Quantum Machine Learning: A Review and Current Status

Chapter · January 2021

DOI: 10.1007/978-981-15-5619-7_8

Quantum Machine Learning: A Review and Current Status

Nimish Mishra, Manik Kapil, Hemant Rakesh, Amit Anand, Nilima Mishra, Aakash Warke, Soumya Sarkar, Sanchayan Dutta, Sabhyata Gupta, Aditya Prasad Dash, Rakshit Gharat, Yagnik Chatterjee, Shuvarati Roy, Shivam Raj, Valay Kumar Jain, Shreeram Bagaria, Smit Chaudhary, Vishwanath Singh, Rituparna Maji, Priyanka Dalei, Bikash K. Behera, Sabyasachi Mukhopadhyay, and Prasanta K. Panigrahi

Abstract Quantum machine learning is at the intersection of two of the most sought after research areas—quantum computing and classical machine learning. Quantum machine learning investigates how results from the quantum world can be used to solve problems from machine learning. The amount of data needed to reliably

N. Mishra
Department of Computer Science and Engineering, Indian Institute of Information Technology, Kalyani, West Bengal, India

M. Kapil
Department of Physics, Indian Institute of Technology Guwahati, Guwahati 781039, Assam, India

H. Rakesh
Department of Computer Science and Engineering, Nitte Meenakshi Institute of Technology, Yelahanka, Bangalore, India

A. Anand
Department of Mechanical Engineering, Indian Institute Of Engineering Science And Technology, Shibpur 711103, West Bengal, India

N. Mishra
Department of Electronics and Telecommunication, International Institute of Information Technology, Bhubaneswar 751003, Odisha, India

A. Warke · V. Kumar Jain
Department of Physics, Bennett University, Greater Noida 201310, Uttar Pradesh, India

S. Sarkar · R. Gharat
Department of Physics, National Institute of Technology Karnataka, Karnataka 575025, India

S. Dutta
Department of Electronics and Telecommunication, Jadavpur UniverGreater Noidasity, Kolkata, India

S. Gupta
Department of Physics, Panjab University, Chandigarh 160036, Chandigarh, India

A. Prasad Dash · S. Roy
Department of Physical Sciences, Indian Institute of Science Education and Research Berhampur, Berhampur 760010, Odisha, India

N. Sharma et al. (eds.), *Data Management, Analytics and Innovation*,
Advances in Intelligent Systems and Computing 1175,
https://doi.org/10.1007/978-981-15-5619-7_8

train a classical computation model is evergrowing and reaching the limits which normal computing devices can handle. In such a scenario, quantum computation can aid in continuing training with huge data. Quantum machine learning looks to devise learning algorithms faster than their classical counterparts. Classical machine learning is about trying to find patterns in data and using those patterns to predict further events. Quantum systems, on the other hand, produce atypical patterns which are not producible by classical systems, thereby postulating that quantum computers may overtake classical computers on machine learning tasks. Here, we review the previous literature on quantum machine learning and provide the current status of it.

Keywords Quantum machine learning · Quantum renormalization procedure · Quantum hhl algorithm · Quantum support vector machine · Quantum classifier · Quantum artificial intelligence · Quantum entanglement · Quantum neural network · Quantum computer

Y. Chatterjee
Department of Physics and Astronomy, National Institute of Technology Rourkela, Odisha 769008, India

S. Raj
School of Physical Sciences, National Institute of Science Education and Research, HBNI, Jatni 752050, Odisha, India

S. Bagaria
Department of Chemical Engineering, MBM Engineering College, Jodhpur, India

S. Chaudhary
Department of Physics, Indian Institute Of Technology, Kanpur, Kalyanpur, Kanpur, India

V. Singh
Indian Institute of Technology (Indian School of Mines), Dhanbad 826004, Jharkhand, India

R. Maji
Department of Physics, Central University of Karnataka, Karnataka 585367, India

P. Dalei · B. K. Behera (✉)
Bikash's Quantum (OPC) Private Limited, Balindi, Mohanpur, Nadia 741246, West Bengal, India
e-mail: bkb18rs025@iiserkol.ac.in

B. K. Behera · P. K. Panigrahi
Department of Physical Sciences, Indian Institute of Science Education and Research Kolkata, Mohanpur 741246, West Bengal, India
e-mail: pprasanta@iiserkol.ac.in

S. Mukhopadhyay
BIMS Kolkata, Kolkata 700 097, West Bengal, India
e-mail: mukhopadhyaysabyasachi@gmail.com

1 Introduction

Everyday experience in our life makes up our classical understanding, however, it's not the ultimate underlying mechanism of nature. Our surrounding is just the emergence of the underlying and more basic mechanics known as quantum mechanics. Quantum phenomenons don't match with our everyday intuition. In fact, for a very long time in the history of science and human understanding, these underlying mechanics were hidden from us. It is only in the last century we came to observe this aspect of nature. As the research progressed, we developed theories and mathematical tools from our renowned scientists. Quantum theory being a probabilistic theory attracts a lot of philosophical debates with it. Many quantum phenomenona such as the collapse of the wave function, quantum tunnelling, quantum superposition, etc still fascinates us. The true quantum nature of reality is still a mystery to our understanding. Quantum technologies aim to use these physical laws to our technological advantage. It is in the last 10 to 20 years the applications based on quantum mechanical laws have improved leaps and bound with the aim to replace or go parallel to the classical machines.

Today quantum technologies have three main specializations: quantum computing (using quantum phenomena to computational tasks), quantum information (using quantum phenomena to aid in transfer, storage and reception of information) and quantum cryptography (using quantum phenomena to devise superior cryptography techniques). The power of quantum computation comes from the expansive permutations which make quantum computers twice as memory-full with the addition of each qubit. To specify N bits classical bits system, we need to have N bits of binary numbers. Now, we know in quantum systems the two possible definite states that are $|0\rangle$ and $|1\rangle$. A general state of a bipartite quantum system can be represented as $\Phi = \alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|10\rangle + \delta\,|11\rangle$, we can easily see that from a two-qubit quantum system we get four classical bits of information (α, β, γ, δ). Similarly, from the N-qubit quantum system, we can get 2^N bits of classical information. A mathematical model of computation that conceptually defines the aspect of a machine: contains an automaton, a strip with some symbols, a read-write head and a set of rules (encapsulated by the transition function) which can manipulate symbols on the strip, can be defined as a Turing machine. Quantum computers are universal Turing machines. Quantum mechanics allows superposition of quantum states resulting in quantum parallelism which can be exploited to perform probabilistic tasks much faster than any classical means.

Quantum computers are known to solve problems that cannot be solved using a classical computer. One such example includes the factorization of large numbers using Shor's algorithm [1]. Moreover, if classical and quantum computers are simultaneously utilized for the same purpose, cases can exist in which quantum algorithms prove to be more efficient. These algorithms belong to a particular complexity class called BQP (Bounded-error Quantum Polynomial time) [2]. Another difficult problem in the classical computation model, that is, solving the Pell's equation, is efficiently solvable in quantum computation model. Similarly, the Non-Abelian hidden subgroup problem is shown to be efficiently solved using a quantum algorithm [2]. Quantum computers have shown remarkable improvements in the field of optimization and simulation. It includes computing the properties of partition functions, performing approximate optimization and simulating different quantum systems. Quantum simulations have applications in the field of quantum optics and condensed matter physics as well [3].

"Give machines the ability to learn without explicitly programming them"—Arthur Samuel, 1955. The idea of machine learning can be derived from this statement. Briefly, an algorithm that holds the capacity to analyze a huge amount of data, make a pattern out of it and predict the future outcomes can be termed as a machine learning algorithm. The entire concept of machine learning is to optimize a constrained multivariate function; the algorithms to achieve such optimizations are the core of data mining and data visualization in use today. The decision function works on mapping input points to output points. This is the result of optimization. There are several exceptions to this simplistic rule, however, such optimizations are central to learning theory. Applications of such algorithms lead to artificial intelligence. Classically, there are three types of methods in machine learning—1. *Supervised Machine Learning* [4] where we teach machines to work on the basis of data which are already labeled with some of these' characteristics, 2. *Unsupervised Machine Learning* [5] where no labelled data is provided to machines, they analyze these data on the basis of similarities and dissimilarities of their classes, 3. *Reinforcement Learning* [6] where machines analyze our feedbacks and learn. In quantum machine learning, the most common supervised machine learning algorithm is *Quantum Support Vector Machine* [7] wherein higher dimension vector space optimisation boundary is used to classify the classes of labelled data, *Principal Components Analysis* [8] is one of the most common algorithms. It makes a pattern of huge not-labelled data and effectively reduces it to make it easier for further analysis, this is essentially the quantum counterpart of Unsupervised Machine Learning.

Quantum computing also servers useful for financial modeling [9]. The randomness that is inherent to quantum computers is analogous to the stochastic nature of financial markets [10]. Today classical computers conduct high frequency stock exchange worth millions of dollars every second. The power of quantum computers can be used to solve such systems. Weather forecasting has been a long goal for scientists, however, predicting weather conditions require taking into account an enormously large number of variables causing classical simulations to be unreasonably lengthy. With their parallel computing power, quantum computers can be used to create better climate models. Various difficulties arise when we try to model complex molecules in classical computers. Chemical reactions lead to the formation of highly entangled quantum superposition states, and are thus quantum in nature [11]. Such states can be modeled accurately with the help of a quantum computer.

Yet another fascinating specialization in the field of quantum technologies is quantum cryptography. Quantum cryptography can be defined as the utilization of quantum mechanical properties in order to carry out cryptographic tasks. Publication of Wiesner's paper in 1983, led to the origination of this field. A well known example of quantum cryptography is the quantum key distribution (QKD). In 1984, thanks to Bennett and Brassard, it was possible to obtain a complete protocol of extremely secure quantum key distribution. It is currently known to be the BB84 protocol [12]. This protocol drew a significant amount of attention from the cryptographic community as such security was unattainable by classical means [13].

In this paper, we aim to give a brief description of Quantum Machine Learning and its correlation with AI to unleash the future scope and application of these in human life. We will see how the quantum counterpart of machine learning is way faster and more efficient than the classical machine learning. In the following section, we describe the basic fundamentals of classical machine learning and its methods. Detailed discussion regarding ways by which a machine can learn has also been described in this section. In Sect. 3, aspects of classical machine learning which can be understood and applied to the quantum domain and its implementation have been discussed in detail. Furthermore, subsection on quantum neural networks covers a general introduction to neural networks and variants as they stand in deep learning, background work which has already been processed on quantum neural networks, quantum neuron and quantum convolutional neural network as a mark of deep learning's transition to quantum computers. In Sect. 4, we discuss in detail about how learning and renormalization procedures invite the application of machine learning. Using previously established relationships between the inputs and the outputs, machine learning derives patterns which are then used to make sense of previously unknown inputs. Some features are far too complex for standard numerical modeling methods. This is where techniques of machine learning play a vital role in solving problems. In Sect. 5, we discuss in detail about quantum HHL algorithm's much needed importance in solving linear systems. It is known to be a revolutionary algorithm that attempts to estimate solutions of linear systems of equations in logarithmic time. It is applicable as a subroutine in several quantum machine learning algorithms. As we know, data classification is one of the most important tools of machine learning today, we discuss in detail in Sect. 6 about quantum support vector machine. Quantum support vector machines are data classification algorithms that allow classification of data into one of two different categories. This can be done by drawing a hyperplane between our training data. This helps in the identification of data that belongs to a specific category. After this, we can enter our data to be classified and get our result based on its position relative to the hyperplane. Many quantum SVM algorithms exist today and quantum SVMs have been experimentally tested and turned out to be successful. In the next section, i.e. Sect. 7, we discuss in detail about quantum classifiers and its recent developments. A quantum computing algorithm which analyzes quantum states of existing data in order to determine or categorize new data to respective classes is known to be a quantum classifier. Approaches in quantum classifiers have been discussed in explicit detail in this section. In Sect. 8, an overview of applications of quantum machine learning to physics have been discussed. Machine learning methods can efficiently be used in vivid quantum information processing problems which include quantum signal processing, quantum metrology, quantum control, etc. The following section, i.e. Sect. 9, we firstly cover basics of how machine learning can be applied to the idea of quantum artificial intelligence. This section covers two to three AI simulations and explores whether these simulations can be quantized. It explores the possibility of relating quantum computing to artificial intelligence. Lastly, it cites some new developments in the field of quantum artificial intelligence. Section 10 covers some examples of how entangled-state helps ML to be more accurate, efficient and sensitive. Likewise, machine learning also can be used to measure how entangled a state is, so both can be used to make each other better and more efficient than before. Section 11 describes the motivation of quantum neural networks through classical neural networks. Background work in quantum neural networks has also been discussed in detail. We then put forth the concept of quantum neuron which is then followed by the comprehensive discussion of quantum convolutional neural networks. Section 12 reports the use of artificial neural networks to solve many-body quantum systems. This happens to be one of the most challenging areas of quantum physics. Recent use of neural networks for the variational representation of quantum many-body states has initiated a huge attentiveness in this field. Since neural networks assist in the representation of quantum states efficiently, the question of whether or not a simulation of various quantum algorithms is possible has been addressed in Sect. 13. In the next section, i.e. Sect. 14, we report in detail about learning algorithms that are implemented on real quantum hardware. Recent developments in this area of research have been discussed as well. The last section narrates the use of machine learning frameworks, especially RBM (Restricted Boltzmann Machine) networks to represent the quantum many-body wave functions. The possibility of using neural networks to study quantum algorithms and the recent developments in this direction are discussed briefly, thus giving a proper conclusion and a futuristic vision.

2 Classical Machine Learning

In this section, we discuss basic machine learning types and models to set the context for various methods by which machines learn. Broadly, a machine may learn either from data or by interaction. We discuss both the learning methods in detail in Sect. 2.1. After this, we discuss the most widely used machine learning models that implement the fore mentioned learning types in Sect. 2.2. Machine learning algorithms were built decades ago when fast computation was a difficult task. Nowadays, with increased computational capabilities, implementing these algorithms successfully is a fairly achievable task. A certain characterization on the basis of ease or difficulty in implementation and computational resources required for implementation can be done for ML algorithms. This is discussed under Sect. 2.3, i.e Computational learning theory.

2.1 *How Does a Machine Learn?—Learning from Data and Learning from Interaction*

Machine learning has mainly three canonical categories of learning— supervised, unsupervised and reinforcement learning. Fundamentally, supervised and unsupervised learning are based on data analysis and data mining tasks. Whereas reinforcement learning is an interaction-based learning, where learning enhances sequentially at every step. We discuss each learning type in the following sections.

