Recent advances for quantum classifiers

Weikang Li[1] and Dong-Ling Deng[1,2,*]

[1]*Center for Quantum Information, IIIS, Tsinghua University, Beijing 100084, People's Republic of China*
[2]*Shanghai Qi Zhi Institute, 41th Floor, AI Tower, No. 701 Yunjin Road, Xuhui District, Shanghai 200232, China*

(Dated: September 1, 2021)

Machine learning has achieved dramatic success in a broad spectrum of applications. Its interplay with quantum physics may lead to unprecedented perspectives for both fundamental research and commercial applications, giving rise to an emergent research frontier of quantum machine learning. Along this line, quantum classifiers, which are quantum devices that aim to solve classification problems in machine learning, have attracted tremendous attention recently. In this review, we give a relatively comprehensive overview for the studies of quantum classifiers, with a focus on recent advances. First, we will review a number of quantum classification algorithms, including quantum support vector machine, quantum kernel methods, quantum decision tree, and quantum nearest neighbor algorithm. Then, we move on to introduce the variational quantum classifiers, which are essentially variational quantum circuits for classifications. We will review different architectures for constructing variational quantum classifiers and introduce the barren plateau problem, where the training of quantum classifiers might be hindered by the exponentially vanishing gradient. In addition, the vulnerability aspect of quantum classifiers in the setting of adversarial learning and the recent experimental progress on different quantum classifiers will also be discussed.

CONTENTS

I. INTRODUCTION

The interplay between machine learning and quantum physics may lead to unprecedented perspectives for both fields [1]. In recent years, machine learning has revolutionized many new techniques in both research and commercial fields, ranging from image recognition to automated driven cars [2–4]. More strikingly, some notoriously challenging problems for traditional methods have been cracked successfully. Notable examples include playing the game of Go with deep neural networks and tree search [5, 6], classifying skin cancers with convolutional neural networks [7], and predicting protein structures with the AlphaFold system [8]. Proceeding along this direction, machine learning methods may likewise shed new light on solving complex problems in quantum physics [9, 10]. In parallel, quantum computing has made fascinating progresses in recent years, with the experimental demonstrations of quantum supremacy marked as the latest milestone [11–13]. Nowadays, a number of notable algorithms [14–28], such as the the Harrow-Hassidim-Lloyd algorithm [18], quantum principal component analysis [19], quantum-enhanced feature space [23, 24], quantum generative models [25–27], quantum support vector machines [20], etc., have been proposed and shown to hold the potential of exhibiting quantum advantages over their classical counterparts. These advances have ignited tremendous interest in exploring enhanced machine learning models with quantum devices.

Classifications, as one of the most important branches in machine learning, are nowadays widely applied in commercial and academic applications ranging from face recognition and recommendation systems to earthquake detection and disease diagnosis [2–4, 7]. With the rapid development of quantum computing theory in the past few years, it is promising to develop quantum enhanced classification models that may be able to handle complex classification tasks. To date, a number of quantum classification models and relevant techniques have already been proposed theoretically with some of them even experimentally demonstrated [20, 23, 24, 29–75]. These works have enriched the studies of quantum classifiers from various aspects and may provide valuable guidance for more advanced models in the future [76–78].

In general, the classification process involves a mapping from the input data to different categories. For supervised learning tasks, it requires that there is already some labeled data serving as the training set, e.g. some handwritten digits with their true labels. After the learning process, the trained

* dldeng@tsinghua.edu.cn

model is expected to master the data's pattern and to be used to classify the unseen data. In the early studies of quantum classifiers, some works have extended popular classical classification algorithms to the quantum domain. Renowned examples include the quantum support vector machine [20], quantum nearest neighbor algorithms [51, 52], and quantum decision tree classifiers [53]. Rather than merely implementing the extensions from classical to quantum, among them there are also approaches exhibiting the potential of providing quadratic or even exponential speedups. Moreover, it is desirable to design new quantum classifiers by encapsulating the above mentioned algorithms, such as the Harrow-Hassidim-Lloyd algorithm [18] and quantum principal component analysis [19], as subroutines. Recently, quantum kernel methods have drawn a lot of attention. In Ref. [20], the authors have proposed to use quantum devices to efficiently prepare the linear or polynomial kernel matrices. When dealing with a classification task, the choice of the input feature space is crucial for the classification performance, e.g. the support vector machine with a linear kernel may not work to classify a dataset that is not linearly separable. Along this line, several quantum kernel methods have been reported in both theoretical studies and experimental demonstrations [23, 24, 29, 31, 32, 79]. In particular, Refs. [23, 24, 29] propose to use the quantum devices to estimate the kernel matrices and leave the rest computations to classical computers, which might be applicable in the age of noisy intermediate-scale quantum devices [78]. It is also worth mentioning that, in Ref. [79], the authors have proposed a quantum classification algorithm with a rigorous and robust speedup, assuming the hardness of the renowned discrete logarithm problem.

The most successful programming paradigm in machine learning relies on artificial neural networks, a highly abstracted and simplified model of the human brain [2, 3]. An artificial neural network consists of a collection of connected units or nodes called artificial neurons, which are typically organized in layers. There are various types of neural networks, including feedforward [80, 81], convolutional [82], recurrent [83, 84], and capsule neural networks [85, 86], each of which bears their own particular structures and levels of complexity. These networks usually contain a large amount of parameters that can be adjusted during the training procedure. Due to the well-designed structures and the nonlinear components, e.g. the activation functions and the pooling layers, some neural networks exhibit outstanding expressive power as classifiers. Inspired by the achievements in classical machine learning, variational quantum classifiers, which inherit some features of the classical neural networks, have attracted a wide range of attention and rapid progress has been made over the recent years [77]. Similar to the classical case, variational quantum circuits contain variational parameters that can be optimized during the training procedure. During the optimization process, a key step is to calculate the derivatives of the target functions with respect to the circuit parameters [36–38, 41, 54, 56–63, 72, 73]. In particular, the widely used "parameter shift rule" has been proposed and developed [37, 38, 41, 54, 61]. In addition to the optimization strategies, a variety of variational quantum classifiers with different structures have emerged [36, 42–45, 47, 66–71, 74, 75]. Among these works, some of them extend straightforwardly the concepts from the classical neural networks to the quantum domain, e.g. the quantum convolutional neural networks [45] and the continuous-variable quantum neural networks (which provide the frameworks suitable for constructing convolutional networks, recurrent networks *et al.*) [42]. Some other works use special structures inspired by physics to build quantum classifiers, e.g. the tree tensor network classifiers, the multi-scale entanglement renormalization ansatz classifiers [43] and the multi-level quantum system classifiers [47]. It is worth mentioning that there are algorithms that search for appropriate structures of the variational quantum circuits for certain tasks [87–99], e.g. a quantum neuroevolution algorithm that autonomously finds suitable quantum neural networks [92] and a differentiable quantum architecture search algorithm that allows automated quantum circuit designs in an end-to-end differentiable fashion [89].

Though quantum classifiers have drawn a wide range of interest and investigation, there are crucial challenges that may hinder their performance and should be studied in depth: the barren plateau phenomena during the training process and vulnerability of quantum classifiers. First, the barren plateaus refer to the phenomenon in the optimization procedure where the gradients concentrate to zero exponentially fast as the system size increases [100]. To illustrate the phenomenon, we consider the initial parameters in the high-dimensional parameter space with the cost function being an additional dimension. At ideal conditions, during the training process the parameters can converge to the point corresponding to the minimal value of the cost function with decent speed, just like a snow ball rolling down the hill. However, the barren plateaus present the situation that the cost function with respect to the parameters is too flat in most of the space except a small area around the optimal point, thus it is nearly impossible to converge to the optimal point. Nowadays, this phenomenon has attracted broad interest across communities, with many works analyzing the reasons why barren plateau occurs and designing strategies to avoid it [101–118]. The reasons underlying this phenomenon include the poor design of the quantum circuits [101], non-negligible noises [104], massive entanglement [106], and inappropriate choice of cost functions [102, 118]. Second, the vulnerability of quantum classifiers concerns their performance against adversarial attacks. Adversarial examples refer to input samples that an attacker has crafted in order to fool the machine learning models. For a simple example, a classifier is supposed to assigned a picture to the class "dog". If we add a carefully designed and imperceptible perturbation to this picture, then it may be assigned to the class "cat" incorrectly by the classifier. The vulnerability of quantum machine learning models to crafted adversarial examples as well as the design of defense strategies are actively investigated at the current stage with both numerical simulations and analytical results [39, 119–123].

In parallel to the theoretical development, stimulating progress has also been made in the experimental side [23, 29–32, 55]. For example, short after the theoretical proposal of the quantum support vector machines and quantum nearest

neighbor algorithms, in Refs. [30, 55] the authors have successfully demonstrated their proof-of-principle experiments. More recently, both the kernel based classifications [23] and the estimation of kernel matrices [23, 29] have been implemented on the optical and superconducting platforms, respectively. These quantum experiments may be carried out with higher accuracy and at a lower cost with the development of quantum devices. And by that time, more practical quantum classifiers might be implemented and it is appealing to see some of them served in commercial applications.

This review is organized as follows. In Section 2, we will introduce some important algorithms and models for quantum classifiers. This section can be divided into several subsections including the quantum support vector machines, quantum decision tree, quantum nearest neighbors algorithms, and variational quantum classifiers. To better present these advances, we will introduce some relevant classical concepts first and then focus on the quantum models. In addition to the theoretical progresses, the experimental advances will be reviewed in Section 3. In the last section, we will summarize this review and give an outlook for the future research.

II. QUANTUM CLASSIFIERS

A. Quantum support vector machine and kernel methods

Support vector machines (SVMs) are widely used machine learning models and can be used to find the optimal hyperplane for classification tasks [124–126]. For a linearly separable dataset given as

$$\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_M, y_M)\}, \tag{1}$$

where M is the size of the training dataset with data dimension N, it is feasible to construct an $(N-1)$-dimensional hyperplane with the maximum margin to optimally classify the data in the N-dimensional hyperspace. For simplicity, we only consider the binary classification task as an example and denote the labels as $y_i = \pm 1$. This basic idea is briefly illustrated in Fig. 1(a).

We use a set of parameters $(\vec{\omega}, b)$ to denote a hyperplane in the N-dimensional space. In general, the hyperplane that classifies the dataset may not be unique, i.e., there exists more than one solution for $\vec{w}$ and b such that for every training data, $y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1$ is satisfied. For SVMs, the task is to find the maximum separation between the two classes, i.e., to maximize the distance $\frac{2}{|\vec{w}|}$ between the space $\vec{w} \cdot \vec{x}_i + b \geq 1$ ($+1$ class) and $\vec{w} \cdot \vec{x}_i + b \leq -1$ (-1 class). Then, the decision function can be written as

$$f(\vec{x}) = \operatorname{sgn}\left(\vec{w}^* \cdot \vec{x} + b^*\right), \tag{2}$$

where $\vec{w^*}$ and b^* are the optimal values for the parameters obtained from the optimization procedure.

To solve the original optimization problem (also called the primal problem in the literature), it is helpful to transform it to the corresponding dual problem. With Lagrange multipliers, the goal can be reduced to maximizing

$$L(\vec{\alpha}) = \sum_{j=1}^{M} y_j \alpha_j - \frac{1}{2} \sum_{j,k=1}^{M} \alpha_j (\vec{x}_j \cdot \vec{x}_k) \alpha_k, \tag{3}$$

where $\sum_{j=1}^{M} \alpha_j = 0$ and $y_j \alpha_j \geq 0$. Then the weights and bias can be recovered as $\vec{w}^* = \sum_{j=1}^{M} \alpha_j^* \vec{x}_j$ and $b^* = y_k - \vec{w}^* \cdot \vec{x}_k$ for an index k corresponding to a non-zero α_k^*. Actually, we can directly plug these expressions into the decision function in Eq. (2) and obtain

$$f(\vec{x}) = \operatorname{sgn}\left(\sum_{j=1}^{M} \alpha_j^* (\vec{x}_j \cdot \vec{x}) + b^*\right). \tag{4}$$

For cases where the dataset is nearly linearly separable, i.e., there are a few outliers that violate the linearity, the hard-margin condition can be relaxed to a soft-margin one: $y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i$, where ξ_i is a cost [124–126]. Then we can use the similar optimization procedure to solve the problem. Since noisy data is more general in the physical world, the soft-margin condition is more often used than the hard-margin condition in practice.

However, when the dataset is not linearly separable, such as the example shown in Fig.1(b), the above SVM approach may not work. To handle this issue, kernel methods are applied to help build a hyperplane in a higher dimensional kernel space and make the data linearly separable again [127]. This can be briefly illustrated as follows. As shown in Fig.1(b), the data samples in the feature space (x_1, x_2) are not linearly separable. However, if the feature space can be transformed into $(x_1, x_2, x_1^2 + x_2^2)$, then it is obvious that the samples in the new basis become linearly separable as shown in Fig.1(c). The kernel function is defined as

$$K(x, z) = \phi(x) \cdot \phi(z), \tag{5}$$

where $\phi(x)$ is the function mapping the feature space to the kernel space. With an appropriate choice of the kernel function, the data sample can be mapped into a higher dimensional space and becomes linearly separable. We note that $\phi(x)$ does not need to be explicitly defined, since we will only use the kernel function to compute the inner product of the vectors in the kernel space. In this case, the dual problem becomes maximizing

$$L(\vec{\alpha}) = \sum_{j=1}^{M} y_j \alpha_j - \frac{1}{2} \sum_{j,k=1}^{M} \alpha_j K_{jk} \alpha_k, \tag{6}$$

where $\sum_{j=1}^{M} \alpha_j = 0$ and $y_j \alpha_j \geq 0$. After obtaining the optimal parameters for $\vec{\alpha}$, the decision function can then be formulated similar to the linear case as

$$f(\vec{x}) = \operatorname{sgn}\left(\sum_{j=1}^{M} \alpha_j^* K\left(\vec{x}_j, \vec{x}\right) + b^*\right). \tag{7}$$

In this way, the kernel methods can be utilized to efficiently handle some non-linear problems in the SVM framework.

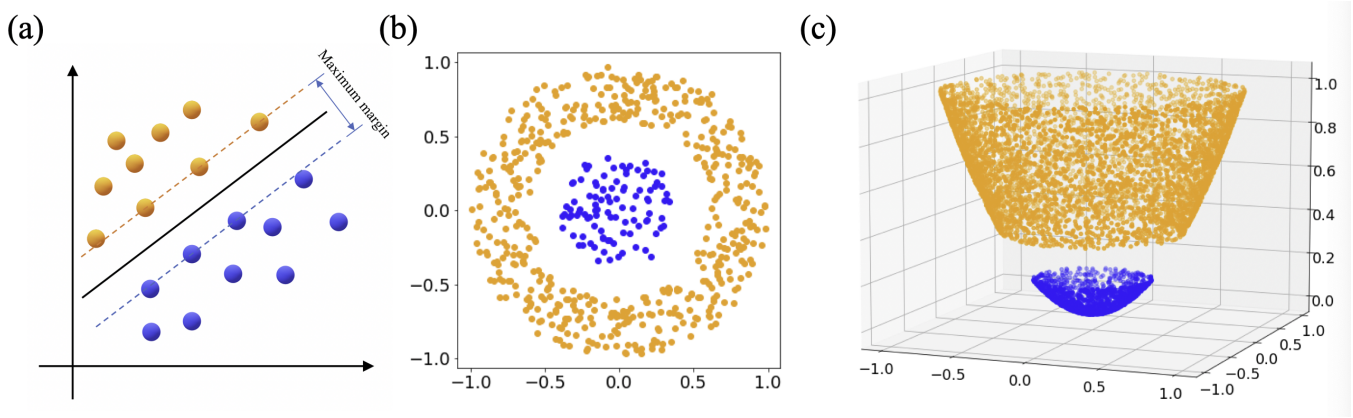

FIG. 1. (a) A schematic diagram for support vector machines classifying a dataset, where the two kinds of data points (marked as orange and blue balls, respectively) are linearly separable. (b) A situation where the dataset is not linearly separable. (c) A method to solve the problem in (b) using kernel methods to map the data to a higher dimensional space.

