

Deep quantum neural networks equipped with backpropagation on a superconducting processor

Xiaoxuan Pan,^{1,*} Zhide Lu,^{1,*} Weiting Wang,¹ Ziyue Hua,¹ Yifang Xu,¹ Weikang Li,¹ Weizhou Cai,¹
Xuegang Li,¹ Haiyan Wang,¹ Yi-Pu Song,¹ Chang-Ling Zou,² Dong-Ling Deng,^{1,3,†} and Luyan Sun^{1,‡}

¹Center for Quantum Information, Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China

²CAS Key Laboratory of Quantum Information, University of Science and Technology of China, Hefei, Anhui 230026, China

³Shanghai Qi Zhi Institute, No. 701 Yunjin Road, Xuhui District, Shanghai 200232, China

Deep learning and quantum computing have achieved dramatic progresses in recent years. The interplay between these two fast-growing fields gives rise to a new research frontier of quantum machine learning. In this work, we report the first experimental demonstration of training deep quantum neural networks via the backpropagation algorithm with a six-qubit programmable superconducting processor. In particular, we show that three-layer deep quantum neural networks can be trained efficiently to learn two-qubit quantum channels with a mean fidelity up to 96.0% and the ground state energy of molecular hydrogen with an accuracy up to 93.3% compared to the theoretical value. In addition, six-layer deep quantum neural networks can be trained in a similar fashion to achieve a mean fidelity up to 94.8% for learning single-qubit quantum channels. Our experimental results explicitly showcase the advantages of deep quantum neural networks, including quantum analogue of the backpropagation algorithm and less stringent coherence-time requirement for their constituting physical qubits, thus providing a valuable guide for quantum machine learning applications with both near-term and future quantum devices.

Machine learning has achieved tremendous success in both commercial applications and scientific researches over the past decade. In particular, deep neural networks play a vital role in cracking some notoriously challenging problems, ranging from playing Go [1] to predicting protein structures [2]. They contain multiple hidden layers and are believed to be more powerful in extracting high-level features from data than traditional methods [3, 4]. The learning process can be fueled by updating the parameters through gradient descent, where the backpropagation algorithm enables efficient calculations of gradients via the chain rule [3].

By harnessing the weirdness of quantum mechanics such as superposition and entanglement, quantum machine learning approaches hold the potential to bring advantages compared with their classical counterpart. In recent years, exciting progress has been made along this interdisciplinary direction [5–10]. For example, rigorous quantum speedups have been proved in classification models [11] and generative models [12] with complexity-theoretic guarantees. In terms of the expressive power for quantum neural networks, there is also preliminary evidence showing their advantages over the comparable feedforward neural networks [13]. Meanwhile, noteworthy progress has also been made on the experimental side [14–22]. For examples, in Ref. [14], the authors realize a quantum convolutional neural network on a superconducting quantum processor. In Ref. [15], an experimental demonstration of quantum adversarial learning has been reported. Similar to deep classical neural networks with multiple layers, a deep quantum neural network (DQNN) with the layer-by-layer architecture is proposed [23, 24], which can be trained via a quantum analog of the backpropagation algorithm. Under this framework, the quantum analog of a perceptron is a general unitary operator acting on qubits from adjacent layers, whose parameters are updated by multiplying the corresponding updating matrix of the perceptron in the training process.

In this paper, we report the first experimental demonstra-

tion of training DQNNs through the backpropagation algorithm on a programmable superconducting processor with six frequency-tunable transmon qubits. We find that a three-layer DQNN can be efficiently trained to learn a two-qubit target quantum channel with a mean fidelity up to 96.0% and the ground state energy of molecular hydrogen with an accuracy up to 93.3% compared to the theoretical prediction. In addition, we also demonstrate that a six-layer DQNN can efficiently learn a one-qubit target quantum channel with a mean fidelity up to 94.8%. Our approach can carry over to other DQNNs with a larger width and depth straightforwardly, thus paving a way towards large-scale quantum machine learning with potential advantages in practical applications.

As sketched in Fig. 1(a), our DQNN has a layer-by-layer structure, and maps the quantum information layerwise from the input layer state ρ^{in} , through L hidden layers, to the output layer state ρ^{out} . Quantum perceptrons are the building blocks of the DQNN. As shown in Fig. 1(b), a single quantum perceptron is defined as a parameterized quantum circuit applied to the corresponding qubit pair at adjacent layers, which is directly implementable in experiments. A sequential combination of the quantum perceptrons constitutes the layerwise operation between adjacent layers. One of the key characteristics of the DQNN is the layer-by-layer quantum state mapping, allowing efficient training via the quantum backpropagation algorithm [23]. We sketch the general experimental training process in Fig. 1(c). When performing the quantum backpropagation algorithm, one only requires the information from adjacent two layers, rather than the full DQNN, to evaluate the gradients with respect to all parameters at these two layers. Such a backpropagation-equipped DQNN bears the following merit: it significantly reduces the requirements for the ability to maintain many coherent qubits, since qubits in each layer only need to keep their coherence for no more than the duration of two-layer operations regardless of the depth of the DQNN. This advantage makes it possible to real-

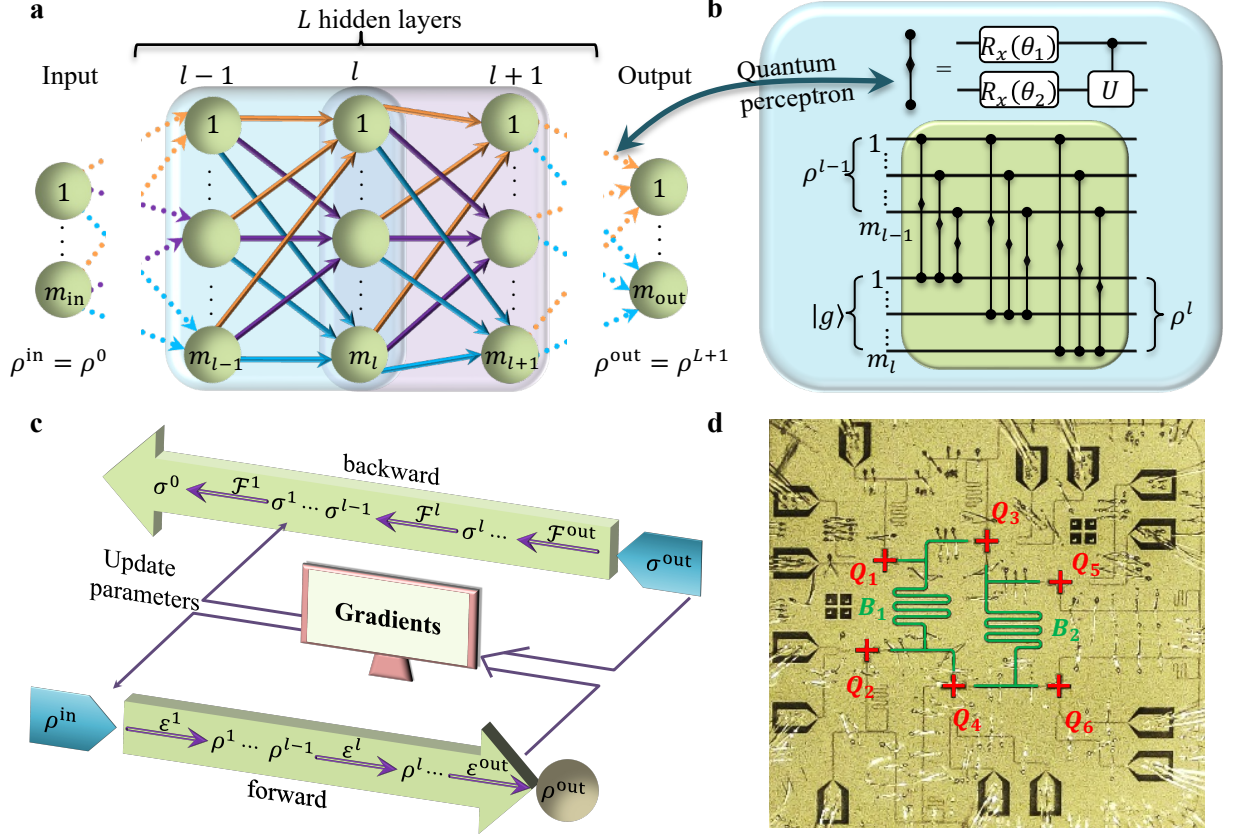


Fig. 1. **A schematic of training deep quantum neural networks.** (a), Architecture exhibition of a general DQNN. Information propagates layerwise from the input layer to the output layer. At adjacent two layers, we apply the quantum perceptron in the order according to the exhibited circuit in (b). A quantum perceptron is realized by applying two single-qubit rotation gates $R_x(\theta_1)$ and $R_x(\theta_2)$ (the rotations along the x axis with variational angles θ_1 and θ_2 , respectively) followed by a fixed two-qubit controlled-phase gate. (c), Illustration of the quantum backpropagation algorithm. We apply forward channels \mathcal{E} on ρ^{in} and successively obtain $\{\rho^1, \rho^2 \dots \rho^{out}\}$, and apply backward channels \mathcal{F} to successively obtain $\{\sigma^{out}, \sigma^L \dots \sigma^1\}$ in the backward process. These forward and backward terms are used for the gradient evaluation. (d), Exhibition of a quantum processor with six superconducting transmon qubits, which are used to experimentally implement the DQNNs. The transmon qubits (Q_1 - Q_6) are marked in red and the bus resonators (B_1 and B_2) are marked in green.

ize DQNNs with reduced number of layers of qubits through qubit reusing [23].

Our experiment is carried out on a superconducting quantum processor, which possesses six two-junction and frequency-tunable transmon qubits [25–32]. As photographed in Fig. 1(d), the chip is fabricated with the layout of the qubits being purposely and carefully optimized for a layer-by-layer structure. Each transmon qubit is coupled to an individual flux control line, XY control line, and quarter-wavelength readout resonator, respectively. All readout resonators are coupled to a common transmission line, which is connected through a Josephson parametric amplifier for high-fidelity single-shot readout of the qubits [33, 34]. In order to implement the two-qubit gates in the quantum perceptrons, two separate half-wavelength bus resonators are respectively used to mediate the interactions among the qubits between layers [16, 35, 36]. The detailed experimental setup and device parameters can be found in Supplementary Information.

We first consider using DQNNs to learn a two-qubit quan-

tum channel. We experimentally implement a three-layer DQNN with two qubits in each layer. This three-layer DQNN is denoted by DQNN₁. Here, we choose $|00\rangle$, $|01\rangle$, $|++\rangle$, and $|+i+i\rangle$ as our input states ρ_x^{in} , where the subscript $x = 1, 2, 3, 4$ is the labeling, $|0\rangle$ and $|1\rangle$ are the eigenstates of Pauli Z matrix, $|+\rangle$ ($|-\rangle$) is the eigenstate of Pauli X matrix, and $|+i\rangle$ is the eigenstate of Pauli Y matrix. The four pairs of $(\rho_x^{in}, \tau_x^{out})$ serve as the training dataset, where τ_x^{out} is the corresponding desired output state produced by the target quantum channel. We learn the target quantum channel by maximizing the mean fidelity between τ_x^{out} and the measured DQNN output ρ_x^{out} averaged over all four input states. The general training procedure goes as follows: 1) Initialization: we randomly choose the initial gate parameters θ for all perceptrons in DQNN₁. 2) Forward process (implemented on our quantum processor): for each training sample $(\rho_x^{in}, \tau_x^{out})$, we prepare the input layer to ρ_x^{in} , then apply layerwise forward channels \mathcal{E}^1 and \mathcal{E}^{out} , and extract ρ_x^1 and ρ_x^{out} successively by carrying out quantum state tomography [37]. 3) Backward process (im-