2.1.1 Supervised Learning

In supervised learning, we are provided with a training set D which contains a number of input–output pairs ($\mathbf{x}$, t). The input $\mathbf{x}$ could be in general an n-dimensional vector. The primary aim is to infer relationship—linear or nonlinear—between the inputs and outputs, and predict the output for yet unobserved input values. We want to be able to predict the output $\hat{t}(x)$ for any input x.

A potent example of this is spam classifier. Based on a training set of emails classified as *spam* or *not-spam*, the classifier labels future emails as either *spam* or *not-spam*.

To characterize the quality of the prediction made, a loss function is used. Depending on the context, a variety of loss functions can be used which quantify how far the prediction $\hat{t}(x)$ has been from the actual output $t(x)$. The goal is to minimize this loss function $f(\hat{t}(x), t(x))$

Predicting the probability distribution function $p(x, t)$, is a three-step process:

1. **Model Selection**: We take the probability distribution function to be from a family of functions parameterized by some vector Θ. This is also called *inductive bias*. We can specify the particular parametric family of distribution functions in two ways—in generative models, we specify the family of point distributions $p(x, t|\theta)$, while in discriminative models, we parameterize the predictive distribution $p(t|x, \theta$
2. **Learning**: The second step is learning where given a training set **D**, we optimize a learning parameter (here, the loss function $f(\hat{t}(x), t(x)$. Thus, we find out the parameter θ and by extension the distribution from the family of distribution parameterized by Θ.
3. **Inference**: The third and the final step is inference. Here the trained model is employed to predict the output $\hat{t}(x)$ in line with minimizing the loss parameter. Had we used the generative model in step 1, we would need to use marginalization to get to the actual predictive distribution $p(t|x)$, while the discriminative model would directly yield the predictive distribution.

2.1.2 Unsupervised Learning

The main difference between unsupervised learning and supervised learning is the fact that in unsupervised learning, we do not have labelled data points. The training set D contains a set of inputs $\{x\}$. Thus, we only have the input data points. The general goal of the process is to extract useful properties from this data. We are interested in the following tasks:

1. **Density estimation**: Based on the training set, here we directly try to estimate the probability distribution $p(x)$.
2. **Clustering**: We could want to segregate the data in various clusters based on their similarities and differences. Here, the notion of what similarity is and what difference is, depends on the case at hand, and the particular domain in which you are working. Thus, even though the input data set was not endowed with any labels or classifications, through clustering we readily partition the data points into groups.
3. **Dimensionality Reduction**: The process involves representing the data point x_n in different spaces, thus being better able to visualize the correlations between various factors. This representation is generally done in a space of lower dimensions that better helps in extracting the relationship between the various components.
4. **Generation of new samples**: Given the data set D, we could want to generate new samples that would follow the probability distribution of the training data approximately. A relevant example is how sports scientists predict the actions of athletes using the same.

Just as was the case for supervised learning, here too it is a three-step process: Model selection, learning and using the learned model for clustering or generation of new samples.

2.1.3 Reinforcement Learning

Reinforcement Learning includes making sequential decisions in order to optimize some parameters. It differs from supervised learning, in that, supervised learning included a training set D with input–output pairs where the output is the "correct" answer or the correct characterization of the input. In the case of reinforcement learning, suboptimal paths taken by the algorithms need not be corrected immediately. Reinforcement learning could be seen as the middle ground between supervised and unsupervised learning as there does not exist immediate correct output to the input but there exists some sort of supervision in terms of if the series of steps taken are in the right direction or not.

The reinforcement algorithm receives feedback from the environment in place of a desired output for each input. And, this happens only after the algorithm has chosen an output for the given input. The feedback tells the algorithm about how well the chosen steps have helped or harmed in fulfilling the goals. The environment with which the algorithm interacts is formulated as Markov Decision Process.

2.2 *Machine Learning Models*

2.2.1 Artificial Neural Networks

Artificial Neural Networks belong to a class of computational models inspired by the biological neural network. Broadly, they mimic the workings of the natural neural network. Just as is required for a certain excitation for a natural neuron to fire, and the series of neurons firing determine the action that is to be taken, for artificial neural networks too, the system abides by the same set of rules at a broader level. Depending on the particular type of problem we are facing, different types of neural network models are used. A general architecture of a neural network could be understood in terms of layers of neurons which fire according to the firing of the neurons in the previous layer.

Feed-Forward Neural Networks are used as classifiers. If one wanted to map any input x to an output category y, one could define a function $y = f(x; \theta)$ that computes and learns the value of the parameter θ that results in the best function approximation. The nomenclature is derived from the fact that the signal flows in only one direction. Given a particular set of input at the first layer of neurons, the neurons in the subsequent layers fire to provide the classification.

Convolutional Neural Network is mostly used to classify images. Here, there is a vector of weights and biases which determines the output value given the inputs. There are multiple hidden layers between the input and the output layers. There is also an activation function (commonly taken to be RELU Layer) and is followed by subsequent pooling layers. The nomenclature derives from the fact that convolution operation is performed using a kernel. There could also be backpropagation to optimize how the weights are distributed.

Recurrent Neural Network is a type of neural network where the input to the current step is the output of the previous step. Predictive text is one of the most user-known applications of these. To predict the next word, the algorithm needs to store the previous word. RNNs turn the independent activation into dependent activation by making the weights and biases uniform across different layers.

Implementation of the Artificial Neural Network The neural network design consists of the input layer, output layer and the hidden layers. The input dataset is converted into an array of input to be fed into the network. Each input set comes with label value. Once it is fed to the network, the network is trained to determine the output label function of the fed dataset. The deviation from the true label value gives the error, the training parameter thus determines which corresponds to the minimum error.

The basic neural network operates with the help of three processes—forward propagation, backward propagation and updating the weight associated with the neuron. A neuron in any m layer receives the input from the (m–1) layer and conveys the output to the (m+1) layer. The value obtained with multiplying the input parameter with the weighted vector is fed to the neurons. In the input layer, a bias is also given.

Forward Propagation employs the methods of preactivation and activation. Preactivation involves feeding the network with weighted inputs. It is the linear transformation of the inputs which decides on the further passing of the input through the network. Activation [14] is the nonlinear transformation of the weighted inputs.
Backpropagation serves the most important step in the operation of a neural network. When the deviation of the obtained label value from the true label value is calculated, backpropagation helps in minimizing the error value. The training parameter is updated through each iteration until the error value is minimized. In backpropagation, we travel from the output layer to the input layer. It functions by employing the basic types of gradient descent algorithms to optimize the function. After forward propagation of any input through a neuron with respect to an assumed weight, the error is calculated. The weight with respect to the neuron corresponding to the error is then updated and the error function is changed. Similarly, the weight of every neuron is updated while backpropagating from output to input layer. Thus, the final training parameter is obtained. The analysis of the error value with the corresponding training parameter over each iteration presents the method in which the function of the neural network is implemented for performing different algorithms. Thus, the network is trained.

2.2.2 Support Vector Machines for Supervised Learning

The concept of Support Vector Machine (SVM), was developed by Boser, Guyon and Vapnik in COLT-92, in the year 1992. SVMs are one of the several models belonging to the family of generalized linear classifiers based on supervised learning, and are used for classification and regression. SVMs are systems based on hypothesis space of a linear function in a high dimensional feature space. They are trained with optimization learning algorithms that implement learning bias (based on statistical theory).

A potent example of SVM is handwriting recognition. Pixel maps are fed as inputs to such classifiers, and their accuracy is comparable to complicated neural networks with specialized features for handwriting recognition. Many applications, especially for pattern classification and regression-based applications due to promising features like better empirical performance of SVMs have come forth. Some examples include handwriting analysis, face analysis and so on.

SVMs perform better than neural networks in some scenarios because it uses the Structural Risk Minimization (SRM) principle, which is superior to traditional Empirical Risk Minimization (ERM) principle in use in conventional neural networks. The core difference between the two—minimization of the upper bound on the expected risk by SRM as opposed to the minimization of an error on training data by ERM—allows SVMs to generalize better than conventional neural networks, and are thus better suited to statistical learning. Due to this, modern times have seen increased applications of SVMs to regression problems, in addition to traditional classification problems.

Why SVM?: Neural networks show good results for several supervised and unsupervised learning problems. The most common—Multilayer perceptrons (MLPs)—include several properties like universal approximation of continuous nonlinear functions, learning with input–output patterns and advanced network architectures with multiple inputs and outputs. The problem with these is the existence of several local minima where the optimization algorithm may get stuck, as well as the problem of determining the optimal number of neurons, to be used in the neural network architecture. There is one other problem: convergence of a neural network solution does not guarantee a unique solution.

2.3 Computational Learning Theory

Given the number of machine learning algorithms available, there arises a need to characterize the capabilities of these machine learning algorithms. It is natural to wonder if we can classify machine learning problems as inherently difficult or easy [15]. With this come the obvious questions about quantifying a 'suitable' or 'successful' machine learning algorithm for various classes of problems. The success of a machine learning algorithm can depend upon several parameters like sample complexity, computational complexity, mistake bound, etc. Computational learning theory or COLT is a subfield of artificial intelligence in which the theoretical limits on machine learning algorithms, encompassing several classes of learning scenarios, are analyzed [16]. The standard procedure in COLT is to mathematically specify an environment or problem, apply learning algorithms on the problem, and characterize the performance of an optimal learning algorithm [15]. To study the aspects of COLT, we consider the *probably approximately correct* learning model or more colloquially called the PAC model.

The basic PAC learning model [17] introduced by Valiant can quantify learnability. We consider the case of learning Boolean valued concepts from training data. Consider the set of all possible instances over which target functions can be defined. Let this set be X. Let C be some set of target concepts our learner L has to learn. C is essentially a subset of X. The instances in X are generated at random according to probability distribution D. The learner L learns from target concepts in C and considers some set H of possible hypotheses describable by conjunctions of n attributes that define elements in X. After learner L learns from a sequence of training examples of target concept c in C, L outputs some hypothesis h from H which is L's estimate of c. Output hypothesis h is only an approximation of actual target concept c. The error in approximation or true error of hypothesis h with respect to concept c, denoted by $\text{error}_D(\text{h})$, is the probability that h will misclassify an instance drawn from D at random. We aim to identify classes of target concepts that can be reliably learned from a reasonable number of training examples. Ideally, for a sufficient number of training examples, we require $\text{error}_D(\text{h})$ to be 0. This assumption is practically impossible owing because multiple consistent hypotheses may exist for c, and there is always a nonzero finite probability that a training example may be misleading for the learner.

Therefore, in a practical scenario, we focus on minimizing error$_{\mathcal{D}}$(h) and limiting the number of training examples required.

For some class C of target concepts learned by learner L using hypothesis space H, let δ be an arbitrarily small constant bounding the probability of failure or the probability of misclassifying c. Also let ϵ denote an arbitrarily small constant bounding the upper limit of error$_{\mathcal{D}}$(h) such that error$_{\mathcal{D}}$(h) is less than ϵ. C is said to be PAC learnable by L using H if for all c belonging to C, distributions D over X, ϵ such that $0 < \epsilon < \frac{1}{2}$ and δ such that $0 < \delta < \frac{1}{2}$, learner L with probability at least $(1 - \delta)$ will output a hypothesis h belonging to H such that error$_{\mathcal{D}}$(h) is less than ϵ, in time that is polynomial in $\frac{1}{\epsilon}$, $\frac{1}{\delta}$, n and size(c) [15]. Thus, for a PAC learnable concept, L must learn from a polynomial number of training examples. Here we defined PAC learnability of conjunctions of Boolean literals. PAC learnability can be defined for other concept classes too.

PAC learnability greatly depends upon the number of training examples required by the learner. The sample complexity of the problem is growing in the number of training examples with problem size. Mathematical formulations for sample complexity for finite and infinite hypotheses spaces are available. Generally, the sample complexity of infinite hypotheses spaces is illustrated by the Vapnik-Chervoneniks dimension [18]. Collectively, PAC learnability and bounds of sample complexity relate to computational complexity. A number of training examples from which learner learns in polynomial time of PAC learnable bounds defines a finite time per training example. Here comes the need of handling enormous data in the least possible time.

3 Quantum Machine Learning

Training the machine to learn from the algorithms implemented to handle data is the core of machine learning. This field of computer science and statistics employs artificial intelligence and computational statistics. The classical machine learning method, through its subsets of deep learning (supervised and unsupervised) helps to classify images, recognize pattern and speech, handle *big data* and many more. However, as of today, a huge amount of data is being generated. New approaches, therefore, are required to manage, organize and classify such data. Classical machine learning can usually recognize patterns in data, but the few problems requiring huge data cannot be efficiently solved by classical algorithms. Companies dealing in big database management are aware of these limitations, and are thus actively looking for alternatives: quantum machine learning being one of them.

The use of quantum phenomena like superposition and entanglement to solve problems in classical machine learning paves the way to quantum machine learning [19]. algorithms in quantum systems, by Quantum computers, use the several superposition states $|0\rangle$ and $|1\rangle$ to allow any computation procedure at the same time, providing an edge over classical machine learning. In the quantum machine learning techniques, we develop quantum algorithms to operate the classical algorithms with the use of a quantum computer. Thus, data can be classified, sorted and analyzed using the quantum algorithms of supervised and unsupervised learning methods. These methods are again implemented through models of a quantum neural network or support vector machine.

3.1 Implementation of Quantum Machine Learning Algorithms

In the implementation of algorithms, we broadly discuss the process of the two major learning processes—supervised and unsupervised learning [20]. The pattern is learned observing the given set of training examples in case of supervised learning. While finding the structure from some clustered data set is done in unsupervised learning.

Quantum clustering technique [20] uses the quantum Lloyd's algorithm to solve the k-means clustering problem. It basically uses repetitive procedures to obtain the distance of the centroid of the cluster. The basic methods involve choosing randomly an initial centroid and assigning every vector to the cluster with the closest mean. Repetitive calculation and updating the centroid of the cluster should be done till the stationary value is obtained. The quantum algorithm speeds up the process in comparison to the classical algorithm. For an N-dimensional space, it demands O(Mlog(MN)) time to run the step for the quantum algorithm.

Quantum Neural Network [21] model is the technique of deep supervised learning to train the machine to classify data, recognize patterns or images. It is a feed-forward network. The basic principle is to design the circuits with qubits (being the analogue of neurons) and rotation gates (being the analogous of weights used in classical neural networks). The network learns from a set of training examples. Every input string comes with a label value. The function of the network is to obtain the label value of the data set and minimize the deviation of the obtained label from the true label. The focus is to obtain the training parameter that gives the minimum error. The training parameter is updated through every iteration. Error minimization is done by the backpropagation technique, which is based on gradient descent principle.

