

## Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor

Vlothuizen, W.; Marques, J. F.; van Straten, J.; Ali, H.; Muthusubramanian, N.; Zachariadis, C.; van Someren, J.; Beekman, M.; Haider, N.; Bruno, A.

**DOI**

[10.1038/s41534-023-00779-5](https://doi.org/10.1038/s41534-023-00779-5)

**Publication date**

2023

**Document Version**

Final published version

**Published in**

NPJ Quantum Information

**Citation (APA)**

Vlothuizen, W., Marques, J. F., van Straten, J., Ali, H., Muthusubramanian, N., Zachariadis, C., van Someren, J., Beekman, M., Haider, N., Bruno, A., Almudever, C. G., DiCarlo, L., & More Authors (2023). Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor. *NPJ Quantum Information*, 9(1), Article 118. <https://doi.org/10.1038/s41534-023-00779-5>

**Important note**

To cite this publication, please use the final published version (if applicable).  
Please check the document version above.

**Copyright**

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

**Takedown policy**

Please contact us and provide details if you believe this document breaches copyrights.  
We will remove access to the work immediately and investigate your claim.

## ARTICLE OPEN



# Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor

M. S. Moreira<sup>1,2</sup>, G. G. Guerreschi<sup>3</sup>, W. Vlothuizen<sup>1,4</sup>, J. F. Marques<sup>1,2</sup>, J. van Straten<sup>1,5</sup>, S. P. Premaratne<sup>3</sup>, X. Zou<sup>3</sup>, H. Ali<sup>1,2</sup>, N. Muthusubramanian<sup>1,2</sup>, C. Zachariadis<sup>1,2</sup>, J. van Someren<sup>1,5</sup>, M. Beekman<sup>1,4</sup>, N. Haider<sup>1,4</sup>, A. Bruno<sup>1,2</sup>, C. G. Almudever<sup>1,5</sup>, A. Y. Matsuura<sup>3</sup> and L. DiCarlo<sup>1,2</sup>✉

Artificial neural networks are becoming an integral part of digital solutions to complex problems. However, employing neural networks on quantum processors faces challenges related to the implementation of non-linear functions using quantum circuits. In this paper, we use repeat-until-success circuits enabled by real-time control-flow feedback to realize quantum neurons with non-linear activation functions. These neurons constitute elementary building blocks that can be arranged in a variety of layouts to carry out deep learning tasks quantum coherently. As an example, we construct a minimal feedforward quantum neural network capable of learning all 2-to-1-bit Boolean functions by optimization of network activation parameters within the supervised-learning paradigm. This model is shown to perform non-linear classification and effectively learns from multiple copies of a single training state consisting of the maximal superposition of all inputs.

npj Quantum Information (2023)9:118; <https://doi.org/10.1038/s41534-023-00779-5>

## INTRODUCTION

Deep learning is an established field with pervasive applications ranging from image classification to speech recognition<sup>1</sup>. Among the most intriguing recent developments is the extension to the quantum regime and the search for advantage based on quantum mechanical effects<sup>2</sup>. This effort is pursued in a variety of ways, often inspired by the diversity of classical models and based on the concept of artificial neural networks. Prior works have proposed quantum versions of perceptrons<sup>3,4</sup>, support vector machines<sup>5,6</sup>, Boltzmann machines<sup>7</sup>, autoencoders<sup>8</sup>, and convolutional neural networks<sup>9–11</sup>. The advantage ranges from reducing the model size by exploiting the exponentially large number of amplitudes defining multi-qubit states, to speeding-up either training or inference by applying efficient quantum algorithms such as HHL<sup>12</sup> to solve systems of linear equations or reducing the number of samples needed for accurate learning.

A promising implementation is based on variational quantum algorithms in which parametrized quantum circuits are used to prepare approximate solutions to the problem at hand. These solutions are then refined by classically optimizing circuit parameters<sup>13</sup>. However, fundamental questions must be answered on the parameter landscape<sup>14</sup>, on the cost of the classical optimization loop, and on the expressive power of circuit ansatz<sup>15</sup>. Encouraging results suggest that trainability is possible for quantum convolutional neural networks<sup>16,17</sup>. Still, it is recognized that loading training set data into a quantum machine accurately and efficiently is an unsolved problem<sup>18</sup> and, although promising results<sup>19–21</sup>, current solutions work only under specific assumptions. Nevertheless, the exponential complexity of states generated by ever larger quantum computers<sup>22</sup> suggests that machine learning techniques will become increasingly important at directly processing large-scale quantum states<sup>23</sup>.

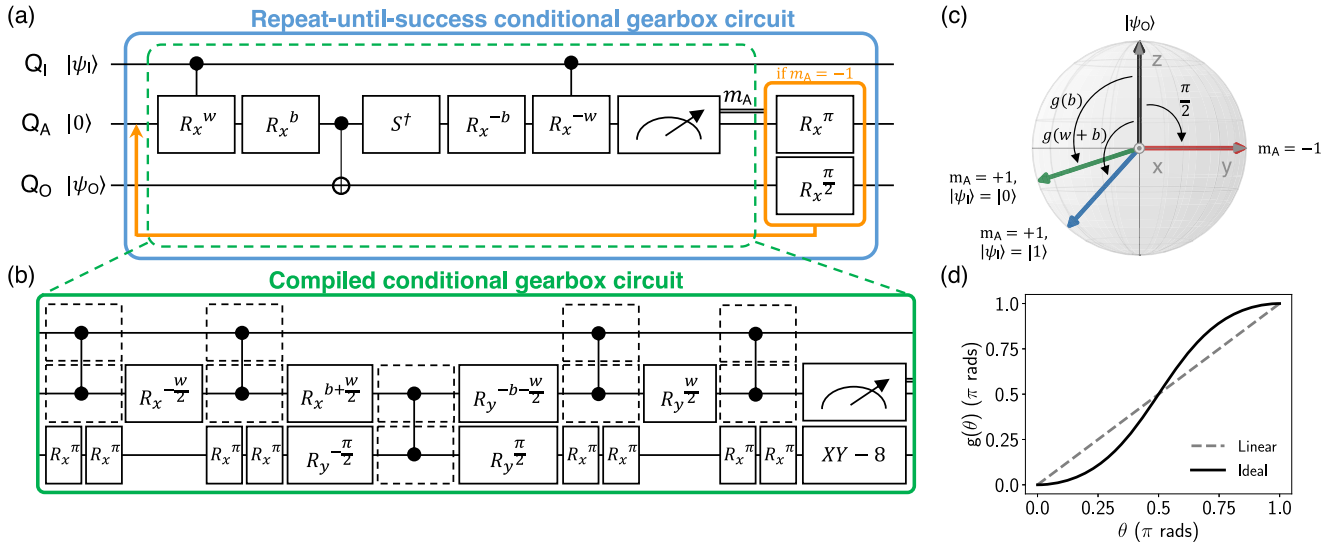
It was noted in traditional machine-learning literature that non-linear activation functions for neurons are superior<sup>24</sup>. To translate