We now introduce the quantum support vector machines (QSVMs) [20], which are based on a least-squares version of classical SVMs [128]. Compared with the traditional SVMs we introduced above, the least-squares SVMs consider equality type constraints rather than inequality type constraints with slack variables e_j introduced to transform the constraints as

$$y_j(\vec{w}\cdot\vec{x}_j+b)\geq 1 \Rightarrow \vec{x}_j+b=y_j-y_je_j. \tag{8}$$

Besides, a regularization term $\frac{\gamma}{2}\Sigma_{j=1}^{M}e_j^2$ is added to the Lagrangian to be optimized, where γ is a hyperparameter describing the ratio of the Lagrangian's components. As a consequence, the problem after applying the optimization of the Lagrangian can be formulated as a linear equation:

$$\mathbf{F}\begin{pmatrix} b \\ \vec{\alpha}\end{pmatrix}=\begin{pmatrix} \mathbf{0} & \vec{1}^T \\ \vec{1} & \mathbf{K}+\gamma^{-1}\mathbf{I}\end{pmatrix}\begin{pmatrix} b \\ \vec{\alpha}\end{pmatrix}=\begin{pmatrix} 0 \\ \vec{y}\end{pmatrix}, \tag{9}$$

where $\mathbf{K}$ is an $M\times M$ kernel matrix and the matrix $\mathbf{F}$ is $(M+1)\times(M+1)$ dimensional. In the quantum case, the above equation can be rewritten compactly as $\hat{\mathbf{F}}\left|b,\vec{\alpha}\right\rangle=\left|\vec{y}\right\rangle$, where $\hat{\mathbf{F}}=\mathbf{F}/\operatorname{Tr}(\mathbf{F})$.

With this formulation, one may consider the Harrow-Hassidim-Lloyd (HHL) algorithm to solve the above linear equations [18]. For this purpose, it is necessary that $\hat{\mathbf{F}}$ can be both created and exponentiated efficiently. For simplicity, $\hat{\mathbf{F}}$ can be written as $(\gamma^{-1}\mathbf{I}+\mathbf{J}+\mathbf{K})/\operatorname{Tr}(\mathbf{F})$ with $\mathbf{J}=\begin{pmatrix}\mathbf{0} & \vec{1}^T \\ \vec{1} & \mathbf{0}\end{pmatrix}$. Actually, with the help of Lie product formula, $e^{-i\hat{\mathbf{F}}\Delta t}$ can be decomposed as

$$e^{-i\hat{\mathbf{F}}\Delta t}=e^{-\frac{i\gamma^{-1}\mathbf{I}\Delta t}{\operatorname{Tr}(\mathbf{F})}}e^{-\frac{i\mathbf{J}\Delta t}{\operatorname{Tr}(\mathbf{F})}}e^{-\frac{i\mathbf{K}\Delta t}{\operatorname{Tr}(\mathbf{F})}}+O(\Delta t^2). \tag{10}$$

The exponentiation operations for $\mathbf{I}$ and $\mathbf{J}$ are straightforward, as the eigenvalues and eigenstates for these two matrices can be analytically calculated. The main challenges are to create and exponentiate $\mathbf{K}$ efficiently. To overcome these challenges, the first step is to call the training data oracle [19]:

$$1/\sqrt{M}\sum_{i=1}^{M}\left|i\right\rangle \Rightarrow 1/\sqrt{N_\chi}\sum_{i=1}^{M}\left|\vec{x}_i\right\|i\rangle\left|\vec{x}_i\right\rangle, \tag{11}$$

where $N_\chi=\sum_{i=1}^{M}\left|\vec{x}_i\right|^2$ denotes the normalization factor. Then we trace out the second register and obtain the reduced density matrix $\hat{\mathbf{K}}=\mathbf{K}/\operatorname{Tr}(\mathbf{K})$, which accomplishes the challenge of creating $\mathbf{K}$. For the exponentiation of $\hat{\mathbf{K}}$, Lloyd, Mohseni, and Rebentrost have introduced a powerful method in Ref. [19]. Suppose we want to apply $e^{-i\hat{\mathbf{K}}\Delta t}$ to any density matrix ρ. Instead of trying to construct $e^{-i\hat{\mathbf{K}}\Delta t}$ directly, the following fact can be utilized

$$\begin{aligned} e^{-i\hat{\mathbf{K}}\Delta t}\rho e^{i\hat{\mathbf{K}}\Delta t} &\approx \operatorname{Tr}_1\left\{e^{-iS\Delta t}\hat{\mathbf{K}}\otimes\rho e^{iS\Delta t}\right\} \\ &= \rho-i\Delta t[\hat{\mathbf{K}},\rho]+O\left(\Delta t^2\right), \end{aligned} \tag{12}$$

where Tr_1 means tracing out the first register, and S is a SWAP operator which is sparse and easy to exponentiate. This result indicates that we can approximately evolve the density matrix ρ by time Δt efficiently using S as the Hamiltonian and tracing out the first register. Back to the task of $\hat{\mathbf{F}}$'s exponentiation, $\hat{\mathbf{K}}$ and $\mathbf{K}/\operatorname{Tr}(\mathbf{F})$ only differ by a constant, which can be easily fixed via multiplying the time by $\operatorname{Tr}(\mathbf{K})/\operatorname{Tr}(\mathbf{F})$.

After these preparations, it is now straightforward to see that the HHL algorithm can be utilized to solve the linear Eq. (9). With the phase estimation algorithm, the eigenvalues of $\hat{\mathbf{F}}$ can be extracted to the ancillary register. Then with a controlled rotation and the uncomputing strategy, one arrives at the target state $\left|b,\vec{\alpha}\right\rangle$. This process can be summarized as

$$\left|\vec{y}\right\rangle\left|0\right\rangle\rightarrow\sum_{j=1}^{M+1}\left\langle u_j\mid\vec{y}\right\rangle\left|u_j\right\rangle\left|\lambda_j\right\rangle\rightarrow\sum_{j=1}^{M+1}\frac{\left\langle u_j\mid\vec{y}\right\rangle}{\lambda_j}\left|u_j\right\rangle, \tag{13}$$

where $\left|u_j\right\rangle$ represents the eigenstate of $\hat{\mathbf{F}}$ with eigenvalue λ_j, and the final state is the target state: $\left|b,\vec{\alpha}\right\rangle=\frac{1}{\sqrt{C}}\left(b\left|0\right\rangle+\sum_{k=1}^{M}\alpha_k\left|k\right\rangle\right)=\sum_{j=1}^{M+1}\frac{\langle u_j\left|\vec{y}\right\rangle}{\lambda_j}\left|u_j\right\rangle$.

With the above procedure, we arrive at a quantum state which encodes all the optimal parameters for the corresponding SVM. For a classification task, the training data oracle can be called again to get $\left|\vec{u}\right\rangle$ using $\left|b,\vec{\alpha}\right\rangle$ and construct the query

state $|\vec{x}\rangle$:

$$|\vec{u}\rangle = \frac{1}{\sqrt{N_{\vec{u}}}}\left(b|0\rangle|0\rangle + \sum_{k=1}^{M}\alpha_k |\vec{x}_k| |k\rangle |\vec{x}_k\rangle\right), \quad (14)$$

$$|\vec{x}\rangle = \frac{1}{\sqrt{N_{\vec{x}}}}\left(|0\rangle|0\rangle + \sum_{k=1}^{M}|\vec{x}||k\rangle|\vec{x}\rangle\right), \quad (15)$$

where $N_{\vec{u}} = b^2 + \sum_{k=1}^{M}\alpha_k^2 |\vec{x}_k|^2$ and $N_{\vec{x}} = M|\vec{x}|^2 + 1$ are normalization factors. To classify $|x\rangle$, intuitively we can calculate the inner product of these two states since we have

$$\langle\vec{u} \mid \vec{x}\rangle = \frac{1}{\sqrt{N_{\vec{x}}N_{\vec{u}}}}\left(b + \sum_{k=1}^{M}\alpha_k |\vec{x}_k||\vec{x}| \langle\vec{x}_k \mid \vec{x}\rangle\right), \quad (16)$$

which exactly corresponds to the decision function. For the physical implementation of the inner product, a SWAP test [129] can be utilized to project the ancillary qubit in state $1/\sqrt{2}(|0\rangle|\vec{u}\rangle + |1\rangle|\vec{x}\rangle)$ to state $1/\sqrt{2}(|0\rangle - |1\rangle)$. Then the success probability is $P = 1/2(1 - \langle\vec{u} \mid \vec{x}\rangle)$, which can be used to determine $|x\rangle$'s label according to the sign of $1/2 - P$.

So far we have discussed the case of linearly separable dataset for quantum support vector machines. The extension to the nonlinear case is straightforward: one only need to map each vector for the data sample $|\vec{x}_k\rangle$ into its d-times tensor product $|\vec{x}_k\rangle \otimes \cdots \otimes |\vec{x}_k\rangle$ to obtain a polynomial kernel with degree d [20]. We mention that this trick is universally applicable, namely it can be used to construct arbitrary polynomial kernels. From the computational cost perspective, it is straightforward to obtain that the time complexity for classical SVMs scales as $O(\log(\frac{1}{\epsilon})\,\mathrm{poly}(N, M))$, where ϵ characterizes the desired accuracy. In sharp contrast, the quantum support vector machines introduced above feature a time complexity of $O(\log(NM))$ for both training and inference, giving rise to an exponential speedup over their classical counterparts. However, it is worthwhile to mention that there indeed exist several possible caveats as discussed in depth by Aaronson [130].

From the above discussion on quantum support vector machines, we see that kernel methods are very useful in certain classification tasks. More recently, a number of different quantum classification models with kernel methods have also been introduced [23, 24, 29, 31, 32]. For instance, in Ref. [31] a distance-based classifier has been proposed with the decision function

$$\tilde{y} = \mathrm{sgn}\left(\sum_{m=1}^{M} y^m\left[1 - \frac{1}{4M}|\tilde{\mathbf{x}} - \mathbf{x}^m|^2\right]\right), \quad (17)$$

where $\tilde{\mathbf{x}}$ is the input to be predicted and M is the size of the training set. The component $1 - \frac{1}{4M}|\tilde{\mathbf{x}} - \mathbf{x}^m|^2$ here can be treated as a kernel. For a balanced training set, i.e. the number of samples belonging to the two different classes is equal, the decision function can be reduced to

$$\tilde{y} = \mathrm{sgn}\left(\sum_{m=1}^{M}\left[-\frac{y^m}{4M}|\tilde{\mathbf{x}} - \mathbf{x}^m|^2\right]\right). \quad (18)$$

Then it is clear that if $\tilde{\mathbf{x}}$ belongs to the $+1$ class, the distance will be small between $\tilde{\mathbf{x}}$ and the samples in the $+1$ class, and large between $\tilde{\mathbf{x}}$ and the samples in the -1 class. As a result, the sign function can be used to assign the input sample to the right class with high probability.

To obtain a quantum analogue of the above algorithm, the amplitude encoding scheme can be used and the data with 2^n features can be embedded into a quantum state with only n qubits. In this way, the quantum state can be written as $|\psi_{\mathbf{x}}\rangle = \sum_{i=0}^{N-1} x_i|i\rangle$ up to an irrelevant normalization factor. With some data preparation methods such as quantum random access memory (QRAM) [131], the initial state can be prepared to be

$$|\mathcal{D}\rangle = \frac{1}{\sqrt{2M}}\sum_{m=1}^{M}|m\rangle\left(|0\rangle|\psi_{\tilde{\mathbf{x}}}\rangle + |1\rangle|\psi_{\mathbf{x}^m}\rangle\right)|y^m\rangle, \quad (19)$$

where $\psi_{\tilde{\mathbf{x}}}$ represents the input data and the index m represents the labeled data. Then the key step is to apply a Hadamard gate on the ancillary qubit to transform the state to

$$\frac{1}{2\sqrt{M}}\sum_{m=1}^{M}|m\rangle\left(|0\rangle|\psi_{\tilde{\mathbf{x}}+\mathbf{x}^m}\rangle + |1\rangle|\psi_{\tilde{\mathbf{x}}-\mathbf{x}^m}\rangle\right)|y^m\rangle, \quad (20)$$

where $|\psi_{\tilde{\mathbf{x}}\pm\mathbf{x}^m}\rangle = |\psi_{\tilde{\mathbf{x}}}\rangle \pm |\psi_{\mathbf{x}^m}\rangle$. If a conditional measurement on the ancillary qubit is used to project it into the $|0\rangle\langle 0|$ subspace, it will succeed with probability $\mathrm{p}_{\mathrm{acc}} = \frac{1}{4M}\sum_m |\tilde{\mathbf{x}} + \mathbf{x}^m|^2$ and the state will collapse into

$$\frac{1}{2\sqrt{M\mathrm{p}_{\mathrm{acc}}}}\sum_{m=1}^{M}\sum_{i=1}^{N}|m\rangle\left(\tilde{x}_i + x_i^m\right)|i\rangle|y^m\rangle. \quad (21)$$

In the last step, a measurement will be applied on the label qubit $|y^m\rangle$. For different outcomes, the corresponding probability will be

$$\mathrm{p}(\tilde{y} = k) = \frac{1}{4M\mathrm{p}_{\mathrm{acc}}}\sum_{m|y^m=k}|\tilde{\mathrm{x}} + \mathrm{x}^m|^2. \quad (22)$$

Then it is straightforward to assign the input sample to the class with higher probability.

Actually, after applying the Hadamard gate, the next operations can be flexible. The post-selection can also be used to choose the state $|1\rangle$ while keeping the rest steps unchanged. In doing so, the probability can be expressed as

$$\mathrm{p}(\tilde{y} = k) = \frac{1}{4M\mathrm{p}'_{\mathrm{acc}}}\sum_{m|y^m=k}|\tilde{\mathrm{x}} - \mathrm{x}^m|^2, \quad (23)$$

which also corresponds to the decision function above.