Quantum Decision Tree [22] employs quantum states to create classifiers. Decision trees are like normal tree structures in Computer Science: with one starting node named the root having no incoming edge and all outgoing edges leaving to other internal nodes or leaves. In these structures, the answer to a question is classified as we move down. The node contains a decision function that decides the direction of movement of the input vector along the branches and leaves. The quantum decision tree learns from training data: each node basically splits the training data set into subsets based on discrete function. Each leaf of the decision tree is assigned to an output class based on the target attributes desired. Thus, the quantum decision tree classifies data among its different components: leaves and root/internal nodes (which contain decisions based on which one of their child nodes are traversed for further classification).

Quantum machine learning provides a huge scope in computing the techniques done in classical machine learning on a quantum computer. The entanglement and superposition of the basic qubit states provide an edge over classical machine learning. Apart from neural networks, clustering methods, decision trees, quantum machine algorithms have been proposed for several other applications of image and pattern classification, and data handling. Further implications of the algorithms has been discussed in the paper.

4 Application of Machine Learning for Learning Procedure and Renormalization Procedure

Machine learning derives patterns from data in order to make sense of previously unknown inputs. Some features are far too complex for standard numerical modelling methods. This is where techniques of machine learning play a vital role in solving problems. Recent progress in the field of machine learning has shown significant promise in applying ML tools like classification or pattern recognition to identify phases of matter or nonlinear approximation of arbitrary functions using neural networks. Machine learning has become the central aspect of modern society: in web searching, in recommendation systems, in content filtering, in cameras and smartphones, speech-to-text and text-to-speech converters, identify fake news, sentiment analysis and many more. More often, such applications require deep neural networks (having several hidden layers with many neurons) called deep learning.

The renormalization group (RG) approach underlies the discovery of asymptotic freedom in quantum chromodynamics and of the Kosterlitz–Thouless phase transition as well as the seminal work on critical phenomena. The RG approach is a conceptual framework comprising techniques such as density matrix RG, functional RG, real-space RG, among others. The essence of RG lies in determining the 'relevant' degrees of freedom and integrating the 'irrelevant' ones iteratively. We thus arrive at a universal, low-energy effective theory. The RG procedure, which reveals the universal properties that, in turn, determine their physical characteristics, systematically retains the 'slow' degrees of freedom and integrates out the rest. The main problem, however, is to identify the important degrees of freedom.

Consider a feature map which transforms any data X to a different, more coarse grain scale

$$x \rightarrow \phi_\lambda(x) \tag{1}$$

The RG theory requires that the Free Energy F(x) is scaled, to reflect that the free Energy is both Size-Extensive and Scale- Invariant near a Critical Point. The Fundamental Renormalization Group Equation (RGE):

$$\mathcal{F}(x) = g(x) + \frac{1}{\lambda}\mathcal{F}(\phi_\lambda(x)) \tag{2}$$

5 Quantum HHL Algorithm

A quantum algorithm for solving linear systems of equations was put forward by Aram Harrow, Avinatan Hassidim and Seth Lloyd in 2009 [23]. More specifically, the algorithm can estimate the result of a *scalar measurement* on the solution vector **b** to a given linear system of equations $\mathbf{Ax} = \mathbf{b}$. In this context, **A** is a $N \times N$ Hermitian matrix with a spectral norm bounded by 1 and a finite condition number $\kappa = |\lambda_{\max}/\lambda_{\min}|$.

The HHL algorithm can be *efficiently* implemented only when the matrix **A** is sparse (at most poly(log N) entries per row) and well-conditioned (that is, its singular values lie between $1/\kappa$ and 1). We also emphasize the term "scalar measurement" here: the solution vector **x** produced by the HHL subroutine is actually (approximately) encoded in a quantum state $|\tilde{x}\rangle$ of $\lceil \log_2 N \rceil$ qubits and it cannot be directly readout; in one run of the algorithm we can at most determine some statistical properties of $|x\rangle$ by measuring it in some basis or rather *sampling* using some quantum mechanical operator M, i.e. $\langle \tilde{x}|M|\tilde{x}\rangle$. Even determining a specific entry of the solution vector would take approximately N iterations of the algorithm.

Furthermore, the HHL requires a quantum RAM (in theory): that is, a memory which can create the superposition state $|b\rangle$ (encoded **b**) all at once, from the entries $\{b_i\}$ of **b** without using parallel processing elements for each individual entry. Only if all these conditions are satisfied, the HHL runs in the claimed $\mathcal{O}(\log N s^2 \kappa^2/\epsilon)$ time, where s is the sparsity parameter of the matrix **A** (i.e. the maximum number of nonzero elements in a row) and ϵ is the desired error [24, 25].

Given all these restrictions, at first sight, the algorithm might not seem to be too useful; however, it is important to understand the context here. The HHL is primarily used as a *subroutine* in other algorithms and not meant as an independent algorithm for solving systems of linear equations in logarithmic time. In other words, the HHL is suitable for application in special circumstances where $|b\rangle$ can be prepared efficiently, the unitary evolution e^{-iAt} can be applied in a reasonable time frame and when only some observables of the solution vector **x** are desired rather than all its elements. The 2013 paper by Clader et al. [26], is a concrete demonstration of such a use case of the HHL with a very real-world application, i.e., calculation of electromagnetic scattering cross-sections of any arbitrary target faster than any classical algorithm (Fig. 1).

The HHL algorithm comprises of three steps: **phase estimation**, **controlled rotation** and **uncomputation** [27, 28].

For the first step of the algorithm, let $\mathbf{A} = \sum_j \lambda_j \left|u_j\right\rangle\left\langle u_j\right|$. Considering the case when the input state is one of the eigenvectors of **A**, $|b\rangle = \left|u_j\right\rangle$. Given a unitary operator **U** with eigenstates $\left|u_j\right\rangle$ and corresponding complex eigenvalues $e^{i\phi_j}$, the technique of quantum phase estimation allows for the following mapping to be implemented:

$$|0\rangle \left|u_j\right\rangle \rightarrow \left|\tilde{\phi}\right\rangle \left|u_j\right\rangle \tag{3}$$

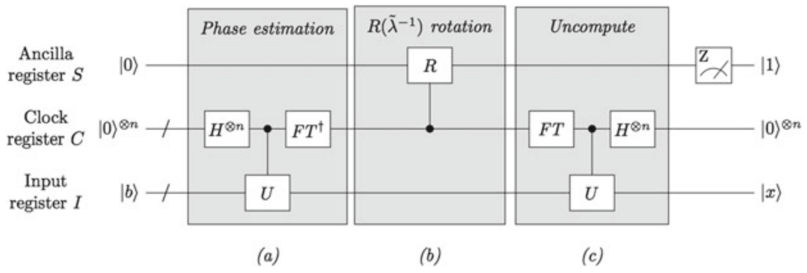

Fig. 1 HHL algorithm schematic: **a** *Phase estimation* **b** $R(\tilde{\lambda}^{-1})$ *rotation* **c** *Uncomputation*

Here, $\tilde{\phi}$ is the binary representation of ϕ to a certain precision. In the case of a Hermitian matrix A, with eigenstates $|u_j\rangle$ and corresponding eigenvalues λ_j, the matrix e^{iAt} is unitary, with eigenvalues $e^{i\lambda_j t}$ and eigenstates $|u_j\rangle$. Therefore, the technique of phase estimation can be applied to the matrix e^{iAt} in order to implement the mapping

$$|0\rangle |u_j\rangle \to |\tilde{\lambda_j}\rangle |u_j\rangle \tag{4}$$

where λ_j is the binary representation of λ_j.

In the second step of the algorithm, a controlled rotation conditioned on $|\lambda_j\rangle$ is implemented. For this purpose, a third ancilla register is added to the system in state $|0\rangle$, and performing the controlled Pauli-Y rotation produces a normalized state of the form

$$\sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}}\,|\tilde{\lambda_j}\rangle |u_j\rangle |0\rangle + \frac{C}{\tilde{\lambda}_j}|\tilde{\lambda_j}\rangle |u_j\rangle |1\rangle \tag{5}$$

where C is the normalization constant. This can be done by applying the operator

$$e^{-i\theta\sigma_y} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{6}$$

where $\theta = \tan^{-1}(C/\tilde{\lambda})$.

Since by definition $\mathbf{A} = \sum_j \lambda_j |u_j\rangle\langle u_j|$, therefore, $\mathbf{A}^{-1} = \sum_j (1/\lambda_j)|u_j\rangle\langle u_j|$. We assume that we are given a quantum state $|b\rangle = \sum_i b_i |i\rangle$. This state can be expressed in the eigen basis $|u_j\rangle$ of operator $\mathbf{A}$, such that $|b\rangle = \sum_j \beta_j |u_j\rangle$. Using the above procedure on this superposition state, we get the state

$$\sum_{j=1}^{N} \beta_j |\tilde{\lambda_j}\rangle |u_j\rangle \left(\sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}}\,|0\rangle + \frac{C}{\tilde{\lambda}_j}|1\rangle\right). \tag{7}$$

We uncompute the first register, giving us

$$|0\rangle \otimes \sum_{j=1}^{N} \beta_j \left|u_j\right\rangle \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} \, |0\rangle + \frac{C}{\tilde{\lambda}_j} \, |1\rangle \right). \tag{8}$$

It is to be noted that, $\mathbf{A}^{-1}\,|b\rangle = \sum_{j=1}^{N} \frac{\beta_j}{\tilde{\lambda}_j} \left|u_j\right\rangle$. Thus, a quantum state close to $|x\rangle = \mathbf{A}^{-1}\,|b\rangle$ can be constructed in the second register by measuring the third register and post selecting on the outcome'1', modulo the constant factor of normalization C. Amplitude amplification can be used at this step instead of simply measuring and postselecting to boost the success probability.

Notably, Tang's 2018, thesis titled *A quantum-inspired classical algorithm for recommendation systems* [29] essentially demonstrated that solutions based on the HHL for several linear algebra problems, which were earlier believed to have no equivalent to HHL in terms of time complexity, can be dequantized and solved with equally fast classical algorithms. Furthermore, the only caveat for Tang's algorithm is the allowance of *sample* and *query access*, and that is far more reasonable than efficient state preparation as demanded by the HHL. However, this doesn't imply the HHL has been rendered obsolete; we must be careful to note that Tang's algorithm is specifically aimed at low- dimensional matrices, whereas the original HHL was meant for sparse matrices, albeit quantum machine learning for low- dimensional problems are the most practical algorithms in the literature as of now. Nevertheless, generation of arbitrary quantum evolutions for state preparation remains as hard as ever [30–34].

6 Quantum Support Vector Machine

Data classification is one of the most important tools of machine learning today. It can used to identify, group and study new data. These machine learning classification tools have been used in computer vision problems [35], medical imaging [36, 37], drug discovery [38], handwriting recognition [39], geostatistics [40] and many other fields. Classification tools have machines to identify data, and therefore, know how to react to a particular data. In machine learning one of the most common methods of data classification using is using Support Vector Machines (SVM). The SVM is particularly useful because it allows us to classify data into one of two categories by giving in an input set of training data by drawing a hyperplane between the two categories. Quantum SVM machines have been recreated both theoretically [7] and experimentally [41]. These machines use qubits instead of classical bits to solve our problems. Many such quantum SVM [42–44] and quantum-inspired SVM [45, 46] algorithms have been developed.

In a support vector machine, in general, we shall have our training data of n points given as $\{\vec{x}_1, y_1\}$,$\{\vec{x}_2, y_2\}$...$\{\vec{x}_n, y_n\}$ according to the form $\{\{\vec{x}_i, y_i\} : \vec{x}_i \in \mathbb{R}^N, y_i = \pm 1\}_{i=1,2,\ldots,n}$ where $\vec{x}_i$ indicated the location of the point in the space $\mathbb{R}^N$ and y_i

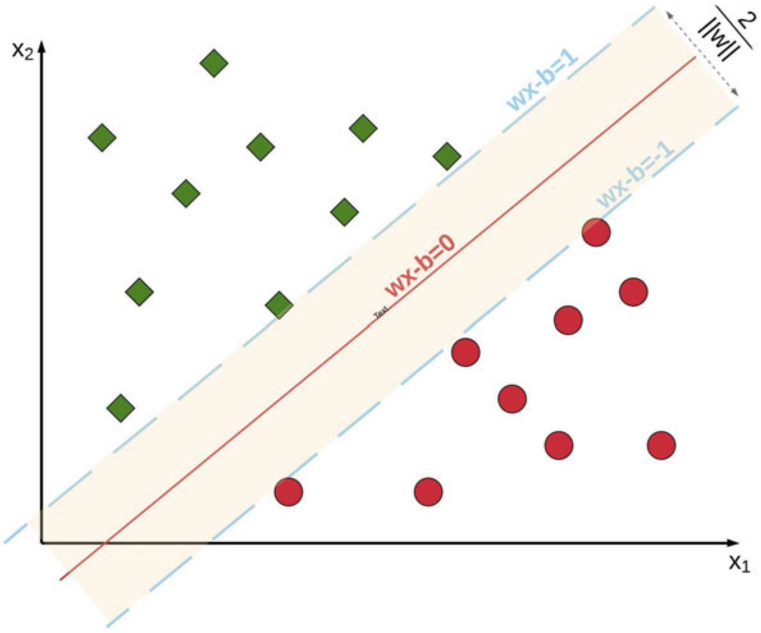

Fig. 2 Maximum margin hyperplane for a SVM

classifies the data as either +1 or –1 as to indicate the class to which it belongs. One of the simplest ways to divide such a data (if it is linearly separable) is by using any plane that satisfies the equation.

$$\vec{w}.\vec{x}_i - b = 0 \tag{9}$$

Here $\vec{w}$ is a vector normal to the hyperplane and $\frac{b}{|\vec{w}|}$ is the offset from the origin. In general, it's sometimes possible that many planes satisfy this equation so to draw a hard margin SVM where we try to construct two parallel hyperplanes with a maximum possible distance of $\frac{2}{|\vec{w}|}$ in between. The construction of these hyperplanes is done so that $\vec{w}.\vec{x}_i - b \geqslant 1$ for $y_i = 1$ and $\vec{w}.\vec{x}_i - b \leqslant 1$ for $y_i = -1$. We can write this as $y_i(\vec{w}.\vec{x}_i - b) \geqslant 1$. This hyperplane clearly discriminates between out two types of data points (Fig. 2).