this observation to the design of quantum neural networks (QNNs), several methods have been proposed to break the intrinsic linearity of quantum mechanics. These solutions range from the use of quantum measurements and dissipative quantum gates<sup>25</sup>, to the quadratic form of the kinetic term<sup>26</sup>, reversible circuits<sup>27</sup>, recurrent neural networks<sup>28</sup> and the SWAP test<sup>29</sup> with phase estimation<sup>30</sup>.

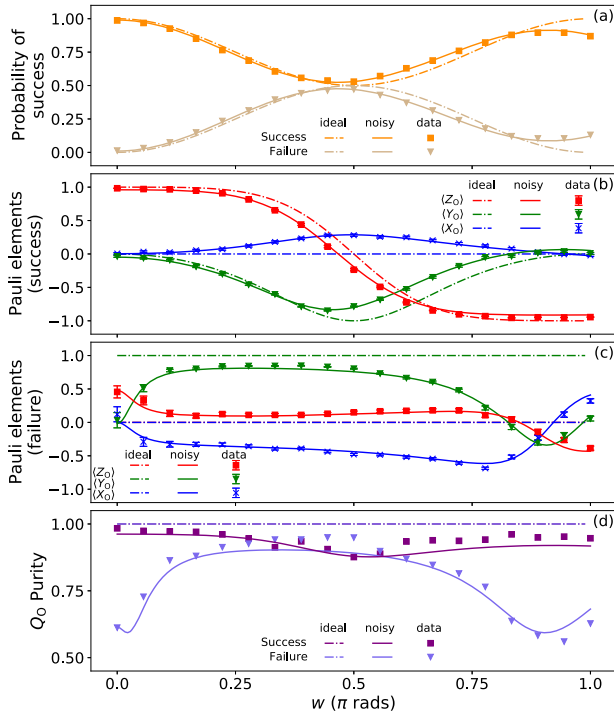
Previous work in this context<sup>10</sup> has shown the implementation of neural networks applied in post-processing to the classical results of measurements. Here, we experimentally demonstrate a quantum neural network architecture based on variational repeat-until-success (RUS) circuits<sup>31,32</sup>, that is implemented in a fully coherent way, handling quantum data directly, and in which real-time feedback is used to perform the internal update of neurons. In this model, each artificial neuron is substituted by a single qubit<sup>33</sup>. The neuron update is achieved with a quantum circuit that generates a non-linear activation function using control-flow feedback based on mid-circuit measurements. This activation function is periodic but locally resembles a sigmoid function. Despite the mid-circuit measurement, this approach does not suffer from the collapse of relevant quantum information. Rather, the measurement outcome signals that the neuron update is either successfully implemented or that a fixed, input-independent operation is performed. This other operation can be undone by feedback and the circuit rerun as necessary until success, leading to a constant, not exponential, overhead in the number of elementary operations required by RUS. Note that the overall fidelity of RUS circuits critically depends on the architecture and speed of the active feedback mechanism.

Our experiment uses 4 of the 7 transmons in a circuit QED processor<sup>34</sup> to implement a feedforward QNN with two inputs, one output, and no intermediate layers. We demonstrate that the QNN can learn each of the 16 2-to-1-bit Boolean functions by changing the weights and bias associated with the output neuron.

<sup>1</sup>QuTech, Delft University of Technology, P.O. Box 5046, 2600 Delft, GA, The Netherlands. <sup>2</sup>Kavli Institute of Nanoscience, Delft University of Technology, P.O. Box 5046, 2600 Delft, GA, The Netherlands. <sup>3</sup>Intel Labs, Intel Corporation, 97124 Hillsboro, OR, USA. <sup>4</sup>Netherlands Organisation for Applied Scientific Research (TNO), P.O. Box 96864, 2509 The Hague, JG, The Netherlands. <sup>5</sup>Computer Engineering Lab, Delft University of Technology, Mekelweg 4, 2628 Delft, CD, The Netherlands. ✉email: l.dicarlo@tudelft.nl



**Fig. 1 Conditional gearbox circuit using repeat until success.** **a** Three-qubit circuit with input parameters  $(w, b)$  ideally implementing  $R_x^{g(w+b)}$  on  $Q_O$  for  $Q_1 = |1\rangle$  and  $R_x^{g(b)}$  for  $Q_1 = |0\rangle$ , heralded by  $m_A = +1$  (success). For  $m_A = -1$  (failure), the circuit ideally implements  $R_x^{w/2}$  on  $Q_O$ . The probabilistic nature of the circuit is rectified using RUS: in case of failure,  $Q_A$  and  $Q_O$  are first reset ( $R_x^\pi$  and  $R_x^\pi$ , respectively), and the circuit re-run. The  $S^\dagger$  symbol corresponds to  $R_x^{w/2}$ . **b** Compilation into the native gate set after circuit optimization and added error mitigation (two refocusing pulses on  $Q_O$  during  $Q_1$ - $Q_A$  CZ gates and  $XY-8$  sequence<sup>38</sup> during measurement). **c** Illustration of the ideal action of the conditional gearbox circuit on  $Q_O$  when starting in  $|0_0\rangle$ . **d** Comparison of the ideal  $g(\theta)$  to a Rabi oscillation of  $Q_O$ , showing the non-linearity of  $g$ .