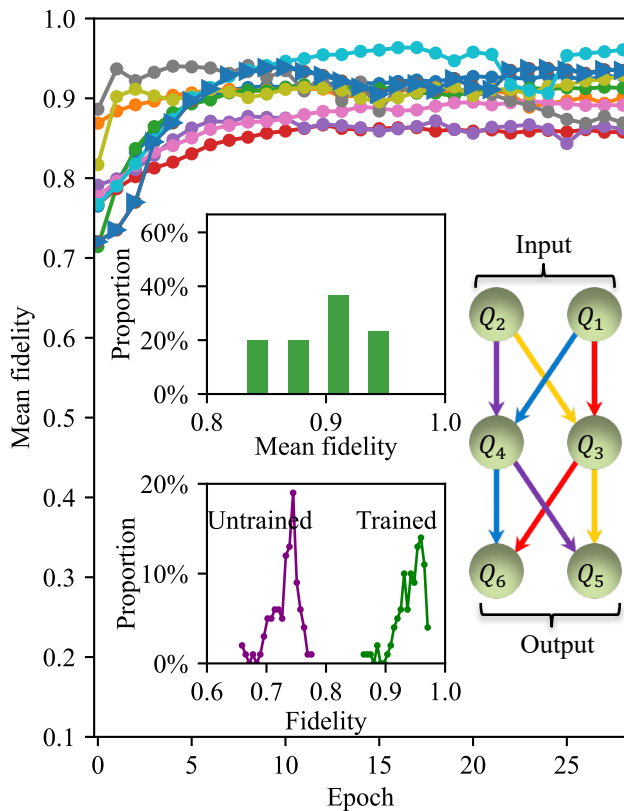


Fig. 2. Experimental results for learning a two-qubit quantum channel. We train the three-layer DQNN₁ with 30 different initial parameters, and plot the mean fidelity as a function of training epochs for 10 of them for clarity. The upper left inset shows the distribution of the converged mean fidelities of these 30 different initial parameters. We choose one of the learning curves (marked with dark blue triangles), then randomly generate 100 different input quantum states, and test the fidelity between their output states given by the target quantum channel and the trained (untrained) DQNN₁. In the lower left inset, the green (purple) curve shows the distribution of the fidelities for the trained (untrained) DQNN₁. The right inset is a schematic illustration of DQNN₁. At adjacent layers, we apply the quantum perceptrons in the order indicated with the colors: red, yellow, blue, and purple.

plemented on a classical computer): we initialize the output layer to σ_x^{out} , which is determined by ρ_x^{out} and τ_x^{out} (see Supplementary Note 1), and then apply backward channels \mathcal{F}^{out} and \mathcal{F}^1 on σ_x^{out} to successively obtain σ_x^1 and σ_x^0 . 4) Based on $\{(\rho_x^{l-1}, \sigma_x^l)\}$, we evaluate the gradient of the fidelity with respect to all the variational parameters in the adjacent layers $l-1$ and l . Then we take the average over the whole training dataset for the final gradient, which is used to update the variational parameters θ . 5) Repeat 2), 3), 4) for s_0 rounds. The pseudocode for our algorithm is provided in Supplementary Note 1.

In Fig. 2, we randomly choose 30 different initial parameters θ , and then train DQNN₁ to learn the same target quantum channel. We observe that DQNN₁ converges quickly during the training process, with the highest fidelity above 96%.

Compared with the numerical simulation results (see Supplementary Note 2), the deviation of the final converged fidelities is due to experimental imperfections, including qubit decoherence and residual ZZ interactions between qubits [38–40]. In the upper left inset of Fig. 2, we show the distribution for all the converged fidelities from these 30 repeated experiments. We expect that the distribution will concentrate to a higher fidelity for improved performance of the quantum processor.

To evaluate the performance of DQNN₁, we choose one training process from the 30 experiments, and refer the DQNN₁ with parameters corresponding to the ending (starting) epoch of the training curve as the trained (untrained) DQNN₁. We generate other 100 different input quantum states and experimentally measure their corresponding output states produced by the trained (untrained) DQNN₁. We test the fidelity between output states given by the target channel and the trained (untrained) DQNN₁. As shown in the lower left inset of Fig. 2, for the trained DQNN₁, 43% of the fidelities exceed 0.95 (green curve) and 95% of the fidelities are higher than 0.9, which separate away from the distribution of the results of the untrained DQNN₁ (purple curve). This contrast illustrates the effectiveness of the training process of DQNN₁.

Another application of DQNNs is learning the ground state energy of a given Hamiltonian H by minimizing the energy estimate $\text{tr}(\rho^{\text{out}}H)$ for the output state of the DQNN. Here we aim to learn the ground state energy of the molecular hydrogen Hamiltonian [41]. By exploiting the Bravyi-Kitaev transformation and certain symmetry, the Hamiltonian of molecular hydrogen can be reduced to the effective Hamiltonian acting on two qubits: $\hat{H}_{\text{BK}} = g_0\mathbf{I} + g_1Z_0 + g_2Z_1 + g_3Z_0Z_1 + g_4Y_0Y_1 + g_5X_0X_1$, where X_i, Y_i, Z_i are Pauli operators on the i -th qubit, and coefficients g_j ($j = 0, \dots, 5$) depend on the fixed bond length of molecular hydrogen. We consider the bond length 0.075 nm in this work and the corresponding coefficients g_i can be found in Ref. [41].

We use DQNN₁ again as the variational ansatz to learn the ground state of molecular hydrogen with the following procedure, similar to the previous one of learning a quantum channel: 1) Initialization: we prepare the input layer to the fiducial product state $|00\rangle$, and randomly generate initial gate parameters θ for DQNN₁. 2) In the forward process (implemented on the quantum processor), we apply forward channels \mathcal{E}^1 and \mathcal{E}^{out} in succession, and extract quantum states of the hidden layer (ρ^1) and the output layer (ρ^{out}) by quantum state tomography. 3) In the backward process (implemented on a classical computer), we initialize the quantum state of the output layer to σ^{out} , and then obtain σ^1 and σ^0 after successively applying backward channels \mathcal{F}^{out} and \mathcal{F}^1 on σ^{out} . 4) Based on $\{(\rho^{l-1}, \sigma^l)\}$, we calculate the gradient of the energy estimate with respect to all the variational parameters in the adjacent layers $l-1$ and l , and then update all gate parameters in DQNN₁. 5) Repeat 2), 3), 4) for s_0 rounds. The pseudocode for our algorithm is provided in Supplementary Note 1.

We train DQNN₁ with 30 different initial parameters and show our experimental results in Fig. 3(a). We observe that

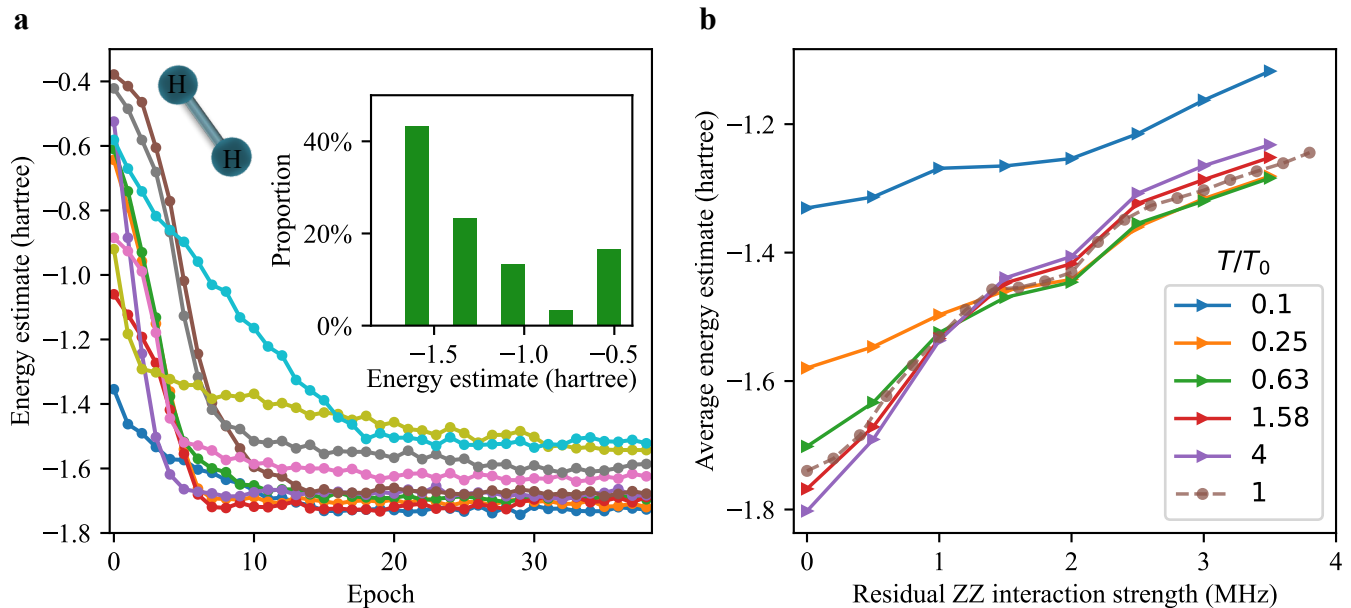


Fig. 3. **Experimental and numerical results for learning the ground state energy of molecular hydrogen.** (a), Experimental energy estimate at each epoch during the learning process for different initial parameters. The inset displays the distribution of converged energy estimates of 30 different initial parameters. (b), Numerical results for the mean energy estimates with different coherence times and residual ZZ interaction strengths between qubits. Specifically, we adjust both the energy relaxation time and the dephasing time with the same ratio (T/T_0), where T and T_0 are the coherence times in the simulation and the experiment respectively, and vary the residual ZZ interaction strengths.

DQNN₁ converges within 20 epochs. The lowest ansatz energy estimate reaches below -1.727 (hartree) in the learning process, with an accuracy up to 93.3% compared to the theoretical value of the ground state energy -1.851 (hartree). This shows the good performance of DQNN₁ and the accuracy of our experimental system control. The inset of Fig. 3(a) shows the distribution of all the converged energy from these 30 repeated experiments with different initial parameters, six of which have an accuracy above 90%.

To numerically investigate the effects of experimental imperfections on training DQNNs, we consider two possible sources of errors: decoherence of qubits and residual ZZ interactions between qubits. Taking into consideration these errors, we numerically train DQNN₁ with 30 different initial parameters. We find that for four of these initial parameters DQNN₁ converges to local minima instead of the global minimum, which is also observed in the experiment as shown in the inset of Fig. 3(a). Excluding these abnormal instances with local minima, we plot the average energy estimate as a function of the strength of the residual ZZ interaction with different coherence times in Fig. 3(b). We find that the increase of the coherence time around the experimental value has a minor effect on learning the ground state energy, while the reduction of the residual ZZ interactions provides larger improvements of the ground state energy estimation. These experimental imperfections can be suppressed after introducing advanced technologies in the design and fabrication of better superconducting quantum circuits, such as tunable couplers [42–44]

and tantalum based qubits [45, 46].

To further illustrate the efficiency of the quantum backpropagation algorithm, we construct another DQNN with four hidden layers (denoted as DQNN₂) by rearranging our six-qubit quantum processor into a six-layer structure, with one qubit respectively in each layer. We focus on the task of learning a one-qubit quantum channel. We choose $|0\rangle$, $|1\rangle$, $|-\rangle$ as our input states and compare the measured output states of DQNN₂ with the desired ones from the target single-qubit quantum channel. The general training procedure is similar as in training DQNN₁ discussed above. Our experimental results are summarized in Fig. 4, which shows the learning curves for 10 different initial parameters. We find that DQNN₂ can learn the target quantum channel with a mean fidelity up to 94.8%. We notice that the variance among the converged mean fidelity in DQNN₂ is smaller than that for DQNN₁, which may be attributed to the smaller total circuit depth and thus less error accumulation due to experimental imperfections. To study the learning performance, we choose one of these learning curves (marked in triangles), and refer DQNN₂ with parameters corresponding to the ending (starting) epoch of the learning curve as the trained (untrained) DQNN₂. We then use other 100 different input quantum states to test the trained and untrained DQNN₂ by measuring the fidelities between the experimental output states and the corresponding desired ones given by the target quantum channel. As shown in the upper inset of Fig. 4, the fidelity distribution concentrates around 0.92 for the trained DQNN₂, which stands in stark contrast to that of

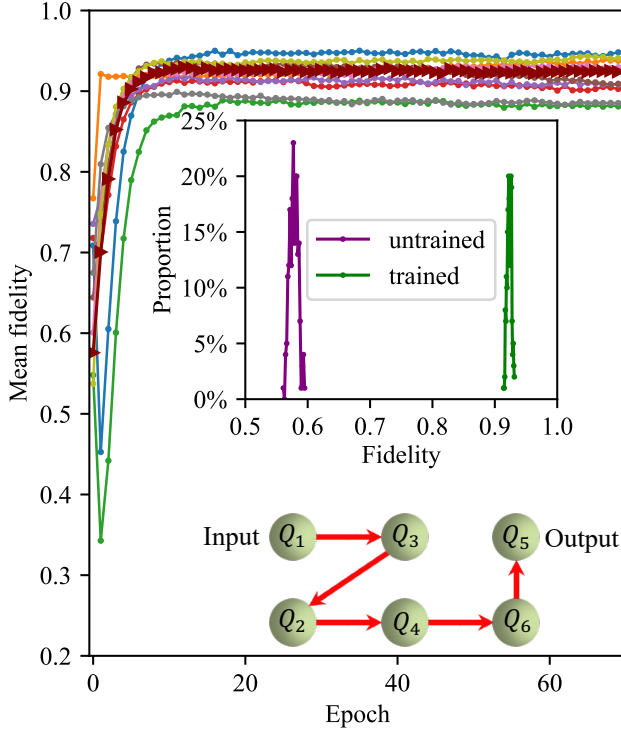


Fig. 4. **Experimental results for learning a one-qubit quantum channel.** The mean fidelity of training the six-layer DQNN₂ is plotted as the function of training epochs for different initial parameters. We randomly generate 100 different single-qubit states, and evaluate the fidelities between their output states produced by DQNN₂ and their desired output states given by the target quantum channel. The upper inset displays the distribution in two cases: a well-trained DQNN₂ (green) and an untrained (purple) DQNN₂, both are defined with the learning curve marked in triangles. The lower inset is a schematic illustration of DQNN₂, where we apply the perceptrons in the order indicated by the direction of the arrows.

the untrained DQNN₂ and thus indicates a good performance after training.