While data often can be classified into two sets using the aforementioned method, often the data is nonlinear and method cannot be used. The common method to solve such problems is using the kernel trick where the problem is brought to a higher dimension where a hyperplane can be easily used to solve the issue. To do this, we need to use Lagrangian multipliers ($\alpha = (\alpha_1, \alpha_2, \ldots \alpha_n)$) and solve the dual formulation for optimization. Hence we can use the formula to get our solution

$$\max\left(\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i y_i \alpha_j y_j K_{ij}\right) \tag{10}$$

where K_{ij} is the kernel matrix and the dot product of the space given as $K_{ij} = \vec{x}_i.\vec{x}_j$ subject to the constraints $\sum_{i=1}^{n}\alpha_i y_i = 0$ and $\alpha_i y_i \geqslant 0$ hence the decision function of the hyperplane becomes

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{n}\alpha_i y_i K(x_i, x) + b\right) \tag{11}$$

where we can write $w = \sum_{i=1}^{n}\alpha_i y_i .x_i$. Hence even nonlinear problem can be solved using a SVM. But sometimes the number of dimensions needed to solve a problem inadvertently turns extremely high. This leads what can be called as the Curse of Dimensionality where we have increased complexity and over-fitting due to an increasing sparse matrix defining the location of data points. This is where the quantum SVMs become important. While problems in higher dimensions are extremely tedious to solve using classical computers, the exponential speedup observed in quantum computers by using the quantum SVM algorithms very effectively sorts our data.

But to solve it using a quantum computer, we need to bring our solution to a form where it will be easy for a quantum algorithm to solve it. One of the primary things to do at this point is to provide the algorithm with a certain scope of misclassification so that we do not have a problem with over-fitting, and have a variable ξ_i called a slack variable where $\xi_i \geqslant 0$ using which we can measure the misclassification. We can now write out the following optimization problem

$$\min\left(\frac{1}{2}||w|| + C\sum_{i=1}^{n}\xi_i\right) \tag{12}$$

where C is the cost parameter. Let also take $C = \gamma/2$ where γ is also a form of cost parameter. Once we have set these values we can write our equation as $\vec{w}.\vec{x}_i - b = 1 - \xi_i$. Looking for the saddle points of the above equation using our given constraints we get the equation.

$$F\begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & 1^T \\ 1 & K + \gamma^{-1}I \end{pmatrix}\begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ y_i \end{pmatrix} \tag{13}$$

Now to solve our classical algorithm in our quantum computer, we need to transform our algorithm into a quantum one. For this, firstly, we shall convert our training instances to quantum states in the form $|x_i\rangle$. Now we shall convert our matrix $F = J + K_\gamma$ where

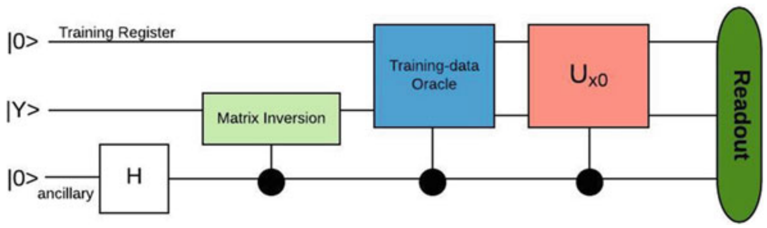

Fig. 3 Circuit of quantum SVM

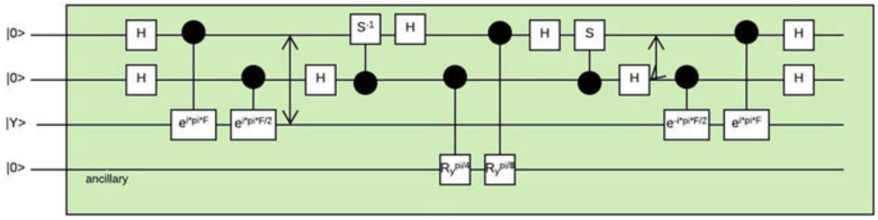

Fig. 4 Matrix inversion

$$J = \begin{pmatrix} 0 & 1^T \\ 1 & 0 \end{pmatrix} \qquad K_\gamma = \begin{pmatrix} 0 & 0 \\ 0 & K + \gamma^{-1} I \end{pmatrix} \tag{14}$$

Now we shall normalize F as $\hat{F} = \frac{F}{\text{tr}(F)} = \frac{F}{\text{tr}(K_\gamma)}$ and now using Baker–Campbell-Hausdorff formula we get our equation as

$$e^{-i\hat{F}\Delta t} = e^{\frac{-iJ\Delta t}{K_\gamma}} .e^{\frac{-i\gamma^{-1}I\Delta t}{K_\gamma}} .e^{\frac{-iK_\gamma \Delta t}{K_\gamma}} \tag{15}$$

This simplifies our equation to a form where we can find the eigenvalues and eigenbasis of our equation to find out desired values of b and α. Therefore, we can now find the hyperplane. One of the main advantages of using the quantum SVM is that the speed of execution is increased exponentially [47]. While this method can only be used for a dense training vector, other algorithms have been proved for sparse training vectors [44]. We can also create a circuit diagram [41] of this (Fig. 3).

In this circuit, we use the matrix inversion to get the parameters of the hyperplane. Then we enter the training data. After this is done we enter the data x0 to get what classification our data belongs to. These can be drawn as (Figs. 4 and 5).

Where F is the $(M + 1) \times (M + 1)$ matrix which contains the part of the Kernel K and th1 and th2 are the training data, and th0 is the data of the position of U_{x0}. Hence, we can see that quantum SVMs are one of the most effective methods of classifying data. These equations are faster than all other methods to perform data classification. They can also be implemented with ease in most systems. There are some limitations of these systems though. Firstly, these systems can often massively overfit data. That could lead to very data point being a support vector. This is something that is not

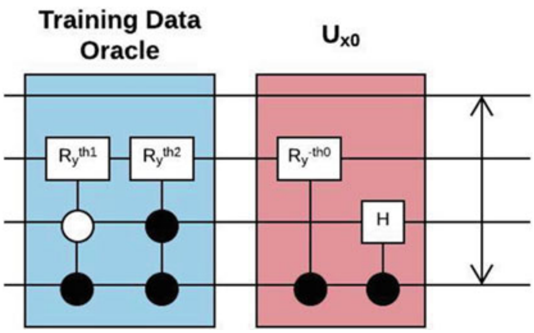

Fig. 5 Training oracle data and U_{x0}

desirable and could lead to issues in large data sets. It can also make the hyperplane very rigid and would leave very little scope for error. We would have to increase the scope for a soft SVM. Secondly, these systems work well with linear and polynomial kernels but can cause issues in other kernels. But since most of these systems are either polynomial or linear this is usually not an issue, Non Symmetrical kernels can also cause issues. These also form one of the important problems in the future. Solving a general kernel will especially be an important problem to solve. These algorithms will allow us to solve more complicated and specific problems. These would increase the scope of Quantum SVM into a more general application.

7 Quantum Classifier

A quantum classifier is a quantum computing algorithm which uses quantum states of the existing data to determine or categorize new data into their respective classes. In the following subsection we discuss about the background work on quantum classifiers, and how they have been implemented on a quantum computer.

7.1 Current Work on Quantum Classifiers

In a recent paper by Microsoft [48] presented a quantum framework for supervised learning based on variational approaches (Fig. 6).

Further actions are performed by the QPU(the quantum processing unit), consisting of a state preparation circuit S_x that encodes the input x into amplitudes, a model circuit U_θ, and single-qubit measurement. Such a QPU serves to perform inference with the model, or mathematically, determination of $f(x, \theta) = y$ by measuring the amplitudes prepared by the state preparation circuit and operated upon by the model circuit. Such measurements yield a 0 or a 1 from which binary prediction can be inferred. The classification circuit parameters θ are learnable and can be trained by

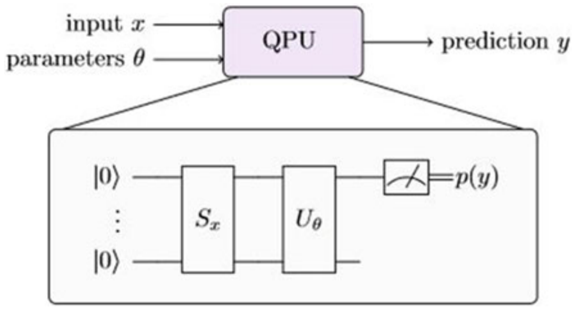

Fig. 6 Idea of the circuit-centric quantum classifier [48]

a variational scheme. Given a n-dimensional ket vector ψ_x representing an encoded feature vector, the model circuit maps this to another ket $\psi' = U_\theta \psi(x)$ by U_θ, where U_θ is necessarily unitary, parameterised by θ (Fig. 7).

The above circuit has code blocks B_1 and B_3 with control $r = 1$ and $r = 3$, respectively. There are 17 trainable single-qubit gates $G = G(\alpha, \beta, \gamma)$ and 16 trainable controlled single-qubit gates $C(G)$ which must be decomposed into elementary constant gate set for the underlying quantum hardware to use it. If optimization methods are employed such that all controlled gates are reduced to a single parameter, we get $3 \times 33 + 1 = 100$ learnable parameters in the model circuit, which, in turn, can classify between inputs belonging to

$$2^8 = 256$$

dimensions. Such flexibility such a model is much more compact than conventional feed-forward neural networks.

Farhi and Neven's [49] paper discussed about a quantum neural network (QNN), capable of representing labelled classical or labelled quantum data, and being trained by supervised learning techniques. For instance, a data set may consist of strings $z = z_1 z_2 \ldots z_n$ such that each z_i represents a bit taking the value $+1$ or -1 and a

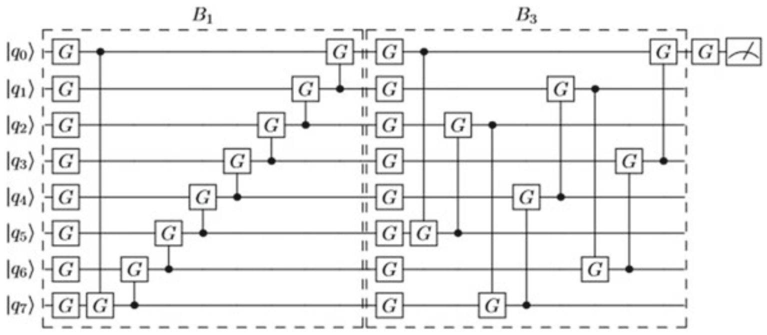

Fig. 7 Generic model circuit architecture for 8 qubits [48]

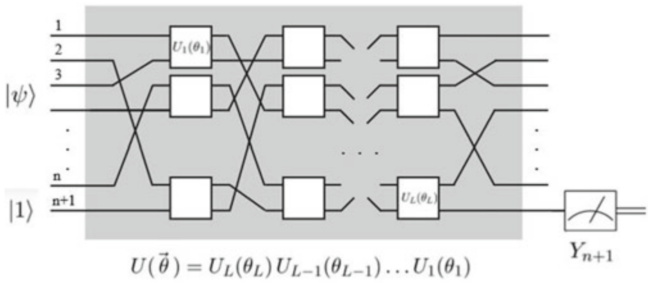
1
2
3
$|\psi\rangle$
n
n+1
$|1\rangle$
$U_1(\theta_1)$
$U_L(\theta_L)$
Y_{n+1}
$U(\vec{\theta}) = U_L(\theta_L)U_{L-1}(\theta_{L-1})\ldots U_1(\theta_1)$

Fig. 8 Schematic proposed of the quantum neural network on a quantum processor by Farhi and Neven [49]

binary label $l(z)$ chosen as $+1$ or -1. Also, there exists a quantum processor acting on $n+1$ qubits (ignoring the possibility of requiring ancilla qubits). The very last qubit shall be used as a readout. This processor applies unitary transforms on given input states; these are the unitaries that we have come from some toolbox of unitaries, maybe determined by experimental considerations [50] (Fig. 8).

Now, preparation of the input state $|\Psi, 1\rangle$ occurs followed by transformation via a sequence of qubit unitary transformations $U_i(\Theta_i)$ depending on parameters Θ_i. They are automatically adjusted during the operation, and the measurement of Y_{n+1} on the readout qubit produces the desired label for $|\Psi\rangle$.

A paper by Grant et al. [51], discusses the application of hierarchy-structured quantum circuits to applications involving binary classification of classical data encoded in a quantum state. The authors come up with more expressive circuits successfully applied to the problem of classification of highly entangled quantum states. Such circuits are tree-like, parameterized by a very simple gate set which currently available quantum computers can handle. First of these is called a tree tensor network (TTN) [52]. We further consider more complex circuit layout: multiscale entanglement renormalisation ansatz (MERA) [53]. MERAs are different from TTNs in the sense that they use additional unitaries to capture quantum correlations more effectively. Both one-dimensional (1D) and two-dimensional (2D) versions of TTN, and MERA circuits have been proposed in the literature [54, 55] (Fig. 9).

Earlier this year, Turkpençe et al.'s [56] paper on steady state quantum classifier demonstrates exploitation of additivity and divisibility properties of completely positive (CP) quantum dynamical maps. He also numerically demonstrates a quantum unit in its steady state, when subjected to different information environment behaves as a quantum data classifier. Dissipative environments influence the reduced system dynamics in a way that they affect the evolution of pure quantum states into mixed steady states [57]. Such mixed states are mixtures of classical probability distributions carrying no quantum signature. This model was used to demonstrate the usefulness of a small quantum system in classifying data contained in the quantum environments when the former is left in contact with these quantum environments.

Figure 10b depicts a few commonly used activation functions. For instance, a step function yields $f(y) = 1$ if $y = \Sigma_i x_i w_i \geqslant 0$ and yields $f(y) = -1$ else. After these

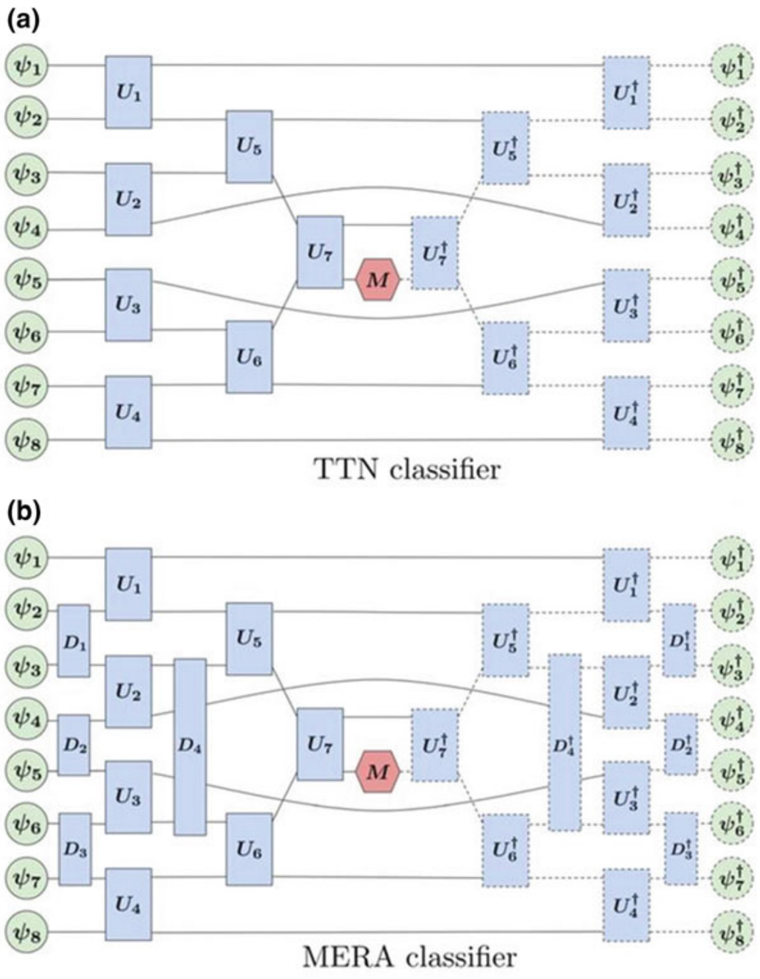

Fig. 9 TTN and MERA classifiers for eight qubits. **a** TNN Classifier, **b** MERA classifier [51]

results, if a line correctly separates the data instances, this corresponds to a properly functioning perceptron.