**Fig. 2 Synthesis of non-linear functions using a conditional gearbox circuit.** **a** Probability of success and failure at the first iteration of the conditional gearbox circuit (Fig. 1) as a function of  $w$  ( $b = 0$ ). **b**, **c** Pauli components of  $Q_O$  assessed by quantum state tomography conditioned on **(b)** success and **(c)** failure, for  $Q_1$  prepared in  $|1\rangle$ . **d** Purity of  $Q_O$  for success and failure. All panels include experimental results (symbols), ideal simulation (dashed curves), and noisy simulation (solid curves).

It is particularly noteworthy that this architecture allows implementation of the XOR Boolean function using a single neuron since this is a fundamental example of the limitations of classical artificial neuron constructions, which cannot capture the linear inseparability<sup>1</sup> of such a function.

We follow the supervised learning paradigm, in which a set of training examples provides information to the network about the specific function to learn. Our experiment uses multiple copies of a single input state (the maximal superposition of 4 inputs), demonstrating that the QNN can learn from a superposition. Finally, we investigate the specificity of parameters learned for each of the Boolean functions by characterizing how well the values learned for one function can be used for any other. This provides indications on using the QNN to discriminate between the Boolean functions when provided as a quantum black box.

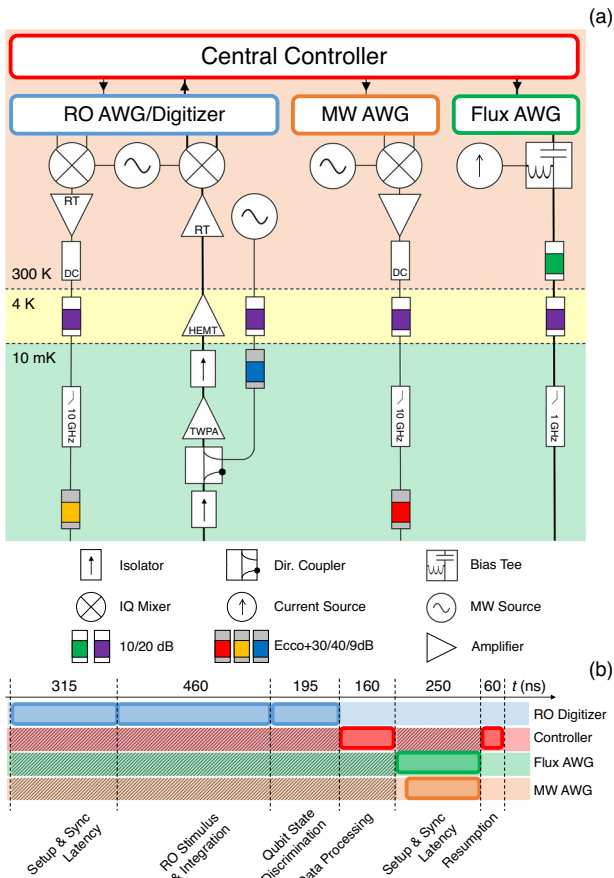
## RESULTS

### Synthesizing non-linear functions using conditional gearbox circuits

The conditional gearbox circuit<sup>35</sup> belongs to a class of RUS circuits<sup>31</sup> that use one ancilla qubit  $Q_A$  and mid-circuit measurements to implement a desired operation. The three-qubit version (Fig. 1a) has input qubit  $Q_1$ , output qubit  $Q_O$ , and angles  $w$  and  $b$  as classical input parameters. For an ideal processor starting with  $Q_A$  in  $|0_A\rangle$ ,  $Q_1$  in computational state  $|k_1\rangle$  ( $k \in \{0, 1\}$ ), and  $Q_O$  in the arbitrary state  $|\psi_0\rangle$ , the coherent operations produce the state

$$|0_A\rangle|k_1\rangle|\psi_0\rangle \rightarrow \sqrt{p_S(\theta_k)}|0_A\rangle|k_1\rangle R_x^{g(\theta_k)}|\psi_0\rangle + \sqrt{1-p_S(\theta_k)}|1_A\rangle|k_1\rangle R_x^{w/2}|\psi_0\rangle, \quad (1)$$

where  $\theta_k = kw + b$  and  $p_S(\theta_k) = \cos^4(\frac{\theta_k}{2}) + \sin^4(\frac{\theta_k}{2})$ . A measurement of  $Q_A$  in its computational basis produces outcome  $m_A = +1$  (projection to  $|0_A\rangle$ ) with probability  $p_S(\theta_k)$ . In this case, the net effect on  $Q_O$  is a rotation around the  $x$  axis of its Bloch sphere by



**Fig. 3 Quantum control setup.** **a** Schematic of wiring and control electronics, highlighting critical feedback path between outputs of the quantum processor, the analog-interface devices, controller, and the flux-drive lines. **b** Timing diagram for the critical feedback path. Hashed regions indicate idling operations for each instrument. Latency concerns the time necessary to ensure synchronicity, since instruments are delayed with respect to the AWGs to account for the difference in propagation/latency time between readout and qubit drive instrumentation.

angle  $g(\theta_k)$ , where

$$g(\theta_k) = 2 \arctan \left( \tan^2 \left( \frac{\theta_k}{2} \right) \right) \quad (2)$$

is a non-linear function with a sigmoid shape (Fig. 1d). This outcome constitutes success.

