Research
A SCIENCE PARTNER JOURNAL

RESEARCH ARTICLE

Quantum Neural Network for Quantum Neural Computing

Min-Gang Zhou[1†], Zhi-Ping Liu[1†], Hua-Lei Yin[1†], Chen-Long Li[1], Tong-Kai Xu[2], and Zeng-Bing Chen[1,2*]

[1]National Laboratory of Solid State Microstructures, School of Physics, Collaborative Innovation Center of Advanced Microstructures, Nanjing University, Nanjing 210093, China. [2]MatricTime Digital Technology Co. Ltd., Nanjing 211899, China.

*Address correspondence to: zbchen@nju.edu.cn
†These authors contributed equally to this work.

Citation: Zhou MG, Liu ZP, Yin HL, Li CL, Xu TK, Chen ZB. Quantum Neural Network for Quantum Neural Computing. *Research* 2023;6:Article 0134. https://doi.org/10.34133/research.0134

Submitted 21 December 2022
Accepted 11 April 2023
Published 8 May 2023

Neural networks have achieved impressive breakthroughs in both industry and academia. How to effectively develop neural networks on quantum computing devices is a challenging open problem. Here, we propose a new quantum neural network model for quantum neural computing using (classically controlled) single-qubit operations and measurements on real-world quantum systems with naturally occurring environment-induced decoherence, which greatly reduces the difficulties of physical implementations. Our model circumvents the problem that the state-space size grows exponentially with the number of neurons, thereby greatly reducing memory requirements and allowing for fast optimization with traditional optimization algorithms. We benchmark our model for handwritten digit recognition and other nonlinear classification tasks. The results show that our model has an amazing nonlinear classification ability and robustness to noise. Furthermore, our model allows quantum computing to be applied in a wider context and inspires the earlier development of a quantum neural computer than standard quantum computers.

Introduction

Developing new computing paradigms [1–4] has attracted considerable attention in recent years due to the increasing cost of computing and the von Neumann bottleneck [5]. Conventional (hard) computing is characterized by precision, certainty, and rigor. In contrast, "soft computing" [1,2] is a newer approach to computing that mimics human thinking to learn and reason in an environment of imprecision, uncertainty, and partial truth. This approach aims to address real-world complexities with tractability, robustness, and low solution costs. In particular, neural networks (NNs), a subfield of soft computing, have rapidly evolved in both theory and practice during the current machine learning boom [6,7]. With backpropagation algorithms, NNs have achieved impressive breakthroughs in both industry and academia [8,9] and may even alter the way computation is performed [4]. However, the training cost of NNs can become very expensive as the network size increases [10]. More seriously, it is difficult for NNs to simulate quantum many-body systems with exponentially large quantum state spaces [11], which restricts basic scientific research and the intelligent development of biopharmaceutical and material design.

Quantum computing [3] is another paradigm shift in computing, and it promises to solve the aforementioned difficulties of NNs. How to effectively develop NNs on quantum computing devices is a challenging open problem [11–13] that is still in its initial stages of exploration. In recent years, many novel and original works have attempted to develop well-performing quantum NN models [14–25] on noisy intermediate-scale quantum devices [26], and these networks can be used to learn tasks involving quantum data or to improve classical models. However, despite the remarkable progress in the physical implementation of quantum computing in recent years, a number of important challenges remain for building a large-scale quantum computer [27–29]. Thus, if the quest for quantum NNs heavily relies on standard quantum computing devices, the scope of applying quantum NNs might be quite restrictive.

A real-world quantum system is always characterized by nonunitary, faulty evolutions and is coupled with a noisy and dissipative environment. The real-system complexities in the quantum domain call for a new paradigm of quantum computing aiming at nonclassical computation using real-world quantum systems. Thus, the new quantum computing paradigm, called soft quantum computing to be compared with classical soft computing, deals with classically intractable computation under the conditions of noisy and faulty quantum evolutions and measurements, while being tolerant of those effects that are detrimental for the standard quantum computing paradigm.

Here, we propose for the first time a quantum NN model to illustrate soft quantum computing. Unlike other quantum NN models, we develop NNs for quantum neural computing based on "soft quantum neurons", which are building blocks of soft quantum computing and subject to only single-qubit operations and classically controlled single-qubit operations and measurements, thus markedly reducing the difficulties of

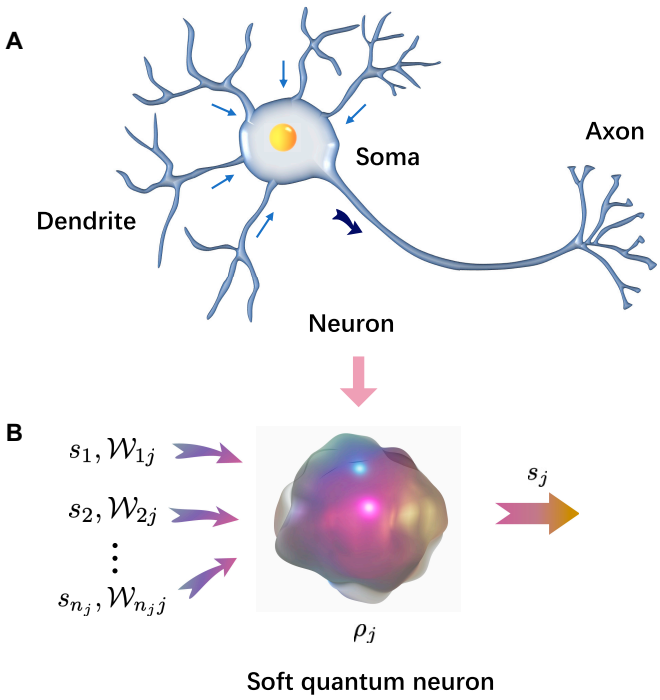

Fig. 1. A drastically simplified drawing of a neuron and the soft quantum neuron model. (A) The neuron integrates hundreds or thousands of impinging signals through its dendrites. After processing by the cell body, the neuron outputs a signal through its axon to another neuron for processing in the form of an action potential when its internal potential exceeds a certain threshold. (B) Similarly, a "soft quantum neuron" can, in principle, receive hundreds or thousands of input signals. These signals affect the evolution of the soft quantum neuron. The evolved soft quantum neuron is measured and decides whether to output a signal according to the measurement result.

physical implementations. We demonstrate that quantum correlations characterized by non-zero quantum discord are present for quantum neurons in our model. The simulation results show that our quantum perceptron can be used to classify nonlinear problems and simulate the XOR gate. In contrast, classical perceptrons do not possess such nonlinear classification capabilities. Furthermore, our model is able to classify handwritten digits with an extraordinary generalization ability even without hidden layers. Our model also has a marked accuracy advantage over other quantum NNs for the above-mentioned tasks. Prominently, the proposed soft quantum neurons can be integrated into quantum analogues of typical topological architectures [30–33] in classical NNs. The respective advantages of quantum technology and classical network architectures can thus be well combined in our quantum NN model.