To benchmark these calculations, the authors make contact between the single spin and the data reservoir in the $\rho_\pi = |\downarrow\rangle\langle\downarrow|$ fixed quantum state. They then apply information as shown in Fig. 10d. The authors observe that the time evolution of spin magnetization converges to $(\sigma_z(t)) = -1$ as the spin density matrix approaches to the unit fidelity

$$F(t) = \mathrm{Tr}\sqrt{\sqrt{\rho\pi\sigma S(t)}\sqrt{\rho\pi}} = 1 \tag{16}$$

with the fixed reservoir state monotonically.

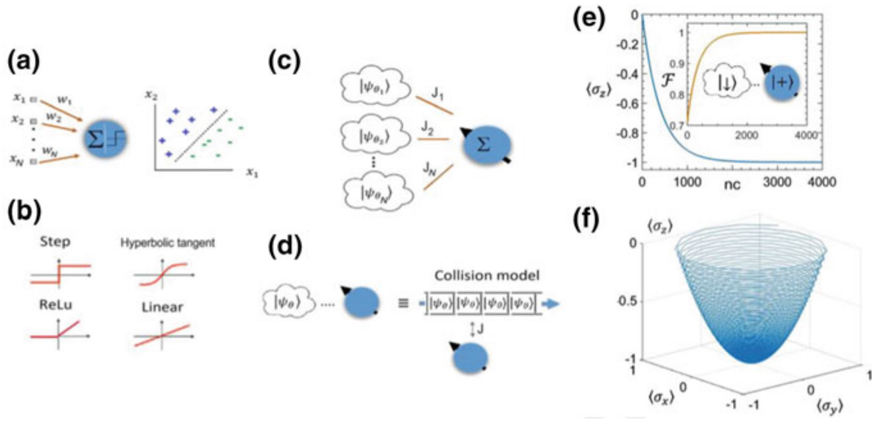

Fig. 10 A general view of the proposed model. **a** A classical perceptron with N inputs. **b** A few of the activation functions for the perceptron, **c** The scheme of the proposed quantum classifier. **d** Collision model to simulate quantum dynamic systems. **e** Time evolution of the single spin magnetization depending on the number of collisions. **f** The Bloch ball vector trajectory of the single spin during the evolution [56]

8 Application of Quantum Machine Learning to Physics

Machine learning methods have been effectively used in various quantum information processing problems including quantum metrology, Hamiltonian estimation, quantum signal processing, problems of quantum control and many others. The construction of advanced quantum devices including quantum computers use the techniques of quantum machine learning and artificial intelligence. Machine learning and Reinforcement learning techniques are used to create entangled states. Automated machines can control complex processes, for example, the execution of a sequence of simple gates, as used in quantum computation. While performing the quantum computation, decoherence or noise can be dealt with, by using advanced techniques of machine learning. Optimization algorithms are also used in the optimization of QKD-type cryptographic protocols in noisy channels [58]. Online Machine learning optimization can be used for determining the optimal evaporation ramps for Bose-Einstein condensates production [59]. The overlap between the theoretical foundations of machine learning and quantum theory is due to the underlying statistical nature of both. In the field of condensed matter physics, the identification of different phases and determining the order parameters can be done with the help of unsupervised learning. The problem of the Ising model configurations of thermal states can be solved using unsupervised learning techniques. Besides detecting the phases, the order parameters (for example magnetization in this case) can also be identified. Even without any prior knowledge about the system Hamiltonian, we can get information about the phases using these techniques.

9 Quantum Machine Learning and Quantum Artificial Intelligence

Quantum Artificial Intelligence is still a much more debatable concept. However, in the following few subsections, we try to understand some basic AI and machine learning terminologies, and finally see how they can be modified using quantum information processing and quantum computing. At the end of this section, we cite some recent developments in this field.

9.1 Basic Terminologies

Human intelligence allows us to accumulate knowledge, understand it and use it to make the best decisions. The field of AI aims to simulate such kind of process. The most important part of AI is machine learning (ML). ML tries to formalize algorithms which can learn and predict using some initial data. ML broadly can be classified into two fields viz. Supervised Intelligence and Unsupervised Intelligence. Supervised intelligence maps input to output using labels. Unsupervised learning, on the other hand, doesn't use labels and rather uses samples based on some specified rules. Another class of ML is Responsive Learning (RL). This is important with a quantum information perspective. In RL the environment is interactive rather than being static. The agent interacts with the environment and gets rewarded if its behaviour is correct. The agent learns through its cumulative experiences. An intelligent agent may be defined as an autonomous entity which can store data and act to achieve some goals.

9.2 Quantum Artificial Intelligence

The bigger question now is "Can quantum world offer something to the field of AI ?". We will now try to relate quantum information processing to AI using some kind of simulations. Quantum Computing (QC) can simulate large quantum data and can enable faster search and optimization. This in particular is very helpful for AI. For example, Govern algorithms and its several variants have a quadratic speed-up in search problems, as well as recent advances in Quantum Machine learning, have caused exponential gains in machine learning problems. We now try to understand Projective Simulation (PS). The agent is placed in some environment, such that the agent can act on the environment, and the environment responds as certain physical inputs. Hence, the agent learns from experience. The main part of the PS model is Episodic Computational Memory (ECM). ECM helps the agent to project itself, and thus induces a random walk through episodic memory space. PS model can easily be quantized. A quantum-enhanced autonomous agent is any agent interacting with classical environments but having a memory with quantum degrees of freedom.

The agent now takes a quantum walk through its memory space. The transitions generated are now quantum superpositions and can interfere. Also, quantum jumps are generated between different clip states. Since the PS model can be quantized, the model can potentially reach high speed-ups in exploring memory. Hence, the extension of PS model to quantum regime defines for the first time the meaning of embodied quantum agents.

9.3 Recent Developments

A team at the University of Pavia in Italy, conducted early research into artificial neural networks, experimented upon IBM's 5-qubit quantum hardware. Furthermore, there has been a collaboration between IBM and Raytheon BBM, in 2017, to improve the efficiency of certain black-box machine learning tasks. Currently, superconducting electronics has received attention as being a viable candidate for the creation of quantum hardware, with Google's Quantum AI Lab and UC Santa Barbara's partnership in 2014, being the latest venture. According to a recent research paper on "Quantum Computing for AI Alignment", as of now, we can't expect QC to be relevant to current AI Alignment research due to safety reasons until some protocols are made as efficient as possible.

We conclude this section by quoting Sankar Das Sarma and Dong-Ling Deng and Lu-Ming Duan, who wrote "It is hard to foresee what the quantum future will look like. Yet one thing is certain: The marriage of machine learning and quantum physics is a symbiotic relationship that could transform them both."

10 Entanglement in Quantum Machine Learning

Quantum Non-locality and Entanglement was recognized as a key feature of Quantum Physics. Entanglement can be described as correlations between distinct subsystems which cannot be created by local actions on each subsystem separately. In quantum Entanglement, two or more particles which are separated (space like separated) are correlated in such a way that local measurements in any one of the particles will affect the other particle(s) far away, i.e. 'The spooky action at a distance' stated by Albert Einstein. This basic nature of quantum particles is due to entanglement. Entanglement received a lot of attention since the advent of quantum mechanics (EPR paradox [60]), and still remains an area of active research. Let us consider a very simple example. Let's take two shocks (qubits) and each of the shocks can be right one or left one or superposition of both right and left with some probabilities. The right shock is represented by $|0\rangle$ and left one by $|1\rangle$. Now if we want to represent a group of two shocks we will take a tensor product of the two. The composite system of two shock is represented by $|\psi\rangle$ such as $|\psi\rangle = (a\,|0\rangle + b\,|1\rangle) \otimes (c\,|0\rangle + d\,|1\rangle)$ $|\psi\rangle = (ac\,|0\rangle\,|0\rangle + ad\,|0\rangle\,|1\rangle + bc\,|1\rangle\,|0\rangle + bd\,|1\rangle\,|1\rangle)$, where a,b,c,d are probabilities coefficient. If we want to form a pair of shocks from these set then the coefficients must be selected such as to cancel the two terms of the sum that is $|00\rangle$ *and* $|11\rangle$ conserving the other two. Now we cannot obtain the quantum state as a tensor product of the individual shocks. This is because between them to form a pair they have some correlation. That when these qubits are separated (space like separated), their correlations remains, even without the existence of any interaction. It is said the shocks (qubits) are entangled; it is not possible to represent the composite system as a tensor product of individual qubit states. These qubits are interconnected, such that measurement on one qubit affects the other. This feature is extensively used in Machine learning as it reduces no of qubits required to perform the same task in classical machine learning. However, there are some demerits of using Quantum Machine Learning as well which is discussed in the work of Cristian [61] and later in our conclusion.

In 2015, Cai [62] and his group did a work in which they demonstrate a very difficult task for conventional machine learning—manipulation of high dimensional vectors and the estimation of the distance and inner product between vectors—can be done with quantum computers. It is, for the first time, reported using a small-scale photonic quantum computer to experimentally implement classification of 2, 4 and 8 dimensional vectors into different clusters. These clusters are then used to implement supervised and unsupervised machine learning. Theoretically, this can be scaled to a larger number of qubits, thereby paving a way for accelerating machine learning.

In 2018, another work is done by Liu [63] and his group. They implemented simple numerical experiments, related to pattern/image classification, in which they represent the classifiers by many-qubit quantum states written in the matrix product states (MPS). Classical machine learning algorithm is applied to these quantum states to learn the classical data. They explicitly show how quantum entanglement (i.e. single-site and bipartite entanglement) can emerge in such represented images. Entanglement characterizes here the importance of data, and such information is practically used to guide the architecture of MPS, and improve the efficiency. The number of needed qubits can be reduced to less than 1/10 of the original number, which is within the access of the state-of-the-art quantum computers.

In recent work by Yoav Levine [64] and his group, it is established that deep convolutional neural networks and recurrent neural networks can represent highly entangled quantum systems. They constructed Tensor Network equivalents of such architectures and identified that the information in the network operation can be reused; this trait distinguishes these from other standard Tensor Network-based representations, thereby increasing their entanglement capacity. These results show that volume-law entanglement can be supported by such architectures, and these are polynomially more efficient than presently employed RBMs. Analysis of deep learning architectures lead to quantification of the entanglement capacity, as well as formally motivating a shift of trending neural network-based wave function representations, closer to the state-of-the-art in machine learning.

Neural Network is one of the most significant sides of machine learning and artificial intelligence. To make machines learn from the data patterns, analyze the data on its own, scientists made algorithms to simulate our natural neural network. Warren

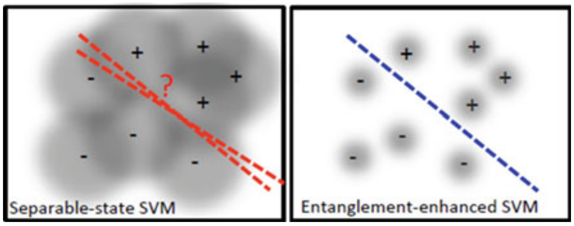

Fig. 11 Performance contrast between entangled-state SVM and separable-state SVM [69]

McCulloch of the University of Illinois and Walter Pitts of the University of Chicago, developed the theoretical basis of neural networks in 1943. In 1954, Belmont Farley and Wesley Clark of MIT, developed the first neural network for pattern-recognition skills [65]. In this context, we see that as the demand for machine learning is increasing day by day, understanding the physical aspects of neural network is to be increased certainly, and this is one of the sides where the study of entanglement properties has to be done. Focusing on the RBM [66] (Restricted Boltzmann Machine), Dong-Ling Deng, Xiaopeng Li and S. Das Sarma, in 2017, studied [67] the entanglement properties, and they found that for short RBM states entanglement entropy follows the area law which is also inspired by the holographic principle [68] that states all the informations reside on the surface of the black hole, hence the entropy depends on its surface not on volume. For any dimension and arbitrary bipartition geometric R-range RBM states, entanglement entropy becomes

$$S \leq 2a(A)R \log 2 \tag{17}$$

where a(A) is the surface area of subsystem A.

In the limit, N $\rightarrow \infty$ (N qubits), the entropy starts to vary linearly with the size of the system—entanglement volume law.

Supervised Learning can be enhanced by Entangled Sensor Network as shown by Zhuang and Zhang, early this year [69]. So far existing quantum supervised learning schemes depend on quantum random access memories (qRAM) to store quantum-encoded data given a priori in a classical description. However, the data acquisition process has not been used while this process makes the maximum usage of input data for different supervised machine learning tasks, as constrained by the quantum Cramér-Rao bound. They introduced the Supervised Learning Enhanced by an Entangled sensor Network (SLEEN). The entanglement states become handy in quantum data classification and data compression. They used SLEEN to construct entanglement-enhanced SVM and entangled-enhanced PCA and for both the cases, they got genuine advantages of entangled states—data classification and data compression, respectively (Fig. 11).

In the case of SVMs, while separable-state SVM becomes inaccurate due to measurement uncertainty making the data classification less contrasting when entangled-state SVM is not affected by the uncertainty keeping the output as expected (Fig. 12).

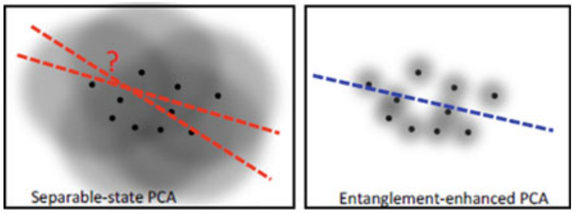

Fig. 12 Performance contrast between entangled-state PCA and separable-state PCA [69]

In the case of PCAs, the same uncertainties factor prevent the entangled-state PCA from making a perfect principal axis, while entangled-state PCA precisely finds the principal axis.

Hence, this entanglement stuff makes the Supervised Machine Learning ultrasensitive to various fields in biological, thermal systems.