In summary, we have demonstrated the training of deep quantum neural networks on a six-qubit programmable superconducting quantum processor. We experimentally exhibit its intriguing ability to learn quantum channels and learn the ground state energy of a given Hamiltonian. The quantum backpropagation algorithm demonstrated in our experiments can be directly applied to DQNNs with extended widths and depths. This approach significantly reduces the requirements for the coherence time of superconducting qubits, regardless of how many hidden layers DQNNs include. With further improvements in experimental conditions, the quantum perceptrons in our DQNNs can be constructed with deeper circuits to improve the expressive capacity, which allows DQNNs to tackle more challenging tasks in the future.

Methods

Framework. We consider a deep quantum neural network (DQNN) that includes L hidden layers with a total number

m_l of qubits in layer l . The qubits in two adjacent layers are connected with quantum perceptrons and each perceptron consists of two single-qubit rotation gates $R_x(\theta_1)$ and $R_x(\theta_2)$ along the x axis with variational angles θ_1 and θ_2 , respectively, followed by a fixed two-qubit controlled-phase gate. The unitary of the quantum perceptron that acts on the i -th qubit at layer $l-1$ and the j -th qubit at layer l in the DQNN is written as $U_{(i,j)}^l(\theta_{(i,j),1}^l, \theta_{(i,j),2}^l)$. Then the unitary product of all quantum perceptrons acting on the qubits in layers $l-1$ and l is denoted as $U^l = \prod_{j=m_l}^1 \prod_{i=m_{l-1}}^1 U_{(i,j)}^l$. The DQNN acts on the input state ρ^{in} and produces the output state ρ^{out} according to

$$\rho^{\text{out}} \equiv \text{tr}_{\text{in,hid}} (\mathcal{U} (\rho^{\text{in}} \otimes |0 \cdots 0\rangle_{\text{hid,out}} \langle 0 \cdots 0|) \mathcal{U}^\dagger), \quad (1)$$

where $\mathcal{U} \equiv U^{\text{out}} U^L U^{L-1} \cdots U^1$ is the unitary of the DQNN, and all qubits in the hidden layers and the output layer are initialized to a fiducial product state $|0 \cdots 0\rangle$. The characteristic of the layer-by-layer architecture enables ρ^{out} to be expressed as a series of maps on ρ^{in} :

$$\rho^{\text{out}} = \mathcal{E}^{\text{out}} (\mathcal{E}^L (\cdots \mathcal{E}^2 (\mathcal{E}^1 (\rho^{\text{in}})) \cdots)), \quad (2)$$

where $\mathcal{E}^l(\rho^{l-1}) \equiv \text{tr}_{l-1} (U^l (\rho^{l-1} \otimes |0 \cdots 0\rangle_l \langle 0 \cdots 0|) U^{l\dagger})$ is the forward quantum channel.

In Supplementary Information, we prove that for the two machine learning tasks in our work, the derivative of the mean fidelity or the energy estimate with respect to $\theta_{(i,j),k}^l$ can be calculated with the information of layers $l-1$ and l , which can be written as $G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l)$ with $\boldsymbol{\theta}^l$ incorporating all parameters in layers $l-1$ and l . We note that $\rho^{l-1} = \mathcal{E}^{l-1} (\cdots \mathcal{E}^2 (\mathcal{E}^1 (\rho^{\text{in}})) \cdots)$ refers to the quantum state in layer $l-1$ in the forward process, and $\sigma^l = \mathcal{F}^{l+1} (\cdots \mathcal{F}^{\text{out}} (\cdots) \cdots)$ represents the backward term in layer l with \mathcal{F}^l being the adjoint channel of \mathcal{E}^l .

Generating random input quantum states. To evaluate the learning performance in the task of learning a target quantum channel, we need to generate many different input quantum states and test the fidelity between their output states produced by DQNN₁ and their desired output states given by the target quantum channel.

For the task of learning a two-qubit quantum channel, we generate these input quantum states by separately applying single-qubit rotation gates $R_{a_1}(\Omega_1) \otimes R_{a_2}(\Omega_2)$ on the two qubits initialized in $|00\rangle$. Here each rotation gate has a random rotation axis a_i in the x - y plane and a random rotation angle Ω_i .

For the task of learning a one-qubit quantum channel, we generate the input quantum states by applying single-qubit rotation gates $R_b(\Phi)$ on the input qubit initialized in $|0\rangle$ with a random rotation axis b in the x - y plane and a random rotation angle Φ .

Data availability The data for experimental results presented in the figures is provided in <https://>

[//github.com/luzd19/Deep-quantum-neural-networks_ equipped-with-backpropagation](https://github.com/luzd19/Deep-quantum-neural-networks_equipped-with-backpropagation)

Code Availability The codes for numerical simulations and the numerical results are available at https://github.com/luzd19/Deep-quantum-neural-networks_ equipped-with-backpropagation

Acknowledgement We thank Wenjie Jiang for helpful discussions. We acknowledge the support by the National Natural Science Foundation of China (Grants No. 92165209, No. 11925404, No. 11874235, No. 11874342, No. 11922411, No. 12061131011, No. T2225008, No. 12075128), the National Key Research and Development Program of China (Grants No. 2017YFA0304303), Key-Area Research and Development Program of Guangdong Province (Grant No. 2020B0303030001), Anhui Initiative in Quantum Information Technologies (AHY130200), China Postdoctoral Science Foundation (BX2021167), and Grant No. 2019GQG1024 from the Institute for Guo Qiang, Tsinghua University. D.-L. D. also acknowledges additional support from the Shanghai Qi Zhi Institute.

Author contributions X.P. carried out the experiments and analyzed the data with the assistance of Z.H. and Y.X.; L.S. directed the experiments; Z.L. formalized the theoretical framework and performed the numerical simulations under the supervision of D.-L.D.; W.L. and C.-L.Z. provided theoretical support; W.C. fabricated the parametric amplifier; W.W. and X.P. designed the devices; X.P. fabricated the devices with the assistance of W.W., H.W., and Y.-P.S.; Z.H., W.C., and X.L. provided further experimental support; X.P., Z.L., W.L., C.-L.Z., D.-L.D., and L.S. wrote the manuscript with feedback from all authors.

Competing interests All authors declare no competing interests.

* These two authors contributed equally to this work.

† E-mail: dldeng@tsinghua.edu.cn

‡ E-mail: luyansun@tsinghua.edu.cn

- [1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, Mastering the game of Go without human knowledge, *Nature* **550**, 354 (2017).
- [2] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, *et al.*, Highly accurate protein structure prediction with AlphaFold, *Nature* **596**, 583 (2021).
- [3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (The MIT Press, Cambridge, 2016).
- [4] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, *Nature* **521**, 436 (2015).
- [5] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature* **549**, 195 (2017).
- [6] 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).
- [7] S. Das Sarma, D.-L. Deng, and L.-M. Duan, Machine learning meets quantum physics, *Phys. Today* **72**, 48 (2019).
- [8] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, Challenges and opportunities in quantum machine learning, *Nat. Comput. Sci.* **2**, 567 (2022).
- [9] A. Dawid, J. Arnold, B. Reuena, A. Gresch, M. Płodzień, K. Donatella, K. A. Nicoli, P. Stornati, R. Koch, M. Büttner, *et al.*, Modern applications of machine learning in quantum sciences, [arXiv:2204.04198](https://arxiv.org/abs/2204.04198) (2022).
- [10] H.-Y. Huang, R. Kueng, and J. Preskill, Information-Theoretic Bounds on Quantum Advantage in Machine Learning, *Phys. Rev. Lett.* **126**, 190505 (2021).
- [11] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, *Nat. Phys.* **17**, 1013 (2021).
- [12] X. Gao, Z.-Y. Zhang, and L.-M. Duan, A quantum machine learning algorithm based on generative models, *Sci. Adv.* **4**, eaat9004 (2018).
- [13] 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).
- [14] J. Herrmann, S. M. Lima, A. Remm, P. Zapletal, N. A. McMahon, C. Scarato, F. Swiadek, C. K. Andersen, C. Hellings, S. Krinner, *et al.*, Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases, *Nat. Commun.* **13**, 4144 (2022).
- [15] W. Ren, W. Li, S. Xu, K. Wang, W. Jiang, F. Jin, X. Zhu, J. Chen, Z. Song, P. Zhang, *et al.*, Experimental quantum adversarial learning with programmable superconducting qubits, *Nat. Comput. Sci.* **2**, 711 (2022).
- [16] 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).
- [17] L. Hu, S.-H. Wu, W. Cai, Y. Ma, X. Mu, Y. Xu, H. Wang, Y. Song, D.-L. Deng, C.-L. Zou, *et al.*, Quantum generative adversarial learning in a superconducting quantum circuit, *Sci. Adv.* **5**, eaav2761 (2019).
- [18] C. Blank, D. K. Park, J.-K. K. Rhee, and F. Petruccione, Quantum classifier with tailored quantum kernel, *npj Quantum Inf.* **6**, 1 (2020).
- [19] Z. Li, X. Liu, N. Xu, and J. Du, Experimental Realization of a Quantum Support Vector Machine, *Phys. Rev. Lett.* **114**, 140504 (2015).
- [20] 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, *et al.*, Training of quantum circuits on a hybrid quantum computer, *Sci. Adv.* **5**, eaaw9918 (2019).
- [21] M. Gong, H.-L. Huang, S. Wang, C. Guo, S. Li, Y. Wu, Q. Zhu, Y. Zhao, S. Guo, H. Qian, *et al.*, Quantum Neuronal Sensing of Quantum Many-Body States on a 61-Qubit Programmable Superconducting Processor, [arXiv:2201.05957](https://arxiv.org/abs/2201.05957) (2022).
- [22] H.-Y. Huang, M. Broughton, J. Cotler, S. Chen, J. Li, M. Mohseni, H. Neven, R. Babbush, R. Kueng, J. Preskill, *et al.*, Quantum advantage in learning from experiments, *Science* **376**, 1182 (2022).
- [23] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, Training deep quantum neural networks, *Nat. Commun.* **11**, 808 (2020).
- [24] Z. Liu, L.-M. Duan, and D.-L. Deng, Solving quantum master equations with deep quantum neural networks, *Phys. Rev. Research* **4**, 013097 (2022).

