the classical computer. Similarly, quantum computers can perform Fourier transforms over N data points, invert sparse $N \times N$ matrices, and find their eigenvalues and eigenvectors in time proportional to a polynomial in $\log_2 N$, where the best known algorithms for classical computers take time proportional to $N\log_2 N$: thus the quantum computer exhibits an exponential speedup over the best classical computer algorithms.

In the Box 1 Table, speedups are taken with respect to their classical counterpart(s)—hence, $O(\sqrt{N})$ means quadratic speedup and $O(\log(N))$ means exponential relative to their classical counterpart.

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].

quantum basic linear algebra subroutines (BLAS)—Fourier transforms, finding eigenvectors and eigenvalues, solving linear equations—exhibit exponential quantum speedups over their best known classical counterparts[8–10]. This quantum BLAS (qBLAS) translates into quantum speedups for a variety of data analysis and machine learning algorithms including linear algebra, least-squares fitting, gradient descent, Newton's method, principal component analysis, linear, semidefinite and quadratic programming, topological analysis and support vector machines[9,11–19]. At the same time, special-purpose quantum information processors such as quantum annealers and programmable quantum optical arrays are well matched to deep learning architectures[20–22]. Although it is not clear yet to what extent this potential can be realized, there are reasons to be optimistic that quantum computers can recognize patterns in data that classical computers cannot.

The learning machines we consider can be either classical[23–32] or quantum[8,9,11,13,33–36]. The data they analyse can be either classical or quantum states produced by quantum sensing or measuring apparatus[30,37]. We briefly discuss conventional machine learning—the use of classical computers to find patterns in classical data. We then turn to quantum machine learning, where the data that the quantum computer analyses can be either classical data, which ends up encoded as quantum states, or quantum data. Finally, we discuss briefly the problem of using classical machine learning techniques to find patterns in quantum dynamics.

Classical machine learning

Classical machine learning and data analysis can be divided into several categories. First, computers can be used to perform 'classic' data analysis methods such as least-squares regression, polynomical interpolation and data analysis. Machine learning protocols can be supervised or unsupervised. In supervised learning, the training data are divided into labelled categories, such as samples of handwritten digits together with the actual number the handwritten digit is supposed to represent, and the job of the machine is to learn how to assign labels to data outside the training set. In unsupervised learning, the training set is unlabelled, and the goal of the machine is to find the natural categories into which the training data falls (for example, different types of photos on the internet) and then to categorize data outside the training set. Finally, there are machine learning tasks, such as playing Go, that involve combinations of supervised and unsupervised learning, together with training sets that may be generated by the machine itself.

Linear-algebra-based quantum machine learning

A wide variety of data analysis and machine learning protocols operate by performing matrix operations on vectors in a high-dimensional vector space. But quantum mechanics is all about matrix operations on vectors in high-dimensional vector spaces.

The key ingredient behind these methods is that the quantum state of n quantum bits or qubits is a vector in a 2^n-dimensional complex vector space; performing a quantum logic operations or a measurement on qubits multiplies the corresponding state vector by $2^n \times 2^n$ matrices. By building up such matrix transformations, quantum computers have been shown to perform common linear algebraic operations such as Fourier transforms[38], finding eigenvectors and eigenvalues[39], and solving linear sets of equations over 2^n-dimensional vector spaces in time that is polynomial in n, exponentially faster than their best known classical counterparts[8]. This latter is commonly referred to as the Harrow, Hassidim and Lloyd (HHL) algorithm[8] (see Box 2). The original variant assumed a well conditioned matrix that is sparse. Sparsity is unlikely in data science, but later improvements relaxed this assumption to include low-rank matrices as well[10,33,40]. Going past HHL, here we survey several quantum algorithms which appear as subroutines when linear algebra techniques are employed in quantum machine learning software.

Quantum principal component analysis

For example, consider principal component analysis (PCA). Suppose that the data are presented in the form of vectors $\boldsymbol{v}_j$ in a d-dimensional vector space, where $d = 2^n = N$. For example, $\boldsymbol{v}_j$ could be the vector of changes in prices of all stocks in the stock market from time t_j to time t_{j+1}. The covariance matrix of the data is $C = \sum_j \boldsymbol{v}_j \boldsymbol{v}_j^{\mathrm{T}}$, where superscript T denotes the transpose operation: the covariance matrix summarizes the correlations between the different components of the data, for example, correlations between changes in the prices of different stocks. In its simplest form, principal component analysis operates by diagonalizing the covariance matrix: $C = \sum_k e_k \boldsymbol{c}_k \boldsymbol{c}_k^{\dagger}$, where the $\boldsymbol{c}_k$ are the eigenvectors

of C, and e_k are the corresponding eigenvalues. (Because C is symmetric, the eigenvectors $\boldsymbol{c}_k$ form an orthonormal set.) If only a few of the eigenvalues c_k are large, and the remainder are small or zero, then the eigenvectors corresponding to those eigenvalues are called the principal components of C. Each principal component represents an underlying common trend or form of correlation in the data, and decomposing a data vector $\boldsymbol{v}$ in terms of principal components, $\boldsymbol{v}=\sum_k v_k \boldsymbol{c}_k$, allows one both to compress the representation of the data and to predict future behaviour. Classical algorithms for performing PCA scale as $O(d^2)$ in terms of computational complexity and query complexity. (We note that we make use of 'big O' notation to keep track of the leading term that dominates scaling.)

For quantum principal component analysis of classical data[11], we choose a data vector $\boldsymbol{v}_j$ at random, and use a quantum random access memory (qRAM)[41] to map that vector into a quantum state: $\boldsymbol{v}_j \rightarrow |v_j\rangle$. The quantum state that summarizes the vector has $\log d$ qubits, and the operation of the qRAM requires $O(d)$ operations divided over $O(\log d)$ steps that can be performed in parallel. Because $\boldsymbol{v}_j$ was chosen at random, the resulting quantum state has a density matrix $\rho=(1/N)\sum_j |v_j\rangle\langle v_j|$, where N is the number of data vectors. By comparison with the covariance matrix C for the classical data, we see that the density matrix for the quantum version of the data actually is the covariance matrix, up to an overall factor. By repeatedly sampling the data, and using a trick called density matrix exponentiation[42] combined with the quantum phase estimation algorithm[39], which finds eigenvectors and eigenvalues of matrices, we can take the quantum version of any data vector $|\boldsymbol{v}\rangle$ and decompose it into the principal components $|c_k\rangle$, revealing the eigenvalue of C at the same time: $|v\rangle \rightarrow \sum_k v_k|c_k\rangle|\tilde{e}_k\rangle$. The properties of the principal components of C can then be probed by making measurements on the quantum representation of the eigenvectors of C. The quantum algorithm scales as $O[(\log N)^2]$ in both computational complexity and query complexity. That is, quantum PCA is exponentially more efficient than classical PCA.