On the other hand, machine learning is also used to determine the entanglement of systems—how much entangled they are. Jaffali and Oeding showed, mainly focusing on pure states in their paper how Artificial Neural Network can be trained to predict the entanglement type of a quantum state and distinguish them. This may help in processing quantum information, increasing the efficiency of the quantum algorithms, cryptographic schemes, etc.

From the above works, we can see that by using the Quantum Entanglement we cannot only outperform the results of classical computers but it also requires less resources. The most motivating work is merging many-body quantum systems to machine learning using Tensor Network. Such an interdisciplinary field was recently strongly motivated, due to the exciting achievements in the so-called "quantum technologies". Thanks to the successes in quantum simulations/computations, including the D Wave and the quantum computers by Google and others ("Quantum Supremacy"), it becomes unprecedentedly urgent and important to explore the utilization of quantum computations to solve machine learning tasks. The low demands on the bond dimensions, and particularly, on the size, permit to simulate machine learning tasks by quantum simulations or quantum computations in the near future.

11 Quantum Neural Network

The following few subsections elaborate the merger of classical neural networks and quantum computing, producing a more powerful version of the former. In subsection *A*, we provide a brief introduction to classical neural networks. In subsection *B*, we discuss the previous works on quantum neural networks. Subsequent sections onwards, we describe the quantum neuron and its implementation to the quantum computer, and present the latest development on quantum convolutional neural networks.

11.1 Classical Neural Networks

One of the most basic neural networks in classical deep learning is the deep feed-forward networks, mathematically defined by a function $y = f(x; \theta)$, where x is the input n-dimensional vector, y is the output m-dimensional vector ($m < n$ usually), and θ represents the parameters that guide the network to map x to y [70]. Neural networks are usually organized in layers (especially the *hidden* layers between the input layer from which propagation occurs to different hidden layers and the output layer to which propagation occurs from some hidden layer) to divide computation into steps. At each step (or hidden layer), some degree of nonlinearity is added, allowing the network to learn complicated functions.

Training a network is choosing the combination of parameters θ that can map the input vectors x as closely as possible to the actual output vectors y [70]. Training involves the initial random choice of parameters, followed by gradual updates as the same vectors x are passed on to the network, and predictions by the network are compared to the actual, real outputs y externally provided to the network. This training happens through the classical procedures—gradient descent and backpropagation [70].

Several different types of architectures have been developed, for instance, Convolutional Neural Networks (CNNs), Long-Short Term Memory networks (LSTMs), Recurrent Neural Networks (RNNs), Generative Adversarial Neural networks (GANs), variational autoencoders and many more [70]. Together, they are able to drive the AI revolution, finding increasing applications to image and sound processing, self-driving cars, online recommender systems, reinforcement learning-based game playing bots, stock market price predictors, virtual assistants and several other applications in all walks of life. For further technical details on these, we refer to *Deep Learning* by Ian Goodfellow, Yoshua Bengio and Aaron Courville [70].

11.2 Background Work in Quantum Neural Networks

In a paper by Kak in 1995 [71], attempts were made to model a neural network in quantum computing, and discussions were presented on the versatility of the quantum neural computer with respect to the classical computers. In the same year, Menneer and Narayanan's [72] work introduced a method based on a multi-universe interpretation of the quantum theory that made neural network training powerful than ever before—superposition of several single layered neural networks to form a bigger quantum neural network. Perus [73] used the quantum version of classical gradient descent, coupled with CNOT gates, to demonstrate the use of parallelism in quantum neural architectures. Menneer [74] did a comprehensive study of the contemporary NN architectures in his Ph.D. thesis. Faber et al. [75] addressed the question of implementation of an artificial neural network architecture on a quantum hardware. Schuld [76] gave guidelines for quantum neural network models: (1) ability of the network to produce the binary output of length separated from the length of the binary input by some distance measure, (2) reflect some neural computing mechanisms and (3) utilize the quantum phenomenon and be fully consistent with the quantum theory.

In the recent past, several advancements have been made to bridge the gap between classical and quantum deep learning. In 2014, Wiebe et al. [77] demonstrated the power of quantum computing over classical computing in deep learning and objective function optimization. Adachi et al. Recent research hasb demonstrated the superiority of quantum annealing (using D-Wave quantum annealing machine) to contrastive divergence based methods, and tested the same on preprocessed MNIST data set.

Other notable works include quantum perception model [78], quantum neural networks based on Deutsch's model of quantum computational network [79], quantum classification algorithm based on competitive learning neural network and entanglement measure [80], a novel autonomous perceptron model for pattern classification applications [81] and a quantum version of the generative adversarial networks [82] to name a few.

11.3 Quantum Neuron

The current issue in quantum neural networks is the problem of introducing nonlinearity, as is the case in classical neural networks. Nonlinearity is central to learning complex functions, and thus efforts have been made to resolve this: use of quantum measurements Kak et al. [83] and Zak et al. [84], using dissipative quantum gates [83], and the idea that a quantum neural network based on the time evolution of the system is intrinsically nonlinear.

A quantum neuron is strongly correlated to the actual neuron of the human system. The latter, based on the electrochemical signals received, either fires or not. Similar is the model of the classical neuron in deep learning. An input vector x (corresponding to the stimulus in humans) is combined with a set of weights θ (corresponding to the neurotransmitters), and the result of this combination determines whether the neuron fires or not. Mathematically, a n-dimensional input vector $X = x_1, x_2, ..., x_n$ is combined with weights $\theta = \theta_1, \theta_2, ..., \theta_n$ to yield the combination: $x_1\theta_1 + x_2\theta_2 + ... + x_n\theta_n + b$ where b is the *bias* added to the computation to incorporate functions not passing through the origin of the $n-$dimensional space considered here [85]. To introduce nonlinearity in the same, several activation functions are used, which have been shown to benefit neural network training [86]. Recent advances have explored learning activation functions for separate neurons using gradient descent [87], approximation of neural networks using ReLU as the activation function [88], and other conventional functions like the sigmoid function and the step function.

The quantum equivalent of the classical neuron: the quantum neuron, is used to build the quantum neural networks, which benefit from the intrinsic property of quantum mechanics of storing co matrices and performing linear algebraic operations on those matrices conveniently Neukart et al. [89], Schuld et al. [90], Alvarez et al. [91], Wan et al. [92], Rebentrost et al. [93], Otterbach et al. [94], Lamata et al. [95].

To implement a quantum neuron, a set of n qubits is prepared and operated upon by some unitary transformation, and the result is prepared in a separate ancilla qubit that is then measured—the measurement being the decision whether the quantum neuron fires or not. Specific details follow as under.

To encode a m dimensional classical input vector x, n qubits are used such that $m = 2^n$, $n < m$, thereby exploiting the advantage of quantum information storage allowing the exponential reduction in the number of input nodes required. The following transformation is done on the input qubits: $U|0\rangle^{\otimes n} = |\psi\rangle$.

Assuming the computational basis of the already defined n dimensional Hilbert space is $|1\rangle$, $|2\rangle$, $|3\rangle$, ..., $|n\rangle$, the input vector x and the weight vector θ can be defined in quantum terms as

$$|\psi\rangle = \frac{1}{n^{1/2}} \sum_{j=1}^{n} x_j |j\rangle \tag{18}$$

where x_j is represents the usual jth component of the classical input vector x. Likewise, the weight vector θ can be encoded in the quantum realm as

$$|\phi\rangle = \frac{1}{n^{1/2}} \sum_{j=1}^{n} \theta_j |j\rangle \tag{19}$$

where θ_j represents the usual jth component of the classical weight vector θ.

Tacchino et al. [96] defines a unitary operation that performs the inner product of the two terms defined above, and updates an ancilla qubit based on a multi-controlled NOT gate. The authors introduce a nonlinearity by performing a quantum measurement on the ancilla qubit.

11.4 Quantum Convolutional Neural Network

Convolutional Neural Network is a type of deep neural network architecture motivated by the visual cortex of animals [97]. CNNs provide great power over a variety of tasks: object tracking [98], text detection and recognition [99], pose estimation [100], action recognition [101], scene labelling [102], saliency detection using multi-context deep learning [103]. Further review of deep convolutional deep learning is referred [104]. The power of CNNs arises from the several convolutional layers, pooling layers, followed by few densely, fully connected layers that help to reduce the huge size of various matrices of images to few hundred nodes which can then be used for the final output layer of a few nodes (for instance equal to the number of classes in a multi-classification problem). The weights are optimized by training on huge data sets fed into the network through multiple passes. CNNs also involve parameters that directly affect the parameters/weights, called the hyperparameters. Hyperparameters are fixed for specific networks based on experiments and comparisons across several models.

On the quantum side, neural networks have been used to study properties of quantum many-body systems, as in Carleo et al. [105], Nieuwenburg et al. [106], Maskara et al. [107], Zhang et al. [108], Carrasquilla et al. [109], Wang et al. [110], and Levine et. al. [111]. The use of quantum computers to enhance conventional machine learning tasks has gained traction in the modern world Biamonte et al. [112], Dunjko et al. [113], Farhi et al. [114], and Huggins et al. [115].

A QCNN circuit model has been proposed by Cong et al. [116]. The proposed model upper bounds the n input parameters by $O(\log(n))$. Like conventional CNN, the authors continued the training on the quantum version of the *mean squared error*:

$$\text{MSE} = \frac{1}{2M} \sum_{i=0}^{m} (y_i - h(\psi))^2 \tag{20}$$

where y_i is the actual output of the input state ψ, and $h(\psi)$ is the computation done by the quantum network. The *mean squared error* tends to reduce the distance between the predicted value from the network. The authors discuss the efficient implementation of experimental platforms: regarding efficiently preparing quantum many-body input states, two-qubit gates application and projective measurements. With the success of the quantum convolutional neural networks, it is hoped that other conventional deep learning networks will be soon converted, thus increasing the range of quantum neural networks.

To solve highly complex problems like quantum phase recognition (QPR), which probes if an arbitrary input quantum state ρ_{in} belongs to the particular quantum phase of matter, quantum error correction (QEC) optimization which probes for an optimal quantum error correcting code for a given, a priori unknown error model such as dephasing or potentially correlated depolarization, quantum convolution neural networks has stood out to be the best possible solution.

These problems are intrinsically quantum in nature, and therefore, unsolvable by classical machine learning techniques that are in use today. The classical machine learning algorithms with large, deep neural networks can only solve classical problems such as image recognition; the quantum systems, however, involve exponentially large Hilbert spaces and are difficult to implement in a classical framework. Quantum algorithms are able to solve these problems, due to parallelism offered in the design of such algorithms; however, the limited coherence times of short-term quantum devices prevent the realization of large scale deep quantum neural networks.

12 Use of Artificial Neural Networks to Solve Many-Body Quantum Systems

Studying the many-body quantum systems is one of the most challenging areas of research in physics. It is mainly due to the exponential complexity of the many-body wave function and the difficulty that arises in describing the non-trivial correlations encoded in its wavefunction [105]. However, recently the use of neural networks for the variational representation of the quantum many-body states has generated a huge interest in this field [117–119]. This representation was first introduced by Carleo and Troyer in 2016, in which they had used the Restricted Boltzmann Machine (RBM) architecture with a variable number of hidden neurons. Using this procedure, they could find the ground state of the transverse-field Ising (TFI) and the antiferromagnetic Heisenberg (AFH) models with high accuracy [105]. Moerover, they could also describe the unitary time evolution of complex interacting quantum systems. Since then neural networks have been extensively used to study various physical systems. The representational power and the entanglement properties of the RBM states have been investigated, and the RBM representation of different systems such as the Ising Model, Toric code, graph states and stabilizer codes have been constructed [117]. Also, the representational power of the other neural network architectures such as the Deep Boltzmann Machine (DBM) [120] are under active investigation.

12.1 Variational Representation of Many-Body Systems in RBM Networks

A neural network is an architecture that uses its parameters to represent the quantum state of a physical system [105]. The Restricted Boltzmann Machine architecture consists of a visible layer of N neurons and a hidden layer of M neurons. The neurons of the visible layer and hidden layers are connected but there are no intra-layer connections. As the spin of the neurons in the RBM network can have the values ± 1, the spins of the neurons of the visible layer can be mapped to the spins of the physical system they represent. Moreover a set of weights is assigned to the visible (a_i for the i_{th} visible neuron), hidden (b_i for the i_{th} hidden neuron) and to the couplers connecting them (W_{ij} for the coupler connecting the i_{th} visible neuron with the j_{th} hidden neuron) [121]. Then, wave function ansatz for the N-dimensional quantum state of spin variable configuration $\mathcal{S} = \{s_i\}_{i=1}^{N}$ would be given by [105, 121].

$$\psi_M(\mathcal{S}, \mathcal{W}) = \sum_{\{h_i\}} e^{\sum_i a_i s_i + \sum_j b_j h_j + \sum_{ij} W_{ij} s_i h_j} \tag{21}$$

where s_i and h_i denote the spins of the visible and hidden neurons, respectively, and the whole state is given by the superposition of all the spin configuration states with $\psi(\mathcal{S})$ as the amplitude of the $|\mathcal{S}\rangle$ state [121, 122].

$$|\psi\rangle = \sum_{\mathcal{S}} \psi(\mathcal{S}) \, |\mathcal{S}\rangle \tag{22}$$

13 Classical Simulation of Quantum Computation Using Neural Networks

Since the neural networks are able to represent various quantum states efficiently, a natural question to be posed is whether they can also simulate various quantum algorithms. Interestingly, the networks are also able to simulate the action of various quantum gates. This has been investigated in the DBM and the RBM architectures [120, 123]. As mentioned before, the representation of a quantum state by a neural network depends on its network parameters. Thus, the action of various gates can be simulated by appropriately changing the network parameters in a way that the new quantum state represented by the network with the new parameters is the same as would have been obtained by applying the quantum gate to the initial quantum state. Also in a recent work, the methods to prepare specific initial states in RBM analogous to those used as initial states while implementing a quantum algorithm in a quantum circuit model has been discussed [121]. The prepared states were shown to efficiently simulate the action of the Pauli X gate. These results have opened up a great possibility of solving various quantum mechanical problems using neural networks. Future investigations in this direction may include the implementations of quantum algorithms in various neural network architectures and the exploitation of the machine learning techniques to achieve higher accuracy in solving the quantum mechanical problems [121].

14 Implementation of Quantum Machine Learning Algorithms on Quantum Computers

In this section, we discuss the implementation of some quantum machine learning algorithms with the help of quantum logic and quantum gates.