- [25] J. Koch, T. M. Yu, J. M. Gambetta, A. A. Houck, D. I. Schuster, J. Majer, A. Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf, Charge insensitive qubit design from optimizing the cooper-pair box, *Phys. Rev. A* **76**, 042319 (2007).
- [26] R. Barends, J. Kelly, A. Megrant, D. Sank, E. Jeffrey, Y. Chen, Y. Yin, B. Chiaro, J. Mutus, C. Neill, *et al.*, Coherent Josephson Qubit Suitable for Scalable Quantum Integrated Circuits, *Phys. Rev. Lett.* **111**, 080502 (2013).
- [27] X. Li, Y. Ma, J. Han, T. Chen, Y. Xu, W. Cai, H. Wang, Y. P. Song, Z.-Y. Xue, Z.-q. Yin, and L. Sun, Perfect Quantum State Transfer in a Superconducting Qubit Chain with Parametrically Tunable Couplings, *Phys. Rev. Appl.* **10**, 054009 (2018).
- [28] W. Cai, J. Han, F. Mei, Y. Xu, Y. Ma, X. Li, H. Wang, Y. P. Song, Z.-Y. Xue, Z.-q. Yin, *et al.*, Observation of topological magnon insulator states in a superconducting circuit, *Phys. Rev. Lett.* **123**, 080501 (2019).
- [29] S. Kono, K. Koshino, D. Lachance-Quirion, A. F. van Loo, Y. Tabuchi, A. Noguchi, and Y. Nakamura, Breaking the trade-off between fast control and long lifetime of a superconducting qubit, *Nat. Commun.* **11**, 3683 (2020).
- [30] I. Carusotto, A. A. Houck, A. J. Kollár, P. Roushan, D. I. Schuster, and J. Simon, Photonic materials in circuit quantum electrodynamics, *Nat. Phys.* **16**, 268 (2020).
- [31] V. Negîrmeac, H. Ali, N. Muthusubramanian, F. Battistel, R. Sagastizabal, M. Moreira, J. Marques, W. Vlothuizen, M. Beekman, C. Zachariadis, *et al.*, High-Fidelity Controlled-Z Gate with Maximal Intermediate Leakage Operating at the Speed Limit in a Superconducting Quantum Processor, *Phys. Rev. Lett.* **126**, 220502 (2021).
- [32] A. Blais, A. L. Grimsmo, S. M. Girvin, and A. Wallraff, Circuit quantum electrodynamics, *Rev. Mod. Phys.* **93**, 025005 (2021).
- [33] T. Roy, S. Kundu, M. Chand, A. M. Vadiraj, A. Ranadive, N. Nehra, M. P. Patankar, J. Aumentado, A. A. Clerk, and R. Vijay, Broadband parametric amplification with impedance engineering: Beyond the gain-bandwidth product, *Appl. Phys. Lett.* **107**, 262601 (2015).
- [34] K. W. Murch, S. J. Weber, C. Macklin, and I. Siddiqi, Observing single quantum trajectories of a superconducting quantum bit, *Nature* **502**, 211 (2013).
- [35] J. Majer, J. M. Chow, J. M. Gambetta, J. Koch, B. R. Johnson, J. A. Schreier, L. Frunzio, D. I. Schuster, A. A. Houck, A. Wallraff, *et al.*, Coupling superconducting qubits via a cavity bus, *Nature* **449**, 443 (2007).
- [36] C. Song, K. Xu, H. Li, Y.-R. Zhang, X. Zhang, W. Liu, Q. Guo, Z. Wang, W. Ren, J. Hao, *et al.*, Generation of multicomponent atomic schrodinger cat states of up to 20 qubits, *Science* **365**, 574 (2019).
- [37] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2010).
- [38] P. Zhao, P. Xu, D. Lan, J. Chu, X. Tan, H. Yu, and Y. Yu, High-contrast zz interaction using superconducting qubits with opposite-sign anharmonicity, *Phys. Rev. Lett.* **125**, 200503 (2020).
- [39] J. Ku, X. Xu, M. Brink, D. C. McKay, J. B. Hertzberg, M. H. Ansari, and B. L. T. Plourde, Suppression of unwanted zz interactions in a hybrid two-qubit system, *Phys. Rev. Lett.* **125**, 200504 (2020).
- [40] A. Kandala, K. X. Wei, S. Srinivasan, E. Magesan, S. Carnevale, G. A. Keefe, D. Klaus, O. Dial, and D. C. McKay, Demonstration of a high-fidelity cnot gate for fixed-frequency transmons with engineered zz suppression, *Phys. Rev. Lett.* **127**, 130501 (2021).
- [41] P. J. J. O'Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, *et al.*, Scalable Quantum Simulation of Molecular Energies, *Phys. Rev. X* **6**, 031007 (2016).
- [42] X. Li, T. Cai, H. Yan, Z. Wang, X. Pan, Y. Ma, W. Cai, J. Han, Z. Hua, X. Han, *et al.*, Tunable coupler for realizing a controlled-phase gate with dynamically decoupled regime in a superconducting circuit, *Phys. Rev. Appl.* **14**, 024070 (2020).
- [43] M. C. Collodo, J. Herrmann, N. Lacroix, C. K. Andersen, A. Remm, S. Lazar, J.-C. Besse, T. Walter, A. Wallraff, and C. Eichler, Implementation of conditional phase gates based on tunable zz interactions, *Phys. Rev. Lett.* **125**, 240502 (2020).
- [44] Y. Sung, L. Ding, J. Braumüller, A. Vepsäläinen, B. Kannan, M. Kjaergaard, A. Greene, G. O. Samach, C. McNally, D. Kim, *et al.*, Realization of high-fidelity cz and zz -free $iswap$ gates with a tunable coupler, *Phys. Rev. X* **11**, 021058 (2021).
- [45] A. P. M. Place, L. V. H. Rodgers, P. Mundada, B. M. Smitham, M. Fitzpatrick, Z. Leng, A. Premkumar, J. Bryon, A. Vrajitoarea, S. Sussman, *et al.*, New material platform for superconducting transmon qubits with coherence times exceeding 0.3 milliseconds, *Nat. Commun.* **12**, 1779 (2021).
- [46] C. Wang, X. Li, H. Xu, Z. Li, J. Wang, Z. Yang, Z. Mi, X. Liang, T. Su, C. Yang, *et al.*, Towards practical quantum computers: transmon qubit with a lifetime approaching 0.5 milliseconds, *npj Quantum Inf.* **8**, 1 (2022).
- [47] R. Barends, C. M. Quintana, A. G. Petukhov, Y. Chen, D. Kafri, K. Kechedzhi, R. Collins, O. Naaman, S. Boixo, F. Arute, *et al.*, Diabatic gates for frequency-tunable superconducting qubits, *Phys. Rev. Lett.* **123**, 210501 (2019).
- [48] M. A. Rol, L. Ciorciaro, F. K. Malinowski, B. M. Tarasinski, R. E. Sagastizabal, C. C. Bultink, Y. Salathe, N. Haandbaek, J. Sedivy, and L. DiCarlo, Time-domain characterization and correction of on-chip distortion of control pulses in a quantum processor, *Appl. Phys. Lett.* **116**, 054001 (2020).

Supplementary Information: Deep quantum neural networks equipped with backpropagation on a superconducting processor

SUPPLEMENTARY NOTE 1: THEORETICAL DETAILS FOR DEEP QUANTUM NEURAL NETWORKS

In classical machine learning, deep neural networks are characterized by the ability to extract high-level features from data. With the rapid development in quantum machine learning [5–9], we expect a quantum generalization of a deep neural network architecture to bring promising insights. Recently, a deep quantum neural network (DQNN) and a quantum analog of the backpropagation algorithm have been proposed [23]. In this ansatz, the quantum analog of a perceptron is a unitary operation which acts on qubits in two adjacent layers. During the training process, the unitary operator of a quantum perceptron is updated by multiplying the corresponding updating matrix.

In this paper, we experimentally demonstrate the training of parameterized DQNNs with a superconducting quantum processor. Equipped with the backpropagation algorithm, we can efficiently calculate the gradients during the training process. Our scheme is feasible for the experimental implementation in the noisy intermediate scale quantum era. In this section, we will introduce the basic structures, optimization strategies, and training procedures for DQNNs.

Basic structures

As mentioned in the main text, our DQNNs have layer-by-layer structures, and qubits in two adjacent layers are connected with the quantum perceptrons. In our ansatz, the quantum perceptrons are engineered as parameterized quantum circuits. For simplicity in this paper, we consider that each quantum perceptron acts on only two qubits in two adjacent layers. The circuit structure of a quantum perceptron is composed of two single-qubit rotation gates $R_x(\theta_1)$ and $R_x(\theta_2)$ with θ_1 and θ_2 as the variational parameters, followed by a fixed two-qubit controlled-phase gate, which is shown in Fig. 1(b) in the main text. A sequential combination of the quantum perceptrons constitutes the layer-by-layer transition mapping between adjacent layers. In this way, the DQNN maps the information layerwise from the input layer to the output layer through hidden layers.

Now we consider a DQNN including L hidden layers. The total number of qubits in layer l is denoted as m_l . The unitary of a quantum perceptron which acts on the i -th qubit at layer $l - 1$ and the j -th qubit at layer l is written as $U_{(i,j)}^l(\theta_{(i,j),1}^l, \theta_{(i,j),2}^l)$, where $\theta_{(i,j),k}^l$ ($k = 1, 2$) denote the variational parameters of the two R_x gates in the quantum perceptron $U_{(i,j)}^l$. The unitary product of all quantum perceptrons acting on the qubits in layers $l - 1$ and l is denoted as:

$$U^l = \prod_{j=m_l}^1 \prod_{i=m_{l-1}}^1 U_{(i,j)}^l.$$

We note that qubits in layer l are initialized to a fiducial product state $|0 \cdots 0\rangle$, and then the quantum state ρ^l of the qubits in layer l can be written as the layer-by-layer transition mapping on ρ^{l-1} :

$$\rho^l = \mathcal{E}^l(\rho^{l-1}) \equiv \text{tr}_{l-1} \left(U^l (\rho^{l-1} \otimes |0 \cdots 0\rangle_l \langle 0 \cdots 0|) U^{l\dagger} \right). \quad (\text{S1})$$

In this way, the output state ρ^{out} can be expressed as a series of maps on ρ^{in} :

$$\rho^{\text{out}} = \mathcal{E}^{\text{out}} \left(\mathcal{E}^L \left(\dots \mathcal{E}^2 \left(\mathcal{E}^1 \left(\rho^{\text{in}} \right) \dots \right) \right) \right). \quad (\text{S2})$$

Optimization strategies

With the basic structures discussed above, now we can specify the learning tasks. In this paper, we consider two machine learning tasks. The first task is learning a target quantum channel. We expect the output state given by the DQNN to be as close as possible to the output state given by the target quantum channel for each input state. We aim to maximize the mean fidelity between output states given by the DQNN (ρ_x^{out}) and the target quantum channel (τ_x^{out}) averaged over N training data:

$$F = \frac{1}{N} \sum_{x=1}^N F_x(\rho_x^{\text{out}}, \tau_x^{\text{out}}) = \frac{1}{N} \sum_{x=1}^N \left[\text{tr} \sqrt{\sqrt{\tau_x^{\text{out}}} \rho_x^{\text{out}} \sqrt{\tau_x^{\text{out}}}} \right].$$

The second task is learning the ground state of a Hamiltonian H . We aim to minimize the energy estimate \bar{E} of this Hamiltonian computed with the DQNN output state ρ^{out} :

$$\bar{E} = \text{tr}(\rho^{\text{out}} H).$$

To maximize the mean fidelity or minimize the energy estimate, we adapt the gradient descent method. In the main text, we mention that our DQNNs with the layer-by-layer architecture allow the quantum backpropagation algorithm. Via this algorithm, one only requires the information from two adjacent layers to calculate the gradients with respect to all gate parameters at these two layers. In other words, the derivative of the mean fidelity or the energy estimate with respect to $\theta_{(i,j),k}^l$ can be written as $G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l)$, where $\boldsymbol{\theta}^l$ incorporates all gate parameters in layers $l-1$ and l .