Quantum support vector machines and kernel methods

The simplest examples of supervised machine learning algorithms are linear support vector machines and perceptrons. These methods seek to find an optimal separating hyperplane between two classes of data in a dataset such that, with high probability, all training examples of one class are found only on one side of the hyperplane. The most robust classifier for the data is given when the margin between the hyperplane and the data are maximized. Here the 'weights' learned in the training are the parameters of the hyperplane. One of the greatest powers of the support vector machine lies in its generalization to nonlinear hypersurfaces via kernel functions[43]. Such classifiers have found great success in image segmentation as well as in the biological sciences.

Like its classical counterpart, the quantum support vector machine is a paradigmatic example of a quantum machine learning algorithm[13]. A first quantum support vector machine was discussed in the early 2000s[44], using a variant of Grover's search for function minimization[45]. Finding s support vectors out of N vectors consequently takes $\sqrt{N/s}$ iterations. Recently, a least-squares quantum support vector machine was developed that harnesses the full power of the qBLAS subroutines. The data input can come from various sources, such as from qRAM accessing classical data or from a quantum subroutine preparing quantum states. Once the data are made available to the quantum computing device, they are processed with quantum phase estimation and matrix inversion (the HHL algorithm). All the operations required to construct the optimal separating hyperplane and to test whether a vector lies on one side or the other can in principle be performed in time that is polynomial in $\log N$, where N is the dimension of the matrix required to prepare a quantum version of the hyperplane vector. Polynomial[13] and radial basis function kernels[46] are discussed, as well as another kernel-based method called Gaussian process regression[47]. This approach to quantum support machines has been experimentally demonstrated in a nuclear magnetic resonance testbed for a handwritten digit recognition task[48].

BOX 2

HHL algorithm

The HHL algorithm for inverting systems of equations is a fundamental and easy-to-understand subroutine, underpinning many quantum machine learning algorithms. The algorithm seeks to solve $A\boldsymbol{x}=\boldsymbol{b}$ using a quantum computer. HHL quantizes the problem by expressing the vector $\boldsymbol{b}\in C^N$ as a quantum state $|b\rangle$ over $\log_2 N$ qubits, and the vector $\boldsymbol{x}$ as a quantum state $|x\rangle$. The matrix A can be assumed to be Hermitian without loss of generality because the space can always be expanded to make this true. The equation $A|x\rangle = |b\rangle$ can then be solved by multiplying both sides of the equation by A^{-1}, where A^{-1} is the inverse of A. The HHL algorithm then allows one to construct the quantum state proportional to $A^{-1}|b\rangle$. More generally, when A is not square or has zero eigenvalues, the algorithm can be used to find the state $|x\rangle$ that minimizes[9] $|A|x\rangle - |b\rangle|$.

The algorithm works as follows. Assume $|b\rangle=\sum_n b_n|E_n\rangle$ where $|E_n\rangle$ is an eigenvector of A with eigenvalue $\lambda_n \geq \Lambda$. By applying phase estimation under A to compute λ_n and by rotating an ancillary qubit through an angle of $\arcsin(\Lambda/\lambda_n)$ and then uncomputing the phase estimation we obtain:

$$\sum_n b_n|E_n\rangle\left(\frac{\Lambda}{\lambda_n}|1\rangle+\sqrt{1-\frac{\Lambda^2}{\lambda_n^2}}|0\rangle\right)$$

If the ancillary qubit is measured and if 1 is observed then each eigenstate is divided through by λ_n, which affects the inverse. The number of times that the state preparation circuit needs to be applied to succeed, after applying amplitude amplification, is $O(\|A\|/\Lambda)$, which is the condition number for the matrix.

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 $\boldsymbol{x}$ using the best known method on a classical computer.

There are several important caveats to the HHL algorithm. First, finding the full answer $\boldsymbol{x}$ from the quantum state $|x\rangle$ requires $O(N)$ repetitions to reconstruct the N components of $\boldsymbol{x}$. Generalizations to HHL, such as least-squares fitting, sidestep this problem by allowing the output to have many fewer dimensions than the input. In general, however, HHL can provide only features of the data such as moments of the solution vector or its expectation value $\boldsymbol{x}^\dagger B\boldsymbol{x}$ over other sparse matrices B. The second caveat is that the input vector $|b\rangle$ needs to be prepared, either on a quantum computer or using qRAM, which may be expensive. The third caveat is that the matrix must be well conditioned and it must be possible to simulate e^{-iA} efficiently. Finally, although the HHL algorithms scales as $O[(\log N)^2]$, current estimates of the cost of the algorithm for practical problems are prohibitive[109], which underlines the importance of investigating further improvements[10]. In general, the promise of exponential speedups for linear systems should be tempered with the realization that they apply only to certain problems.

qBLAS-based optimization

Many data analysis and machine learning techniques involve optimization. Of increasing interest is the use of D-Wave processors to solve combinatorial optimization problems by means of quantum annealing. Some optimization problems can also be formulated as a single-shot solution of a linear system, such as the optimization of a quadratic function subject to equality constraints, a subset of quadratic programming problems. If the matrices involved are sparse or low rank, such problems can be solved in time that is polynomial in $\log d$, where d is the system dimension via the HHL matrix inversion algorithm, yielding an exponential speedup over classical algorithms, which run in time that is polynomial in d.