Earlier this year H. Liu's [124] paper first proposed a quantum algorithm to obtain the classical gradients. can be regarded as the inner product of two vectors $(p(x_1; w) - y_1, ..., p(x_N; w) - y_N)$ and $(x_1^j, ..., x_N^j)$. To achieve this, their quantum algorithm consists of two steps: generate an intermediate quantum state $\frac{1}{\sqrt{N}}\Sigma_{i=1}^N|i\rangle|p(x_i; w)\rangle$ based mainly on amplitude estimation; (2)perform swap test to obtain ∇w_j in the classical form. Then the parameters w is updated according to the iterative rules via simple calculations. The entire algorithm process is shown in Fig. 13.

(1) Generate an intermediate quantum state

(1.1) Preparing three quantum registers in the state $|0 \log N\rangle$ $|0\rangle$ $|0^{\log_M}\rangle$ and perform the Hadamard gates $H^{\otimes \log N}$ on the first register, then the system becomes

$$\frac{1}{\sqrt{N}}\Sigma_{i=1}^N|i\rangle|0\rangle|0^{\log M}\rangle \tag{23}$$

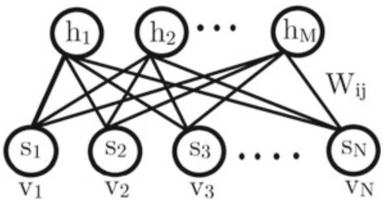

Fig. 13 The structure of a restricted Boltzmann machine [121]. The spin configuration of the N visible neurons is represented by $\{s_i\}_{i=1}^{N}$ (s_i is the spin value of the $i_t h$ neuron). Also, there are M hidden neurons in one more layer (called the hidden layer). The coupler connecting the i_{th} visble neuron with the j_{th} hidden neuron has the weight W_{ij}. However, there are no intra-layer connections. The wavefunction ansatz for the system represented by this network is given by Eq. 21

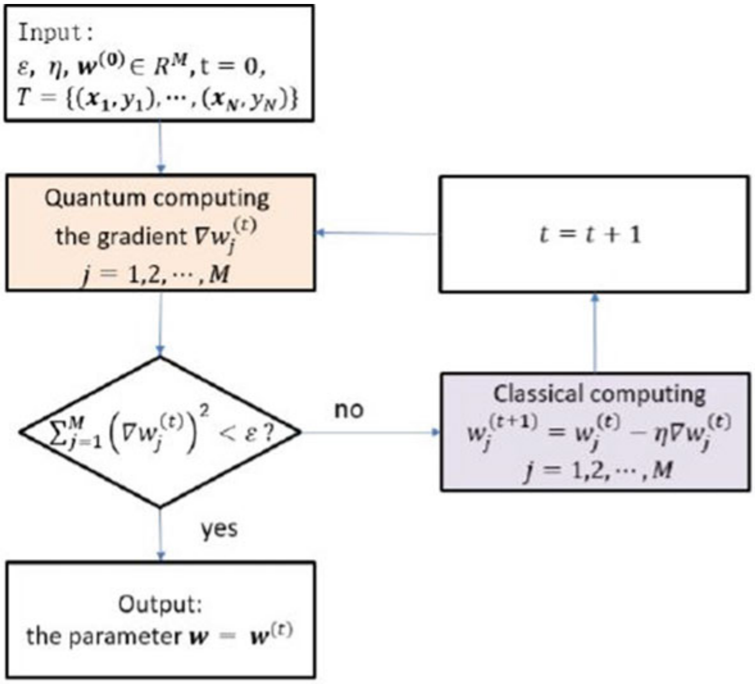
Input: $\varepsilon, \eta, \mathbf{w}^{(0)} \in R^M, t = 0$, $T = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$
Quantum computing the gradient $\nabla w_j^{(t)}$ $j = 1,2,\cdots,M$
$t = t + 1$
$\sum_{j=1}^{M}\left(\nabla w_j^{(t)}\right)^2 < \varepsilon$?
no
Classical computing $w_j^{(t+1)} = w_j^{(t)} - \eta\nabla w_j^{(t)}$ $j = 1,2,\cdots,M$
yes
Output: the parameter $\mathbf{w} = \mathbf{w}^{(t)}$

Fig. 14 The whole process of the entire algorithm for quantum logistic regression [124]

where i is represented in binary as $i_1, i_2, \ldots, i_{\log N}$.
(1.2) Perform H on the second register (Fig. 14)

$$\frac{1}{\sqrt{N}}|i\rangle\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0^{\log M}\rangle \tag{24}$$

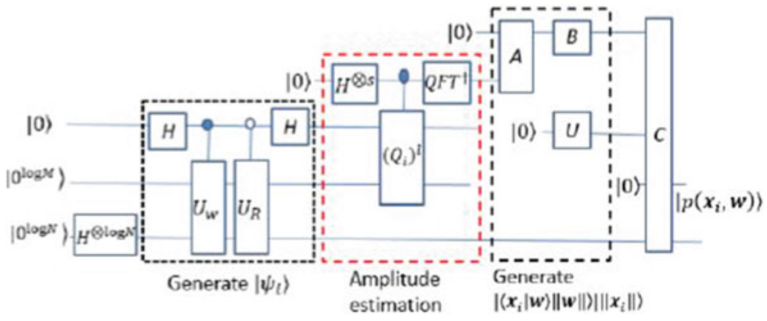

Fig. 15 Quantum circuit diagram to generate an immediate quantum state [124]

In the above circuit, where ϵ is the precision, η is the step factor, $w^{(0)}$ is the initial value, $t = 0, 1, 2, \ldots$ is the iteration number, T is the data set. The brown rectangle represents the quantum algorithm, and the light purple rectangle represents the classical iterative update algorithm (Fig. 15).

In the above circuit, A denotes the $2 \sin 2(\pi) - 1$ gate to estimate $\langle x_i | w \rangle$. s is the number qubits for estimating $x_i | w \rangle$., B, U are the unitary gates to access $||w||$,$||x_i||$, respectively. C represents the $\frac{1}{1+\exp(-kx)}$ gate to obtain $|p(x_i; w)\rangle$.

In addition to the above quantum machine learning algorithm implementation, there have been other algorithms that have been implemented on quantum computers as well, for example, S. Lloyd [125] and the team have work Quantum Hopfield neural network which uses qHop, encoding data in order $\log_2(d)$ qubits.

15 Conclusion and Future Works

In this review, we tried to compile the effects that quantum computers are having and will have on machine learning. While only a few years ago, most of the research works in these fields were largely theoretical, we now have demonstrable quantum machine learning algorithms. And as expected these algorithms are significantly faster and more effective than their classical counterparts. This amalgamation of machine learning and quantum computers allows us to run classical algorithms significantly faster in many cases. The effect that quantum computers can have on machine learning is extremely vast. As quantum computers with larger number of qubits are realized, we will be able to test more quantum algorithms and then truly access the effect that quantum computers will end up having on machine learning. Many of realized on an actual quantum computer due to the large number of qubits they require. But as research in this field progresses, we shall have better quantum computers and better algorithms to solve our machine learning problems. It is always possible that a more effective algorithm to solve machine learning problems is yet to be discovered. This is still one of the biggest problems while working with quantum computers since quantum algorithms are often unintuitive and it may take a lot of time to discover a better algorithm. Using quantum computers, we are able to implement classical machine learning classifiers for better, faster and accurate classification. While only time can tell the true effect that quantum computers will have on machine learning the possibilities seem endless and with every new algorithm machine learning seems to be something that can be definitely improved upon by quantum computers. In our society where huge amounts of data is collected and needs to be processed every minute, and where new and novel research methods can have huge impacts on both life and economy quantum machine learning definitely seems to be a methodology that will lead to a better future.

Here, we discussed a number of different methods of quantum machine learning algorithms. The research in this field is still mostly on a theoretical level, with few dedicated experiments on the demonstration of the usefulness of quantum mechanics for machine learning and artificial intelligence.

Quantum computation exhibits promising applications in machine learning and data analysis with much more advance in time and space complexity.

Experimental verification of quantum algorithms requires dedicated quantum hardware, and that is not presently available. It is hoped that the research community shall soon have access to scale-scale quantum computers (500–1000 qubits) via quantum cloud computing ('Qloud'). However, the world shall still continue to see advancements, in size and complexity, of special-purpose quantum information processors: quantum annealers, nitrogen vacancy centres (NV)-diamond arrays, made to order superconducting circuits, integrated photonic chips and qRAM. Moreover, silicon-based integrated photonics have been used to construct programmable quantum optic arrays having around 100 tunable interferometers; however, as the circuits are scaled up, we experience the loss of quantum effects.

There is yet not a general theory that can be used to analyze and engineer new quantum machine learning algorithms. Further, there are still some interesting questions yet to be unanswered, and problems yet to be solved. One of the main problems is that the proposed implementations are limited in the quantity of input data they can handle. Since the dimension of Hilbert space increases as the size does, if we permit to store a huge quantity of data, we must ensure accurate and efficient initialization of the quantum system. This is problematic because ensuring this is difficult and we need a large amount of data for reliable training. In addition to this, we have the problem of retaining memory. In machine learning, we take for granted the memory of the network, which is able to remember the weights of the network. But obtaining this memory in quantum dynamics is very difficult due to the unitary evolution of the system. Another problem pertains to quantum annealers: to improve connectivity and implement more general tunable couplings between qubits. Another problem is that irrespective of the ability of quantum algorithms to provide speed-ups in data processing, there is no advantage to be taken in reading data; on the other hand, the latter sometimes overshadows the speed-up offered by certain algorithms. There is yet another problem relating to obtaining a full solution from quantum algorithms. This is because obtaining a string of bits requires observing an exponential number of bits, rendering few applications of quantum machine learning infeasible. A potential solution for this problem is to only learn the summary statistics for the solution state. The main challenge is the costing. The theoretical bounds on the complexity suggest that for sufficiently large problems, huge advantages can be extracted; it is unclear, however, when that crossover point occurs. Another problem is the benchmarking problem. To properly assert the supremacy of a quantum algorithm, it must be shown better than known classical machine learning algorithms. This requires extensive benchmarking against modern heuristic methods.

We can avoid a few of the above said problems by applying quantum machine learning for a system which follows the principle of quantum mechanics rather than applying on classical data. An aim can be to use quantum machine learning to control quantum computers, enabling an innovative cycle like that in classical computation, where each generation of processors is used to design processors of the next generation. For quantum algorithms for linear algebra, their practical performance is hindered due to issues for data access and restrictions on problem classes that can be solved. The true potential of such techniques can only be evaluated by near future advances in quantum hardware development. Given the great majority of QML literature developed within the quantum community, further advances can come after significant interactions between the two communities. This is the reason behind us structuring this review to be familiar to both quantum scientists and ML researchers. To achieve this goal, we put great emphasis on the computational aspects of ML.

Acknowledgments A.P.D. acknowledges the support of KVPY fellowship. S.R. and S.C. acknowledge the support of DST Inspire fellowship. B.K.B acknowledges the support of IISER-K Institute fellowship.

References

1. P.W. Shor, Algorithms for quantum computation: discrete logarithms and factoring, *Proceedings 35th Annual Symposium on Foundations of Computer Science* (IEEE Comput. Soc. Press, 1994)
2. J. Bermejo-Vega, K.C. Zatloukal, *Abelian Hypergroups and Quantum Computation* (2015). arXiv:1509.05806
3. R.D. Somma, Quantum simulations of one dimensional quantum systems. Quantum. Inf. Comput. **16**, 1125 (2016)
4. S.J. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach* (Pearson Education Limited, Malaysia, 2016)
5. O. Bousquet, U.V. Luxburg, G. Ratsch, (eds.), *Advanced Lectures on Machine Learning: ML Summer Schools 2003* (Canberra, Australia, 2003; Tubingen, Germany, 2003). Revised Lectures (Springer, 2011), p. 3176
6. L.P. Kaelbling, M.L. Littman, A.W. Moore, Reinforcement learning: a survey. J. Artif. Intell. Res. **4**, 237–285 (1996)
7. P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification. Phys. Rev. Lett. **113**, 130503 (2014)
8. H. Abdi, L.J. Williams, Principal component analysis. Wil. Inter. Rev.: Comput. Stat. **2**, 433 (2010)
9. R. Orus, S. Mugel, E. Lizaso, Quantum computing for finance: overview and prospects. Rev. Phys. **4**, 100028 (2019)

10. M.S. Palsson, M. Gu, J. Ho, H.M. Wiseman, G.J. Pryde, Experimentally modeling stochastic processes with less memory by the use of a quantum processor. Sci. Adv. **3** (2017)
11. J. Li, S. Kais, Entanglement classifier in chemical reactions. Sci. Adv. **5**, eaax5283 (2019)
12. C.H. Bennett, F. Bessette, G. Brassard, L. Salvail, J. Smolin, Experimental quantum cryptography. J. Crypt. **5**, 3 (1992)
13. A. Pathak, *Elements of Quantum Computation and Quantum Communication* (CRC Press, 2018)
14. S. Shahane, S. Shendye, A. Shaikh, Implementation of artificial neural network learning methods on embedded platform. Int. J. Electric. Electron. Comput. Syst. **2**, 2347 (2014)
15. T.M. Mitchell, *Machine Learning* (McGraw-Hill, 2006)
16. D. Angluin, Computational learning theory: Survey and selected bibliography, in *Proceedings of 24th Annual ACM Symposium on Theory of Computing* (1992), pp. 351–369
17. L. Valiant, Commun. ACM **27**(11), 1134–1142 (1984)
18. V.N. Vapnik, A. Chervonenkis, On uniform convergence of relative frequencies of events to their probabilities. Theor. Prob. Appl. **16**, 264 (1971)
19. M. Schuld, I. Sinayskiy, F. Petruccione, An introduction to quantum machine learning. arXiv:1409.3097v1
20. S. Lloyd, M. Mohseni, P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411
21. E. Farhi, H. Neven, Classification with quantum neural networks on near term processors. arXiv:1802.06002v2
22. S. Lu, S.L. Braunstein, Quantum decision tree classifier. Quantum. Inf. Process (2013)
23. S. Lloyd, Quantum algorithm for solving linear systems of equations. APS March Meeting Abstracts (2010)
24. S. Aaronson, Read the fine print. Nat. Phys. **11**(4), 291 (2015)
25. A.M. Childs, R. Kothari, R.D. Somma, SIAM J. Comput. **46**, 1920 (2017)
26. B.D. Clader, B.C. Jacobs, C.R. Sprouse, Preconditioned quantum linear system algorithm. Phys. Rev. Lett. **110.25**, 250504 (2013)
27. D. Dervovic et al., *Quantum Linear Systems Algorithms: A Primer* (2018). arXiv:1802.08227
28. S. Dutta et al., *Demonstration of a Quantum Circuit Design Methodology for Multiple Regression* (2018). arXiv:1811.01726
29. E. Tang, A quantum-inspired classical algorithm for recommendation systems, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (ACM, 2019)
30. E. Tang, Quantum-inspired classical algorithms for principal component analysis and supervised clustering (2018). arXiv:1811.00414
31. A. Gilyén, S. Lloyd, E. Tang, Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension (2018). arXiv:1811.04909
32. N.-H. Chia, H.-H. Lin, C. Wang C, Quantum-inspired sublinear classical algorithms for solving low-rank linear systems (2018). arXiv:1811.04852
33. E. Tang, An overview of quantum-inspired classical sampling (2019). https://ewintang.com/blog/2019/01/28/an-overview-of-quantum-inspired-sampling/
34. E. Tang, Some settings supporting efficient state preparation (2019). https://ewintang.com/blog/2019/06/13/some-settings-supporting-efficient-state-preparation/
35. S. Nowozin, C.H. Lampert, Structured learning and prediction in computer vision. Found. Trends Comput. Graph. Vis. **6**, 185–365 (2011)
36. M.N. Wernick, Y. Yang, J.G. Brankov et al., Machine learning in medical imaging. IEEE **27**, 25–38 (2010)
37. B.J. Erickson, P. Korfiatis, Z. Akkus, T.L. Kline, Machine learning for medical imaging. Rad. Graph. **37**, 505–515 (2017)
38. A. Lavecchia, Machine-learning approaches in drug discovey: methods 5nd applications. Sci. Dir. **20**, 318–331 (2015)
39. C. Bahlmann, B. Haasdonk, H. Burkhardt, Online handwriting recognition with support vector machines—a kernel approach, in *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition* (2002)