It should be noted that the two different measurement outcomes of the ancillary qubit can both serve as a classifier, while the post-selection scheme will abandon one of them. This idea is reviewed and discussed in Ref. [32] with a classification protocol proposed without post-selection. In this work, instead of choosing a particular value of the ancillary qubit to handle the task, a joint measurement is used to measure the expectation value $\left\langle\sigma_z^{(a)}\sigma_z^{(l)}\right\rangle$ of the ancillary qubit

and label qubit. In this way, both measurement outcomes of the ancillary qubit will be covered and the expectation value is

$$\left\langle \sigma_z^{(a)} \sigma_z^{(l)} \right\rangle = \sum_{m=1}^{M} (-1)^{y_m} \mathrm{Re}(\langle \tilde{\mathbf{x}} \mid \mathbf{x}_m \rangle). \tag{24}$$

The decision function can then be constructed by using the sign function. In fact, the classification scheme introduced in [31] and its variant discussed in [32] only use the real part of the quantum state overlap. To further enrich this classification model, a SWAP test classifier is also proposed in [32]. Assume that the initial state is prepared to be

$$\sum_{m=1}^{M} \sqrt{w_m} |0\rangle |\tilde{\mathbf{x}}\rangle^{\otimes n} |\mathbf{x}_m\rangle^{\otimes n} |y_m\rangle |m\rangle, \tag{25}$$

where ω_m is a controllable non-negative value. The basic idea is to apply a "$\mathrm{H}_a \cdot \mathrm{C-SWAP}^n \cdot \mathrm{H}_a$" gate (a Hadamard gate on the ancillary qubit, a Controlled-SWAP gate on the input qubits and the training data qubits controlled by the ancillary qubit, and a Hadamard gate on the ancillary qubit again) to the initial state and transform it to

$$|\Psi_f^s\rangle = \sum_{m=1}^{M} \frac{\sqrt{\omega_m}}{2} \left(|0\rangle |\psi_{n+}\rangle + |1\rangle |\psi_{n-}\rangle \right) |y_m\rangle |m\rangle, \tag{26}$$

where $|\psi_{n\pm}\rangle = |\tilde{\mathbf{x}}\rangle^{\otimes n} |\mathbf{x}_m\rangle^{\otimes n} \pm |\mathbf{x}_m\rangle^{\otimes n} |\tilde{\mathbf{x}}\rangle^{\otimes n}$. In this way, the joint measurement as used above will yield

$$\mathrm{Tr}\left(\sigma_z^{(a)} \sigma_z^{(l)} \left|\Psi_f^s\right\rangle \left\langle \Psi_f^s \right| \right) = \sum_{m=1}^{M} (-1)^{y_m} w_m \left| \langle \tilde{\mathbf{x}} \mid \mathbf{x}_m \rangle \right|^{2n},$$

and the quantum state fidelity is obtained to determine the result for the classification. For the preparation of the initial state, it is also mentioned that in many cases creating the desired state can be quite expensive and conditional. However, in this work the state preparation does not require the prior knowledge of the training data and the input test data, and has a relatively low cost. All the data can be initially prepared in separate registers in parallel. Then the Controlled-SWAP gates are applied to the test data register and different training data registers controlled by the index qubits. This operation will transform the state

$$\sum_{m}^{M} \sqrt{w_m} |0\rangle_a |\tilde{\mathbf{x}}\rangle^{\otimes n} |0\rangle_d^{\otimes n} |0\rangle_l |m\rangle \bigotimes_{k=1}^{M} |\mathbf{x}_k\rangle^{\otimes n} |y_k\rangle \tag{27}$$

to

$$\sum_{m}^{M} \sqrt{w_m} |0\rangle_a |\tilde{\mathbf{x}}\rangle^{\otimes n} |\mathbf{x}_m\rangle_d^{\otimes n} |y_m\rangle_l |m\rangle |\mathrm{junk_m}\rangle . \tag{28}$$

Since the junk register is normalized and will not influence the expectation value $\left\langle \sigma_z^{(a)} \sigma_z^{(l)} \right\rangle$, the initial state preparation is already accomplished.

Until now we have introduced several quantum algorithms that utilize kernel methods to handle classification tasks. As mentioned in Ref. [24], this approach can be viewed from some different perspectives. First, the quantum computers can be used to map the classical data x to a Hilbert space state $U_\phi(x)|0\cdots0\rangle = |\phi(x)\rangle$ and estimate the inner products between these states to obtain the kernel matrix. After that, the rest of the work is left to classical computers by passing the kernel matrix to them. Indeed, this year a rigorous quantum speedup is proved in Ref. [79], where a quantum computer is utilized to estimate a kernel function. In this work, Liu *et al*. constructed a special kind of classification problem which cannot be solved efficiently by a classical computer assuming the widely-believed hardness of the discrete logarithm problem. Whereas in the supervised learning scenario and with Shor's algorithm [15] as a subroutine, the quantum computer is able to efficiently estimate a kernel matrix to map the data to an exponentially large dimensional space and make the training data linearly separable. In addition to constructing a classification problem in a special situation, this work provides valuable guidance for exploring practical quantum algorithms with advantages over the classical computers. Second, it is proposed that the classification can be explicitly performed in the feature Hilbert space [24]. For example, the "linear weight state" $|w(\theta)\rangle$ can be prepared using a variational quantum circuit $W(\theta)$ such that $|w(\theta)\rangle = W(\theta)|0\rangle$ and the function $f(x,w) = \langle w|\phi_x\rangle$ can be used as a classification criterion. This idea about using parameterized circuits will be discussed more comprehensively in the section of variational quantum classifiers.

B. Quantum decision tree classifiers

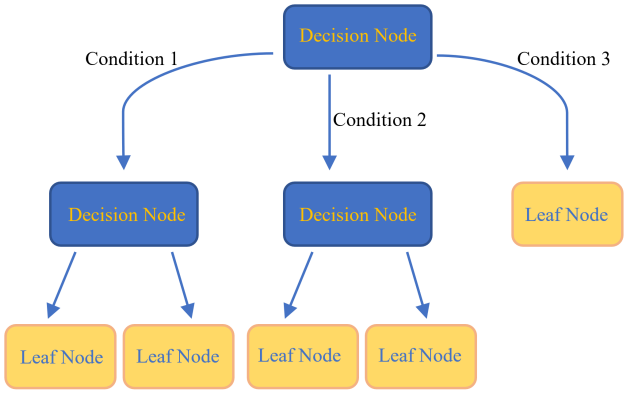

FIG. 2. A schematic illustration of a decision tree classifier. Given a new input, it will start at the top node and go down according to its features until arriving at a leaf node. Then the label of the leaf node will be assigned to this input.

A decision tree classifier is a tree-like model and a popular method for classification tasks. As shown in Fig.2, a decision tree classifier is usually build from a top decision node and then split into several branches according to different conditions, which can be viewed as a directed acyclic graph. Then the leaf nodes at the end of the directed graph represent the

classification decisions. Given a training dataset

$$\mathcal{D} = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_M, y_M)\}, \tag{29}$$

where M is the size of the dataset with d being the number of attributes, our goal is to build a decision tree according to different features and labels. After the building process of the decision tree, it is supposed to be able to classify some unseen datasets with decent accuracy and generalization performance.

Some basic ideas and algorithms of decision trees have already been summarized in [132–134] including CART algorithm, ID3 algorithm and C4.5 algorithm. Due to the decent efficiency and interpretability, decision tree method has been already applied in many classification scenarios such as in data mining [135–137].

In the first step of building a decision tree, it is important to choose appropriate feature for each node. Usually, we need to build a subtree using the features that can best improve the classification ability of the current tree, and the concept of "information gain" will be very helpful [132]. For a decision tree denoted by D, the information gain of D from feature X represents the reduction of the uncertainty of D after knowing the value of X, which can be formulated as

$$I(D; X) = H(D) - H(D|X), \tag{30}$$

where $H(D)$ is the Shannon entropy of D and $H(D|X)$ is the conditional entropy. After the enumeration of information gain over all features, it is intuitively effective to find the feature that reduces the entropy of the dataset(or sub dataset) most and pick it as the splitting node. This idea of feature choosing is reminiscent of greedy algorithms and is a suboptimal strategy. We mention that finding the optimal decision tree is an NP-complete problem [138].

With the feature choosing strategy introduced above or with some other strategies, e.g. information gain ratio, which is a modified version of information gain, the decision tree can be build accordingly. As the structure of the decision tree may grow complicated during the building procedure, it may fail to generalize to classify the unseen data due to overfitting. To deal with this problem, a pruning step might be used. More concretely, one can add a penalty term $\alpha|T|$ to the cost function, which can be expressed as

$$C_\alpha(T) = C(T) + \alpha|T|, \tag{31}$$

where $|T|$ is the complexity of the tree and α is a tunable hyperparameter. Then in the pruning process, at each iteration, we will calculate the cost function before and after pruning a leaf node. If the cost function becomes smaller after pruning, the leaf node will be removed and the algorithm will go on to check other leaf nodes until the procedure is completed.

To build a bridge between decision trees and quantum computation, the model of a quantum decision tree classifier has been proposed by Lu and Braunstein [53]. Similar to the classical setting, this model starts with a training set

$$\mathcal{D} = \{(|x_1\rangle, |y_1\rangle), (|x_2\rangle, |y_2\rangle), \ldots, (|x_M\rangle, |y_M\rangle)\}. \tag{32}$$

Following the classical algorithm, the task for building the decision tree then becomes choosing the attribute a_i that can achieve the best uncertainty reduction. When facing the choice of attributes to split the nodes, the von Neumann entropy is chosen for the quantum case instead of Shannon entropy applied in the classical case.

We mention that some other works about quantum decision trees have also been introduced in the literature [139, 140], although their major targets may not be classification problems. For instance, in Ref. [139], Farhi and Gutmann proposed quantum decision tree algorithms which are able to solve some computational problems exponentially faster than their classical counterparts. In addition, in Ref. [140] Shi connected the query complexity of quantum decision tree algorithms to Shannon entropy and provided a lower bound for computing any total function f, which is an interesting result in the computational complexity field.

C. Quantum nearest neighbor algorithm

In classical machine learning, the k-nearest neighbors algorithm is a non-parametric algorithm which can be applied to classification tasks. The original idea of it was introduced by Cover and Hart [141]. The basic idea for the k-nearest neighbors algorithm is straightforward as indicated by the name. Suppose we already have a set of training data. Then with this algorithm we are able to directly use the training data to predict some unseen data's labels without a model training procedure.

In the first step, we need to define a distance between the feature vectors. In the literature, the L_p distance is often chosen as a candidate:

$$L_p(\vec{x}_i, \vec{x}_j) = (\Sigma_{k=1}^{N} |\vec{x}_{i(k)} - \vec{x}_{j(k)}|^p)^{\frac{1}{p}}, \tag{33}$$

where N is the dimension of the feature space and $\vec{x}_{i(k)}$ represent the kth component of the vector $\vec{x}_i$. Moreover, the choice of distance measures can also be problem specific. For instance, hamming distance can be utilized to handle binary data. Then for each unseen input to be predicted, we search for the nearest k vectors in the training feature set according to the chosen distance measure. After obtaining these k vectors with their labels, the major label can be taken as the output prediction, which is often called the "majority voting rule". This process is sketched Fig.3(a). When applying this strategy, the computational cost plays an important role. If the training set is very large, simply calculating all the distances(between the vectors to be predicted and the vectors of the training set) and sorting them will be too computationally expensive. To solve this problem, sometimes we will need to use some special data structures such as k-d tree [142] to store the training data and optimize the process of finding the nearest points. In addition, there are different forms of nearest neighbor algorithms. As a popular variation, the nearest centroid classification is designed to compute the mean value of all samples in the training set labeled a_i to represent the centroid of a_i. That is, $C_{a_i} = \frac{1}{N_{a_i}} \Sigma_{[y(\vec{x}_{i(k)})=a_i]} \vec{x}_{i(k)}$, where N_{a_i} is number of samples labeled a_i and y is the function that outputs the corresponding label. Then the new data will

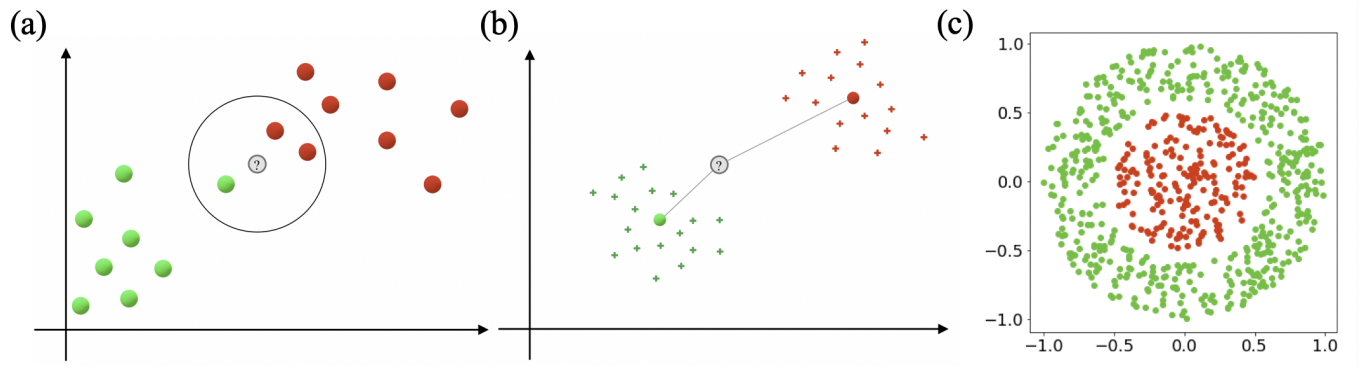

FIG. 3. (a) A schematic illustration of the k-nearest neighbors algorithm, where the value of k is 3 in this case. Here we have an unlabeled data point and two kinds of data points marked as red and green balls, respectively. The label of the majority of the nearest k samples, which is the label of the red balls, will be the output. (b) A schematic illustration of the nearest centroid algorithm. In this algorithm, the centroids of the different kinds of data points are calculated first. Then the unlabeled data will be assigned to the class corresponding to the closest centroid. (c) A situation that the nearest centroid algorithm may result in poor performance, where the centroid of the two sets lies in the same region.

be categorized to the class according to the nearest centroid, which is illustrated in Fig.3(b).

After a brief introduction to the classical algorithms, we now introduce two quantum algorithms. First, Lloyd *et al*. have proposed the quantum version of the nearest centroid algorithm [51]. In this task, the key step is to evaluate the distance between the vector $\vec{u}$ to be assigned and the centroids of different labels as shown in Fig.3(b). To accomplish this, firstly we write the set of samples of the same label v as $\{|\vec{v_j}||v_j\rangle\}$ to explicitly show the amplitude and the normalized state. With the help of quantum random access memory [131], we prepare the state to be $|\psi\rangle = (1/\sqrt{2})\left(|0\rangle|u\rangle + (1/\sqrt{M})\sum_{j=1}^{M}|j\rangle|v_j\rangle\right)$ in $O(\log(N))$ steps, where M is the size of the subset $\{|\vec{v_j}||v_j\rangle\}$ and N is the dimension of the vectors. Then a SWAP test [129] can be used to measure the success probability of the first qubit found in $|\phi\rangle = (1/\sqrt{Z})\left(|\vec{u}||0\rangle - (1/\sqrt{M})\sum_{j=1}^{M}|\vec{v_j}||j\rangle\right)$ where Z is a normalization factor. After a simple calculation, the success probability can be written as

$$P = \frac{1}{2Z}\left|\vec{u} - (1/M)\sum_{j}\vec{v}_j\right|^2. \tag{34}$$

In other words, we can get an estimation of the target distance by multiplying the success probability by $2Z$. The remaining task about constructing the state $|\phi\rangle$ can be accomplished by applying a time evolution of a specially designed Hamiltonian to an easy-to-construct state, which is also discussed in detail in [51]. Finally by comparing these distances, the classification decision can be made. The runtime of the process scales as $O(\log(MN))$, which shows an exponential speedup for both the size of the training set and the dimension of the vectors.