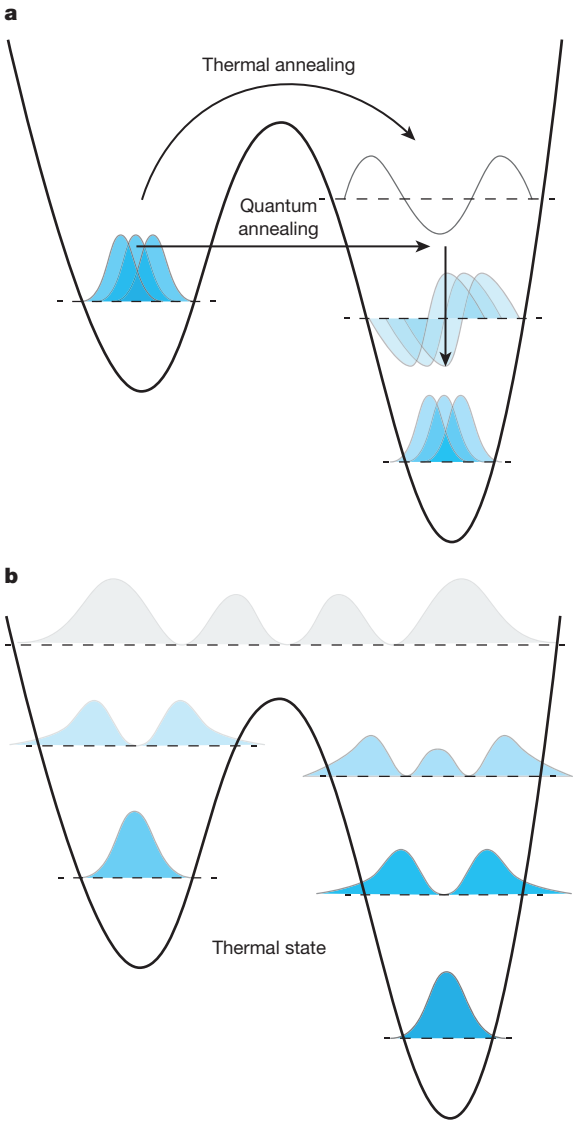


Figure 1 | Quantum tunnelling versus thermalization. A quantum state tunnels when approaching a resonance point before decoherence induces thermalization. Shades of blue illustrate occupation of energy levels (black dashes). **a**, A quantum state must traverse a local minimum in thermal annealing, whereas a coherent quantum state can tunnel when brought close to resonance. **b**, Coherent effects decay through interaction with an environment, causing the probability distribution of the occupancy of a system's energy levels to follow a Gibbs distribution.

Most methods in machine learning require iterative optimization of their performance. As an example, inequality constraints are often handled via penalty functions[49] and variations of gradient descent or Newton's method. A modification of the quantum PCA method implements iterative gradient descent and Newton's methods for polynomial optimization, and can again provide an exponential speedup over classical methods[19]. Multiple copies of the present solution, encoded in a quantum state, are used to improve that solution at each step. Brandao and Svore provide a quantum version of semi-definite programming that holds out the possibility of super-polynomial speedups[18]. The quantum approximate optimization algorithm (the QAO algorithm)[50] provides a unique approach to optimization based on alternating qubit rotations with the application of the problem's penalty function.

Reading classical data into quantum machines

Classical data must be input before being processed on a quantum computer. This 'input problem' often has little overhead but can present a serious bottleneck for certain algorithms. Likewise, the 'output problem' is faced when reading out data after being processed on a quantum device. Like the input problem, the output problem often causes a noticeable operational slowdown.

In particular, if we wish to apply HHL, least-squares fitting, quantum principal component analysis, quantum support vector machines, and related approaches to classical data, the procedure begins by first loading considerable amounts of data into a quantum system, which can require exponential time[51]. This can be addressed in principle using qRAM but the cost of doing so may be prohibitive for big data problems[52]. Apart from combinatorial-optimization-based approaches, the only known linear-algebra-based quantum machine learning algorithm that does not rely on large-scale qRAM is the quantum algorithm for performing topological analysis of data (persistent homology)[14]. With the notable exceptions of least-squares fitting and quantum support vector machines, linear-algebra-based algorithms can also suffer from the output problem because desirable classical quantities such as the solution vector for HHL or the principal components for PCA are exponentially hard to estimate.

Despite the potential for exponential quantum speedups, without much effort put into optimization, the circuit size and circuit depth overhead can balloon (to around 10^{25} quantum gates in one proposed realization of HHL[53]). Ongoing work is needed to optimize such algorithms, provide better cost estimates and ultimately to understand the sort of quantum computer that we would need to provide useful quantum alternatives to classical machine learning.

Deep quantum learning

Classical deep neural networks are highly effective tools for machine learning and are well suited to inspire the development of deep quantum learning methods. Special-purpose quantum information processors such as quantum annealers and programmable photonic circuits are well suited for constructing deep quantum learning networks[21,54,55]. The simplest deep neural network to quantize is the Boltzmann machine (see Box 3 and Box 3 Figure). The classical Boltzmann machine consists of bits with tunable interactions: the Boltzmann machine is trained by adjusting those interactions so that the thermal statistics of the bits, described by a Boltzmann–Gibbs distribution (see Fig. 1b), reproduces the statistics of the data. To quantize the Boltzmann machine one simply takes the neural network and expresses it as a set of interacting quantum spins, corresponding to a tunable Ising model. Then by initializing the input neurons in the Boltzmann machines into a fixed state and allowing the system to thermalize, we can read out the output qubits to obtain an answer.

An essential feature of deep quantum learning is that it does not require a large, general-purpose quantum computer. Quantum annealers are special-purpose quantum information processors that are much easier to construct and to scale up than are general-purpose quantum computers (see Fig. 1a). Quantum annealers are well suited for implementing deep quantum learners, and are commercially available. The D-Wave quantum annealer is a tunable transverse Ising model that can be programmed to yield the thermal states of classical systems, and certain quantum spin systems. The D-Wave device has been used to perform deep quantum learning protocols on more than a thousand spins[56]. Quantum Boltzmann machines[22] with more general tunable couplings, capable of implementing universal quantum logic, are currently at the design stage[57]. On-chip silicon waveguides have been used to construct linear optical arrays with hundreds of tunable interferometers, and special-purpose superconducting quantum information processors could be used to implement the quantum approximate optimization algorithm.