Results

Soft quantum neurons

Quantizing the smallest building block of classical NNs, namely, the neuron, is a key challenge in building quantum NNs. Our soft quantum neuron model is inspired by biological neurons (Fig. 1), which can be implemented on realistic quantum systems. The term "soft" utilized here highlights the ability of our model to handle realistic environments and evolutions, distinguishing it from the standard quantum computing models. It is worth noting that soft computing is a proprietary term that is conceptually opposite to hard computing. In our proposal, a quantum neuron is modeled by a noisy qubit, which can be coupled with its surrounding environment. The initial state of the jth neuron can be described by a density matrix ρ_j^{in} in the computational basis $|0\rangle$ and $|1\rangle$. The quantum neuron ρ_j^{in} accepts n_j outputs s_i ($i = 1, 2, \ldots, n_j$) from the final states ρ_i^{out} ($i = 1, 2, \ldots, n_j$) of the other possible n_j neurons. The output s_i is determined by a 2-outcome projective measurement on ρ_i^{out} in the computational basis. It is therefore a classical binary signal, namely, $s_i = 0$ or 1. When $s_i = 1$, corresponding to the case where ρ_i^{out} is measured and collapses to the state $|1\rangle$, the quantum neuron ρ_j^{in} is acted upon by an arbitrary superoperator $\mathcal{W}_{ij}$, while when $s_i = 0$ (ρ_i^{out} collapses to the state $|0\rangle$), nothing happens to ρ_j^{in}. Ideally, $\mathcal{W}_{ij}$ can be replaced by a corresponding unitary operator W_{ij}. As a result, the evolution of the whole system from the state $\otimes_{i=1}^{n_j} \rho_i^{out} \otimes \rho_j^{in}$ is

$$\begin{aligned}\rho_{\{i\}j}^{mid} &= \mathcal{T} \otimes_{i=1}^{n_j} \mathcal{O}_{ij}\left(\rho_i^{out} \otimes \rho_j^{in}\right) \\ &\equiv \mathcal{T} \otimes_{i=1}^{n_j} \left[\mathcal{P}_{|0\rangle_i} \otimes \hat{I}_j + \mathcal{P}_{|1\rangle_i} \otimes \mathcal{W}_{ij}\right]\left(\rho_i^{out} \otimes \rho_j^{in}\right)\end{aligned} \tag{1}$$

where, $\mathcal{O}_{ij}$ is a classically controlled single-qubit operation, the superprojectors $\mathcal{P}_{|s\rangle}$ are defined by $\mathcal{P}_{|s\rangle}\rho = |s\rangle\langle s|\rho|s\rangle\langle s|$, $\hat{I}$ is the identity operator, and $\mathcal{T}$ represents a time-ordering operation. All $\mathcal{W}_{ij}$ act upon the target neuron ρ_j^{in} with specific temporal patterns. As different quantum operations $\mathcal{W}_{ij}$ might be noncommutative, the time-ordering of these operations is important. The state of the target neuron after the evolution of Eq. 1 can be obtained by tracing out all the input neurons ρ_i^{out}, namely, $\rho_j^{mid} = \mathrm{tr}_{\{i\}}\rho_{\{i\}j}^{mid} = \mathcal{T}\prod_{i=1}^{n_j}\left[p_i\hat{I}_j + (1-p_i)\mathcal{W}_{ij}\right]\rho_j^{in}$, where $p_i \equiv p_i(0) = \mathrm{tr}(|0\rangle_i\langle 0|\rho_i^{out})$.

After the evolution of Eq. 1, the target neuron ρ_j^{mid} is independently acted upon by a local bias superoperator $\mathcal{U}_j$. This operator is designed to improve the flexibility and learning ability of quantum neurons. Ideally, $\mathcal{U}_j$ can be replaced by a corresponding unitary operator U_j. The action of $\mathcal{U}_j$ is similar to adding bias to neurons in classical NNs [6,7]. The final state of the target neuron is thus $\rho_j^{out} = \mathcal{U}_j\left(\rho_j^{mid}\right)$. Similarly, the output s_j of the target neuron is obtained by a 2-outcome projective measurement on ρ_j^{out} in the computational basis. The output signal s_j of the target neuron is

$$s_j = \begin{cases} 0 & \text{with probability } p_j(0) \\ 1 & \text{with probability } 1 - p_j(0). \end{cases} \tag{2}$$

The output s_j can be accepted by all other connecting quantum neurons and affects the evolution of the quantum neurons that accept the output. This completes the specification of our proposed quantum neuron model. Strikingly, our model contains noisy cases, which allow our model to work under the conditions of noisy and faulty quantum evolutions and measurements. An elementary setup of our model is the soft quantum perceptron, which consists of a soft quantum neuron accepting inputs of n other soft quantum neurons and providing a single output, though in probability.

Quantumness of quantum neurons

All final states of our quantum neurons are mixed states, as the evolution of these neurons depends on the measurements of their input neurons, thus introducing classical probability. Although such measurements make the neurons evolve into mixed states, the proposed quantum neurons can still develop quantum correlations arising from quantum discord. To make this clear, we consider the simplest 2-neuron case. For the 2 neurons in the states $\rho_1^{out} = p_1|0\rangle_1\langle 0| + (1-p_1)|1\rangle_1\langle 1|\ (p_1 \neq 0, 1)$ and ρ_2^{in}, the action of an operation $\mathcal{O}_{12}$ results in the state

$$\begin{aligned}\rho_{12}^{mid} &= \left(\mathcal{P}_{|0\rangle_1}\otimes\hat{I}_2 + \mathcal{P}_{|1\rangle_1}\otimes\mathcal{W}_{12}\right)\left(\rho_1^{out}\otimes\rho_2^{in}\right)\\ &= p_1|0\rangle_1\langle 0|\otimes\rho_2^{in} + (1-p_1)|1\rangle_1\langle 1|\otimes\mathcal{W}_{12}\left(\rho_2^{in}\right)\end{aligned} \tag{3}$$

where $\mathcal{W}_{12}$ represents a specific quantum channel. Quantum correlations, if any, of ρ_{12}^{mid} can be quantified by the quantum discord [34]. Any bipartite state is called fully classically correlated if it is of the form [35] $\rho_{12}^{c} = \sum_{i,j} p_{ij}|i\rangle_1\langle i|\otimes|j\rangle_2\langle j|$; otherwise, it is quantum-correlated. Here, $|i\rangle_1$ and $|j\rangle_2$ are the orthonormal bases of the 2 parties, with nonnegative probabilities p_{ij}.

Obviously, for ρ_{12}^{mid} in Eq. 3, the first neuron becomes quantum-correlated with the second as long as $\mathcal{W}_{12}\left(\rho_2^{in}\right)$ and ρ_2^{in} are nonorthogonal [36–38]. In particular, Refs. [37,38] show the creation of discord, from classically correlated 2-qubit states, by applying an amplitude-damping process only on one of the qubits; for the phase-damping process, see Ref. [35]. Actually, ρ_{12}^{mid} in Eq. 3 is the classical-quantum state, as dubbed in Ref. [37]. While the discord is zero for measurements on neuron-1, measurements on neuron-2 in general lead to nonzero discord.