For failure (i.e., outcome  $m_A = -1$  and projection onto  $|1_A\rangle$ ), the effect on  $Q_O$  is an  $x$  rotation by  $-\pi/2$ , independent of  $k$ ,  $w$ , and  $b$ . In this case, the effect of the circuit can be undone using feedback, specifically  $R_x^\pi$  and  $R_x^{\pi/2}$  gates on  $Q_A$  and  $Q_O$ , respectively. The circuit can then be re-run with feedback corrections until success is achieved. For an ideal processor, the average number of runs to success,  $\langle N_{RTS} \rangle$ , is bounded by  $1 \leq \langle N_{RTS} \rangle = 1/p_S(\theta_k) \leq 2$ . This bound holds even when  $Q_I$  is initially in a superposition state  $|\psi_I\rangle = \alpha|0_I\rangle + \beta|1_I\rangle$ . In this general case, the output state upon success is still a superposition but with potentially different amplitudes:

$$|0_A\rangle \sum_{k=0}^1 \alpha'_k |k_I\rangle R_x^{g(\theta_k)} |\psi_O\rangle. \quad (3)$$

The probability amplitudes can change, from  $a_k$  to  $\alpha'_k$ , depending on the initial  $|\psi_I\rangle$ ,  $w$ ,  $b$ , and  $N_{RTS}$ . This distortion of probability amplitudes can be mitigated using amplitude amplification<sup>36</sup>, which we do not employ here.

We compile the three-qubit conditional gearbox circuit into the native gate set of our processor (Fig. 1b) and evidence its action after one round using state tomography of  $Q_O$  conditioned on success and failure. Figure 2 shows experimental results when preparing  $Q_I$  ( $Q_O$ ) in  $|1_I\rangle$  ( $|0_O\rangle$ ), setting  $b=0$  and sweeping  $w$ , alongside simulation for both an ideal and a noisy processor. Qualitatively, the experimental results reproduce the key features of the ideal circuit: we observe a  $\pi$ -periodic oscillation in  $p_S(w)$  with minimal value 0.5 at  $w = \pi/2$ , and a sharp variation in  $Z_O$  from  $+1$  to  $-1$  centered at  $w = \pi/2$ . However, the nonzero  $Z_O$  components observed for both success and failure indicate that the action on  $Q_O$  for both cases is not purely an  $x$ -axis rotation. The noisy simulation captures all key nonidealities observed. This simulation includes nonlinearity in single-qubit microwave driving, cross resonance<sup>37</sup> effects between  $Q_A$  and  $Q_O$ , phase errors in CZ gates, readout error in  $Q_A$ , and qubit decoherence [see Supplementary Materials]<sup>38</sup>.

We note the existence of different activation circuits that can implement non-linear activation without the need for RUS strategies. Despite the disadvantages of softer non-linearities implemented by such schemes, they may be viable candidates for implementations of QNNs optimized to noisy processors. We provide a detailed comparison of these schemes in the Supplementary Material.

### Control-flow feedback on a programmable superconducting quantum processor

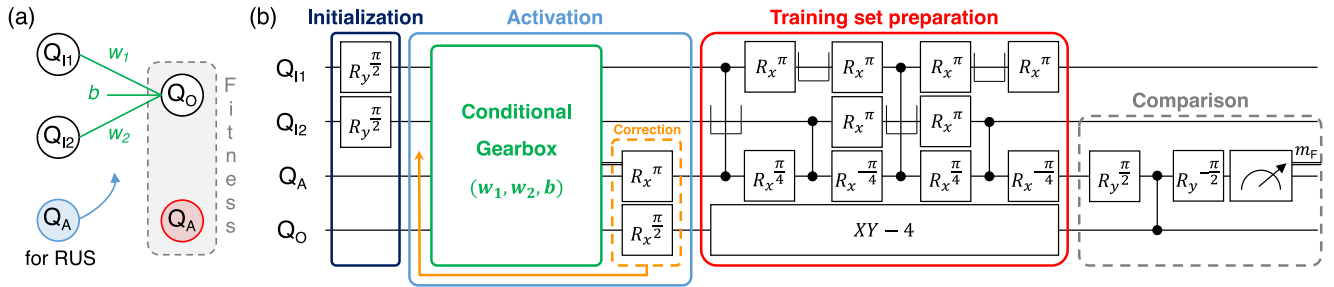
Active feedback is important for many quantum computing applications, including quantum error correction (QEC). Past demonstrations of QEC relied on the storage of measurements without real-time feedback<sup>39,40</sup>. Moreover, real-time feedback has been demonstrated using data-flow mechanisms, where individual operations are applied conditionally<sup>41</sup>. In contrast, the implementation of RUS hinges on support for control-flow mechanisms in the control setup (Fig. 3), where the entire sequence of operations has to be assessed and executed, depending on the results of measurements, in real-time.

In our quantum control architecture, a controller sequences the sets of operations to be performed in real-time, controlling various arbitrary waveform generators (AWG) and digitizers to implement the desired program. Therefore, our implementation of control-flow feedback focuses on this controller and achieves a maximum latency of 160 ns. The latency to complete the full feedback loop of the overall control system (controller, analog-interface devices, and the entire analog chain) was measured to be 980 ns. This represents 3% of the worst coherence time, and sets an upper bound on the efficiency of RUS execution with the quantum processor. Further improvements could be achieved by optimizing the design of our readout (RO) AWG for trigger latency and speeding up the task of digital signal processing within the digitizers.

Note that the critical feedback path consists of the entire readout chain in addition to the slowest instrument, whose latency must also be accounted for before the branching condition is assessed and implemented. In our control setup, the slowest instrument is the Flux AWG, due to the latency introduced by various finite input responses and exponential filters implemented in hardware for the correction of on-chip distortion of control pulses<sup>42</sup>.