There are several ways that quantum computers can provide advantages here. First, quantum methods can make the system thermalize quadratically faster than its classical counterpart[20,58–60]. This can make accurate training of fully connected Boltzmann machines practical. Second, quantum computers can accelerate Boltzmann training by providing improved ways of sampling. Because the neuron activation pattern in the Boltzmann machine is stochastic, many repetitions are needed to determine success probabilities, and in turn, to discover the effect that

BOX 3

Training quantum Boltzmann machines

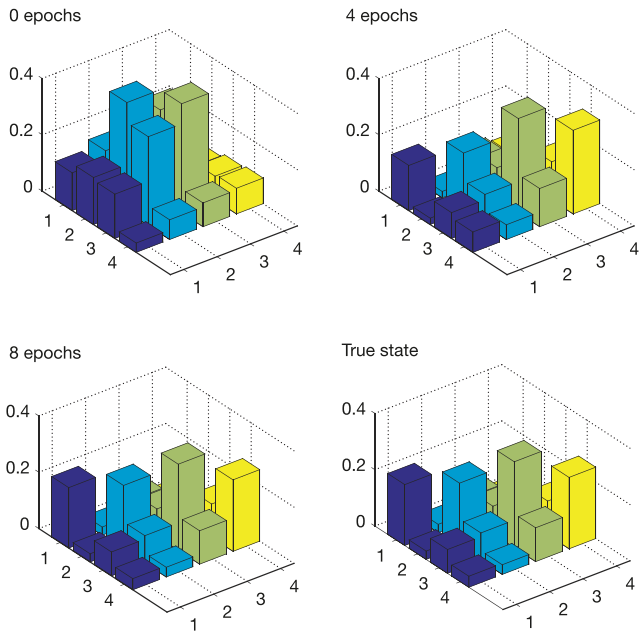
In quantum Boltzmann machine training we wish to learn a set of Hamiltonian parameters (w_j) such that for a fixed set of H_j we have that our input state ρ_{train} is well approximated[22,61] by $\sigma = e^{-\sum_j w_j H_j}/\text{Tr}(e^{-\sum_j w_j H_j})$. For all visible Boltzmann machines, the quantum relative entropy $S(\rho_{\text{train}}\|\sigma) = \text{Tr}[\rho_{\text{train}} \log(\rho_{\text{train}}) - \rho_{\text{train}} \log(\sigma)]$ is the most logical way to measure the quality of the approximation. It is easy to see (assuming that the kernels of ρ and σ coincide) that the quantum relative entropy provides an upper bound for the distance between the two states. Thus, minimizing it minimizes the error in approximating the state.

Although the relative entropy is an excellent measure of the distance between two states, it can be difficult to discover experimentally. However, the gradient (that is, the direction of greatest change) of the relative entropy is easy to estimate[61]:

$$\partial_{w_j} S(\rho_{\text{train}}\|\sigma) = \text{Tr}(\sigma H_j)\,\text{Tr}(\rho H_j)$$

Given an experimental dataset of expectation values for ρ_{train} and a quantum simulator for $\text{Tr}(\sigma H_j)$ we can find the direction of greatest improvement in the quantum relative entropy. Gradient descent then is used to update $\mathbf{w}$ via $\boldsymbol{w} \rightarrow \boldsymbol{w} - \eta \nabla S\,(\rho_{\text{train}}\|\sigma)$ for $\eta > 0$. Stoquastic (quantum stochastic) Hamiltonians have the property that all off-diagonal matrix elements in the standard basis are real and non-positive (equivalently non-negative). No efficient classical analogue of this method is known in general for non-stoquastic H (see ref. 57).

We show this protocol below for learning a random state formed from a uniform mixture of four random states—random with respect to the unique and unitarily invariant Haar measure. Fewer than ten gradient steps (epochs) are needed to train it to approximately generate ρ_{train} using a complete set of Hamiltonian terms.



Box 3 Figure | Learning a random state using a quantum Boltzmann machine.

changing a weight in the neural network has on the performance of the deep network. When training a quantum Boltzmann machine, in contrast, quantum coherence can quadratically reduce the number of samples needed to learn the desired task. Furthermore, quantum access to the training data (that is, qRAM or a quantum blackbox subroutine) allows the machine to be trained using quadratically fewer access requests to the training data than are required by classical methods: a quantum algorithm can train a deep neural network on a large training dataset while reading only a minuscule number of training vectors[20].

Quantum information processing provides new, fundamentally quantum, models for deep learning. For example, adding a transverse field to the Ising model quantum Boltzmann machine can induce a variety of quantum effects such as tunnelling[22,61]. Adding further quantum couplings transforms the quantum Boltzmann machine into a variety of quantum systems[57,62]. Adding a tunable transverse interaction to a tunable Ising model is known to be universal for full quantum computing[57]: with the proper weight assignments this model can execute any algorithm that a general-purpose quantum computer can perform. Such universal deep quantum learners may recognize and classify patterns that classical computers cannot.

Unlike classical Boltzmann machines, quantum Boltzmann machines output a quantum state. Thus deep quantum networks can learn to generate quantum states representative of a wide variety of systems, allowing the network to act as a form of quantum associative memory[63]. This ability to generate quantum states is absent from classical machine learning. Thus quantum Boltzmann training has applications beyond classifying quantum states and providing richer models for classical data.

Quantum machine learning for quantum data

Perhaps the most immediate application of quantum machine learning is to quantum data—the actual states generated by quantum systems and processes. As described above, many quantum machine learning algorithms find patterns in classical data by mapping the data to quantum mechanical states, and then manipulating those states using basic quantum linear algebra subroutines. These quantum machine learning algorithms can be applied directly to the quantum states of light and of matter to reveal their underlying features and patterns. The resulting quantum modes of analysis are frequently much more efficient and more illuminating than the classical analysis of data taken from quantum systems. For example, given multiple copies of a system described by an $N \times N$ density matrix, quantum principal component analysis can be used to find its eigenvalues and to reveal the corresponding eigenvectors in time $O[(\log N)^2]$, compared with the $O(N^2)$ measurements needed for a classical device to perform tomography on a density matrix, and the $O(N^2)$ operations needed to perform the classical PCA. Such quantum analysis of quantum data could profitably be performed on the relatively small quantum computers that are likely to be available over the next several years.

A particularly powerful quantum data analysis technique is the use of quantum simulators to probe quantum dynamics. Quantum simulators are 'quantum analogue computers'—quantum systems whose dynamics can be programmed to match the dynamics of some desired quantum system. A quantum simulator can either be a special-purpose device constructed to simulate a particular class of quantum systems, or a general-purpose quantum computer. By connecting a trusted quantum simulator to an unknown system and tuning the model of the simulator to counteract the unknown dynamics, the dynamics of the unknown system can be efficiently learned using approximate Bayesian inference[64–66]. This exponentially reduces the number of measurements needed to perform the simulation. Similarly, the universal quantum emulator algorithm[67] allows one to reconstruct quantum dynamics and the quantum Boltzmann training algorithm of ref. 61 allows states to be reconstructed, in time logarithmic in the dimension of the Hilbert space, which is exponentially faster than reconstructing the dynamics via classical tomography.