Thus, we reveal a crucial property of our quantum neuron model. Namely, quantum correlations arising from quantum discord can be developed between the proposed quantum neurons in our model, although these neurons are generally in mixed states. Note that the existing quantum NN models are mainly based on variational quantum circuits requiring 2-qubit gates.

More remarkably, our model can be equated to quantum circuits generating quantum entanglement. To illustrate this more clearly, we take 3 neurons in Fig. 2A as an example and represent their interactions by a quantum circuit model shown in Fig. 2B. To facilitate the demonstration, we consider the ideal case where ρ_2^{in} and ρ_3^{in} are pure states and $\mathcal{W}_{ij}$ and $\mathcal{U}_j$ are replaced by corresponding unitary operators W_{ij} and U_j, respectively. According to the principle of deferred measurement [3], measurements can always be moved from the middle step of a quantum circuit to the end of the circuit. Therefore, the circuit in Fig. 2C is equivalent to that in Fig. 2B. In the equivalent circuit, the unitaries that are conditional on the measurement results are replaced by controlled unitary operations on ρ_2^{in} and ρ_3^{in}. It is easy to verify that quantum entanglement can exist between neuron-1 and neuron-2 (as well as neuron-3) in Fig. 2C. Another example of the principle of deferred measurement can be found in teleportation [3]. Nonetheless, it remains unclear whether this equivalence can be effectively utilized in computing tasks. We leave this matter for future work.

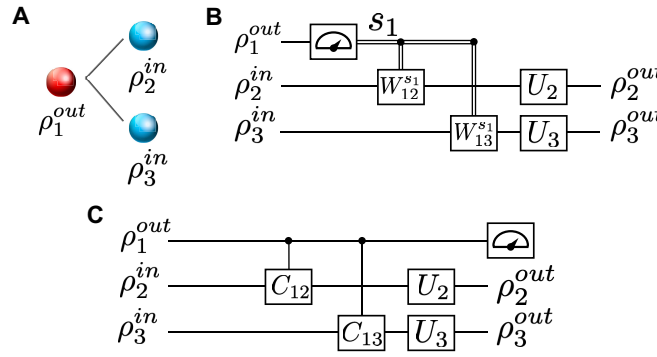

Fig. 2. Simplified demonstration of the principle of deferred measurement as applied to our model. (A) A simple example of our model. The quantum neuron ρ_1^{out} sends signals to ρ_2^{in} and ρ_3^{in}. (B) The quantum circuit model of (A). (C) The deferred measurement quantum circuit of (A), which is equivalent to (B). Here, C_{12} and C_{13} are controlled unitary operators acting on ρ_2^{in} and ρ_3^{in}, respectively.

Soft quantum neural network

Quantum neurons are connected together in various configurations to form quantum NNs with learning abilities, thus representing a quantum neural computing device obeying the evolution-measurement rules provided above. Our neurons can, in principle, be combined into quantum analogues of any classical network architecture that has proven effective in many applications. In this work, we present a fully connected soft quantum feedforward NN (SQFNN) for application to supervised learning.

Neurons are arranged in layers in a fully connected feedforward NN (FNN). Each neuron accepts all the signals sent by the neurons in the previous layer and outputs the integrated signal to each neuron in the next layer. Note that there is no signal transmission between neurons within the same layer. To date, there has been no satisfactory quantum version of this simple model. Because no neuron can perfectly copy its quantum state in multiple duplicates as an output to the next layer due to the quantum no-cloning theorem [39], the output is not perfectly shared by neurons in the next layer. Because of the same theorem, quantum neural computing and standard quantum computing have incompatible requirements that are difficult to reconcile [12]. Our quantum NN model resolves this incompatibility by measuring each soft quantum neuron to give classical information as the integrated signal. This feature is essential for our model to be a genuine quantum NN model, which, while incorporating a neural computing mechanism, uses quantum laws consistently throughout neural computing.

In fact, many studies have made bold attempts in this challenging area. For example, Ref. [14] introduces a general "fan-out" unit that distributes information about the input state into several output qubits. The quantum neuron in Ref. [15] is modeled as an arbitrary unitary operator with m input qubits and n output qubits. These attempts provide new perspectives for resolving the abovementioned incompatibility. Unfortunately, none of them directly confronts this incompatibility. The neurons in these schemes still cannot share the outputs of the neurons in the previous layer; conversely, each neuron can only send different signals to different neurons in the next layer. In that sense, our NN is quite different from these quantum NNs.

Figure 3 shows the concept of an SQFNN. Without loss of generality, we specify that signals propagate from top to bottom

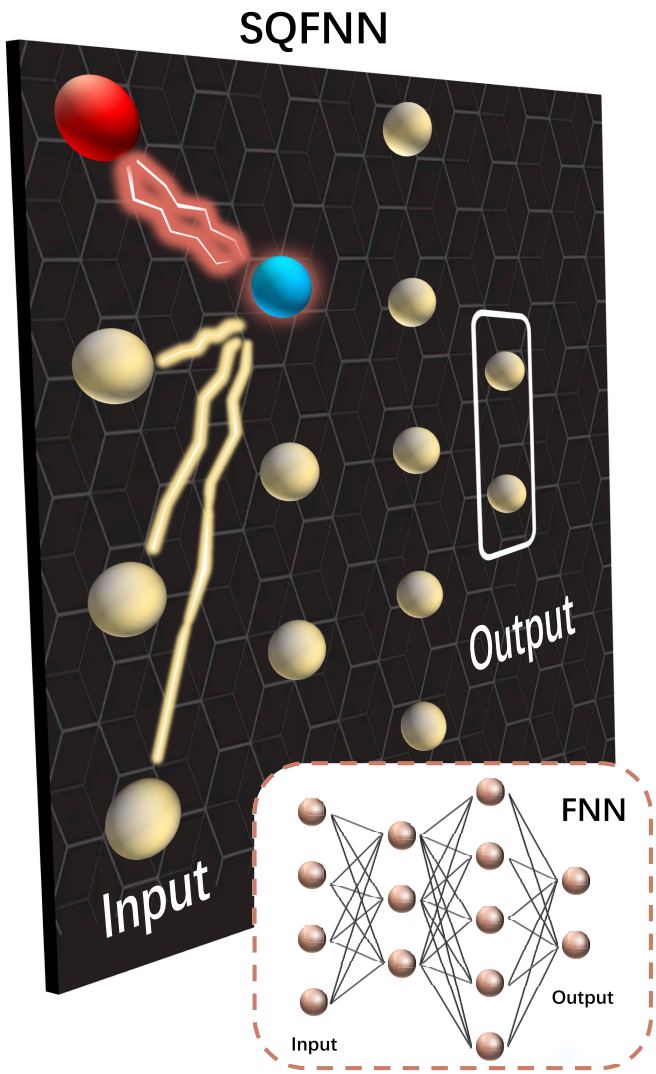

Fig. 3. The concept of SQFNNs. The network architecture of a soft quantum feedforward NN (SQFNN) is similar to that of the fully connected feedforward NN (FNN) displayed at the bottom right corner. There is no feedback in the entire network. Signals propagate unidirectionally from the input layer to the output layer. The first action occurs between the red neuron and the blue neuron. The part in the white box represents the output layer, whose average output is defined as the final output of the network.