### Constructing a QNN using RUS circuits

The characteristic threshold shape of  $g$  makes it useful in the context of neural networks: the conditional gearbox circuit can be seen as a non-linear activation function, whose rotations are controlled by the input qubits to mimic the propagation of information between network layers. We use these concepts<sup>33</sup> to implement a minimal QNN capable of learning any of the



**Fig. 4 Quantum neural network using the repeat-until-success conditional gearbox circuit.** **a** Schematic representation of simplest feedforward network, highlighting the role played by parameters  $(w_1, w_2, b)$  in weighing the sum of input signals, before the result is passed through a non-linear activation function.  $Q_{11}$  and  $Q_{12}$  are input nodes,  $Q_O$  is the output node and  $Q_A$  is an ancilla used first within the RUS circuit and then as expected output for the training set. **b** Quantum circuit for a 3-neuron feedforward network. This circuit is divided into four steps. Input  $(Q_{11}, Q_{12})$  preparation into maximal superposition; threshold activation into  $Q_O$  using RUS conditional gearbox circuit with  $(w_1, w_2, b)$ ; unitary encoding of Boolean function (AND, in this case) using oracle; and comparison of  $Q_A$  with  $Q_O$ .  $XY-4$  specifies an error mitigation sequence<sup>38</sup> applied during measurement and the symbol  $\sqcup$  denotes parking of spectator qubit  $Q_{12}(Q_{11})$  during  $CZ(Q_A, Q_{11})$  ( $CZ(Q_A, Q_{12})$ ) gates [Supplementary Materials].

2-to-1-bit Boolean functions (see Supplementary Materials for their definition and naming convention). These 16 functions can be separated into three categories: two constant functions (NULL and IDENTITY) have the same output for all inputs; 6 balanced functions (e.g., XOR) output 0 for exactly two inputs; and 8 unbalanced functions (e.g. AND) have the same output for exactly three inputs.

The 4-qubit circuit shown in Fig. 4 corresponds to a 3-neuron feedforward network. Two quantum inputs ( $Q_{11}$  and  $Q_{12}$ ) are initialized in a maximal superposition state. Next, the RUS-based conditional gearbox circuit (now with three input angles  $w_1$ ,  $w_2$ , and  $b$ ) performs threshold activation of  $Q_O$ , with no bounds placed on the number of times the conditional circuit is allowed to re-run until success. Following RUS (i.e.,  $Q_A$  projected to  $|0_A\rangle$ ),  $Q_A$  is reused for training set preparation. Here, the Boolean function  $f$  is encoded in a quantum oracle mapping  $|k_{11}\rangle|l_{12}\rangle|0_A\rangle \rightarrow |k_{11}\rangle|l_{12}\rangle|f(k, l)_A\rangle$ . At this point, the 4-qubit register is ideally in state

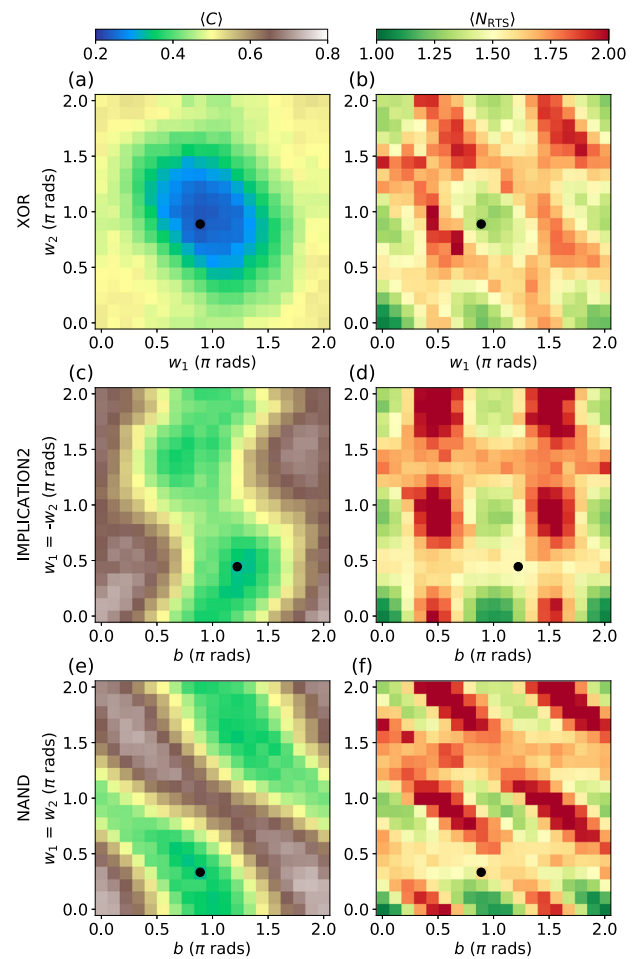
$$\sum_{k,l=0}^1 \alpha'_{kl} |k_{11}\rangle |l_{12}\rangle |f(k, l)_A\rangle R_x^{g(\theta_{kl})} |0_O\rangle, \quad (4)$$

where  $\theta_{kl} = kw_1 + lw_2 + b$ . Finally,  $Q_A$  and  $Q_O$  are compared by mapping their parity onto  $Q_A$  and performing a final measurement on  $Q_A$  on a computational basis.

We define  $C = (1 - m_A)/2$  from the output  $m_A$  and estimate  $\langle C \rangle \in [0, 1]$  by averaging over 10,000 repetitions of the full circuit. Training the QNN to learn a specific Boolean function thus amounts to minimizing  $\langle C \rangle$  over the 3-D input parameter space. Beforehand, we explore the feature space landscapes. Figure 5 shows 2-D slices of  $\langle C \rangle$  and  $\langle N_{RTS} \rangle$  for three examples: XOR, IMPLICATION2, and NAND (see Supplementary Materials for slices of all 16 functions). These slices are chosen to include the optimal settings minimizing  $\langle C \rangle$  for an ideal quantum processor [see Supplementary Materials]. These landscapes exemplify the complexity of the feature space and highlight the various symmetries and local minima that can potentially affect the efficient training of parameters.