To use a quantum computer to help characterize a quantum system[65,66] or to accept input states for use in a quantum PCA algorithm, we must

face the substantial technical challenge of loading coherent input states. Nonetheless, because such applications do not require qRAM and offer the potential for exponential speedups for device characterization[22,61,65,66] they remain among the promising possibilities for near-term application of quantum machine learning.

Designing and controlling quantum systems

A major challenge in the development of quantum computation and information science involves tuning quantum gates to match the exacting requirements needed for quantum error correction. Heuristic search methods can help to achieve this in a supervised learning scenario[68,69] (for instance in the case of nearest-neighbour-coupled superconducting artificial atoms[69] with gate fidelity above 99.9% in the presence of noise) and thus to reach an accepted threshold for fault-tolerant quantum computing. A similar methodology has been successful in constructing a single-shot Toffoli gate, again reaching gate fidelity above 99.9%[70]. Genetic algorithms have been employed to reduce digital and experimental errors in quantum gates[71]. They have been used to simulate controlled-NOT gates by means of ancillary qubits and imperfect gates. Besides outperforming protocols for digital quantum simulations, it has been shown that genetic algorithms are also useful for suppressing experimental errors in gates[72]. Another approach used stochastic gradient descent and two-body interactions to embed a Toffoli gate into a sequence of quantum operations or gates without time-dependent control using the natural dynamics of a quantum network[73]. Dynamical decoupling sequences help to protect quantum states from decoherence, which can be designed using recurrent neural networks[74].

Controlling a quantum system is just as important and complex. Learning methods have also been very successful in developing control sequences to optimize adaptive quantum metrology, which is a key quantum building block in many quantum technologies. Genetic algorithms have been proposed for the control of quantum molecules to overcome the problem caused by changing environmental parameters during an experiment[75]. Reinforcement learning algorithms using heuristic global optimization, like the algorithm used for designing circuits, have been widely successful, particularly in the presence of noise and decoherence, scaling well with the system size[76–78]. One can also exploit reinforcement learning in gate-based quantum systems. For instance, adaptive controllers based on intelligent agents for quantum information demonstrate adaptive calibration and compensation strategies to an external stray field of unknown magnitude in a fixed direction.

Classical machine learning is also a powerful tool with which to extract theoretical insights about quantum states. Neural networks have recently been deployed to study two central problems in condensed matter, namely phase-of-matter detection[79,80] and ground-state search[81]. These succeeded in achieving better performances than established numerical tools. Theoretical physicists are now studying these models to understand analytically their descriptive power compared to traditional methods such as tensor networks. Interesting applications to exotic states of matter are already on the market, and have been shown to capture highly non-trivial features from disordered or topologically ordered systems.

Perspectives on future work

As we have discussed in this review, small quantum computers and larger special-purpose quantum simulators, annealers and so on seem to have potential use in machine learning and data analysis[15,21,22,36,48,82–95]. However, the execution of quantum algorithms requires quantum hardware that is not yet available.

On the hardware side, there have been great strides in several enabling technologies. Small-scale quantum computers with 50–100 qubits will be made widely available via quantum cloud computing (the 'Qloud'). Special-purpose quantum information processors such as quantum simulators, quantum annealers, integrated photonic chips, nitrogen vacancy centres (NV)-diamond arrays, qRAM, and made-to-order superconducting circuits will continue to advance in size and complexity. Quantum machine learning offers a suite of potential applications for small quantum computers[23–31,96–98] complemented and enhanced by special-purpose quantum information processors[21,22], digital quantum processors[70,73,78,99,100] and sensors[76,77,101].

In particular, quantum annealers with around 2,000 qubits have been built and operated, using integrated superconducting circuits that are, in principle, scalable. The biggest challenges for quantum annealers to implement quantum machine learning algorithms include improving connectivity and implementing more general tunable couplings between qubits. Programmable quantum optic arrays with around 100 tunable interferometers have been constructed using integrated photonics in silicon, but loss of quantum effects increases as such circuits are scaled up. A particularly important challenge for quantum machine learning is the construction of interface devices such as qRAM that allow classical information to be encoded in quantum mechanical form[52]. A qRAM to access N pieces of data consists of a branching array of $2N$ quantum switches, which must operate coherently during a memory call. In principle, such a qRAM takes time $O(\log N)$ to perform a memory call, and can tolerate error rates of up to $O(1/\log N)$ per switching operation, where $\log N$ is the depth of the qRAM circuit. Proof-of-principle demonstrations of qRAM have been performed, but constructing large arrays of quantum switches is a difficult technological problem.

These hardware challenges are technical in nature, and clear paths exist towards overcoming them. They must be overcome, however, if quantum machine learning is to become a 'killer app' for quantum computers. As noted previously, most of the quantum algorithms that have been identified face a number of caveats that limits their applicability. We can distill the caveats mentioned above into four fundamental problems.

(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) 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.
(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.
(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.

To avoid some of these problems, we could apply quantum computing to quantum, rather than classical, data. One aim therein is to use quantum machine learning to characterize and control quantum computers[66]. This would enable a virtuous cycle of innovation similar to that which occurred in classical computing, wherein each generation of processors is then leveraged to design the next-generation processors. We have already begun to see the first fruits of this cycle with classical machine learning being used to improve quantum processor designs[23–31,102–104], which in turn provide powerful computational resources for quantum-enhanced machine learning applications themselves[8,9,11,13,33–36].

Received 20 February; accepted 4 July 2017.

1. Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **65,** 386 (1958).
2. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521,** 436–444 (2015).