Here, we derive the formula for $G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l)$. We first consider a function f with the form $f(\rho^{\text{out}}, X) = \text{tr}(X \rho^{\text{out}})$, where X is a Hermitian matrix related to specific tasks. The derivative of f with respect to $\theta_{(i,j),k}^l$ can be expressed as:

$$\begin{aligned} \frac{\partial f(\rho^{\text{out}}, X)}{\partial \theta_{(i,j),k}^l} &= G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l) \\ &= \text{tr} \left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} \left(\rho^{l-1} \otimes |0\rangle_l \langle 0| \right) U^{l\dagger} \left(\mathbb{I}_{l-1} \otimes \sigma^l \right) \right) + \text{h.c.}, \end{aligned} \quad (\text{S3})$$

where h.c. stands for the Hermitian conjugate of the preceding terms. We show the proof as follows:

Proof.

$$\begin{aligned} \frac{\partial f(\rho^{\text{out}}, X)}{\partial \theta_{(i,j),k}^l} &= G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l) = \text{tr} \left(\frac{\partial \rho^{\text{out}}}{\partial \theta_{(i,j),k}^l} X \right) \\ &= \text{tr} \left(\left(\text{tr}_{\text{in,hidden}} \left(U^{\text{out}} \dots U^{l+1} \frac{\partial U^l}{\partial \theta_{(i,j),k}^l} U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^\dagger \right) + \text{h.c.} \right) X \right) \\ &= \text{tr} \left(\left(U^{\text{out}} \dots U^{l+1} \frac{\partial U^l}{\partial \theta_{(i,j),k}^l} U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^\dagger \right) \cdot \left(\mathbb{I}_{\text{in,hidden}} \otimes X \right) \right) + \text{h.c.} \\ &= \text{tr} \left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} U^{(l-1)} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U^{(l-1)\dagger} \right. \\ &\quad \left. U^{(l)\dagger} U^{(l+1)\dagger} \dots U^{\text{out}\dagger} \left(\mathbb{I}_{\text{in,hidden}} \otimes X \right) U^{\text{out}} \dots U^{(l+1)} \right) + \text{h.c.} \\ &= \text{tr} \left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} \left(\text{tr}_{l,\dots,\text{out}}(T_1) \otimes |0 \dots 0\rangle_{l,\dots,\text{out}} \langle 0 \dots 0| \right) U^{l\dagger} \left(\mathbb{I}_{0,\dots,l-1} \otimes \text{tr}_{0,\dots,l-1}(T_2) \right) \right) + \text{h.c.} \\ &= \text{tr} \left(\left(\left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} \left(\text{tr}_{l,\dots,\text{out}}(T_1) \otimes |0\rangle_l \langle 0| \right) U^{l\dagger} \right) \otimes \mathbb{I}_{l+1,\dots,\text{out}} \right) \right. \\ &\quad \left. \left(\mathbb{I}_{0,\dots,l} \otimes |0\rangle_{l+1,\dots,\text{out}} \langle 0| \right) \left(\mathbb{I}_{0,\dots,l-1} \otimes \text{tr}_{0,\dots,l-1}(T_2) \right) \right) + \text{h.c.} \\ &= \text{tr} \left(\left(\left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} \left(\rho^{l-1} \otimes |0\rangle_l \langle 0| \right) U^{l\dagger} \right) \otimes \mathbb{I}_{l+1,\dots,\text{out}} \right) \left(\mathbb{I}_{l-1,l} \otimes |0\rangle_{l+1,\dots,\text{out}} \langle 0| \right) \left(\mathbb{I}_{l-1} \otimes \text{tr}_{0,\dots,l-1}(T_2) \right) \right) + \text{h.c.} \\ &= \text{tr} \left(\frac{\partial U^l}{\partial \theta_{(i,j),k}^l} \left(\rho^{l-1} \otimes |0\rangle_l \langle 0| \right) U^{l\dagger} \left(\mathbb{I}_{l-1} \otimes \sigma^l \right) \right) + \text{h.c.}, \end{aligned}$$

where $\mathcal{U} \equiv U^{\text{out}} U^L U^{L-1} \dots U^1$. We use the shorthands $T_1 = U^{(l-1)} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U^{(l-1)\dagger}$, and $T_2 = 1/(2^{\sum_{i=m_0}^{m_l-1}}) U^{(l+1)\dagger} \dots U^{\text{out}\dagger} \left(\mathbb{I}_{\text{in,hidden}} \otimes X \right) U^{\text{out}} \dots U^{(l+1)}$. We define $\rho^{l-1} = \text{tr}_{1,\dots,l-2,l,\dots,\text{out}}(T_1)$ as the quantum states of the qubits in layer $l-1$ in the forward process, and $\sigma^l = \text{tr}_{l+1,\dots,\text{out}} \left(\left(\mathbb{I}_l \otimes |0 \dots 0\rangle_{l+1,\dots,\text{out}} \langle 0 \dots 0| \right) \cdot \text{tr}_{1,\dots,l-1}(T_2) \right)$ as the backward term in layer l . From this formula we obtain the recursive relation between σ^{l-1} and σ^l :

$$\sigma^{l-1} = \mathcal{F}^l(\sigma^l) = \text{tr}_l \left(\left(\mathbb{I}_{l-1} \otimes |0\rangle_l \langle 0| \right) U^{l\dagger} \left(\mathbb{I}_{l-1} \otimes \sigma^l \right) U^l \right), \quad (\text{S4})$$

where \mathcal{F}^l is the adjoint channel of \mathcal{E}^l , and $\sigma^{\text{out}} = X$. From this recursive relation, we can obtain the backward terms layerwise from the output layer to the input layer in the backward process.

Specially, if the gate with parameter $\theta_{(i,j),k}^l$ in the DQNN is of the form $e^{-\frac{i}{2}\theta_{(i,j),k}^l P_n}$ with P_n belonging to the Pauli group, we can utilize the ‘‘parameter shift rule’’ to calculate the gradient of f :

$$\frac{\partial f(\rho, \sigma)}{\partial \theta_{(i,j),k}^l} = G(\theta^l, \rho^{l-1}, \sigma^l) = \frac{1}{2}(h_+ - h_-), \quad (\text{S5})$$

where $h_{\pm} = \text{tr} \left(U_{\pm}^l (\rho^{l-1} \otimes |0\rangle_l \langle 0|) U_{\pm}^{l\dagger} (\mathbb{I}_{l-1} \otimes \sigma^l) \right)$, and U_{\pm}^l denotes the unitary that replaces the parameter $\theta_{(i,j),k}^l$ in U^l with $\theta_{(i,j),k}^l \pm \frac{\pi}{2}$. We show the proof as follows:

Proof.

$$\begin{aligned} 2 \cdot \frac{\partial f(\rho^{\text{out}}, X)}{\partial \theta_{(i,j),k}^l} &= 2 \cdot G(\theta^l, \rho^{l-1}, \sigma^l) = 2 \cdot \text{tr} \left(\frac{\partial \rho^{\text{out}}}{\partial \theta_{(i,j),k}^l} X \right) \\ &= \text{tr} \left(\left(\text{tr}_{\text{in,hidden}} \left(U^{\text{out}} \dots U^{l+1} U_+^l U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U_+^{l\dagger} \dots U^{\text{out}\dagger} \right) \right) X \right) \\ &\quad - \text{tr} \left(\left(\text{tr}_{\text{in,hidden}} \left(U^{\text{out}} \dots U^{l+1} U_-^l U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U_-^{l\dagger} \dots U^{\text{out}\dagger} \right) \right) X \right). \end{aligned}$$

Now we prove the first term equals to h_+ . In the same way, we can prove the second term equals to h_- . The first term can be written as:

$$\begin{aligned} &\text{tr} \left(\left(\text{tr}_{\text{in,hidden}} \left(U^{\text{out}} \dots U^{l+1} U_+^l U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U_+^{l\dagger} \dots U^{\text{out}\dagger} \right) \right) X \right) \\ &= \text{tr} \left(\left(U^{\text{out}} \dots U^{l+1} U_+^l U^{l-1} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U_+^{l\dagger} \dots U^{\text{out}\dagger} \right) \left(\mathbb{I}_{\text{in,hidden}} \otimes X \right) \right) \\ &= \text{tr} \left(\left(U_+^l U^{(l-1)} \dots U^1 \rho^{\text{in}} \otimes |0 \dots 0\rangle_{\text{hid,out}} \langle 0 \dots 0| U^{1\dagger} \dots U^{(l-1)\dagger} U_+^{l\dagger} U^{(l+1)\dagger} \dots U^{\text{out}\dagger} \right. \right. \\ &\quad \left. \left. \left(\mathbb{I}_{\text{in,hidden}} \otimes X \right) U^{\text{out}} \dots U^{(l+1)} \right) \right) \\ &= \text{tr} \left(U_+^l \left(\text{tr}_{l,\dots,\text{out}} (T_1) \otimes |0 \dots 0\rangle_{l,\dots,\text{out}} \langle 0 \dots 0| \right) U_+^{l\dagger} \left(\mathbb{I}_{0,\dots,l-1} \otimes \text{tr}_{0,\dots,l-1} (T_2) \right) \right) \\ &= \text{tr} \left(\left(\left(U_+^l \left(\text{tr}_{l,\dots,\text{out}} (T_1) \otimes |0\rangle_l \langle 0| \right) U_+^{l\dagger} \right) \otimes \mathbb{I}_{l+1,\dots,\text{out}} \right) \left(\mathbb{I}_{0,\dots,l} \otimes |0\rangle_{l+1,\dots,\text{out}} \langle 0| \right) \right. \\ &\quad \left. \left(\mathbb{I}_{0,\dots,l-1} \otimes \text{tr}_{1,\dots,l-1} (T_2) \right) \right) \\ &= \text{tr} \left(\left(\left(U_+^l (\rho_{l-1} \otimes |0\rangle_l \langle 0|) U_+^{l\dagger} \right) \otimes \mathbb{I}_{l+1,\dots,\text{out}} \right) \left(\mathbb{I}_{l-1,l} \otimes |0\rangle_{l+1,\dots,\text{out}} \langle 0| \right) \left(\mathbb{I}_{l-1} \otimes \text{tr}_{0,\dots,l-1} (T_2) \right) \right) \\ &= \text{tr} \left(U_+^l (\rho_{l-1} \otimes |0\rangle_l \langle 0|) U_+^{l\dagger} (\mathbb{I}_{l-1} \otimes \sigma_l) \right) \\ &= h_+. \end{aligned}$$

With the gradients of f obtained above, we need to derive the gradients of the mean fidelity F and the energy estimate \bar{E} in the two tasks that are discussed in the main text. In the task of learning a target quantum channel, for each input state, we consider the derivative of F_x with respect to $\theta_{(i,j),k}^l$. For convenience, we omit the superscript and subscript of F_x , ρ_x^{out} , and τ_x^{out} , and use the shorthand $A = \tau^{1/2} \rho \tau^{1/2}$, $B = \sqrt{A}$, then

$$\frac{\partial F}{\partial \theta_{(i,j),k}^l} = \text{tr} \left(\frac{\partial B}{\partial \theta_{(i,j),k}^l} \right).$$

Now, we further omit the superscript and subscript of $\theta_{(i,j),k}^l$.

$$\frac{\partial A}{\partial \theta} = \frac{\partial (B^2)}{\partial \theta} = B \cdot \frac{\partial B}{\partial \theta} + \frac{\partial B}{\partial \theta} \cdot B \Rightarrow \frac{\partial A}{\partial \theta} \cdot B^{-1} = B \cdot \frac{\partial B}{\partial \theta} B^{-1} + \frac{\partial B}{\partial \theta},$$

hence,

$$\text{tr} \left(\frac{\partial A}{\partial \theta} \cdot B^{-1} \right) = \text{tr} \left(B \cdot \frac{\partial B}{\partial \theta} B^{-1} \right) + \text{tr} \left(\frac{\partial B}{\partial \theta} \right) = \text{tr} \left(\frac{\partial B}{\partial \theta} B^{-1} B \right) + \text{tr} \left(\frac{\partial B}{\partial \theta} \right) = 2 \text{tr} \left(\frac{\partial B}{\partial \theta} \right).$$

This yields

$$\frac{\partial F}{\partial \theta} = \frac{1}{2} \text{tr} \left(\frac{\partial A}{\partial \theta} \cdot B^{-1} \right) = \frac{1}{2} \text{tr} \left(\tau^{1/2} \frac{\partial \rho}{\partial \theta} \tau^{1/2} \cdot B^{-1} \right) = \frac{1}{2} \text{tr} \left(\frac{\partial \rho}{\partial \theta} \cdot \tau^{1/2} B^{-1} \tau^{1/2} \right),$$

which has the same form as the derivative of $\text{tr}(\rho^{\text{out}} X)$ with $\tau^{1/2} B^{-1} \tau^{1/2}$ analogous to X . In the task of learning the ground state energy of a Hamiltonian H , the energy estimate $\text{tr}(\rho^{\text{out}} H)$ has the same form as $\text{tr}(\rho^{\text{out}} X)$, where H is analogous to X . So we can derive the gradients of the energy estimate \bar{E} according to $G(\theta^l, \rho^{l-1}, \sigma^l)$. With the gradients obtained, we can update the variational parameters in the DQNN by gradient descent methods.