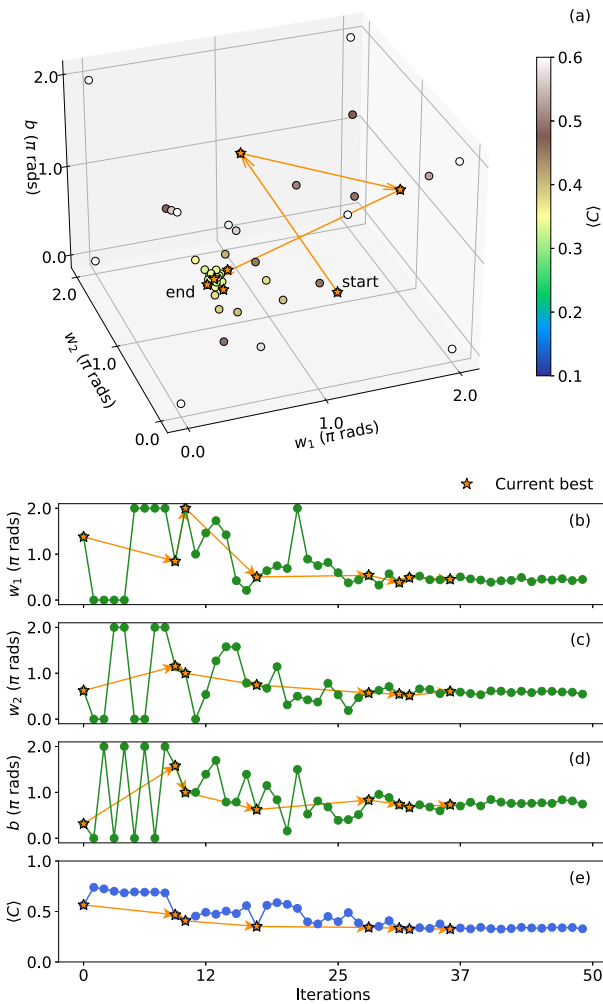
### Training a QNN from superpositions of data

To train the QNN, we employ an adaptive learning algorithm<sup>43</sup> to minimize  $\langle C \rangle$  over the full 3-D parameter space. Figure 6 shows the training process for NAND, chosen for the complexity of its feature space. The parameters evolve with each training step, starting from a randomly chosen initial point, then exploring the bounds, and subsequently converging to the global minimum in  $\sim 50$  training steps, requiring  $\sim 1$  min per step. This satisfactory behavior is observed for all the Boolean functions.



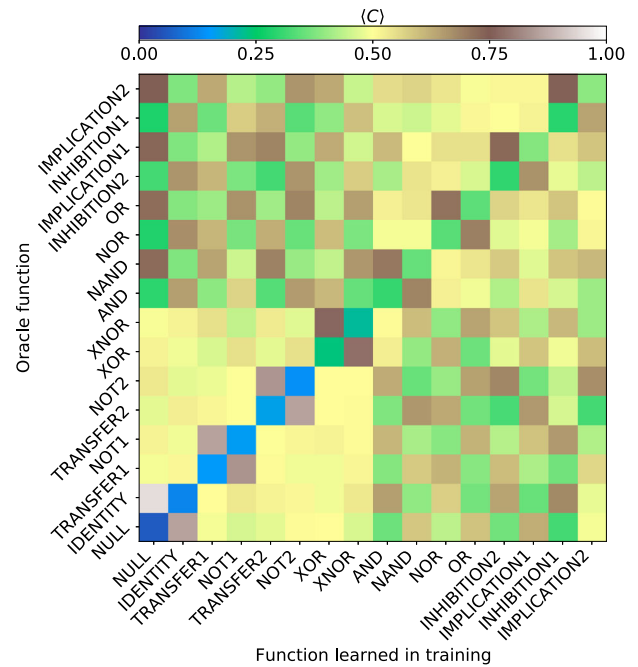
**Fig. 5 Feature space landscapes of three Boolean functions.** 2-D slices of  $\langle C \rangle$  and  $\langle N_{RTS} \rangle$  for XOR **(a, b)** IMPLICATION 2 **(c, d)**, and NAND **(e, f)**. For each function, the slice includes  $(w_1, w_2, b)$  parameters that minimize  $\langle C \rangle$  for an ideal quantum processor. Black dots indicate the experimental parameters achieving minimal  $\langle C \rangle$  within each slice.

Following training of the QNN for each Boolean function, we investigate the specificity of learned parameters by preparing the 256 pairs of trained parameters and function oracles and measuring  $\langle C \rangle$  for each pair. To understand the structure of the experimental specificity matrix (Fig. 7), it is worthwhile to first



**Fig. 6 Learning the NAND function.** **a** Training the QNN to learn NAND over the full parameter space ( $w_1, w_2, b$ ) by minimizing  $\langle C \rangle$  with an adaptive algorithm. Training starts from a randomly-chosen point, then explores the boundaries, and ultimately converges within  $\sim 50$  steps. **b–e** Evolution of training parameters ( $w_1, w_2, b$ ) and  $\langle C \rangle$  as a function of training step. The current best setting achieved is marked by a star.