40. L. Chen, C. Ren, L. Li et al., A comparative assessment of geostatistical, machine learning, and hybrid approaches for mapping topsoil organic carbon content. Int. J. Geo-Inf. **8**(4), 174 (2019)
41. Z. Li, X. Lui, N. Xu, J. Du, Experimental realization of a quantum support vector machine. Phys. Rev. Lett. **114**, 140504 (2015)
42. I. Kerenidis, A. Prakash, D. Szilágyi, Quantum algorithms for second-order cone programming and support vector machines (2019). arXiv:1908.06720
43. A.K. Bishwas, A. Mani, V. Palade, Big data quantum support vector clustering (2018). arXiv:1804.10905
44. Arodz, T., Saeedi, S. Quantum sparse support vector machines (2019). arXiv:1902.01879
45. C. Ding, T. Bao, H. Huang, Quantum-inspired support vector machine (2019). arXiv:1906.08902
46. D. Anguita, S. Ridella, F. Rivieccio, R. Zunino, Quantum optimization for training support vector machines. Neural Netw. **16**, 763–770 (2003)
47. B.J. Chelliah, S. Shreyasi, A. Pandey, K. Singh, Experimental comparison of quantum and classical support vector machines. IJITEE **8**, 208–211 (2019)
48. M. Schuld, A. Bacharov, K. Svore, N. Wiebe (2018). arXiv:1804.00633v1
49. E. Farhi, H. Neven (2018). arXiv:1802.06002v2
50. M. Schuld, I. Sinayskiy, F. Petruccione, An introduction to quantum machine learning. Cont. Phys. **56**, 172–185 (2015)
51. E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. Green, S. Severini, npj Quantum Inf. **4**, 65 (2018)
52. Y.-Y. Shi, Y.-Y., Duan, L.-M., and Vidal,G., Classical simulation of quantum many-body systems with a tree tensor network, Phys. Rev. A **74**, 022320 (2006)
53. G. Vidal, Class of quantum many-body states that can be efficiently simulated. Phys. Rev. Lett. **101**, 110501 (2008)
54. L. Cincio, J. Dziarmaga, M.M. Rams, Multiscale entanglement renormalization ansatz in two dimensions: quantum ising model. Phys. Rev. Lett. **100**, 240603 (2008)
55. G. Evenbly, G. Vidal, Entanglement renormalization in noninteracting fermionic systems. Phys. Rev. B **81**, 235102 (2010)
56. D. Turkpençe et al., A Steady state quantum classifier. Phys. Lett. A **383**, 1410 (2019)
57. H.P. Breuer, F. Petruccione, *The Theory of Open Quantum Systems* (Oxford University Press, Oxford, 2007)
58. W.O. Krawec, M.G. Nelson, E.P. Geiss, Automatic generation of optimal quantum key distribution protocols, in *Proceedings of Genetic and Evolutionary Computation* (New York, ACM, 2017), p. 1153
59. P.B. Wigley, P.J. Everitt, A. van den Hengel, J.W. Bastian, M.A. Sooriyabandara, N.P. McDonald, G.D. Hardman, K.S. Quinlivan, C.D. Manju, P. Kuhn, C.C.N. Petersen, I.R. Luiten, A.N. Hope, J.J. Robins, M.R. Hush, Fast machine-learning online optimization of ultra-cold-atom experiments. Sci. Rep. **6**, 25890 (2016)
60. A. Einstein, B. Podolsky, N. Rosen, Can quantum-mechanical description of physical reality be considered complete? Phys. Rev. **47**, 777 (1935)
61. G. Cristian Romero. https://www.qutisgroup.com/wp-content/uploads/2014/10/TFG-Cristian-Romero.pdf
62. X.D. Cai, D. Wu, Z.-E. Su, M.C. Chen, X.L. Wang, L. Li, N.L. Liu, C.Y. Lu, J.W. Pan, Entanglement-based machine learning on a quantum computer. Phys. Rev. Lett. **114**, 110504 (2015)
63. Y. Liu, X. Zhang, M. Lewenstein, S.J. Ran, Entanglement-guided architectures of machine learning by quantum tensor network (2018). arXiv:1803.09111
64. Y. Levine, O. Sharir, N. Cohen, A. Shashua, Quantum entanglement in deep learning architectures. Phys. Rev. Lett. **122**, 065301 (2019)
65. B. Farley, W. Clark, Simulation of self-organizing systems by digital computer. Trans. IRE Profess. Gr. on Inf. Theor. **4**, 76 (1954)

66. P. Smolensky, Chapter 6: information processing in dynamical systems: foundations of harmony theory, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1*
67. D.L. Deng, X. Li, S.D. Sarma, Quantum entanglement in neural network states. Phy. Rev. X **7**, 021021 (2017)
68. L. Susskind, J. Lindesay, An introduction to black holes, information and the string theory revolution: the holographic universe. World Sci. **200** (2004)
69. Q. Zhuang, Z. Zhang, Supervised learning enhanced by an entangled sensor network (2019). arXiv:1901.09566
70. I. Goodfellow, Y. Bengio, A. Courville, Deep learning. Gen. Program. Evolv. Machin. **19**, 305 (2018)
71. S.C. Kak, Quantum neural computing. Adv. Imag. Elect. Phys. **94**, 259 (1995)
72. T. Menneer, A. Narayanan, Quantum-inspired neural networks. Tech. Rep. **R329** (1995)
73. M. Perus, Neuro-quantum parallelism in brain-mind and computers. Informatica **20**, 173 (1996)
74. T. Menneer, Quantum artificial neural networks, Ph.D. thesis, University of Exeter (1998)
75. Faber, J., and Giraldi, G. A., Quantum Models for Artificial Neural Networks, LNCC–National Laboratory for Scientific Computing
76. M. Schuld, I. Sinayskiy, F. Petruccione, The quest for a quantum neural network. Quantum Inf. Process. **13**, 2567 (2014)
77. N. Wiebe, A. Kapoor, K.M. Svore, Quantum deep learning (2014). arXiv:1412.3489
78. M.V. Altaisky, Quantum neural network (2000) arxiv:quant-ph/0107012
79. S. Gupta, R.K.P. Zia, Quantum neural networks. J. Comput. Sys. Sci. **63**, 355 (2001)
80. M. Zidan, A.-H. Abdel-Aty, M. El-shafei, M. Feraig, Y. Al-Sbou, H. Eleuch, M. Abdel-Aty, Quantum classification algorithm based on competitive learning neural network and entanglement measure. Appl. Sci. **9**, 1277 (2019)
81. A. Sagheer, M. Zidan, M.M. Abdelsamea, A novel autonomous perceptron model for pattern classification applications. Entropy **21**, 763 (2019)
82. S. Lloyd, C. Weedbrook, Quantum generative adversarial learning. Phys. Rev. Lett. **121**, 040502 (2018)
83. S. Kak, On quantum neural computing. Inf. Sci. **83**, 143 (1995)
84. M. Zak, C.P. Williams, Quantum neural nets. Int. J. Theor. Phys. **37**, 651 (1998)
85. Y. Cao, G.G. Guerreschi, A. Aspuru-Guzik, Quantum neuron: an elementary building block for machine learning on quantum computers (2017). arXiv:1711.11240
86. S. Hayou, A. Doucet, J. Rousseau, On the impact of the activation function on deep neural networks training (2019). arXiv:1902.06853
87. F. Agostinelli, M. Hoffman, P. Sadowski, P. Baldi, Learning activation functions to improve deep neural networks (2015). arXiv:1412.6830
88. I. Daubechies, R. DeVore, S. Foucart, B. Hanin, G. Petrova, Nonlinear approximation and (Deep) ReLU networks (2019). arXiv:1905.02199
89. F. Neukart, S.A. Moraru, On quantum computers and artificial neural networks. Sig. Process. Res. **2**, 1 (2013)
90. M. Schuld, I. Sinayskiy, F. Petruccione, Simulating a perceptron on a quantum computer. Phys. Lett. A **7**, 660 (2015)
91. U. Alvarez-Rodriguez, L. Lamata, P.E. Montero, J.D. Martín-Guerrero, E. Solano, Supervised quantum learning without measurements. Sci. Rep. **7**, 13645 (2017)
92. K.H. Wan, O. Dahlsten, H. Kristjánsson, R. Gardner, M.S. Kim, Quantum generalisation of feedforward neural networks. npj Quantum Inf. **3**, 36 (2017)
93. P. Rebentrost, T.R. Bromley, C. Weedbrook, S. Lloyd, Quantum Hopfield neural network. Phys. Rev. A **98**, 042308 (2018)
94. J.S. Otterbach, et al., Unsupervised machine learning on a hybrid quantum computer (2017). arXiv:1712.05771
95. L. Lamata, Basic protocols in quantum reinforcement learning with superconducting circuits. Sci. Rep. **7**, 1609 (2017)

96. F. Tacchino, C. Macchiavello, D. Gerace, D. Bajoni, An artificial neuron implemented on an actual quantum processor. npj Quantum Inf. **5**, 26 (2019)
97. D.H. Hubel, T.N. Wiesel, Receptive fields and functional architecture of monkey striate cortex. J. Physiol. (1968)
98. J. Fan, W. Xu, Y. Wu, Y. Gong, Human tracking using convolutional neural networks. IEEE Trans. Neural Netw. **21**, 1610 (2010)
99. M. Jaderberg, A. Vedaldi, A. Zisserman, Deep features for text spotting, in *European Conference on Computer Visions* (2014)
100. A. Toshev, C. Szegedy, Deep-pose: human pose estimation via deepneural networks. CVPR (2014)
101. J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: a deep convolutional activation feature for generic (2014)
102. C. Farabet, C. Couprie, L. Najman, Y. LeCun, Learning hierarchical features for scene labeling. PAMI (2013)
103. R. Zhao, W. Ouyang, H. Li, X. Wang, Saliency detection by multicontext deep learning, in *CVPR* (2015)
104. N. Aloysius, M.A. Geetha, Review on deep convolutional neural networks, in *International Conference on Communication and Signal Processing* (India, 2017)
105. G. Carleo, M. Troyer, Solving the quantum many-body problem with artificial neural networks. Science **355**, 602 (2017)
106. E.P. Van Nieuwenburg, Y.H. Liu, S.D. Huber, Learning phase transitions by confusion. Nat. Phys. **13**, 435–439 (2017)
107. N. Maskara, A. Kubica, T. Jochym-O'Connor, Advantages of versatile neural-network decoding for topological codes. Phys. Rev. A **99**, 052351 (2019)
108. Y. Zhang, E.A. Kim, Quantum loop topography for machine learning. Phys. Rev. Lett. **118**, 216401 (2017)
109. J. Carrasquilla, R.G. Melko, Machine learning phases of matter. Nat. Phys. **13**, 431–434 (2017)
110. L. Wang, Discovering phase transitions with supervised learning. Phys. Rev. B **94**, 195105 (2016)
111. Y. Levine, N. Cohen, A. Shashua, Quantum entanglement in deep learning architectures. Phys. Rev. Lett. **122**, 065301 (2019)
112. J. Biamonte et al., Quantum machine learning. Nature **549**, 195–202 (2017)
113. V. Dunjko, J.M. Taylor, H.J. Briegel, Quantum-enhanced machine learning. Phys. Rev. Lett. **117**, 130501 (2016)
114. E. Farhi, H. Neven, Classification with quantum neural networks on near term processors (2018). arXiv: 1802.06002
115. W. Huggins, P. Patil, B. Mitchell, K.B. Whaley, E.M. Stoudenmire, Towards quantum machine learning with tensor networks. Quantum Sci. Tech. **4**, 024001 (2018)
116. I. Cong, S. Choi, M.D. Lukin, Quantum convolutional neural networks. Nat. Phys. **15**, 1273 (2019)
117. Z.A. Jia, B. Yi, R. Zhai, Y.-C. Wu, G.-C. Guo, G.-P. Guo, Quantum neural network states: a brief review of methods and applications. Adv. Quantum Tech. **1800077** (2019)
118. C. Monterola, C. Saloma, Solving the nonlinear schrodinger equation with an unsupervised neural net-work. Opts. Exp. **9**, 72 (2001)
119. C. Caetano, J. Reis Jr., J. Amorim, M.R. Lemes, A.D. Pino Jr., Using neural networks to solve nonlinear differential equations in atomic and molecular physics. Int. J. Quantum Chem. **111**, 2732 (2011)
120. X. Gao, L.-M. Duan, Efficient representation of quantum many-body states with deep neural networks. Nat. Comm. **8**, 662 (2017)
121. A.P. Dash, S. Sahu, S. Kar, B.K. Behera, P.K. Panigrahi, Explicit demonstration of initial state construction in artificial neural networks usingNetKet and IBM Q experience platform. ResearchGate- (2019). https://doi.org/10.13140/RG.2.2.30229.17129

122. B. Gardas, M.M. Rams, J. Dziarmaga, Quantum neural networks to simulate many-body quantum systems. Phys. Rev. B **98**, 184304 (2018)
123. J. Bjarni, B. Bela, G. Carleo, Neural-network states for the classical simulation of quantum computing (2018). arXiv:1808.05232v1
124. H. Liu, C. Yu, S. Pan, S. Qin, F. Gao, Q. Wen (2019). arXiv:1906.03834v2 [quant-ph]
125. P. Rebetrost, T.R. Bromley, C. Weedbrook, S. Lloyd, Quantum hopfield neural network. Phy. Rev. A **98**, 042308 (2018)