Training procedures

In this section, we give a detailed description of how our DQNNs are trained via the quantum backpropagation algorithm for different tasks.

For the task of learning a quantum channel, first we need to generate the training dataset. Here, we randomly choose parameters θ_t in the DQNN to generate a specific target quantum channel that we aim to learn. Then we apply the target quantum channel on each input state to obtain the corresponding output state to constitute the training dataset $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}_{x=1}^N$ with N being the size of the training dataset. We assume the DQNN used in this task includes L hidden layers with a total number m_l of qubits in layer l . Now we describe the general training procedure as follows:

1. Initialization:

Randomly choose initial gate parameters for all perceptrons in the DQNN, which is denoted as θ_I .

2. Forward process:

For each training data $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}$, apply forward channels $\mathcal{E}^1, \mathcal{E}^2, \dots, \mathcal{E}^{\text{out}}$ on ρ_x^{in} to obtain $\rho_x^1, \rho_x^2, \dots, \rho_x^{\text{out}}$ successively.

Forward channel \mathcal{E}^l : According to the main text, the forward channel \mathcal{E}^l applies on qubits in layer $l-1$ of the quantum state ρ^{l-1} , and produces ρ^l in layer l according to $\rho^l = \mathcal{E}^l(\rho^{l-1}) \equiv \text{tr}_{l-1} \left(U^l (\rho^{l-1} \otimes |0 \dots 0\rangle_l \langle 0 \dots 0|) U^{l\dagger} \right)$. In our experiment, we prepare m_l qubits in layer l to the fiducial product state $|0 \dots 0\rangle$ at first. Then we apply all quantum perceptrons acting on qubits in layers $l-1$ and l . Finally, we carry out quantum state tomography to extract ρ^l .

3. Backward process:

For each training data $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}$, calculate $\sigma^{\text{out}} = (\tau_x^{\text{out}})^{1/2} ((\tau_x^{\text{out}})^{1/2} \rho_x^{\text{out}} (\tau_x^{\text{out}})^{1/2})^{-1/2} (\tau_x^{\text{out}})^{1/2}$, and then apply backward channels $\mathcal{F}^{\text{out}}, \mathcal{F}^L, \dots, \mathcal{F}^1$ on σ^{out} to successively obtain $\sigma_x^L, \sigma_x^{L-1}, \dots, \sigma_x^0$.

Backward channel \mathcal{F}^l : The backward channel \mathcal{F}^l applies on backward term σ^l and produces σ^{l-1} according to $\sigma^{l-1} = \mathcal{F}^l(\sigma^l) = \text{tr}_l \left((\rho_{l-1} \otimes |0\rangle_l \langle 0|) U^{l\dagger} (\mathbb{I}_{l-1} \otimes \sigma^l) U^l \right)$. In this paper, we carry out the backward channel on a classical computer due to the experimental challenges in preparing the quantum states for the backward terms σ^l . We expect an efficient proposal for the experimental implementation of the backward process, which is important and remains as a future work.

4. Evaluate the mean fidelity and the gradients:

Compute the mean fidelity:

$$F = \frac{1}{N} \sum_{x=1}^N F_x(\rho_x^{\text{out}}, \tau_x^{\text{out}}) = \frac{1}{N} \sum_{x=1}^N \left[\text{tr} \sqrt{\sqrt{\tau_x^{\text{out}}} \rho_x^{\text{out}} \sqrt{\tau_x^{\text{out}}}} \right].$$

Calculate the gradient with respect to $\theta_{(i,j),k}^l$ for each training data: $\frac{\partial F_x(\rho_x^{\text{out}}, \tau_x^{\text{out}})}{\partial \theta_{(i,j),k}^l} = G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l)$, and then take the average over the whole training dataset: $\frac{1}{N} \sum_{x=1}^N G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l)$. Finally, update each $\theta_{(i,j),k}^l$ with the learning rate ϵ according to

$$\theta_{(i,j),k}^l \rightarrow \theta_{(i,j),k}^l + \epsilon * \frac{1}{N} \sum_{x=1}^N G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l).$$

5. Repeat 2, 3 and 4 for s_0 steps.

We summarize the pseudocode in Algorithm 1.

Algorithm 1 Training the DQNN for learning quantum channels via the quantum backpropagation algorithm

Input The DQNN model with L hidden layers, initial parameters $\boldsymbol{\theta}_I$, input quantum states $\{(\rho_x^{\text{in}})\}_{x=1}^N$, iteration steps s_0 , learning rate ϵ ,

Output The trained DQNN

Generate the training dataset: choose parameters $\boldsymbol{\theta}_t$ for the DQNN, which serves as the target quantum channel, and then apply it to each input state to obtain the corresponding output state, which constitute the training dataset $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}_{x=1}^N$.

for $s = 1$ to s_0 **do**

Forward: for each training data $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}$, apply forward channels $\mathcal{E}^1, \mathcal{E}^2, \dots, \mathcal{E}^{\text{out}}$ on ρ_x^{in} to obtain $\rho_x^1, \rho_x^2, \dots, \rho_x^{\text{out}}$ successively.

Backward: for each training data $\{(\rho_x^{\text{in}}, \tau_x^{\text{out}})\}$, calculate $\sigma^{\text{out}} = (\tau_x^{\text{out}})^{1/2} ((\tau_x^{\text{out}})^{1/2} \rho_x^{\text{out}} (\tau_x^{\text{out}})^{1/2})^{-1/2} (\tau_x^{\text{out}})^{1/2}$, and then apply backward channels $\mathcal{F}^{\text{out}}, \mathcal{F}^L, \dots, \mathcal{F}^1$ on σ^{out} to successively obtain $\sigma_x^L, \sigma_x^{L-1}, \dots, \sigma_x^0$.

Gradients: calculate the gradient with respect to $\theta_{(i,j),k}^l$ for each training data: $\frac{\partial F_x(\rho_x^{\text{out}}, \tau_x^{\text{out}})}{\partial \theta_{(i,j),k}^l} = G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l)$, and then take the

average over the whole training dataset: $\frac{1}{N} \sum_{x=1}^N G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l)$.

Update: update each $\theta_{(i,j),k}^l$ according to $\theta_{(i,j),k}^l \rightarrow \theta_{(i,j),k}^l + \epsilon * \frac{1}{N} \sum_{x=1}^N G(\boldsymbol{\theta}^l, \rho_x^{l-1}, \sigma_x^l)$.

end for

Output the trained DQNN

For the task of learning the ground state energy of a Hamiltonian H , we provide the pseudocode in Algorithm 2.

Algorithm 2 Training the DQNN for learning the ground state for some Hamiltonian via the quantum backpropagation algorithm

Input The DQNN model with L hidden layers, initial parameters $\boldsymbol{\theta}_I$, Hamiltonian H , iteration steps s_0 , learning rate ϵ .

Output The trained DQNN

for $s = 1$ to s_0 **do**

Forward: apply forward channels $\mathcal{E}^1, \mathcal{E}^2, \dots, \mathcal{E}^{\text{out}}$ on initial fiducial product state $|0 \dots 0\rangle$ to obtain $\rho^1, \rho^2, \dots, \rho^{\text{out}}$ successively.

Backward: apply backward channels $\mathcal{F}^{\text{out}}, \mathcal{F}^L, \dots, \mathcal{F}^1$ on $\sigma^{\text{out}} = H$ to successively obtain $\sigma^L, \sigma^{L-1}, \dots, \sigma^0$.

Gradients: calculate the gradient with respect to $\theta_{(i,j),k}^l$: $\frac{\partial \bar{E}(\rho^{\text{out}}, H)}{\partial \theta_{(i,j),k}^l} = G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l)$.

Update: update each $\theta_{(i,j),k}^l$ according to $\theta_{(i,j),k}^l \rightarrow \theta_{(i,j),k}^l - \epsilon * G(\boldsymbol{\theta}^l, \rho^{l-1}, \sigma^l)$.

end for

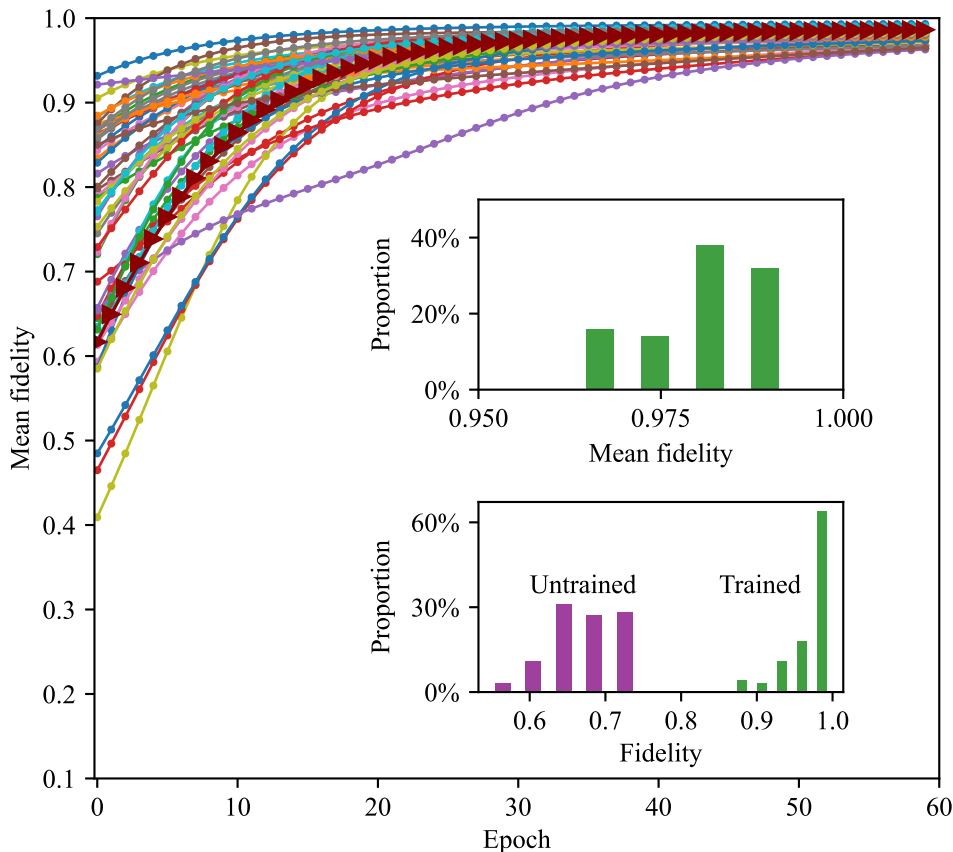
Output the trained DQNN

SUPPLEMENTARY NOTE 2: NUMERICAL RESULTS FOR SEVERAL MACHINE LEARNING TASKS

In this section, we simulate the training of DQNNs by realizing the forward channels and the backward channels with matrix calculations on a classical computer, and present some numerical results.

Task: learning a two-qubit quantum channel. Here, we choose DQNN₁ mentioned in the main text to learn a two-qubit target quantum channel. The training dataset is the same as that in the main text. We numerically train DQNN₁ with 50 different parameters and show our numerical results in Supplementary Fig. S1. We observe that DQNN₁ shows high convergence performance, with the average converged mean fidelity above 98%.