consider the case of an ideal processor (see Supplementary Materials). Along the diagonal, we expect  $\langle C \rangle = 0$  for constant and balanced functions, which can be perfectly learned, and  $\langle C \rangle \approx 0.029$  for unbalanced functions, which cannot be perfectly learned due to the finite width of the activation function  $g(\theta)$ . In other words, no combination of ( $w_1, w_2, b$ ) can lead to perfect contrast for these functions. For off-diagonal terms, we expect  $\langle C \rangle$  at or close to multiples of 0.25, the multiple being set by the number of 2-bit inputs for which the paired training function and oracle function have different 1-bit output. For example, NAND and XOR have different outputs only for input 00, while TRANSFER1 and NOT1, which are complementary functions, have different output for all inputs. Note that every constant or balanced function, when compared to any unbalanced function, has a different output for exactly two inputs. Evidently, while the described pattern is discerned in the experimental specificity matrix, deviations result from the compounding of decoherence, gate-calibration, crosstalk, and measurement errors. These errors affect the 256 pairs differently for two main reasons. First, the average circuit depth of the RUS-based conditional gearbox circuit is higher for unbalanced functions. Second, the fixed circuit depth of oracles is also significantly higher for unbalanced functions, as



**Fig. 7 Specificity of the quantum neural network.** Cost function of the optimized parameter set for every training function (horizontal axis) against all oracle functions (vertical axis). In each axis, the functions are ordered from constant, to balanced, to unbalanced. Functions are put alongside their complementary function (NULL and IDENTITY, TRANSFER1 and NOT1, etc.). For an ideal processor,  $\langle C \rangle$  values are expected at or close to multiples of 0.25, due to the varying overlap between the 16 Boolean functions (i.e., the number of 2-bit inputs producing different 1-bit outcomes). Further differences arise in the experiment due to variations in the average circuit depth of the RUS-based activation functions and in the fixed circuit depth of oracle functions.

these all require a Toffoli-like gate which we realize using CZ and single-qubit gates. Noisy simulation [Supplementary Materials] modeling the main known sources of error in our processor produces a close match to Fig. 7.

Despite the evident imperfections, we have shown that it is possible to train the network across all functions, arriving at parameters that individually optimize each landscape. The circuit is thus able to learn different functions using multiple copies of a single training state corresponding to the superposition of all inputs, despite the complexity of feature space landscapes for various Boolean functions.

## DISCUSSION

We have seen that RUS is an effective strategy to address the probabilistic nature of the conditional gearbox circuit, allowing the deterministic synthesis of non-linear rotations. Even at the error rates of current superconducting quantum processors, it allowed the implementation of a QNN that reproduced a variety of classical neural network mechanisms while preserving quantum coherence and entanglement. Moreover, we have shown that this QNN architecture could be trained to learn all 2-to-1-bit Boolean functions using superpositions of training data.

This minimal QNN represents a fundamental building block that can be used to build larger QNNs. With larger numbers of qubits, these neurons could form multi-layer feed-forward networks containing hidden layers between inputs and outputs. Beyond feedforward networks, this minimal QNN is amenable to the implementation of various other network architectures, from Hopfield networks to quantum autoencoders<sup>33</sup>.

Finally, this work highlights the importance of real-time feedback control performed within the qubit coherence time and the quantum-classical interactions governing RUS algorithms. The ability to implement RUS circuits is in itself a useful result, as the active feedback architecture demonstrated is crucial for various other applications of a quantum computer, including active-reset protocols and the synthesis of circuits of shorter depth relative to purely unitary circuit design<sup>31</sup>, of value in areas such as quantum chemistry. Moreover, recent work into quantum error correction (QEC) highlights the importance of real-time quantum control in protocols for the distillation of magic states or, when coupled to a real-time decoder, the correction of errors. Similarly to real-time feedback, the construction of a real-time decoder that meets the stringent requirements for QEC with superconducting qubits requires application-specific hardware developments that are the focus of ongoing work.

## DATA AVAILABILITY

The data shown in all figures of the main text and supplementary material are available at [http://github.com/DiCarloLab-Delft/Quantum\\_Neural\\_Networks\\_Data](http://github.com/DiCarloLab-Delft/Quantum_Neural_Networks_Data).

## CODE AVAILABILITY

The code used for numerical simulations and data analyses is available from the corresponding author upon reasonable request.

Received: 20 March 2023; Accepted: 17 October 2023;