Nevertheless, the classification criterion used above is the distance between the input vector and different centroids. In some situations, this might result in a relatively poor performance: when the centroid of the subset lies in the space of other subsets, e.g. the example shown in Fig.3(c). To deal with this issue, Wiebe *et al*. have proposed a quantum nearest-neighbor algorithm to assign a vector $\vec{u}$ to a subset according to the label of $\vec{v_j}$ which satisfies the minimization of $|\vec{u} - \vec{v_j}|$ [52]. The main idea can be sketched in three steps. First, with the help of quantum random access memory, we can make a query to obtain the state $\frac{1}{\sqrt{M}}\sum_{j=1}^{M}|j\rangle\left(\sqrt{1-|\vec{v}_j-\vec{u}|}|0\rangle + \sqrt{|\vec{v}_j-\vec{u}|}|1\rangle\right)$ from $\frac{1}{\sqrt{M}}\sum_{j=1}^{M}|j\rangle|0\rangle$. Then the method of amplitude estimation is applied to encode the success probability into an ancillary register [143], which can be formulated as $\frac{1}{\sqrt{M}}\sum_{j=1}^{M}|j\rangle||\vec{v_j}-\vec{u}|\rangle$ with $||\vec{v_j}-\vec{u}|\rangle$ being the quantum state representing the distance $|\vec{v_j} - \vec{u}|$. In the last step, the Dürr Høyer minimization algorithm [144], which takes the Grover search algorithm [14] as a subroutine, can be applied to search for the closest element to $\vec{u}$ and assign $\vec{u}$ to the corresponding label. This process can be done with a quadratic speedup over the classical counterparts.

D. Variational quantum classifiers

Artificial neural networks are one of the most powerful approaches in the field of machine learning. Since the model of artificial neural networks proposed by McCulloch and Pitts and the concept of perceptron proposed by Rosenblatt in the last century [145–147], this direction has been actively studied and developed for a long time. In recent years, new breakthroughs of neural networks such as AlexNet [148] have shown us the magical power of feature extraction and brilliant outlooks for this field. Now artificial neural networks have already been widely applied in many scenarios from face recognition and natural language processing to robotics and automated driven cars, etc. [2, 3].

Partly inspired by biological neural networks, artificial neural networks usually consist of layers of connected artificial

neurons and activation functions. The parameters that can be optimized during the training procedure are contained in the transformations between the layers such as weights and biases. For an input sample, when it goes through the neural networks, each layer of the neural networks will make some transformations on it such as linear transformations and pooling operations. Then the activation function can be applied to the transformed results to gain more nonlinearity before passing it to the next layer.

For classification tasks, during the training process we have a neural network model where we can input the training data and receive the output result. Then a cost function is often needed to measure the distance between the current output and the training target, and a proper choice of cost functions will be helpful with the process of optimization. Some cost functions are commonly used for theoretical analysis and real-life applications, such as the mean square error and cross entropy. In addition, sometimes in order to improve the generalization behavior and to avoid overfitting, we will add a regularization term to the cost function, which is relevant to the complexity of the model. With properly chosen loss function L, the learning task reduces to an optimization problem [3]:

$$\min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f\left(\vec{x}_i\right)\right) + \lambda R(f), \tag{35}$$

where the first term represents the empirical risk and $R(f)$ denotes the complexity of the model with λ being a tunable hyperparameter characterizing the importance ratio between them. An appropriate choosing of this tunable hyperparameter may greatly improve the training accuracy, while preventing the model from being too complex. This is crucial for the generalization performance of the neural networks. From another point, if two models with different complexity achieve the same empirical error, the one with a simpler structure will be taken, which is in line with the principle of Occam's razor. Taking a step forward, with an already well-defined cost function, the training process can simply be a process of minimizing the cost function by tuning the weights and biases in the neural networks, during which common optimization techniques, such as gradient descent and backpropagation, can be exploited. After the training process, the trained classifiers can be used to make predictions for new unseen data samples. Compared with the training process, the computational cost of the predicting process is typically much lower and the trained model can then be encapsulated into some commercial products for real-life applications.

In recent years, quantum neural networks have gained much attention and rapid development [36, 42–45, 47, 66–71, 74, 75]. As a concept originally from the classical machine learning field, quantum neural networks usually show up in the form of variational quantum circuits. Thus in classification tasks, they are also called variational quantum classifiers. The variational parameters of these classifiers can be chosen flexibly, e.g. the rotation angle θ of the component $R_x(\theta)$ or some parameterized controlled rotations. When handling classification tasks, the parameters will be optimized during the training process to minimize a predefined loss function. Then this trained model can be utilized to classify some unseen data. Based on the above ideas, a variety of structures and protocols have already been proposed. To introduce this relatively more diverse subfield, we will split this section into several subsections to make it more organized. First, we will introduce the gradient descent methods used in the optimization process of variational quantum classifiers. Then some popular structures of variational quantum classifiers will be reviewed in the following up subsection. After this, we will turn to introduce some important phenomena and properties in variational quantum classifiers: the barren plateaus and vulnerability.

1. *Gradients evaluation*

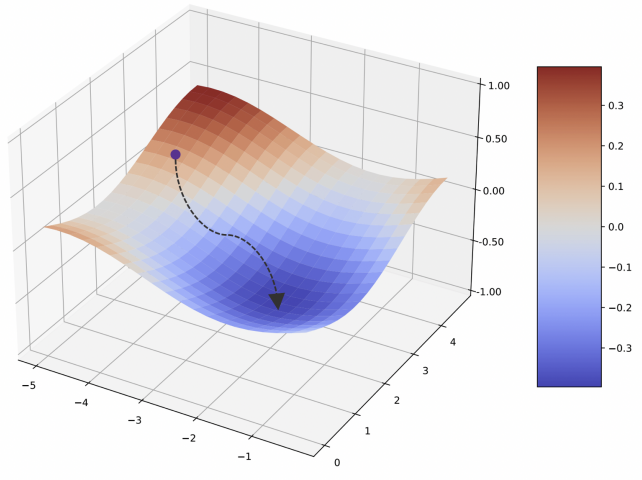

FIG. 4. A schematic illustration of the gradient descent process. At each point in the space of the cost function and parameters, the gradient descent strategy provides a direction to effectively reduce the value of the cost function.

When handling classification tasks using neural networks in the classical setting, gradients descent methods and their derivatives are the most frequently used tools during the training process. In general, for a given cost function L with parameters denoted collectively as Θ, the update at gradient descent step t can be expressed as

$$\Theta_{t+1} = \Theta_t - \gamma \nabla L\left(\Theta_t\right), \tag{36}$$

where γ is the learning rate. This is an intuitive idea to minimize the cost function, which is illustrated in Fig.4. For practical tasks, with appropriate learning rate and suitable optimizers like Adam [149], the training process can achieve desirable performance.

For this approach, obtaining the gradients usually takes the major computational resources. So for a variational quantum circuit, when trying to minimize a cost function, it is also wondered whether it is possible to efficiently obtain the gradients during the optimization process. Focusing on this question, some works have already been done with valuable ideas proposed. In Refs. [38, 41, 65], it is proposed that if the gate

with parameter θ is in the form $\mathcal{G}(\theta) = e^{-i\theta P_n}$ with P_n being an n-qubit operator in the Pauli group, then the derivative can be evaluated with $\frac{\partial\langle B\rangle}{\partial\theta} = \frac{\langle B\rangle^+ - \langle B\rangle^-}{2}$, where $\langle B\rangle^\pm$ means the expectation value of B with the original parameter θ being $\theta \pm \frac{\pi}{2}$. This is usually called the "parameter shift rule". Compared with the finite difference method expressed as $\frac{\partial\langle B\rangle}{\partial\theta} \approx \frac{\langle B\rangle_{\theta+\frac{\Delta\theta}{2}} - \langle B\rangle_{\theta-\frac{\Delta\theta}{2}}}{\Delta\theta}$, the gradient obtained through the parameter shift rule is exact with no discretization error. More importantly, the result of the exact gradient method is more robust to errors in the near-term quantum devices due to their expressions. Furthermore, this "parameter shift rule" can be generalized to compute arbitrary order derivatives by applying different shifts to the parameters as proposed in Ref. [54].

In Refs. [36, 37], the theory about gradient-based optimization above has been further developed. In Ref. [36], the gradient descent scheme is provided for a parameterized circuit with single-qubit gates expressed as

$$G(\alpha,\beta,\gamma) = \begin{pmatrix} e^{i\beta}\cos(\alpha) & e^{i\gamma}\sin(\alpha) \\ -e^{-i\gamma}\sin(\alpha) & e^{-i\beta}\cos(\alpha) \end{pmatrix} \quad (37)$$

and their corresponding controlled gates. After using analytic methods to express the derivative of the circuit unitary with respect to the parameters, the task of computing the partial derivatives can be reduced to measuring the overlap of two quantum states, which can be achieved with an inference circuit construction. In Ref. [37], the gradient-based approach has been further enriched from several aspects. Firstly, the "parameter shift rule" gets expanded to a more generalized setting. For a gate $\mathcal{G}(\theta) = e^{-i\theta G}$ where G is a Hermitian operator with two distinct eigenvalues $\pm r$, the target analytical gradient contains the terms $\frac{1}{\sqrt{2}}(I \pm ir^{-1}G)$ to be constructed. Observing the fact that $\mathcal{G}$ can be written in a more convenient form: $\mathcal{G}(\theta) = I\cos(r\theta) - ir^{-1}G\sin(r\theta)$, the above target can be achieved by setting θ as $\pm\frac{\pi}{4r}$. With this idea, it is easy to see that the circuit with one-qubit operators generated by Pauli matrices [38] is a special case for the generalized parameter shift rule. Noting that if the operator has more than two eigenvalues, some new strategies need to be developed. The Ref. [37] has also proposed a possible solution to deal with such a case. The key idea is to use an ancillary qubit and decompose the gradient $\partial_\theta\mathcal{G}$ into a linear combination of unitary matrices A_1 and A_2 [150]:

$$\partial_\theta\mathcal{G} = \frac{\Gamma}{2}\left[\left(A_1 + A_1^\dagger\right) + i\left(A_2 + A_2^\dagger\right)\right], \quad (38)$$

where Γ is a real number depending on the decomposition. The desired gradient can be evaluated from the probabilities and expectation values, as explained in detail in Ref. [37]. The parameter shift rule for Gaussian gates in the continuous variable setting has also been discussed in Ref. [37]. In addition, we mention that other methods [56–63, 72, 73], such as quantum natural gradient [56, 57] and L-BFGS method [58, 59], have been introduced to obtain the gradients as well in the literature. Different methods bears their pros and cons, and the choice of which one to use depends on the specific problem in practice.

2. *Different variational quantum classifiers*

In classical machine learning, different neural networks are used for different tasks. Various neural networks with different structures have been developed, including feedforward [80, 81], convolutional [82], recurrent [83, 84], and capsule neural networks [85, 86]. Similarly, in the emerging field of quantum machine learning, different variational quantum neural networks with distinct structures have also been introduced to tackle different problems[36, 42–45, 47]. In this subsection, we review some important advances about the structures and corresponding optimization strategies of variational quantum classifiers.

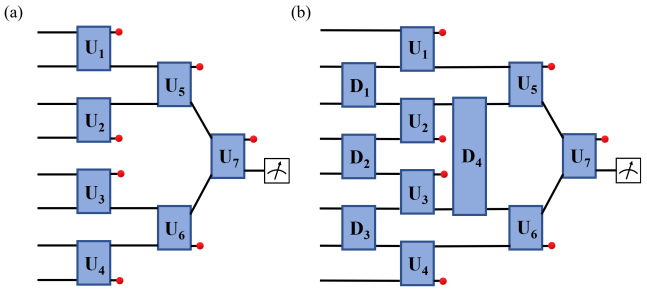

FIG. 5. (a) A schematic illustration of a TTN classifier, where a half of the qubits will be traced out after each layer. (b) A schematic illustration of a MERA classifier. The U_i and D_i are unitary blocks.

In Ref. [43], a parameterized tree-like quantum classifier has been proposed. The structures of the tree tensor networks (TTN) [151] and the multi-scale entanglement renormalization ansatz (MERA) [152, 153] are chosen as the layouts of the hierarchical circuits as shown in Fig.5. For an input state $|\psi\rangle$, the output is generally an expectation value expressed as

$$M_{\boldsymbol{\theta}}\left(\psi^d\right) = \left\langle \psi^d \left| \hat{U}_{\mathrm{QC}}^\dagger\left(\{U_i(\theta_i)\}\right)\hat{M}\hat{U}_{\mathrm{QC}}\left(\{U_i(\theta_i)\}\right)\right| \psi^d \right\rangle,$$

where $\hat{U}_{\mathrm{QC}}(\{U_i\})$ stands for the quantum circuit with parameterized unitaries $U(\theta)$ and $\hat{M}$ denotes the measurement on a single qubit. This value can be further used as a component in the optimization and classification process, e.g. as a component in the loss function: $L(\boldsymbol{\theta}) = \frac{1}{D}\sum_{d=1}^{D}\left(M_{\boldsymbol{\theta}}\left(\psi^d\right) - y^d\right)^2$, where D is the size of the training set and y^d is the label corresponding to the training data of index d. In this work, different datasets have been used for the training, including Iris dataset, handwritten digits, and quantum data. For the classical data, the qubit encoding method is chosen instead of amplitude encoding, which indicates that the classical data and the quantum state have the same dimension and the element-wise mapping can be written as $|\psi_n^d\rangle = \cos\left(x_n^d\right)|0\rangle + \sin\left(x_n^d\right)|1\rangle$. It should be noted that if the dimension of the classical data is beyond the reach of the quantum devices at hand, e.g. the 784 dimensional handwritten digits, dimension reduction methods like principal component analysis (PCA) could be utilized, which will inevitably cause some information loss. For the quantum

data, we typically assume that the data is generated from certain quantum physical process and can be input directly into the quantum classifiers. As shown in Ref. [43], the classifier can achieve decent performance for the above mentioned datasets with proper strategies. In addition, the performance of the classifier has been further benchmarked through comparisons between scenarios with and without beforehand PCA dimension reduction for the handwritten digits case, and scenarios with and without ancillary qubits for the quantum data case.

In addition to the classification model above, Schuld *et al.* have also proposed a parameterized circuit model with a different structure [36]. The circuit is composed of variational single-qubit gates and two-qubit controlled gates. Similar to the hierarchical quantum classifiers discussed above, here the training and classification processes make use of the expectation values of the output states as well. However, instead of only using this circuit to handle classification tasks on different datasets, this work provides several fresh angles to look into variational quantum classifiers. As already mentioned in the gradient evaluation part, a hybrid gradient descent scheme is given in this work to serve as the subrutine of the training. In the numerical simulations, five different datasets are chosen with feature dimensions varying from 13 to 256. The classification results are listed with benchmarking against six classical models, where a comparably good performance is achieved with fewer parameters. Considering the inevitability of noise in the quantum devices, this work has also numerically studied the effect of noise in the inputs and circuit parameters, and it turns out that this approach is reasonably resilient to noise.