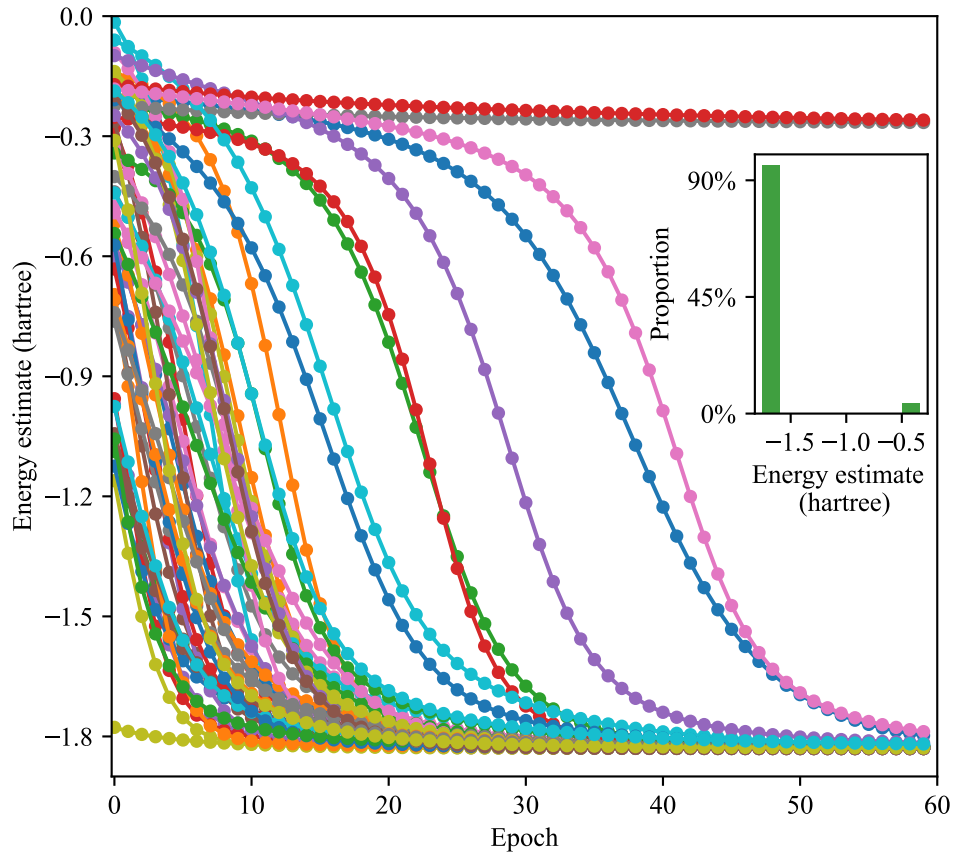
We choose one learning curve (marked in triangles in Supplementary Fig. S1) to test the learning performance of DQNN₁. We refer DQNN₁ with parameters corresponding to the ending (starting) epoch of this training curve as the trained (untrained) DQNN₁, and then use 100 different input quantum states to test the fidelities between their corresponding output states and the desired output states given by the target quantum channel. As shown in the lower inset of Supplementary Fig. S1, for the trained DQNN₁, the mean fidelity exceeds 0.97 (green bars), which separates away from the distribution of the results of the untrained DQNN₁ (purple bars). This contrast indicates a satisfying performance of DQNN₁.



Supplementary Figure S1. **Numerical results for learning a two-qubit quantum channel.** The numerical results for training DQNN₁ with 50 different initial parameters. We plot the mean fidelity as a function of the training epochs. The upper inset shows the distribution of the converged mean fidelities for these 50 different initial parameters. We choose one of the learning curves (marked with triangles), and then randomly generate 100 different input quantum states to test the fidelities between their output states given by the target quantum channel and the trained (untrained) DQNN₁. The results are displayed in the lower inset with the green (purple) bars showing the distribution of the fidelities for the trained (untrained) DQNN₁.

Task: learning the ground state of molecular hydrogen (H_2). We also use DQNN₁ to learn the ground state energy of the molecular hydrogen Hamiltonian. We choose 50 different initial parameters and classically simulate the training process as presented in the main text. The results are shown in Supplementary Fig. S2. We observe that DQNN₁ converges quickly, and the average of the mean ansatz energy estimate reaches -1.826 (hartree) when excluding two abnormal instances with local minima, which is very close to the theoretical value -1.85 (hartree). This indicates the successful application of our model.

Task: learning a one-qubit quantum channel. We choose DQNN₂ mentioned in the main text to learn a one-qubit target quantum channel. In our simulation, the training dataset is the same as in the main text. Our numerical results for 50 different initial parameters are summarized in Supplementary Fig. S3. We observe that DQNN₂ shows high convergence performance in the training process, with the average converged mean fidelity above 99.5%. We also choose one of these learning curves (marked in triangles in Supplementary Fig. S3), and refer DQNN₂ with parameters corresponding to the ending (starting) epoch of the training curve as the trained (untrained) DQNN₂. We then use 100 different input quantum states to test the fidelities between their corresponding output states and the desired output states given by the target quantum channel. As shown in the lower inset of Supplementary Fig. S3, for the trained DQNN₁, the mean fidelity exceeds 0.999 (green bars), which separates away from the distribution of the untrained DQNN₂ (purple bars) with the mean fidelity below 0.4. This contrast indicates a satisfying performance of DQNN₂.



Supplementary Figure S2. **Numerical results for learning the ground state energy of molecular hydrogen.** Energy estimate as a function of the training epochs for 50 different initial parameters. The distribution of their converged energy estimates is displayed in the inset.

SUPPLEMENTARY NOTE 3: EXPERIMENTAL IMPLEMENTATION OF THE DQNN

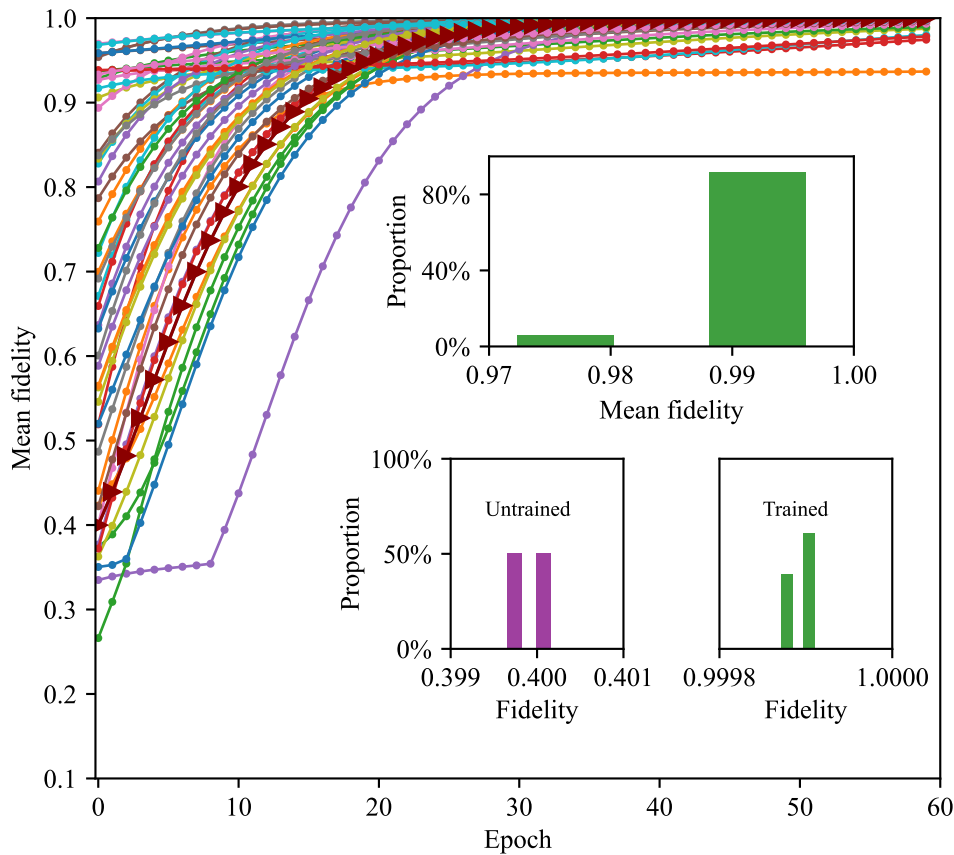
Characterization of the quantum processor

Our experiment is performed on a six-qubit superconducting quantum processor. As shown in Fig. 1(d) in the main text, the layout of qubits is carefully optimized to be a layer-by-layer structure. We denote these qubits as Q_j , where $j = 1, 2, \dots, 6$, and the labels are the same as those in the main text figures. The detailed experimental wiring of qubit control lines and measurement lines are shown in Supplementary Fig. S4. We summarize the characteristic parameters of our quantum processor in Table S1.

Synthesize the microwave control signals

Timing and microwave switch control. We note that the microwave control signals for the single-qubit gates of Q_1, Q_2, Q_5 and Q_3, Q_4, Q_6 are directly generated by the two DAC channels of a Tektronix AWG70002A (sampling rate 25 Gs per second), respectively. The single-qubit gates are implemented by 40 ns pulses with Gaussian envelopes. The tunability of the parameter θ in the DQNN is experimentally realized with a linear map between θ and the pulse amplitude. The individual addressing of each qubit is realized with the time domain separation of the microwave drives. In order to minimize the off-resonance crosstalk coming from the signal multiplexing, we add a fast microwave switch (activation time < 10 ns, on-off ratio > 40 dB) to each input XY control line, and turn on the switches only when the single-qubit gates are needed to be applied to the specific qubits.

Implementation of two-qubit gates with flux modulation. To realize the controlled-phase gate in a quantum perceptron, we adiabatically tune one of the qubit frequency to bring the $|ee\rangle$ state of the control and the target qubits into resonance with the $|gf\rangle$ state. Then the two-qubit state undergoes a periodic evolution path based on the coupling Hamiltonian of the two

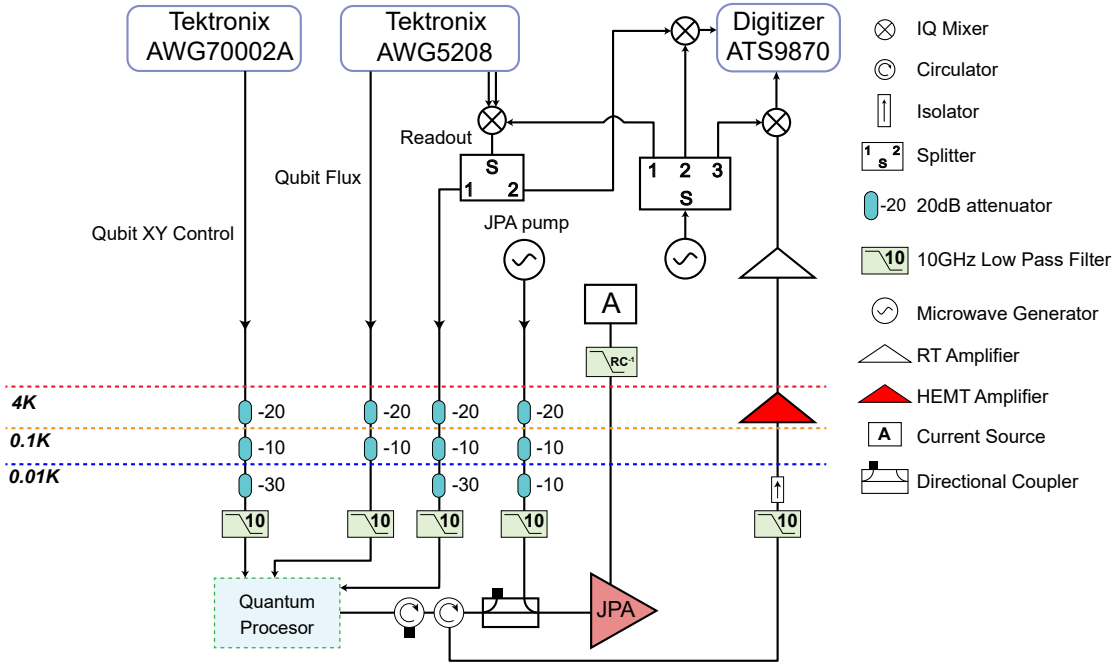


Supplementary Figure S3. **Numerical results for learning a one-qubit quantum channel.** The mean fidelity is plotted as the function of the training epochs for 50 different initial parameters. The distribution of their converged mean fidelities is displayed in the upper inset. We choose one of the learning curves (marked with triangles), then randomly generate 100 single-qubit states, separately produce their output states given by the target quantum channel and the well-trained (untrained) DQNN₂, and finally evaluate the corresponding fidelities between them. The distributions of the fidelities are shown in the lower inset.

qubits, leaving the population of the $|ee\rangle$ state intact, but a conditional geometric phase being accumulated in the $|ee\rangle$ state. Such an operation is realized with a fast step pulse of the external current threading the junction loop of the qubit to modify its frequency [47].