and from left to right. Therefore, the evolution equation of the jth neuron in the lth layer is

$$\rho_{j^{(l)}}^{\text{out}} \equiv \mathcal{U}_{j^{(l)}} \text{tr}_{\{i^{(l-1)}\}} \left(\otimes_{i^{(l-1)}=1}^{n^{(l-1)}} \mathcal{O}_{i^{(l-1)}j^{(l)}} \left(\rho_{i^{(l-1)}}^{\text{out}} \otimes \rho_{j^{(l)}}^{\text{in}} \right) \right), \tag{4}$$

where $\mathcal{O}_{i^{(l-1)}j^{(l)}}$ acts on the ith neuron in the $(l-1)$th layer and the jth neuron in the lth layer. The final state of the output layer of the network can be obtained by calculating the final state of each neuron layer by layer with Eq. 4 after considering the local bias superoperator acting upon each neuron. Note that due to the randomness introduced by the measurement operations, the result of a single run of the quantum NN is unstable, i.e., probabilistic. One way to prevent this instability is to obtain the average output of the network by resetting and rerunning the entire network multiple times. This average output is more representative of the prediction made by our quantum NN and is therefore defined as the final output of the network. For each neuron of the output layer, the average output includes the binary outputs in the computation basis and their corresponding probabilities. Although running the network multiple times seems to consume more time and resources, this increase is only equivalent to an additional constant factor on the original consumption [15] and has no serious consequences. Therefore, running the network multiple times is common for extracting the information of quantum NNs and is also widely adopted by other quantum NN models [15,20]. Strikingly, this repetitive operation is easy and fast for a quantum computer. For example, the "Sycamore" quantum computer executed an instance of a quantum circuit a million times in 200 s [40].

In supervised learning, the NN must output a value close to the label of the training point. The closeness between the output and the label is usually measured by defining a loss function. The loss function in our model can be defined in various ways, e.g., by the fidelity between the output and the expected output or by a certain distance measure. In the simulations shown below, a mean squared error (MSE) loss function is adopted, which can be written as

$$\mathcal{L} = \sum_{k=1}^{N} \frac{1}{N} \left| y^k - \tilde{y}^k \right|^2, \tag{5}$$

where N represents the size of a training set, y^k represents the label of the kth training point, and $\tilde{y}^k$ represents the predicted label of our network for the kth training point, which is the average value of the output layer of the network obtained by resetting and rerunning the entire network multiple times. This loss function can be driven to a very low value by updating the parameters of the network, thereby improving the network performance. However, the loss function is nonconvex and thus requires iterative, gradient-based optimizers. As information is forward-propagated in our network, we can use a back-propagation algorithm to update the parameters of the quantum operations. Moreover, since only single-qubit gates are involved in our model, the total number of parameters is not large and is approximately $\sum_{l=1}^{L-1} 3\left(n_l + 1\right) \times n_{l+1}$, where L is the total number of layers in a network and n_l is the number of neurons in the lth layer. This number is directly proportional to the length L of the network and the square of the average width of the network (i.e., the average number of neurons per layer). In particular, the state space involved in computing the gradients is always that of a single neuron, thus circumventing the problem that the state-space size grows exponentially with the number of neurons. Many optimization algorithms widely used in classical NNs are therefore effectively compatible with our quantum NN, such as Adagrad [41], RMSprop [42], and Adam [43].

Both classical and quantum samples are available for our network, which is similar to other quantum NNs. For classical data, the input features need to be encoded into qubits and fed to the input layer. For quantum data, the quantum states can be decomposed into a tensor product of the qubits in the input layer, as in quantum circuits.

Simulations

In this section, we benchmark soft quantum perceptrons and SQFNNs with simple XOR gate learning, classifying nonlinear datasets and handwritten digit recognition. Our models show extraordinary generalization abilities and robustness to noise in the numerical simulations.

XOR gate learning

The XOR gate is a logic gate that cannot be simulated by classical perceptrons because the input–output relationship of the gate is nonlinear. Figure 4 reports the results of XOR gate learning with a soft quantum perceptron. Figure 4A and B shows the structure and setting of the soft quantum perceptron (see Methods for details). The results clearly show that the soft quantum perceptron is able to learn the data structure of the XOR gate with very high accuracy (Fig. 4C). Figure 4D shows the training process, where the training accuracy of our model converges quickly after even one epoch. These results show that our soft quantum perceptron has an extraordinary nonlinear classification ability.

In addition, we add the bit flip channel, the phase flip channel and the bit–phase flip channel to this task to further demonstrate the performance of our model on realistic quantum systems. We assume that each quantum neuron passes through the same type of quantum noise channel with probability p while waiting to be operated. To make the results more reliable, we repeat the prediction 100 times with the trained model and use the average accuracy as the evaluation metric. We set the highest noise level in the simulations to $p = 0.50$. Measurements in the simulations are calculated within the limit of the infinite shot number. Details of the simulation results can be found in Table 1 in Methods. The result shows that our model is robust to these different quantum channels. A remarkable result is that our model is fully tolerant to the phase flip channel for the XOR gate learning task. In particular, our model achieves up to 75% accuracy even with a probability of a bit flip or bit–phase flip up to 0.40. When the probability of a bit flip or bit–phase flip reaches 0.50, the noise makes qubits $|0\rangle$ and $|1\rangle$ completely indistinguishable. Our model also naturally does not work in this case, which is consistent with theoretical predictions.

Classifying nonlinear datasets

Two standard 2-dimensional datasets ("circles" and "moons") are studied to further demonstrate the ability of soft quantum perceptrons to classify nonlinear datasets and form decision boundaries (see Methods for details). For each dataset, 200 (100) points are generated as the training (test) set. Figure 5A visualizes the training set for the 2 datasets, where the red (blue) dots represent class 1 (class 2). Obviously, the 2 datasets are linearly inseparable. Figure 5B reports the results of classifying