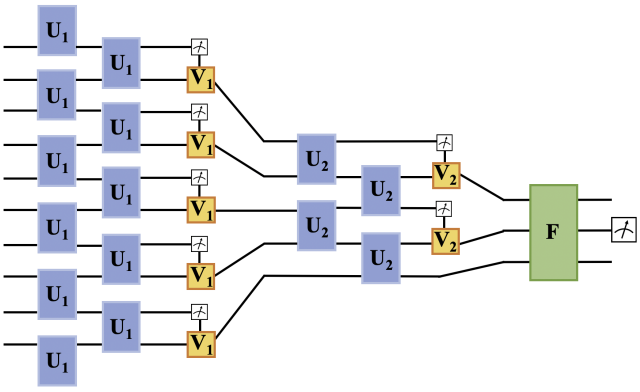

FIG. 6. A schematic illustration of the quantum convolutional neural networks, where the U_i, V_i, and F are unitary blocks and the V_i is controlled by some measurement outcomes.

Inspired by the classical convolutional neural networks, Cong, Choi, and Lukin introduced a quantum version of convolutional neural networks in Ref. [45]. In general, the classical convolutional neural networks consist of convolution layers, pooling layers and fully connected layers. This architecture has been translated into the quantum version as shown in Fig.6. The convolutional layer in the circuit is represented by the single quasilocal unitary (U_i). Moreover, the pooling procedure is achieved by measuring a fraction of the qubits and using the outcomes to determine unitary operations (V_j) applied to the nearby qubits. When the number of the remained qubits is reasonably small, a unitary F will serve as a fully connected layer followed by a final measurement. During the training process, the unitaries will be optimized with $O(log(N))$ parameters where N is the number of input qubits. As discussed in Ref. [45], the quantum convolutional neural networks can be applied to quantum phase recognition to detect a one-dimensional (1D) symmetry-protected topological phase. It turns out that the circuit can achieve a high accuracy when the depth is large enough.

For more diverse structures and protocols for variational quantum classifiers, there are many other works done from different perspectives [66–71, 74, 75]. For example, the work in Ref. [46] aims to train a deep quantum neural network to achieve a unitary operation. By replacing each target state with a state representing a "label", it can also be utilized as a classifier. Moreover, the variational quantum classifier can also be built in the continuous-variable architecture [42], and classical models such as convolutional, recurrent, and residual networks can all be embedded into this setting. The variational quantum classifiers can also be utilized in delegated learning protocols, e.g. in Ref. [64], the variational classifiers are designed to fit in the blind quantum computation structure [154] as a component in the multiparty federated learning scheme. Furthermore, there is increasingly more attention paid to the algorithms that search for appropriate structures of the variational quantum circuits [87–99], which shed light on maximizing the efficiency of quantum devices on the supervised learning tasks. In particular, a quantum neuroevolution algorithm that autonomously finds near-optimal quantum neural networks for different machine learning tasks has been introduced in Ref. [92]. Moreover, in Ref. [89] the authors have also introduced a differentiable quantum architecture search algorithm, which is capable of automated quantum circuit designs in an end-to-end differentiable fashion. This field is growing rapidly. With the development of quantum devices, the applications based on these models may bring some practical quantum advantages.

3. *Barren plateaus*

So far a variety of variational quantum classifiers have been introduced above. When a variational quantum circuit is being trained, the update of the parameters in the circuit is expected to efficiently optimize the model. However, in recent years, several works have shown the existence of gradient vanishing phenomena in the training process of variational quantum circuits. This is the so called barren plateau problem, which is illustrated pictorially in Fig.7. In this subsection, we will review some of the recent progresses on barren plateaus for quantum classifiers.

In Ref. [101], McClean *et al.* have shown that, for a wide class of parameterized quantum circuits, the probability that the gradient with respect to any variational parameters is exponentially suppressed as the system size increases. In other

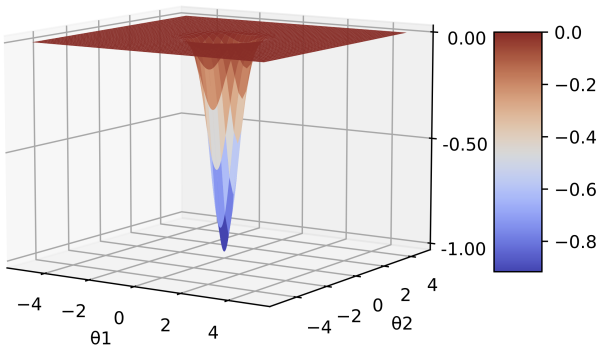

FIG. 7. A schematic illustration of the barren plateaus in the optimization process, where the landscape of the loss function is flat for nearly all the parameter space and the gradient descent methods will fail due to the exponentially vanishing gradients.

words, the loss function landscapes of these variational quantum circuits are mostly flat and barren plateaus prevail in the training process. To illustrate the essential ideas, we consider the randomly parameterized quantum circuits

$$U(\boldsymbol{\theta}) = U\left(\theta_1, \ldots, \theta_L\right) = \prod_{l=1}^{L} U_l\left(\theta_l\right) W_l, \tag{39}$$

where $U_l\left(\theta_l\right) = \exp\left(-i\theta_l V_l\right)$ with V_l being a Hermitian operator, and W_l is a given unitary without variational parameters. The expectation value of the output state over a Hermitian operator H is $E(\boldsymbol{\theta}) = \left\langle 0 \left| U(\boldsymbol{\theta})^\dagger H U(\boldsymbol{\theta}) \right| 0 \right\rangle$, and the derivative of $E(\boldsymbol{\theta})$ with respect to θ_k can be expressed as

$$\partial_k E \equiv \frac{\partial E(\boldsymbol{\theta})}{\partial \theta_k} = i \left\langle 0 \left| U_-^\dagger \left[V_k, U_+^\dagger H U_+\right] U_- \right| 0 \right\rangle, \tag{40}$$

where $U_- \equiv \prod_{l=0}^{k-1} U_l\left(\theta_l\right) W_l$, $U_+ \equiv \prod_{l=k}^{L} U_l\left(\theta_l\right) W_l$. With tools of unitary t-designs [155–157], the authors prove that if either U_+ or U_- is 1-design, the expectation value of the gradient with respect to the parameter θ_k is zero, i.e., $\langle \partial_k E \rangle = 0$. Moreover, if either U_+ or U_- is 2-design, then an exponential decay of the gradient variance will occur as the system size increases. These results explicitly show the exponential concentration of the gradients to zero for the randomly parameterized quantum circuits, which may pose a notable challenge for training quantum classifiers. From a high-level theoretical perspective, the barren plateaus proved here have a geometric understanding in terms of concentration of measure for high-dimensional spaces [158–160].

The presence and absence of barren plateaus depends crucially on the depths of the variational quantum circuits and the properties of the loss functions used. In Ref. [102], it is proved that if a global loss function is used, then barren plateaus typically appear even for circuits with small constant depths. Whereas, for local loss functions the gradient vanishes at most polynomially as long as the circuit depth is bounded by $O(\log n)$. To obtain an intuitive understanding of these results, the authors considered a toy problem about state preparation with the target state being $|\mathbf{0}\rangle$. They started with a tensor-product varational ansatz $V(\boldsymbol{\theta}) = \otimes_{j=1}^{n} e^{-i\theta^j \sigma_x^{(j)}/2}$, with the goal of finding out the angles θ^j such that $V(\boldsymbol{\theta})|\mathbf{0}\rangle = |\mathbf{0}\rangle$. Assume the global cost function is chosen as

$$C_G = \mathrm{Tr}\left[O_G V(\boldsymbol{\theta}) \left|\psi_0\right\rangle \left\langle\psi_0\right| V(\boldsymbol{\theta})^\dagger\right], \tag{41}$$

where $O_G = I - |\mathbf{0}\rangle\langle\mathbf{0}|$. This cost function can be simplified as $C_G = 1 - \prod_{j=1}^{n} \cos^2 \frac{\theta^j}{2}$. Direct calculations yield $\left\langle \frac{\partial C_G}{\partial \theta^j} \right\rangle = 0$ and $\mathrm{Var}\left[\frac{\partial C_G}{\partial \theta^j}\right] = \frac{1}{8}\left(\frac{3}{8}\right)^{n-1}$, which shows that the gradient concentrates exponentially around zero and a barren plateau appears independent of the depth of the circuits. In contrast, if we choose a local cost function

$$C_L = \mathrm{Tr}\left[O_L V(\boldsymbol{\theta}) |\mathbf{0}\rangle\langle\mathbf{0}| V(\boldsymbol{\theta})^\dagger\right], \tag{42}$$

where $O_L = I - \frac{1}{n}\sum_{j=1}^{n} |0\rangle\langle 0|_j \otimes I_{\bar{j}}$ ($I_{\bar{j}}$ denotes the identity operator on all qubits except the one with index j), then direct calculations lead to $C_L = 1 - \frac{1}{n}\sum_{j=1}^{n} \cos^2 \frac{\theta^j}{2}$. In this case, it is straightforward to obtain $\left\langle \frac{\partial C_L}{\partial \theta^j} \right\rangle = 0$ and $\mathrm{Var}\left[\frac{\partial C_L}{\partial \theta^j}\right] = \frac{1}{8n^2}$, which indicates that the gradient vanishes polynomially as the system size increases and hence exhibits no barren plateau.

In addition to the above two studies, we also mention that barren plateaus for variational quantum circuits have also been investigated from other aspects in the literature [103–118]. For instance, in Ref. [103], the absence of barren plateaus in quantum convolutional neural networks has been rigorously proved. This guarantees the trainability of randomly initialized quantum convolutional neural networks and singles out them as being efficiently trainable unlike many other quantum neural network models. Besides, noise and entanglement induced barren plateaus have also been investigated in Ref. [104] and Refs. [106, 107], respectively. We also remark that in practical applications the barren plateau problem might be mitigated through a variety of strategies, such as proper initialization [112], pre-training [161, 162], and judiciously designed quantum circuit structures [89, 92].

4. *Vulnerability*

The vulnerability of a machine learning model is often relevant to the performance of the model against adversarial attacks. For a specific example, given an input labeled "panda" and a fixed classification model, the unperturbed input will be categorized to the label "panda" with high probability. The adversarial attack aims to generate an imperceptible perturbation on the original input to deceive the classifier, i.e. to efficiently reduce the probability of assigning the input to "panda" or to efficiently increase the probability of assigning the input to some wrong labels like "gibbon". This property of machine learning models has already been widely investigated in recent years [163–176]. Similarly in the quantum domain, when a quantum classifier is given, it is important to characterize the vulnerability of the model in consideration of adversarial perturbations. In this subsection, we review recent advances about the vulnerability of quantum classifiers.

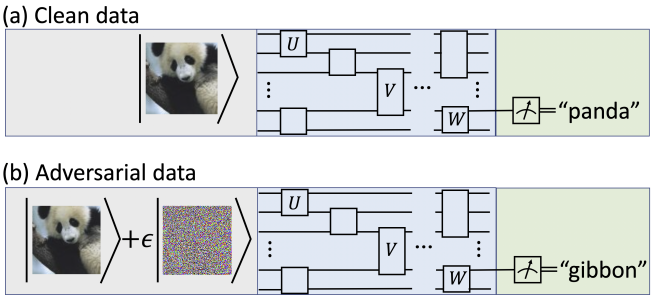

FIG. 8. A schematic illustration of quantum adversarial machine learning. (a) The quantum classifier can correctly assign the label "panda" to the figure. (b) By adding a tiny amount carefully-crafted perturbation, the quantum classifier will assign the label "gibbon" to the figure incorrectly.

In Ref. [39], Lu, Duan, and Deng studied the vulnerability of quantum machine learning systems in different adversarial scenarios. They found that, similar to classical neural network classifiers, quantum classifiers are likewise vulnerable to crafted adversarial examples, regardless of whether the input data is classical or quantum. In particular, a quantum classifier that achieves nearly the state-of-the-art accuracy can be conclusively deceived by adversarial examples, which are obtained by adding imperceptible perturbations to the original legitimate samples. The essential idea can be illustrated pictorially in Fig.8. More concretely, suppose we have a trained model with optimized parameters collectively denoted as Θ^*, the loss function reads $L(h(|\psi\rangle_{\rm in};\Theta^*),\mathbf{a})$, where $h(|\psi\rangle_{\rm in};\Theta^*)$ denotes the hypothesis function determined by the variational quantum classifier with parameters fixed to be Θ^* and $\mathbf{a}$ is the label corresponding to the input state $|\psi\rangle$. An imperceptible perturbation is used as an adversarial attack and implemented as a perturbation operator U_δ acting on $|\psi\rangle_{\rm in}$. Here the perturbation operator U_δ can be chosen from a set Δ of all unitaries close to the identity operator. For an untargeted attack, the goal is to minimize the probability of assigning the right label $\mathbf{a}$ to the state $|\psi\rangle_{\rm in}$ without caring about which label will be the output. So finding out the adversarial perturbations becomes searching a U_δ within Δ so as to maximize the loss function:

$$U_\delta \equiv \underset{U_\delta\in\Delta}{\operatorname{argmax}}\, L\left(h\left(U_\delta|\psi\rangle_{\rm in};\Theta^*\right),\mathbf{a}\right). \tag{43}$$

For a targeted attack, the goal is to maximize the probability to assign a target label $\mathbf{a}^{(t)}$ to the state $|\psi\rangle_{\rm in}$ where $\mathbf{a}\neq\mathbf{a}^{(t)}$. In this case, the task can be formalized as finding out the $U_\delta^{(t)}$ that minimizes the targeted loss function:

$$U_\delta^{(\rm t)} \equiv \underset{U_\delta^{(\rm t)}\in\Delta}{\operatorname{argmin}}\, L\left(h\left(U_\delta^{(\rm t)}|\psi\rangle_{\rm in};\Theta^*\right),\mathbf{a}^{(\rm t)}\right). \tag{44}$$

To demonstrate how to obtain adversarial examples in practice, the authors carried out extensive numerical simulations for several concrete examples covering different scenarios with diverse types of data (such as, handwritten digit images in the dataset MNIST, simulated time-of-flight images in a cold-atom experiment, and quantum data from a one-dimensional transverse field Ising model) and different attack strategies. In addition, they also discussed possible defense strategies to enhance the robustness of quantum classifiers against adversarial perturbations.

On a more analytical level, Liu and Wittek proved rigorously that the amount of perturbation needed for an adversary to induce a misclassification for quantum classifiers scales inversely with dimensionality [119]. This vulnerability property is a fundamental feature for quantum classifiers independent of the details of the classification protocol, which originates from the concentration of measure phenomenon of high-dimensional Hilbert space. Furthermore, in Ref. [123] Gong and Deng studied the universality of adversarial examples and perturbations for quantum classifiers. They proved that there exist universal adversarial examples that can fool a set of different quantum classifiers: for a set of k classifiers with each input data sample encoded by n qubits, an $O(\frac{\ln k}{2^n})$ increase of the perturbation strength is sufficient to warrant a moderate universal adversarial risk. In addition, they also proved, based on the quantum no free lunch theorem [177, 178], that the universal adversarial risk is bounded from both below and above and approaches unit exponentially fast as as the size of the classifier increases.

We also mention that there are other works studying quantum adversarial machine learning from different aspects [120–122]. For example, the Ref. [122] explored quantum noises to develop a defense strategy against adversarial attacks. Furthermore, in Ref. [121] a fundamental link between binary quantum hypothesis testing and probably robust quantum classification has been established, which provides a tight robustness condition for the amount of noise (either natural or adversarial) a classifier can tolerate.