However, the limited bandwidth of the electronics as well as parasitic capacitances and inductances in the wiring cables lead to the distortion of the step pulses and thus the degradation of the gate performances. We adapt the method in Ref. [48] to mitigate this problem with the flux pulse compensation. We model the AWG response and the on-chip responses to the step pulse as low-pass filters, and apply real-time predistortions to the step pulse for compensations. The height and duration of the step pulse are optimized to minimize the errors in the swap process between $|ee\rangle$ and $|gf\rangle$. The comparison between the compensated and the uncompensated flux pulses is shown in Supplementary Fig. S5. We note that the optimized gate parameters do not necessarily lead to a conditional π phase, therefore, we just record the conditional phase ϕ and use it in the two-qubit gate in the perceptron. Moreover, the flux pulse of the target qubit will generally lead to the magnetic flux change not only in the target qubit loop, but also in other qubit loops, which causes the frequency and phase change of other qubit states. We calibrate the single-qubit phase of each qubit during the flux pulse through quantum state tomography, and compensate the flux-induced single-qubit phase in software by a phase shift of the following driving pulses. Based on the above implementation, the average fidelity of the controlled-phase gates in the DQNN is around 0.95 for all qubit pairs.

Different working frequencies and phase compensation due to reference frame change. In our experiment, since the two-qubit gate requires the frequency modulation of the qubit, it is possible that the energy level resonances other than the wanted $|ee\rangle$ and $|gf\rangle$ hybridization could occur during the modulation process. Such unwanted resonances will lead to undesired state swapping that degrades the gate fidelity. In order to avoid the unwanted frequency resonances, we have set the working frequencies of the six qubits to several different configurations when executing different perceptrons in DQNN₁ and DQNN₂ (see Table S2 for details). Meanwhile, the frequency changes of the qubits require additional time-dependent phase compensation. As illustrated in Supplementary Fig. S6(a), we calibrate the time-dependent phase change of each qubit by preparing the state

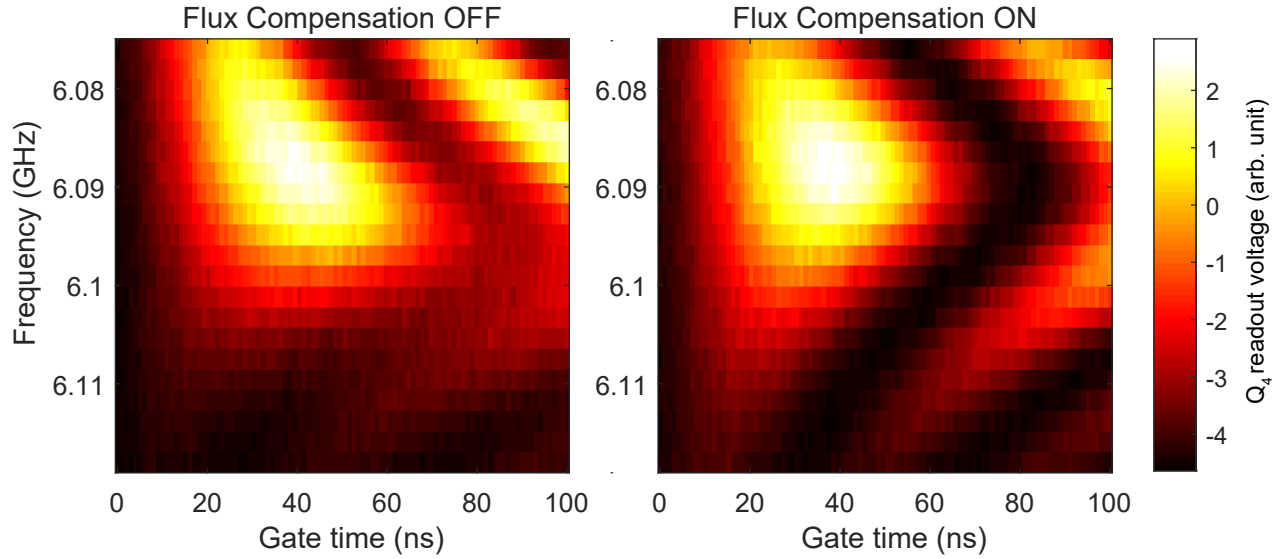


Supplementary Figure S4. **The experimental wiring of qubit control lines and measurement lines.** We plot one of two identical qubit XY control lines and one of six identical qubit flux lines for simplification purpose.

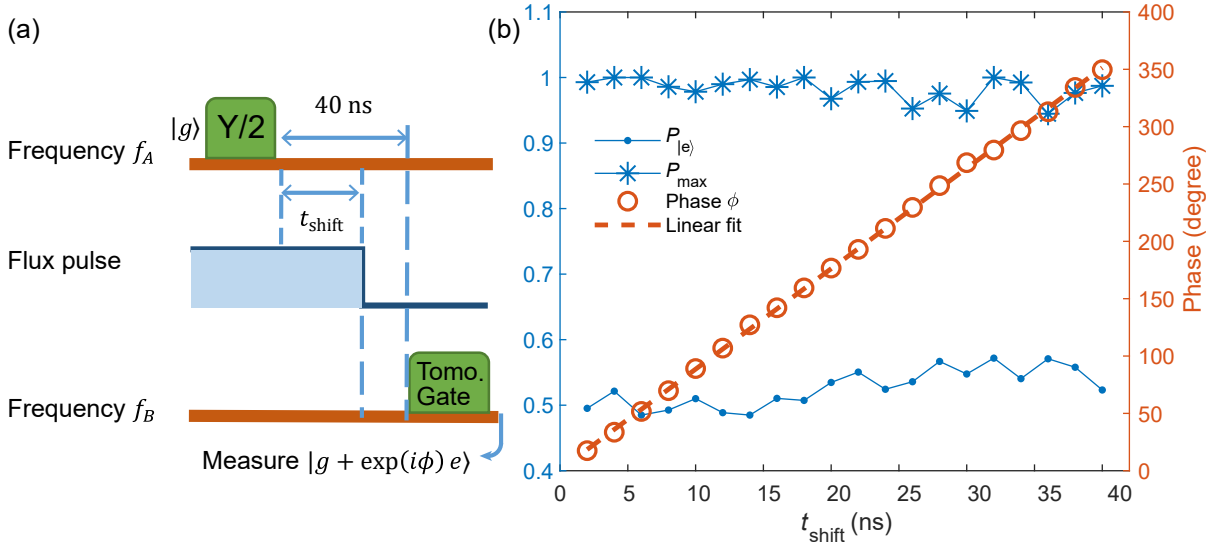
Parameters / Qubit	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6
Qubit working frequency f_Q (GHz)	6.413	6.363	6.328	6.453	6.083	6.191
Qubit energy relaxation time T_1 (μs)	4.2	6.1	5.1	8.2	10.0	10.6
Qubit Ramsey dephasing time T_2 (μs)	2.2	1.9	4.8	8.4	18.2	11.8
Qubit anharmonicity $E_C/2\pi$ (MHz)	194	217	206	196	207	208
Readout resonator frequency f_R (GHz)	7.10	7.16	7.13	7.22	7.12	7.21
Qubit-readout-resonator coupling strength $g_{QR}/2\pi$ (MHz)	69	72	81	66	78	65
Qubit-bus-resonator1 coupling strength $g_{QB1}/2\pi$ (MHz)	0	0	36	35	37	34
Qubit-bus-resonator2 coupling strength $g_{QB2}/2\pi$ (MHz)	32	31	34	32	0	0
Internal quality factor of the readout resonator $Q_{I,R}$ (10^3)	102	83	130	15	85	92
Coupled quality factor of the readout resonator $Q_{C,R}$ (10^3)	20	15	20	7.6	7.4	8.0

TABLE S1. **Characteristic parameters of the quantum processor.**

$(|g\rangle + |e\rangle)/\sqrt{2}$ with a microwave driving frequency f_A when the qubit frequency is also at f_A , and then apply a predistorted step pulse in the flux control line to shift the qubit frequency to f_B . A quantum state tomography of the qubit with a driving frequency f_B is performed to extract the phase accumulation caused by the frequency change. Here we fix the time interval between the tomography pulse and the state preparation pulse, and vary the time t_{shift} between the state preparation pulse and the step pulse to calibrate the phase accumulation with t_{shift} . The calibration result is shown in Supplementary Fig. S6(b). Such a phase accumulation is also corrected in software by shifting the phases of the driving pulses after the frequency change.



Supplementary Figure S5. **The comparison between the swap operations with and without the flux compensation.** With the frequencies of all other qubits well below 5.8 GHz, we prepare Q_4 and Q_5 in the $|eg\rangle$ state, and take a step pulse in the flux line to modulate the frequency of Q_4 down to reach the resonance with the $|ge\rangle$ state. The modulated qubit frequency and the duration of the step pulse are varied in the experiment. Compared with the uncompensated step pulse, the predistortion compensation method has successfully recovered the chevron pattern of the expected $|ge\rangle$ and $|eg\rangle$ swap process.



Supplementary Figure S6. **Calibration of the single-qubit phase induced by the shift of the working frequency.** (a) The experimental pulse sequence. (b) The experimental result for the frequency shift process of Q_4 . The blue dots denote the probability of measuring $|e\rangle$ state $P_{|e\rangle}$. The blue star marks denote P_{\max} , which is the larger eigenvalue of the single-qubit density matrix. The near-unity values of P_{\max} indicate the measured final quantum states are close to pure states. The red circles denote the phase ϕ extracted from the final quantum states in the form $|g\rangle + e^{i\phi}|e\rangle$. The dashed line is a linear fit to the red circles to infer the desired frequency shift.

Quantum state tomography

We extract the quantum state ρ^l of the qubits in each layer of the DQNN by carrying out the quantum state tomography. To reconstruct a single-qubit state, we perform single-qubit Pauli measurements on four bases $\mathcal{S}_1 = \{|g\rangle, |e\rangle, |+\rangle, |i\rangle\}$. To reconstruct a two-qubit state, we perform two-qubit Pauli measurements on 16 bases $\mathcal{S}_2 = \{|v_1\rangle \otimes |v_2\rangle; v_1, v_2 \in \mathcal{S}_1\}$. In our experiment, we repeat the measurement in each basis 10^4 times to obtain a probability distribution \vec{r} on the two and four computational bases for the single-qubit and two-qubit cases, respectively. \vec{r} is sent to a classical convex optimizer to find the

DQNN₁

Perceptron	Qubits	Q_1 (GHz)	Q_2 (GHz)	Q_3 (GHz)	Q_4 (GHz)	Q_5 (GHz)	Q_6 (GHz)	Time (ns)	Phase
$U_{1,1}^1$	Q_1, Q_3	6.413	6.364	6.320	6.453	< 5.8	< 5.8	62	175°
$U_{2,1}^1$	Q_2, Q_3							52	180°
$U_{1,2}^1$	Q_1, Q_4							92	180°
$U_{2,2}^1$	Q_2, Q_4							82	-155°
$U_{1,2}^2$	Q_3, Q_6	< 5.8	< 5.8	6.328	6.453	6.08	6.191	64	176°
$U_{1,1}^2$	Q_3, Q_5			64				-117°	
$U_{2,2}^2$	Q_4, Q_6			< 5.8				64	-157°
$U_{2,1}^2$	Q_4, Q_5			60				-165°	

DQNN₂

Perceptron	Qubits	Q_1 (GHz)	Q_2 (GHz)	Q_3 (GHz)	Q_4 (GHz)	Q_5 (GHz)	Q_6 (GHz)	Time (ns)	Phase
$U_{1,1}^1$	Q_1, Q_3	6.413	6.364	6.320	6.453	< 5.8	< 5.8	62	175°
$U_{1,1}^2$	Q_2, Q_3							52	180°
$U_{1,1}^3$	Q_2, Q_4							82	-155°
$U_{1,1}^4$	Q_4, Q_6	< 5.8	< 5.8	< 5.8	6.453	6.08	6.191	64	-157°
$U_{1,1}^5$	Q_5, Q_6							60	-130°

TABLE S2. **Experimental parameters for training DQNNs.** In our experiments for training DQNN₁ and DQNN₂, we apply the quantum perceptrons in the order from the top to the bottom in the first column. Each perceptron acts on the qubits listed in the second column. When applying different perceptrons, we need to set the qubits to different frequencies as listed. We also show the operation time and the rotation angle for the controlled-phase gate in each perceptron.

density matrix ρ^j that produces the distribution as close as \vec{r} .