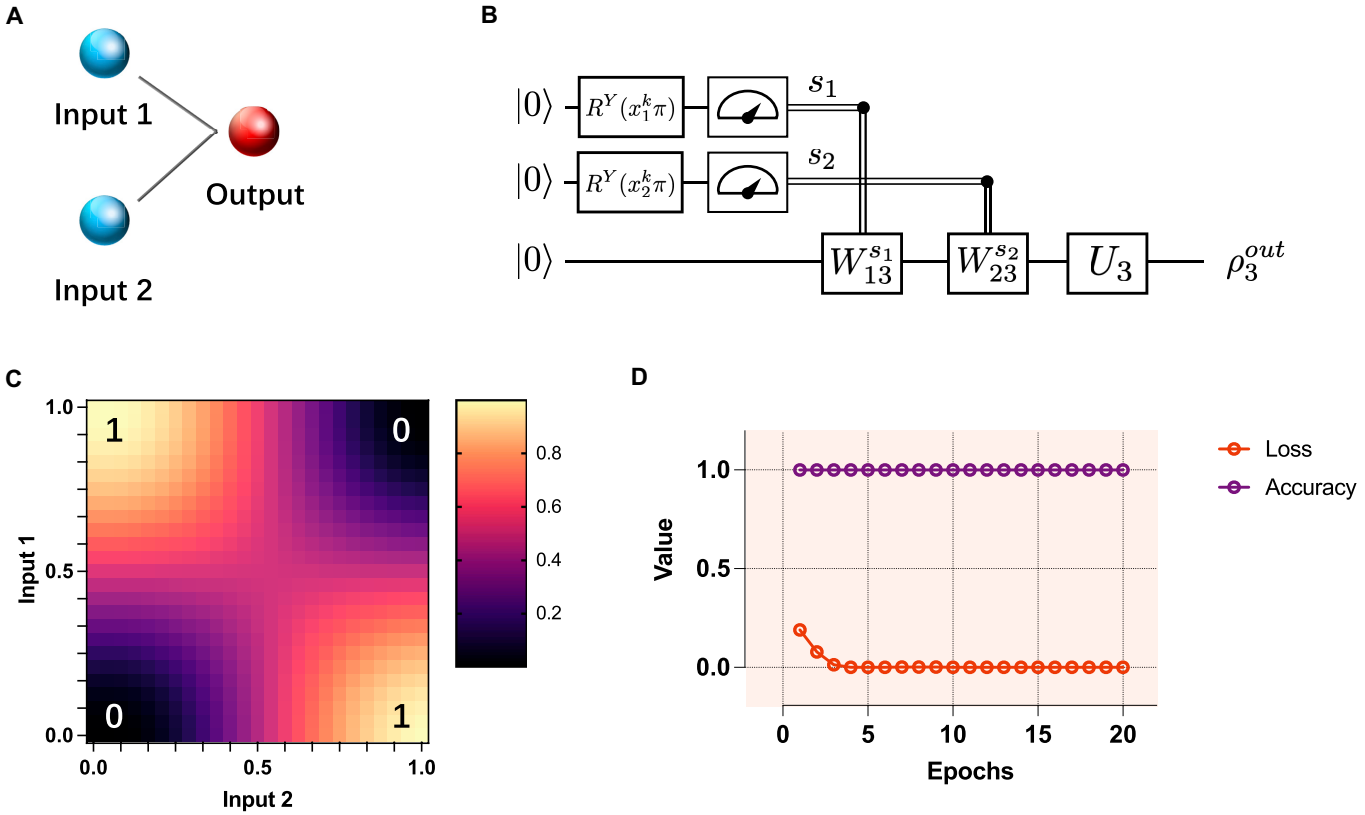

Fig. 4. Results of XOR gate learning with a soft quantum perceptron. (A) The structure of a soft quantum perceptron for learning the XOR gate. The input layer receives 2 features of the XOR gate, namely, input 1 and input 2 of XOR. The output layer predicts the results. (B) The quantum circuit model corresponding to (A). $R^Y(x_1^k\pi)$ and $R^Y(x_2^k\pi)$ are used to encode the input features (see Methods for details). $W_{13}^{s_1}$, $W_{23}^{s_2}$, and U_3 are single-qubit gates with parameters, where the values of the parameters converge during the learning process. (C) The simulation results of learning the XOR gate. The yellow (black) area represents an output of 1 (0), which is consistent with the truth table of the XOR gate. The true table of the XOR gate is displayed at the 4 corners of the figure. The soft quantum perceptron fully learns the data structure of the XOR gate. (D) The training process of the XOR gate. Loss (accuracy) is the value of the loss function (the test accuracy). The soft quantum perceptron achieves 100% test accuracy after the first epoch.

Table 1. Test accuracies of learning the XOR gate with the soft quantum perceptron after a bit flip channel, phase flip channel, or bit–phase flip channel with different flip probabilities.

Noise channels	Flip probability					
	0.10	0.20	0.30	0.35	0.40	0.50
Bit flip	100%	100%	100%	100%	75%	50%
Phase flip	100%	100%	100%	100%	100%	100%
Bit–phase flip	100%	100%	100%	100%	75%	50%

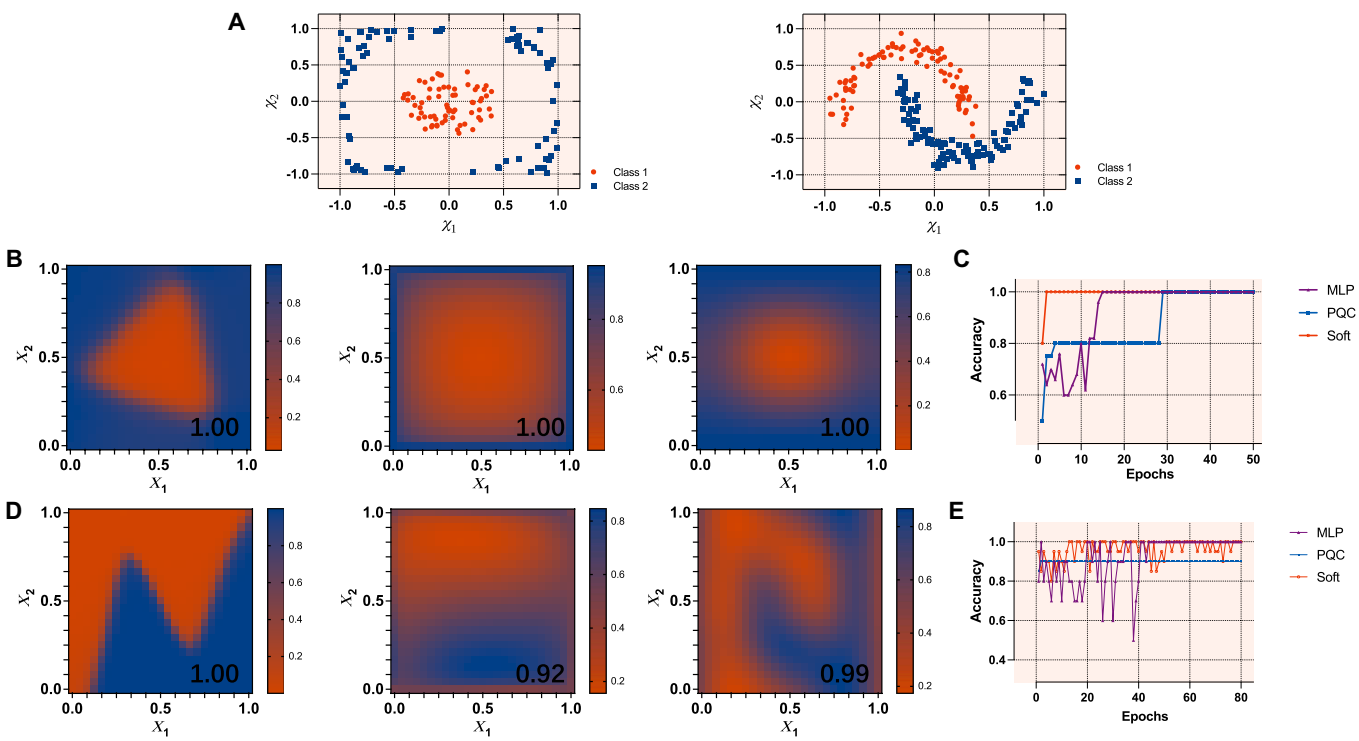

Fig. 5. Nonlinear decision boundaries by the classical MLP, the PQC model, and our model. (A) Displayed from left to right are the visualizations of the "circles" dataset and the "moons" dataset. X_1 (X_2) represents the horizontal (vertical) coordinate of the input point. The red (blue) dots represent class 1 (class 2). Both datasets are linearly inseparable. (B) Displayed from left to right are the simulation results for the classical multilayer perceptron (MLP), the PQC model, and our model. The classification accuracy is displayed at the bottom right corner of each subfigure. (C) The training process of learning the "circles" dataset with different models. The same results of the "moons" dataset are shown in (D) and (E).