III. EXPERIMENTAL PROGRESS

A. Quantum support vector machine and kernel methods

In the theory section, we have introduced the quantum support vector machine and some other quantum kernel methods. The extensions from classical to quantum domain exhibit some attractive properties including potential exponential speedups for certain tasks. In parallel to the theoretical progresses, there are also noteworthy experimental advances and some proof-of-principle physical implementations of these classification models have already been demonstrated in laboratory [23, 29–32]. Here, we review recent experimental progresses on quantum support vector machine and other quantum classification models based on kernel methods.

The first experimental demonstration of quantum support vector machines has been reported in Ref. [30]. In this work, the classification of handwritten digits has been experimentally implemented using a small-size quantum circuit. Due to limited available qubits, the classification task considered in this experiment was focused on classifying two handwritten digits "6" and "9". Moreover, a preprocessing of the images was also carried out to reduce the number of features to 2:

the vertical ratio and the horizontal ratio according to the pixels in the left (upper) half over the right (lower) half. The preprocessed images corresponding to digits "6" and "9" are defined to be in the positive and negative classes, respectively. After these preparations, the quantum support vector machine scheme for this classification task can be delegated to a 4-qubit nuclear magnetic resonance quantum processor. As shown in the paper, the experimental predictions are consistent with the true labels of the handwritten digits.

For the experimental advances in quantum classifiers utilizing kernel methods, several works have been done in recent years [23, 29, 31, 32]. For instance, the distance-based quantum classifier mentioned above has been experimentally demonstrated on the superconducting platform of IBM Quantum Experience [31]. This experiment achieves the classification of the Iris dataset [179] using four non-error-corrected superconducting qubits. It is worthwhile to mention that, although the simulation results are close to the theoretical predictions, the experimental results deviate far from them due to the lack of error correction.

In addition, the quantum devices can be used to estimate the kernel functions and the results can be fed to classical computers to do the rest of the training [23, 24]. This idea is attractive especially for the stage of noisy intermediate-scale quantum (NISQ) devices, where the quantum technologies are not advanced enough and the cooperation with the classical computers might be helpful. This hybrid approach has been experimentally demonstrated on both superconducting [23] and optical [29] platforms. The results show that classical computers can indeed achieve decent performance by utilizing the kernel matrices evaluated from quantum devices.

B. Quantum nearest neighbor algorithm

As mentioned in the above sections, with quantum random access memory, the quantum nearest neighbor algorithm has the potential to offer an exponential speedup. Moreover, the first proof-of-principle demonstration has also been implemented [55]. This experiment is conducted on the optical platform to assign different entangled states to their nearest clusters. Here, the two clusters A and B are represented by reference vectors $\vec{v}_A$ and $\vec{v}_B$, respectively. For the job of assigning a new vector $\vec{u}$, it can be transformed into comparing the distance between $\vec{u}$ and the reference vectors: $D_A = |\vec{u} - \vec{v}_A|$, and $D_B = |\vec{u} - \vec{v}_B|$ and assigning it to the cluster with a smaller distance.

To achieve this goal, the single photons are used as qubits and $|0\rangle$ and $|1\rangle$ are encoded with their horizontal and vertical polarization. A key step is to create the entanglement resource states for data encoding. Firstly, the polarization-entangled photon pairs $(|0\rangle_{\rm anc}|0\rangle_{\rm vec} + |1\rangle_{\rm anc}|1\rangle_{\rm vec})/\sqrt{2}$, where one photon is used as an ancillary qubit, can be generated through spontaneous parametric down-conversion [180]. Then the four-photon resource states can be generated using two entangled photon pairs with a post-selection scheme, which results in the four-qubit Greenberger-Horne-Zeilinger entangled state [181]:

$$\frac{1}{\sqrt{2}}\left(|0\rangle_{\rm anc}|000\rangle_{\rm vec} + |1\rangle_{\rm anc}|111\rangle_{\rm vec}\right). \tag{45}$$

The three-qubit state can be generated straightforwardly by a projection.

With these states prepared, the data encoding can be accomplished by firstly sending the single photons through a polarizing beam splitter and splitting them into two spatial modes. Then the controlled unitary gates can be applied to these two spatial modes, respectively [182]. After recombining these two modes on a nonpolarizing beam splitter, the final state $\left(|0\rangle|u_1\rangle_{\rm new} + |1\rangle|v_1\rangle_{\rm ref}\right)/\sqrt{2}$ can be generated. With all the preparations, the classification scheme can be deployed on the optical platform. It turns out to achieve decent classification accuracy of the two-, four-, and eight-dimensional vectors that are encoded in the two-, three-, and four-qubit entanglement states. The misclassified cases partly come from "boundary conditions", where the error is comparable to the distance $|D_A - D_B|$ and the performance of the distance evaluation which is affected by the fidelity of the entangled states [55].

C. Variational quantum classifiers

Variational quantum classifiers have drawn broad interest over recent years. However, due to the system size limitation of current quantum devices, most of the classifiers reviewed in the above sections are not readily feasible to be implemented with a large number of qubits in the laboratory at the current stage. To connect the theories with the experimental implementations, some proof-of-principle experiments have been carried out. In this subsection, we will introduce several experimental advances of the classifiers composed of variational quantum circuits [23, 43].

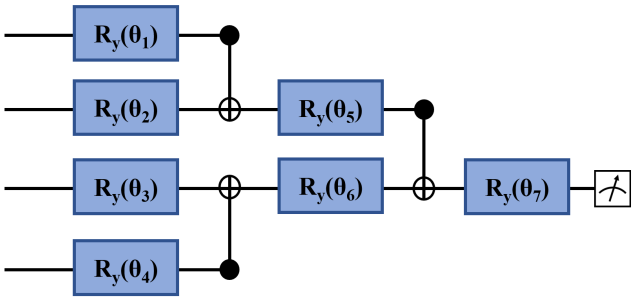

FIG. 9. The structure of a proof-of-principle TTN classifier.

In the above sections, we have introduced the structures of hierarchical quantum classifiers including tree tensor networks and multi-scale entanglement renormalization ansatz [43]. In Ref. [43], a proof-of-principle experiment has been demonstrated to classify two classes of the Iris data, where a tree tensor network quantum classifier is deployed on the ibmqx4 quantum computer. The circuit is designed as shown in Fig.9 and it is trained classically first and then deployed

on the quantum device, which is finally able to implement the classification of the test data with 100% accuracy.

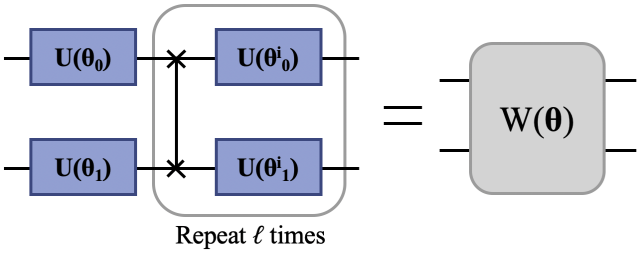

FIG. 10. The structure of the variational quantum circuit for the training task. The entangling gates used in this protocol are controlled-Z gates and the local gates belong to SU(2).

In Ref. [23], a variational quantum classifier has been experimentally implemented on the superconducting platform to classify artificially generated data. To start with, the training data $\boldsymbol{x}$ is encoded in a quantum state $|\Phi(\boldsymbol{x})\rangle$ by applying an operator $\mathcal{U}_{\Phi(\boldsymbol{x})} = U_{\Phi(\boldsymbol{x})}H^{\otimes n}U_{\Phi(\boldsymbol{x})}H^{\otimes n}$ to $|0\rangle^n$ where the component $U_{\Phi(\boldsymbol{x})}$ can be expressed as [23]:

$$U_{\Phi(\boldsymbol{x})} = \exp\left(i\sum_{S\subseteq[n]}\phi_S(\boldsymbol{x})\prod_{i\in S}Z_i\right), \tag{46}$$

with $\phi_{\{i\}}(\boldsymbol{x}) = x_i$ and $\phi_{\{1,2\}}(\boldsymbol{x}) = (\pi - x_1)(\pi - x_2)$. Then to assign the training data into different classes, a random unitary $V \in SU(4)$ and the observable $\boldsymbol{f}$=Z_1Z_2 are chosen. The label of $\boldsymbol{x}$ will be assigned to $+1$ if $\langle\Phi(\boldsymbol{x})|V^\dagger \boldsymbol{f} V|\Phi(\boldsymbol{x})\rangle \geq \Delta$ and to -1 if $\langle\Phi(\boldsymbol{x})|V^\dagger \boldsymbol{f} V|\Phi(\boldsymbol{x})\rangle \leq -\Delta$ with Δ being 0.3. After these data preparations, the short-depth variational circuit $W(\theta)$ designed as shown in Fig.10 can be applied to the data states and takes the role of assigning the data to different categories. For the basic settings of the training process, the binary measurement $\{M_y\}$ in the Z-basis is performed on the state $W(\theta)\mathcal{U}_{\Phi(\boldsymbol{x})}|0\rangle^n$ with success probabilities $p_y(\boldsymbol{x}) = \langle\Phi(\boldsymbol{x})|W^\dagger(\boldsymbol{\theta})M_yW(\boldsymbol{\theta})|\Phi(\boldsymbol{x})\rangle$. In the physical experiments, these probabilities can be evaluated by repeating the measurement and taking the empirical distribution $\hat{p}_y$. Then the data $\boldsymbol{x}$ will be assigned to class y if $\hat{p}_y(x) > \hat{p}_{-y}(x) - yb$, where the bias parameter $b \in [-1, 1]$ can also be optimized. During the training process, the cost function is defined as the error probability and it is optimized with Spall's simultaneous perturbation stochastic approximation algorithm [183, 184]. After the training procedure, the parameters of the circuit are fixed and can be used for classifying some unseen data. With these settings, the experiments with error mitigation techniques and different circuit depths have been carried out, where a 100% success rate can be achieved with the depth four circuit.

IV. CONCLUSION AND OUTLOOKS

Over the past two decades, the world has witnessed revolutionary development in both quantum computing and machine learning. As an appealing interdisciplinary application, the quantum classifiers have attracted a wide range of interest. The models of quantum classifiers have been developed from a number of aspects. Some quantum classifiers can be viewed as the quantum extensions of some classical classification algorithms. What is more, these quantum classifiers may have the potential to exhibit advantages over their classical counterparts. In addition, there are other quantum classifiers that utilize the structures drawing inspiration from physical concepts, e.g. the tree tensor network classifiers, the multi-scale entanglement renormalization ansatz classifiers, and the multi-level quantum system classifiers. These explorations might be helpful to find models that are feasible to tackle certain classification problems with some inherent physical structures.

Yet, despite the rapid development, there are crucial challenges for quantum classifiers that stand in the way to future practical applications. First, for the quantum devices, the quantum random access memory is a key component for a number of algorithms to demonstrate the potential advantages, while it is still far from implementing QRAM with a desirable scale at the current stage. Second, since the noise in the quantum devices seems inevitable, increasing the fidelity of the quantum operations and developing better error correction techniques are of crucial importance. Third, for the variational quantum classifiers, there are already some works trying to measure their representation power [185–190]. To find whether there is a separation between these models and their classical counterparts for potential future applications, further studies are highly desirable. In addition, the barren plateau phenomena and the vulnerability of quantum classifiers to adversarial attacks mentioned in this review also present the potential problems against the future applications, though there are already a variety of works trying to formalize and handle them. For the future development of this field, it is important to further develop quantum classification theories and models to find more potential applications. With the help of the softwares used for simulating these quantum models [191–204], it gets easier to quantify the performance of these models. Moreover, the size of the quantum platforms, such as the superconducting platforms and the ion trap systems, are expected to grow fast to meet the requirement of the theories of the various quantum classifiers. The combination of the advanced theories of quantum classifiers and the moderate-size quantum computers may demonstrate useful advantages in practical applications in the near future.

From the perspective of computational complexity, in Ref. [205] Bittel and Kliesch proved that the classical optimization problems for variational quantum algorithms are NP-hard, even for classically tractable systems which are composed of only logarithmically many qubits or free fermions. This work showed the intrinsic hardness for the classical optimization and the training landscape may have many local minima far from optimal, which presents a severe challenge for variational quantum classifiers. In addition, the quantum advantage in machine learning can also be viewed in terms of the number of times to access a quantum process $\mathcal{E}$ in the classical or quantum setting. In Ref. [206], Huang, Kueng and Preskill proved that for any input distribution, a classi-

cal machine learning model can achieve accurate predictions *on average* by accessing $\mathcal{E}$ with times comparable to the optimal quantum machine learning model. By comparison, an exponential quantum advantage is possible for achieving an accurate prediction *on all inputs*. For instance, predicting the expectations of all Pauli observables in an n-qubit system ρ requires $O(n)$ copies of ρ in a quantum machine learning model, exponentially less than $2^{\Omega(n)}$ copies in a classical one. Moreover, the sample complexity has also been investigated in Ref. [207, 208]. These works provide helpful insights for understanding quantum advantages in machine learning tasks.

Without a doubt, it is hard to say when the first quantum classifier will be commercially available. Harder even is to predict what the special advantage the first commercial quantum classifier will bring us in real life. This emergent research direction is largely still in its infancy, with many important issues remaining barely explored. Yet, one thing is for sure: the combination of supervised learning and quantum physics is a win-win cooperation that has great potential to revolutionize many aspects of our modern world.

Weikang Li would like to thank Zhide Lu and Li-Wei Yu for helpful discussions and advices about the manuscript. This work is supported by the start-up fund from Tsinghua University (Grant. No. 53330300320), the National Natural Science Foundation of China (Grant. No. 12075128), and the Shanghai Qi Zhi Institute.

[1] S. D. Sarma, D.-L. Deng, and L.-M. Duan, Machine learning meets quantum physics, Phys. Today **72**, 48 (2019).
[2] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature **521**, 436 (2015).
[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[4] M. I. Jordan and T. M. Mitchell, Machine learning: Trends, perspectives, and prospects, Science **349**, 255 (2015).
[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, Mastering the game of Go with deep neural networks and tree search, Nature **529**, 484 (2016).
[6] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, Mastering the game of Go without human knowledge, Nature **550**, 354 (2017).
[7] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature **542**, 115 (2017).
[8] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, Improved protein structure prediction using potentials from deep learning, Nature **577**, 706 (2020).
[9] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91**, 045002 (2019).
[10] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: A review of recent progress, Rep. Prog. Phys. **81**, 074001 (2018).
[11] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).
[12] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, P. Hu, X.-Y. Yang, W.-J. Zhang, H. Li, Y. Li, X. Jiang, L. Gan, G. Yang, L. You, Z. Wang, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan, Quantum computational advantage using photons, Science **370**, 1460 (2020).
[13] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, M. Gong, C. Guo, C. Guo, S. Guo, L. Han, L. Hong, H.-L. Huang, Y.-H. Huo, L. Li, N. Li, S. Li, Y. Li, F. Liang, C. Lin, J. Lin, H. Qian, D. Qiao, H. Rong, H. Su, L. Sun, L. Wang, S. Wang, D. Wu, Y. Xu, K. Yan, W. Yang, Y. Yang, Y. Ye, J. Yin, C. Ying, J. Yu, C. Zha, C. Zhang, H. Zhang, K. Zhang, Y. Zhang, H. Zhao, Y. Zhao, L. Zhou, Q. Zhu, C.-Y. Lu, C.-Z. Peng, X. Zhu, and J.-W. Pan, Strong quantum computational advantage using a superconducting quantum processor, arXiv:2106.14734 (2021).
[14] L. K. Grover, Quantum Mechanics Helps in Searching for a Needle in a Haystack, Phys. Rev. Lett. **79**, 325 (1997).
[15] P. W. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. **26**, 1484 (1997).
[16] A. M. Childs and W. van Dam, Quantum algorithms for algebraic problems, Rev. Mod. Phys. **82**, 1 (2010).
[17] G. L. Long, Grover algorithm with zero theoretical failure rate, Phys. Rev. A **64**, 022307 (2001).
[18] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).
[19] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. **10**, 631 (2014).
[20] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. **113**, 130503 (2014).
[21] V. Dunjko, J. M. Taylor, and H. J. Briegel, Quantum-Enhanced Machine Learning, Phys. Rev. Lett. **117**, 130501 (2016).