Published online: 21 November 2023

## REFERENCES

- Goodfellow, I., Bengio, Y. & Courville, A. Deep Learning (2016). <http://www.deeplearningbook.org>.
- Biamonte, J. et al. Quantum machine learning. *Nature* **549**, 195–202 (2017).
- Schuld, M., Sinayskiy, I. & Petruccione, F. Simulating a perceptron on a quantum computer. *Phys. Lett. A* **379**, 660–663 (2015).
- Tacchino, F., Macchiavello, C., Gerace, D. & Bajoni, D. An artificial neuron implemented on an actual quantum processor. *NPJ Quantum Inf.* **5**, 26 (2019).
- Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **113**, 130503 (2014).
- Havlíček, V. et al. Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).
- Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Estimation of effective temperatures in quantum annealers for sampling applications: a case study with possible applications in deep learning. *Phys. Rev. A* **94**, 022308 (2016).
- Romero, J., Olson, J. P. & Aspuru-Guzik, A. Quantum autoencoders for efficient compression of quantum data. *Quantum Sci. Technol.* **2**, 045001 (2017).
- Cong, I., Choi, S. & Lukin, M. D. Quantum convolutional neural networks. *Nat. Phys.* **15**, 1273–1278 (2019).
- Herrmann, J. et al. Realizing quantum convolutional neural networks on a superconducting quantum processor to recognize quantum phases <https://arxiv.org/abs/2109.05909> (2021).
- Gong, M. et al. Quantum neuronal sensing of quantum many-body states on a 61-qubit programmable superconducting processor <https://arxiv.org/abs/2201.05957> (2022).
- Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502 (2009).
- Cerezo, M. et al. Variational quantum algorithms. *Nat. Rev. Phys.* **3**, 625–644 (2021).
- McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R. & Neven, H. Barren plateaus in quantum neural networks training landscapes. *Nat. Commun.* **9**, 4812 (2018).
- Abbas, A. et al. The power of quantum neural networks. *Nat. Comput. Sci.* **1**, 403–409 (2021).
- Pesah, A. et al. Absence of barren plateaus in quantum convolutional neural networks. *Phys. Rev. X* **11**, 041011 (2021).
- Killoran, N. et al. Continuous-variable quantum neural networks. *Phys. Rev. Res.* **1**, 033063 (2019).
- Aaronson, S. Read the fine print. *Nat. Phys.* **11**, 291–293 (2015).
- Harrow, A. W. Small quantum computers and large classical data sets (2020). <https://arxiv.org/abs/2004.00026>.
- Zoufal, C., Lucchi, A. & Woerner, S. Quantum generative adversarial networks for learning and loading random distributions. *NPJ Quantum Inf.* **5**, 103 (2019).
- Holmes, A. & Matsuura, A. Y. Efficient quantum circuits for accurate state preparation of smooth, differentiable functions. 2020 IEEE International Conference on Quantum Computing and Engineering (QCE)169–179 (2020).
- Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
- Huang, H.-Y. et al. Quantum advantage in learning from experiments. *Science* **376**, 1182–1186 (2022).
- Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Networks* **2**, 359–366 (1989).
- Kak, S. On quantum neural computing. *Inf. Sci.* **83**, 143–160 (1995).
- Behrman, E., Nash, L., Steck, J., Chandrashekar, V. & Skinner, S. Simulations of quantum neural networks. *Inf. Sci.* **128**, 257–269 (2000).
- Wan, K. H., Dahlsten, O., Kristjánsson, H., Gardner, R. & Kim, M. S. Quantum generalisation of feedforward neural networks. *NPJ Quantum Inf.* **3**, 36 (2017).
- Rebentrost, P., Bromley, T. R., Weedbrook, C. & Lloyd, S. Quantum hopfield neural network. *Phys. Rev. A* **98**, 042308 (2018).
- Zhao, J. et al. Building quantum neural networks based on a swap test. *Phys. Rev. A* **100**, 012334 (2019).
- Li, P. & Wang, B. Quantum neural networks model based on swap test and phase estimation. *Neural Networks* **130**, 152–164 (2020).
- Paetznick, A. & Svore, K. M. Repeat-until-success: non-deterministic decomposition of single-qubit unitaries. *Quantum Inf. Comput.* **14**, 1277–1301 (2014).
- Bocharov, A., Roetteler, M. & Svore, K. M. Efficient synthesis of universal repeat-until-success quantum circuits. *Phys. Rev. Lett.* **114**, 080502 (2015).
- Cao, Y., Guerreschi, G. G. & Aspuru-Guzik, A. Quantum neuron: an elementary building block for machine learning on quantum computers <https://arxiv.org/abs/1711.11240>. (2017).
- Versluijs, R. et al. Scalable quantum circuit and control for a superconducting surface code. *Phys. Rev. Appl.* **8**, 034021 (2017).
- Wiebe, N. & Kliuchnikov, V. Floating point representations in quantum circuit synthesis. *New J. Phys.* **15**, 093041 (2013).
- Guerreschi, G. G. Repeat-until-success circuits with fixed-point oblivious amplitude amplification. *Phys. Rev. A* **99**, 022306 (2019).
- Chow, J. M. et al. Simple all-microwave entangling gate for fixed-frequency superconducting qubits. *Phys. Rev. Lett.* **107**, 080502 (2011).
- Maudsley, A. Modified carr-purcell-meiboom-gill sequence for nmr fourier imaging applications. *J. Magn. Reson. (1969)* **69**, 488–491 (1986).
- Krinner, S. et al. Realizing repeated quantum error correction in a distance-three surface code <https://arxiv.org/abs/2112.03708> (2021).
- Chen, Z. et al. Exponential suppression of bit or phase errors with cyclic error correction. *Nature* **595**, 383–387 (2021).
- Andersen, C. K. et al. Entanglement stabilization using ancilla-based parity detection and real-time feedback in superconducting circuits. *NPJ Quantum Inf.* **5**, 1–7 (2019).
- Rol, M. A. et al. Fast, high-fidelity conditional-phase gate exploiting leakage interference in weakly anharmonic superconducting qubits. *Phys. Rev. Lett.* **123**, 120502 (2019).
- Nijholt, B., Weston, J., Hoofwijk, J. & Akhmerov, A. Adaptive: parallel active learning of mathematical functions <https://doi.org/10.5281/zenodo.1182437> (2019).

## ACKNOWLEDGEMENTS

We thank G. Calusine and W. Oliver for providing the traveling-wave parametric amplifiers used in the readout amplification chain. This research is supported by Intel Corporation and by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office Grant No. W911NF-16-1-0071. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government.

## AUTHOR CONTRIBUTIONS

M.S.M. performed the experiment and data analysis. M.B., N.H., and L.D.C. designed the device. N.M., C.Z., and A.B. fabricated the device. M.S.M., J.F.M., and H.A. calibrated the device. M.S.M., W.V., J.S., J.S., and C.G.A. designed the control electronics. G.G.G. and L.D.C. performed the numerical simulations and motivated the project. M.S.M., G.G.G., and L.D.C. wrote the manuscript with input from A.Y.M., S.P.P., and X.Z. A.Y.M. and L.D.C. supervised the theory and experimental components of the project, respectively.

## COMPETING INTERESTS

The authors declare no competing interests.

## ADDITIONAL INFORMATION

**Supplementary information** The online version contains supplementary material available at <https://doi.org/10.1038/s41534-023-00779-5>.

**Correspondence** and requests for materials should be addressed to L. DiCarlo.

**Reprints and permission information** is available at <http://www.nature.com/reprints>

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2023