the "circles" datasets with different models. Displayed from left to right are the simulation results for the classical multilayer perceptron (MLP), the parameterized quantum circuit (PQC) model, and our model. The settings of these models are discussed in detail in Methods. Figure 5C shows that all 3 models achieved 100% classification accuracy on the test set of "circles". However, the soft quantum perceptron converges faster and learns more robust decision bounds. It is worth reemphasizing that soft quantum perceptrons do not have hidden layers and do not require 2-qubit gates.

We also test the tolerance of soft quantum perceptrons for different noise types on this task (see Table 2 in Methods). The noise types are added in a manner consistent with the XOR gate learning task described above. The results show that the soft quantum perceptron maintains 100% accuracy on the test set of "circles", even when the probability of a bit flip or bit–phase flip is as high as 0.40. In particular, a soft quantum perceptron can achieve up to 96% accuracy when the probability of a bit flip is as high as 0.49. In addition, the soft quantum perceptron can maintain over 90% accuracy when the probability of a phase flip is as high as 0.50. We also found that the robustness of our model can be greatly enhanced when we use SQFNNs. For example, we obtain 100% accuracy when the probability of a phase flip is as high as 0.50 by adopting a 2-4-2-1 network structure. This suggests that the capabilities of our model can be enhanced by building more complex network structures, which provides strong confidence in handling more complex classification problems with our model.

Table 2. Test accuracies of learning the "circles" dataset with the soft quantum perceptron after a bit flip channel, phase flip channel, or bit–phase flip channel with different flip probabilities.

Noise channels	Flip probability				
	0.10	0.20	0.30	0.40	0.50
Bit flip	100%	100%	100%	100%	32%
Phase flip	100%	100%	93%	92%	91%
Bit–phase flip	100%	100%	100%	100%	68%

Figure 5D and E shows the results of classifying the "moons" datasets with different models. The MLP achieved 100% accuracy, which is slightly higher than the 99% accuracy of the soft quantum perceptron. However, the soft quantum perceptron learns a decision boundary that is better suited to the original data. For comparison, the PQC model can only achieve 92% accuracy. In the experimental setup currently used, our model shows absolute advantages over the PQC model in some tasks.

Handwritten digit recognition

Finally, we use QuantumFlow, the classical MLP, the PQC model, soft multioutput perceptron (SMP), and the SQFNN to recognize handwritten digits to demonstrate the ability of our models to solve specific practical problems (see Methods for details). QuantumFlow is a codesign framework of NNs and quantum circuits, and it can be used to design shallow networks that can be implemented on quantum computers [18]. SMP can also be regarded as an SQFNN without hidden layers. The simulation setting is discussed in Methods. Figure 6 shows the results of different classifiers for classifying different subdatasets from the Mixed National Institute of Standards and Technology (MNIST) database [44]. The results show that the classical MLP performs better than the other 4 quantum models on all these subdatasets except for {3, 9}. This may be caused by the fact that the classical optimization algorithm has better adaptability to the classical MLP model. Strikingly, the performance of our models (i.e., SMP and the SQFNN) is markedly better than that of QuantumFlow as the number of classes in the dataset increases, implying that our models may have more advantages in dealing with more complex classification problems. For the datasets with 2 or 3 classes, our models also perform markedly better than the PQC model and perform comparably to QuantumFlow. For example, the SQFNN achieves 89.67% accuracy on the {3, 8} dataset, which is 2.47% and 4.34% higher than those of QuantumFlow and the PQC model, respectively. However, our models require only classically controlled single-qubit operations and single-qubit operations, whereas QuantumFlow requires a large number of controlled 2-qubit gates or even Toffoli gates to implement the task. In particular, SMP is able to effectively classify handwritten digits with a structure without hidden layers, which is not possible in classical multioutput perceptrons.

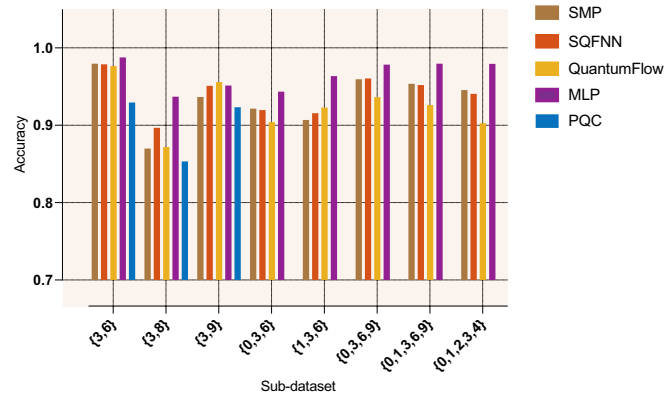

Fig. 6. Handwriting recognition with the classical MLP, the PQC model, QuantumFlow, and our models. The brown, orange, yellow, purple, and blue bars represent the soft multioutput perceptron (SMP), SQFNN, QuantumFlow, and the classical MLP and PQC models, respectively. SMP can also be regarded as an SQFNN without hidden layers and accurately identifies handwritten digits that are impossible for classical multioutput perceptrons.

Discussion

In this work, we develop a new routine for quantum NNs as a platform for quantum neural computing on real-world quantum systems. The proposed soft quantum neurons are subject merely to local or classically controlled single-qubit gates and single-qubit measurements. The simulation results show that soft quantum perceptrons have the ability, beyond that of classical perceptrons, of nonlinear classification. Furthermore, our model is able to classify handwritten digits with extraordinary generalization ability, even in the absence of hidden layers. This performance, combined with the quantum correlations arising from quantum discord in our model, makes it possible to perform nonclassical computations on realistic quantum devices that are extensible to a large scale. Thus, the proposed computing paradigm is not only physically easy to implement, but also predictably exciting beyond classical computing capabilities.

The soft quantum neurons are modeled as independent signal processing units and have more flexibility in the network architecture. Similar to classical perceptrons [6,7], such units can receive signals from any number of neurons and send their outputs to any number of neurons. This similar property allows our quantum NNs to take classical network architectures that have been proven effective, thereby exploiting the respective advantages of quantum technology and classical network architectures. For example, soft quantum neurons can be combined into quantum convolutional NNs based on convolutional NNs that are widely used in large-scale pattern recognition [30]. Moreover, our model enables the construction of quantum-classical hybrid NNs by introducing classical layers. As the final output of our quantum NN is the classical information, part of the classical information can also be processed by classical perceptrons. This advantage makes our model more flexible and thus more adaptable to various problems.