[22] A. Montanaro, Quantum algorithms: An overview, npj Quantum Inf **2**, 1 (2016).

[23] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature **567**, 209 (2019).

[24] M. Schuld and N. Killoran, Quantum machine learning in feature hilbert spaces, Phys. Rev. Lett. **122**, 040504 (2019).

[25] X. Gao, Z.-Y. Zhang, and L.-M. Duan, A quantum machine learning algorithm based on generative models, Sci. Adv. **4**, eaat9004 (2018).

[26] S. Lloyd and C. Weedbrook, Quantum generative adversarial learning, Phys. Rev. Lett. **121**, 040502 (2018).

[27] L. Hu, S.-H. Wu, W. Cai, Y. Ma, X. Mu, Y. Xu, H. Wang, Y. Song, D.-L. Deng, C.-L. Zou, and L. Sun, Quantum generative adversarial learning in a superconducting quantum circuit, Sci. Adv. **5**, eaav2761 (2019).

[28] J. M. Martyn, Z. M. Rossi, A. K. Tan, and I. L. Chuang, A Grand Unification of Quantum Algorithms, arXiv:2105.02859 (2021).

[29] K. Bartkiewicz, C. Gneiting, A. Černoch, K. Jiráková, K. Lemr, and F. Nori, Experimental kernel-based quantum machine learning in finite feature space, Sci. Rep. **10**, 1 (2020).

[30] Z. Li, X. Liu, N. Xu, and J. Du, Experimental realization of a quantum support vector machine, Phys. Rev. Lett. **114**, 140504 (2015).

[31] M. Schuld, M. Fingerhuth, and F. Petruccione, Implementing a distance-based classifier with a quantum interference circuit, Europhys. Lett. **119**, 60002 (2017).

[32] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, Quantum classifier with tailored quantum kernel, npj Quantum Inf. **6**, 1 (2020).

[33] D. Zhu, N. M. Linke, M. Benedetti, K. A. Landsman, N. H. Nguyen, C. H. Alderete, A. Perdomo-Ortiz, N. Korda, A. Garfoot, C. Brecque, L. Egan, O. Perdomo, and C. Monroe, Training of quantum circuits on a hybrid quantum computer, Sci. Adv. **5**, eaaw9918 (2019).

[34] J. Zhao, Y.-H. Zhang, C.-P. Shao, Y.-C. Wu, G.-C. Guo, and G.-P. Guo, Building quantum neural networks based on a swap test, Phys. Rev. A **100**, 012334 (2019).

[35] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, An artificial neuron implemented on an actual quantum processor, npj Quantum Inf. **5**, 1 (2019).

[36] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, Phys. Rev. A **101**, 032308 (2020).

[37] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, Phys. Rev. A **99**, 032331 (2019).

[38] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A **98**, 032309 (2018).

[39] S. Lu, L.-M. Duan, and D.-L. Deng, Quantum adversarial machine learning, Phys. Rev. Research **2**, 033212 (2020).

[40] J.-G. Liu and L. Wang, Differentiable learning of quantum circuit born machines, Phys. Rev. A **98**, 062324 (2018).

[41] J. Li, X. Yang, X. Peng, and C.-P. Sun, Hybrid quantum-classical approach to quantum optimal control, Phys. Rev. Lett. **118**, 150503 (2017).

[42] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, Continuous-variable quantum neural networks, Phys. Rev. Research **1**, 033063 (2019).

[43] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, Hierarchical quantum classifiers, npj Quantum Inf. **4**, 1 (2018).

[44] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, arXiv:1802.06002 (2018).

[45] I. Cong, S. Choi, and M. D. Lukin, Quantum convolutional neural networks, Nat. Phys. **15**, 1273 (2019).

[46] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, Training deep quantum neural networks, Nat. Commun. **11**, 1 (2020).

[47] S. Adhikary, S. Dangwal, and D. Bhowmik, Supervised learning with a quantum classifier using multi-level systems, Quantum Inf. Process. **19**, 89 (2020).

[48] K. Wang, L. Xiao, W. Yi, S.-J. Ran, and P. Xue, Quantum image classifier with single photons, arXiv:2003.08551 (2020).

[49] S. Y.-C. Chen, C.-M. Huang, C.-W. Hsing, and Y.-J. Kao, Hybrid quantum-classical classifier based on tensor network and variational quantum circuit, arXiv:2011.14651 (2020).

[50] D. Türkpençe, T. Ç. Akıncı, and S. Şeker, A steady state quantum classifier, Phys. Lett. A **383**, 1410 (2019).

[51] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning, arXiv:1307.0411 (2013).

[52] N. Wiebe, A. Kapoor, and K. Svore, Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning, arXiv:1401.2142 (2014).

[53] S. Lu and S. L. Braunstein, Quantum decision tree classifier, Quantum Inf. Process. **13**, 757 (2014).

[54] A. Mari, T. R. Bromley, and N. Killoran, Estimating the gradient and higher-order derivatives on quantum hardware, Phys. Rev. A **103**, 012405 (2021).

[55] X.-D. Cai, D. Wu, Z.-E. Su, M.-C. Chen, X.-L. Wang, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan, Entanglement-based machine learning on a quantum computer, Phys. Rev. Research **114**, 110504 (2015).

[56] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum Natural Gradient, Quantum **4**, 269 (2020).

[57] B. Koczor and S. C. Benjamin, Quantum natural gradient generalised to non-unitary circuits, arXiv:1912.08660 (2019).

[58] D. C. Liu and J. Nocedal, On the limited memory BFGS method for large scale optimization, Mathematical Programming **45**, 503 (1989).

[59] W. Lavrijsen, A. Tudor, J. Müller, C. Iancu, and W. de Jong, Classical Optimizers for Noisy Intermediate-Scale Quantum Devices, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2020) pp. 267–277.

[60] T. Haug and M. S. Kim, Natural parameterized quantum circuit, arXiv:2107.14063 (2021).

[61] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, General parameter-shift rules for quantum gradients, arXiv:2107.12390 (2021).

[62] R. Sweke, F. Wilde, J. Meyer, M. Schuld, P. K. Faehrmann, B. Meynard-Piganeau, and J. Eisert, Stochastic gradient descent for hybrid quantum-classical optimization, Quantum **4**, 314 (2020).

[63] B. Koczor and S. C. Benjamin, Quantum analytic descent, arXiv:2008.13774 (2020).

[64] W. Li, S. Lu, and D.-L. Deng, Quantum Private Distributed Learning Through Blind Quantum Computing, arXiv:2103.08403 (2021).

[65] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz, Quantum Sci. Technol. **4**, 014008 (2018).

[66] A. Blance and M. Spannowsky, Quantum machine learning for particle physics using a variational quantum classifier, J. High Energ. Phys. **2021** (2), 212.

[67] I. Kerenidis, J. Landman, and A. Prakash, Quantum

Algorithms for Deep Convolutional Neural Networks, arXiv:1911.01117 (2019).
[68] J. Liu, K. H. Lim, K. L. Wood, W. Huang, C. Guo, and H.-L. Huang, Hybrid Quantum-Classical Convolutional Neural Networks, arXiv:1911.02998 (2021).
[69] S. Wei, Y. Chen, Z. Zhou, and G. Long, A Quantum Convolutional Neural Network on NISQ Devices, arXiv:2104.06918 (2021).
[70] H. Yano, Y. Suzuki, R. Raymond, and N. Yamamoto, Efficient Discrete Feature Encoding for Variational Quantum Classifier, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2020) pp. 11–21.
[71] Y. Li, R.-G. Zhou, R. Xu, J. Luo, and W. Hu, A quantum deep convolutional neural network for image recognition, Quantum Sci. Technol. **5**, 044003 (2020).
[72] G. G. Guerreschi and M. Smelyanskiy, Practical optimization for hybrid quantum-classical algorithms, arXiv:1701.01450 (2017).
[73] O. Kyriienko and V. E. Elfving, Generalized quantum circuit differentiation rules, arXiv:2108.01218 (2021).
[74] T. Hur, L. Kim, and D. K. Park, Quantum convolutional neural network for classical data classification, arXiv:2108.00661 (2021).
[75] I. MacCormack, C. Delaney, A. Galda, N. Aggarwal, and P. Narang, Branching Quantum Convolutional Neural Networks, arXiv:2012.14439 (2020).
[76] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum (NISQ) algorithms, arXiv:2101.08448 (2021).
[77] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, Variational Quantum Algorithms, arXiv:2012.09265 (2020).
[78] J. Preskill, Quantum computing in the nisq era and beyond, Quantum **2**, 79 (2018).
[79] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. , 1 (2021).
[80] G. Bebis and M. Georgiopoulos, Feed-forward neural networks, IEEE Potentials **13**, 27 (1994).
[81] D. Svozil, V. Kvasnicka, and J. Pospichal, Introduction to multi-layer feed-forward neural networks, Chemometrics and Intelligent Laboratory Systems **39**, 43 (1997).
[82] S. Lawrence, C. Giles, A. C. Tsoi, and A. Back, Face recognition: A convolutional neural-network approach, IEEE Trans. Neural Netw. **8**, 98 (1997).
[83] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur, Extensions of recurrent neural network language model, in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2011) pp. 5528–5531.
[84] W. Zaremba, I. Sutskever, and O. Vinyals, Recurrent Neural Network Regularization, arXiv:1409.2329 (2015).
[85] E. Xi, S. Bing, and Y. Jin, Capsule Network Performance on Complex Data, arXiv:1712.03480 (2017).
[86] Z. Xinyi and L. Chen, Capsule Graph Neural Network, in *International Conference on Learning Representations* (2018).
[87] A. G. Rattew, S. Hu, M. Pistoia, R. Chen, and S. Wood, A domain-agnostic, noise-resistant evolutionary variational quantum eigensolver for hardware-efficient optimization in the hilbert space, arXiv:1910.09694 (2019).
[88] R. Li, U. Alvarez-Rodriguez, L. Lamata, and E. Solano, Approximate quantum adders with genetic algorithms: an ibm quantum experience, Quantum Meas. Quantum Metrol. **4**, 1 (2017).
[89] S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, Differentiable quantum architecture search, arXiv:2010.08561 (2020).
[90] L. Cincio, Y. Subaşı, A. T. Sornborger, and P. J. Coles, Learning the quantum algorithm for state overlap, New J. Phys. **20**, 113022 (2018).
[91] L. Cincio, K. Rudinger, M. Sarovar, and P. J. Coles, Machine Learning of Noise-Resilient Quantum Circuits, PRX Quantum **2**, 010324 (2021).
[92] Z. Lu, P.-X. Shen, and D.-L. Deng, Markovian quantum neuroevolution for machine learning, arXiv:2012.15131 (2020).
[93] D. Chivilikhin, A. Samarin, V. Ulyantsev, I. Iorsh, A. Oganov, and O. Kyriienko, Mog-vqe: Multiobjective genetic variational quantum eigensolver, arXiv:2007.04424 (2020).
[94] M. Pirhooshyaran and T. Terlaky, Quantum circuit design search, arXiv:2012.04046 (2020).
[95] M. Ostaszewski, E. Grant, and M. Benedetti, Structure optimization for parameterized quantum circuits, Quantum **5**, 391 (2021).
[96] L. Li, M. Fan, M. Coram, P. Riley, and S. Leichenauer, Quantum optimization with a novel Gibbs objective function and ansatz architecture search, Phys. Rev. Research **2**, 023074 (2020).
[97] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Reinforcement learning with neural networks for quantum feedback, Phys. Rev. X **8**, 031084 (2018).
[98] A. Bolens and M. Heyl, Reinforcement Learning for Digital Quantum Simulation, arXiv:2006.16269 (2020).
[99] H. Wang, Y. Ding, J. Gu, Y. Lin, D. Z. Pan, F. T. Chong, and S. Han, QuantumNAS: Noise-Adaptive Search for Robust Quantum Circuits, arXiv:2107.10845 (2021).
[100] S. Shalev-Shwartz, O. Shamir, and S. Shammah, Failures of Gradient-Based Deep Learning, arXiv:1703.07950 (2017).
[101] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 1 (2018).
[102] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat Commun **12**, 1791 (2021).
[103] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, and P. J. Coles, Absence of barren plateaus in quantum convolutional neural networks, arXiv:2011.02966 (2020).
[104] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, Noise-induced barren plateaus in variational quantum algorithms, arXiv:2007.14384 (2020).
[105] P. Huembeli and A. Dauphin, Characterizing the loss landscape of variational quantum circuits, Quantum Sci. Technol. **6**, 025011 (2021).
[106] T. L. Patti, K. Najafi, X. Gao, and S. F. Yelin, Entanglement devised barren plateau mitigation, Phys. Rev. Research **3**, 033090 (2021).
[107] C. O. Marrero, M. Kieferová, and N. Wiebe, Entanglement induced barren plateaus, arXiv:2010.15968 (2020).
[108] A. V. Uvarov and J. D. Biamonte, On barren plateaus and cost function locality in variational quantum algorithms, J. Phys. A: Math. Theor. **54**, 245301 (2021).
[109] A. Arrasmith, M. Cerezo, P. Czarnik, L. Cincio, and P. J. Coles, Effect of barren plateaus on gradient-free optimization, arXiv:2011.12245 (2020).
[110] D. Wierichs, C. Gogolin, and M. Kastoryano, Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer, Phys. Rev. Research **2**, 043246 (2020).

[111] M. Cerezo and P. J. Coles, Higher order derivatives of quantum neural networks with barren plateaus, Quantum Sci. Technol. (2021).

[112] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, An initialization strategy for addressing barren plateaus in parametrized quantum circuits, Quantum **3**, 214 (2019).

[113] T. Haug and M. S. Kim, Optimal training of variational quantum algorithms without barren plateaus, arXiv:2104.14543 (2021).

[114] Z. Holmes, A. Arrasmith, B. Yan, P. J. Coles, A. Albrecht, and A. T. Sornborger, Barren Plateaus Preclude Learning Scramblers, Phys. Rev. Lett. **126**, 190501 (2021).

[115] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, arXiv:2101.02138 (2021).

[116] C. Zhao and X.-S. Gao, Analyzing the barren plateau phenomenon in training quantum neural networks with the ZX-calculus, Quantum **5**, 466 (2021).

[117] M. Kieferova, O. M. Carlos, and N. Wiebe, Quantum Generative Training Using R\'enyi Divergences, arXiv:2106.09567 (2021).

[118] Z. Liu, L.-W. Yu, L.-M. Duan, and D.-L. Deng, The Presence and Absence of Barren Plateaus in Tensor-network Based Machine Learning, arXiv:2108.08312 (2021).

[119] N. Liu and P. Wittek, Vulnerability of quantum classification to adversarial perturbations, Phys. Rev. A **101**, 062331 (2020).