3. Le, Q. V. Building high-level features using large scale unsupervised learning. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* 8595–8598 (IEEE, 2013).
4. Schuld, M., Sinayskiy, I. & Petruccione, F. An introduction to quantum machine learning. *Contemp. Phys.* **56,** 172–185 (2015).
5. Wittek, P. *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (Academic Press, New York, NY, USA, 2014).
6. Adcock, J. *et al.* Advances in quantum machine learning. Preprint at https://arxiv.org/abs/1512.02900 (2015).
7. Arunachalam, S. & de Wolf, R. A survey of quantum learning theory. Preprint at https://arxiv.org/abs/1701.06806 (2017).
8. Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103,** 150502 (2009).
9. Wiebe, N., Braun, D. & Lloyd, S. Quantum algorithm for data fitting. *Phys. Rev. Lett.* **109,** 050505 (2012).
10. Childs, A. M., Kothari, R. & Somma, R. D. Quantum linear systems algorithm with exponentially improved dependence on precision. Preprint at https://arxiv.org/abs/1511.02306 (2015).
11. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nat. Phys.* **10,** 631–633 (2014).
12. Kimmel, S., Lin, C. Y.-Y., Low, G. H., Ozols, M. & Yoder, T. J. Hamiltonian simulation with optimal sample complexity. Preprint at https://arxiv.org/abs/1608.00281 (2016).
13. Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **113,** 130503 (2014).
This study applies quantum matrix inversion in a supervised discriminative learning algorithm.
14. Lloyd, S., Garnerone, S. & Zanardi, P. Quantum algorithms for topological and geometric analysis of data. *Nat. Commun.* **7,** 10138 (2016).
15. Dridi, R. & Alghassi, H. Homology computation of large point clouds using quantum annealing. Preprint at https://arxiv.org/abs/1512.09328 (2015).
16. Rebentrost, P., Steffens, A. & Lloyd, S. Quantum singular value decomposition of non-sparse low-rank matrices. Preprint at https://arxiv.org/abs/1607.05404 (2016).
17. Schuld, M., Sinayskiy, I. & Petruccione, F. Prediction by linear regression on a quantum computer. *Phys. Rev. A* **94,** 022342 (2016).
18. Brandao, F. G. & Svore, K. Quantum speed-ups for semidefinite programming. Preprint at https://arxiv.org/abs/1609.05537 (2016).
19. Rebentrost, P., Schuld, M., Petruccione, F. & Lloyd, S. Quantum gradient descent and Newton's method for constrained polynomial optimization. Preprint at https://arxiv.org/abs/1612.01789 (2016).
20. Wiebe, N., Kapoor, A. & Svore, K. M. Quantum deep learning. Preprint at https://arxiv.org/abs/1412.3489 (2014).
21. Adachi, S. H. & Henderson, M. P. Application of quantum annealing to training of deep neural networks. Preprint at https://arxiv.org/abs/arXiv:1510.06356 (2015).
22. Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B. & Melko, R. Quantum Boltzmann machine. Preprint at https://arxiv.org/abs/arXiv:1601.02036 (2016).
23. Sasaki, M., Carlini, A. & Jozsa, R. Quantum template matching. *Phys. Rev. A* **64,** 022317 (2001).
24. Bisio, A., Chiribella, G., D'Ariano, G. M., Facchini, S. & Perinotti, P. Optimal quantum learning of a unitary transformation. *Phys. Rev. A* **81,** 032324 (2010).
25. Bisio, A., D'Ariano, G. M., Perinotti, P. & Sedlák, M. Quantum learning algorithms for quantum measurements. *Phys. Lett. A* **375,** 3425–3434 (2011).
26. Sentís, G., Calsamiglia, J., Muñoz-Tapia, R. & Bagan, E. Quantum learning without quantum memory. *Sci. Rep.* **2,** 708 (2012).
27. Sentís, G., Guţă, M. & Adesso, G. Quantum learning of coherent states. *EPJ Quant. Technol.* **2,** 17 (2015).
28. Paparo, G. D., Dunjko, V., Makmal, A., Martin-Delgado, M. A. & Briegel, H. J. Quantum speedup for active learning agents. *Phys. Rev. X* **4,** 031002 (2014).
29. Dunjko, V., Friis, N. & Briegel, H. J. Quantum-enhanced deliberation of learning agents using trapped ions. *New J. Phys.* **17,** 023006 (2015).
30. Dunjko, V., Taylor, J. M. & Briegel, H. J. Quantum-enhanced machine learning. *Phys. Rev. Lett.* **117,** 130501 (2016).
This paper investigates the theoretical maximum speedup achievable in reinforcement learning in a closed quantum system, which proves to be Grover-like if we wish to obtain classical verification of the learning process.
31. Sentís, G., Bagan, E., Calsamiglia, J., Chiribella, G. & Muñoz Tapia, R. Quantum change point. *Phys. Rev. Lett.* **117,** 150502 (2016).
32. Faccin, M., Migdał, P., Johnson, T. H., Bergholm, V. & Biamonte, J. D. Community detection in quantum complex networks. *Phys. Rev. X* **4,** 041012 (2014).
This paper defines closeness measures and then maximizes modularity with hierarchical clustering to partition quantum data.
33. Clader, B. D., Jacobs, B. C. & Sprouse, C. R. Preconditioned quantum linear system algorithm. *Phys. Rev. Lett.* **110,** 250504 (2013).
34. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. Preprint at https://arxiv.org/abs/1307.0411 (2013).
35. Wiebe, N., Kapoor, A. & Svore, K. M. Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning. *Quantum Inf. Comput.* **15,** 316–356 (2015).
36. Lau, H.-K., Pooser, R., Siopsis, G. & Weedbrook, C. Quantum machine learning over infinite dimensions. *Phys. Rev. Lett.* **118,** 080501 (2017).
37. Aïmeur, E., Brassard, G. & Gambs, S. in *Machine Learning in a Quantum World* 431–442 (Springer, 2006).
38. Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **26,** 1484–1509 (1997).
39. Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information* (Cambridge Univ. Press, 2000).
40. Wossnig, L., Zhao, Z. & Prakash, A. A quantum linear system algorithm for dense matrices. Preprint at https://arxiv.org/abs/1704.06174 (2017).
41. Giovannetti, V., Lloyd, S. & Maccone, L. Quantum random access memory. *Phys. Rev. Lett.* **100,** 160501 (2008).
42. Lloyd, S. Universal quantum simulators. *Science* **273,** 1073–1078 (1996).
43. Vapnik, V. *The Nature of Statistical Learning Theory* (Springer, 1995).
44. Anguita, D., Ridella, S., Rivieccio, F. & Zunino, R. Quantum optimization for training support vector machines. *Neural Netw.* **16,** 763–770 (2003).
45. Dürr, C. & Høyer, P. A quantum algorithm for finding the minimum. Preprint at https://arxiv.org/abs/quant-ph/9607014 (1996).
46. Chatterjee, R. & Yu, T. Generalized coherent states, reproducing kernels, and quantum support vector machines. Preprint at https://arxiv.org/abs/1612.03713 (2016).
47. Zhao, Z., Fitzsimons, J. K. & Fitzsimons, J. F. Quantum assisted Gaussian process regression. Preprint at https://arxiv.org/abs/1512.03929 (2015).
48. Li, Z., Liu, X., Xu, N. & Du, J. Experimental realization of a quantum support vector machine. *Phys. Rev. Lett.* **114,** 140504 (2015).
49. Whitfield, J. D., Faccin, M. & Biamonte, J. D. Ground-state spin logic. *Europhys. Lett.* **99,** 57004 (2012).
50. Farhi, E., Goldstone, J. & Gutmann, S. A quantum approximate optimization algorithm. Preprint at https://arxiv.org/abs/1411.4028 (2014).
51. Aaronson, S. Read the fine print. *Nat. Phys.* **11,** 291–293 (2015).
52. Arunachalam, S., Gheorghiu, V., Jochym-O'Connor, T., Mosca, M. & Srinivasan, P. V. On the robustness of bucket brigade quantum RAM. *New J. Phys.* **17,** 123010 (2015).
53. Scherer, A. *et al.* Concrete resource analysis of the quantum linear system algorithm used to compute the electromagnetic scattering cross section of a 2D target. Preprint at https://arxiv.org/abs/1505.06552 (2015).
54. Denil, M. & De Freitas, N. Toward the implementation of a quantum RBM. In *Neural Information Processing Systems (NIPS) Conf. on Deep Learning and Unsupervised Feature Learning Workshop* Vol. 5 (2011).
55. Dumoulin, V., Goodfellow, I. J., Courville, A. & Bengio, Y. On the challenges of physical implementations of RBMs. Preprint at https://arxiv.org/abs/1312.5258 (2013).
56. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Estimation of effective temperatures in quantum annealers for sampling applications: a case study with possible applications in deep learning. *Phys. Rev. A* **94,** 022308 (2016).
57. Biamonte, J. D. & Love, P. J. Realizable Hamiltonians for universal adiabatic quantum computers. *Phys. Rev. A* **78,** 012352 (2008).
This study established the contemporary experimental target for non-stoquastic (that is, non-quantum stochastic) D-Wave quantum annealing hardware able to realize universal quantum Boltzmann machines.
58. Temme, K., Osborne, T. J., Vollbrecht, K. G., Poulin, D. & Verstraete, F. Quantum metropolis sampling. *Nature* **471,** 87–90 (2011).
59. Yung, M.-H. & Aspuru-Guzik, A. A quantum–quantum metropolis algorithm. *Proc. Natl Acad. Sci. USA* **109,** 754–759 (2012).
60. Chowdhury, A. N. & Somma, R. D. Quantum algorithms for Gibbs sampling and hitting-time estimation. *Quant. Inf. Comput.* **17,** 41–64 (2017).
61. Kieferova, M. & Wiebe, N. Tomography and generative data modeling via quantum Boltzmann training. Preprint at https://arxiv.org/abs/1612.05204 (2016).
62. Lloyd, S. & Terhal, B. Adiabatic and Hamiltonian computing on a 2D lattice with simple 2-qubit interactions. *New J. Phys.* **18,** 023042 (2016).
63. Ventura, D. & Martinez, T. Quantum associative memory. *Inf. Sci.* **124,** 273–296 (2000).
64. Granade, C. E., Ferrie, C., Wiebe, N. & Cory, D. G. Robust online Hamiltonian learning. *New J. Phys.* **14,** 103013 (2012).
65. Wiebe, N., Granade, C., Ferrie, C. & Cory, D. G. Hamiltonian learning and certification using quantum resources. *Phys. Rev. Lett.* **112,** 190501 (2014).
66. Wiebe, N., Granade, C. & Cory, D. G. Quantum bootstrapping via compressed quantum Hamiltonian learning. *New J. Phys.* **17,** 022005 (2015).
67. Marvian, I. & Lloyd, S. Universal quantum emulator. Preprint at https://arxiv.org/abs/1606.02734 (2016).
68. Dolde, F. *et al.* High-fidelity spin entanglement using optimal control. *Nat. Commun.* **5,** 3371 (2014).
69. Zahedinejad, E., Ghosh, J. & Sanders, B. C. Designing high-fidelity single-shot three-qubit gates: a machine-learning approach. *Phys. Rev. Appl.* **6,** 054005 (2016).
70. Zahedinejad, E., Ghosh, J. & Sanders, B. C. High-fidelity single-shot Toffoli gate via quantum control. *Phys. Rev. Lett.* **114,** 200502 (2015).
71. Zeidler, D., Frey, S., Kompa, K.-L. & Motzkus, M. Evolutionary algorithms and their application to optimal control studies. *Phys. Rev. A* **64,** 023420 (2001).
72. Las Heras, U., Alvarez-Rodriguez, U., Solano, E. & Sanz, M. Genetic algorithms for digital quantum simulations. *Phys. Rev. Lett.* **116,** 230504 (2016).
73. Banchi, L., Pancotti, N. & Bose, S. Quantum gate learning in qubit networks: Toffoli gate without time-dependent control. *npj Quant. Inf.* **2,** 16019 (2016).