Our results provide an easier and more realistic route to quantum artificial intelligence. However, some limitations are worth noting. Although the quantum state space involved in computing the gradients in our model is always that of a single neuron, there may also be a barren plateau in the loss function landscape, which hinders the further optimization of the network. Additionally, while soft quantum NNs are much easier to build than standard ones, we need to do more work to understand what kinds of tasks they do well in learning. Future work should therefore include further research on optimization algorithms and building various soft quantum NNs inspired by classical architectures to solve problems that are intractable with classical models.

Methods

Soft quantum perceptron for XOR gate learning

We now discuss the details of the simulation setting for XOR gate learning. Figure 4A shows the model structure for learning an XOR gate, where 2 neurons in the input layer receive and encode data points, and the neuron in the output layer predicts the outcome. We adopt a simpler and more efficient angle encoding method instead of the method adopted in Ref. [45] to encode the data (Fig. 4B) that accelerates the convergence of the training process. Specifically, for an input set $\{x^k\}$, we encode the ith feature x_i^k of the kth data point by applying a single-qubit rotation gate $R^Y\left(x_i^k\pi\right)$ on the initial qubit $|0\rangle$, where Y represents the rotation along the Y axis and $x_i^k\pi$ represents the rotation angle. Note that a common MSE loss function and the Adam algorithm [43] are used in the training processes for all tasks in this study. The soft quantum perceptron for learning XOR is optimized for 20 epochs, and the learning rate is set to 0.1. Table 1 shows how the test accuracy of our model for the XOR gate learning task varies as the flip probability p increases.

Classifying nonlinear datasets

A 2-4-1 MLP structure is used for comparison in the task of classifying the "circles" dataset, as classical perceptrons are unable to classify nonlinear datasets. The reason for using this structure is that the 2-4-1 MLP needs to learn 17 parameters, which is approximately the same number of parameters that our model needs to learn. The structure of the PQC model used for comparison is adopted from Ref. [46]. This common layered PQC model is denoted as

$$U\left(\overline{\theta}\right) = B_d\left(\overline{\theta}_d\right)\cdots B_\ell\left(\overline{\theta}_\ell\right)\cdots B_1\left(\overline{\theta}_1\right) \tag{6}$$

where $\overline{\theta}$ represents the overall learnable parameters of the PQC, $B_\ell\left(\overline{\theta}_\ell\right)$ is a parameterized block consisting of a certain number of single-qubit gates and entangling controlled gates, and depth d represents the total number of such blocks. These qubits and controlled gates in the same block $B_\ell\left(\overline{\theta}_\ell\right)$ form a cyclic code. The control proximity range of a cyclic code, denoted as r, defines how the controlled gates work. For any qubit index $j \in [0, N-1]$ of an N-qubit circuit, the entangling code clock has one controlled gate with the jth qubit as the target and the qubit with the index $k = (j + r) \bmod (N)$ as the control qubit (see Ref. [46] for details). In each block $B_\ell\left(\overline{\theta}_\ell\right)$ of our setting, each qubit is acted on by a parameterized universal single-qubit gate. Then, the code block follows. One more optimizable single-qubit gate R^Y acts on each qubit in the final $B_\ell\left(\overline{\theta}_\ell\right)$. The control proximity range of a cyclic code r is fixed to 1. Specifically, a 2-qubit circuit of depth $d = 1$ and size $s = 6$ is used to classify the "circles" dataset, where d is the number of blocks $B_\ell\left(\overline{\theta}_\ell\right)$ and s is the total number of gates in the circuit other than in the encoding layer. In particular, the encoding method in Ref. [45] is also used in this PQC model for classifying nonlinear datasets. To enrich the expressivity of our model, we adopt the "parallel encodings" strategy mentioned in Ref. [47] when classifying the "moons" dataset, that is, using multiple neurons to repeatedly encode the same input in the input layer. In the task of classifying the "moons" dataset, we repeatedly encode each input with 3 neurons. For comparison, we also simulate the results of a 2-10-1 MLP and a 4-qubit PQC with $d = 2$ and $s = 24$. The MLP has 41 parameters to learn. The PQC model also adopts the "parallel encodings" strategy in this task. In particular, the soft quantum perceptron does not have hidden layers, so it is a simpler structure compared to the MLP. In fact, in addition to the results presented in the main text, we also found that a 4-qubit PQC with $d = 10$ and $s = 40$ could only achieve 94% accuracy when classifying the "moons" dataset. Table 2 shows how the test accuracy of our model for classifying the "circles" datasets varies as the flip probability p increases. Note that this effect is continuous, but the presentation of our results in a discrete table format may create an impression of discontinuity. Moreover, when the probability of bit flip or bit–phase flip reaches 0.50, the $|0\rangle$ and $|1\rangle$ components of the corresponding quantum state become indistinguishable in the computational basis, resulting in the inability to extract any relevant information. This causes a sudden drop in the probability of successful learning. This effect is particularly pronounced in close proximity to the 0.50 probability threshold.

Simulation setting for handwritten digit recognition

The specific simulation seting is as follows. First, we extract several subdatasets from MNIST. For example, $\{3, 6\}$ represents the subdataset containing 2 classes of the digits 3 and 6. After that, we apply the same downsampling size to all images from the same subdataset of MNIST. Specifically, we downsample the resolution of the original images from 28×28 to 4×4 for the datasets with 2 or 3 classes, and to 8×8 for datasets with 4 or 5 classes. Finally, we use the structure from Ref. [18] that contains a hidden layer for QuantumFlow, the classical MLP, and the SQFNN, where the hidden layer contains 4 neurons for 2-class datasets, 8 neurons for 3-class datasets, and 16 neurons for 4- and 5-class datasets. The input and output layers of these models (including SMP) are determined by the downsampling size and the number of digits in the subdatasets. Note that the PQC model is designed as a 4-qubit circuit of $d = 10$ and $s = 120$ due to the lack of the concept of neurons. The PQC model is usually used as a binary classifier in the current study. Therefore, the PQC model is only used to classify the datasets with 2 classes in this task. Other QuantumFlow settings, such as accuracy, are consistent with those in Ref. [18].

Acknowledgments

Funding: We gratefully acknowledge the support from the National Natural Science Foundation of China (No. 12274223), the Natural Science Foundation of Jiangsu Province (No. BK20211145), the Fundamental Research Funds for the Central Universities (No. 020414380182), the Key Research and Development Program of Nanjing Jiangbei New Aera (No. ZDYD20210101), and the Program for Innovative Talents and Entrepreneurs in Jiangsu (No. JSSCRC2021484). **Author contributions:** Z.-B.C. conceived and supervised the study. M.-G.Z., Z.-P.L., H.-L.Y., and Z.-B.C. built the theoretical model. M.-G.Z., Z.-P.L., and H.-L.Y. performed the simulations. M.-G.Z., Z.-P.L., H.-L.Y., and Z.-B.C. cowrote the manuscript, with inputs from the other authors. All authors have discussed the results and proofread the manuscript. **Competing interests:** The authors declare that they have no competing interests.