[120] H. Liao, I. Convy, W. J. Huggins, and K. B. Whaley, Robust in practice: Adversarial attacks on quantum machine learning, Phys. Rev. A **103**, 042427 (2021).

[121] M. Weber, N. Liu, B. Li, C. Zhang, and Z. Zhao, Optimal provable robustness of quantum classification via quantum hypothesis testing, npj Quantum Inf **7**, 1 (2021).

[122] Y. Du, M.-H. Hsieh, T. Liu, D. Tao, and N. Liu, Quantum noise protects quantum classifiers against adversaries, Phys. Rev. Research **3**, 023153 (2021).

[123] W. Gong and D.-L. Deng, Universal Adversarial Examples and Perturbations for Quantum Classifiers, arXiv:2102.07788 (2021).

[124] C. Cortes and V. Vapnik, Support-vector networks, Mach. Learn. **20**, 273 (1995).

[125] W. S. Noble, What is a support vector machine?, Nat Biotechnol **24**, 1565 (2006).

[126] S. Suthaharan, Support Vector Machine, in *Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning*, Integrated Series in Information Systems, edited by S. Suthaharan (Springer US, Boston, MA, 2016) pp. 207–235.

[127] B. E. Boser, I. M. Guyon, and V. N. Vapnik, A training algorithm for optimal margin classifiers, in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92 (Association for Computing Machinery, New York, NY, USA, 1992) pp. 144–152.

[128] J. A. Suykens and J. Vandewalle, Least squares support vector machine classifiers, Neural Process Lett **9**, 293 (1999).

[129] H. Buhrman, R. Cleve, J. Watrous, and R. De Wolf, Quantum fingerprinting, Phys. Rev. Lett. **87**, 167902 (2001).

[130] S. Aaronson, Read the fine print, Nat. Phys. **11**, 291 (2015).

[131] V. Giovannetti, S. Lloyd, and L. Maccone, Quantum random access memory, Phys. Rev. Lett. **100**, 160501 (2008).

[132] J. R. Quinlan, Induction of decision trees, Mach. Learn. **1**, 81 (1986).

[133] J. Quinlan, *C4. 5: programs for machine learning* (Elsevier, 2014).

[134] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees* (CRC press, 1984).

[135] D. R. Carvalho and A. A. Freitas, A hybrid decision tree/genetic algorithm method for data mining, Inform. Sci. **163**, 13 (2004).

[136] R. Kohavi and J. R. Quinlan, Data mining tasks and methods: Classification: Decision-tree discovery, in *Handbook of Data Mining and Knowledge Discovery* (Oxford University Press, Inc., USA, 2002) pp. 267–276.

[137] Y.-Y. Song and L. Ying, Decision tree methods: applications for classification and prediction, Shanghai Arch Psychiatry **27**, 130 (2015).

[138] H. Laurent and R. L. Rivest, Constructing optimal binary decision trees is np-complete, Inform. Process. Lett. **5**, 15 (1976).

[139] E. Farhi and S. Gutmann, Quantum computation and decision trees, Phys. Rev. A **58**, 915 (1998).

[140] Y. Shi, Entropy lower bounds for quantum decision tree complexity, Inform. Process. Lett. **81**, 23 (2002).

[141] T. Cover and P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory **13**, 21 (1967).

[142] J. L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM **18**, 509 (1975).

[143] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, Quantum amplitude amplification and estimation, Contemp. Math. **305**, 53 (2002).

[144] C. Durr and P. Hoyer, A quantum algorithm for finding the minimum, arXiv:quant-ph/9607014 (1996).

[145] W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bull Math Biophys **5**, 115 (1943).

[146] F. Rosenblatt, *The perceptron, a perceiving and recognizing automaton Project Para* (Cornell Aeronautical Laboratory, 1957).

[147] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain., Psychol. Rev. **65**, 386 (1958).

[148] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM **60**, 84 (2017).

[149] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980 (2014).

[150] A. M. Childs and N. Wiebe, Hamiltonian simulation using linear combinations of unitary operations, arXiv:1202.5822 (2012).

[151] Y.-Y. Shi, L.-M. Duan, and G. Vidal, Classical simulation of quantum many-body systems with a tree tensor network, Phys. Rev. A **74**, 022320 (2006).

[152] G. Vidal, Class of quantum many-body states that can be efficiently simulated, Phys. Rev. Lett. **101**, 110501 (2008).

[153] L. Cincio, J. Dziarmaga, and M. M. Rams, Multiscale entanglement renormalization ansatz in two dimensions: quantum ising model, Phys. Rev. Research **100**, 240603 (2008).

[154] A. Broadbent, J. Fitzsimons, and E. Kashefi, Universal Blind Quantum Computation, in *2009 50th Annual IEEE Symposium on Foundations of Computer Science* (2009) pp. 517–526.

[155] J. M. Renes, R. Blume-Kohout, A. J. Scott, and C. M. Caves, Symmetric informationally complete quantum measurements, J. Math. Phys. **45**, 2171 (2004).

[156] C. Dankert, R. Cleve, J. Emerson, and E. Livine, Exact and approximate unitary 2-designs and their application to fidelity estimation, Phys. Rev. A **80**, 012304 (2009).

[157] A. W. Harrow and R. A. Low, Random quantum circuits are approximate 2-designs, Comm. Math. Phys. **291**, 257 (2009).

[158] S. Popescu, A. J. Short, and A. Winter, Entanglement and the foundations of statistical mechanics, Nature Phys **2**, 754

(2006).
[159] M. J. Bremner, C. Mora, and A. Winter, Are Random Pure States Useful for Quantum Computation?, Phys. Rev. Lett. **102**, 190502 (2009).
[160] D. Gross, S. T. Flammia, and J. Eisert, Most Quantum States Are Too Entangled To Be Useful As Computational Resources, Phys. Rev. Lett. **102**, 190501 (2009).
[161] G. E. Hinton, S. Osindero, and Y.-W. Teh, A Fast Learning Algorithm for Deep Belief Nets, Neural Computation **18**, 1527 (2006).
[162] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, Greedy Layer-Wise Training of Deep Networks, in *Advances in Neural Information Processing Systems*, Vol. 19 (MIT Press, 2007).
[163] N. Carlini and D. Wagner, Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods, in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17 (Association for Computing Machinery, New York, NY, USA, 2017) pp. 3–14.
[164] A. Kurakin, I. Goodfellow, and S. Bengio, Adversarial examples in the physical world, arXiv:1607.02533 (2016).
[165] D. J. Miller, Z. Xiang, and G. Kesidis, Adversarial learning in statistical classification: A comprehensive review of defenses against attacks, arXiv:1904.06292 (2019).
[166] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, Adversarial machine learning, in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, AISec '11 (Association for Computing Machinery, New York, NY, USA, 2011) pp. 43–58.
[167] Y. Vorobeychik and M. Kantarcioglu, Adversarial machine learning, Synth. Lect. Artif. Intell. Mach. Learn. **12**, 1 (2018).
[168] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, Black-box adversarial attacks with limited queries and information, arXiv:1804.08598 (2018).
[169] V. Tjeng, K. Xiao, and R. Tedrake, Evaluating robustness of neural networks with mixed integer programming, arXiv:1711.07356 (2017).
[170] I. J. Goodfellow, J. Shlens, and C. Szegedy, Explaining and harnessing adversarial examples, arXiv:1412.6572 (2014).
[171] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, Intriguing properties of neural networks, arXiv:1312.6199 (2013).
[172] A. Athalye, N. Carlini, and D. Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, arXiv:1802.00420 (2018).
[173] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, Practical Black-Box Attacks against Machine Learning, in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17 (Association for Computing Machinery, New York, NY, USA, 2017) pp. 506–519.
[174] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, Towards deep learning models resistant to adversarial attacks, arXiv:1706.06083 (2017).
[175] B. Biggio and F. Roli, Wild patterns: Ten years after the rise of adversarial machine learning, Pattern Recognition **84**, 317 (2018).
[176] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models, in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17 (Association for Computing Machinery, New York, NY, USA, 2017) pp. 15–26.
[177] K. Poland, K. Beer, and T. J. Osborne, No Free Lunch for Quantum Machine Learning, arXiv:2003.14103 (2020).
[178] K. Sharma, M. Cerezo, Z. Holmes, L. Cincio, A. Sornborger, and P. J. Coles, Reformulation of the No-Free-Lunch Theorem for Entangled Data Sets, arXiv:2007.04900 (2020).
[179] R. A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugen. **7**, 179 (1936).
[180] P. G. Kwiat, K. Mattle, H. Weinfurter, A. Zeilinger, A. V. Sergienko, and Y. Shih, New high-intensity source of polarization-entangled photon pairs, Phys. Rev. Lett. **75**, 4337 (1995).
[181] J.-W. Pan, Z.-B. Chen, C.-Y. Lu, H. Weinfurter, A. Zeilinger, and M. Żukowski, Multiphoton entanglement and interferometry, Rev. Mod. Phys. **84**, 777 (2012).
[182] X.-Q. Zhou, P. Kalasuwan, T. C. Ralph, and J. L. O'Brien, Calculating unknown eigenvalues with a quantum algorithm, Nature Photon **7**, 223 (2013).
[183] J. C. Spall, A one-measurement form of simultaneous perturbation stochastic approximation, Automatica J. IFAC **33**, 109 (1997).
[184] J. C. Spall, Adaptive stochastic approximation by the simultaneous perturbation method, IEEE Trans. Autom. Control **45**, 1839 (2000).
[185] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, The power of quantum neural networks, Nat Comput Sci **1**, 403 (2021).
[186] J. J. Meyer, Fisher Information in Noisy Intermediate-Scale Quantum Applications, arXiv:2103.15191 (2021).
[187] X. Wang, Y. Du, Y. Luo, and D. Tao, Towards understanding the power of quantum kernels in the NISQ era, arXiv:2103.16774 (2021).
[188] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Phys. Rev. A **103**, 032430 (2021).
[189] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, arXiv:2106.03880 (2021).
[190] Y. Wu, J. Yao, P. Zhang, and H. Zhai, Expressivity of quantum neural networks, Phys. Rev. Research **3**, L032049 (2021).
[191] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Yao. jl: Extensible, efficient framework for quantum algorithm design, arXiv:1912.10877 (2019).
[192] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, Julia: A Fresh Approach to Numerical Computing, SIAM Rev. **59**, 65 (2017).
[193] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, S. V. Isakov, P. Massey, M. Y. Niu, R. Halavati, E. Peters, M. Leib, A. Skolik, M. Streif, D. Von Dollen, J. R. McClean, S. Boixo, D. Bacon, A. K. Ho, H. Neven, and M. Mohseni, TensorFlow Quantum: A Software Framework for Quantum Machine Learning, arXiv:2003.02989 (2020).
[194] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri, K. McKiernan, J. J. Meyer, Z. Niu, A. Száva, and N. Killoran, PennyLane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968 (2020).
[195] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook, Strawberry Fields: A Software Platform for Photonic Quantum Computing, Quantum **3**, 129 (2019).
[196] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernández, J. Carballo-Franquis, A. Chen, C.-F. Chen, J. M. Chow, A. D. Córcoles-Gonzales, A. J. Cross, A. Cross, J. Cruz-Benito, C. Culver, S. D. L. P. González, E. D. L. Torre, D. Ding, E. Dumitrescu, I. Duran, P. Eendebak, M. Everitt,

I. F. Sertage, A. Frisch, A. Fuhrer, J. Gambetta, B. G. Gago, J. Gomez-Mosquera, D. Greenberg, I. Hamamura, V. Havlicek, J. Hellmers, Ł. Herok, H. Horii, S. Hu, T. Imamichi, T. Itoko, A. Javadi-Abhari, N. Kanazawa, A. Karazeev, K. Krsulich, P. Liu, Y. Luh, Y. Maeng, M. Marques, F. J. Martín-Fernández, D. T. McClure, D. McKay, S. Meesala, A. Mezzacapo, N. Moll, D. M. Rodríguez, G. Nannicini, P. Nation, P. Ollitrault, L. J. O'Riordan, H. Paik, J. Pérez, A. Phan, M. Pistoia, V. Prutyanov, M. Reuter, J. Rice, A. R. Davila, R. H. P. Rudy, M. Ryu, N. Sathaye, C. Schnabel, E. Schoute, K. Setia, Y. Shi, A. Silva, Y. Siraichi, S. Sivarajah, J. A. Smolin, M. Soeken, H. Takahashi, I. Tavernelli, C. Taylor, P. Taylour, K. Trabing, M. Treinish, W. Turner, D. Vogt-Lee, C. Vuillot, J. A. Wildstrom, J. Wilson, E. Winston, C. Wood, S. Wood, S. Wörner, I. Y. Akhalwaya, and C. Zoufal, Qiskit: An Open-source Framework for Quantum Computing, Zenodo (2019).

[197] K. Svore, A. Geller, M. Troyer, J. Azariah, C. Granade, B. Heim, V. Kliuchnikov, M. Mykhailova, A. Paz, and M. Roetteler, Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL, in *Proceedings of the Real World Domain Specific Languages Workshop 2018*, RWDSL2018 (Association for Computing Machinery, New York, NY, USA, 2018) pp. 1–10.

[198] F. Zhang, C. Huang, M. Newman, J. Cai, H. Yu, Z. Tian, B. Yuan, H. Xu, J. Wu, X. Gao, J. Chen, M. Szegedy, and Y. Shi, Alibaba Cloud Quantum Development Platform: Large-Scale Classical Simulation of Quantum Circuits, arXiv:1907.11217 (2019).

[199] C. Huang, M. Szegedy, F. Zhang, X. Gao, J. Chen, and Y. Shi, Alibaba Cloud Quantum Development Platform: Applications to Quantum Algorithm Design, arXiv:1909.02559 (2019).

[200] D. Nguyen, D. Mikushin, and Y. Man-Hong, HiQ-ProjectQ: Towards user-friendly and high-performance quantum computing on GPUs, in *2021 Design, Automation Test in Europe Conference Exhibition (DATE)* (2021) pp. 1056–1061.

[201] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, Quipper: A scalable quantum programming language, in *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '13 (Association for Computing Machinery, New York, NY, USA, 2013) pp. 333–342.

[202] A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, ScaffCC: Scalable compilation and analysis of quantum programs, Parallel Computing Computing Frontiers 2014: Best Papers, **45**, 2 (2015).

[203] N. Khammassi, I. Ashraf, X. Fu, C. Almudever, and K. Bertels, QX: A high-performance quantum computer simulation platform, in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017* (2017) pp. 464–469.

[204] J. R. Johansson, P. D. Nation, and F. Nori, QuTiP: An open-source Python framework for the dynamics of open quantum systems, Computer Physics Communications **183**, 1760 (2012).

[205] L. Bittel and M. Kliesch, Training variational quantum algorithms is NP-hard – even for logarithmically many qubits and free fermionic systems, arXiv:2101.07267 (2021).

[206] H.-Y. Huang, R. Kueng, and J. Preskill, Information-Theoretic Bounds on Quantum Advantage in Machine Learning, Phys. Rev. Lett. **126**, 190505 (2021).

[207] K. Bu, D. E. Koh, L. Li, Q. Luo, and Y. Zhang, On the statistical complexity of quantum circuits, arXiv:2101.06154 (2021).

[208] H. Cai, Q. Ye, and D.-L. Deng, Sample Complexity of Learning Quantum Circuits, arXiv:2107.09078 (2021).