74. August, M. & Ni, X. Using recurrent neural networks to optimize dynamical decoupling for quantum memory. Preprint at https://arxiv.org/abs/1604.00279 (2016).
75. Amstrup, B., Toth, G. J., Szabo, G., Rabitz, H. & Loerincz, A. Genetic algorithm with migration on topology conserving maps for optimal control of quantum systems. *J. Phys. Chem.* **99,** 5206–5213 (1995).
76. Hentschel, A. & Sanders, B. C. Machine learning for precise quantum measurement. *Phys. Rev. Lett.* **104,** 063603 (2010).
77. Lovett, N. B., Crosnier, C., Perarnau-Llobet, M. & Sanders, B. C. Differential evolution for many-particle adaptive quantum metrology. *Phys. Rev. Lett.* **110,** 220501 (2013).
78. Palittapongarnpim, P., Wittek, P., Zahedinejad, E., Vedaie, S. & Sanders, B. C. Learning in quantum control: high-dimensional global optimization for noisy quantum dynamics. *Neurocomputing* https://doi.org/10.1016/j.neucom.2016.12.087 (in the press).
79. Carrasquilla, J. & Melko, R. G. Machine learning phases of matter. *Nat. Phys.* **13,** 431–434 (2017).
80. Broecker, P., Carrasquilla, J., Melko, R. G. & Trebst, S. Machine learning quantum phases of matter beyond the fermion sign problem. Preprint at https://arxiv.org/abs/1608.07848 (2016).
81. Carleo, G. & Troyer, M. Solving the quantum many-body problem with artificial neural networks. *Science* **355,** 602–606 (2017).
82. Brunner, D., Soriano, M. C., Mirasso, C. R. & Fischer, I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat. Commun.* **4,** 1364 (2013).
83. Cai, X.-D. *et al.* Entanglement-based machine learning on a quantum computer. *Phys. Rev. Lett.* **114,** 110504 (2015).
84. Hermans, M., Soriano, M. C., Dambre, J., Bienstman, P. & Fischer, I. Photonic delay systems as machine learning implementations. *J. Mach. Learn. Res.* **16,** 2081–2097 (2015).
85. Tezak, N. & Mabuchi, H. A coherent perceptron for all-optical learning. *EPJ Quant. Technol.* **2,** 10 (2015).
86. Neigovzen, R., Neves, J. L., Sollacher, R. & Glaser, S. J. Quantum pattern recognition with liquid-state nuclear magnetic resonance. *Phys. Rev. A* **79,** 042321 (2009).
87. Pons, M. *et al.* Trapped ion chain as a neural network: error resistant quantum computation. *Phys. Rev. Lett.* **98,** 023003 (2007).
88. Neven, H. *et al.* Binary classification using hardware implementation of quantum annealing. In *24th Ann. Conf. on Neural Information Processing Systems (NIPS-09)* 1–17 (2009).
This paper was among the first experimental demonstrations of machine learning using quantum annealing.
89. Denchev, V. S., Ding, N., Vishwanathan, S. & Neven, H. Robust classification with adiabatic quantum optimization. In *Proc. 29th Int. Conf. on Machine Learning (ICML-2012)* (2012).
90. Karimi, K. *et al.* Investigating the performance of an adiabatic quantum optimization processor. Quantum Inform. *Process.* **11,** 77–88 (2012).
91. O'Gorman, B. A. *et al.* Bayesian network structure learning using quantum annealing. *EPJ Spec. Top.* **224,** 163–188 (2015).
92. Denchev, V. S., Ding, N., Matsushima, S., Vishwanathan, S. V. N. & Neven, H. Totally corrective boosting with cardinality penalization. Preprint at https://arxiv.org/abs/1504.01446 (2015).
93. Kerenidis, I. & Prakash, A. Quantum recommendation systems. Preprint at https://arxiv.org/abs/1603.08675 (2016).
94. Alvarez-Rodriguez, U., Lamata, L., Escandell-Montero, P., Martín-Guerrero, J. D. & Solano, E. Quantum machine learning without measurements. Preprint at https://arxiv.org/abs/1612.05535 (2016).
95. Wittek, P. & Gogolin, C. Quantum enhanced inference in Markov logic networks. *Sci. Rep.* **7,** 45672 (2017).
96. Lamata, L. Basic protocols in quantum reinforcement learning with superconducting circuits. *Sci. Rep.* **7,** 1609 (2017).
97. Schuld, M., Fingerhuth, M. & Petruccione, F. Quantum machine learning with small-scale devices: implementing a distance-based classifier with a quantum interference circuit. Preprint at https://arxiv.org/abs/1703.10793 (2017).
98. Monràs, A., Sentís, G. & Wittek, P. Inductive supervised quantum learning. *Phys. Rev. Lett.* **118,** 190503 (2017).
This paper proves that supervised learning protocols split into a training and application phase in both the classical and the quantum cases.
99. Tiersch, M., Ganahl, E. J. & Briegel, H. J. Adaptive quantum computation in changing environments using projective simulation. *Sci. Rep.* **5,** 12874 (2015).
100. Zahedinejad, E., Ghosh, J. & Sanders, B. C. Designing high-fidelity single-shot three-qubit gates: a machine learning approach. Preprint at https://arxiv.org/abs/1511.08862 (2015).
101. Palittapongarnpim, P., Wittek, P. & Sanders, B. C. Controlling adaptive quantum phase estimation with scalable reinforcement learning. In *Proc. 24th Eur. Symp. Artificial Neural Networks (ESANN-16) on Computational Intelligence and Machine Learning* 327–332 (2016).
102. Wan, K. H., Dahlsten, O., Kristjánsson, H., Gardner, R. & Kim, M. S. Quantum generalisation of feedforward neural networks. Preprint at https://arxiv.org/abs/1612.01045 (2016).
103. Lu, D. *et al.* Towards quantum supremacy: enhancing quantum control by bootstrapping a quantum processor. Preprint at https://arxiv.org/abs/1701.01198 (2017).
104. Mavadia, S., Frey, V., Sastrawan, J., Dona, S. & Biercuk, M. J. Prediction and real-time compensation of qubit decoherence via machine learning. *Nat. Commun.* **8,** 14106 (2017).
105. Rønnow, T. F. *et al.* Defining and detecting quantum speedup. *Science* **345,** 420–424 (2014).
106. Low, G. H., Yoder, T. J. & Chuang, I. L. Quantum inference on Bayesian networks. *Phys. Rev. A* **89,** 062315 (2014).
107. Wiebe, N. & Granade, C. Can small quantum systems learn? Preprint at https://arxiv.org/abs/1512.03145 (2015).
108. Wiebe, N., Kapoor, A. & Svore, K. M. Quantum perceptron models. *Adv. Neural Inform. Process. Syst.* **29,** 3999–4007 (2016).
109. Scherer, A. *et al.* Concrete resource analysis of the quantum linear-system algorithm used to compute the electromagnetic scattering cross section of a 2D target. *Quantum Inform. Process.* **16,** 60 (2017).

Acknowledgements J.B. acknowledges financial support from AFOSR grant FA9550-16-1-0300, Models and Protocols for Quantum Distributed Computation. P.W. acknowledges financial support from the ERC (Consolidator Grant QITBOX), Spanish Ministry of Economy and Competitiveness (Severo Ochoa Programme for Centres of Excellence in R&D SEV-2015-0522 and QIBEQI FIS2016-80773-P), Generalitat de Catalunya (CERCA Programme and SGR 875), and Fundacio Privada Cellex. P.R. and S.L. acknowledge funding from ARO and AFOSR under MURI programmes. We thank L. Zheglova for producing Fig. 1.

Author Contributions All authors designed the study, analysed data, interpreted data, produced Box 3 Figure and wrote the article.

Author Information Reprints and permissions information is available at www.nature.com/reprints. The authors declare no competing financial interests. Readers are welcome to comment on the online version of the paper. Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations. Correspondence should be addressed to J.B. (jacob.biamonte@qubit.org).

Reviewer Information *Nature* thanks L. Lamata and the other anonymous reviewer(s) for their contribution to the peer review of this work.