Data Availability

Data generated and analyzed during the current study are available from the corresponding author upon reasonable request.

References

1. Zadeh LA. Fuzzy logic, neural networks, and soft computing. *Commun ACM*. 1994;37(3):77–84.
2. Amit K. *Artificial intelligence and soft computing: Behavioral and cognitive modeling of the human brain*. Boca Raton (FL): CRC Press; 2018.
3. Nielsen MA, Chuang I. *Quantum computation and quantum information*. New York: Cambridge University Press; 2002.
4. Zhang Y, Qu P, Ji Y, Zhang W, Gao G, Wang G, Song S, Li G, Chen W, Zheng W, et al. A system hierarchy for brain-inspired computing. *Nature*. 2020;586:378–384.
5. Waldrop MM. The chips are down for Moore's law. *Nature News*. 2016;530:144–147.
6. Goodfellow I, Bengio Y, Courville A. *Deep learning*. Cambridge (MA): MIT Press; 2016.
7. Nielsen MA. Neural networks and deep learning. San Francisco (CA): Determination Press; 2015. vol. 25.
8. Jordan MI, Mitchell TM. Machine learning: Trends, perspectives, and prospects. *Science*. 2015;349:255–260.
9. Bishop CM, Nasrabadi NM. *Pattern recognition and machine learning*. New York: Springer; 2006. vol. 4.
10. Brown T, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, et al. Language models are few-shot learners. *Adv Neural Inf Process Syst*. 2020;33:1877–1901.
11. Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S. Quantum machine learning. *Nature*. 2017;549:195–202.
12. Schuld M, Sinayskiy I, Petruccione F. The quest for a quantum neural network. *Quantum Inf Process*. 2014;13:2567–2586.
13. Zhou M-G, Cao XY, Lu YS, Wang Y, Bao Y, Jia ZY, Fu Y, Yin HL, Chen ZB. Experimental quantum advantage with quantum coupon collector. *Research*. 2022;2022:9798679.
14. Wan KH, Dahlsten O, Kristjánsson H, Gardner R, Kim M. Quantum generalisation of feedforward neural networks. *npj Quantum Inf*. 2017;3:36.
15. Beer K, Bondarenko D, Farrelly T, Osborne TJ, Salzmann R, Scheiermann D, Wolf R. Training deep quantum neural networks. *Nat Commun*. 2020;11:808.
16. Bondarenko D, Feldmann P. Quantum autoencoders to denoise quantum data. *Phys Rev Lett*. 2020;124:Article 130502.
17. Cong I, Choi S, Lukin MD. Quantum convolutional neural networks. *Nat Phys*. 2019;15:1273–1278.
18. Jiang W, Xiong J, Shi Y. A co-design framework of neural networks and quantum circuits towards quantum advantage. *Nat Commun*. 2021;12:579.
19. McClean JR, Boixo S, Smelyanskiy VN, Babbush R, Neven H. Barren plateaus in quantum neural network training landscapes. *Nat Commun*. 2018;9:4812.
20. Farhi E, Neven H. Classification with quantum neural networks on near term processors. arXiv. 2018. https://doi.org/10.48550/arXiv.1802.06002
21. Sharma K, Cerezo M, Cincio L, Coles PJ. Trainability of dissipative perceptron-based quantum neural networks. *Phys Rev Lett*. 2022;128:Article 180505.
22. da Silva AJ, Ludermir TB, de Oliveira WR. Quantum perceptron over a field and neural network architecture selection in a quantum computer. *Neural Netw*. 2016;76:55–64.
23. Torrontegui E, Garcia-Ripoll JJ. Unitary quantum perceptron as efficient universal approximator. *Europhys Lett*. 2019;125(3):30004.
24. Herrmann J, Llima SM, Remm A, Zapletal P, McMahon NA, Scarato C, Swiadek F, Andersen CK, Hellings C, Krinner S, et al. Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases. *Nat Commun*. 2022;13:4144.
25. Huang H-Y, Broughton M, Cotler J, Chen S, Li J, Mohseni M, Neven H, Babbush R, Kueng R, Preskill J, et al. Quantum advantage in learning from experiments. *Science*. 2022;376:1182–1186.
26. Preskill J. Quantum computing in the NISQ era and beyond. *Quantum*. 2018;2:79.
27. Kjaergaard M, Schwartz ME, Braumüller J, Krantz P, Wang JIJ, Gustavsson S, Oliver WD. Superconducting qubits: Current state of play. *Annu Rev Condens Matter Phys*. 2020;11:369–395.
28. Ladd TD, Jelezko F, Laflamme R, Nakamura Y, Monroe C, O'Brien JL. Quantum computers. *Nature*. 2010;464:45–53.
29. Barends R, Kelly J, Megrant A, Veitia A, Sank D, Jeffrey E, White TC, Mutus J, Fowler AG, Campbell B, et al. Superconducting quantum circuits at the surface code threshold for fault tolerance. *Nature*. 2014;508:500–503.
30. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst*. 2012;25.
31. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. *IEEE Trans Neural Netw*. 2009;20(1):61–80.
32. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial nets. *Adv Neural Inf Process Syst*. 2014;27.
33. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9:1735–1780.
34. Ollivier H, Zurek WH. Quantum discord: A measure of the quantumness of correlations. *Phys Rev Lett*. 2001;88:Article 017901.
35. Streltsov A, Kampermann H, Bruß D. Behavior of quantum correlations under local noise. *Phys Rev Lett*. 2011;107:Article 170502.
36. Dakić B, Vedral V, Brukner Č. Necessary and sufficient condition for nonzero quantum discord. *Phys Rev Lett*. 2010;105:Article 190502.
37. Ciccarello F, Giovannetti V. Creating quantum correlations through local nonunitary memoryless channels. *Phys Rev A*. 2012;85:Article 010102.
38. Lanyon B, Jurcevic P, Hempel C, Gessner M, Vedral V, Blatt R, Roos CF. Experimental generation of quantum discord via noisy processes. *Phys Rev Lett*. 2013;111:Article 100504.
39. Wootters WK, Zurek WH. A single quantum cannot be cloned. *Nature*. 1982;299:802–803.
40. Arute F, Arya K, Babbush R, Bacon D, Bardin JC, Barends R, Biswas R, Boixo S, Brandao FGSL, Buell DA, et al. Quantum supremacy using a programmable superconducting processor. *Nature*. 2019;574:505–510.
41. Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. *J Mach Learn Res*. 2011;12:2121–2159.

42. Tieleman T, Hinton G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning. 2012;4:26–31.
43. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv. 2014. https://doi.org/10.48550/arXiv.1412.6980
44. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proc IEEE*. 1998;86:2278–2324.
45. Mitarai K, Negoro M, Kitagawa M, Fujii K. Quantum circuit learning. *Phys Rev A*. 2018;98:Article 032309.
46. Schuld M, Bocharov A, Svore KM, Wiebe N. Circuit-centric quantum classifiers. *Phys Rev A*. 2020;101:Article 032308.
47. Schuld M, Sweke R, Meyer JJ. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys Rev A*. 2021;103:Article 032430.